KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Paulius Gelažius

Viešų finansinių duomenų automatizuoto surinkimo ir panaudojimo tyrimas

Magistro darbas

Vadovas: doc. dr. E. Karčiauskas

KAUNAS, 2009

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Paulius Gelažius

Viešų finansinių duomenų automatizuoto surinkimo ir panaudojimo tyrimas

Magistro darbas

Vadovas: doc. dr. E. Karčiauskas
2009 – 05 – 22

Recenzentas: prof. dr. R. Butleris
2009 – 05 – 25

Atliko: IFM-3/2 gr. studentas
Paulius Gelažius
2009 – 05 – 22

Kaunas, 2009

TURINYS

LENTELIŲ IR PAVEIKSLŲ TURINYS

SUMMARY

Nowadays technology evolves at a very fast pace. The information amount generated and used by people through all this technological progress is humongous. Every single day the information keeps adding up and up, thus we face a big problem of these times – ineffective usage of information. To partially solve this problem and to use all the provided information more effectively, first of all we must be able to collect it automated way.

In my master thesis work I've done a research of existing data collection model. Described in details the data publishing, analysis and storage possibilities Which includes the ways information is provided to public use, web scraping techniques and XML analysis, verification and transformation possibilities. This research goal was to find the possibilities of creating Lithuania's banks financial data automated collection. Research outcome has proven that best way to do that is by using web scraping techniques on data provided by banks information systems.

As a result of the research, web financial calculator system was created to prove the possibility of automated data collection and usage in Lithuania. A system that consists of services for gathering 3 different types of data from 9 biggest Lithuania's banks and 3 financial calculators operating with that particular data.

1. ĮVADAS

Spartus informacinių technologijų tobulėjimas ir plėtimasis suteikia žmonėms galimybes skleisti informaciją be jokių apribojimų visas įmanomais būdais. Su kiekviena diena paskleidžiamas informacijos kiekis auga milžinišku tempu. Didžioji dalis šios informacijos yra perkeliama į pasaulinį kompiuterių tinklą – internetą. Norint internete sėkmingai surasti ir panaudoti mus dominančią informaciją, kuriamos duomenų indeksavimo ir paieškos sistemos. Stengiamasi kuo labiau automatizuoti duomenų surinkimą. Tačiau ir šios sistemos nevisada randa tai ko ieškome. Taigi iškyla aktuali informacinių sistemų problema – automatizuotas duomenų surinkimas, analizė, struktūrizavimas ir sėkmingas panaudojimas.

Informacija gali būti įvairių tipų, todėl šiame darbe bus tiriama tik viešos finansinės informacijos ir iš jos gautų formalių duomenų surinkimas ir panaudojimas. Terminas „finansiniai duomenys“ taip pat apima labai plataus spektro ir įvairių tipų finansinę informaciją. Tai gali būti informacija apie asmenis, įmones ar valstybes, todėl yra būtina apibrėžti šio termino ribas vykdomam tyrimui. Magistriniame darbe „finansiniais duomenimis“ bus laikomi bankų finansiniai duomenys pateikiami bankų informacinėse sistemose (toliau tekste IS). Kadangi šie duomenys yra laisvai be apribojimų prieinami visiems bankų IS lankytojams, todėl juos vadinsime viešais. Interneto erdvę, kurioje yra talpinamos šios informacinės sistemos vadinsime viešąja erdve.

Duomenų surinkimas kaip funkcija yra reikalingas norint atlikti bankų finansinių duomenų analizę, paslaugų palyginimą ar panaudoti teikiamus duomenis įvairiems statistiniams skaičiavimams. Viešoje erdvėje pateikiamų duomenų surinkimas gali būti vykdomas tiek rankiniu, tiek ir automatiniu būdu. Mūsų tyrime bus skiriamas dėmesys automatiniam surinkimui. Darbe išnagrinėsime galimus duomenų surinkimo būdus, jų priklausomybę nuo duomenų pateikimo būdų, bei pažymėsime privalumus ir trukumus, lyginant su rankiniu duomenų surinkimu.

Darbo tikslas yra išnagrinėti galimybes automatiškai surinkti ir panaudoti analizei bei skaičiavimams Lietuvos bankų teikiamus finansinius duomenis. Darbe bus koncentruojamasi išsiaiškinti ir aprašyti duomenų surinkimo modelį, bei jo dalis. Kadangi duomenų surinkimas priklauso nuo to kokiais būdais jie yra pateikti, išnagrinėsime naudojamus duomenų pateikimo internete būdus, tokius kaip interneto svetainės, sklaidos kanalai, skaitmeniniai dokumentai ir k.t. Taip bus aptarta kokios technologijos bei standartai aktualūs šiai sričiai. Ištirsime egzistuojančius duomenų pateiktų HTML ir XML formatais apdorojimo būdus. Taip pat aprašysime kokie yra galimi šių automatizuotai surenkamų finansinių duomenų panaudojimo būdai.

Šiame darbe tiriamos galimybės yra panaudotos realizuojant Lietuvos bankų finansinių paslaugų palyginamųjų skaičiuoklių sistemą. Išnagrinėjus kokiais būdais Lietuvos bankai teikia duomenis savo informacinių sistemų lankytojams, buvo realizuotas automatinis duomenų surinkimas ir tų duomenų panaudojimas kuriant šią skaičiuoklių IS.

Darbo pabaigoje yra pateikiamos išvados, nurodoma naudota literatūra, bei terminų ir santrumpų žodynas.

1.1.Darbo tikslas ir uždaviniai

Pagrindinis darbo tikslas yra išanalizuoti galimybes automatizuotai surinkti Lietuvos bankų teikiamus viešus finansinius duomenis ir juos praktiškai panaudoti. Šis darbas yra atliekamas norint ištirti viešoje erdvėje egzistuojančius automatizuoto duomenų surinkimo ir panaudojimo būdus.

Pagrindiniai darbo uždaviniai:

- Aprašyti automatizuoto duomenų surinkimo procesą
- Išanalizuoti esamus finansinių duomenų pateikimo būdus viešoje erdvėje
- Išanalizuoti esamus finansinių duomenų apdorojimo būdus
- Išanalizuoti esamus apdorotų finansinių duomenų saugojimo būdus
- Aprašyti galimus surinktų duomenų panaudojimo būdus
- Ištirti kokais būdais Lietuvos bankai teikia duomenis ir kokios yra galimybės juos automatiškai surinkti
- Realizuoti informacinę sistemą, automatiškai surenkačią duomenis ir juos panaudojančią skaičiavimams.

2. DUOMENŲ SURINKIMO IR PANAUDOJIMO ANALIZĖ

2.1.Duomenų analizė

Dažnai kalbant apie duomenis informatikos kontekste yra maišomos dvi savokos: duomenys ir informacija. Aiškumo dėlei apibrėžime kiekvieną iš jų. *Informacija* - žinios apie faktus, ivykius, daiktus, procesus, idejas, savokas ir kitus objektus, kurios kuriame nors kontekste turi kokia nors prasme [1]. *Duomenys* - formalizuotas informacijos vaizdinys, tinkamas perduoti kitiems, suvokti ir apdoroti. Duomenys gali būti apdorojami neautomatizuotai arba panaudojant automatines priemones [1]. Kalbant bendrai apie informacinės sistemos turinį vartosime terminą „informacija“, o kalbant kaip apie formalų surinkimo objektą vartosime „duomenys“.

Viešoje erdvėje pateikiami duomenys gali būti struktūrizuoti arba nestruktūrizuoti. *Struktūrizuoti duomenys* – tai duomenys turintys tam tikrą modelį. Duomenų modelis formaliai apibūdina duomenų elementus ir saryšius tarp šių elementų pasirinktai sričiai [2]. Pvz.:

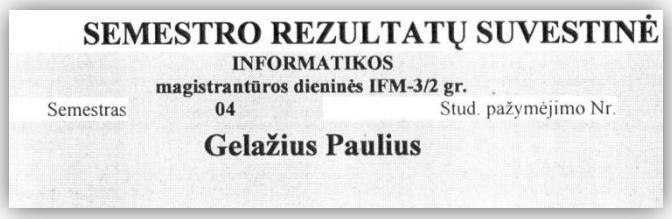
```
<Studentas>
     <Vardas>Paulius</Vardas>
     <Pavardė>Gelažius</Pavardė?
     <Grupė>IFM-3/2</Grupė>
</Studentas>
```

Pav. 1 Struktūrizuotų duomenų pavyzdys (XML dokumento fragmentas)

Nestruktūrizuoti duomenys – tai duomenys neturintys modelio arba turintys modelį, kurį sudėtinga panaudoti programinėje įrangoje [3]. Pvz.:

```
IFM-3/2 gr. studentas
Paulius Gelažius
```

Pav. 2 Nestruktūrizuotų duomenų pavyzdys (Tekstinio failo fragmentas)


SEMESTRO REZULTATŲ SUVESTINĖ
INFORMATIKOS
magistrantūros dieninės IFM-3/2 gr.
Semestras 04 Stud. pažymėjimo Nr.
Gelažius Paulius

Pav. 3 Nestruktūrizuotų duomenų pavyzdys (Paveiksliuko fragmentas)

Nestruktūrizuoti duomenys susideda iš 2 paprastų kategorijų:

- Taškinės grafikos objektų (3 pav.)
- Teksto objektų (2 pav.)

Dauguma šių dienų technologijų nukreiptos į teksto objektus [2].

Kadangi tiriamos finansinės institucijos duomenis pateikia ne grafikos objektų forma, o tekstu, todėl magistriniame darbe bus tiriamas nestruktūrizuotų tekstinių duomenų automatinis surinkimas, struktūrizavimas ir panaudojimas.

2.2.Automatizuoto duomenų surinkimo aktualumas

Norint surinkti finansinę informaciją rankiniu būdu reikėtų aplankyti kiekvienos finansinės institucijos, kurios duomenys jums reikalingi, informacinę sistemą su interneto naršykle ir perrašyti arba nukopijuoti ir įterpti duomenis į saugyklą, priklausomai nuo to kokiu būdų bus saugojami šie duomenys. Jei vedama į duomenų bazę naudojant formomis paremtą programinę įrangą (toliau tekste PĮ), reikės suvesti kiekvieną lauką atskirai. Jeigu tokiu duomenų kiekis yra didelis, šis surinkimo procesas taps neefektyvus, nes užims labai daug laiko. Jeigu duomenys bus saugojami skaitmeniniuose dokumentuose nukopijavus ir įterpus, tai užims mažiau laiko lyginant su saugojimu suvedant duomenis, tačiau šiuo būdų išsaugoti duomenys bus nestruktūrizuoti ir neparuošti tolimesniam jų panaudojimui.

Norint surinkti finansinę informaciją automatiniu būdų reikalinga programinė įranga, kuri, priklausomai nuo pateikimo formos parsiųstų reikiamus duomenis, imituodama rankinį naršyklės apsilankymą ar kitu būdų, atliktų parsiusto turinio analizę ir išgavus duomenis išsaugotų juos struktūrizuotoje formoje. Tokiu būdų galima automatiškai surinkti duomenis iš daugelio finansinių institucijų žiniatinklių. Tačiau, kad galima būtų tai atlikti reikalinga programinė įranga puslapių turinio parsiuntimui ir saugojimui kiekvienam prieigos prie informacijos būdui, bei turinio analizatoriai kiekvienai skirtingai duomenų aibei.

Apibendrinant abu būdus svarbu pasakyti, kad automatinis surinkimas yra naudingas ir efektyvus tik tuomet jei surenkami duomenys keičiasi palyginti dažnai. Pvz.: (Kas dieną, kas valandą ar net kas minute ar sekundę). Jei duomenys keičiasi kas savaitę ar rečiau, reikia gerai išanalizuoti ar tikrai reikalinga šiuos duomenis surinkti automatiškai, kadangi kiekvienas automatizuotas procesas reikalauja papildomų lėšų ir priežiūros. Šiame magistriniame darbe tiriame bankų finansinių duomenų, kurie keičiasi sąlyginai dažnai surinkimą, todėl šio proceso automatizavimas mums yra labai aktualus.

2.3.Duomenų surinkimo modelis

Tikslus duomenų surinkimo modelis tiesiogiai priklauso nuo to kokiu būdu pateikiamus duomenis rinksime. Todėl pirmiausia abstrakčiai apibrėšime neperiodinį ir periodinį duomenų surinkimo modelius.

Taigi duomenų surinkimas susideda iš šių pagrindinių dalių: a) duomenų paėmimas b) duomenų apdorojimas c) duomenų išsaugojimas.

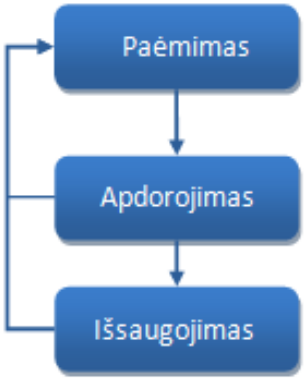
Duomenų tiekėjo funkciją atlieka minėtos bankų informacinės sistemos. Duomenų apdorojimo dalyje atliekamas duomenų validavimas ir struktūrizavimas. Jei apdorojimas buvo sėkmingas, pereinama į išsaugojimo dalį, kur jau struktūrizuotus duomenis galima išsaugoti pasirinktu būdų (Duomenų bazė, Standartus atitinkantys formatai ir pan.).

Neperiodinio duomenų surinkimo atveju modelis būtų tiesinis (6 pav.)

