KAUNO TECHNOLOGIJOS UNIVERSITETAS
Informatikos fakultetas
Informacijos sistemų katedra

Vita Misevičiūtė

Medicininių duomenų informacijos sistemos, naudojančios objektines technologijas, tyrimas

Magistro darbas

Vadovas
prof. habil. dr.
Arūnas Lukoševičius

Kaunas, 2006

KAUNO TECHNOLOGIJOS UNIVERSITETAS
Informatikos fakultetas
Informacijos sistemų katedra

TVIRTINU
Katedros vedėjas
doc. Rimantas Butleris
2006-01-

Medicininių duomenų informacijos sistemos, naudojančios objektines technologijas, tyrimas

Informatikos inžinerijos mokslo magistro baigiamasis darbas

Vadovas
prof. habil. dr.
A. Lukoševičius
2006-01-

Recenzentas
doc. E. Karčiauskas
2006-01-

Atliko
IFM-0/4 gr. stud.
V. Misevičiūtė
2006-01-

Kaunas, 2006

Turinys

Paveikslai

Lentelės

Misevičiūtė V. Research of medical data information system, which uses object-oriented technologies: Master's work in science of informatics/ supervisor Prof. habil. dr. A. Lukoševičius; Department of Information System Engineering, Faculty of Informatics, Kaunas University of Technology. – Kaunas, 2006. = 84 p

Summary

Information systems for medical data storing, managing and retrieving are specific, because of the data that has different in-between associations. This data consists of text and multimedia. The purpose of this work is to propose and develop a medical data system, which uses object oriented data base. The user could store various medical data and perform data mining by selecting desirable criterions. After analysis of object-oriented technologies, were chosen object-oriented programming environment and object-oriented data base management system. For its implementation, the suitable technologies were analyzed, according to the specifics of medical data. The advantages of these technologies where shown by an experiment. Developed information system is functional and flexible enough to work with evolving medical data.

Santrauka

Informacijos sistemos (IS), skirtos medicininių duomenų kaupimui, saugojimui ir išgavimui, yra specifinės tarpusavio ryšiais susietų duomenų požiūriu. Šie duomenys susideda ne tik iš tekstinės, bet ir iš vaizdinės informacijos. Darbo tikslas – suprojektuoti ir realizuoti medicininės paskirties IS, naudojančią objektinę duomenų bazę, kurioje vartotojas galėtų įkelti įvairiarūšius medicininis duomenis ir juose atlikti paiešką pagal skirtingus kriterijus. Šiam tikslui įgyvendinti ieškota sprendimų, atitinkančių medicininių duomenų specifiką. Taigi, atliekant analizę su šiuo metu intensyviai besivystančiomis objektinėmis technologijomis, pasirinkta objektinio programavimo aplinka ir objektinė duomenų bazių valdymo sistema. Minėtų technologijų privalumai parodyti eksperimentiškai. Objektinėmis technologijomis sukurta IS pasižymi funkcionalumu bei lankstumu, kurio reikia kintantiems medicininiams duomenims saugoti.

1. Įvadas

Medicininės paskirties informacijos sistemos (IS) yra viena iš pastaruoju metu intensyviai besiplėtojančios e. sveikatos sistemos komponentų. Pagrindiniai šiuolaikinių medicininių IS skiriamieji bruožai yra: 1) saugomi ir kaupiami duomenys yra įvairialypiai (tekstinė informacija, statiniai ir judantys vaizdai, signalai, ir pan.); 2) informacija yra surišta sudėtingais semantiniais ryšiais, kurie, kaupiantis žinioms, neišvengiamai evoliucionuoja; 3) informacijos kiekiai yra dideli, nes apima visą gyvenimą trunkančius pacientų sveikatos įrašus (sveikatos paslaugų epizodų metu gautus tyrimų duomenis, atliktas diagnostikos ir terapijos procedūras, demografijos ir gyvensenos parametrus, skirtus vaistus ir kt.); 4) vartotojų ratas (pavyzdžiui gydytojai) yra plačiai pasiskirstęs per visus tris sveikatos paslaugų lygius, todėl reikalingos saugios ir patogios kompiuterinio tinklo prieigos visapusiam ryšiui su IS; 5) IS turi būti suderintos su Europos Sąjungos ir tarptautiniais standartais, reglamentuojančiais sveikatos paslaugų tęstinumą laike ir tarp institucijų, informacijos apsikeitimą, jos apsaugą, elektroninę sveikatos istoriją (ESI), kodifikavimą bei terminologiją; 6) duomenys turi tarnauti klinikinių sprendimų palaikymui, t.y. paslaugų teikimo vietoje gydytojas turi gauti daugeliu prieš tai atliktų tyrimų bei gydymo rezultatais pagrįstus patarimus naudingus klinikinėje praktikoje.

Tai kelia gana specifinius reikalavimus IS, jų architektūrai ir verčia ieškoti naujų įgyvendinimo instrumentų. Šiuo metu pasaulyje yra sukurta ir kuriami vis nauji standartai optimizuojantys ir palengvinantys medicininių vaizdų saugojimą. Tačiau jų apdorojimas ir prieinamumas išlieka problema, kadangi ilgą laiką naudotos technologijos nėra pakankamai lanksčios tokiems duomenims saugoti ir apdoroti. Vaizdai saugomi reliacinėse duomenų bazėse (RDB), nors objektinės duomenų bazės (ODB) įrodo turinčios vis daugiau privalumų [1]. Šios duomenų bazės (DB) ypatingos tuo, kad duomenys jose saugomi objektais, kurie savyje gali turėti kitus objektus, sukurdami hierarchiją. Šios objektinės struktūros užpildomos ne tik konkrečiais duomenimis, bet ir objektais. Tokią technologiją labai patogu pritaikyti medicininių diagnostinių vaizdų saugojimui. 2005 metais sukurtas nekomercinis tinklalapis (www.ODBMS.org) skirtas objektinių duomenų bazių technologijoms. Tai pabrėžia objektinių duomenų bazių svarbą ir raidą pastaraisiais metais. Be to pasaulio universitetuose yra skaitomos paskaitos apie

ODB. Tai taip pat išreiškia, kad ODB yra svarbios kaip ir RDB. Tačiau ODB nėra RDB pakeitimas, tai tik papildymas.

Saugios ir lanksčios internetinės prieigos reikalavimas taip pat verčia ieškoti efektyvių programavimo kompiuteriniuose tinkluose technologijų. To siekiama naudojantis taip pat nauja technologija – .NET.

Darbo tikslas ir uždaviniai. *Darbo tikslas* – suprojektuoti ir realizuoti medicininės paskirties IS, naudojančią objektinę duomenų bazę, kurioje vartotojas galėtų įkelti įvairiarūšius, kintančius, tarpusavio ryšiais susietus medicininis duomenis ir atlikti jų paiešką.

Darbo uždaviniai: Palyginti ODB su RDB specifinių reikalavimų medicininiams duomenims ir standartams požiūriu. Atlikti J2EE ir .NET objektinių programavimo technologijų palyginamąją analizę. Suprojektuoti ir realizuoti IS naudojant pasirinktą technologiją ir ODBVS. Eksperimentiškai parodyti ODB privalumus, realizuotam uždaviniui, įgyvendinant naujai atsiradusių medicininių duomenų įkėlimą ir atnaujinimą.

Darbo originalumas.

Šiuo metu medicininių (oftalmologinių) duomenų saugojimas yra nestruktūrizuotas. Vaizdai saugomi keliose reliacinėse duomenų bazėse. O juos lydinti informacija – kituose resursuose. Paieška duomenų bazėse atliekama rankiniu būdu. Darbo metu sukurta objektinė duomenų bazė, pritaikytos internetinės technologijos palengvinančios vartotojų darbą su šiais duomenimis, parodyti ODB privalumai šiems duomenims saugoti.

2. Objektinės technologijos ir duomenų saugojimo standartai

2.1. Objektinių duomenų bazių analizė

Programinės įrangos krizės išsprendimui nepakanka vien naujų metodų programų kūrimui. Reikalinga nauja informacijos valdymo technologija. Šiandien informacinės sistemos turi apdoroti ne tik tekstus, bet ir skaičiavimo lenteles (spreadsheets), vaizdus, diagramas, žemėlapius, garso įrašus, kitokią daugialypės terpės informaciją. Be to, visa ši informacija turi būti organizuota kaip vieninga struktūra, patogi vartotojui.

Objektinė technologija potencialiai pajėgi užtikrinti naują informacijos saugojimo ir paieškos technologiją, suteikti informacijos valdymui naujų galimybių.

Išskiriamos trys objektinės technologijos, panaudojimo duomenų bazių valdymui, sritys:

- objektinių programų palaikymas,
- kompleksinės informacijos saugojimas,
- intelektualių DB sudarymas.

Dauguma objektinių programų sukuria objektus ir turi juos saugoti tam, kad galėtų juos pakartotinai panaudoti kito vykdymo metu. Pavyzdžiui, Smalltalk sukurtas ir naudojamas originalus būdas - saugoma visa sistemos būsena specialioje byloje (image file): visi sukurti objektai, ekrano informacija, kita valdanti informacija. Pakartotinai vykdant programą, Smalltalk automatiškai nuskaito „image file“ ir tęsia darbą nuo šios būsenos. Šis būdas tinka tik individualiam darbui su viena programa. Jei kelios programos naudoja vieną objektą, toks būdas netinka. Geresnis būdas yra saugoti objektus bylose. Taip atsiranda galimybė bendrai naudoti tuos pačius objektus. Tačiau kai du vartotojai vienu metu kreipiasi į tą patį objektą, sistema nepajėgi patikrinti, kas su kuriuo objektu dirba. Taigi, bylos neturi reikiamo apsaugos mechanizmo ir negalima kontroliuoti į juos siunčiamų kreipinių.

Akivaizdu, kad objektai turi būti saugomi DBVS (duomenų bazių valdymo sistemose). Tačiau šiuo metu paplitusios įprastinės DBVS, pirmiausia pritaikytos tekstinių ir skaitmeninių duomenų saugojimui. Įprastinės DBVS nepajėgios susitvarkyti su neapibrėžtais iš anksto duomenų tipais, kurie susiformuoja objektinėse kalbose. Be to, šios DBVS visiškai nepritaikytos objektų sudėtinėms dalims – metodams saugoti.

Vienas iš šios problemos sprendimų yra sukurti papildomą sluoksnį virš įprastinės DBVS, kuris suskaidytų objektus į paprastesnes komponentes, kurias pajėgtų saugoti

įprastinės DBVS. Tinklinėse ir hierarchinėse DBVS galima būtų realizuoti šį būdą, nes jos pritaikytos sudėtingų struktūrų saugojimui. Tačiau lieka neaišku, kaip saugoti paveldėjimo taisykles. Reliacinės DB yra pakankamai lanksčios, tačiau jų duomenų struktūros paprastos, nepajėgios natūraliai atvaizduoti objektus [2]. Geriausias sprendimas – objektams saugoti skirtos DB, vadinamos objektinėmis DBVS (ODBVS).

ODBVS užtikrina visą įprastinį servisą, objektai saugomi tiesiogiai, jų nekonvertuojant. Objektinė metodologija reikalauja, kad būtų užtikrinti tokie bruožai:

- Apgaubimas (encapsulation) – galimybė sujungti duomenų struktūras ir jų apribojimo metodus į vieną struktūrinį metodą;
- Paveldėjimas (inheritance) – objekto sugebėjimas paveldėti protėvių objekto atributus ir metodus;
- Objekto identitetas (object identity) – unikalus objekto identifikavimas, tai turi nepriklausyti nuo objekto vietos (atmintyje) ar savybių (kadangi jos gali keistis)

Objektinių duomenų bazių standartus tvarko ODMG (Object Database Management Group) [3]. ODMG remiasi tokiais objektiniais modeliais (specifikacijomis):

- OM – Object Model – objektų modeliu (užtikrina ODB funkcionalumą);
- ODL – Object Definition Language – objektų aprašymo kalba (aprašo ODB schemas);
- OQL – Object Query Language – objektų užklausos kalba (SQL tipo kalba objektų paieškai);
- LB – Language Binding – kalbų sąsajos specifikacija su C++, JAVA ir Smalltalk (palaiko objektines užklausų kalbas).

Objektinės DB saugo kompleksinę informaciją ir nepraranda reliacinėms DB būdingo lankstumo. ODBVS saugo ir sudėtingas struktūras – sudėtinius objektus (composite objects), kurie susideda iš kitų objektų. Sudėtiniai objektai saugomi sudarant adresines nuorodas (references) į žemesnio lygio objektus, lygių skaičius gali būti labai didelis. Kiekvienas objektas gali būti įtrauktas į daugelį kitų objektų kaip komponentė. Taip susiformuoja struktūra, kuri būdinga ir tinkliniam duomenų modeliui. ODB gali palaikyti eilę alternatyvių struktūrų, sudarytų iš tų pačių duomenų elementų. Skirtingai negu RDB, šios alternatyvios struktūros nėra tik "langai" (views). Visos šios struktūros yra vienodai traktuojamos, visiškai nepriklausomos viena nuo kitos. Galima kurti naujas struktūras ir

modifikuoti senąsias. Jei naujoji objektų struktūra turi daugiau laukų negu senoji, senųjų objektų skaitymo metu, naujieji objektai užpildomi iš anksto nustatytomis reikšmėmis. Priešingu atveju, jeigu naujoje struktūroje yra mažiau objektų, skaitymo metu neegzistuojantys laukai praleidžiami [4].

Taigi, ODBVS užtikrina plačias struktūrizavimo galimybes, tuo jos panašios į tinklines DB, bet neturi pastarųjų trūkumų.

Kaip ir įprastose DB, objektinėse DB yra vidinių duomenų tipų. Tačiau galima sukurti naujus reikalingus konkrečiu atveju duomenų tipus, jungti juos į hierarchines struktūras. Tokiu būdu sukuriamos objektinės DB, atitinkančios vartotojo poreikius.

Naujų duomenų tipų sudarymo galimybė ypatingai reikalinga ODB savybė, kuri įgalina saugoti daugialypės terpės informaciją: paveikslus, nuotraukas, vaizdo įrašus, kalbos įrašus, muziką ir kitokių tipų informaciją.

Paveldėjimo mechanizmas suteikia ODB dar vieną papildomą lankstumo lygmenį. Tradicinėse DB visų vieno tipo esybių struktūra vienoda. Jei naujiems modeliuojamos probleminės srities elementams reikia kitokios struktūros, yra dvi galimybės:

- sutalpinti naująsias esybes į senąsias struktūras;
- apibrėžti naują struktūrą, kuri tenkintų visus reikalavimus ir konvertuoti visas reikalingas esybes į naująją struktūrą.

ODB suteikia naują galimybę – sukurti poklasę, kuri aprašo modeliuojamą specialų atvejį, saugodama papildomą informaciją.

ODB atlieka sudėtingų struktūrų paiešką greičiau už tradicines DB, nes struktūros saugomos kaip tiesioginės nuorodos tarp objektų [2].

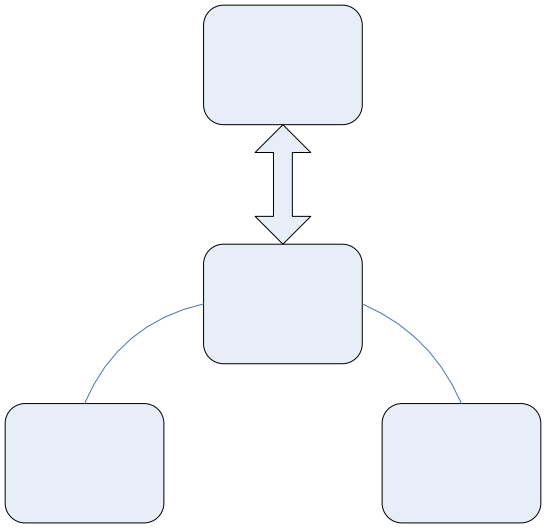
Skirtingi duomenų rinkiniai dažnai siejami daugelis-su-daugeliu ryšiu. Tokie ryšiai modeliuojami RDBVS reikalauja tarpinės lentelės. Tuo tarpu, dauguma ODBVS duomenų rinkinius atskiria į skirtingus objektus (veikiausiai „einant“ gilyn) ir leidžia išrinkti ir įkelti objektus viena užklausa [4].

ODB skirstomos į dvi kategorijas:

- pasyvias ODB, kurios saugo objektų struktūrą, bet nerealizuoja objektų elgsenos,
- aktyvias ODB, kurios leidžia objektams sąveikauti tiesiog duomenų bazėje.

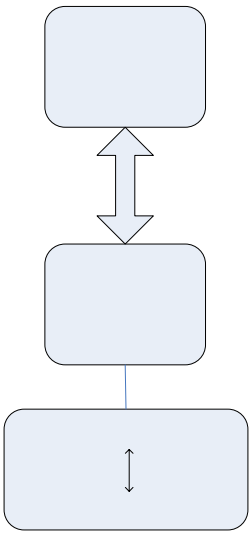
Pasyvioje ODB objekto duomenys atskiriami nuo metodų. Metodai saugomi atskirame faile. Tai reiškia, kad kol objektas yra DB, metodo įvykdyti negalima. Objektas turi būti

iškviestas į taikomąją programą ir tik tada gali būti vykdomas. 1 paveiksle pavaizduota pasyvios ODB schema.



1 pav. Pasyvi objektinė DB

Aktyvioje ODB duomenys ir objektai saugomi kartu, todėl metodai gali būti aktyvuoti tiesiogiai duomenų bazėje. Sukurti aktyvias objektines DB nėra lengva, kadangi tenka tokioje ODVS realizuoti visas programavimo kalbos galimybes. Reikia pabrėžti, kad tik aktyviomis ODB pilnai įgyvendinami objektinio metodo privalumai. 2 paveiksle pavaizduota aktyvios ODB schema.



2 pav. Aktyvi objektinė DB

Pasyvios ODB labai tinka sudėtinių objektų saugojimui ir paskirstytam naudojimui. Tačiau reikia objektinių taikomųjų programų, kurios galėtų aktyvuoti šiuos objektus.

Tuo tarpu aktyvias ODB gali naudoti visos, ir neobjektinės, programos. Tai reiškia, kad aktyvios ODB yra platesnės paskirties. Dėl šių ir kitų privalumų, aktyvios ODB tampa intelektualių IS kūrimo pagrindu.

Didžiausia problema, su kuria susiduria DBVS yra bendrų kelioms skirtingoms programoms duomenų valdymas. Bendri duomenys reiškia ryšį tarp programų, o tai pažeidžia modulinio programavimo principą – visišką programinių modulių nepriklausomumą.

Objektinis metodas talpina duomenų paieškos procedūras tiesiog DB, duomenų objektų apibrėžimuose. Taip pašalinamas procedūrų dubliavimasis, sutrumpėja programų tekstai, sumažėja klaidos tikimybė. Lengviau atlikti duomenų struktūrų pakeitimus, nes jie liečia tik atskirą klasę.

Objektinėse DB galima duomenų reikšmių kontrolė (monitoringas). Paprastai duomenų reikšmės kontroliuojamos programose. Aktyvūs objektai gali kontroliuoti savo kintamuosius ir pranešti kitiems objektams apie kritinius atvejus. Tai įgyvendinama papildant kontrolės procedūromis objektų metodus, susietus su kintamųjų reikšmių modifikavimu. Toks sprendimas ne tik logiškas, jis efektyvus, kadangi kintamieji yra tikrinami tik tada, kai keičiasi jų reikšmė. Tokia vidinė duomenų reikšmių kontrolė yra vadinama trigeriais (*triggers*), nes ji informuoja vartotoją apie kritines situacijas. Nors šis mechanizmas yra ir RDB, tačiau ODB jis paprasčiau valdomas. Aktyvūs objektai gali atlikti skaičiavimus, apklausti kitus objektus duomenų bazėje, pareikalauti informacijos iš vartotojo, atlikti įvairiausius duomenų reikšmių patikrinimus. Trigerio atliktos kontrolės (monitoringo) rezultatas gali būti pateiktas įvairiu pavidalu – pradedant pranešimu ekrane, baigiant išsamia atliktos analizės ataskaita.

Aktyvioje ODB galima atlikti pačias sudėtingiausias operacijas su duomenimis, visa taikomoji programa gali būti vykdoma ODB aplinkoje. Toks taikomųjų programų vykdymas turi eilę privalumų, lyginant su įprastiniu jų realizavimu. ODB esančios taikomosios programos yra prieinamos visiems sistemos vartotojams visą laiką.

Besivystant objektinei technologijai, įvairios kompanijos siūlo objektinėms programavimo kalboms skirtas DBVS. Nemažai ODBVS yra mokamos. Tačiau vis dar

vystantis technologijai nemaža dalis programinės įrangos kūrėjų siūlo nemokamus produktus. Šiame darbe apžvelgiamos nemokamos (atviro kodo) ODBVS. Jos pateiktos 1 lentelėje [5-10].

1 lentelė. Nemokamos ODBVS.

ODBVS	Programavimo aplinka	Trūkumai
db4o	• J2EE • .NET	Pastebėti sistemos realizavimo metu;
SOD2	• J2EE • kompiliuojančios C++	Neturi užklausų mechanizmo;
ozone	• J2EE	Realizuota JAVA kalba ir veikia tik J2EE aplinkoje; Neturi užklausų mechanizmo;
ObjectDB	• J2EE	Realizuota JAVA kalba ir veikia tik J2EE aplinkoje; Nemokamas paketas nepritaikytas kliento-serverio architektūrai (nebent DB yra pačiame serveryje);
GigaBASE	• J2EE • kompiliuojančios C++	Nėra galimybės dirbti su *null* reikšmėmis;
XL2	• J2EE	Neturi duomenų atstatymo priemonių; Neturi užklausų mechanizmo;

Matome, kad dauguma atviro kodo duomenų bazių sukurtos J2EE programavimo aplinkai. Tačiau ne visos jos atitinka ODMG specifikacijas ir užtikrina pilną ODBVS būdingą lankstumą. Todėl pasirinkti vieną iš pateiktų duomenų bazių nėra sudėtinga. Vis tik, iki sistemos realizavimo, nepavyko įžvelgti nė vieno trūkumo db4o ODBVS, kuomet kitos J2EE realizuotos DBVS turi keletą trūkumų.

2.1.1. db4o objektinė DBVS

db4o duomenų bazę sukūrusi ir tebevystanti kompanija db4objects pradėjo veiklą nuo 1993 metų. Šiuo metu yra pristatyta 6 DBVS versija, kuri naudota darbui realizuoti. db4o yra atviro kodo ODBVS, leidžianti kurti programinę įrangą (PĮ) Java ir .NET aplinkose. Kuriamame programiniame pakete tereikia įdėti aplinkai pritaikytą biblioteką (*.jar* – Java programavimo aplinkai arba *.dll* - .NET programavimo aplinkai). Bibliotekos dydis tik 350 KB, reikalingas minimalus RAM – 1MB. Taigi db4o reikalauja mažai kompiuterinių resursų. Ši ODBVS veikia su Java JDK 1.1x iki 5.0 versijomis, taip pat palaiko J2ME

dialektus (CDC, Symbian ir kt.). .NET aplinkoje veikia visose platformose ir dirba su visomis .NET kalbomis (C#, VB.NET ir kt.). Prie šių funkcionalumo bruožų dar galima pridėti, tai, kad db4o sukurtos dvi užklausų kalbos (QBE ir S.O.D.A.), bei vartotojo sąsają (ObjectManager), skirtą ODB bylai peržiūrėti. Ši byla kiekvieno programos vykdymo pradžioje yra atidaroma, o vykdymo pabaigoje – uždaroma. ODB objektus galima lengvai perkelti ir kopijuoti iš vienos bylos į kitą, taip pat duomenis saugoti užšifruotus. db4o byloje galima įdiegti automatinį duomenų atstatymo mechanizmą, nuolat daryti atsargines kopijas, tekstinius duomenis saugoti „Unicode“ koduote. Šios bylos maksimalus dydis – 254GB.

db4o veikia kliento-serverio architektūroje. Ši ODB sėkmingai įdiegta mobiliuose bei medicininiuose įrenginiuose, tai pat automobilių pramonėje, tinklo paslaugų kūrimui ir kitur. ODB galimybės ir lankstumas plačiai naudojamos biomedicinoje, finansų rinkoje, transporto įmonėse ir kitose ūkio šakose.

2.2.Medicininių duomenų saugojimo standartai

Šiuolaikinės informacinės technologijos sudaro prielaidą saugoti duomenų bazėse ir perduoti ryšio kanalais visą ligos istorijos tekstą su lydinčia informacija, tame tarpe medicininius vaizdus. Tačiau iškyla svarbi problema – ligos istorijos duomenų, sukauptų DB, panaudojimas. Efektyviam reikiamų duomenų išrinkimui, ligos istorijos duomenis reikia saugoti tokioje struktūroje, kuri palengvintų jų išrinkimą ir apibendrinimą. Taigi IS turi būti lanksti, kad galėtume lengvai duomenis patalpinti ir išgauti.

Iškilus suderinamumo ir tarpnacionalinių sveikatos paslaugų užtikrinimo poreikiams bei vystantis kompiuterinėms technologijoms, medicinoje didėja poreikis vieningo tarptautinio standarto skirto medicinos duomenų mainams. Kiekvienoje šalyje šie klausimai sprendžiami skirtingai, todėl šiuo metu egzistuoja skirtingi standartai: ASTM, ASC X12, IEEE/MEDIX, NCPDP, HL7 [11], DICOM [12], CEN/TC 251 [13] ir kt. (išsamus standartų sąrašas pateiktas 1 priede 1 lentelėje). Standartų grupę kurianti organizacija, grupė ar komitetas specializuojasi, pvz.: ASC X12N užsiima išorinių standartų skirtų elektroninių dokumentų mainams, AST E31.11 – laboratorinių testų duomenų mainų standartais, IEEE – P1157 – medicinos duomenų mainų standartais

(MEDIX), ACR/NEMA DICOM – vaizdų mainų standartais ir pan. 1996 m. JAV buvo įtvirtintas Amerikos nacionaliniame standartų institute (ANSI) nacionalinis medicininių elektroninių duomenų mainų standartas HL7 (Health Level 7). Šis standartas skirtas medicinos duomenų mainams tarp medicininių IS. Europoje įkurtas standartizavimo komitetas CEN/TC 251, jis yra paskelbęs sistemą tarpusavyje susijusių standartų [žr. 1 Priedo 1 lentelę], į kuriuos orientuojasi Europos ir kitų šalių gamintojai.

Gyventojų mobilumas ir lengvai prieinamos aukštos kokybės sveikatos apsaugos poreikis skatina tarptautinius standartus kuriančių organizacijų bendradarbiavimą. Standartai globalizuojasi: kūrėjai vienija pastangas, vyksta bendri HL7 bei CEN/TC251 komitetų posėdžiai ir koncepcijos konverguoja į globaliai priimtinus ir stabilius sprendimus. Atsiradus bendradarbiavimui, atsiranda ir geresnė kuriamų produktų kokybė. O tai turi reikšmės ir PĮ kūrimui, kuriamos sistemos turi gebėti dirbti su šių standartų duomenis.

