ŠIAULIŲ UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

INFORMATIKOS KATEDRA

Darius Augaitis

Informatikos specialybės II kurso dieninio skyriaus studentas

PROGRAMŲ SINTEZĖ LYGIAGREČIUOJU PROGRAMAVIMO METODU

MAGISTRO DARBAS

Darbo vadovas :

Lekt. V. Giedrimas

Recenzentas :

Lekt. G. Felinskas

Šiauliai, 2007/2008 m.

Žodynėlis

JDL – (*angl. job description language*) užduoties apibūdinimo kalba.

Grid – tai tinklynas. Tinklyną sudaro kompiuteriai, sujungti tinklu. Grid reikalingas užduotims vykdyti [5].

GUI – (*angl. Grafical user interface*) grafinė vartotojo sąsaja.

UI – (*angl. User interface*) vartotojo sąsaja, tekstinė eilutė.

gLite – programinė įranga skirta kurti Grid sistemas.

---> – sudaro failą.

Migrating Desktop – tai grafinė vartotojo, sąsaja skirta saveikauti su Grid tinklynu.

Paveikslų sąrašas

Turinys

I Įvadas

1.1 Temos aktualumas

1.1.1 Naujų programų ir programavimo priemonių poreikis

Tobulėjant technologijoms, auga ir programų poreikis. Suprantama, programos ir užduotys darosi sudėtingesnės, todėl dažniausiai bandoma pritaikyti jau egzistuojančias sistemas. Bet kartais kuriamos naujos programos, kurios sutaupo laiko, žmogiškojo kapitalo ir kt. kaštus, o ne bandoma prisitaikyti prie egzistuojančios sistemos ar jai rašyti paketus ir bandyti pritaikyti šioms programoms. Dažniausiai pritaikomos programos būna lėtos ir turi klaidų, kurių negalima ar neturima galimybių pataisyti.

Naujų programų poreikis skatina kūrimą naujų programavimo įrankių, kurie užtikrintų lengvą pritaikymą prie besikeičiančių sąlygų. Programavimo įrankiai ar metodai pritaikomi dažniausiai pasitaikančių uždavinių automatizavimui. Daugeliu atveju be naujų programavimo įrankių ar net programavimo kalbų darbas būtų komplikuotas.

1.1.2 Grid sistemų plitimas

Grid sistemos skirtos sudėtingiems uždaviniams spręsti. Šios sistemos nėra skirtos paprastiems vartotojams, nes dažniausiai jie nesuformuoja užduočių, kurių nesugebėtų apdoroti vienas kompiuteris. Grid sistemos taikomos mokslinėse institucijose, ten kur prireikia didelių skaičiavimų. Šiose sistemose užduotims aprašyti naudojama JDL kalba, kurios populiarumą mažina prasta dokumentacija, bet tokia sistema turi ateitį, nes jos dėka galima realizuoti sudėtingus uždavinius (automatinis programų kūrimas ar sudėtingi matematiniai skaičiavimai) su žymiai mažesniais laiko kaštais [10].

1.2 Darbo tikslai

Egzistuojančios Grid sistemos dažniausiai neturi patogios vartotojo sąsajos, nuo kurios priklauso darbo greitis ir paprastumas, todėl buvo pasirinkta Grid sistemai grafinės sąsajos kūrimas. Iškelti šie darbo tikslai:

- Pirminis tikslas – sukurti scenarijų ir diagramų kalbą, kuri leistų greitai ir paprastai aprašyti norimas operacijas. Dažnas scenarijų trūkumas yra jų sudėtingumas, todėl keliamas reikalavimas kuriamai scenarijų kalbai – išvengti šio trūkumo.
- Antrinis tikslas – suprojektuoti ir sukurti įrankį (grafinę vartotojo sąsają), kuri leistų greitai ir paprastai kurti įvairaus sudėtingumo scenarijus. Dažniausias aplikacijų trūkumas – vystymasis ir priežiūra, todėl keliamas reikalavimas kuriamai aplikacijai – išvengti šio trūkumo.

1.3 Darbo uždaviniai

Siekiant darbo tikslų, turi būti įgyvendinti šie uždaviniai:

- Išskirti pagrindines Grid sistemos dalis.
- Apžvelgti JDL kalbos struktūrą, išskirti pagrindinius šios sistemos privalumus ir trūkumus.
- Apžvelgti dažniausiai naudojamas Grid sistemos vartotojo sąsajas, išskirti pagrindinius šių sistemų privalumus ir trūkumus.
- Suformuoti reikalavimus kuriamai scenarijų kalbai.
- Sukurti ir aprašyti scenarijų kalbą, tenkinančią kiek įmanoma daugiau iškeltų reikalavimų.
- Suprojektuoti ir realizuoti programą, leidžiančią sukurti scenarijų, tenkinantį kiek įmanoma daugiau iškeltų reikalavimų.
- Suprojektuoti ir realizuoti programą, leidžiančią dirbti neprisirišant prie darbo vietos.

1.4 Mokslinis naujumas

Nors Grid sistemos kuriamos ir vystomos kaip ir vartotojo sąsajos, Grid sistema nėra vientisas produktas. Egzistuojančios Grid sistemos susideda iš kelių programų, kurios atsako už tam tikras operacijas. Viena dalis yra vartotojo sąsaja (*angl. UI – User Interface*).

Šis darbas išsiskiria ir yra naujas savo požiūriu į tokią Grid sistemą, kadangi pagrindinis dėmesys skiriamas vartotojo sąsajai, nepamirštant vientisumo ir automatizuotą programų kūrimą. Kuriama sistema gali būti pritaikyta įvairiose mokslo, darbo srityse [11].

1.5 Praktinė nauda

Bet kuris įrankis ar jų sistema, didinantis programuotojų ir vartotojų darbo našumą ir galimybes, turi praktinės naudos: mažiau reikia programavimo žinių (užtenka pagrindų), mažiau programuotojų, kurie per trumpesnį laiką gali sukurti sudėtingesnes aplikacijas. Atsižvelgiant į tai, kad mokslas nestovi vietoje, reikalavimai ir skaičiavimo sudėtingumas didėja, tokios aplikacijos, kurią siekiama sukurti, nauda yra akivaizdi.

Šiame darbe realizuota aplikacija nebus pilnavertė, todėl jos praktinis pritaikymas mažai tikėtinas. Kita vertus, atliktas tyrimas ir sukurta aplikacija bus naudingi vystant šį įrankį ir kuriant naujus.

II Grid sistemos, vartotojo sąsajos ir JDL kalbos analizė

2.1 Vartotojo sąsajos

Darbui su Grid sistemomis reikalingas patogus įrankis. Dažniausiai kuriami įrankiai daugiau skirti darbo padavimui Grid sistemai ir iš jos rezultatų paėmimui. Kadangi Šiaulių universitete yra naudojama gLite programinė įranga, tai testavimui pasirinktos programos: gLite komandinė eilutė ir Migrating Desktop.

2.1.1 Migrating Desktop

Patogiam darbui daugelis renkasi grafinį įrankį kaip Migrating Desktop. Tai galingas grafinis įrankis nepriklausomas nuo operacinės sistemos, nes parašytas Java kalba. Sukurtas vartotojams su nepastovia darbo vieta. Paleidžiama ant visų OS kuriose yra Java virtualioji mašina (MS Windows, Linux, Solaris) [3]. Jame yra nemažai įrankių palengvinančių darbą su Grid sistemomis (žr. 3 priedas).

Privalumai:

- Platformos nepriklausomumas;
- Prisijungimo paprastumas;
- Darbo aplinkos keitimo galimybė;
- Operacijos su failais Grid sistemoje ir vietiniame kompiuteryje;
- Užduoties vykdymo būsenos stebėjimas;
- Lengvas užduoties padavimas Grid sistemai.

Trūkumai:

- Nėra redaktoriaus skirto kurti, redaguoti bash, java ar python kalba parašytas programas.

2.1.2 gLite komandinė eilutė

gLite programinė įranga yra netik vartotojo sąsaja bet ir pati Grid sistema. Vartotojui leidžiama pasirinkti ar nori tik naudotis Grid sistema, ar patapti jos dalimi, nuo to priklauso programinės įrangos paketai. Pati gLite susideda iš keleto programų ir ji skirta tik linux operacinei sistemai [9]. Pagrindinis trūkumas – reikia mokėti komandas.

gLite vartotojo sąsaja susideda iš dviejų dalių [9]:

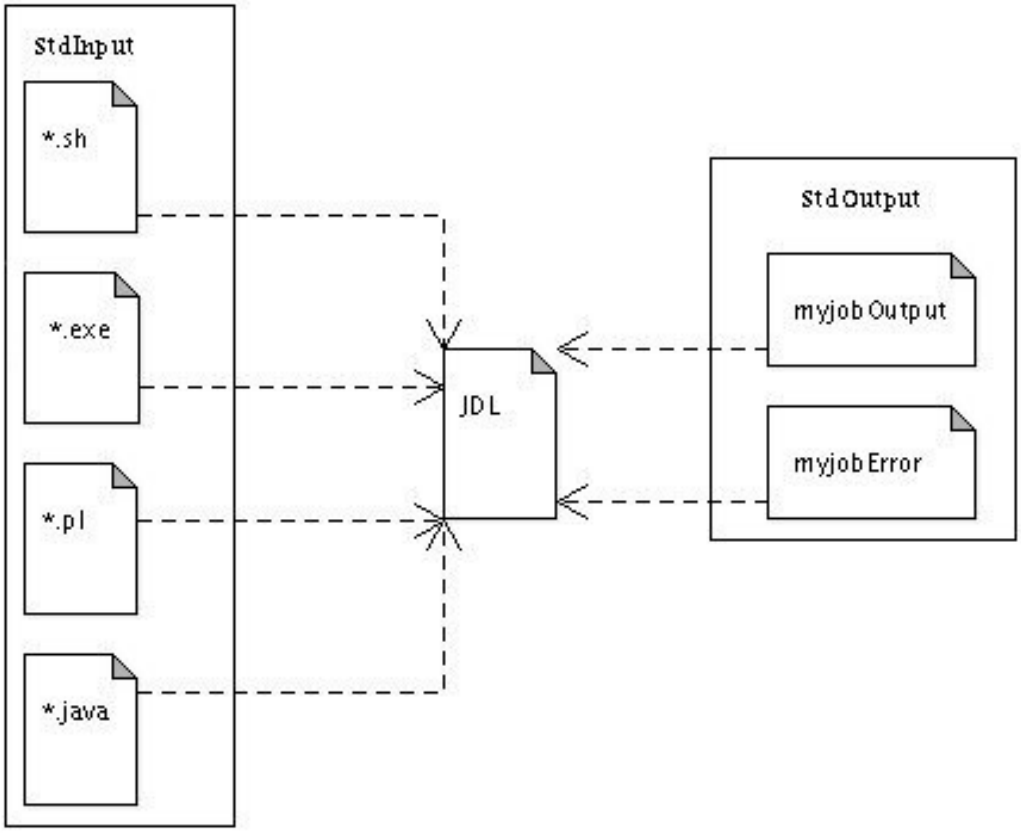
- PPS glite UI – gLite programinė įranga
- PPS glite UI external – papildomos bibliotekos reikalingos gLite veikimui

Su gLite vartotojo sąsaja galima:

- Nusiųsti užduotį;
- Peržiūrėti rezultatus;
- Peržiūrėti tarpinius rezultatus.

2.2 JDL kalbos struktūra

JDL (*Job Description Language*) išvertus iš anglų kalbos – užduoties apibūdinimo kalba. Šia kalba aprašoma: ką turi įvykdyti Grid sistema, kur talpinti rezultatą, kur įvykus klaidai patalpinti pranešimą apie nesklandumus ar kokio tipo darbą reikia atlikti [1].



