KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Ramūnas Liuokaitis

AKMENĖS GIMNAZIJOS INTERAKTYVI INFORMACINĖ SISTEMA

Magistro darbas

Darbo vadovas
doc. V. Šakys

KAUNAS, 2005

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

TVIRTINU
Katedros vedėjas
doc. R. Butleris
2005-05-...

AKMENĖS GIMNAZIJOS INTERAKTYVI INFORMACINĖ SISTEMA

Informatikos mokslo magistro baigiamasis darbas

Kalbos konsultantė
Lietuvių kalbos katedros lektorė
dr. J. Mikelionienė
2005 05 ...

Vadovas
doc. V. Šakys
2005 05 ...

Recenzentas
doc. dr. A. Lenkevičius
2005 05 ...

Atliko
IFM– 9/3gr. stud.
R. Liuokaitis
2005-05-...

KAUNAS, 2005

Turinys:

Įvadas

Šiuolaikinių technologijų progresas įpareigoja ieškoti efektyviausių sprendimų optimizuojant švietimo organizacijų veiklos procesą. Automatizuota švietimo organizacijos duomenų kontrolė gerina darbo kokybę, tiesiog gyvybiškai svarbus tinkamas resursų panaudojimas valdant mokinių, mokytojų, mokyklos duomenis. Tačiau daugeliu atvejų duomenys valdomi rankiniu metodu ir rezultatas atitinka iš anksto nusibrėžtus reikalavimus. Pakeitimams ir analizei sugaištama daug brangaus laiko.

Šis darbas skirtas elektrininio dienyno valdymo programinės įrangos tobulinimui, naudojant naujausias technologijas, tokias kaip Žiniatinklio paslauga, realizuota kaip taikomųjų programų serveris. Žiniatinklio paslaugos architektūros pagrindas yra į paslaugą orientuota architektūra. Jo paslaugoje naudojami XML, SOAP, UDDI and WSDL. Žiniatinklio paslaugos pateikia abstraktų modelį, aprašantį produkto pateikimo ir paieškos būdą žiniatinklio vartotojui. Trumpas sistemos sukūrimo laikas, sumažinta savikaina ir investicijos pakartotiniam panaudojamumui. Taip pat atveriamos naujos perspektyvos elektroninio dienyno visos sukūrimo programinės įrangos integracijai.

Tokio elektroninio dienyno sukūrimas yra gana sudėtingas darbas, nes reikalauja, kad visi mokyklos duomenys būtų lengvai ir saugiai tvarkomi. Šiuo metu jau egzistuoja keletas eksperimentinių elektroninių dienynų, tačiau jie yra kuriami pagal konkrečių mokyklų pageidavimus, atsižvelgiant tik į išskirtines mokyklos ypatybes bei reikalavimus. Toks dienynas reikalingas šiuo metu todėl, kad besivystant technologijai, tobulėja ir pati visuomenė. Tėvai pageidauja, kad jie galėtų patys kiekvieną dieną stebėti vaiko lankomumą, pažangumą ir aktyvumą mokyklos veikloje. Taip gali būti skatinamas tėvų bei vaikų tarpusavio supratimas ir geresnis bendradarbiavimas.

Šiuo metu, atsižvelgiant į ankstesnių programų trūkumus bei konkrečius mokyklos reikalavimus, yra kuriami elektroniniai dienynai, kurie, be abejo, nuolat tobulėja. Tačiau universalios programos visoms mokykloms sukurti neįmanoma, nes visos mokyklos turi savo išskirtinius bruožus.

Šio darbo tikslas – sukurti mokyklos informacinę sistemą – elektroninį dienyną, kuris kauptų, redaguotų ir analizuotų įvairius duomenis, naudojamus mokykloje mokymosi metu, pateiktų įvairias elektronines ataskaitas, ieškotų klaidų, perspėtų apie tai MIS (mokyklos informacinės sistemos) vartotoją.

Norėdamas įgyvendinti šį tikslą, sau kėliau uždavinius:

1) atlikti teorinę žiniatinklio paslaugų ir duomenų bazių valdymo sistemų analizę;
2) atlikti jau sukurtų analogiškų elektroninių dienynų analizę;

3) apklausti būsimus vartotojus;
4) nustatyti sistemos reikalavimus;
5) sukurti informacinės sistemos prototipą;
6) sukurti lengvai valdomą informacinę sistemą;
7) įdiegti informacinę sistemą Akmenės gimnazijoje.

1. Analitinė dalis

1.1. Tyrimo sritis

Kiekvienas IT specialistas yra girdėjęs apie žiniatinklio paslaugas (WS). Žiniatinklio paslauga yra apibūdinama kaip naujas evoliucinis etapas vystant programinę įrangą, susijusią su pasauliniu tinklalapiu (WWW). Ši nauja koncepcija išvystė ir paplito po pasaulį neįtikėtinai sparčiu tempu. Žiniatinklio paslaugos naudojimas neišsprendžia visų programinės įrangos kūrimo problemų, tačiau pagreitina jos kūrimo ir vystymo procesą. Per pastaruosius keletą metų pasaulinis žiniatinklis pasidarė pagrindinis faktorius paskirstytų skaičiavimų tinkle. Pagrindinės priežastys, kodėl jis prigijo ir tapo taip plačiai naudojamas, yra jo paprastumas ir platus panaudojamumas (visuotinai priimtas kaip standartas). Visus šiuos teigiamus veiksmus perėmė ir žiniatinklio paslauga. Žiniatinklio paslauga turi paprastą architektūrą ir naudoja labai plačiai paplitusius protokolus HTTP ar XML. WWW yra platforma, suteikianti galimybę visuomenės individų tarpusavio sąveikai. Ši savybė žiniatinklio paslaugoje yra transformuojama į taikomųjų programų tarpusavio sąveiką. Kai yra naudojama WS, tuomet išsprendžiamos tinklo heterogeniškumo ir decentralizacijos problemos. Žvelgiant atgal, žiniatinklio paslauga buvo pakankamai neefektyvi, kad galėtų dominti sistemos kūrėją. Tačiau besivystant tinklo pralaidumui ir duomenų saugojimui, tinklo platformoms bei skaičiavimo įrenginiams poreikis realizuoti sistemos funkcionalumą kaip WS tapo vis didesnis. WS vystymosi raidą galima palyginti su tinklo panaudojimu komerciniais tikslais evoliucija. Anot Billo Gateso – „naujas skaičiavimo modelis yra naujų pramonės poreikių rezultatas“. Šis modelis turėtų pateikti standartinį taikomųjų programų kūrimo būdą bei galimybę keistis informacija tinkle. Panaudojant WS orientuotą skaičiavimo modelį duomenų bazės valdymo programinės įrangos gamintojai galėtų sumažinti produktų savikainas ir atitinkamai pakeisti investicijų planus. Taip pat atsirastų galimybė skirtingų gamintojų modulių integracijai.

IBM aprašo WS kaip „savyje turinčią, pati save aprašančią, suskaldytą moduliais taikomąją programą, kuri galėtų būti paviešinta, surandama ir paleidžiama vykdyti per žiniatinklį. Kai WS konfigūracija yra nustatyta, tada ji paviešinama ir po to gali būti naudojama kitų taikomųjų programų.

1.2. Pagrindinės priežastys, kodėl žiniatinklio paslauga yra tokia svarbi yra:

- ✓ jos gali būti paleidžiamos vykdyti skirtingų tipų skaičiavimo įrenginiuose pradedant nuo stalinio kompiuterio baigiant didžiuoju kompiuteriu;
- ✓ nėra skirtumo, ar WS yra naudojama įmonės ribose ar išoriniame kontekste;
- ✓ jos leidžia glaudesnį verslo objektų bendradarbiavimą ir efektyvesnį resursų panaudojimą.

Žiniatinklio paslaugos (WS) yra naujai besivystanti technologija, suteikianti daugybę privalumų programinės įrangos kūrimo procese. Prie šios technologijos sukūrimo ir vystymo prisidėjo ir tebedirba daug žymių kompanijų. Vienos didžiausių – IBM ir Microsoft. Nepaisant to, kad šios dvi kompanijos tarpusavyje konkuruoja, jų tarpusavio bendras darbas padėjo WS pagrindus:

- ✓ UDDI
- ✓ SOAP
- ✓ WSDL

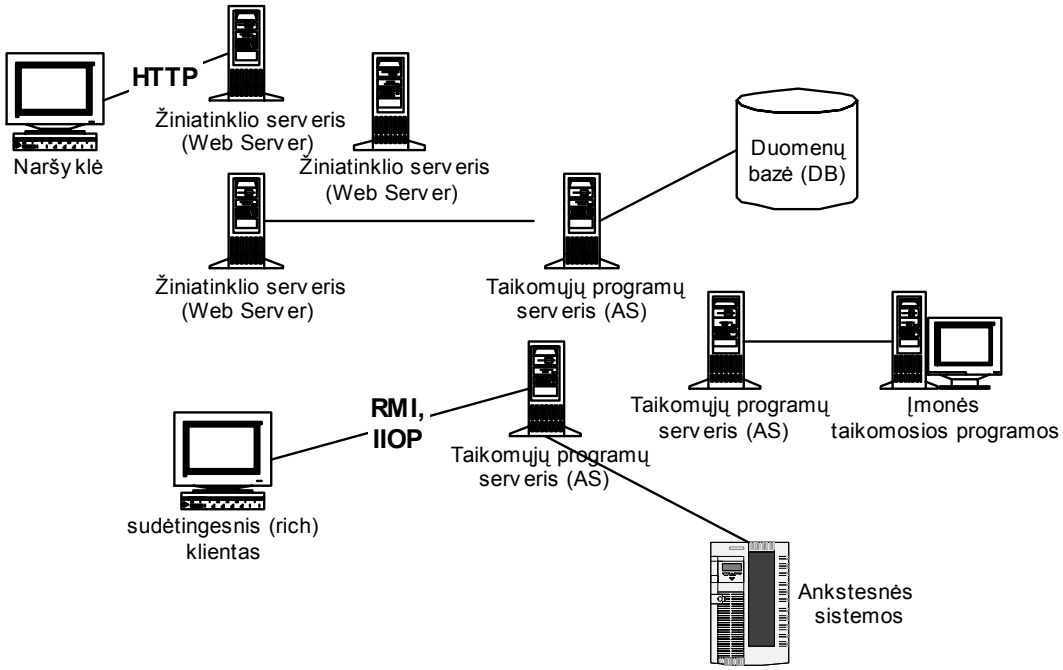
Šie standartai naudojami daugelio programinės įrangos kūrimo kompanijų (įskaitant tokius taikomųjų programų serverio gamintojus, kaip Sun Microsystems, IBM, Hewlett Packard ir BEA Systems). Didėja įmonių portalų kaip strateginių verslo objektų apimtis, ir dėl šios priežasties vykdomieji procesai tampa vis sudėtingesni ir painesni. WS technologija palengvina informacijos paiešką ir teisingą apdorojimą. Be abejo, reikalingi taikomųjų programų serverių gamintojai šiems portalams sukurti ir palaikyti.

1.3. Taikomųjų programų serveris

Terminas „Taikomųjų programų serveris“ (AS) turi daug reikšmių (pav. 1). Kiekviena programinės įrangos įmonė, kuri išleidžia AS, tuo pačiu apibrėžia ir jo prasmę. Taikomųjų programų serveris susideda iš integruotos žiniatinklio kūrimo platformos. Galima teigti, kad kiekvieno AS tikslas yra valdyti įmonės resursus – žiniatinklio serverius ir duomenų bazes. AS suteikia savo klientams galimybę susijungti su duomenų integracijos pakopoje esančiais duomenimis („backend source“). Taip pat realizuoja taikomosios programos logiką ir, apdorojus užklausą, rezultatą grąžina klientui. AS yra naudojamas siekiant užtikrinti kritinių taikomųjų programų darbo nuoseklumą ir pastovumą, taip pat ir sistemos nepriklausomybę nuo proporcingai didėjančio vartotojų skaičiaus. Naudojant AS, taikomosioms programoms:

- ✓ įvykdomas saugumo reikalavimas;
- ✓ sumažinama jų savikaina;
- ✓ pagerinamas pakartotinio panaudojamumo faktorius;
- ✓ sutrumpėja programavimo laikas.

Mažos apimties programoms pakanka žiniatinklio serverio, tačiau apimčiai augant AS poreikis auga.

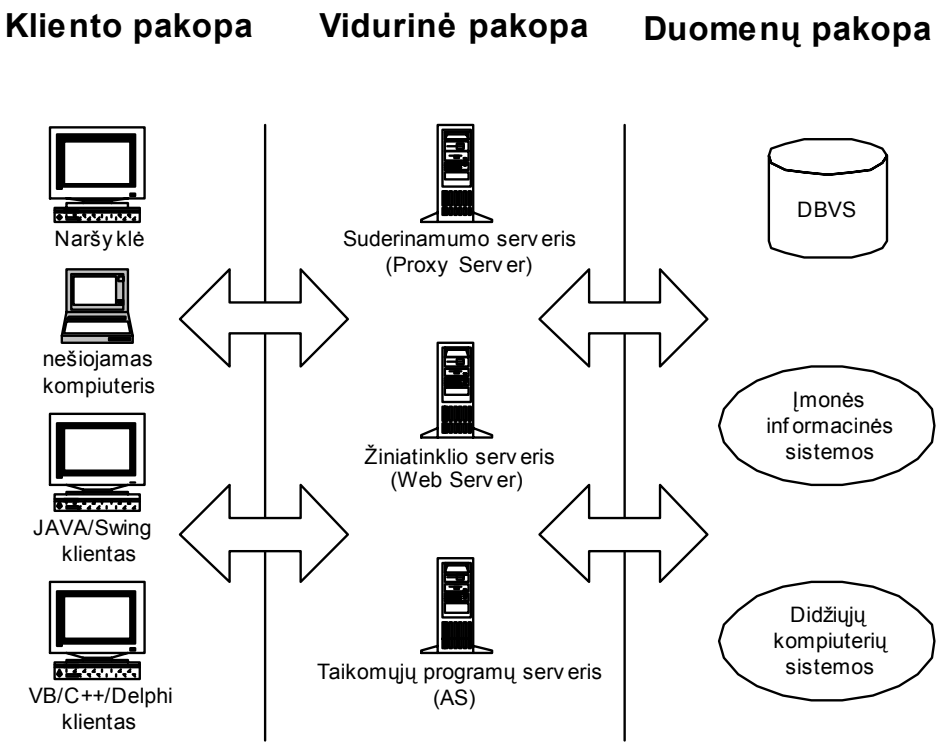


1 pav. Taikomųjų programų serverio panaudojimas

Taikomųjų programų serverio architektūra yra suskirstyta į tris pakopas (3T architektūra):

- ✓ vartotojo (kliento) pakopa;
- ✓ tarpinė pakopa;
- ✓ duomenų pakopa.

Pagrindinė AS užduotis yra taikomosios programos verslo logikos realizacija ir pateikimas vartotojui. Tai pateikiama per sujungimą tarp naršyklės ir taikomosios programos verslo logikos duomenų. Įėjimo duomenų srautas per naršyklę yra nukreipiamas į žiniatinklio serverį. Žiniatinklio serveris persiunčia užklausą taikomųjų programų serveriui, kur jame ir įvykdoma verslo logika susieta su taikomąja programa. Kitaip tariant, taikomųjų programų serveris yra tarpinė grandis tarp vartotojo ir įmonės duomenų. Ši trijų pakopų architektūra fiziškai atskiria vartotoją (klientą), kuris išsiunčia užklausą, taikomąja programą, kuri realizuoja verslo logiką ir duomenis, kurie yra naudojami taikomosios programos. 3T architektūra išsivystė iš centralizuotos, kurioje vartotojas (klientas), verslo logika ir duomenys buvo saugojami kartu. Taikomųjų programų serveriai suteikia galimybę greitai ir lengvai sukurti bei nustatyti vartotojui taikomąsias programas, kartu ir padidinti sistemos vartotojų skaičių be papildomo sistemos perprogramavimo, nes sistema yra suskaldyta atskiromis pakopomis.



2 pav. Daugelio pakopų architektūra

1.4. Trys daugiapakopės architektūros pakopos

3T skaičiavimas atskiria atvaizdavimo klientui logiką nuo verslo logikos. Šis atskyrimas reiškia, kad verslo pakopoje esantis kodas yra nepriklausomas nuo jo atvaizdavimo ir panaudojamumo. Verslo logikos vidurinio lygmens pakopa nereikalauja informacijos apie tai, kaip vartotojas vaizduoja duomenis. 3T architektūra yra lengvai pernešama, palaiko skirtingų tipų platformas ir leidžia valdyti vartotojo užklausas skirtingiems serveriams. Sistemos saugumas yra lengviau realizuojamas, kada verslo logika yra atskiriama nuo vartotojo. Tuo pačiu ir sumažėja sistemos savikaina. Tačiau realizuojant šį lygmenį tampa yra sudėtingiau realizuoti transakcijų apdorojimą ir priėjimą prie duomenų pakopos. Taikomųjų programų serverio technologija bando šį dalyką išspręsti.

- ✓ **Vartotojo pakopa**

Kliento pakopa apima visas vartotojo vykdomas taikomąsias programas ir modulius. Tokios programos, pavyzdžiui, yra žiniatinklio naršyklės. Šios taikomosios programos yra nepriklausomos nuo programavimo kalbos ir darbo platformos. Prie vartotojo pakopos galima priskirti nešiojamus, belaidžius, rankinius ir kitus skaičiavimo įrenginius su atitinkama konfigūracija. Visi šie klientai yra tarpusavyje nevienalyčiai. Tačiau darbui su sistema yra reikalinga vidurinioji pakopa, kuri vienodai sąveikauja su visais klientais.

- ✓ **Vidurinioji (tarpinė) pakopa**

Šiai pakopai priklauso taikomųjų programų serveris ir visi kiti serveriai, kurie yra adresuojami tiesiogiai iš vartotojo pakopos. Pavyzdžiui, žiniatinklio serveriai, „proxy“ serveriai[1]. Šioje pakopoje saugoma taikomųjų programų logika ir realizuojamas ryšys tarp duomenų ir taikomosios programos logikos. Taip ši pakopa apibrėžia, kaip vartotojo pakopa sąveikauja su duomenų pakopa.

- ✓ **Duomenų pakopa**

Duomenų pakopai priklauso įmonės turimi resursai, tokie kaip DBVS, ankstesnės sistemos, didieji kompiuteriai, taikomosios programos moduliai ir įmonės duomenys. Šie resursai dažniausiai yra sukurti naudojant ankstesnes technologijas, tokias kaip didžiojo kompiuterio DB.

1.5. Taikomųjų programų loginės pakopos

- ✓ **Atvaizdavimo logikos pakopa.**

Ši pakopa susideda iš dviejų pagrindinių funkcinių vienetų: atvaizdavimo logikos ir atvaizdavimo išdėstymo. Atvaizdavimo išdėstymas parodo taikomosios programos vartotojo sąsają. Daugelio atveju vartotojo sistemoje nustatoma žiniatinklio naršyklė. Taip elgiamasi dėl to, kad yra pakankamai sunku nustatyti kliento modulius (programas) į kiekvieną vartotojo kompiuterį. Pagrindinis tikslas yra atskirti atvaizdavimo logiką nuo atvaizdavimo išdėstymo. Taip pat galima naudoti skirtingus išdėstymus skirtingiems įrenginiams, tuo tarpu, kai išdėstymo logika lieka ta pati. Žiniatinklio naršyklės kliento programinė įranga yra kaip elektroninės prekybos taikomųjų programų standartas. Žiniatinkliui orientuotų taikomųjų programų vartotojo sąsaja susideda iš HTML dokumentų, JSP. Žiniatinklio naršyklė pateikia lokaliai (vartotojo kaip kliento pusėje) žiniatinklio puslapį. Žiniatinklio naršyklės panaudojimo vietoj kitų – sudėtingesnių kliento programų privalumas yra tas, kad nereikia realizuoti vartotojo sąsajos pateikimo.

