KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMŲ INŽINERIJOS KATEDRA

Juozas Dovydaitis

„E-Portfolio“ projektavimas ir metodų tyrimas

Magistro darbas

Darbo vadovas

Habil. Dr. Vytautas Štuikys

Kaunas, 2008

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

PROGRAMŲ INŽINERIJOS KATEDRA

Juozas Dovydaitis

„E-Portfolio“ projektavimas ir metodų tyrimas

Magistro darbas

Recenzentas

2008 05 26

Darbo vadovas

Habil. Dr. Vytautas Štuikys

2008 05 26

Atliko

IFM-2/2 gr. Stud.

Juozas Dovydaitis

2008 05 26

Kaunas, 2008

Turinys

Design and Analysis of e-portfolio

Summary

The main objective of this project was to design and develop an e-portfolio system „ePortfelis”. During this process the analysis of alternative e-portfolio systems was performed, as well as gathering of e-portfolio requirements. The design of developed system was carried according to RUP process. There were reviewed three different software design methods during the software design methods analysis part: SSADM, Booch method and JSP. It was determined that of those three above mentioned methods JSP was most unsuitable for designing e-portfolio systems.

PAVEIKSLĖLIŲ SĄRAŠAS

LENTELIŲ SĄRAŠAS

1 Įvadas

1.1 Dokumento paskirtis

Šiame dokumente nagrinėjama tyrimo sritis susijusi su „E-Portfolio“ tipo sistemų projektavimu ir galimų projektavimo metodų parinkimu. Analitinėje darbo dalyje pristatomos „E-Portfolio“ sistemos, jos vėliau palyginamos. Projektinėje dalyje pateikiama suprojektuotos tokios sistemos – „ePortfelis“, - architektūra. Tiriamojoje darbo dalyje apžvelgiami skirtingi projektavimo metodai ir jų tinkamumas „E-Portfolio“ sistemų projektavimui.

2 „E-Portfolio“ sistemos „ePortfelis“ projektavimas

2.1 Įžanga

Elektroniniams portfeliams, arba el. portfeliams (ang. „portfolio“, „eportfolio“ ar „folio“) skiriama vis daugiau dėmesio kaip galimiems mokymosi, vertinimo ir įvertinimo įrankiams pedagogams, mokiniams ir akademinėms organizacijoms [4]. Šių sistemų idėja yra paprasta, bet galinga – tai suteikia studentams galimybę turėti asmeninė sritį internete, kurioje jie gali užduoti klausimus, skelbti atliktus (atliekamus) darbus, ar tiesiog juos komentuoti.

Galima teigti, kad el. portfelis – tai žmogaus atliktų darbų ir už juos gautų įvertinimų internetinė saugykla-informacinė sistema, leidžianti pristatyti save, savo pasiekimus, įgūdžius ir tikslus tiek tokios sistemos vidiniams vartotojams, tiek ir išoriniams. Tokia sistema tinkama naudoti ten, kur vyksta mokymo(si) procesas – mokyklose, universitetuose ar įvairiose organizacijose. Tokia sistema gali naudotis ne tik studentai (mokiniai), bet ir dėstytojai (mokytojai). Tačiau el. portfelis neapsiriboja vien įvairių failų saugojimu – jį galima panaudoti ir administruojant pačius failus, kontroliuoti jų pasiekiamumą. El. portfelis universitete gali tarnauti kaip tarpininkas tarp studento ir dėstytojo, kaip priemonė įvertinti studentų žinias, kaip galimybė studentams užsibrėžti tikslus, juos įgyvendinti ir stebėti studijų progresą. El. portfelis leidžia greitai praplėsti dėstytojų pateikiamą informaciją, kadangi jame galima saugoti ne tik objektus, bet ir nuorodas į juos – keisdamiesi papildoma informacija studentai ne tik praplečia savo žinias, bet šiame procese dalyvaudamas dėstytojas gali užtikrinti šios informacijos teisingumą. Yra įrodymų, kad elektroninė el. portfelio struktūra leidžia studentams sutelkti dėmesį į turinį ir sąryšius tarp teorijos ir praktikos negu tradiciniai metodai [5].

2.2 Projekto tikslas ir adresatas

Projekto tikslas – sukurti el. portfelio informacinę sistemą „ePortfelis“. Dėstytojai ir studentai sistema gali naudotis interneto naršyklės pagalba – tuo buvo siekiama sumažinti reikalavimus vartotojų programinei įrangai.

Dėstytojai, naudodamiesi sistema, gali:

- Individualiai pritaikyti modulių aplinkas savo reikalavimams;
- Internetu įvertinti studentų darbus;
- Sudaryti ir išsaugoti įvairius testus ar klausimynus;
- Naudoti ne vien tekstinę medžiagą medžiagos pateikimui;
- Bendrauti su studentais.

Naudodamiesi „ePortfeliu“ studentai gali:

- Kurti, įdėti, redaguoti ir ištrinti failus, juos organizuotai išdėstyti;
- Susikurti profilį savęs pristatymui kitiems sistemos vartotojams;
- Susikurti CV;
- Pristatyti savo pasiekimus, įgūdžius ir tikslus;
- Bendrauto su kitais vartotojais įvairiose bendruomeninėse grupėse;
- Komentuoti ne tik savo, bet ir kitų veiklą ir jos rezultatus;
- Prisitaikyti el. portfelio aplinką pagal jų norus;
- Pagal raktinius žodžius ieškoti el. portfelio aplinkoje talpinamos informacijos;
- Bendrauti su dėstytojais.

Būtina atkreipti dėmesį, kad šis sistemos funkcionalumas projekto vykdymo pradžioje tebuvo pradiniai „ePortfelio“ reikalavimai, tačiau iš jų jau galima nustatyti pagrindines sistemos vartotojų veikas, o ir pagrindinius sistemos vartotojus – studentus ir dėstytojus.

2.3 „ePortfelis“ projektavimas

2.3.1 Naudota projektavimo metodika

Projektas buvo vykdomas taikant „Rational Unified Process“ (RUP) metodiką, kuri užtikrina gerą produkto kokybę bei produkto kūrimo proceso valdymą. Projekto dokumentacija paruošta naudojant „Unified Modeling Language“ (UML), suteikiančią galimybę pakartotinai panaudoti projekte surastus problemų sprendimus.

2.3.2 Egzistuojantys sprendimai

Egzistuoja daug el. portfelio sistemų – vienos jų atviro kodo ir visiems laisvai prieinamos, kitos – komercinės, trečios sukurtos pačių jas naudojančių organizacijų, pritaikytos jų specifiniams poreikiams ir reikalavimams. Apžvelgsiu kiekvieno tipo el. portfelio sistemų konkrečius pavyzdžius: „BlackBoard“, „eFolio Minnesota“ ir „OSPortfolio“.

2.3.2.1 Komercinė sistema „Blackboard“

„Blackboard“ – komercinė el. portfelio sistema, kuriama ir plėtojama „Blackboard Inc.“ kompanijos. Šios kompanijos produkcija naudojama ne tik mokyklose ar universitetuose, bet ir įvairiose privačiose įmonėse.

Privalumai [6]:

1. Galimybė naudoti sistemą skirtingose operacinėse sistemose ir su skirtingomis duomenų bazių valdymo sistemomis;
2. Galimybė individualiai pritaikyti kiekvieno modulio aplinką;
3. Administravimo modulis leidžia lengvai publikuoti modulio dokumentus;
4. „Blackboard“ atpažįsta įvairius dokumentų tipus – nuo „Microsoft Word“ iki PDF, taip pat ir video medžiagą;
5. Galimybė modulio medžiagą padaryti pasiekiamą nuo tam tikro laiko;
6. Galimybė internetu įvertinti studentų darbus;
7. Lengvai sudaromi testai ir klausimynai;
8. Galimybė išsaugoti paruoštus testus tolimesniam naudojimui;
9. Testuose galima naudoti keletą pasirinkimo atvejų turinčius klausimus;
10. Testų rezultatai iš karto pateikiami tiek studentui, tiek ir dėstytojui;
11. Integruoti diskusijų ir internetinių pokalbių kambarių moduliai;
12. Studentai gali valdyti visą studijų procesą tiesiai iš sistemos: darbų publikavimą, įvertinimų peržiūrėjimą, peržiūrėti atsiskaitymų kalendorių ir planuoti savo laiką;
13. Studentai ir dėstytojai gali praplėsti pateikiama kurso medžiagą savo pasiūlymais ar nuorodomis į kitus šaltinius;
14. Galimybė apriboti medžiagos pasiekiamumą tik tam tikriems sistemos vartotojams;
15. Galimybė stebėti sistemos vartotojų apsilankymo statistiką kurso aplinkoje;
16. „Building Blocks“ – atviros architektūros iniciatyva, leidžianti pasinaudoti „Blackboard“ API ir pateiktomis specifikacijomis, kurios suteikia galimybę kurti programinę įranga naudojantis „Blackboard“ platforma, taip pat praplėsti šios sistemos funkcijas ar integruoti išorines sistemas.

Trūkumai[6]:

1. Varginantis didelės apimties testo ar klausimyno paruošimo procesas;
2. Brangi sistema;
3. Apribotas įvertinimų įrankis [7].

Bendras „Blackboard“ sistemos įvertinimas – teigiamas. El. portfelio sistemos įtraukimas į studijų procesą „leido studentams sutelkti dėmesį į medžiagą – pagilinti bendrą supratimą, spręsti problemas“[8].

2.3.2.2 Pritaikyta universitetui „eFolio Minnesota“ sistema

Šios sistemos atsiradimą inicijavo „Minnesota State Colleges and University System“ (liet. Minestos valstijos koledžų ir universitetų sistema) (MNSCU), kad būtų galima pristatyti nemokamą internetu pasiekiamą el. portfelį visiems Minesotos dėstytojams, studentams ir darbininkams. Pagrindinė eFolio paskirtis – saugoti vartotojų CV, pakeisti įvairias saugyklas bei pristatyti su konkrečiu asmenim susijusią mokslo, tobulėjimo ir karjeros medžiagą [9].

Privalumai:

1. Galimybė suasmeninti savo dalį šioje sistemoje, publikuoti dokumentus, garso ir vaizdo medžiagą, pateikti nuorodas į vidinius ar išorinius šaltinius ir atlikti apklausas;
2. Galimybė kurti dokumentus pasinaudojant sistemoje esančiais įrankiais;
3. Galimybė susipažinti su įvairių sistemoje esančių įrankių panaudojimo galimybėmis;

Trūkumai:

1. Failų publikavimui skirta sistemos dalis smarkiai apriboja failų dydį;
2. Atpažįstami tik šie dokumentų tipai: HTML, PDF ir „Microsoft Word“;

Kadangi ši sistema naudojama tik Minesotos valstijos mokslo įstaigose, nepateikiama papildomos informacijos apie „eFolio Minnesota“ reikalavimus programinei įrangai.

2.3.2.3 Atvirojo kodo sistema „OSPortfolio“

Vienas didžiausių atviro kodo projektų akademiniame pasaulyje yra „Open Source Portfolio Initiative“ (liet. „Atvirojo Kodo el. portfelio Iniciatyva“) (OSP) [10] – specializuota turinio valdymo sistema, panaši į „WebCT“ [10]. Pagrindines šios sistemos funkcijas galima išskirti tokias:

1. padėti valdyti WEB serverius;
2. padėti dėstytojams paruošti kurso medžiagą;
3. padėti dėstytojams ir studentams susikurti el. portfelį.

Skirtumas tarp dėstytojų ir studentų el. portfelių – pastarųjų skirti parodyti pastangoms, kurios skiriamos įsisavinant kurso medžiagą. Tuo tarpu dėstytojų el. portfelis labiau skirtas pristatyti kurso medžiagą bei darbų pavyzdžius.

Privalumai:

1. Vartotojai gali ne tik peržiūrėti priskirtų kursų medžiagą, bet taip pat prisijungti prie tokių pačių pomėgių turinčių vartotojų, taip sukurdami tam tikras vartotojų grupes;
2. Vartotojai gali valdyti skelbimų ir el. pašto laiškų gavimą;
3. Galimybė naudotis laiko planavimo priemonėmis;
4. Galimybė valdyti sistemos vartotojus
5. Galimybė dėstytojams įvertinti studentų darbus;

6. Galimybė dokumentus ne tik publikuoti sistemoje, bet ir pasinaudojant grafiniu HTML redaktoriumi juos sukurti pačioje sistemoje;

Trūkumai:

1. Sistemos atsako laikas priklauso nuo to, kiek konkrečiu metu ja naudojasi vartotojų;
2. Kiek paini vartotojo sąsaja;
3. Prasta ir neišbaigta pagalbos dalis;

2.3.3 Apžvelgtų sistemų palyginimas

Lyginant sistemas svarbu ne tik jų galimybės ar atliekamos funkcijos. Svarbu ir tai, kokias jų dalis galima redaguoti, ar tai atviro kodo programinė įranga ar ne, ar galima rasti integruotas turinio valdymo sistemas, galų gale – kokia kaina. Įvairių el. portfelio sistemų palyginimas šiais aspektais pateiktas 1 lentelėje [11].

1 lent. El. portfelio sistemų palyginimas

Sistema	**Modifikuojamas e-portfelis**	**Atviro kodo programinė įranga**	**Turinio valdymo sistema**	**Internetinis dienoraštis ar žurnalas**	**Kaina**
eFolio Minnesota	+	-	-	-	**Nemokama**
FolioLive	+	-	-	-	**35$ metams**
TaskStream	+	-	-	-	**25$ semestrui**
Blackboard	+	-	+	-	**10000$ metams**
College Live Text	+	-	-	-	**89$ studentui**
FolioTek	+	-	-	+	**30$ metams**
OSPortfolio	+	+	-	-	**Nemokama**

ePortaro	+	**-**	**-**	+	**10$ vartotojui metams**

2.3.4 „E-Portfolio“ sistemos pagrindinės savybės ir funkcijos

Apžvelgus aukščiau minėtas el. portfelių sistemas, galima išskirti bendras tokių sistemų savybes:

1. Vartotojų valdymas;
2. Vartotojo aplinkos suasmeninimas;
3. Galimybė publikuoti įvairių tipų dokumentus, garso bei vaizdo medžiagą;
4. Dėstytojams leidžiama sukurti kurso aplinką, kurioje galima saugoti su kursu susijusią medžiagą, įvertinti studentų pristatytus darbus bei atlikti jų įgytų žinių patikrinimą;
5. Leidimas sistemos vartotojams bendrauti forumuose, internetiniuose pokalbių kambariuose arba keistis asmeninėmis žinutėmis;
6. Saugoti sukauptus vartotojų dokumentus, nuorodas į kitus šaltinius ir gautus įvertinimus;
7. Dėstytojams leidžiama pakartotinai panaudoti jau sudarytus testus ar klausimynus;
8. Leidžiama naudotis laiko planavimo įrankiais;
9. Užtikrinama, kad publikuojamus dokumentus gali pasiekti tik tie vartotojai, kuriems numatytos tokios teisės.

Toliau panagrinėsime šias savybes plačiau.