2.2 pav. Užduoties struktūra

2.2.1 JDL kalbos ir darbo tipai

Tipo (Type) pagalba nurodome, koks bus darbo užklausos tipas JDL faile. Jeigu nenurodomas tipas, tokiu atveju, pagal nutylėjimą, pati Grid sistema priskiria „Job“. JDL faile užrašoma taip [1]:

Type=“Job“;

Galimos reikšmės:

- Job
- DAG

Darbo tipo (JobType) reikšmė nurodo, koks bus atliekamas užduoties tipas. JDL faile užrašoma:

JobType=“Interactive“;

Galimos reikšmės [1]:

- Normal – (yra naudojamas pagal nutylėjimą)
- Interactive
- MPICH
- Checkpointable
- Partitionable

Jeigu parenkamas DAG tipas, tuomet galimos reikšmės:

- Normal – (yra naudojamas pagal nutylėjimą)
- MPICH
- Checkpointable

2.2.2 Vykdymo, įvesties ir išvesties komandos

Sekantis žingsnis – nurodyti, kokie bus vykdomi failai ir jiems perduodami parametrai, kokiame faile saugomi rezultatai ar pranešimai apie klaidą. Tam aprašyti naudojamos komandos [1]:

- executable – šia komanda nurodoma, kuris failas bus vykdomas, pvz., exe.
- Attribute – šia komanda perduodami atributai vykdomajam failui.
- StdInput – šia komanda nurodomi, kurių failų kodai bus vykdomi.
- StdOutput – šia komanda nurodoma, kuriame faile bus saugomi rezultatai.
- StdError – šia komanda nurodoma, kuriame faile bus saugomi pranešimai apie klaidą.
- InputSandbox – šia komanda nurodomas sąrašas failų, kurie bus reikalingi įvykdyti užduočiai.
- OutputSandbox – šia komanda nurodomas sąrašas failų, kuriame bus surašyti rezultatai.

Pavyzdys:

```
Type = "job"; // nurodomas JDL kalbos aprašymo tipas
JobType = "Normal"; //nurodomas darbo tipas
VirtualOrganisation = "su.lt"; // organizacijos pavadinimas
NodeNumber = 6; // nurodomas procesorių skaičius užduoties įvykdymui
Executable = "testas.exe"; // nurodomas vykdomasis failas
```

Arguments = “N1 N2 –out testas.out”; // paduodamos reikšmės ir rezultatas išsaugomas faile.

StdOutput = "testas.out"; //rezultato failas

StdError = "testas.err"; // klaidos failas

InputSandbox = {“file:///home/darius/testas.exe”}; //nurodoma failo egzistavimo vieta

OutputSandbox = { "testas.err", "testas.out" }; // nurodomi išvedimo failai

RetryCount = 3; //kiek sykių bandyti išnaujo vykdyti įvykus klaidai

2.3 Grid sistemos pagrindinės dalys

Kadangi Šiaulių universitetas yra virtualios organizacijos Litgrid partneris, tai analizuojama tarpinė programinė įranga:

- LCG;
- GLITE;

Šios tarpinės programos nėra vientisos. Jos susideda iš keleto programų [5], kurios turi tam tikras atsakomybes:

- openMosix, mosix, globus ir t.t. – programinė įranga, garantuojanti darbą telkinyje.
- GridFTP,OGSA-DAI, Metadata catalog service – programinė įranga, garantuojanti duomenų gavimą ir išsiuntimą.
- MyProxy, PKINIT, Globus Sertificate Service – programinė įranga, garantuojanti Grid sistemos saugumą.
- Ganglija, Inca, MonALISA, gridICE – programinė įranga, garantuojanti grid sistemos stebėjimą.
- MPI, PVM, LAM – programinė įranga, skirta skaičiavimams.
- Scientific Linux – operacinė sistema Redhat pagrindu, pritaikyta dideliems skaičiavimams.

III Įrankių ir scenarijų specifikacijos

3.1 Darbo įrankiui keliami reikalavimai

Tam, kad galėtume suprojektuoti ir realizuoti vartotojo sąsają reikalingi įrankiai ir programos. Kad lengviau būtų išsirinkti įrankius, tam yra keliami reikalavimai:

- Nepriklausomas nuo OS;
- Nemokami;
- Saugumas;
- Našumas.

3.1.1 Duomenų bazės

- SQLite reikalavimų atitikimas ir trūkumai:
  - Nemokama, nepriklausoma nuo OS, naši, saugi.
  - Tik vietiniame kompiuteryje.
- PostgreSQL reikalavimų atitikimas ir trūkumai:
  - Nemokama, nepriklausoma nuo OS, saugi, galimybė prisijungti iš nutolusio kompiuterio.
  - Lėta.
- MS SQL reikalavimų atitikimas ir trūkumai:
  - Yra nemokama versija, saugi, galimybė prisijungti iš nutolusio kompiuterio.
  - Tik windows OS.
- IBM DB2 reikalavimų atitikimas ir trūkumai:
  - Saugi, galimybė prisijungti iš nutolusio kompiuterio, nepriklauso nuo OS.
  - Nors yra nemokama versija ji apribota procesorių kiekiu ir ramų dydžiu, reikalauja galingo kompiuterio.
- ORACLE reikalavimų atitikimas ir trūkumai:
  - Saugi, galimybė prisijungti iš nutolusio kompiuterio, nepriklauso nuo OS.
  - Nors yra nemokama versija ji apribota procesorių kiekiu ir ramų dydžiu, reikalauja galingo kompiuterio.
- MySQL reikalavimų atitikimas ir trūkumai:
  - Saugi, galimybė prisijungti iš nutolusio kompiuterio, nepriklauso nuo OS, yra

nemokama versija, greita.

Iš visų išnagrinėtų duomenų bazių sąlygas labiausiai tenkina MySQL. Todėl duomenų saugojimui bus pasirinkta ši duomenų bazė.

3.1.2 Duomenų bazės projektavimo ir kūrimo įrankis

Kadangi pasirinkta duomenų bazė MySQL, tai galima pasirinkti ir įrankius skirtus tik šiai duomenų bazei:

- EMS SQL Manager 2007 for MySQL reikalavimų atitikimas ir trūkumai:
  - Saugi, paprasta ir lengva naudotis, grafinis duomenų bazių projektavimo įrankis.
  - Mokama, tik windows OS.
- phpMyAdmin reikalavimų atitikimas ir trūkumai:
  - Nemokama, paprasta naudotis, nepriklauso nuo OS.
  - Reikia papildomų aplikacijų, nėra galimybių grafiškai projektuoti duomenų bazes.
- MySQL Query Browser reikalavimų atitikimas ir trūkumai:
  - Nemokama, nepriklausoma nuo OS, paprasta naudotis.
  - Nėra galimybių grafiškai projektuoti duomenų bazes.

Kadangi pagrindinis kriterijus yra patogumas ir paprastumas, todėl šis kriterijus nusveria EMS SQL Manager 2007 for MySQL trūkumus, todėl parenkamas kaip įrankis šiai užduočiai atlikti.

3.1.3 UML diagramų braižymo įrankis

- Umbrella reikalavimų atitikimas ir trūkumai:
  - Paprasta naudotis, greita, saugi, nemokama, kodo generavimas iš diagramų ir atvirkščiai.
  - Tik Linux OS.
- argouml reikalavimų atitikimas ir trūkumai:
  - Paprasta naudotis, greita, saugi, nemokama, yra iš klasių degramos vertimas į kodą.
  - Tik Linux OS, nėra iš kodo vertimo į klasių diagramas.
- Enterprise Architect reikalavimų atitikimas ir trūkumai:
  - Paprasta naudotis, nepriklausoma nuo OS, kodo generavimas iš diagramų ir atvirkščiai, saugi.
  - Lėta ir reikalauja daug kompiuterio resursų, mokama.
- Magick Draw reikalavimų atitikimas ir trūkumai:
  - Paprasta naudotis, kodo generavimas iš diagramų ir atvirkščiai, saugi.

- Lėta, dažnai „lūžinėja“, mokama.

Argouml labiausiai tenkina poreikius norint atlikti užduotį. Nors kiti įrankiai galingesni savo galimybių atžvilgiu, jų šiai užduočiai atlikti neprireiks, todėl pasirenkamas paprastesnis įrankis.

3.1.4 Programavimo kalba ir įrankis

Kadangi keliami kriterijai yra aplikacijos greitis, nepriklausomumas nuo OS, todėl pasirinkta C++ kalba ir biblioteka QT kuri puikiai veikia Linux, Windows, MacOS OS.

Įrankiai parinkti C++ kalbai:

- Qdevelop reikalavimų atitikimas ir trūkumai:
  - Greitas, skirtas QT bibliotekai, yra klaidų sekimo galimybė.
  - Prastas automatinis žodžių surinkėjas, retkarčiais „užlūžta“.
- Eclipse reikalavimų atitikimas ir trūkumai:
  - Greita, daug galimybių.
  - Dažnai „užlūžta“, ne visuomet iš pirmo karto pavyksta įdiegti įskiepį, prastas automatinis žodžių surinkėjas.
- Visual Studio reikalavimų atitikimas ir trūkumai:
  - Gera klaidų sekimo galimybė, puikus žodžių pasiūlos ir užbaigimo variklis, lengvas naudojimasis, saugi.
  - Norint integruoti QT bibliotekas reikalingas papildomas mokamas produktas. Lėtas.
- Kdevelop reikalavimų atitikimas ir trūkumai:
  - Paprasta, greita, daug galimybių.
  - Nor QT yra palaikomas dažnai „užlūžta“, prastas automatinis žodžių surinkėjas.

Įrankiai šuo atveju renkami ir windows OS, nes galutiniame variante turi būti galimybė išbandyti windows vartotojams. Todėl pasirinkti įrankiai Qdevelop ir Visual Studio.

3.2 Scenarijui keliami reikalavimai

Nuo 1965 metų yra kuriama ir vystoma vizuali programavimo kalba [12]. Ši kalba kol kas naudojama tik mokslinėse įstaigose, nes sudėtingesnius algoritmus ir matematines formules lengviau užrašyti tekstiniu pavidalu.

Atsižvelgiant į tai, kad siekiama sukurti paprastą naudojimui įrankį taip vadinamą vartotojo sąsaja. Vienas iš reikalavimų tenkinimų būtų galimybė kurti su ja scenarijus, kurie palengvintų patį kūrybos procesą. Tam yra formuojami scenarijui tokie reikalavimai:

- Scenarijai turi būti lengvai skaitomi, prižiūrimi ir taisomi.
- Turi būti galimybė scenarijų užrašyti dviem būdais:
  - tekstinis – tradicinis būdas kai viskas užrašoma programavimo kalba;
  - grafinis – tai grafinis kodo atvaizdavimas;
- Galimybė vartotojui redaguotis ir kurti naujus kalbos atpažinimo šablonus. Šablono paskirtis iš kodo (pvz. C++) nubraižyti grafinį vaizdą.

3.2.1 Kalbos dalys ir taisyklės

Kalboje naudojami penki rūšių elementai:

- Pradžia – nurodoma, nuo kurios vietos scenarijus turi būti vykdomas.
- Pabaiga – nurodoma, kurioje vietoje scenarijaus vykdymas baigiasi.
- Veiksmas – nurodama, kad bus atliekamas kažkoks veiksmas:
  - matematinis – galima nurodyti matematinį veiksmą ar veiksmų seką;
  - kodas – galima nurodyti programinį kodą parašyta kalba, kurią palaiko ta Grid sistema su kuria dirbama;
  - ciklai – galimybė nurodyti ciklus:
    - while;
    - for.
- Rezultatai/reikšmės – nurodomi, kokie rezultatai domina ir bus naudojami sekančiame veiksme. Tai gali būti vienas ar keli kintamieji.
- Tikrinimo – tikrinama sąlyga (if atitikmuo) kur galimi tik du variantai TRUE arba FALSE (išvertus iš anglų kalbos: tiesa arba netiesa).

Pagrindinės scenarijaus užrašymo taisyklės (žr. 5 priedas):

- Scenarijuje privalo sudaryti tik vienas pradžios elementas;
- Scenarijuje gali sudaryti viena ar daugiau pabaigų;

- Į rezultatą ar reikšmę gali būti keli veiksmai;
- Po pradžios elemento seka veiksmas. Pirmas veiksmas gali būti tuščias.
- Po veiksmo seka rezultatai. Pirmas rezultatas ar reikšmės gali būti tušti.
- Griežtai draudžiama, kad po veiksmo sektų veiksmas ar rezultatas po rezultato.
- Į pabaigą veiksmas nurodomas tuščias.