Pav. 4 Neperiodinio duomenų surinkimo modelis

Šis modelis apibūdina vienkartinį duomenų surinkimą. Dažniausiai tai būna atliekama rankiniu būdu suvedant pradinę nekintamą duomenų aibę. Tokio modelio panaudojimo pavyzdys galėtų būti buhalterinės sistemos, kurios skačiavimams naudoja pastovius labai retai kintančius ekonominius koeficientus. Tokiais atvejais suvedimą reikia atlikti tik sudarant pradinę duomenų aibę. Pvz.: gyventojų mokesčių tarifai, PVM ir pan.

Periodiško duomenų surinkimo modelis skiriasi nuo tiesinio tuo, kad dalis procesų vykdomi pastoviais PĮ numatytais laiko tarpais. Šis modelis labiausiai tinka automatizuotam duomenų surinkimui, nes periodiškumas pabrėžia automatizavimo svarbą. Kuo mažesnis periodas tarp surinkimų, tuo daugiau naudos automatizavimas suteikia.



Pav. 5 Periodinio duomenų surinkimo modelis

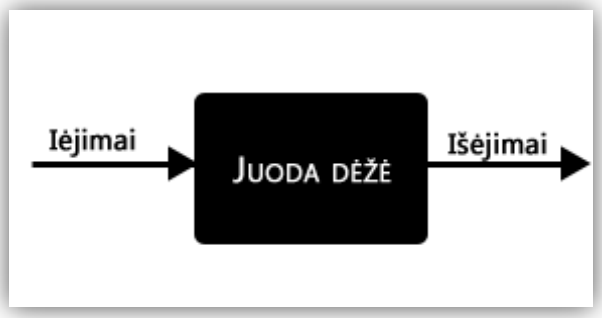
Jei naujų duomenų nėra, jie neatitinka kriterijų reikalingų saugoti arba gauti duomenys negali būti apdorojami, tokiu atveju užfiksuojama klaida, praleidžiamas išsaugojimas ir pradedamas naujas užklausų ciklas. Jei duomenys apdoroti sėkmingai ir tinkami saugojimui, pereinama į saugojimo fazę. Išsaugojus struktūrizuotus duomenis, vėl pereinama prie naujų duomenų užklausų.

Toks būtų automatizuoto duomenų surinkimo modelis, kuris ir yra aktualus mūsų darbe. Toliau detaliai išanalizuosime kiekvieną iš aprašyto proceso dalių.

2.4.Duomenų pateikimo būdų viešoje erdvėje analizė

Duomenų surinkimo būdas ir atlikimo efektyvumas priklauso nuo to, kokiu būdu yra pateikiami duomenys, kuriuos norime surinkti. Todėl yra labai svarbu išnagrinėti kokia forma ir būdais jie yra pateikiami interneto (Web) erdvėje.

Finansinius duomenis teikiančioms informacinėms sistemoms galime taikyti programų inžinerijoje gerai žinomą „juodos dėžės“ modelį (5 pav.), kuris turi 3 sudedamąsias dalis: a) juodą dėžę b) įėjimus c) išėjimus



Pav. 6 Juodos dėžės modelis

Mūsų tiriamu atveju juoda dėžė reprezentuoja informacinę sistemą, iš kurios gauname duomenis. Įėjimais vadiname užklausas sistemos duomenis gauti, o išėjimai būtų sistemos teikiami duomenys, kurių užklausėme.

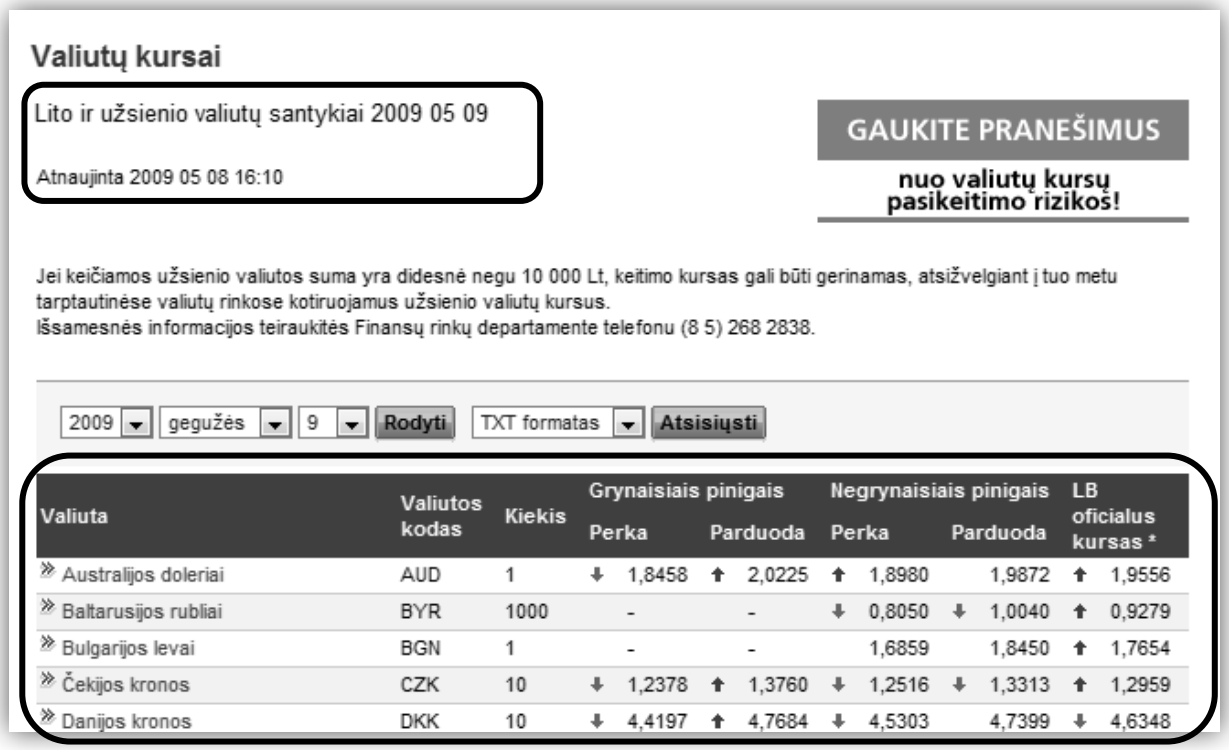
Viešoje erdvėje duomenys pateikiami šiais pagrindiniais būdais:

- Interneto puslapyje
- Skaitmeniniais dokumentais
- Sklaidos kanalais
- Interneto aplikacijų programavimo sąsaja

2.4.1. Duomenų pateikimas interneto puslapyje

Interneto puslapis yra web serveryje informaciją laikantis dokumentas. Šis dokumentas gali būti sukurtas rankiniu būdu arba sugeneruotas specialios programinės įrangos. Dokumento pagrindą sudaro hiperteksto žymėjimo kalba aprašyta informacija. Nors dokumentas gali būti sugeneruotas įvairiomis web serverio pusėje naudojamomis technologijomis (ASP.NET, PHP, JSP ar k.t.) tačiau dokumento turinys vistiek bus aprašytas HTML. Šią žymių kalbą įkūrė ir palaiko pasaulinio tinklo konsorciumas W3C. Kadangi šios kalbos pagrindu veikia visi interneto tinklalapiai, duomenų pateikimas šiuo būdu yra populiariausias.

Interneto tiklalapyje skelbiama informacija yra nestruktūrizuota. Ji pateikiama teksto blokais arba lentelėmis (7 pav. Išskirtini blokai apibrėžti).



Valiutų kursai

Lito ir užsienio valiutų santykiai 2009 05 09

Atnaujinta 2009 05 08 16:10

GAUKITE PRANEŠIMUS

nuo valiutų kursų pasikeitimo rizikos!

Jei keičiamos užsienio valiutos suma yra didesnė negu 10 000 Lt, keitimo kursas gali būti gerinamas, atsižvelgiant į tuo metu tarptautinėse valiutų rinkose kotiruojamus užsienio valiutų kursus.
Išsamesnės informacijos teiraukitės Finansų rinkų departamente telefonu (8 5) 268 2838.

2009 | gegužės | 9 | Rodyti | TXT formatas | Atsisiųsti

Valiuta	Valiutos kodas	Kiekis	Grynaisiais pinigais		Negrynaisiais pinigais		LB oficialus kursas *
			Perka	Parduoda	Perka	Parduoda	
Australijos doleriai	AUD	1	1,8458	2,0225	1,8980	1,9872	1,9556
Baltarusijos rubliai	BYR	1000	-	-	0,8050	1,0040	0,9279
Bulgarijos levai	BGN	1	-	-	1,6859	1,8450	1,7654
Čekijos kronos	CZK	10	1,2378	1,3760	1,2516	1,3313	1,2959
Danijos kronos	DKK	10	4,4197	4,7684	4,5303	4,7399	4,6348

Pav. 7 Teksto blokais ir lentelės forma pateikiami duomenys

Iš interneto svetainės surinkti duomenys dažniausiai yra apdorojami ir sustruktūrizuojami prieš juos panaudojant. Apdorojimo būdus aprašysime darbe po visų duomenų pateikimo būdų analizės.

2.4.2. Duomenų pateikimas skaitmeniniais dokumentais

Skaitmeniniai dokumentai - tai dokumentai išsaugoti tam tikro formato failuose. Šiuos formatus aprašo, atnaujina ir palaiko standartų organizacijos, įmonės ar asmenys, kurie juos sukūrė. Kadangi formatų yra daugybė, mūsų darbui svarbūs tik formatai kuriais pateikiami finansiniai dokumentai:

- Microsoft Word dokumentas
- Microsoft Excel dokumentas
- Adobe PDF
- Kablelių atskirtų reikšmių formatas (angl. CSV)
- Finansiniais standartais paremti dokumentai

Dažniausiai šie dokumentai yra publikuojami kaip papildomi duomenų šaltiniai interneto svetainėse. Adobe PDF ir Microsoft Word dokumentai dažnai naudojami įvairaus tipo ataskaitoms, aprašymams, kitai tekstinei informacijai pateikti. Microsoft Excel ir CSV formatų dokumentai naudojami pateikti sarašų tipo duomenims. Pvz.: Vartotojo atliktų operacijų banko sistemoje ataskaita ar vartotojo banko sąskaitos išlaidų ir pajamų ataskaita. Dažniausiai sarašo tipu pateikiami duomenys yra skirti tolimesniam jų apdorojimui ar panaudojimui.

Sparčiai plečiantis finansinėms veikloms įvairios finansinės institucijos prikuria savų duomenų saugojimo formatų, tokiu būdų apsikeitimas duomenimis tarp šių institucijų tampa vis sudėtingesnis, dėl naudoja skirtingų duomenų formatų. Todėl sparčiai tobulėjant technologijoms ir augant informacijos kiekiui, verslo atstovai ir finansinės institucijos sukūrė standartus, aprašančius apibendrintus finansinių duomenų saugojimo ir apsikeitimo formatus. Standartais paremti dokumentai dažniausiai yra kuriami finansinėms ataskaitoms, apsikeitimui finansiniais duomenimis tarp institucijų, bei skelbimui viešoje erdvėje. Šiuo metu pasaulyje yra gerai žinomi šie pagrindiniai finansiniai standartai ir specifikacijos:

- IFX (Interactive Financial Exchange)
- FIXML (Financial XML)
- OFX (Open Financial Exchange)
- XBRL (Extensible Business Reporting Language)

Interactive Financial Exchange (IFX) – žinučių siuntimo standartas finansiniams servisams. Tai yra XML pagrindu sukurta specifikacija finansiniams pervedimams, tokiems kaip įmonė-su-įmone ar įmonė-su-vartotoju bankiniai pervedimai (balansai ir k.t. informacija), apmokėjimams ir bankomatų komunikacijoms. [4]

Open Financial Exchange (OFX) - tai vieninga specifikacija elektroniniam apsikeitimui finansiniais duomenimis tarp finansinių institucijų, įmonių ir vartotojų internete. Ši specifikacija yra laisvai licenzijuota, leidžianti programų kūrėjams suprojektuoti savo sąsają, kuri realizuotų šioje specifikacijoje apibrėžtą finansinių duomenų srautų formatą [5].

Financial XML (FiXML) – tai elektroninio komunikavimo protokolas tarptautiniam realaus laiko apsikeitimui saugumumo bei finansų rinkų duomenimis [6].

Extensible Business Reporting Language (XBRL) **–** tai standartas skirtas finansinės informacijos, tokios kaip finansinės ataskaitos, apsikeitimui tarp verslo ir finansinių institucijų. Šis standartas palaiko informacijos modeliavimą ir semantinės prasmės išreiškimą. Standartas yra atviros licencijos ir pagrįstas XML [7].

2.4.3. Duomenų pateikimas sklaidos kanalais

Sklaidos kanalas – tai duomenų formatas naudojamas teikti vartotojams dažnai atnaujinamą informaciją. Turinio tiekėjai sukuria sklaidos kanalą (t.y. prieigą prie to pačio dokumento, kuris yra pastoviai atnaujinamas) ir leidžia vartotojams prisijungti prie jo naudojant specialias peržiūros programas arba specialias interneto svetaines, kurios dažniausiai vadinamos agregatoriais. Ir svetainių, ir programų veikimo principas yra paprastas – surinkti nurodytų kanalų teikiamą informaciją į vieną vietą, susisteminti pagal vartotojo pateiktus kriterijus ir parodyti nurodytu aspektu [8]. Pagal nustatymus agregatoriai išsiunčia užklausas visiems užregistruotiems kanalams ar jie neturi naujo turinio. Jei randamas naujas kanalo turinys jis yra parsiunčiamas arba parodomas pranešimas vartotojui, kad naujas turinys egzistuoja. Agregatorių privalumas yra tas, kad galima nustatyti periodinį kanalų turinio patikrinimą.

Sklaidos kanalo turinys paprastai yra HTML formato informacija arba nuorodos į interneto puslapius ar kitą skaitmeninę laikmeną.

