



doc. dr. Diana Kalibatienė

INFORMACINĖS SISTEMOS PROGRAMŲ KŪRIMAS
Kompleksinis projektas

Reikalavimai

Programų inžinerija, valstybinis kodas 612I30003

Informacijos sistemos, valstybinis kodas 612I20003

Vilnius, 2021

TURINYS

Įvadas.....	5
Studijų dalyko tikslas.....	5
Studijų dalyko aprašas.....	5
Darbo tikslai ir paskirtis.....	5
Darbų atlikimo ir vertinimo tvarka.....	5
Studento pasiekimų vertinimo formulė.....	6
Įvado struktūra.....	6
Darbo bendrosios dalies struktūra.....	7
Terminų ir trumpinių žodynėlis.....	7
Verslo žodynas.....	8
1 Poreikių specifikacija.....	9
1.1 Pradinė verslo analizė.....	9
1.1.1 Esamos būsenos analizė.....	9
1.1.2 Verslo vykdomų užduočių analizė.....	9
1.2 Verslo problemų, grėsmių ir neišnaudotų galimybių (SSGG) analizė.....	11
1.3 Verslo tobulinimo strategija.....	11
1.4 Užsakovo poreikių analizė.....	12
1.5 Sistemos naudojimo scenarijus.....	12
1.5.1 Esamoji būklė.....	13
1.5.2 Scenarijaus aprašas.....	13
1.5.3 Priemonės scenarijui įgyvendinti.....	13
1.6 PS įgyvendinamumo analizė.....	13
1.6.1 Operacinis įgyvendinamumas.....	13
1.6.2 Techninis sistemos įgyvendinamumas.....	14
1.6.3 Juridinis sistemos pagrindimas.....	14
1.6.4 Ekonominis sistemos įgyvendinamumas.....	14
1.6.5 Projekto veiklos planavimas.....	15
1.6.5.1 Projekto veiklų planas.....	15
1.6.5.2 Projekto resursų sąrašas.....	15
1.7 Teikiamos naudos analizė.....	15
1.8 Išvados ir siūlymai.....	15
2 Reikalavimų specifikacija.....	16
2.1 Formuluojamos užduotys.....	16
2.1.1 Pagrindinės užduotys.....	16
2.1.2 Užduočių formulavimo kalbos reikalavimai.....	16
2.1.3 Interfeiso darnos ir standartizavimo reikalavimai.....	16
2.1.4 Pranešimų formulavimo reikalavimai.....	16
2.1.5 Interfeiso individualizavimo reikalavimai.....	16

2.2	Funkciniai sistemos reikalavimai.....	16
2.2.1	Dalykiniai reikalavimai.....	17
2.2.2	Pagalbinės sistemos funkcijos.....	17
2.3	Nefunkciniai reikalavimai.....	17
2.3.1	Poskyrio “Vidinių interfeiso reikalavimai” paskirtis, struktūra ir turinys.....	18
2.3.1.1	Skyrelio “Operacinės sistemos naudojimo reikalavimai” paskirtis ir turinys	18
2.3.1.2	Skyrelio “Sąveikos su duomenų bazėmis reikalavimai” paskirtis ir turinys..	18
2.3.1.3	Skyrelio “Dokumentų mainų reikalavimai” paskirtis ir turinys.....	18
2.3.1.4	Skyrelio “Darbo kompiuterių tinkluose reikalavimai” paskirtis ir turinys....	18
2.3.1.5	Skyrelio “Programavimo aplinkos reikalavimai” paskirtis ir turinys.....	18
2.3.2	Poskyrio “Veikimo reikalavimai” paskirtis, struktūra ir turinys.....	18
2.3.2.1	Skyrelio “Tikslumo reikalavimai” paskirtis, struktūra ir turinys.....	18
2.3.2.2	Skyrelio “Patikimumo reikalavimai” paskirtis ir turinys.....	19
2.3.2.3	Skyrelio “Robastiškumo reikalavimai” paskirtis ir turinys.....	19
2.3.2.4	Skyrelio “Našumo reikalavimai” paskirtis ir turinys.....	19
2.3.3	Poskyrio “Diegimo reikalavimai” paskirtis, struktūra ir turinys.....	19
2.3.3.1	Skyrelio “Ruošinio reikalavimai” paskirtis ir turinys.....	19
2.3.3.2	Skyrelio “Instaliavimo reikalavimai” paskirtis ir turinys.....	19
2.3.3.3	Skyrelio “Pradinio duomenų bazių kaupimo reikalavimai” paskirtis ir turinys	19
2.3.3.4	Skyrelio “Sistemos įsisavinamumo reikalavimai” paskirtis ir turinys.....	19
2.3.4	Poskyrio “Aptarnavimo ir priežiūros reikalavimai” paskirtis, struktūra ir turinys	19
2.3.5	Poskyrio “Tiražuojamumo reikalavimai” paskirtis, struktūra ir turinys.....	19
2.3.6	Poskyrio “Apsaugos reikalavimai” paskirtis, struktūra ir turinys.....	20
2.3.7	Poskyrio “Juridiniai reikalavimai” paskirtis, struktūra ir turinys.....	20
3	Programų sistemos projektavimas.....	21
3.1	Programų sistemos projekciniai reikalavimai.....	21
3.1.1	Programų sistemos dekompozicija.....	21
3.1.2	Reikalavimų lokalizavimo matrica.....	21
3.1.3	Reikalavimų ryšio matrica.....	22
3.2	Programų sistemos architektūra.....	22
3.2.1	Užduotys ir jų vykdymo scenarijai.....	22
3.2.2	PS struktūros modelis.....	23
3.3	Programų sistemos įgyvendinimo architektūra.....	23
3.4	Programų sistemos prototipas.....	23
4	Testavimas.....	24
4.1	Testavimo scenarijai.....	24
4.2	Testavimo atvejai.....	24
4.3	Automatinis testavimas.....	24

5	Diegimas – kodo ir veikiančios sistemos demonstravimas.....	25
6	Išvados.....	26
7	Naudoti informacijos šaltiniai.....	27
8	Priedai.....	28

Išvadas

Studijų dalyko tikslas

Igyti naujus bei išvystyti turimus gebėjimus bei įgūdžius programų sistemoms (PS) kurti ir naudoti PS analizės, projektavimo metodus, programinei įrangai kurti.

Studijų dalyko aprašas

Modulis skirtas studentų gebėjimams bei įgūdžiams projektuoti bei realizuoti programų sistemas (PS) formuoti ir vystyti. Modulio vykdymo metu studentai turi praktikoje panaudoti studijų metu įgytas žinias apie pagrindinius PS analizės ir projektavimo metodus, technikas ir metodikas, bei išvystyti struktūrinio ir objektinio PS projektavimo gebėjimus. Be to, studentai įsigys praktinius įgūdžius dirbti analitikų ir projektuotojų vaidmenyse PS kūrėjų komandoje, kuri projektuoja modernias technologijas naudojančią PS, skirtą pasirinktos verslo sistemos (VS) informacijos apdorojimo poreikiams tenkinti. Studentai vystys PS kūrimo įgūdžius.

Darbo tikslai ir paskirtis

Šiuo darbu siekiama:

- mokyti ir gilinti prieš tai įgytas žinias ir gebėjimus analizuoti verslą, specifiкуoti naudotojų poreikius, specifiкуoti sistemos reikalavimus, projektuoti sistemą pagal reikalavimų specifiкуaciją, įgyvendinti suprojektuotą sistemą, ištestuoti ją;
- gilinti UML kalbos žinias ir gebėjimus;
- rengti techninę dokumentaciją;
- dirbti pagal formalius reikalavimus;
- dirbti grupėse ir savarankiškai;
- planuoti darbą ir baigti jį numatytais terminais;
- viešai pristatyti darbą, argumentuoti priimtus sprendimus.

