MOKSLAS – LIETUVOS ATEITIS
SCIENCE – FUTURE OF LITHUANIA

ISSN 2029-2341 print / ISSN 2029-2252 online
2012 4(1): 31–34 doi:10.3846/mla.2012.07

Sistemų inžinerija, kompiuterių technologija T 120
System Engineering, Computer Technology T 120

Elektronika ir elektrotechnika
Electronics and Electrical Engineering

ŠIUOLAIKINIŲ TINKLALAPIŲ AUTOMATIZUOTAM NARŠYMUI IR TESTAVIMUI SKIRTŲ PRIEMONIŲ ANALIZĖ IR PRITAIKOMUMAS INFORMACIJAI RINKTI

Tomas Grigalis[1], Leonardas Marozas[2], Lukas Radvilavičius[3]

Vilniaus Gedimino technikos universitetas

El. paštas: [1]tomas.grigalis@vgtu.lt; [2]leonardas.marozas@vgtu.lt; [3]lukas.radvilavicius@vgtu.lt

Santrauka. Internetui tapus milžiniška informacijos duomenų baze, susiduriama su informacijos rinkimo problema – kaip iš itin gausaus kiekio informacijos šaltinių pasirinkti tokį, kuris gebėtų informacijos naudotojui pateikti tinkamą ir jį dominančią aktualią informaciją. Taip pat svarbu gebėti analizuoti šiuolaikinius tinklalapius saugumo prasme ir ieškoti juose, pavyzdžiui, įterpto slapto kenkėjiško kodo, o tai galima padaryti tik surinkus informaciją iš tinklalapio. Be to, nauja WEB 2.0 interneto karta priverčia keisti įprastinius informacijos rinkimo metodus, nes *Flash*, *Javascript*, *Ajax* ir kitos naujos technologijos trukdo surinkti informaciją vien tik analizuojant įprastą HTML kodą. Šiame straipsnyje analizuojamos sudėtingų šiuolaikinių tinklalapių naršymo automatizavimui ir testavimui skirtos priemonės, kurios gali būti panaudotos informacijai rinkti.

Reikšminiai žodžiai: informacijos rinkimas, dinamiški tinklalapiai, automatinis naršymas, *Quick Test Pro*, *Sahi*, *Selenium*, *Telerik*, *TestComplete*, *Watir*, *Windmill*.

Įvadas

Internetui tapus milžiniška informacijos duomenų baze, susiduriama su informacijos pertekliaus problema – kaip iš itin gausaus kiekio informacijos šaltinių pasirinkti galintį informacijos naudotojui pateikti tinkamą ir jį dominančią aktualią informaciją. Taip pat svarbu gebėti analizuoti šiuolaikinius tinklalapius saugumo prasme ir ieškoti juose, tarkim, įterpto slapto kenkėjiško kodo.

Tokios informacijos paieškos sistemos, kaip *Google*, *Bing*, *Altavista* ir daugelis kitų, iki šiol gana sėkmingai sprendė šią problemą, tačiau interneto naršytojai laipsniškai pradėjo reikalauti dar konkretesnės informacijos – būtent to, ko jiems reikia, ir iš ne vieno patikimo šaltinio. Atsirado metapaieškos ir vertikaliosios paieškos sistemos, kurios iš gausybės interneto portalų ieško ir vartotojui pateikia konkrečią, filtruotą informaciją, o ne tiesiog visą puslapį ar portalą, kaip tai daro *Google*.

Taigi, tiek didžiųjų informacijos paieškos korporacijų, tiek pasaulio mokslininkų (Xia 2009; Duda *et al*. 2009; Mesbah *et al*. 2008; Baumgartner *et al*. 2009) dėmesys nukrypo į informacijos rinkimo ir skirtingų šaltinių integracijos problemų nagrinėjimą (angl. *web information extraction*, *schema matching*). *Microsoft* kuria *Academic search*, *Google* tobulina ekspermentinės produktų paieškos sistemą *Froogle*, mokslinių straipsnių – *Google Scholar*.

