АВТОМАТИЗАЦІЯ ТЕСТУВАННЯ САЙТІВ ЩОДО НЕДІЙСНИХ ПОСИЛАНЬ

УДК 004.457

БИКОВА Ольга Дмитрівна
магістр, Чорноморський національний університет імені Петра Могили, м. Миколаїв, Україна.
Наукові інтереси: Web-додатки, Web-розробка, UML-діаграми.
e-mail: odbykova@gmail.com

ФІСУН Микола Тихонович
доктор технічних наук, професор, завідувач кафедри інженерії програмного забезпечення,
Чорноморський національний університет імені Петра Могили, м. Миколаїв, Україна.
Наукові інтереси: інтелектуальні інформаційні системи та CASE-засоби їх створення,
системи автоматизованого проектування, бази даних та бази знань.
e-mail: mykola.fisun@gmail.com

ДАВИДЕНКО Євген Олександрович
андидат технічних наук, доцент кафедри інженерії програмного забезпечення,
Чорноморський національний університет імені Петра Могили, м. Миколаїв, Україна.
Наукові інтереси: програмні продукти та технології Microsoft, бази даних та бази знань, системи
автоматизованого проектування, Web-технології, системи підтримки прийняття рішень, системний аналіз.

У зв'язку з тим, що в сучасному світі інформаційних технологій на перший план вийшла мережа Internet, то більшість компанії розробляють web-додатки на різноманітні теми. У команді проекту завжди є працівники, які відповідають за якість IT-продукту та його підтримку вже після виходу у мережу Internet.

Одними з важливих частин на web-сайтах є посилання (як внутрішні, так і зовнішні) та метадані, що передають інформацію пошуковим роботам. У той час як легко перевірити вручну невеликі сайти, часто буває так, що web-додатки занадто великі, щоб пройти сторінку за сторінкою, та перевірити все. На щастя, для web-розробників і фахівців [1-3] з тестування є інструменти та програми, які допомагають визначити непрацюючі посилання на сайті та перевірити наявність метаданих.

За останні роки запропоновано безліч різноманітних програм, інструментів та Add On розширень для браузерів Chrome, IE та Mozilla Firefox [4, 5]. Більшість з них діляться на такі категорії, як перевірка окремо посилань, та перевірка метаданих. Такі інструменти як Screaming Frog або Netpeak Spider занадто вартісні. Велика проблема полягає в тому, що на web-сайтах та додатках стоїть авторизація, яка не дозволяє пройти до інших сторінок, спочатку не ввівши логін та пароль, а отже, перевірка стає неможливою для вище представлених програм. Тому створення web-додатків для пошуку так званих битих посилань є *актуальним* питанням.

Метою даної роботи є удосконалення підходів щодо пошуку непрацюючих посилань та виводу метаданих шляхом створення web-додатку для підвищення продуктивності та скорочення часу на тестування.

Непрацюючі посилання – це область, що часто «страждає» в результаті внутрішнього і зовнішнього зв'язування зсередини web-сторінок.

У той час як легко перевірити вручну, часто буває так, що сайти просто занадто великі, щоб пройти сторінку за сторінкою, та перевірити. Для web-

розробників і фахівців з маркетингу є інструменти, які допоможуть визначити непрацюючі посилання на сайті.

Користувачі можуть розчаруватись у web-сайті через помилки та проблеми при внутрішніх переходах. Але також і пошукові роботи можуть не бачити багатьох сторінок та деталей, а отже скорочується можливість зайняти топове місце у пошуку. Якщо пошукові системи не зможуть сканувати та індексувати сайт легко, вони будуть менш налаштовані відповідним чином ранжувати його.

Під час досліджень проаналізовано основні існуючі інструменти для перевірки посилань, їх переваги та недоліки (табл. 1):

- Darcy SEO Checke;
- Netpeak Spider;
- Check My Links;
- ScreamingFrog.

По-перше, чого хотілося б всім користувачам це отримати просту і легку для розуміння систему. Краще всього, якщо віна буде безкоштовна, тому що не багато людей захоче брати дорогий продукт для такої задачі (більшість все ж буде тестувати вручну).