2.3.4.1 Failų saugykla

Kaip ir kiekvienos turininio valdymo sistemos, taip ir el. portfelio viena iš pagrindinių dalių – failų saugykla. Čia vartotojai įkelia failus, juos apdoroja ir susistemina. Failai čia taip pat gali atsidurti iš kitų sistemų. Tad visi failai, kurie naudojami ir kuriais dalinamasi, gali būti pasiekti iš el. portfelio aplinkos arba išorinių programinių priemonių pagalba.

Turi būti įgyvendintos tokios failų saugyklos funkcijos [12]:

1. Failai
    a. Vartotojas gali įkelti naują failą, arba keisti ir ištrinti jau esamą;
    b. Kiekvienas failas gali būti aprašomas;
2. Failų hierarchija:
    a. Vartotojai asmeninėse srityse gali kontroliuoti failų hierarchiją;
    b. Vartotojai gali kurti katalogus ir pakatalogius;
    c. Visiems vartotojams turi būti sukurta išankstinė failų hierarchija;
    d. Tiek numatytoji, tiek vartotojo failų hierarchijos gali būti keičiamos bet kuriuo metu, neribojant keitimų kiekio;
3. Paieška
    a. Vartotojai gali ieškoti saugykloje esančių failų pagal vardą ar aprašymą;

Paieška pagal aprašymą leidžia vartotojams tiksliau išskirti paieškos sritį - kursą ar dominančią sritį.

2.3.4.2 Vidinė struktūra

Tam tikri sistemos elementai, vidinė struktūra ir failų saugykla yra sukurti tam, kad padėtų vartotojui plėsti savo skaitmeninius aplankus – e-portfelius. Tam tikri elementai bus prieinami visiems el. portfelio vartotojams, kiti automatiškai bus pridedami priklausimai nuo kurso,

studijų metų ar disciplinų, kurios reikalauja papildomų elementų. Informacija įvedama įvairių formų pagalba.

Funkcijos [12]:

1. El. portfelio vartotojo profilio elementai:
    a. Vartotojai pateikia savo vardą bei kontaktinę informaciją;
    b. Leidžia sukurti kelis profilius;
    c. Vartotojai gali pasirinkti tinkamiausią profilį kiekvienam viešam prisistatymui (draugams, dėstytojams, darbdaviams) – tekstas dinamiškai įtraukiamas į reprezentacinį e-portfelį;
    d. Kai profilis pasikeičia, automatiškai atnaujinami ir šį profilį naudojantys e-portfeliai.
2. CV įrankis:
    a. Skirtingi šablonai funkciniams ir chronologiniams CV;
    b. Galima sukurti keletą CV, kuriuos vėliau galima modifikuoti;
3. Praktinių pasiekimų ataskaitos:
    a. Kursui pritaikyta struktūra, leidžianti vartotojams sukurti ir pateikti praktinių įgūdžių įsisavinimo ataskaitas;
    b. Šios ataskaitos leidžia nustatyti patirties įsisavinimą;

2.3.4.3 Bendradarbiavimo galimybės

Bendradarbiavimo funkcijos el. portfelio vartotojams leidžia dalintis failais, kartu kurti naujus ar redaguoti esamus failus, įvertinti kitų vartotojų darbus. Ši el. portfelio sritis turi būti lanksti, jos vidinė struktūra ir turinys turi priklausyti nuo atskirų bendradarbiavimo grupių.

Savybės [12]:

1. Bendradarbiavimo grupių kūrimas:
    a. Bet kuris el. portfelio vartotojas gali sukurti bendradarbiavimo grupę ir nustatyti, kurie vartotojai gali pasiekti šią grupę;
    b. Šios grupės gali būti sukurtos įvairioms vartotojų grupėms: nuo tai pačiai akademinei grupei priklausančių studentų iki atskiro projekto dalyvių;
    c. Grupės autorius tampa jos savininku, o tuo pačiu ir atsakingu už naujų narių įtraukimą, grupės naudojimo prižiūrėjimą ir jos narių veiklą grupės viduje bei kt.
2. Veiksmai su failais:
    a. Bet kuris grupės narys gali įkelti naujus failus ar redaguoti jau esančius;
    b. El. portfelio vartotojas gali įkelti į grupę jo asmeniniame e-portfelyje esančius failus;
    c. Nurodomas įkelto failo autorius ir įkėlimo data;
    d. Redaguojant failą, kartu nurodoma ir keitimo data bei autorius;
    e. Redagavus failą ir išsaugojus atliktus pakeitimus, apie tai iš karto pranešama visiems grupės nariams;
3. Saugumas:
    a. Visi grupei priskirti failai gauna vieną iš dviejų saugumo lygių:
        i. „Viešas“ – toks failas gali būti panaudojamas el. portfelio vartotojų srityse, juo galima dalintis kaip ir kitais dokumentais;
        ii. „Privatus“ – toks failas pasiekiamas tik grupės nariams, o šios grupės savininkas gali peržiūrėti ir įvertinti šį failą.
    b. Kiekvieno failo saugumo lygis gali būti keičiamas tik grupės savininko arba failo autoriaus;
    c. Failus galima užblokuoti, neleidžiant jų daugiau redaguoti.
4. Bendradarbiavimo grupės struktūra:

a. Grupės autorius gali papildomai patalpinti rubrikas, matricas ar kitokias papildomas struktūras, kurios bus pasiekiamos visiems grupės nariams;

5. Komentarai:

   a. Grupės nariai gali komentuoti visus grupės failų saugykloje esančius failus;

   b. Komentarai gali būti pakeisti arba ištrinti tik jų autorių arba grupės savininko;

Kai failai perkeliami iš grupės failų saugyklos, vartotojai gali pasirinkti, ar išsaugoti jų komentarus, ar perkelti tik failą.

2.3.4.4 Reprezentaciniai el. Portfeliai

Ši el. portfelio savybė leidžia vartotojams kurti suasmenintus e-portfelius ar internetines svetaines, jų turinį užpildant jų el. portfelio srities duomenimis. Vienu metu gali egzistuoti keletas reprezentacinių e-portfelių, kuriuos galima sukurti pasinaudojant pateiktais šablonais. Patys šablonai taip pat gali būti keičiami ir pritaikomi kiekvieno el. portfelio vartotojo poreikiams.

Savybės [12]:

1. Prisistatymo įrankis:

   a. Vartotojai gali pasirinkti norimą šabloną;

   b. Vartotojai nurodo failus iš failų saugyklos;

   c. Šablonai gali būti modifikuojami pagal vartotojo poreikius;

   d. Daugialypės terpės šablonai gali būti naudojami įkeliant garso, vaizdo ar kt. failus;

   e. Specifiniai šablonai sukurti specifinėms sistemos struktūroms (pvz., CV šablonas suderintas su tais CV, kurie sukurti CV įrankio pagalba);

2. Šablonų kūrimo įrankis:

a. ”WYSIWYG” tipo šablonų kūrimo ar redagavimo įrankis skirtas tiems, kurie neturi reikiamų žinių;

b. Visiškas šablonų išeities tekstų priėjimas tiems, kurie turi reikiamų žinių;

c. Sukurti šablonai gali būti išsaugojami ir prienami visiems sistemos vartotojams.

3. Saugumas:

a. Vartotojai gali nustatyti skirtingus pasiekiamumo lygius kiekvienam jų reprezentaciniam e-portfeliui;

b. Vartotojai gali nustatyti kada jų reprezentaciniai e-portfeliai gali būti pašalinami;

4. Komentarai:

a. Vartotojai gali nurodyti kurios reprezentacinio e-portfelio dalys gali būti komentuojamos, o kurios ne;

b. Naršytojai gali palikti komentarus prie kiekvienos e-portfelio dalies, jei tai leidžia e-portfelio savininkas. Komentarai gali būti redaguojami arba šalinami tiek jų autorių, tiek e-portfelių savininkų;

Vartotojui pranešama apie jo e-portfelyje paliktus komentarus;

2.3.4.5 Socialinis tinklas

Socialinis tinklas leidžia el. portfelio vartotojams susisiekti su kitais pagal bendras studijų kryptis, pomėgius, tikslus ir kt. Be to, kad tai padės studentams greičiau rasti juos dominančią informaciją, socialinis tinklas taip pat leis sudaryti nuomonę, kad el. portfelis gali būti naudojamas ir kitokiam bendravimui.

Funkcijos [12]:

1. Vartotojo profilis:

a. Vartotojo profilis prieinamas visiems el. portfelio vartotojams;

b. Tokia informacija kaip studijų metai, moduliai, kursai automatiškai pateikiama vartotojo profilyje;

c. Vartotojai gali keisti profilius pridėdami kontaktinę informaciją, nuorodas, aprašydami savo pomėgius, tikslus, pasiekimus ir pan.;

2. Kitų šaltinių įtraukimas:

   a. Vartotojai gali pasirinktinai portfelyje įdėti nuorodas į reprezentacinius e-portfelius;

   b. Vartotojai taip pat gali profilį papildyti nuorodomis į kitas internetines svetaines (pvz., asmeninis puslapis);

3. Draugai:

   a. Kitus el. portfelio vartotojus galima priskirti ”draugams”, nurodant kurias vartotojo el. portfelio sistemos dalis jie gali pasiekti, o kurias ne;

   b. Draugų sąrašas patalpinamas vartotojo profilyje, iš kurio galima pasiekti draugų profilius;

4. Ryšys tarp pomėgių ir kitų panašumų:

   a. Visi vartotojo profilyje išvardinti pomėgiai yra kartu ir nuorodos į tokius pačius pomėgius turinčių vartotojų sąrašus;

   b. Patekę į tokį sąrašą vartotojai gali peržvelgti, kas dar turi tokius pačius pomėgius, studijuoja tame pačiame kurse ar tokį patį modulį;

5. Paieška:

   a. Vartotojų paieška gali būti vykdoma pagal profilių informaciją;

Paieška gali būti vykdoma pagal tokius parametrus kaip prisijungimo vardas, el. pašto adresas, ICQ, MSN, AOL, Skype vartotojų vardus;

2.3.4.6 Kitos el. portfelio savybės

1. Failų tipai [12]:
    a. Sistema turėtų „suprasti“ daug failų tipų;
    b. Tarp tokių failų tipų turi būti populiariausi: „Word“ ir „Excel“ dokumentai, „PowerPoint“ skaidrės, „MediaPlayer“ ir „QuickTime“ garso ir vaizdo formatai, „JPEG“, „GIF“ ir „PNG“ formatai, naudojami grafiniuose failuose, taip pat ir neformatuoto teksto failai;
2. Failų saugyklos apribojimai [12]:
    a. Kiekvienam el. portfelio vartotojui nustatoma failų saugyklos kvota;
    b. Vartotojai gali prašyti padidinti šią nustatyta kvotą;
    c. Vartotojams, studijuojantiems tam tikrus modulius, kvota gali būti automatiškai padidinama;

2.3.4.7 Papildomi reikalavimai

El. portfelio vartotojo sąsaja turi būti greitai perprantama ir netrukdyti sistemos naudojimui.

Sąsaja taip pat turi būti suderinta su neįgaliųjų vartotojų sistemoms keliamais reikalavimais. Sistemoje turi egzistuoti šablonų biblioteka, kurioje vartotojai galėtų keistis e-portfelių šablonais.

2.3.4.8 „Europass“

Deja, kol kas sunku rasti tokią el. portfelio sistemą, kurioje būtų integruotas „Europass“ gyvenimo aprašymas ir „Europass“ kalbų pasas. Todėl kuriamą el. portfelio sistemos CV modulį reikės pritaikyti „Europass“ keliamiems reikalavimams [13]:

1. „Europass“ CV turi būti rengiamas pagal bendrą europinę curriculum vitae (CV) formą, pasiūlytą Rekomendacija 2002/236/EB;
2. „Europass CV“ apima šiuos skyrius:
    a. Asmeninė informaciją;
    b. Kalbų mokėjimas;
    c. Darbo patirtis;
    d. Įgytas išsilavinimas ir mokslas;
    e. Papildomi gebėjimai;
    f. Kita papildoma informacija.

2.3.5 „E-Portfolio“ sistemos taikymas universitete

Jau išnagrinėtas JAV Minesotos valstijos sistemos pavyzdys rodo, kad el. portfelio taikymas universitete nėra sunki užduotis. Nors KTU universitete jau taikoma panašaus tipo sistema – WebCT, joje pagrindinis akcentas skiriamas dėstomiems moduliams, o ne studentams. Kaip buvo nustatyta „eFolio Minnesota“ atveju [9], paaiškėjo, kad:

1. naudojant el. portfelį pagerėja bendravimas tarp studentų ir dėstytojų;
2. studentai labiau susidomi dėstomu kursu dėl internete lengvai pasiekiamų papildomų resursų.

Tačiau gali iškilti kartais atrodančios paprastos, tačiau reikšmingos problemos [14]:

1. sistemoje naudojant įvairias technologijas reikia užtikrinti, kad bent iš universitete esančių kompiuterių studentai galėtų be didesnių problemų naudotis el. portfeliu;

2. taip pat reikia įvertinti ir el. portfelio korektišką veikimą tiems vartotojams, kurie pasiekia šią sistemą iš kitų vietų (namų, interneto kavinių ir t.t.) – kai kuriems vartotojams gali iškilti nenumatytų problemų, pvz., nebuvimas programinės įrangos, kuri privaloma naudojantis el. portfeliu;

3. kadangi el. portfelis – internetinė sistema, reikia užtikrinti jos pasiekiamumą įvairių naršyklių aplinkose;

4. neretai studentams gali iškilti sunkumų orientuojantis sistemoje ar paprasčiausiai ja naudojantis, todėl reikia paruošti ne tik gerą vartotojo gidą, bet ir skirti laiko mažiau techniškai pažengusių studentų apmokymams.

2.3.6 Įgyvendinimo problemos

Organizacijoms gali iškilti daug klausimų diegiant el. portfelio sistemas, kad ir kam jos būtų skirtos – studentams, dėstytojams ar mokymui [15].

Viena iš tokių probleminių sričių – organizacijos jau naudojama techninė ir programinė įranga – kokią įtaka turės el. portfelis vidinėms organizacijos informacinėms sistemoms? Ar prireiks papildomos techninės įrangos? Jei prireiks, tai kokios? Kokie bus keliami nefunkciniai reikalavimai? Taip pat svarbu žinoti, ar organizacijoje yra tokių žmonių, kurie būtų pajėgus ne tik prižiūrėti naują sistemą, bet ir ją plėtoti. Jei reikės vartotojus supažindinti su el. portfelio naudojimo galimybėmis ir būdais, reikia turėti ne tik infrastruktūrą, bet ir žmones, kurie sugebėtų tai atlikti. Naudojantis sistema gali iškilti abejonių dėl talpinamos informacijos saugumo, o tuo pačiu – ir vartotojų darbų autentiškumo. Kas užtikrins, kad visi dokumentai, prezentacijos ir kt. tikrai sukurti nurodytų autorių? Nemažiau svarbu užtikrinti ir intelektinės nuosavybės apsaugą. Reikia nustatyti, kas gali būti skelbiama el. portfelio aplinkoje, o kas ne.

Vertinant sistemos vartotojų darbus, reikia numatyti kokie įvertinimo standartai bus taikomi, kaip ir kiek ilgai bus saugojami pažangumo rodikliai.