Vienas iš intensyviai vystomų duomenų mainų standartų yra DICOM (Digital Imaging and Communications in Medicine). Pirmoji šio standarto versija sukurta 1985 metais Amerikos radiologijos koledže (American College of Radiology, ACR) ir Nacionalinės elektroninės aparatūros kūrėjų asociacijoje (National Electrical Manufacturers Association, NEMA). DICOM tai industrinis standartas skirtas radiologinių vaizdų ir medicinos duomenų apsikeitimui tarp sistemų. Atskira DICOM byla susideda iš antraštės ir vaizdinės informacijos. Antraštėje saugomi duomenys apie pacientą, atvaizdą ir kita informacija. Standartas DICOM aprašo paciento asmeninius duomenis, tyrimų atlikimo sąlygas, paciento padėtį vaizdo formavimo metu ir pan. Dar vienas svarus DICOM privalumas yra tas, kad daugialypės terpės informacijai galime nustatyti kokybę (suspaudimo lygį), tai svarbu keičiantis duomenimis internetu. Šios bylos sudaromos naudojantis specialiomis programomis. Sukurta daug nemokamų programų DICOM byloms kurti ir peržiūrėti. Šiuo metu yra aktuali 3.0 standarto versija (paruošta 1993 metais) ir skirta medicinos vaizdų mainams. Standarte pateikta 29 diagnostinių metodų aprašymai. Šis standartas plačiai naudojamas JAV, Japonijoje, Vokietijoje ir kitose šalyse, taip pat ir Lietuvoje.

Taigi, standartai sukurti, tačiau patrauklios IS yra vis dar kuriamos. Didžiausia problema yra centralizuotas medicininių vaizdų standartu suformuotų bylų saugojimas, tokių DB apsauga ir duomenų paieška. IS būtinumą galime įžvelgti iš 2 lentelės, kurioje pateikta

Lietuvos sveikatos įstaigose užregistruotų susitikimų su pacientais atvejų skaičius per metus, per mėnesį ir per dieną [14].

2 lentelė. Lietuvos sveikatos įstaigose užregistruotų susitikimų su pacientais atvejų skaičius per metus, per mėnesį ir per dieną

Įvykiai	Per metus	Per mėnesį	Per dieną	Viso per parą
Gimdymas	29,765	2,480	113	113
Susitikimai PCP lygyje	22,910,700	1,909,225	86,783	
Skubi pagalba	780,700	65,058	2,957	2,957
BPG, PP, ambulatorija	14,695,500	1,224,625	55,665	55,665
Nukreipta į SSP	7,434,500	619,542	28,161	28,161
Epikrizė po ASP SP	7,434,500	619,542	28,161	18,305
Hospitalizavimas	811,300	67,608	3,073	3,073
Epikrizė po hospitalizavimo	811,300	67,608	3,073	1,229
Labo. tyrimai PSP lygyje	15,896,417	1,324,701	60,214	3,011
Labo. tyrimų PSP lygyje rezultatai	15,896,417	1,324,701	60,214	3,011
Radiologija PSP lygyje	2,180,201	181,683	8,258	83
Radiologijos PSP lygyje rezultatai	2,180,201	181,683	8,258	83
Labo. tyrimai ASP lygyje	12,007,240	1,000,603	45,482	2,274
Labo. tyrimų ASP lygyje rezultatai	12,007,240	1,000,603	45,482	2,274
Radiologija ASP lygyje	892,430	74,369	3,380	34
Radiologijos ASP lygyje rezultatai	892,430	74,369	3,380	34
Receptai	25,000,000	2,083,333	94,697	94,697
Nedarbingumo lapeliai	1,787,686	148,974	6,772	6,772
Viso transakcijų	**120,737,827**	**10,061,486**	**457,340**	**221,774**

2.3.J2EE ir .NET technologijų palyginamoji analizė

Tinklo paslaugų (Web service) technologijos šiuo metu yra labai plačiai naudojamos, dėl to, kad jos suteikia daug galimybių programuotojui, jų realizavimas nėra sudėtingas. Pagrindinės vyraujančios platformos yra J2EE ir .NET. Kompanijos siūlančios šiems standartams skirtus programinius paketus yra vienos didžiausių konkurenčių. Taigi programuotojui, ketinančiam naudoti vieną iš jų iškyla sunkus uždavinys – kurį pasirinkti. Sun Microsystems siūlo naudoti Java 2 Platform, Enterprise Edition (J2EE), o Microsoft - .NET technologiją. Abi šios technologijos siūlo įrankius tinklo paslaugų

kūrimui. Pagrindinis J2EE privalumas yra tas, kad juo sukurtus programinius modulius ir paketus galima naudoti bet kurioje operacinėje sistemoje. Microsoft platforma sukurta Windows operacinėms sistemoms, tačiau šiuo metu firma siūlo atitinkamus įrankius ir Linux operacinei sistemai. Abi technologijos naudoja objektines programavimo kalbas: .NET – Visual Basic ir kt., J2EE – Java. .NET privalumas yra daugiakalbiškumas, ji palaiko Visual Basic, C#, Fortran ir COBOL programavimo kalbas. Vienas .NET komponentas gali būti parašytas viena iš programavimo kalbų, kitas – kita. Sėkmingą darbą tarp jų užtikrina įdiegtos technologijos. Visų pirma programos kodas verčiamas į Microsoft tarpinę kalbą (Microsoft Intermediate Language, trumpinama MSIL arba IL). Šis kodas yra neutralus kalbų atžvulgiu. Po to jis pervedamas į specialią programą-interpretatorių – Common Language Runtime (CLR). Šis interpretatorius turi daug naudingų savybių, t.y. automatinis programavimo šiukšlių rinktuvas, išimčių (situacijų) valdymas, sintaksės klaidų taisymas ir kitos. Java kodu parašytas programos kodas kompiliuojamas į objektinį kodą – baitkodą. Baitkodas yra analogas MSIL. Šį kodą toliau vykdo Java abstrakti virtualioji mašina (JVM) – Java Runtime Environment (JRE). Ji yra analogas .NET CLR.

Dar vienas .NET technologijos privalumas yra tas, kad ji geba atvaizduoti puslapius įvairiais HTML formatais. Nors J2EE technologijoje servletais galima gauti tokį patį rezultatą, tačiau jie reikalauja papildomo programavimo [15].

Microsoft siūlo vizualinę .NET programavimo aplinką, tai Visual Studio .NET. Joje galima kuri programas visomis .NET programavimo kalbomis. J2EE atitinkami įrankiai nepasižymi visomis reikiamomis galimybėmis.

1 lentelėje pateikta J2EE ir .NET technologijų analizė jų funkcionalumo požiūriu [16-18].

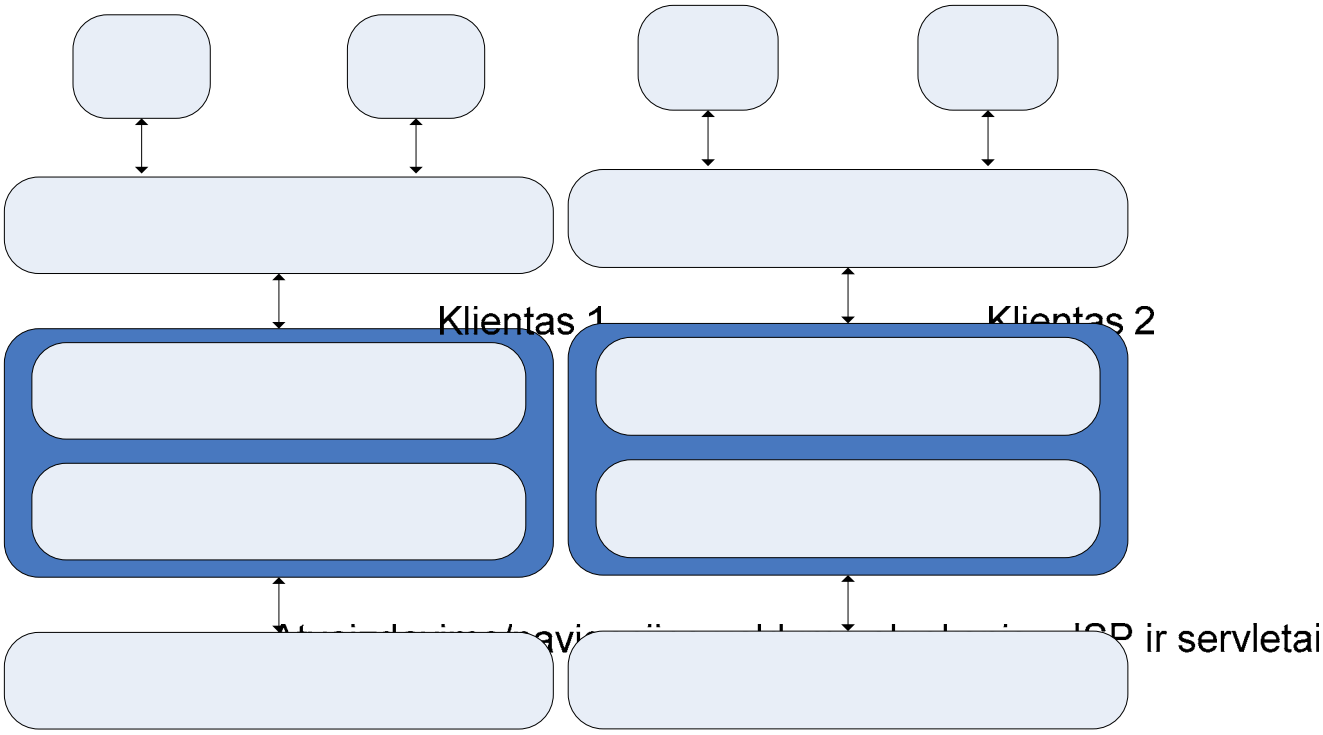
3 lentelė. J2EE ir .NET technologijų suvestinė.

Funkcija	J2EE	.NET
Priėjimas prie reliacinių DB	Java Database Connectivity (JDBC) arba SQL/J	Active Data Objects (ADO.NET)
Duomenų atvaizdavimas naršyklėje	Java server pages (JSP) ir servletai	Active Server Pages (ASP.NET)
Vartotojo sąsajos atvaizdavimas	AWT/Swing	Windows arba internetinės formos

Užklausų valdymas	Java Messaging Service (JMS)	Microsoft Message Queue (MSMQ)
Tinklo paslaugų kūrimas	Java Web Services Developer Pack (JWSDP)	Galima kurti naudojant Visual Studio programavimo aplinką
Technologija	J2EE	.NET
Infrastruktūra	Enterprise Java Beans (EJB)	COM+
Komponentų registracija	Java Naming and Directory Interface (JNDI)	Active Directory Services Interface (ADSI)
Duomenų saugykla	-	SQLServer[1]

Kaip matome abi technologijos yra ypatingai panašios. Svarbiausia, kad jomis galima kurti tinklo paslaugas, atvaizduoti duomenis internete. .NET tinklo paslauga gali būti vartotoju J2EE tinklo paslaugoms ir atvirkščiai, kadangi standartai, kuriuos pastarosios naudoja, yra vienodi. Be to tinklo paslaugų programas, galima pakartotinai panaudoti, nesvarbu kokia platforma naudojantis jos sukurtos. 3 paveiksle pavaizduota pavyzdinė tinklo paslaugų architektūra J2EE ir .NET technologijoms, remiantis trijų lygių architektūros standartu. Klientas yra interneto naršyklė (Internet Explorer, Opera ir kt.). Atvaizdavimo sluoksnis apibūdina kokios technologijos generuoja HTML formato bylas. Biznio logikos sluoksnis apima transakcijas, nutolusių objektų paskirstymą ir XML tinklo paslaugų palaikymą. Duomenų atvaizdavimo sluoksnis apima technologijas naudojamas duomenims saugomiems reliacinėse DB pasiekti. Duomenų bazė šiuo atveju yra reliacinė. Galima naudoti objektines duomenų bazes. Šių DB integracija su J2EE ir .NET technologijomis pateikta 2.1 skyriuje.

[1] SQLServer yra .NET DB technologija.



a) b)

3 pav. Pavyzdinė tinklo paslaugų architektūra J2EE (a) ir .NET (b) technologijoms

Kaip matome iš 3 paveikslo J2EE ir .NET technologijų tinklo paslaugų architektūros yra analogiškos.

Kiekviena kompanija stengiasi reklamuoti savo produktą, taigi sunku rasti objektyvių straipsnių. Dauguma autorių, dėl technologijų panašumo, nenurodo ypatingų priežasčių, kurios galėtų lemti pasirinkimą [18-20]. Apsisprendimą gali lemti tai, kad J2EE technologija pradėta vystyti 3 metais vėliau negu .NET. Dažniausiai apsispręsti turi pats programuotojas kurią platformą pasirinkti arba kompanija, kuri ketina diegti vieną ar kitą technologiją. Vis tik galima įžvelgti kai kuriuos .NET privalumus ir jos plėtojimo, bei naudojimo galimybes.

2.4. Objektinių technologijų ir duomenų saugojimo standartų analizės išvados

J2EE ir .NET objektinio programavimo technologijos yra labai panašios tiek funkciniu, tiek ir technologiniu aspektu. Darbo realizacijai pasirinkta .NET technologija dėl jos

plėtojimo ir naudojimo galimybių, taip pat daugiakalbiškumo galimybės ir patogios Visual Studio .NET programavimo aplinkos. ODBVS palygintos su RDBVS. Pateikti ODBVS privalumai saugant daugialypės terpės informaciją, taip pat analizuotos šių DB savybės. Apžvelgtos atviro kodo ODBVS, darbo realizacijai pasirinkta db4o ODBVS. Pateikti šios ODBVS privalumai. Aprašyti medicininių duomenų saugojimo standartai ir jų panaudojimo problemos kuriant IS. Apibūdintas DICOM standartas, kadangi jis yra vienas plačiausiai naudojamų medicininių duomenų mainų standartų, taigi siekta, kad sukurta IS dirbtų su šio standarto bylomis.

3. Vartotojų poreikiai ir medicininės IS

3.1.Medicininių IS vartotojų poreikiai

Šiuo metu Kauno medicinos universiteto Biomedicininių tyrimo instituto Oftalmologijos laboratorijoje įdiegta ir naudojama medicininių duomenų keitimosi sistema MediPas [21]. Ši sistema skirta realaus laiko konferencijų organizavimui, kurių metu vyksta specialistų vaizdinės konsultacijos, diagnostinių vaizdų kaupimas, saugojimas DB, paieška ir jų perdavimas tarp telemedicinos stočių. Tačiau sistema netenkina visų svarbiausių vartotojų poreikių. Sistemoje duomenys kaupiami nutolusioje arba asmeniniame kompiuteryje įdiegtoje duomenų bazėje, tačiau nėra centralizuoto duomenų valdymo, informatyvios duomenų paieškos, ribotos duomenų įvedimo galimybės. Vaizdai iš skirtingų aparatūrinių įrenginių įkeliami naudojantis Twain sąsaja, kuri neužtikrina patikimo duomenų perdavimo į DB. Esama sistema saugo vaizdus lokalioje RDB (MS Access) arba serveryje įdiegtoje RDB (MS SQL server). Tačiau saugomi vaizdai, nenurodant dalies vartotojui svarbios informacijos. Ši programa dalinai apdoroja DICOM standarto bylas, kuris šiuo metu yra naudojamas vaizdams ir su juo susijusiai informacijai saugoti. 4 lentelėje apibendrintos šios vartotojams aktualios problemos ir pateikti sprendimo būdai.

4 lentelė. Vartotojų tikslai ir problemos

Problema	Dabartinė sistema	Sprendimas
Šiuo metu yra daugybė įvairių tyrimų vaizdinių duomenų, kurie saugomi skirtingose DB, kai kurie atskirai nuo juos lydinčių duomenų.	MediPas nepritaikyta įvairiems medicininiams duomenims saugoti.	Galima įgyvendinti įvairialypių duomenų kaupimą ir saugojimą, pritaikyti DB įvestos informacijos paieškos ir apdorojimo algoritmus.
Vartotojas informacijos gali ieškoti tik tekstiniuose laukuose.	Reliacinė DB neleidžia atlikti paieškos *binary* tipo duomenyse.	Galima sukurti kliniškai reikšmingų požymių atpažinimo algoritmus tiek tekstinėje, tiek ir vaizdinėje informacijoje. Tai galima efektyviai panaudoti sprendžiant diagnostikos klausimus.
Didelėse DB paieška vyksta lėtai, sunku taikyti sudėtingus paieškos algoritmus.	Nėra sistemos kuri atrinktų duomenis pagal skirtingus parametrus.	Paieška greitesnė, nes ODB duomenis saugo naudodamos adresines struktūras tarp objektų.

Keitimasis duomenimis	Nėra RDB, skirtų medicininės informacijos kaupimui, struktūros sudarymo standarto, todėl keitimasis duomenimis tarp tokių DB yra sudėtingas, reikia sukurti konvertavimo algoritmus.	DICOM bylomis galimas tiesioginis keitimasis tarp kelių DB.
Vartotojas gali sukurti DICOM bylas, tačiau negali atlikti išsamios paieškos tarp laukų.	MediPas pritaikyta tik saugoti, skaityti ir keistis DICOM tipo bylomis.	ODB tiesiogiai bendrauja su DICOM standarto bylomis. Galima pritaikyti juose įvestos informacijos paieškos ir apdorojimo algoritmus.

3.2.Medicininių IS tyrimas

Šiuo metu gan plačiai naudojamos galingos sistemos, skirtos medicininiams skaitmeniniams vaizdams saugoti – PACS (Picture Archive and Communication Systems). Šios sistemos yra įvairiai tobulinamos, siekiant tenkinanti skirtingus vartotojų poreikius. Jos pagrįstos daugialypės terpės duomenų saugojimu reliacinėse duomenų bazėse, yra sujungtos tinklais su šalies ligoninėmis ir leidžia gydytojams prieiti prie sistemos ir joje saugomų duomenų. Tam, kad sistemos užtikrintų duomenų vieningumą, jos naudoja standartus (DICOM ar kitus), kurie tinklais keliauja naudojant SQL ir HTTP protokolus. Taip pasiekiama standartų integracija su kitus standartus naudojančiais servisais. Tačiau tokios DB sukūrimas yra sudėtingas procesas, kadangi reikia atsižvelgti į specifikacijas, gydytojų poreikius, taip pat duomenų modelis turi atitikti standartus. Įvykus pasikeitimams bet kuriame sistemos lygyje gali subyrėti dalis sistemos, reikia atlikti daug programinių pakeitimų. Šią problemą galima paprasčiau išspręsti naudojant objektines duomenų bazes. Taip pat jose galima lanksčiai realizuoti duomenų įkėlimą, atnaujinimą ir paiešką, naudojant specialius algoritmus ir klinikinių sprendimų medžių metodologiją. PACS sistemos yra labai brangios, todėl ne visos gydymo įstaigos gali jas įsigyti. Lietuvoje PACS sistema yra įdiegta Vilniaus Santariškių ligoninėje.

Sukurta IS skirta oftalmologiniams vaizdams saugoti ir analizuoti, todėl tolimesnė sistemų analizė skirta šioms sistemoms apžvelgti.

Dauguma oftalmologinius vaizdus saugančių IS yra sukurtos ir pritaikytos įvairių oftalmologinių tyrimų aparatams. Juose yra integruoti reikiami įvesties/išeities įrenginiai,

techninė įranga (TĮ) ir PĮ, kuri geba apdoroti vaizdus, juos kaupti ir pateikti. Tačiau dažniausiai vaizdai yra saugomi atskirai nuo jį lydinčių skaitinių ir tekstinių duomenų. Norint šiuos duomenis transportuoti, yra formuojamos bylos vienu iš medicininių duomenų saugojimo standartu (dažniausiai DICOM). Tokiuose aparatuose taip pat įdiegta prieiga prie interneto tinklo. Galimas analizės tipo bendravimas tarp šių specifinių sistemų arba tarp tos pačios kompanijos sukurtų aparatų, kadangi dažniausiai diegiama ta pati PĮ. Reikiama informacija gali būti siunčiama tiesiog elektroniniu paštu, sukuriant DICOM ar kito standarto bylą. Tačiau centralizuotos DB, kaip ir priėjimo prie jos nėra. 5 lentelėje pateiktos lyderiaujančios kompanijos siūlančios minėtus sprendimus.

5 lentelė. Kompanijos siūlančios oftalmologinius vaizdus saugančias IS

Gamintojas	Ophthalmic Imaging Systems	Topcon Medical Systems, Inc.	Zeiss	Clarity Medical Systems, Inc.
Produktas/IS	Ophthalmology Office	IMAGEnet Professional	VISUPAC software	RetCam
Duomenų saugykla	Network Attached Storage (NAS)	IMAGEnet	SQL DB	ODB db4o
Interneto prieiga	LAN, WAN, interneto naršyklė	el. pašto paslauga	interneto naršyklė	LAN, interneto naršyklė
Palaikomi standartai	DICOM	DICOM	DICOM	-
PĮ	Windows XP Professional, OIS WinStation, PC Anywhere, Norton AntiVirus, PaintShop Pro	Specifinės programos	Windows 2000	Specifinės programos

Šie komerciniai produktai yra galingos sistemos, siūlančios diagnostinius, analizės ir laboratorinių tyrimų sprendimus, be kurių šiuolaikinė medicina negali apsieiti. Iš lentelės matyti, kad duomenų mainams šios IS naudoja DICOM standartą. Taigi, šio darbo metu sukurta IS apjungia tokių sistemų sukurtus duomenis ir juos saugo centralizuotai, bei suteikia nuolatinę prieigą.

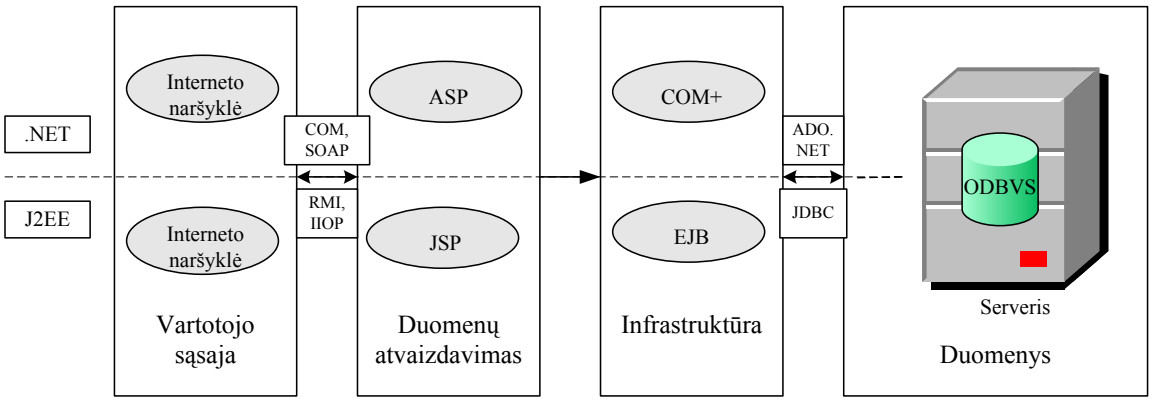
2 Priede 1 lentelėje pateikta daugiau kompanijų siūlančių įvairius produktus.

Panaši funkciniu pažiūriu į analizuotas sistemas yra ir minėta sistema MediPas.

4. IS architektūra, savybės ir reikalavimai

4.1. IS architektūra ir įgyvendinimo priemonės

Medicininių duomenų paieškos IS realizuota naudojantis .NET technologija Visual C# kalba. Tačiau nepriklausomai nuo .NET ar J2EE objektinių technologijų pasirinkimo galima apibūdinti sistemos architektūrą. 4 paveiksle pavaizduota realizuotos IS architektūra.



4 pav. Informacijos sistemos architektūra.

Schemoje dar kartą įžvelgiame objektinių technologijų tarpusavio panašumą. Šiame paveiksle pavaizduota interneto naršyklė gali būti bet kokia šiuo metu prieinama vartotojams, ASP skriptai realizuoti Visual C# programavimo kalba, ODBVS yra atviro kodo db4o duomenų bazė.

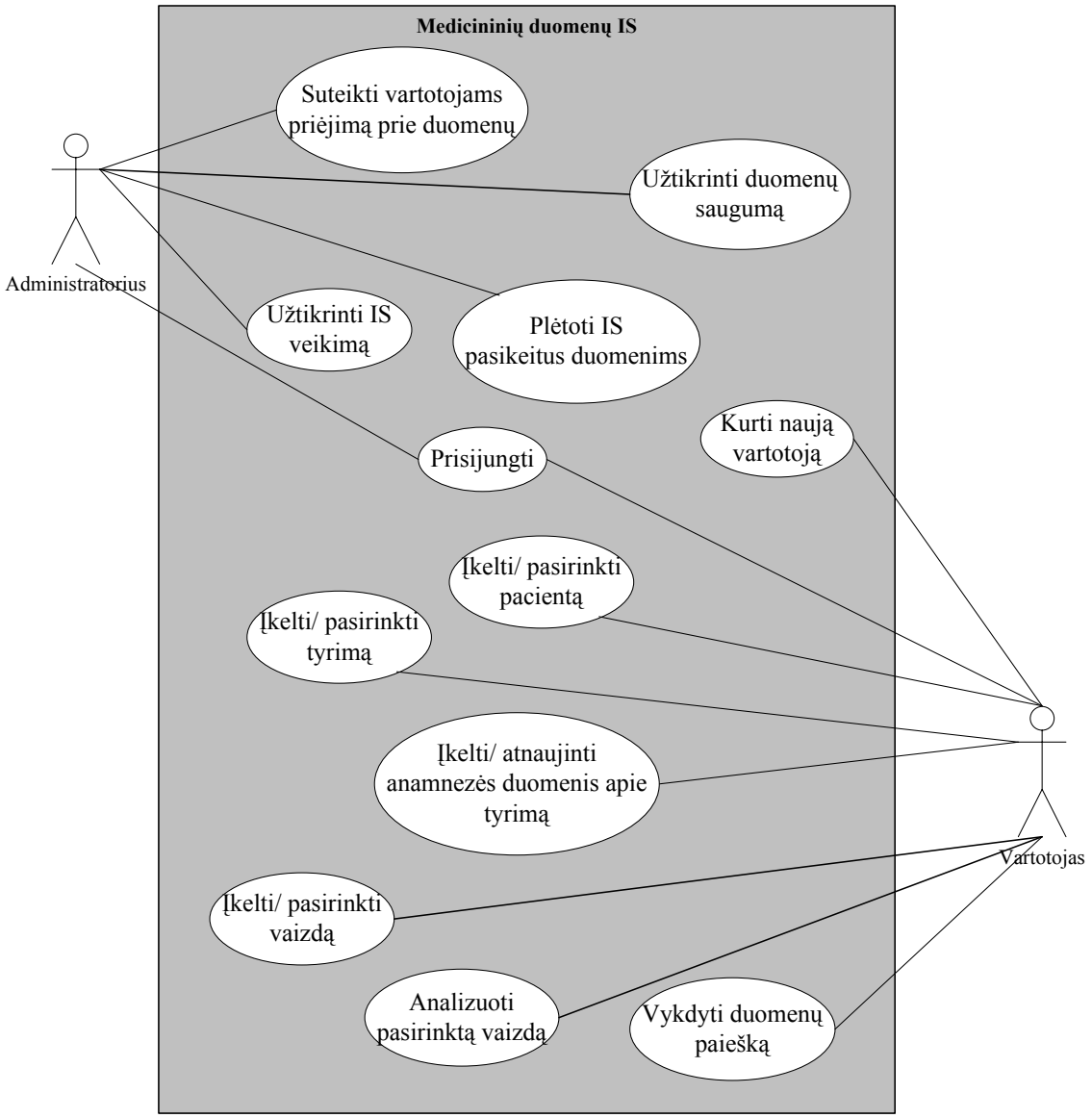
4.2. IS savybės ir tikslas

6 lentelėje pateiktos sistemos savybės, nustatytas kiekvienos savybės įgyvendinimo būtinumas ir paaiškinta kodėl savybė yra būtina, kokios yra jos įgyvendinimo alternatyvos.