- ✓ **Verslo logikos pakopa.**

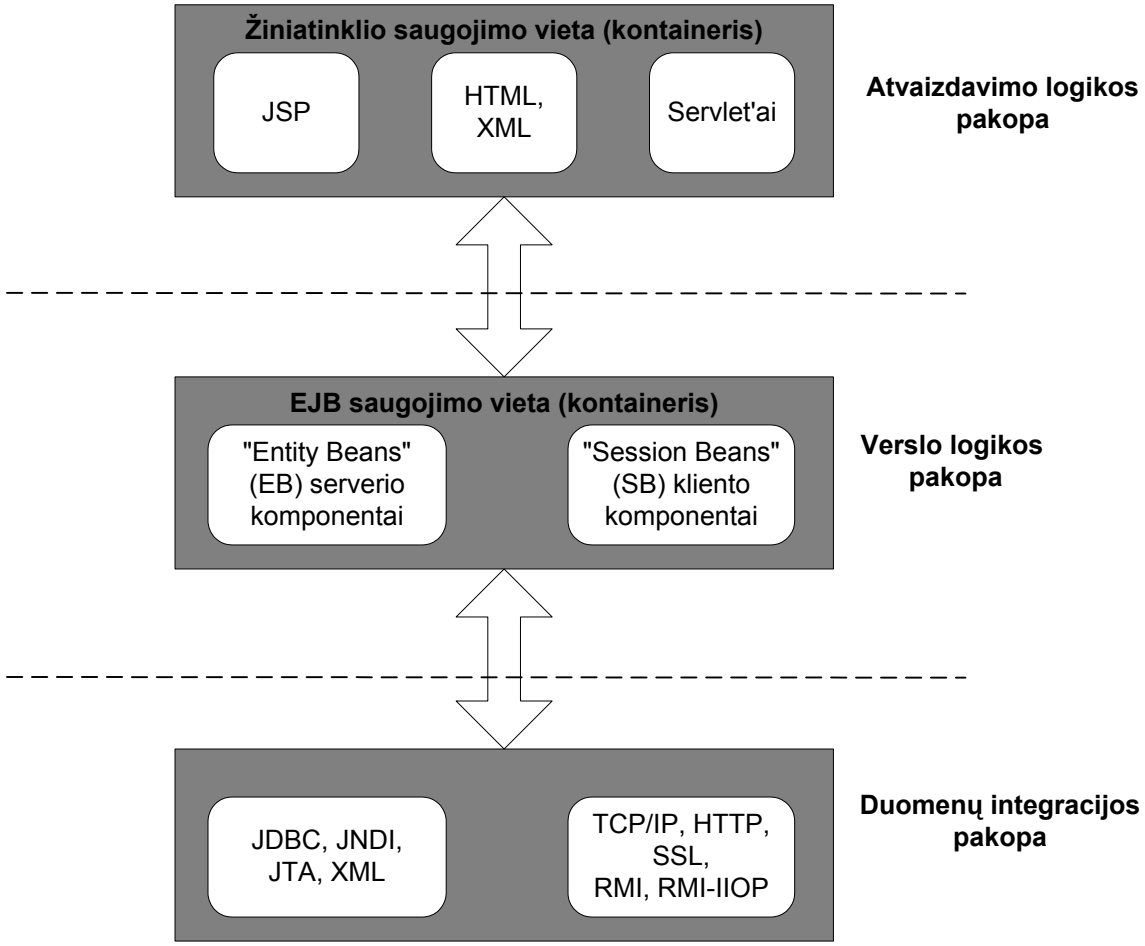
Verslo logikos pakopa aprašoma kaip taisyklių, pagal kurias taikomoji programa realizuoja verslo užklausą iš kliento pusės, visuma. Pagrindinis tikslas yra atskirti verslo logiką nuo vartotojo sąsajos, nes taip yra lengviau programą palaikyti ir keisti pačią verslo logiką. Verslo logikos pakopoje atliekami visi reikalingi skaičiavimai ir priėjimas prie duomenų, jų validavimas ir valdymo sekos kontrolė. Verslo pakopa apibrėžia, kaip elgsis puslapis ir valdo sistemos saugumą –

[1] „Proxy“ serveris – tai serveris, kuris atlieka tarpininko f-jas. Kitaip tariant, užklausos keliauja per “proxy” serverį, kuris užklausas deleguoja pagrindiniam serveriui. Taip pat tai serveris, kuris atlieka lokaliam tinkle apsaugos nuo nesankcionuoto priėjimo per internetą funkcijas.

kiekvieno vartotojo teisės atlikti norimą užduotį yra apribojamos. Pavyzdys – EJB komponentai J2EE architektūroje.

- ✓ **Duomenų integracijos pakopa.**

Duomenų integracijos pakopa susideda iš pačių duomenų ir informacijos apie jų konfigūraciją. Dažniausiai tai būna gerų vykdymo charakteristikų ir patikima sąryšinė DB, suteikianti taikomajai programai universalumą ir darbo efektyvumą. Ši pakopa suteikia galimybę jungtis su skirtingų tipų duomenų šaltiniais ir yra atsakinga už duomenų valdymą. Šios pakopos užduotys vykdomos tada, kai verslo logikos pakopoje vykdomai užklausai reikia nuskaityti arba įrašyti duomenis. Taip pat per duomenų integracijos pakopą yra pasiekiamos ankstesnės sistemos arba kiti duomenų šaltiniai. J2EE modelis realizuoja šią pakopą per JDBC API. Duomenų integracijos pakopoje neturėtų būti saugoma verslo logika, pvz., duomenų validavimas. Kartais tai sunku padaryti, nes net tokie smulkūs apribojimai kaip „not null“[2] gali būti laikomi verslo pakopos dalimi.



3 pav. Taikomųjų programų serverio pakopos

[2] „not null“ – nėra tuščias (DB terminologijoje). Gali būti laukas, visas įrašas.

Realybėje šios architektūros egzemplioriai yra daug sudėtingesni. Klientai dažniausiai yra realizuojami panaudojant „permatomo“ kliento technologiją siekiant sumažinti programinės įrangos administravimo išlaidas. Daugelyje atvejų klientas yra žiniatinklio naršyklė. Ši tendencija sumažina išsibarstymą, nes nebelieka būtinybės sukurti naujo sujungimo tarp kliento ir serverio. Pagrindinė priežastis, kodėl „permatomo“ kliento modelis yra toks svarbus didelėms kompanijoms yra lankstumo poreikis. Keičiantis verslo poreikiams yra labai svarbu greita reakcija į pasikeitimus. Siekiant greitai įgyvendinti pasikeitimus verslo ir taikomosios programos taisyklės yra saugomos tarpinėje pakopoje, t. y. AS, tada, kai logika ar taisyklės gali būti efektyviai panaudotos daugelio taikomųjų programų. Sistemos administravimas, verslo ir apdorojimo logika tokiu būdu yra centralizuoti, o informacija reikalinga vartotojams yra paskirstoma per naršyklę. Nėra prasmės patalpinti visą programos logiką tarpiniame lygmenyje, nes tai padidintų tinklo apkrautumą, validuojant vartotojo įvedamus duomenis (vyktų sąveika tarp kliento ir serverio). Todėl dalis taikomosios programos logikos yra vykdoma lokaliai.

Žiniatinklio serverį galima būtų sutapatinti su tarpine pakopa. Nedidelei taikomųjų programų grupei pakanka naudoti servlet’us ir žiniatinklio serverio nustatymo programas („script“), bet šis sprendimas netinka sudėtingesniems sprendimams.

1.6. Taikomųjų programų serverio privalumai

- ✓ užkrauties balansavimas – galimybė siųsti užklausas skirtingiems serveriams priklausomai nuo prieinamumo prie serverio ir serverio užkrovimo trukmės;
- ✓ tinkama reakcijos į sistemos klaidą – nebelieka taip vadinamos “vieno taško“ problemos. Gali būti nustatyta sistemos reakcijos į klaidas politika tam tikram objektui ar jų grupei;
- ✓ saugumas. AS leidžia apsaugoti ir užkoduoti informaciją, siunčiamą tarp DB serverio, AS ir kliento. AS užtikrina vartotojo autentifikaciją, SSL sertifikatų apsikeitimą, vartotojų teises ir persiunčiamų duomenų kodavimą;
- ✓ patikimumas – automatinė klaidų paieška ir sistemos pakartotinis startavimas;
- ✓ pritaikomumas siekiant užtikrinti greitą naujų vartotojo poreikių realizaciją;
- ✓ administravimo įrankis – grafinis administravimo įrankis nutolusių vartotojų ir serverių darbo sekimui;
- ✓ AS gali susijungti su sistemomis realizuotas naudojant skirtingą aparatūrinę įrangą ir operacines sistemas.
- ✓ integruota vystymo aplinka;
- ✓ AS naudoja pakartotinio panaudojamumo modulius, tokius kaip EJB;
- ✓ sujungimų su DB „baseino“ technologija ir sujungimų laikinu saugojimu atmintyje.

- ✓ būsenų valdymas;
- ✓ transakcijų valdymas ir paskirstytos taikomosios programos;
- ✓ pranešimų tarp objektų talpinimas į eilę;
- ✓ verslo ir taikomosios programos logikos sąsaja;
- ✓ daugiagijiškumas ir lygiagretūs skaičiavimai siekiant maksimizuoti CPU išnaudojimą.

1.7. Į paslaugą orientuota architektūra

Paskutinį dešimtmetį kliento – serverio architektūra nėra pakankamai lanksti pateikiant sprendimus vartotojui kaip WS. Pramonėje iškilusiom techninėms problemoms spręsti vis daugiau taikoma į paslaugą orientuota architektūra. Ši architektūra aprašo nutolusių objektų ir funkcijų iškvietimą ir dinaminę paslaugų paiešką. Nutolusios funkcijos yra vadinamos paslaugomis.

Šioje architektūroje pagrindinis dėmesys yra kreipiamas į objektų tarpusavio sąveiką. Prieš sukuriant šią architektūrą buvo naudojama nutolusios procedūros iškvietimo (RPC) architektūra. Nutolusios procedūros iškvietimo technologija leidžia taikomosioms programoms kviesti nutolusias funkcijas taip pat kaip lokalias. Privalumas tas, kad sistemos kūrėjui nereikia rūpintis apie skirtingas operacines sistemas, duomenų perdavimo tinklo protokolus ir programavimo kalbas, kuriomis realizuoti sistemos komponentai. Tai suteikia galimybę sukurti aukšto pakartotinio panaudojamumo komponentus.

Į paslaugą orientuota architektūra yra lengvai pritaikoma ir lanksti. Jos pagrindas yra komponentų agregacija. Komponentas – tai programinio kodo dalis, kuri atlieka nurodytą funkciją. Komponentas turi griežtai apibrėžtą sąsają ir daugeliu atveju tik vieną užduotį, kurį realizuoja. Paslauga yra grupė komponentų, atliekančių tarpusavyje susijusį darbą. Šie komponentai yra vykdomosios programos. Programinės įrangos industrijos vizija yra gamintojų komponentų rinka. Kuriant naują sistemą, perkami reikalingi komponentai ir tarpusavyje sujungiami. Šios vizijos privalumas yra tas, kad kritiškai sumažėja naujo funkcionalumo programavimas. Toks taikomosios programos kūrimo metodas yra orientuotas į verslo reikalavimų išpildymą.

Ši architektūra sumažina kaštus panaudojant visus turimus resursus – programuotojus, programavimo kalbas, operacines sistemas, aparatūrines platformas, duomenų bazes ir duomenų bazių taikomąsias programas. Egzistuojančias sistemas daug paprasčiau suskirstyti į komponentus ir juos modifikuoti, negu perrašyti visą sistemą iš naujo.

Paskirstytų objektų protokolai tokia kaip CORBA, RMI išsivystė iš RPC technologijos kaip standartinių funkcijų iškvietimo papildas.

1.8. Į paslaugą orientuotos architektūros privalumai

Paskirstytos taikomosios programos trūkumas yra tas, kad jas pakankamai sudėtinga išvystyti. Ši problema į paslaugą orientuotoje architektūroje sprendžiama dinamine paslaugų paieška per paslaugų registrą, kuris aprašo paslaugos įėjimų ir išėjimų sąsajas. Paslaugų registras palengvina reikalingo funkcionalumo paslaugų paiešką, vykdomą galutinio vartotojo. Šiuo metu programinių ir aparatūrinių komponentų konfigūracija yra statinė. Informacija apie paslaugą yra dažnai įsiūnama į programinius komponentus arba saugoma sistemos konfigūracijos failuose. Nėra vienos tinklinės sistemos visiškai statiškos. Tik tuo atveju, jei pasikeitimai yra minimalūs įmanoma išlaikyti sistemos darbą su statine konfigūracija. Augant sistemų dinamiškumo poreikiui, iškyla problemų norint administruoti sistemas, kurių konfigūracija nustatoma statiškai. Problemos sprendžiamos 2 būdais: paslaugos išankstine paieška ir atradimu. Tuomet konfigūracijos duomenys nėra statiškai suvedami, tiesiog komponentai susiranda reikiamas paslaugas iš anksto, programos nustatymo metu ir atitinkamai, pagal rezultatus, nustato sistemos darbo aplinką.

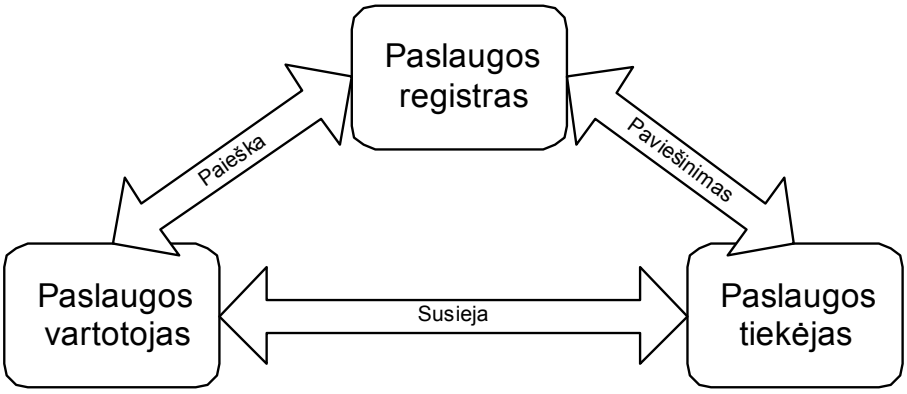
Žiniatinklio paslaugos yra į paslaugą orientuotos architektūros pagrindas. Jos sukuriamos panaudojant paskirstytą, į komponentus orientuotą aplinką. Jų tikslas yra supaprastinti pakartotinį naudojamų technologijų resursų panaudojamumą ir palengvinti naujų taikomųjų sistemų kūrimą. Į paslaugą orientuotos architektūros privalumai yra:

- ✓ produktyvumas vystant naujas taikomąsias sistemas;
- ✓ paslaugų patikimumas ir tikslumas;
- ✓ supaprastintas testavimas;
- ✓ greita taikomųjų programų integracija su žiniatinklio paslaugomis;
- ✓ pakartotinis paslaugų panaudojamumas;
- ✓ sumažinti kaštai ir rizika;
- ✓ lengvesnis sistemos darbo režimo sekimas;
- ✓ mažiau techninio palaikymo darbų.

1.9. Pagrindinės į paslaugą orientuotos architektūros esybės

Į paslaugą orientuotos architektūros standartai yra XML, SOAP, UDDI ir WSDL. Šie standartai pateikia bendrą komunikacijos aprašymo kalbą tarp naudotojo, tiekėjo ir registruotojo. Taigi susiformavo trys pagrindinės esybės, kurios sąveikauja tarpusavyje, pateikdamos tarpusavyje mažai susieto („*loose coupling*“) skaičiavimo paradigmą:

- ✓ paslaugų tiekėjai;
- ✓ paslaugų vartotojai;
- ✓ paslaugų registruotojai.



4 pav. Į paslaugą orientuota architektūra

1.10. Į paslaugą orientuotos architektūros paslaugos

Loginis komponentų grupavimas siekiant patenkinti verslo reikalavimus yra vadinamas paslauga. Tai yra veiksmas arba funkcija, kuri atliekama kaip dalis sandėrio tarp paslaugos tiekėjo ir jos klientų. Paslauga – tai bet kokio tipo informacijos šaltinis, kuris gali būti naudojamos iš taikomosios programos ar galutinio vartotojo, patogumo dėlei patalpintas į rinkinį. Prieš naudojant paslaugą ji turi būti užregistruota. Taip pat turi būti apibrėžta sąsaja su paslauga. Paslaugos susideda iš 2 dalių:

- ✓ išorinės sąsajos. Išorinė sąsaja aprašo, kaip naudoti pateikiamą paslaugą. Ji specifikuoja priėjimą prie paslaugos ir apibrėžia jos įėjimo reikšmių diapazoną bei rezultatus, grąžinamus atgal;
- ✓ vidinio funkcionalumo. Paslaugos yra surenkamos iš keleto komponentų – tokiu būdu paslaugos funkcionalumas nusako, kaip tie komponentai tarpusavyje sąveikauja. Paslaugos funkcionalumas yra talpinamas pačioje paslaugoje taip, kad vartotojai sąveikautų su paslauga tik per sąsają, o funkcionalumas nuo jų būtų paslėptas;

Yra keletas metodų kaip pasinaudoti verslo paslaugomis:

- ✓ kontraktų. Kontraktas – tai apibrėžtas susitarimas, kuris nusako pačią sąsają. Sąsajos aprašymo kalba (IDL) yra kaip vienas iš galimų pavyzdžių;
- ✓ pranešimų. Pranešimas – tai išbaigtas duomenų blokas, dažniausiai su antraštės sekcija kurioje saugomi logistinės transporto detalės. Jie naudojami asinchroninėse komunikacijose ir tinka ten, kur reikalingas nesusietas bendravimas tarp objektų;
- ✓ saugyklos. Saugykla – tai bendras nukreipimo taškas į kryžminius komponentus ir kryžminius paslaugų resursus. Šis apibendrinta nuoroda leidžia dinaminį paslaugos apibrėžimą ir valdymą.

1.11. Žiniatinklio paslaugos

Žiniatinklio paslauga nėra vizualiai matomas komponentas, kaip pavyzdžiui žiniatinklio puslapiai, bet iš jos kaip duomenų šaltinio į žiniatinklio puslapį yra perduodami duomenys. Šios paslaugos pakeičia būdą kaip pateikiami žiniatinklio resursai: kompanija – kompanija („business–to–business“) ir kompanija – vartotojas („business–to–consumer“). Žiniatinklio paslauga gali būti laikoma programine įranga, kuri sugeba sąveikauti per tinklą su kito tipo programine įranga, o tiksliau programa su tam tikru funkcionalumu. WS funkcionalumas pateikiamas taip, kad kitos taikomosios sistemos galėtų juo naudotis. WS išsivystė iš RPC technologijos, tačiau su tam tikrais skirtumais:

- ✓ perduodami duomenys suformatuojami naudojant XML. Todėl tapo nebereikalingas persiunčiamų duomenų pakavimas ir išpakavimas;
- ✓ WS gali būti surastos naudojant UDDI, o aprašomos naudojant WSDL;
- ✓ Duomenims persiųsti naudojami HTTP ir SMTP. Šie protokolai turi griežtai apiubrėžtus standartus.