3.3 Vartotojo sąsajai keliami reikalavimai

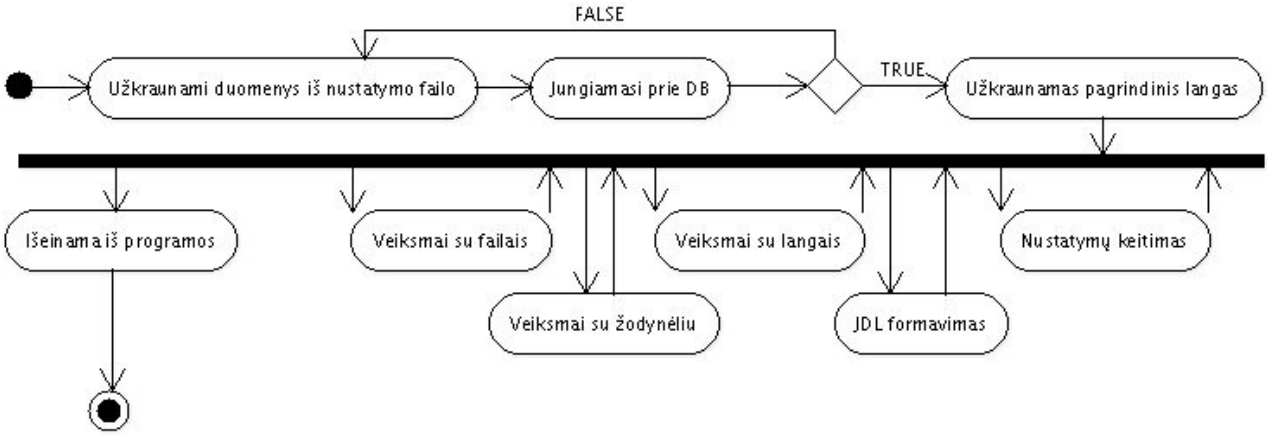
Atsižvelgiant į nagrinėtų vartotojų sąsajų teikiamas galimybes ir trukumus, galima suformuoti reikalavimus, keliamus kuriamajai sąsajai:

- Šia vartoto sąsaja turi būti galimybė lengvai kurti scenarijus.
- Vartotojo sąsaja turi būti nepriklausoma nuo OS. Palaikoma windows, linux, macos.
- Vartotojo nepririšanti prie darbo vietos. Duomenų saugojimas nutolusioje duomenų bazėje su galimybe išsaugoti vietiniame kompiuteryje.
- Saugus prisijungimas prie Grid sistemos.
- Apsaugoti scenarijai nuo pašalinių prieigos.

3.4 Bendras vartotojo sąsajos kūrimo aprašymas

Norima suprojektuoti vartotojo sąsaja, kuri leistų kurti scenarijus. Tam, kad užtikrinti lengvesnį programos kodo rašymą, reikia suprojektuoti ir numatyti, kokias galėtų atlikti operacijas ši programa.

3.4.1 Bendra principinė veikimo schema



3.1 pav. Bendra principinė veikimo schema

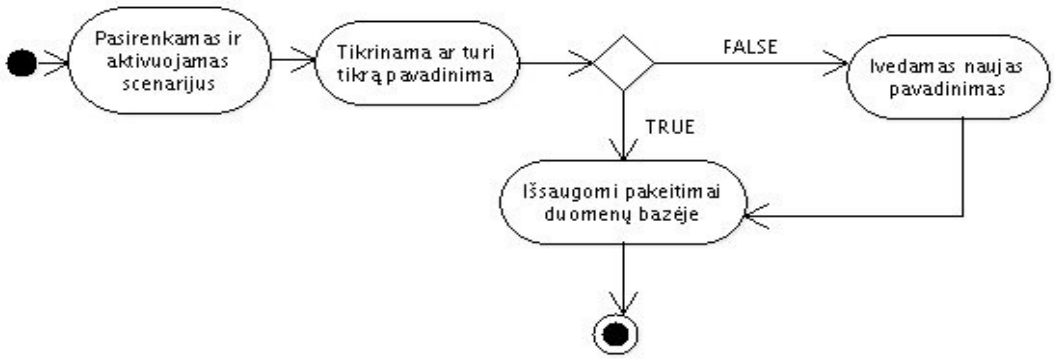
Bendroje veiksmų schemoje pavaizduoti pagrindiniai veiksmai:

- Užkraunami duomenys iš nustatymo failo – užkraunami visi reikalingi parametrai (išskyrus slaptažodį) prisijungti prie duomenų bazės.
- Užkraunamas pagrindinis langas – šis langas yra vartotojo sąsaja. Iš jos atliekami visi pagrindiniai veiksmai, kurie suteikiami tik prisijungus.
- JDL formavimas – vedlio pagalba iš duomenų bazėje esančių failų formuojama užduotis.

3.4.2 Veiksmų su failais principinė schema

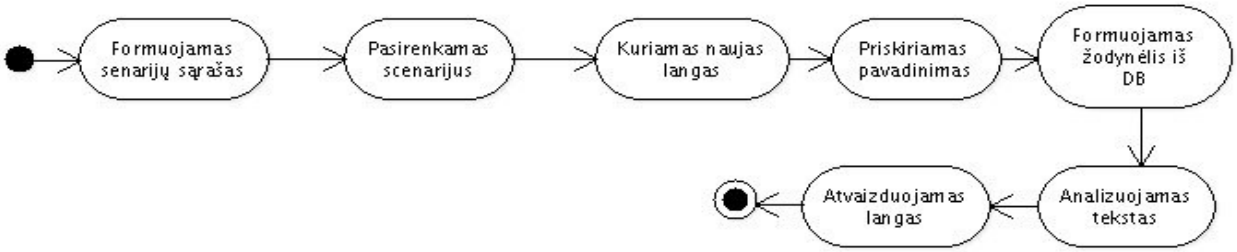
Veiksmų su failais operacijos:

- Išsaugoti – išsaugomas scenarijus duomenų bazėje (žr. 1 priedas, 1.1 pav.).



3.2 pav. Išsaugojimo principinė schema

- Užkrauti – scenarijus užkraunamas iš duomenų bazės (žr. 1 priedas, 1.2 pav.).



3.3 pav. Scenarijų užkrovimo principinė schema

- Ištrinti – pašalinami scenarijai iš duomenų bazės (žr. 1 priedas, 1.3 pav.).
- Importuoti – importuojamas failas į duomenų bazę (žr. 1 priedas, 1.4 pav.).
- Eksportuoti – eksportuojamas iš duomenų bazės į failą (žr. 1 priedas, 1.5 pav.).

3.4.3 Veiksmų su langais principinė schema

Veiksmų su langais operacijos:

- Sumažinti – aktyvi programa sutraukiama į mažiausią galimą dydį.
- Padidinti – padaroma programa maksimalaus dydžio.
- Uždaryti – uždaromas aktyvus langas (žr. 1 priedas, 1.6 pav.).
- Uždaryti visus – uždaromi visi aktyvūs langai (žr. 1 priedas, 1.7 pav.).
- Langų sąrašas – atidarytų failų sąrašas.

3.4.4 Klasių diagrama

Aplikacijos pagrindinės operacijos apgalvotos ir nubraižytos veiksmų diagramos (žr. 1 priedas). Sekantis žingsnis suprojektuoti klases ir funkcijas. Klasės ir jų pagrindinės atsakomybės (žr. 2 priedas):

- mainwindow – ši klasė atsako už pagrindinio lango formavimą;
- Dictionary – ši klasė atsako už žodynėlio formavimą;
- IntroPage – ši klasė atsako už JDL failo formavimo nustatymus;
- ConclusionPage – ši klasė atsako už JDL failo formavimą;
- Settings – ši klasė atsako už nustatymų parametrų keitimą ir nustatymo lango formavimą;

- JDL – ši klasė atsako už JDL failo generavimą;
- Login – ši klasė atsako už prisijungimą prie grid sistemos;
- Save – atsako už scenarijaus išsaugojimą duomenų bazėje ir už išsaugojimo lango formavimą;
- NewName – atsako už naujo pavadinimo sudarymą egzistuojančiam scenarijui;
- LogTo – atsako už prisijungimą prie duomenų bazės;
- SintaxEdit – atsako už:
    - sintaksės lango formavimą;
    - už sintaksės papildymą;
    - už sintaksės šalinimą.
- SmallWindow – atsako už scenarijų redagavimo langų formavimą;
- textBold – atsako už sintaksės paryškinimą;
- Load – atsako už scenarijaus užkrovimą iš duomenų bazės.

3.4.5 Saugomų duomenų struktūra

Duomenų saugojimui pasirinkta duomenų bazė. Kad efektingai išnaudoti duomenų bazę, reikia tinkamai suprojektuoti duomenų bazę (žr. 4 priedas).

IV Scenarijų projektavimo ir vartotojo sąsajos realizacija

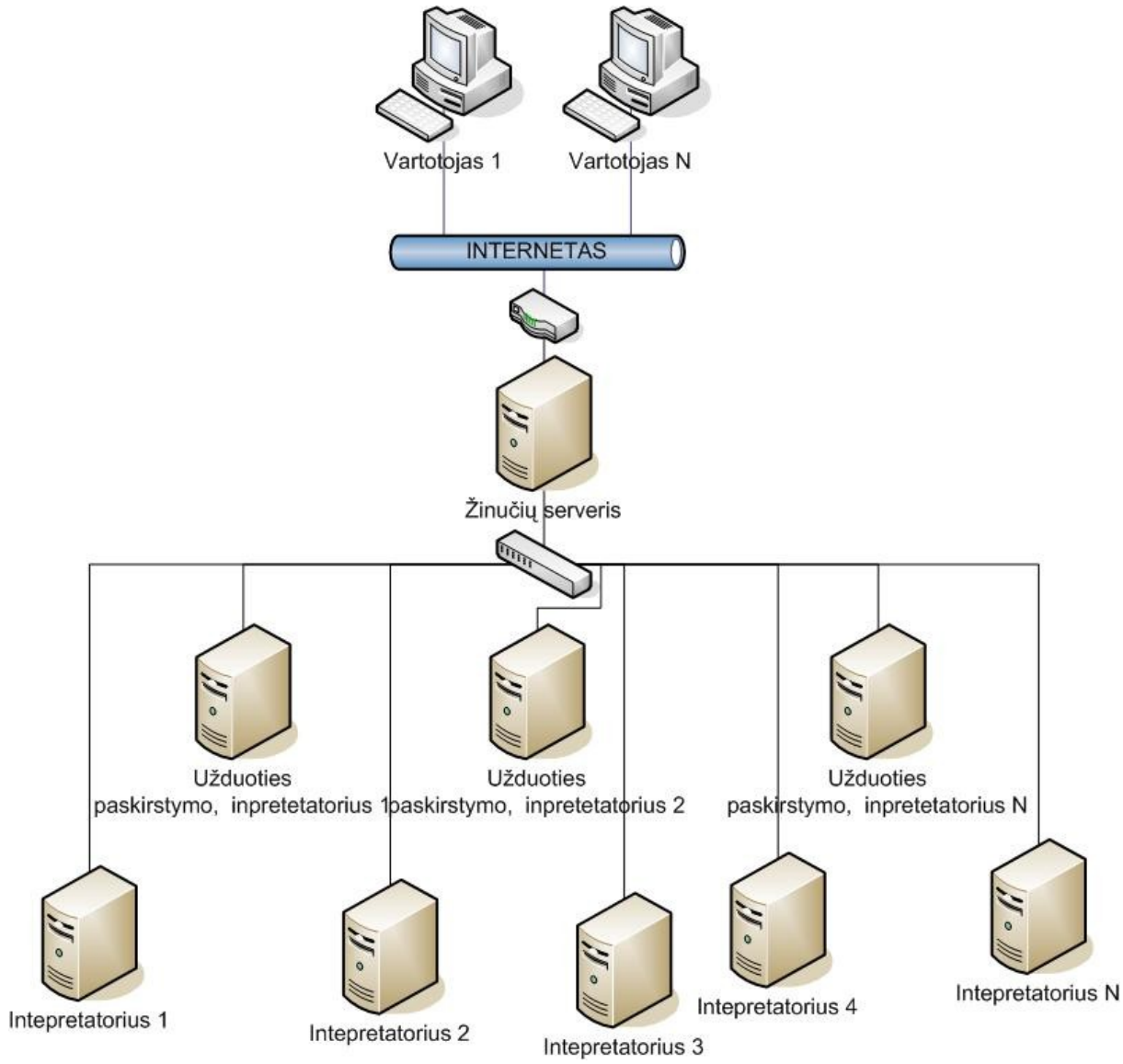
4.1 Naujos Grid sistemos poreikis

Programuoti dažniausiai suprantama, kaip kažkokia kalba užrašomas kodas, kurios dažniausias rezultatas veikianti aplikacija. Bet programavimą galima įsivaizduoti kaip ir kažkokių veiksmų ar įvykių (pvz., mašinos variklio darbą ar vėjo pūtimą) seką užrašytą kompiuteryje. Projektuojant grafinę scenarijų kalbą buvo pastebėta, kad nepavyks realizuoti šios scenarijų kalbos ant gLite Grid sistemos. Todėl buvo lygiagrečiai pradėta mąstyti nauja Grid sistema, kuri išspręstų iškilusias problemas. Žinoma dauguma problemų būtų galima išspręsti, bet tai užimtų žymiai daugiau laiko ir tektų įgyti daug žinių tokiai realizacijai. Tai pat įvertinus tam tikras problemas, aprašytas 4.3.2 skyriuje. Tai nereiškia, kad negalima pritaikyti kitoms Grid sistemos. Tai tiesiog atimtų perdaug taip brangaus laiko.