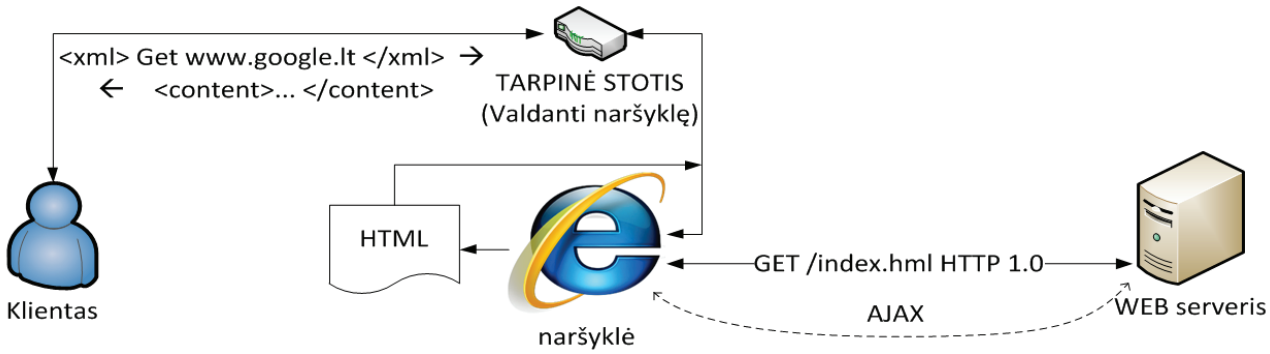
Nors informacija, esanti tinklalapiuose, žmogui vizualiai atrodo vienoda ir gana lengvai suprantama, programiniu požiūriu jos atvaizdavimas yra sudėtingas, o įgyvendinimas (programavimas) gali būti visiškai skirtingas. Tai reiškia, kad skirtingu programiniu kodu, pavyzdžiui, HTML kodu, galima pateikti vizualiai vienodą informaciją. Šiuolaikiniai tinklalapiai yra ypač sudėtingi. Pavyzdžiui, CNN pagrindinis puslapis naršyklėje yra sugeneruojamas iš atsisiųstų 53 statiškų paveikslėlių, 39 dinamiškų paveikslėlių, trijų FLASH rinkmenų, 30 *JavaScript* rinkmenų iš septynių skirtingų domenų, 29 *HTML* bylų ir septynių *CSS* (angl. *Cascading Style Sheet*) dokumentų. Žinoma, vartotojai naršydami to nepastebi, nes šiuolaikinės interneto naršyklės viską daro automatiniu foniniu režimu.

Iš dinamiškų tinklalapių surinkta informacija gali būti naudojama labai plačiai: ją galima analizuoti saugumo prasme ir ieškoti kenkėjiško programinio kodo, galima rinkti ir indeksuoti.

Straipsnyje pateikta 14 priemonių, skirtų sudėtingų šiuolaikinių tinklalapių naršymui automatizuoti ir testuoti, jos taip pat gali būti naudojamos informacijai rinkti. Šios priemonės varijuoja nuo visiškai paprastų, nemokamų ir lengvų iki tūkstančius kainuojančių sistemų, gebančių atlikti gausybę veiksmų. Analizės esmė yra pateikti informaciją apie šias priemones, apžvelgti jų pritaikymo galimybes išgaunant informaciją iš aplikacijų, įvertinti jų veikimo tikslumą, paprastumą ir panaudojimo galimybes.