6 lentelė. Sistemos savybių analizė.

Sistemos savybė	Vertinimas	Paaiškinimas
Aiški ir patogi vartotojo sąsaja	Būtina savybė	Vartotojo sąsaja turi būti suprantama, kadangi būsimi vartotojai neturi pakankamai kompiuterinių žinių.
Vartotojo sąsaja internetu	Būtina savybė	Tai vienas iš pagrindinių sistemos privalumų.
Duomenų paieška	Būtina savybė	Vartotojams turi būti užtikrinama efektyvi duomenų paieška pagal pasirinktus norimus kriterijus.
Vaizdų saugojimas duomenų bazėje	Būtina savybė	Vartotojams labai svarbu saugoti ne tik tekstinę informaciją apie pacientą ir jam atliktus tyrimus, bet ir vaizdus.
Sistemos administravimas	Būtina savybė	Sistema turi būti administruojama, tobulinama ir prižiūrima.
Saugus duomenų perdavimas internetu	Būtina savybė	ODB ir DICOM standarto bylose saugomi asmeniniai pacientų duomenys, todėl tinklais keliaujanti informacija turi būti apsaugota.
Vartotojo teisių nustatymas	Nebūtina savybė	Ši savybė galėtų būti įgyvendinta ateityje, skirstant vartotojus į galinčius įvesti duomenis ir galinčius juos peržiūrėti.
Klinikinių sprendimų medžių formavimas	Nebūtina savybė	Informatyvi duomenų paieška yra svarbesnė sistemos savybė, todėl šios savybės įgyvendinimas atidedamas kitam etapui.

Šio darbo tikslas yra sukurti IS, kuri užtikrintų saugojimą ne tik tekstinės informacijos apie pacientą, jam atliktus tyrimus ir tyrimų vaizdus, bet ir greitą, bei kokybišką sistemos išplėtimą esant pasikeitimams šių duomenų struktūrose ir vartotojų reikalavimuose. Vieni iš pagrindinių sistemos bruožų yra efektyvi medicininių duomenų paiešką ir jų saugojimas patogioje struktūroje. Šios sistemos funkcijos pavaizduotos 5 paveiksle, UML panaudojimo atvejų diagrama.



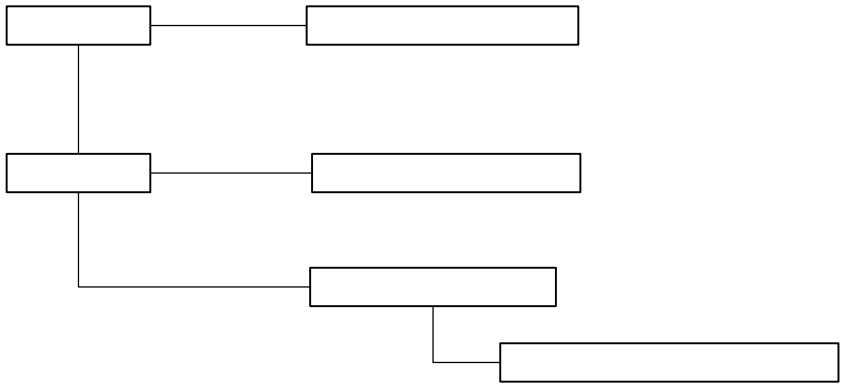
5 pav. IS panaudojimo atvejų diagrama.

Panaudojimo atvejų diagramoje pavaizduotos pagrindinės administratoriaus ir vartotojo funkcinės galimybės. Panaudojimo atvejai specifikuoti 5.2 skyriuje ir 4 priede. Klinikinių sprendimų medžio formavimas nėra įtrauktas į šią schemą, kadangi šiame etape nėra įgyvendintas.

4.3. Reikalavimai

4.3.1. Reikalavimai duomenims

6 paveiksle pateikti IS reikalavimai duomenims.



6 pav. Reikalavimai duomenims

Esybių *Vartotojas* ir *Informacija apie vartotoją* yra duomenys apie vartotoją (gydytoją ir administratorių), kurie reikalingi prisijungimui prie IS. Esybių *Pacientas* ir *Informacija apie pacientą* yra duomenys aprašantys asmeninę informaciją apie pacientą. Esybė *Informacija apie tyrimą* yra duomenys aprašantys tyrimo informaciją. Esybė *Vaizdinė informacija apie tyrimą* yra vaizdas ir duomenys aprašantys vaizdą.

4.3.2. Nefunkciniai reikalavimai

Medicininių duomenų paieškos IS nefunkciniai reikalavimai pateikti 7 lentelėje.

7 lentelė. Nefunkciniai reikalavimai ir apribojimai sistemai.

1.Reikalavimai realizavimo priemonėms	IS turi būti sukurta programavimo priemonėmis skirtomis atvaizduoti informaciją internete. Išanalizavus du tinkamas objektines technologijas (.NET ir J2EE), pasirinkta .NET (priežastys pateiktos 2.3 skyriuje). Programavimo kalbai reikalavimų nėra, taigi pasirinkta Visual C#. DB turi būti lanksti atsižvelgiant į medicininių duomenų kintamumą.
2. Reikalavimai turimų duomenų panaudojimui	Išplėstos sistemos funkcijos turi būti realizuotos pritaikant esančius nestruktūrizuotus duomenis ir užtikrinant galimybę įkelti naujus duomenis ir plėsti sistemos funkcionalumą.

3. Reikalavimai vartotojų identifikavimui ir duomenų saugumui	Sistema gali naudotis tik tam teises turintys vartotojai. Vartotojas identifikuojamas vardu ir slaptažodžiu. Prisijungimo metu sukuriamas ODB įvykis, kuris susiejamas su vartotojo identifikavimu ir visais jo įvestais ar modifikuotais svarbių duomenų įrašais.
4. Reikalavimai vartotojo sąsajai	Sistemos vartotojo sąsaja turi būti aiški ir intuityvi, kadangi būsimi vartotojai neturi pakankamai kompiuterinių įgūdžių. Navigacija tarp langų turi būti lanksti.

4.3.3. Funkciniai reikalavimai

Šiai sistemai keliami funkciniai reikalavimai:

- Galimybė vartotojui (gydytojui) įkelti asmeninius duomenis apie pacientą;
- Galimybė vartotojui (gydytojui) įkelti tyrimo duomenis apie pasirinktą pacientą;
- Galimybė vartotojui (gydytojui) įkelti anamnezės duomenis apie pasirinktą tyrimą;
- Galimybė vartotojui (gydytojui) atnaujinti anamnezės duomenis apie pasirinktą tyrimą;
- Galimybė vartotojui (gydytojui) įkelti tyrimo vaizdus;
- Galimybė vartotojui (gydytojui) peržiūrėti pasirinkto tyrimo vaizdus;
- Galimybė vartotojui (gydytojui) atlikti paiešką ODB pagal pasirinktus kriterijus;
- Galimybė vartotojui (gydytojui) reikalauti priėjimo prie duomenų;
- Galimybė vartotojui (gydytojui) pasiekti pagalbos šaltinius;
- Galimybė administratoriui suteikti vartotojams priėjimą prie duomenų;
- Galimybė administratoriui pašalinti netinkančius priėjimui prie duomenų vartotojus;
- Galimybė administratoriui įgyvendinti pakeitimus sistemoje.

5. Medicininių duomenų IS projektas

5.1. IS sprendimo modelis

IS projektuojama atsižvelgiant į sistemos suderinamumą su standartais. Kaip minėta 2.2 skyriuje DICOM bylos sudarytos iš dviejų dalių – antraštės ir daugialypės terpės informacijos. Antraštė susideda iš specialių žymių, kurios pagal sudėtį skirstomos į modulius. 8 lentelėje apibūdintas paciento modulis.

8 lentelė. Paciento modulio sudėtis.

Atributo pavadinimas	Žymė	Tipas	Apibūdinimas
Diagnozė	(0008,1080)	3	Nustatyta diagnozė
Paciento amžius	(0010,1010)	3	Paciento amžius metais
Paciento dydis	(0010,1020)	3	Paciento ūgis ar kitoks matmuo metrais
Paciento svoris	(0010,1030)	3	Paciento svoris kilogramais
Profesija	(0010,2180)	3	Paciento profesija
Papildoma paciento istorija	(0010,21B0)	3	Papildoma informacija apie paciento medicininę istoriją

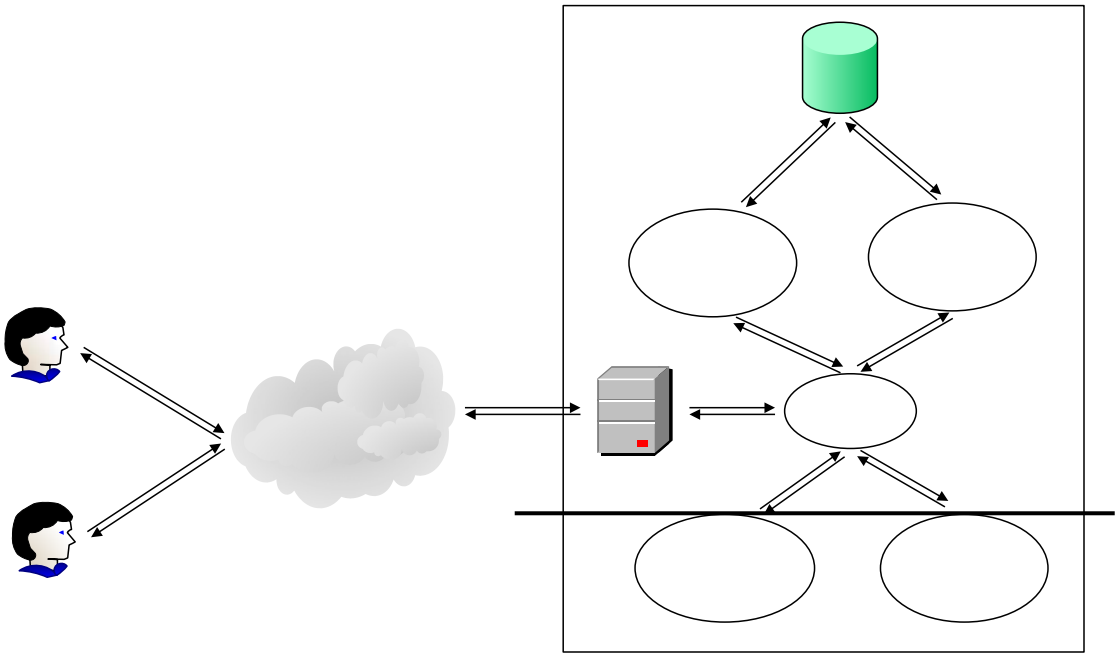
Pastaba: paciento modulis yra tipas, žymimas skaičiumi 3.

Sistemoms dirbančioms su DICOM bylomis, reikalingi specialūs įrankiai vykdantys paiešką šiuose laukuose, kadangi standartinis DICOM užklausų mechanizmas yra nelankstus [22]. PACS sistemos šią problemą sprendžia skirtingai:

1. programiškai nurodomi reikalingiausi laukai ir pagal juos vykdoma paieška (pvz. įvedant paciento ID) arba suformuojama užklausa iš vieno ar kelių galimų kriterijų [23, 24]. Šis sprendimas nereikalauja pažeisti standartu suformuotos bylos struktūros, bet paieška sistemoje priklauso nuo programinio kodo kokybės;
2. suformuojama RDB iš DICOM antraštėse saugomos informacijos. Tai leidžia vykdyti išsamesnę duomenų paiešką, bet procesas yra gan ilgas, o tokios DB administravimas sudėtingas ir lengvai pažeidžiamas [25];
3. konvertuoja DICOM antraštėse esančią informaciją į XML tipo bylas [26]. Nors reikalingas papildomas programinis modulis, tačiau šis sprendimas įgalina atlikti informatyvią paiešką, sistema nepraranda lankstumo.

Sistemos realizavimui pasirinktas antrasis būdas. Tačiau vietoj RDBVS, naudojama ODBVS. Tokiu būdu parodomi šių DBVS privalumai medicininiams duomenims saugoti prieš reliacines DBVS. Darbui su DICOM standarto bylomis reikalingas papildomas

programinis modulis. Sistemos realizavime galėtų būti panaudotas programinio paketo Matlab „Image Processing“ įrankis. 7 paveiksle pavaizduota tokios sistemos struktūra.



7 pav. IS struktūra.

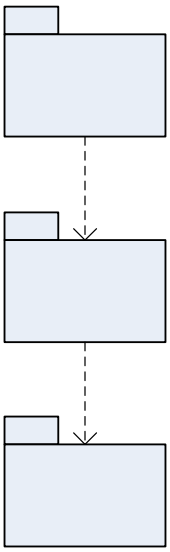
Vartotojas internetu kreipiasi į tinklo paslaugų serverį siųsdamas užklausą. Ši skriptais ją analizuoja, iškviečia reikalingas taikomąsias programas ir tiesiogiai bendrauja su objektine duomenų bazių valdymo sistema. Bendravimas susideda iš duomenų įkėlimo, išgavimo ir atnaujinimo. Duomenų paieška apima svarbiausius kriterijus, pagal kuriuos išrenkami duomenys. DICOM bylų peržiūros programa reikalinga vartotojui norinčiam peržiūrėti visos DICOM bylos turinį. DICOM bylų konvertavimo įrankis reikalingas duomenų esančių ODBVS konvertavimui į DICOM standarto bylą. Tačiau šios funkcijos (7 paveiksle esančios žemiau brūkšnio) neįgyvendintos.

Vartotojas 1

Internetas

Sistemą galima suskirstyti į tris lygmenis, kurie pateikti 8 paveiksle.

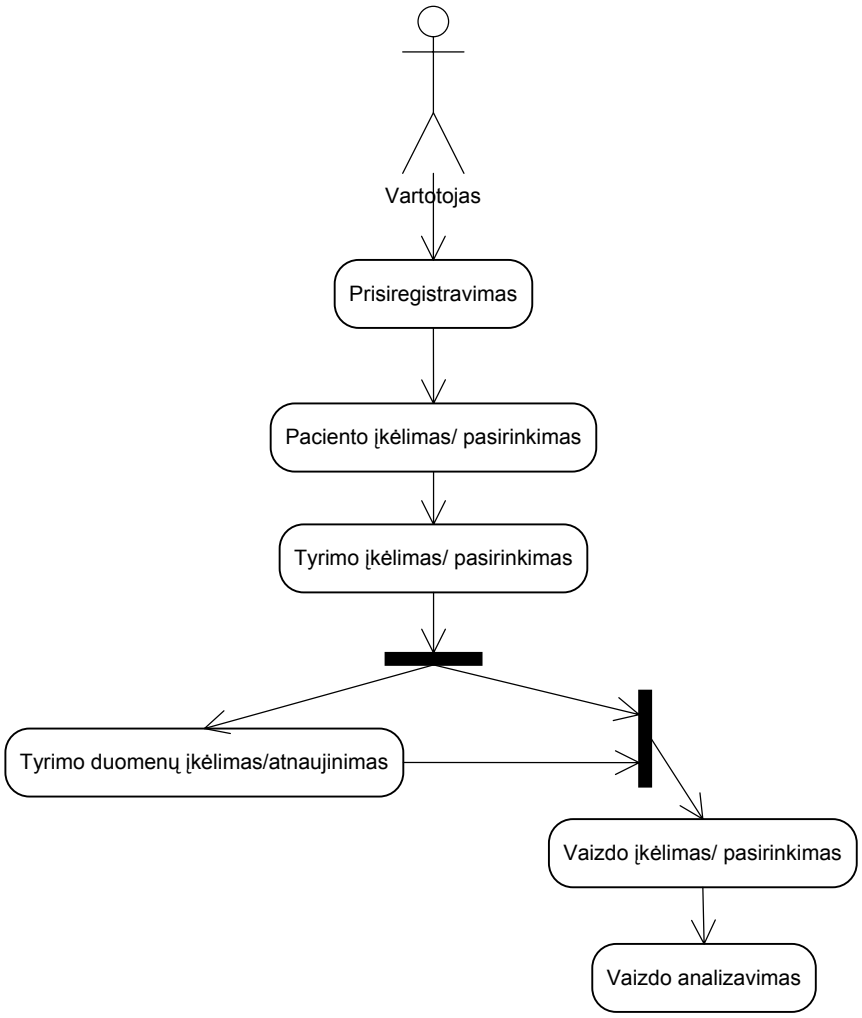
Vartotojas N



8 pav. IS lygmenys.

Tai supaprastinta šios IS trijų lygių architektūros schema. Išplėstinės schemos skirtingoms technologijoms pateiktos 3 paveiksle. Sistemoje vyksta bendravimas tarp vartotojo sąsajos ir duomenų per programinę sąsają.

Vartotojui skirtos sistemos dalies duomenų įkėlimo, atnaujinimo ir vaizdų peržiūros procedūros atliekamos pagal nustatytų veiksmų seką. Ši seka pavaizduota 9 paveiksle.



9 pav. Loginė vartotojo veiksmų seka.

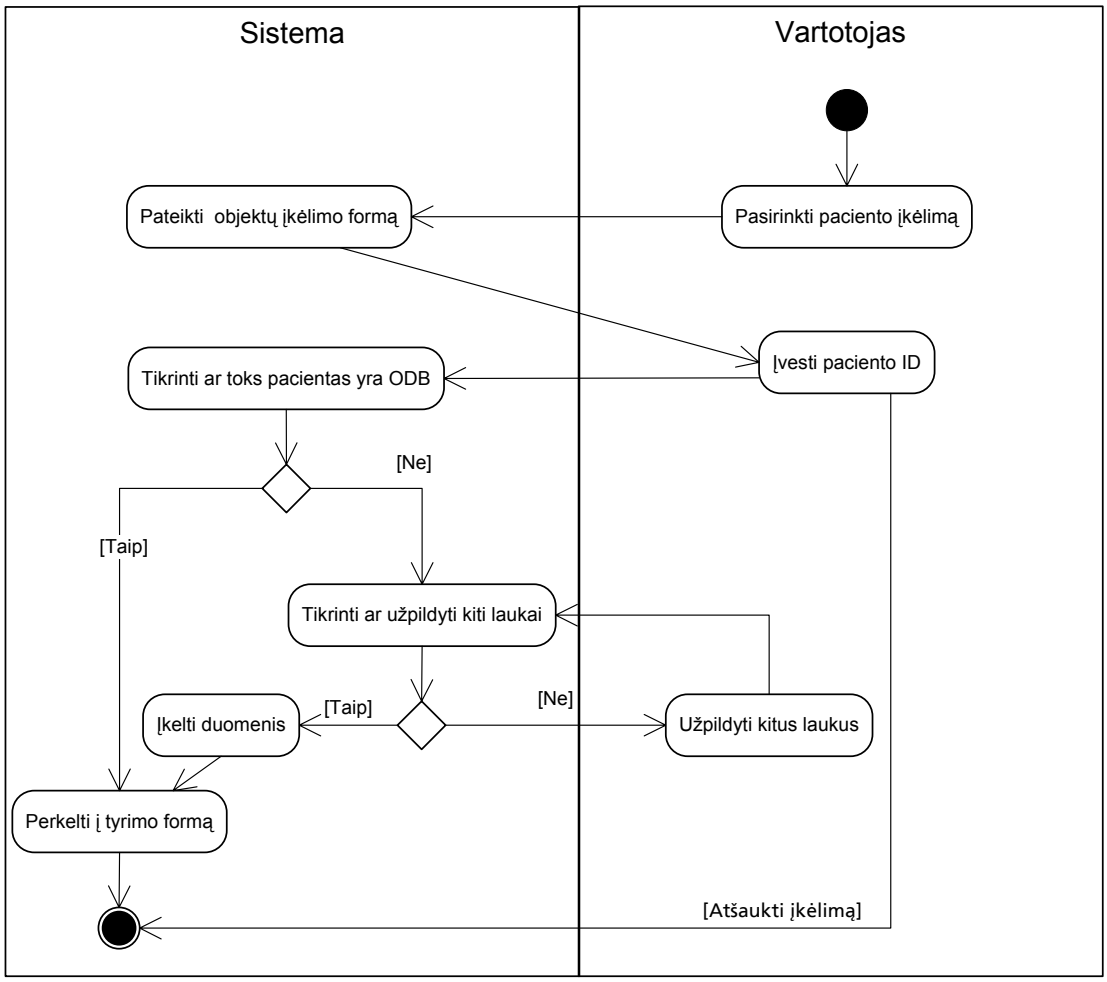
Prisiregistravęs vartotojas papildomai sistemoje gali atlikti duomenų paiešką, kuri yra nepriklausoma nuo kitų galimybių pavaizduotų 9 paveiksle.

5.2. Veiklos diagramos

Remiantis informacinės sistemos panaudojimo atvejų diagrama (5 pav.) ir logine vartotojo veiksmų sekos diagrama (9 pav.) suformuotos IS UML veiklos diagramos. 10 paveiksle pavaizduota pacientų įkėlimo/pasirinkimo diagrama, 11 paveiksle – tyrimo įkėlimo/ pasirinkimo diagrama, 12 paveiksle – oftalmologinio vaizdo įkėlimo/ pasirinkimo diagrama, 13 – duomenų paieškos diagrama, o 14 – naujų vartotojų registravimo diagrama. Kitos ne tokios svarbios ir sudėtingos diagramos pateiktos 3

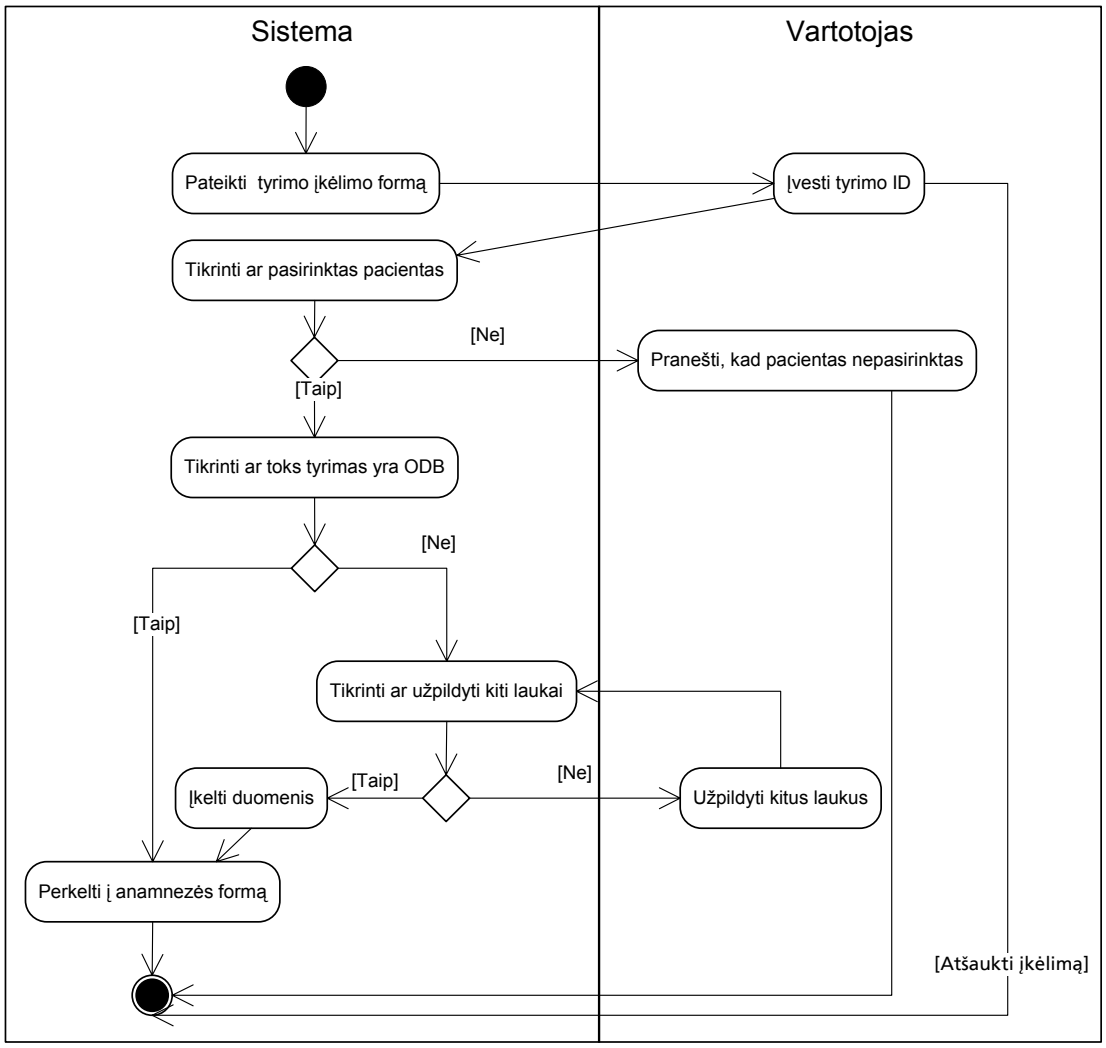
priede 1-4 paveiksluose. Klinikinių sprendimų medžio formavimas nėra įgyvendintas, todėl ši diagrama analizuojama 3 priede (4 pav.).

Registruodamasis prie sistemos tiek vartotojas, tiek administratorius įveda savo vartotojo vardą ir slaptažodį. Šie duomenys saugomi MD5 duomenų perdavimo formatu. Jei vartotojas yra registruotas sistemoje, tikrinamos jo teisės ir jam suteikiamas leidimas dirbti sistemoje. Pirminiame sprendime yra dviejų tipų vartotojų teisės – paprastas vartotojas ir administratorius. Paprastas vartotojas prieina prie visų duomenų – gali juos įkelti, peržiūrėti ir vykdyti paiešką. Administratorius valdo vartotojus, mato koks vartotojas kokius pakeitimus atliko ODB. Jei vartotojas neregistruotas sistemoje gali užpildyti specialią formą ir prašyti leidimo prieiti prie duomenų. 14 paveiksle pateikta naujų vartotojų registravimo veiklos diagrama. Prisijungimo procedūrą galima pakartoti, jei vartotojas pamiršo slaptažodį arba neteisingai jį įvedė. Pamiršus slaptažodį kreipiamasi į administratorių. Vartotojų prisijungimo prie sistemos veiklos diagrama pateikta 3 priede 1 paveiksle. Prisijungęs vartotojas gali prieiti prie sistemos resursų.