WS nėra tik verslo modelis. Tai naujos technologijos vystymo modelis, kuris nepriklauso konkrečiam verslo modeliui. Taip pat WS nėra tobula technologija, tiesiog, tai – žingsnis į priekį, naujas paslaugų pateikimo būdas. WS sudaryta iš save aprašančių komponentų, kurie sugeba automatiškai atlikti paiešką žiniatinklyje ir naudotis kitomis paslaugomis ar taikomosiomis sistemomis, siekiant realizuoti nubrėžtą tikslą. Pagal „Stencil Group“ WS apibrėžiama orientuojantis į konteksto specifiką ir koncepciją:

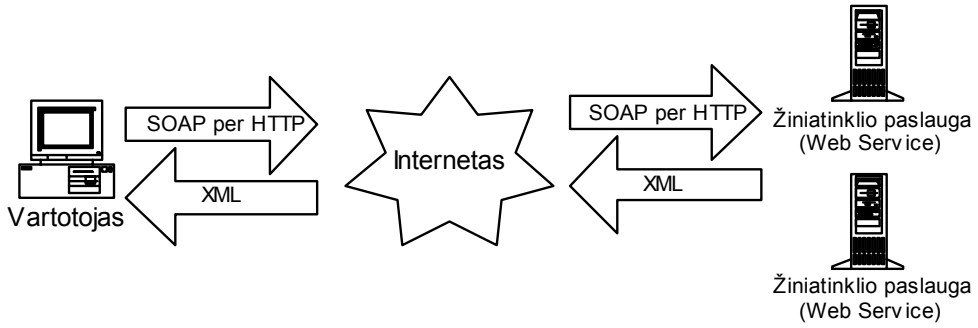
- ✓ **specifinė reikšmė.** Tai sąrašas besivystančių standartų, kurie aprašo į paslaugą orientuotą, komponentinę taikomųjų programų architektūrą;
- ✓ **konceptuali reikšmė.** Tai modelis, kuriame e–komercijos procesų užduotis yra paskirstomos Internete.

Tradicinės taikomosios programos susideda iš tvirtai tarpusavyje sujungtų komponentų. Tačiau tada programuotojams yra būtina turėti supratimą apie abi sujungimo puses. WS programinės įrangos komponentai tarpusavyje yra silpnai susieti, todėl jie gali dinamiškai susirasti ir naudoti kitus komponentus Internete. WS gali būti aprašomos, publikuojamos, surandamo, nustatoma jų konfigūracija per žiniatinklį. Be to, jas paprasčiau valdyti. Kompanijos gali automatizuoti daug verslo procesų, sukuriant didesnį operacinį efektyvumą. WS technologija suteikia šiuos privalumus:

- ✓ greitas e–komercijos taikomųjų programų sukūrimas, sumažinant kaštus ir sukūrimo laiką;
- ✓ sustiprinta verslo ir IT tarpusavio sąveika;
- ✓ investicijų išsaugojimas;
- ✓ sisteminės kvalifikacijos reikalavimų sumažinimas;

- ✓ mažesnis sistemos sudėtingumas naudojant inkapsuliaciją;
- ✓ sąveika su ankstesnėmis sistemomis;
- ✓ padidintos verslo galimybės;
- ✓ pagerintas ryšys su vartotoju;
- ✓ dauguma gamintojų palaiko SOAP, o tuo pačiu ir WS technologiją;
- ✓ šie standartai yra lengvai įsisavinami;
- ✓ sukurta daug standartinių įrankių, padedančių greitai sukurti ir pateikti vartotojui naujas WS.

Kaip minėta ankščiau, WS inkapsuliuoja informaciją ir diskretinį funkcionalumą (galima sulyginti su „applet“, kuris atlieka vieną užduotį). WS aprašo savo įėjimus ir išėjimus, iškvietimo būdus ir rezultatus taip, kad kita programinė įranga galėtų naudotis jų pateikiamu funkcionalumu. WS taikomųjų programų logika yra programuojama ir pasiekiama per standartinius Interneto protokolus. Taip apjungiama komponentinio ir žiniatinklio programavimo privalumai. WS naudoja daug įvairių būdų komunikacijoms užtikrinti, tačiau pagrindinis transporto pakopos mechanizmas yra XML ir HTTP.



5 pav. Kliento komunikacija su WS

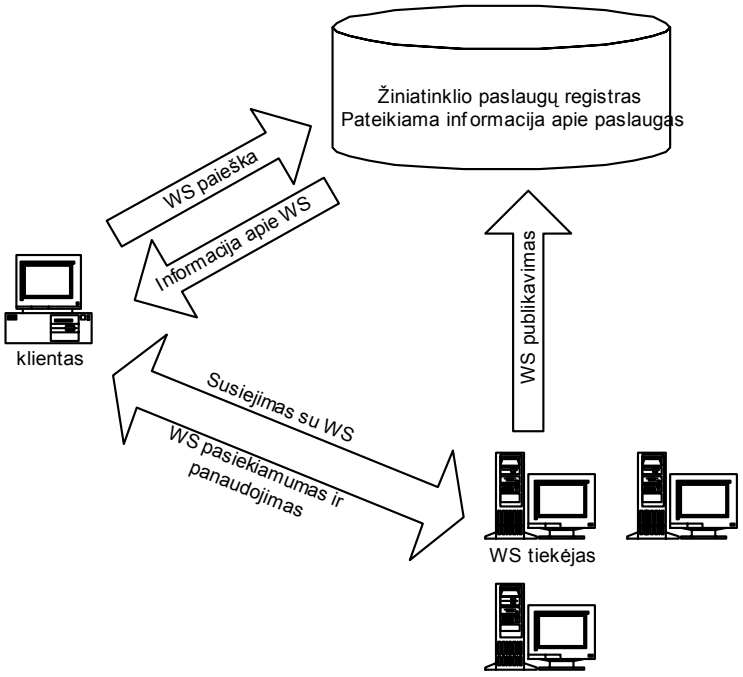
1.12. Žiniatinklio paslaugos komponentai

WS tapo taip plačiai naudojamos dėl to, kad WS suprogramuota Java kalba gali būti naudojama iš kitos WS, parašytos C kalba, kitame taikomųjų programų serveryje ir nepaisant to, kaip WS yra realizuota. WS funkcionalumas yra paslepiamas, naudojamas juodos dėžės principas[3]. Programuojant sistemą kaip WS, nereikia sukurti visos sistemos iš naujo, išrenkama dalis komponentų, atliekančių dalinį funkcionalumą ir atliekama komponentų funkcijos praplečiamos (jei reikia), jie tarpusavyje surišami, atliekama kompozicija. WS pateikia verslo objektus (EJB, COM objektus), prieinamus SOAP per HTTP. Taip vartotojai gali kviesti nutolusias šių objektų funkcijas. Jokia grafinė sąsaja nepateikiama, nes WS nėra skirtas tiesioginiam bendravimui su vartotoju. Tam reikalinga kita programinė įranga, o WS operuojama kodo lygmenyje.

[3] Juoda dėžė – („black box”) – sistemai nurodomos įėjimo reikšmės ir gaunami rezultatai išėjimuose, tačiau nesigilinama, kaip yra gaunami rezultatai.

Programuotojai, kurdami WS, į pagalbą gali pasitelkti J2EE įrankinę. Tuo pačiu egzistuojantys žiniatinklio puslapiai ar J2EE taikomosios programos gali būti praplečiamos ir pagerinamos. Naudojami trys pagrindiniai modeliai, nusakantys paskirstytą skaičiavimą:

- ✓ paslaugos registravimo mechanizmas;
- ✓ paslaugos paieškos mechanizmas;
- ✓ dviejų dalių tarpusavio komunikacijos mechanizmas.



6 pav. WS architektūros vaizdas

WS tiekėjas pateikia WS aibę. Informacija apie WS yra publikuojama ir centralizuotai saugoma WS saugykloje. Kada WS yra eksportuojama į WS saugyklą, klientas ją gali naudoti. Klientai savo ruožtu atlieka WS saugykloje reikalingos WS paiešką. Po to klientas susiejamas su WS ir gali naudotis jos teikiamu funkcionalumu. WS iškviečiamos statiniu būdu, naudojant WSDL paslaugos sąsają ir WS realizacijos dokumentus. Jei WS tipo apibrėžimas ir realizacija nuskaitoma per UDDI, WS gali būti iškviečiama dinamiškai. Taip užtikrinamas saugumas ir kokybė, vartotojo autentifikacija ir transakcijų valdymas.

HTTP

HTTP yra visuotinai Internete naudojamas protokolas. Jis suteikia galimybę žiniatinklio naršyklėms ir serveriams sąveikauti per TCP/IP sujungimą. Viskas, kas vyksta žiniatinklyje, yra apgaubta HTTP transakcijomis. Kaip ir kitų standartinių Interneto protokolų kontrolinė informacija yra perduodama kaip nekoduotas tekstas per TCP/IP sujungimą. HTTP pasiunčia naršyklės

užklausą į žiniatinklio serverį ir nukopijuoja serverio atsakymą atgal į naršyklę. HTTP yra maksimaliai orientuotas į pristatymo paslaugą. Čia naudojamas neturintis būsenos duomenų persiuntimo mechanizmas.

XML

XML tai specifikacija kalbos, skirtos kurti žiniatinklio puslapiams. Ši specifikacija yra kaip mechanizmas apibrėžiantis dokumento struktūras. Pasaulyje vis daugėja taikomųjų programų, kurios naudoja XML dokumentus. Priešingai nuo HTML, XML nespecifikuoja nei semantikos nei žymių aibės. XML pateikia galimybę apibrėžti žymes ir jų tarpusavio santykius. XML dokumento semantika yra apibrėžiama pačios taikomosios programos, kuri ją apdoroja. XML sukūrimo pagrindinis tikslas buvo dokumentų su specifikuota struktūra naudojimas žiniatinklyje. XML privalumai:

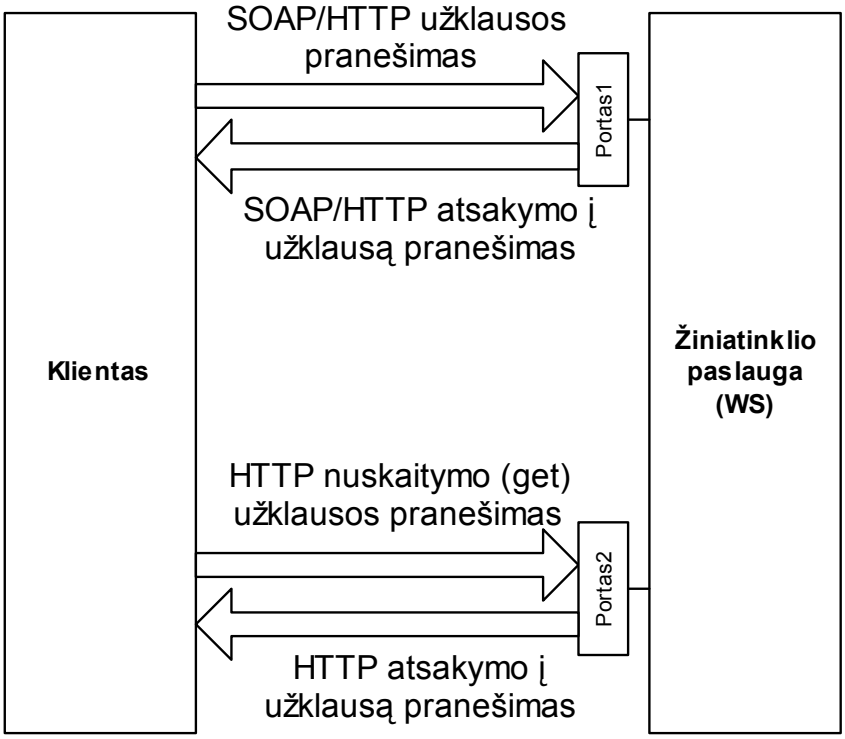
- ✓ sumažėja taikomojo programavimo apimtis;
- ✓ validavimui skirtas XML interpretatorius automatiškai patikrina dokumento sintaksę ir pritaiko verslo taisykles pagal jam pateiktą DTD. Taikomajai programai telieka validuoti simbolinius duomenis tarp žymių;
- ✓ XML dokumentai gali aprašyti patys save. Tekstinė XML žymių prigimtis ir teisingai apibrėžto DTD naudojimas labai smarkiai sumažina spėliojimais pagrįsto darbo programuojant transliatorių;
- ✓ XML suteikia programuotojams galimybę sukurti atviras, standartizuotas sąsajas egzistuojančioms sistemoms, panaudojant plačiai prieinamus ir patikimus įrankius.

SOAP naudoja XML ir dėlto gaunama papildoma apkrova, nes XML suformuoti duomenys yra 4 kartus didesni, nei atvaizdavimas yra keturi kartus didesni nei binariniai. Šis duomenų padidėjimas veda prie duomenų perdavimo tinklu trukmės ilgėjimo. Naudojant WS, duomenų perdavimo trukmė turi būti nedidelė, nes priešingu atveju taikomosios programos bus lėtos. Todėl dažnai perduodami XML duomenys yra suspaudžiami.

SOAP

SOAP naudojimas iš esmės pakeičia visą žiniatinklio taikomųjų programų kūrimo procesą. SOAP – tai protokolo specifikacija, skirta perduodamų XML duomenų kodavimui. Tuo pačiu tai mechanizmas, skirtas nutolusiom procedūrom iškviesti. SOAP naudoja HTTP kaip pagrindą duomenų perdavimui. SOAP apjungia XML ir HTTP kviečiant metodus, realizuotus skirtingose tinklų platformose. SOAP nubrėžia modelį, kuris naudoja paprastus užklausos/atsakymo pranešimus parašytus XML.

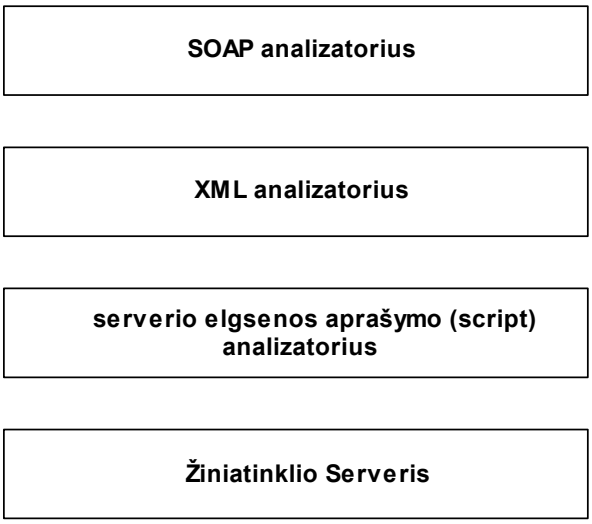
SOAP yra nepriklausantis nuo platformos nutolusios procedūros iškvietimo mechanizmas, nepaisant kokią OS, objektų modelį ar programavimo kalbą naudoja taikomosios programos. Nutolusios procedūros vykdymas yra tapatus užklausos į nutolusį kompiuterį siuntimui. Įvykdžius užklausą, nutolusio kompiuterio programa siunčia atsakymą. Nutolusios procedūros vykdymas yra baigtas tada, kai užklausiančioji programa gauna atsakymą į užklausą. SOAP dirba pagal šį užklausos/atsakymo modelį. Kliento užklausa susideda iš teisingo SOAP XML dokumento, kuris per HTTP siunčiamas į serverį. Kai serveris gauna SOAP užklausą, jis ją išanalizuoja ir validuoja. Po to nurodytas metodas yra įvykdomas su perduotais parametrais, ir serveris suformuoja atitinkamą SOAP XML atsakymą, kuris siunčiamas atgal per HTTP.



7 pav. WS aktyvacija per SOAP/HTTP get

SOAP naudoja keletą HTTP antraščių, kurios suteikia „proxy“ ar „firewall“ filtravimą. SOAP yra labai svarbus, nes XML žiniatinklis pagal specifikaciją turi būti atviras, išplečiamas ir nepriklausomas nuo platformų. Kadangi SOAP pagrindas yra HTTP, todėl jis yra lengvai išplečiamas. Nors ir atliekamas SOAP paketų dekodavimas ir išanalizuota juose saugoma XML informacija, tačiau SOAP duomenų perdavimo charakteristikos yra pakankamai priimtinos. Saugumas SOAP nerealizuotas, tačiau SOAP gali naudoti per HTTP saugius “socket“us.

SOAP susideda iš SOAP antraštės ir kūno. Antraštė nėra privaloma, tačiau jei ji egzistuoja, tuomet ten saugoma vienas ar daugiau antraštės elementų, suteikiančių meta–informaciją reikalingą metodo iškvietimui. Kūne saugoma užkoduota nuosekli informacija apie metodo parametrus. Metodo iškvietimo XML elementas turi turėti tą patį vardą, kaip ir nutolęs metodas.



8 pav. SOAP per HTTP serverio komponentai

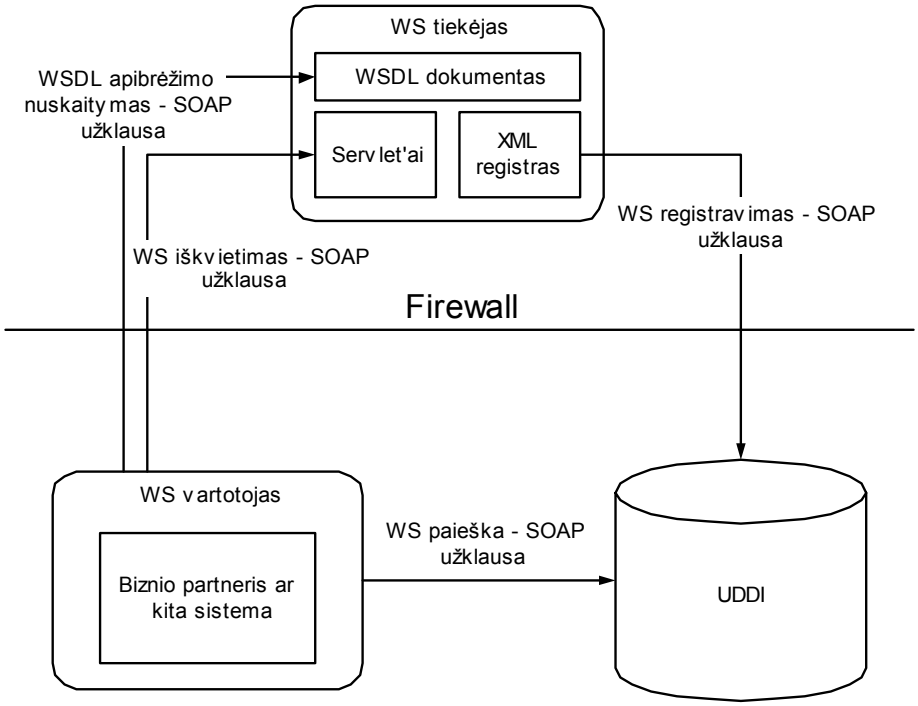
WSDL

WSDL aprašo WS kaip aibę galinių taškų, dirbančių pranešimų pasikeitimo režimu, o kartu ir kaip tie taškai gali būti pasiekiami. WSDL naudoja XML kaip duomenų perdavimo formatą ir yra UDDI pagrindinė sudedamoji dalis. Pranešimai turi informaciją orientuotą į funkcijas arba į dokumentus. WSDL abstrakčiai aprašo operacijas ir pranešimus. Tada tos operacijos yra susiejamos su konkrečiu tinklo protokolu ir pranešimo formatu, taip apibrėžiant galinį tašką. WSDL dokumento specifikacija pagerina taikomųjų sistemų tarpusavio sąveiką. Nebelieka skirtumo, koks protokolas ar kodavimo schema yra naudojama. WSDL dokumentas aprašo, kaip iškviesti paslaugą, ir pateikia informaciją apie metodus ir duomenis, kuriais apsikeičiama. Dar daugiau WSDL saugo pranešimų seką kiekvienai operacijai, protokolo susietumo ir paslaugos saugojimo vietos informaciją. WSDL atskiria abstraktų apibrėžimą nuo konkrečios realizacijos. WSDL panaudojimo atvejai WS architektūroje atvaizduojami 9 pav. Konkretus protokolas ir duomenų formatas išeities/įeities taškui sukuria pakartotinai panaudojamą susietumą. Šis išeities/įeities taškas apibrėžiamas susiejant tinklo adresą su pakartotinai panaudojamu susietumu. Rinkinys tokių išeities/įeities taškų sudaro WS. WSDL dokumentas turi specifinius elementus:

- ✓ tipus. Duomenų tipų apibrėžimų saugykla. Aprašomi tipai, naudojami pranešimuose;
- ✓ pranešimus. Aprašomi duomenys perduodami iš vieno taško į kitą, vykdant iškvietimą;
- ✓ operacijas. Apibrėžia įėjimo, išėjimo ir klaidos pranešimų kombinacijas;
- ✓ išeities/įeities taško tipus. Apibrėžia operacijos rinkinį, kuris palaikomas viename ar kitame galiniame taške;
- ✓ susietumą arba, kitaip tariant, protokolą, naudojamą WS komunikacijai. Palaikomi protokolai: SOAP, HTTP, GET, HTTP POST, MIME;

- ✓ išeities/įeities tašką. Vienas galinis taškas – susietumo ir tinklo adreso kombinacija;
- ✓ paslaugą. Susideda iš išeities/įeities taškų kombinacijos. Kiekviena paslauga susiejama su vienu išeities/įeities taško tipu ir pateikia skirtingus būdus kaip iškviesti operacijas tam išeities/įeities taškui;
- ✓ įėjimą. Pranešimas siunčiamas į serverį;
- ✓ išėjimą. Pranešimas siunčiamas klientui;
- ✓ klaidą. Klaida grąžinama vietoj rezultato, apdorojant pranešimą.