Šiuolaikinių dinamiškų tinklalapių naršymo metodika

Tradiciškai tinklalapio programinis kodas gaunamas prisijungus prie serverio ir atsisiuntus failą. Naujosios *WEB 2.0* technologijos suteikia galimybes kurti dinamiškus interneto



Pav. Šiuolaikinių tinklalapių *HTML* išeities kodo gavimas per internetinę naršyklę
Fig. Reception of the *HTML* source code via the modern Internet web browser

puslapius, kur informacija ir duomenys yra atnaujinami pasitelkiant *Javascript* programavimo kalbą. Dinaminiuose tinklapiuose per vieną užklausą gaunama tik dalis turinio. Kita dalis formuojama naršyklėje (*Javascript*), papildomai asinchroniškai (*Ajax*) atsisiunčiami duomenys iš serverių (Kriučkova 2007).

Norint matyti tokį puslapio vaizdą, kurį mato vartotojas per interneto naršyklę, reikia būtinai įvykdyti visas *Javascript* programėles, esančias *HTML* kode. Todėl tinklalapius visų pirma būtina įkelti šiuolaikine naršykle (žr. pav.), kuri atlieka visą darbą: surenka paveikslėlius, *Javascript* kodus ir juos įvykdo. Tuomet jau pilnutinai įkeltą tinklalapį galima analizuoti, simuliuoti vartotojo veiksmus.

Tinklalapių automatizuotam naršymui ir testavimui skirtų priemonių techninė analizė

Analizei pasirinktos septynios svetainių informacijos gavybos priemonės. Jos atrinktos remiantis vartotojų rekomendacijomis ir naudojimo populiarumu, atliekant tokio pobūdžio veiklą.

Analizės metodika. Pateiksime programinių priemonių įvertinimą pagal iš anksto apsibrėžtus kriterijus:

- greitis;
- gautų duomenų tikslumas;
- prisitaikymas prie svetainių pokyčių;
- gebėjimas apdoroti *JavaScript*;
- gebėjimas apdoroti *Ajax*;
- gebėjimas apdoroti *Flash*;
- įgyvendinimo paprastumas;
- palaikymas;
- kaina;
- platformos;
- ataskaitų aiškumas.

Taip pat buvo atsižvelgiama į: demonstracinės versijos buvimą, galimybę programuoti įskiepius, palaikomas naršykles, *Javascript* variklį, programavimo kalbas, kuriomis programuojami skriptai.

Analizės metu iš galutinio vartotojo pozicijos santykinėje skalėje iki 10 balų įvertinamos programinių paketų galimybės atsižvelgiant į minėtus kriterijus. Be abejo, kai kuriuos paketus vieną su kitu sudėtinga lyginti, pvz., atvirojo kodo specializuotą testavimo aplinką su korporacijos plėtojamu brangiu ir universaliu produktu. Atliekant analizę dėmesys buvo sutelktas ties duomenų gavimu iš internetinės aplinkos ir jų korektiškumu, tad dažnai brangios ir į universalumą orientuotos programos įvertintos žemiau nei paprastos greitai ir korektiškai užduotį atliekančios atvirojo kodo programos. Lentelėje pateikti atliktos analizės įvertinimai.

Quick Test Pro yra HP sukurtas produktas, skirtas funkciniams ir regresijos testams automatizuoti. Jis naudoja *VBScript* kalbą skriptams kurti ir manipuliuoti objektais. Dažniausiai šis įrankis naudojamas vartotojo sąsajai testuoti. Patikrai naudojamos tam tikros gairės (9 tipų), kuriomis remdamasis vartotojas gali nustatyti, ar funkcionalumas yra toks, koks numatytas, ar ne – duomenys lentelėje, įrašai duomenų bazėje, paveikslėliai ir pan. Taip pat galima apibrėžti savo gaires. *Quick Test Pro* turi keletą plėtinių, pritaikytų *Web*, *.NET*, *Java* ir *Delphi*. Ataskaitos pateikiamos keliais formatais – *HTML*, *doc, PDF*. Viena iš neigiamų šio programinio paketo savybių yra ta, kad jis veikia tik *Windows* operacinėje sistemoje. Jis negali testuoti emuliuodamas šių naršyklių: *Opera*, *Safari* ar *Chrome*. Be to, šis įrankis nemažai kainuoja. *Quick Test Pro* vartotojams nepateikia demonstracinės versijos, skirtos išmėginti. Nėra galimybių papildomai suprogramuoti savo įskiepių prie šio įrankio. Vienas didžiausių įrankio trūkumų yra ribotas palaikomų naršyklių skaičius.