4.2 Naujos Grid sistemos projektas

Suprojektuota nauja Grid sistema, kuri susideda iš tokių aplikacijų 4.1 pav.:

- Vartotojo sąsaja – tai įrankis, per kurį:
  - siunčiamos užduotys serveriui ir gaunami iš jo rezultatai;
  - kuriami redaguojami scenarijai;
- Žinučių serveris – tai aplikacija, kuri atsako:
  - už saugumą;
  - už žinutės nusiuntimą atitinkamui mazgui;
  - už būsenos stebėjimą mazguose;
  - už tinkamo serverio parinkimą užduočiai įvykdyti;
- Užduoties paskirstymo serveris – atsako už:
  - žinutės analizę;
  - užduoties analizę;
  - interpretatoriaus dalyje naujo scenarijaus vykdymo variklio sukūrimą;
- Interpretatorius – atsako už jame besisukančius lygiagrečiai scenarijų vykdymo variklius;



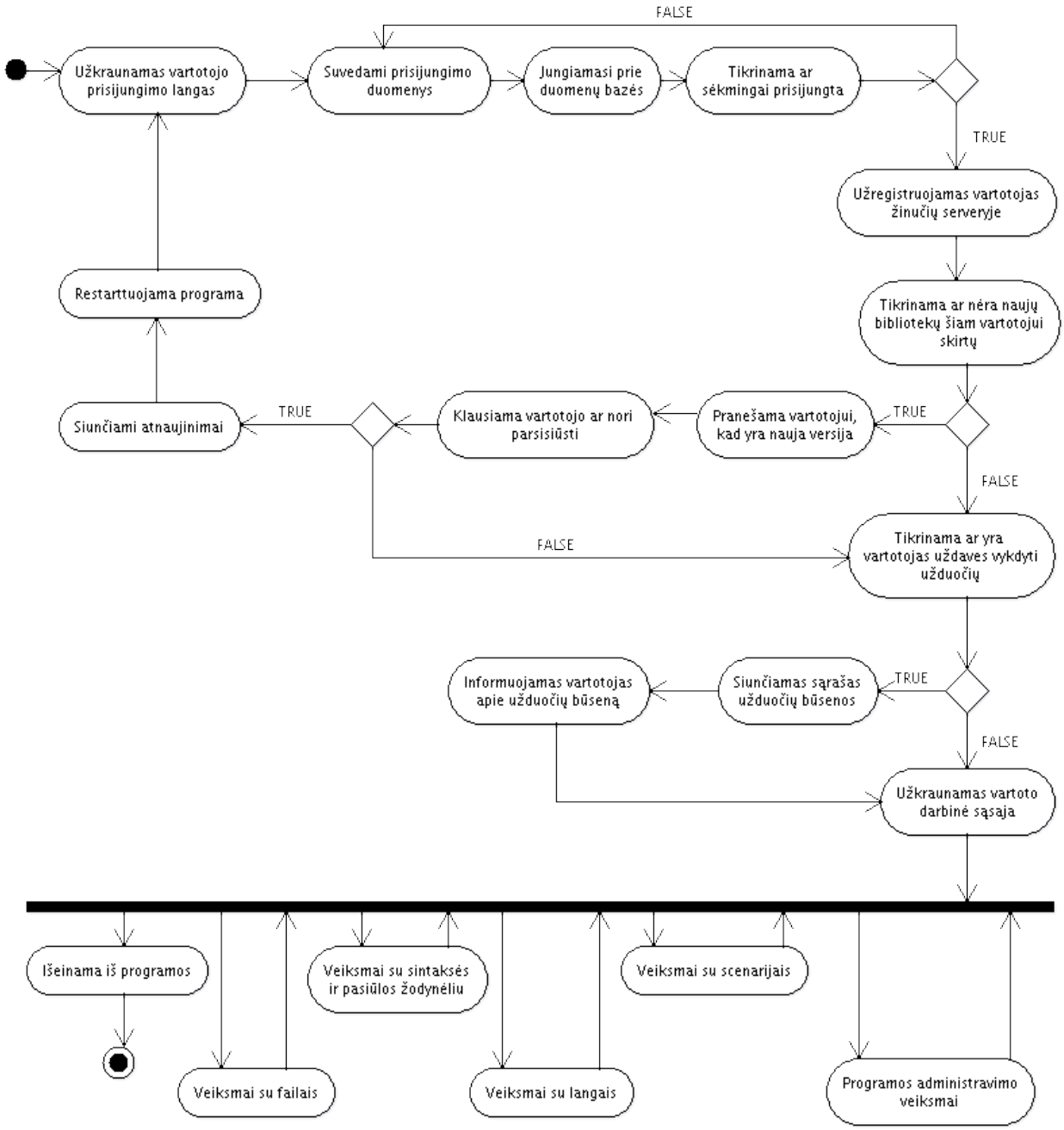
4.1 pav. Bendros Grid sistemos loginės dalys

Kiekviena dalis elgiasi skirtingai išskyrus užduoties paskirstymo serveris. Jis gali veikti dvejopai: kaip užduoties paskirstymo serveris ir tuo pačiu metu atlikti interpretatoriaus vaidmenį.

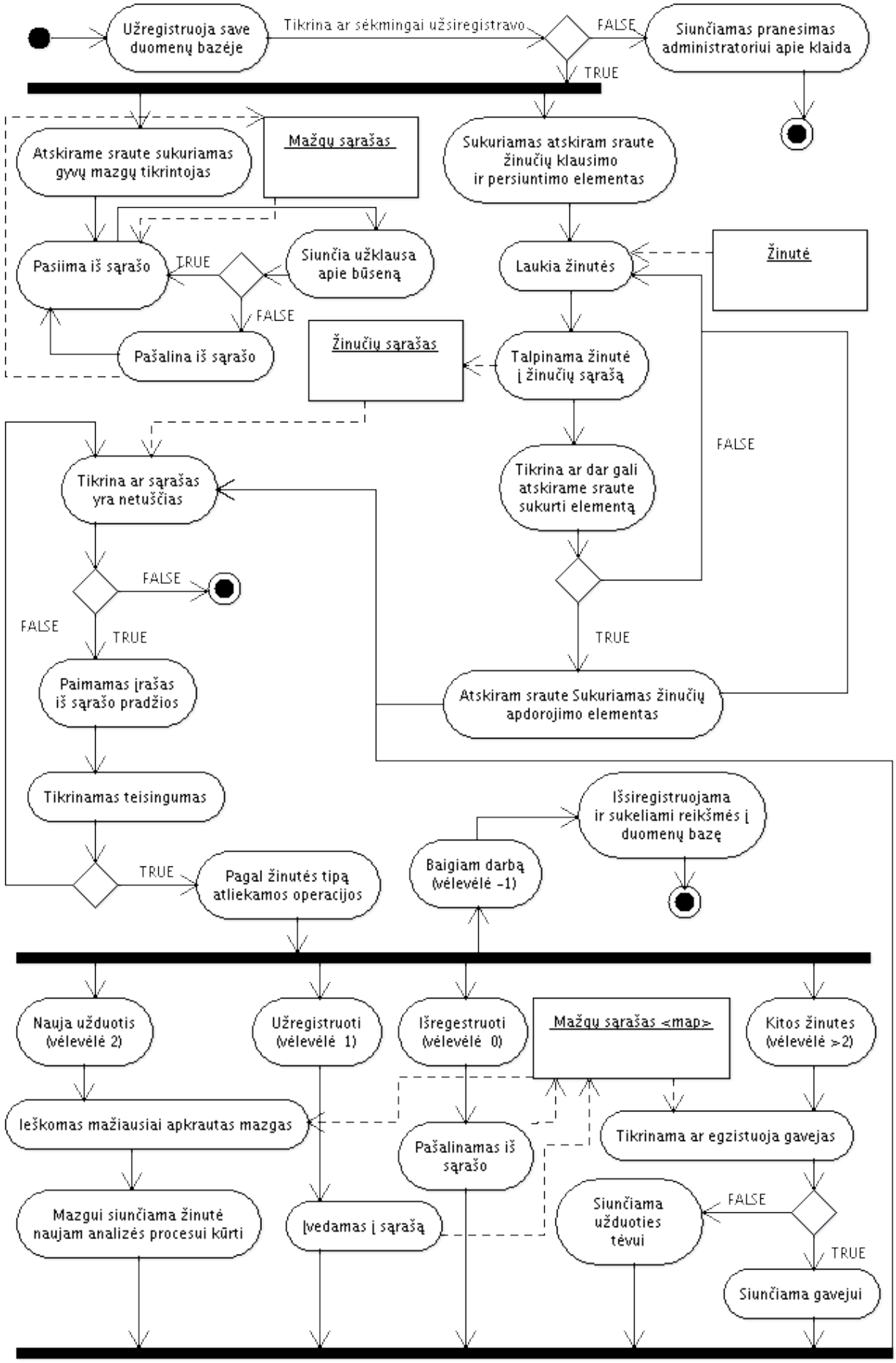
Suprojektuotos kiekvienos dalies veiksmų sekos:

- Vartotojo sąsajos schema (žr. 4.2 schemą) – pavaizduota vartotojo sąsajos veikimo schema.
- Žinučių serverio schema (žr. 4.3 schemą) – pavaizduota žinučių serverio veikimo schema.
- Užduoties paskirstymo serverio schema (žr. 4.4 schemą) – pavaizduota užduoties paskirstymo schema.

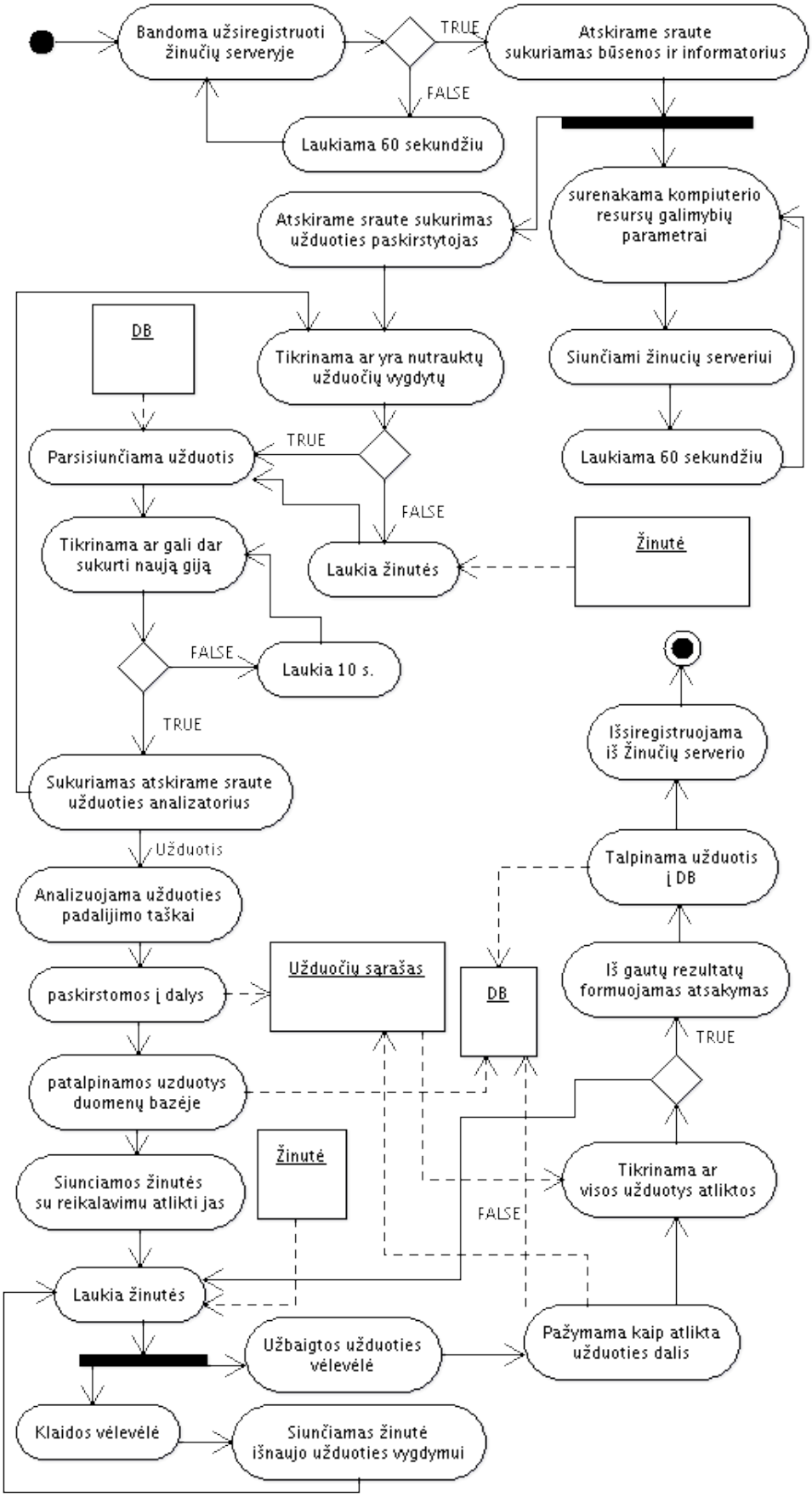
- Interpretatoriaus schema (žr. 4.5 schemą) – pavaizduota užduoties vykdymo schema.



