KAUNO TECHNOLOGIJOS UNIVERSITETAS
PROGRAMŲ INŽINERIJOS KATEDRA

EGIDIJUS GEGECKAS

SILVERLIGHT TECHNOLOGIJOS TYRIMAS RENGINIŲ ORGANIZAVIMO SISTEMOJE

MAGISTRO DARBAS

KAUNAS, 2009

KAUNO TECHNOLOGIJOS UNIVERSITETAS
PROGRAMŲ INŽINERIJOS KATEDRA

SILVERLIGHT TECHNOLOGIJOS TYRIMAS RENGINIŲ ORGANIZAVIMO SISTEMOJE

MAGISTRO DARBAS

Recenzentas: .....
doc. Aleksas Riškus

2009 gegužės d.

Vadovas: .....
prof. Eduardas Bareiša

2009 gegužės d.

Atliko: .....
Egidijus Gegeckas, IFM-3/2 gr. stud.

2009 gegužės d.

KAUNAS, 2009

TURINYS

Summary

Events publishing and invitations to events organizing and management system was created after analysis of events management business model. The system was delivered and installed to its client. The main project goal was to create a system which is reachable over the internet and improves events management for its users.

System graphical user interface (GUI) had to be comfortable and look simple. One of the main requirements was to develop a system which is easy extendable. Initial requirements were gathered before designing system architecture, but a lot of requirements were added during development and maintenance stages. Some business logic has been changed after system deployment.

Modern Web 2.0 and Rich internet applications set a new level of expectations for enterprises on the Web. Developers face heightened requirements for "richer" user interfaces at the same time Web applications increase in size and complexity. It gets more and more complex to reach these new goals using old technologies. Silverlight is one of the new type technologies, which enables easy Rich internet applications development. In this work I will analyze how to extend system usability and improve its development stage using Silverlight technology.

Santrauka

Išanalizavus renginių organizavimo veiklos modelį, buvo sukurta Renginių paskelbimo ir pakvietimų į renginius organizavimo ir valdymo sistema. Sistema buvo realizuota ir įdiegta užsakovui. Projekto tikslas buvo sukurti internetu valdomą sistemą, kuri padeda pavieniams asmenims ar organizacijoms paskelbti renginius, pranešti žmonėms apie organizuojamus renginius, sukviesti žmones į paskelbtus privačius bei viešus renginius, o taip pat ir registruoti renginio dalyvius.

Sistemos grafinė vartotojo sąsaja iš pirmo žvilgsnio turėjo pasižymėti patogumu ir paprastumu. Vienas iš svarbiausių reikalavimų, kad sistema būtų lengvai praplečiama. Pradiniai reikalavimai buvo surinkti ir nustatyti prieš sistemos projektavimą, tačiau dauguma reikalavimų buvo papildyta sistemos kūrimo ir jos palaikymo metu. Sistemos priežiūros ir tobulinimo metu keitėsi dalis sistemos veiklos logikos, o taip pat buvo įdiegta ir naujų funkcionalumų.

Nuolatos augant vartotojų poreikiui, o taip pat tobulėjant ir technologijom, kuriamom sistemom keliami vis didesni reikalavimai. Šiandien internetinės sistemos vis labiau panašėja į taikomąsias kompiuterines programas. Tarp internetinių sistemų ryškėja Web 2.0 ir Raiškiųjų internetinių sistemų (RIA) kūrimo tendencijos, kurios pasižymi turtinga ir interaktyvia grafine vartotojo sąsaja ir išplėstu savo funkcionalumu. Įprastinų technologijų pagalba tampa vis sunkiau tai realizuoti. Silverlight – tai nauja ir viena iš sparčiausiai tobulėjančių naujo tipo technologijų, naudojamų Raiškiosioms internetinėms sistemoms kurti. Šiame darbe apžvelgiamas sistemos galimybių praplėtimas ir jos kūrimo etapų optimizavimas su Silverlight technologija.

Paveiksliukų sąrašas

Lentelių sąrašas

1 Analitinė dalis

1.1 Taikymo sritis

1.1.1 Projekto tikslas ir adresatas

Šis projektas skirtas renginio planavimo etapui supaprastinti. Veiksmai, kuriuos atlieka sistema, palengvina naudotojo darbą, išsprendžiant konkrečius renginio organizavimo etapo klausimus. Sistema yra orientuota į šias renginius organizuojančias žmonių grupes:

- Renginių organizavimo kompanijos.
- Verslo kompanijos.
- Fiziniai asmenys.
- Valstybinės institucijos.

Sistema yra aktuali visoms aukščiau išvardintoms žmonių grupėms, kuriuos organizuoja privačius arba viešus renginius. Sistemoje galima paskelbti tiek privačius, tiek viešus renginius, tačiau sistema yra labiau orientuota į asmenis, organizuojančius privačius renginius, kur yra svarbu pakviesti konkrečius asmenis į renginį.

Kuriant sistemą keliamas tikslas ją padaryti lengvai naudojamą įvairių sričių renginių organizavimui. Sistemos taikymas buvo kuriamas sprendžiant asmenines renginių organizatoriaus problemas, naudojantis jo ilgamete patirtimi organizavimo srityje.

1.1.2 Problemos sprendimas pasaulyje

Pirmiausiai reikėtų atkreipti dėmesį kokie yra organizatoriaus lūkesčiai, jam organizuojant tam tikrus renginius.

- Ar jiem paprasčiausiai reikia greito, nebrangaus ir lengvo kelio surinkti mokėjimams
- Ar jie planuoja organizuoti sudėtingus ir didelius renginius ar konferencijas.
- Ar jiem reikalingas kitoks apmokėjimų būdas.
- Ar jų svarbiausias poreikis yra svečių sąrašas, saugomas ir analizuojamas sistemos duomenų bazėje.

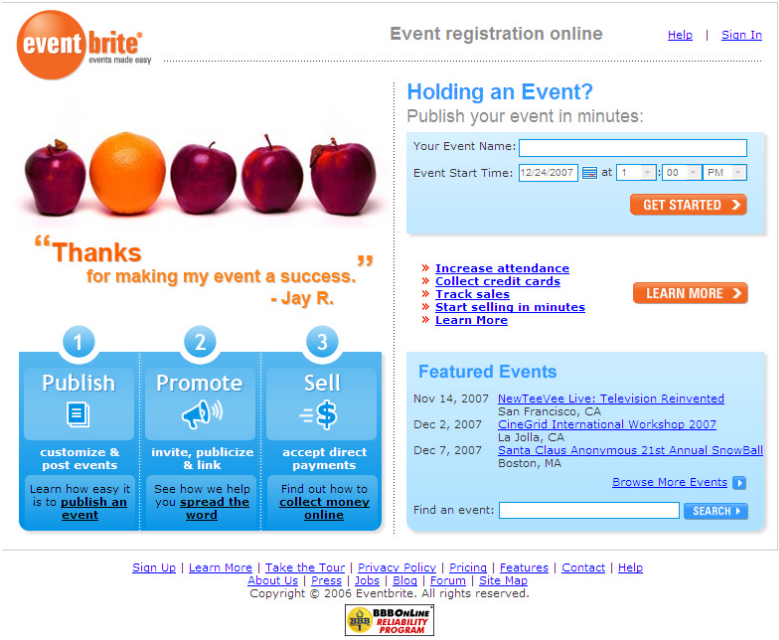
Pasaulyje yra keletas panašios paskirties projektų, tačiau dauguma jų skirti profesionaliems vartotojams. Šios sistemos turi begalę funkcijų, tačiau jomis gana sudėtinga naudotis. Tam kad pilnai išnaudot jų funkcionalumą reikia būti patyrusiu kompiuterio vartotoju. Be to norint išbandyti kai kurias sistemas reikia skambinti arba parašyti vadybininkui, kad jis tau suteiktų priėjimą prie demonstracinės versijos. Taigi norint paskelbti renginį, reikia laukti kelias dienas kol galėsit tai padaryti.

Mūsų nuomone geriausia šias problemas išsprendė toliau pateiktos sistemos. Žinoma kiekviena iš jų turi savo privalumų bei trūkumų, tačiau būtent šias sistemas mes laikome savo tiesioginiais konkurentais pasauliniu mastu.

1.1.2.1 Eventbrite

Adresas: www.eventbrite.com

Pradinis puslapis:



Pav. 1 Eventbrite pagrindinis langas

Privalumai:

Ši sistema leidžia labai greitai paskelbti renginį ir sukviesti į jį svečius. Pakankamai aiškus vartotojo interfeisas leis net ir nepatyrusiam vartotojui gana lengvai sukurti renginį. Sistema kaip manome sukurta panaudojus Python programavimo kalbą, dėl ko funkcinis sistemos veikimas yra greitas. Leidžiama tinkinti registracijos į renginį puslapį, šios funkcijos dėka renginio organizuotojui suteikiama vietos pasireikšti fantazijai, kad prijauktų kuo daugiau lankytojų į savo renginį. Taip pat organizuotojui leidžiama nurodyti Paypal sąskaitą į kurią tiesiogiai būtų pervesti pinigai už įsigytus bilietus į renginį.

Trūkumai:

Kadangi sistema yra pakankamai neseniai sukurta tai vis dar pasitaiko keletas klaidelių, taip pat nėra galimybės tinkinti elektroninio pašto laiškų, siunčiamų kviečiamiesiems asmenims. Kalbant apie naudojamumą, kadangi funkcijų šioje sistemoje yra tikrai daug, ne visada pavyksta

pilnai jas visas išnaudoti, nes paprasčiausiai sunku surasti kur ji yra, tam reikia truputėlį daugiau laiko kad priprasti prie funkcijų nuorodų išdėliojimo.

1.1.2.2 Eventsbot

Adresas: www.eventsbot.com

Pradinis puslapis:

Pav. 2 Events bot pagrindinis langas

Privalumai:

Visų pirma lyginant su eventsbrite.com sistema šios sistemos dizainas yra malonesnis.

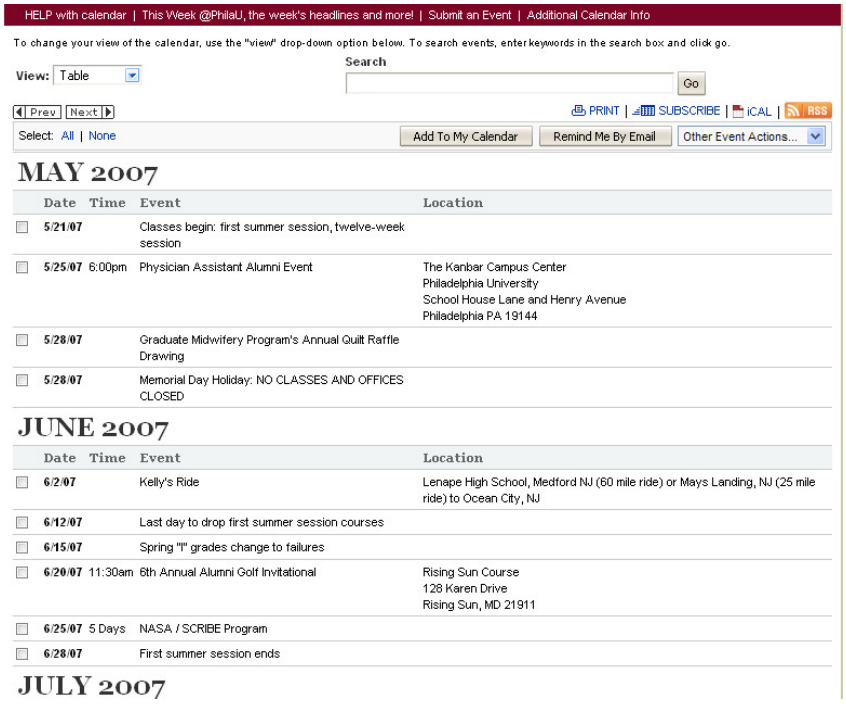
Renginių kūrimas taip pat yra nesudėtingas, joje daugiau iššokinėjančių langų, kurie leidžia neperkraunant puslapio įvesti renginio informaciją. Taip pat patogu kad tik įvedus renginio pavadinimą ir laiką iš karto esate nuvedami į puslapį, kuriame matote kaip atrodys renginio registracijos puslapis. Šiame puslapyje galite keisti visus renginio, bei registracijos puslapio parametrus.

Trūkumai:

Truputėlį skurdokas renginio statistikos peržiūros langas, matomai puslapio kūrėjai nespėjo tinkamai jo išdirbti, labiau koncentravosi į renginio kūrimą.

1.1.2.3 Trumba.com

Adresas: www.trumba.com



Pav. 3 Trumba pagrindinis langas

Privalumai:

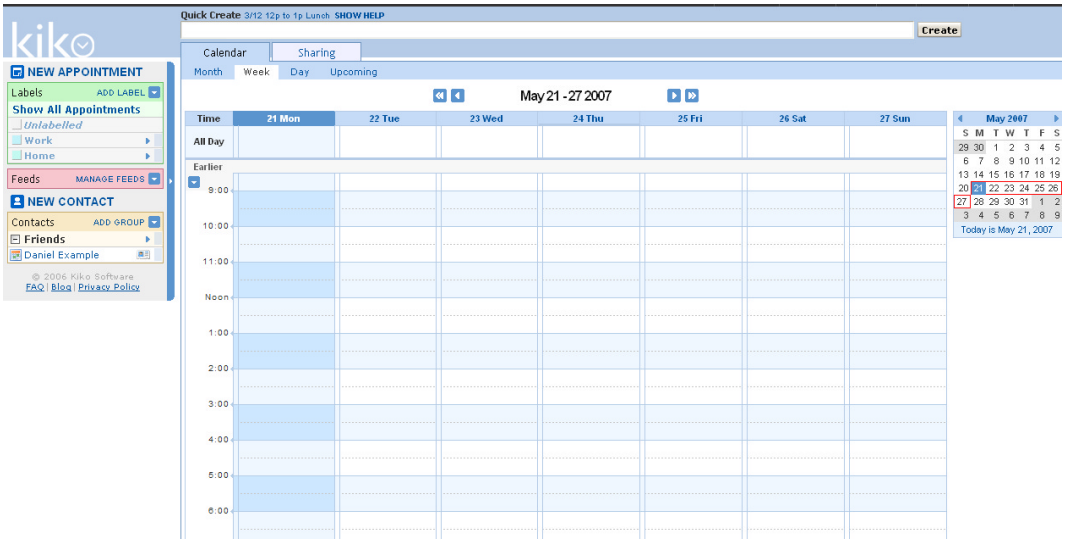
Kompanijos veikimas pagrįstas interaktyvių komponentų diegimu skirtingose svetainėse, pvz. Kompanija savo svetainėje nori turėti kalendorių, kuriame būtu atvaizduojamas renginių sąrašas su aprašymais. Šią paslaugą būtent ir tiekia Trumba. Kalendoriaus komponentas yra valdomas ir prižiūrimas Trumba korporacijos.

Trūkumai:

Taip pat neatitinka pagrindinė idėja mūsų projektuojamos sistemos. Trumba siekia patogiai atvaizduoti didelį kiekį renginių viename kalendoriaus komponente, o mūsų sistemos pagrindinis tikslas, padėti renginių organizatoriams išsiuntinėti ir valdyti pakvietimus. Brangus sistemos diegimas, palaikymas ir priežiūra. Sudėtinga pradinė išvaizda.

1.1.2.4 Kiko

Adresas: www.kiko.com



Pav. 4 Kiko pagrindinis langas

Privalumai:

Sistema yra Google Calendar analogas. Pagrindinis tikslas, internetinis asmeninis renginių kalendorius, prie kurio galima prijungti ir kitus – viešus kalendorius. Patogus kalendoriaus valdymas, nes sistema veikia Ajax technologijos pagrindu. Priminimai apie renginius el. paštu ir SMS žinutėmis.

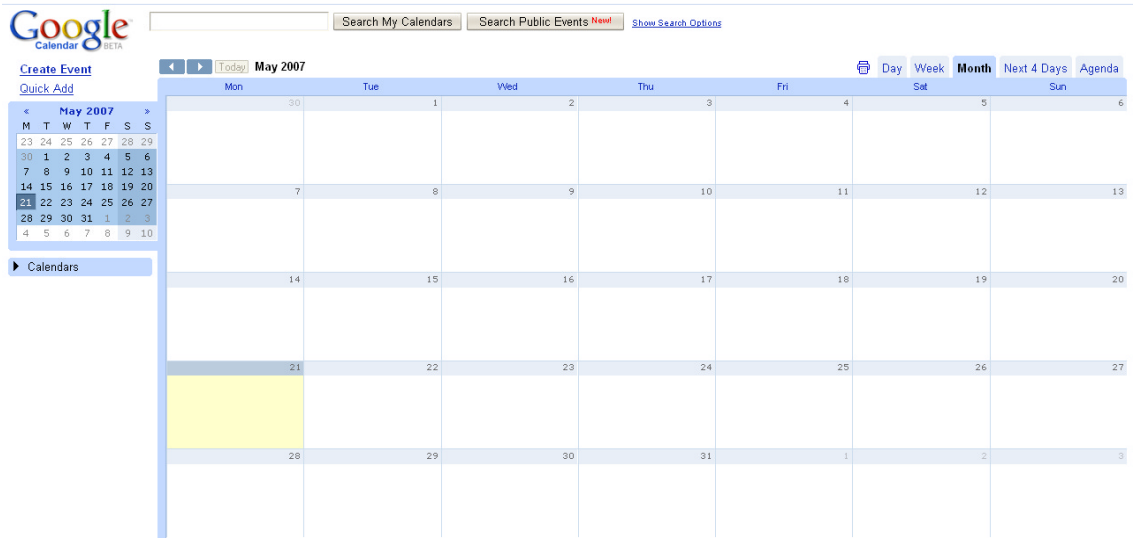
Trūkumai:

Šia sistema negalima sukurti pakvietimų. Negalima parinkti kvietimų dizaino, bei turinio. Pagrindinė paskirtis kalendorius, o ne renginio organizatoriaus padėjėjas.

Rinkoje egzistuoja keletas panašaus pobūdžio sistemų, tačiau jų visų pagrindinis tikslas nesutampa su mūsų sistemos tikslu. Didelės kompanijos, turi savo viduje panašaus tikslo sistemas, tačiau viešai jų neskelbia, ir naudoja savo poreikiams. Paanalizuokime tas sistemas kurios yra viešai prieinamos atskirai:

1.1.2.5 Google Calendar

Adresas: www.google.com/calendar



Pav. 5 Google Calendar pagrindinis langas

Privalumai:

Ši sistema yra sparčiai populiarėjanti, nors šiuo metu yra tik beta stadijoj. Apie Google produktus labai greitai sužino visas pasaulis, nes jų paieškos sistema naudojasi milijonai žmonių. Įdėję nors ir nedidelį užrašą paieškos sistemoje apie naują produktą, jis kaip mat tampa populiarius. Taip pat Google Calendar yra susietas su el. pašto sistema Gmail, kas dar labiau išpopuliarina šią sistemą.

Sistemoje yra galimybė kurti renginius, į juos pakviesti savo draugus iš kontaktų sąrašo. Automatiškai išsiuntinėjami el. laiškai kviečiamiems asmenims. Taip pat, jei gavėjas turi Google sistemoje įsivedęs savo mob. nr. Tai jis gauna ir trumpąją žinutę nemokamai, kurioje pranešama apie renginį. Suderinama su iCal.

Trūkumai:

Pagrindinė idėja skiriasi nuo mūsų projektuojamos sistemos. Tai yra, Google Calendar pritaikytas daugiau privačiam sektoriui, pavieniams asmenims, kurie neorganizuoja oficialių renginių, kuriems nėra svarbus įvaizdis. Negalima redaguoti el. laiško turinio. Jį automatiškai sugeneruoja Google sistema. Negalima pasirinkti laiško dizaino, jis visada bus vienodas. Negalima nurodyti kokias būdais bus nusiųstas pakvietimas. Negalima nurodyti gavėjų, visada bus siunčiama visiems pakviestiesiems. Komplikuotas gavėjų pasirinkimo būdas, reikia mintinai žinoti kviečiamų žmonių vardus ar pavardes, nes pasirinkimas atsiranda tik pradėjus vesti pirmas kontakto vardo raides.

1.2 Sistemos apibūdinimas

1.2.1 Programų sistemos funkcijos

Organizuojant privatų renginį yra svarbu turėti kviečiamų asmenų sąrašą. Šiame sąraše dominuoja visa kontaktinė informacija apie kviečiamus asmenis. Sudarius kviečiamų asmenų sąrašą, kitas etapas – pakviesti šiuos asmenis į renginį. Atsižvelgiant į kiekvieno sąraše esančio asmens kontaktinius duomenis, reikia parinkti kokiu būdu jį būtų galima pakviesti į organizuojamą renginį. Galimi pakvietimo būdai:

- Elektroniniu paštu.
- SMS žinute.
- MMS žinute.
- Telefonu.
- Paštu.

Organizuojantis asmuo parenka kokiu būdu bus kviečiamas konkretus asmuo. Po pakvietimų būdų parinkimo, reikia informuoti kiekvieną asmenį pasirinktu būdu. Šis organizavimo etapas užima daug laiko. Tačiau, išskyrus tiesioginį pokalbį telefonu, pranešus asmeniui tam tikru būdu apie renginį, nėra aišku ar asmuo dalyvaus renginyje. Turint kviečiamų asmenų sąrašą yra svarbu surinkti informaciją apie renginyje dalyvausiančius asmenis. Būtent sistema ir užtikrins renginyje dalyvausiančių asmenų registraciją. Asmuo gavęs pakvietimą, tuo pačiu gaus ir unikalų kodą, su kurio, prisijungęs prie sistemos, galės plačiau sužinoti apie renginį, peržiūrėti renginyje dalyvaujančių asmenų sąrašą, užsiregistruoti renginyje. Asmuo gali pasirinkti vieną iš būdų:

- Renginyje dalyvaus
- Renginyje nedalyvaus
- Dar neapsisprendė ar dalyvaus renginyje

Sistema registruos kiekvieno pakviestojo asmens apsisprendimą, t.y. ar asmuo:

- Dalyvauja
- Nedalyvauja
- Neapsisprendęs
- Į pakvietimą nesureagavo

Renginį organizuojantis asmuo matydamas pakviestųjų reakciją, atitinkamai galės spręsti apie renginyje dalyvausiančių asmenų skaičių, o taip pat matydamas jog pakviestasis asmuo nesureagavo, bandyti jį pakviesti pakartotinai arba kitu būdu

Registruotiems naudotojams, kurie kuria naują renginį, yra funkcija, palengvinanti ir paspartinanti darbą. Renginį kuriantis naudotojas gali priskirti moderatorius, kurie taip pat galės redaguoti naudotojo renginį. Moderatoriai yra taip Registruoti sistemos naudotojai.

Registruoti naudotojai galės naudoti pakvietimų šablonus, kurie pagražins, pagyvins ir padarys naudotojo pakvietimus išskirtiniais. Pakvietimų šablonus naudotojai gali susikurti patys arba užsisakyti jau esamus sistemos šablonus. Ši funkcija padės naudotojams turėti ir išsiuntinėti, būtent jiems patinkančio dizaino pakvietimus, kiek galima mažiau suvaržant naudotojo poreikius ir norus.

Užsiregistravęs naudotojas galės:

- susikurti ir naudotis nuolatine savo kontaktų duomenų bazę sistemoje, kurios pagalba galės per kelias minutes juos pridėti į kviečiamų asmenų sąrašą;
- kurdamas naują renginį galės sudaryti renginio aprašą, dienotvarkę, kviečiamųjų asmenų sąrašą, pakvietimų šablonus, kurie atitiks naudotojo poreikius;
- paskirti moderatorius, kurie galės padėti redaguoti organizuojamą renginį;
- paskelbti renginį, kuris galės būti viešas arba privatus;
- nustatyti būdus, kuriais mūsų sistema pasirūpins informuoti kviečiamus asmenis;
- stebėti kviečiamųjų asmenų reakciją į gautus pakvietimus (ar asmuo užsiregistravo, kad dalyvaus, ar kad nedalyvaus, ar dar neapsisprendęs, ar jokio atsako iš to asmens nebuvo sulaukta);
- matydamas kviečiamųjų asmenų reakciją, galės nuspręsti kokių veiksmų toliau imtis;
- matyti viešai skelbiamus renginius.

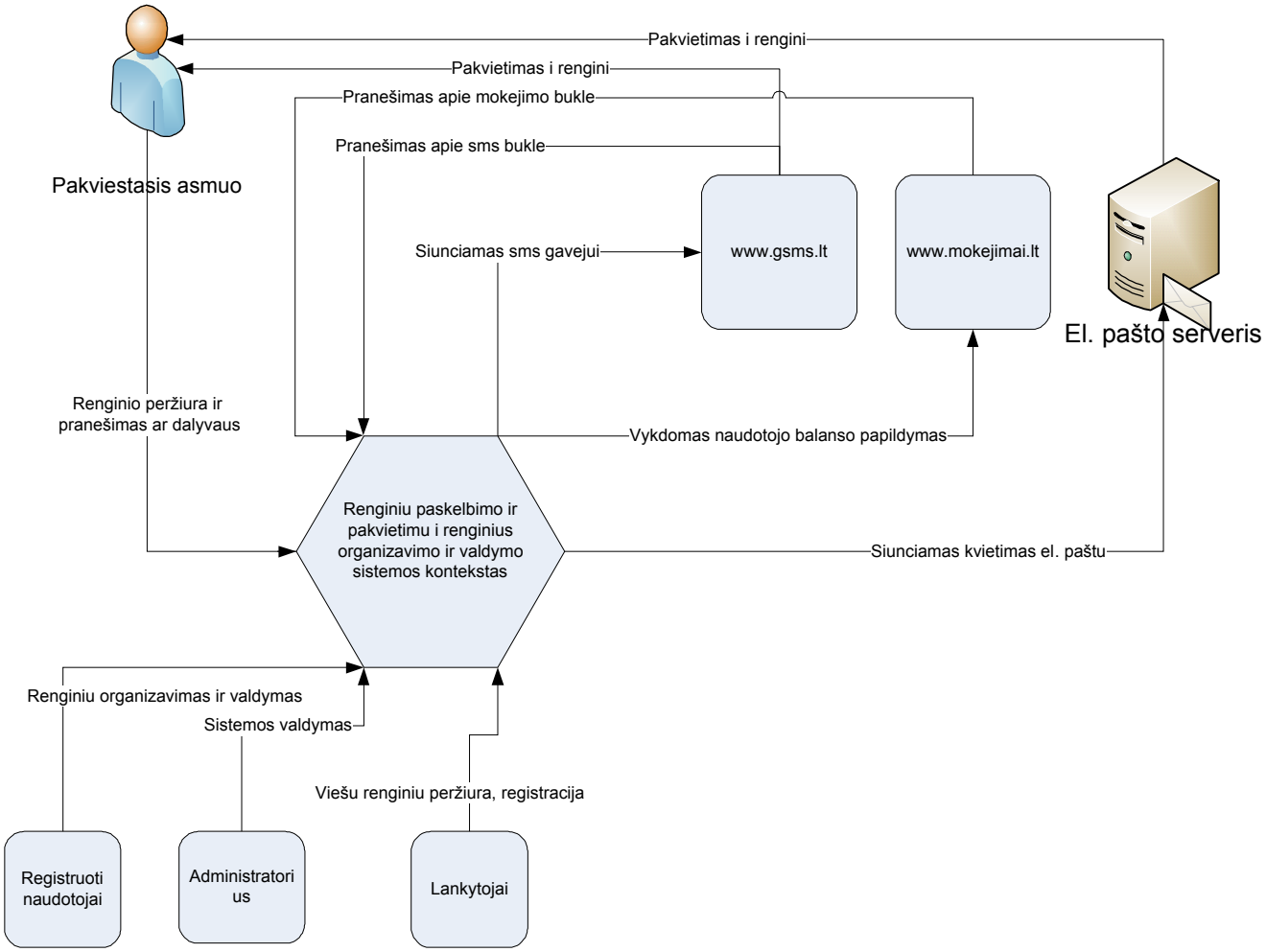
1.2.2 Sistemos kontekstas

Kuriama sistema bus įgyvendinta kaip tinklo programa (WEB application). Ją galima bus pasiekti iš bet kurio kompiuterio, kuris yra prijungtas prie interneto ryšio, adresu www.invup.com . Visa sistemos naudojama informacija bus saugojama duomenų bazėje. Sistema naudoja savo vartotojų autentifikaciją. Visi veiksmai kuriamoje sistemoje atliekami naudojant tą pačią vartotojo sąsają, skiriasi tik sąsajos panaudojimo galimybės, priklausomai nuo vartotojo teisių sistemoje. Sistema yra pritaikyta konkrečiai problemai spręsti ir pakartotinis jos panaudojimas yra įmanomas tik tam tikriems sistemos komponentams. Kuriama sistema yra priklausoma nuo kitų sistemų:

www.mokejimai.lt – sistema per kuria atliekami piniginiai pervedimai.

www.gsms.lt – sistema per kurią išsiuntinėjamos trumposios sms žinutės.