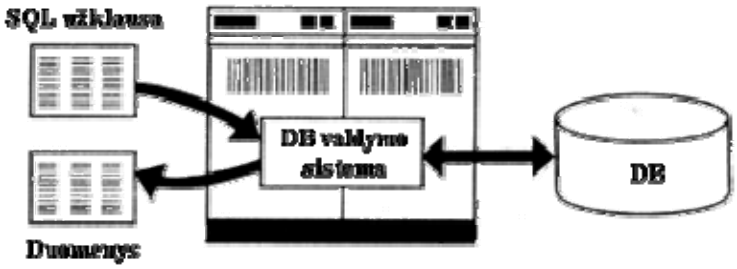
Pranešimai nėra nepriklausomi nuo protokolo, Jie gali būti naudojami kartu su SOAP, HTTP, GET ar kitu protokolu. Tačiau apibendrinant WS privalo turėti savyje aprašytus 2 tipų pranešimus: įėjimo ir išėjimo. Įėjimo pranešimas yra siunčiamas iš kliento į paslaugą, o atsakant klientui siunčiamas išėjimo pranešimas. Nėra reikalo aprašyti WSDL dokumentus ranka, yra daug programinės įrangos įrankių, automatizuojančių šią užduotį.



9 pav. WSDL bendras vaizdas

1.13. MySQL duomenų bazės serveris

SQL – tai instrumentas, skirtas duomenų, esančių kompiuterinėje duomenų bazėje nuskaitymui ir apdorojimui. SQL (Structured Query Language) – struktūrizuota užklausų kalba.
SQL dirba tik su vieno tipo duomenų bazėmis, t.y. reliacinėmis DB 10 pav. pavaizduota SQL darbo schema:



10 pav. SQL panaudojimas dirbant su DB

Kompiuterinė programa, valdanti DB vadinama duomenų bazės valdymo sistema (DBVS). Vartotojas SQL pagalba kreipiasi į DBVS, kuri apdoroja užklausą, randa reikalingus duomenis ir pasiunčia juos vartotojui. Ši procedūra vadinama DB užklausa – iš čia ir pavadinimas struktūrizuota užklausų kalba.

Tačiau dabar SQL panaudojama ne tik užklausų sudarymui, ji naudojama visų funkcinių galimybių, kurias vartotojui suteikia DBVS, realizavimui:

- ✓ duomenų organizavimas. SQL suteikia vartotojui galimybę keisti duomenų pateikimo struktūrą, taip pat nustatyti santykius tarp DB elementų;
- ✓ duomenų nuskaitymas. SQL suteikia vartotojui galimybę gauti duomenis esančius DB ir jais naudotis.

Duomenų apdorojimas. SQL pagalba galima keisti DB turinį, t.y. įvesti naujus duomenis, trinti nebereikalingus, atnaujinti senus.

Priėjimo prie duomenų valdymas. SQL pagalba galima apsaugoti duomenis nuo nesankcionuoto vartojimo, apriboti vienų ar kitų vartotojų galimybes dirbant su DB.

Kolektyvinis darbas su DB. SQL suteikia galimybę keliems vartotojams vienu metu naudotis ta pačia DB netrukdant vienas kitam.

DB apsauga. SQL padeda užtikrinti DB vientisumą apsaugodama ją nuo sugriovimo dėl įvairių nesuderintų pakeitimų DB ar tiesiog atsakius DBVS.

Visgi SQL yra nepilnavertė kalba kaip pavyzdžiui COBOL, FORTRAN ar C. SQL neturi operatoriaus IF sąlygos patikrinimui; GOTO – perėjimų ir nuorodų organizavimui; DO arba FOR – ciklų sudarymui. SQL yra DB pokalbe turinčia apie 30 operatorių skirtų valdyti DB. SQL operatoriai įsiterpia į bazinę kalbą, pavyzdžiui COBOL, FORTRAN ar C ir suteikia galimybę naudotis DB.

Nežiūrint į ne visai tikslų apibrėžimą SQL šiandien yra vienintelė standartinė kalba skirta dirbti su reliacinėmis DB.

1.13.1. SQL vaidmuo

Pačios SQL negalime pavadinti nei DBVS, nei atskiru programiniu produktu. SQL yra neatsiejama DBVS dalis, instrumentas, kurio pagalba realizuojamas vartotojo ryšys su DBVS:

SQL – tai interaktyvi užklausų kalba. Vartotojas panaudoja SQL komandas interaktyviose programose, skirtose duomenų nuskaitymui ir atvaizdavimui ekrane. Tai patogus būdas atlikti specialias užklausas.

SQL – tai duomenų bazių programavimo kalba. Tam, kad gauti priėjimą prie DB programuotojai į savo programas įveda SQL komandas. Ši metodika naudojama tiek vartotojų kuriamose programose, tiek tarnybinėse duomenų bazių programose (kaip pvz. Ataskaitų generatoriuose arba duomenų įvedimo įrankiuose).

SQL – duomenų bazių administravimo kalba. DB administratorius, esantis kompiuteryje, naudoja SQL DB struktūros nustatymui ir vartotojų sąsajos su DB valdymui.

SQL – priedų klientas/serveris sudarymo kalba. Programose personaliniams kompiuteriams SQL naudojamas ryšio tarp kompiuterių ir DB serverio lokaliniame tinkle organizavimui. Dabar dažnai panaudojama kliento/serverio architektūra. Tai leidžia iki minimumo suvesti tinklo apkrovimą kada vienu metu ta pačia DB naudojasi daug kompiuterių, taip pat paspartina kaip personalinių kompiuterių, taip ir serverio darbą.

SQL – paskirstytų duomenų bazių kalba. Tokių DB valdymo sistemose SQL padeda paskirstyti duomenis tarp kelių atskirų duomenų sistemų. Ryšys tarp jų organizuojamas naudojant SQL užklausas.

SQL naudojamas kaip duomenų bazių šliuzas. Atskiruose kompiuterių tinkluose su skirtingomis DBVS SQL naudojamas šliuzo programoje, kurios pagalba organizuojamas ryšys tarp skirtingų tipų DBVS.

Tokiu būdu SQL tapo galingu instrumentu naudojantis informaciją, esančia reliacinėse DB. Be to SQL – lengvai suprantama ir universali programinė duomenų valdymo priemonė.

1.13.2. Pagrindiniai SQL privalumai:

Nepriklausoma nuo konkrečių DBVS;
standartizuota;
perkėlimo iš vienų kompiuterinių sistemų į kitas galimybė;
pripažinta IBM (DBVS DB2) ir Microsoft (ODBC protokolas);
reliacinis pagrindas;
aukšto lygio struktūra;
duomenų pateikimo įvairovė;

galimybė dinamiškai keisti ir plėsti DB struktūrą tuo pat metu naudojantis jos turiniu;

galimybė atlikti specialias interaktyvias užklausas;

programinio priėjimo prie duomenų bazių užtikrinimas;

pilnavertė kalba darbui su DB;

palaiko kliento/serverio architektūrą.

1.13.3. Pagrindiniai SQL trūkumai

SQL neturi operatoriaus IF sąlygos patikrinimui; GOTO – perėjimų ir nuorodų organizavimui; DO arba FOR – ciklų sudarymui.

1.13.4. Microsoft Access 2000

Tai viena iš populiarių duomenų bazių. Ji priklauso standartiniam Microsoft Office 2000 paketui. Šių versijų yra 97, 2000, Xp.

Duomenų bazė yra valdoma įprastais būdais: visos operacijos yra parenkamos standartinėmis OS Windows terpės priemonėmis – pele bei klaviatūra valdomomis meniu lentelėmis, įrankių juostomis, dialogo langais.

Duomenų bazės elementai turi juos tarpusavyje siejančius ryšius.

Kiekviena duomenų bazė turi „raktą“. Duomenų bazės raktas – tai kitas įrašo reikšmes vienareikšmiškai identifikuojantis duomenų tipas.

Raktas susidedantis iš kelių duomenų elementų tipų yra sudėtinis raktas.

Duomenų bazės duomenys yra saugomi dvimatėse lentelėse. Šios lentelės yra vadinamos (Forms).

Kortelę sudaro laukeliai (Fields) reikalingiems duomenims įvesti. Visi vienoje kortelėje esantys duomenys sudaro duomenų bazės įrašą (Record).

Vieno tipo kortelės dedamos į joms skirtą segtuvą. Segtuvo turinys atvaizduojamas lentele (Table). Vienoje segtuvo lentelės eilutėje (Record) būna informacija, paimta iš vienos to paties segtuvo kortelės, o lentelės stulpelyje (Field) – iš visų kortelių to paties tipo laukelių. Duomenų bazėje gali būti daug segtuvų su įvairių tipų kortelėmis, taip pat įvairių išvestinių lentelių.
Informaciją, prieš išvedant į ekraną arba išspausdinant, galima įvairiai apdoroti, pavyzdžiui, iškviesti tik tam tikrus požymius (Criteria) turinčias korteles, duomenų sąrašus (List), ataskaitas (Reports), sukurti užklausas (Query).

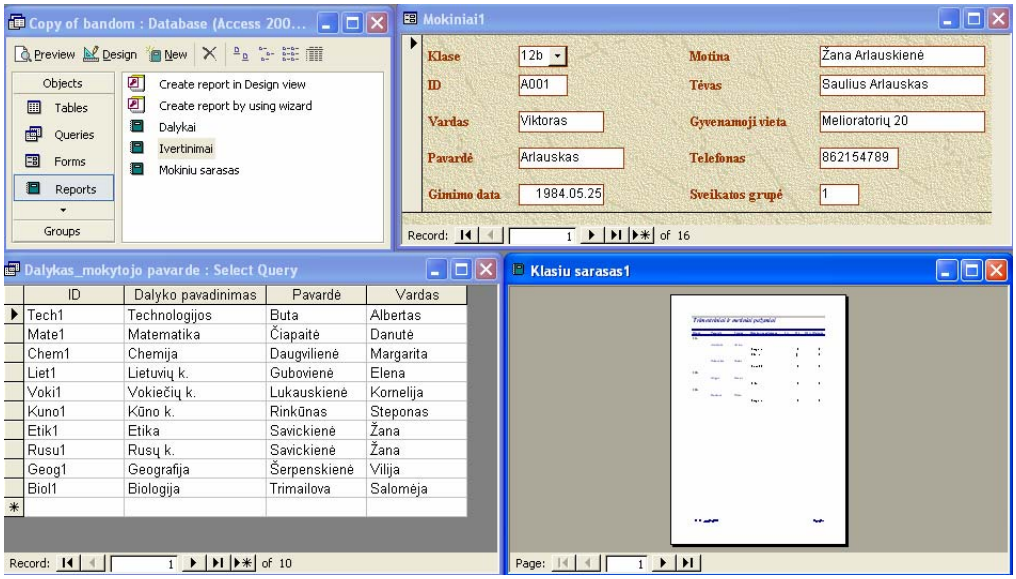
Kompiuterinė duomenų bazė patogi tuo, kad lengva kurti ir keisti korteles ištrinti nereikalingus bei įterpiant naujus laukelius duomenims įrašyti, labai lengva rūšiuoti ir išrinkti reikalingus duomenis, juos analizuoti įvairiais aspektais.

1.13.5. Access duomenų bazės privalumai:

greitis;
lengva naudotis;
galima saugoti labai daug duomenų;
lengva papildyti ir redaguoti duomenis;
automatinis duomenų perskaičiavimas;
lengvai paieška ir duomenų atrinkimas;
galimybė įvairiai atvaizduoti saugomus duomenis;
duomenų apsikeitimas tarp įvairių programų;
tinkle vienu metu tais pačiais duomenimis gali naudotis daug vartotojų;
nėra duomenų dubliavimosi.

1.13.6. Trūkumai:

Lėtai dirba su dideliais duomenų kiekiais. Gali „pakibti“.



11 pav. Access darbastalis

1.14. Firebird

„Firebird“ yra gana populiari komercinio DB serverio „Borland Interbase 6.x“ atviro kodo atšaka. Tai duomenų bazės valdymo sistema, pasižyminti greičiu ir siūlanti daugelį ANSI SQL–92 ypatybių. Ji veikianti „MS Windows“, „Linux“ bei kitose „UNIX“ platformose. Ji yra galinga ir tuo

pačiu metu mažai reikalaujanti konfigūracijos bei administravimo. Vienas „Firebird“ serveris gali dirbti su daug atskirų duomenų bazių.

Privalumai:

- nemokama atviro kodo programa;
- galingos saugojimo procedūros;
- pilna operacinė kontrolė;
- dirba keliais priėjimo lygiais;
- nėra jokios būtinybės uždaryti bazę jos tvarkymui;
- naudojimosi bibliotekos gali būti parašytos C, C++, Delphi, Piton kalbomis.

1.15. PHP

PHP yra HTML pagrindu veikianti scenarijų rašymo kalba. Nors PHP paprasta naudotis, tai labai galinga kalba. Joje derantys vieni geriausių šiuolaikinių programavimo bruožų daro ją unikalia ir įdomia interneto programų rašymo priemone. PHP nuo pat pradžių buvo numatyta kurti tinklalapiams. Dinamiškos, duomenimis pagrįstos taikomosios programos ir interneto svetainės tapo labi svarbios organizuojant bendravimą ir verslą. PHP yra nemokama ir veikia beveik visose populiariausiose techninės įrangos platformose. PHP galima derinti su daugybe atvirojo kodo sąryšio duombazių valdymo sistemomis (RDBVS), pavyzdžiui, MySQL, PostgreSQL ar Interbase, ir turėsime galingą, visiškai nemokamą projektavimo platformą. Be to prieinamas pirminis kodas. Tokiu atveju su PHP galima kurti ir nedidelius projektus, ir labai didelės apimties programas.

PHP buvo keletas versijų. Dabartiniu metu naudojama PHP 4.1. Viena iš šios versijos pastebimiausių ir svarbiausių priemonių – nepaprastai sparti scenarijų rašymo sistema „Zend“. „Zend Technoloegies“ gamina daugybę įvairių PHP skirtų priedų, tarp jų ir „Zend Cache“. Daugybė patobulinimų ir priedų buvo sukurti ir PHP branduoliui. Kai kurie priedai neteikiami nemokamai ir palaikomi komercinių įmonių. Tačiau dauguma naujovių pateikiamos atviruoju kodu ir nemokamai. Atsirado ir kelios naujos bibliotekos ir programavimo sąsajos.

PHP yra išaugusi į tikrą elektroninės komercijos sprendimą: ji pripažįstama ir dažniausiai naudojamu Apache moduliu. Remiantis 2002 m. sausio mėn. korporacijos „E–soft“ atliktos apklausos duomenimis, kaip modulis PHP įdiegta į daugiau nei 46% visų Apache sistemų. Iš visų serveriams skirtų programavimo priemonių artimiausia varžovė yra mod_perl, įdiegta beveik 20% visų apklausoje dalyvavusių Apache serverių.

Versijos 4.1 naujovės. Lyginant su versija 4, naujoje PHP versijoje yra keli svarbūs pakeitimai. Vienas svarbiausių pakeitimų susijęs su apsauga, t. y. pakeistas prieigos prie vartotojų

įvesties duomenų būdas. Pagal naująjį modelį „išorinės“ reikšmės, tokios kaip per formas pateikti ir slapukuose esantys duomenys, dabar prieinamos specialiuose visuotiniuose masyvuose:

1 lentelė. PHP kintamieji

Masyvas	**Turinys**
$_GET	Bet kokios reikšmės, pateiktos naudojant metodąč HTTP GET
$_POST	Bet kokios reikšmės, pateiktos naudojant metodąč HTTP POST
$_COOKIE	Bet kokios reikšmės HTTP slapuko antraštėse
$_SERVER	Bet kokių vardų ir reikšmių poros, sukurtos žiniatinklio serverio
$_ENV	Bet kokie kintamieji iš žiniatinklio serverio aplinkos
$_REQUEST	Bet kokios reikšmės iš HTTP užklausos
$_SESSION	Bet kokie seanso kintamieji

Nauji kintamieji, kurie veikia kaip senieji $HTTP_*_VARS kintamieji ($HTTP_GET_VARS, $HTTP_POST_BVARS ir pan.), automatiškai prieinami bet kurioje srityje. Šiais pokyčias siekiama padąsinti programuotojus nepasikliauti vien automatiniu išorinių kintamųjų registravimu visuotinėje srityje, naudojant konfigūravimo parinktį register_globals, dėl kurios kyla saugumo problemų.

PHP privalumai. PHP pateikia daug priemonių turinčią platformą, skirtą žiniatinklio taikomosioms programoms. PHP siūlo platų programavimo sąsajų pasirinkimą ir galimybę susisiekti su daugybe atvirojo kodo ir komercinių duombazių, kitaip tariant, ja labai lengva operuoti, nes specialiai buvo sukurta dirbti su dideliu technologijų asortimentu. PHP yra labai mobili ir tinka įvairiuosiems žiniatinklio serveriams ir operacijų sistemoms, tarp jų Linux ir Windows. PHP netrūksta pažangių priemonių, tačiau išsaugota ir galimybė nesunkiai jos mokytis ir naudotis. Tai vienas iš svarbiausių veiksnių, renkantis konkrečią programavimo kalbą taikomajai programai kurti.