El. portfelis – naujas įrankis, tad reikia numatyti universiteto bendruomenės reakciją į naują sistemą, į tai kaip ji bus priimta ir naudojama.

2.3.7 Principinė sistemos architektūra

Nors el. portfelio architektūros sprendimų pateikiama nedaug, JAV Kalifornijos universitete atliktoje analizėje [16] pateikiami tokie keturi pagrindiniai sistemos komponentai:

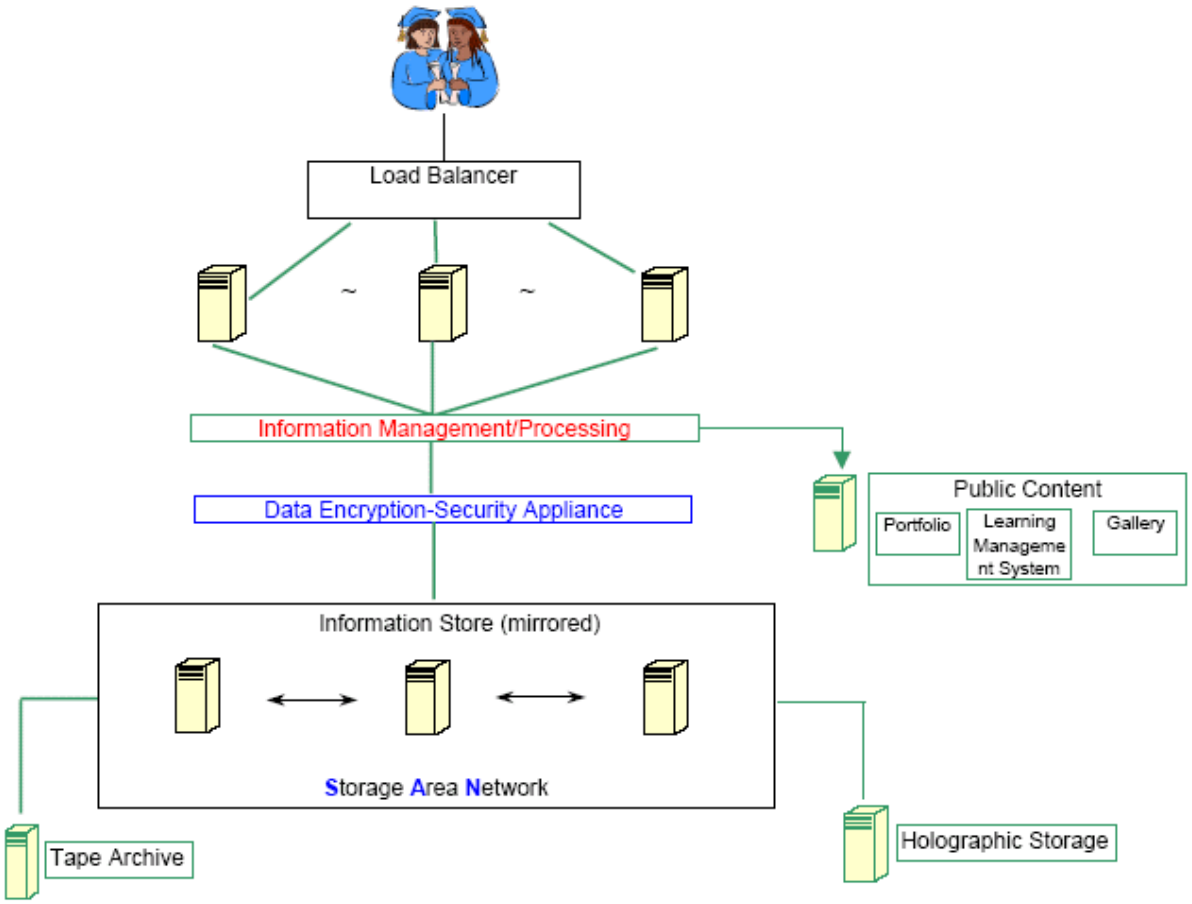
Sistemos apkrovimo reguliatorius (studentų paskirstymas sistemoje)

Informacijos valdymo procesorius (įvairios informacijos pateikimas)

Duomenų kodavimas (užtikrinamas duomenų saugumas)

Informacijos saugykla (duomenų saugojimas)

Kartu pateikiama ir grafiškai pavaizduota architektūra [16]:



1 pav. JAV Kalifornijos universiteto analizėje nustatyta el. portfelio sistemos architektūra

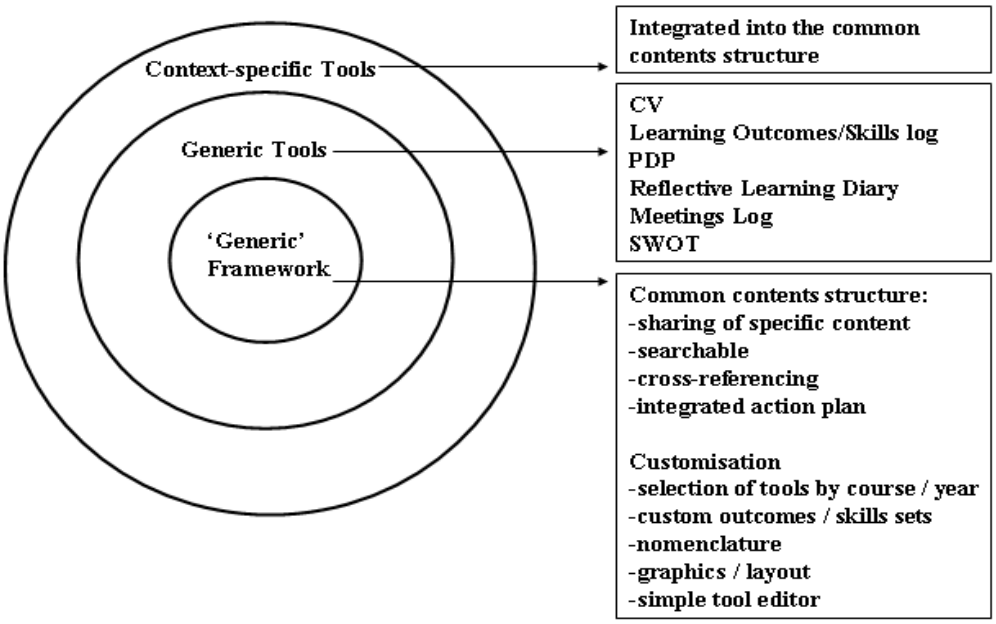
Kadangi sistema naudosis daug vartotojų, būtina užtikrinti jos sklandų veikimą. Tam naudojamas sistemos apkrovimo reguliatorius („Load Balancer“). Informacijos valdymo procesorius („Information Management/Processing“) paskirsto informaciją, kuri saugoma saugykloje („Storage Area Network“). Duomenų saugumas užtikrinamas vykdant duomenų kodavimą („Data Encryption – Security Appliance“). Tokios sistemos informacija saugoma ne tik saugykloje, bet taip pat archyvuojama magnetinėse juostose ir holografinėse laikmenose.

Jungtinės Karalystės Newcastle universitete sukurtoje apibendrintoje ePortfolio sistemoje [17] pateikiama kiek kitokia architektūra (2 pav). Čia ji susideda iš trijų dalių:

1. Nuo konteksto priklausantys įrankiai:

    a. Integruoti į bendrąją informacijos struktūrą;

2. Bendrieji įrankiai:
    a. CV;
    b. Mokymosi proceso rezultatai;
    c. Asmeninio tobulėjimo planas;
    d. Dienoraštis, atspindintis asmeninę mokymosi patirtį;
    e. Susitikimų protokolas;
    f. Stiprybių, silpnybių, grėsmių ir galimybių dalis;
3. Bendroji sistema:
    a. Konkretaus turinio dalijimas;
    b. Paieška;
    c. Integruoti veiksmų planai;
    d. Sistemos įrankių parinkimas pagal kursą ar metus;
    e. Įprasti pasiekimų rinkiniai;
    f. Veiklos logika;
    g. Grafinė sąsaja;
    h. Paprastas įrankių redaktorius.



2 pav. Newcastle universiteto pateikta el. portfelio architektūra

2.3.8 Sistemos realizacija

2.3.8.1 Diegimo aplinka

Kuriama sistema bus įdiegta serveryje ir pasiekiama per interneto naršyklę, tad jos vartotojas papildomų priemonių nereikės. Minimalūs reikalavimai vartotojų programinei įrangai:

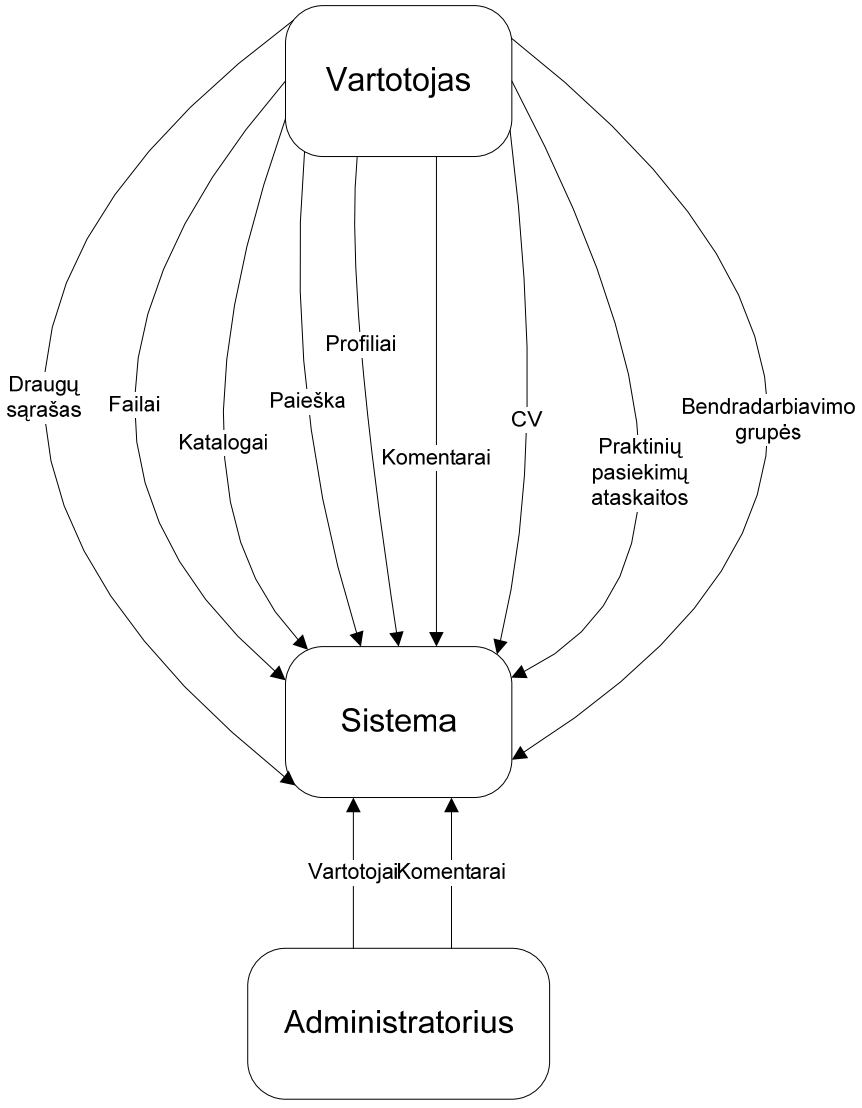
- Operacinė sistema su grafine vartotojo aplinka;
- Interneto naršyklė:
  - Internet Explorer - 6.0 arba naujesnė versija;
  - Mozilla Firefox – 2.0 arba naujesnė versija;
  - Opera – 9.0 arba naujesnė versija;

Minimalūs reikalavimai serverio techninei įrangai:

- Procesorius – 1 GHz Intel arba AMD;
- Spartinančioji atmintis – bent 512 MB;
- Kietasis diskas – bent 40 GB;
- Tinklo plokštė – Ethernet 10/100.

2.3.8.2 Veiklos kontekstas

Pateiktoje diagramoje (3 pav.) vaizduojamas sistemos veiklos kontekstas.



3 pav. Sistemos veiklos konteksto diagrama

Iš aukščiau pavaizduotos diagramos matome sistemos veikimo metu vykstančius duomenų mainus, o taip pat galime susidaryti bendrą supratimą apie sistemai keliamus funkcinius reikalavimus.

2.3.8.3 Veiklos padalinimas

Sistemos veiklos įvykių sąrašas pateikiamas 2 lentelėje:

2 lent. Sistemos veiklos įvykių sąrašas.

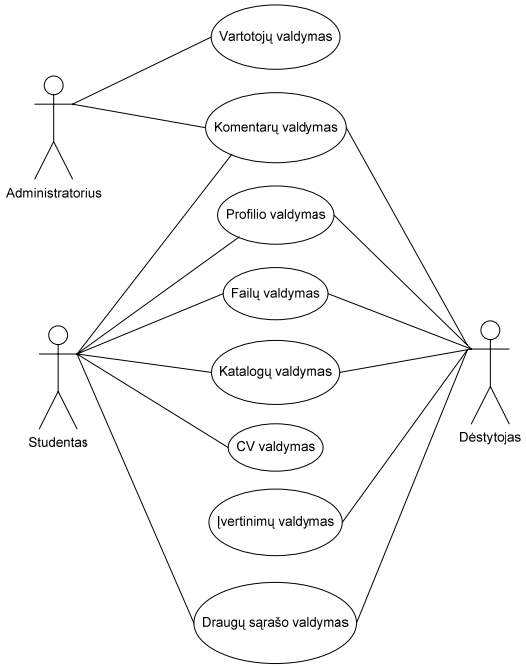
Eil. Nr.	**Įvykio pavadinimas**	**Įeinantys ir išeinantys informacijos srautai**
1	Administratorius registruoja/šalina sistemos vartotoją	Vartotojo vardas, pavardė ir prisijungimo duomenys
2	Vartotojas įkelia naują failą/keičia failą/šalina failą	Failas, jo raktiniai žodžiai ir aprašymas
3	Katalogų hierarchijos kūrimas/redagavimas	Katalogai, jų raktiniai žodžiai ir aprašymai
4	Vartotojas vykdo paiešką	Paieškos užklausa
5	Vartotojas sukuria savo profilį	Vartotojo informacija
6	Vartotojas sukuria savo CV	CV duomenys
7	Vartotojas sukuria praktinių pasiekimų ataskaitą	Kurso praktinių įgūdžių įsisavinimo rezultatai
8	Vartotojas sukuria/redaguoja/panaikina bendradarbiavimo grupę	Informacija apie bendradarbiavimo grupę, pasiekiamumo teisės
9	Vartotojas sukuria/redaguoja draugų sąrašą	Kitų sistemos vartotojų prisijungimo vardai

10	Vartotojas pateikia/redaguoja komentarą	Komentaro tekstas, sukūrimo/atnaujinimo data, vartotojo prisijungimo vardas

Šios lentelės pagalba galime susidaryti bendrą kurtos sistemos struktūros vaizdą. Galima išskirti modulius, bendrus duomenis bei jų tarpusavio ryšius.