Sahi – informacijos įkėlimo iš puslapių automatizavimo įrankis. Šis įrankis įterpia *JavaScript* naudodamas *proxy* į internetinius puslapius, suteikdamas galimybę juos automatizuoti. Pagrindiniai pastebėti šio įrankio trūkumai: freimai / puslapiai su *frames* / *iframes* nepalaikomi iš keleto domenų; problemos, susijusios su failų siuntimu, naudojant *Javascript*; *Sahi* vartotojams nepateikiama demonstracinės versijos. Tačiau įrankis pasižymi galimybėmis programuoti įskiepius automatinei gavybai iš internetinių resursų. *Sahi* neapsiriboja veikimu tik tam tikrose naršyklėse. Sistema yra nepriklausoma. *Sahi* naudoja *Rhino* pagrįstą *Javascript* variklį, kuris apdoroja skriptus ir kuriuo remiasi visos sistemos veikimas.

Selenium – vienas iš populiariausių ir geriausių įrankių, skirtų duomenų gavybai iš internetinių puslapių automatizuoti. Tai nemokamas atvirojo kodo įrankis. Egzistuoja dvi *Selenium* variacijos – IDE, įskiepis naršyklei ir *Webdriver*, specifinių komandų naršyklės valdymo rinkinys. IDE skirtas nedideliems skriptams, klaidoms atkurti ir automatizuoti. Šis įrankis išskirtinai gerai naudojamas *Ajax web* puslapiuose. Tiesa, dėl ypatybių gausos *Selenium* kiek nukenčia greičio atžvilgiu, lyginant su *Canoo* ir kitais į greitį orientuotais testavimo įrankiais. Norint korektiškai analizuoti *Ajax* / *CSS* puslapiuose, reikia turėti gerų programavimo žinių, nes automatizavimas yra ganėtinai sudėtingas. *Selenium* veikia *Linux*, *Windows* ir *Macintosh* platformose. Taip pat egzistuoja ir primityvi skriptų kalba, skirta *IDE – Selenese*, tačiau dėl jos primityvumo ir pernelyg didelio paprastumo pritaikymas tampa gana sudėtingas. *Selenium* nesiūlo vartotojams išmėginti demonstracinės versijos. *Selenium* yra lengvai plečiamas ir tai yra vienas iš didžiausių privalumų lyginant su kitais automatizavimo sprendimais, kurie buvo pateikti atliekant šią analizę.

Telerik Test Studio – mokamas, gana nesudėtingas įrankis, skirtas užduotims automatizuoti testuojant internetinius puslapius ir aplikacijas. *Test Studio* geba automatizuoti *HTML*, *Ajax*, *Silverlight* ir *WPF* aplikacijas. Mėginant kuo labiau supaprastinti testų kūrimą, vartotojui faktiškai nereikia programavimo žinių, norint sukurti testus, tačiau testai būna gana riboto taikymo. Testai gali būti taikomi kelioms populiariausioms naršyklėms. Viena iš stiprių jų šio įrankio pusių yra ta, kad nenaudojant absoliučių koordinačių, testavimo aplinka yra pakankamai pakanti pasikeitimams puslapiuose. *Telerik Test Studio* galimi du variantai – kaip savarankiška versija ir kaip įskiepis *Visual Studio*. Pastarasis variantas labiau skirtas programuotojams, kurie linkę patys testuoti.