Sistemos projekto vizualus ryšių atvaizdavimas:



1.2.3 Vartotojo charakteristikos

Naudotojas norėdamas naudotis mūsų informacine sistema privalo mokėti naudotis internetine naršykle, bei turėti bendrą supratimą kaip naudotis teksto redaktoriumi (šablonams kurti).

Taip pat naudotojas turi turėti personalinį kompiuterį, kuris būtų prijungtas prie interneto. Naudotojo kompiuterio minimalūs parametrai turi būti tokie, kad palaikytų normalų šiuolaikinės internetinės naršyklės darbą. Kiekviena naršyklė turi skirtingus reikalavimus aparatinei įrangai. Mes siūlome naudotis Firefox internetine naršykle, kadangi ji veikia sparčiau, bei yra saugesnė, be to galima gauti įvairių priedų, kurie labai palengvina darbą internete, ir yra nemokami. Firefox naršyklės rekomenduojami reikalavimai kompiuteriui yra tokie:

- 500 Mhz procesorius
- 256 operatyviosios atminties

- 100 MB laisvos vietos diske
- Microsoft Windows XP operacinė sistema

1.2.4 Vartotojo tikslai

Produktas bus orientuotas į vyresniąją vartotojų kategoriją. Sistemos išvaizda turi būti solidi, patraukli akiai, tačiau ne per daug spalvota ir neapkrauta. Iš pirmo žvilgsnio sistemos išvaizda turi sudaryti pasitikėjimo ir ramumo jausmą. Išvaizdai keliamas pagrindinis reikalavimas, kad ji atrodytų paprasta ir elementari, lengva naudotis, tačiau tuo pačiu turi netrūkti komponentų, palaikančių visišką sistemos funkcionalumą. Taip pat išvaizda turi sudaryti ir privatumo bei konfidencialumo jausmą, kad vartotojas, tik išvydęs dizainą, nesusidarytų pigaus ir nepatikimo puslapio įspūdžio. Tačiau to vien tik nepakanka, kad pritraukti naujų vartotojų dėmesį, privalėtų būti ir šiokia tokia intriga, kad vartotojas būtų susidomėjęs užsiregistruoti renginių organizavimo sistemoje.

Kadangi vartotojo sąsaja yra vienas iš svarbiausių programos komponentų, todėl labai svarbu, kad ji pasižymėtų šiomis savybėmis:

- Patogumu;
- Lengva navigacija;
- Paprastumu;
- Neapkrauta aplinka;
- Lengvu įsisavinimu.

1.2.5 Bendri apribojimai

Sistema veiks visuose kompiuteriuose, kurie turės priėjimą prie interneto, nepriklausomai nuo kompiuteriuose esančios operacinės sistemos. Būtina sąlyga, kad kompiuteryje, turinčiame priėjimą prie interneto, būtų įdiegta interneto naršyklė. Interneto naršyklės pagalba vartotojas turės galimybę pasiekti Invup sistemą. Rekomenduotina atviro kodo interneto naršyklė „Mozilla Firefox“, kuri yra greita ir patogi, tačiau sistema puikiai funkcionuos ir su kitomis interneto naršyklėmis: MS Internet Explorer, Opera, Netscape. Tam, kad sistema funkcionuotų reikia, kad naudojama naršyklė palaikytų XHTML 1.0 žymėjimo standartą, Javascript programavimo kalbą (be to naršyklėje turi būti nustatyta, kad būtų galima vykdyti Javascript komandų interpretavimą).

Visos funkcijos, aprašytos dokumentacijoje yra įgyvendintos galutiniame produkte. Projekto eigoje sprendėsi apimtis, bendraujant su užsakovu, keičiant ir papildant funkcijų sąrašą, išvardintą Reikalavimų specifikacijoje.

Prie sistemos galės prisijungti naudotojas, susikūręs anketą sistemoje. naudotojas galės prisijungti tiktai turėdamas priėjimą prie interneto. Prisijungimas bus patvirtinamas prisijungimo vardu ir slaptažodžiu, kuriuos žinos tik užsiregistravęs naudotojas. Naudotojų skaičius nebus ribojimas ir jų skaičius neįtakos sistemos funkcionavimo.

Renginių organizavimo sistema internete yra Web programa skirta renginių paskelbimui ir organizavimui. Kontaktus galima įkelti iš failo tik pagal tam tikrus nustatytus standartus. Kontaktus išsisaugoti į failą galima pagal tam tikrus standartus. Pildymo formose nustatyti laukai, kai kurie privalomi, kiti pasirinktini.

Sistema užtikrins dalyvių pakvietimą nustatytu būdu ir laiku. Pakvietimo laikas priklausys nuo pasirinktos šalies laiko juostos nustatymų.

Sistema turės apsaugą nuo naudotojo klaidingai įvestų duomenų. Taip pat bus apribojimai tyčiniams klaidingų duomenų įvedimams. Naudotojui bus paaiškinta ir nurodyta kokio tipo duomenys yra priimtini.

Kad būtų galima teikti paslaugas, reikia turėti serverį arba užsisakyti hostingo paslaugas. Serveryje turėtų būti veikiantis Apache servisas, įdiegta MySQL duomenų bazės valdymo sistema ir taip pat įdiegtas PHP kalbos palaikymas, be kurių neįmanomas dinaminės sistemos veikimas, ši programinė įranga pasirinkta todėl, kad ji labai puikiai sąveikauja tarpusavyje, o tuo pačiu yra nemokama.

Sistema užtikrina naudotojų duomenų konfidencialumą apie jų turimus kontaktus ir organizuojamus renginius. Sistemos veikimas stebimas ir prižiūrimas, tai padės greitai reaguoti į iškilusias klaidas. Taip pat, php kalbos ir serverio apsaugų pagalba bus pasirūpinta ir apsauga nuo išorinių įsilaužėlių. Užtikrintas sistemos veikimas ir priėjimas prie jos 24 valandas per parą ir 7 dienas per savaitę.

1.3 Projekto įgyvendinimo planai ir kokybės vertinimas

Produkto įvertinimo būdai ir priemonės.

Projektas buvo įvertintas, atsižvelgiant į dabartines įrangos ir technologijų kainas bei vidutinio dydžio atlyginimus. Projekto vertinimas atliktas kartu su sistemos užsakovu, Mantu Žvironu, ir projekto vadovu. Pagrindinę projekto sumos dalį sudaro žmoniškieji ištekliai.

Kokybės vertinimo kriterijų pasiūlymas.

Sistemos kokybę įvertinti galima tokiais kriterijais:

- Veikimo greitis.
- Paprastumas naudotis.
- Saugumas.
- Paslaugų ir papildomų funkcijų kiekis ir kainos.

Konkurentų palyginimas pagal kriterijus

Didelį funkcionalumą siūlo „Eventbrite“ ir „Eventsbot“ sistemos, tačiau jomis naudotis nėra pigu. „Trumba“ turi visa pagrindinį funkcionalumą, organizuojant renginį. „Kiko“ ir „Google calendar“ yra pigios alternatyvos, tačiau šios sistemos nėra būtent sutelktos į renginių bei bilietų valdymą ir statistikos vedimą. Veikimo greičiu pasižymi „Kiko“ ir „Google calendar“, tačiau jų funkcionalumas yra mažas ir tai truputį kitokio pobūdžio sistemos.

Skaičius prie atitinkamo konkurento ir kriterijaus parodo to kriterijaus įvertinimą. Įvertinimas gali būti nuo 1 – blogas, iki 10 – geras.

Pavadinimas	**Greitis**	**Paprastumas**	**Saugumas**	**F-jų kiekis**	**Kaina**	**Dizainas**	**Bendras**
Eventbrite	7	8	7	7	9	7	**7,50**
Eventsbot	7	8	7	7	9	8	**7,67**
Google calendar	8	8	10	1	10	7	7,33
Kiko	8	8	8	1	10	8	7,17
Trumba	6	6	9	9	5	6	6,83

Lentelė Nr. 1 Konkurentų palyginimas pagal kriterijus

Tolesnės plėtros galimybės – tai saugoma ir kaupiama informacija apie visus svečius, kaip konkrečius asmenis, ir apie jų dalyvavimą tam tikruose renginiuose. Tai yra labai vertinga informacija renginių organizatoriams, kurie yra labai susidomėję nuolatos plėsti savo lankytojų ir dalyvių ratą.

Projektą įgyvendinantis personalas ir jų pareigos.

Mantas Žvironas - įmonės vadovas, projekto užsakovas, rūpinasi projekto palaikymu.

Mindaugas Vilniškis – projekto vadovas, rūpinasi visu projekto plėtojimu, nuo pradžių iki galo.

Egidijus Gegeckas – atliekami darbai: reikalavimų sudarymas, detalus projektavimas, programavimas, testavimas.

Domas Greičius - atliekami darbai: reikalavimų sudarymas, detalus projektavimas, programavimas, testavimas.

Projekto įgyvendinimo fazės.

Mūsų kuriamos sistemos projekto įgyvendinimo fazės yra šios:

- Reikalavimų surinkimas
- Reikalavimų specifikavimas
- Detalus projektavimas
- Realizavimas
- Testavimas
- Palaikymas

Reikalavimų analizės metu pagrindinis dėmesys yra kreipiamas į tai, ką sistema turi atlikti ir kokie yra jos apribojimai , prie kurių sistema turi vykdyti savo funkcijas. Yra nustatomos visos funkcijos kurias turės sistema atlikti, tačiau kaip tos funkcijos bus atliekamos yra detalizuojama projektavimo etape.

Kuriant sistemą yra laikomasi krioklio metodo, kuomet pirmiausiai yra sudaroma reikalavimų specifikacija, tuomet pereinama prie detalaus projektavimo ir realizavimo ir galiausiai atliekamas sistemos testavimas.

1.4 Įgyvendinimo problemos

1.4.1 Sparta

Web technologijų sparta yra matuojama pralaidumu, gaišties laiku, vykdymo laiku ir operacijos laiku. Pralaidumas yra skaičius, parodantis pateiktų užklausų skaičių per tam tikrą laiką. Gaištis yra laikas, per kurį yra nusiunčiama užklausa ir gaunamas atsakas. Vykdymo laikas yra laikas, skirtas sistemai atlikti visą jai skirtą veiklų seką. Operacijos laikas parodo Web sistemos užtrunkumą laiką, per kurį yra įvykdoma viena pilna operacija. Kuo didesnis pralaidumas, mažesnis gaišties, vykdymo laikas ir kuo greitesnis operacijos laikas, tuo web sistema greičiau veikia.

Visa Web sistemos sparta priklauso nuo programos logikos, tinklo ir labiausiai nuo esančių viduje žinučių siuntinėjimo ir duomenų perdavimo protokolų, tokių kaip SOAP ir HTTP.

Nepaisant SOAP, reikia atkreipti dėmesį ir į XML apdorojimo žingsnius. Keli iš būdų, kaip optimizuoti XML apdorojimo žingsnius [1]:

- Naudoti veiksmingus ir lengvus analizatorius
- Efektyvus XML validavimo naudojimas kūrimo stadijoje
- Suspausto XML naudojimas, siunčiant žinutes per tinklą
- Kiek galima paprastesnių duomenų tipų naudojimas SOAP žinutėse.

1.4.2 Paprastumas

Ar niekados neteko stebėtis nepaprastu bendrumu tarp be galo funkcionalių svetainių? Net nereikia skirti daug laiko, norint palyginti tam tikras savybes tokių svetainių, kaip Cisco, Marshall , Dell, Amazon ir pan., ir atsakymas bus neišvengiamas. Taip, jos visos suprantamos ir paprastos, draugiškos ir funkcionalios. Jokių bereikalingų kosmetinių pagražinimų, jokios painiavos. Paprastumas Web dizaine turi tiek pat svarbos, kaip ir bet kurioje kitoje gyvenimo sferoje.

Web dizainas juda draugiškumo ir paprastumo naudojamume linkme. Norint tą pasiekti, reikia, visų pirma, suformuluoti Web sistemos pagrindines atliekamas funkcijas ir gerai perprasti savo planuojamą vartotojų ratą.

Pagrindinių funkcijų suformulavimas, kuria kryptimi kompanija judės yra vienas iš svarbiausių dalykų. Tai suteiks tam tikrą stabilumą, kurio pagalba internetinės sistemos kūrimas turės pradinį tašką ir aiškią kryptį, kuria ir reikia toliau judėti. Reikėtų pradėti nuo tokių klausimų:

- Koks yra kuriamos svetainės tikslas?
- Kokie bus svetainės vartotojai?
- Kaip bus išmatuojama svetainės sėkmė?
- Kaip gauti atsakomąjį ryšį iš vartotojų ir įtraukti juos į svetainės kūrimo procesą?

Reikia gerai apgalvoti apie internetinės svetainės išdėstymą ir navigaciją. Reikia pasistenkti išlaikyti tokį patį dizainą visoje svetainėje, kad naršant po ją, visas išdėstymas išliktų nepakitęs, keistųsi tik išvedamas turinys. Reikia pilnai įsitikinti ar tikrai svetainės užkrovimo laikas yra pats optimaliausias. Norint sumažinti kiek įmanoma svetainės užkrovimo laiką, reikia [2]:

- Optimizuoti HTML ir skripto kodą: įsitikinti ar tikrai nėra nereikalingų HTML žymų ar nenaudojamų scenarijumi.
- Naudoti serverio pusėje turimus (SSI) failus: SSI failai vieną kartą iškviesti iš Web serverio pasilieka laikinojoje atmintyje ir vėlesnėms užklausoms krovimo laikas sumažėja.
- Bekuriant dizainą, apsvarstyti visas įmanomas ekrano skiriamąsias gebas.

Kuriama svetainė turi būti interaktyvi, tai yra bendraujanti su vartotoju ir vertinga savo turiniu.

Norint, kad svetainė būtų skrupulinga, patartina naudoti tokių interaktyvių funkcijų, kaip [2]: svetainės apklausos, tyrimai, svečių knyga, įvykių kalendorius, naujienų laiškų užsiprenumeravimas. Tačiau visa tai reikėtų naudoti su saiku. Patartina išlaikyti nuoseklumą visoje internetinėje svetainėje.

Beprojektuojant svetainę yra labai svarbu, kad ji būtų patraukli, greitai užsikraunanti, draugiška vartotojui ir visas dėmesys būtų sutelktas į svetainės turinį. Šie faktoriai lemia profesionalumą išvaizdoje ir kokybėje, tačiau nemažesnį indėlį turi ir vertė, sutelkta į funkcionalumą.

1.4.3 Naudojamumas

Vis dažniau yra akcentuojama kuo paprastesnė svetainių navigacija, kuo mažiau apkrauta vartotojo sąsaja, tačiau pilnas funkcionalumas, reikalui esant, privalo išlikti, visa informacija yra centruojama per vidurį.

Kuo paprastesnis naudojamumas jau tapo privalomu akcentu, kuris yra būtinas Web plėtojimui. Vis labiau el. verslas supranta, kad sistemos, kurios pasižymi skurdžia vartotojo sąsaja, taip pat ir pačiam verslui neša mažą investicijų grąžą [3]. El. verslo klientai pradeda reikalauti kuo patogesnės ir paprastesnės vartotojo sąsajos iš Web kūrėjų, tačiau išlaikomas ir pilnas funkcionalumas.

1.4.4 Patikimumas

Patikimumas yra vienas iš svarbiausių kriterijų, vertinant Web sistemų kokybę. Skaičius sutrikusių funkcionavimų per dieną, per savaitę ar per metus yra svarbiausias dydis, parodantis sistemos patikimumą. Šiuo metu Web technologijos pasikliauja tokiais duomenų perdavimo protokolais, kaip HTTP. Tačiau tai dar negarantuoja, kad žinutė bus pristatyta į vietą. Vienas iš būdų sumažinti riziką yra asinchroninės žinučių eilės, kitas būdas būtų paveldėjimas naujų patikimų duomenų perdavimo protokolų, tokių kaip HTTPR, REST, BEEP [1].

Patikimumas apibrėžia ar sistema bus prieinama tam tikru laiko momentu. Su patikimumu yra susijusi TTR (angl. Time-to-Repair) technologija. TTR atstovauja laiką, kuris bus sugaištas, tvarkant tam tikrą Web paslaugą.

Kurti web paslaugas, kurios net ir įvykus tam tikrai klaidai veiktų toliau, yra brangu. Kompanijos, pasirinkusios šį kelią, susiduria su skirtingose vietose išdalintų technologijų problemomis. Kyla tokie klausimai, kaip [1]:

- Ar tik ne vienas iš pagrindinių serverių tapo nepasiekimas?
- Ar sistema netapo per daug apkrauta?
- Kodėl užklausos trunka taip ilgai?

Dauguma renkasi multifunckinį web paslaugų valdymą, web servisų „klasteriavimą“.

1.4.5 Saugumas

Didėjant web technologijų paklausai, kurios yra prieinamos per viešą internetą, auga susirūpinimas dėl saugumo. [5] Web kūrėjai vis dažniau susiduria su nuolatos didėjančiu spragų kiekiu, kurios tik dar labiau apsunkina darbą. Web paslaugų saugumas reiškia konfidencialių duomenų perdavimą, tik įgaliotiems asmenims, žinučių šifravimą, ir priėjimo kontrolės tiekimą. Web sistemoje gali būti skirtingų lygių konfidenciali informacija, kurią yra būtina apsaugoti. Su saugumu susijusios problemos negali būti paliktos nuošalyje, sistemos saugumas privalo užtikrinti vartotojų pasitikėjimą.

Vienas iš didžiausių rizikos šaltinių organizacijoms yra Web serveriai. Įsilaužimas į serverius gali padaryti labai didelių problemų organizacijoms. Šias pažeidžiamas spragas galima suskirstyti į keletą kategorijų [6]:

Standartinė konfigūracija. Web serveriai yra dažnai įdiegiami tiesiog su standartine konfigūracija, kuri gali būti nesaugi. Pažeidžiamos vietos gali būti standartiniai pavyzdžiai, šablonai, administravimo įrankiai, nuspėjamos paslaugų programų vietos. Be atitinkamo rizikos apsisaugojimo tai gali nuvesti į keletą skirtingo tipo įsilaužimų, po kurių piktavaliai vartotojai gali perimti pilną Web serverio valdymą

Vartotojo įvedamų duomenų tikrinimas. Internetinės svetainės turi būti interaktyvios, kad būtų naudingos vartotojui. Neapsisaugojus nuo vartotojo įvedamų duomenų, gali kilti daugybė saugumo problemų, kuriomis gali pasinaudoti piktavalis vartotojas ir įsilaužti į Web serverį bei prieiti prie duomenų, esančių duomenų bazėse.

Duomenų užšifravimas. Kad ir kaip bebūtų gaila, nors ir dabartiniai šifravimo algoritmai virtualiai yra nenulaužiami, tačiau jie nėra pakankamai naudojami. Taip yra dėl didelio kainos ir kokybės skirtumo tarp procesorių ir specializuotų šifravimo greitintuvų. Beveik visas šiuo metu tinklo srautas keliauja nešifruotas ir gali būti lengvai perimtas piktavalių vartotojų

Duomenų laikymo apsaugojimas. Dauguma konfidencialių failų yra saugomi viešai prieinamuose serveriuose, vietoj to, kad būtų talpinami specialiai apsaugotuose serveriuose. Kai kurios programos skiria per mažai dėmesio laikinų failų išvalymui, palikdamos svarbią informaciją lengvai prieinamą.

Sesijos valdymas. Dauguma svetainių per mažai dėmesio skiria unikalių sesijų valdymui. Tai gali būti ir naudojami silpni autentifikavimo metodai, prastas sistemos sausainiukų (angl. cookies) valdymas, nesugebėjimas nustatyti sesijos neveiksnumo laiko ir panašios sesijų silpnybės. Tai dažnai veda į sesijų užgrobimus, kur užgrobėjas gali perimti visus vartotojo duomenis.

Dauguma šių svarbių problemų ir yra pagrindinės saugumo spragos Web sistemoje. Organizacijoms, kurioms saugumas yra labai svarbus faktorius privalo užtikrinti sistemos saugumo palaikymą ne tik sistemos kūrimo metu, tačiau ir jos veikimo bei plėtojimo metu.

1.4.6 Praplečiamumas

Kuriant internetinę sistemą yra būtina atsižvelgti ir numatyti sistemos praplečiamumą ateityje. Į tai įeina ir naujų savybių įdėjimas, ir sistemos galimybių praplečiamumas, prijungimas naujų modulių.

Kai prireikia tvarkytis su gerokai išaugusiu lankytojų srautu, yra susiduriama su dviem, gana svarbiais parametrais – tai tinklo perdavimo greitis ir serverio atsako laikas. [9] Tinklo duomenų perdavimo greitis priklauso nuo interneto prieigos spartos, o serverio atsako laikas priklauso nuo tokių resursų, kiek greitas procesorius, daugybė operatyvinės atminties ir gero įvesties, išvesties įrenginių veikimo. Norint pagerinti sistemos veikimą, galima statyti galingesnius procesorius, įdėti daugiau

operatyvinės atminties, taip pat galima pakonfigūruoti operacinę sistemą arba Web serverio programinę įrangą. Tačiau tai ilgainiui neišgelbės. Tuomet šią problemą reikia spręsti kitu būdu. Pagerinti veikimą galima padidinus web serverių kiekį. Tai yra nukreipti duomenų srautą į giliau esančius serverius. Pagrindinį serverį paliekant, kurį gali pasiekti vartotojai, tačiau iš jo visus duomenų srautus nukreipti į atskirą serverių grupę (angl. cluster). Šie serveriai nebūtinai turi būti pačios optimaliausios konfigūracijos, toks duomenų srauto išskaidymas puikiai veikia ir su vidutinio galingumo serveriais.

1.4.7 Palaikomumas

Kol kuri ir programuoji web sistemą, struktūra, kurią naudoji, puslapio išdėstymas ir viskas, kas susiję tarpusavyje tampa gerai žinoma ir pažįstama. Tačiau praeina tam tikras laiko tarpas, pridedi vienur šiek tiek, pridedi kitur, ir po kurio laiko sistema tampa didžiulė. Po kurio laiko atmintis išblėsta ir ilgainiui tampa nebeaišku, kodėl vienas dalykas yra čia, o ne ten, kam yra skirtas šitas principas. Dabar, jeigu norėsi ką nors pakeisti, yra didelė galimybė, kad keičiant vieną dalyką, gali nustoti funkcionavę kiti. Kad to išvengti, reikėtų būti nuolatos pastoviam ir naudoti vieną pasirinktą metodologiją ir principą, nedaryti tų pačių dalykų skirtingais būdais. Verta naudoti komentarus. [10] Yra blogai manoma, kad komentarai skirti tik keistuoliams, kad jie galėtų suprasti ar net pasisavinti jūsų metodus. Nes labiausiai tikėtina, kad tas keistuolis jūs pats ir būsite, kuris mokinsis iš komentarų.

Patartina naudoti pastovią tiek pačio tinklapio, tiek pačios sistemos projektavimo struktūrą. Vienas iš būdų įvesti sistemingą struktūrą yra MVC (modelis-vaizdas-kontroleris) naudojimas. Tai tam tikra loginė bei praktinė metodologija, atskirianti veiksmus su duomenų bazėmis, išskirianti sistemos objektų klases, skirtas kontroliuoti sistemos vaizdą ir duomenų srautus.

1.4.8 HTML žymos el. laiškuose

Norint sudaryti siunčiamų el. laiškų sąrašą ir juos išsiųsti, verta panagrinėti problemas, su kuriomis galima susidurti. Visų pirma žvelgiant iš techninės pusės, reikia išsiųsti laiškus, kuriuos žmonės galėtų perskaityti. [11] Gryno teksto el. laiškai bus priimtini visuose el. pašto klientuose, bet jie nebus tokie patrauklūs, kaip iš HTML žymų sudaryti laiškai. Tačiau, nelaimei, nebus dviejų el. pašto kliento, kurie atvaizduotų el. laišką taip pat. Reikia patikrinti dažniausiai naudojamus el. pašto klientus ir sąsajas, kad pavyktų įsitikinti, kaip atrodys galutinis produktas. Mintis, kad el. pašto klientai supranta HTML žymas, dar nereiškia, kad reikia naudoti tokius pačius metodus, kaip projektuojant web puslapio dizainą. Negalima įtraukti jokių kliento pusėje naudojamų „skriptų“ (pvz. JavaScript), nes jie bus daugelio programų užrakinti. Negalima naudoti įvedimo formų, nes jos dažniausiai bus nukirptos. Mažesnė tikimybė, kad el. laiškai bus palaikyti kaip šiukšlės (angl. spam) ir užblokuoti jeigu HTML kodas yra validus. Jeigu laiškai yra siunčiami iš serverio, reikia įsitikinti, kad yra nustatytas vartotojo vardas, o ne pvz.: nuobody@ arba apache@. Reikia įsitikinti, kad yra sutvarkyti atgaliniai DNS nustatymai. Reikia įsitikinti, kad gerai nustatyta el. pašto apatinė antraštė (angl. footer).

Yra daugybė priežasčių, dėl kurių HTML žymos nėra pageidautinos el. laiškuose [12]:

- HTML el. laiškai yra pavojingi, kodas gali būti automatiškai įvykdytas, vartotojui to net nežinant.
- HTML el. laiškai be reikalo apkrauna tinklo pralaidumą.
- HTML el. laiškai ne visada atvaizduojami, kaip buvo tikėtasi.
- HTML el. laiškai yra sugeneruojami daug lėčiau.

Taigi yra daugybė problemų, su kuriomis susiduriama, norint patraukti klientų dėmesį, gražiais ir spindinčiais HTML el. laiškais.

1.4.9 Masinis SMS žinučių siuntimas

Prieš išsiunčiant SMS žinutę yra daugybė darbų, kurie turi būti atlikti. Taigi, štai žingsniai, kurie turi būti atlikti kiekvienai žinutei, norint ją išsiųsti [13]:

- SMS centras turi būti informuotas apie žinutės priėmimą.
- SMS centras turi nustatyti iš kurio siųstuvo bus priiminėjama SMS žinutė. Tai įtraukia CCITT7 (C7) tinklo užkrovimą.
- SMS centras turi nustatyti gavėjo siųstuvą.
- SMS centras turi nustatyti žinutės perdavimo kanalus
- Perduodant žinutę, SMS centras turi įsitikinti ar gavėjas gavo žinutę, ar ją reikia išsaugoti ir perduoti vėliau.

Ir daugybė įvairių kliūčių, su kuriomis galima susidurti atliekant didelius žinučių kiekių siuntimus tuo pačiu metu.

1.4.10 Daugiakalbiškumas

Jeigu kuriama sistema yra orientuota į pasaulinę rinką, jeigu sistemos klientai kalba skirtingomis kalbomis, tuomet reikia kurti tokią sistemą, kuri palaikytų daugiau negu vieną kalbą. Šiuo metu yra maždaug 128 milijonai [14] interneto vartotojų, kalbančių angliškai ir apie 88 milijonai [14] vartotojų, kurių gimtoji kalba nėra anglų.

Yra atlikta daugybė tyrimų, kurie parodo , kad vartotojai dažniausiai renkasi naršyti po sistemą savo gimtąja kalba [15].

Taigi kuriant pasaulinio masto sistemą reikia atsižvelgti ir į sistemos daugiakalbiškumo funkciją, kurios pagalba sistema taptų draugiškesnė vartotojams. Reikia teikti pirmenybę anglų kalbai, tačiau nepamiršti ir kitų šalių kalbų, į kurių rinką kuriamos sistemos užsakovai pretenduoja.