2.3.8.4 Panaudojimo atvejai

Atsižvelgiant į sistemos veiklos kontekstą bei surinktus reikalavimus, sudaryta 4 pav. Pateikta sistemos panaudojimo atvejų diagrama. Čia pateikiami tik aukščiausio lygio panaudojimo atvejai ir kiekvienas jų detaliau nenagrinėjamas.



4 pav. Sistemos panaudojimo atvejų diagrama

2.3.8.5 Funkciniai reikalavimai

Sistemos funkciniai reikalavimai rinkti naudojantis Volere šablonu [18]. Toliau pateikiami funkcinių reikalavimų pavadinimai:

FR1. Vartotojui turi būti leidžiama įkelti failus, juos atnaujinti ir ištrinti;

FR2. Vartotojui turi būti leidžiama kurti katalogus, juos pervardinti ir ištrinti;

FR3. Vartotojui turi būti leidžiama vykdyti paiešką;

FR4. Sistemoje turi būti galimybė kurti vartotojus ir juos šalinti;

FR5. Vartotojas turi turėti galimybę sukurti savo profilį, jį redaguoti ir ištrinti;

FR6. Sistema turi suteikti galimybes vartotojui kurti CV, juos redaguoti ir šalinti;

FR7. Vartotojas turi turėti galimybę kurti pasiekimų ataskaitas;

FR8. Vartotojas gali sukurti bendradarbiavimo grupę, ją redaguoti ir šalinti;

FR9. Vartotojas gali sukurti draugų sąrašą, jį redaguoti ir šalinti;

FR10. Sistema turi suteikti priemones vartotojams rašyti komentarus, juos redaguoti ir trinti.

2.3.8.6 Nefunkciniai reikalavimai

Nefunkciniai reikalavimai pateikiami sugrupuoti į kelias pagrindines kategorijas.

- Reikalavimai sistemos išvaizdai:
  - Vartotojo sąsajoje egzistuosiantys elementai turi būti standartiniai interneto naršyklių palaikomi HTML elementai: mygtukai, nuorodos, teksto įvedimo laukai, sąrašai, lentelės ir t.t.;
  - Grafinė vartotojo sąsaja turi būti kuo aiškesnė ir paprastesnė: neperpildyta dėmesį nukreipiančiais elementais, suderintomis spalvomis, aiškiai atskirtais meniu punktais, aiškia meniu hierarchija;

- Jei vietoj teksto naudojami grafiniai elementai, jei turi būti aiškiai suvokiami ir atitikti keičiamą tekstą;

- Reikalavimai panaudojamumui:
  - Sistema turi būti paprasta naudotis;
  - Sistemoje turi būti numatyta galimybė vartotojams ištaisyti jų padarytas klaidas;
  - Vartotojo sąsaja turi būti lietuviška;
  - Jei įmanoma, naudoti iškrentančius sąrašus;
  - Vartotojui turi būti pateikiami raktinės informacijos paaiškinimai;
  - Vartotojas turi lengvai pasiekti reikšmių sąrašus ir jų reikšmes;
- Reikalavimai vykdymo charakteristikoms:
  - Kadangi „ePortfelis“ pasiemias interneto naršyklės pagalba, sistema turi būti suderinama su „Internet Explorer 6.0“, „Mozilla Firefox 2.0“ ir „Opera 9.0“ naršyklėmis ar naujesnėmis jų versijomis;
  - Vienu metu sistema be trukdžio turi aptarnauti iki 50 žmonių;
- Reikalavimai veikimo sąlygoms:
  - Sistema galima pasinaudoti bet kuriuo kompiuteriu, turinčiu interneto ryšį;
- Reikalavimai sistemos priežiūrai:
  - Sistemos priežiūra atlieka tam sukurtas vartotojas – administratorius;
- Reikalavimai saugumui:
  - Sistema ir jos funkcijomis gali pasinaudoti tik registruoti vartotojai; turintys reikalingas teises;
- Kultūriniai – politiniai reikalavimai:
  - Vartotojo sąsaja turi būti realizuota lietuvių kalba;
- Teisiniai reikalavimai:
  - Bet kokia sistemoje talpinama informacija turi atitikti Lietuvos Respublikos įstatymus bei tarptautinius įsipareigojimus;

2.3.8.7 Sistemos architektūros specifikacija

Sistema suskaidyta į 3 pagrindinius paketus (5 pav.):

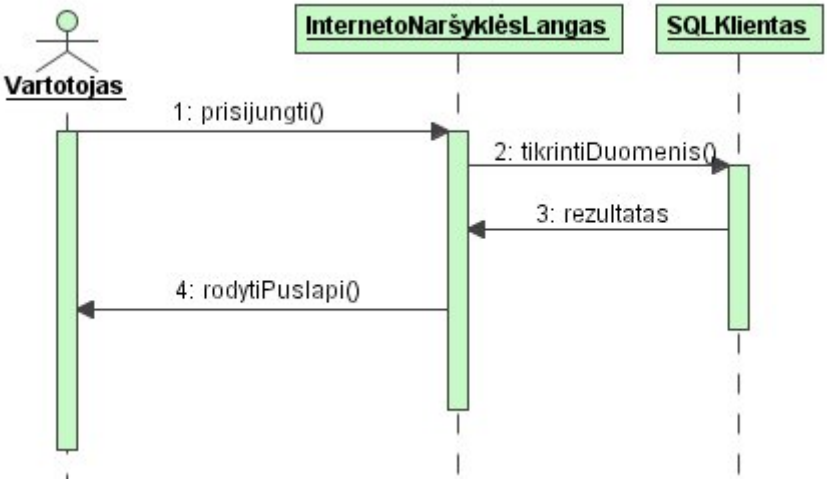
5 pav. Sistemos suskaidymas į paketus

Toliau pateikiami paketų aprašai.

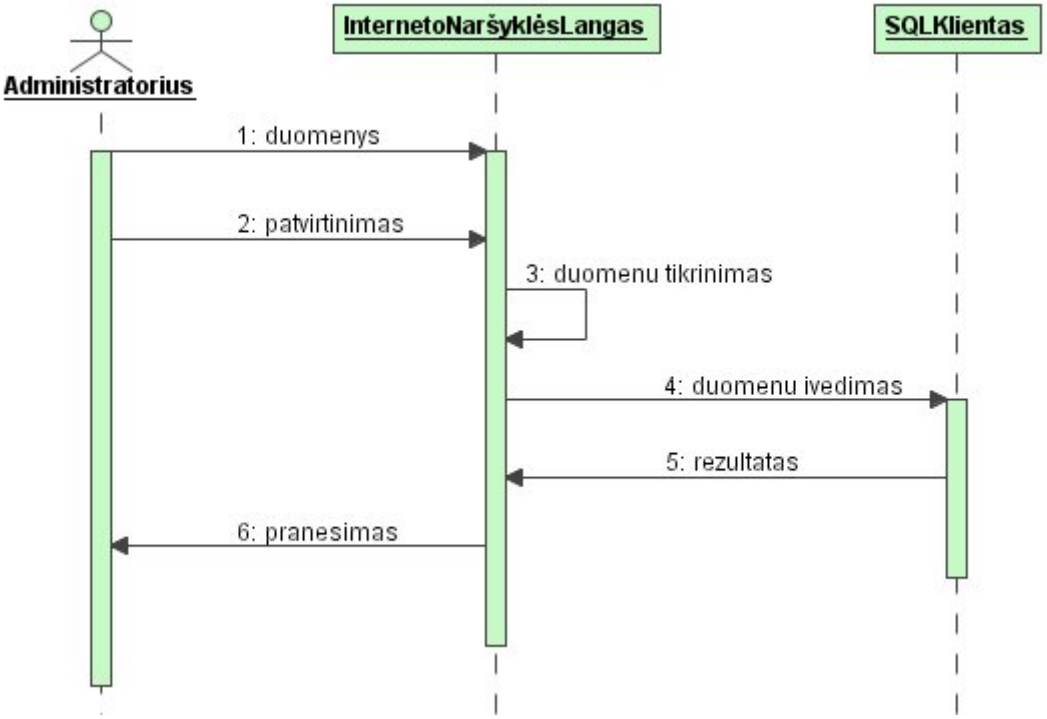
- „Vartotojo sąsajos“ paketą sudaro vartotojui matomi meniu punktai, interneto naršyklės lange pateikiama informacija, priklausomai nuo pasirinkto meniu punkto ir papildomi veiksmai, kuriuos vartotojas gali atlikti iš karto (atnaujinti tuo metu peržiūrimą informaciją arba ją panaikinti);
- Veiklos logiką sudaro sistemos atliekami veiksmai, kuriais vartotojas manipuliuoja sistemoje saugoma informacija: ją įveda, atnaujina ir ištrina;
- Šis paketas atitinka sistemos ryšį su duomenų baze. Paketą sudaro viena klasė „SQLKlientas“, kuri vykdo iš sistemos veiklos logikos pateikiamas užklausas ir gražina jų rezultatus;

2.3.8.8 Sistemos dinaminis vaizdas

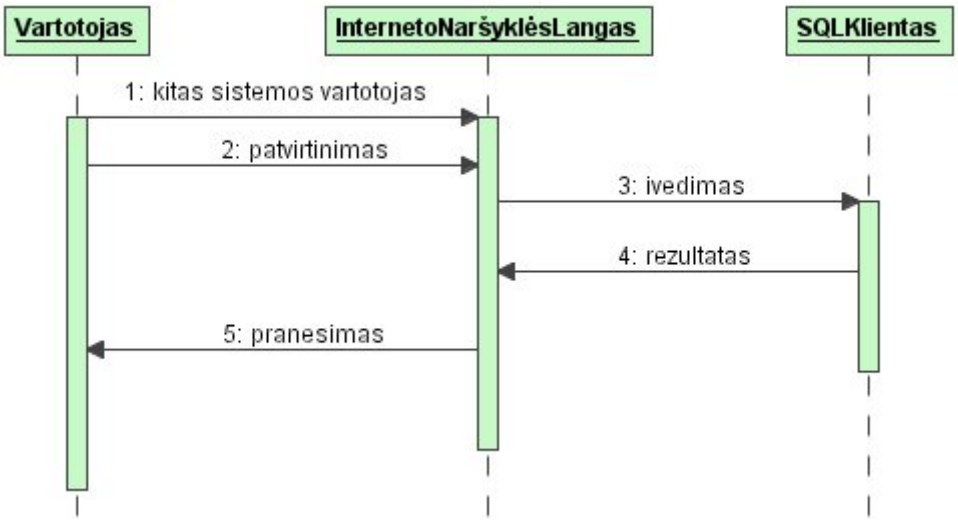
2.3.8.8.1 Sekų diagramos



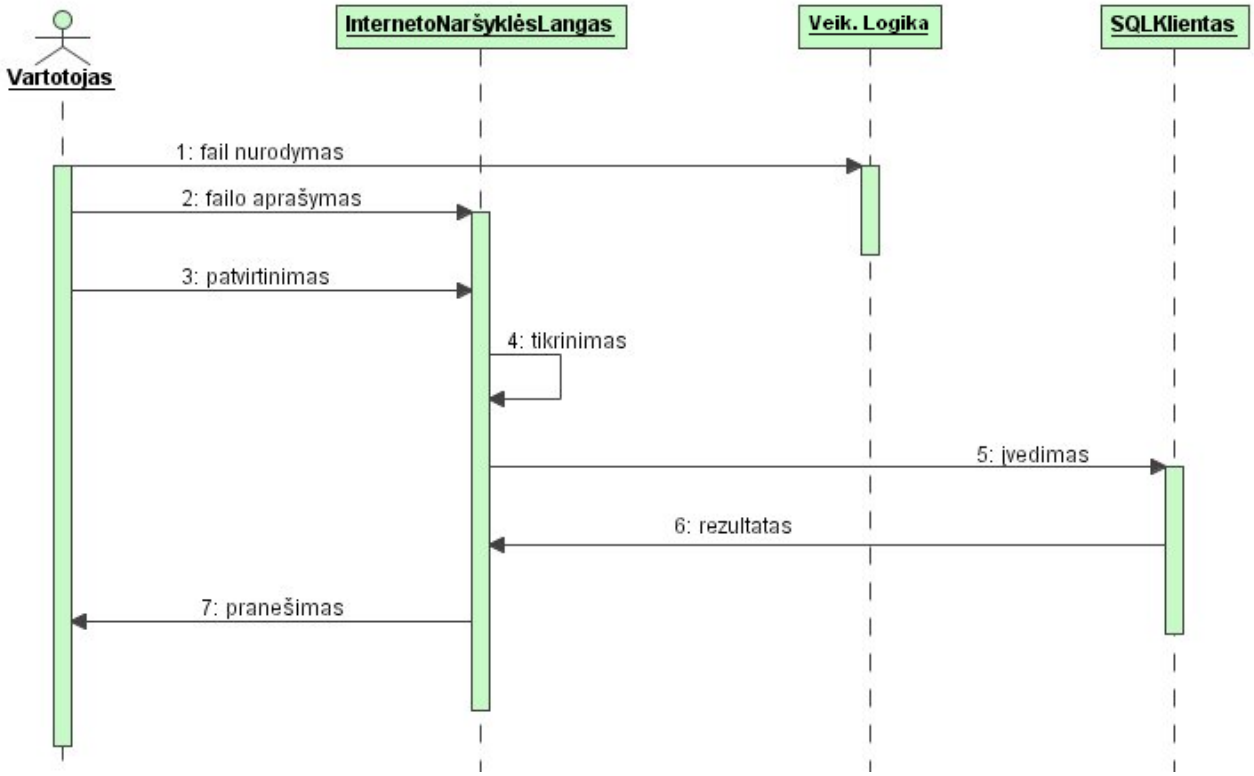
6 pav. Vartotojo prisijungimo prie sistemos sekų diagrama