Darbų atlikimo ir vertinimo tvarka

Studentai darbą atlieka komandomis po 3 studentus. Individualiai (po 1 studentą) atlikti darbai nėra vertinami. Konkretus kiekvieno studento indėlis į darbą privalo būti aprašytas to darbo anotacijoje.

Kiekviena komanda pasirenka uždavinį, susijusį su tam tikra dalykine sritimi. **Uždavinys turi būti toks, kad jam spręsti skirtai PS būtų galima suformuluoti visas nefunkcinių reikalavimų rūšis (našumas, patikimumas, apsauga, teisiniai reikalavimai, interfeiso reikalavimai), bei yra naudotojo interfeisas.**

Prieš atliekant toliau darbus, pasirinktas uždavinys (tema) yra pateikiamas dėstytojui patvirtinti. **Studentai toliau gali atlikti numatytus KP darbus tik dėstytojui patvirtinus pasirinktą uždavinį (temą).** Tema tvirtinama per Moodle.

Toliau studentai pagal numatytą darbų atlikimo grafiką (kuris skelbiamas Moodle), rengia numatytas KP dalis ir jas atsiskaitinėja. Kiekvienai daliai yra rengiamas atskiras dokumentas, pagal pateiktą paskaitų metų šablono, pristatomas ir įvertinamas pažymiu.

Norėdamas atsiskaityti, komanda atitinkamos dalies ataskaitą talpina į Moodle. Pristatymo/atsiskaitymo trukmė 10 minučių. Darbą pristatinėja **VISI** komandos nariai, kiekvienas savo dalį. Jiems pateikiami klausimai. Atsiskaitymo metu dėstytojas peržiūri Moodle patalpintą darbo ataskaitą, pateikia pastabas (jeigu yra) ir užduoda klausimus. Kiekviena ataskaita yra vertinama, atsižvelgiant į jos esmę, gebėjimus apipavidalinti pagal pateiktus reikalavimus, taisyklingą lietuvių kalbą, taisyklingą terminų vartoseną. **Ataskaitos, neatitinkančios slenkstinį pasiekimų lygmenį** (t.y. yra NETENKINAMAS bent vienas iš esamų reikalavimų: (1) aprašyti visi numatyti pagal reikalavimus skyriai bent 50 proc., (2) darbe pateiktos visos numatytos UML diagramos, (3) UML diagramos yra sudarytos su MagicDraw, (4) UML diagramose yra mažiau

negu 5 esminės klaidos, (5) tekste yra mažiau kaip 10 gramatinių arba sintaksės klaidų, arba (6) ataskaita yra tvarkinga), **yra pateikiamos atgal studentui su pastabomis taisymui ir gynimui kitą numatytą pagal grafiką datą.** Už darbą visi grupės nariai gauna tą patį pažymį (nuo 0 iki 10 balų).

Pavėlavus darbą apginti daugiau kaip 1 savaitę, pažymys sumažinamas 1 balu, pavėlavus daugiau kaip 2 savaites – 2 balais. Išimtys daromos tik ligos ar kitais ypatingais atvejais.

Visos kompleksinio projekto dalys yra atsiskaitomos **trimis iteracijomis (sprintais).** Kiekvienos iteracijos metu studentas praeina PS įgyvendinimo ciklą nuo pradžios (poreikio) iki pabaigos (demonstracijos). Už kiekvieną iteraciją yra rašomas pažymys nuo 0 iki 10. **Galutinį pažymį sudaro trijų pažymių vidurkis. Vieno atsiskaitymo metu galima atsiskaityti tik už vieną iteraciją. Iteracija nėra vertinama pažymiu, jeigu ji yra atlikta nepilnai.** Pirmos dvi iteracijos turi būti pristatytos ir apgintos **semestro metu (iki sesijos) pagal pateiktą darbų atlikimo grafiką. Po semestro pabaigos šios dalys nėra vertinamos.** Trečia iteracija gali būti pristatyta egzamino metu.

Studentų pasiekimų vertinimo formulė

Studentų žinios vertinamos remiantis dešimtbale kriterine vertinimo skale, taikant kaupiamojo balo sistemą.

Iteracija
(i – bauda)

$$GP = \frac{1}{3} \sum_{i=1}^3 i$$

kur GP – galutinis pažymys, $Iteracija_i$ – PS kūrimo iteracija, $bauda_i$ – bauda už vėlavimą.

Už visas darbo dalis studentas turi surinkti teigiamus (≥ 5 balus) vertinimus, kad galutinis pažymys būtų teigiamas.

Darbų vertinimo tvarka

Puikus pasiekimų lygmuo (9-10) – (1) aprašyti visi numatyti skyriai bent 90 proc., (2) pateiktos visos numatytos UML diagramos, (3) **UML diagramos yra MagicDraw Scrum projektas**, (4) UML diagramose nėra esminių klaidų, (5) tekste yra ne daugiau 2 gramatinių arba sintaksės klaidų, ir (6) ataskaita yra tvarkinga. Studentas pagrindžia kiekvieno projekto gyvavimo ciklo etape taikomų veiksmų metodikos pasirinkimą, atsako į visus pateiktus klausimus.

Žinios ir su jomis susiję praktiniai gebėjimai yra išsamūs, neapsiribojama informacija pateikiama per studijas. Darbas pristatytas ir apgintas laiku pagal pateiktą grafiką iki sesijos pradžios.

Tipinis pasiekimų lygmuo (7-8) – (1) aprašyti visi numatyti skyriai bent 70 proc., (2) pateiktos visos numatytos UML diagramos, (3) **UML diagramos yra MagicDraw Scrum projektas**, (4) UML diagramose yra mažiau negu 2 esminės klaidos, (5) tekste yra mažiau kaip 5 gramatinių arba sintaksės klaidų, ir (6) ataskaita yra tvarkinga. Studentas paaiškina visą programinių projektų gyvavimo ciklą ir jo metu vykdomas veiklas. Darbas pristatytas ir apgintas laiku iki sesijos pradžios pagal pateiktą grafiką.

Slenkstinis pasiekimų lygmuo (5-6) – (1) aprašyti visi numatyti skyriai bent 50 proc., (2) pateiktos visos numatytos UML diagramos, (3) **UML diagramos yra MagicDraw Scrum projektas**, (4) UML diagramose yra mažiau negu 5 esminės klaidos, (5) tekste yra mažiau kaip 10 gramatinių arba sintaksės klaidų, ir (6) ataskaita yra tvarkinga. Studentas dalinai pagrindžia pasirinkimus projektavimo ir įgyvendinimo etapuose. Darbas apgintas vėluojant.

Ivado struktūra

Ivada sudaro šie poskyriai:

- PS pavadinimas
- Dalykinė sritis

- Probleminė sritis
- Naudotojai
- Darbo pagrindas
- Naudoti dokumentai
- Akronimai

PS privalo turėti du pavadinimus: pilną (pvz., "Gamyklos sandėlio apskaitos sistema") ir trumpą (pvz. sistema "Sandėlis"). Pilnas pavadinimas naudojamas tituliname lape ir šiame poskyryje, trumpas – dokumento tekste.

Poskyryje "Dalykinė sritis" aprašoma sritis (pvz. buhalterinė apskaita), kurioje numatoma naudoti kuriamą PS. Poskyryje "Probleminė sritis" aprašomi uždaviniai (problema), kuriuos privalo spręsti kuriamoji PS.

Poskyryje "Naudotojai" nurodoma kokie dalykinės srities specialistai (pvz., sandėlininkas) naudosis kuriama sistema ir kokią kvalifikaciją (mokyklinis informatikos kursas, ECDL, koledžo diplomas, bakalauro diplomas, magistro diplomas, koks nors sertifikatas) **informatikos srityje** jie privalo turėti, kad galėtų ta sistema tinkamai naudotis. Naudotojai ir jų kvalifikacija yra pateikiami tokia lentelė (žr. 0-1 lentelė):

0-1 lentelė. Naudotojai ir jų IT kvalifikacija

Nr.	Naudotojas (t.y. dalykinės srities specialistas)	IT kvalifikacija
1.		
2.		
...		