1.5 Pasirinktos technologijos sistemos realizavimui

1.5.1 Php

PHP (PHP hypertext preprocessor) tai scenarijų kalba (scripting language) kuri pačioje pradžioje buvo orentuota tik į internetą, nors šiuo metu ją galima "drąsiai" pavadinti programavimo kalba, kadangi su ja pilnai galima programuoti ne tik internetui. PHP pasirinkta, nes:

- php yra visiškai nemokamas
- php veikia įvairiose operacinėse sistemose: Win, *nix, MacOS, Solaris, HP-UX, AIX ir t.t.
- php yra atviro kodo projektas kurį kuria didelė grupė žmonių, todėl iškilusios klaidos yra greitai ištaisomos
- veikia ir ant daugelio WEB serverių: Apache, IIS, PWS, OmniHTTP, BadBlue ir t.t.
- išmokti PHP programavimo pagrindų yra labai lengva
- pasižymi dideliu greičiu serverio pusėje, bei dirbant su duomenų bazėmis
- nedideliuose projektuose PHP jūs paprastai galite įterpti į savo HTML'ą
- kūrėjai yra pirašę praplėtimų kiekvienam gyvenimo atvejui
- kadangi php programuotojų yra be galo daug, daugumą jau parašytų skriptų galima rasti internete: HotScripts, FreeScripts, PHPClasses.upperdesign.com ir t.t. ir t.t.

1.5.2 CakePHP karkasas

Tai turbūt pats aktyviausias karkasas šiuo metu (2007 m. lapkritis 14 d.). Nors jis yra tik beta versijoje tačiau kūrėjui siūlomų funkcijų gausa lenkia daugumą senesnių karkasų [7.4 02].

CakePHP pasirinkta dėl:

- MVC architektūros.
- Integruoto validavimo komponento. Formų duomenų patikrinimas atliekamas automatiškai, tereikia modelio faile nurodyti validavimo parametrus
- Crud programos kūrimo naudojant komandinę eilutę
- Duomenų valymo. Išvalo duomenis gautus iš formų nuo scenarijų neleistinų simbolių.
- Daugybės įvairių papildomų komponentų. Yra komponentai skirti sesijų valdymui, užklausų apdorojimui ir pan.
- Aktyvi, draugiška bendruomenė. Jei iškilo kokia problema, tau tikrai kas nors padės

1.5.3 jQuery

„Jūs pradedate nuo 10 jQuery kodo eilučių, kuris atitiktų 20 nuobodaus DOM Javascript kodo eilučių. Jums baigus kodas sutrumpėja iki dviejų ar trijų eilučių, ir jis jau negalėtų būti trumpesnis nebent jis skaitytų Jūsų mintis“ – Dave Methvin

Jquery sukurtas taip kad pakeistų jūsų supratimą apie javascript kodo rašymą. Kodas parašytas su jquery yra greitas ir glaustas. Be to visas jquery branduolys yra labai mažas - tik 14KB, kas yra labai svarbu, nes kuo mažiau duomenų reikia siųstis, tuo greičiau atverčiamas puslapis. Taip šis kodas be problemų veikia skirtingose naršyklėse.

1.5.4 Ajax

Ajax ("Asynchronous JavaScript and XML") – yra svetainių programavimo technologija. Naudojant Ajax technologiją yra pasiekimas svetainės maksimalus interaktyvumas. Ši technologija, nes naudoja šias priemones:

- HTML, bei stilių lenteles (Cascading Style Sheets) informacijos vaizdavimui
- DOM (Dokumento Objektinis Modelis), bei JavaScript kalbą dinamiškam vaizdavimui bei interaktyvumui
- XML, XSLT ir XMLHttpRequest objektą asinchroniniam duomenų apsikeitimui su serveriu

1.5.5 MySQL

MySQL - viena iš reliacinių duomenų bazių valdymo sistemų (liet. santrumpa RDBVS, angl. - RDBMS), palaikanti daugelį naudotojų, dirbanti SQL kalbos pagrindu. MySQL yra atviro kodo programinė įranga, kuri puikiai veikia kartu su PHP.

1.6 Raiškiųjų internetinių sistemų technologijų analizė

Šiame darbe planuoju apžvelgti Renginių organizavimo sistemos tobulinimą Silverlight technologija. Silverlight technologijos integracija, panaudojimas ir teikiami jos privalumai.

1.6.1 Web 2.0 tendencija

Pastaruoju metu vis dažniau, kalbant apie internetines sistemas, naudojamas Web 2.0 terminas. Taigi, kas yra ir ką reiškia Web 2.0?

Web 2.0 (antrosios kartos internetas) – specifinis bendradarbiavimas tarp žiniatinklio vartotojų (pvz., keitimasis informacija), kuriam būdingos sparčiai besikeičiančios komunikacijos internete bei žiniatinklio technologijos, specifinis dizainas, atitinkamos tendencijos, „internetinis“ kūrybiškumas. Antrosios kartos žiniatinklis prisidėjo prie įvairių svetainių bendruomenių bei socialinių paslaugų vystymosi. Būdinga socialiniai bendravimo tinklai (interneto bendruomenės), tinklaraščiai (internetiniai dienoraščiai), RSS prenumerata, wiki svetainės (MediaWiki programinė įranga), interneto „įskiepiai“ ir kt. Antrosios kartos žiniatinklio pasekėjas - vad. Trečiosios kartos žiniatinklis, Web 3.0 (arba netgi „Ketvirtosios kartos žiniatinklis“, Web 4.0).

Terminą WEB 2.0 pirma kartą paminėjo Tim O‘Reilly savo straipsnyje, apie šiuolaikines internetines sistemas. WEB 2.0 galima apibrėžti įvairiai, bet bendresne prasme tai yra technologijų, architektūrinių stilių, savybių, panaudojimo variantų rinkiniai, kurie naudojami šiuolaikinių internetinių aplikacijų, paslaugų kūrime. Galima būti išvardinti tokius pagrindinius WEB 2.0 terminus: Naudojamumas, Paprastumas, Mobilumas, atviri API (Aplication Programming Interface), SEO, SOAP, XML, Internetiniai standartai, RSS, Atom, Sindikavimas, Socialiniai tinklai, internetiniai dienoraščiai, AJAX, RIA ir pan [17].

WEB 2.0 visatoje internetas tampa platforma, kurioje turinį gali kurti ir naudoti eiliniai interneto vartotojai, blogų, socialinių tinklų, tinklalapių pagalba. Nors WEB 2.0 ir nėra nei standartas nei technologija tačiau jis parodo naujus turimų technologijų bei standartų panaudojimo būdus.

1.6.2 Raiškiosios internetinės sistemos (Rich Internet application)

Raiškiosios internetinės sistemos (angl. Rich Internet application - RIA) – tai tokios internetinės sistemos, kurios turi taikomųjų kompiuterinių programų charakteristikų, kurios veikia internetinių naršyklių įskiepiu pagalba arba veikiančios nepriklausomai virtualiosiose mašinose [18]. Raiškiųjų internetinių sistemų panaudojimo pavyzdžiai būtų tokie karkasai, kaip Adobe Flex/AIR, Java/JavaFX [19], uniPaaS [20] and Microsoft Silverlight [21].

Šis terminas buvo pristatytas 1990 metais, tokių prekinių ženklų kaip Macromedia, kurie akcentavo esamų laikų apribojimus kuriamoms internetinėms sistemoms, ir pasiūlydami patentuotas technologijas, kurti raiškiasnėms internetinėms sąsajoms [22].

Pagrindinis skirtumas tarp RIa sistemų ir paprastų internetinių sistemų yra interkatyvumo kiekio ir galimybių panaudojimas vartotojo sąsajoje. Tradicinėse internetinėse sistemose interkatyvumas yra apribotas maža naudojamų kontrolės elementų aibe, pvz. (angl - checkboxes, radio buttons, form fields and buttons). Būtent dėl šių priežasčių ir buvo toks didelis skirtumas tarp taikomųjų programų ir internetinių sistemų. Tai taip pat įtakojo sistemų panaudojimą daug siauresnei sferai. RIA sistemos būtent ir suteikia platesnį valdymo įrankių diapazoną, padidina sistemos interaktyvumą su vartotoju, leidžia tai padaryti labai efektyviai, siūlo geresnį klaidų valdymą, o visa tai ir sukuria gera atsakomąjį ryšį iš vartotojo pusės [23].

Pagrindinės tokių sistemų savybės:

- Vartotojo tiesioginis interaktyvumas su puslapio elementais (tiesioginis redagavimas, elementų slankiojimas po darbinę aplinką, atskirų kadrų užkrovimas).
- Atnaujinama tik dalis puslapio (vietoj to jog būtų perkraunamas visas puslapis).
- Daug daugiau detalesnės informacijos gali būti sutalpinta tame pačiame puslapyje (vietoj tos informacijos sudėjimo pagalbiniuose puslapiuose).
- Iš karto gaunamas patvirtinimų, validavimų, klaidų atsakas.

Visi šios raiškiosios ypatybės būtent ir būna didžiausias iššūkis tiek dizaineriams tiek ir programuotojams, kurie nori sistemas sukurti plačiai naudojamas.

1.6.3 Kas yra Silverlight

Silverlight – tai nepriklausantis nuo platformos ir naršyklės įskiepis, leidžiantis web-programuotojams ir dizaineriams kurti šiuolaikines dinamines web-aplikacijas .NET aplinkoje (iš dalies technologija panaši į Flash, bet yra pagrįsta .NET platforma) [24].

Silverlight yra naujas .NET karkaso skirtingose internetinėse naršyklėse ir skirtingose platformose panaudojimo būdas [25]. Silverlight technologija suteikia naujas galimybes naujos kartos raiškiųjų internetinių sistemų kūrimui ir panaudojimui. Ji veikia visose populiariausiose internetinėse naršyklėse, įskaitant Microsoft Internet Explorer, Mozilla Firefox, Apple Safari, Opera. Šioms naršyklėms reikalingas Silverlight įskiepis, kuris užima labai nedaug vietos ir yra iš karto instaliuojamas kliento kompiuteryje, jeigu pastarasis šio įskiepio dar neturėjo.

Tai skirtingų technologijų kombinacija, apjungta po viena kūrimo platforma, kuri leidžia naudoti įrankius ir kalbus, kurie yra patogesni kūrėjams. Silverlight galima lengvai integruoti su jau egzistuojančiu JavaScript ir ASP.NET AJAX kodu, kad būtų galima praplėsti jų esamą funkcionalumą. Silverlight konkuruoja su Adobe Flash ir įvairiais Ajax karkasais. Adobe Flash atsakomasis produktas yra Adobe Flex, kuris paremtas Flash ir ActionScript technologijų pagrindu.

Šiuo metu yra išleistos ir veikiančios dvi Silverlight versijos – tai Silverlight 1.0 ir Silverlight 2.0 (prieš tai planuota ir vadinta Silverlight 1.1 versija) [26].

Silverlight teikiami privalumai [29]:

- Silverlight įskiepis leidžia orientuotis į vieną platformą ir nekreipti dėmesio į skirtingas naršyklių savybes
- Galima vykdyti .NET kodą ir nereikia įdiegtos .NET veikimo aplinkos
- Silverlight veikimo našumas daug žadantis. Silverlight našumą galima palyginti puslapyje [27], kuriame skaičiavimai įgyvendinti Flash ir Silverlight technologijomis, kur yra skaičiuojami pirminiai skaičiai. Pabandymui buvo parinkta išrinkti pirminius skaičius iki 1000000. Flash technologijos pagrindu tai užtruko: 2.723 s, o Silverlight: 1.044 s, t. y. 2.6 karto greičiau. Beje, buvo paimtas šis kodas ir paverstas į JavaScript kodą, atlikto tyrimo rezultatai [28]: Safari, iMac 2.4Ghz užtruko 5.75 s. Firefox naršyklė užtruko 7.077 s.
- Vystomas Moonlight projektas skirtas atviro kodo platformoms.
- Silverlight interpretuoja XMAL kodą tiesiogiai, kaip tuo tarpu Adobe XML GUI kalba, MXML, paverčia į SWF formatą kompiliavimo metu. XAML puslapiai yra įterpti kaip atskiri resursai sukompiliuotame .XAP tipo faile, kuriame būna visa Silverlight programa. Iš tikrųjų .XAP tipo failas yra tas pats ZIP tipo failas, tik

pavadintas kitokiu plėtiniu. O tai reiškia kad paieškos sistemos gali indeksuoti tekstą, esantį Silverlight programoje.

- Trečiųjų šalių kuriami komponentai jau taip pat yra prieinami kaip Silverligt įskiepiai.
- Kadangi Silverlight programų logika realizuojama Visual Basic arba C# kalbomis, tai šias programas yra lengva pernešti nuo vienos platformos prie kitos.
- Silverlight kūrimo aplinkos įrankis integruojamas į Visaul Studio, kuris yra populiarus visame pasaulyje.
- Galima pasirinkti kokia kalba nori programuoti Silverlight programą. Nuo Silverlight 2 versijos, jos logiką galima realizuoti Visual Basic arba C# kalbomis, o taip pat ir Dynamic Language Runtime (DLR) aplinkos pagalba ir Iron Ruby or Iron Python kalbomis.

Silverlight trūkumai

- Jeigu Apple neleido naudoti Flash iPhone telefonuose, ar leis naudoti Silverlight?
- Silverlight yra naujas ir dar nebrandus projektas, lyginant jį su Flash kūriniais.
- Pakolkas niekas nenaudojo tokių dizaino įrankių kaip Expression Blend and Expression Design.
- Nors ir gerai išspręstas suderinamumas tarp Expression Blend ir Visual Studio įrankių, tačiau tai vistiek du atskiria ir nepigūs įrankiai.
- Nepalaiko H.264 video koduotės, o naudoja mažiau žinomą VC-1.
- Tai dar vienas būdas tobulinti ir skatinti Microsoft įrankių naudojimą
- Moonlight nors ir kuriamas sprendimas Linux platformoms, tačiau gerokai atsilieka nuo Silverlight pirmtako.
- Pakolkas visi Expression kūrimo įrankiai skirti tik Windows platformoms.

1.6.4 Silverlight versijos

Silverlight 1.0

Silverlight 1.0 susideda iš pateikiamo pagrindinio karkaso, kuris atsakingas už vartotojo sąsają. Interaktyvumą ir vartotojo įvedamus duomenis, pagrindiniai vartotojo sąsajos valdymo įrankiai, grafika, animacija, audio/video įrašų peržiūros ir DOM integracijos.

Silverlight 2.0

Silverlight 2.0 versija panaudoja .NET karkasą, kartu su pilnu CLR ir .NET Framework 3.0 palaikymu. Gali vykdyti bet kokį kodą, parašytą VB.NET and C# kalbomis. Taip pat susieta grafinė aplinka XAML kalbos formatu.

Šioje versijoje yra virš 30 vartotojo sąsajos valdiklių, įskaitant ir tekstinius blokus, slinkties juostas, kalendoriaus valdiklius ir pan., taip pat ir tokius elementus, kaip StackPanel, Grid, DataGrid ar ListBox, skirtus paruošti raiškiems šablonams.

Numatomos versijos

„Microsoft“ žada, kad trečioji ir ketvirtoji interaktyvaus internetinio turinio generavimo bei perteikimo „Silverlight“ posistemės versijos bus tobulinamos visiškai naujomis kryptimis. Anot korporacijos kūrimo padalinio viceprezidento Scotto Guthrie, „Silverlight 3“ ir „Silverlight 4“ leis daryti tai, ko kol kas neleidžia jokia internetinė interaktyvaus turinio sistema [30].

Žadama, kad „Silverlight 3“ bus numatyta aibė naujų funkcijų, pagerinsiančių grafikos perteikimo galimybes. Pavyzdžiui, internetiniu grotuvu bus galima atkurti H.264 formato vaizdo įrašus. Šio formato (Lietuvoje juo transliuojama skaitmeninė televizija) atkūrimo spartinimo funkcijos yra įtrauktos į daugelį naujausių vaizdo posistemių. „Silverlight 3“ taip pat turėtų gebėti išnaudoti vaizdo posistemių trimatės grafikos spartinimo funkcijas bei gausias duomenų susiejimo galimybes, leidžiančias išnaudoti „Visual Studio“ ir „Visual Web Developer Express“. galimybes. Anot S. Guthrie, „Silverlight“ naršykle leis daryti tai, ko negalima, pasitelkus AJAX ar „Flash“ [30].

Kompanija taip pat ketina imti tobulinti ir mažiau įdomias, tačiau ne mažiau reikalingas funkcijas, pavyzdžiui duomenų formas, reikiamas duomenų keitimuisi tarp programų.

„Microsoft“ kol kas nepraneša, kada ji ketina išleisti trečiąją „Silverlight“ versiją, tačiau manoma, kad ji pasirodys jau šiais metais. Anot S. Guthrie, „Silverlight“ jau yra įsidiegę šimtai milijonų vartotojų, tačiau neaišku ar tai yra kalbama apie visą dviejų metų periodą ir ar yra įskaičiuojamos bandomosios „Silverlight“ versijos. Pirmoji „Silverlight“ versija pasirodė 2007 metų rugsėjo mėnesį, tačiau bandomosios jau buvo siūlomos nuo balandžio. Norėdama paspartinti „Silverlight“ sistemų populiarinimą. Reikia pastebėti „Microsoft“ iš tiesų nori įtikti rinkai. Kompanija ne tik pateikė visą reikiamą informaciją apie „Silverlight“ sistemos pritaikymą alternatyvioms operacinėms sistemoms, nors pati ir nesiėmę kurti „Linux“ įskiepių versijų. Be to, „Linux“ vartotojui užsukus į „Microsoft“ svetainę ir spragtelėjus „Get Silverlight“ nuorodą, vartotojas yra nukreipiamas į svetainę, kuri pateikiamas alternatyvus „Moonlight“ įskiepis, užuot sausai pareiškus, kad vartotojo operacinė sistema nėra suderinama su „Microsoft“ produkcija.

Tiek „Microsoft“ išsamių internetinių programų („rich internet application“ - RIA), tiek „Adobe“ AIR padaliniai praneša, kad jų parengtus naršyklių įskiepius ir programas per 10 mėnesių nuo išleidimo atsisiuntė per 100 mln. vartotojų nuo šių sistemų starto 2008 metais [31].

Žemiau esančioje lentelėje pateikta informacija apie Silverlight palaikomas skirtingų versijų naršykles[24].

Lentelė 1 Silverlight palaikomos naršyklių versijos

OS/browser	IE 6 SP1	IE 6 SV1	IE 7	IE 8	Firefox	SeaMonkey	Safari	Konqueror	Opera	Google Chrome
Windows Vista/2008	N/A	N/A	1.0, 2.0, 3.0	1.0, 2.0, 3.0	1.0, 2.0, 3.0	1.0, 2.0	1.0, 2.0; via NPAPI	N/A	Unofficially[45][46]	2.0
Windows XP/2003/Home Server	N/A	1.0, 2.0, 3.0	1.0, 2.0, 3.0	1.0, 2.0, 3.0	1.0, 2.0, 3.0	N/A	1.0, 2.0; via NPAPI	N/A	Unofficially[45][46]	2.0
Windows 2000	2.0	N/A	N/A	N/A	2.0 Unofficially[47]	N/A	2.0; via NPAPI	N/A	Planned[45]	N/A
Windows Mobile 6	2.0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Mac OS 10.4/10.5 PowerPC	N/A	N/A	N/A	N/A	1.0	N/A	1.0	N/A	Planned[45]	N/A
Mac OS 10.4/10.5 Intel	N/A	N/A	N/A	N/A	1.0, 2.0	N/A	1.0, 2.0	N/A	Planned[45]	N/A

1.6.5 XAML – dizainerio ir programuotojo ateities darbo eiga

XAML (angl. Extensible Application Markup Language) – tai XML standartu apibrėžta kalba, ji sukurta Microsoft kompanijos ir yra naudojama įvairiems objektams ir jų reikšmėms struktūriškai apibrėžti.

Būtent XAML kalba ir yra pagrindas dizainerio ir programuotojo tarpusavio bendradarbiavimui. XAML, kaip atskiras technologijos vienetas šio bendradarbiavimo nesukurtų, tačiau ji yra pagrindas sistemos, paremtos Silverlight technologija, grafinės aplinkos kūrimui.

Expression Blend įrankis leidžia dizaineriams kurti raiškias vartotojų sąsajas, kurių formatas būtent ir yra XAML tipo. Šis įrankis taip pat leidžia kurti ir vartotojo sąsajos interaktyvumą ir paruošti prezentacinį sluoksnį kuriamai Silverlight programai. Programuotojas, kuris realizuoja sistemos logiką su Visual Studio įrankiu. Būtent šios logikos prezentacinis sluoksnis ir yra aprašomas XAML kalba. Dizaineris ir programuotojas gali pagaliau dirbti prie to pačio projekto vieningoje platformoje. Visual Studio ir Expression Blend naudoja tuos pačius projekto failus. Jeigu dizaineris dirbo prie sistemos ir programuotojui prireikia atlikti tam tikrus pakeitimus architektūroje, jis tai gali atlikti visai neliesdamas prezentacinio sluoksnio. Atvirkštinis variantas taip pat įmanomas, dizaineriai gali dirbti prie vartotojo sąsajos, nepaliečiant egzistuojančios sistemos architektūros. Tuo pačiu realizuojama ir integracija ir sąsajos bei logikos atskyrimas sistemoje [32].

1.6.6 Flash ir Silverlight

Po daugelio metų dominavimo, Adobe Flash (buvęs Macromedia flash) susilaukė iššūkio Raiškiųjų internetinių sistemų sferoje. Microsoft šiuo metu dirba prie trečios Silverlight versijos, raiškiosios medijos technologijų, kurios turi daug panašumų su Flash, ir pagaliau tampa pastebima sistemų kūrėjų. Flash technologija, buvusi pioniere Raiškiųjų internetinių sistemų srityje nuo 1996 metų [33], pagaliau susilaukė rimto konkurento. Nors Adobe Flash irgi nestovi vietoje. Nuo 9 Flash versijos pereinant

prie 10 flash versijos taip pat buvo realizuota daugybė patobulinimų, o taip pat aktyviai dirbama ir prie Flex projekto – tiesioginio konkurento Silverlight platformos teikiamom galimybėm.

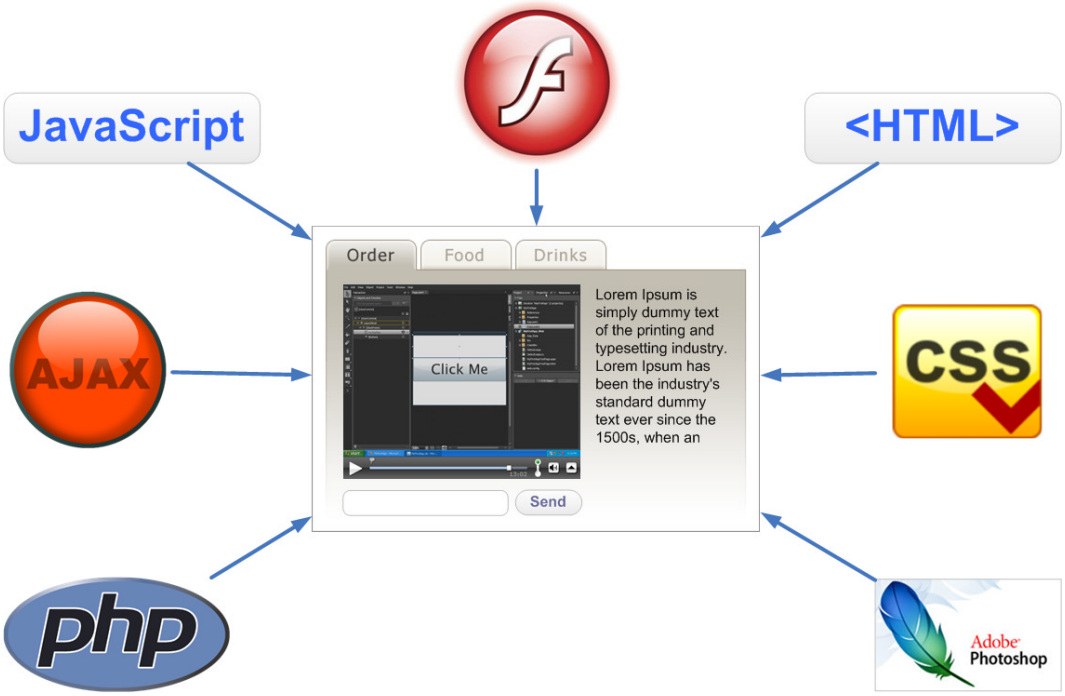
Silverlight palyginimas Flash atžvilgiu

- Silverlight kodas gali būti parašytas bet kuria .NET kalba (pavyzdžiui, C#, VB, Ruby, Python (tiesa, Ruby ir Python eina per DLR) ir kt.) - vadinasi greičiausiai nereikės mokintis papildomos – ActionScript, kurios panaudojimo sritis siauresnė nei .NET kalbų (čia aktualu tuo atveju, kai nemoki nei vienos programavimo kalbos;
- Iki ActionScript 3.0 kodas buvo sunkiai atskiriamas nuo dizaino;
- Daug geresnė aplinka programavimui – daug protingesnis kodo redaktorius, daug didesnės klaidų ieškojimo (debug) galimybės;
- CLR palaikomų kalbų galimybės daug didesnės nei ActionScript;
- Silverlight palaiko pilno ekrano rėžimą;
- Silverlight UI yra tekstiniam formate – gerokai lengviau redaguoti/pakartotinai naudoti fragmentus ir panašiai;
- Flash nesinchronizuoja kadrų su laiku (nebent naudojami tušti MP3), todėl 3 sekundžių animacija priklausomai nuo kompiuterio, jo užkrautumo ir pan. gali tęstis nuo 2 sekundžių iki 6 ir panašiai;
- Video formatai – Flash (paskutinis) naudoja Sorenson's H.263 uždarą formatą, taip pat didžioji dalis naudojamų garso suspaudimo formatų taip pat uždari (išskyrus MP3 ir ADPCM, kurio niekas nenaudoja dėl prastų suspaudimo rezultatų), tuo tarpu Silverlight naudoja industrinio standarto VC-1 video formatą, taip pat palaiko WMV ir WMA, kuriems yra nemokamas Microsoft paruoštas SDK, taigi galima suspausti šiais formatais savo turimą informaciją paties pasirašytomis priemonėmis (automatiškai dėl atvirumo ir kitų prirašytų priemonių daugiau);
- Galima nesunkiai panaudoti taikomųjų aplikacijų kodo klases Silverlight programose, kas nelabai įmanoma Flash atveju, nes paprasčiausiai niekas nerašo taikomųjų aplikacijų su ActionScript;
- Labai nedaug yra tikrų ActionScript programuotojų, kurie supranta ką daro, o ne tik kopijuoja įklijuoja kodo gabaliukus iš interneto, todėl sudėtingesnei problemai gauti sprendimą forumuose yra pakankamai sudėtinga (ypač, jeigu kalbam apie internetines paslaugas (webservices), duomenų bazes) [34];
- Atviras kodas. Taip taip, Microsoft ir atviras kodas. Jau dabar kas netingi gali dekompiliuoti .NET bibliotekas (kam neteko dekompiliuoti .NET bibliotekų ir kas mano, kad galima dekompiliuoti ir Adobe kodą – tai du skirtingi dalykai, nes .NET dekompiliuotą kodą žmogus gali lengvai suvokti ir skaityti, žinoma jeigu nebuvo taikomos apsauginės priemonės (.NET karkaso atveju jos taikomos nebuvo)), o kartu su Visual Studio 2008 bus standartiškai duodamas .NET karkaso išeities kodas, todėl klaidų ieškojimo (debugging) procesas bus dar paprastesnis.