7 pav. Vartotojo įvedimo į sistemą sekų diagrama

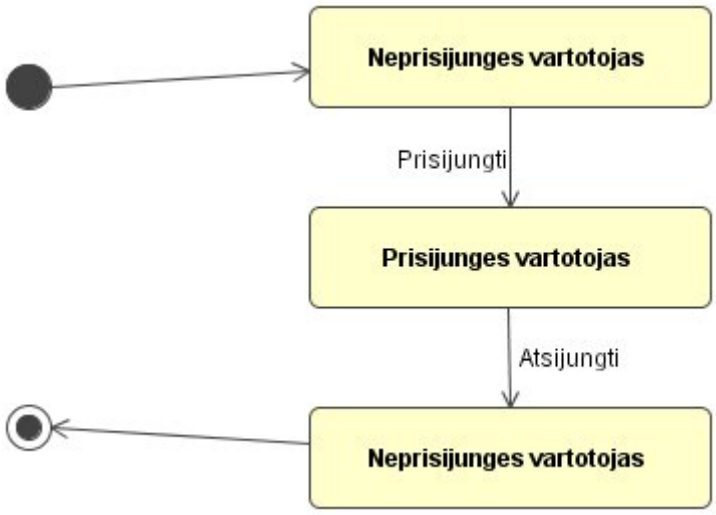


8 pav. Draugų sąrašo papildymo sekų diagrama

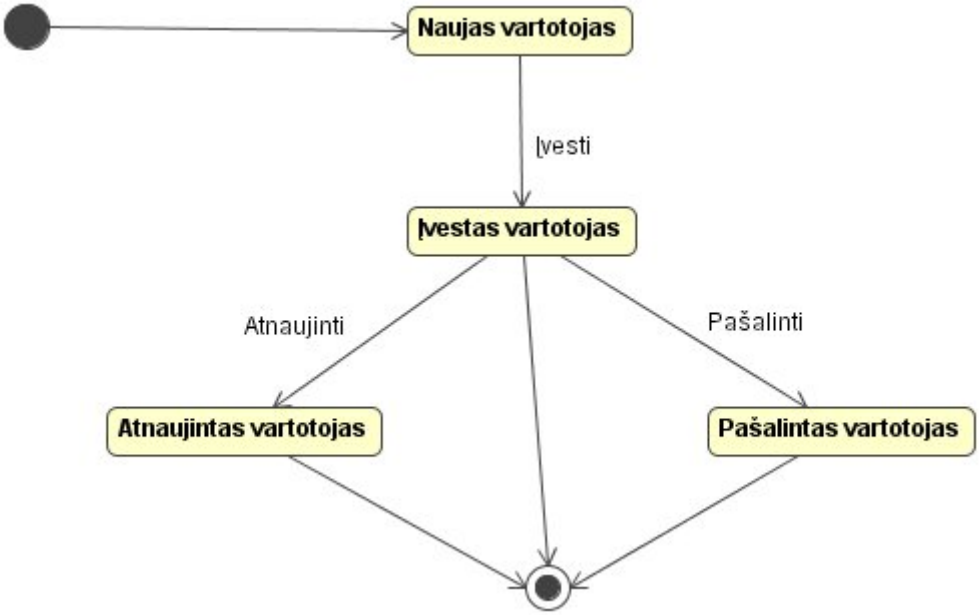


9 pav. Failo įkėlimo sekų diagrama

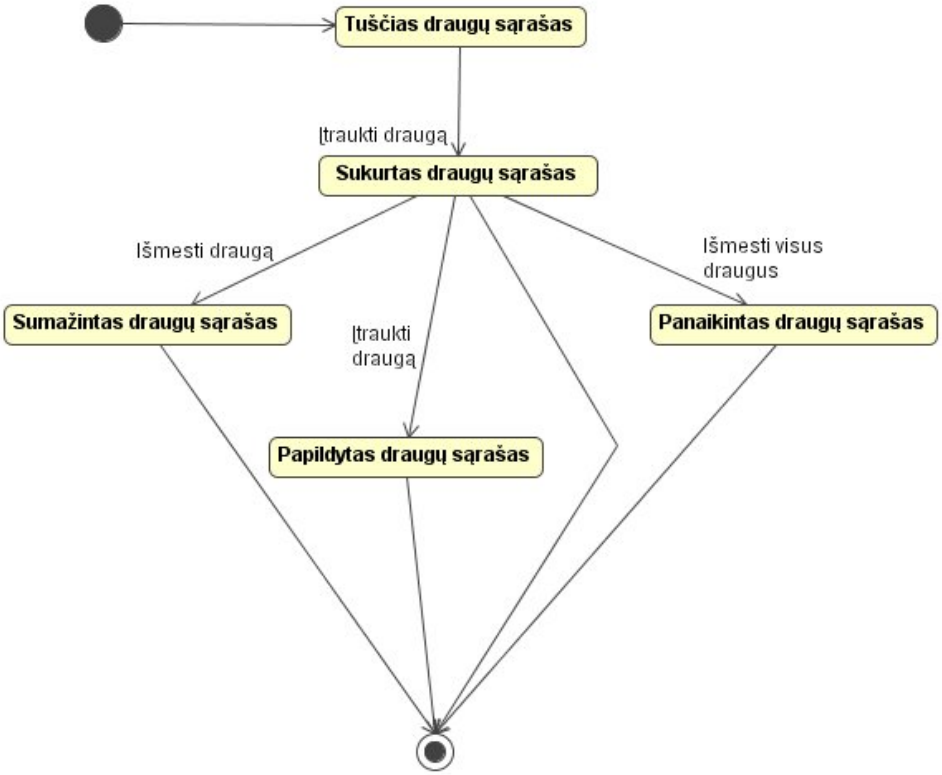
2.3.8.8.2 Būsenų diagramos



10 pav. Vartotojo prisijungimo prie sistemos būsenų diagrama



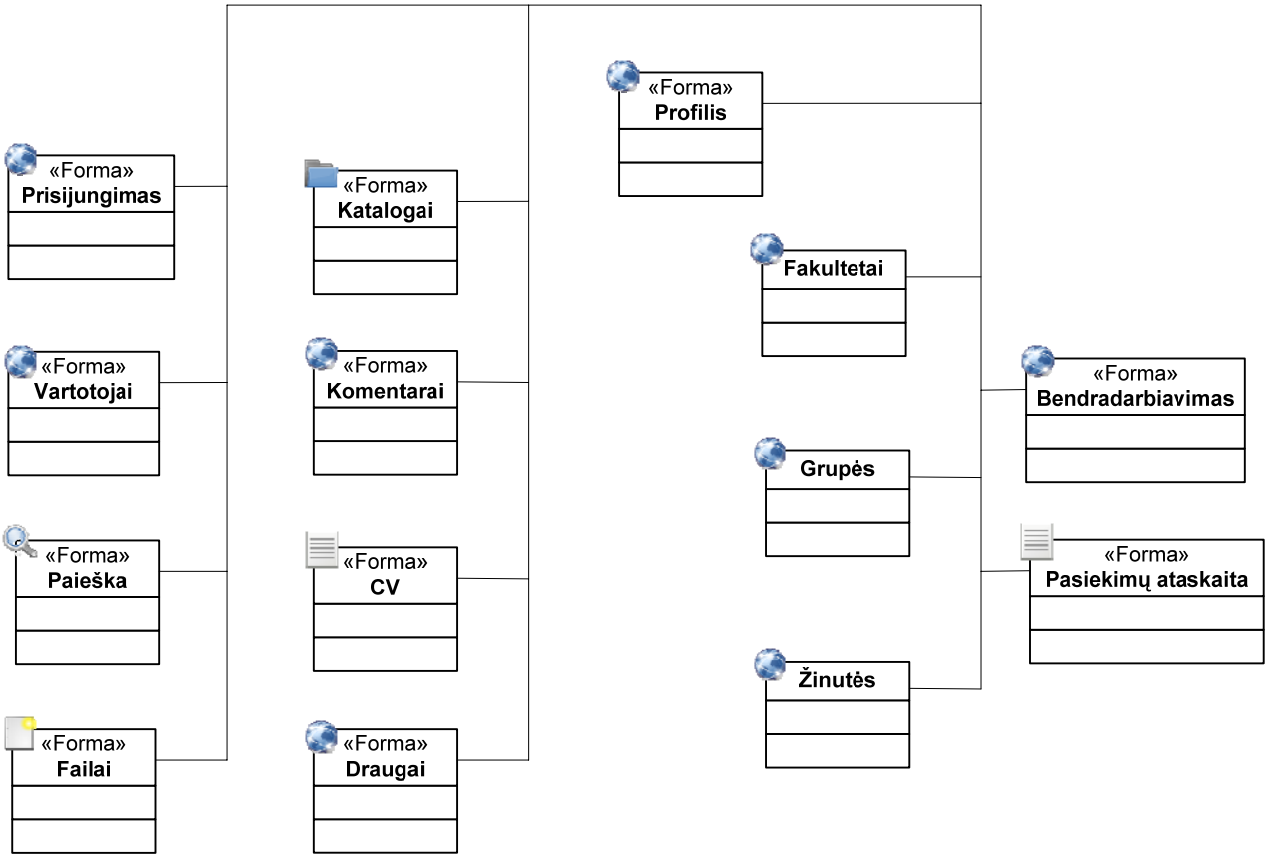
11 pav. Vartotojo įvedimo į sistemą būsenų diagrama



12 pav. Draugų sąrašo sukūrimo būsenų diagrama

2.3.8.9 Vartotojo sąsaja

Vartotojo sąsajos komponento struktūra pateikiama 13 pav.:



13 pav. Vartotojo sąsajos struktūra

Toliau trumpai aprašomas kiekvienas vartotojo sąsajos komponentas:

- Prisijungimas - prisijungimo prie sistemos forma, skirta prisijungimo duomenų įvedimui;
- Vartotojai - vartotojų valdymo forma, skirta vartotojų įvedimui, peržiūrėjimui, redagavimui ir šalinimui;
- Fakultetai - fakultetų valdymo forma, skirta fakultetų įvedimui, peržiūrėjimui, redagavimui ir šalinimui;

- Grupės - grupių valdymo forma, skirta grupių įvedimui, peržiūrėjimui, redagavimui ir šalinimui;
- Žinutės - žinučių forma, skirta žinučių siuntimui tarp vartotojų, skaitymui ir trynimui;
- Paieška - paieškos forma, skirta paieškos užklausų įvedimui;
- Failai - failų įvedimo ir ištrynimo forma;
- Katalogai - katalogų kūrimo, pervardinimo ir trynimo forma;
- Komentarai - komentarų įvedimo ir redagavimo forma;
- CV - CV įvedimo, redagavimo ir trynimo forma;
- Draugai - draugų sąrašo redagavimo forma;
- Profilis - profilio sukūrimo, redagavimo ir trynimo forma;
- Bendradarbiavimas - bendradarbiavimo grupės forma, skirta bendradarbiavimo grupės valdymui;
- Pasiekimų ataskaita - pasiekimų ataskaitų kūrimo forma;

2.3.8.10 Pasiektas rezultatas

Užbaigus vykdytą projektą, gautas galutinis produktas – „ePortfelis“ sistema. Taip pat sukurta ir programinės įrangos techninė dokumentacija, skirta sistemos aprašymui. Dokumentaciją sudaro projekto paraiškos dokumentas, egzistuojančių panašių sprendimų analizė, projekto planas, reikalavimų ir architektūros specifikacija, programinės įrangos licencija, testavimo planas ir vartotojo dokumentacija.

3 Nagrinėjami projektavimo metodai

3.1 Įvadas

Programinės įrangos projektavimo metodų iki šiol sukurta be galo daug, tačiau juos visus galima priskirti ketveriems tipams [20]:

- Funkcinis (struktūrinis) projektavimas;
- Objektiškai orientuotas projektavimas;
- Duomenų struktūromis besiremiantis projektavimas;
- Komponentinis projektavimas;

Funkcinis projektavimas – vienas iš klasikinių programinės įrangos projektavimo metodų, kuriame dekompozicija vykdoma identifikuojant pagrindines PĮ funkcijas, kurios vėliau detalizuojamos „iš apačios žemyn“ (ang. „top-down“) būdu. Struktūrizuotas projektavimas paprastai pradedamas po struktūrinės analizės, o jo rezultatai – duomenų srautų diagramas ir su jomis susijusių procesų aprašymus. Nagrinėjamas metodas – SSADM.

Objektiškai orientuotas projektavimas – strategija, kai kuriamą sistemą sudaro tarpusavyje sąveikaujantys objektai, saugantys savo būsenas ir galintys atlikti tam tikras operacijas. OOP proceso metu projektuojamos objektų klasės ir jų tarpusavio ryšiai. Programavimo metu reikalingi objektai dinamiškai sukuriami atsižvelgiant į jų klasių aprašymus. Darbe bus apžvelgiamas Boocho metodas.

Duomenų struktūromis besiremiantis projektavimas prasideda nuo duomenų struktūrų, kuriomis manipuliuoja kuriama programa, o ne nuo funkcijų, kurias ji atlieka. Pirmiausia nustatomos įeinančių ir išeinančių duomenų struktūros, o programavimas atliekamas atsižvelgiant į gautas diagramas. Nagrinėjamas metodas – JSP.

Komponentinio projektavimo atveju komponentu laikomas nepriklausomas vienetas, turintis gerai aprašytas sąsajas ir priklausomybes. Toks vienetas yra platinamas laisvai. C. Szyperski pateikia tokius kriterijus, kuriuos turi atitikti PĮ komponentas [22]:

- Keleriopas panaudojimas;
- Nuo konteksto nepriklausomas;
- Komponuojamas su kitais komponentais;
- Enkapsuliuotas, t.y. neatsiskleidžiantis per savo sąsajas;
- Nepriklausomo platinimo ir versijavimo vienetas.

Kadangi „E-Portfolio“ sistema nėra komponentinė, šiame darbe nėra tikslinga nagrinėti komponentinio projektavimo metodus.

3.2 Struktūrinio projektavimo metodas SSADM

SSADM sukurtas praėjusio amžiaus devintojo dešimtmečio pradžioje sistemų analizei ir projektavimui Jungtinėje Karalystėje. Šis metodas naudoja rašytinį tekstą ir diagramas viso sistemos projektavimo ciklo metu, nuo pradinės idėjos iki fizinio projekto. Galima išskirti tris pagrindines atliekamas veiklas [23], naudojant šį metodą:

- Loginis duomenų modeliavimas. Identifikuojami, modeliuojami ir dokumentuojami projektuojamos sistemos duomenų reikalavimai. Šie duomenys išskaidomi į esybes ir ryšius tarp tų esybių.
- Duomenų srautų modeliavimas. Čia svarbu duomenų srautai sistemoje. Tiriami procesai, duomenų saugyklos, išorinės esybės ir duomenų srautai.
- Esybių veiklos modeliavimas. Identifikuojami, modeliuojami ir dokumentuojami esybių įvykiai sistemoje bei tvarkai, kaip tie įvykiai iššaukiami.

Kiekvienos iš šių veiklų rezultatai (modeliai) suteikia skirtingą požiūrį į tą pačią sistemą, ir kiekvienas toks modelis reikalingas pilnam projektuojamos sistemos modelio sudarymui. Atliekant šias veiklas, jų rezultatai tarpusavyje palyginami, užtikrinant kuriamos sistemos užbaigtumą ir tikslumą.