Poskyryje "Darbo pagrindas" nurodoma, kad dokumentas yra parengtas kaip dalyko „Informacinės sistemos programų kūrimas“ kompleksinis projektas.

Poskyryje "Naudoti dokumentai" pateikiami bibliografiniai aprašai dokumentų (pvz., buhalterinės apskaitos įstatymas), į kuriuos daromos nuorodos darbo tekste. Bibliografiniai aprašai turi būti pateikiami **APA stiliumi**, naudojant teksto redaktoriaus **Citations&Bibliography**. **Poskyryje aprašomi tik tie dokumentai, į kuriuos toliau daromos nuorodos**. Cituoti neaprašytus dokumentus ar daryti į juos nuorodas negalima.

Poskyryje "Akronimai" aprašomi darbe įsivesti trumpiniai.

Darbo bendrosios dalies struktūra

Darbo bendrąją dalį sudaro šie skyriai:

1. Produkto poreikių specifikacija
2. Reikalavimų specifikacija
3. Programų sistemos projektas
4. Testavimas ir diegimas
5. Išvados

Atkreipkite dėmesį, kad kiekvienas skyrius turi būti logiškai gaunamas iš ankstesnių skyrių. Visi skyriai sudaro logiškai susietą visumą ir pirmuosiuose skyriuose pateikta medžiaga turi būti panaudota kituose darbuose.

INITIAL PRODUCT SPEC - A MODEL SPEC

This document was created by Monterail as an example of a filled out product spec. It was not submitted by any company to build an app and is not using any confidential information.

Table of Contents

1. Glossary
2. Purpose of the document
3. Deadline
4. General Context
 - About the project
 - Goals of the product
 - Strategy
 - Market
 - User personas
 - Business model
 - Budget
 - Timeline
5. Functional requirements and scope
 - User roles
 - Features: user interface
 - Features: admin panel
 - Integrations
6. Non-functional Requirements
 - Usage and accessibility
 - Design
 - Benchmarks
7. Additional materials
8. FAQ

Terminų ir trumpinių žodynėlis

Verslo žodynas

1 Produkto poreikio specifikacija

1.1 Pasirinktos dalykinės srities specifikacija

Šiame skyriuje apibūdinama pasirinkta dalykinė sritis: pobūdis, apimtys ir kt.

Dalykinė sritis yra vaizduojama prieš kuriant programų sistemą, t.y. **esamos būsenos** (situacijos) **vaizdusis paveikslėlis**, ir **siekiamos būsenos vaizdusis paveikslėlis**. Abu paveikslėliai yra aprašomi tekstu apie 5-10 sakinius. Po pirmo vaizdžio paveikslėlio turi būti aprašyta esanti probleminė situacija, kurią spęs kuriama PS.

Esamos situacijos vaizdusis paveikslėlis (VP) turi atitikti šiuos reikalavimus:

1. VP turi būti apibrėžtos nagrinėjamos *dalykinės srities ribos*.

2. VP turi būti išskirti ir įvardinti pagrindiniai *aktoriai* ir *objektai*. Aktorių ir objektų pavadinimai turi būti užrašyti po jį vaizduojančiu paveikslėliu.
3. VP turi būti nurodyti *ryšiai* tarp aktorių ir/arba objektų. Ryšiai būtinai turi turėti pavadinimą, bei tipą: materialieji srautai (ištininė linija) ir nematerialieji (informaciniai) srautai (brūkšnine linija).
4. VP turi būti pažymėtos *problemos* (konflikinės situacijos) ir/arba svarbūs įvairių rūšių *klausimai*.
5. VP turi būti pavaizduoti *išorės stebėtojai*.
6. VP apačioje turi būti *elementų paaiškinimas*.
7. VP esantys elementai turi būti paveikslėliai, o ne UML notacija.

Siekiamos būsenos VP turi atitikti tuos pačius reikalavimus, kaip ir esamos būsenos VP, išskyrus 4 reikalavimą, kadangi jis vaizduoja situaciją, kai probleminės ir konflikinės situacijos yra išspręstos, įdiegiant sukurta PS.

Pagal sudarytą vaizdų paveikslėlį, turi būti parašytas **esminis verslo apibrėžimas**, kurio struktūra yra tokia:

Veikianti *globalioje aplinkoje {W}* ir priklausanti *savininkui {O}* Sistema *aktoriaus {A}* pagalba atlieka *verslo procesą {T}*, esant *išoriniams ribojimams {E}* tam, kad patenkinti *kliento {C}* poreikius, pagaminant/suteikiant *produktą/paslaugą {P}*.

Esminio verslo apibrėžimo aiškinimas yra pateikiamas 2 1 lentelėje.

2-2 lentelė. Esminio verslo apibrėžimo aiškinamoji lentelė (JOS PILDYTI IR DARBE PATEIKTI NEREIKIA)

Nr.	Subjektas / objektas	Apibūdinimas	Pavyzdys
1.	Klientas (C)	Kam tai yra daroma? Veiklos iniciatorius arba veiklos veikiamas. Tai galutinis produkto/paslaugos gavėjas	Abiturientas
2.	Aktorius (A)	Kas vykdo procesą? Aktorius, vykdomi verslo procesą, t.y. transformaciją. Tai, kas iš tikrųjų dirba.	Dėstytojas
3.	Verslo procesas (T)	Kas vyksta arba kas į ką transformuojama? Vykstantys sistemos pokyčiai. Įeiga → Išeiga	Studijų procesas
4.	Globalinė aplinka (W)	Kokioje globalioje aplinkoje vyksta veiksmas?	Universitetas
5.	Savininkas (O)	Kas yra atsakingas už sistemoje naudojamus išteklius?	Rektorius
6.	Išoriniai ribojimai (E)	Etiniai ribojimai, įstatymai, resursų ribojimai,...	Aukštojo mokslo įstatymas
7.	Produktas/paslauga (P)	Kas yra verslo proceso galutinis rezultatas?	Absolventas

Pavyzdys:

Veikiantis *universitete* ir priklausanti *rektoriui* Sistema *dėstytojo* pagalba atlieka *studijų procesą*, veikiant *Aukštojo mokslo įstatymui* tam, kad patenkinti *abituriento* poreikius, išugdant *absolventą*.

1.2 Kuriamo PS produkto specifikacija

Kontekstas ir turinys

<https://uxdesign.cc/how-i-write-product-specifications-d7d759c7d471>

1.2.1 Kuriamo produkto strategija

Šiame poskyryje formuluojama kokios strategijos yra laikomasi, kuriant produktą.

Verslo strategija detalizuojama, išreiškiant ją strateginių tikslų rinkiniu. Strateginiai tikslai detalizuojami, kiekvieną iš jų išreiškiant operacinių tikslų rinkiniu. Kitaip tariant, siekiamų verslo tikslų visuma sudaro medį.

Toliau yra vykdoma užsakovo poreikių analizė ir aprašytiems operaciniams tikslams įgyvendinti nustatomos savybės (1-2 lentelė).

1-3 lentelė. Verslo problemos, grėsmės, siekiai ir savybės

Nr.	Verslo problema arba grėsmė	Verslo operacinis tikslas	Savybė, reikalingas tikslui įgyvendinti
1.		Operaciniai tikslai yra konkretūs, konstruktyvūs, terminuoti, matuojami. Su viena verslo problema arba grėsme gali būti susiję keli operaciniai tikslai.	Funkcija/savybė, duomenys/ataskaita sistemoje ir pan.