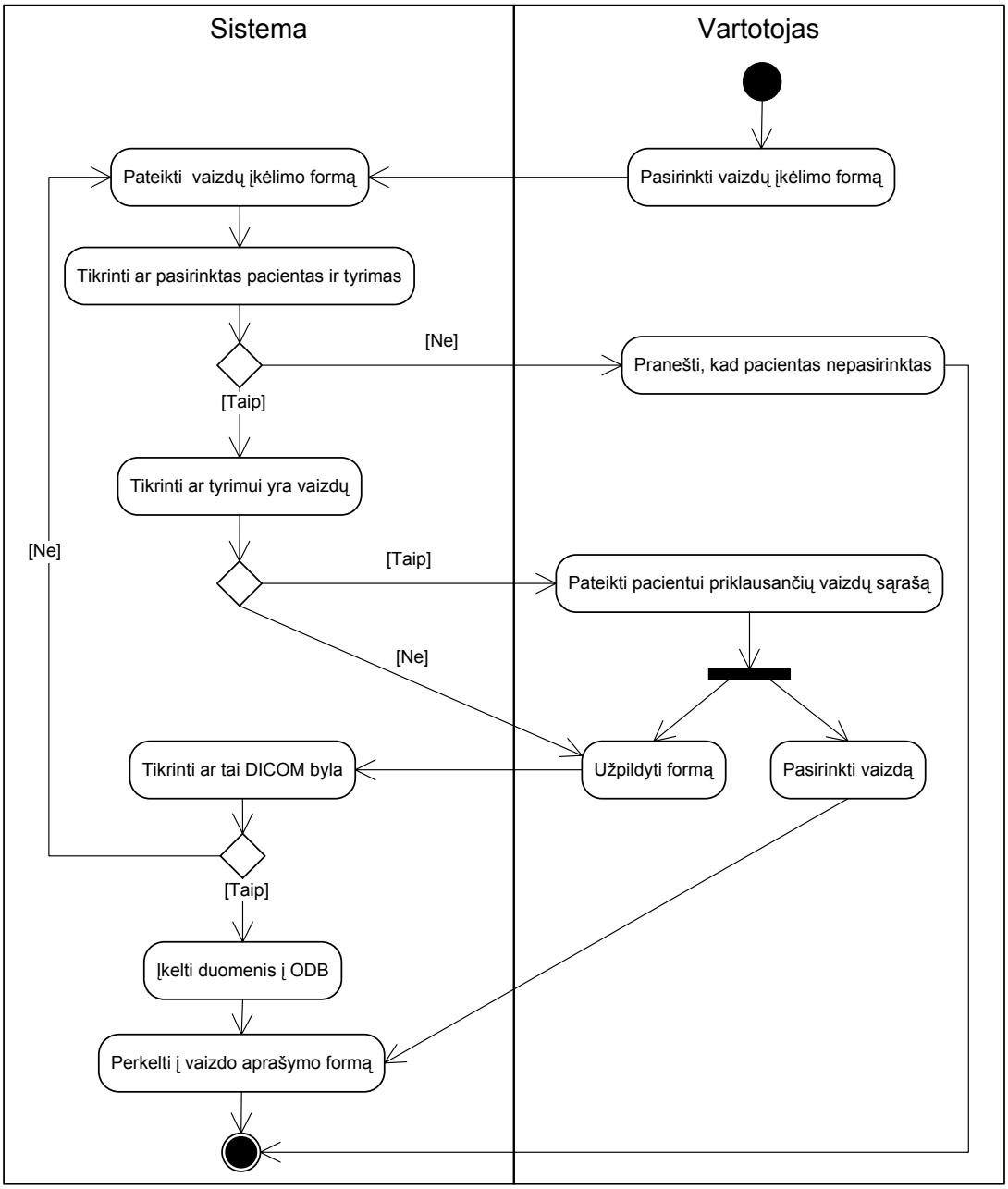
10 pav. Pacientų įkėlimo/pasirinkimo veiklos diagrama.

Autentifikavus ir autorizavus vartotoją (gydytoją) jam suteikiama galimybė valdyti ODB esančius duomenis apie pacientą. Pacientų įkėlimo/ pasirinkimo veiklos diagrama nurodo kokius veiksmus turi atlikti vartotojas norėdamas įkelti pacientą į ODB, jei ten jo nėra. Jei pacientas yra ODB, vartotojas pervedamas į sekančią formą – tyrimo įkėlimo/ pasirinkimo. Pacientas tikrinamas įvedus jo ID, ODB asmeniniai duomenys apie pacientą nesaugomi, tik jo lytis ir gimimo data.



11 pav. Tyrimo įkėlimo/ pasirinkimo veiklos diagrama.

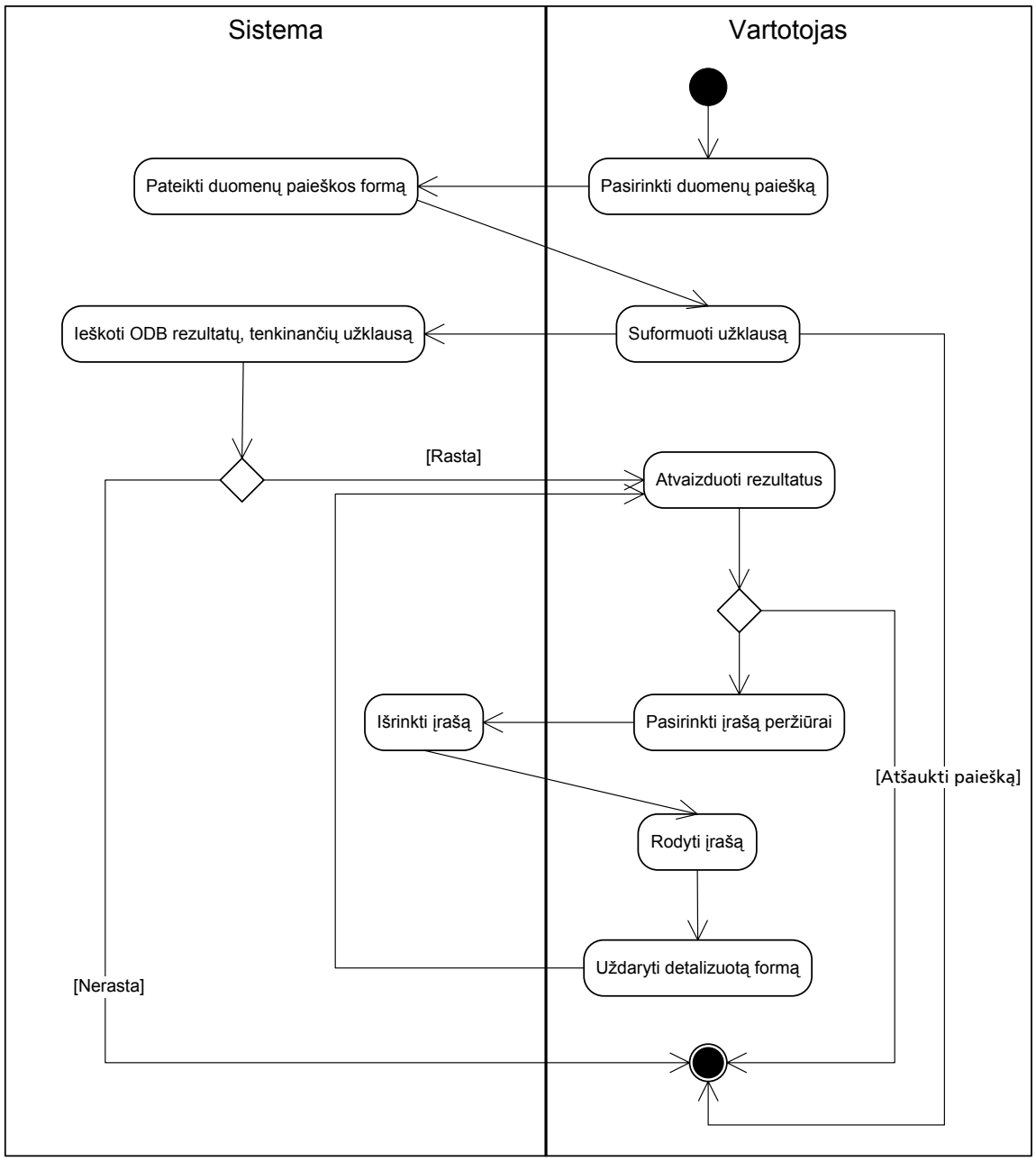
Pacientų įkėlimo/ pasirinkimo veiklos diagrama nurodo kokius veiksmus turi atlikti vartotojas norėdamas įkelti tyrimą į ODB, jei ten jo nėra. Jei tyrimas yra ODB, vartotojas pervedamas į sekančią formą – anamnezės įkėlimo (3 priedo 2 pav.). Jei vartotojas nenori koreguoti ar įvesti šių duomenų, jis gali pasirinkti vaizdų įkėlimo/ pasirinkimo formą (12 pav.). Nepasirinkus paciento, negalima įkelti ar pasirinkti tyrimo. Tyrimas tikrinamas įvedus jo ID, ODB apie tyrimą saugojama jo data.



12 pav. Oftalmologinio vaizdo įkėlimo/ pasirinkimo veiklos diagrama.

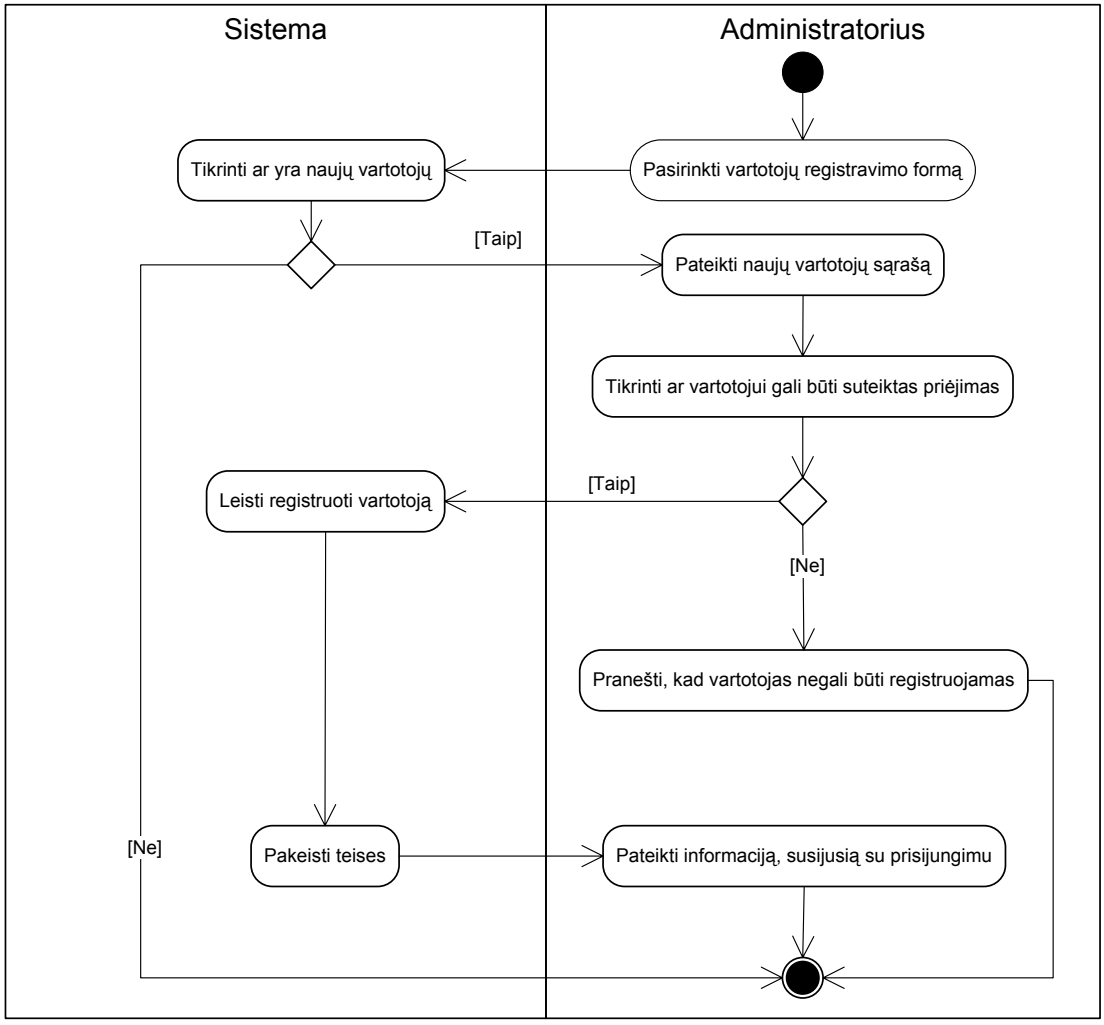
Oftalmologinių vaizdų įkėlimo/ pasirinkimo veiklos diagrama nurodo kokius veiksmus turi atlikti vartotojas norėdamas įkelti vaizdą į ODB. Nepasirinkus paciento ir tyrimo, negalima įkelti ar pasirinkti vaizdo. Jei pacientas ir tyrimas pasirinktas, atidarius formą automatiškai ieškoma šio tyrimo vaizdų. Jos atvaizduojamos sąrašu, atskiriant kairės ir dešinės akių vaizdus, jei vaizdų nėra, pranešama, kad jų nėra. Norint pasirinkti vaizdą, reikia paspausti ant bylos pavadinimo, automatiškai pervedama į vaizdo aprašymo

(peržiūros) formą (3 priedo 3 pav.). Nuspaudus bylos įkėlimo mygtuką tikrinamas bylos plėtinys, jei jis netinkamas, vartotojui pranešama ir prašoma pakartoti procedūrą. Jei byla tinkama, ji įkeliama į ODB ir atvaizduojama bylų sąraše. Vartotojas iš karto gali ją pasirinkti aprašymui. Apie bylą saugoma papildoma informacija (bylos pavadinimas, dydis ir plėtinys) automatiškai generuojama ODB odb4, vartotojas turi pažymėti kurios akies (kairės ar dešinės) yra įkeliamas vaizdas.



13 pav. Duomenų paieškos veiklos diagrama.

Duomenų paieškos veiksmas yra nepriklausomas nuo aukščiau aprašytų kitų veiksmų, vartotojas visada gali pasirinkti šį veiksmą. Duomenų paieškos diagrama nurodo kokius veiksmus atlieka vartotojas vykdydamas paiešką ODBVS. Pasirinkdamas įrašą vartotojas mato visą informaciją esančią ODBVS. Diagramoje nepažymėta, tačiau vartotojas gali atsisiųsti vaizdą susijusį su dominančia informacija.



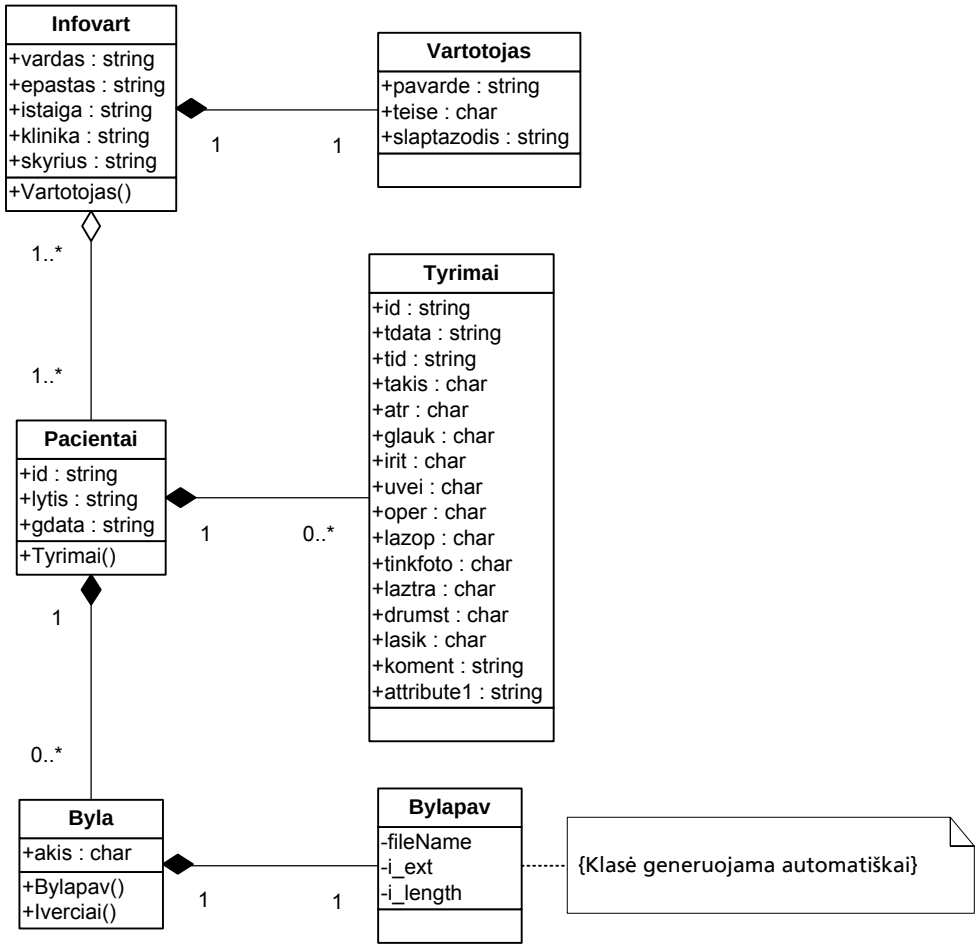
14 pav. Naujų vartotojų registravimo veiklos diagrama.

Vartotojų registravimo veiklos diagrama nurodo kokius veiksmus atlieka administratorius, kuomet užsiregistruoja naujas vartotojas, norintis naudotis sistema. Vartotojų registravimas nedetalizuotas, nes užpildoma forma nėra sudėtinga.

Administratorius turi įsitikinti, kad naujas vartotojas gali prieiti prie pacientų informacijos (pavyzdžiui tikrindamas gydytojo licencijos numerį) ir ją įkelti. Tik tada leisti sistemai sukurti vartotojui priėjimą prie vidinių resursų. Bendravimas su vartotoju numatytas elektroniniu paštu. Registruotiems vartotojams suteikiama galimybė keisti asmeninę informaciją.

5.3. ODB klasės

Apibendrinus vartotojo funkcijas suformuojamos klasių diagramos. Detalizuota klasių diagrama pateikta 15 paveiksle.

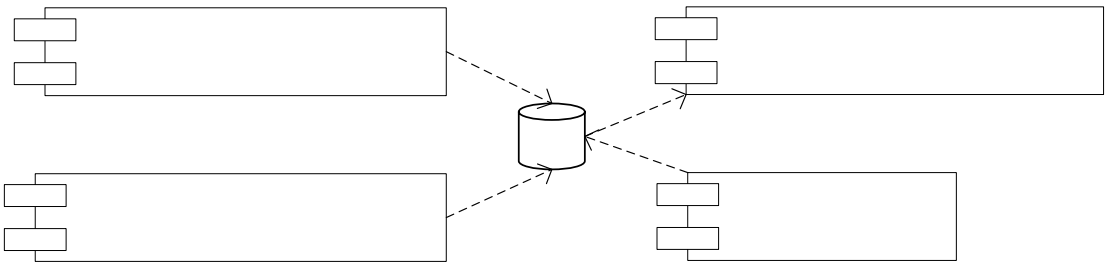


15 pav. IS klasių diagrama.

Klasių diagrama yra ODB modelis, kiekvienos klasės atributas yra ODB sukuriamas atributas objektui saugoti. Klasė *Infovart* agreguoja klasę *Vartotojas*, šios – klasę *Pacientai*, klasė *Pacientai* agreguoja *Tyrimai* ir *Byla*, o *Byla* – automatiškai ODBVS db4o generuojamą klasę *Bylapav*. Klasės *Infovart* ir *Vartotojas* saugo informaciją apie vartotojus, kurie gali jungtis prie sistemos vartotojo teisėmis, administratoriaus teisėmis ir tuos, kurie prašo leidimo būti registruotais sistemos vartotojais. Klasėse *Pacientai* ir *Tyrimai* saugoma informacija apie pacientus ir jiems atliktus tyrimus. Klasė *Byla* skirta pacientams atliktų tyrimų vaizdinės informacijos saugojimui.

5.4. Sistemos komponentai

Medicininė IS susideda iš 4 komponentų, kurie pateikti 16 paveiksle.



16 pav. IS skirstymas į komponentus.

„Medicininių duomenų įvedimo“ komponentas skirtas vartotojui (gydytojui) įvesti informaciją apie pacientą ir jam atliktus oftalmologinius tyrimus. Duomenų įvedimas priklauso tik tam teisę turintiems vartotojams, kurią suteikia sistemos administratorius. Komponentas surenka pateiktą informaciją ir ją patalpina į ODB

„Medicininių duomenų paieškos“ komponentas skirtas vartotojui (gydytojui) kurti jo reikalavimus tenkinančias užklausas ir peržiūrėti ODB esančius duomenis. Komponentas išrenka reikalaujamą informaciją ir ją pateikia vartotojui.

„Medicininių duomenų pateikties“ komponentas skirtas vartotojui (gydytojui) atsisiųsti vaizdinius duomenis.

„Administravimo komponentas“ skirtas IS administravimui. Suteikia galimybę kurti administratorius ir vartotojus.

Medicininių duomenų įvedimo komp.

ODB

Visi komponentai naudojasi ODBVS esančiais resursais.

Sistemos realizacijos metu sukurti visi programiniai komponentai, sukurtos vartotojo sąsajos formos pagal pateiktas 5.2 skyriuje veiklos diagramas, ODB klasės atitinka pateiktas 5.3 skyriuje. Pastebėta, kad dėl netinkamos klasių struktūros sudarymo visiško duomenų dubliavimo nebuvo išvengta. Taip pat įžvelgta keletas naudotos ODBVS db4o trūkumų, kurie buvo paslėpti naudojantis programavimo priemonių galimybėmis.

5.5. Testavimas

Vienas iš svarbiausių medicininės IS reikalavimų yra patikimumas [27]. Kiekvieno programinio modulio veikimas patikrintas naudojantis realiais duomenimis. Testavimui naudojami oftalmologiniai duomenys sukaupti Kauno medicinos universiteto Biomedicininių tyrimų instituto Oftalmologijos laboratorijoje. Šiuose duomenyse asmeninės informacijos apie pacientą nėra, nurodytas tik tam tikras jo ID. Šiais duomenimis užpildyta objektinė DB. Testuojami visi realizuoti sistemos komponentai. Testavimas skirstomas į dvi dalis:

- funkcijų testavimas,
- vartotojo sąsajos testavimas.

Vartotojo funkcijų testavimas

Suformuotos šešios vartotojo funkcijos pagal panaudojimo atvejų modelį (5 pav.):

1. Įkelti/ pasirinkti pacientą
2. Įkelti/ pasirinkti tyrimą
3. Įkelti/ atnaujinti tyrimo duomenis
4. Įkelti/ pasirinkti vaizdą
5. Analizuoti pasirinktą vaizdą
6. Vykdyti paiešką

1-4 punktai apima „Medicininių duomenų įvedimo“ komponentą 5 - „Medicininių duomenų pateikties“, o 6 - „Medicininių duomenų paieškos“. Šių testavimui skirtų funkcijų analizė pateikta 9 lentelėje.

9 lentelė. Vartotojo funkcijų testavimo procedūros.

Nr.	Testavimo procedūra	Tikėtina išeitis	Pastabos	?
1	Įvesti paciento informaciją (ID, lytį, gimimo datą) ir vykdyti.	Paciento informacija įkelta į ODB, pacientas pasirinktas, automatiškai pervedama į tyrimo langą.	Jei paciento ID neįvestas, pranešama vartotojui. Jei pacientas jau yra ODB, tik pervedama į tyrimo langą. Jei paciento nėra ODB ir kiti duomenys neįvesti, pranešama vartotojui.	Taip
2	Įvesti tyrimo informaciją (ID, atlikimo datą) ir vykdyti.	Tyrimo informacija įkelta į ODB, tyrimas pasirinktas, automatiškai pervedama į anamnezės langą.	Jei pacientas nepasirinktas, pranešama vartotojui. Jei tyrimo ID neįvestas, pranešama vartotojui. Jei tyrimas jau yra ODB, tik pervedama į anamnezės langą. Jei tyrimo nėra ODB ir kiti duomenys neįvesti, pranešama vartotojui.	Taip
3	Įvesti anamnezės informaciją (iš iššokančio sąrašo pasirinkti reikšmes) ir vykdyti.	Anamnezės informacija įkelta į ODB, automatiškai pervedama į vaizdo langą.	Jei pacientas ir tyrimas nepasirinktas, pranešama vartotojui. Duomenys įkeliami į ODB, pervedama į vaizdo langą.	Taip
4	Pasirinkti vaizdą iš kompiuteryje esančių bylų ir vykdyti. Pažymėti kurios akies tai vaizdas. Iš sąrašo pasirinkti vaizdą peržiūrai.	Vaizdas įkeliamas į ODB. Vaizdas pasirinktas, automatiškai pervedama į vaizdo peržiūros langą.	Jei pacientas ir tyrimas nepasirinktas, pranešama vartotojui. Jei byla nepasirinka įkėlimui, pranešama vartotojui. Jei bylos netinkamas plėtinys, pranešama vartotojui. Jei pasirenkamas vaizdas, tik pervedama į vaizdo peržiūros langą.	Taip
5	Atvaizduoti pasirinktą vaizdą	Vaizdas matomas naršyklės lange.	Jei vaizdas nepasirinktas, pranešama vartotojui.	Taip
6	Suformuoti užklausą ir ją vykdyti.	Rezultatai tenkinantys užklausą	Jei nėra tenkinančių rezultatų, pranešama vartotojui. Jei užklausa nesuformuota, pranešama vartotojui.	Taip

Pastaba: ? = Ar sėkmingai įvykdyta

Administratoriaus funkcijų testavimas

Pasirinkta viena administratoriaus funkcija pagal panaudojimo atvejų modelį (5 pav.):

1. Suteikti vartotojams priėjimą prie duomenų.

Ši funkcija apima „Administravimo“ komponentą, jo analizė pateikta 10 lentelėje.

10 lentelė. Administratoriaus funkcijų testavimo procedūros.

Nr.	Testavimo procedūra	Tikėtina išeitis	Pastabos	?
1	Pasirinkti naują vartotoją iš sąrašo ir pakeisti jo priėjimo prie sistemos teisę.	Vartotojui pakeista priėjimo prie IS teisė. Naujų vartotojų sąraše jo nėra.	Administratorius šioje vietoje turi tris naujo vartotojo teisių pakeitimo galimybes: suteikti vartotojo ar administratoriaus teisę arba jį pašalinti. Jei pasirinkta teisė yra vartotojo, jam suteikiama vartotojo teisė. Jei pasirinkta teisė yra administratoriaus, jam suteikiama administratoriaus teisė. Jei pasirinkta pašalinti, duomenys pašalinami iš ODBVS.	Taip

Pastaba: ? = Ar sėkmingai įvykdyta

Vartotojo sąsajos testavimas

Vartotojo sąsajos testavimas patiktina ar visi sąsajos elementai veikia teisingai. Šio testavimo veiksmai pateikiami 11 lentelėje.

11 lentelė. Vartotojo sąsajos testavimas.

Nr.	Vartotojo sąsajos elementas	Veiksmai	Pastabos	?
1	Navigacijos nuorodos	Patikrinti ar kiekviena nuoroda veda į tam skirtą internetinį puslapį.		Taip
2	„Vykdyti“ mygtukai	Patikrinti ar teisingai veikia navigacija paspaudus šiuos	Įsitikinti, kad naudojantis šiais mygtukais duomenys teisingai įkeliami,	Taip

		mygtukus kiekvienoje duomenų įkėlimo/ atvaizdavimo formoje.	atnaujinami ir išrenkami iš ODB.	
3	Paieškos rezultatai	Patikrinti ar teisingi paieškos rezultatai.		Taip
4	Vaizdo pasirinkimas	Patikrinti ar sąraše esantys vaizdai tikrai priklauso tam tyrimui ir pacientui.		Taip
5	Vaizdo peržiūra	Patikrinti ar atvaizduojamas pasirinktas vaizdas.		Taip
6	Naršyklės langas	Patikrinti kaip vartotojo sąsajos elementai reaguoja į lango dydžio, rezoliucijos, naršyklės pakeitimus.	Įsitikinti, kad įvykus šiems pakeitimams vartotojo sąsaja nepraranda savo struktūros, patrauklumo.	Taip

Pastaba: ? = Ar sėkmingai įvykdyta

Iš 9-11 lentelėje pateiktų testavimo rezultatų matome, kad procedūros įvykdytos sėkmingai ir sistema veikia patikimai. Kadangi testavimą atliko šios IS kūrėjas, tolimesnis jos testavimas bus perduotas šios sistemos tikriesiems vartotojams – gydytojams oftalmologams. Nors testavimo procedūros įvykdytos teisingai, tačiau pastebėta ir keletas sistemos veikimo trūkumų, kuriuos galima ištaisyti atliekant pakeitimus sistemos programiniame lygyje.