Pagrindiniai šio būdo privalumai:

- Privatumas – nereikia pateikti el. Pašto ar kitos asmeninės informacijos
- Efektyvumas – užtenka vieną kartą užregistruoti kanalą ir informaciją vartotojas gauna automatiškai
- Tikslumas – vartotojas gauna informaciją tik iš pasirinkto kanalo
- Lengvas pašalinimas – Jei nenorima gauti informacijos, tereikia pašalinati kanalą iš stebimų sarašo
- Rūšiavimas – informacija rušiuojama pagal kanalą, todėl niekada nesusimaišo.

Trūkumai:

- Kanalo skaitytojai privalo turėti specialią programinę įrangą arba servisą teikiančius kanalo skaitymo paslaugas.
- Kanalo turinio tiekėjai turi rūpintis teikiamos informacijos kokybe ir tikslumu.

Internete yra paplitę 2 populiarų sklaidos kanalų formatai: RSS ir Atom.

RSS (Really Simply Syndication) – tai sklaidos kanalų formatų šeima naudojama dažnai atnaujinamiems darbams – tinklaraščių įrašams, naujienų antraštėms, audio ir video – standartizuotu formatu [9]. RSS dokumentas laiko pilną ar santraukos tekstą ir metaduomenis, tokius kaip dokumento išleidimo data ir autorystė. RSS yra specifikuotas naudojant XML.

Atom – šis sklaidos kanalų formatas kaip ir RSS naudojamas dažnai atnaujinamai informacija pateikti, tačiau pagrindinė Atom sukūrimo priežastis buvo tai, kad RSS formatas buvo ribotas savo galimybėmis ir turėjo spragų. Atom taip pat yra specifikuotas naudojant XML. Pagrindiniai skirtumai nuo RSS tai, kad turi turinio modelį, nurodantį kokio tipo duomenys yra pateikiami (HTML, XHTML, XML ar k.t.), naudoja standartizuotą datos formatą, naudoja XML žymę *xml:lang* informacijos kalbai nurodyti, bei yra modulinės struktūros, todėl atskiros Atom dokumento dalys gali būti pakartotinai panaudojamos kituose dokumentuose.

Tačiau dauguma interneto vartotojų informaciją patiekia tik vienu formatu ir tai dažniausiai yra RSS, dėl to kad jis buvo išleistas ir naudojamas anksčiau. Todėl Atom formato plitimas ir adaptacija yra lėtesnis nors pats formatas yra tobulesnis nei RSS.

2.4.4. Duomenų pateikimas per interneto aplikacijų programavimo sąsają

Aplikacijų programavimo sąsaja (API) yra aprašoma kaip „aibė ciklų, duomenų struktūrų, objektinių klasių ir/arba protokolų, kurias teikia bibliotekos ir/arba operacinių sistemų servisai, tam kad palaikyti aplikacijų kūrimą“ [10]. Viena iš pagrindinių API funkcijų yra suteikti programuotojui galimybę įgyvendinti tam tikrą funkcionalumą, naudojantis tam skirtos bibliotekos funkcijomis, klasėmis ar metodais. Finansų srityje tokios sąsajos naudojamos suteikti programuotojams galimybę operuoti finansinėmis funkcijomis, kurias teikia API sukūrusi finansinė institucija. Tai gali būti duomenų gavimas, statistikos išskaičiavimas ar finansinių grafikų nubrėžimas ir pan.

Interneto API – tai API skirta naudoti per internetą. Populiariausi interneto API realizacijos būdai:

- Web Servisai
- Objektų valdymo grupės (OMG)
- CORBA standartų sistema
- Paskirstytų sistemų komponentų objektų modelis (DCOM)
- Java/Nuotolinio metodo iškvietimas (RMI)

Interneto finansų srityje plačiausiai naudojamas API tipas yra web servisai, todėl panagrinėsime juos detaliau.

Web Servisai dažnai yra internetinės aplikacijų programavimo sąsajos, kurios yra prieinamos per tinklą ir vykdomos servisus teikiančiame serveryje. Web servisas yra apibūdintas konsorciumo W3C kaip „programų sistema suprojektuota palaikyti suderinamą mašina-su-mašina sąveiką per tinklą“ [11]. Šis apibūdinimas aprašo klientus ir serverius, kurie komunikuoja tarpusavyje internete naudojamu HTTP protokolu. Tokie web servisai išskiriami į dvi grupes:

- Dideli web servisai
- REST architektūros tipo servisai

Dideli web servisai naudoja išplėstos žymių kalbos (XML) žinutes, kurios atitinka paprastų objektų prieigos protokolą (SOAP). Tokios sistemos paprastai turi aprašytą sąrašą galimų operacijų, kurias teikia tas servisas. Pats servisas paprastai yra aprašomas web servisų aprašymo kalba (WSDL) [12]. Ši kalba nėra privaloma aprašant servisus, tačiau yra siūlytina, nes dauguma SAOP programavimo karkasų, sukurtų naudojant JAVA ir .NET technologijas, naudoja WSDL kalba aprašytus servisus automatizuotam programos kodo generavimui.

Reprezentacinės būsenos perkėlimo (REST) architektūra [13] pagrįsti servisai nereikalauja naudoti XML žinučių ar serviso aprašymų WSDL kalba. Šios architektūros išskirtinis požymis, kad ji operuoja resursais. Resursu gali būti bet koks specifinės informacijos duomenų šaltinis. REST architektūros servisai paprastai naudoja gerai žinomą ir standartizuotą HTTP protokolą serviso veikimui įgyvendinti. Tokio tipo servisą galima apibūdinti trimis aspektais [14]:

- Universaliu resursų adresu (URI) pvz.: http://www.pavyzdys.lt/resursai/knygos
- Kokius MIME duomenų tipus servisas palaiko. (JSON, XML, YAML ar kiti)
- Serviso palaikomų operacijų aibė naudojant HTTP metodus (POST, GET, DELETE, PUT)

Abi web servisų grupės tarnauja tam pačiam tikslui – suteikti programuotojui prieigą prie duomenų, tačiau skirtingais būdais. SAOP su WSDL pagrindu veikiančių servisų privalumas tai, kad programavimo karkasai iš serviso aprašymų sugeneruoja reikalingas objektų klases, metodus ir kitus serviso panaudojimui reikalingus atributus, tačiau dažnai tokius web servisus sudėtinga suprast, ypač jei jie teikia daug funkcijų. Taip pat dažnai šie servisai yra priklausomi nuo stambių programinės įrangos kūrėjų, o ne nuo atviro kodo realizacijų. Kitas trūkumas yra tai, kad naudojant XML, serviso greitaveika yra įtakojama žinučių analizavimo ir indeksavimo greitaveikos. Tuo tarpu REST architektūros principu veikiantys web servisai naudoja HTTP protokolo metodus, kuriems nereikia analizės ir tokiu būdu išvengiama greitaveikos sumažėjimo. Šio tipo privalumas - servisai yra lengvai suprantami.

2.5.Duomenų apdorojimas automatinio duomenų surinkimo procese

Iš informacinės sistemos gavus duomenis reikia juos perkelti į objektines duomenų struktūras. Jei duomenys gaunami HTML kalbos formatu, reikia atlikti dokumento analizę, kad būtų galima duomenis sustruktūrizuoti. Jei duomenys yra pateikti XML arba XML paremto standarto formatu, tuomet reikia patikrinti ar gauti duomenys atitinka nurodyto dokumento schemą ir neturi loginių klaidų. Aptarsime duomenų, paimtų iš informacinių sistemų, apdorojimo būdus.

2.5.1. Tinklalapio iškarpymas

Tinklalapio „iškarpymas“ (Web Scraping) yra programinės įrangos technika informacijos išgavimui iš interneto tinklalapių. Tokio tipo programinė įranga simuliuoja žmogaus apsilankymą svetainėje naudodama HTTP protokolą arba išnaudodama interneto naršykles, pvz.: Internet Explorer arba Mozilla Firefox. Žiniatinklio „iškarpymas“ yra susijęs su žiniatinklio indeksavimu bei automatizavimu, kai naršymo robotas indeksuoja tiklalapių turinį, todėl ši technika yra plačiai paplitusi ir naudojama interneto paieškų varikliuose. Tačiau dažniausiai žiniatinklio „iškarpymas“ yra naudojamas norint transformuoti nestruktūrizuotą tinklalapių turinį, kuris tipiškai būna pateiktas HTML formatu, į struktūrizuotus duomenis. Toliau šie duomenys gali būti saugomi duomenų bazėse ar analizuojami skaičiuoklėmis. Tipiniai žiniatinklio iškarpymo atvejai būtų interneto kainų palyginimai, orų duomenų stebėjimas, tinklalapių pasikeitimų nustatymas, žiniatinklio tyrinėjimai, tinklalapių turinio apjungimai bei žiniatinklio duomenų integracija. Mūsų darbe šis būdas būtų skirtas bankų paslaugų duomenims išgauti.

Duomenų „iškarpymas“ pagal automatizavimo kiekį gali būti keleto lygių.

- Nukopijuok ir įklijuok (angl. Copy - Paste)
- Dėsningų išraiškų atitikimas (angl. Regular Expressions arba regex)
- HTTP programavimas
- DOM analizė
- HTML analizatoriai
- Žiniatinklio „iškarpymo“ PĮ
- Semantikos pastabų atpažinimas

Trumpai aprašysime kiekvieną paminėtų lygių.

Iškirpk ir Įklijuok – tai rankinis duomenų išgavimo būdas, kai automatizuoti būdai yra neveiksmingi, ar dokumentai turi barjerų neleidžiančių automatiškai surinkti duomenis. Vartotojas suranda norimus saugoti duomenis, juos nusikopijuoja ir išsaugo norimu būdu, ar tai būtų naujas tekstinis dokumentas ar įrašas duomenų bazėje.

Dėsningų išraiškų atitikimas – tai paprastas, tačiau labai veiksmingas būdas išgauti informaciją iš tinklalapio panaudojant dėsningos išraiškos (RegEx) atitikimą, atitinkamoje programavimo kalboje.

HTTP programavimas – tiek statiniai, tiek dinaminiai tinklalapiai gali būti gaunami siučiant HTTP užklausas į nuotolinį interneto serverį naudojant prievadų[1] programavimą.

DOM analizė – programos panaudodamos įterpiamus interneto naršyklių komponentus (pvz. Internet Explorer arba Mozilla Firefox) gali gauti iš svetainės dinaminį tinklalapio turinį, kuris yra sugeneruojamas serverio pusėje vykdomo tinklalapio kodo. Ir toliau išnagrinėti ši turinį sudarant dokumento objektų modelį (angl. DOM).

HTML analizatoriai – kai kurios pusiau struktūrizuotos duomenų užklausų kalbos kaip XML užklausų kalba (XQL) ir hiperteksto užklausų kalba (HTQL) gali būti naudojamos interneto puslapio analizei tam, kad būtų galima išgauti ir transformuoti tinklalapio turinį.

Žiniatinklio iškarpymo PĮ – egzistuoja daug trečių šalių programų, kurios gali būti pritaikytos spręsti web iškarpymo problemą. Šios programos dažniausiai pateikia web terpės įrašymo sąsają arba skriptų funkcijas, skirtas web turinio išskyrimui ir transformavimui, bei taip pat pateikia duomenų bazių sąsajas, kad išskirtus duomenis būtų galima išsaugoti lokaliose duomenų bazėse.

Semantikos pastabų atpažinimas – jei puslapiai naudoja meta duomenų žymes ir anotacijas, juos galima panaudoti specifinių duomenų blokų suradimui. Panaudojant tokias anotacijas puslapiuose, jas galima apjungti į semantinį sluoksnį. Web iškarpymo PĮ gali išgauti duomenų schemas iš šio semantinio sluoksnio prieš pradedant iškarpymą. [15]

[1] Prievadų programavimas (socket programming) – programinės įrangos, kuri bendrauja su kitomis programomis kompiuterių tinkle, kūrimas.

2.5.2. XML dokumentų nagrinėjimas

Norint patikrinti ar XML dokumentas yra korektiškos sandaros ar norint jį panaudoti, dokumentas pirmiausia turi būti apdorotas. Tai atlieka programos, vadinamos XML nagrinėtojais (angl. XML Parser). Jos sugeba nuskaityti ir patikrinti XML dokumento duomenų korektiškumą. Nagrinėtojas leidžia programai naudotis dokumente esančiais duomenimis. Egzistuoja 2 pagrindiniai nagrinėtojų tipai [5]:

- Medžiais pagrįsti nagrinėtojai, kurie XML dokumentą programoje paverčia į vidinę atminties struktūrą (DOM).
- Įvykiais pagrįsti nagrinėtojai, kurie skaitydami XML dokumentą generuoja įvykius ir praneša apie tai programai (SAX, angl. Simple API for XML)

Pasirinkimą nulemia programos reikalavimai. Dažniausiai yra pasirenkamas DOM medžio nagrinėtojas, kadangi jo programavimo modelis žymiai aiškesnis, nei SAX įvykių apdorojimo. Tačiau, kad ir koks neįprastas būtų SAX programavimo modelis, jis gali būti šiektiek efektyvesnis. SAX programos gali būti žymiai spartesnės už DOM variantus ir beveik visada naudoja mažiau atminties. SAX patogu naudoti dirbant su dokumentais srautiniu režimu. Jeigu įmanoma suskaidyti sudėtingus apdorojimo procesus į kelis sluoksnius, SAX tipo nagrinėjimas yra efektyvesnis. DOM nagrinėjimą naudojančios programos linkusios būti daugiau monolitiškos, kur viena programa atlieka visą darbą.