Savybė (angl. *feature*) – tai yra tai ko naudotojas nori arba jam reikia. Ji yra apibūdinama pagal tokią formulę:

<veiksmas> <rezultatas> <objektas>

Pagal naudotojo istoriją (angl. *user story*):

Kaip autorius aš noriu tvarkyti knygos tekstą skyriais ir poskyriais.

Gali būti apibūdinama tokia savybė:

Sukurti talpyklą dokumentams.

Turi būti aprašytos bent 2 verslo problemos arba grėsmės. Kiekvienai verslo problemai ir grėsmei turi būti surašyta bent po 2 operacinius tikslus, o kiekvienam operaciniam tikslui – bent po 2 savybes. Pasiskirstymas gali būti toks, kad kiekvienas studentas pogrupyje aprašo po 1 verslo problemą su pilna detalizavimu. Tokios analizės pavyzdys yra pateikiamas 1-3 lentelėje.

1-4 lentelė. Verslo problemos, grėsmės, siekiai ir savybės

Verslo problema arba grėsmė	Verslo operacinis tikslas	Sprendimas sistemoje
1. Daug transporto priemonių techninių gedimų	1.1. Sumažinti transporto priemonių techninius gedimus nuo 50 proc iki 10 proc.	1.1.1. Automobilių techninės apžiūros priminimo funkcija. 1.1.2. Sutarčių valdymo funkcija.
	2.1. Pagreitinti transporto priemonių gedimų šalinimą nuo 2 savaičių iki 3 dienų.	1.1.3. Laisvų transporto priemonių ataskaitų sudarymo funkcija 1.1.4. El. klientų aptarnavimas
4. Daug nesusistemintų duomenų	4.1. Pagerinti duomenų saugojimo struktūrą nuo visai nestruktūrizuotų duomenų iki struktūrizuotų duomenų.	4.2.1. Duomenų bazė 4.2.2. Ataskaitų sudarymo funkcija 4.2.3. Dažniausiai sudaromų ataskaitų šablonų parengimas
	4.2. Pagerinti/pagreitinti ataskaitų sudarymą nuo 3 dienų iki 30 min.	4.2.4. Automatinis pasirinktų ataskaitų sudarymas mėnesio/savaitės pabaigoje
5.	5.1.	5.1.1.

1.3 Produkto naudojimo scenarijus

Šio skyriaus paskirtis – nustatyti, kaip bus dirbama organizacijoje įdiegus kuriamą PS produktą ir ką reikia padaryti, kad būtų galima pradėti šitaip dirbti. Pateikiama **UML sekų diagrama** aprašo, kaip bus dirbama su sistema ją įdiegus. Sistemai ir kiekvienai darbo vietai diagramoje imama atskira objekto gyvavimo atkarpa. Parodomos ne tik tiesiogiai su sistema

dirbančių asmenų darbo vietos, bet ir asmenų ruošiančių sistemai pateikiamus duomenis ar besinaudojančių jos darbo rezultatais darbo vietos.

Šiame poskyryje nustatoma, ką reikia padaryti (išsigyti techninę bei programinę įrangą, įrengti tinklus ar darbo vietas, pasamdyti naujus darbuotojus, apmokyti darbuotojus, sukaupti duomenis į duomenų bazes ir kt.), kad būtų galima pradėti dirbti pagal aukščiau aprašytą scenarijų. Techninė ir sisteminė programinė įrangą specifikuojama bendrais bruožais, nenurodant konkrečių detalių, nes jos čia dar negali būti žinomos.

1.4 PS įgyvendinamumo analizė

Šio skyriaus paskirtis – išsiaiškinti ar reikiamą programų sistemos produktą tikrai įmanoma sukurti, kokią konkrečią naudą jis duos ir ar jį verta kurti.

1.4.1 Operacinis įgyvendinamumas

Šiame poskyryje analizuojami PS diegimo inovaciniai slenksčiai ir parodoma, kad jie gali būti pašalinti. Čia reikia įvardinti operacines rizikas ir nustatyti jų pašalinimo būdus. Tai yra, reikia atsakyti į du pagrindinius klausimą:

1. Kokios galimos rizikos, kad vartotojas nenorės naudotis sistema?
2. Kaip pašalinti įvardintas rizikas?

Kitaip tariant, parodoma kad siūlomas sistemos naudojimo scenarijus bus priimtinas tiems, kas pagal jį dirbs ir kad jie galės pagal šį scenarijų dirbti efektyviai.

Rizikos gali būti išvardinamos, kaip pateikta 1-4 lentelėje.

1-5 lentelė. Projekto operacinių rizikų valdymo planas

Rizika	Rizikos įvertinimas			Prioritetas	Sumažinimo priemonės
	Tikimybė	Įtaka	Vertė		
1. Vyresnio amžiaus naudotojai nesugebės naudotis sistema	Didelė	Vidutinė	Vidutinė	Aukštas	✓ Kompiuterinio raštingumo mokymai ✓ Sistemos naudojimo demonstracijos ✓ ...
					✓ ✓ ✓
					✓ ✓ ✓

Nagrinėjant įgyvendinamumą, **turi būti įvardintos bent 3 rizikos**, kurios gali atsirasti darbo vykdymo metu. Atlikite jų įvertinimą bei pateikite priemones joms sumažinti.

1.4.2 Techninis įgyvendinamumas

Šiame poskyryje reikia parodyti, kokiais techniniais būdais galima įgyvendinti kuriamą PS, t.y. kokios yra techninės alternatyvos PS įgyvendinti. Turi būti aprašytos ir palygintos bent **trys** techninės alternatyvos (1-5 lentelė).

1-6 lentelė. Techninių alternatyvų palyginimas

Nr.	Alternatyva, jos trumpas aprašas	Palyginimo kriterijus 1	Palyginimo kriterijus 2	Palyginimo kriterijus 3	...
1.		Kaina, darbų apimtis, ...			

1.4.3 Juridinis sistemos pagrindimas

Šiame poskyryje reikia parodyti, kad sukuriant sistemą nebus pažeisti LR Konstitucijos (valdžių atskyrimo principas), Asmens duomenų apsaugos įstatymo, Statistikos įstatymo ar kitų LR teisės aktų numatyti draudimai. Rizikos gali būti išvardinamos, kaip pateikta 1-6 lentelėje.

1-7 lentelė. Projekto juridinių rizikų valdymo planas

Rizika	Rizikos įvertinimas			Prioritetas	Sumažinimo priemonės
	Tikimybė	Įtaka	Vertė		
					✓ ✓ ✓
					✓ ✓ ✓
					✓ ✓ ✓

1.5 Panaudojamumo ir teikiamos naudos analizė

Šiame poskyryje pateikiama kuriamo PS produkto **UML užduočių diagrama**, kurioje parodoma, kokie aktoriai ir kokias užduotis vykdys naudodamiesi PS produktu.

Po diagrama yra pateikiama viena lentelė (1-7 lentelė), kuri paaiškina sistemos panaudojimą, ir antra lentelė (1-8 lentelė) – sistemos teikiamą naudą.

1-8 lentelė. Sistemos panaudojimas

Užduotys	Sistemos panaudojimas
	kaip sistema padės tas užduotis vykdyti

1-9 lentelė. Sistemos teikiama nauda

Aktoriai	Gaunama nauda
	kokią konkrečią naudą (operatyvesnę informaciją, išsamesnę informaciją ir pan.) kiekvienas aktorius iš to gaus

1.6 Išvados ir siūlymai

Šiame skyriuje, naudojantis kituose skyriuose pateikta medžiaga, parodoma, kad sistemą tikrai yra verta kurti.

2 Darbo atlikimo tvarka

Darbui atlikti turi būti sukuriamas **MagicDraw Scrum projektas**, kurio struktūra yra pateikiama (1.1 pav.). Šis projektas yra sukurtas pagal MagicDraw esantį šabloną, todėl jame jau yra pavyzdžiai. **Studentai patys turi peržiūrėti sukuriamą projektą ir išsinagrinėti pateiktus pavyzdžius. Dėstytojas čia veikia, kaip pagalbininkas, bet ne viską aiškinantis. Studentai, turi, visu pirma, peržiūrėti esamus aprašus savarankiškai.**



1.1 pav. MagicDraw Scrum projekto struktūra.