6. Eksperimentas

Eksperimentui naudoti tie patys oftalmologiniai duomenys kaip ir testavimui. Testavimo metu įkelti duomenys buvo pašalinti (ODB byla pašalinta, kartu su įkeltais vaizdais) ir eksperimentui užpildyta pakartotinai. Tuo siekta pašalinti testavimo metu dėl atsiradusių klaidų įkeltus dubliuotus duomenis. Šiuos duomenis sudaro 6 pacientų sveikatos įrašai, kuriuose yra 11 vaizdų su jais lydinčia tekstine informacija. Eksperimento metu siekta įvertinti objektinių technologijų, naudotų IS realizavimui, funkcionalumą. Didžiausias dėmesys skirtas parodyti ODBVS privalumus medicininiams duomenims saugoti. Šiai užduočiai įgyvendinti sukurtas scenarijus realiai egzistuojantis ir praktikoje. Kaip minėta, medicininiai duomenys yra evoliucionuojantys, t.y.:

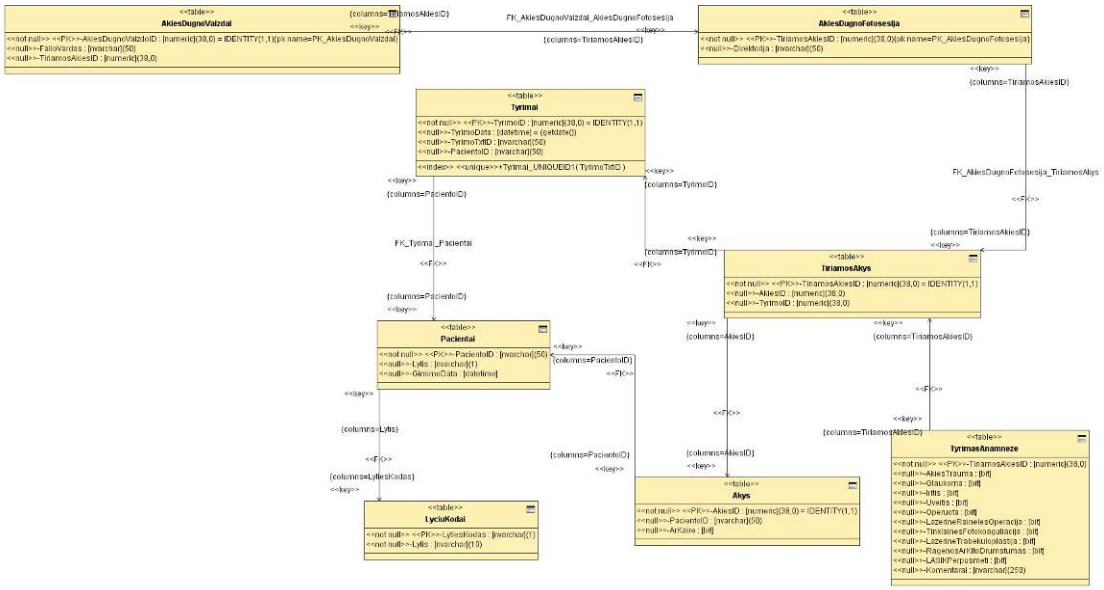
- atsiranda pasikeitimai duomenų struktūrose,
- atsiranda nauji tyrimai ir nauji duomenys, taip pat ryšiai tarp jų,
- atsiranda būtinybė analizuoti duomenis laike, įvertinant ir pasikeitimus.

Šiems reikalavimams įgyvendinti RDB reikėtų nemažai pakeitimų, ypač duomenų atnaujinimo ir papildymo požiūriu. Eksperimentas atliekamas įvykdant scenarijų šio darbo metu sukurtai Medicininių duomenų IS. Realiems duomenims sukurto scenarijaus aprašas:

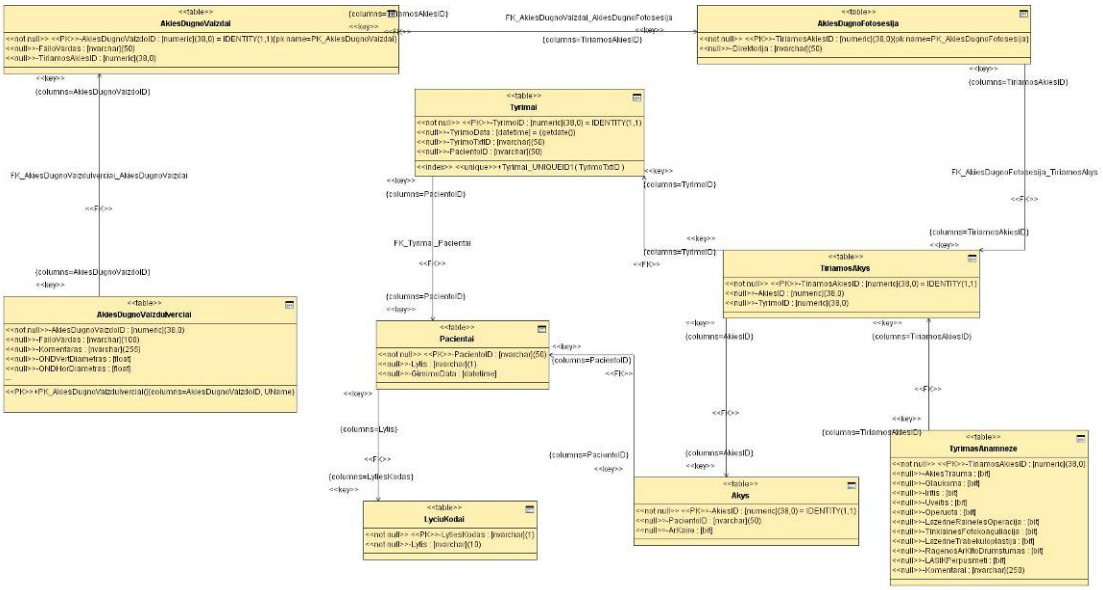
Nufotografavus akies dugno vaizdą, jį galima išsaugoti ir automatizuotai išmatuoti keletą parametrų. Toks vaizdo parametrizavimas yra labai svarbus kliniškai, nes šiuolaikinės technologijos leidžia atlikti objektyvius tikslius matavimus, o gydytojas, nors ir patyręs, įvertina subjektyviai. Atsiradus galimybei saugoti vaizdus struktūrizuotoje duomenų bazėje, iškilo poreikis saugoti ir šiuos parametrus kartu su vaizdais.

Eksperimentui įgyvendinti pasirinkti du svarbiausi akies dugno vaizdo parametrai. Kiti reikalavimai nepateikti, tačiau naudojami duomenys yra ODB esančių vaizdų, susietų su pacientais ir tyrimais, parametrai. Pakeitimus atlikti reikėjo visuose IS lygiuose (vartotojo sąsajos, programos logikos ir duomenų). Taigi, eksperimentas suskirstytas į tris etapus, kiekvienam lygiui atskirai.

Pirmas eksperimento etapas yra atlikti pakeitimus duomenų lygyje. Tam, kad vaizdžiai parodyti šiuos pakeitimus DB, sukurtas RDB modelis tokiam uždaviniui spręsti. 17 paveiksle pateiktas RDB modelis prieš pakeitimus, o 18 – po pakeitimų. Padidinti paveikslai pateikti 4 priede 1 ir 2 paveiksluose.

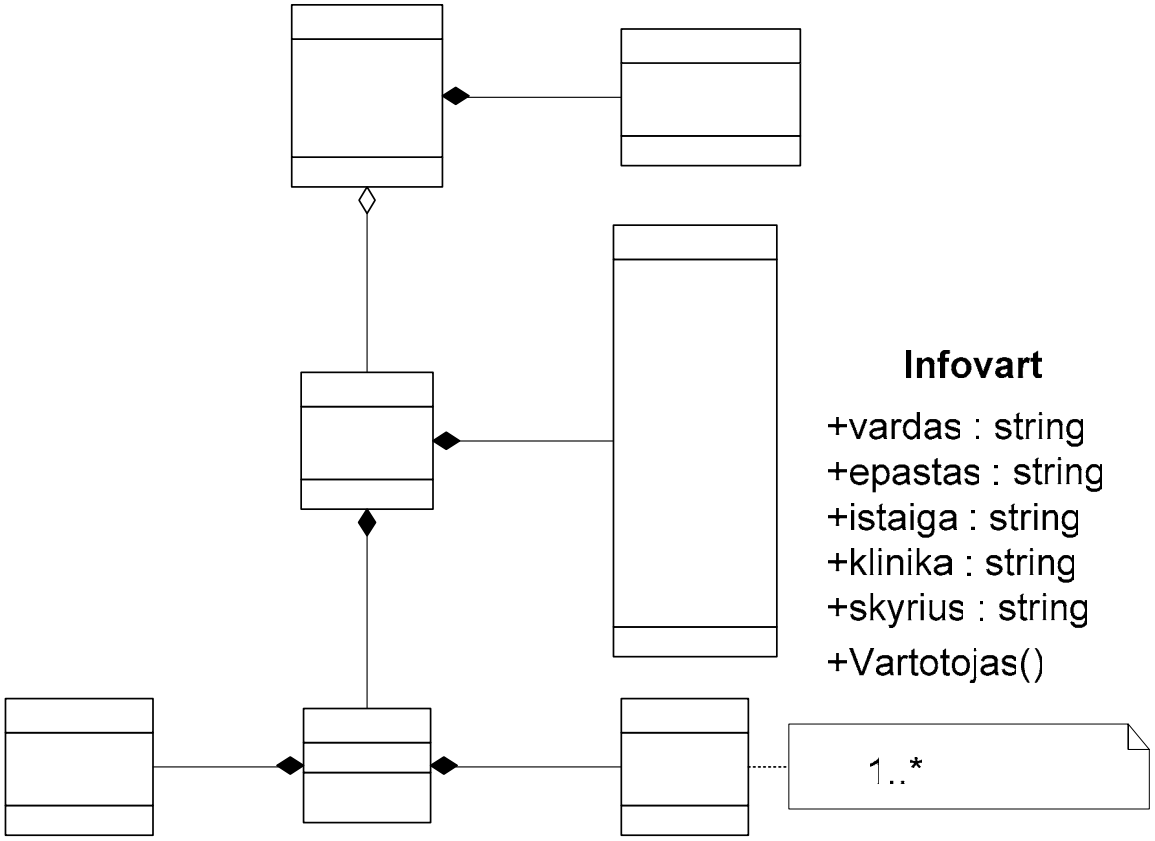


17 pav. RDB modelis prieš pakeitimus.



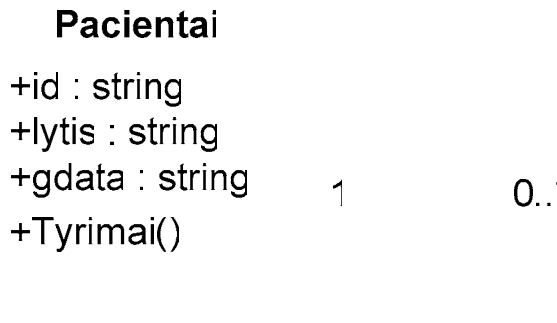
18 pav. RDB modelis po pakeitimų.

ODB klasės prieš pakeitimus pateiktos 15 paveiksle, o po pakeitimų – 19.



19 pav. ODB klasės po pakeitimų.

Kad būtų aiškiau DB po pakeitimų pavaizduotos kartu 20 paveiksle.



a) b)

20 pav. DB po pakeitimų: a) RDB, b) ODB.

Iš 20 a paveikslo matome, kad egzistuoja ryšiai tarp lentelių, kurie gaunami įkeliant nesikartojančius duomenis į lenteles – pirminius raktus. ODB ryšys sukuriamas naudojat adresinę nuorodą, kuri gaunama įrašius operaciją į kitą klasę. Nauja klasė *Iverciai* prijungta prie klasės *Byla*, todėl kad nauji duomenys yra tiesiogiai susiję su vaizdais. Klasėje *Byla* sukurta operacija *Iverciai*(), kuri sukuria nuorodą į klasę *Iverciai* (20b pav.), taip pat sukurta pati klasė su trimis atributais. Duomenys nedubliuojami, taigi nėra ir tokios sudėtingos struktūros kaip 18 paveiksle, kuriame atvaizduota RDB struktūra. Lyginant 18 ir 19 paveikslus galima pastebėti, kad ODB paprasčiau saugo vaizdinę informaciją – nereikia saugoti bylos kelio, kuriame yra saugoma pati byla. Šis kelias ODB nurodomas programiškai vieną kartą, ir nereikia papildomai kreipinių pasiekiant bylas. Taip pat pastebėtina, kad ODB nereikia lentelės *AkiesDugnoVaizdai* (20a pav.), kadangi šią informacija ODB generuojama automatiškai (19 pav.). Pakanka sukurti specialaus plėtinio operaciją (*Blob Bylapav*()) bylos pavadinimui ir kitai informacijai saugoti (19, 20 pav.). ODBVS taip pat automatiškai keičia bylų pavadinimus, taigi programuotojui nereikia papildomai atlikti jokių su pervadinimu susijusių veiksmų. Taigi, sumažėja programinio kodo kiekis.

Antras eksperimento etapas yra aukščiau aprašytų pakeitimų įgyvendinimas vartotojo sąsajos lygyje. Vartotojo sąsajos pakeitimai dirbant su RDB ir ODB yra ekvivalentūs, kadangi atvaizduojami tie patys duomenys. Pirmiausia įvedama nauja nuoroda

navigacijos struktūroje, po to išdėstomi sąsajos elementai. Šie pakeitimai pavaizduoti 21 paveiksle.

21 pav. Vartotojo sąsajos pakeitimai.

Pastaba: navigacijos medis sukurtas sistemos realizacijos metu, tačiau eksperimento rezultatams tai įtakos neturi, kadangi neužpildytoms nuorodoms nepateikti reikalavimai.

Trečias eksperimento etapas yra pakeitimai programiniame lygyje. ODB modeliui įgyvendinti šie pakeitimai:

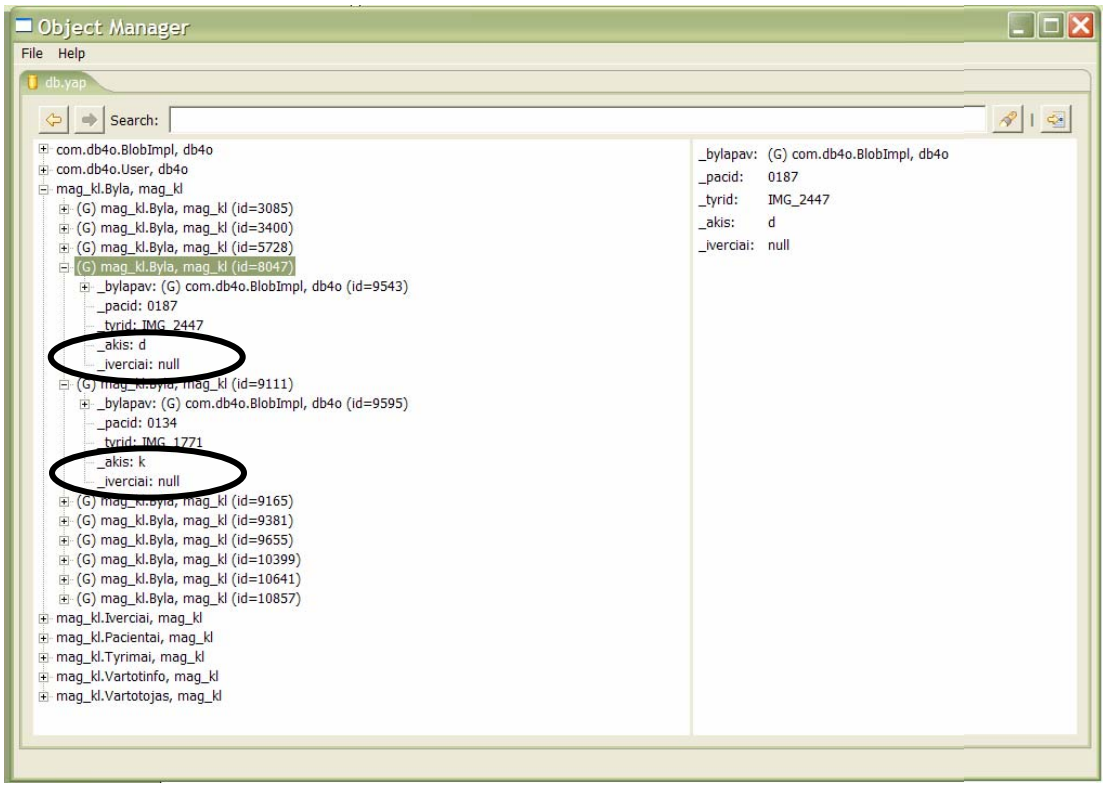
- sukurtas programinis kodas naujiems vartotojo sąsajos elementams,
- atliktas testavimas šiems elementams, naudojant papildomus duomenis,
- realizuotas naujų duomenų įkėlimas.

RDB modeliui aprašytos tik operacijos su duomenimis. Pirmiausia atlikti minėti pakeitimai ODB sistemai. Sukurtas papildomas programinis kodas šio eksperimento metu neanalizuojamas, vėliau bus aptartos operacijos su duomenimis abiem atvejams. Tačiau, prieš tai parodoma ODB elgsena atsiradus naujai klasei. Paveikslai šiam procesui vaizdžiai parodyti sukurti naudojant ODBVS db4o duomenų peržiūros ir išrinkimo programą „Object Manager“. Šiuo konkrečiu atveju ODB įvyksta tokie veiksmai:

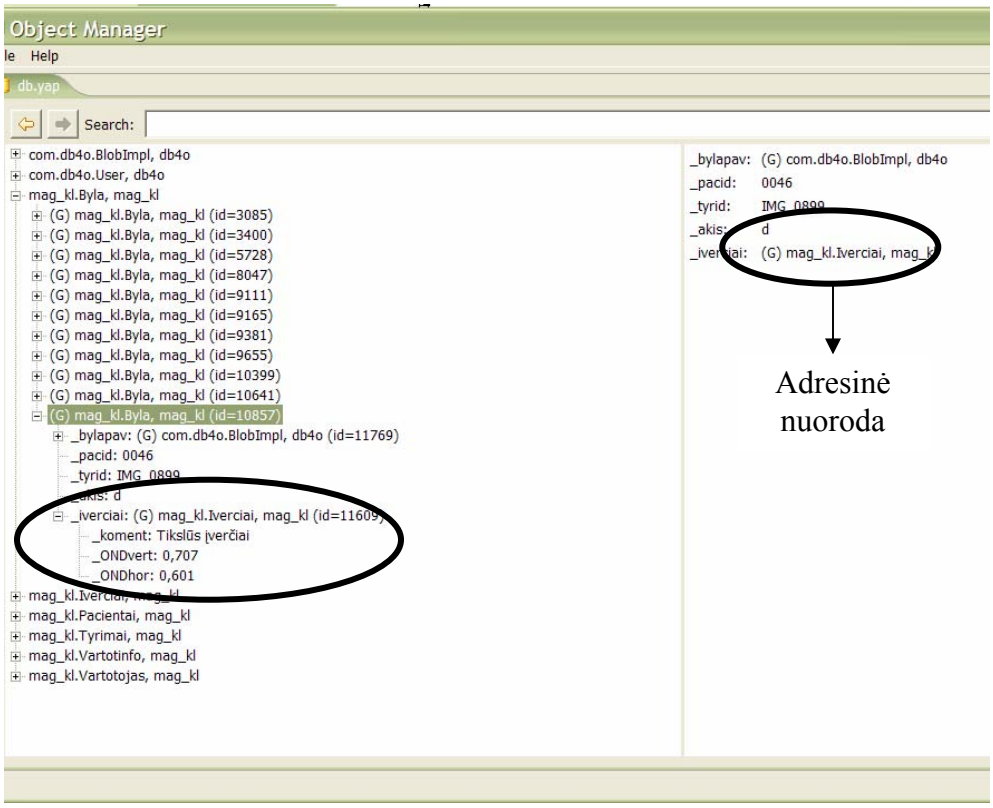
- transakcijos pradžia
  - atidaroma ODB byla (*db.yap*),
  - vaizdinių duomenų skaitymo metu iškviečiamas objektas Byla,

- į šį objektą įkeliamas naujas atributas *iverciai* (22 pav.),
- pagal nutylėjimą jo reikšmė priskiriama *null* (22 pav.),
- naujame vartotojo sąsajos lange užpildomi laukai naujais duomenimis,
- duomenų įkėlimo metu sukuriamas naujas objektas *Iverciai* su trimis atributais,
- atribute *iverciai* sukuriama adresinė nuoroda į objektą *Iverciai* (23 pav.),
- atributai užpildomi naujais duomenimis (23 pav.).

- transakcijos pabaiga



22 pav. Naujas atributas *iverciai* su reikšme *null*.



23 pav. Adresinė nuorodą į objektą su naujais duomenimis.

Matome, kad ODB reikia atlikti pakeitimus klasėse programiniame lygyje, DB sužadinimo metu struktūros yra atnaujinamos automatiškai. RDB struktūra keičiama kitais būdais, o programiniame lygyje SQL sakiniais duomenys įkeliami ir atnaujinami. Tačiau duomenų įkėlimui ir atnaujinimui naudojami skirtingi kreipiniai: *Insert Into* ir *Update*. ODB šiems veiksmams atlikti naudojama tik viena operacija – *set*(). Tarkim, kad turim sukurtą RDB pagal 18 paveiksle pateiktą modelį ir vartotojo sąsajos langą sukurtą pagal 21 paveikslą. RDB užpildyta duomenimis, išskyrus lentelę *AkiesDugnoVaizduIverciai*. RDB pagal pasirinktą pacientą, tyrimą ir vaizdą reikia išrinkti *AkiesDugnoVaizdoID* ir įkelti naujus duomenis. Taip atrodytų mysql kodu užrašytas sakinys:

```
$sql = mysql_query ("INSERT INTO AkiesDugnoVaizduIverciai (AkiesDugnoVaizdoID, Failovardas, Komentaras, ONDVertDiametras, ONDHorDiametras) VALUES ('…')");
```

Tačiau pakartotinai vykdant operaciją, jei norime tuos pačius dugno įverčius atnaujinti, reikės naudoti papildomai keletą operacijų: išrinkti *AkiesDugnoVaizdoID* ir ieškoti ar pagal šį ID yra įrašų. Jeigu įrašų nėra atlikti anksčiau pavaizduotą operaciją. Jeigu yra, reikės esančius duomenis atnaujinti. Taip atrodytų mysql kodu užrašytas sakinys:

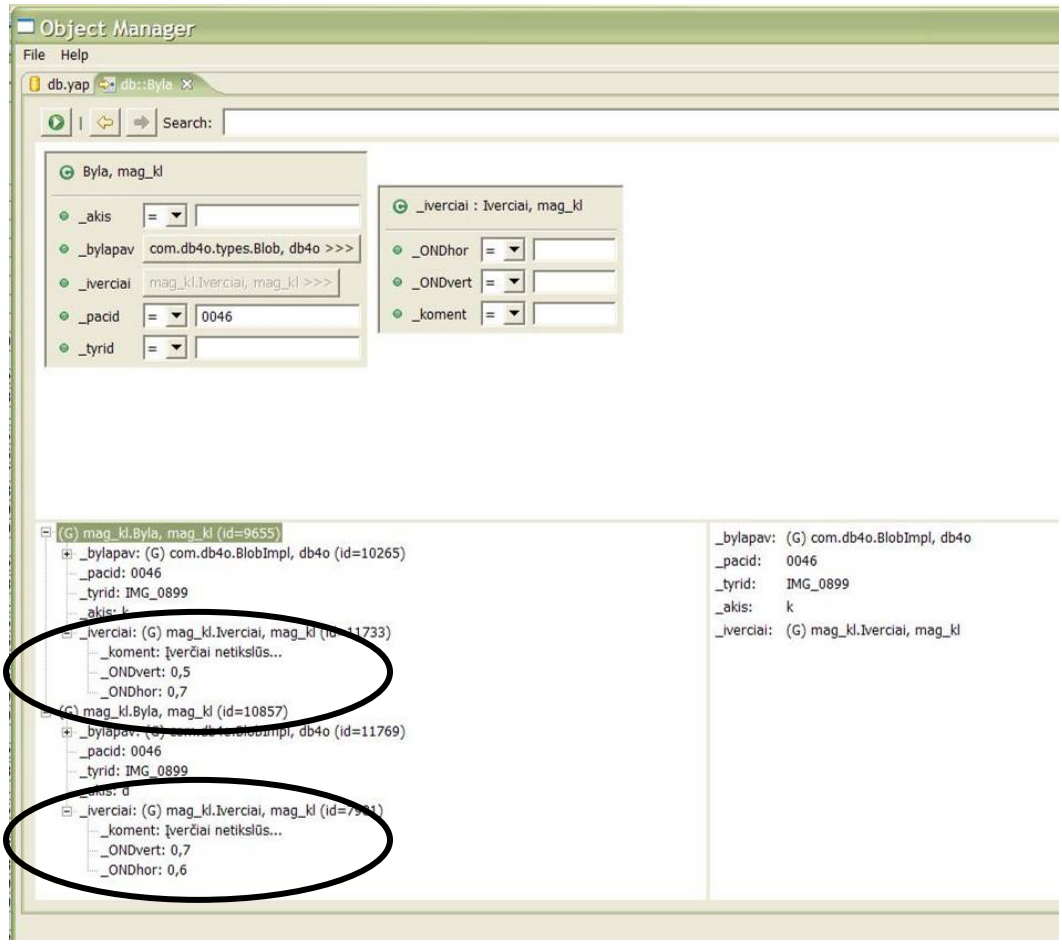
```
$sql = mysql_query ("UPDATE AkiesDugnoVaizduIverciai SET Failovardas ='$...',
Komentaras ='$...', ... WHERE AkiesDugnoVaizdoID =$AkiesDugnoVaizdoID ");
```

Pilnas programinis kodas susidės iš duomenų išrinkimo, dviejų minėtų (pavyzdinių) SQL užklausos sakinių ir loginio šių užklausų atskyrimo.