Таблиця 1

Переваги та недоліки програм пошуку посилань та метаданих

Назва програми	Переваги	Недоліки
Darcy SEO Checke	Є можливість переглянути історію тестування.	Немає функції генерування звіту.
Netpeak Spider	Показує кількість знайдених помилок для оптимізації сайту.	12 місяців – $117 Ціна!
Check My Links	Підсвічує биті посилання на сторінці.	Немає функції перевірки метаданих. Немає функції генерування репорту.
ScreamingFrog	Генерує Sitemap.xml сайту.	Немає можливості пройти авторизовані сайти. 12 місяців – $185 Ціна!
RainBow	Є вікно авторизації, яке дозволяє пройти далі по сайту та вивести всі дані. Безкоштовна та з відкритим кодом.	Немає функції пошуку саме помилок та підрахування їх кількості.

Також можна переглянути порівняльну діаграму (рис. 1) проходження програмами по сторінкам сайту за деякий проміжок часу. Дана діаграма створена, щоб показати можливості розробленого програмного забезпечення та порівняти з вже існуючими додатками. Оцінку швидкості сканування сторінок було отримано наступною перевіркою: кожен програмний додаток отримував посилання на сайт, який перевіряється; запускалася програма пошуку та одночасно секундомір. Потім порівнювалися результати і заносилися у таблицю, на даних якої можна побудувати відповідну діаграму.

Візуалізація даних [6] – це подання даних у вигляді, який забезпечує найбільш ефективну роботу користувача по їх вивченню. Візуалізація даних знаходить широке застосування в наукових і статистичних дослідженнях (зокрема, в прогнозуванні, інтелектуальному аналізі даних, бізнес-аналізі), в педагогічному дизайні для навчання і тестування, в аналітичних оглядах. Візуалізація даних пов'язана з виведенням інформації, відображенням наукових даних, розвідувальним аналізом даних і статистичної графікою.

Візуальна інформація краще сприймається і дозволяє швидко і ефективно донести до користувача власні думки та ідеї. Фізіологічно, сприйняття візуальної інформації є основною для людини.

Цілі візуалізації – це реалізація основної ідеї інформації, це те, заради чого потрібно показати вибрані дані, якого ефекту потрібно досягнути – виявлення взаємозв'язків в інформації, відображення розподілу даних, композиції або порівняння даних.

Композиція даних – об'єднання даних з метою аналізу загальної картини в цілому, порівняння компонентів, що складають відсоток від якогось цілого. Ключовими фразами для композиції є «склало х%», «частка», «відсоток від цілого».

Порівняння даних – об'єднання даних, з метою порівняння деяких показників, виявлення того, як об'єкти співвідносяться один з одним. Також це порівняння компонентів, що змінюються з часом.