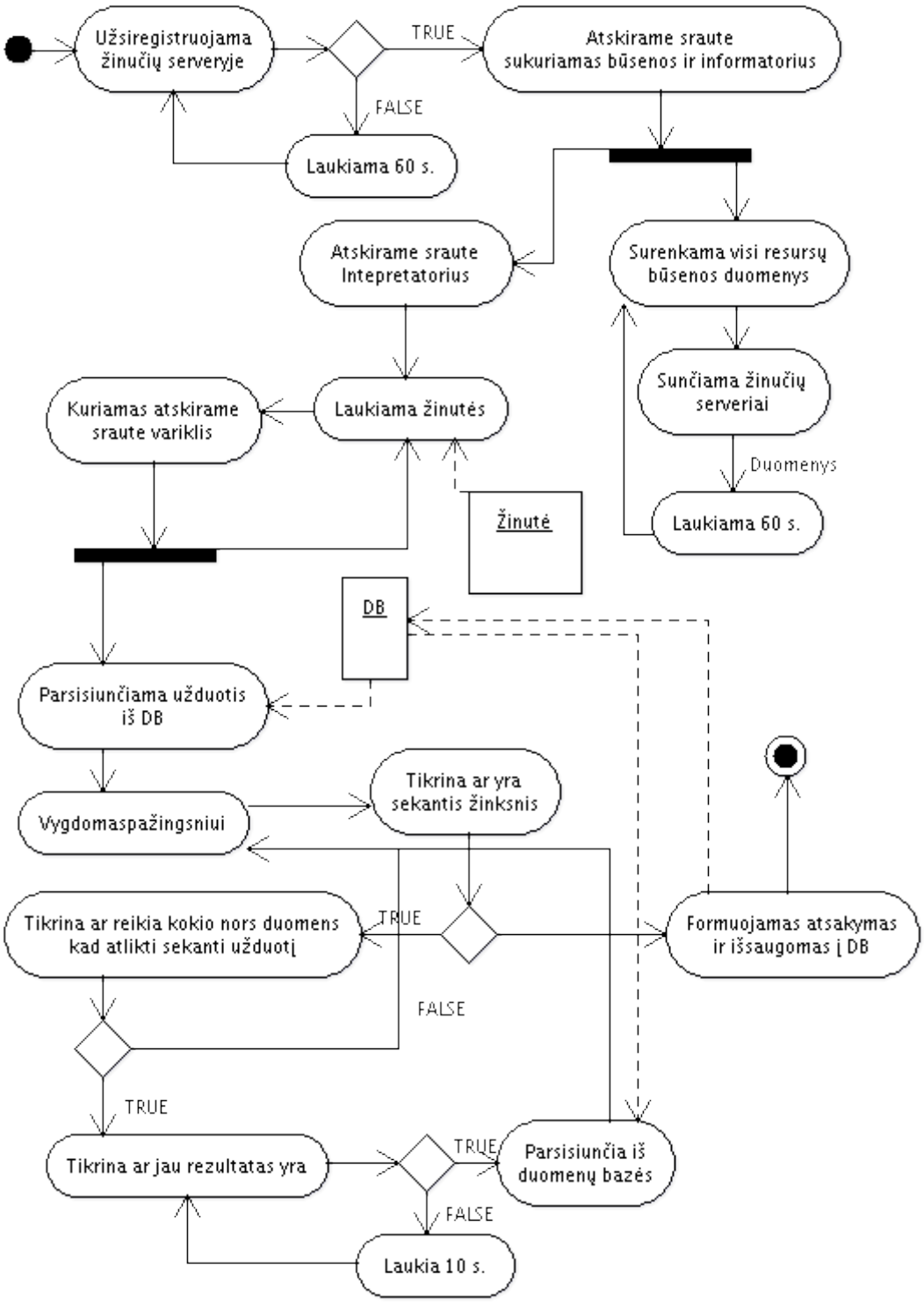
4.2 pav. Bendra vartotojo sąsajos principinė veikimo schema



4.3 pav. Bendra žinučių serverio principinė veikimo schema



4.4 pav. Užduoties paskirstymo serverio principinė veikimo schema



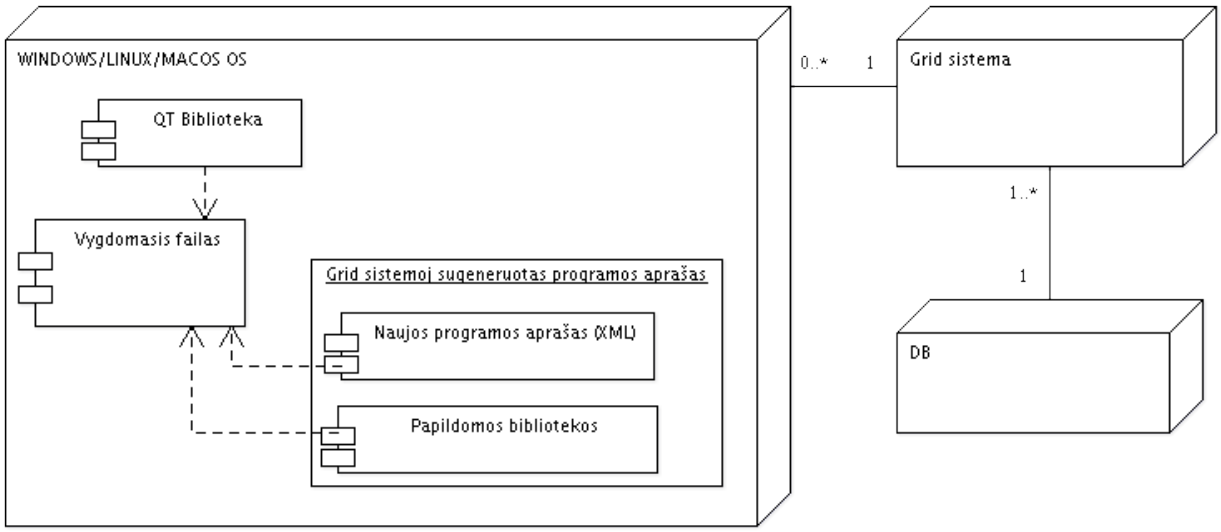
4.5 pav. Intepretatoriaus principinė veikimo schema

4.2.1 Programų sintezė

Vienas iš programavimo palengvinimų yra automatinis programų kūrimas, kai iš anksčiau aprašytų kodų ar jų dalių generuojamos naujos programos.

Tam kad automatiškai sugeneruotos programos būtų nepriklausomos nuo OS, reikia nepriklausomos bibliotekos kaip QT, kuri garantuotų programos veikimą windows, linux, macos OS. 3.4 schemoje matoma, kad windows ar linux, ar macos OS turi būti suinstaliuota QT biblioteka. Grid sistema sugeneruoja kodą XML kalba ir, jei reikia, paruošia reikalingas bibliotekas.

Vykdomieji failai yra skirtingi ir priklauso nuo OS, kuriai generuojama programa. Vartotojui jie yra pateikiami ir yra nekintantys (nebent išleidžiama nauja versija). Šitas failas naudojasi QT biblioteka ir vykdo Grid sistemoje sugeneruotą programos aprašą. Aprašas yra generuojamas vienodas visoms OS sistemoms, skiriasi tik bibliotekos, kurios jau būna paruoštos, o vartotojas jas gauna kartu su Grid sistemos sugeneruotu programos aprašu.



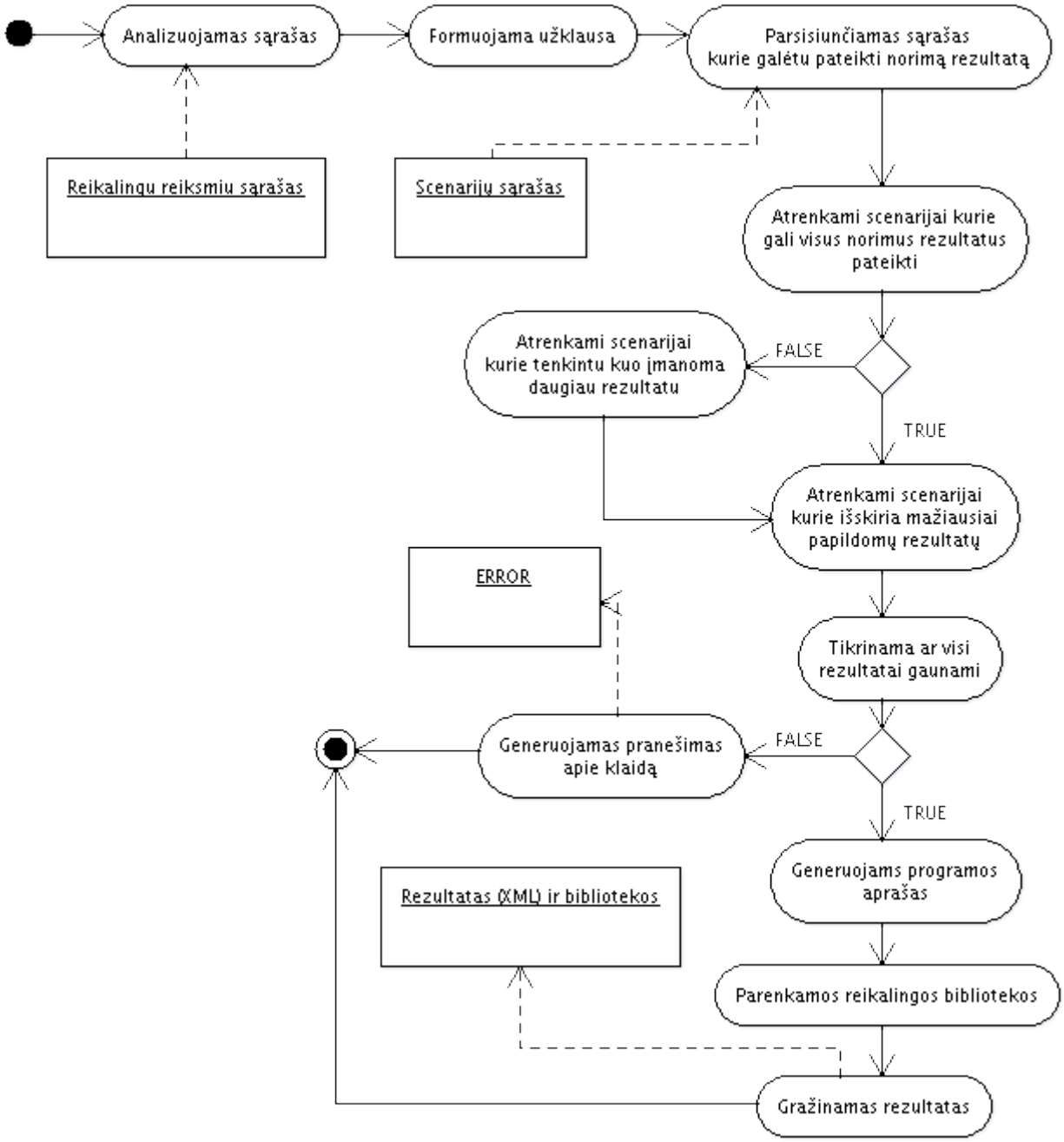
4.6 pav. Sugeneruotos programos aprašo vykdymas

4.2.2 Programų generavimo algoritmas

Norint, kad programa kurtų programą, reikia sukurti algoritmą ir taisykles, kuriomis vadovaujantis būtų generuojami programų aprašai. Algoritmas privalo:

- Išvengti užsiciklinimo;
- Išvengti aklaviečių.

Sugeneruojamas programos aprašas XML kalba. Programos varianto parinkimas gali priklausyti nuo daug kriterijų, kuriuos turėtų nurodyti vartotojas. Jeigu nenurodomi kriterijai, tokiu atveju generuojama programa atsitiktinai (3.5 pav.).