Norint, kad PHP veiktų kaip įmonėms skirtas sprendimas, pirmiausia reikėtų ją paversti tikru taikomųjų programų serveriu. PHP veiktų kaip nuolatinio serverio procesas su taikomųjų programų serveriu, turinčiu keletą gijų. Veikdama kaip savarankiškas taikomųjų programų serveris, PHP kur kas veiksmingiau susidorotų su duombazių prisijungimais. Tai leistų nuolat laikyti atvirą tam tikrą skaičių nuolatinių duombazių prisijungimų, kur užklausas prižiūrėtų ir apdorotų tariamasis PHP taikomųjų programų serveris. Toks dizainas žymiai pagerintų našumą. Nuolatinis serverio procesas praverstų naudojant PHP kartu su nutolusių procedūrų iškvietimo protokolu, pavyzdžiui, SOAP ar net žemesnio lygmens.

Pavyzdžiui, gavęs nutolusios buhalterinės programos iškvietimą, PHP taikomųjų programų serveris apdoroja rezultatus ir siunčia juos atgal buhalterinei programai. Tą pačią akimirką, kai PHP

serveris apdoroja vidinės buhalterinės programos užklausą, XHTML dokumentu ji gali atsakyti į interneto naršyklės užklausą. Tokios situacijos pasitaiko gan dažnai, o privalumas akivaizdus.
Iš viso šito matome, kad PHP yra galinga ir efektyvi priemonė kurti tinklalapiams.

1.16. Programinių produktų apžvalgos išvados

Apžvelgiau keletą programinių produktų, tinkamų vienoms ar kitoms funkcijoms atlikti. Dauguma jų specializuoti ir tinkami profesionaliam darbui. Tačiau man reikalingas produktas, kuris tiktų vidutinius darbo su kompiuteriu įgūdžius turinčiam vartotojui, ir pasižymėtų intuityvia ir „draugiška“ vartotojo sąsaja. Tikslas: „Laiko ir vartotojo darbo sąnaudomis kaupiami ir tvarkomi duomenys“.

Visų pirma numatoma programa turėtų būti pilnavertis duomenų bazių kūrimo ir palaikymo įrankis. Pagal tai galima pasirinkti MySQL arba Firebird programinius produktus. Iš jų pasirenkam MySQL nes:

Kaina. MySQL daugeliu atvejų yra nemokama, o jos parama nebrangi.

Parama. „MySQL AB“ siūlo nebrangios paramos sutartis, be to, yra didelė ir veikli MySQL bendruomenė.

Sparta. MySQL lenkia daugelį savo konkurentų.

Funkcionalumas. MySQL siūlo daugumą savybių, kurių reikalauja rimti programuotojai, tokių, kaip visiškas suderinamumas su ACID, suderinamumas su didžiąja ANSI SQL dalimi, atsarginių kopijų darymas prisijungus, kopijavimas, suderinamumas su saugiu lizdų lygmeniui (Secure Sockets Layer, sutr. SSL) ir integracija į beveik visas programavimo aplinkas. Be to, ji kuriama ir atnaujinama sparčiau nei daugelis jos konkurentų, taigi beveik visos „standartinės“ savybės, kurių MySQL kol kas neturi, yra kuriamos.

Mobilumas. MySQL veikia didžiojoje daugumoje operacinių sistemų ir daugeliu atvejų duomenys be jokių sunkumų gali būti persiųsti iš vienos sistemos į kitą.

Naudojimo paprastumas. MySQL lengva naudotis ir administruoti. Daugelis senesnių duomenų bazių kenčia dėl palikimo problemų, be reikalo paversdamos administravimą sudėtingu darbu. MySQL įrankiai galingi ir lankstūs, neprarandantys naudos.

Antra, reikia programos palaikančios žiniatinklių darbą. Šiam darbui pasirinkau PHP, nes PHP pateikia daug priemonių turinčią platformą, skirtą žiniatinklio taikomosioms programoms. PHP siūlo platų programavimo sąsajų pasirinkimą ir galimybę susisiekti su daugybe atvirojo kodo ir komercinių duombazių, kitaip tariant, ja labai lengva operuoti, nes specialiai buvo sukurta dirbti su dideliu technologijų asortimentu. PHP yra labai mobili ir tinka įvairiuosiems žiniatinklio serveriams

ir operacijų sistemoms, tarp jų Linux ir Windows. PHP netrūksta pažangių priemonių, tačiau išsaugota ir galimybė nesunkiai jos mokytis ir naudotis. PHP yra nemokama.

1.17. Problemos sprendimo būdas

Pagal atliktą galimų sprendimų analizę, pasirenkamas sprendimas

1. Eksperimente bus naudojama į paslaugą orientuota architektūra.
2. Eksperimente bus naudojama objektiškai orientuotas programavimas.
3. Bus pateikta žiniatinklio paslaugos (WS) kaip taikomųjų programų serverio (AS) realizacija.
4. Sistema bus realizuojama atsižvelgiant į:

- ✓ apkrovimo balansavimą;
- ✓ tinkamą reakcijos į sistemos klaidą;
- ✓ duomenų saugumą;
- ✓ patikimumą;
- ✓ pritaikomumą siekiant užtikrinti greitą naujų vartotojo poreikių realizaciją.

Sistemoje bus realizuota dauguma AS technologijos privalumų:

- ✓ administravimo įrankis – grafinis administravimo įrankis nutolusių vartotojų ir serverių darbo sekimui;
- ✓ testavimo įrankis – skirtas verslo logikos testavimui;
- ✓ integruota vystymo aplinka;
- ✓ sujungimų su DB „baseino“ technologija ir sujungimų laikinu saugojimu atmintyje;
- ✓ būsenų valdymas;
- ✓ transakcijų valdymas ir paskirstytos taikomosios programos;
- ✓ verslo ir taikomosios programos logikos sąsaja;
- ✓ daugiagijiškumas ir lygiagretūs skaičiavimai siekiant maksimizuoti CPU išnaudojimą.

5. MIS bus bendros sistemos vienas komponentas, realizuojamas naudojant komponentinį programavimą.
6. Bus realizuoti papildomi komponentai, kuriuos bus galima panaudoti pakartotinai (komponentinis programavimas).
7. Mokyklos informacinė sistema – elektroninis dienynas (MIS) bus realizuojamas kaip AS verslo logikos dalis.
8. MIS bus bendros sistemos vienas komponentas, realizuojamas naudojant komponentinį programavimą.
9. Bus realizuoti papildomi komponentai, kuriuos bus galima panaudoti pakartotinai (komponentinis programavimas)

1.18. Dalyvių tyrimas

Akmenės gimnazijoje ir Ventos vidurinėje mokykloje atlikau apklausą. Iš apklausos paaiškėjo, ko iš šios sistemos tikisi būsimi vartotojai:

1. Mokyklos administracija:
    a) direktorius – stebėti moksleivių mokymosi rezultatus ir lankomumą.
    b) pavaduotojas (ugdymui) – žinoti, kada klasės auklėtojai numato klasės valandėles, matyti būrelių tvarkaraštį, klasių auklėtojų sąrašą, mokytojų rengiamus projektus, būsimų konkursų ir olimpiadų datas.
    c) pavaduotojas (vykdymui) – matyti mokinių sąrašus, sudarytus pagal dalykus, kuriuos jie mokosi, pagal mokinių mokymosi rezultatus sudarytas suvestines: 10 balų, nuo 9 iki 10 balų, nuo 7 iki 10 balų, nuo 4 iki 6 balų, nepažangius. Stebėti bendrą klasės lankomumą ir kiekvieno mokinio atskirai, matyti pateisintas ir nepateisintas pamokas. Gauti 4, 10 ir 12 klasių egzaminų (mokykliniai: bendra lentelė ir 7–10 balų, valstybiniai: bendra lentelė ir 51 – 100 balų, neišlaikiusiųjų) pažymių suvestines. Matyti visų mokytojų sąrašą, seminarų, kuriuose jie dalyvavo, sąrašą, stebėti kvalifikacijos kėlimą, patalpinti informaciją apie numatomus posėdžius.
    d) sekretorė – matyti bendrą mokinių skaičių, atvykusius ir išvykusius mokinius, mokinių asmeninius duomenis.

2. Mokytojai:
    a) klasės auklėtojai – stebėti lankomumą, visų mokslo metų, trimestrų ir metinį kiekvieno mokinio visų dalykų pažymius, patalpinti informaciją apie numatomas klasės valandėles, tėvų susirinkimus, ekskursijas, matyti pamokų tvarkaraštį.
    b) dalyko mokytojai – nesudėtingai įvesti ir taisyti duomenis apie moksleivių mokymąsi ir lankomumą, žinoti, kad bus atkreiptas mokyklos administracijos, klasės auklėtojo ir tėvų dėmesys į nesilankančius ir nepažangius moksleivius.

3. Socialinis darbuotojas – stebėti mokinių lankomumą ir pažangumą, mokyklos ugdymo planus, būrelių tvarkaraštį, spec. poreikių moksleivius.

4. Bibliotekos darbuotojai – matyti tvarkaraštį, mokytojų numatomas pamokas skaitykloje, sudarinėti naujų knygų sąrašus (kad galėtų matyti mokytojai).

5. Tėvai – stebėti savo vaikų mokymosi rezultatus, lankomumą, gauti informaciją apie klasės auklėtojo numatytus renginius: klasės valandėles, susirinkimus ir pan. Turėti galimybę pateikti pasiūlymus ar prašymus.

1.19. Analogiškų programų apžvalga

Kurdamas savo dienyno programą, rėmiausi kitų autorių jau sukurtais dienynais. Kaip pavyzdžius būtų galima pateikti: Šiaulių miesto Lieporių mokyklos elektroninį dienyną, Kauno miesto „Jėzuitų“ gimnazijos elektroninį dienyną. Šie dienynai yra gana paprasti ir su trūkumais, kuriuos stengiausi pašalinti kurdamas savo elektoninį dienyną. Ankstesniuose dienynuose išryškėja tokie trūkumai kaip:

- ✓ tvarkaraštyje visos dienos yra aktyvios ir neaišku kokią funkciją jos atlieka;
- ✓ visų semestrų pažymius galima koreguoti, o taip neturėtų būti, nes taip būtų galima nuolat keisti pažymius;
- ✓ norint atspausdinti ataskaitą, tenka naudotis Microsoft paketo Excel programa, nes atsakaita nėra sukuriama pačiame dienyno puslapyje;
- ✓ tėvams pateikiamas viso semestro pažymių sąrašas, norint juos peržiūrėti nėra patogu pasirinkti būtent pageidaujamą mėnesį;
- ✓ ne darbo dienų neišskyrimas sukelia taip pat tam tikrų nepatogumų;
- ✓ nėra informacijos apie specialiųjų poreikių moksleivius.

Kurdamas savo dienyną, stengiausi pašalinti visus tam tikrus trūkumus. Dienyne sukuriamos ataskaitos pačio dienyno puslapyje bei su Microsoft paketo Excel programa, tėvams pranešama apie numatomus renginius, vyksiančius tėvų susirinkimus, mokyklos pavaduotoja gali stebėti, kokie mokiniai nelanko mokyklos, pavaduotoja gali siųsti pranešimus mokytojams apie vyksiančius posėdžius ir kitas funkcijas. Visa tai skiria mano dienyną nuo kitų, nes juose tokios funkcijos neegzistuoja arba tik dalinai realizuotos.

2. Sistemos projektavimas

2.1. Sistemos aprašymas

Mokyklos informacinė sistema – elektroninis dienynas (toliau MIS) yra įrankis, automatizuojantis organizacijos duomenų valdymą. Jos paskirtis – kaupti, redaguoti ir analizuoti įvairius duomenis, kurie naudojami mokykloje mokymosi metu, pateikti įvairias ataskaitas, ieškoti klaidų, perspėti apie tai MIS vartotoją. Vartotojo pageidavimu ši programinė įranga turi būti adaptuota Akmenės gimnazijai.

Kompiuterizuojamų funkcijų sąrašas:

- kaupti ir redaguoti asmeninius darbuotojų ir moksleivių duomenis;
- kaupti duomenis moksleivių mokymosi eigoje apie jų pasiekimus;
- pateikti ataskaitas mokytojams, administracijai, auklėtojams ir mokinių tėvams.

2.2. Sistemos vartotojai

Programine įranga gali naudotis kiekvienas, mokantis dirbti kompiuteriu (mokantis naudotis standartine Microsoft Windows operacinės sistemos grafine vartotojo sąsaja). Tačiau darbas bus efektyvus, jei vartotojas turės bent minimalias žinias apie taikomąją sritį.

Vartotojams būtinos lietuvių kalbos žinios. Programa yra skirta lietuviškai kalbančių vartotojų grupei, todėl visi paaiškinimai pateikiami būtent šia kalba.

Išskiriamos keturios vartotojų grupės: mokinys/tėvai, mokytojas, administracija, administratorius.

Mokinys/Tėvai

Vartotojo sprendžiami uždaviniai (galimos funkcijos):

- ✓ Užsiregistruoti.
- ✓ Peržiūrėti ir atspausdinti mokymosi rezultatus.

Šios vartotojų grupės poreikiai (nuomonė ir pageidavimai) turi žemiausią prioritetą.

Mokytojas

Vartotojo sprendžiami uždaviniai (galimos funkcijos):

- ✓ Užsiregistruoti.
- ✓ Peržiūrėti ir atspausdinti savo dalyko rezultatus.
- ✓ Įvesti naujus duomenis apie moksleivių mokymosi rezultatus.

Šios vartotojų grupės poreikiai (nuomonė ir pageidavimai) turi vidutinį prioritetą.

Administracija

- ✓ Užsiregistruoti.

- ✓ Peržiūrėti, atspausdinti esamus rezultatus bei kitokio pobūdžio ataskaitas.
- ✓ Įvesti tvarkaraščius. Panaudojama visa surinkta ir prieinama informacija apie modulius, mokytojus, mokytojus ir pan.

Šios vartotojų grupės poreikiai (nuomonė ir pageidavimai) turi aukščiausią prioritetą.

Administratorius

- ✓ Instaliuoti ir nustatyti elektroninio dienyno programinę įrangą.
- ✓ Nustatyti sistemos konfigūraciją pagal vartotojų atsiliepimus.

Šios vartotojų grupės poreikiai (nuomonė ir pageidavimai) turi aukščiausią prioritetą.

2.3. Apribojimai sprendimui

Sistema turėtų veikti Microsoft Windows NT/2000/XP/ME operacinėse sistemose.

Naudojamas mokyklos duomenų bazės serveris turi būti Apache, kur versija 1.3.23. ir aukštesnė. Sistema dirba ir su Interbase reliacine DB (MySQL adaptacija v. 4.1 ir aukštesne).

Projektuojant bus remiamasi objektiškai orientuota projektavimo ir paslaugą orientuotos architektūros metodologija. Architektūra aprašoma Visio diagramomis.

Programuojant bus naudojama PHP 4.3.9 programavimo kalba. Bus naudojamos žiniatinklio paslaugos ir taikomojo serverio kūrimo technologijos.

2.4. Sistemos funkciniai reikalavimai

1. Naudojantis sistema turi būti galima ištrinti, redaguoti, peržiūrėti ir atspausdinti mokinių mokymosi rezultatus.
2. Duomenų įvedimui gali būti įvedami apribojimai pagal nustatytą tvarką. Kiti apribojimai turi būti suteikiami pagal nutylėjimą. Jei apribojimų nesilaikoma, sistema automatiškai praneša apie jų pažeidimus.
3. Suteikiami apribojimai vertinimams:
   - ✓ pagal balų sistemą;
   - ✓ pagal dalykus (negali būti vertinimų iš neegzistuojančių dalykų);
   - ✓ pagal dalyko pobūdį (būna tik įskaitos);
   - ✓ pagal projektinius darbus.

Suteikiami apribojimai ataskaitoms:

- ✓ pagal klases;
- ✓ pagal dalykus;
- ✓ pagal lankomumą.

Suteikiami apribojimai administracijai:

- ✓ negalima keisti rezultatų;
- ✓ pranešimų rašymams (gali būti tik bendri);
- ✓ specialūs pageidavimai (tam tikros ataskaitos).

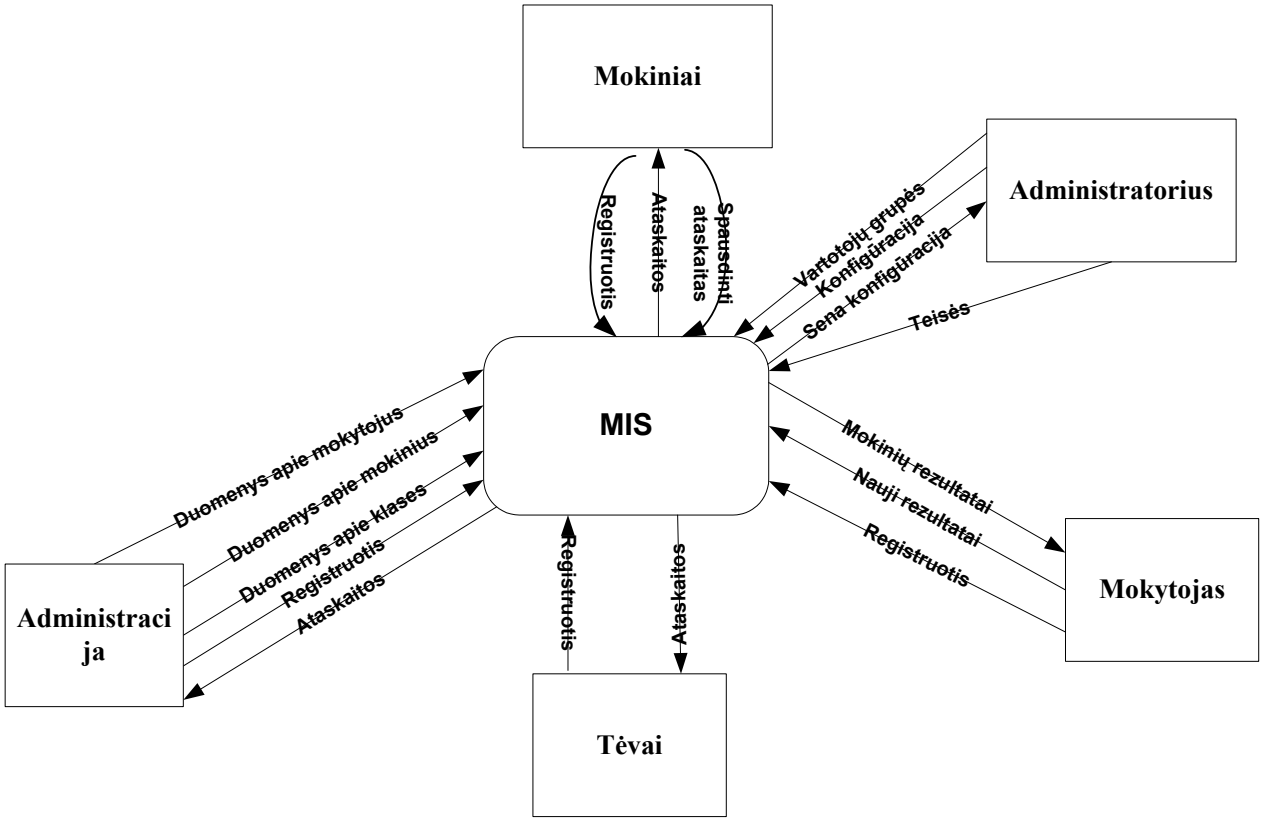
2.5. Sistemos nefunkciniai reikalavimai

1. Sąsaja turėtų būti patraukli vartotojui: spalvos neturėtų „skaudinti“ akių, turėtų būti suderintos tarpusavyje.
2. Vartotojo sąsaja privalo būti grafinė, naudojami tokie sąsajos elementai kaip:
   - ✓ langai;
   - ✓ dialogai;
   - ✓ meniu.
3. Lengvai skaitoma sąsaja. Kompiuterio ekrane visa informacija turi būti vaizduojama tvarkingai. Tekstinės informacijos išvedimui turi būti parinktas lengvai skaitomas šriftas.
4. Vartotojo sąsaja turi atspindėti taikomąją sritį: naudoti meniu ir mygtukus, pavadintus dalykinės srities sąvokomis.
5. Programa orientuota į Lietuvos rinką, t.y. sukurta programinė įranga turės lietuviškus tekstus ir pagalbos sistemas, tačiau turi būti numatyta galimybė kalbos keitimui.
6. Sistemos darbo metu neturi būti neatlaisvinamų resursų (memory, resource leak’ų).
7. Programa turi veikti stabiliai, nepadarant žalos aplinkai, vartotojo programinei bei techninei įrangai.
8. Sistemos turi būti suprojektuota taip, kad būtų galima ją lengvai papildyti naujo funkcionalumo komponentais (komponentinis programavimas).
9. Sistemos komponentai gali būti pakartotinai panaudojami (objektiškai orientuota).
10. Sistemos komponentai turi būti tarpusavyje mažai susieti (“loose coupling“)
11. Sistema gali naudotis daug vartotojų vienu metu
12. Sistema turi dirbti su skirtingų platformų duomenimis (duomenų integralumas).
13. Sistema turi veikti Windows NT/ 2000/XP/ME platformose.
14. Sistemos verslo logika turi būti atskirta nuo taikomosios logikos.
15. Sistema turi būti apsaugota nuo nesankcionuotų priėjimo prie jos resursų, t.y.: prisijungimas prie duomenų bazės turi būti apsaugotas slaptažodžiu, norint atlikti svarbesnį redagavimą ar valdymą reikia identifikuoti vartotoją.
16. Sistemos kontekste neturėtų būti žargonų, barbarizmų ar necenzūrinių žodžių.
17. Sistemai kurti naudoti tik licenzijuotą programinę įrangą.
18. Sistema turi nepažeisti Lietuvos Respublikoje galiojančių įstatymų bei teisės normų.
19. Visos teisės į produktą priklauso užsakovui.