Toliau studentai dirba pagal Scrum metodiką (**kiek tai įmanoma studijų aplinkoje**). Studentai turi kurti pasirinktos programų sistemos prototipą per tris iteracijas – sprintus. Kiekvienas sprintas apima tokius žingsnius, kaip:

1. **Naudotojų istorijų** aprašymas ir pasirinkimas sprintui;
2. **Reikalavimų modeliavimas**;
3. **Architektūros modeliavimas**;
4. **Įgyvendinimas**
5. **Ataskaitos rengimas** ir pristatymas

2.1 Naudotojų istorijos



Čia, pagal atliktą dalykinės srities analizę 2 skyriuje, yra aprašomi naudotojų poreikiai per naudotojų istorijas. Kiekvienas pogrupio **studentas turi aprašyti bent po 3 naudotojų istorijas**, susijusias su verslo problemomis, grėsmėmis ir siekiais.

Naudotojo istorija (angl. *User Story*) – tai struktūrizuotas naudotojo poreikio aprašas sakiniu. Naudotojo istorija dažniausiai yra tokios formos:

Aš [asmuo/rolė], noriu [poreikis], tam kad [tikslas].

As a [type of user/role], I want [user's need] so that [user's goal].

Naudotojų istorijos MagicDraw Scrum projekte atrodo taip (3.2 pav.).

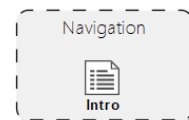
User Stories										
Criteria										
Element Type: UserStory			Scope (optional): User Story package			Filter: <input type="text"/>				
#	ID	Name	Documentation	Acceptance Criteria	User Story Owner	Risk	Rank	Sprint	Release	Reference
1	US-1	 Make book reservation	As a registered user I want to make a book reservation so that I do not have to wait in line at the library.	1. A user cannot submit the reservation form without completing all the mandatory fields. 2. A confirmation email is sent to the user after submitting the reservation form.	Billy Bob	High	Medium	1	v0.1	http://jira.project.com/123
2	US-2	 Another User Story								

3.2 pav. MagicDraw Scrum naudotojų istorijos pavyzdys.

2.2 Sprintų planavimas

Toliau turi būti planuojami sprintai (3.3 pav.). Studentai turi paskirstyti naudotojų istorijų įgyvendinimą į trys sprintus.

Sprint Planning



The Sprint planning view represents what user stories are planned for development in each sprint.

Plan your sprint using packages

Drag user stories from the User Story package into your sprint table.



3.3 pav. MagicDraw Scrum Sprintų planavimo pavyzdys.

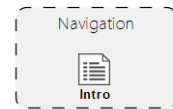
Pasirenkamas pirmas sprintas vykdymui. Analogiškai vykdomi ir kiti du sprintai.

2.3 Reikalavimų modelis

Pasirinkus sprintą vykdymui, peržiūrimi šiame sprinte esančios naudotojų istorijos. Jos transformuojamos į reikalavimų modelį (3.4 pav.).

Requirements Model

The Requirements model refines user stories. It is a set of Use Case and Activity diagrams, and traceability from user stories to use cases.

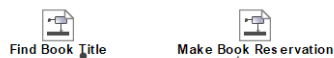


Use Cases



Use cases show a big picture of the value provided by the system to the actors. Use cases show who uses the system and what they can do with it.

Use Case Scenarios



The Activity diagram allows capturing workflows between actors and the system without further specifying what the system exactly does or performs to achieve the goal/result of the use case.

You can choose a style for the Activity diagram.

3.4 pav. MagicDraw Scrum reikalavimų modelio pavyzdys.

Reikalavimų modelis yra aprašomas per užduočių modelį. Užduočių modelį sudaro:

1. UML esminių užduočių diagrama
2. Kiekvienos esminės užduoties modelis ir aprašas.

2.3.1 Esminės užduotys


Kaip atrodo esminių užduočių diagrama, žiūrėti pavyzdį **MagicDraw Scrum projekte**. Ji rodo bendrą sistemos naudotojams suteikiamos vertės vaizdą. Žiūrėdami į diagramą, mes matome kokie naudotojai gali naudotis su sistema, vykdydami tam tikras užduotis, t.y. ką naudotojai gali padaryti su sistema.

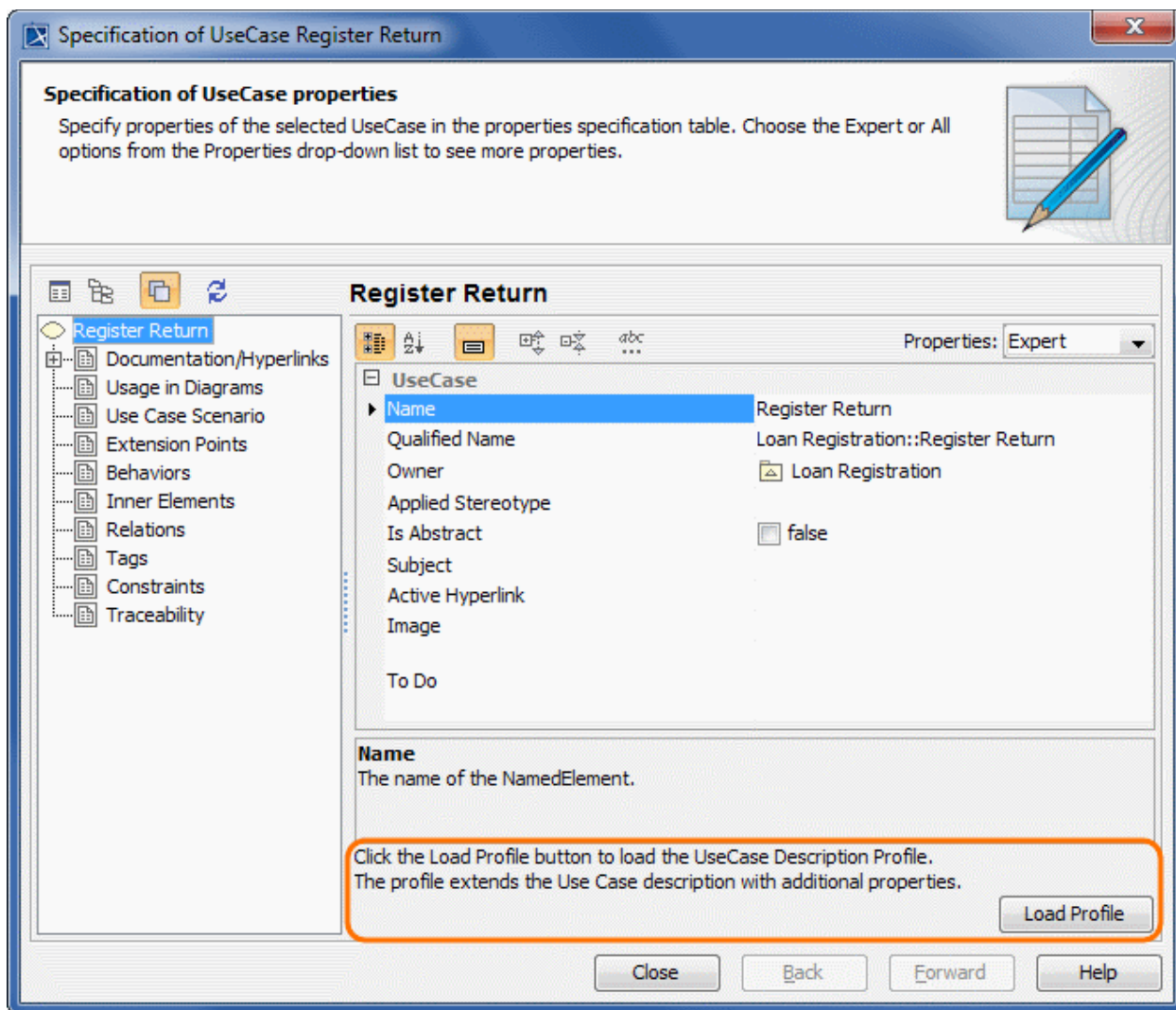
Kiekviena užduotis užduočių diagramoje turi būti aprašyta **veiklos diagrama**, kaip pateikta MagicDraw Scrum projekte.