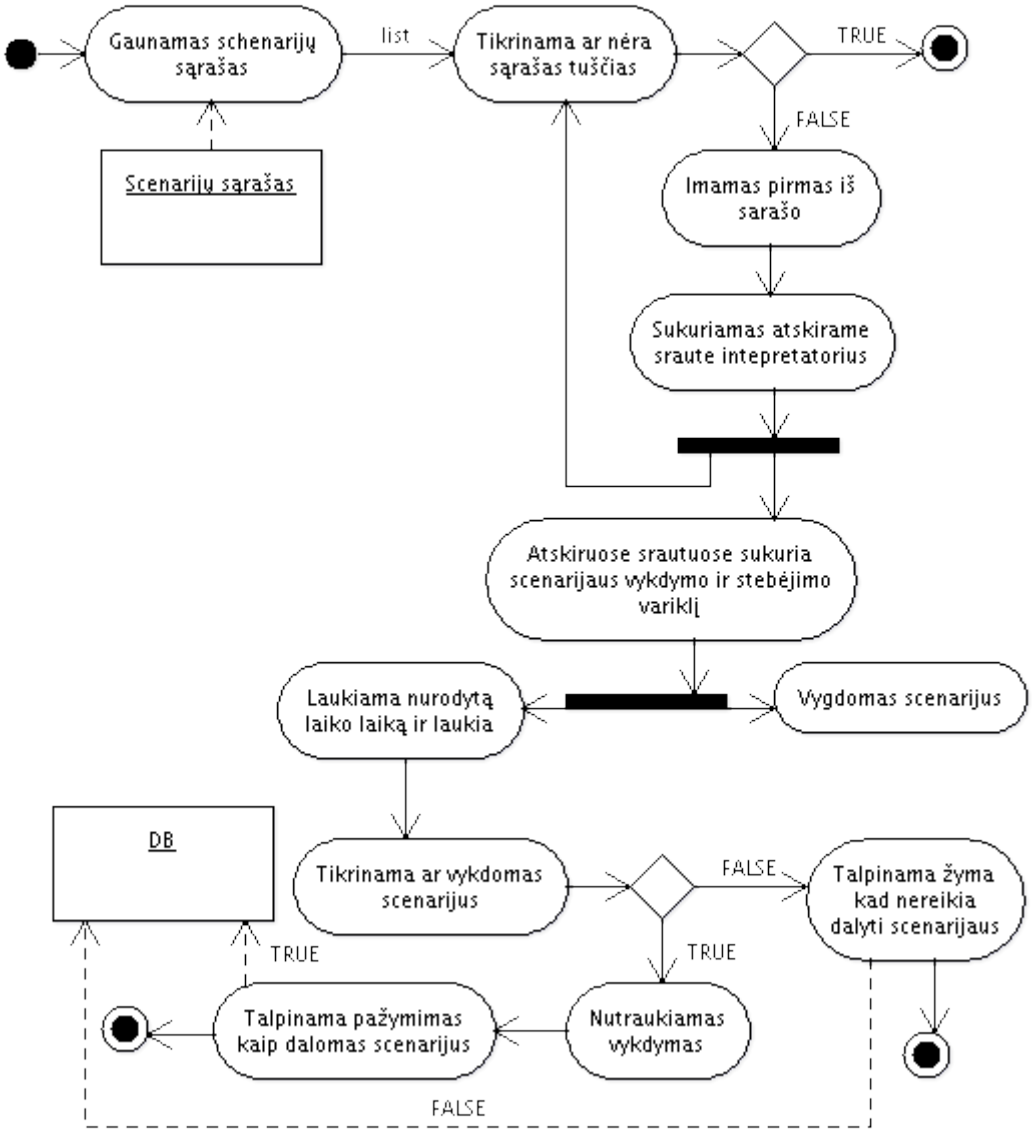
4.7 pav. Generuojamas programos aprašymas

4.2.3 Užduoties dalijimo algoritmas

Viena iš sudėtingiausių dalių – tai tinkamai padalyti užduotį. Užduoties padalijamas gali

būti:

- Vartotojas pats nusprendžia, kuriose ir kurie scenarijai bus dalijami;
- Kompiuteris atlieka dalijimą be analizės. Kiekvienas scenarijus įvykdomas atskirai;
- Kompiuteris atlieka scenarijų analizę ir pagal kriterijus atlieka dalijimą (3.6 pav.).



4.8 pav. Užduoties dalijimo algoritmas

4.3 Scenarijų ir vartotojo sąsajos būsena

Suprojektuotas ir sukurtas produktas. Suprantama jis nėra pilnavertis. Tekstinius

scenarijus galima pilnai užrašinėti. Grafiniai scenarijai suprojektuoti ir paruošti realizacijai. Jų šiuo metu negalima braižyti su vartotojo sąsaja, nes nebuvo tikslas realizuoti, o tik išanalizuoti tokią galimybę. Žinoma padėti pagrindai, kuriais galbūt būtų galima pasinaudoti ir išvystyti į pilnavertį produktą.

4.3.1 Dabartinė būsena

Sukurta grafinė scenarijų kalba pilnai tenkina užsibrėžtus tikslus. Nusakoma kalbos struktūra ir taisyklės. Projektinė dalis pilnai baigta.

Sukurta vartotojo sąsaja su scenarijų rašymo įrankiu. Tekstinis redaktorius galutinai yra užbaigtas. Yra nemažai įgyvendinta palengvinimų skirtų vartotojui. Realizuoti užsibrėžti tikslai: nuotolinis duomenų saugojimas su galimybe išsaugoti ir vietiniame kompiuteryje, veiksmai su žodynėliu ir t.t.

Programavimo modelis pagal užsibrėžtus tikslus yra pilnai įgyvendintas. Pagal projektą realizuotos programuotojui darbą palengvinančios funkcijos, kaip automatinis žodžių užbaigėjas, teksto sintaksės išskyrimas – tai yra pirminis šio įrankio tikslas. Sekantis etapas turėtų būti grafinis scenarijų rašymo įrankis, kuris kol kas yra tik projektiniame lygyje. Pagal užsibrėžtus tikslus jį reikėjo tik suprojektuoti ir išanalizuoti jo realizacijos galimybes ant Grid sistemos. Jam yra suformuoti reikalavimai ir taisyklės.

Atsižvelgiant į darbo metu iškilusias problemas, parengtos rekomendacijos ir kai kurių problemų sprendimai.

4.3.2 Rekomendacijos, pastebėjimai, problemos ir jų sprendimai

Nepavyko iš sukurtos aplikacijos prisijungti prie Šiaulių universiteto Grid sistemos. Aplikacija nebuvo prileidžiama prie sistemos. Buvo naudojamas sertifikatas ir ssl saugumo protokolas. Problemos nepavyko išspręsti.

Projektuojant grafinę scenarijų kalbą paaiškėjo, kad būtų lengviau sukurti naują Grid sistemą negu pritaikyti esamai sistemai. Analizės metu paaiškėjo, kad ne visas idėjas pavyksta realizuoti ant pasirinktos sistemos, todėl buvo lygiagrečiai suprojektuota nauja grid sistema.

Projektuojant Grid sistemą, susidurta su užduoties padalijimo problema. Užduoties paskirstymo taisykles turės nurodyti pats vartotojas, kuriantis scenarijus, nes priešingu atveju dažniausiai bus sugaištama ne užduoties vykdyme, o jos siuntinėjimui ir dalijimui.

Dinaminių aplikacijų kūrimas. Nėra galimybės aprašyti užduoties taip, kad jos rezultatas būtų aplikacija (pvz. exe vykdomasis failas) ir bet kokiai OS paruoštų tą pačią aplikaciją. Vienas iš sprendimų naujos Grid sistemos kūrimas.

Nerastas būdas kaip pastoviai vykdyti procesą, o prireikus iš kito proceso prisijungti ir jį įtakoti. Šiai problemai išspręsti pasirinkta naujos Grid sistemos kūrimas.

Rekomenduoju, vystant ir kuriant panašias programas, naudoti QT biblioteką, kuri palengvins programavimą ir procesų valdymą.

4.3.3 Tolimesni žingsniai

Tolimesni žingsniai vartotojo sąsajos su programavimo galimybėmis ir scenarijaus projektavimu būtų:

Sukurti grafinę scenarijų kūrimo aplinką, kuri leistų:

- lengvai redaguoti scenarijus;
- kurti naujus.

Tolimesni žingsniai priklauso nuo poreikių: pirmiausiai būtų baigiama suprojektuoti nauja Grid sistema ir jai vartotojo sąsaja. Naujos Grid sistemos poreikis kyla iš atlikto tyrimo, nes, norint pritaikyti egzistuojančiai sistemai, reikia mažiausiai įdėti dvigubai pastangų, ir, kaip matoma iš aukščiau aprašytų pastebėjimų, ne visas idėjas galima realizuoti.

V Išvados

- Suprojektuota grafinė scenarijų kalba, aprašytos užrašymo taisyklės, kalbos dalys, padarytas grafinis scenarijaus pavyzdys. Darbo metu paaiškėjo, kad grafinis scenarijų užrašymas nėra būtinas, tiesiog kartais jis palengvintų scenarijų kūrimą ir analizę.
- Sukurta vartotojo sąsaja su scenarijų užrašymo įrankiu. Žinoma ji nėra pilnavertė ir vis dar yra vystymo stadijoje. Galima užrašyti tik tekstinius scenarijus. Tekstinių scenarijų pakanka realizuoti užduotį Grid sistemai. Sekančiame etape būtų galima realizuoti grafinį scenarijų užrašymo įrankį.
- Suprojektuotos pagrindinės naujos Grid sistemos dalys. Kad ją pilnai suprojektuoti prireiktų kokių 3 metų, o programavimo ir testavimo darbams dar 1 metų. Tokie terminai parinkti, nes reikia nemažai padaryti rimtų sprendimų, kurie leistų pilnavertiškai dirbti. Plačiau analizuotos Grid funkcijos yra automatinis programų kūrimas ir užduoties padalijimas.

VI Literatūros ir informacinių šaltinių sąrašai

[1] JOB DESCRIPTION LANGUAGE ATTRIBUTES SPECIFICATION. Prieiga per internetą: https://edms.cern.ch/document/555796/1 [žiūrėta 2008 m. balandžio 25 d.]

[2] LCG Application Area Software. Prieiga per internetą: https://twiki.cern.ch/twiki/bin/view/LCG/LCGAASoftware [žiūrėta 2008 m. balandžio 25 d.]

[3] Migrating Desktop. Prieiga per internetą: http://desktop.psnc.pl/docs/BG-MDTutorial-v1.3-PSNC.doc [žiūrėta 2008 m. balandžio 25 d.]

[4] Migration Desktop installation guide. Prieaiga per internetą:http://ras.man.poznan.pl/cgi-bin/viewcvs.cgi/desktop/resources/doc/MD_RAS_Installation_Guide.doc?view=log [žiūrėta 2008 m. balandžio 25 d.]

[5] Albinas Vladyka,Grid Projektavimas, Šiauliai 2007 m.

[6] JDL ATTRIBUTES. Prieiga per internetą: http://server11.infn.it/workload-Grid/docs/DataGrid-01-TEN-0142-0_2.pdf [žiūrėta 2008 m. balandžio 25 d.]

[7] Computer cluster. Prieiga per internetą: http://en.wikipedia.org/wiki/Computer_cluster [žiūrėta 2008 m. balandžio 25 d.]

[8] PVM Prieiga per internetą: http://209.85.135.104/search?q=cache:IAGcOUWo7YkJ:www.netlib.org/pvm3/pvm-eet.ps+pvm+math&hl=lt&ct=clnk&cd=1&gl=lt [žiūrėta 2008 m. balandžio 25 d.]

[9] gLite 3.1 UI tarball distribution Prieiga per internetą: https://twiki.cern.ch/twiki/bin/view/LCG/UiTarInstall [žiūrėta 2008 m. balandžio 29 d.]

[10] K. Czajkowski C. Kesselman, Grid Information Services for Distributed Resource Sharing, University of Southern California, 2001 m.

[11] R. Buyya D. Abramson J. Giddy, An Architecture for a Resource Management and Scheduling System in a Global Computational Grid, 2000 m.

[12] A. Survey M. Boshernitsan M. Downes, Visual Programming Languages, 1997, University of California, Berrkeley

VII Anotacija

Programų sintezė lygiagrečiuoju programavimo metodu. Šio darbo tikslai: sukurti, išanalizuoti ir suprojektuoti grafinę scenarijų kalbą, kuri leistų vartotojui nesunkiai piešimo būdu kurti sudėtingas programas, skirtas Grid sistemoms, suprojektuoti ir sukurti vartotojo sąsaja su programavimo galimybėmis, išanalizuoti tiriamos Grid sistemos trūkumus ir pasiūlyti alternatyvų sprendimą.