1.7 Raiškiųjų internetinių sistemų kūrimo sudėtingumas

Tobulėjant technologijoms tobulėja ir pačios sistemos. Naujai kuriamom sistemom vis dažniau pritaikomi Web 2.0 ir Raiškiųjų interneto sistemų kūrimo principai. Tačiau vis sudėtingesniu uždaviniu tampa šiuos principus realizuoti nuo seno naudojamomis technologijomis. Šiame projekte buvo naudotos technologijos, kurios puikiai dera tarpusavyje – tai: HTML, CSS, JavaScript, AJAX, PHP ir MySQL. Tačiau nepaisant jų suderinamumo, kūrimo procesai, kuriais bandoma sukurti Raiškiąsias internetines sistemas, vis sudėtingėja. Susidaro daug kodo eilučių, naudojama vis daugiau įskiepių ir pagalbinių klasių, kodas tampa sunkiai skaitomas ir dar sunkiau redaguojamas, dažnai vienas pakeitimas iššaukia daugybę naujų klaidų. Norint sukurti išraiškingas ir patrauklias sistemas reikia visas šias technologijas apjungti į vieną visumą. Kūrimo, o tuo labiau palaikymo, procesai sudėtingėja. Patrauklių sistemų realizavimas aukščiau minėtomis technologijomis tampa labai sudėtingas.

Vien tik norint sukurti gražiai atrodantį kortelėmis (angl. tabs) valdomą elementą, prireikia daugybės technologijų bei įrankių, o tai pat ir savo srities specialistų. Žemiau būtent ir pateiktas paveiksliukas iliustruoja kortelėmis valdomą elementą internetinėje sistemoje bei technologijas, kurios buvo pasitelktos jam sukurti ir įgyvendinti.



Pav. 6 Raiškiųjų internetinių sistemų kūrimas senomis technologijomis

.

2 Projektinė dalis

Renginių paskelbimo ir pakvietimų į renginius organizavimo ir valdymo sistema.

Pagrindinė problema, kuriai išspręsti skirtas šis projektas, yra renginio paskelbimas svečiams, pakvietimų išsiuntinėjimas į renginį kviečiamiems asmenims, bei svečių registravimas renginyje. Kvietimų išsiuntinėjimas turi būti supaprastintas iki minimumo. Dalyvių reakcijos užfiksavimas, neturėtų kelti jokių sunkumų kvietimą gavusiam asmeniui, net jei jo kompiuterinis raštingumas yra žemas. Kvietimą gavęs asmuo turėtų turėti galimybę pranešti organizatoriui apie dalyvavimo statusą, net nesinaudodamas kompiuteriu. Sistema turi būti patogi tiek kviečiamam asmeniui (kuris yra perspektyvus sistemos naudotojas kaip organizatorius), tiek ir renginių organizatoriui. Tai turėtų pasireikšti patogia ir aiškia vartotojo sąsaja, neįkyriu dizainu, neperkrautu informacijos kiekiu.

Ši sistema skirta organizatoriams, norintiems padidinti lankytojų skaičių ir parduoti daugiau bilietų.

Sistema siūlo lengvą ir paprastą renginių registraciją, bilietų pardavimą ir apmokėjimų tvarkymą, prieinamą tiesiog per internetą.

Sistemos privalumai pašalina administravimo sunkumus, kurie yra susiję su dalyvių registravimu ir tvarkymu. Sistema padeda automatiškai surinkti, rikiuoti ir tvarkyti visą renginio informaciją internete.

Organizatorius gali tiekti tiesioginę registraciją visiems savo svečiams. Su šia sistema tai galima padaryti per kelias minutes, kas užtruktų tikrai labai ilgai, jeigu norėtum sukurti prezentacinį renginio puslapį su registracijos ir apmokėjimų valdymu, realaus laiko duomenų surinkimu, ir tai kainuotų labai daug.

Renginį galima sukurti per tris žingsnius: 1 - užsiregistruoti sistemoje, 2 - užpildyti informaciją apie renginį ir paruošti pakvietimus, 3 - sudaryti svečių sąrašą. Įvykdžius šiuos tris žingsnius, belieka paskelbti renginį.

Sistemoje galima matyti įvairiapusiškas ataskaitas, jas tvarkyti ir pritaikyti savo poreikiams.

Informacinė visuomenė vis didėja ir plečiasi, internetinės paslaugos tampa kasdienybe, žmonės supranta, kad internetu problemas galima išspręsti daug greičiau ir paprasčiau.

2.1 Pagrindiniai sistemos veiklos įvykiai

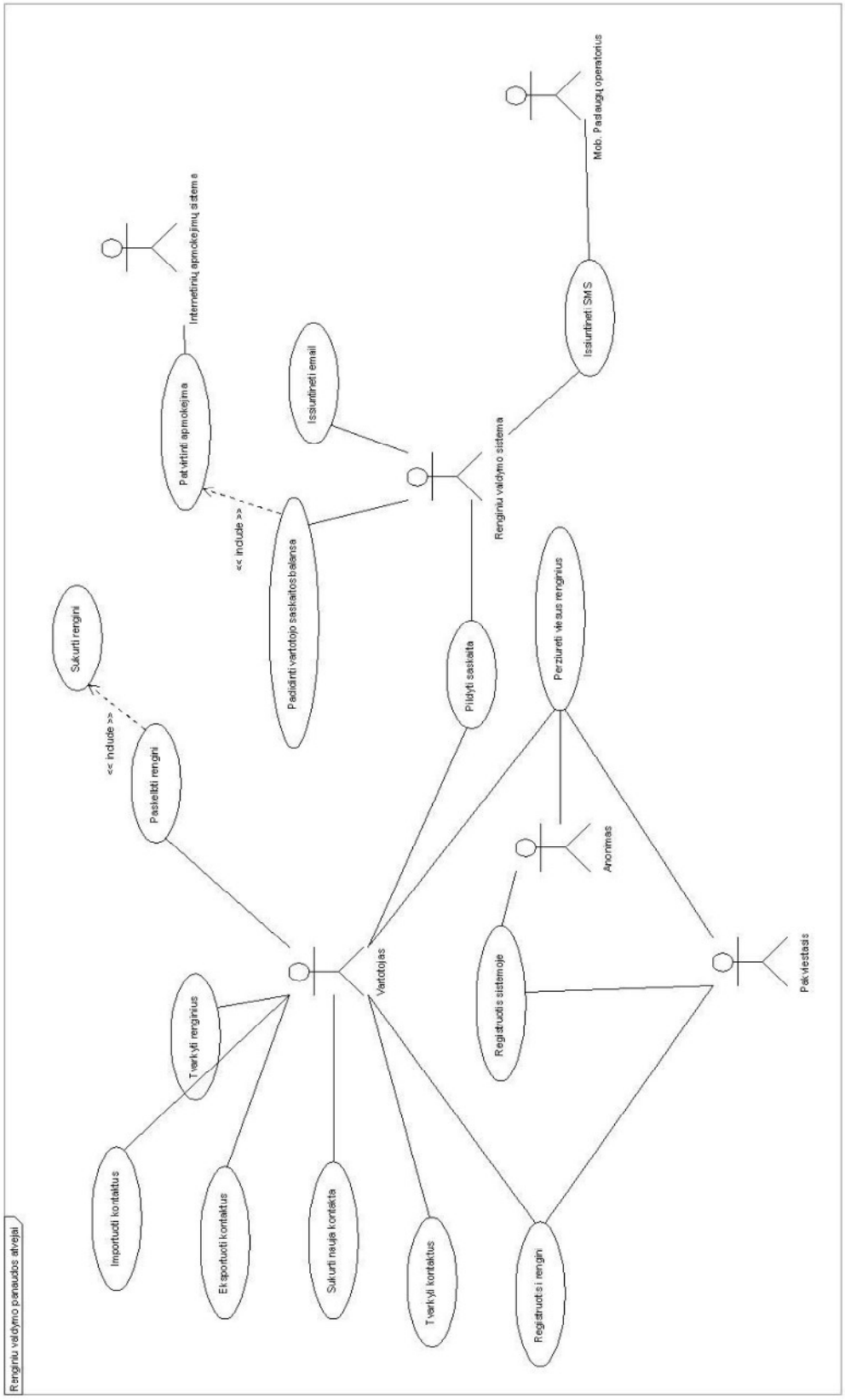
Žemiau yra pateiktas pagrindinių sistemos veiklos įvykių sąrašas, bendaram sistemos funkcionalumui apibrėžti.

Įvykio nr.	Įvykio pavadinimas	Įeinantys/išeinantys duomenys	Aprašymas
1	Vartotojas sukuria/redaguoja renginį	Renginio duomenys(Į)	Išsaugomas vartotojo sukurtas renginys
2	Lankytojai peržiūri renginius	Data(Į) Renginiai(Iš)	Pasirinkus datą lankytojas gali matyti renginius
3	Formuojamas ir siunčiamas pakvietimas el. paštu	Renginys(Į) El. Laiškas su renginio info(Iš)	Pagal renginio duomenis suformuojamas ir išsiunčiamas el. laiškas
4	Formuojamas ir siunčiamas pakvietimas sms‘u	Renginys(Į) sms su renginio kodu(Iš)	Pagal renginio duomenis suformuojama ir išsiunčiama sms žinutė
5	Gaunamas pranešimas apie sms būklę	Būklė(Į)	Serveris užfiksuoja sms būklę
6	Formuojamas mokėjimas, sąskaitos papildymui	Mokėjimo duomenys(Į)	Suformuotas mokėjimas siunčiamas į mokėjimai.lt
7	Gaunamas pranešimas apie mokėjimo būklę, sąskaitos papildymas	Mokėjimo būklė(Į)	Serveris užfiksuoja mokėjimo būklę, papildoma sąskaita
8	Pakviestasis peržiūri renginį su pakvietimo kodu	Pakvietimo kodas(Į) Renginio informacija(Iš)	Surandamas renginys, kurio pakvietimo kodas buvo įvestas
9	Pakviestojo atsakymo fiksavimas	Atsakymas(Į)	Užfiksuojamas pakviestojo atsakymas
10	Vartotojas sukuria/valdo kontaktus	Kontakto duomenys(Į)	Sukuriamas/pakeičiamas kontaktas su nurodytais duomenimis
11	Siunčiama sms žinutė kontaktui	Žinutės tekstas(Į) Kontaktas(Į) Atsakymas(Į)	Nurodytam kontaktui siunčiama žinutė, ir parodomas pranešimas apie pristatymo būklę
12	Administratorius valdo vartotojus	Vartotojas(Į)	Kuriami/redaguojami/trinami sistemos vartotojai
13	Vartotojas keičia savo duomenis	Duomenys(Į)	Sistema išsaugo naujus vartotojo duomenis
14	El.pašto serveris siunčia pakvietimą	Pakvietimas(Į)	Siunčiamas pakvietimas el. paštu nurodytam žmogui
15	Gsms.lt siunčia pakvietimą žmogui	Pakvietimas(Į)	Siunčiamas pakvietimas sms žinute nurodytam žmogui
16	Naudotojas peržiūri išsiųstų sms būklę	Naudotojas(Į) Žinučių būklė(Iš)	Sistema išveda nurodyto naudotojo išsiųstų sms žinučių būklę
17	Naudotojas peržiūri atliktų mokėjimų būklę	Naudotojas(Į) Mokėjimų būklė(Iš)	Sistema išveda nurodyto naudotojo atliktų mokėjimų būklę
18	Vartotojas peržiūri savo renginių istoriją	Naudotojas(Į) Renginių sąrašas(Iš)	Sistema išveda nurodyto naudotojo renginius, į kuriuos buvo kviečiamas, sąrašą
19	Naujo vartotojo užregistravimas	Vartotojo duomenys(Į)	Sistemoje užregistruojamas naujas vartotojas

Lentelė 2 Veiklos įvykių sąrašas

2.2 Sistemos panaudos atvejai

Pagrindiniams vartotojų veiksmams apibrėžti, žemiau yra pateikta sistemos panaudos atvejų diagrama.

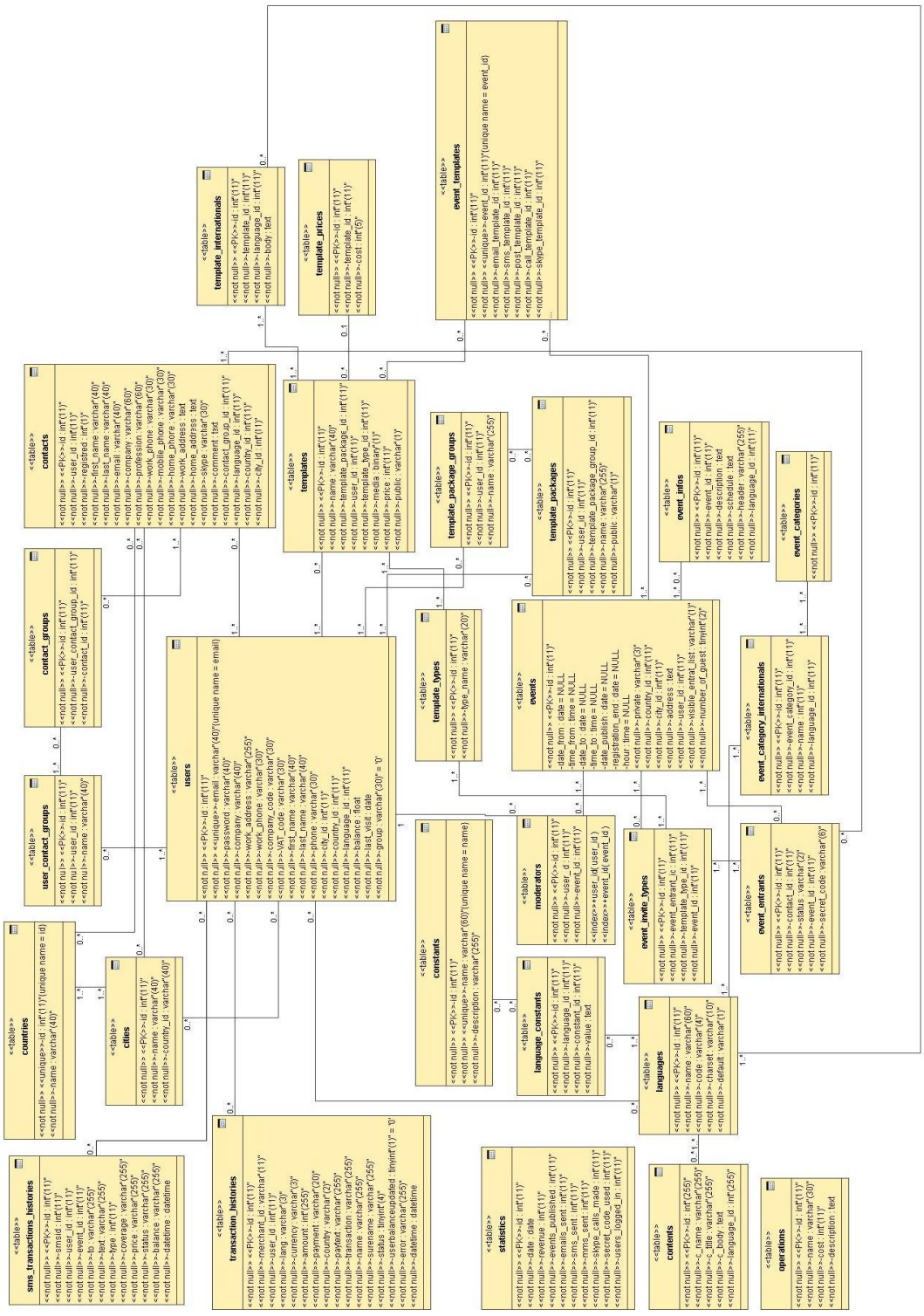


Pav. 7 Panaudos atvejų diagrama

2.3 Reikalavimai duomenims

Visi sistemos duomenys ir rezultatai yra saugomi MySQL duomenų bazės valdymo sistemoje (tik laikiniems duomenims saugoti yra naudojami atskiri failai). Dalis duomenų bazės lentelių yra sistemos lentelės, kitos lentelės yra sukurtos siekiant išgauti reikalingą funkcionalumą pagal užsakovo poreikius.

Duomenų bazės loginę schemą pateikiame Pav. 8 Duomenų bazės schema.



Pav. 8 Duomenų bazės schema

2.4 Nefunkciniai reikalavimai

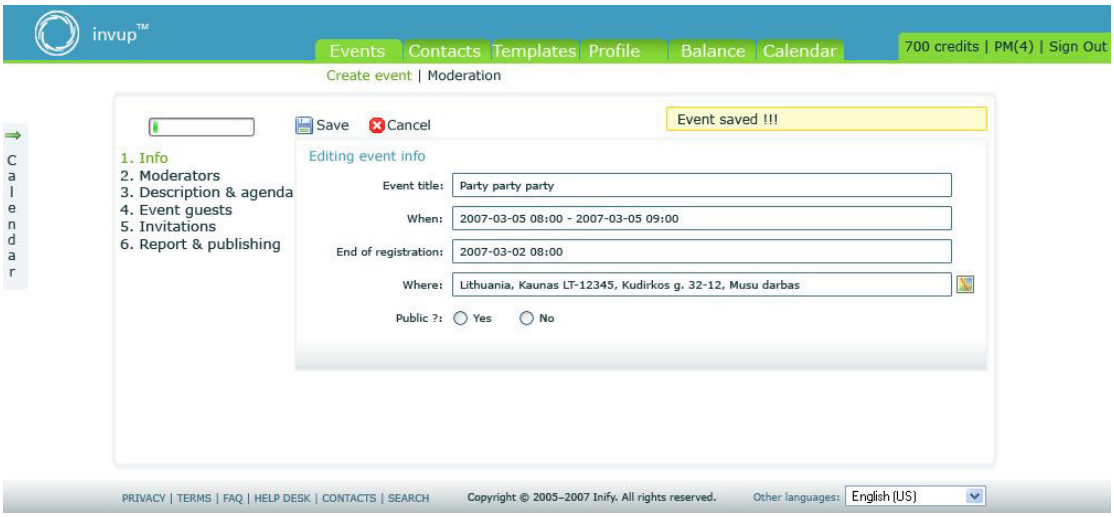
2.4.1 Sistemos išvaizda

Produktas bus orientuotas į vyresniąją vartotojų kategoriją. Sistemos išvaizda turi būti solidi, patraukli akiai, tačiau ne per daug spalvota ir neapkrauta. Iš pirmo žvilgsnio sistemos išvaizda turi sudaryti pasitikėjimo ir ramumo jausmą. Išvaizdai keliamas pagrindinis reikalavimas, kad ji atrodytų paprasta ir elementari, lengva naudotis, tačiau tuo pačiu turi netrūkti komponentų, palaikančių visišką sistemos funkcionalumą. Taip pat išvaizda turi sudaryti ir privatumo bei konfidencialumo jausmą, kad vartotojas, tik išvydęs dizainą, nesusidarytų pigaus ir nepatikimo puslapio įspūdžio. Tačiau to vien tik nepakanka, kad pavyktų pritraukti naujų vartotojų dėmesį, privalėtų būti ir šiokia tokia intriga, kad vartotojas būtų susidomėjęs užsiregistruoti renginių organizavimo sistemoje.

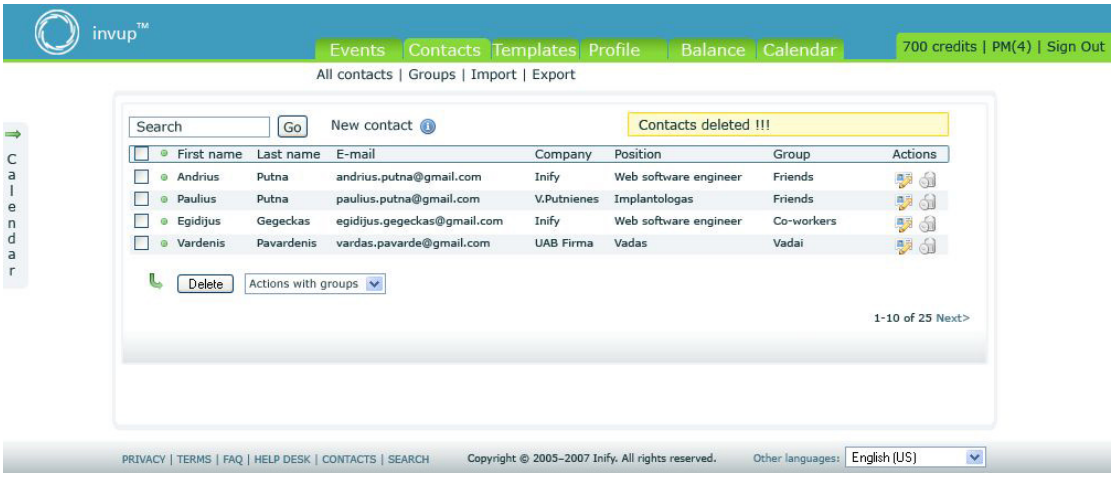
Kadangi vartotojo sąsaja yra vienas iš svarbiausių programos komponentų, todėl labai svarbu, kad ji pasižymėtų šiomis savybėmis:

- Patogumu;
- Lengva navigacija;
- Paprastumu;
- Neapkrauta aplinka;
- Lengvu įsisavinimu.

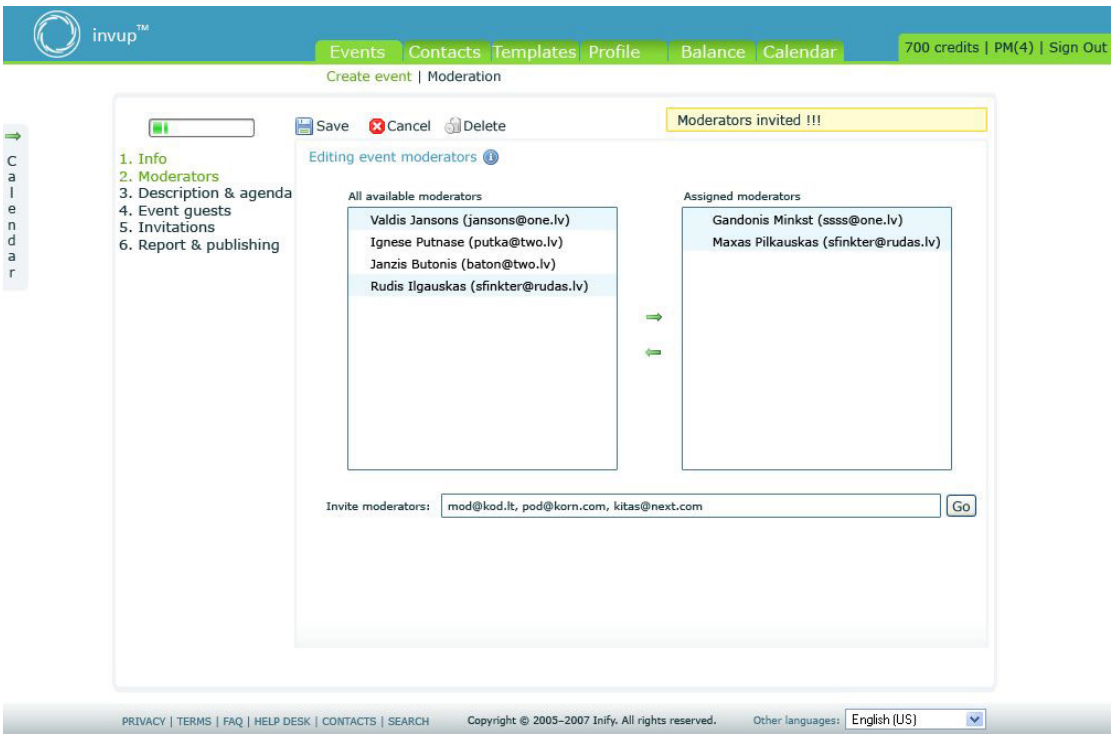
Žemiau yra pateikta keletas paveiksliukų, kurie atspindi kurtą ir planuotą sistemos aplinkos dizainą:



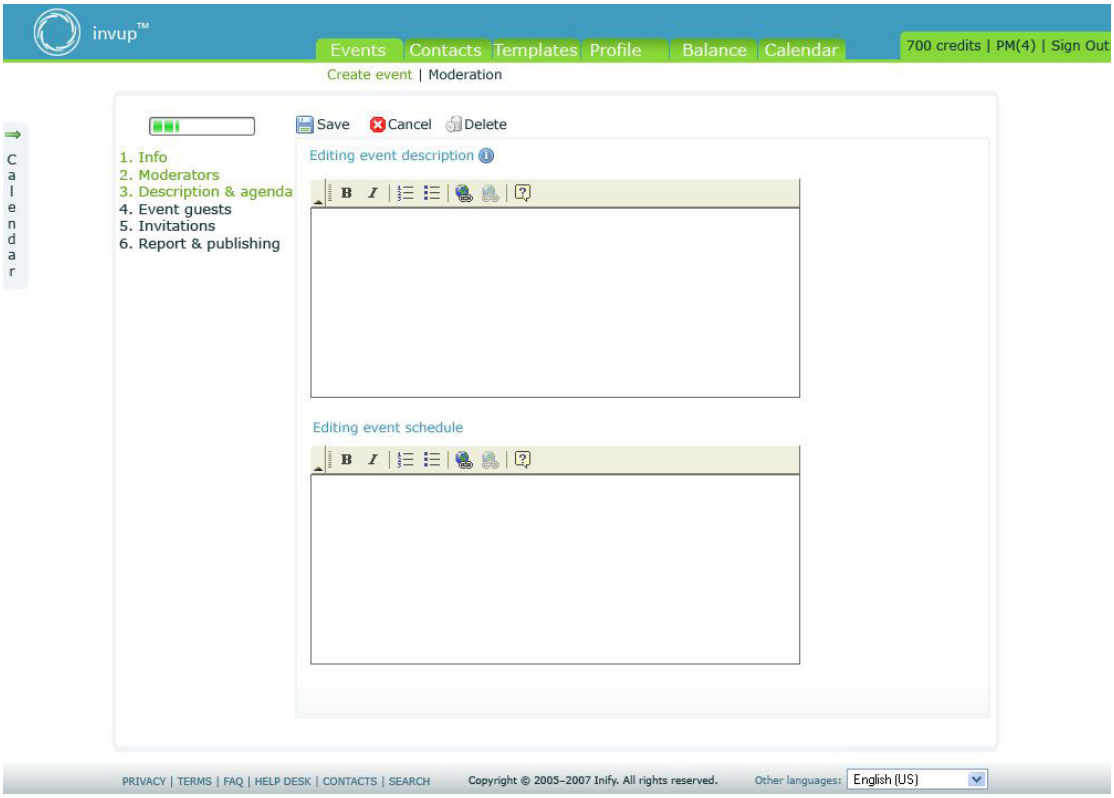
pav. 9 Renginių redagavimo ir sukūrimo langas



pav. 10 Sistemos atvaizduojamų duomenų sąrašas



pav. 11 Moderatorių arba kontaktų priskyrimas



pav. 12 Tekstiniai redagavimo laukai

2.4.2 Panaudojamumas ir prieinamumas

Sistema turi būti pasiekiama visiems interneto vartotojams (įskaitant ir neįgaliuosius ir sveikuosius), nepaisant kokią naršymo technologiją jie naudoja. Visiems prieinamas tinklalapis gali turėti didelės naudos tiek pačiam tinklalapiui, tiek ir verslui.

Internetinė svetainė turi funkcionuoti su skirtingomis naršymo technologijomis. Pirma ir viena iš svarbiausių prieinamumo taisyklių yra, kad ne visi vartotojai naudoja naujausios versijos Internet Explorer naršyklę, su visais įmanomais atnaujinimais ir papildiniais. Kaip pavyzdžiui, galima pateikti keletą skirtingo tipo interneto naršymo technologijų:

- Lynx naršyklė – tai yra tik tekstinė naršyklė, kuri nepalaiko lentelių, css, paveiksliukų, JavaScript, Flash, audio ar video turinio.
- Naršymas per televizorių – 560 taškų pločio ir nėra horizontalios slintkies juostos.
- Ekrano skaitytojas – tai programa, balsu skaitanti puslapio turinį tokia tvarka, kaip jis atsiranda HTML dokumente
- Delninukas – labai mažas ekranas su ribotomis JavaScript ir didelių paveiksliukų galimybėmis.
- Ekrano didintuvas – vienu metu gali būti rodomi tik 3 – 4 žodžiai.
- Lėtas interneto ryšys – vartotojai gali išjungti paveiksliukus, Flash technologijas

- Plačiaekraniai vaizduokliai.