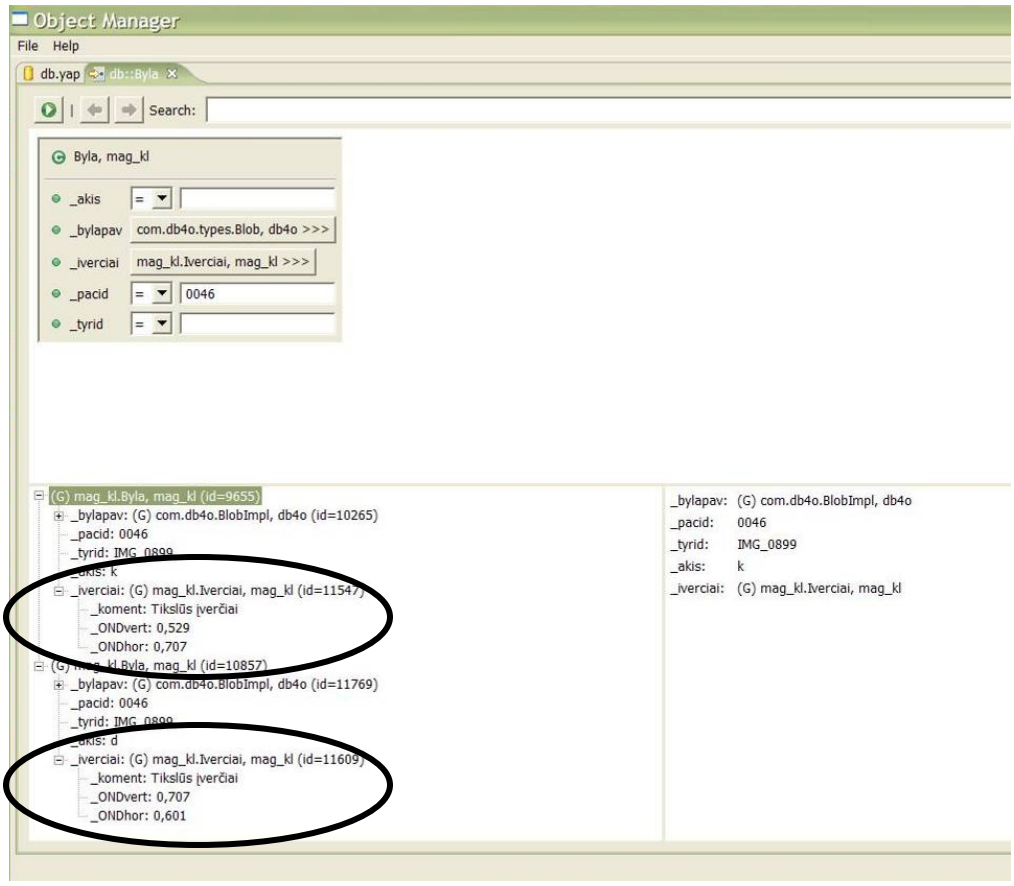
ODB pagal pasirinktą pacientą, tyrimą ir vaizdą reikia išrinkti vietą, kurioje įkelsime naujus duomenis. Kaip jau minėta duomenų įkėlimui ir atnaujinimui ODB naudoja tą pačią operaciją, taigi nereikia papildomo programinio kodo dviems užduotims įgyvendinti. Eksperimento metu sukurtas programinis kodas šioms užduotims įgyvendinti (C# kalbos kodas):

```
ObjectSet result1 = client.get(byl1);     //išrenkame atnaujintiną vietą
if (result1.size()>0)                     //jei vieta rasta
{
        Byla found1 = (Byla)result1.next();     //pasirenkam pirminį objektą
        found1.Iverciai = new Iverciai(kom.Value,kvert.Text,khor.Text);
                                          //adresine nuoroda priskiriam naujus duomenis
        client.set(found1);               //duomenis atnaujiname
}                                         //jei vieta nerasta – vaizdų šiaip akiai nėra
```

Matome, kad vienintelis reikalingas sakinys dviems operacijoms atlikti yra *client.set(found1)*. Pateiktas pilnas kodas yra trumpas ir suprantamas. Taigi naudojant ODB sumažėja ne tik kodo kiekis, bet už ir klaidų tikimybė. Eksperimento metu atliktas duomenų įkėlimas ir atnaujinimas. Užklausos naujiems duomenims vaizdžiai parodyti atliktos naudojantis „Object Manager“ programa. Rezultatai patiekti 24 ir 25 paveiksluose.



24 pav. Užklausos rezultatai po netikslaus duomenų įkėlimo.



25 pav. Užklausos rezultatai po duomenų patikslinimo.

Iš šių paveikslų matome, kad duomenys ODB atnaujinti sėkmingai, nekeičiant ir nepildant programinio kodo.

Sistemai sukurtas vartotojo vadovas (5 priedas) yra pilnas esamai sistemai, įtraukti eksperimento metu atlikti pakeitimai.

Šie pakeitimai kaip ir visa sistema realizuoti Visual Studio .NET programinėje aplinkoje. Šis paketas yra patogus ir tinkamas dirbti du pasirinkta ODB, kadangi juo paprasta kurti ir modifikuoti ODB klases.

7. Išvados

1. Medicinos informacijos sistemos yra specifinės dėl kintančių, tarpusavio ryšiais susietų duomenų.
2. Šiuolaikiniais medicinos duomenų saugojimo standartais siekiama užtikrinti patogų ir vieningą bendravimą tarp skirtingomis technologijomis kurtų medicininės paskirties IS.
3. Objektinių technologijų analizė parodė .NET technologijos privalumus įvairios paskirties sistemoms kurti. Ši technologija geresnė nei J2EE dėl plėtojimo, daugiakalbiškumo galimybės, patogaus programavimo aplinkos integruotumo.
4. Daugumai nemokamų objektinių duomenų bazių rasti trūkumai, todėl pasirinkta db4o ODBVS, skirta darbui su .NET ir J2EE.
5. Darbo metu suprojektuota ir realizuota IS, skirta medicininiams-oftalmologiniams duomenims ir vaizdams saugoti. Tokio pobūdžio IS šioje srityje dar nebuvo sukurta.
6. Tyrimo metu palyginta ODB ir RDB, analizuojant ir atliekant eksperimentą su kintančių medicininių duomenų saugojimo pavyzdžiu, ir nustatyta, kad pagrindiniai ODB privalumai prieš RDB yra šie:
    a. galimybė įkelti/atnaujinti duomenis ta pačia komanda,
    b. sumažėja programinio testo ir klaidos tikimybės,
    c. esant adresinėms nuorodoms tarp objektų nereikia kurti papildomų laukų ryšiams tarp lentelių palaikyti,
    d. duomenys nedubliuojami,
    e. lankstesnis vaizdinės informacijos saugojimo mechanizmas,
    f. pildant DB nereikia kurti ryšių ir lentelių duomenų bazės lygyje.
7. .NET programavimo aplinkoje sėkmingai atlikti visi realizacijos darbai. Ši aplinka pasirodė tinkama darbui su ODB, nes joje patogu dirbti su ODB klasėmis.
8. Analizės, tyrimo ir realizacijos metu nustatyta, kad šios technologijos yra tinkamos automatinių algoritmų įgyvendinimui ir klinikinių sprendimų palaikymui.

9. Šia tema parengtas straipsnis „The Use Of Object Oriented Technologies For Medical Data Storing And Retrieving“ spausdinti žurnale „Information Technology and Control“ (6 priedas).

8. Literatūra

1. Mikel Sorli, Jesus Romo, Rene Stach, Bernd Bredehorst. *REMOTE: GRD1-2000-25433*. 2001, p. 40-42. [žiūrėta 2005-07-27]. Prieiga per internetą: www.remote-project.org/downloads/REMOTE.D1.1-1.1--2001-07-09.pdf
2. Saba Zamir. *Handbook of object technology*. CRC Press, 1999 – 35skr.
3. Object Database Management Group [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.odmg.org
4. Rick Grehan. *When to Use an ODBMS*. 2005. [žiūrėta 2005-09-23].Prieiga per internetą: http://www.odbms.org/introduction_whenODBMS.html
5. ODBVS db4o [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.db4o.com/
6. ODBVS SOD2 [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.tneumann.de/sod/index.html
7. ODBVS ozone [žiūrėta 2005-02-27]. Prieiga per internetą: http://ozone-db.org/frames/home/what.html
8. ODBVS ObjectDB [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.objectdb.com/
9. O-RDBVS GigaBASE [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.garret.ru/~knizhnik/gigabase.html
10. ODBVS XL2 [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.xl2.net/
11. HL7 svetainė [žiūrėta 2005-02-27]. Prieiga per internetą:http://www.hl7.org/
12. DICOM svetainė [žiūrėta 2005-02-27]. Prieiga per internetą: http://medical.nema.org/
13. CEN/TC 251 svetainė [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.centc251.org/
14. Sveikatos apsaugos ministerijos svetainė [žiūrėta 2001-10-16] . Prieiga per internetą: http://www.sam.lt/images/Dokumentai/eSveikata/
15. Jonathan Lurie, R. Jason Belanger. *Does one Web services platform dominate the other?*. 2002. [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.javaworld.com/javaworld/jw-03-2002/jw-0308-j2eenet.html
16. Jim Farley. *Microsoft .NET vs. J2EE: How Do They Stack Up?*. 2000. [žiūrėta 2005-02-27]. Prieiga per internetą: http://java.oreilly.com/news/farley_0800.html
17. Byron Taylor Estes, Oriel Maxime. *J2EE vs .Net: The choice depends on your needs*. 2003. [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.computerworld.com/printthis/2003/0,4814,84155,00.html
18. Roger Sessions. *Java 2 Enterprise Edition (J2EE) versus The .NET Platform Two Visions for eBusiness*. 2001. [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.objectwatch.com/FinalJ2EEandDotNet.doc
19. J. Jeffrey Hanson. *.NET versus J2EE Web Services. A Comparison of Approaches*. [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.webservicesarchitect.com/content/articles/hanson01.asp
20. Humphrey Sheil, Michael Monteiro. *Rumble in the jungle: J2EE versus .Net. How do J2EE and Microsoft's .Net compare in enterprise environments?*. 2002. [žiūrėta 2005-02-27]. Prieiga per internetą: http://www.javaworld.com/javaworld/jw-06-2002/jw-0628-j2eevsnet_p.html

21. MediPas sistemos svetainė [žiūrėta 2005-02-27]. Prieiga per internetą: http://tmc.kmu.lt/medipas.htm
22. DICOM specifikacija, http://medical.nema.org/dicom/2004.html, žr. PS3.4, Annex C.
23. Joseph A. Maldjian, John Listerud, Satjeet Khalsa. *Integrating Postprocessed Functional MR Images with Picture Archiving and Communication Systems*. American Journal of Neuroradiology 23:1393-1397, 2002 09 [žiūrėta 2005-06-02]. Prieiga per internetą: http://www.ajnr.org/cgi/content/full/23/8/1393
24. Kompanijos SECTRA svetainė. [žiūrėta 2005-06-02]. Prieiga per internetą: http://www.sectra.se/medical/
25. Thomas M Lehmann, Berthold B Wein, Hayit Greenspan. *Integration of Content-based Image Retrieval to Picture Archiving and Communication Systems*. 2003. [žiūrėta 2005-06-02]. Prieiga per internetą: http://libra.imib.rwth-aachen.de/irma/ps-pdf/MIE_2003-final.pdf
26. Simona Cohen, Flora Gilboa, Uri Shani. *PACS and Electronic Health Records.* Konferencija "Medical Imaging 2002" [San Diego, Kalifornija, JAV, 2002 02 23-28]. [žiūrėta 2005-06-02]. Prieiga per internetą: www.haifa.il.ibm.com/projects/software/imr/papers/mi4685-40.pdf
27. Vaidotas Marozas, Rytis Jurkonis, Algirdas Kazla, Mantas Lukoševičius, Arūnas Lukoševičius, Adas Gelžinis, Darius Jegelevičius. *Development of Teleconsultations Systems for e-Health*. IOS Press, 2004 – 337-348psl.

9. Sutrumpinimai

.NET – Microsoft objektinio programavimo technologija

ADO.NET – Active Data Objects; .NET priėjimas prie RDB

ADSI – Active Directory Services Interface; .NET komponentų registravimo technologija

ASP.NET – Active Server Pages; .NET vartotojo sąsajos programavimo priemonė

AWT – Abstract Windowing Toolkit; Java vartotojo sąsajos programavimo priemonė

CLR – Common Language Runtime; speciali .NET programa-interpretatorius

COM+ – .NET tarpininkavimo technologija

DB – duomenų bazė

DBVS – duomenų bazių valdymo sistema

DICOM – Digital Imaging and Communications in Medicine; duomenų mainų standartas

EJB – Enterprise Java Beans; Java tarpininkavimo technologija

ESI – elektroninė sveikatos istorija

HTML – Hypertext Markup Language – hiperteksto kūrimo kalba

IS – informacijos sistema

J2EE – Java 2 Platform, Enterprise Edition; objektinio programavimo standartas

JDBC – Java Database Connectivity; Java priėjimas prie RDB

JMS – Java Messaging Service; Java užklausų valdymo standartas

JNDI – Java Naming and Directory Interface; Java komponentų registravimo technologija

JRE – Java Runtime Environment – Java abstrakti virtualioji mašina

JSP – Java server pages; Java vartotojo sąsajos programavimo priemonė

JVM – Java virtualioji mašina

JWSDP – Java Web Services Developer Pack; Java tinklo paslaugų kūrimo įrankis

KS – klinikiniai sprendimai

LB – Language Binding – kalbų sąsajos specifikacija

MD5 – duomenų kodavimo algoritmas

MSIL (IL) – Microsoft Intermediate Language – Microsoft tarpinė kalba

MSMQ – Microsoft Message Queue; Microsoft užklausų valdymo standartas

ODB – objektinė duomenų bazė

ODBVS – objektinė duomenų bazių valdymo sistema

ODL – Object definition language – objektų aprašymo kalba

ODMG – Object Database Management Group; darbo grupė, valdanti ODB standartus

OM – Object Model – objektų modeliu

OQL – Object Query Language – objektų užklausos kalba

O-RDBVS – objektinė-reliacinė duomenų bazių valdymo sistema

OSI – Open System Interconnection

PACS – Picture Archive and Communication Systems; medicininių duomenų saugojimo sistemos

PĮ – programinė įranga

RDB – reliacinė duomenų bazė

TĮ – techninė įranga

10. Priedai

10.1. Priedas. CEN/TC 251 paskelbti standartai

1 lentelė. CEN/TC 251 paskelbti standartai [17]

Standarto charakteristika	Pavadinimas originalo kalba
CEN/TS 14271:2003	Health informatics - File exchange format for vital signs
CEN/TS 14463:2003	Health informatics - A syntax to represent the content of medical classification systems (CIaML)
CEN/TS 14796:2004	Health Informatics - Data Types
CR 12069:1995	Profiles for medical image interchange
CR 12161:1995	A method for defining profiles for healthcare
CR 12587:1996	Medical Informatics - Methodology for the development of healthcare messages
CR 12700:1997	Supporting document to ENV 1613:1994 - Messages for Exchange of Laboratory Information
CR 1350:1993	Investigation of syntaxes for existing interchange formats to be used in health care
CR 13694:1999	Health Informatics - Safety and Security Related Software Quality Standards for Healthcare (SSQS)
CR 14300:2002	Health Informatics - Interoperability of healthcare multimedia report systems
CR 14301:2002	Health informatics - Framework for security protection of healthcare communication
CR 14302:2002	Health informatics - Framework for security requirements for intermittently connected devices
EN 14484:2003	Health informatics - International transfer of personal health data covered by the EU data protection directive - High level security policy
EN 14485:2003	Health informatics - Guidance for handling personal health data in international applications in the context of the EU data protection directive
EN 1828:2002	Health informatics - Categorial structure for classifications and coding systems of surgical procedures
EN ISO 18104:2003	Health Informatics - Integration of a reference terminology model for nursing (ISO 18104:2003)
EN ISO 18812:2003	Health informatics - Clinical analyser interfaces to laboratry information systems - Use profiles (ISO 18812:2003)
EN ISO 21549-1:2004	Health informatics - Patient healthcard data - Part 1: General structure (ISO 21549-1:2004)
EN ISO 21549-2:2004	Health informatics - Patient healthcard data - Part 2: Common objects (ISO 21549-2:2004)
EN ISO 21549-3:2004	Health informatics - Patient healthcard data - Part 3: Limited clinical data (ISO 21549-3:2004)

ENV 1064:1993	Medical informatics - Standard communication protocol - Computer-assisted electrocardiography
ENV 1068:1993	Medical informatics - Healthcare information interchange - Registration of coding schemes
ENV 12017:1997	Medical Informatics - Medical Informatics Vocabulary (MIVoc)
ENV 12018:1997	Identification, administrative, and common clinical data structure for Intermittently Connected Devices used in healthcare (including machine readable cards)
ENV 12251:2000	Health informatics - Secure user identification for health care - Management and security of authentication by passwords
ENV 12264:1997	Medical informatics - Categorical structures of systems of concepts - Model for representation of semantics
ENV 12381:1996	Health care informatics - Time standards for healthcare specific problems
ENV 12388:1996	Medical Informatics - Algorithm for Digital Signature Services in Health Care
ENV 12435:1999	Medical informatics - Expression of the results of measurements in health sciences
ENV 12443:1999	Medical Informatics - Healthcare Information Framework (HIF)
ENV 12537-1:1997	Medical informatics - Registration of information objects used for EDI in healthcare - Part 1: The Register
ENV 12537-2:1997	Medical informatics - Registration of information objects used for EDI in healthcare - Part 2: Procedures for the registration of information objects used for electronic data interchange (EDI) in healthcare
ENV 12538:1997	Medical informatics - Messages for patient referral and discharge
ENV 12539:1997	Medical informatics - Request and report messages for diagnostic service departments
ENV 12610:1997	Medical informatics - Medicinal product identification
ENV 12611:1997	Medical informatics - Categorial structure of systems of concepts - Medical devices
ENV 12612:1997	Medical informatics - Messages for the exchange of healthcare administrative information
ENV 12924:1997	Medical Informatics - Security Categorisation and Protection for Healthcare Information Systems
ENV 12967-1:1998	Medical informatics - Healthcare Information System Architecture (HISA) - Part 1: Healthcare Middleware Layer
ENV 13606-1:2000	Health informatics - Electronic healthcare record communication - Part 1: Extended architecture
ENV 13606-2:2000	Health informatics - Electronic healthcare record communication - Part 2: Domain term list

ENV 13606-3:2000	Health informatics - Electronic healthcare record communication - Part 3: Distribution rules
ENV 13606-4:2000	Health informatics - Electronic healthcare record communication - Part 4: Messages for the exchange of information
ENV 13607:2000	Health informatics - Messages for the exchange of information on medicine prescriptions
ENV 13608-1:2000	Health informatics - File exchange format for vital signs
ENV 13608-2:2000	Health informatics - A syntax to represent the content of medical classification systems (CIaML)
ENV 13608-3:2000	Health Informatics - Data Types
ENV 13609-2:2000	Profiles for medical image interchange
ENV 13729:2000	A method for defining profiles for healthcare
ENV 13730-1:2001	Medical Informatics - Methodology for the development of healthcare messages
ENV 13730-2:2002	Supporting document to ENV 1613:1994 - Messages for Exchange of Laboratory Information
ENV 13734:2000	Investigation of syntaxes for existing interchange formats to be used in health care
ENV 13735:2000	Health Informatics - Safety and Security Related Software Quality Standards for Healthcare (SSQS)
ENV 13940:2001	Health Informatics - Interoperability of healthcare multimedia report systems
ENV 1613:1995	Health informatics - Framework for security protection of healthcare communication
ENV 1614:1995	Health informatics - Framework for security requirements for intermittently connected devices

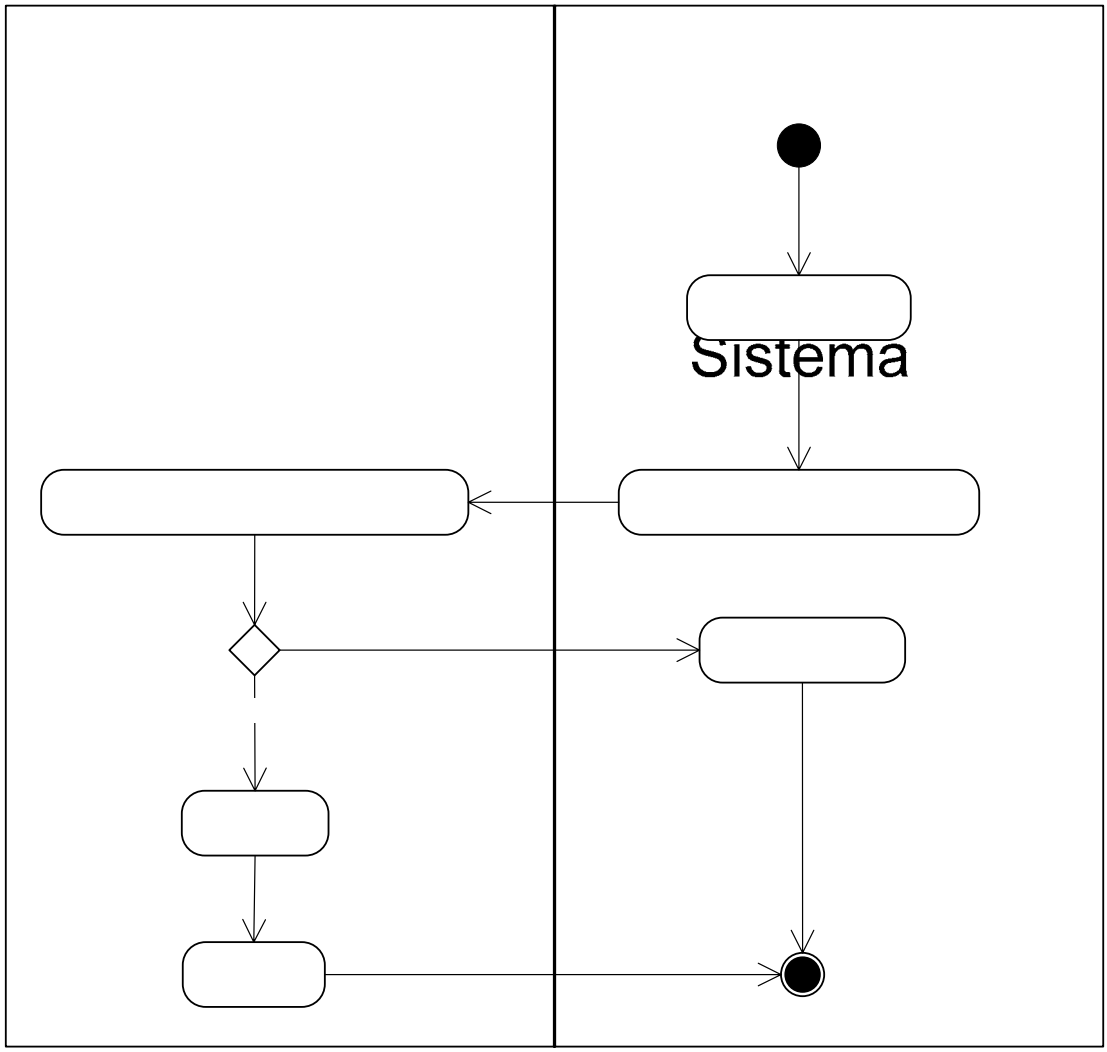
10.2. Priedas. Oftalmologinius vaizdus fotografuojančios ir saugančios sistemos

1 lentelė. Kompanijos siūlančios oftalmologinius vaizdus fotografuojančias ir saugančias sistemas

Gamintojas	Produktas/IS	Prieiga per internetą
Ophthalmic Imaging Systems	Ophthalmology Office	www.oisi.com
Topcon Medical Systems	IMAGEnet Professional	www.topcon.com
Zeiss	VISUPAC software	www.meditec.zeiss.com
Clarity Medical Systems, Inc.	RetCam	www.claritymsi.com www.retcam.com
Heidelberg Engineering	HRT3	www.heidelbergengineering.com
Einstein Medical	Einstein Medical (interneto paslaugų tiekėjas oftalmologams)	www.einsteinmedical.com
First Insight	Maxim Eyes	www.first-insight.com
Innovative Imaging	I3SYSTEM-ABD	www.eye-imaging.com
ManagementPlus	ManagementPlus	www.managementplus.com
Medflow	MDOffice	www.mdoffice.com
MDTeknix	eyeTeknix	www.mdteknix.com
Medibell Medical Vision Technologies	Panoret	www.medibell.com
Medical 101 Websites	Medical 101	www.medical101.com
NextGen Healthcare Information Systems	NextGen	www.nextgen.com
TelScreen	EyeRes System	www.telscreen.com

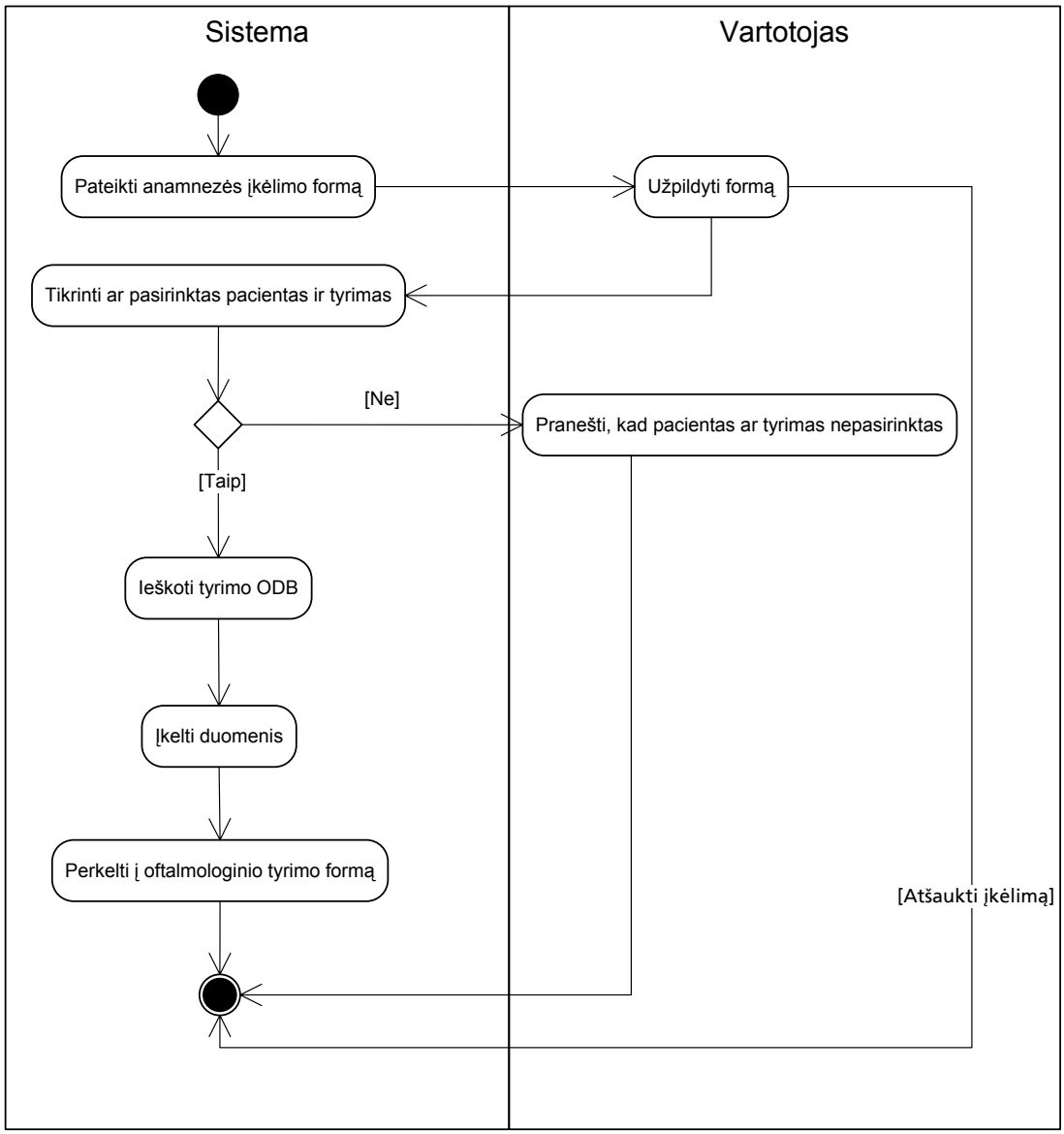
Informacija iš konferencijos American Academy of Ophthalmology, Anual Meeting, 2005.