Summary

Synthesis of programs for parallel programming method. Goals of this work are: to analyze, make project of visual scenario language, which will provide ability for user easily create complicated programs for Grid system with drawing method, to analyze and make project of graphical user interface for the Grid system with ability to write program source, to analyze the Grid systems limitations and suggest alternative solution.

VIII Priedai

1 priedas

Vartotojo sąsajos veiksmų diagramos

2 priedas

Vartotojo sąsajos klasių diagrama

3 priedas

Migrating Desktop programos vaizdas

4 priedas

Duomenų bazės lentelės

5 priedas

Grafinis scenarijų pavyzdys

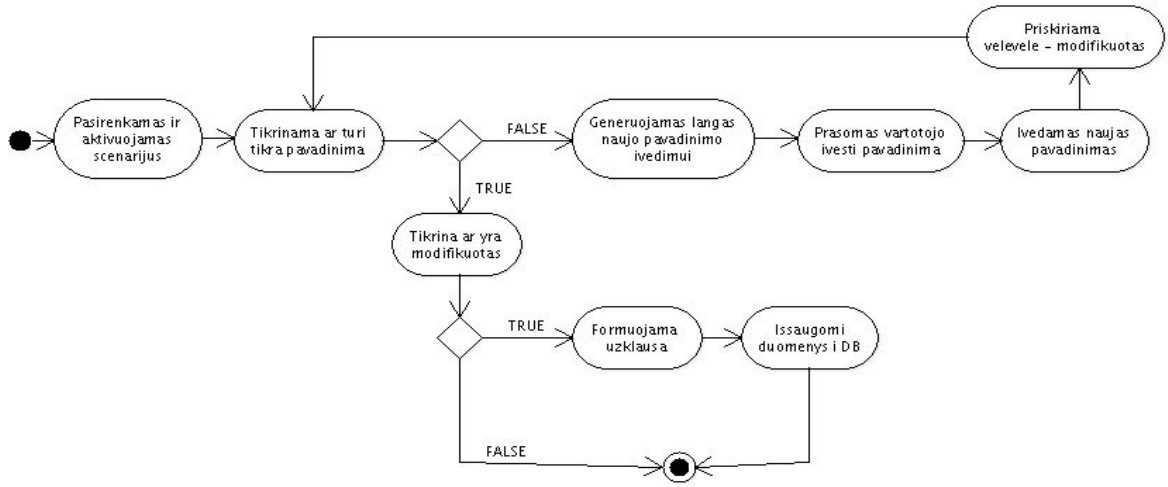
6 priedas. CD turinys

1. darbo_aprašymas.pdf – darbo aprašymas;
2. yrankis – programos išeities tekstai;
3. yrankis_l (linux OS), yrankis_w (windows OS) – sukompiliuota ir paruošta programa;
4. sqlquery.sql – sql užklausos skirtos sukurti duomenų bazę;
5. darbo_aprašymas.doc– darbo aprašymas;
6. darbo_aprašymas.odt – darbo aprašymas;
7. Install – papildomos programos.

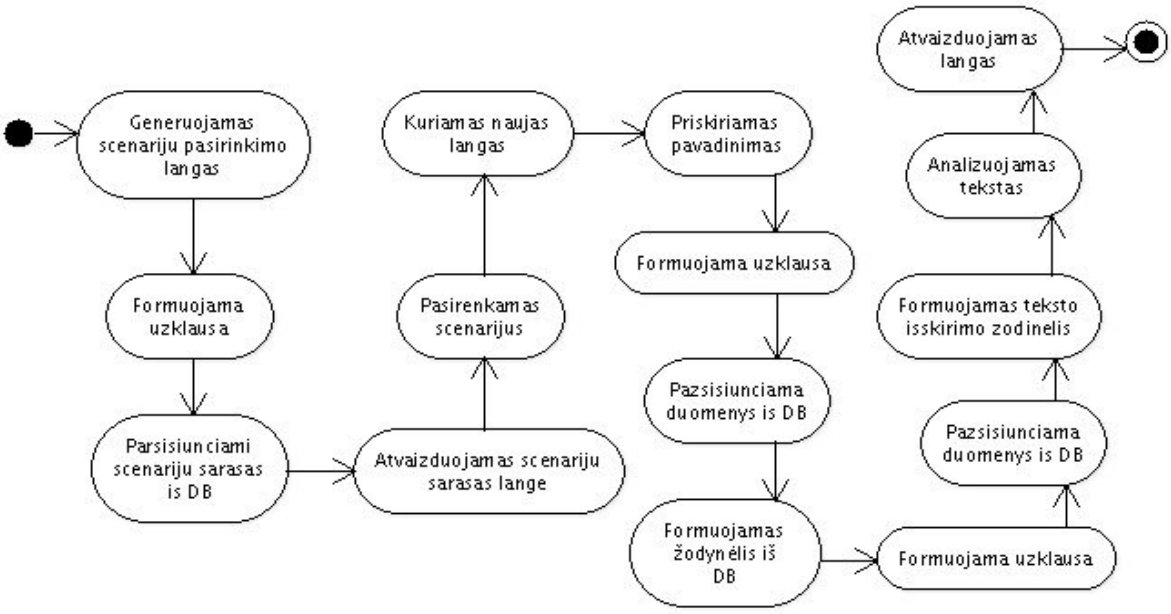
1 Priedas

Veiksmų diagramos

1.

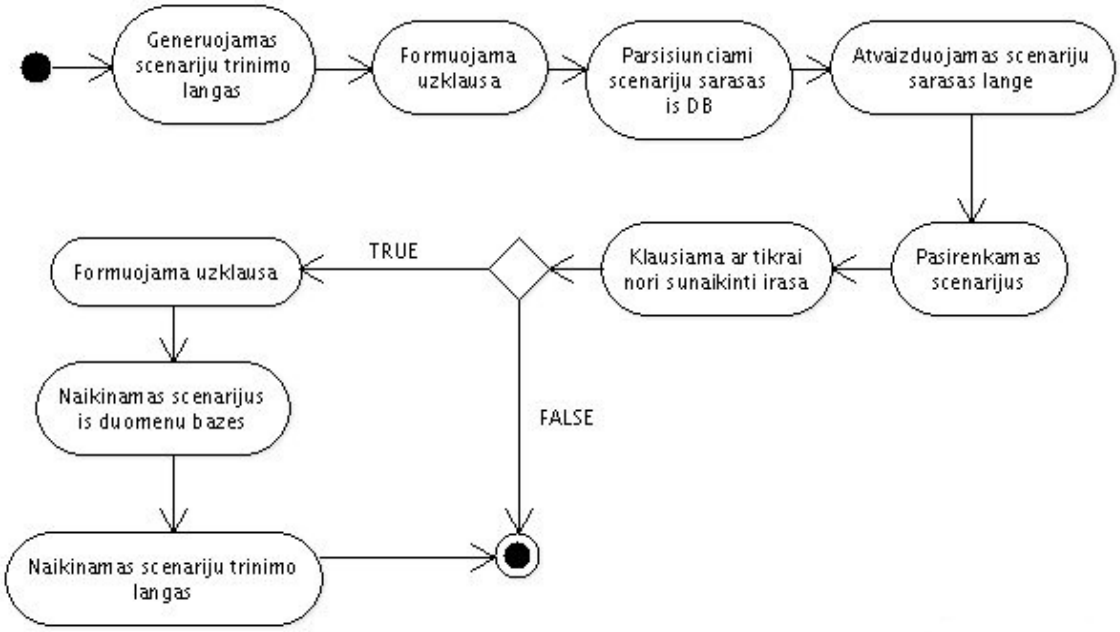


1 pav. Saugoti - veiksmo diagrama

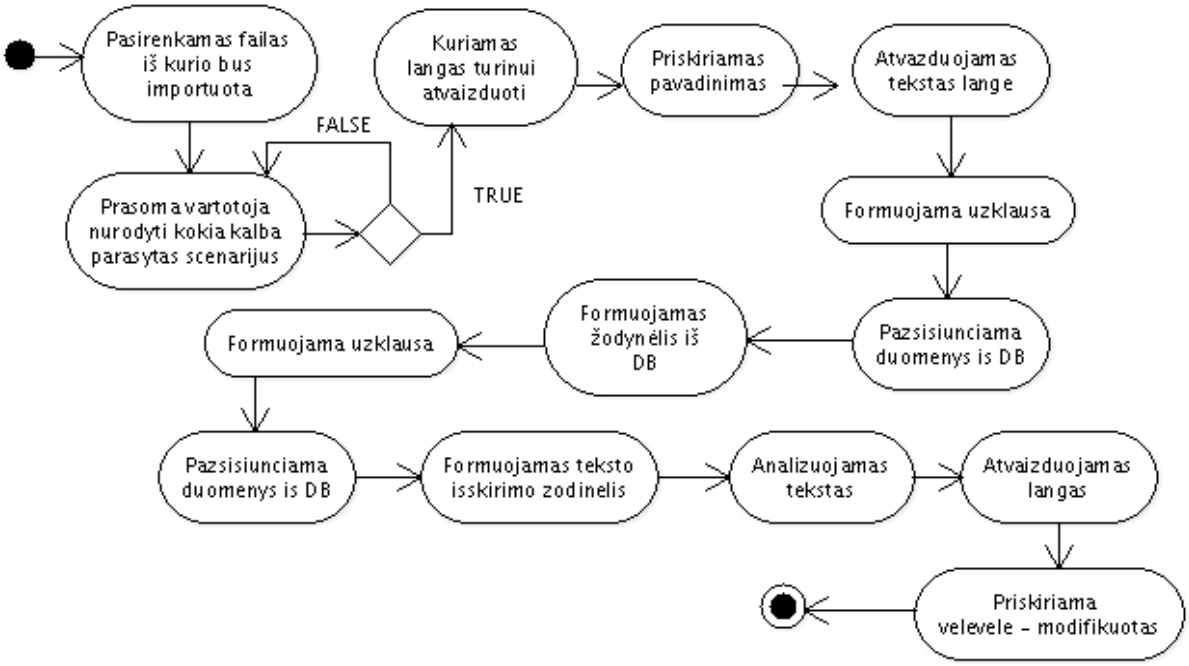


1.