2.3.2 Užduoties modelis

Jeigu reikia, esminė užduotis yra skaidoma į smulkesnes použduotys, kurios taip pat aprašomos veiklos diagramomis.

Tam, kad MagicDraw įrankyje atsirastų visi reikalingi (papildomi) užduočių aprašai, kaip užduoties numeris, prieš sąlygos, po sąlygos ir kt., reikia paleisti užduočių aprašymo profilį (angl. *the Use Case Description Profile*) rankiniu būdu, kaip parodyta žemiau (3-1 pav.):

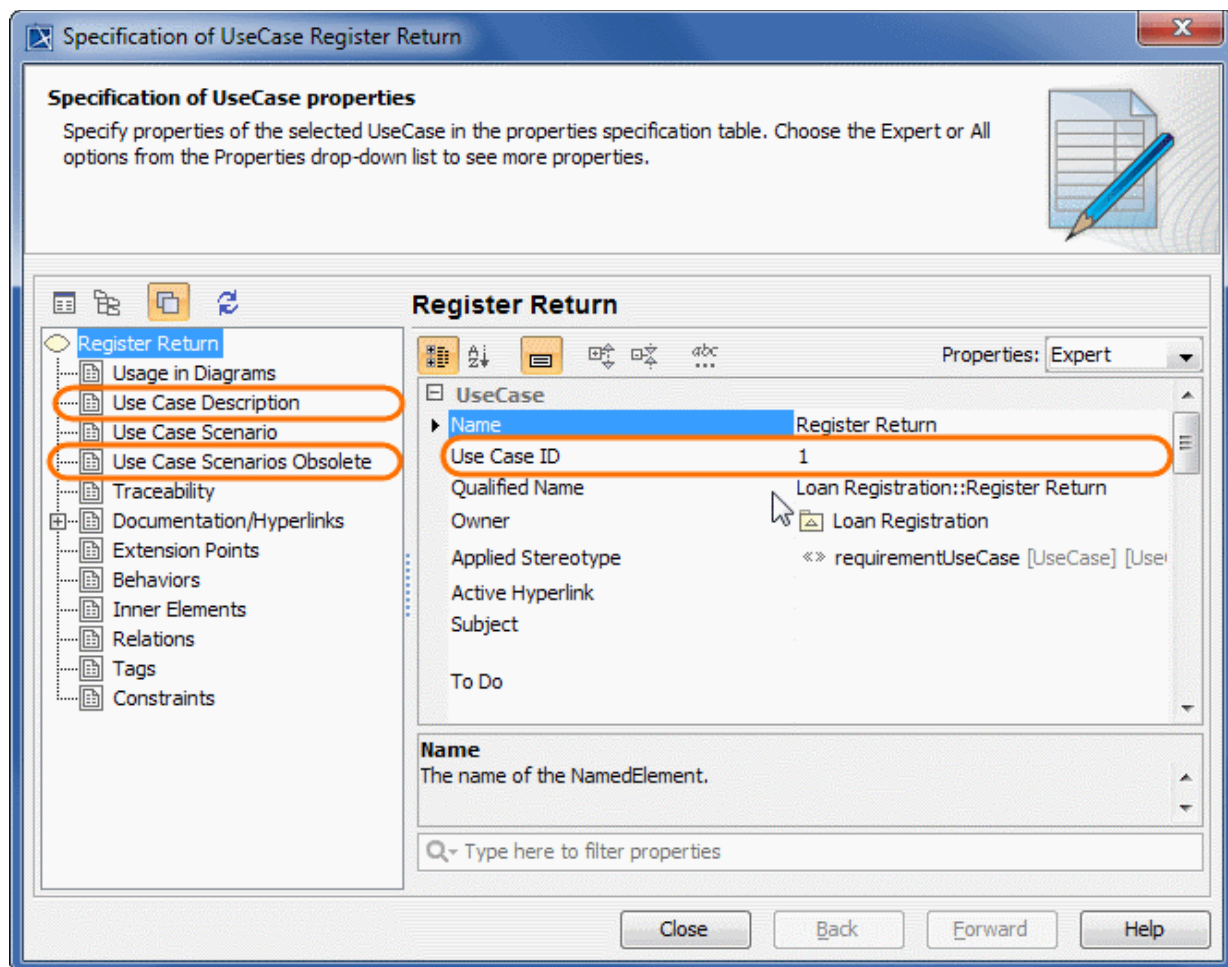
1. Atidaryti **Use Case**  **Specification window**.
2. Pasirinkti **Load Profile**. Užduočių aprašymo profilis bus pakrautas ir papildomos sąvybės įtrauktos į aprašą.



2-5 pav. Užduočių aprašymo profilio paleidimas

Įkėlus Užduočių aprašymo profilį (2-5 pav.), užduoties aprašymo lange bus rodomos šios papildomos savybės:

- **Use Case ID**
- **Use Case Description** – tai grupė savybių, kaip autorius (Author), data (Date), tikslas (Goal), ir kitos.
- **Use Case Scenario Obsolete** (As of MagicDraw or other modeling tool developed by No Magic Inc. version 17.0.2, Use Case scenarios of projects created with earlier versions are stored in the Use Case Scenario Obsolete property group, in the Use Case Specification window.). Naujoje versijoje to nėra.

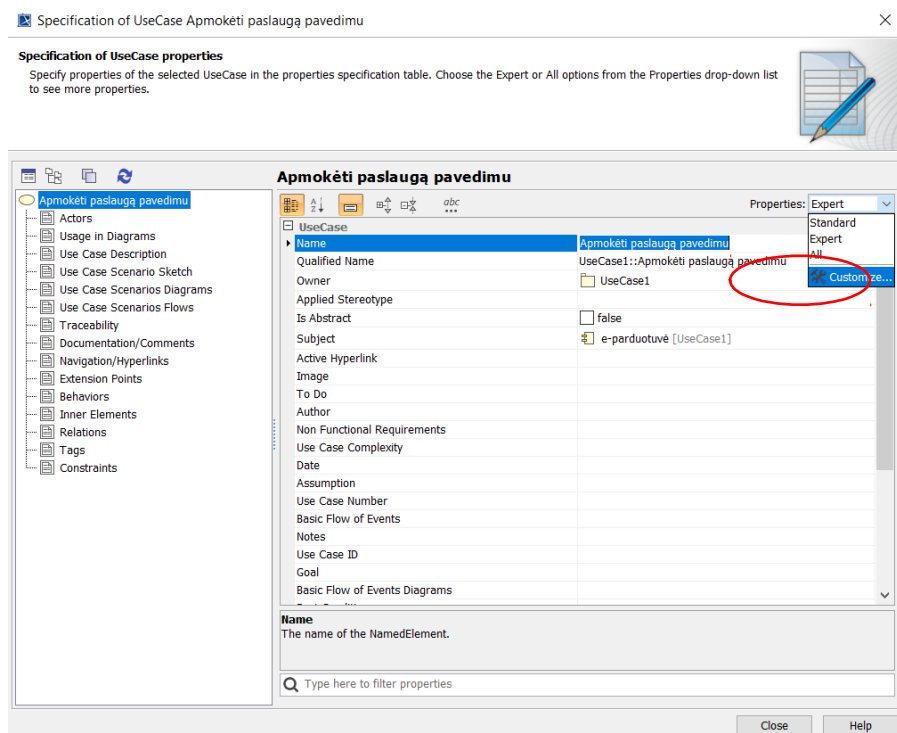


2-6 pav. Užduočių aprašymo profilio pavyzdys

Svarbu, kad į užduočių aprašą būtų įtrauktos papildomos reikalingos savybės (žr. 2-1 lentelę). Jeigu kai kurios savybės neatsiranda arba jų yra per daug, jas galima pridėti/pašalinti pasinaudojus **Expert** **Customize** (2-7 pav.).