10.3. Priedas. Veiklos diagramos



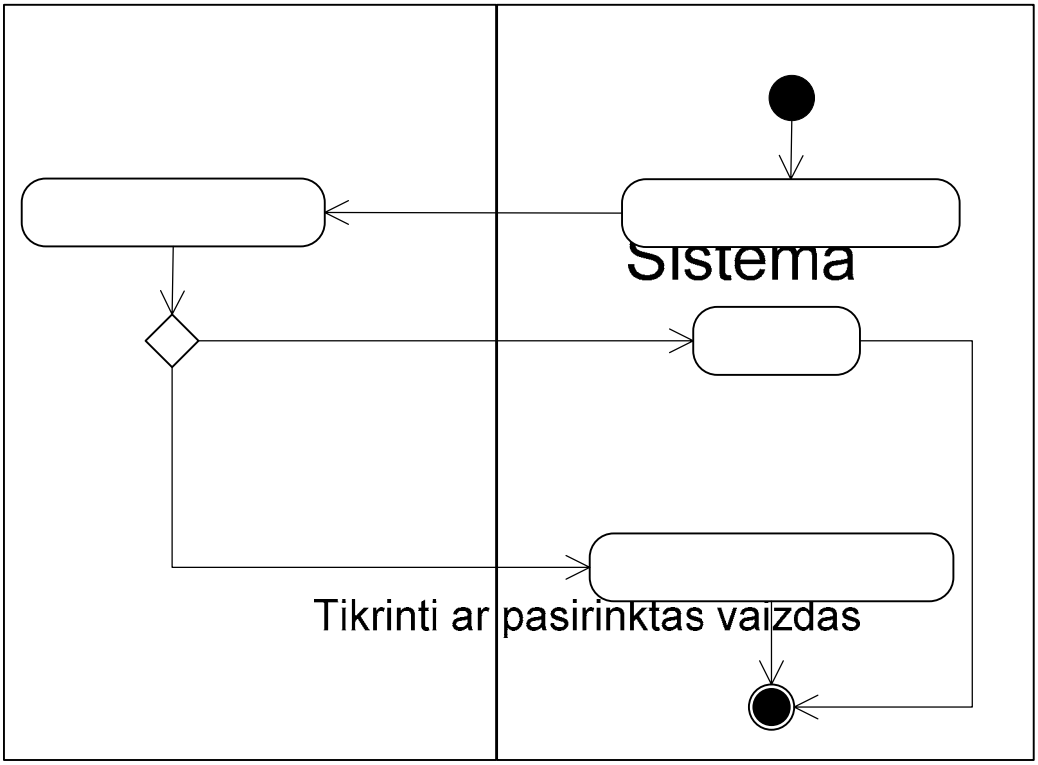
1 pav. Prisiregistravimo prie IS veiklos diagrama.

Registruodamasis prie sistemos tiek vartotojas, tiek administratorius įveda savo vartotojo vardą ir slaptažodį. Jei vartotojas yra registruotas sistemoje, tikrinamos jo teisės ir jis yra priregistruojamas prie sistemos.



2 pav. Anamnezės įkėlimo/ atnaujinimo veiklos diagrama.

Anamnezės įkėlimo/ atnaujinimo veiklos diagrama nurodo kokius veiksmus turi atlikti vartotojas norėdamas įkelti duomenis apie tyrimą į ODB. Nepasirinkus paciento ir tyrimo, negalima įkelti ar atnaujinti duomenų apie tyrimą. Jei pacientas ir tyrimas pasirinktas, tyrimui prikirti duomenys atnaujinami ODB. Nuspaudus anamnezės duomenų įkėlimo mygtuką automatiškai perkeliama į oftalmologinio tyrimo (vaizdo įkėlimo/ pasirinkimo) formą (10 pav.).

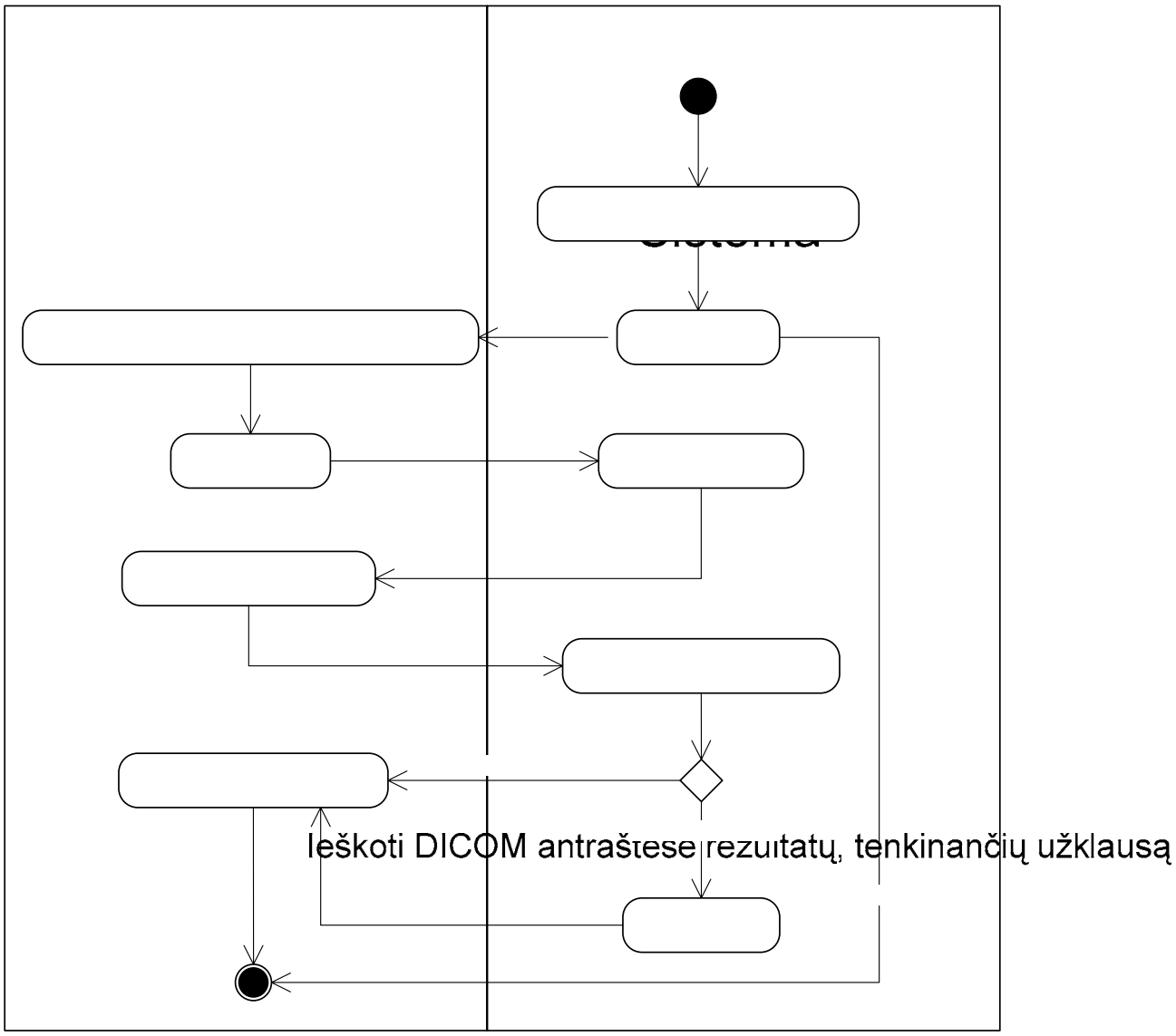


3 pav. Vaizdo peržiūros veiklos diagrama.

[Taip]

Vaizdo peržiūros diagrama nurodo kaip atvaizduojamas pasirinktas vaizdas. Jei vaizdas nepasirinktas, vartotojui apie tai pranešama. Pasirinktas vaizdas atvaizduojamas naršyklės lange įprasto (.jpg formato) paveikslo pavidalu.

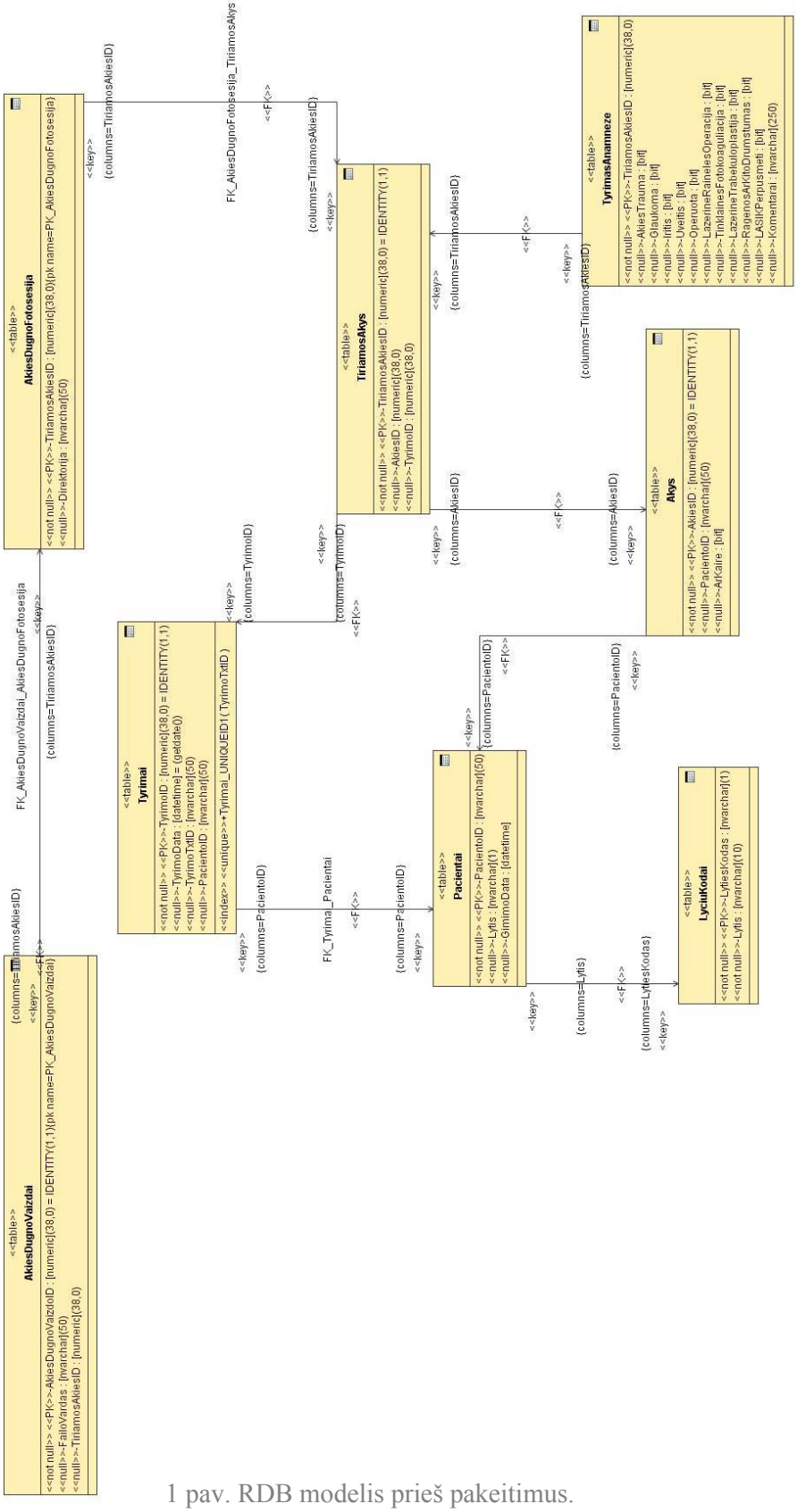
[Ne]



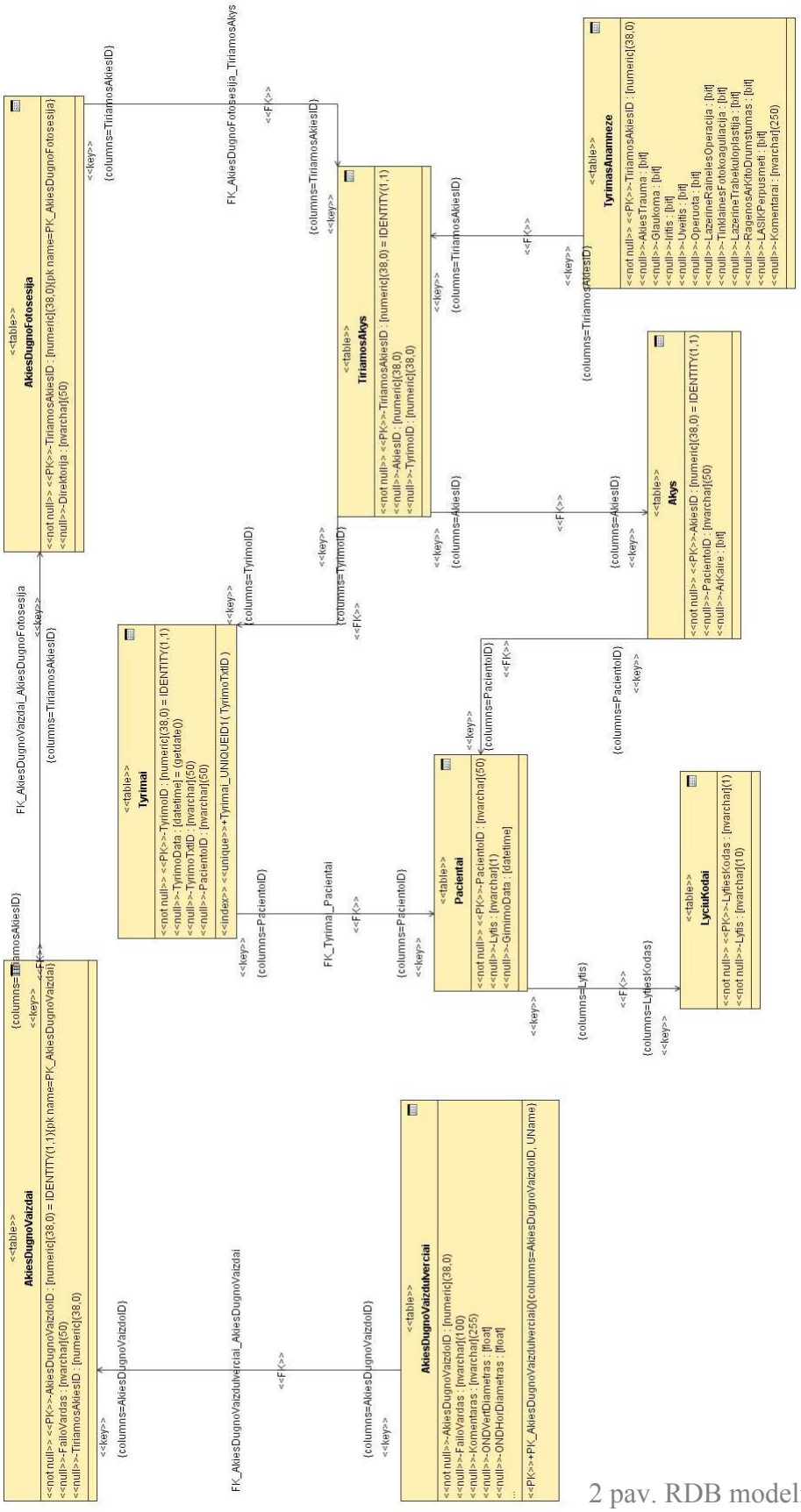
4 pav. Klinikinių sprendimų (KS) medžio formavimo veiklos diagrama.

Klinikinių sprendimų medžio formavimo diagrama nurodo kokius veiksmus atlieka vartotojas formuodamas KS medį. Gydytojas matydamas KS medį priima pirminę diagnozę. Ji užregistruojama sistemoje. Nustatytu laiku (pavyzdžiui kartą per mėnesį) gydytojui primenamos jo preliminarios diagnozės, tuo metu jis žino ar ta diagnozė pasitvirtino ar ne. Jai pasitvirtinus byla įkeliama į DB ir užregistruojama kaip tinkama naudoti kitiems KS medžių formavimams. Jei diagnozė atmetama, ją reikia pakeisti. Taip KS medžių formavimo algoritmas apsimoko ir tobulėja pateikdamas patikimesnes pirmines diagnozes.

10.4. Priedas. Eksperimento metu suformuoti RBD modeliai



1 pav. RDB modelis prieš pakeitimus.



2 pav. RDB modelis po pakeitimų.

10.5. Priedas. Vartotojo vadovas

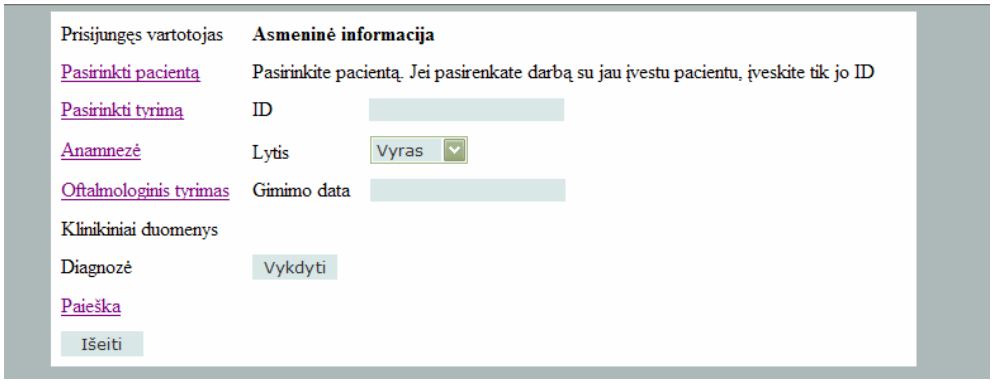
„Oftalmologinių vaidų informacijos sistema“ skirta gydytojams oftalmologams įkelti, atnaujinti ir išrinkti duomenis susijusius su pacientais, ir jiems atliktais tyrimais. Sistema saugo tyrimo metu užfiksuotus pacientų akies dugno vaizdus. Ši pagalba skirta pradedantiems dirbti su sistema gydytojams. Atsidarius pirmąjį sistemos langą reikalaujama prisiregistruoti. Šis langas pateiktas 1 paveiksle.

1 pav. Prisijungimo prie sistemos langas.

Neregistruotiems vartotojams galima registruotis, tokiu būdu prašyti priėjimo prie sistemos ir joje saugomų duomenų. Naujo vartotojo registravimo forma pateikta 2 paveiksle.

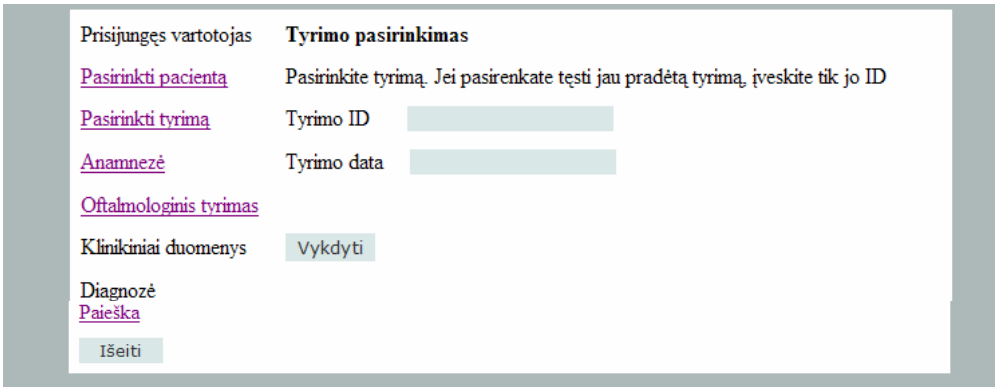
2 pav. Naujo varotojo registravimo forma.

Besiregistruojantis vartotojas privalo užpildyti visus formos laukus, kad gelėtų vėliau jungtis prie sistemos. Leidimą naudotis sistemos resursais suteikia administratorius. Prisijungęs vartotojas gali įkelti, atnaujinti duomenis esančius sistemos resursuose. Taip pat vykdyti paiešką pagal paties pasirinktu kriterijus. Prisiregistravus vartotojui atveriamas paciento įkėlimo/ pasirinkimo langas, pavaizduotas 3 paveiksle.



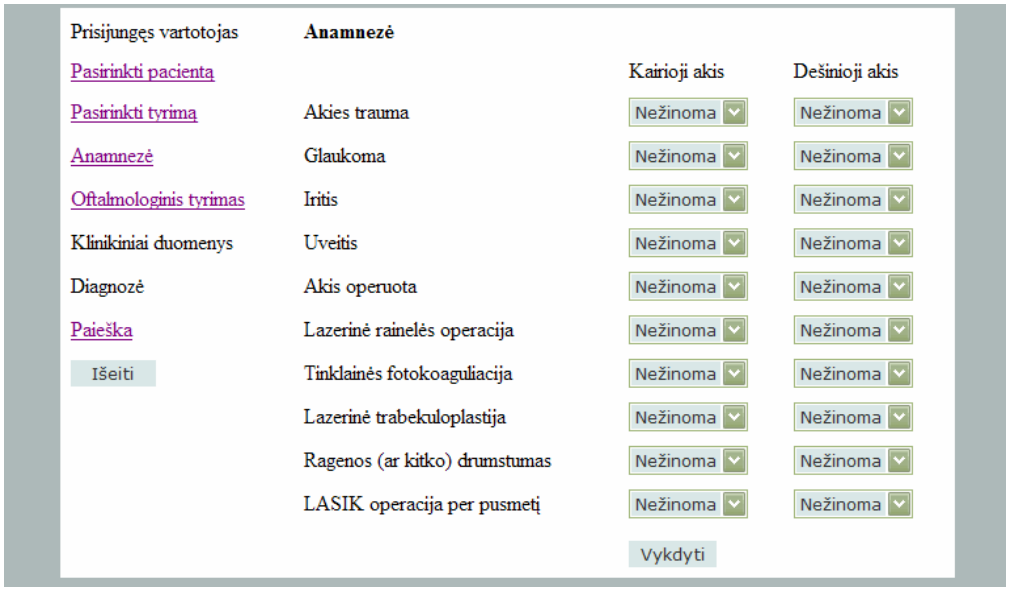
3 pav. Paciento įkėlimo / pasirinkimo forma.

Įrašius tik paciento ID ir paspaudus „Vykdyti“ vartotojas bus pervestas į tyrimo įkėlimo langą. Tačiau jeigu duomenų bazėje tokio paciento nebus, vartotojui bus liepiama užpildyti likusius formos langus. Po to bus perkeliama į 4 paveiksle pavaizduotą formą.



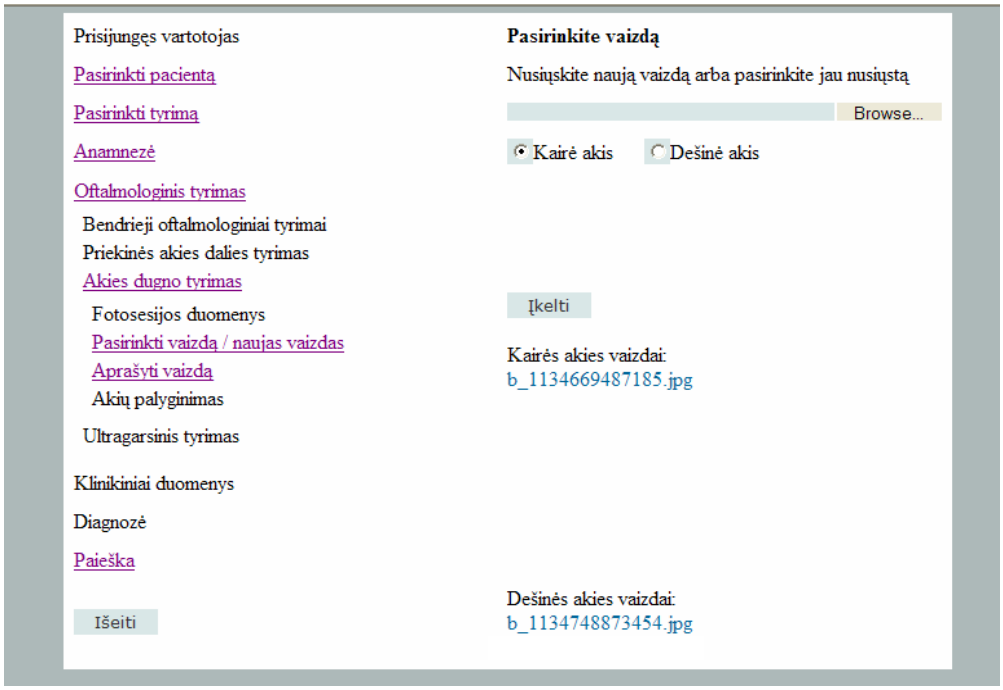
4 pav. Tyrimo įkėlimo / pasirinkimo forma.

Šioje formoje užpildžius laukus bus tikrinamas ar yra įvestas tyrimo ID. Jei tokio tyrimo nebus duomenų bazėje, sistema reikalaus įvesti tyrimo datą. Po to pervedama į anamnezės įkėlimo langą. Jei pacientas nepasirinktas, sistema reikalaus pirma jį pasirinkti, po to tik galima vykdyti tyrimo įkėlimą ar pasirinkimą. Anamnezės įkėlimo / atnaujinimo forma pavaizduota 5 paveiksle.



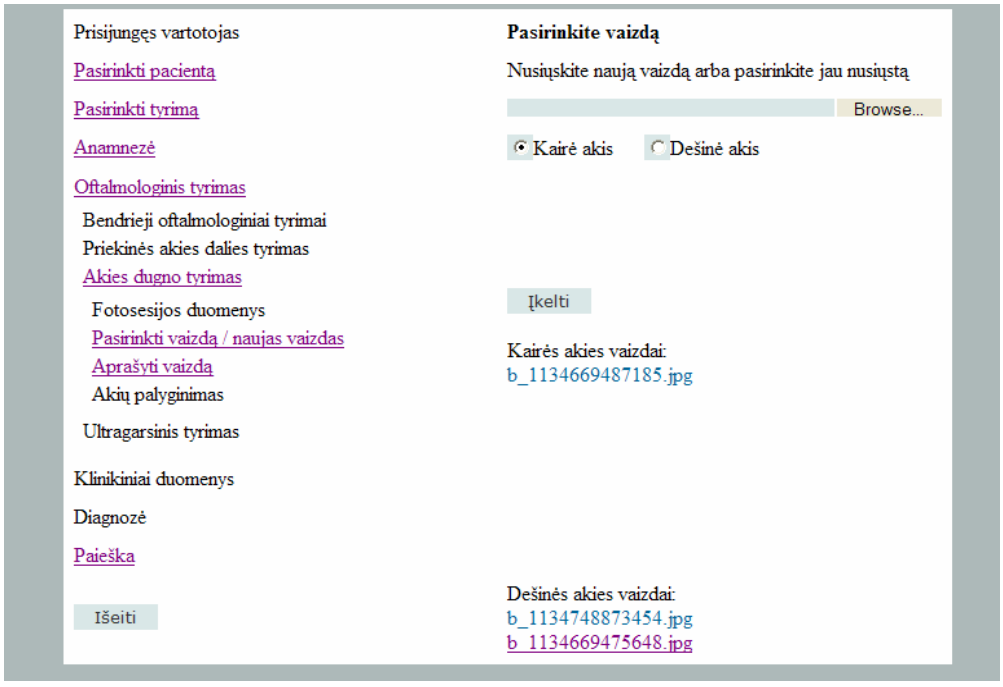
5 pav. Anamnezės įkėlimo / atnaujinimo forma.