3. Sistemos architektūra

3.1. Veiklos kontekstas

Sistemos kontekstas yra bet kokios švietimo įstaigos duomenų valdymo procesas, tačiau orientuojant sistemą į vartotoją, ji adaptuota Akmenės gimnazijai. Suprastintas veiklos kontekstas matyti 12 pav.



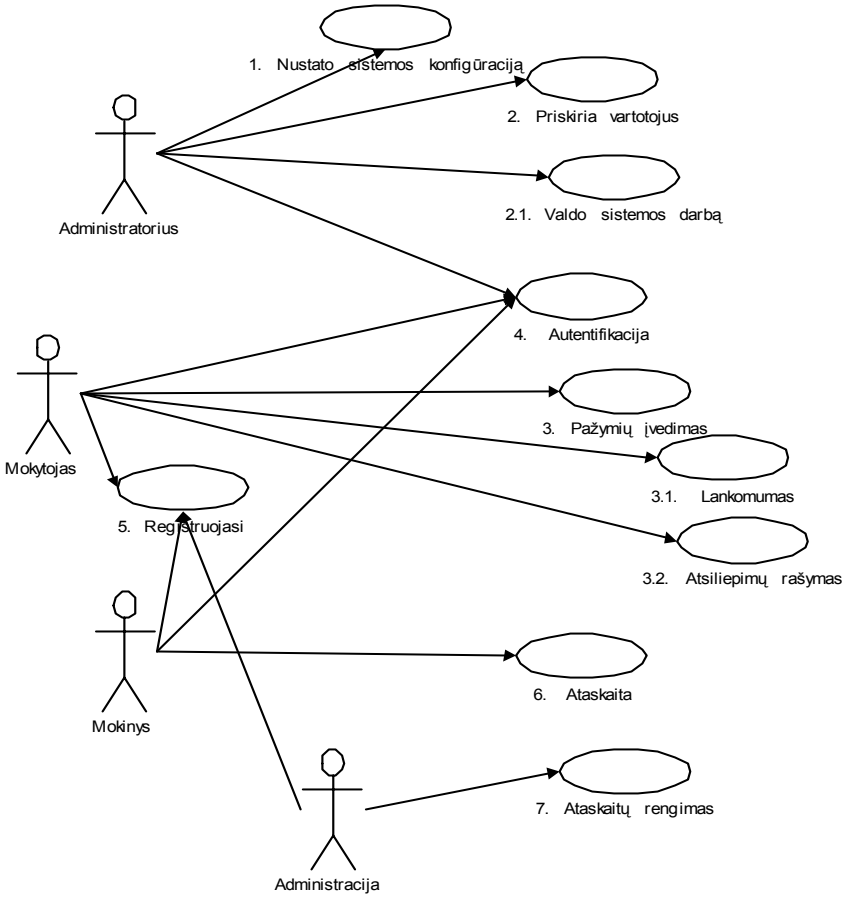
12 pav. MIS veiklos kontekstas

Lentelė 2. MIS veiklos padalinimas

Eil.nr.	Įvykio pavadinimas	Įeinantys/išeinantys informacijos srautai
1.	Spausdinti ataskaitas	Sumaketuotas ataskaitų dokumentas (out).
2.	Administracijos pateikiami duomenys	Informacija apie mokinius, mokytojus, modulius ir klases.(in)
3.	Įvesti rezultatus.	Esami rezultatai (in) Įvesti rezultatai.(out)
4.	Atnaujinti vidinės DB informaciją	Informacija apie mokinius, mokytojus, modulius ir kalses (in). Informacija apie mokinius, mokytojus, modulius ir klases (out).
6.	Sukurti naują konfigūraciją	Nauja konfigūracija(in) Naujas vartotojas(in) Nauja grupė(in) Nauja teisė(in) Sena konfigūracija(out)
7.	Registruotis	vartotojo duomenys(in)

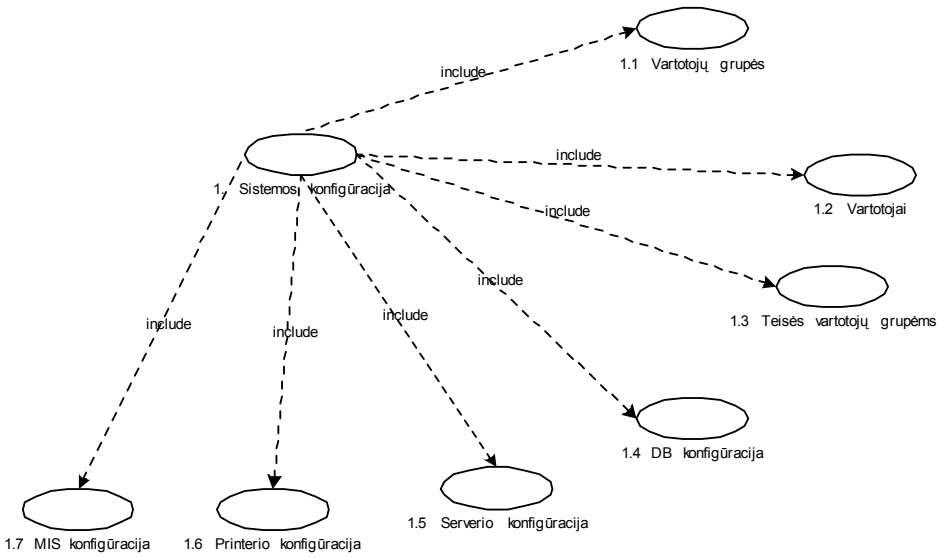
3.2. Sistemos apibendrinti panaudojimo atvejai

Sistemos vartotojai gali atlikti sistemoje tam tikrus veiksmus. Šiems veiksmams apibrėžti sudaromas vartotojų panaudojimo atvejų modelis (13 pav.).

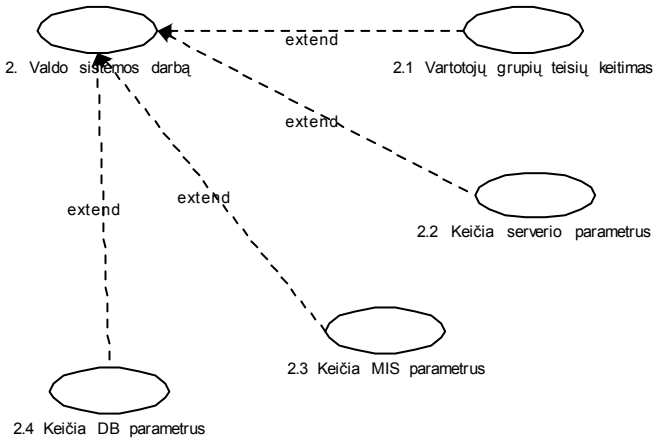


13 pav. Darbas su sistema

Sistemos funkcionalumo reikalavimai yra pateikiami atitinkamais panaudojimo atvejais. Toliau taikomas architektūros projektavimo „iš viršaus“ metodas – pagrindiniai panaudojimo atvejai yra padalinami į mažesnius panaudojant išplėtimo („extend“), įtraukimo („include“), paveldimumo („generalize“) ryšius (14, 15 pav.):



14 pav. Sistemos konfigūracijos nustatymas



15 pav. Sistemos darbo valdymas

3.3. Apibendrintų panaudojimo atvejų aprašymas

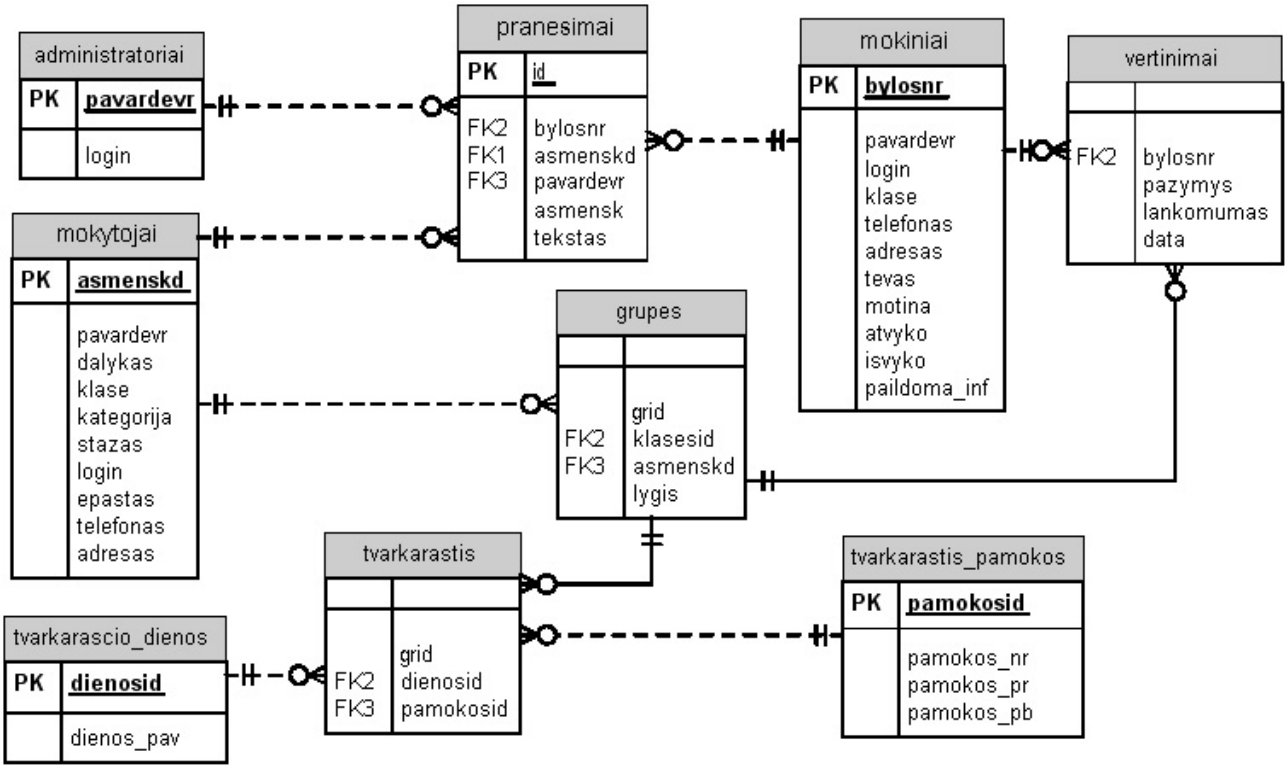
Apibendrintų panaudojimo atvejų aprašas pateikiamas lentelėje. Išskiriami aktoriai su kuriais panaudojimo atvejis sąveikauja, jo aprašas, ir tenkinimo kriterijus – kitaip tariant įvykdymo požymis:

Panaudojimo Atvejo Nr.	**Aktoriaus pavadinimas**	**Aprašas**	**Tenkinimo kriterijus**
1.	Administratorius	Sistemos darbo konfigūracijos nustatymas	Sistema paruošiama darbui.
2.	Administratorius	Sistemos darbo valdymas. Atliekami vis veiksmai, kurie reikalingi užtikrinti darbui sus sistema	Iškilus reikalui sistema gali būti nustatyta darbui su naujais parametrais, tenkinančius naujo sistemos vartotojo reikmes.
3.	Mokytojas	Atliekami veiksmai su rezultatais	Naudojant sistemą įvedami nauji pažymiai, lankomumo

			duomenys, rengiamos atasktaitos.
4.	Vartotojas, Mokytojas	Užsiregistruoja sistemoje	Vartotojas pateikia inf. apie save(užfiksuojama DB).
5.	Vartotojas, Mokytojas, Administratorius	Prisijungia prie sistemos	Prisijungusiam vartotojui sistema leidžia pasirinkti veiksmus, kuriuos bus galima atlikti pagal nurodytas teises vartotojų grupei, kuriai tas vartotojas priklauso
1.1.	Administratorius	Sukuria vartotojų grupes	Vartotojai priskiriami grupėms
1.2.	Administratorius	Sukuria vartotojus	Sukurti vartotojai
1.3.	Administratorius	Priskiria teises vartotojų grupėms	Egzistuojantys vartotojai gali dirbti sus sistema pagal teises nurodytas tai grupei, kuriai jie priklauso.
1.4.	Administratorius	Nustato DB konfigūraciją	Paruošta darbui vidinė MIS DB.
1.5.	Administratorius	Nustato Application serverio konfigūraciją	MIS gali siųsti užklausas į MIS DB
1.6.	Administratorius	Nustato spausdintuvo konfigūraciją	Spausdintuvas gali spausdinti ataskaitas.
1.7.	Administratorius	Nustato MIS konfigūraciją	Sistema pilnai paruošta darbui.
2.	Administratorius	Sistemos darbo valdymas	Sistemos nustatymai gali būti pakeisti pagal sistemos vartotojo pageidavimus.
2.1.	Administratorius	Pakeičia teises vartotojų grupėms	Vartotojai gauna naujas teises arba jiems jos atimamos.
2.2.	Administratorius	Keičia AS parametrus	Pakeistas priėjimas prie taikomųjų programų serverio (pagal sistemos vartotojo pageidavimus) Pvz. Suinstaliuotas kitame PC.
2.3.	Administratorius	Keičia sistemos parametrus	Nauja konfigūracija (pagal sistemos vartotojo pageidavimus). Pvz. Nustatomas naujas spausdintuvas.
2.4.	Administratorius	Keičia vidinės DB parametrus	Nauja DB konfigūracija (pagal sistemos vartotojo pageidavimus). Pvz. Kitame PC, kito tipo DB.

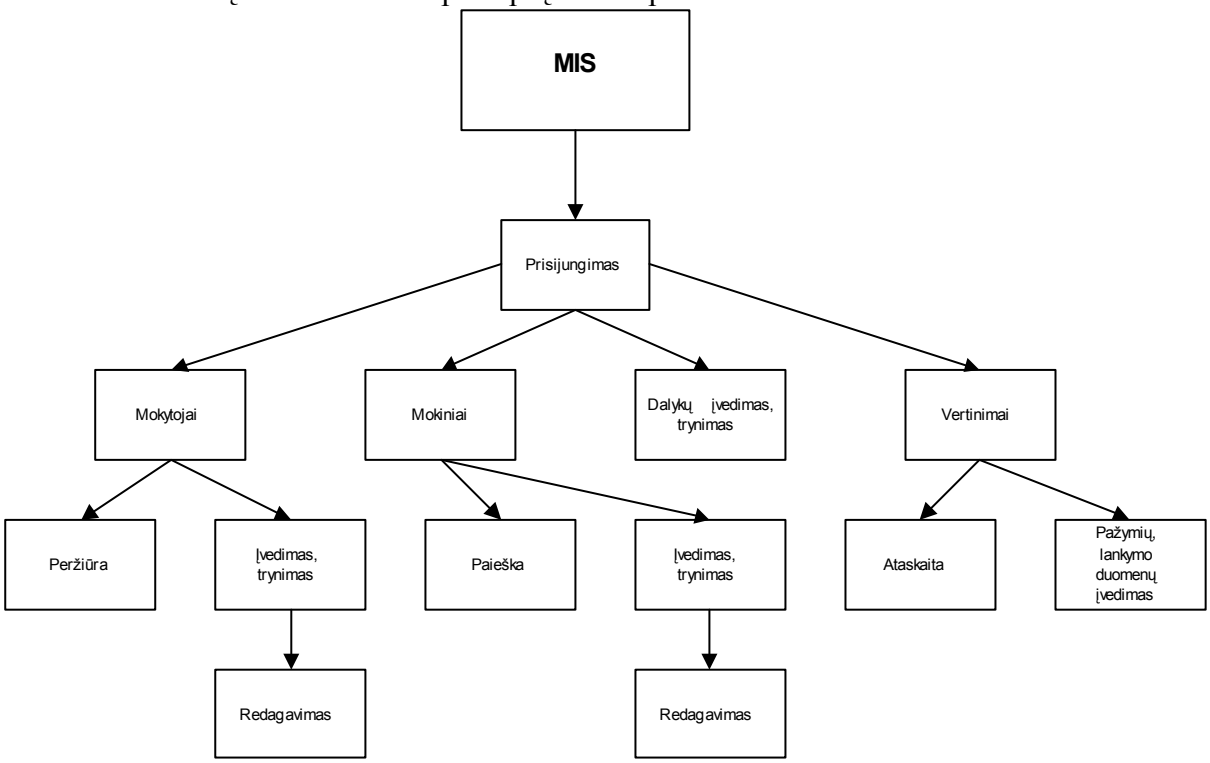
4. Sistemos realizacija

Duomenų bazės sistemos projektavimas. Suformuoti lentelių ryšiai, palengvinantys duomenų paiešką ir padedantys sudaryti formas bei ataskaitas naudojant kelių susietų lentelių duomenis, pateikti pav. 16.



16 pav. Duomenų bazės diagrama

Sudaromas duomenų bazės sistemos puslapių žemėlapis.



16 pav. Duomenų bazės puslapių žemėlapis

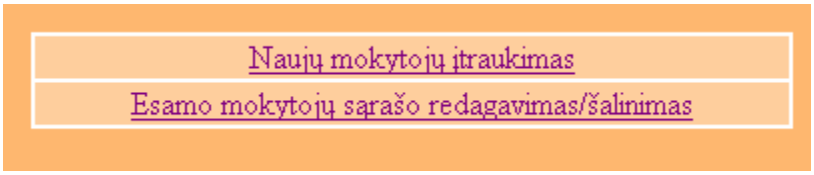
4.1. Uždaviniai*:*

1. Sudaryti duomenų bazės diagramą;
2. Sukurti duomenų bazės lenteles;
3. Sukurti grafinę vartotojo sąsają;
4. Parašyti programinę dalį duomenų įvedimui, redagavimui, trynimui;
5. Parašyti programinę dalį paieškai ir ataskaitoms.