Visuomet turi būti alternatyvos paveiksliukams, JavaScript, Flash technologijoms, audio ir video turiniui. Taip pat atkreipti dėmesį ir į tai kaip svetainė atrodo be CSS. Taip pat ir užpildymo formos turi būti prieinamos visiems interneto vartotojams. Visiems vartotojams turėtų būti lengva ir greita pasiekti turinį tinklalapyje. Patartina naudoti antraštes, nuorodas, paryškinimus, numeravimus, taip pat konkrečius aprašymus, kaip pavyzdžiai: niekada nepavadinti nuorodos – „spausti čia".

Patartina atskirti puslapio struktūrą nuo pateikimo. Struktūra nusako kaip dokumentas yra išdėstytas, dažniausiai naršymo meniu elementai, antraštės, pastraipos, sąrašai ir nuorodos. Puslapio pateikimas nusako kaip visa tai yra pateikta vartotojui. Pagrindinis principas yra naudoti CSS, o ne lentelių tipo išdėstymą. Taip pat patartina pasitelkti į pagalbą ir interaktyvumą, kaip pavyzdžiui: leisti vartotojams keisti teksto dydį, perspėti juos, kai nuoroda atsidarys kitame naršyklės lange ir pan.

Aukšto prieinamumo web paslaugos gali būti pasiektos, kuriant plataus naudojimo sistemas. Tačiau tokias sistemas sukurti yra brangu, ir tai gali įtakoti mažesnes kompanijas atsisakyti šių reikalavimų.

2.4.3 Našumas

Web technologijų sparta yra matuojama pralaidumu, gaišties laiku, vykdymo laiku ir operacijos laiku. Pralaidumas yra skaičius, parodantis pateiktų užklausų skaičių per tam tikrą laiką. Gaištis yra laikas, per kurį yra nusiunčiama užklausa ir gaunamas atsakas. Vykdymo laikas yra laikas, skirtas sistemai atlikti visą jai skirtą veiklų seką. Operacijos laikas parodo Web sistemos užtrunkamą laiką, per kurį yra įvykdoma viena pilna operacija. Kuo didesnis pralaidumas, mažesnis gaišties, vykdymo laikas ir kuo greitesnis operacijos laikas, tuo web sistema greičiau veikia.

Visa Web sistemos sparta priklauso nuo programos logikos, tinklo ir labiausiai nuo esančių viduje žinučių siuntinėjimo ir duomenų perdavimo protokolų, tokių kaip SOAP ir HTTP.

Nepaisant SOAP, reikia atkreipti dėmesį ir į XML apdorojimo žingsnius. Keli iš būdų, kaip optimizuoti XML apdorojimo žingsnius:

- Naudoti veiksmingus ir lengvus analizatorius
- Efektyvus XML validavimo naudojimas kūrimo stadijoje
- Suspausto XML naudojimas, siunčiant žinutes per tinklą
- Web paslaugų padėjimas į podelį
- Kiek galima paprastesnių duomenų tipų naudojimas SOAP žinutėse.

2.4.4 Veikimo reikalavimai

Patikimumas yra vienas iš svarbiausių kriterijų, vertinant Web sistemų kokybę. Skaičius sutrikusių funkcionavimų per dieną, per savaitę ar per metus yra svarbiausias dydis, parodantis sistemos patikimumą. Šiuo metu Web technologijos pasikliauja tokiais duomenų perdavimo protokolais, kaip HTTP. Tačiau tai dar negarantuoja, kad žinutė bus pristatyta į vietą. Vienas iš būdų sumažinti riziką yra asinchroninės žinučių eilės, kitas būdas būtų paveldėjimas naujų patikimų duomenų perdavimo protokolų, tokių kaip HTTPR, REST, BEEP.

Patikimumas apibrėžia ar sistema bus prieinama tam tikru laiko momentu. Su patikimumu yra susijusi TTR (angl. Time-to-Repair) technologija. TTR atstovauja laiką, kuris bus sugaištas, tvarkant tam tikrą Web paslaugą.

Kurti web paslaugas, kurios net ir įvykus tam tikrai klaidai veiktų toliau, yra brangu. Kompanijos, pasirinkusios šį kelią, susiduria su skirtingose vietose išdalintų technologijų problemomis. Kyla tokie klausimai, kaip:

- Ar tik ne vienas iš pagrindinių serverių tapo nepasiekimas?
- Ar sistema netapo per daug apkrauta?
- Kodėl užklausos trunka taip ilgai?

Dauguma renkasi multifunkcinį web paslaugų valdymą, web servisų „klasteriavimą“.

2.4.5 Palaikomumas

Kol kuri ir programuoji web sistemą, struktūra, kurią naudoji, puslapio išdėstymas ir viskas, kas susiję tarpusavyje tampa gerai žinoma ir pažįstama. Tačiau praeina tam tikras laiko tarpas, pridedi vienur šiek tiek, pridedi kitur, ir po kurio laiko sistema tampa didžiulė. Po kurio laiko atmintis išblėsta ir ilgainiui tampa nebeaišku, kodėl vienas dalykas yra čia, o ne ten, kam yra skirtas šitas principas. Dabar, jeigu norėsi ką nors pakeisti, yra didelė galimybė, kad keičiant vieną dalyką, gali nustoti funkcionavę kiti. Kad to pavyktų išvengti, reikia būti nuolatos pastoviam ir naudoti vieną pasirinktą metodologiją ir principą, nedaryti tų pačių dalykų skirtingais būdais. Verta naudoti komentarus. Yra blogai manoma, kad komentarai skirti tik keistuoliams, kad jie galėtų suprasti ar net pasisavinti jūsų metodus. Nes labiausiai tikėtina, kad tas keistuolis jūs pats ir būsite, kuris mokinsis iš komentarų.

Patartina naudoti pastovią tiek pačio tinklapio, tiek pačios sistemos projektavimo struktūrą. Vienas iš būdų įvesti sistemingą struktūrą yra MVC (modelis-vaizdas-kontroleris) naudojimas. Tai tam tikra loginė bei praktinė metodologija, atskirianti veiksmus su duomenų bazėmis, išskirianti sistemos objektų klases, skirtas kontroliuoti sistemos vaizdą ir duomenų srautus.

2.4.6 Saugumas

Didėjant web technologijų paklausai, kurios yra prieinamos per viešą internetą, auga susirūpinimas dėl saugumo. Web kūrėjai vis dažniau susiduria su nuolatos didėjančiu spragų kiekiu, kurios tik dar labiau apsunkina darbą. Web paslaugų saugumas reiškia konfidencialių duomenų perdavimą, tik įgaliotiems asmenims, žinučių šifravimą, ir priėjimo kontrolės tiekimą. Web sistemoje gali būti skirtingų lygių konfidenciali informacija, kurią yra būtina apsaugoti. Su saugumu susijusios problemos negali būti paliktos nuošalyje, sistemos saugumas privalo užtikrinti vartotojų pasitikėjimą.

Vienas iš didžiausių rizikos šaltinių organizacijoms yra Web serveriai. Įsilaužimas į serverius gali padaryti labai didelių problemų organizacijoms. Šias pažeidžiamas spragas galima suskirstyti į keletą kategorijų:

Standartinė konfigūracija. Web serveriai yra dažnai įdiegiami tiesiog su standartine konfigūracija, kuri gali būti nesaugi. Pažeidžiamos vietos gali būti standartiniai pavyzdžiai, šablonai, administravimo įrankiai, nuspėjamos paslaugų programų vietos. Be atitinkamo rizikos apsisaugojimo tai gali nuvesti į keletą skirtingo tipo įsilaužimų, po kurių piktavaliai vartotojai gali perimti pilną Web serverio valdymą

Vartotojo įvedamų duomenų tikrinimas. Internetinės svetainės turi būti interaktyvios, kad būtų naudingos vartotojui. Neapsisaugojus nuo vartotojo įvedamų duomenų, gali kilti daugybė saugumo problemų, kuriomis gali pasinaudoti piktavalis vartotojas ir įsilaužti į Web serverį bei prieiti prie duomenų, esančių duomenų bazėse.

Duomenų užšifravimas. Kad ir kaip bebūtų gaila, nors ir dabartiniai šifravimo algoritmai virtualiai yra nenulaužiami, tačiau jie nėra pakankamai naudojami. Taip yra dėl didelio kainos ir kokybės skirtumo tarp procesorių ir specializuotų šifravimo greitintuvų. Beveik visas šiuo metu tinklo srautas keliauja nešifruotas ir gali būti lengvai perimtas piktavalių vartotojų

Duomenų laikymo apsaugojimas. Dauguma konfidencialių failų yra saugomi viešai prieinamuose serveriuose, vietoj to, kad būtų talpinami specialiai apsaugotuose serveriuose. Kai kurios programos skiria per mažai dėmesio laikinų failų išvalymui, palikdamos svarbią informaciją lengvai prieinamą.

Sesijos valdymas. Dauguma svetainių per mažai dėmesio skiria unikalių sesijų valdymui. Tai gali būti ir naudojami silpni autentifikavimo metodai, prastas sistemos sausainiukų (angl. cookies) valdymas, nesugebėjimas nustatyti sesijos neveiksnumo laiko ir panašios sesijų silpnybės. Tai dažnai veda į sesijų užgrobimus, kur užgrobėjas gali perimti visus vartotojo duomenis.

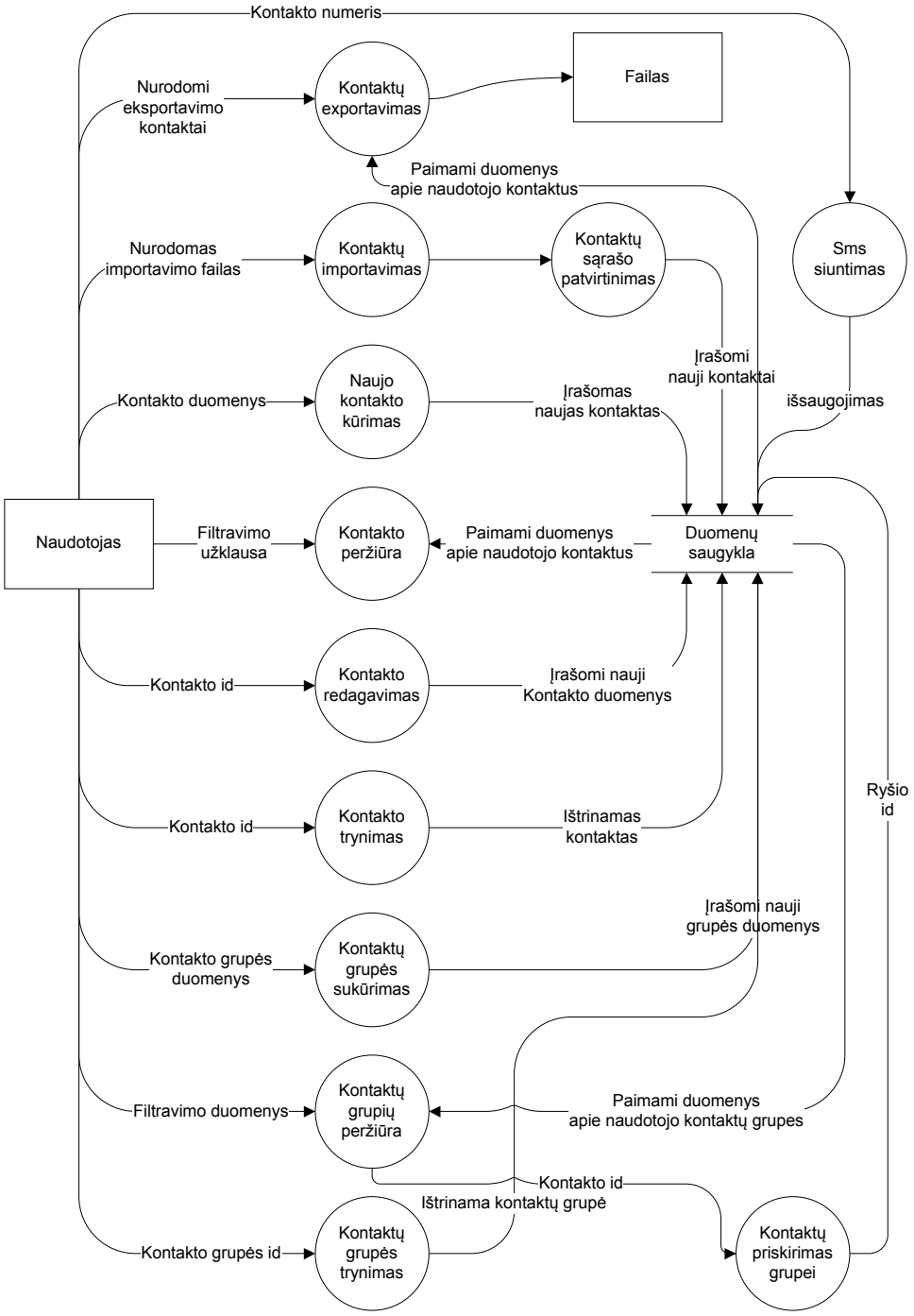
Dauguma šių svarbių problemų ir yra pagrindinės saugumo spragos Web sistemoje. Organizacijoms, kurioms saugumas yra labai svarbus faktorius privalo užtikrinti sistemos saugumo palaikymą ne tik sistemos kūrimo metu, tačiau ir jos veikimo bei plėtojimo metu.

2.5 Projektavimo etapų planavimas

Projektavimo etapus suskirstėme į kuriamos sistemos modulius. Pateikiame kiekvieno modulio aprašymą bei srautų diagramą.

2.5.1 Kontaktų valdymo modulis

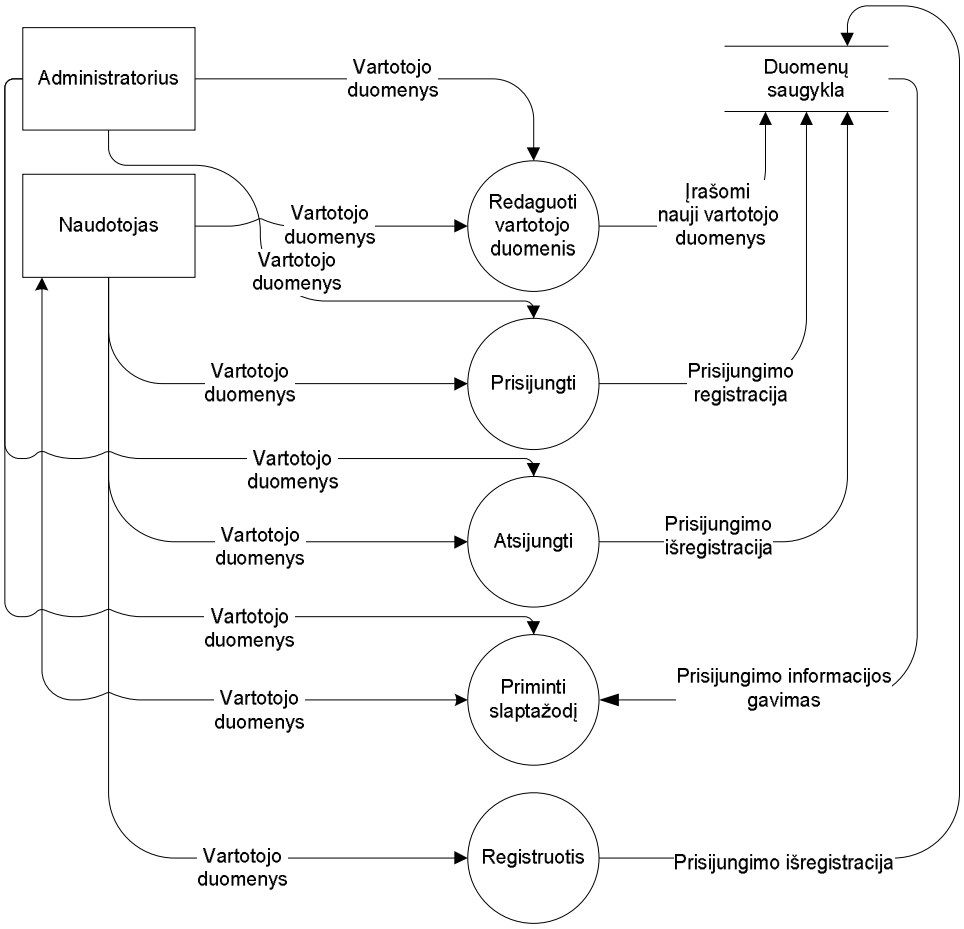
Kontaktų modulį valdo naudotojas. Proceso „Kontaktų eksportavimas“ metu naudotojas nurodo „eksportuojamų kontaktų sąrašą“ sistema iš duomenų saugyklos paima reikalingus duomenis ir naudotojui pasiūloma išsisaugoti failą. Proceso „Kontaktų importavimas“ metu perduodamas „Naudotojo importavimo failas“, suformuojamas sąrašas kurį reikia patvirtinti, po to patvirtintas sąrašas įrašomas į duomenų saugyklą. Proceso „Naujo kontakto sukūrimas“ metu perduodami „Kontaktų duomenys“ ir įrašomi į duomenų saugyklą. Proceso „Kontakto peržiūra“ metu paimami duomenys iš duomenų saugyklos ir atvaizduojami naudotojui. Proceso „Kontakto redagavimas“ metu perduodami „Nauji kontakto duomenys“ ir atnaujinami duomenų saugykloje. Detalūs duomenų srautai yra pateikti žemiau esančioje diagramoje.



Pav. 13 Srautų diagrama: Kontaktų modulis

2.5.2 Vartotojo modulis

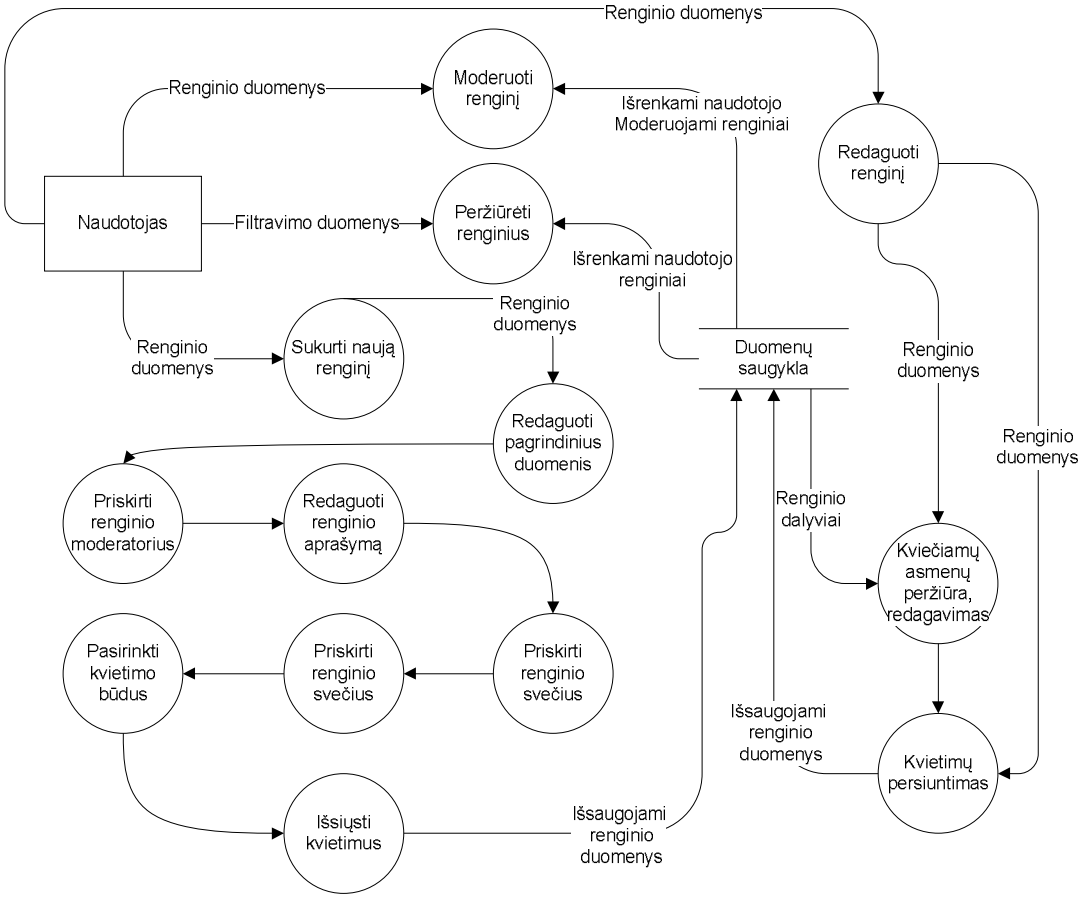
Modulis valdomas ir vartotojo ir administratoriaus. Proceso „Registruotis“ metu perduodami „registracijos duomenys“ ir įrašomi į duomenų saugyklą. Proceso „Prisijungti“ metu perduodami „prisijungimo duomenys“ ir įrašomi į duomenų saugyklą. Proceso „Atsijungti“ metu perduodami „vartotojo duomenys“ ir įrašomi į duomenų saugyklą. Proceso „Priminti slaptažodį“ metu perduodami „Vartotojo duomenys“ paimami duomenys iš duomenų saugyklos ir nusiunčiami vartotojui. Proceso „Redaguoti vartotojo duomenis“ metu perduodami „Nauji vartotojo duomenys“ ir įrašomi į duomenų saugyklą. Detalūs duomenų srautai yra pateikti žemiau esančioje diagramoje.



Pav. 14 Srautų diagrama: Vartotojo modulis

2.5.3 Renginių paskelbimo modulis

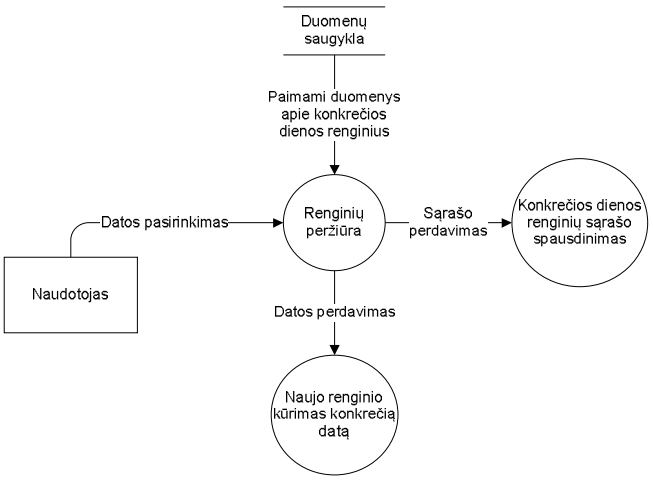
Renginių modulį valdo vartotojas. Proceso „Peržiūrėti renginius“ metu paimami duomenys iš duomenų saugyklos ir atvaizduojami vartotojui. Proceso „Sukurti renginį“ metu perduodami „Nauji renginio duomenys “ ir įrašomi į duomenų saugyklą. Proceso „Moderuoti renginį“ metu paimami duomenys iš duomenų saugyklos ir pateikiami moderatoriui. Detalūs duomenų srautai yra pateikti žemiau esančioje diagramoje.



Pav. 15 Srautų diagrama: Renginių modulis

2.5.4 Kalendoriaus modulis

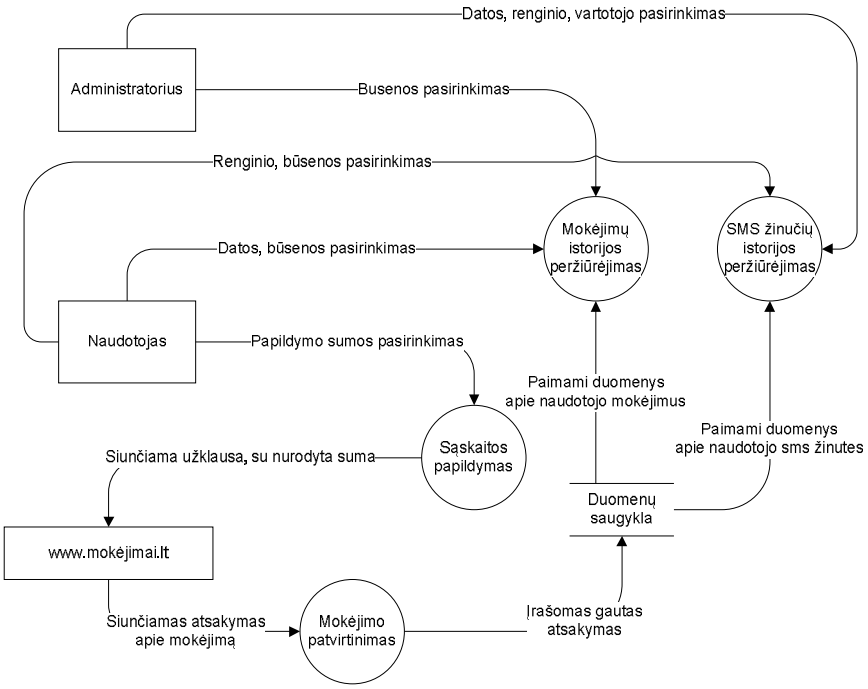
Valdo vartotojas. Proceso „Renginių peržiūra“ metu perduodami „Datos duomenys“ ir iš duomenų saugyklos atrenkami reikalingi duomenys. Detalūs duomenų srautai yra pateikti žemiau esančioje diagramoje.



Pav. 16 Srautų diagrama: Kalendoriaus modulis

2.5.5 Vartotojo sąskaitos valdymo modulis

Valdo administratorius ir vartotojas. Proceso „Sąskaitos papildymas“ metu vartotojas nurodo sumą pinigų už kurią jis nori pasipildyti sąskaitą, duomenys apdorojami www.mokėjimai.lt sistemoje ir gražinti duomenys įrašomi i duomenų saugyklą. Detalūs duomenų srautai yra pateikti žemiau esančioje diagramoje.

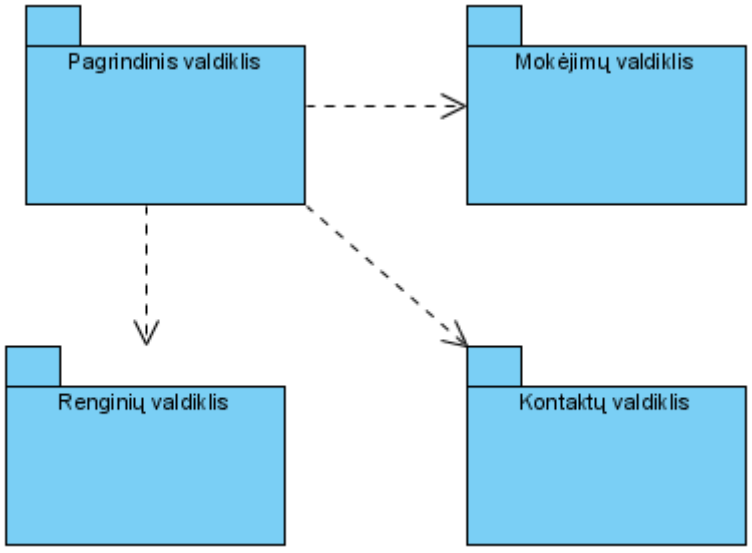


Pav. 17 Srautų diagrama: Vartotojo sąskaitos modulis

2.6 Sistemos paketų struktūra

Šis skyrius aprašo sistemos loginę struktūrą. Pateikia sistemos išskaidymą į paketus ir juos sudarančias klases.

2.6.1 Apžvalga



Pav. 18 Kuriamos sistemos paketų struktūra

2.7 Detali sistemos architektūra

Naudosime MVC šablonu paremtą architektūrą. Pagrindinis komponentas *Pagrindinis valdiklis*. Šis valdiklis valdo visus kitus klasių objektus, nuo jo priklauso sistemos veikiamumas, funkcionalumas, priėjimas. Šiam paketui taip pat priklauso ir vartotojų klasė, vartotojų sąskaitų klasė ir pagalbinių puslapių klasės. Taip pat šiame pakete pavaizduoti pagrindinio valdiklio sąryšiai su renginių ir kalbų klasėmis.

2.7.1 Pagrindinis valdiklis

Klasifikacija

Paketas

Apibrėžimas

Šiame pakete yra pateiktos išsamios klasės, susijusios su vartotojų valdymu, taip pat priėjimo prie sistemos, kalbų, turinio elementų valdymui.

Atsakomybės

Šio paketo pagalba vartotojai gali prisijungti prie sistemos, pasirinkti kokia kalba peržiūrėti puslapius, taip pat skaityti informacinius puslapius.