2-10 lentelė. Užduočių aprašymo šablonas.

užduoties numeris	
užduoties pavadinimas	
siekiamas tikslas	
užduoties vykdymą inicijuojantis įvykis (triggeris)	Išorinis įvykis, inicijuojantis užduoties vykdymo pradžią
užduoties prioritetas	Svarba, lyginant ją su kitomis užduotimis.
užduoties vykdymo sritis	Visa verslo sistema arba kokia nors tos sistemos dalis.
užduoties lygmuo	sumarinis tikslas, naudotojo tikslas, sub-funkcija.
pirminis aktorius	Inicijuoja sąveiką su sistema tam, kad pasiektų savo tikslus.
antriniai aktoriai	be kurio pagalbos sistema negali įvykdyti jos vykdomų užduočių
"prieš" sąlygos	Sistemos arba aktorių būsenos, kurioms esant pradedama vykdyti užduotis.
sėkmingos baigties "po" sąlygos	Sistemos arba aktorių būsenos, baigus vykdyti užduotį.
pagrindinis sėkmės scenarijus	
nesėkmingos baigties "po" sąlygos	Sistemos arba aktorių būsenos, nepavykus įvykdyti užduotį.
bendresnioji užduotis	Užduotis, kurios po-užduotimi yra nagrinėjama užduotis.
použduotys («include» priklausomybė)	Siauresnės (dalinės) užduotys.
plėtiniai («extend» priklausomybė)	Po-užduotys, susidarius ypatingoms situacijoms.



2-7 pav. Užduočių aprašymo savybių papildymas

2.3.3 Poskyrio “Užduoties “.....” modelis” turinys ir struktūra

Šitokių poskirių turi būti tiek, kiek yra esminių užduočių.

Šiame kurse kiekvienas studentas grupėje privalo detalizuoti po vieną esminę užduotį iki elementarių užduočių.

Užduočių modeliavimas yra vykdomas MagicDraw įrankyje. Kiekvienai užduočiai ir jos požduotims turi būti sudaromi:

- užduoties realizavimo scenarijus, aprašytas **UML sekų diagrama** ir ją aiškinančiu tekstu, ir
- **použduotys**.

Poskyris kartojamas tiek kartų, kol yra požduočių. UML sekų diagramos yra aprašomos užduoties scenarijaus sekcijoje, o ne atskira nepriklausoma diagrama.

2.3.3.1 Skyrelio “Užduoties “.....” požduotys” turinys ir struktūra

Kiekviena esminė užduotis aprašoma hierarchine požduočių **UML užduočių diagrama** ir scenarijų hierarchija. Jei hierarchijos lygmenų yra daugiau negu du, tai šiame poskyryje atsiranda papildomi skyreliai vadinami “Požduoties “.....” modelis”. Kada baigti detalizuoti požduotis, sprendžiama vadovaujantis požduoties elementarumo kriterijumi. Kitaip tariant, analitikas mano, kad požduoties realizavimas yra akivaizdus, ji neturi nei plėtinių, nei variantų, nei požduočių ir gali būti traktuojama kaip reikalavimas atlikti kokį nors elementarų veiksmą.

Hierarchinis užduoties modeliavimas yra vykdomas su MagicDraw įrankiu ir po to generuojama ataskaita. Rankiniu būdu aprašinėti užduočių nereikia.

2.4 Architektūros modelis

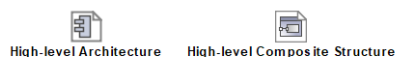
Aprašius reikalavimus per užduočių modeliavimą, yra kūriamas architektūros modelis (3.5 pav.). Šiose studijose studentai turi sudaryti:

- Konceptinio lygmens programų sistemos architektūra (angl. *High-Level Architecture*)
- Duomenų modelis (angl. *Data model*)
- Sąveikos modelis (angl. *Interaction*)
- Įgyvendinimo architektūra (angl. *Development Architecture*)

Architecture Model

The model of solution architecture represents the main components, integrations with other systems, interfaces, and deployment.

High-level Architecture



Shows the major components of the system and dependencies between them.

Data Model



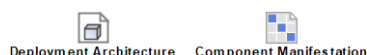
Classes represent types of objects that are handled in the system.

Interactions



Focuses on the interaction between different parts of the system.

System Deployment



Shows how the artifacts manifesting software components can be deployed on hardware devices and how these hardware devices are

3.8 pav. MagicDraw Scrum architektūros modelio pavyzdys.

2.5 Programų sistemos architektūra (High-level architecture)

Pateikiama **UML komponentų diagrama**, vaizduojanti į kokius komponentus yra dekomponuota PS, ir aprašomi kiekvieno komponento funkciniai ir nefunkciniai reikalavimai, gauti lokalizavus jame aukštesniojo lygmens reikalavimus ir nuleidus juos žemyn. Atkreipkite dėmesį į tai, kad kiekvienas komponentas vykdo tam tikras užduotis (po-užduotis), todėl sistemos dekompozicija turi atitikti skyriuje „Programų sistemos architektūra“ atliekamai užduočių dekompozicijai. Formuluoju reikalavimus, kiekvienas komponentas traktuojamas kaip „juodoji dėžė“, kurios struktūra atskleidžiama dekomponuojant jį į žemesnio lygmens komponentus. Dekompozicija atliekama vadovaujantis pasirinktąja programų sistemos architektūra. Rekomenduojama rinktis Koudo-Jodano architektūrą, bet galima rinktis ir bet kurią kitą architektūrą. Priklausomai nuo pasirinktos architektūros, komponentams yra formuluojami vadinamieji papildomi reikalavimai.

Žemiau yra pateiktas reikalavimo iš reikalavimų specifikacijos **dekompozicijos pavyzdys**:
Sistemos reikalavimas:

DR-01. Sistema turi leisti naudotojui užsiregistruoti, įvedant tokius duomenis kaip: vardas, pavardė, gimimo data, lytis, prisijungimo vardas, slaptažodis, el.paštas.

Šis reikalavimas dekomponuojamas į tokius **projektinius reikalavimus**:

GVS-1. Grafinės vartotojo sąsajos (GUI) komponentas turi leisti naudotojui užsiregistruoti, įvedant tokius duomenis kaip: vardas, pavardė, gimimo data, lytis, prisijungimo vardas, slaptažodis, el.paštas.

GVS-2. GUI komponentas turi perduoti naudotojo registracijos

duomenis Autentifikacijos komponentui.

AK-1. Autentifikacijos komponentas, gavęs iš GUI komponento vartotojo registracijos duomenis ... ir turi perduoti juos Duomenų bazės komponentui.

DB-1 Duomenų bazės komponentas, gavęs iš Autentifikacijos komponento vartotojo registracijos duomenis, turi juos išsaugoti.

Kitas sistemos reikalavimo dekompozicijos pavyzdys yra pateikiamas žemiau:

Sistemos reikalavimas:

001 SR. Sistemoje turi būti numatytos priemonės prieigai prie kitų sistemų (Navision, RP/3, etc.) sukurtų failų..

Šis reikalavimas dekomponuojamas į tokius *projektinius reikalavimus*:

F001 1PS. Per interfeisą Naudotojas turi turėti galimybę nurodyti išorinio failo tipą.