Sistemos pagrindą sudaro 9 lentelės. 2 lentelės: mokiniai ir mokytojai, skirstos asmeninei informacijai. Čia yra įtraukti asmeniniai duomenys. Lentelėje klasės laikoma informacija apie mokinių suskirstymą į grupes. Ši informacija skirta tam, kad vartotojas galėtų lengviau orientuotis. Identifikaciniai numeriai vartotojams nerodomi. Lentelė grupės, skirta grupių sudarymui, kur nurodome koks mokytojas, kokį dalyką, kokiu lygiu kokiai klasei veda. Lentelėse, sukurtose pagal mokinių bylų numerius, laikoma informacija apie moksleivių mokymosi rezultatus bei lankomumą.

4.2. Mokytojų duomenų tvarkymas.

Prisijungus administratoriaus teisėmis su mokytojų duomenis galime atlikti šiuos veiksmus: įtraukti, redaguoti, ištrinti.



17 pav. Duomenų bazės mokytojų lentelė

Kad būtų patogiau naujų mokytojų įtraukimas yra atskirtas nuo bendro mokytojų sąrašo.

18 pav. Duomenų bazės mokytojų lentelė

Kad galėtume dirbti su šia lentele reikalingas PHP kodas:

```
if(isset($_POST['ivestas'])) {
        $asmenskd = $_POST['asmenskd'];
        $pavardevr = $_POST['pavardevr'];
        $dalykas = $_POST['dalykas'];
        $klase = $_POST['klase'];
        $kategorija = $_POST['kategorija'];
        $stazas = $_POST['stazas'];
        $login = $_POST['login'];
        $epastas = $_POST['epastas'];
        $telefonas = $_POST['telefonas'];
        $adresas = $_POST['adresas'];
        mysql_query("Insert INTO mokytojai (asmenskd,pavardevr,dalykas,klase,kategorija,stazas,
login,epastas,telefonas, adresas) VALUES ('$asmenskd','$pavardevr', '$dalykas',' $klase',' $kategorija',
'$stazas','$login', '$epastas', '$telefonas','$adresas')") or die ("Įrašymas nepavyko")
;.....
```

Pasirinkus punktą „redagavimas/šalinimas“ atsidaro langas, kuriame galime redaguoti bei naikinti jau esančių mokytojų duomenis.

Nr.	Asmens kodas	Pavardė vardas	Dalykas	Aukl. klasė	Kategorija	Stažas	Slapt.	E-paštas	Telefonas	Adresas	Naikinti
1	35510152356	Kaminskas Romas	kūno kultūra	9b	mokytojas	5	k564	kamin@one.lt	841029063	Vytauto 1-16	Naikinti
2	37810060609	Liuokaitis Ramūnas	informatika		mokytojas	6	l123	r.liuokaitis@gmail.com	861029062	Gedimino 5	Naikinti
3	37810050508	Matulaitienė Jūratė	informatika	12b	metodininkas	15	m123	jutar@mokykla.lt	861046562	Jurbarko 5-11	Naikinti
4	4785689124	Pocevičius Jonas	fizika		ekspertas	20	p159	pojo@kaunas.lt	861015062	Tunelio 2-15	Naikinti
5	861029065	Pocevičiūtė Živilė	chemija	12a	vyr. mokytoja	10	p123	a@one.lt	861027840	Šiaulių 41-4	Naikinti

Įrašyti pakeitimus

19 pav. Duomenų bazės mokytojų redagavimo lentelė

Generuojama visų mokytojų lentelė. Papildomai įvedamas stulpelis, kurio paskirtis: trinti mokytojus iš duomenų bazės. Ši lentelė patogi tuo, kad galima pataisyti, bet kurio mokytojo duomenis. Pataisius keletą laukelių, nebūtinai to paties mokytojo, visi pakeitimai išsaugomi duomenų bazėje mygtuko „Įrašyti pakeitimus“ pagalba.

Prisijungimas prie duomenų bazės:

```
<?
//prisijungimas prie duomenu bazes
$db="darbas";
$link = mysql_connect("localhost","root","");
```

```
if (!$link)
 die("Negaliu prisijungti prie duomenu bazes");
$d = mysql_select_db($db , $link)or die("Negaliu atidaryti duomenu bazes");

//trynimo veiksmo aprasymas
$dd=$HTTP_GET_VARS["dd"];
$action=$HTTP_GET_VARS["action"];
if($action=='trinti'){
mysql_query("DELETE FROM mokytojai WHERE asmenskd='$dd'");
}
//update vykdymas
if(isset($_POST['siunciama']))  {
        $result = mysql_query("SELECT * FROM mokytojai");
        while ($get_info = mysql_fetch_object ($result, MYSQL_ASSOC)){
                $asmenskd = $get_info–>asmenskd;
                $pavardevr = $_POST[$asmenskd.'_pavardevr'];
                $dalykas = $_POST[$asmenskd.'_dalykas'];
                $klase = $_POST[$asmenskd.'_klase'];
                $kategorija = $_POST[$asmenskd.'_kategorija'];
                $stazas = $_POST[$asmenskd.'_stazas'];
                $login = $_POST[$asmenskd.'_login'];
                $epastas = $_POST[$asmenskd.'_epastas'];
                $telefonas = $_POST[$asmenskd.'_telefonas'];
                $adresas = $_POST[$asmenskd.'_adresas'];
                mysql_query("UPDATE mokytojai SET pavardevr='$pavardevr',dalykas='$dalykas',klase='$klase',kategorija='$kategorija', stazas='$stazas', login='$login', epastas='$epastas', telefonas='$telefonas',adresas='$adresas'  WHERE asmenskd='$asmenskd'") or die("klaida");
}
}

//duomenu vaizdavimas lentele
$result = mysql_query("SELECT * FROM mokytojai ORDER BY pavardevr") or die("Klaidingas rezultatas uzklausoje");
$i=0;
$bc="#E2E2E2";
while ($get_info = mysql_fetch_object ($result, MYSQL_ASSOC)){
if ($bc == "#E2E2E2") {
$bc = "#ffffff";
} else {
$bc = "#E2E2E2";
}
$i=$i+1;
```

```
$asmenskd=$get_info–>asmenskd;
?>

<tr bgcolor="<? echo $bc; ?>">
<td align=center><font size="2"><? echo $i; ?></td>
<td align=center><font size="2"><? echo $get_info–>asmenskd; ?></td>
<td align=left><font size="2"><input type="text" size="17" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_pavardevr'; ?>" value="<? echo $get_info–>pavardevr ?>"> </td>
<td align=left><font size="2"><input type="text" size="10" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_dalykas'; ?>" value="<? echo $get_info–>dalykas ?>"> </td>
<td align=left><font size="2"><input type="text" size="3" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_klase'; ?>" value="<? echo $get_info–>klase ?>"> </td>
<td align=center><font size="2"><input type="text" size="10" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_kategorija'; ?>" value="<? echo $get_info–>kategorija ?>"> </td>
<td align=center><font size="2"><input type="text" size="3" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_stazas'; ?>" value="<? echo $get_info–>stazas ?>"> </td>
<td align=center><font size="2"><input type="text" size="5" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_login'; ?>" value="<? echo $get_info–>login ?>"> </td>
<td align=center><font size="2"><input type="text" size="20" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_epastas'; ?>" value="<? echo $get_info–>epastas ?>"> </td>
<td align=center><font size="2"><input type="text" size="8" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_telefonas'; ?>" value="<? echo $get_info–>telefonas ?>"> </td>
<td align=center><font size="2"><input type="text" size="10" style="font–size: 8pt" name="<? echo $get_info–>asmenskd.'_adresas'; ?>" value="<? echo $get_info–>adresas ?>"> </td>
<td align=center><font size="2"><? echo "<a href=mokytojutvarkymas.php?action=trinti&dd=".$asmenskd.">Naikinti"; ?> </td>
</tr>
<?
}
?>
</table>
<input name="siunciama" type="hidden" value="set">
<input name="submit" type="submit" value="Įrašyti pakeitimus">
</form>
```

Trynimui ir redagavimui naudojamas metodas POST. Jį naudojant naršyklės laukelyje nerodoma informacija.

4.3. Mokinių duomenų tvarkymas

Prie punkto „Mokiniai“ yra du pasirinkimai: „Naujų mokinių įvedimas“ ir „Redagavimas/šalinimas“.

Nr.	Bylos Nr.	Pavardė vardas	Slaptažodis	Telefonas	Adresas	Tėvas	Motina	Klasė	Atvyko	Išvyko	Naikinti
1	p159	Dekaminavičius Arminas	k147		Gedimino 5-5	Jonas	Edita	9a	2000-09-01	0000-00-00	Naikinti
2	k456	Janulionis Aurimas	jau12	845925133	Žeimenos 5	Algis	Jurgita	9a	2000-09-01	0000-00-00	Naikinti
3	p456	Jurgaitis Algis	j54	846516874	Kauno 1	Aurimas	Indrė	9a	2003-09-10	0000-00-00	Naikinti
4	145	Petraitis Jurgis	g4611	498413216	Kauno 7-5	Jonas	Odeta	10a	2001-09-01	0000-00-00	Naikinti
5	k123	Pocaitis Arūnas	132	861029062	Kauno 7-5	Alvydas	Žaneta	12a	2000-09-01	0000-00-00	Naikinti

Įrašyti pakeitimus

20 pav. Mokinių duomenų tvarkymas

Mokinių dalies PHP kodas yra panašus į mokytojų. Tik turi tam tikrų skirtumų. Įtraukiant naują mokinį automatiškai sukuriama jo „pažymių knygelė“. Tai atskira lentelė, kurios pavadinimą sudaro mokinio asmens bylos kodas. Šioje lentelėje laikoma informacija apie mokinio įvertinimus bei lankomumą, suskirstytus pagal dalykus ir datas.

Pocaitis Arūnas	Rugsejis																												
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Etika						9												8												
Lietuvių kl.											10																			
Vokiečių kl.																				8										
Anglų kl.																														
Matematika																														
Chemija									8								7													
Fizika						9								7							10									
Kūno kultūra										6														7						
Dailė														8																
Menai						4														5										

21pav. Mokinio pažymių knygelė

Ši lentelė generuojama automatiškai, įvedus mokinio asmens bylos numerį bei slaptažodį. Lentelės generavimo PHP kodo fragmentas.

```
include("prisijungimas.php");
$_SESSION['id'] = 'k123';
if ($_SESSION['id']) {
$a = 30;
$id = $_SESSION['id'];
$r = mysql_query("SELECT * FROM mokiniai WHERE bylosnr='$id'");
$ra = mysql_fetch_array($r, MYSQL_ASSOC);
        echo '
        <table border="0" id="table1" cellspacing="1">
        <tr>
                <td height="25" width="172" bgcolor="#006262">
                <p align="center"><font face="Verdana" size="2" color="#FFFFFF"><b>'. $ra["pavardevr"].'
</b></font></td>
```

```
                <td height="25" bgcolor="#00A8A8" colspan="'.$a.'">
                <p align="center"><b><font face="Verdana" size="2" color="#FFFFFF">
                Rugsejis</font></b></td>
        </tr>
        <tr>
                <td width="172" height="21" bgcolor="#FFCE9D"> </td>';
for ($i=1; $i<=$a; $i++) {
if ($i<10) $i = '0'.$i;
        echo '
                <td width="8" height="20" bgcolor="#027D58">
                <p align="center"><b><font face="Verdana" size="2" color="#FFFFFF">'.$i.'</font></b></p></td>
        ';
}
        echo '

        <tr>
```

4.4. Grupių sudarymas

Punktas „Grupės“, skirtas nurodyti mokytojams, kurie dėstys konkrečias disciplinas, klasėms, kurios šių disciplinų mokysis.

22 pav. Dalykų skirstymo puslapis

Viršuje galima užrašyti grupės pavadinimą, pasirinkti mokytoją ir t.t. Mokytojų duomenys pateikti „Combo box“ pavidalu ir paimti iš duomenų bazės, kad vartotojas negalėtų nurodyti pvz. neegzistuojančių mokytojų. „Combo box“ sudarymui skirtas kodas:

```
<select name="mokytojas">
<option value="">Mokytojas</option>
<?
$query = mysql_query("SELECT pavarde FROM mokytojai") or die(mysql_error);
while ($row = mysql_fetch_array($query))
{ echo "<option value=\"".$row["pavardevr"]."\">".$row["pavardevr"]."</option>"; }
?>
</select>
```

Be to šioje dalyje vartotojams yra sukurtas palengvinimas. Tai yra pasirinkus mokytoją automatiškai parodomas dalykas, kurį šis mokytojas dėsto. Kad tai įgyvendinti PHP galimybių neužteko. Čia pasinaudojau JAVA SCRIPT kalba. Šios dalie programinis kodas:

```
td width="30%"><select size="1" name="asmenskd" id="asmuo" onchange="keisti (document.getElementById
('asmuo').options[selectedIndex].text)">
   <option selected value="">Pasirinkite mokytoją</option>
   <?
   $j = '';
   while ($get_info = mysql_fetch_object ($langelis, MYSQL_ASSOC)){
        echo '<option '.$po.' value="'.$get_info–>asmenskd.'">'.$get_info–>pavardevr.'</option>';
        $j = $j.'if (vp==\".$get_info–>pavardevr.'\') document.getElementById(\'dalykas\').value=\".$get_info–
>dalykas.'\';'
        ?>
        <?
        }
        echo '<script language="JavaScript">
        function keisti(vp) {
        if (vp==\'Pasirinkite mokytoją\') document.getElementById(\'dalykas\').value=\'\';
        '.$j.'
        }
        </script>
        ';
        ?>
```

4.5. Vartotojų identifikacija

Kad prie šios sistemos negalėtų prisijungti pašaliniai vartotojai yra sukurti prisijungimo ir identifikacijos sistema. Norint prisijungti, bet kuriai vartotojų grupei yra reikalaujama prisijungimo vardo bei slaptažodžio. Šie duomenys yra saugomi duomenų bazėje. Kad pašalinis žmogus žinodamas bylų išdėstymo struktūra negalėtų apeiti prisijungimo sistemos, PHP bylos būtinai reikalauja patvirtinimo iš identifikacijai skirtų bylų.

```
$vardas = $_POST['pavardevr'];
$pass = $_POST['login'];
$vardas = addslashes($vardas);
$pass = addslashes($pass);
$result=mysql_query("select * from administratoriai
  where pavardevr='" . $vardas . "' and login='" . $pass. "'");

-----------------------------------------------------------------------------------------------
if ($num==1) {
        $r = mysql_fetch_array($result, MYSQL_ASSOC);
        $_SESSION['id'] = $r['asmenskd'];
        include("admin.htm");
}
```

5. Vartotojo dokumentacija

„Elektroninis dienynas“ – tai specializuota informacinė sistema skirta mokymo įstaigoms. Sistemos pagrindinė paskirtis – pakeisti dienyną ir pažymių knygeles, padarant jas elektronines.

Sistema veikia tinklalapio principu, t.y. duomenys suvedami bei peržiūrimi naudojantis interneto naršykle. Duomenys saugomi duomenų bazėje, kompiuteryje-serveryje. Tame pačiame serveryje yra patalpinta ir pati informacinė sistema.

Sistemoje yra trijų rūšių vartotojai: mokiniai (jų tėveliai), mokytojai ir administratoriai.

Kiekvienas programos naudotojas turi atskirą vartotojo vardą ir slaptažodį, kurių pagalba yra atpažįstamas programos. Taip pat jų pagalba yra užtikrinamas duomenų saugumas ir konfidencialumas, t.y. mokiniai negali peržiūrėti kitų mokinių pažymių, mokytojai negali koreguoti kitų mokytojų įvestos informacijos.

Sistema labiau orientuota į mokymo įstaigas, kurios turi nuolatinį interneto ryšį. Nes vos tik įvedami nauji pažymiai, juos jau gali išvysti mokinių tėvai.

Norint naudotis šia sistema, kompiuteriuose nereikia instaliuoti jokios programinės įrangos – užtenka tik paprasčiausios interneto naršyklės. Jūs galite naudotis sistema iš bet kurios pasaulio vietos, bet kokios operacinės sistemos kompiuteriu ar tiesiog interneto terminalu – tereikia atsiminti savo mokyklos tinklalapio adresą bei vartotojo vardą ir slaptažodį.

5.1. Sistemos galimybės

Visada laiku ir tik aktualūs duomenys

Be pažymių, taip pat galima sužinoti šią informaciją:

- kada mokinys pavėlavo į pamoką ar tiesiog joje nebuvo;
- ar yra mokiniui (ar visai klasei) skirtų pastabų ar vadovybės pranešimų;
- mokinio pamokų tvarkaraštį;
- vykusių pamokų aprašymus;
- užduotus namų darbus.

Kadangi tiek mokytojai suvedinėjantys pažymius, tiek mokinių tėvai juos peržiūrintys dirba su viena ir ta pačia duomenų baze, informacija yra prieinama vos tik ji yra suvedama.

Mokiniai nelankę mokyklos gali sužinoti kas buvo mokoma per praleistas pamokas, kas buvo užduota namų darbams.

Prisijungę prie sistemos mokinio tėvai sužino ar yra naujų pažymių ar pastabų. Dienyne tie pažymiai yra išskiriami iš visų likusiųjų.

Taip pat mokinio dienyne prognozuojamas trimestro ar semestro pažymys, yra informacija apie tam tikros dienos mokinio vidutinį pažymį.

Sistema palengvina darbą tiek mokytojams tiek mokyklos administracijai:

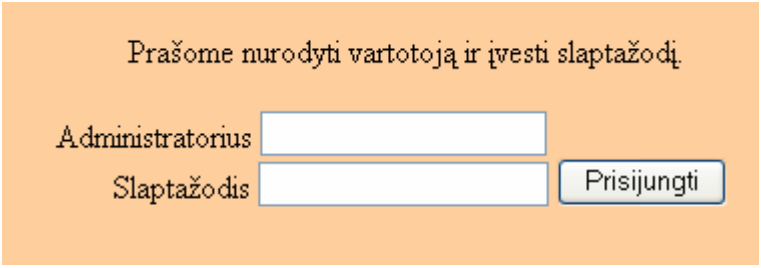
- mokytojams besiruošiant išvesti trimestrinius/semestrinius pažymius, sistema paskaičiuoja aritmetinį vidurkį, taip pat išveda informaciją apie visus gautus pažymius išskiriant juos pagal spalvas (pvz. raudoni – kontrolinių darbų pažymiai ir pan.);

- mokyklos administracija gali greitai suformuoti norimas ataskaitas. Ataskaitų mechanizmas yra labai lankstus, duomenis galima gauti už pasirinktą periodą, išskiriant norimą klasę, dalyką pažymio tipą ir t.t. (pvz. galima už pasirinktus mokslo suskaičiuoti matematikos kontrolinių darbų pažymių vidurkius 10A klasei). Galima sužinoti geriausiai/blogiausiai besimokinančius mokinius, ar informaciją apie praleistas pamokas.

- norint duomenis analizuoti kitais pjūviais ar tiesiog išsaugoti ateičiai, galima informaciją apie gautus pažymius „ištraukti“ iš sistemos į MS Excelio duomenų failą CSV formate.

Duomenų saugumo užtikrinimas

Prisijungimui prie sistemos kiekvienas vartotojas turi įvesti vartotojo vardą ir slaptažodį:



22 pav. Vartotojo autorizacija

Pamiršus ar įtarus kad slaptažodį galėjo kas nors sužinoti, galima jį pakeisti (slaptažodį gali pakeisti tik sistemos administratoriai). Vedant vartotojo vardą, bei slaptažodį, reikia atkreipti dėmesį į didžiąsias/mažąsias raides ir jų nesumaišyti.