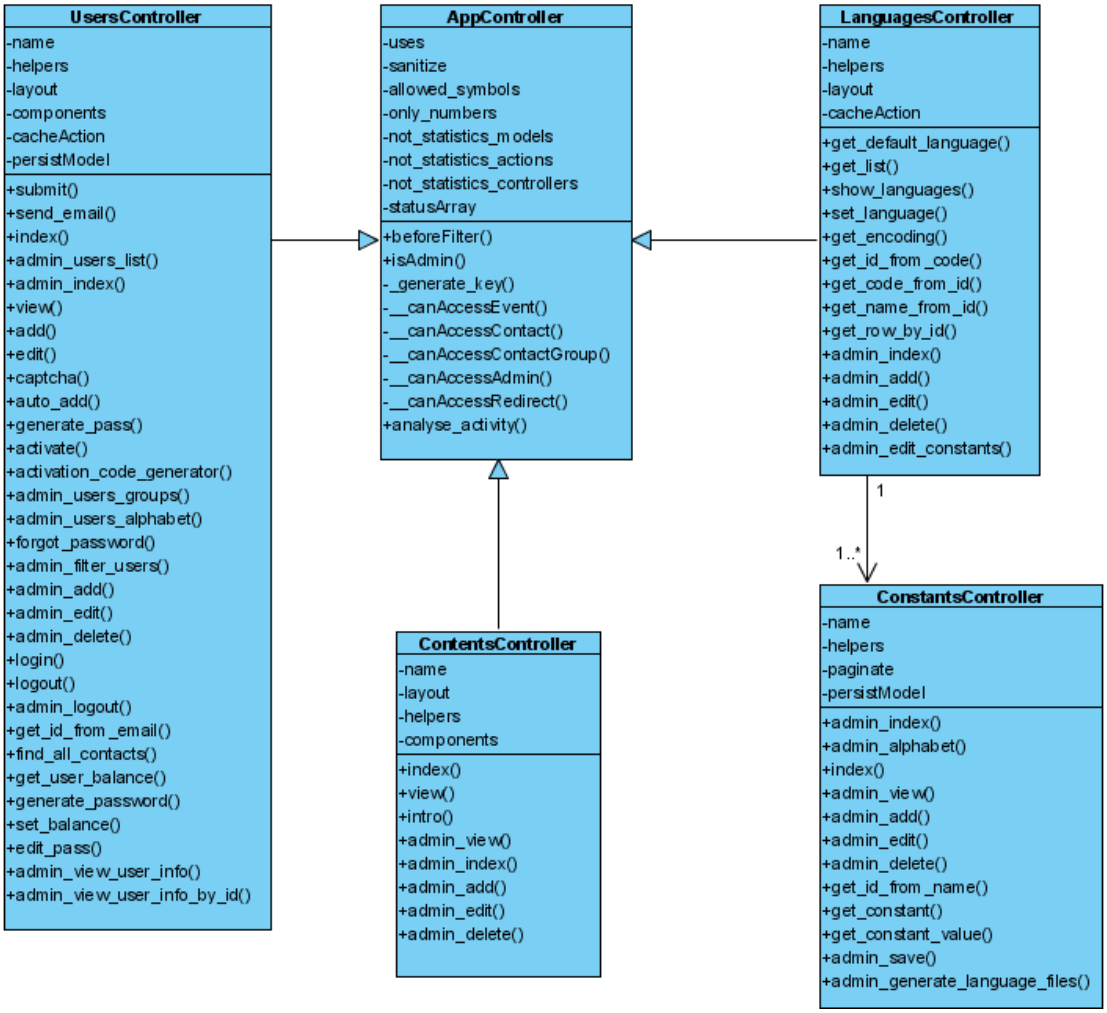
Sistema šio paketo dėka gali sekti vartotojo veiksmų statistiką, taip pat leisti ar drausti priėjimą prie sistemos sričių.

Apribojimai

Šiuo komponentu gali pasinaudoti visi sistemos vartotojai, tačiau paprasti vartotojai negali prieiti prie administravimui skirtų metodų.

Struktūra

Paketo struktūra pateikiama pav.19



Pav. 19 Pagrindinis valdiklis

Šį komponentą sudaro penkios klasės.

Sąsaja/eksportas

Šis komponentas įgalina priėjimą prie vartotojų, kalbų, konstantų, turinio elementų duomenų. Taip pat nustato priėjimo teises kiekvienai vartotojų grupei.

Kadangi daugumai metodų kintamieji yra paduodami per globalius kintamuosius, jie neatsispindi kaip metodų parametrai.

Visos šio paketo klasės paveldi klasę AppController. O klasė LanguagesController turi sąryšį su ConstantsController.

2.7.2 Renginių valdiklis

Klasifikacija

Paketas

Apibrėžimas

Šiame pakete yra pateiktos išsamios klasės, susijusios su renginių tvarkymu ir valdymu. Į šį klasių paketą taip pat yra prijungtos ir renginių dalyvių tvarkymo bei pranešimų siuntinėjimo klasės.

Atsakomybės

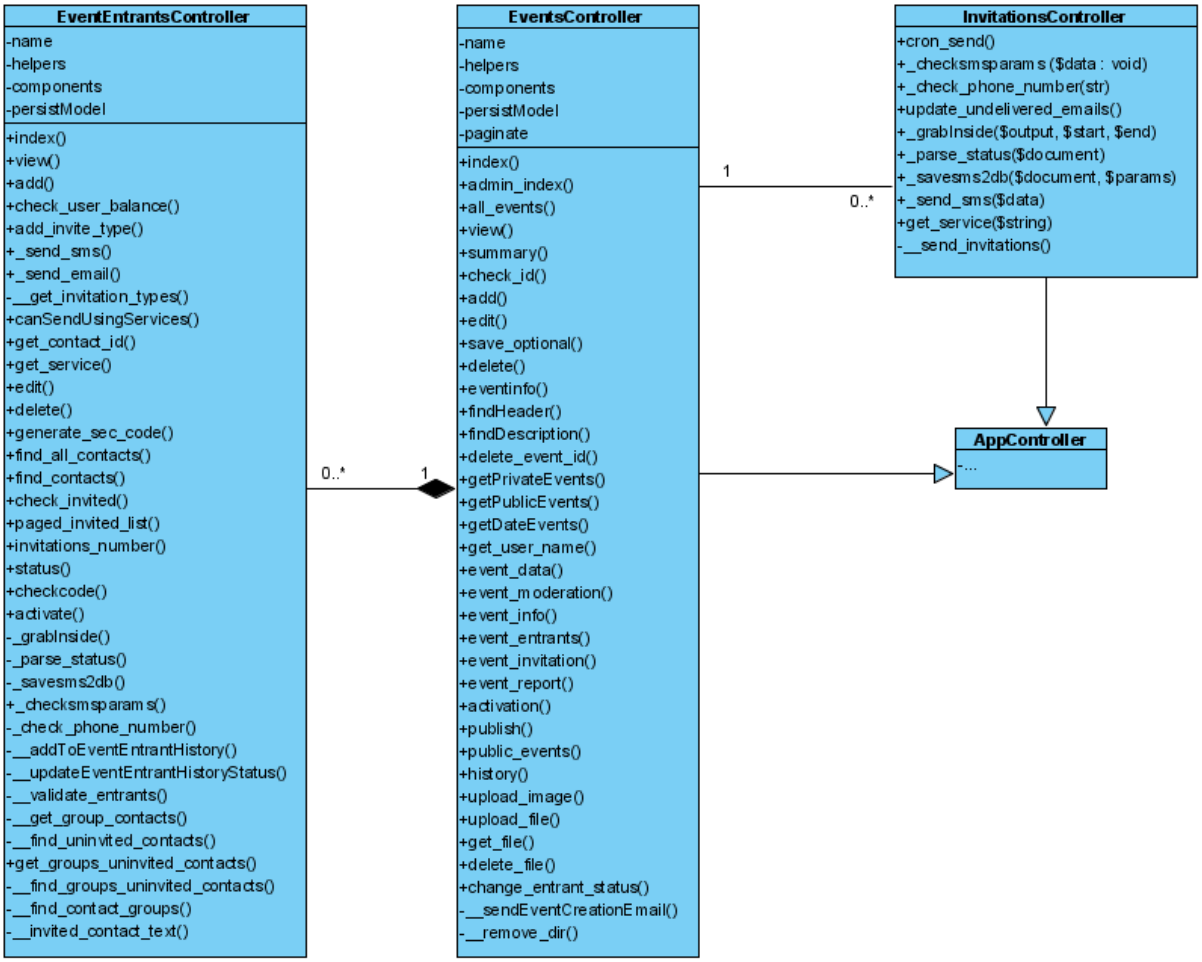
Šio paketo pagalba sistemos vartotojas gali valdyti savo renginius, taip pat nurodyti, kuriuos kontaktus kviesti, bei kuriem renginio svečiams išsiųsti pakvietimus.

Apribojimai

Šiuo komponentu gali pasinaudoti tik prisijungę prie sistemos vartotojai.

Struktūra

Paketo struktūra pateikiama pav.20



Pav. 20 Renginių valdiklis

Šį komponentą sudaro trys klasės.

Sąsaja/eksportas

Šis komponentas įgalina priėjimą prie renginių, pakvietimų, renginio dalyvių duomenų. Taip pat suteikia teisę siųsti pakvietimus renginio dalyviams.

Kadangi daugumai metodų kintamieji yra paduodami per globalius kintamuosius, jie neatsispindi kaip metodų parametrai.

Visos šio paketo klasės paveldi klasę AppController. O klasė EventsController turi sąryšį su InvitationsController.

2.7.3 Mokėjimų valdiklis

Klasifikacija

Paketas

Apibrėžimas

Šiame pakete yra pateiktos išsamios klasės, susijusios su mokėjimu priėmimu iš vartotojų ir mokėjimų istorijos sekimu.

Atsakomybės

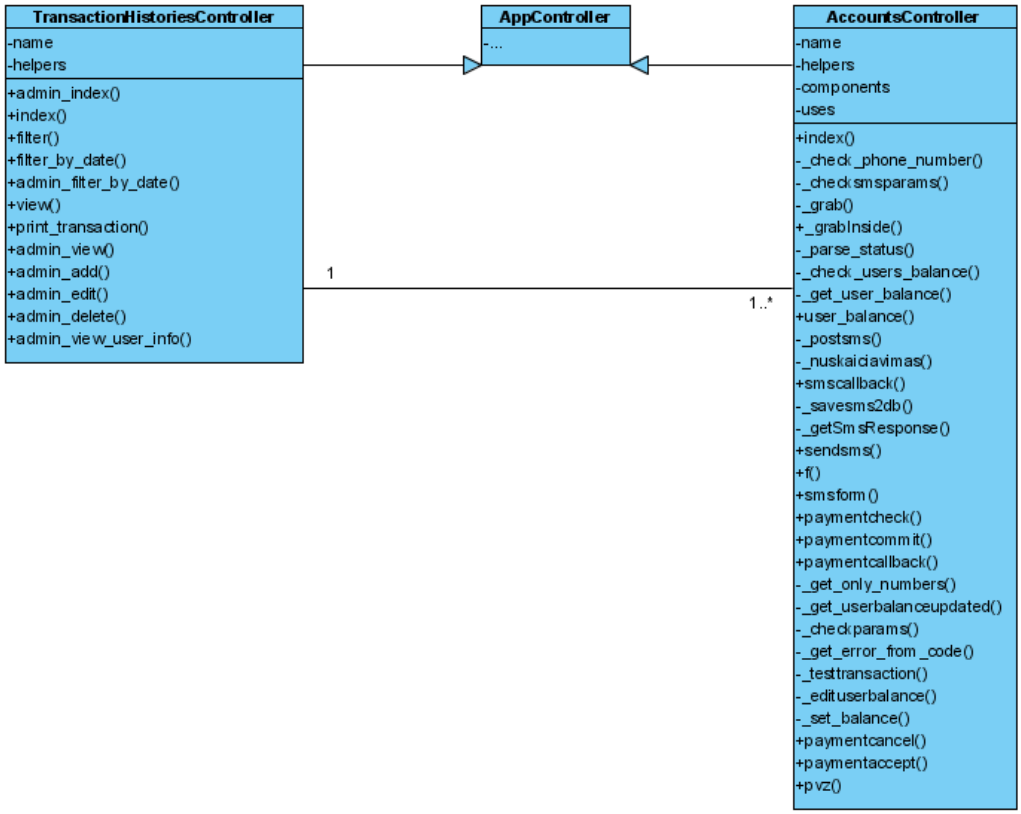
Šio paketo pagalba sistemos vartotojas gali pasididinti kreditų skaičių sistemoje, taip pat matyti kokius mokėjimus ir kada jis yra atlikęs

Apribojimai

Šiuo komponentu gali pasinaudoti tik prisijungę prie sistemos vartotojai.

Struktūra

Paketo struktūra pateikiama pav.21



Pav. 21 Mokėjimų valdiklis

Šį komponentą sudaro dvi klasės.

Sąsaja/eksportas

Šis komponentas įgalina priėjimą prie mokėjimų, ir mokėjimų istorijos duomenų. Taip pat šio komponento pagalba yra kreipiamasi į išorinę sistemą.

Kadangi daugumai metodų kintamieji yra paduodami per globalius kintamuosius, jie neatsispindi kaip metodų parametrai.

Visos šio paketo klasės paveldi klasę AppController. O klasė AccountsController turi sąryšį su TransactionHistoriesController.

2.7.4 Kontaktų valdiklis

Klasifikacija

Paketas

Apibrėžimas

Šis paketas yra išskirtas vartotojo kontaktų ir kontaktų grupių valdymui. Šiame pakete yra pateiktos dvi klasės, viena apibrėžia kontakto objektą, kita kontaktų grupės objektą.

Atsakomybės

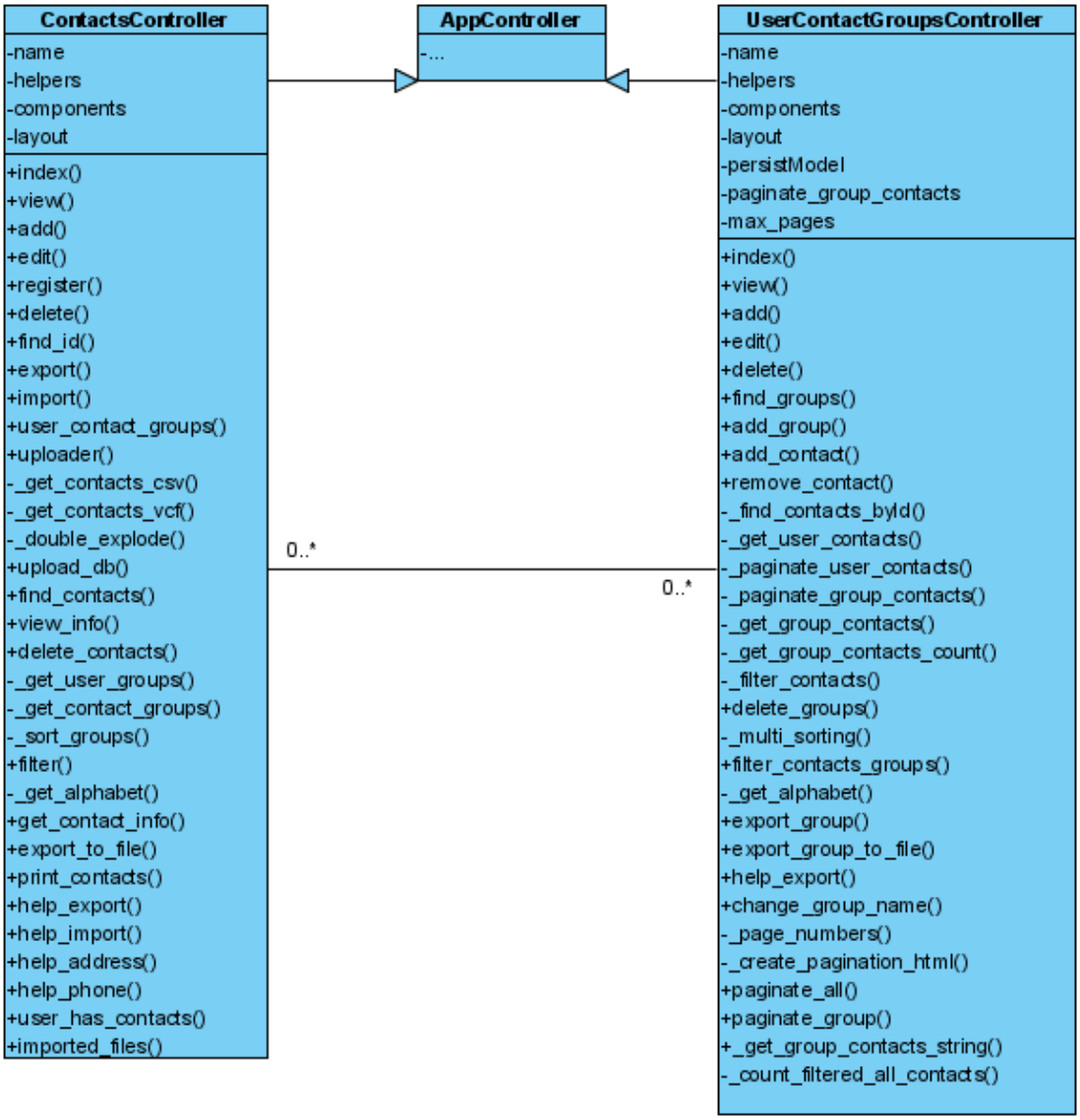
Šio paketo pagalba sistemos vartotojas gali tvarkyti savo kontaktų ir kontaktų grupių sąrašą. Vartotojas gali importuoti kontaktus iš išorinių failų, failų standartai: csv (kableliu arba „tab“ simboliu atskirtos reikšmės), vCard. Vartotojas taip pat gali ir eksportuoti savo kontaktus arba jų grupes į csv tipo failus. Vartotojo importuoti failai yra saugomi sistemoje, vartotojas visuomet juos gali parsisiųsti. Kontaktus galima sukurti, redaguoti, ištrinti. Jeigu yra nurodytas teisingas mob. numeris ir vartotojo sąskaitos balansas pakankamas, vartotojas gali tiesiogiai nusiųsti kontaktui sms žinutę. Vartotojas gali sukurti, redaguoti ir ištrinti kontaktų grupes. Sukurtoms kontaktų grupėms galima priskirti jau turimus arba importuojamus kontaktus.

Apribojimai

Šiuo komponentu gali pasinaudoti tik prisijungę prie sistemos vartotojai. Vartotojai sms žinutę kontaktui gali išsiųsti tik tuomet, jeigu yra nurodytas teisingas mobilaus telefono numeris ir vartotojo sąskaitos balansas pakankamas. Vienu metu galima importuoti ne daugiau kaip 1000 kontaktų arba importuojamo failo dydis neturi viršyti 600 Kb. Vartotojo importuotų failų saugomas kiekis yra 20, jeigu viršijama, trinami seniausiai įkelti failai. Vartotojo kontaktų skaičius yra ribojamas iki 5000. Vartotojo kontaktų grupių skaičius yra ribojamas iki 100. Kontaktų ir kontaktų grupių sąrašai yra puslapiuojami po 30 įrašų.

Struktūra

Paketo struktūra pateikiama pav. 22.



Pav. 22 Kontaktų valdiklis

Šį komponentą sudaro dvi klasės.

Sąsaja/eksportas

Šis komponentas įgalina priėjimą prie kontaktų ir kontaktų grupių duomenų.

Kadangi daugumai metodų kintamieji yra paduodami per globalius kintamuosius, jie neatsispindi kaip metodų parametrai.

Visos šio paketo klasės paveldi klasę AppController. Klasė ContactsController turi sąryšį su UserContactGroupsController.

2.8 Išvados

Sudarant specifikaciją buvo susidurta su reikalavimų išgavimo iš užsakovo problemomis, kurių analizė užima daug laiko. Tai tikrai labai naudinga patirtis tolimesniems projektams. Buvo atlikta analizė ir sudaryta išsami reikalavimų specifikacija. Įgyta daug patirties, sudarinėjant

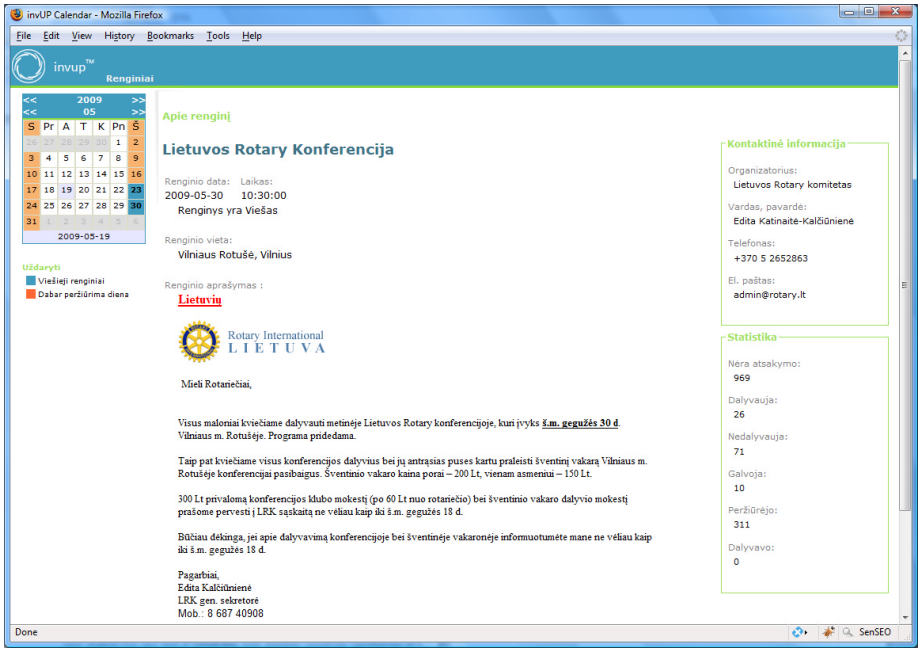
reikalavimų specifikaciją, pagal „Volere“ šabloną. Buvo pasinaudota daugybe puikių patarimų ir pasiūlymų. Įgyta patirties dirbant komandinį darbą. Sudaryti pradiniai reikalavimai kuriamai sistemai.

Įgauta patirties reikalavimų išgavime iš užsakovo, taip pat pastebėta, kad esant skirtingiems suvokimo lygiams, kartais priimamos klaidingos prielaidos. Taip pat buvo įgyta daug patirties dirbant komandoje ir bandant suderinti bei paskirstyti darbus tolygiai tarp komandos narių. Taip pat įgauta daug patirties sprendimų priėmime, esant išoriniams trukdžiams, kaip pvz.: laiko, lėšų, turimų technologijų ir kitokių resursų. Optimizuojant darbą ir taupant laiką bei kitus resursus teko domėtis įvairiomis naujovėmis ir gerai įsisavinti darbo priemones.

3 Tyrimo dalis

3.1 Sistemos tobulinimo kryptis

Atsirandant naujiems poreikiams ir tobulėjant technologijų galimybėms, atsiranda ir naujos sistemos tobulinimo galimybės. Renginių organizavimo sistemoje, analizuojant vartotojų poreikius ir norus, buvo nustatytos naujos tobulinimo galimybės. Pastebėta, kad niekuo neišsiskiriantys, vienodi visų renginių pakvietimai nepatraukia vartotojų dėmesio. Taip pat ir organizatoriai pareiškė savo norus, kad jų renginiai išsiskirtų, nes kiekvieno žmogaus skoniai skirtingi. Nuspręsta realizuoti skirtingų renginio aprašo šablonų pasirinkimą. Prieš tai buvęs standartinis renginių aprašo dizainas pateiktas žemiau esančiame paveikslėlyje.



Pav. 23 Standartinis renginių aprašo dizainas

3.2 Sistemos kūrimo ir tobulinimo modelis

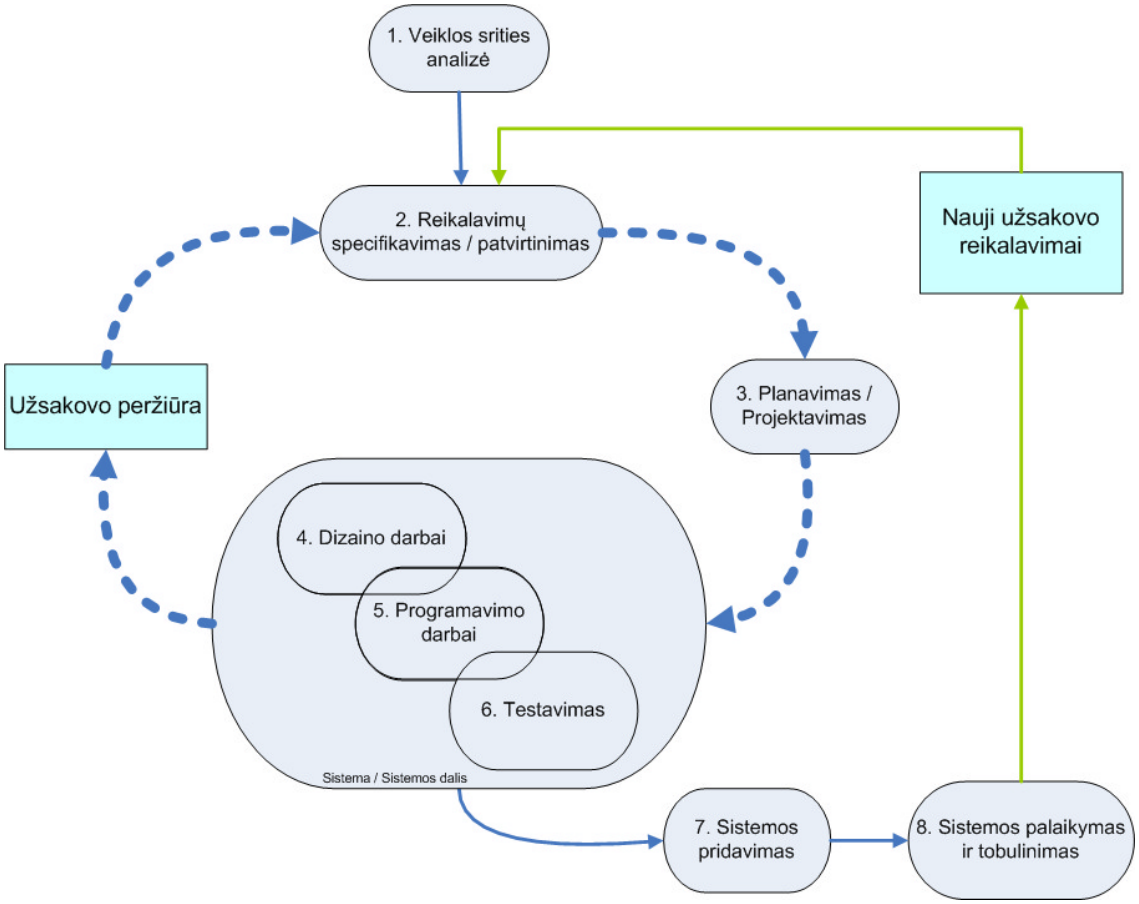
Internetinės sistemos kūrimo ir palaikymo ciklas susideda iš šių etapų:

1. Veiklos srities analizė
2. Reikalavimų specifikavimas
3. Planavimas – projektavimas
4. Dizaino darbai
5. Programavimo darbai
6. Testavimas
7. Sistemos pridavimas
8. Sistemos palaikymas ir tobulinimas

Teoriškai sistemą galima būtų sukurti nuosekliai vykdant šiuos etapus vieną po kito. Tačiau praktikoje, kuriant vidutinės apimties internetines sistemas, dauguma šių etapų persidengia, vykdomi lygiagrečiai ir dažniausiai būna iteraciniai. Klientas dažniausiai negali iš karto nusakyti visų savo norų ir reikalavimų, todėl pradedant reikalavimų specifikavimu ir

baigiant testavimu, šiuos etapus reikia vykdyti iteratyviai, pateikiant dalinai realizuotą sistemą užsakovui. Dauguma sistemų realizuojamos iteraciniu metodu.