Žemiau išvardinti programos bruožai nurodo, kada programoje geriau būtų naudoti SAX nagrinėtoją [16]:

- Dokumentai netelpa į atmintį. Tai pati svarbiausia priežastis pasirinkti vieną ar kitą programavimo sąsają. Jeigu dokumentai netelpa į atmintį, belieka naudoti srautinę XML darbo sąsają.
- Dokumentą galima apdoroti trumpais gabaliukais. Nereikia užsikrauti viso dokumento, kad galėtume pradėti su juo dirbti.
- Apdorojant dokumentą, užtenka tik žinių apie prieš tai buvusius elementus, nereikia žiūrėti į priekį.
- Apdorojimas gali būti išskaidytas į kelias nuoseklias operacijas.

Analogiškai pateikiami DOM nagrinėtojo pasirinkimą lemiantys programos bruožai [16]:

- Programa vienu metu turi kreiptis į atskiras dokumento dalis ar net kelis dokumentus vienu metu.
- Vidinės duomenų struktūros yra panašaus sudėtingumo kaip ir pats dokumentas.
- Programa turi modifikuoti dokumentą pakartotinai.
- Jei dokumentas saugomas ilgą laiką ir su juo periodiškai atliekamos operacijos

2.5.3. XML dokumentų korektiškumo tikrinimo priemonės

XML dokumentas yra tekstas, kuris aprašo hierarchinę duomenų struktūrą. XML standartas aprašo sintaksę, kaip turi būti sudarytas dokumentas. Atitinkantis XML sintaksę dokumentas vadinamas taisyklingu. Susidūrę su netaisyklingais dokumentais, įrankiai turėtų pranešti apie klaidas, nes tai jau nėra teisingas XML dokumentas. Tačiau norint sužinoti ar dokumentas yra prasmingas, neužtenka vien taisyklingumo tikrinimo. Teisingi duomenys turi atitikti tam tikrą loginę struktūrą, kurią aprašo schemų kalbos. Būtent schemų kalbomis aprašomos naujos XML kalbos. Bet kuris dokumento korektiškumą tikrinantis nagrinėtojas gali užtikrinti, kad nagrinėjamas dokumentas atitinka nustatytas kalbos taisykles. Taip palengvinamas programuotojo darbas, kad nereiktų programoje pačiam tikrinti ar pateikiami duomenys yra korektiški. Kadangi kartu su XML standartu pateikta schemų kalba DTD turėjo mažai galimybių, įvairios organizacijos sukūrė daug įvairių schemų kalbų [10].

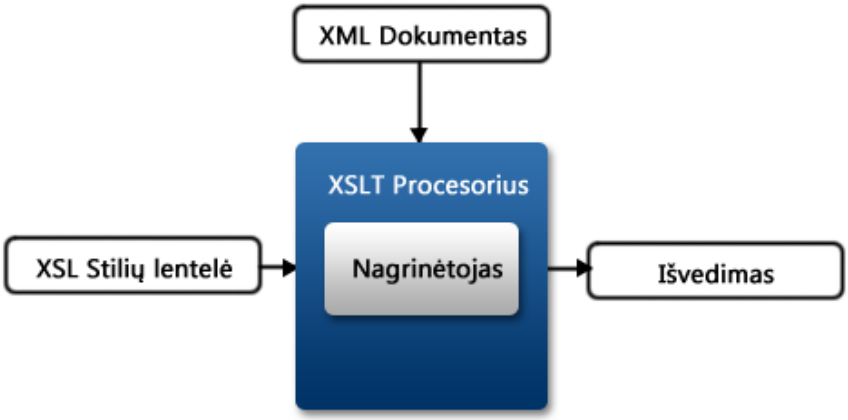
Seniausia schemų kalba yra dokumento tipo apibrėžimas (angl. DTD – Document Type Definition), kuris susideda iš taisyklių apie dokumento struktūrą ir turinį. DTD išvardina visus galimus elementus, jų pasirodymo tvarką ir atributus. Nors, palyginus su kitomis schemų kalbomis, DTD turi mažai galimybių, bet jis yra pirmasis priimtas standartas, populiarus ir dėl savo paprastumo. DTD yra sudedamoji XML standarto dalis, todėl jame yra keletas dalykų, kurie nepriklauso schemų kalbai, pvz., esybės. Nors dokumento tipo apibrėžimo metodas puikiai tiko pirmiems bandymams, bet kuriant sudėtingas sistemas iš karto paaiškėjo jo trūkumai. Viena pagrindinių DTD problemų – tai nėra korektiškas XML dokumentas. Todėl darbui su juo nebetinka standartiniai XML nagrinėtojai.

Dar daugiau problemų kelia tai, kad DTD turi tik ribotas galimybes dokumento struktūros ir turinio aprašymui. Juo neįmanoma aprašyti sudėtingų dokumento elementų ryšių, be to, vienam elementui aprašytos taisyklės negali būti panaudotos kito elemento aprašymui. Šias problemas išsprendžia kitas standartas – XML Schema. Šie dokumentai yra taisyklingi XML dokumentai, jie palaiko vardų sritis bei turi priemones elementams priskirti tipus, taip leisdama tikrinti dokumentus ne tik pagal elementų struktūrą, bet ir pagal jų turinį. Viena, labiausiai išsiskirianti iš kitų schemų kalbų XML Schema savybė, yra tipų paveldimumas.

2.5.4. XML dokumentų transformavimas

Vienas pagrindinių XML technologijos privalumų – tai galimybė konvertuoti XML duomenis iš vienos formos į kitą, naudojantis bendrinėmis priemonėmis. Technologija, tam atlikti yra išplečiama stilių kalba transformacijoms XSLT (angl. eXtensible Stylesheet Language for Transformations).

XSL stilių lentelė – tai transformavimo instrukcijų rinkinys, kuriomis išeitinis XML dokumentas verčiamas kitu dokumentu (nebūtinai XML). Norint atlikti XSLT transformacijas reikalingas XSLT procesorius, kuris transformuoja išeities XML dokumento duomenis, susiedamas juos su XSL stilių lentelės šablonais. Tačiau, XSLT procesoriai yra ganėtinai lėti, todėl kartais reikalingi specialūs pritaikymai norint generuoti dokumentus realiu laiku.



Pav. 8 XSLT procesoriaus darbo schema

XSLT kalba yra aukšto lygio programavimo kalba, kurios pagrindai ateina iš funkcinio programavimo teorijos. Ji žymiai galingesnė nei panašios paskirties standartai CSS ar CSS2, todėl vidutiniam programuotojui ją įvaldyti yra gana sunku. Lygiai tokia pati situacija buvo ir su SQL jos atsiradimo metu, tai tik dar kartą įrodo, kad XSLT nepaprastai galinga technologija.

XSL Formatavimo objektai – tai instrukcijos, kurios apibrėžia informacijos išsidėstymą ir išvaizdą. XSL-FO leidžia naudoti žymiai sudėtingesnius puslapio elementų išsidėstymo modelius nei HTML+CSS, todėl formatavimo objektais galima aprašyti visą spausdintą knygą. Dažniausiai XSL-FO naudojami norint gaminti PDF failus, kurie skirti verslo duomenų atvaizdavimui. Konvertavimui į galutinį formatą naudojami formatavimo objektų procesoriai.

Nors XML-FO dokumentus galima kurti ir rankomis, bet dažniausiai tam yra rašoma XSL transformacija ir formatavimo objektai generuojami iš duomenų failų.

2.5.5. XML dokumentų duomenų išgavimas ir manipuliavimas

Atskirų XML dokumento dalių išskyrimas ir manipuliavimas jomis yra svarbios veiklos, todėl sukurta daug susijusių specifikacijų, kurios apibrėžia, kaip nurodyti tam tikras XML dokumento vietas ar atlikti informacijos paiešką su parametrais. Toliau aptariamos dvi svarbiausias šios srities kalbos.

XPath yra kalba, skirta XML dokumento dalių adresavimui, kurios sintaksė primena kelius failų sistemoje ar URL. XPath palaiko funkcijas darbui su atrinktais duomenimis. Funkcijos skirtos gauti informaciją apie dokumento atšakas bei darbui su eilutėmis, skaičiais ir loginiais kintamaisiais. Programuotojai gali plėsti prieinamų funkcijų biblioteką.

XPath naudoja kompaktišką, ne XML sintaksę, kad galėtų būti naudojama XML atributuose arba nuorodose. XPath rekomendacija remiasi XSLT, XML Schema, XPointer specifikacijomis. XPath dirba su abstrakčia, logine XML dokumento struktūra, o ne tekstine sintakse. Ji skirta darbui su vienu XML dokumentu, į kurį žvelgia kaip atšakų medį. XPath užklausos rezultato reikšmės konceptualiai irgi yra laikomos atšakomis. XPath duomenų modelyje egzistuoja šių tipų atšakos: tekstinės, elementų, atributų, šakninės, vardų sričių, apdorojimo instrukcijų, komentarų atšakos [17].

XQuery – tai bandymas pateikti užklausų kalbą, kuri būtų panašaus funkcionalumo ir turėtų stiprų formalizmą, kaip SQL reliacinėms duomenų bazėms. XQuery yra funkcinė kalba, kurioje kiekviena užklausa yra išraiška. XQuery išraiškos skirstomos į septynias plačias kategorijas: kelio išraiškos (XPath), elementų konstruktoriai, FLWR išraiškos (FROM, LET, WHERE, RETURN), išraiškos su operatoriais ir funkcijomis, sąlyginės išraiškos, įvertinimo išraiškos ir išraiškos, kurios tikrina ar keičia duomenų tipus. Skirtingų išraiškų sintaksė ir semantika labai skiriasi, todėl, kad XQuery kūrimui turėjo įtakos daug skirtingų sistemų.

XQuery turi sudėtingą tipų sistemą, kuri paremta XML Schema duomenų tipais ir, kitaip nei XPath, palaiko manipuliavimą dokumento atšakomis. Taip pat XQuery duomenų modelis sukurtas darbui ne tik su pavieniu XML dokumentu, bet ir taisyklingu jo fragmentu, dokumentų seka ar dokumentų fragmentų seka [18].

2.5.6. Skaitmeninių dokumentų apdorojimas

Informacija pateikiama skaitmeniniais dokumentais nevisada gali būti apdorota. Tai priklauso nuo dokumento formato. Jei dokumentuose yra sarašo tipo duomenys pvz.: CSV arba Excel failai, tokiems dokumentams yra parašomos specialios paprogramės, kurios pagal dokumento formatą duomenis nuskaito į objektines struktūras. Tačiau tokiems dokumentas, kuriuose pateiktas įvairaus tipo tekstas su paveiksliukais ar be, išskaidyti ir atrinkti duomenis yra sudėtinga ir dažnai netikslinga. Retais atvejais, jei dokumentas yra sudarytas pagal šabloną, duomenis išskirti galima, bet tai padaryti be klaidų pavyksta nevisada.

2.5.7. Sklaidos kanalais ir web servisais teikiamų duomenų apdorojimas

Informacija pateikiama sklaidos kanalais paprastai yra XML ar HTML, todėl jos apdorojimas yra vykdomas anksčiau minėtais šių formatų apdorojimo būdais. Skirtumas tik toks, kad jei duomenys pateikiami HTML formate, jų nagrinėjimas yra paprastesnis, nes nereikalinga nagrinėti viso tinklalapio puslapio turinio.

Jeigu duomenys yra išgaunami naudojant web servisus papildomas apdorojimas yra nereikalingas, nes web servisai gražina jau struktūrizuotus duomenis. Kreipimasi į web servisus vyksta kaip į programos funkciją, kuri turi aiškiai apibrėžtą gražinamų duomenų formatą.

2.6. Duomenų išsaugojimas automatinio duomenų surinkimo procese

Atlikus paimtų finansinių duomenų apdorojimą paprastai šie struktūrizuoti duomenys išsaugomi tolimensiam jų panaudojimui. Jeigu duomenys buvo pateikti standartuose aprašomais formatais ar yra kitų tipų korektiški XML dokumentai, dažniausiai šių dokumentų saugoti atskira forma nereikia. Paprastai šie dokumentai suskirstomi į turinį atitinkančius katalogus, kur toliau bus panaudoti programinėje įrangoje. Kai kuriais atvejais, kai norima atlikti turinio indeksavimą, bei naudojant XML manipuliacijas sukurti kitų tipų dokumentus, šie dokumentai yra saugomi duomenų bazėje specialaus tipo laukuose leidžiančiuose laikyti visą dokumento failą. Jei apdoroti duomenys yra laikinai saugomi kompiuterio atmintyje struktūrizuotų objektų forma (Pvz. Atlikus HTML analizę), juos galima išsaugoti šiais būdais:

- Duomenų bazėje
- XML pagrįstame faile

Jei dokumentai nebuvo XML failo pavidale, galima juos tokiu pavidalu išsaugoti savo sukurtu formatu arba remiantis esamais finansinių duomenų standartais. Nepriklausomai nuo formato tokiems dokumentams sukuriamos jau anskčiau darbe minėtos XML dokumento schemos, kad ateityje galima būtų patikrinti šių dokumentų korektiškumą. Jei remiamasi standartu galima naudoti trečių šalių įrankius arba pačiam realizuoti standarte specifikuojamą formatą.

Dažniausiai surinkti duomenys yra saugojami duomenų bazėse. Saugojant duomenų bazėse užtikrinamas duomenų struktūriškumas, išlaikomi duomenų saryšiai, bei suteikiama galimybė lengvai pasinaudoti duomenimis. Šio būdo privalumas, tai kad lengva atlikti duomenų analizę, naudojant duomenų bazių užklausas, palyginus su saugojimu atskiruose failuose interneto serveryje.