F002 1PS. Su kiekvieno tipo failu turi būti susieta jam apdoroti skirta priemonė

F003 1PS. Kiekvienas leistinas tipas ekrane turi būti pavaizduotas specialia piktograma.

F004 1PS. Naudotojas turi turėti galimybę apibrėžti naujus failų tipus ir su jais susieti naujas piktogramas.

F005 1PS. Naudotojui parinkus failą vaizduojančią piktogramą ir nurodžius failo pavadinimą, turi būti iškviesta tam failo tipui apdoroti skirta priemonė ir jai kaip parametras turi būti perduotas apdorojamo failo pavadinimas.

2.5.1 Reikalavimų lokalizavimo matrica

Šiame poskyryje pateikiama reikalavimų lokalizavimo matrica (žr. 3-1 lentelę).

3-11 lentelė. Reikalavimų lokalizavimo matrica

Reikalavimai	1 posistemis/ komponentas	2 posistemis/ komponentas	...	N posistemis/ komponentas
F001 1PS.	X			
F002 1PS	X			
F003 1PS.	X			

2.5.2 Reikalavimų ryšio matrica

Šiame poskyryje pateikiama reikalavimų ryšio matrica (2-3 lentelė). Stulpelis “Reikalavimo aprobavimo būdas” yra neprivalomas. Jei jis pildomas, tai nurodoma kaip numatoma patikrinti reikalavimą programų sistemos baigiamųjų bandymų metu. Stulpelis “Aprobavimo rezultatai” nepildomas.

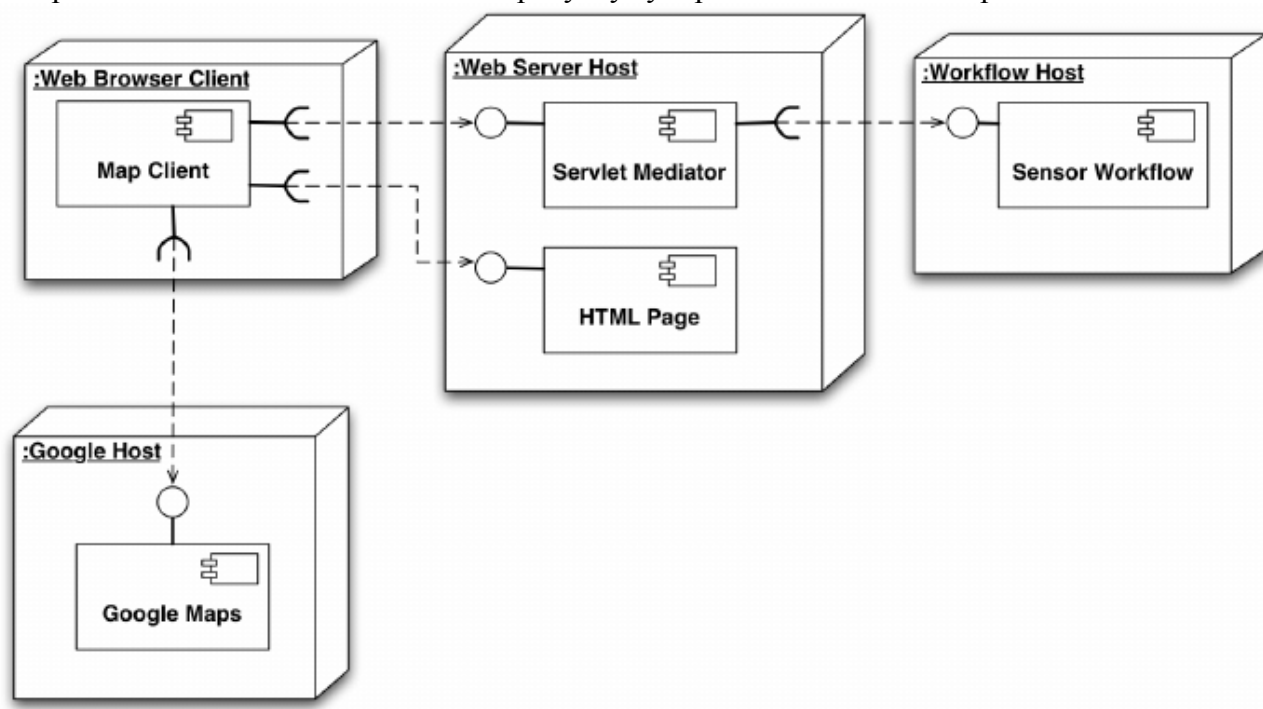
Pastaba. Atliekant reikalavimų lokalizavimą, jų nuleidimą žemyn ir aprašant reikalavimų aprobavimo būdus, gali paaiškėti, kad dokumente “Reikalavimų specifikacija” pateikti reikalavimai buvo suformuluoti netinkamai ir juos reikia taisyti.

2-12 lentelė. Reikalavimų ryšio matrica

Reikalavimas	Iš ko išvestas	Kur lokalizuotas	Reikalavimo aprobavimo būdas
F001 1PS.	001 SR	1PS	Testavimas
F002 1PS	001 SR	1PS	
F003 1PS.	001 SR	1PS	

2.6 Programų sistemos įgyvendinimo architektūra (Deployment architecture)

Šiame skyriuje yra pateikiama **UML įgyvendinimo diagrama** (angl. *deployment diagram*), parodanti kaip planuojama įgyvendinti projektinę architektūrą, kaip komponentai bus išskirstomi kompiuterio tinkle. Tokios architektūros pavyzdys yra pateikiamas žemiau 1 pav.



9 pav. Programų sistemos įgyvendinimo architektūros pavyzdys

2.7 PS struktūros modelis (Data model)

Skyriuje pateikiama UML **konceptinio lygmens** duomenų struktūros diagramos (**UML klasių diagrama**) ir ją paaiškinantis tekstas. Paaiškinamajame tekste trumpai aprašoma klasių paskirtis. Atkreipkite dėmesį, kad čia nagrinėjami klasės ir objektai, o ne sistemos komponentai.

Sąveikos diagramos (**Interactions**) aprašo kiekvienos klasės objektų galimas sąveikas.

2.8 Įgyvendinimas

2.8.1 Programų sistemos prototipas

Šiame skyriuje pateikiamas Sistemos prototipas. Prototipas turi pademonstruoti aprašytų funkcinių ir nefunkcinių reikalavimų įgyvendinimą. Išbaigtumas 60 %.

Be to pateikiama reikalavimų įgyvendinimo lentelė (2-4 lentelė).

2-13 lentelė. Reikalavimų įgyvendinimas

Reikalavimo Nr.	Įgyvendinimo laipsnis	Patikrinimo būdas

2.9 Testavimas

2.9.1 Testavimo scenarijai

Testavimo scenarijų aprašai.

2.9.2 Testavimo atvejai

2.9.3 Automatinis testavimas

Pateikus automatinį testavimą su aprašais, už testavimo skyrių rašomas didžiausias vertinimas, jeigu nėra kitų klaidų.

Reikia pateikti testavimo Print Screen'us.

3 Diegimas – kodo ir veikiančios sistemos demonstravimas

Demonstravimo Print Screen'ai.

4 Ataskaitos rengimas

Pasinaudojus MagicDraw ataskaitų generatoriumi, yra sugeneruojamos reikalingos ataskaitos kiekvienam sprintui. Ataskaitose turi būti pateikti visi aukščiau aprašyti skyriai. Ataskaita yra teikiama dėstytojui gynimui ir kartu rodomas projektas.

Antras ir trečias sprintai yra atliekami analogiškai, tik per trumpesnę laiką, nes laikoma, kad per pirmą sprintą studentams prireikė daugiau laiko dalyko įsisavinimui.

4.1 Išvados

Pateikiamos galutinės išvados.

4.2 Naudoti informacijos šaltiniai

4.3 Priedai