Renginių organizavimo sistemos kūrimo etapai pateikti žemiau esančioje diagramoje:



Pav. 24 Renginių organizavimo sistemos kūrimo etapai

Kaip matome iš diagramos, 2, 3, 4, 5, ir 6 etapai yra vykdomi iteraciniu būdu, t.y. kas kartą pateikiant užsakovui sukurtą versiją, jis ją peržiūri ir patvirtina arba nustato tolimesnius reikalavimus. Atsiradus naujiems reikalavimams po sistemos sukūrimo taip pat kartojami tie patys etapai. Dizaino, programavimo ir testavimo darbai apjungti į vieną sritį, nes dalį jų galima vykdyti lygiagrečiai arba nebūtinai visi trys etapai vykdomi kiekvienoje iteracijoje.

Didžiausią sistemos kūrimo laiką užėmė šie etapai:

- Dizaino darbai
- Programavimo darbai
- Grafinės vartotojo sąsajos testavimas
- Sistemos palaikymas ir tobulinimas

Ypač sistemos palaikymas ir tobulinimas užėmė labai daug laiko, nes užsakovo reikalavimai keitėsi ir jų nuolatos daugėjo.

Šiame darbe plačiau išnagrinėsiu šiuos sistemos kūrimo etapus: dizaino, programavimo ir grafinės vartotojo sąsajos testavimo darbus.

3.3 Naudoto sistemos kūrimo ir tobulinimo modelio problemos

Sistema buvo kurta naudojant HTML, CSS, JavaScript, AJAX, PHP ir MySQL technologijas. Pradžioje vienas arba keli grafiniai dizaino šablonai yra sukuriami pagal užsakovo reikalavimus ir pageidavimus. Kai užsakovas pasirenka konkretų dizaino variantą, jis yra ištobulinamas. Dizainas buvo kurtas grafinio redaktoriaus Adobe Photoshop pagalba. Dizaineris sukuria bendrą sistemos išdėstymą, stilių, sukuria konkrečius sistemos grafinius elementus. Žinant sistemos struktūrą, pradedami programavimo darbai. Programuotojas kuria sistemos funkcionalumą.

Ir čia iškyla pagrindinis klausimas, į kurį nelabai lengva rasti atsakymų. Kaip tarpusavyje dirba dizaineris su programuotoju? Dizaineriai turi gerus įrankius, su kuriais gali sukurti nuostabius grafinius sistemų dizainus, o programuotojai turi gerus įrankius ir pažangias technologijas, kurių pagalba gali sukurti visą sistemos veiklos logiką. Tačiau kodėl yra taip sunku šiuos du etapus sulipdyti į vieną? Problema ir yra jog šie, gana skirtingi, etapai yra bandomi sulipdyti tarpusavyje. Dizainas ir programavimas turėtų būti kaip ingridientai, papildantys vienas kitą, skirti sukurti visai visumai. Nors internetinių sistemų kūrimas yra gerokai pažengęs, tačiau ir dabar išlieka prastas bendradarbiavimas tarp grafinės aplinkos dizainerių ir programuotojų. Dėl to labai dažnai nesusikalbama tarpusavyje. Dizaineriai dažnai būna nepatenkinti, kai pamato sistemoje tam tikrus netinkamai programuotojų realizuotus elementus, kaip pavyzdžiui: mygtukas trejais „pikseliais“ per daug paslinktas į kairę, arba mygtuko šešėlis visai ne tokio atspalvio, kaip turėtų būti. Ir dažnai būna atvirkščiai, kai dizaineris nesupranta, kad programuotojas turės įdėti nemažai pastangų, perrašydamas sistemos kodą, jog grafinį elementą perkeltų kitur arba jį pakeistų kitu elementu.

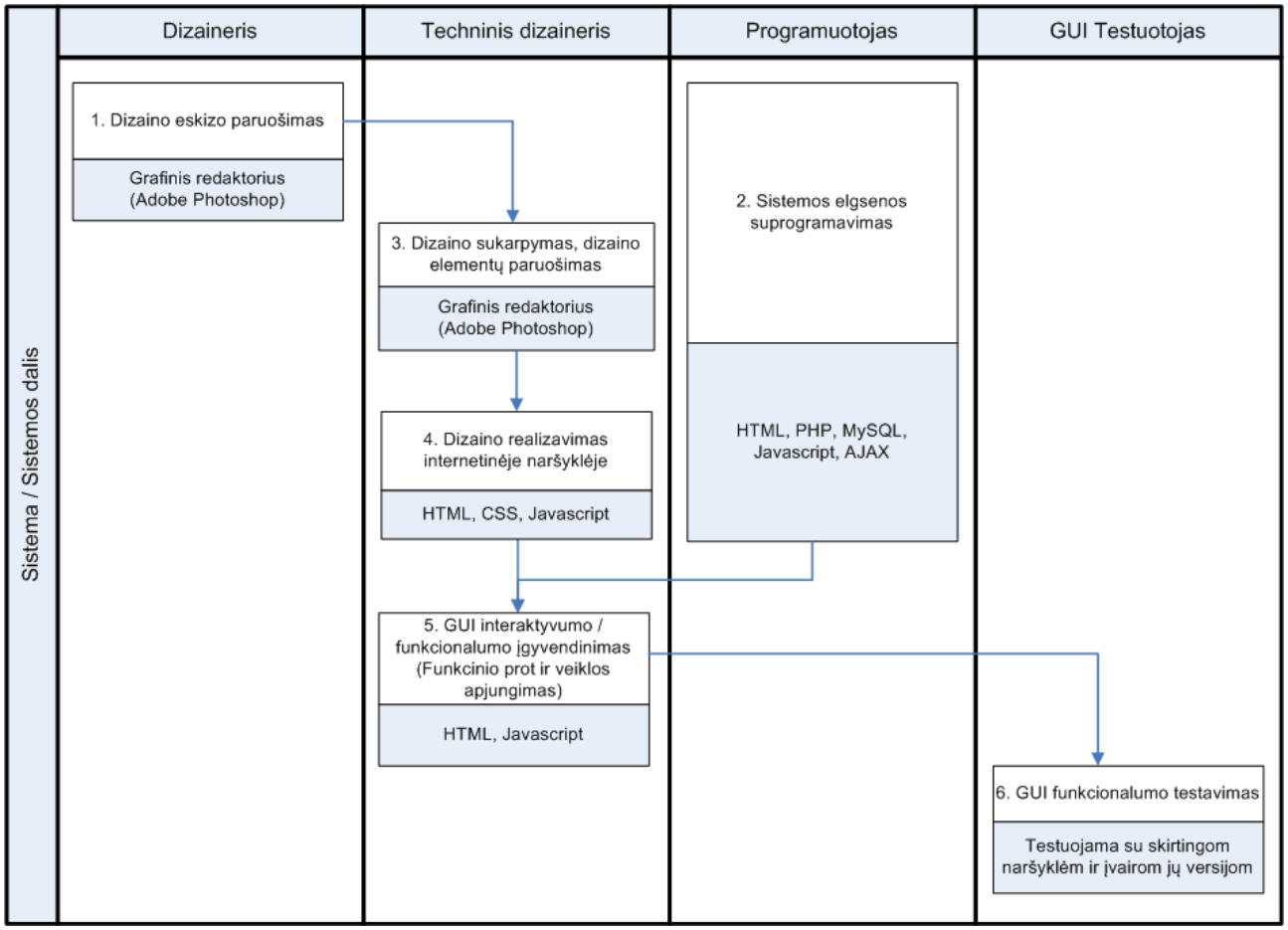
Tarp paminėtų dviejų rolių (dizainerio ir programuotojo) atsiranda dar viena, kuri skirta dizainerių ir programuotojų darbui palengvinti ir apjungti – tai techninis dizaineris. Jis turi priimti iš dizainerių PSD failą, paversti jį į xHTML (transitional) + CSS formatą ir perduoti programuotojams. Darbas nėra lengvas tik dėl to, kad tenka dirbti tarp žemės ir dangaus, t.y. šiam žmogui tenka bendrauti su fantazijos ribų neturinčiais dizaineriais ir tikslumo reikalaujančiais programuotojais. Nuolatos kuriamom naujom naršyklėm ir dažnai leidžiamom naujom jų versijom, jis turi pritaikyti kodą pagal W3C rekomendacijas (HTML ir CSS elementams).

Taigi, į dizaino ir programavimo darbus įsitraukia dar viena rolė – tai techninis dizaineris. Tiesioginio bendradarbiavimo tarp dizainerio ir programuotojo nebelieka. Kaip jau buvo minėta, tokiu metodu realizuotos sistemos pakeitimai tampa labai skausmingi, nes sistemos grafinė vartotojo aplinka yra įterpta į programos kodą. Kad ir menkiausias dizaino pasikeitimas gali įtakoti nemažus sistemos rekonstrukcijos darbus.

Susiduriama su dviem labai skaudžiomis kliūtimis:

- Norint sukurti sistemą, padidėja reikalingas resursų skaičius – reikia dar vienos rolės – tai techninio dizainerio, gerai išmanančio savo darbą. Nebelieka bendradarbiavimo tarp programuotojo ir dizainerio. Atsiradus dar vienai rolei daugiau laiko sugaištama tarpusavio komunikavimui.
- Norint atlikti pakeitimus jau realizuotoje sistemos grafinėje aplinkoje tai pat gerokai išauga resursų skaičius. Į pakeitimo etapą įtraukiamos visos trys rolės. Dažnai programuotojui ar techniniam dizaineriui gali iškilti daugiau problemų realizuojant pakeitimą, negu dizaineriui, kuris ilgai neužtruko pakeisti tam tikrus šešėlius, atspalvius ar apvalumus PSD faile.

Žemiau pateiktame paveikslėlyje pavaizduotos darbuotojų rolės, jų atliekami darbai, darbų eiliškumas ir naudojamos technologijos ar įrankiai.



Pav. 25 Sistemos dalies realizavimo darbų eigos diagrama

25 pav. pateiktoje diagramoje galime matyti, kad kai kurie darbai yra priklausomi vieni nuo kitų ir jų negalima pradėti, kol nebus pabaigti pirminiai darbai. Norint pradėti 3 darbą, pirmiau reikia atlikti 1 darbą. Realizavus 3 darbą, galima pradėti 4 darbą. Norėdami atlikti 5

darbą, turime pabaigti 4 ir 2 darbus. Ir tik galiausiai gali būti pradėtas 6 darbas. Realybėje vieni darbai užtrunka ilgiau, kiti trumpiau, dėl to vieno darbo vėlavimai gali prailginti kitų darbų atlikimo terminus.

Vartotojo sąsajos testuotojas privalo ištestuoti sukurtą sistemą ar jos dalį su skirtingomis, dažniausiai naudojamomis internetinėmis naršyklėmis, kaip pvz.: Microsoft Internet Explorer (versijos: 6,7), Mozilla Firefox (versijos: 2, 3), Opera, Apple Safari, Google Chrome. Čia tik dalis dažniausiai naudojamų naršyklių, kurios tam tikrus HTML elementų stilius ar JavaScript kodo interpretavimą gali įvykdyti skirtingai.

Tobulinant šiuo metodu iš karto galime pamatyti, kad daugybę kartų bus kartojami tie patys, sudėtingi vartotojo grafinės aplinkos kūrimo ir testavimo veiksmai.

3.4 Naujo funkcionalumo realizavimo metodai

Skirtingų šablonų pasirinkimą renginio aprašui galima realizuoti ir šiame projekte jau naudotomis technologijomis, tačiau žvelgiant į ateitį, jos sukeltų tam tikrų apribojimų tolimesniam tobulinimui. Visų pirma, naudotos technologijos neturi audio/video turinio palaikymo. Kita labai svarbi priežastis – jos neleistų vartotojams patiems įsikelti naujų šablonų, nes renginio aprašas yra generuojamas serverio pusėje, naudojant PHP kalbą. Taigi, atsižvelgiant į dar tolimesnius sistemos tobulinimus, buvo pasirinkta Silverlight technologija. Kadangi sistema realizuota PHP kalbos pagrindu, buvo išnagrinėti keli apjungimo atvejai. Pirmasis – tai pasinaudojant “WebORB for PHP” sukurtos aplinkos pagalba. “WebORB for PHP” yra vykdymo aplinka, skirta lengvai ir paprastai sujungti Flex, Flash ir Silverlight klientų programas su PHP klasėmis ir jų valdomomis realiacinėmis duomenų bazėmis. Šis variantas tikrai palengvintų sistemos susiejimą su Silverlight platforma, tačiau jis dar vystymo stadijoje. Kiek teko bandyti – neįmanoma susieti PHP klasių su tam tikromis Silverlight versijomis. Kol kas tai projektas, kuris yra labiau koncentruotas į Flex ir Flash technologijas. Pilnai ir gerai veikiančio Silverlight palaikymo dar teks palaukti. Todėl šį variantą teko atmesti. Buvo nuspręsta Silverlight programai duomenis perduoti XML formatu.

3.5 Naujo funkcionalumo realizavimas Silverlight technologija

3.5.1 Privalomi sistemos logikos pakeitimai

Renginio aprašo dizainą buvo planuota realizuoti su Silverlight technologija, kuri veikia kliento naršyklės pusėje ir tiesioginio priėjimo prie duomenų bazės neturi. Buvo pasirinkta duomenis pateikti XML kalbos formatu. Visa informacija apie renginį paimama iš duomenų bazės ir patalpinama faile, XML formatu.

Taip pat reikia pertvarkyti ir renginio peržiūrą generuojantį valdiklį, kuriame ir buvo realizuotas esamo renginio peržiūros dizainas. Dizaino generavimas tampa nebereikalingas, tereikia pasirūpinti, kad būtų užkrautas reikiamas šablonas.

3.5.2 Rengino aprašo grafinės aplinkos kūrimas

Renginio aprašo komponento grafinei sąsajai sukurti panaudota XAML kalba.

Būtent XAML kalba ir yra pagrindas dizainerio ir programuotojo tarpusavio bendradarbiavimui. XAML, kaip atskiras technologijos vienetas šio bendradarbiavimo nesukurtų, tačiau ji yra pagrindas sistemos, paremtos Silverlight technologija, grafinės aplinkos kūrimui.

XAML ne tik suteikia jau egzistuojančių žymų (angl. Markups) kalbų funkcionalumą (kaip pvz. HTML), tačiau jį ir praplečia, ko negali kitos kalbos. XAML gali aprašyti ne tik valdymo mygtukus, sąrašus, tekstus, paveiksliukus ar struktūros elementų išdėstymą, tačiau ir vektorinę grafiką, spalvų perėjimus (angl. gradient), kampų apavalumus, įvairias figūras ir pan.

XAML kalbai Microsoft kompanija yra sukūrusi ir grafinį redaktorių – tai Expression Blend, kurio pagalba ir buvo sukurta grafinė vartojo sąsaja. Kūrimas su šiuo įrankiu yra patogus ir intuityvus, visi reikalingi objektai gali būti pasirenkami iš objektų panelės. Sistemos dizainą galima kurti ar redaguoti XAML kalboje tiek tekstiniu, tiek ir grafiniu redaktoriumi. XAML kalba aprašytos vartotojo sąsajos ir joje esančio valdymo mygtuko kodo pavyzdys:

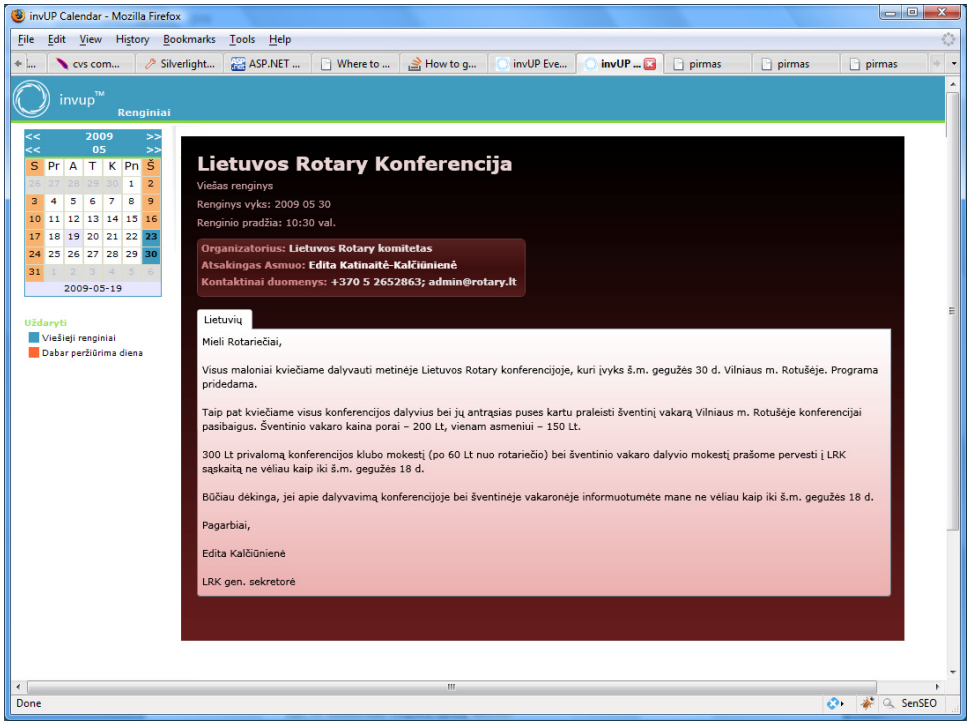
```
<UserControl x:Class="UserCtrl.Card"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Width="200" Height="360" x:Name="Icon">
           <Button Name="pirmas_mygtukas" Content="Paspausk" Width="75"
           Height="23" Click="pirmas_metodas">
           </Button>
</UserControl>
```

Silverlight sistemos programuotojas gali glaudžiai bendradarbiauti su dizaineriu. Būtent sistemos grafinės aplinkos aprašymas XAML kalboje ir sukuria šį tarpusavio bendradarbiavimą. Sukurtas dizainas gali būti iš karto kompiliuojamas ir paleidžiamas naršyklėje. Iš sukurto dizaino iš karto gaunamas dalinai funkcinis sistemos prototipas, nepriklausomai ar sistemos funkcionalumas jau realizuotas ar dar ne.

Viskas ką sukūrė dizaineris yra aprašyta XAML tipo failuose. Programuotojas arba dizaineris valdomiems elementams priskiria vardus XAML faile. Renginio elementų reikšmės paimamos iš sugeneruoto XML failo ir parinktiems vardams XAML faile vėliau priskiriamos

reikšmės Silverlight programoje, C# kalba. Šiam etapui realizuoti buvo pasirinkta Silverlight 2 versija.

Gautas tyrimo rezultatas – tai kitoks renginio aprašo vaizdas, kurį galima lengvai pakeisti, arba jo pagrindu sukurti kitus renginio aprašo šablonus. Vartotojo sąsajos sluoksnis visiškai atskiriamas nuo sistemos logikos, nors jis turi pilną priėjimą prie duomenų. Žemiau pateiktas naujai realizuoto renginio aprašo dizaino šablonas:



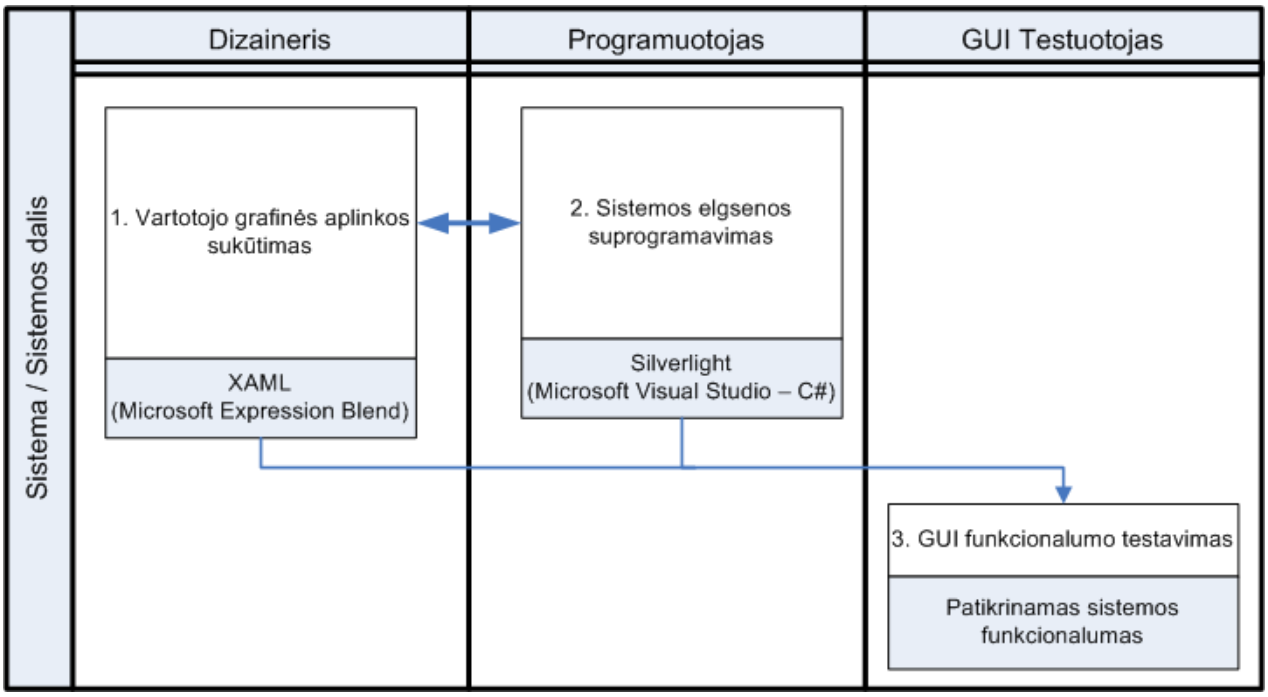
Pav. 26 Naujai realizuoto renginio aprašo šablonas

3.6 Silverlight technologija paremtas sistemos kūrimo modelis

Silverlight technologija kartu su XAML leidžia dizaineriui tiesiogiai dalyvauti projekto kūrime ir prisidėti savo sprendimais. Nebereikia, kad jis aiškintų techniniam dizaineriui savo idėjas ir bandytų perteikti numatytą interaktyvumą. Dizaineris gali lengvai bet kuriuo metu įsitraukti į sistemos kūrimo procesą ir nesunkiai pakeisti tam tikrus grafinės aplinkos aspektus. Dėl dizainerio pakeitimų grafinėje sistemos aplinkoje, nereikės nieko rekonstruoti sistemoje (nebent keičiamas sistemos funkcionalumas).

Dizainerio kuriamas prototipas iš karto yra ir funkcinis prototipas. Dizaineriui nebereikia prašyti techninio dizainerio pagalbos, kad jis nupieštą dizainą paverstų funkcionaliu. Dizaineris kartu su programuotoju dirba prie to pačio projekto, toje pačioje platformoje, kurioje sprendimus gali priimti bendrai. Naujų sprendimų priėmimas gali būti vietoje ir iš karto realizuojamas. Visa tai veda prie greito ir patogaus rezultato gavimo. 27 pav. pateikta diagrama, apibūdinanti dizaino ir programavimo etapų procesus. Galima būtų pabrėžti, kad atlikus pradinius programavimo

darbus, apžvelgtus 3.5.1 poskyryje, iškrenta ir programuotojo rolė. Dizaineris gali lengvai kurti naujus dizaino šablonus ir juos įdiegti be programuotojo pagalbos.



Pav. 27 Sistemos kūrimo etapai Silverlight technologija

Šiame kūrimo modelyje nebelieka techninio dizainerio rolės. Dizaineris gali kurti Raiškiųjų internetinių sistemų aplinkas, naudodamas tik vieną – tai XAML. XAML kalbos pagrindu galima realizuoti visus Raiškiųjų internetinių sistemų principus. Programuotojas sistemos elgseną gali realizuoti pasinaudodamas Silverlight ir .NET technologijų teikiamomis funkcijomis, jam užtenka Visual Studio įrankio.

Kuriant Silverlight technologija paremtą sistemos dalį, dingo techninio dizainerio rolė. Atsiranda glaudus dizainerio ir programuotojo bendradarbiavimas komandoje.

Grafinės aplinkos testuotojui, tereikia patikrinti ar sukurta sistema, ar jos dalis veikia pagal reikalavimus. Testuotojui nebereikia patikrinti sistemos su skirtingomis naršyklėmis bei skirtingomis jų versijomis.

Projektas tampa komandiniu darbu, kuriame yra glaudžiai ir patogiai bendradarbiaujama tarpusavyje.

3.7 Silverlight technologija paremto kūrimo modelio įvertinimas

Buvo realizuotas naujas renginių šablonų modulis ir įvertinti naujo šablono kūrimo laikai. Po šablono sukūrimo buvo atlikti dizaino pakeitimo darbai ir taip pat įvertinti jų laikai. Remiantis prieš tai apžvelgtais kūrimo proceso modeliais, įvertinti darbų atlikimo laikai. Atlikti kūrimo ir pakeitimo darbai dviem būdais: vienu atveju buvo kuriama, pasitelkiant HTML, CSS ir JavaScript technologijas, kitu atveju buvo realizuota su Silverlight technologijomis. Sistemos logikos pakeitimo darbai, kad įdiegti Silverlight technologiją sistemoje, neįskaičiuoti.

Darbai yra suskirstyti pagal darbuotojų roles ir kiekvienos rolės atliekamus etapus. Buvo paimti vidutiniai darbų atlikimo laikai, įvertinant, kad darbuotojai turi panašią patirtį su visom technologijomis ir žino kaip šiuos darbus atlikti, t.y. laikai išmokti konkrečiam darbui atlikti (jeigu to dar nemoka) neįskaitomi.

3.7.1 Naujo modulio sukūrimo įvertinimas

Žemiau esančioje 3 lentelėje pateikti naujo šablono sukūrimo laikai, naudojant Adobe Photoshop ir Zend Development Environment įrankius, HTML, CSS, JavaScript ir PHP kalbų pagrindu.

Rolė	Darbai	Laikas, val
Dizaineris	Dizaino nupiešimas	2,5
Techninis dizaineris	Dizaino paruošimas ir sukarpymas	2
	Funkcinio maketo paruošimas	-
	Pagrindinis funkcinio maketo sukūrimas	3
	Pritaikymas IE6 naršyklei	1
	Pritaikymas IE7 naršyklei	0,5
	Vartotojo sąsajos interaktyvumo sukūrimas	3
Programuotojas	Programavimo darbai	2
GUI testuotojas	Dizaino sulyginimas su pradiniu ir funkcionalumo testavimo darbai	-
	Testavimas su Firefox 2 naršykle	0,3
	Testavimas su Microsoft IE 6 naršykle	0,5
	Testavimas su Microsoft IE 7 naršykle	0,4
	Testavimas su Safari naršykle	0,2
	Testavimas su Opera naršykle	0,2
	Viso:	15,6

Lentelė 3 Naujo šablono sukūrimo darbų laikai

Žemiau esančioje 4 lentelėje pateikti naujo šablono sukūrimo laikai, naudojant Expression Blend ir Visual Studio įrankius, XAML ir C# kalbų pagrindu.

Rolė	Darbai	Laikas, val
Dizaineris	Dizaino kūrimas	3
	Vartotojo sąsajos interaktyvumo sukūrimas	3
Programuotojas	Modulio programavimo darbai	2
GUI testuotojas	Dizaino sulyginimas su pradiniu ir funkcionalumo testavimo darbai	-
	Testavimas su Firefox 2 naršykle (neturi reikšmės kokia naršyklė naudojama)	0,3
	Viso:	8,3

Lentelė 4 Naujo šablono sukūrimo darbų laikai su Silverlight

3.7.2 Sukurto modulio pakeitimo įvertinimas

Žemiau esančioje 5 lentelėje pateikti naujo šablono sukūrimo laikai, naudojant Adobe Photoshop ir Zend Development Environment įrankius, HTML, CSS, JavaScript ir PHP kalbų pagrindu.

Rolė	Darbai	Laikas, val
Dizaineris	Dizaino perpiešimas	1
Techninis dizaineris	Dizaino paruošimas ir sukarpymas	1
	Funkcinio maketo paruošimas	-
	Pagrindinis funkcinio maketo sukūrimas	1,5
Programuotojas	Modulio programavimo darbai	0,3
GUI testuotojas	Dizaino sulyginimas su pradiniu ir funkcionalumo testavimo darbai	
	Testavimas su Firefox 2 naršykle	0,2
	Testavimas su Microsoft IE 6 naršykle	0,2
	Testavimas su Microsoft IE 7 naršykle	0,2
	Testavimas su Safari naršykle	0,2
	Testavimas su Opera naršykle	0,2
	Viso:	4,8

Lentelė 5 Sukurto šablono pakeitimo darbų laikai

Žemiau esančioje 4 lentelėje pateikti naujo šablono sukūrimo laikai, naudojant Expression Blend ir Visual Studio įrankius, XAML ir C# kalbų pagrindu.