Mokytojai gali matyti visų mokinių pažymius, bet koreguoti tik savo įvestus, tuo tarpu mokiniai negali matyti kitų mokinių pažymių.

Prie duomenų bazės negalima prisijungti iš kitų kompiuterių naudojantis kitomis programomis.

Visi svarbesnių duomenų pakeitimai yra fiksuojamai atskiroje įvykių lentelėje, išsaugant įvyko datą, laiką bei vartotoją kuris padarė pakeitimą.

Baigus darbą su sistema, reikia tiesiog paspausti nuorodą „Išeiti”, bei nepamiršti uždaryti naršyklės lango.

5.2. Sistemos galimybės mokiniams bei jų tėveliams

Prisijungęs prie sistemos mokinys yra informuojamas apie gautus naujus pažymius bei pastabas.

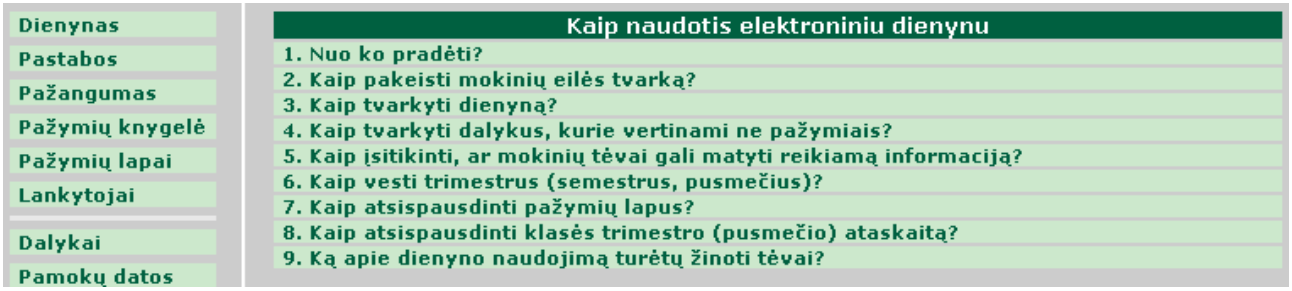
Jis gali pasižiūrėti tik savo pažymius, pastabas bei pamokų tvarkaraštį. Keisti jam nieko negalima.

Pocaitis Arūnas	Rugsejis																												
	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
Etika						9												8												
Lietuvių kl.											10																		
Vokiečių kl.																				8									
Anglų kl.																													
Matematika																													
Chemija									8								7												
Fizika						9								7							10								
Kūno kultūra										6														7					
Dailė														8															
Menai						4														5									

23 pav. Mokinio pažymių knygelė

5.3. *Sistemos nauda mokytojams*

Tokį vaizdą išvysta mokytojas prisijungęs prie sistemos:



24 pav. Mokytojo meniu

Vedant šios (ar praeitos) savaitės pamokų rezultatus mokytojui nebūtina nurodyti klasę, datą, laiką ar dalyką.

	Gruodis											
Nr	02	03	07	09	10	14	16	17	21	23	24	
1.		10	n,n	n	n,n	n,n	10	n,n				Ugnė Andriuškaitė
2.	n	n										Martynas Baliutis
3.		10			9	n,n	n	n,n				Aušra Banevičiūtė
4.					8	10						Laimonas Čekanauskas
6.		10			4	n,n	n	n,n		n		Paulius Damašius
7.					7							Simas Dunauskas
8.		10			9		10					Audronė Eismontaitė
9.					7					n		Vilija Gajauskaitė
10.					2							Žanas Gudlevičius
11.					7			n,n				Edgaras Joneikis
12.					6		n	n,n	n,n	n		Robertas Kazarjan
13.					7							Paulius Morkūnas
14.		10			7		10					Monika Olbakaitė
15.					n,n	8						Gabrielė Paskaitė
16.					5							Kristina Pilelytė
17.					3							Jūratė Preimantaitė
18.	n				n,n	4						Tadas Radavičius
20.					2		n	n,n		n		Linas Rinkevičius
21.					10	10						Kristina Rintautaitė
22.					9					n		Milda Šiukštaitė
23.	n				3							Domas Šulcas
24.					6		10					Jovita Tamkutė
25.					2			n,n	n,n			Aidas Tavoras
26.					4			n,n	n,n			Violeta Virbauskaitė
27.					6		10			n		Domininkas Virbickas

Taip atrodo pažymių įvedimo langas apjungtoje pamokoje.

Kaip ir visur, čia galima parašyti pastabą konkrečiam mokiniui, ar visai klasei. Įrašyti bendrą klasės ir namų darbų informaciją.

25 pav. Dalyko pažymiai

Mokytojas taip pat gali peržiūrėti savo pamokų tvarkaraščius pagal skirtingas klases.

Dalyko (kurso) pamokų datos: *anglu k (II gr.)*

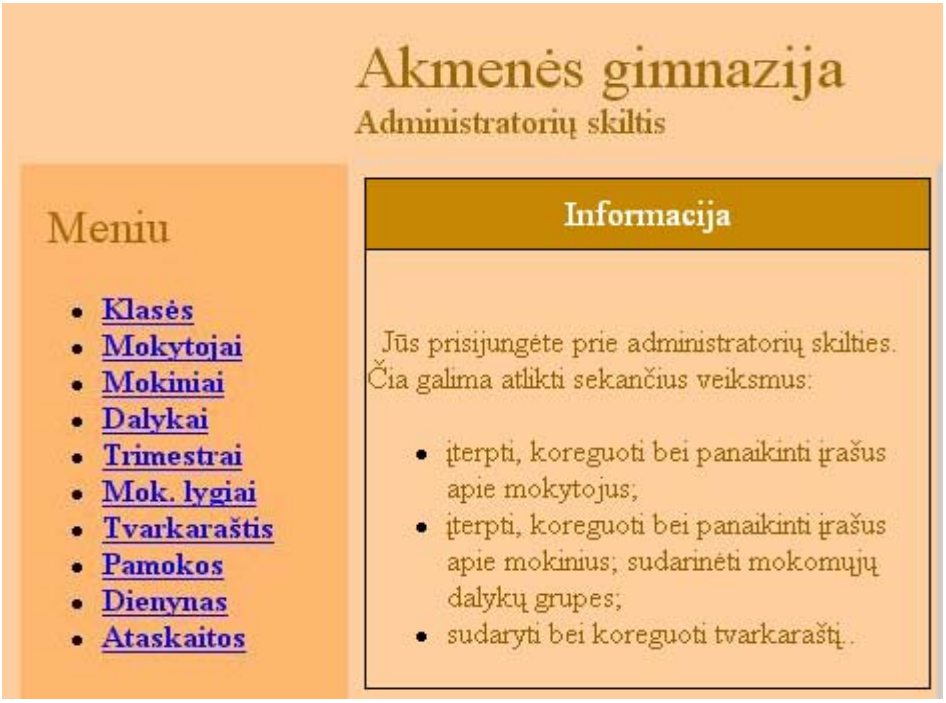
	08	Rugsėjis				Spalis				Lapkritis				Gruodis			Sausis				Vasaris				
Pr		6	13	20	27	4	11	18	25	8	15	22	29	6	13	20	10	17	24	31	7	14	21	28	7
An		7	14	21	28	5	12	19	26	9	16	23	30	7	14	21	11	18	25	1	8	15	22	1	8
Tr	1	8	15	22	29	6	13	20	27	10	17	24	1	8	15	22	12	19	26	2	9		23	2	9
Kt	2	9	16	23	30	7	14	21	28	11	18	25	2	9	16	23	13	20	27	3	10	17	24	3	10
Pt	3	10	17	24	1	8	15	22	29	12	19	26	3	10	17	24	14	21	28	4	11	18	25	4	
Š																									
S																									

26 pav. Tvarkaraštis

5.4. *Sistemos administravimas*

Mokykloje paprastai yra paskiriamas vienas mokytojas, kuris yra atsakingas už sistemos administravimą - sistemos administratorius.

Administratoriui tenka suvesti ir vėliau administruoti visą sistemos veikimui reikalingą informaciją, t.y. mokinių, mokytojų, klasių, trimestrų, dėstomų dalykų, ir galimų pažymių sąrašus. Taip pat administratorius gali keisti vartotojų slaptažodžius. Didesnėje mokykloje administratoriaus darbą gali daryti keli asmenys. Sistemos administracijos teisėmis taip pat gali būti tam reikalui paskirti mokytojai. Jie lygiai taip pat gali turėti auklėjamas klases, vesti pamokas, bei pildyti dienyną. Be viso to jie gali konfigūruoti sistemos darbą, tvarkyti klasifikatorius, registruoti vartotojus.



27 pav. Administratoriaus meniu

Šis meniu matomas prisijungus administratoriaus vardu.

Pagrindiniai administravimo meniu punktai:

- galima nustatyti klasės pamokų laiką, bei išsaugoti klasės rezultatus;
- suvedama visa informacija apie mokytojus – vardas, pavardė, telefonas, e-paštas bei adresas, taip pat nurodyta kokį dalyką mokytojas moko ir kokia yra jo auklėjamoji klasė;
- suvedama informacija apie mokinius, be pagrindinių duomenų tokių kaip vardas, pavardė ir klasė, galima nurodyti e-paštą, adresą bei telefoną; taip pat šioje dalyje galima pakeisti mokinių slaptažodžius;
- suvedama ne tik dėstomų dalykų pavadinimas, bet ir jų eiliškumas kuriuo remiantis yra rodomi dalykai dienyne.

Mokytojų duomenų tvarkymas:

28 pav. Mokytojų duomenų tvarkymas

Mokinių grupių sudarymas pagal dalykus:

29 pav. Grupių sudarymas

Ataskaitų pavyzdžiai:

Apie pažymių vidurkius rezultato pavyzdys:

30 pav. Pažymių vidurkių ataskaita

Šiame ataskaita TOP mokiniai pavyzdyje matome 10 geriausiai besimokinančių mokyklos mokinių:

Ataskaita: TOP mokiniai
Kriterijai: Klasė: Visos
Trimestras: Visi
Dalykas: Visi
Pažymio tipas: Visi
Pažymys: Visi
Rodyti tik didesnius
Rodyti 10 didžiausių

Eil.Nr.	Mokinys	Vidurkis
1	Narimantas Giedrius	10.0000
2	Galinis Mykolas	9.0000
3	Patrauskas Jonas	9.0000
4	Janulynas Jonas	8.5000
5	Tamulis Petras	8.0000
6	Kingas Haris	7.0000
7	Testuotojas Testas	6.8889
8	Baltusis Albertas	6.3333
9	Markevičiūtė Marytė	6.1250
10	Kubilinskaitė Gintarė	6.0000

31 pav. Klasės geriausi mokiniai

Išvados

1. Remiantis darbe atlikta teorine žiniatinklio paslaugų ir duomenų bazių valdymo sistemų analize, savo darbui kurti pasirinkau MYSQL duomenų bazę ir PHP, HTML pagrindu veikiančią scenarijų rašymo kalbą, kuri turi daug priemonių talpinančią platformą. MySQL siūlo daugumą savybių, kurių reikalauja rimti programuotojai, tokių, kaip atsarginių kopijų darymas prisijungus, kopijavimas ir integracija į beveik visas programavimo aplinkas.

2. Darbe atlikta naudojamų elektroninių dienynų analizė padėjo išsiaiškinti pagrindinius jų trūkumus, kuriuos stengiausi kuo geriau išanalizuoti. Analizė leido išvengti tokių trūkumų ar net patobulinti kai kurias sritis kuriant savo elektroninį dienyną.

3. Darbe atlikta apklausa leido nustatyti papildomas funkcijas, kurių pageidauja būsimi vartotojai:

- pavaduotojai – žinoti, kada klasės auklėtojai numato klasės valandėles, matyti būrelių tvarkaraštį, klasių auklėtojų sąrašą, mokytojų rengiamus projektus, būsimų konkursų ir olimpiadų datas, mokinių sąrašus, sudarytus pagal dalykus, kuriuos jie mokosi, pagal mokinių mokymosi rezultatus sudarytas suvestines. Stebėti bendrą klasės lankomumą ir kiekvieno mokinio atskirai, matyti pateisintas ir nepateisintas pamokas.
- klasės auklėtojai – stebėti lankomumą, visų mokslo metų, semestrų ir metinį kiekvieno mokinio visų dalykų pažymius, patalpinti informaciją apie numatomas klasės valandėles, tėvų susirinkimus, ekskursijas, matyti pamokų tvarkaraštį.
- dalyko mokytojai – nesudėtingai įvesti ir taisyti duomenis apie moksleivių mokymąsi ir lankomumą.
- bibliotekos darbuotojai – matyti tvarkaraštį, mokytojų numatomas pamokas skaitykloje, sudarinėti naujų knygų sąrašus (kad galėtų matyti mokytojai).
- tėvai – stebėti savo vaikų mokymosi rezultatus, lankomumą, gauti informaciją apie klasės auklėtojo numatytus renginius: klasės valandėles, susirinkimus ir pan. Turėti galimybę pateikti pasiūlymus ar prašymus.

4. Darbe nustačiau sistemos funkcinius ir nefunkcinius reikalavimus, kuriais remiantis buvo sukurta sistema.

5. Sukūręs sistemos prototipą su pagrindinėmis funkcijomis ir išbandęs jį, pastebėjau keletą trūkumų: nepakanka duomenų apie mokytojus ir mokinius, vartotojai neturėtų matyti mokytojo asmens kodo, įvestų duomenų redagavimas labai nepatogus ir neekonomiškas, vienu metu galima įvesti pažymį tik vienam mokiniui.

6. Sukūriau sistemą remdamasis prototipu. Pašalinau rastus trūkumus išbandant prototipą. Įdiegiau keletą naujų funkcijų: ataskaitų kūrimas pagal pažangumą 10-ties balų skalėje, lankomumą, tėvai

gali rašyti atsiliepimus auklėtojui, direktoriaus pavaduotojams, mokytojai gali peržiūrėti naujų bibliotekos knygų sąrašą.

7. Informacinė sistema įdiegta Akmenės gimnazijoje, kuria naudojasi mokytojai, mokyklos administracija bei mokinių tėvai. Pastebėjau, kad tik dalis mokytojų noriai dirba, kiti nedrįsta, nes mano, kad tai nereikalinga ar bijo išbandyti. Tėvai informacine sistema deja negalėjo pasinaudoti, nes ji įdiegta tik lokaliame tinkle.

8. Planuoju šią sistemą tobulinti, atsiradus galimybei suteikti mokinių tėvams prieigą prie elektroninio dienyno.

Terminų ir santrumpų žodynas

1. WS

   Žiniatinklio Paslauga (Web Service).
2. WWW

   Pasaulinis Žiniatinklis (World Wide Web).
3. HTTP

   Duomenų perdavimo tinklu protokolas (hyper transfer transmission protocol).
4. XML

   Programavimo kalba, aprašanti dokumento struktūrą (Extensible Mark–up Language).
5. UDDI

   (Universal Description, Discovery and Integration).
6. SOAP

   (Simple Object Access Protocol).
7. WSDL

   (Web Service Definition Language).
8. OS

   Operacinė sistema (Operation System).
9. MSDOS

   „Microsoft“ diskinė operacinė sistema (MicroSoft Disk Operation System).
10. DB

    Duomenų bazė.
11. PC

    Personalinis kompiuteris (Personal Computer).
12. RAM

    Operatyvinė atmintis (Random Access Memory).
13. CPU

    Centrinis procesorius (Central Processing Unit).
14. MAPI

    Taikomųjų programų elektroninio pašto sąsajos programavimo priemonė.
15. AS

    Taikomųjų programų serveris(Application Server).
16. MIS

    Mokyklos informacinė sistema.
17. 3T

trijų pakopų architektūra (three–tiered).

18. DBVS

    Duomenų bazių valdymo sistema.

19. HTML

    Žiniatinkliu perduodamo hiperteksto programavimo kalba (Hyper Text Markup Language).

20. JSP

    Žiniatinklio programavimo technologija (Java Server Pages).

21. EJB

    (Enterprise Java Beans).

22. J2EE

    Programinės įrangos kūrimo Java kalba įrankis(Java 2 SDK– Software Development Kit Enterprise Edition).

23. JDBC

    Java taikomųjų programų sąsaja su duomenų bazėmis tinkle.

24. API

    Taikomųjų programų sąsajos su nurodyta sistema programavimo priemonė (Application Programming Interface). Pvz. Windows API.

25. SSL

    Duomenų perdavimo tinklu protokolas (Secure Socket Layer).

26. RPC

    Nutolusios procedūros iškvietimo technologija (Remote Procedure Call).

27. CORBA

    Nutolusio objekto metodų iškvietimo technologija panaudojant objekto užklausų paskirstytoją (Object Request Broker).

28. RMI

    Nutolusio objekto metodo iškvietimo technologija (Remote Method Invocation).

29. COM

    Komponentų kaip objektų kūrimo modelis (Component Object Model).

30. CBD

    Komponentinis programavimas (Component–Based Development).

31. IDL

    Sąsajos aprašymo kalba (Interface Definition Language).

32. SMTP

    (Simple Mail Transfer Protocol).

33. IT

Informacinės technologijos (Information Technologies).

34. TCP/IP

Duomenų perdavimo protokolas (Transmision Control Protocol/Internet Protocol).

35. DTD

Dokumento tipo apibrėžimas (Document Type Definition).

36. SMGL

Standartizuota apibendrinta žymėjimo kalba (Standard Generalized Markup Language).

37. UML

Programinės įrangos projektavimo kalba (Unified Modeling Language).

38. RTTI

Duomenų ir metodų tipų informacija, prieinama programos vykdymo metu(Run–Time Type Information).

39. UC

Panaudojimo atvejis (Use – Case).

Literatūros sąrašas

1. Allen J. PHP 4 vadovas. Kaunas: Smaltija, 2003.
2. Andziulienė B. Duomenų bazių kursas http://www.ik.ku.lt/lessons/konspekt/db/.
3. Apache SOAP V2.2 dokumentacija http://ws.apache.org/soap/faq/faq_chawke.html
4. Будилов В.А. PHP 5. Экспресс-курс. BHV-Спб, 2005.
5. Denisovas V., Maciulevičius S., Šakys V. ir kt. Duomenų bazės. Kaunas: Žara, 2001.
6. Gilfillan I. MySQL 4 vadovas. Kaunas: Smaltija, 2003
7. Gunzer H. Introduction to Web Services. Borland, 2002.
8. Informacinių technologijų apžvalga http://vaidila.vdu.lt/~project2/lindex.htm.
9. Firebird duomenų bazės dokumentacija internete http://www.firebirdsql.org/manual/qsg15.html.
10. Krauklis N. El. pašto siuntimas pasinaudojus mail() funkcija. http://www.php.lt/render/Articles;aid,43.
11. Mysql duomenų bazės dokumentacija internete http://dev.mysql.com/doc/mysql/en/create-procedure.html.
12. Mysql duomenų bazės „stored“ procedūrų aprašymas http://builder.com.com/5100-6388-5178706.html.
13. Ulevičius R. Vartotojai „Online“. http://www.php.lt/render/Articles;aid,75.
14. Хабибуллин И.Ш. Самоучитель Java 2. BHV-Спб, 2005.
15. Хоторн Р. Разработка баз данны Microsoft SQL Server 2000 на примерах. Москва: Вильямс. 2001.
16. Vidžiūnas A., Marčiulynienė R. Access XP. Taikomųjų duomenų bazių projektavimo pagrindai. Kaunas: Smaltija, 2003.
17. William’as N., Francisco C., Sanjiva W. Web Services: Why and how. IBM 2001.