SSADM projektai suskaidomi į penkerius vienas po kito einančius žingsnius, kurie savo ruožtu toliau suskaidomi į smulkesnes hierarchijas [24]:

- Sistemos analizė. Šio etapo metu sukuriama duomenų srautų diagrama (DFD), o pats etapas suskaidomas dar smulkiau:
    - Sukuriamas verslo taisyklių modelis. Kartu tiriamos verslo įvykiai ir verslo taisyklės, esančios tarsi sistemos specifikacijos įėjimai.
    - Reikalavimų nustatymas. Šio žingsnio tikslas – nustatyti sistemos aplinkoje iškylančias problemas, kurias vėliau reikės išspręsti.
    - Informacijos apdorojimo nustatymas. Aptikti duomenų srautai atvaizduojami duomenų srautų modeliu, kuris šiuo atveju atspindi esamą situaciją, į jį neįtraukiant planuojamų pakeitimų.
    - Duomenų nustatymas. Čia tiriami sistemoje naudojami duomenys, neatsižvelgiant į tai kaip jie saugojami. Šio žingsnio rezultatas – duomenų modelis, atspindintis esamą situaciją.
    - Loginis vaizdas. Čia gaunamas loginis esamos situacijas vaizdas, kuris padeda suprasti iškylančias problemas.
- Reikalavimų analizė. Etapas susideda iš dviejų žingsnių:
    - Egzistuojančios aplinkos tyrinėjimas. Nustatomi sistemos reikalavimai ir sumodeliuojama egzistuojanti sistemos aplinka. Modeliavimas susideda iš DFD ir loginių duomenų struktūrų, skirtų sistemos procesams ir duomenų struktūroms. Tuo pačiu paruošiama keletas galimų sistemos sprendimų, atitinkančių reikalavimus.
    - Užsakovui pristatomi paruošti sprendimai, iš kurių išrenkamas tik vienas. Jis atitinka kuriamos sistemos ribas.
- Reikalavimų specifikacija. Šiame etape detalizuojami funkciniai ir nefunkciniai reikalavimai. Tai apima sistemos aprašymą per duomenų srautus ir jos vartotojų roles. Nustatomos sistemos funkcijos. Atliekant reikalavimų specifikaciją, nustatytos sistemos funkcijos gali būti atnaujinamos arba papildomos. Sukuriami specifikacijų prototipai. Jie naudojami apibūdinti sistemą animuota forma. Tokio demonstravimo tikslas – parodyti, kad reikalavimai pilnai suprasti ir gauti papildomų reikalavimų vartotojo sąsajai.
- Loginis sistemos specifikavimas – sukuriamas loginis sistemos projektas. Nurodomos galimos techninės kūrimo aplinkos. Kaip ir reikalavimų analizės etape, užsakovui pateikiami keli techniniai sprendimai, iš kurių išrenkamas vienas, labiausiai tenkinantis užsakovą.

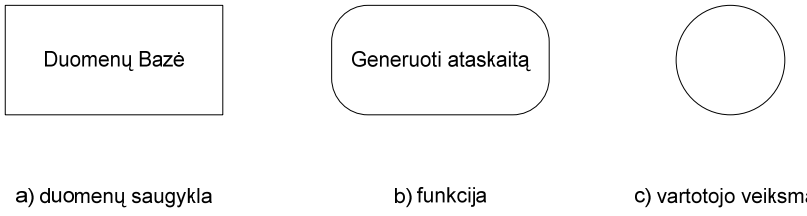
- Fizinis projektas. Loginė ir techninė sistemos specifikacijos panaudojamos suprojektuojant fizinę duomenų bazę. Čia jau naudojama pasirinktos aplinkos programavimo kalba ir savybės.

Žemiau aptarsime SSADM naudojamas diagramas.

3.2.1 Duomenų srautų diagramos

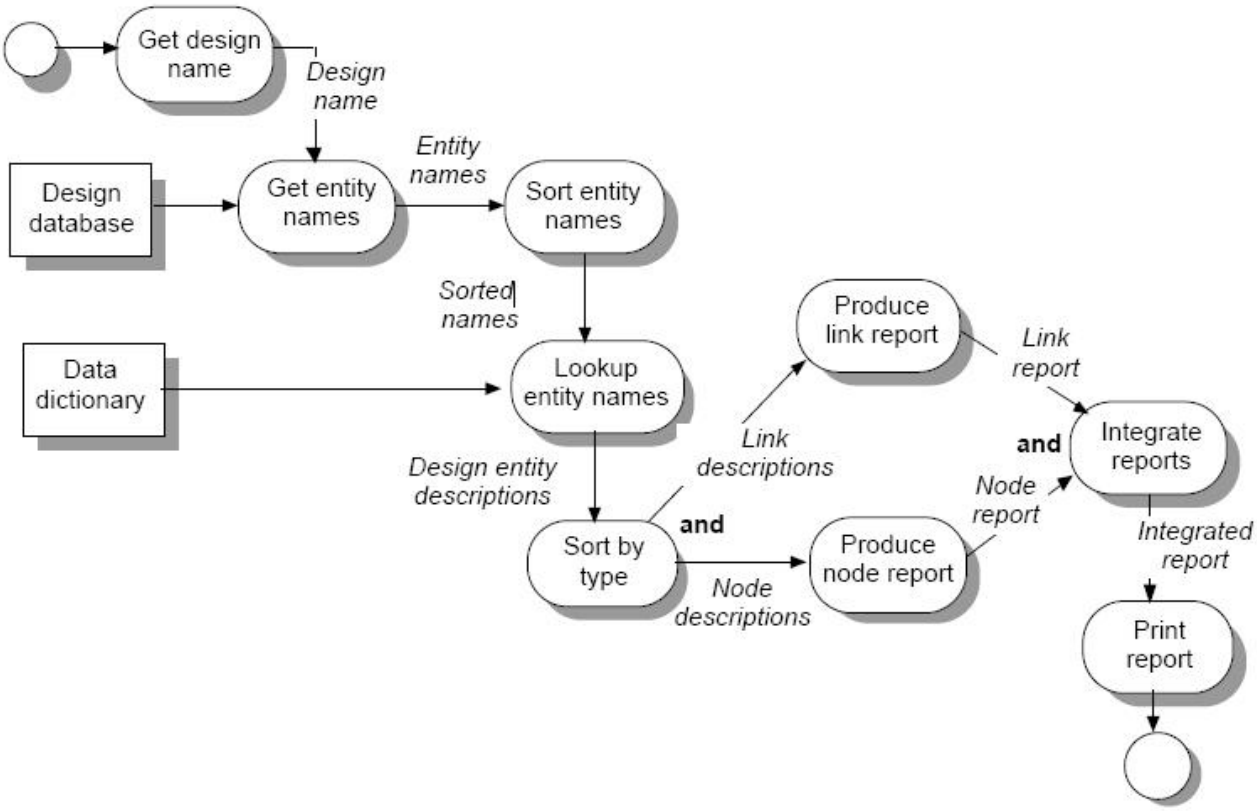
Duomenų srautų diagramos atvaizduoja duomenų kelią sistemoje, ir kokios funkcinės transformacijos apdoroja tuos duomenis. DFD padeda nustatyti, kurioje sistemos vietoje duomenys ateina, ir kurioje išeina. DFD įeina į ne vieną projektavimo metodą, be to, daugelis CASE įrankių palaiko tokių diagramų kūrimą [21]. Nors kiekvienas metodas gali naudoti skirtingus DFD žymėjimus, galima išskirti tokius simbolius:

- Stačiakampis apvaliais kampais (14 pav., b) atitinka funkcijas, kurios įėjimus transformuoja į išėjimus. Transformacijos pavadinimas atitinka tai atliekančią funkciją.
- Stačiakampiais (14 pav., a) vaizduojamos duomenų saugyklos. Joms taip pat suteikiami aprašomieji pavadinimai.
- Apskritimai (14 pav., c) atitinka vartotojų veiksmus su sistema.
- Rodyklės (15 pav.) vaizduoja duomenų judėjimo kryptį, o rodyklių pavadinimas – judančius duomenis.
- Raktiniai žodžiai „ir“ ir „arba“ (15 pav.) atitinka įprastinę loginę reikšmę. Naudojami susieti srautus kai transformacija gali turėti daugiau nei vieną įėjimą ar išėjimą.



14 pav. DFD žymėjimai

Žemiau pateikta DFD diagrama (15 pav.) [21], atvaizduojanti ataskaitų generatorių.



15 pav. DFD diagrama

Ataskaitų generatorius pateikia ataskaitą, aprašančią visas įvardintas esybes duomenų srautų diagramoje. Vartotojas pateikia diagrama vaizduojamo projekto vardą. Tuomet ataskaitų generatorius suranda visus pavadinimus, naudojamus DFD. Informacija apie kiekvieną vardą gaunama iš duomenų žodyno („data dictionary“). Visa tai sudedama į ataskaitą, kurią išveda sistema.

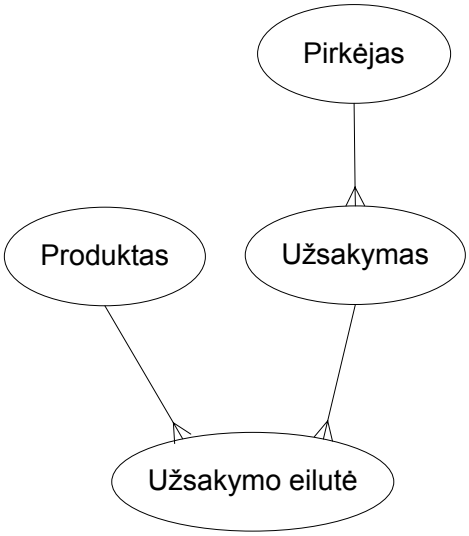
Paprastai naudojamos daugiau nei viena DFD. Pirmiausia pateikiama aukščiausio lygio DFD diagrama, kuri vėliau dekomponuojama į kelias žemesnio lygio DFD, kuriose detaliau analizuojamos tam tikros sistemos funkcijos. Jei reikalinga, dekomponuojama tol, kol tenkinamas norimas detalizacijos laipsnis.

Jau iš pateikto pavyzdžio matosi, kad DFD parodo funkcines transformacijas, tačiau neatskleidžia jų atlikimo būdo.

3.2.2 Loginės duomenų struktūros

Loginės duomenų struktūros naudojamos atvaizduoti sistemoje esančias duomenų struktūras per esybių ir jų tarpusavio ryšių diagramą (16 pav.):

- Esybė – įsivaizduojamas ar realus prasminis objektas, apie kurį kuriamoji sistema turės saugoti informaciją [21];
- Ryšys – reikšminga asociacija tarp esybių [21].



16 pav. Loginė duomenų struktūra

Aukščiau pateiktoje LDS (16 pav.) pavaizduotos keturios sistemos esybės: pirkėjas, produktas, užsakymas ir užsakymo eilutė. Iš jų tarpusavio ryšių matome, kad pirkėjas gali pateikti daugiau nei vieną užsakymą, tačiau užsakymas ateina tik iš vieno pirkėjo. Užsakymą gali sudaryti daug užsakymo eilučių, tačiau užsakymo eilutė įeina tik į vieną užsakymą. Produktas gali įeiti į daugelį užsakymo eilučių, tačiau užsakymo eilutė susideda tik iš vieno produkto.

Šios diagramos SSADM metode gali būti plečiamos ir tobulinamos, priklausomai nuo to, kokiame etape vyksta projektavimas. Iš jų vėliau gaunama fizinė duomenų bazė.

3.2.3 Privalumai ir trūkumai

SSADM privalumai:

- Bent jau teoriškai, SSADM leidžia gerai planuoti, valdyti ir kontroliuoti projektą.
- Naudojant SSADM, dėmesys skiriamas vartotojų poreikių analizei. Tuo pačiu metu sukuriamas sistemos modelis. Tiek poreikių analizės, tiek ir sistemos modeliavimo rezultatai palyginami, ar tinka vienas kitam.
- SSADM nereikalauja gilių ar specifinių projektuotojo įgūdžių, tad galima lengvai apmokyti šio metodo naudojimą. Be to, galima naudoti įvairius modeliavimo ir diagramų kūrimo įrankius, kurie palengvina SSAMD įsisavinimą ir taikymą.
- SSADM gali sumažinti klaidų kiekį vien dėl to, kad jau pačioje projektavimo pradžioje gaunamas tam tikros kokybės sistemos vaizdas, kuris vėliau nuolatos tikrinamas ir tobulinamas.
- Šis metodas atskiria loginį ir fizinį sistemos projektavimą, tad sistemos nereikia iš naujo realizuoti, pasikeitus techninei ar programinei įrangai.

Trūkumai:

- SSADM skiria didelį dėmesį sistemos analizei ir dokumentacijai. Iš čia kyla grėsmė „peranalizuoti“, kas gali užtrukti ilgą laiką ir pareikalauti didelių išlaidų. Didelėms sistemoms aprašyti naudojamos diagramos gali pasidaryti neaiškios, kadangi turi būti įtraukiami visi reikalingi duomenų srautai.

3.3 Boocho OOP metodas

Šis metodas sukurtas Grady Boocho, dirbusio „Rational Software“ kompanijoje. Boocho metode dėmesys kreipiamas į keturis fundamentalius modelius [28]:

- Loginis modelis
- Fizinis modelis
- Statinis modelis

- Dinaminis modelis

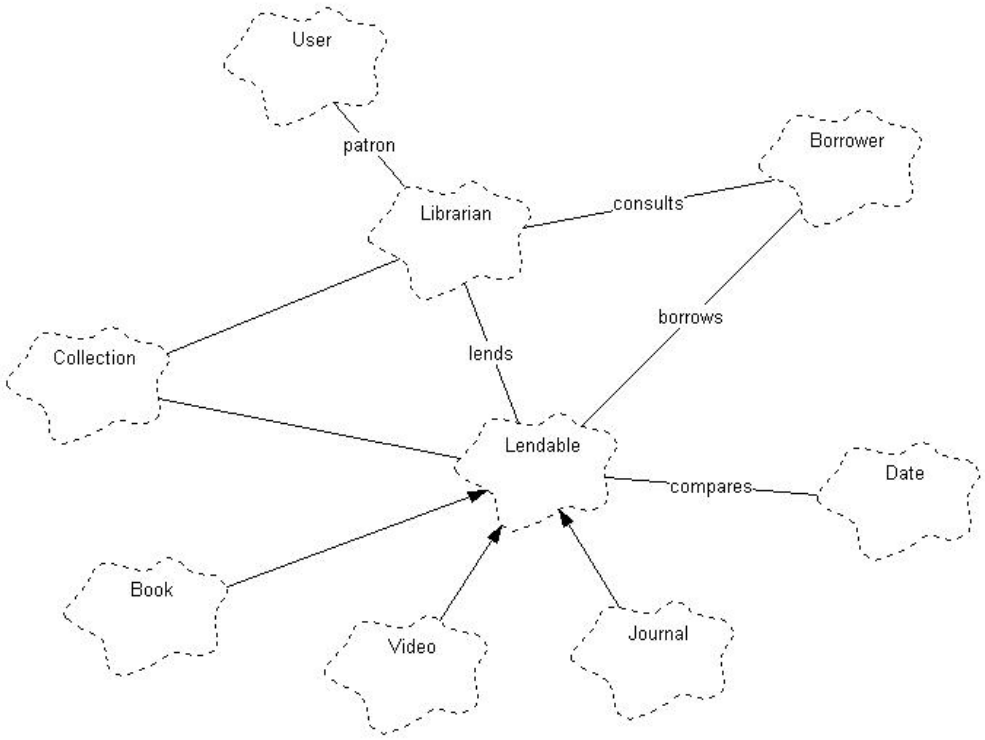
Kiekvieną šių modelių galima dokumentuoti naudojant kelias skirtingas diagramas [29]:

- Klasių diagrama
- Objektų diagrama
- Modulių diagrama
- Būsenų diagrama
- Sąveikų diagrama
- Proceso diagrama

Taikant šį metodą svarbu suprasti, kad vieną modelį gali atitikti daugiau nei viena diagrama. Kai kurios diagramos gali turėti informaciją apie kelis modelius, o kai kurių modelių informacija gali būti padalinta tarp kelių diagramų. Toliau aptarsime Boocho metode naudojamas diagramas. Kadangi naudojant proceso diagramas tenka kalbėti apie daugiagijį („multithreaded“) programavimą, tokio tipo diagramų nenagrinėsime.