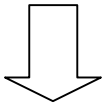
Rolė	Darbai	Laikas, val
Dizaineris	Vartotojo sąsajos pakeitimas	1,6
GUI testuotojas	Dizaino sulyginimas su pradiniu ir funkcionalumo testavimo darbai	-
	Testavimas su Firefox 2 naršykle (neturi reikšmės kokia naršyklė naudojama)	0,3
	Viso:	1,9

Lentelė 6 Sukurto šablono pakeitimo darbų laikai su Silverlight

3.7.3 Kūrimo modelių transformacija

Pagal atliktus laikų įvertinimus galima apibendrinti naujo šablono sukūrimo optimizavimo etapą, pereinant iš vieno kūrimo modelio į kitą.

	Dizaineris	Techninis dizaineris	Programuotojas	GUI testuotojas	Viso
Laikas, val	2,5	9,5	2	1,6	15,6



	Dizaineris	Programuotojas	GUI testuotojas	Viso
Laikas, val	6	2	0,3	8,3

Lentelė 7 Kūrimo modelio transformacija

Pavyko optimizuoti šablono kūrimo etapą, kurio realizavimo laikas sutrumpėjo nuo 15,6 val. iki 8,3 val. ir dingo techninio dizainerio rolė.

3.7.4 Apibendrinimas

Šio tyrimo metu pavyko optimizuoti kūrimo etapą pasirinktai probleminiai sričiai. Sumažėjo ne tik darbų atlikimo bendras laikas, tačiau ir dingo viena darbuotojo rolė – tai techninis dizaineris, be kurio dabartinių internetinių sistemų kūrimas neįsivaizduojamas. Tačiau šis konkretus pavyzdys nepavaizduoja bendros naujo kūrimo modelio naudos, nes kiekvienam konkrečiam sprendimui realizavimo laikai gali skirtis.

Pereinant prie Silverlight technologijos, pirmiau reikėtų gerai įvertinti kokiem tikslam bus naudojama sistema ir kokia linkme ji bus tobulinama. Jeigu numatomi pakeitimai ateityje yra minimalūs, tuomet neverta visos sistemos perkelti ant Silverlight platformos. Verčiau panaudoti tik tam tikrus sprendimus, kuriuos šia technologija būtų galima realizuoti paprasčiau ir greičiau arba ja realizuoti tik tai, ko neleidžia HTML, CSS ir JavaScript galimybės. Jei sistemą numatoma keisti radikaliai, tuomet vertėtų pasvarstyti ar neverta geriau nuo pamatų šią sistemą kurti su Silverlight technologija. Taip pat didelę sprendimo dalį turėtų sudaryti ir numatomas sistemos gyvavimo laikas, jos panaudojimo galimybės, rinka, į kurią sistema yra akcentuota.

3.8 Tolimesnio tobulinimo galimybės

Toks šablonų kūrimo modelis leidžia lengvai kurti naujus ir redaguoti esamus šablonus pagal kliento pageidavimus. Šio modelio pagalba iš karto gaunamas ir matomas rezultatas.

Realizavus tokį šablonų kūrimo modelį ir ištyrus, kad jis veikia, galima planuoti tolimesnį sistemos praplėtimą. Galima leisti patiems vartotojams susikurti šablonus. Vartotojui turėtų būti pateiktas vadovas, nurodant elementų struktūrą, jų vardus ir parodant kaip jis tai turėtų atlikti. Minusas, kad vartotojas privalėtų turėti XAML žinių. Tačiau taip būtų suteikta laisvė vartotojams susikurti savitus dizainus. Sukurtus dizainus jie galėtų sukelti į serverį ir vėliau juos panaudoti. Iki tol naudotos technologijos to neleidžia dėl keleto kliūčių. Visų pirma renginio duomenys yra dinaminiai ir jie užkraunami iš duomenų bazės, o vėliau įterpiami PHP kodo pagalba. Būtų visiškai nesaugu leisti vartotojams vykdyti PHP kodą sistemoje. Kita problema CSS stilių persidengimas. Norint sukurti raiškų dizainą, reikia naudoti ir stilius, kurie gali persidengti su jau esamais sistemos stiliais, dar prie stilių galima pridėti, kad gali tekti įkėlinėti ir paveiksliukus kaip atskirus vienetus. Tačiau su Silverlight pagalba tai realizuoti įmanoma, kadangi Silverlight programa vykdoma kliento naršyklės pusėje ir jos dizainas, esantis XAML faile yra atskirtas nuo sistemos logikos.

Visa tai veda prie metaprogramavimo realizavimo galimybės sistemoje. Naudojant Silverlight platformą galima kurti Silverlight technologija paremtas vartotojo sąsajas. Norint padaryti lengvą ir paprastą skirtingų šablonų kūrimo principą, reikėtų įgyvendinti jų vizualinį kūrimą naršyklėje. Pasinaudojant raiškia ir dinamine Silverlight sąsaja, galima būtų sukurti programą, kurios pagalba vartotojas galėtų kurti XAML tipo šablonus naršyklėje. Sukurti XAML struktūros šablonai galėtų būti panaudojami kaip renginių šablonai. Vartotojas galėtų susikurti naujus šablonus pačioje sistemoje ir iš karto juos panaudoti renginių aprašuose.

Kitas tobulinimo aspektas galėtų būti audio ir video medžiagos panaudojimas sistemoje. Kadangi Silverlight technologija palaiko tokio tipo multimediją, galima realizuoti filmukų ir prezentacijų panaudojimą Renginio apraše.

3.9 Perėjimo prie Silverlight platformos problemos

Iš karto pereiti prie Silverlight platformos nėra lengva, ypač jeigu programuotojai buvo gerai susipažinę su PHP kalbos pagrindu kuriamomis sistemomis, tačiau turėjo mažai arba iš viso neturėjo jokių žinių apie .NET platformą ir ja paremtas programavimo kalbas. Darbuotojai turėtų persikvalifikuoti arba gali prireikti naujų darbuotojų, turinčių daugiau žinių apie šias platformas. Jei programuotojai ir gali persikvalifikuoti, tai grafinių įrankių, kaip pvz. PhostoShop dizaineriams ar techniniams dizaineriams gali būti labai sunku persikvalifikuoti, nes šios technologijos iš pagrindų yra kitokios.

Kita problema būtų naujų technologijų įdiegimas įmonėje, žinant, kad jos yra mokamos ir reikia tikrai ne pigių ir galingų Microsoft įrankių.

4 Išvados

Darbo eigoje buvo susipažinta su jau esamais panašiais produktais ir jų sprendimais, įvertinti jų trūkumai ir privalumai. Išnagrinėtos naujausios technologijos, kurios naudojamos šiuolaikinėse informacinėse sistemose. Projektinėje dalyje trumpai aptarta jau sukurtos sistemos architektūra.

Išnagrinėta su kokiomis problemomis susiduriama, projektuojant Raiškiąsias internetines sistemas, ir kaip tų problemų išvengti ar bent sumažinti jų poveikį. Išanalizuotos ir tarpusavyje sugretintos Flash ir Silverlight technologijos. Dėl siūlomo vartotojo sąsajos sluoksnio atskyrimo nuo sistemos logikos, buvo pasirinkta Silverlight technologija.

Aptarti ir palyginti sistemos kūrimo ir tobulinimo procesų etapai. Tyrimas atliktas, praplečiant sistemos funkcionalumą, t.y. realizuotas naujų renginio šablonų kūrimo modelis su Silverlight. Sistemos kūrimo etapai supaprastėja ir sutrumpėja. Pakeitimų realizavimas tampa paprastas tiek vartotojo sąsajos lygmenyje, tiek sistemos logikos lygmenyje. Eksperimento metu pavyko optimizuoti kūrimo procesų modelį pasirinktai probleminiai sričiai. Pagal pritaikytus Silverlight programų kūrimo principus sumažėjo modulio kūrimo laikas ir buvo eliminuota techninio dizainerio rolė.

Silverlight technologijos dėka, sistema tampa nepriklausoma nuo platformos ir naršyklės. Nebereikia sistemos pritaikyti skirtingoms naršyklėms. Tačiau vartotojai privalo įsidiegti Silverlight įskiepį savo kompiuteriuose.

XAML padeda dizaineriui ir programuotojui artimiau bendradarbiauti, o tuo pačiu struktūriškai atskiria sistemos vartotojo sąsajos sluoksnį nuo sistemos logikos sluoksnio. Naudojamas XAML principas leidžia užsaugoti tam tikrus dizaino elementus ar jų grupes kaip objektus, kuriuos galima panaudoti kituose projektuose. Elementų pakartotinis panaudojimas sutaupo daug laiko kuriant naujas sistemas.

Pasinaudojant Silverlight technologija, galima kurti raiškias renginių prezentacijas, o tai sistemą perkelia į aukštesnį lygmenį, ko neleido prieš tai naudotos technologijos. Po atlikto tyrimo nustatytos tolimesnio sistemos tobulinimo gairės, panaudojant Silverlight technologiją.

Renginių organizavimo ir valdymo sistema sukurta ir įdiegta užsakovui UAB „Inify“. Šio dokumento prieduose pateikiamas „Sistemos įdiegimo aktas“. Veikiančią internetinės sistemos versiją galima išbandyti apsilankius adresu: www.invup.lt

5 Literatūra

[1] Rajesh Sumra and Arulazi, Quality of Service for Web Services — Demystification, Limitations, and Best Practices [žiūrėta 2007 10 28] Prieiga per internetą: http://www.developer.com/services/article.php/10928_2027911_1

[2] Deepak Sharma, Simplicity is the heart of effective web design [žiūrėta 2007 10 28] Prieiga per internetą: http://www.visualbuilder.com/viewpages.php?art_id=30579&pageorder=1

[3] User Vision, Usability in web development [žiūrėta 2007 10 28] Prieiga per internetą: http://www.uservision.co.uk/usability_articles/usability_usability2.asp

[4] Arulazi D, Quality of Service for Web Services [žiūrėta 2007 10 28] Prieiga per internetą: http://www.developer.com/services/article.php/10928_2027911_2

[5] Xavier Spriet, Real-World PHP Security [žiūrėta 2007 10 28] Prieiga per internetą: http://www.linuxjournal.com/article/7237

[6] Caleb Sima, Security Risk Assessment and Management in Web Application Security [žiūrėta 2007 10 28] Prieiga per internetą: http://www.spidynamics.com/spilabs/education/articles/Security-risk-assessment.html

[7] Webcredible, Web accessibility: The basics [žiūrėta 2007 10 28] Prieiga per internetą: http://www.webcredible.co.uk/user-friendly-resources/web-accessibility/basics.shtml

[8] W3C [žiūrėta 2007 10 28] Prieiga per internetą: http://www.w3.org/TR/WCAG10-HTML-TECHS/#index-elements

[9] Ralf S. Engelschall, Load Balancing Your Web Site [žiūrėta 2007 10 28] Prieiga per internetą: http://www.webtechniques.com/archives/1998/05/engelschall/

[10] Vince Barnes, Web Site Maintainability [žiūrėta 2007 10 28] Prieiga per internetą: http://www.htmlgoodies.com/introduction/intro/article.php/3473641

[11] Michele Neylon, Sending bulk email without spamming [žiūrėta 2007 10 28] Prieiga per internetą: http://www.mneylon.com/blog/archives/2005/01/17/mass-emailing-dos-and-donts/

[12] George Dillon, HTML e-mail is STILL evil!!! [žiūrėta 2007 10 28] Prieiga per internetą: http://www.georgedillon.com/web/html_email_is_evil.shtml

[13] Mark Wood, SMS Bulk Messaging, the problem and the solution [žiūrėta 2007 10 28] Prieiga per internetą: http://www.ceasa-int.org/library/7_sms_mass_messaging_problems_V1-2.doc

[14] Balianti, Multilanguage Web Design [žiūrėta 2007 10 28] Prieiga per internetą: http://www.balianti.com/servicedetail.asp?sid=27&bg=20

[15] Eionet, Benefits of a multilingual website [žiūrėta 2007 10 28] Prieiga per internetą: http://www.eionet.europa.eu/software/design/multilinguality/benefits

[16] jQuery JavaScript Library [žiūrėta 2007 10 28] Prieiga per internetą: http://docs.jquery.com/How_jQuery_Works

[17] What Is Web 2.0, Design Patterns and Business Models for the Next Generation of Software [žiūrėta 2009 02 20] Prieiga per internetą: http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html

[18] RIA War Is Brewing [žiūrėta 2009 03 19] Prieiga per internetą: http://etech.eweek.com/content/application_development/ria_war_is_brewing.html

[19] A Convergence of User Interface Paradigms of Web and Desktop [žiūrėta 2009 03 19] Prieiga per internetą: http://www.flomedia.de/diploma

[20] Software CEO, Develop faster and cheaper, from client/server to RIA to mobile to SaaS [žiūrėta 2009 03 19] Prieiga per internetą: http://www.softwareceo.com/products_services/hp_article.aspx?arttype=TT&page=0

[21] Which Rich Internet Application (RIA) Technology Will Lead The Pack [žiūrėta 2009 03 17] Prieiga per internetą: http://www.betadaily.com/2008/03/07/which-rich-internet-application-ria-technology-will-lead-the-pack/

[22] Macromedia March 2002 requirements for Rich Internet Applications [žiūrėta 2009 03 18] Prieiga per internetą: http://download.macromedia.com/pub/flash/whitepapers/richclient.pdf
[23] Usability for Rich Internet Applications [žiūrėta 2009 03 22] Prieiga per internetą: http://www.digital-web.com/articles/usability_for_rich_internet_applications/
[24] Microsoft Silverlight - [žiūrėta 2009 03 22] Prieiga per internetą: http://en.wikipedia.org/wiki/Silverlight
[25] Microsoft implementation studies - [žiūrėta 2008 12 14] Prieiga per internetą: http://www.msuniversity.edu.cn/m_RepositoryIntro/Detail.aspx?id=637
[26] Get Started : The Official Microsoft Silverlight Site - [žiūrėta 2009 03 14] Prieiga per internetą: http://silverlight.net/GetStarted/
[27] Counting Primes in Flash and Silverlight - [žiūrėta 2009 03 02] Prieiga per internetą: http://www.itwriting.com/primetest/index.html
[28] Counting Primes in JavaScript - [žiūrėta 2009 03 02] Prieiga per internetą: http://www.itwriting.com/blog/counting-primes-in-flash-and-silverlight
[29] Microsoft Silverlight: 10 reasons to love it, 10 reasons to hate it - [žiūrėta 2009 03 02] Prieiga per internetą: http://www.theregister.co.uk/2008/08/18/silverlight_pros_and_cons/page2.html
[30] Scott Guthrie: Silverlight plans go beyond Ajax and Flash - [žiūrėta 2009 03 02] Prieiga per internetą: http://arstechnica.com/microsoft/news/2009/02/scott-guthrie-silverlight-plans-go-beyond-ajax-and-flash.ars
[31] Microsoft, Adobe wage verbal battle over RIAs- [žiūrėta 2009 03 22] Prieiga per internetą: http://www.sdtimes.com/link/33291
[32] XAML – the future of the designer developer workflow - [žiūrėta 2009 04 03] Prieiga per internetą: http://differentthings.wordpress.com/2007/01/19/xaml-the-future-of-the-designer-developer-workflow/
[33] Flash Vs Silverlight - [žiūrėta 2009 03 02] Prieiga per internetą: http://www.wearealight.com/Blog/?tag=/ria
[34] Rich Internet Applications The Next Stage of Application Development, Farrell, J.; Nezlek, G.S. Information Technology Interfaces, 2007. ITI 2007.
[35] Model View Controller [žiūrėta 2007 10 28] Prieiga per internetą: http://www.phpwact.org/pattern/model_view_controller
[36] Roger L. Costello, Building Web Services the REST Way [žiūrėta 2007 10 28]
[37] Sparx systems, The Unified Modeling Language [žiūrėta 2007 10 28] Prieiga per internetą: http://www.sparxsystems.com.au/platforms/uml.html
[38] Scott W. Ambler, Introduction to the Diagrams of UML 2.0 [žiūrėta 2007 10 28] Prieiga per internetą: http://www.agilemodeling.com/essays/umlDiagrams.htm
[39] Doug Hughes, What Is UML? [žiūrėta 2007 10 28] Prieiga per internetą: http://www.alagad.com/go/blog-entry/what-is-uml
[40] James Bach, Good Practice Hunting [žiūrėta 2007 10 28] Prieiga per internetą: http://www.satisfice.com/articles/good_practice_hunting.pdf
[41] Ram Chillarege, Software Testinf Best Practices [žiūrėta 2007 10 28] Prieiga per internetą: http://www.chillarege.com/authwork/TestingBestPractice.pdf
[42] Gopal Gupta, Language Based Software Engineering [žiūrėta 2007 10 28] Prieiga per internetą:http://www.utdallas.edu/~gupta/alp.pdf
[43] Patrick Freudens7.2, Jan Buck, Martin Nussbaumer ir Martin Gaedke, Model-driven Construction of Workflow-based

6 Terminų ir santrumpų žodynas

Sąvoka	Apibrėžimas
.NET Framework	Microsoft.NET Framework yra Microsoft Windows operacinės sistemos komponentas, sukurtas 2002 metais. Jis suteikia kitoms programoms galimybę naudotis daugybe jau paruoštu įvairių bibliotekų (pvz., duomenų bazių komponentus, formų komponentus...).
AJAX (Asynchronous JavaScript And XML)	Asinchroninis Javascript ir XML
Apache	Tai yra atviro kodo tinklo serverio programa skirta įvairioms operacinėms sistemoms, kad palaikytų tinklapių veikimą fiziniuose serveriuose.
API (Aplication program interface)	Aibė šablonų bibliotekose, kurios išplečia kalbos funkcionalumą.
ASP.NET	Tai tinklalapio struktūros technologija parduodama Microsoft, kurią programuotojai gali naudoti norėdami sukurti dinaminę internetinę svetainę, žiniatinklio konstrukciją arba paslaugą. Tai dalis Microsoft .NET platformos Microsoft Aktyvių Serverio Puslapių (angl. "Active Server Pages") (ASP.net) technologijos įpėdinis.
CSS (Cascading style sheets)	Stiliaus lentelės, naudojamos kartu su HTML. Leidžia aprašyti HTML elementų stilių atskirose lentelėse.
Karkasas (framework)	Karkasas yra pakartotinio panaudojimo programinė įranga, skirta programinės įrangos sistemoms kurti. Karkasas gali turėti pagalbinių programų, kodo bibliotekų, scenarijų kalbų ar kitokių pagalbinių įrankių, kurių pagalba galima plėtoti ir sudėti skirtingus komponentus į vieną pilnai veikiančią sistemą.
DBVS (Duomenų Bazių Valdymo Sistema)	Kompiuterinė programa ar programų paketas, skirtas duomenų bazės valdymui.
DOM (Document Object Model)	Medžio tipo struktūra, laikanti visus HTML objektus. Naudojama Javascript ir kitų scenarrijų kalbų, kad dinamiškai formuoti ir keisti HTML puslapius.
DUK	Dažnai užduodami klausimai
GUI (Graphic User Interface)	Grafinė vartotojo sąsaja, dažnai dar naudojamas sutrumpintas variantas UI - vartotojo sąsaja.
Hostingas	Hostingas - internetinių paslaugų grupė, apimanti puslapių talpinimą, elektroninio pašto servisą ir kita. Hostingas reiškia tiek paslaugą (tuo atveju, kai tai atlieka firma, kuri teikia hostingo paslaugas), tiek patį procesą, tai fizinė patalpa, serveris (kompiuteris) ir interneto puslapio priežiūra.
HTML	Hiperteksto programavimo kalba: žymių ir taisyklių aibė skirta statinių www tinklapių kūrimui.
Internetinė naršyklė	Naršyklė (angl. browser) yra programa, skirta sklaidyti tinklo serverių (angl. web server) pateikiamą informaciją, dažniausiai naršant žiniatinklyje arba vidiniuose įmonės tinkluose (intranet). Interneto puslapiai gali turėti nuorodas į kitus puslapius ir naudotojas gali greitai ir lengvai pasiekti informacija sekdamas šiomis nuorodomis.
Iteracinis kūrimo metodas	Tai ciklinis sistemų kūrimo metodas, išsivystęs iš gerai žinomo krioklio metodo silpnumų. Jis prasideda nuo pradinių planavimų ir baigiasi sistemos pridavimu, tačiau tarp šių veiksmų yra iteratyviai kartojami sistemos kūrimo metodai, kol gaunamas norimas rezultatas
MySQL	MySQL - viena iš reliacinių duomenų bazių valdymo sistemų (liet. santrumpa RDBVS, angl. - RDBMS), palaikanti daugelį naudotojų, dirbanti SQL kalbos pagrindu. MySQL yra atviro kodo programinė įranga.

MVC (Model-view-controller)	Model-view-controller (MVC) yra architektūrinis šablonas naudojamas programinės įrangos kūrimui ir plėtojimui. Sudėtingose sistemose, kuriuos išvedinėja didžiulį duomenų kiekį vartotojams, dažnai norima atskirti duomenis (modelį) nuo vartotojo sąsajos (vaizdo). Toks išskyrimas neįtakoja sistemos valdomų duomenų, norint pakeisti vartotojo sąsają. Šiuos du komponentus susieja kontroleris, kuris yra tarsi "smegenys".
Operacinė sistema	Operacinė sistema (OS) - tai speciali programinė įranga, abstrahuojanti naudotojo bei programų darbą. Moderniausios operacinės sistemos sudaro galimybę dirbti daugeliui vartotojų vienu metu daugialypėje aplinkoje, užtikrina bylų (failų) apsaugą, turi daug kitų naudingų savybių. Dauguma operacinių sistemų yra pirma programinė įranga, kurią pradeda vykdyti įjungtas kompiuteris.
Panaudos atvejis	Panaudos atvejis yra išorinis sistemos vaizdas, kuris parodo tam tikrus veiksmus, kuriuos naudotojas turi atlikti, norėdamas įvykdyti tam tikrą užduotį. Kiekvienas panaudos atvejis yra sudarytas iš scenarijų. Kiekvienas scenarijus yra žingsnių seka, kuri nusako tarpusio sąveiką tarp naudotojo ir sistemos.
PHP	PHP - plačiai paplitusi dinaminė interpretuojama programavimo kalba, sukurta 1997 m. ir specialiai pritaikyta svetainių kūrimui. PHP sintaksė panaši į daugelį struktūrinių kalbų, ypatingai į C bei Perl. PHP kalba yra atviro kodo ir tai yra viena priežasčių, dėl ko kalba yra nors ir nesudėtinga, bet gana lanksti - veikia daugumoje operacinių sistemų, palaiko nemažai reliacinių duomenų bazių bei veikia su dauguma interneto serverių - CGI, FastCGI, ISAPI ir kitais protokolais.
Programinė įranga	Programinė įranga tai kompiuterio vykdomų instrukcijų seka, skirta tam tikriems veiksmams atlikti. Dažniausiai tokia įranga parašoma naudojant programavimo kalbas, o vėliau kompiliuojant ar interpretuojant parašytą kodą.
PSD (PhotoShop Document)	Formatas PSD – specifinis Adobe programos formatas, leidžiantis išsaugoti idealios kokybės visus vaizdų elementus, kurie buvo sukurti šia programa.
RIA (Rich Internet application)	Raiškiosios internetinės sistemos – tai tokios internetinės sistemos, kurios turi taikomųjų kompiuterinių programų charakteristikų, kurios veikia internetinių naršyklių įskiepiu pagalba arba veikiančios nepriklausomai virtualiosiose mašinose
REST (Representational State Transfer)	Programinės įrangos architektūros stilius
Serveris	Specialios paskirties kompiuteris, skirtas kitų kompiuterių aptarnavimui (neskirtas asmeniniam naudojimui). Dažniausiai išskiriamos kelios serverių rūšys (pagal paskirtį): failų serveriai (FTP, SMB, etc.), pašto serveriai (POP3, SMTP, IMAP), tinklalapių serveriai (angliškai: web server [host]) (HTTP), duomenų bazių serveriai (SQL), aplikacijų serveriai, ir pan. Daugeliu atvejų tas pats kompiuteris gali atlikti visas šias funkcijas. Serverių naudojimas pagrįstas client-server architektūra: klientas (kompiuteris ar programa) prisijungia prie serverio, skirto tam tikrai užduočiai, ir siunčia pastarąjam įsakymus, kurie vykdomi.
Silverlight	Silverlight – tai nepriklausantis nuo platformos ir naršyklės įskiepis, leidžiantis web-programuotojams ir dizaineriams kurti šiuolaikines dinamines web-aplikacijas .NET aplinkoje.
Sistema	Pakvietimų į organizuojamus renginius planavimo ir valdymo informacinė sistema.
SOAP (Service Oriented Architecture Protocol)	Tai XML tipo žinučių perdavimo protokolas.
Techninė įranga	Kompiuterių techninė įranga apima visas fizines kompiuterio dalis, bet ne programinę įrangą, valdančią šias dalis.
UML (Unified Modeling Language)	Vieninga modeliavimo kalba– modeliavimo ir specifikacijų kūrimo kalba, skirta specifikuoti, atvaizduoti ir konstruoti objektiškai orientuotų programų dokumentus.

URL (Uniform Resource Locator)	Universalus resursų lokatorius, standartizuota simbolių eilutė, kuri nurodo resurso vietą žiniatinklyje.
W3C (World Wide Web Consortium)	W3C - tai konsorciumas, leidžiantis programinės įrangos standartus („rekomendacijas“, kaip jie jas vadina) žiniatinkliui (angl. World Wide Web). Vadovauja Tim Berners-Lee, sukūręs URL (Uniform Resource Locator), HTTP (HyperText Transfer Protocol) ir HTML (HyperText Markup Language), interneto technologinį pagrindą.
Web 2.0	Antrosios kartos internetas – specifinis bendradarbiavimas tarp žiniatinklio vartotojų (pvz., keitimasis informacija), kuriam būdingos sparčiai besikeičiančios komunikacijos internete bei žiniatinklio technologijos, specifinis dizainas, atitinkamos tendencijos, „internetinis“ kūrybiškumas.
WSDL (Web Service Description Language)	Interneto paslaugas apibūdinti kalba, paremta XML sintakse.
WWW (World Wide Web)	Žiniatinklis, pasaulinis tinklas (angl. World Wide Web arba WWW) – interneto dalis, resursai, kuriuos internete galima pasiekti naudojant URL (Vieningus Resursų Identifikatorius). Dėl savo naudojimo platumo jis dabar yra neretai supainiojamas su visu internetu apskritai, tačiau tai yra tik interneto poaibis.
XAML (eXxtensible Application Markup Language)	XAML – tai XML standartu apibrėžta kalba, ji sukurta Microsoft kompanijos ir yra naudojama įvairiems objektams ir jų reikšmėms struktūriškai apibrėžti.
XML (eXtensible Markup Language)	W3C rekomenduojama bendros paskirties duomenų struktūrų bei jų turinio aprašomoji kalba.

Lentelė 8 Terminų ir santrumpų žodynas

7 Priedai

1 priedas. Sistemos įdiegimo aktas

SISTEMOS ĮDIEGIMO PATVIRTINIMO AKTAS
2009 m. gegužės mėn. 15 d.
Vilnius

Inify, UAB, įmonės kodas 300655539, buveinės adresas Maumedžių g. 19-17, LT – 08420 Vilnius, pagal įstatus atstovaujama direktoriaus Manto Žvirono, toliau vadinama „**Užsakovu**“, patvirtina, kad Egidijus Gegeckas ir Domas Greičius, toliau vadinami „**Autoriais**“, sėkmingai įdiegė ir perdavė Renginių organizavimo ir valdymo sistemą **Užsakovui**, kurią galima pasiekti internete adresu: www.invup.lt

Užsakovas

Inify, UAB
Įm.k. 300655539
Maumedžių g. 19-17, LT – 08420 Vilnius
Direktorius Mantas Žvironas

(parašas)