2.7.Automatiškai surinktų duomenų panaudojimo galimybių analizė

Surinkus finansinius duomenis atsiveria galimybės juos panaudoti įvairiems tikslams. Pagrindiniai keliai, kur galima būtų panaudoti tokius automatiškai surenkamus duomenis yra šie:

- Finansinės skaičiuoklės
- Statistiniai skaičiavimai ir finansinės prognozės
- Duomenų tiekimas kaip paslauga

2.7.1. Duomenų panaudojimas finansinėms skaičiuoklėms

Automatiškai surinktų finansinių duomenų panaudojimas finansinėms skaičiuoklėms yra vienas populiariausių būdų. Paprastai finansinės insitucijos kuria skaičiuokles naudodamos ne viešus, tačiau vidinius įmonės duomenis, jei skaičiuoklės yra vidiniam įmonės naudojimui. Tačiau, norint sukurti skaičiuoklę, kuri naudotų ne tik tos institucijos duomenis, bet ir kitų institucijų ir galėtų lyginti reikalingi tu įmonių viešai skelbiami doumenys. Čia ir pasitarnauja automatinis tokių duomenų surinkimas. Duomenys išsaugoti duomenų bazėje ar XML failuose yra panaudojami vykdant skaičiavimus, kuriuose šie jie gali būti palyginti ir pateikiamas geriausias rezultatas. Taip pat tokie duomenys nebūtinai gali būti naudojami palyginimui. Dažnai jie gali būti kaip argumentai tam tikroms funkcijoms ar algoritmams. Dažniausiai tokie duomenys yra panaudojami kuriant dideles finansinių skaičiuoklių informacines sistemas, kur galima paskaičiuoti atskirų institucijų paslaugų įverčius ir pan. Pvz.: (Valiutų kursai, indėlių paskaičiavimai, paskolų įmokos, mokesčiai ir t.t.)

2.7.2. Duomenų panaudojimas statistiniams skaičiavimams ir prognozėms

Dažnai automatiškai surenkami finansiniai duomenys yra naudojami įvairiems statistiniams skaičiavimams. Kadangi duomenų kiekis yra didelis, tai suteikia galimybę paskaičiuoti įvairius ekonominius rodiklius ar kitokius finansinius koeficientus tiksliau. Lengviausią tokį duomenų panaudojimą galima būti perteikti paaiškinant pasaulinės Forex valiutų biržos modelį. Forex valiutų biržoje duomenys keičiasi kas sekundę, dėl didelės valiutos prekybos visame pasaulyje. Valiutos kursų duomenų kiekis šioje veikloje yra milžiniškas. Šiems duomenims analizuoti paprastų finansinių skaičiuoklių nepakanka. Todėl yra naudojamos specialios programos, naudojančios šiuos duomenis valiutų kursų grafikams braižyti ir daugybei finansinių rodiklių skaičiuoti ir pažymėti grafikuose. Pvz.: (Paprasti ir eksponentiniai vidurkiai, Boilerio Bangos, Fibonaci atsekamumo lygiai ir k.t.). Tokių indikatorių ir indeksų šioje sferoje yra virš 130, ir visi jie operuoja tais pačiais duomenimis. Pagal šiuos indeksus, grafikus ir papildomus netechninius kriterijus finansų analitikai sudaro finansines prognozes bei vykdo valiutų prekybą. Tai tik vienas iš galimų sudėtingesnių pavyzdžių kaip automatiškai surenkami duomenys gali būti sėkmingai panaudojami statistiniams skaičiavimams.

2.7.3. Duomenų panaudojimas duomenų tiekimui

Labai dažnai finansinės institucijos dirbančios su duomenimis, kurių kiekis yra milžiniškas patiki duomenų surinkimo procesą trečiosioms šalims. Specializuotos įmonės užsiima vien tik finansinių duomenų automatizuotu surinkimu ir šio proceso priežiūra. Tokiu atveju duomenys tampa produktu, o duomenų tiekimas – paslauga. Tokios įmonės įkuria specifinių duomenų bankus ir rūpinasi tik duomenų surinkimu, apdorojimu ir paruošimu naudojimui. Tokių duomenų bankų vartotojai gauna prieigą prie duomenų per web servisus ar sklaidos kanalus su autorizacija. Paprastai ši paslauga yra abonento tipo, mokant nustatytą mokestį kas mėnesį už prieigą arba už panaudotų duomenų kiekį.

3. INFORMACINĖS SISTEMOS PROJEKTAVIMAS IR REALIZACIJA

3.1.Informacinės sistemos kūrimo procesas

Sistemos kūrime buvo atliekami gerai žinomi programų inžinerijos programų kūrimo proceso žingsniai: reikalavimų rinkimas, sistemos projektavimas, sistemos kodavimas, testavimas ir įdiegimas. Prieš pradedant sistemos kūrimą buvo sudaryta kūrimo darbų seka:

1. IS reikalavimų surinkimas
    - Reikalavimų specifikacijos parengimas
2. Sistemos modelių sudarymas
    - IS panaudos atvejų modelio sudarymas
    - IS analitino modelio sudarymas
    - IS projektinio modelio sudarymas
3. Detalios sistemos architektūros specifikacijos sudarymas
4. IS duomenų bazės projektavimas
5. Automatinis duomenų iš bankų surinkimas
6. IS tinklapio kūrimas
    - Prisijungimo prie sistemos modulis
    - Administratoriaus posistemė
    - Skaičiuoklių modulis
    - Vartotojo duomenų valdymo modulis
    - Finansinės informacijos valdymo modulis
7. Specializuotų skaičiuoklių kūrimas ir prijungimas prie sistemos
8. IS testavimas
    - Testavimo plano sudarymas
    - Sistemos testavimas
9. Sistemos įdiegimas
10. Informacinės sistemos vartotojo vadovo parengimas

3.2.Informacinės Sistemos apibūdinimas

Mano magistrinio darbo projektinė užduotis buvo sukurti finansinių skaičiuoklių informacinę sistemą kuri

- Realizuotų automatinį finansinių duomenų surinkimą
- Panaudotų surinktus duomenis skaičiuoklėse

Kadangi finansinių skaičiuoklių tipų yra daug, todėl darbe buvo koncentruojamasi į Lietuvos bankų teikiamas paslaugas ir skaičiuokles, kurios atitiktų šias paslaugas. Lietuvoje dauguma bankų jau turi sukurę finansines skaičiuokles savo teikiamoms finansinėms paslaugoms. Tačiau jos skaičiuoja rezultatus tik nurodytiems duomenims ir neturi galimybės palyginti savo skaičiavimų rezultatų su kitų bankų analogiškos operacijos rezultatais, nevertina papildomų savybių tokių kaip pinigų infliacija ar pan. Kadangi skaičiuoklių tipų gali būti įvairių, magistrinio darbo projekte buvo kuriamos 3 tipų skaičiuoklės:

- Terminuotojų indėlių palūkanų sumos skaičiuoklė
- Kaupiamųjų indėlių palūkanų sumos skaičiuoklė
- Valiutos keitimo skaičiuoklė

Terminuotojų indėlių palūkanų sumos skaičiuoklė - skirta paskaičiuoti, kokią sumą galite sukapti padėję terminuotą indėlį jūsų pasirinktam laikotarpiui skirtinguose Lietuvos bankuose.
Kaupiamojo indėlio palūkanų sumos skaičiuoklė - skirta paskaičiuoti, kokią sumą galite sukapti padėję kaupiamąjį indėlį jūsų pasirinktam laikotarpiui skirtinguose Lietuvos bankuose
Valiutos keitimo skaičiuoklė - skirta paskaičiuoti naudingiausią vartotojui valiutos pirkimą ar pardavimą lyginant skirtingų Lietuvos bankų siūlomus valiutų kursus

Vartotojas ateidamas į šią sistema turi problemą rasti pigiausią ir geriausią jį dominančios banko paslaugos variantą. Taigi jo tikslas yra rasti šio geriausio rezultato šaltinį ir palyginti jį su kitais.

Privalumai:

- Sistema yra lengvai pasiekiama internetu
- Sistema automatiškai atnaujina duomenis naudojamus skaičiavimams
- Sistemos skaičiuoklės turi palyginimo galimybę tarp kelių duomenų šaltinių
- Naujos skaičiuoklės lengvai įdiegiamos į sistemą

Trūkumai:

- Vartotojai įpratę naudotis įprastomis skaičiuoklėmis
- Sistemos veikimui reikalinga interneto prieiga

3.3.Apribojimai sistemos kūrimui

3.3.1. Apribojimai sprendimui

- Sistema turi veikti nepriklausomai, kurioje operacinėje naudojama
- Sistema kuriama naudojant ASP.NET technologiją
- Sistema turi būti lengvai pritaikoma skirtingiems duomenų saugyklų tipams

3.3.2. Diegimo aplinka

- Sistema naudos MS SQL 2005 duomenų bazę.
- Sistema veiks IIS pagrindu ir bus įdiegta Windows 2005 tipo web serveryje.
- Sistema naudos ASP.NET 3.5 SP1 karkasą
- Sistema naudos ASP.NET MVC karkasą

3.3.3. Bendradarbiaujančios sistemos

Sistema tiosioginio ryšio su kitomis sistemomis neturės. Tačiau naudodama specialius metodus iš kitų interneto sistemų išrinks pateikiamus duomenis.

3.3.4. Komerciniai specializuoti programų paketai

Sistemos kūrimui nebus naudojami jokie papildomi specializuoti programų paketai.

3.3.5. Numatoma darbo vietos aplinka

Sistemos reikalavimai vartotojui:

- Turėti prieigą prie interneto
- Web naršyklę

Kadangi sistema yra kuriama web aplinkai, todėl vartotojui tai nesukels didelių problemų ir reikalavimų, kad jis galėtų naudotis sistemos funkcijomis. Visas sistemos darbas bus atliekamas internetinės naršyklės pagalba.

3.4.Nefunkciniai reikalavimai

3.4.1. Reikalavimai sistemos išvaizdai

Sistema turi būti lengvai suprantama vartotojui, intuityvi, neperkrauta daugeliu elementų.

3.4.2. Reikalavimai panaudojamumui

Sistema turi būti suprojektuota taip, kad ateityje būtų galima kurti naujas skaičiuokles ir jas lengvai įdiegti į sistema prie jau egzisuotuojančių.

3.4.3. Reikalavimai vykdymo charakteristikoms

Jei skaičiuoklė naudoja sudėtingus skaičiavimus, tokie skaičiavimai turi būti atliekami klientinėje dalyje, kad nebūtų apkrautas sistemos serveris ir tai nesukeltų sulėtejimo bei kitokių vykdymo problemų kitiems sistemos vartotojams.

3.4.4. Reikalavimai sistemos priežiūrai

Sistema turi būti sukurta taip, kad ją valdyti ir prižiūrėti galėtų ne vien tik sistemos kūrėjai, bet ir sistemos pirkėjai, jei sistema bus parduota.

Pradėjus esploatuoti sistemą, turi būti vykdomi periodiniai duomenų bazės atsarginių kopijų kūrimo darbai.

3.4.5. Reikalavimai saugumui

Sistema turi apsaugoti esamus duomenis nuo neteisėtos prieigos prie jų. Sistemos duomenys turi būti prieinami teisėtam vartotojui, kad jam to reikės. Pateikiami duomenys turi būti teisingi – tokie kokius vartotojas buvo suvędęs. Sistema privalo turėti minimalią apsaugą nuo skriptų ir duomenų bazių užklausų injekcijos atakų.

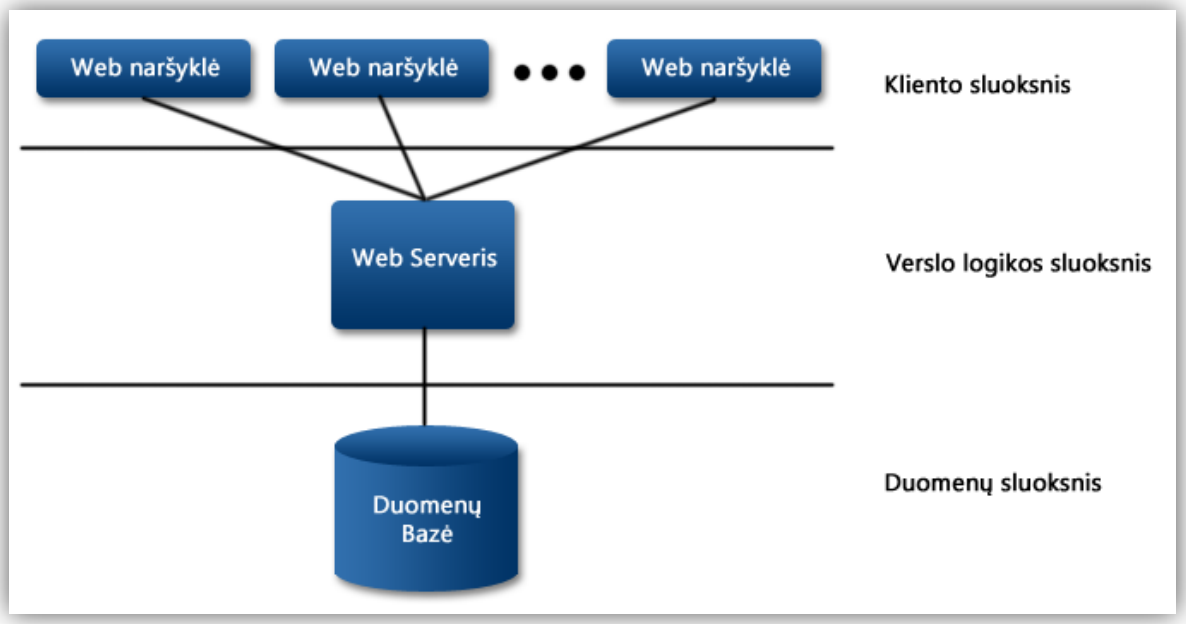
3.5.Architektūros pateikimas

Sistemos architektūros pasirinkimas yra vienas svarbiausių PĮ kūrimo etapų. Neteisingas architektūros parinkimas gali daug kainuoti sistemos kūrėjams, todėl reikalinga gerai išanalizuoti surinktus reikalavimus sistemai.