3.3.1 Klasių diagrama

Boocho metode šios diagramos naudojamos klasių išskyrimui ir jų tarpusavio ryšiams atvaizduoti loginiame sistemos vaizde (modelyje). B. Brooks pateikia tokį Boocho metodo klasių diagramos pavyzdį (17 pav.):

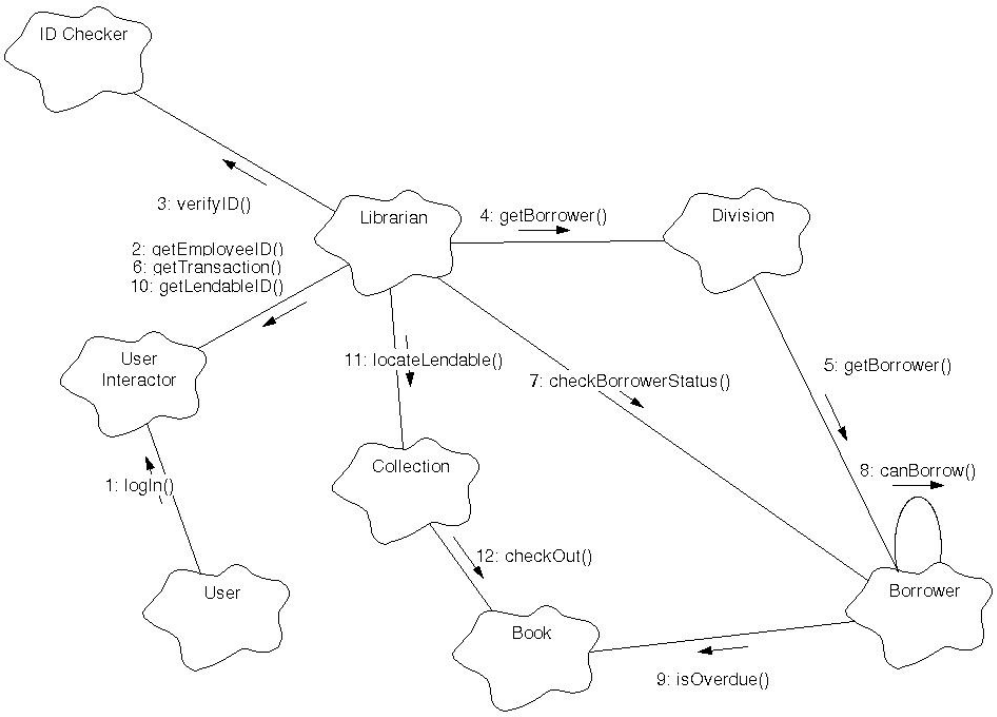


17 pav. Boocho klasių diagrama

Aukščiau pateiktame paveiksle matomos ne tik klasės, bet ir jų ryšiai su kitomis klasėmis. Tokia aukščiausio lygio klasių diagrama gali būti gaunama iš pradinio problemos aprašymo. Ryšiu susietos klasės gali iššaukti viena kitos metodus ar keistis pranešimais. Ryšiuose naudojamos rodyklės žymi paveldimumą. Rodyklės visada nuo vaiko klasės į tėvinę klasę. Klasių ryšiuose galima nurodyti kaip ir kodėl klasės yra susijusios.

3.3.2 Objektų diagrama

Objektų diagrama parodo objektų egzistavimą ir jų ryšius loginiame sistemos vaizde, o taip pat ir veiksmų ar panaudojimo atvejų vykdymo scenarijų (18 pav.).

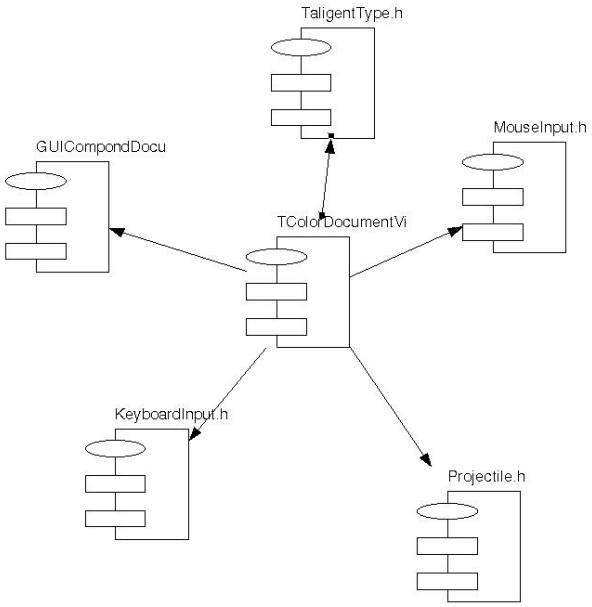


18 pav. Boocho objektų diagrama

Scenarijuje nustatoma kiekviena sistemos vykdoma užduotis, kuri vėliau priskiriama geriausiai tam tinkančiai klasei.

3.3.3 Modulių diagrama

Modulių diagramoje (19 pav.) vaizduojamas klasių ir objektų paskirstymas į modulius fiziniame sistemos vaizde.

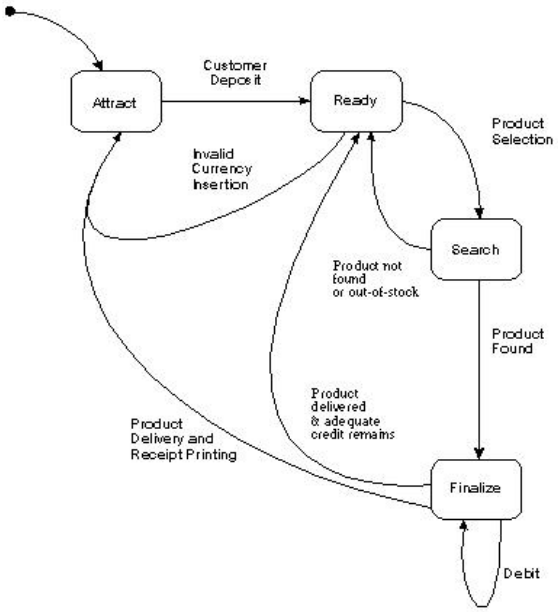


19 pav. Boocho modulių diagrama

19 pav. Pateiktoje modulių diagramoje pavaizduota, kaip viena klasė („TColorDocumentView“) susijusi su kitomis klasėmis.

3.3.4 Būsenų diagrama

Būsenų diagramos (20 pav.) vaizduoja klasių dinaminę elgseną. Diagrama naudojama parodyti duotos klasės visas galimas būsenas, įvykius, kurie iššaukia klasės perėjima iš vienos būsenos į kitą, ir veiksmus, kylančius po būsenos pakeitimo.

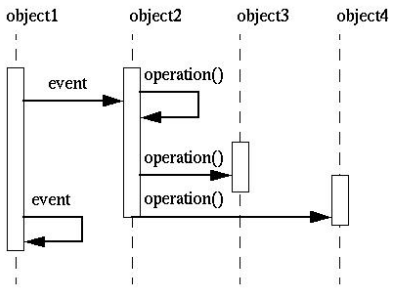


20 pav. Boocho būsenų diagrama

Pateiktoje būsenų diagramoje (20 pav.) matome, kad vaizduojama sistema pradeda dirbti ir iš karto patenka į būseną „attract“. Jei vartotojas pateikia pinigus, sistemos būsena pereina į „ready“. Iš šios būsenos galima išeiti dviem atvejais. Jei vartotojas pasirenka produktą, pereinama į „search“ būseną. Jei vartotojo pateikti pinigai neatpažįstami, sistema grįžta į būseną „attract“.

3.3.5 Sąveikų diagrama

Sąveikų diagramos vaizduoja tą patį ką ir objektų diagramos, tačiau kitokiu būdu (21 pav.).



21 pav. Boocho sąveikų diagrama

Čia ir vėl vaizduojamas vartotojo scenarijus (panaudojimo atvejis), realizuojamas per tam tikrus objektus (klases). Kartu matome ir klasių tarpusavio sąveiką per jų atliekamus veiksmus.

3.3.6 Privalumai ir trūkumai

Privalumai:

- Užbaigus klasių diagramą, gaunamas ne tik analizės modelis, bet ir klasių priklausomumo modelis.
- Modelių ir naudojamų diagramų „persipynimas“ padeda pažvelgti į sistemą iš kelių perspektyvų, o tai gali padėti susidaryti geresnį projektuojamos sistemos vaizdą;

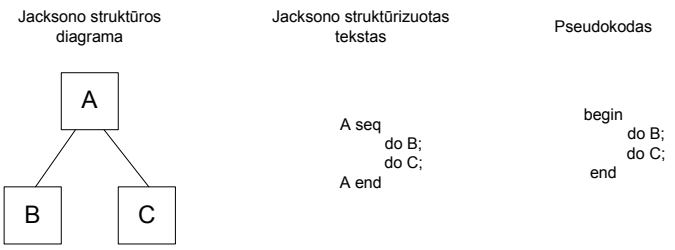
Trūkumai:

- Nė vienam iš keturių Boocho modelių nėra nustatytų baigtumo kriterijų bei nėra apibrėžta, kiek iteracijų bus reikalinga atlikti. Trūksta aiškiai įvardintų kiekvieno modelio nagrinėjimo pradžios ir pabaigos taškų.
- Nors Boocho metode yra naudojama keletas sistemos atvaizdavimo modelių, tačiau trūksta taisyklių juos sujungti į bendrą visumą.

3.4 Duomenų struktūromis besiremiantis JSP metodas

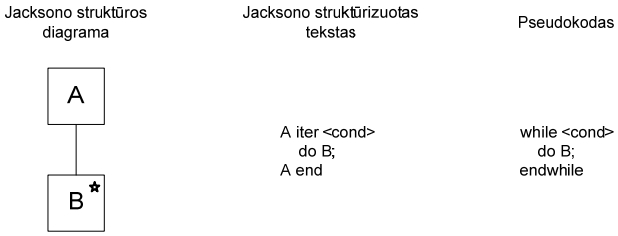
JSP metodas skirtas struktūrizuotam programavimui, jis remiasi atitikimu tarp duomenų struktūrų ir programos struktūros. JSP sustruktūrizuoja programas ir duomenis sekomis (22 pav), iteracijomis (23 pav.) ir atrinkimais (sąlygomis) (24 pav.). Projektuojama programa privalo apdoroti vieną ar daugiau nuoseklių duomenų srautų [31].

Seka – sudėtinis komponentas, turintys dvi ar daugiau dalių, atsirandančių tam tikra tvarka (22 pav.):



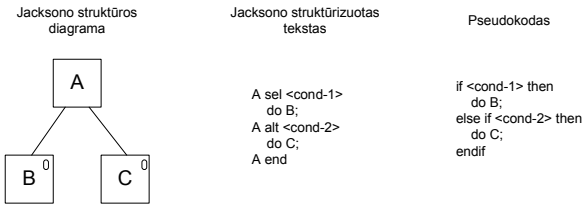
22 pav. Sekos struktūros diagrama

Iteracija (23 pav.) – sudėtinis komponentas, susidedantis iš vienos pasikartojančios dalies. Ši dalis gali ir nesikartoti.



23 pav. Iteracijos struktūros diagrama

Atrinkimas (sąlyga) (24 pav.) – sudėtinis komponentas, susidedantis iš dviejų ar daugiau dalių, iš kurių tik viena dalis parenkama vieną kartą.



24 pav. Atrinkimo (sąlygos) struktūros diagrama

JSP metodas susideda iš šių žingsnių [30]:

- Nubraižoma sistemos diagrama. Šis žingsnis gali būti praleidžiamas, kai sistemos diagrama yra savaime aiški;
- Nubraižomos duomenų struktūros programos įėjimas ir išėjimas;

- Suformuojama programos struktūra, atsižvelgiant į praėjusiame žingsnyje gautas duomenų struktūras;
- Pateikiamas programos struktūros operacijų sąrašas;
- Parengiama detali programos struktūra, paprastą programos struktūrą papildant operacijomis ir sąlygomis;
- Gauta struktūros diagrama paverčiama į struktūrinį tekstą ar programos kodą.

N. Ourusoffas pateikia puikų šių žingsnių atlikimo pavyzdį [32] – sugeneruojama ir atspausdinama daugybos lentelė:

1

2 4

3 6 9

4 8 12 16

...

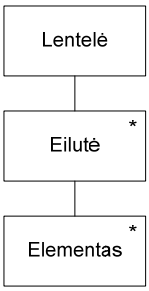
10 20 30 40 50 60 70 80 90 100

Šiuo atveju vykdomi žingsniai:

- Nubraižoma sistemos diagrama (25 pav.):

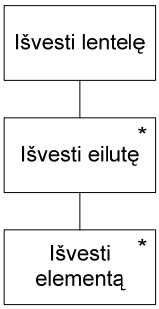
25 pav. Sistemos diagrama

- Nubraižomos duomenų struktūros (26 pav.):



26 pav. Duomenų struktūros diagrama

- Suformuojama programos struktūra (27 pav.):



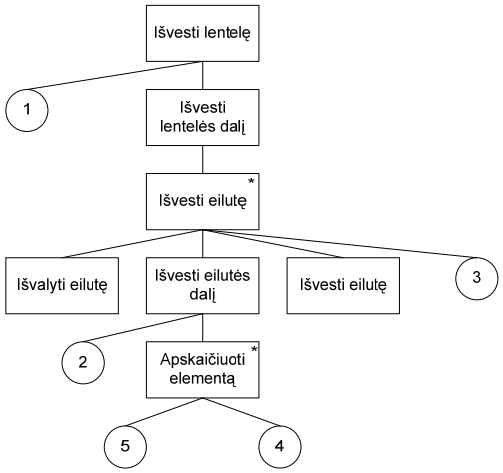
27 pav. Programos struktūros diagrama

- Pateikiamas operacijų sąrašas (3 lent.):

3 lent. Operacijų sąrašas

Operacija	Kaip dažnai?	Kur?
1. eil-nr=1	Vieną kartą	Programos pradžioje
2. stulp-nr=1	Vieną kartą eilutėje	Kur išvedama eilutė, pradžioje
3. eil-nr=eil-nr+1	Devynis kartus	Kur išvedama eilutė
4. stulp-nr=stulp-nr+1	(eil-nr)-1 kartų eilutėje	Kur apskaičiuojamas elementas
5. eilute[stulp-nr]=eil-nr*stulp-nr	Vieną kartą per elementą	Kur apskaičiuojamas elementas

- Parengiama detali programos struktūra (28 pav.):



28 pav. Detali programos struktūra

- Realizuojama programa

3.4.1 JSP privalumai ir trūkumai

Privalumai:

- JSP metodas teikia sistematinį ir nurodantį metodą, dėl kurio nepriklausomi JSP projektuotojai, spręsdami tas pačias problemas, pateikia tuos pačius sprendimus [33].
- Projektuotojas, naudojantis šį metodą, susiduria tik su statinėmis struktūromis. Tai leidžia išvengti „dinaminio“ mąstymo – programos vykdymo apgalvojimo, kas neretai yra klaidų šaltinis.

Trūkumai:

- JSP – individualių programų projektavimo metodas, o ne sistemų.