TestComplete – įrankis, skirtas programinei įrangai *Windows* aplinkoje testuoti. Šis įrankis gali testuoti programas, parašytas *Flash*, C++ ar *Visual ForPro*. Egzistuoja 9 rūšių testai, kuriais galima automatizuotai testuoti, pvz., GUI testai, regresijos ar duomenų testai. *TestComplete* pasižymi išsamiomis ataskaitomis, kurios gali būti XML, HTML ar MHT formatų. Taip pat *TestComplete* turi nemažą skaičių įrankių, skirtų *Visual Studio* sukurtoms programoms testuoti. Šis įrankis lengvai plečiamas, be to, kartu su juo pateikiama daug dokumentacijos. *TestComplete* yra galingas įrankis, skirtas ne vien tik interneto aplikacijoms testuoti. Juo galima įrašyti ir kartoti automatizuotus testus. Kai kurios testų rūšys, pvz., pagrįstos raktiniais žodžiais, yra nesudėtingai rengiamos faktiškai be jokių programavimo žinių.

Watir – nemokamas atvirojo kodo įrankis iš *Ruby* bibliotekų šeimos, skirtas interneto aplikacijų automatizuotam testavimui. Šis įrankis imituoja žmogaus elgseną naršyklėje, pvz., spaudo nuorodas ar įvedinėja formas. Nepaisant to, kad *Watir* remiasi *Ruby*, jis geba analizuoti aplikacijas, sukurtas kitomis kalbomis. Egzistuoja dvi *Watir* versijos: *Watir*, palaikanti tik *Internet Explorer Windows* aplinką, ir *Watir-WebDriver*, palaikanti ir kitas populiariąsias naršykles bei *HTMLUnit* režimą.

Windmill – automatinio testavimo sistema, skirta interneto aplikacijoms. Ši aplinka gali testuoti aplikacijas iš vienos sąsajos. *Windmill* pagrįstas *Python* programavimo kalba. Naršykle galima bendrauti per interaktyviąją *Python* aplinką. *Windmill* turi savo *proxy*, kuris leidžia testuoti esant situacijoms, kai statinis puslapio turinys nepasiekiamas; *Javascript* ir *Junit* galima rašyti testus ir gauti ataskaitas *JUnit* formatu. Kontrolinė API visiškai simuliuoja žmogaus veiksmus internetinėje aplikacijoje. *Windmill* nepalaiko *Flash*.

Išvados

Išanalizuoti pagrindiniai informacijai iš internetinių puslapių gauti dažniausiai naudojami įrankiai (lentelė). Kadangi atliekant analizę buvo orientuojamasi į informacijos gavimą būtent iš internetinių puslapių ir aplikacijų, tai dar nereiškia, kad lentelėje pateikti duomenys apie kai kuriuos produktus vertinami kaip prastesni nei kiti. Žemesnis balas reiškia, kad jų pritaikymas numatytoms informacijos gavyboms užduotims ir įrankiams įgyvendinti yra pernelyg sudėtingas ar brangus. Vienareikšmiškai teigti, kad tam tikras programinis paketas yra geresnis nei kiti negalima, nes vieni jų yra orientuoti į greitį, kiti – į universalumą, treti – į pateikiamų duomenų tikslumą ar testų kūrimo ir pritaikymo paprastumą.

Geriausiai analizės reikalavimus atitiko *Selenium* programinis paketas, nes:

- *Selenium* faktiškai nenusileidžia greičiu atliekant testus kitiems analogiškiems duomenų gavybos įrankiams, o didžiąją dalį jų lenkia.

Lentelė. Analizės metu surinktos informacijos apie šiuolaikinių tinklalapių automatinio naršymo priemones išklotinė

Table. Summary of the collected information about automatic web crawling and testing

	Greitis	Gautų duomenų tikslumas	Prisitaikymas prie svetainių pokyčių	Gebėjimas apdoroti *JavaScript*	Gebėjimas apdoroti *Ajax*	Gebėjimas apdoroti *Flash*	Įgyvendinimo paprastumas	Palaikymas	Kaina, JAV dol.	Platformos	Ataskaitų aiškumas	Bendras
Quick Test Pro	5	6	6	5	4	2	6	7	–	Windows	8	49
Sahi	8	8	7	8	8	5	7	8	0	Visos	8	67
Selenium	8	9	8	9	9	7	7	10	0	Visos	8	75
Telerik	7	8	9	7	8	5	7	8	2 499	Visos	7	67
TestComplete	8	8	8	7	7	6	6	8	2 000	Windows	8	66
Watir	9	8	7	8	8	5	7	9	0	Visos	7	68
Windmill	7	7	8	8	7	0	6	8	0	Visos	6	57