Sistemos kūrimui buvo renkamasi iš dviejų architektūros tipų:

- Kliento – Serverio architektūra
- Trijų sluoksnių architektūra

Nuspręsta sistemos kūrimui pasirinkti trijų sluoksnių architektūrą. Ši architektūra pranašesnė už kliento – serverio architektūrą tuo, kad sistemos logikos dalis yra atskirta nuo klientinės dalies, taigi pasikeitus skaičiuoklių skaičiavimo logikai nereikėtų vykdyti pakeitimų klientinėje dalyje. Tai labai svarbu mano kuriamai sistemai, kadangi skaičiuoklės bus kuriamos ir įtraukiamos į sistemą dinamiškai, todėl kuo didesnis tarpusavio dalių nepriklausomumas, tuo lengvesnis tampa sistemos kūrimas ir palaikymas. Žemiau pateikiamas architektūros grafinis vaizdas (9 Pav.).



Pav. 9 Trijų sluoksnių architektūros modelis

3.6.Architektūros tikslai ir apribojimai

3.6.1. Tikslai

Architektūros tikslai yra sukurti kokybišką, sparčiai veikiančią, lengvai modifikuojamą sistemą sumažinant klaidų ir pakeitimų kainą ateityje.

3.6.2. Apribojimai

Sistemos architektūrai keliami apribojimai:

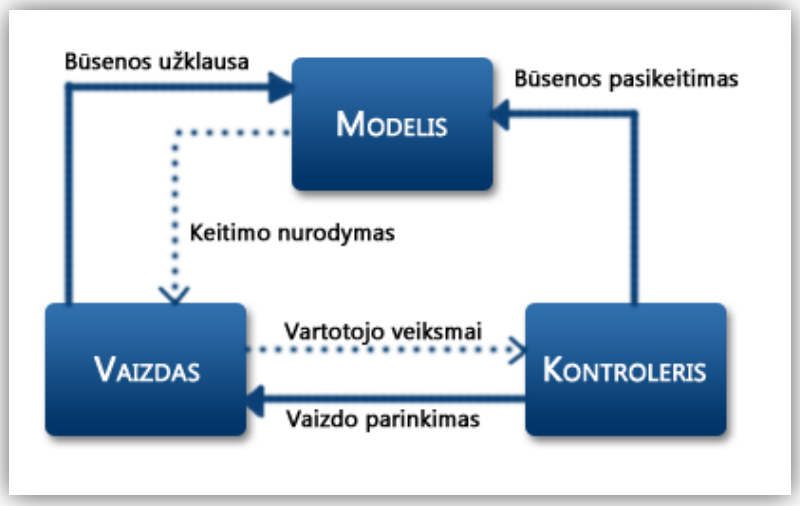
- Kliento sluoksnyje bus naudojamos web naršyklės.
- Logikos sluoksnyje bus naudojamas ASP.NET technologiją palaikantis Windows Web serveris.
- Duomenų sluoksnyje bus naudojama MSSQL duomenų bazė
- Sistema turės veikti Internet Explorer, Mozilla Firefox, Google Chrome ir Opera naršyklėse.

3.7.Kuriamos sistemos technologiniai sprendimai

Prieš pasirenkant technologiją sistemos įgivendinimui, buvo ieškomas programavimo šablonas labiausiai tinkantis įgyvendinti trijų sluoksnių architektūra. Peržiurėjus galimus pagrindinius šablonus buvo nuspręsta pasirinkti modelio – peržiūros – kontrolerio MVC (angl. Model-View-Controller) šablonas, kadangi jis įgyvendina taip vadinamą dalykų atskyrimą (separation of concerns), todėl atskiras dalis lengva atskirti pagal mūsų pasirinktą architektūrą. Sistemos kūrimui pasirinktą ASP.NET technologija kartu su naujai sukurtu ASP.NET MVC karkasu, leidžiančių maksimaliai išnaudoti .NET technologijos teikiamas programavimo galimybes, tokias kaip LINQtoSQL objektinės ir realicinės duomenų bazės surišimas, LINQ integruota kodo užklausų kalba darbui su sarašų tipo objektais bei kitos objektiniam programavimui svarbios savybės. Taip pat, kad padidinti sistemos darbo našumą buvo nuspresta naudoti saugyklos ir filtrų šabloną duomenų valdymui.
Aprašysime šiuos šablonus detaliau.

3.7.1. Modelis – Vaizdas – Kontroleris šablonas

MVC tai yra architektūrinis šablonas naudojamas programų sistemų inžinerijoje. Šio šablono pagrindinė idėja yra izoliuoti sistemos verslo logiką nuo vartotojo aplinkos, tam kad būtų galima žymiau lengviau modifikuoti programos grafinę aplinką arba aprašytas verslo taisykles, neįtakojant vienas kito. Šiame šablone modelis reprezentuoja programos duomenis, vaizdas – duomenų atvaizdavimo grafinę sąsają, o kontroleris atlieka duomenų ir verslo taisyklių manipuliacijas į modelį ir iš modelio. Iš esmės kontroleris kontroliuoja modelio atvaizdavimą atitinkamuose vaizduose, priklausomai nuo vartotojo veiksmų.

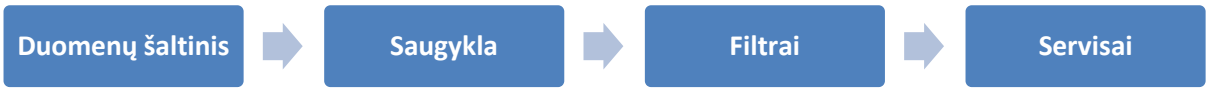


Pav. 10 MVC šablono modelis

3.7.2. Saugyklos ir filtrų šablonai

Saugyklos šablonas veikia paprastu principu. Mūsų programa kreipiaisi į duomenų valdymo paprogramę užklausdama tam tikrų duomenų, nesirupinant kokiu būdu ir iš kur tie duomenys bus paimti, todėl ir vadinama saugyklos vardu. Šis šablonas leidžia programai neprisirišti prie specifinio duomenų saugojimo būdo, todėl mes bet kurie metu galima programai perduoti duomenis iš failų, ar skirtingų tipų ir skirtingas technologijas naudojančių duomenų bazių.

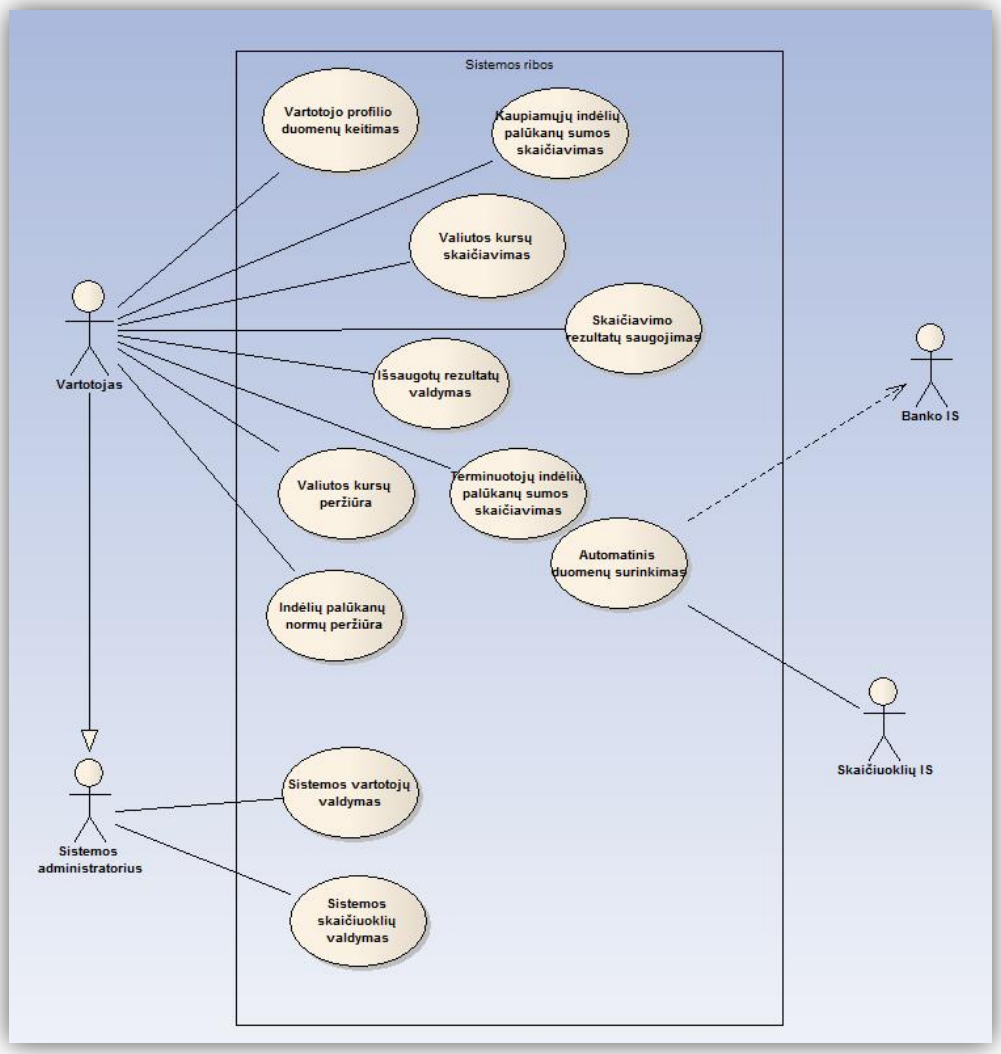
Šiems paimtiems duomenims tolesniame programos žingsnyje pritaikome filtrų šabloną, kuris pagal programos reikalavimus atrenka tik tenkinančius duomenis. Toks būdas leidžia išvengti daugkartinio kreipimosi į duomenų bazę, bei supaprastina duomenų išrinkimą, nes nebūtina sukurti visų reikalingų duomenų atrinkimo kombinacijų, užtenka sukurti reikalingus filtrus, ir tam pačiam duomenų srautui pritaikyti reikalingus atliekamai operacijai.



Pav. 11 Saugyklos ir filtrų šablonų procesas

3.8.Sistemos panaudojimo atvejų vaizdas

Žemiau pateikiamas projektavimo metu sudarytas sistemos galimybes atvaizduojantis panaudojimo atvejų vaizdas.

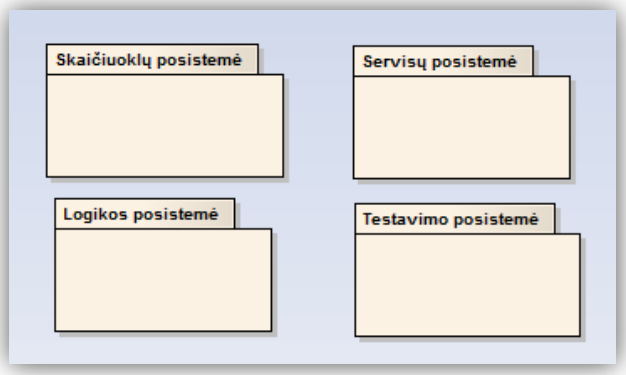


Pav. 12 Sistemos panaudojimo atvejų diagrama

3.9.Sistemos statinis vaizdas

3.9.1. Apžvalga

Žemiau pateikiamas sistemos išskaidymas į posistemes

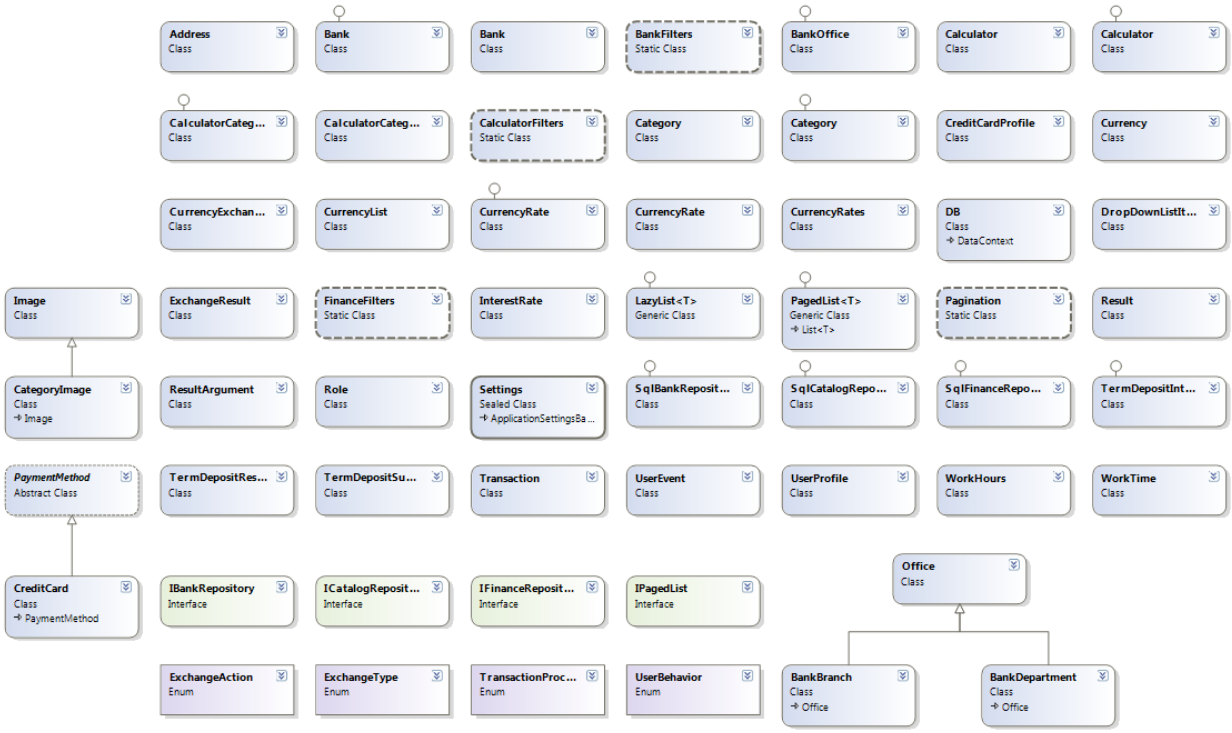


Pav. 13 Sistemos posistemių diagrama

3.9.2. Posistemių detalizavimas

- **Skaičiuoklių posistemė** – ši posistemė yra skirta sistemos kliento sluoksniui bei logikai įgyvendinti. Šioje posistemėje yra sistemos kontroleriai bei sistemos formos.
- **Sistemos servisų posistemė** – posistemė, skirta sistemos funkcijoms reikalingų duomenų gavimui ir valdymui.
- **Sistemos logikos posistemė** – posistemė skirta sistemoje naudojamiems modeliams ir duomenų sluoksniui aprašyti.
- **Testavimo posistemė –** skirta sistemos testavimui.