4 Tirtų metodų galimas teorinis panaudojimas projektuojant „E-Portfolio“ sistemą

Šiame skyriuje teoriškai nagrinėjamas aptartų projektavimo metodų tinkamumas projektuoti „E-Portfolio“ sistemą. Darant šią apžvalgą bus atsižvelgiama į šio darbo autoriaus patirtį projektuojant „E-Portfolio“ sistemą „ePortfolio“.

4.1 JSP tinkamumas

„E-Portfolio“ sistema – tai internetinė sistema, susijusi su mokymo IS sistemomis, naudojama skirtingų vartotojų ne tik sistemos atžvilgiu – studentas-dėstytojas-administratorius, bet ir pomėgių, įpročių bei įgūdžių atžvilgiu. Ji turi būti ne tik patogi naudoti, bet ir suasmeninama, leidžianti bendrauti sistemos vartotojams (socialinė sistema). Tad neišvengiamai tokiai sistemai tenka apdoroti įvedamos ir išvedamos informacijos kiekius, saugoti vartotojų įkeliamus failus ir pan. Vien dėl apdorojamos informacijos skirtingumo bei kiekio ir sistemos dinamiškumo bei 4.4.1 skyriuje „JSP privalumai ir trūkumai“ išvardintų trūkumų JSP metodas netinkamas projektuojant „E-Portfolio“ sistemas.

4.2 SSADM tinkamumas

Kadangi sistemoje apdorojama ir saugojama informacija, atrodytų, jog SSADM metodas puikiai tiktų „E-Portfolio“ projektavimui. Ir iš tiesų, tokioje sistemoje, kurios veikimas remiasi informacijos apdorojimu, tikrai pravartu turėti duomenų srautų modelį, pavaizduotą DFD diagramomis. Iš jų matosi kurioje sistemos dalyje kokia informacija įvedama, kokiose funkcijose ji apdorojama, kur saugojama ir kuriose sistemos vietose išeina. Kadangi nagrinėjama sistema – internetinė, iš principo jos veikimas remiasi vien užklausų DB generavimu ir gautų rezultatų atvaizdavimu naršyklės lange. Todėl šiuo atveju DFD detalizacijos lygis būtų žemas, be to, iš DFD diagramų nesimato funkcinių transformacijų atlikimo būdo.

Projektuojant „E-Portfolio“ sistemą, geriausia būtų panaudoti SSADM metodo logines duomenų struktūras. Kadangi jos naudojamos atvaizduoti sistemoje esančias duomenų struktūras per esybių ir jų tarpusavio ryšių diagramas, tai padėtų greitai susidaryti bendrą sistemos duomenų vaizdą. Jau iš pačios pradinės aukščiausio lygio loginės duomenų struktūros (4.2.2. skyrius, 16 pav.)

pakankamai gerai galima pamatyti, kokiais duomenimis bus manipuliuojama projektuojamoje sistemoje. O pasiekus reikiamą tokių diagramų detalizacijos lygį galima lengvai gauti duomenų bazės modelį.

Kadangi „E-Portfolio“ sistemos smarkiai orientuotos į vartotojų poreikius, SSADM privalumas tas, jog šis metodas skiria didelį dėmesį vartotojų poreikių analizei. Tuo pačiu metu sukurtas sistemos modelis jau iš karto yra tam tikros kokybės, kuri vėliau tik gerinama. Įvairių modeliavimo bei diagramų kūrimo įrankių naudojimas paspartina sistemos modeliavimo procesą, o SSADM paprastumas bei nereikalavimas projektuotojui turėti specifinių įgūdžių gali sumažinti projektavimo klaidų kiekį. Tad, jei projektuojant „E-Portfolio“ sistemą, būtų siūloma naudoti SSADM metodą, šio darbo autorius siūlo pasinaudoti loginėmis duomenų struktūromis ir vartotojų poreikiams kreipiamu dėmesiu, nesigilinant į DFD diagramas.

4.3 Boocho metodo tinkamumas

Taikant Boocho metodą, projektuojant parengiama daug skirtingo tipo sistemos diagramų (vaizdų), o metodo modelių ir diagramų „persipynimas“ leidžia sistemą įvertini iš kelių skirtingų pozicijų – susidaryti geresnį projektuojamos sistemos vaizdą.

Klasių diagramų naudojimas padeda išskirti sistemos klases ir jų tarpusavio ryšius, bei pačių klasių struktūras. Iš klasių diagramos jau matosi projektuojamos sistemos architektūra, o tai vėliau padeda jau programuojant pačią sistemą. Be to, klasių diagramose nebūtina vaizduoti visą sistemos klasių struktūrą – užtenka ir jos dalies. Boocho metode naudojamos objektų diagramos yra dalis objektiškai orientuoto projektavimo notacijos, naudojamos parodyti visiems ar tik daliai objektų ir jų tarpusavio ryšių sistemos loginiame modelyje. Tokios diagramos naudojamos panaudojimo atvejų vykdymo scenarijų parodymui. Taigi, viena objektų diagrama vaizduoja sistemos būseną tam tikru laiko momentu. Modulių diagramos naudojamos parodyti sistemos architektūros išskaidymą. Be to, susijusius modulius galima grupuoti į posistemes, kurios savo ruožtu taip pat gali būti priskiriamos kitoms posistemėms. Kaip ir moduliai, posistemės gali būti priklausomos nuo kitų posistemių ar modulių.

Būsenų diagramos naudojamos parodyti nagrinėjamos klasės visoms galimoms būsenoms, įvykimas, iššaukiantiems perėjimus tarp būsenų ir veiksmams, kylantiems po būsenos

pasikeitimo. Sąveikų diagramos leidžia sekti panaudojimo atvejų vykdymą, tad jos – tarsi kitoks objektų diagramų pateikimas. Kadangi sąveikų diagramos perteikia perduodamus pranešimus pagal jų atsiradimo tvarką, jos tarsi papildo atitinkamas objektų diagramas. Sąveikų diagramos naudojamos atkreipti dėmesį į įvykius, o ne operacijas, tad jos geriau perteikia panaudojimo atvejų scenarijų semantiką nei objektų diagramos.

Taikant Boocho metodą, reikia atsižvelgti į tai, jog sąveikų diagramos persidengia su objektų diagramomis, tad mažinant projektavimo trukmę šio darbo autorius rekomenduoja atsisakyti objektų diagramų, kadangi klasių (objektų) struktūra matoma klasių diagramose, o sąveikų diagramos yra lengviau suprantamos. Taip pat, jeigu sistema nėra sudėtina ar didelė, vertėtų atsisakyti modulių diagramos – ji teiktų tik perteklinę informaciją.

5 Išvados

Analitinėje dalyje buvo apibūdinta el. portfelio sąvoka. Išnagrinėtos trys skirtingos „E-Portfolio“ sistemos bei nustatyti tokioms sistemos keliami reikalavimai. Pasirinktas „ePortfelis“ projektavimo būdas.

Realizacijos dalyje pateikti funkciniai ir nefunkciniai reikalavimai, reikalavimai sistemos diegimo aplinkai bei pačios sistemos architektūra. „ePortfelis“ sistemą realizuoti nuspręsta PHP programavimo kalba, atsižvelgus į jos diegimo aplinką.

Tyrimo dalyje trumpai apžvelgti pagrindiniai PĮ projektavimo metodų tipai ir nurodyti nagrinėjami metodai (SSADM, JSP ir Boocho). Šie metodai toliau buvo detaliai aprašyti, išskirti jų privalumai ir trūkumai. Atlikus metodų tyrimą, buvo analizuojamas jų tinkamumas „E-Portfolio“ sistemų projektavimui. Analizės metu nustatyta, kad netinkamas tokioms sistemoms – JSP metodas. Rekomenduojama taikyti SSADM metodą (be DFD diagramų) kartu su JSP metodu (be objektų ir modulių diagramų).

6 Terminų žodynas

OOP – Objektiškai Orientuotas Projektavimas

SSADM („Structured Systems Analysis and Design Method“) – struktūrizuotas sistemų analizės ir projektavimo metodas

DFD („Data Flow Diagram“) – duomenų srautų diagrama

LDS („Logical Data Structure“) – loginė duomenų struktūra

JSP („Jackson Structured Programming“) – duomenų struktūromis besiremiantis projektavimo metodas

DB – duomenų bazė

HTML („Hypertext Markup Language“) – kompiuterinė žymėjimo kalba, naudojama pateikti turinį internete

WYSIWYG („What You See Is What You Get“) (liet. „tai ką matote atitinka tai ką gausite“) – akronimas, naudojamas kalbant apie kompiuterines programas, norint apibūdinti programinę įrangą, kurios turinio išvaizda yra labai panaši į tos PĮ pateikiamo galutinio produkto išvaizdą. Vartojant teksto rengyklės atveju ši sąvoka reiškia, kad redaguojamo teksto išvaizda ekrane bus identiška ant popieriaus atspausdinto teksto išvaizdai

PDF („Portable Document Format“) – atviro failo formatas, skirtas neutraliam dvimačio dokumento atvaizdavimui

7 Literatūros sąrašas

[1] P. Abrahamsson, O. Salo, J. Ronkainen, J. Warsta. „Agile software development methods. Review and analysis“. 2002

[2] P. Sharma. „An Introduction to Extreme Programming“, http://my.advisor.com/doc/13571c (žiūrėta 2008-05-17)

[3] K. Beck. „Embracing Change With Extreme Programming“. IEEE Computer 32(10): 70-77 psl.

[4] L. C. de Rossi, “Electronic Portfolios: What Are They?”, 2006, http://www.masternewmedia.org/news/2006/03/10/electronic_portfolios_what_are_they.htm, žiūrėta 2006-11-07

[5] J. Morris, H. Buckland, “Electronic portfolios for learning and assessment”, http://www.uvm.edu/~jmorris/site2000electport.html

[6] R. V. Dragan, “Blackboard”, http://www.pcmag.com/article2/0,1895,17187,00.asp, 2001, žiūrėta 2006-10-25

[7] M. Elorriaga, „Blackboard 5.0“, eLearn Magazine, http://www.elearnmag.org/subpage.cfm?section=reviews&article=2-1, žiūrėta 2006-10-28

[8] A. Cartwright, „Blackboard Experience of a Chemistry Teacher“, 2000, http://jchemed.chem.wisc.edu/Journal/Issues/2000/Jun/abs699.html, žiūrėta 2006-10-26

[9] C. C. Chou, “Technology Integration with Standards-Based eFolio for K-12 In-Service Teachers”, 2004, http://efoliomn.com/index.asp?Type=NONE&SEC=%7B0D623B29-EBD3-42B6-AEBC-A4EA8FD87548%7D, žiūrėta 2006-10-27

[10] B. Byfield, “Review: The Open Source Portfolio Initiative”, 2006, http://internet.newsforge.com/article.pl?sid=06/02/01/1527252&tid=13, žiūrėta 2006-10-28

[11] H. C. Barrett, „My Online Portfolio Adventure“, 2006, http://electronicportfolios.org/myportfolio/versions.html, žiūrėta 2006-11-08

[12] K. Sumner-Smith, “Proposed System Features and Program Map”, http://elgg.net/karinas/files/, 2006

[13] Europos Parlamento ir Tarybos Sprendimas Nr. 2241/2004/EB „Dėl bendros Bendrijos sistemos siekiant užtikrinti kvalifikacijų ir gebėjimų skaidrumą (Europass)“, 2004, http://europass.cedefop.europa.eu/img/dynamic/c1399/type.FileContent.file/EuropassDecision_lt_LT.PDF

[14] N. Currant, C. Murray, C. Higgison, J. Taylor, A. Pellow, S. Hennessy, S. Raby, J. Hairsine ir R. Sykes, „Implementing e-Portfolios: Technical and Organisational Issues“, 2006, http://www.lancs.ac.uk/postgrad/ramirezm/e-portfolio/02Currant2.pdf, žiūrėta 2006-11-08

[15] G. Lorenzo, J. Ittelson, “An Overview of E-Portfolios”, 2005, http://www.educause.edu/ir/library/pdf/ELI3002.pdf

[16] „LDP e-Portfolio Report“, http://bearlink.berkeley.edu/ePortfolio/index.html, 2004, žiūrėta 2006-11-13

[17] S. Cotterill, T. McDonald, P. Drummond, G. Hammond, „Design, implementation and evaluation of a „generic“ e-portfolio: the Newcastle experience“, 2004, http://www.eportfolios.ac.uk/FDTL4?pid=49, žiūrėta 2006-11-13

[18] R. Butleris. Reikalavimo specifikavimo oracle case terpėje plėtra, http://www.leidykla.vu.lt/inetleid/inf-mok/19/str6.html, žiūrėta 2008-03-30

[19] K. E. Wiegers, „Read My Lips: No New Models!“, IEEE Software September/October, 1998, 10-13 psl.

[20] H.V. Trung, „Software Desing“, http://cnx.org/content/m14630/latest/, žiūrėta 2008-05-24

[21] I. Sommerville, „Software Engineering. 6th Edition“, 2000

[22] C. Szyperski, „Component Software: Beyond Object-Oriented Programming. 2nd ed.“, 2002

[23] „Structured Systems Analysis and Design Methodology“, http://www.nasscom.org/download/SSAD.pdf, žiūrėta 2008-05-23

[24] SSADM, http://www.webopedia.com/TERM/S/SSADM.html, žiūrėta 2008-05-24

[25] „Structured Systems Analysis and Design Method“, http://en.wikipedia.org/wiki/Structured_Systems_Analysis_and_Design_Method, žiūrėta 2008-05-25

[26] R. Butkienė, R. Butleris, „Taisyklės kompiuterizuotos informacinės sistemos modelio išsamumui ir minimalumui patikrinti“, „Informacijos Mokslai“ 21, 2002, http://www.leidykla.vu.lt/inetleid/inf-mok/21/str10.html, žiūrėta 2008-05-25

[27] S. J. Mellor, „A Comparison of the Booch Method and Shlaer-Mellor OOA/RD“, 1993

[28] B. Brooks, „Automating Aspects Object-Oriented Design using Rational Rose“, 1996, http://users.csc.calpoly.edu/~dbutler/tutorials/winter96/rose/cscpaper.html, žiūrėta 2008-05-25

[29] P. Schneider, „The Booch Method“, http://salmosa.kaist.ac.kr/BoochReferenz/index.html, žiūrėta 2008-05-24

[30] N. Ourusoff, „Using Jackson Structured Programming (JSP) and Jackson Workbench to Teach Program Design“, Informing Science Proceedings, 2003, http://proceedings.informingscience.org/IS2003Proceedings/docs/091Ourus.pdf

[31] M. Jackson, „The Jackson Development Methods“, Wiley Encyclopaedia of Software Engineering, 1992, http://www.jacksonworkbench.co.uk/stevefergspages/papers/jackson--the_jackson_development_methods.pdf

[32] N. Ourusoff, „Introduction to Jackson Design Method: JSP and a little JSD“, 2003, http://www.jacksonworkbench.co.uk/stevefergspages/papers/ourusoff--introduction_to_jackson_design_method.pdf

[33] M. Jackson, „JSP In Perspective“, 2001, http://www.jacksonworkbench.co.uk/resources/JSP%20in%20Perspective%20(MJ%202001).pdf