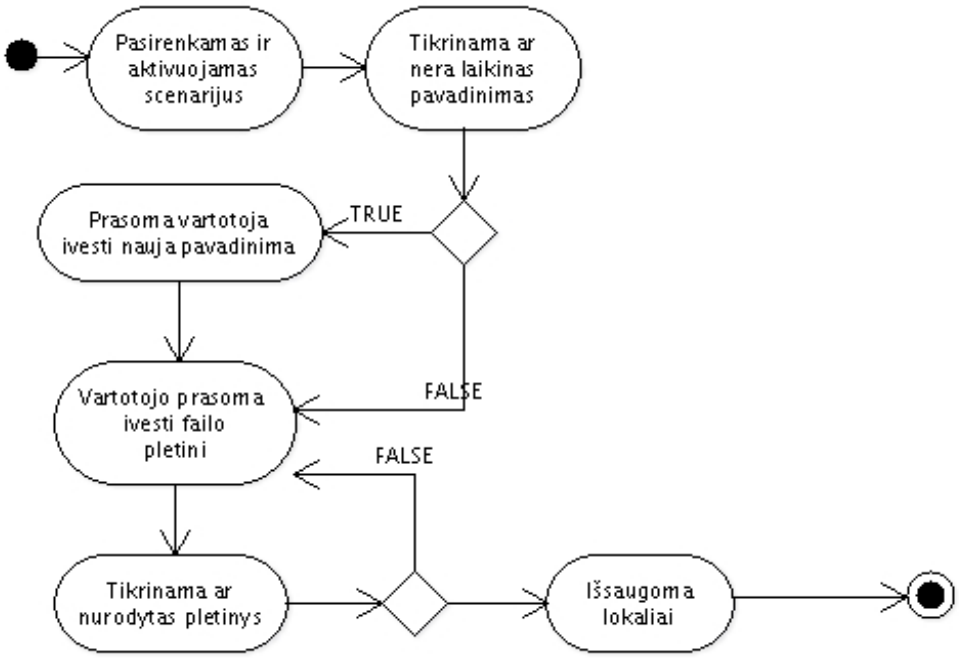
2 pav. Atidaryti - veiksmo diagrama



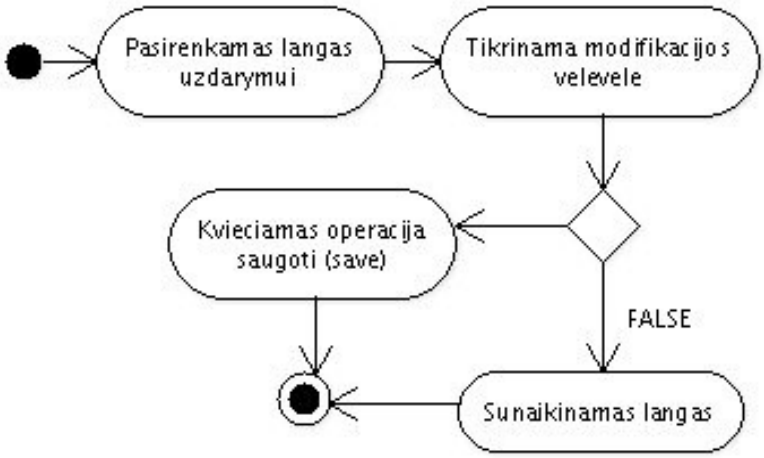
1.3 pav. Trinti - veiksmo diagrama



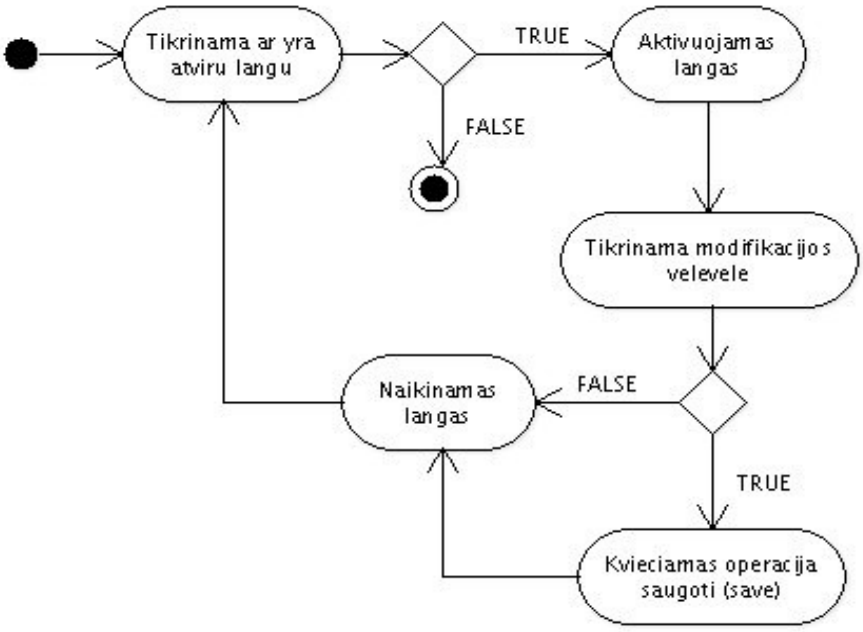
1.4 pav. Importuoti - veiksmo diagrama



1.5 pav. Eksportuoti - veiksmo diagrama



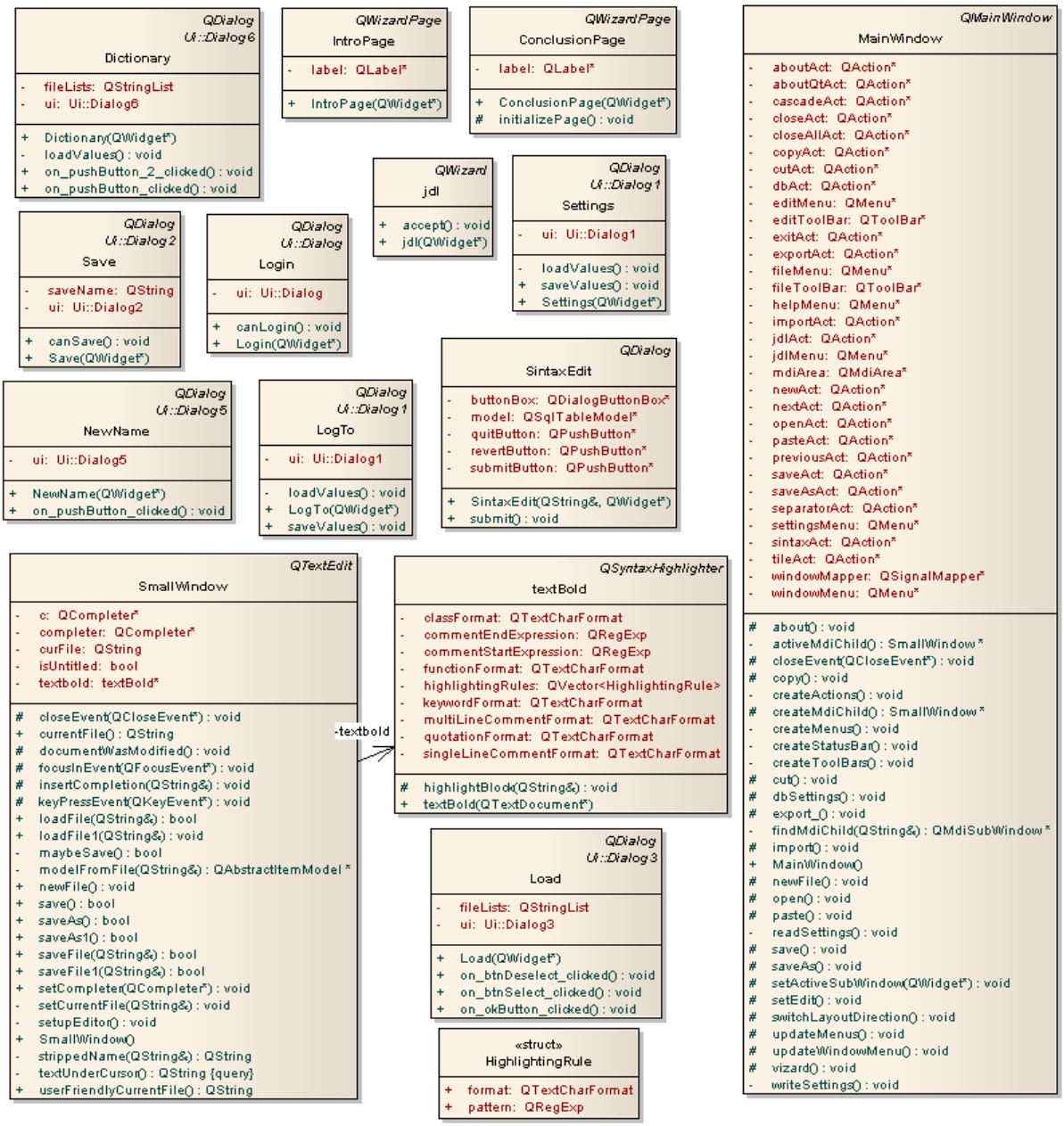
1.6 pav. Uždaryti - veiksmo diagrama



1.7 pav. Uždaryti visus - veiksmo diagrama

2 Priedas

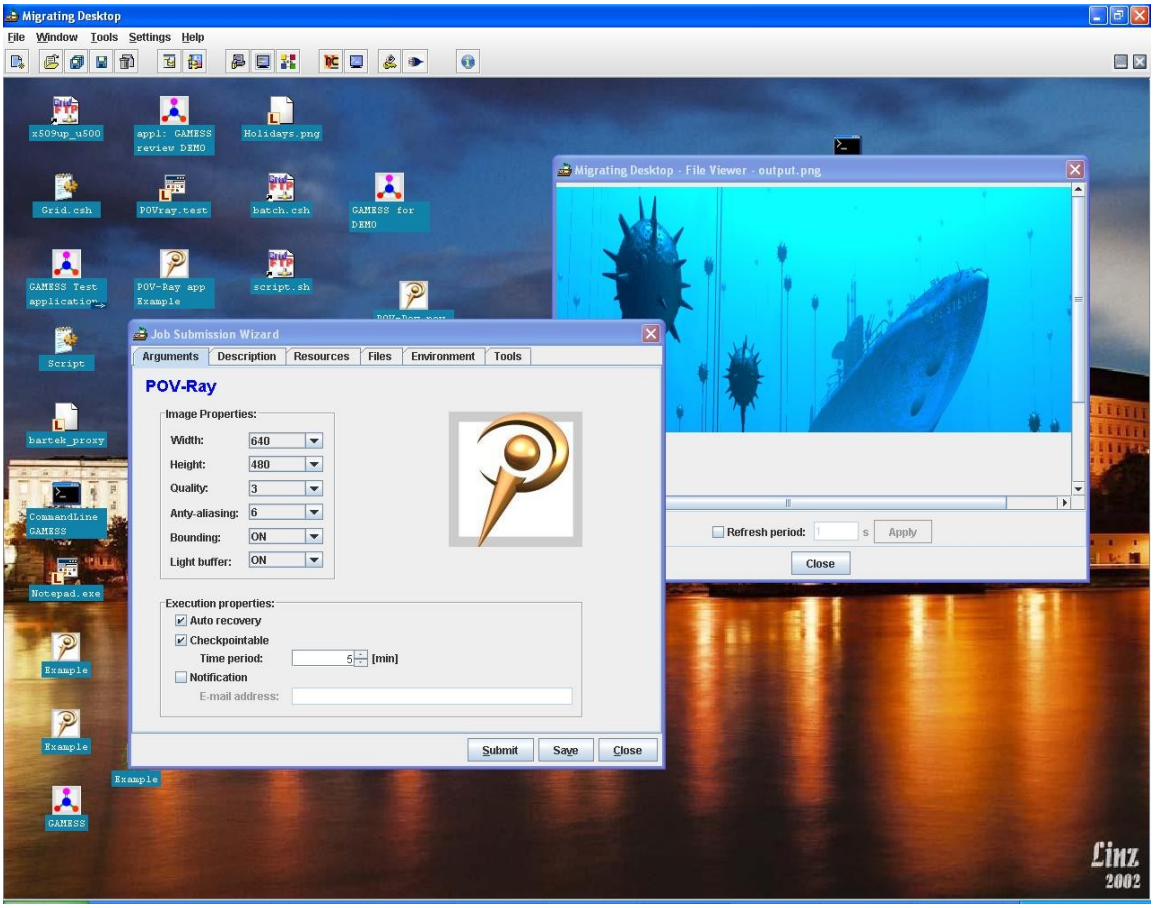
Klasių diagrama



2.1 pav. Klasių diagrama

3 Priedas

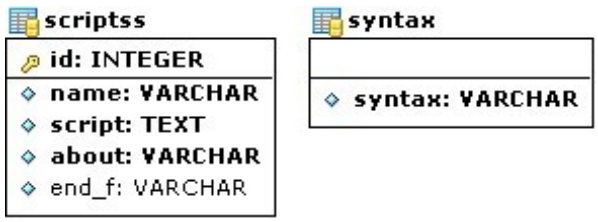
Migrating Desktop



3.1 pav. Migrating Desktop [3]

4 Priedas

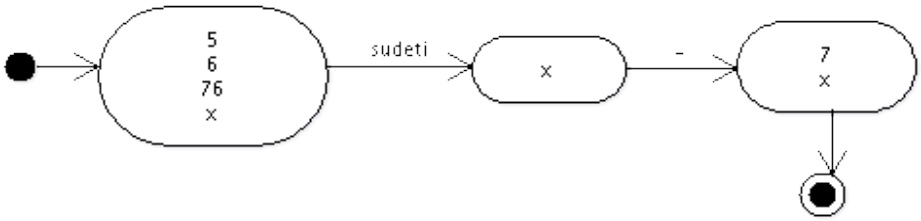
Duomenų bazės lentelės



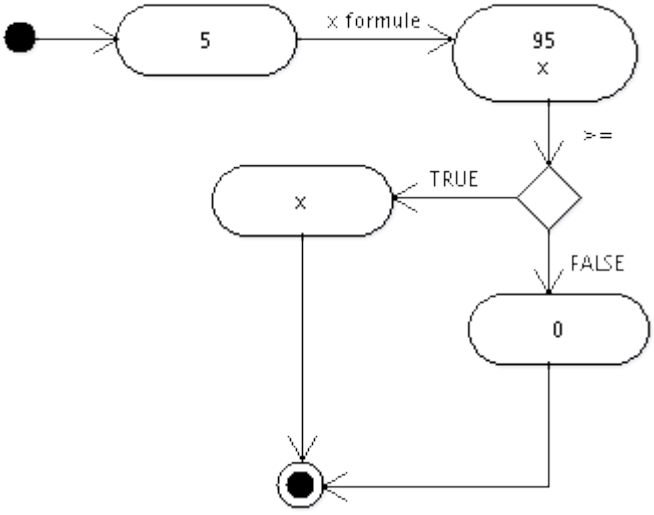
4.1 pav. Duomenų bazės lentelės

5 Priedas

Grafinis scenarijų pavyzdys



5.1 pav. x formulės scenarijus



5.2 pav. x formulės realizacija