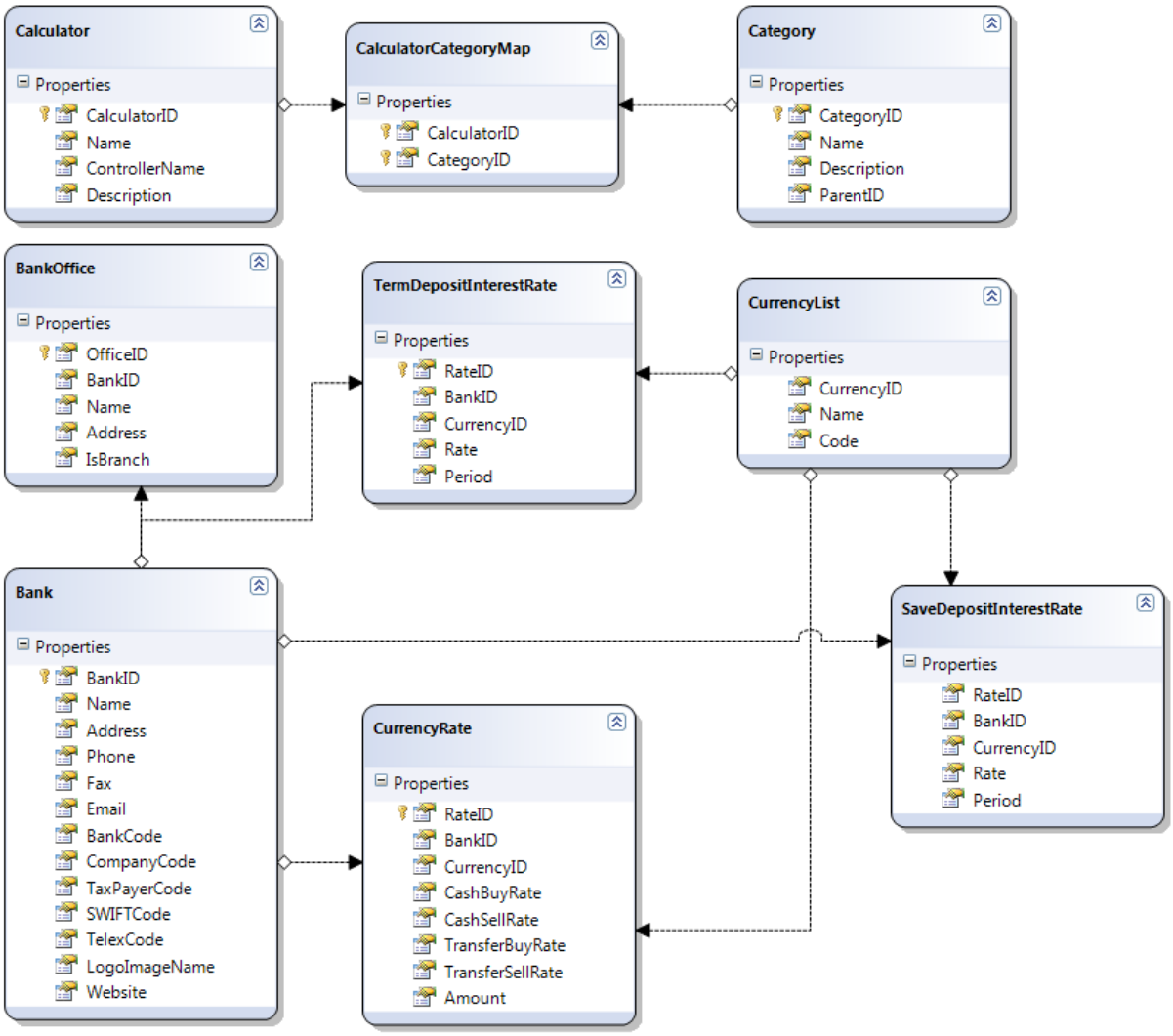
Kadangi posistemių diagramos viena su kita susijusios tai pateikiama bendra visų posistemių klasių diagrama.



Pav. 14 Sistemos klasių diagrama

3.10. Sistemos duomenų vaizdas

Duomenų bazės modelis projektuojant sistemą keitėsi, todėl pateikiamas modelis jau sukūrus sistemą. Sistema susideda 9 pagrindinių lentelių.



Pav. 15 Sistemos duomenų bazės modelis

Calculator – lentelė laikanti duomenis apie sistemoje esančias skaičiuokles

Category – lentelė skirta skaičiuoklių kategorijoms

CalculatorsCategoryMap – lentelė laikanti duomenis apie tai kuriai kategorijai priklauso kiekviena skaičiuoklė

BankOffice – lentelė skirta bankų skyrių adresų ir kontaktų duomenims laikyti

Bank – lentelė laikanti duomenis apie Lietuvos bankus, kurių finansiniai duomenis sistemoje yra naudojami

CurrencyList – lentelė skirta valiutų sarašui laikyti

TermDepositRates – lentelė terminuotojų indėlių palūkanų normoms

CurrencyRate – lentelė valiutos keitimo kursams

SaveDepositInterestRate – taupomųjų indėlių palūkanų normų lentelė

3.11. Esminiai sistemos realizacijos ypatumai

Surinktų duomenų analizei atlikti reikalinga užkrauti HTML dokumentą ir suformuoti jo šakninių ryšių medį (DOM). Tam įvykdyti buvo panaudota trečios šalies HTML funkcijų biblioteka skirta .NET programavimo platformai, kadangi standartinių platformos funkcijų nepakako teisingai atlikti analizę. Ši biblioteka leidžia išanalizuoti dokumentą ir sudaro dokumento DOM medį. Toliau rankiniu būdu naudodamas bibliotekos HtmlDocument klasės funkcijas HasChilds ir Childs[index] išrenkamas reikiamas HTML duomenų blokas laikantis visą reikalingą informaciją. Duomenų bloko išrinkimams buvo parašyta papildoma pagalbinė programa, vizualiai atvaizuojanti medžio struktūrą. Radus norima duomenų bloką, jei duomenys pateikti lentelėse, vykdomas ciklas praeinant visus pasikartojancius elementus. Jei pasikartojimų nėra, tuomet rankiniu būdu išrenkami reikalingi laukai. Visi išrinkti duomenys yra talpinami bendroje visiems bankams sukurtoje struktūroje. Vėliau šie objektai išsaugomi sistemos duomenų bazėje.

Trijų sluoksnių architektūrai vieni iš svarbiausių kokybės faktorių yra aukštas sistemos integruojamumas ir pernešamumas, kadangi sistemos esminiai komponentai yra išskaidyti į atsikirus sluoksnius, tai nėra sudėtinga sistemą intergruoti į kitas sistema. Šio tipo architektūra reikalauja daugiau pastangų sistemos planavimui, tačiau žymiai sumažinami sistemos palaikymo ir kūrimo kaštai, palyginus su dviejų sluoksnių architektūra. Architektūra taip pat pasižymi kodo lankstumu ir pakartotiniu panaudojamumu.

3.12. Sukurtos sistemos funkcijos ir galimybės

- Automatinis periodinis bankų valiutų kursų atnaujinimas iš bankų interneto tinklapių
- Automatinis periodinis bankų terminuotų ir kaupiamųjų indėlių palūkanų normų atnaujinimas iš bankų interneto tinklapių
- Lietuvos bankų sarašas, nuorodos į banko informaciją
- Lietuvos bankų rekvizitai bei padalinių kontaktai
- Lietuvos bankų valiutos kursų lentelės
- Lietuvos bankų terminuotojų ir kaupiamųjų indėlių palūkanų normų lentelės
- Terminuotojų indėlių palūkanų sumos skaičiuoklė
- Kaupiamųjų indėlių palūkanų sumos skaičiuoklė
- Valiutos keitimo skaičiuoklė
- Kredito įmokos skaičiuoklė
- Palyginamoji atlyginimo ir mokesčių skaičiuoklė
- Palyginamoji autorinio atlyginimo ir mokesčių skaičiuoklė
- Informacija apie projektą bei kūrėją

Sistemoje yra sukurtas servisas periodiškai atnaujinantis bankų valiutų kursus, terminuotojų bei kaupiamųjų indėlių palūkanų normas. Duomenys yra gaunami iš banko oficialių tinklalapių, kuriuose skelbiami vieši duomenys, todėl tokių duomenų naudojimas nepažeidžia įstatymų.

Pateikiamas informacinio pobudžio Lietuvoje esnčių bankų sarašas ir detali šių bankų rekvizitų bei filialų informacija, leidžianti vartotojui greitai rasti artimiausią ar tinkamiausią banką ir su juo sukontaktuoti.

Visi automatiškai atnaujinami bankų duomenys (kursai ir normos) yra pateikiami informacinio pobudžio lentelėse, leidžiančiose pilnai peržiūrėti norimo banko duomenis. Pateikiama informaciją apie sistemos kūrėją ir apie patį projektą.

3.13. Testavimo strategija

Kad sistema turėtų kiek įmanoma mažiau klaidų, didelė dalis sistemos kūrimo laiko buvo leidžiama sistemos testavimui. Nuspręsta sistema kurti naudojant TTD (Test Driven Development) programų kūrimo metodologiją, pagal kurią pirmiau yra rašomas testas, o tik poto programos kodas. Pagrindinis šios metodologijos principas yra, tai kad testas diktuoja sistemos kodo kūrimą, taip leidžiant išvengti nereikalingo kodo, kuris gali būti taip ir nepanaudotas sistemoje. Toks kūrimas leidžia sumažinti galimų laidų kiekį.

3.13.1. Vienetų testavimas

Visi vienetų testai buvo atlikti naudojant baltos dėžės testavimą. Buvo testuojama ar objektai yra sukuriami, ar teisingai priskiriamos objektų savybės ir reikšmės, ar teisingai atliekami vykdomi skaičiavimai. Ar teisingai nukreipiamos nuorodos. Buvo stebimos įėjimo ir išėjimo sąlygos. Testai buvo rašomi ir vykdomi Visual Studio integruotoje testavimo aplinkoje, buvo matoma kuris testas praėjo, kuris ne.

3.13.2. Integracijos testavimas

Buvo testuojama ar sistema gerai integruojasi su duomenų bazės sąsaja. Taip pat testuojama ar gaunami duomenys iš duomenų bazės, ar sistema veikiant naršyklėse nekonfliktuoja su kitais naršyklės komponentais ir nestrigdo naršyklės darbo.

3.13.3. Priėmimo testavimas

Buvo tikrinama ar sistemos atliekamos funkcijos atitinka sistemos reikalavimų specifikaciją, ar sistema vykdo visas užsakovui reikalingas funkcijas. Tam buvo naudojamas juodos dėžės testavimo metodas, patikrinti ar sistema teisingai vykdo skaičiavimus. Padavus užsakovo duomenis buvo stebimi gauti rezultatai ir lyginami su tikrais.

3.13.4. Aukšto lygio testavimas

Norint užtinkrinti sistemos saugumą, buvo atliekamas keleto pagrindinių saugumo spragų testavimas. Testuojama nuo SQL įterpimo ir skriptų įterpimo atakų. Taip pat testuojama ar apsaugoti failų serverio katalogai, nuo išorinės neautorizuotos prieigos.

Naudojant į Microsoft Visual Studio integruotus ir trečių šalių kodo profiliavimo įrankius buvo testuojamas kodo veikimo greitumas, kad galima būtų optimizutoti lėtus sitemos algoritmus ir taip padidinti sistemos veikimo greitį.

4. EKSPERIMENTINIS TYRIMAS

4.1.Projekto ir realizacijos tyrimai

Norint žinoti kokios galimybės automatiškai surinkti duomenis iš Lietuvos bankų, turime aprašyti kiekvienos iš, anksčiau darbe aprašyto, automatinio duomenų surinkimo proceso modelio dalių galimybes. Tai reškia ištirti kokiais būdais bankai pateikia informaciją ir tai įtakos apdorojimo ir saugojimo būdo pasirinkimą.