- *Selenium* įrankiu gauti duomenys yra vieni iš atspariausių puslapio išdėstymo pokyčiams, o ataskaitose esantys pranešimai – pakankamai išsamūs, kad leistų greitai susigaudyti, kurioje vietoje suklysta planuojant ir programuojant testinius skriptus.
- Šis įrankis geriausiai veikė su *Javascript* ir *Ajax* technologijomis. *Flash* technologiją *Selenium* taikė vidutiniškai, tačiau lyginant su kitais apžvelgtais įrankiais, rezultatai buvo vieni iš geriausių.
- *Selenium IDE* įdiegimas yra toks pat paprastas kaip bet kokio įskiepio naršyklei įdiegimas. Tam nereikia jokių techninių žinių ar gebėjimų. Kitos *Selenium* įrankio dalys taip pat įdiegiamos itin paprastai.
- Tai populiarus įrankis, todėl daug aspektų jau yra aptarta šaltiniuose, apstu žmonių, kurie aktyviai palaiko *Selenium* bendruomenę ir teikia konsultacijas. Tai atvirojo kodo nemokamas programinis įrankis, pagal galimybes konkuruojantis, o kai kur ir nurungiantis mokamas programas. *Selenium* veikia *Windows*, *Unix* ir *Mac* operacinėse sistemose.
- *Selenium* sistema surinkta informacija gali būti analizuojama saugumo prasme: joje galima ieškoma įterpto kenkėjiško kodo.

Literatūra

Baumgartner, R.; Gottlob, G.; Herzog, M. 2009. Scalable web data extraction for online market intelligence, in *Proceedings of the VLDB Endowment* 2(2): 1512–1523. VLDB Endowment.

Duda, C.; Frey, G.; Kossmann, D.; Matter, R.; Zhou, C. 2009. AJAX crawl: making AJAX applications searchable, in *IEEE 25th International Conference on Data Engineering*, 7889. IEEE.

Kriučkova, M. 2007. AJAX – naujas požiūris į žiniatinklio aplikacijų kūrimą, iš *Kompiuterinė grafika ir projektavimas 10-osios Lietuvos jaunųjų mokslininkų konferencijos „Mokslas – Lietuvos ateitis“ medžiaga.* Vilnius: Technika, 93–99.

Mesbah, A.; Bozdag, E.; Deursen, A. V. 2008. Crawling AJAX by inferring user interface state changes, in *Eighth International Conference on Web Engineering*, 122–134. IEEE.

Xia, T. 2009. Extracting structured data from Ajax site, in *First International Workshop on Database Technology and Applications*, 259–262. IEEE.

ANALYSIS OF AUTOMATED MODERN WEB CRAWLING AND TESTING TOOLS AND THEIR POSSIBLE EMPLOYMENT FOR INFORMATION EXTRACTION

T. Grigalis, L. Marozas, L. Radvilavičius

Abstract

World Wide Web has become an enormously big repository of data. Extracting, integrating and reusing this kind of data has a wide range of applications, including meta-searching, comparison shopping, business intelligence tools and security analysis of information in websites. However, reaching information in modern WEB 2.0 web pages, where HTML tree is often dynamically modified by various JavaScript codes, new data are added by asynchronous requests to the web server and elements are positioned with the help of cascading style sheets, is a difficult task. The article reviews automated web testing tools for information extraction tasks.

Keywords: data extraction, automated crawling, web testing, dynamic webpages, Quick Test Pro, Sahi, Selenium, Telerik, TestComplete, Watir, Windmill.