Šioje formoje vartotojas gali užpildyti anamnezės duomenis, tačiau šis veiksmas nėra privalomas. Tačiau jo nebus galima atlikti jei pacientas ir tyrimas nebus pasirinktas. Paspaudus „Vykdyti“ duomenys įkeliami į duomenų bazę ir vartotojas perkeliamas į vaizdo pasirinkimo / įkėlimo formą. Ši forma pateikta 6 paveiksle.

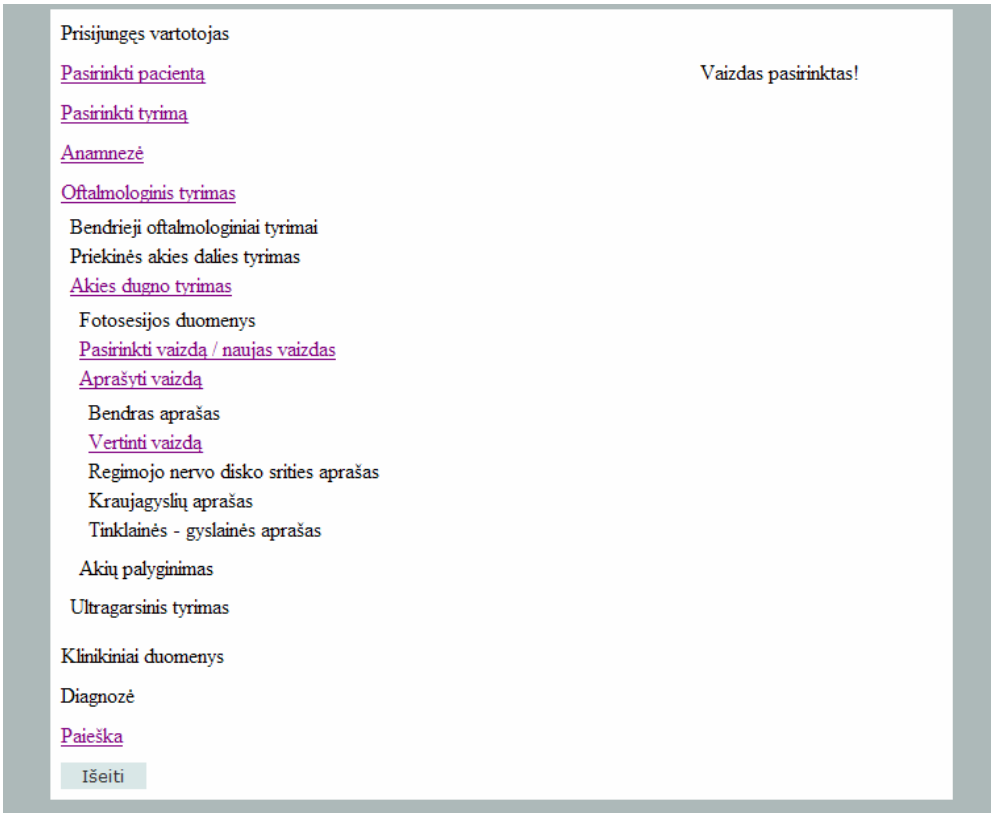


6 pav. Vaizdo įkėlimo / pasirinkimo forma.

Šioje formoje vartotojas gali priskirti paciento tyrimams vaizdus. Tačiau šių veiksmų nebus galima įvykdyti jei nebus pasirinktas pacientas ir tyrimas. Pagal pasirinktus pacientą ir tyrimą rodomi jiems priklausantys akies dugno vaizdų sąrašai. Norint peržiūrėti vaizdą reikia paspausti ant jo pavadinimo, bus perkeliama į langą pavaizduotą 8 paveiksle. Norint įkelti vaizdą, reikės paspausti mygtuką „Browse...“ (naršyti) ir pasirinkti bylą iš kompiuteryje esančių resursų. Po to pažymėti kokios akies tai vaizdas ir spausti „Įkelti“. Jei viskas sėkmingai įgyvendinta, forma atnaujinama ir vaizdą galima pasirinkti, kaip parodyta 7 paveiksle.

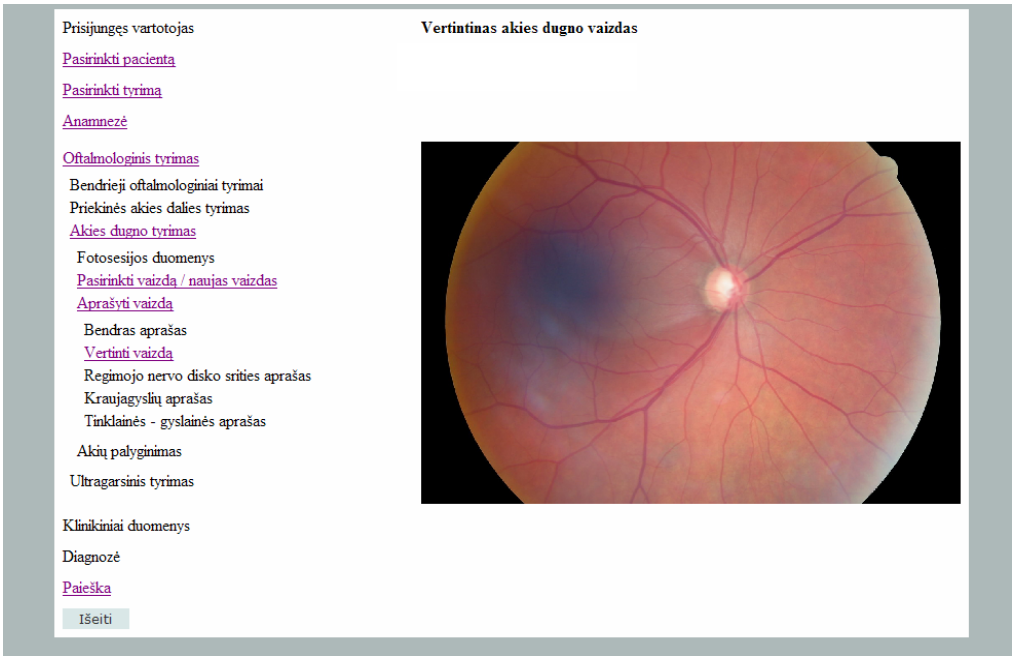


7 pav. Naujas vaizdas struktūroje.



8 pav. Langas atsiveriantis pasirinkus vaizdą.

Pasirinkus vaizdą atveriamas papildomas meniu darbui su vaizdu. Paspaudus ant nuorodos „Vertinti vaizdą“, galima peržiūrėti patį vaizdą, kaip parodyta 9 paveiksle.



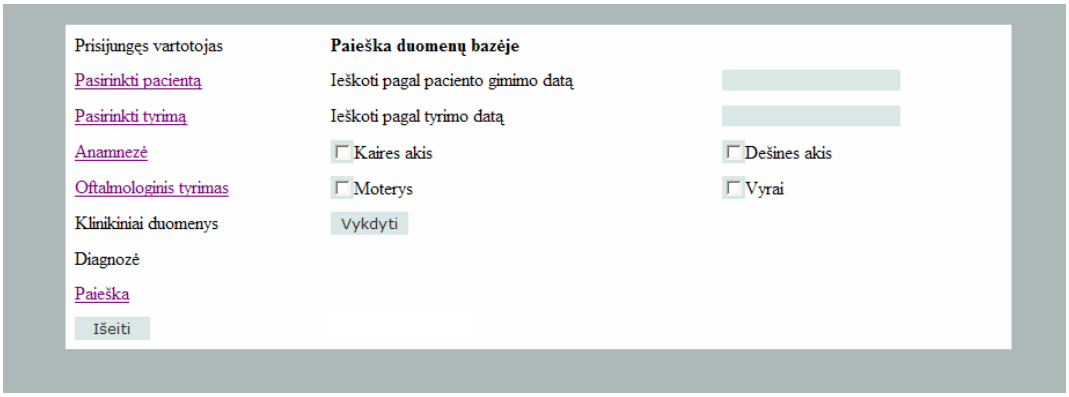
9 pav. Langas su atvaizduotu pasirinktu vaizdu.

Pasirinkus vaizdą galima įrašyti šio vaizdo fotosesijos duomenis. Tam skirta forma pavaizduota 10 paveiksle.

10 pav. Akies dugno fotosesijos duomenų įkėlimo / atnaujinimo forma.

Šią formą galima užpildyti tik tokiu atveju kai pacientas, tyrimas ir vaizdas yra pasirinkti. Užpildžius laukus ir paspaudus „Vykdyti“ duomenis įkeliami į duomenų bazę, nebūtina užpildyti visų laukų.

Šioje sistemoje vartotojas gali atlikti duomenų paiešką. Ši forma pateikta 11 paveiksle.



11 pav. Paieškos forma.

Atlikti paieška vartotojas gali bet kuriuo metu dirbdamas sistema. Tai nepriklauso nuo paciento tyrimo ir vaizdo pasirinkimo. Nurodęs vartotojui reikiamus kriterijus ir paspaudus „Vykdyti“ duomenų bazėje ieškoma duomenų atitinkančių pažymėtus

kriterijus. Jei duomenų rasta, jie atvaizduojami sąrašu, jei nėra, pranešama, kad duomenų nerasta.

Naviguoti sistemoje vartotojas gali pasirinkęs tyrimą pacientą ir kai kuriose vietose vaizdą. Tačiau vartotojas turi atidžiai stebėti ar pacientas ir tyrimas pasirinktas teisingai.

Vartotojas bet kada gali išsiregistruoti iš sistemos. Reiki paspausti mygtuką „Išeiti“.

10.6. Priedas. Straipsnis

THE USE OF OBJECT ORIENTED TECHNOLOGIES FOR MEDICAL DATA STORING AND RETRIEVING

Vita Misevičiūtė, Arūnas Lukoševičius

Kaunas University of Technology, Biomedical Engineering Institute,
Studentų st. 50 LT−51368 Kaunas, Lithuania

Abstract. This article focuses on object oriented technologies and the possibility, using these technologies, to create an information system (IS) for medical applications. We review standards that are used for management of medical data. The rapid evolution of standards, concepts and technologies has created a problem – how to develop a system that would be flexible enough to work with evolving data. After analysis of object-oriented technologies: J2EE and .NET and open source object-oriented data basis (OODB) we have proposed a system based on .NET and OODB db4o. The system would have increased possibilities of informational data mining, creating clinical decision support trees, data storing, retrieving and exchanging. This IS would help medical specialist to reach very important information from any computer in any hospital.

Keywords: object-oriented technology, object-oriented data basis, information system, data mining, patient's records, data storing standards.

Introduction

Information systems (IS) currently used in medical field are one of the most evolving components for e-health. The main features of IS in medicine are 1) storage of multimedia; 2) information connections in complicated and evolving semantic relations; 3) big amounts of information involving all live-long patients' records; 4) wide list of users, spanning between all three health care levels (they create a need of secure and convenient communication between these levels and IS); 5) compatibility of IS with the European Union (EU) [1] and international standards [2, 6]; and 6) usage of data for clinical decision support.

These factors raise specific requirements for IS and their architectures. Therefore, a search for new technologies in order to develop such systems is in demand [3, 4, 5, 7, 16, 17]. A number of standards have been created in the past few years, and they are still being developed. Unfortunately, those standards optimize and facilitate the work with medical data, but data processing and storage is still a technological problem. The technologies that are used for a long time are not flexible enough to work with such data. Medical data is stored in relational data bases (RDB), though studies [18, 19] show that object-oriented data bases (OODB) bring a lot of new advantages. Also the need of secure and flexible Internet is forcing to look for effective programming tools and technologies.

The aim of this article is to offer architecture of IS for medicine which in certain extent will satisfy the above mentioned features. To obtain this goal, object-oriented data bases, compatible programming techniques, data storing methods, and main features of IS for medicine are to be analyzed and applied.

The specifics and problems of IS for medical data storing

Information technologies (IT) offer a possibility to store and transmit all data of Electronic Health Records (EHR) including medical images. The problem is how to make the best benefit of this data for users. The data has to be stored in such a way that it would be easy to generalize and retrieve it. Therefore, standards are under development for interchanging of medical data. Each country has its own way of solving this problem, so one will stumble over a number of different standards describing various IS and data storing aspects: ASTM, ASC X12, IEEE/MEDIX, NCPDP, HL7 (Health Level 7) [6], DICOM [2] and others. The organization developing the standard usually specializes in one particular area, e.g. ASC X12 deal with outer standards for electronic data interchanging; AST E31.11 – messages for exchange of laboratory information; IEEE – P1157 – medical data interchanging standards (MEDIX); ACR/NEMA DICOM – imagery interchanging standards. Another example of a widely used standard is HL7. This standard was created in American National Standards Institute (ANSI), it is designed for data interchanging between IS for medicine and different health care institutions. This standard is

broadly used all over the world. A European standard committee CEN/TC 251 has, also, published a comprehensive system of related standards [1], e.g. CEN/TS 14796:2004 – data types; CR 12069:1995 – profiles for medical image interchange; ENV 13606 – group of EHR communication standards. The producers of medical technologies (Philips, Siemens, Sony, Acuson) collaborate with each of these companies and use their standards in their technical-software equipments.

Since every country has a problem with application of particular standards, health professionals have a very serious problem because they can't share information quickly and efficiently. This problem has especially increased after the EU opened its boarders and every person is free to travel around the continent. So there is a demand of medical data access everywhere all the time to ensure a qualitative and timely health care. This forced the committees to organize international meetings and collaborate in the field of standards. This leads to the better quality of developed products, as well as software – the systems are able to work with different data types.

All medical images today can be quickly generated into a digital format and stored in large medical Picture Archive and Communication Systems (PACS). There are a lot of PACS based systems, which meet different requirements, but they are all storing data in relational DBs which are based on a network between regional hospitals. These systems use SQL and HTTP protocols. The use of these systems seems to be feasible from the programmers' perspective; however, doctors have specific demands and they are very quickly changing. So, if something changes in data structure, half of the system might crumble, and a lot of programming changes are to be done. This problem can be easily solved by OODBs. Using such DBs is easier to implement data mining and clinical decision support algorithms.

The analysis of object-oriented data bases

OODB is a new quickly developing technology which is increasingly applied by many different companies. The benefits of OODB bring a rise in users as well as demands. These DBs have distinctive data storing methods. The data is stored in the tables, which can have tables in themselves – they form a hierarchy. The tables are filed not only with concrete data, but also with objects. This technology can be effectively implemented for storing medical data. Because of the evolution of medical requirements OODBs makes the systems more flexible and stable [18, 19]. It is also enables interaction between objects in the data base. RDBs are convenient in storing text and numerical data; but they can't work with undefined records which are formed in object-oriented programming languages. Also these data base management systems (DBMS) are not fitted for storing methods. ISs have to work not only with text, but also with spreadsheets, images, diagrams, sounds and other multimedia. Besides, all this information has to be organized as united structure, handy for users. One of the solutions to fix this problem could be a construction of an extra layer upon rational DBMS (RDBMS). It would separate objects into simple components which could be managed by RDBMS. While such a solution would solve one problem, it would create another. Data mining would be slow and complicated. The best decision is to implement object-oriented DBMS (OODBMS). We will see that this technology is able to vouch a new approach of data storing and mining.

The standards of OODB are handled by Object Database Management Group (ODMG) [9]. It refers to the following object specifications:

- Object Model – ensures the functionality of OODB;
- Object definition language (ODL) – describes OODB schemes;
- Object Query Language (OQL) – SQL type language for object retrieval;
- Language binding – the specification between programming languages and OQLs.

OODBMS stores composite objects which compound out of other objects. These objects are stored with references to lower level objects. The number of levels is unlimited. Each object can be brought into many other objects as components. This forms a net structure data model. OODB allows constructing new data types; this feature enables to store multimedia: images, audio and video files. These DBs perform data mining faster than RDB because the structure has direct references to objects.

There are two types of object-oriented DBs: passive and active. Passive DBs save object structure but don't control behavior of objects. While, active OODBs allow communication between objects within the DB. A lot of companies offer the products of OODBMS, but most of them are payable. Still there is a part of the software developers who propose free products. In this article we review six open source OODBMS; they are listed in table no. 1 [10, 11, 12, 13, 14, 15].

Table 1. Open source OODBMS

OODBMS	PE[2]	Limitation
db4o	J2EE .NET	-
SOD2	J2EE C++	Doesn't have query mechanism;
ozone	J2EE	Implemented in JAVA and works only in J2EE environment; Doesn't have query mechanism;
ObjectDB	J2EE	Implemented in JAVA and works only in J2EE environment; Free DB edition doesn't support client-server architecture (unless the DB is in the same server);
GigaBASE	J2EE C++	Object-relational DBMS;
XL2	J2EE	Doesn't have data recovery functions; Doesn't have query mechanism;

We can see that open source data bases are implemented for J2EE, and not all of them correspond to ODMG specifications as well as overall OODB flexibility.

A comparison between J2EE and .NET technologies

Web service technologies are very widely used because they provide programmers a lot of capabilities and their implementation is not difficult. The main platforms used are J2EE and .NET. The companies that offer these products are one of the biggest competitors with each other. So, the developer has a dilemma – which one to choose. Sun Microsystems offers to use Java 2 Platform, Enterprise Edition (J2EE) and Microsoft – .NET technology. Both technologies allow implementing web services. The main advantage of J2EE is the fact that it is a multiplatform – it is possible to use J2EE in different operating systems (OS). Microsoft .NET works only in Windows OS, although the company offers equivalent tool for Linux OS. Both technologies use object-oriented programming languages (.NET – Visual Basic and others, J2EE – JAVA). The advantage of .NET is it's capability to render pages in various HTML formats, which abstracts developers from version-specific HTML. Servlets can achieve the same task, though with more manual coding [7]. Another .NET advantage is Multilanguage; it works with FORTRAN, COBOL, C# and Visual Basic. The main features of both technologies are presented in table no. 2 [3, 4, 16].

Table 2. Comparison of J2EE and .NET

Feature	J2EE	.NET
Relational DB Access	Java Database Connectivity (JDBC) or L/J	Active Data Objects (ADO.NET)
Web Client	JAVA server pages (JSP) and servlets	ASP.NET
Graphical user interface (GUI)	AWT/Swing	Windows forms
Messaging	Java Messaging Service (JMS 1.0)	Microsoft Messaging Queuing (MSMQ)
Web Services Support	Java Web Services Developer Pack (JWSDP)	Built directly into .NET and Visual Studio
Technology	J2EE	.NET
Infrastructure	Enterprise Java Beans (EJB)	COM+
Naming and Directory Service	Java Naming and Directory Interface (JNDI)	Active Directory Services Interface (ADSI)
Database storage	-	SQLServer[3]

As we can see both technologies are very similar. Most important is that both of them implement web services and are able to map data on the Internet. A .NET Web service could serve as a consumer to a J2EE Web service and vice versa because of the same Web services standards used [7]. Moreover, web services can be reused no matter what programming language they were coded in. Most authors don't point out particular reasons why one should choose either technology [5, 16, 17]. The decision point can be that the first specifications of J2EE technology came three

[2] Programming environment

[3] SQLServer is the official .NET platform database technology.

years after the first implementation of the equivalent .NET platform technologies [16]. But usually the decision – which technology to choose – is left for the programmer or the company. Still we can penetrate some advantages of .NET, its development and handling capabilities.

System architecture and implementation

In Lithuania a multifunctional telemedicine system MediPas is used for retrieving, storing and exchanging (through video conferences) medical data [8]. The system stores data in remote DBs, but there is no centralized data control or informative data mining, which would allow building clinical decision support system (CDSS). MediPas stores images (of various file formats) in RDB (MS Access) or in the server (MS SQL server). The images are only stored, not referring the topical information. This program reads and writes DICOM standard images; this standard is widely used for storing imagery data. The system is developed for the main purpose – organizing real-time video conferences and acquisition of data. In table No. 3 there is a summary of the problems relevant to users, and possible solutions proposed.
IS for medicine data mining could be implemented using the above mentioned technologies. No matter which object-oriented technology is chosen, the system architecture is needed. The architecture is presented in figure No. 1.
In table No. 4 there are displayed features of proposed IS, pointing the importance of each feature, described why it is important and declared possible alternatives.

Table 3. Users' needs and problems

Problem	Present system	Solutions
Users can read and write DICOM standard images, but they can't execute informative data mining.	MediPas can only read and write DICOM standard images.	OODB directly communicates with DICOM standard images. It is possible to read information saved in DICOM header, as well as perform data mining.
Users can look for data only in text fields.	RDB doesn't look for data in *binary* files.	It is possible to implement data mining in various data files. It can be effectively used for dealing with complicated diagnostic cases.
In big RDBs it is harder to apply data mining algorithms, because of its grand structure.		Data mining is easier implemented in OODBs, because it stores data using references between objects.
Exchanging with data.	There are no RDB designed for storing medical data, or united DB structure standard. So the data exchange between such DBs is inconvenient and needs a converter.	In OODB medical data is stored in a standard way (e.g. DICOM files) they can be directly exchanged between OODBs and other ISs.

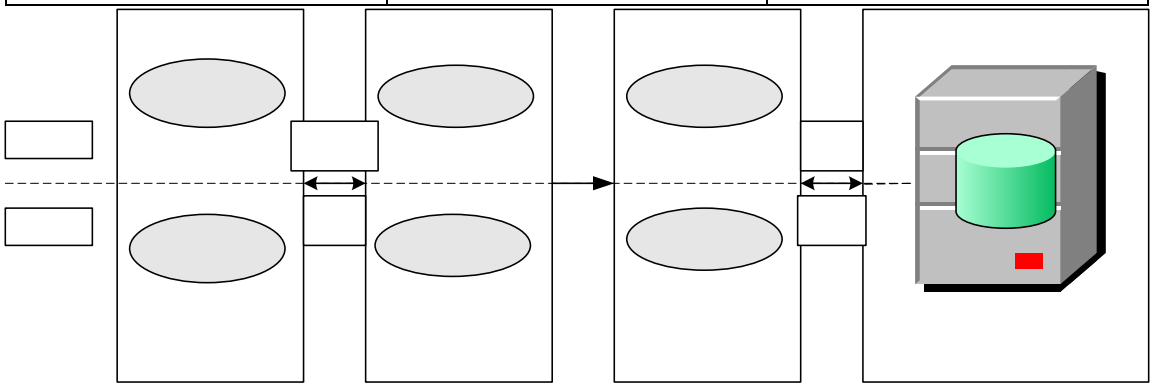


Figure 1. Proposed architecture for IS

Table 4. Characteristics of proposed IS

Feature	Importance	Description
Simple GUI	High	GUI must be simple, because the users don‘t have sufficient computer knowledge.
Connection to the DB through internet	High	It is one of the most important system advantages.
Clinical decision support tree implementation	High	Data mining must be efficient.
System administration	High	System must be administrated and supervised.
Automatic user registration	Low	Users can be registered by the administrator.
Estimation of user rights	Low	This feature could be implemented in the future; users could be divided into different groups.
Safe data transfer through internet	High	Patient’s personal information is stored in DICOM file header, so it must be protected.

As we can see from figure No.1, the architecture of .NET and J2EE technologies are very similar. The advantage of it is that both systems can communicate with the same DB, if there would be a necessity to communicate. The OODB is easier to integrate, so we suggest using OODBs. And we propose db4o OODB that can be implemented in both .NET and J2EE.
Only the basic and most general features are described in table No.4. It should be expanded regarding to the specifics of IS.

Discussion

Extensive international and collaborative work with standards and concepts is gradually leading to the better quality of developed products, as well as software – the systems are able to work with different data types, to ensure compatibility and flexibility.
The analysis of OODBs shows that they are very convenient for storing of semantically and quantitatively evolving medical data. It is possible to implement remote DB access using object-oriented technologies easily.
The proposed IS architecture and analysis of implementation possibilities had shown that such system would be flexible and evolutionistic because of the usage of appropriate technologies which are at the same time compatible with conceptual medical informatics standards. Although a proposed architecture could be implemented by the use of different technologies, we prefer using .NET technology because of its development and handling capabilities, also the Multilanguage ability and the user-friendly environment of Visual Studio .NET software.

References:

1. CEN/TC 251 site [Last viewed on 2005-02-27]. It can be accessed: http://www.centc251.org/
2. DICOM site [Last viewed on 2005-02-27]. It can be accessed: http://medical.nema.org/
3. B. T. Estes, O. Maxime. *J2EE vs .Net: The choice depends on your needs*. 2003. [Last viewed on 2005-02-27]. It can be accessed: http://www.computerworld.com/printthis/2003/0,4814,84155,00.html
4. J. Farley. *Microsoft .NET vs. J2EE: How Do They Stack Up?*. 2000. [Last viewed on 2005-02-27]. It can be accessed: http://java.oreilly.com/news/farley_0800.html
5. J. J. Hanson. *.NET versus J2EE Web Services. A Comparison of Approaches*. [Last viewed on 2005-02-27]. It can be accessed: http://www.webservicesarchitect.com/content/articles/hanson01.asp
6. HL7 site [Last viewed on 2005-02-27]. It can be accessed: http://www.hl7.org/
7. J. Lurie, R. J. Belanger. *Does one Web services platform dominate the other?*. 2002. [Last viewed on 2005-02-27]. It can be accessed: http://www.javaworld.com/javaworld/jw-03-2002/jw-0308-j2eenet.html

8. MediPas [Last viewed on 2005-02-27]. It can be accessed: http://tmc.kmu.lt/medipas.htm
9. Object Database Management Group [Last viewed on 2005-02-27]. It can be accessed: http://www.odmg.org
10. OODB db4o [Last viewed on 2005-02-27]. It can be accessed: http://www.db4o.com/
11. OODB ObjectDB [Last viewed on 2005-02-27]. It can be accessed: http://www.objectdb.com/
12. OODB ozone [Last viewed on 2005-02-27]. It can be accessed: http://ozone-db.org/frames/home/what.html
13. OODB SOD2 [Last viewed on 2005-02-27]. It can be accessed: http://www.tneumann.de/sod/index.html
14. OODB XL2 [Last viewed on 2005-02-27]. It can be accessed: http://www.xl2.net/
15. O-RDB GigaBASE [Last viewed on 2005-02-27]. It can be accessed: http://www.garret.ru/~knizhnik/gigabase.html
16. R. Sessions. *Java 2 Enterprise Edition (J2EE) versus The .NET Platform Two Visions for eBusiness*. 2001. [Last viewed on 2005-02-27]. It can be accessed: http://www.objectwatch.com/FinalJ2EEandDotNet.doc
17. H. Sheil, M. Monteiro. *Rumble in the jungle: J2EE versus .Net. How do J2EE and Microsoft's .Net compare in enterprise environments?*. 2002. [Last viewed on 2005-02-27]. It can be accessed: http://www.javaworld.com/javaworld/jw-06-2002/jw-0628-j2eevsnet_p.html
18. M. Sorli, J. Romo, R. Stach, B. Bredehorst. *REMOTE: GRD1-2000-25433*. 2001, p. 40-42. [Last viewed on 2005-07-27]. It can be accessed: www.remote-project.org/downloads/REMOTE.D1.1-1.1--2001-07-09.pdf
19. S. Zamir. *Handbook of object technology*. CRC Press, 1999 – 35ch.