4.1.1. Duomenų pateikimo būdų tyrimas

Lietuvos bankų asociacijoje yra įregistruoti 11 oficialių bankų ar jų padalinių Lietuvoje [19]. Tyrimas buvo vykdomas visiems šiems bankams:

- AB „Swedbank“
- AB bankas „Snoras“
- AB DnB NORD bankas
- UAB Medicinos bankas
- Nordea Bank Finland Lietuvos skyrius
- AB Parex bankas
- Danske Bank A/S Lietuvos filialias
- AB SEB bankas
- AB Šiaulių bankas
- AS „UniCredit Bank“ Lietuvos skyrius
- AB Ūkio bankas

Atitinkamų bankų naudojami finansinių duomenų pateikimo viešoje erdvėje būdai (Jei būdas egzistuoja bus žymimas pliuso ženklas, jei ne - minusas)

Pateikimo būdas / Bankas	Interneto svetainėje	Skaitmeniniai dokumentai	Sklaidos kanalai	Web Servisai
AB „Swedbank“	+	+	+	-
AB bankas „Snoras“	+	+	-	-
AB DnB NORD bankas	+	+	-	-
UAB Medicinos bankas	+	+	-	-
Nordea Bank Finland Lietuvos skyrius	+	+	-	-
AB Parex bankas	+	+	-	-
Danske Bank A/S Lietuvos filialias	+	+	+	-
AB SEB bankas	+	+	-	-
AB Šiaulių bankas	+	+	-	-
AS „UniCredit Bank“ Lietuvos skyrius	+	+	-	-
AB Ūkio bankas	+	+	-	-

Lentelė 1. Lietuvos bankų naudojami duomenų pateikimo būdai

Pagal atliktą tyrimą matome, kad visi bankai teikia duomenis savo informacinėse sistemose, bei papildomai pateikia kai kurią finansinę infomaciją skaitmeninių dokumentų forma. Nei vienas iš tirtų bankų nenaudoja ir neskelbia apie galimybę gauti duomenis naudojant web servisus. Tik „Swedbank“ ir „Danske Bank“ bankai teikia finansinę informaciją RSS sklaidos kanalų pagalba. Tai būtų galima paaiškinti, tuo kad Lietuvos bankai neturi labai dažnai atnaujinamos finansinės informacijos arba jei tokia informacija egzistuoja, jie pasirenka jų pateikimą interneto tinklalapyje.

Pagal šį tyrimą galima daryti išvadą, kad duomenų surinkimui geriausiai tinka HTML formatu tinklalapiuose pateikiami duomenys, todėl jų apdorojimui reikia rinktis vieną iš galimų HTML analizės būdų, o apdorotus duomenis naudingiausia saugoti duomenų bazėje, kadangi tokiu būdu juos yra lengviau panaudoti kuriant programinę įranga.

Aprašytu būdu įgyvendinus automatinį duomenų surinkimą, sistemai automatiškai atsinaujinant iškilo svarbus DOM analizės trūkumas, stipriai įtakojantis surinkimo proceso kokybę. Pasikeitus tinklalapio struktūrai surinkimas iš to banko sistemos nutrūkdavo užfiksavus klaidą, kadangi duomenų blokų išrinkimas negalėjo vykti teisingai dėl pasikeitusio objektų modelio arba pasikeitusio puslapio adreso. Per 4 sistemos veikimo mėnesius vidutiniškai 3 nutrūkimai per mėnesį 9 bankų sistemoms. Stabiliausiai veikė ir nei karto atnaujinimo darbas nesutriko Swedbank tinklalapyje skelbiamiems duomenims, kadangi duomenis, kuriuos as rinkau jie pateikia specialiai naujame puslapyje, kuris visada išlaiko vienodą struktūrą.

4.1.2. Duomenų surinkimo greitaveikos tyrimas

Taip pat buvo atliktas tyrimas duomenų surinkimo vidutinei greitaveikai paskaičiuoti atskiriems bankams, esant šioms techninėms sąlygoms:

- Dviejų branduolių Athlon Procesorius, taktinis dažnis 2,26Ghz
- 3GB darbinės kompiuterio atminties
- 1.5 Mbit/s interneto ryšys

Tyrimas buvo atliktas 9 bankams, kurie buvo realizuoti automatiniame surinkime.

Rezultatai pateikti lentelėje žemiau:

Bankas	**Atnaujinimo greitaveika, s**
AB „Swedbank“	0,057
AB bankas „Snoras“	0,168
AB DnB NORD bankas	0,054
UAB Medicinos bankas	0,102
Nordea Bank Finland Lietuvos skyrius	0,087
AB Parex bankas	0,234
Danske Bank A/S Lietuvos filialias	0,057
AB SEB bankas	0,078
AB Šiaulių bankas	0,051
Bendra pilno atnaujinimo trukmė	**0,888**

Lentelė 2. Lietuvos bankų duomenų surinkimo vidutinės greitaveikos

Iš tyrimo rezultatų matome, kad surinkimas truko mažiau nei vieną sekundę bendrai sudėjus. Tokia greitaveika pasiekta naudojant palyginti lėtą surinkimo būdą, kadangi atliekama turinio analizė. Sprendžiant pagal šiuos rezultatus sukurta sistema būtų pajėgi surinkti duomenis atnaujinamus net kas 1 sekundę. Jei sistema gautų duomenis iš sklaidos kanalų ar web servisų, atnaujinimo greitaveikų būtų žymiai mažesnė, dėl paprastesnių

4.2.Sukurtos sistemos kokybės analizė

Analizės tikslai:

- Aptikti klaidas funkcionavime, logikoje, realizacijoje
- Patikrinti ar programų sistema atitinka reikalavimų specifikaciją
- Įsitikinti ar programų sistema sukuta pagal standartus

Kokybės įvertinimas

- Klaidų sistemos funkcionavime, bei logikoje nerasta. Tam pasiekti buvo konsultuojamasi su banko darbuotojais, kad sužinoti kaip atliekami skaičiavimai. Realizacijoje klaidų taip pat nerasta, tačiau ateityje, siūlytina optimizuoti kai kurias sistemos dalis, kad būtų pasiekta geresnė sistemos greitaveika.
- Sukurta sistema atitinka reikalavimų specifikaciją, tačiau sistemos metu, ne visus reikalavimus spėta įgyvendinti, todėl jie palikti tikintis sistemą pilnai realizuoti ir įdiegti ateityje.
- Sistemą buvo stengiamasi kurti pagal esamus standartus. Dėl pasirinktų technologinių sprendimų sistema yra lengvai pernešama, palaikoma ir patikima. Sistemą lengva naudoti.

- Sistemoje galimas veikimo efektyvumo bei funkcionalumo tobulinimas, jei sistema plėsis. Šiuos trūkumus įtakoja sistemos kūrėjo patirtis dirbant su panašiomis technologijomis, bei palyginti nedidelė sistemos apimtis šioje kūrimo stadijoje.

Galimų sistemos patobulinimų sąrašas:

- Detalesnis duomenų surinkimas įvertinant išimtinius atvejus
- Sistemos papildymas naujomis skaičiuoklėmis
- Grafinis skaičiavimo rezultatų atvaizdavimas
- Vartotojo funkcijų praplėtimas

4.3.Eksperimentinio tyrimo vertinimas

Praktiškai išbandėme HTML dokumentų apdorojimą pasitelkiant dokumento DOM medžio analizę. Šis darbas aiškiai parodė, kad HTML dokumentų analizavimas iš HTML kodo dažnai sukelia problemų, nes pasikeitus tinklalapio, iš kurio duomenis imame, struktūrai paprogramė vykdanti analizę užfiksuoja klaidą dėl pasikeitusio DOM medžio struktūros. Šį metodą galima patobulinti įvedus tam tikrų dokumento elementų paiešką, tačiau tai bus veiksminga tik jeigu tinklalapis žymėms naudos HTML kalbos ID atributą, pagal kurį būtų galima atrinkti duomenų blokus, net jei išorinė struktūra aplink blokus yra pasikeitusi. Visgi tinkamiausias sprendimas būtų duomenų surinkimui naudoti XML kalba pagrįstą duomenų tiekimo formatą arba web servisus, leidžiančius tiksliai gauti informacija be papildomos dokumento analizės. Bet Lietuvos bankai tokia forma duomenų nepateikia, todėl tenka naudoti mažiau stabilius ir didesnę klaidos tikimybę surenkant turinčius duomenų formatus.

5. IŠVADOS

1) Atlikta automatinio duomenų surinkimo analizė ir nustatytos surinktų duomenų panaudojimo galimybės.
2) Apžvelgti finansinių duomenų pateikimo viešoje erdvėje būdai, paaiškintas duomenų surinkimo modelis, bei detaliai aprašytos sudedamosios modelio dalys.
3) Apibrėžti būdai HTML dokumentų apdorojimui, naudojant HTML nagrinėjimą skirtingais metodais. Taip pat apibrėžti XML dokumentų korektiškumo nustatymo, nagrinėjimo, duomenų transformavimo bei išgavimo būdai.
4) Išanalizuotos galimybės automatiškai surinkti Lietuvos bankų teikiamus finansinius duomenis.
5) Įgyvendintas automatinis 9 Lietuvos bankų finansinių duomenų surinkimas panaudojant HTML DOM medžio analizę. Atliktas realizacijos greitaveikos tyrimas.
6) Sukurta informacinė finansinių skaičiuoklių sistema remiantis ASP.NET MVC karkasu, kuris leidžia atskirti logines sistemos dalis taip palengvinant sistemos pakeitimų įgyvendinimą.
7) Įgyvendinus sistemą pastebėti pasirinkto duomenų surinkimo metodo trūkumai.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

Terminas/Santrumpa	**Paaiškinimas**
IS (angl. Information System)	Informacinė sistema
HTML (angl. Hypertext Markup Language)	Hiperteksto žymių kalba
XML (angl. eXtensible Markup Language)	Išplėsta žymių kalba
W3C	Pasaulinio tinklo konsorciumas
CSV (angl. Comma Separated Values)	Kablelių atskirų reikšmių formatas
RSS (angl. Really Simple Syndication)	Sklaidos kanalų formatas
Atom	Sklaidos kanalų formatas
API (Application Programing Interface)	Aplikacijų programavimo sąsaja
DOM (Document Object Model)	Dokumento objektų modelis
XQL (XML Query Language)	XML užklausų kalba
HTQL (HTML Query Language)	HTML užklausų kalba
HTTP (Hypertext Transfer Protocol)	Hiperteksto perdavimo protokolas
WSDL (Web Services Description Language)	Web servisų aprašymo kalba
SOAP (Simple Object Access Protocol)	Paprastų objetų prieigos protokolas
REST (Representational state transfer)	Reprezentacinės būsenos perkėlimas – architektūros tipas
SAX (Simple API for XML)	Paprasta API sąsaja XML kalbai
DTD (Document Type Definition)	Dokumento tipo aprašas
XSL (eXtensible Stylesheet Language)	Praplėsta stilių kalba
XSLT (eXtensible Stylesheet Language Transformation)	Praplėstos stilių kalbos transformacija
MSSQL (Microsoft SQL Database)	Microsoft kompanijos duomenų bazės tipas
ASP.NET	Microsoft kompanijos technologija interneto sistemų kūrimui
MVC (Model-View-Controller)	Modelio-Vaizdo-ontrolerio šablonas

LITERATŪRA

1. LST ISO 2382-1. *Informacijos technologija. Terminai ir apibrėžimai. 1-oji dalis. Pagrindiniai terminai.* s.l. : Lietuvos standartizacijos departamentas, 1996 m.

2. **Weglarz, Geoffrey.** Two Worlds of Data – Unstructured and Structured. *Information Management Magazine.* 2004 m.

3. Wikipedia. *Structured Data.* [Tinkle] http://en.wikipedia.org/wiki/Structured_data.

4. Interactive Financial eXchange Forum. *About IFX.* [Tinkle] http://www.ifxforum.org/about/.

5. Open Financial Exchange. [Tinkle] http://www.ofx.net/.

6. FIX Protocol. *What is Fix.* [Tinkle] http://www.fixprotocol.org/.

7. eXtensible Business Reporting Language. *What is XBRL.* [Tinkle] http://www.xbrl.org/WhatIsXBRL/.

8. **Mačiulaitis, Laurynas.** Optimizavimas paieškos sistemoms. [Tinkle] http://www.searchengineoptimization.lt/index.php/kas-yra-rss/.

9. **Libby, Dan.** *RSS 0.91 Spec, revision 3.* s.l. : Netscape Communications, 1999 m. 07 10 d.

10. **Orenstein, David.** QuickStudy: Application Programming Interface (API). *ComputerWorld.* [Tinkle] 2000 m. 01 10 d. http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=43487.

11. W3C. *Web Services Glossary.* [Tinkle] 2004 m. 02 11 d. http://www.w3.org/TR/ws-gloss/.

12. **W3C Web Services Architecture Working Group.** Web Services Architecture. [Tinkle] 2004 m. http://www.w3.org/TR/ws-arch/.

13. **Fielding, Roy Thomas.** Architectural Styles and the Design of Network- based Software Architectures. s.l. : University of California, 2000 m.

14. Wikipedia. *Representational State Transfer.* [Tinkle] http://en.wikipedia.org/wiki/Representational_State_Transfer.

15. GeoSeeker. *What is FreeFormat.* [Tinkle] http://www.gooseeker.com/en/node/knowledgebase/freeformat.

16. SAX. *Events vs. Trees.* [Tinkle] http://www.saxproject.org/event.html.

17. XML Path Language (XPath). [Tinkle] http://www.w3.org/TR/xpath.

18. XQuery 1.0: An XML Query Language. [Tinkle] http://www.w3.org/TR/xquery/.

19. Lietuvos bankų asociacija. *Nariai.* [Tinkle] http://www.lba.lt/index.php/lt/26952/.

20. W3C. *XSL Transformations (XSLT).* [Tinkle] http://www.w3.org/TR/xslt#section-Introduction.