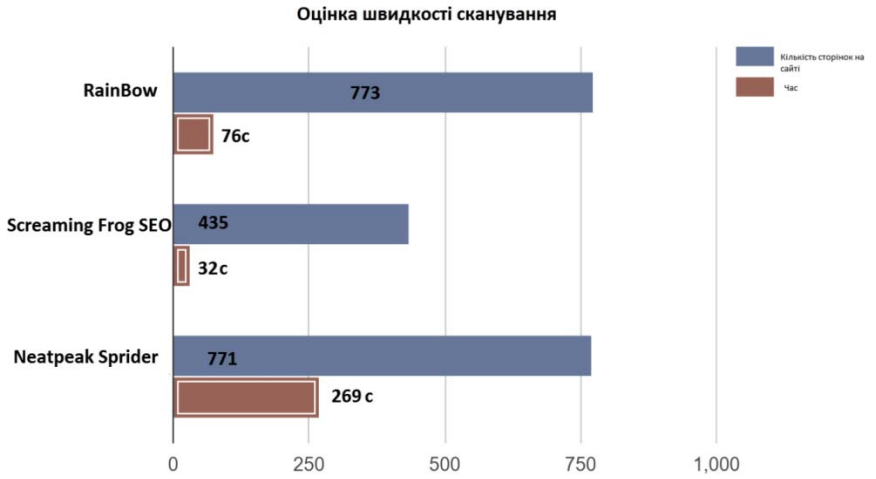

Рис. 1. Порівняльна діаграма

Після визначення мети візуалізації визначаємо типи даних. За своєю структурою вони можуть бути дуже різнорідними, але в самому простому випадку виділяють безперервні числові і тимчасові дані, дискретні дані, географічні та логічні дані. Безперервні числові дані містять в собі інформацію залежності однієї числової величини від іншої, наприклад графіки функцій, такий як *y=2*x*. Безперервні тимчасові містять в собі дані про події, що відбуваються на якомусь проміжку часу, як графік температури, вимірюваної кожен день. Дискретні дані можуть містити в собі залежності категорійних величин, наприклад графік кількості продажів товарів у різних магазинах. Географічні дані містять в собі різну інформацію, пов'язану з місцем розташування, геологією та іншими географічними показниками, яскравий приклад – це звичайна географічна карта. Логічні дані показують логічне розташування компонентів відносно один одного, наприклад генеалогічне древо сім'ї.

Залежно від мети і даних можна вибрати найбільш підходящий їм графік. Найкраще уникати різноманітності заради різноманітності і вибирати за принципом «чим простіше, тим краще». Тільки для специфічних даних використовувати специфічні типи діаграм, в інших же випадках добре підійдуть найпоширеніші графіки:

- лінійний (line);
- з областями (area);
- колонки і гістограми (bar);
- кругова діаграма (pie, doughnut);
- полярний графік (radar);
- точковий графік (scatter, bubble);
- карти (map);
- дерева (tree, mental map, tree map);
- тимчасові діаграми (time line, gantt, waterfall).

Пошук у ширину – алгоритм пошуку на графі: якщо задано граф $G = (V, E)$ та початкову вершину s, алгоритм пошуку в ширину систематично обходить всі досяжні із s вершини. На першому кроці вершина s позначається, як пройдена, а в список додаються всі вершини, досяжні з s без відвідування проміжних вершин. На кожному наступному кроці всі поточні вершини списку відмічаються, як пройдені, а новий список формується із вершин, котрі є ще не пройденими сусідами поточних вершин списку. Для реалізації списку вершин найчастіше використовується черга. Виконання алгоритму продовжується до досягнення шуканої вершини або до того часу, коли на певному кроці в список не включається жодна вершина. Другий випадок означає, що всі вершини, доступні з початкової, уже відмічені, як пройдені, а шлях до цільової вершини не знайдений.

Алгоритм, що використовувався при створенні програмного забезпечення має назву пошуку в ширину, оскільки «фронт» пошуку (між пройденими та не пройденими вершинами) одноманітно розширюється

вздовж всієї своєї ширини. Тобто, алгоритм проходить всі вершини на відстані *k* перед тим як пройти вершини на відстані *k+1*.

Кроки алгоритму:

1. почати з довільної вершини *v*. Виконати *v*=1. Включити вершину *v* у чергу;
2. розглянути вершину, яка перебуває на початку черги; нехай це буде вершина *x*. Якщо для всіх вершин, суміжних із вершиною x, уже визначено номери, то перейти до кроку 4, інакше – до кроку 3;
3. нехай {*x,y*} – ребро, у якому номер *y* не визначено. Позначити це ребро потовщеною суцільною лінією, визначити *y* як черговий номер, включити вершину *y* у чергу й перейти до кроку 2;
4. виключити вершину *x* зі черги. Якщо черга порожня, то зупинитись, інакше – перейти до кроку 2.

Відстань Левенштейна [7] (також функція Левенштейна, алгоритм Левенштейна або відстань редагування) у теорії інформації і комп'ютерній лінгвістиці – міра відмінності двох послідовностей символів (рядків). Обчислюється як мінімальна кількість операцій вставки, видалення і заміни, необхідних для перетворення одної послідовності в іншу.

На практиці дистанція Левенштейна використовується для визначення подібності послідовностей символів, наприклад для корекції орфографії або для пошуку дублікатів.

Для розрахунку відстані Левенштейна найчастіше використовують простий алгоритм, в якому використовується матриця розміром *(n+1)*(m+1)*, де *n* і *m* – довжини порівнюваних рядків. Окрім цього, значення операцій вилучення, заміни та вставки вважається однаковою. Для конструювання матриці використовують таке рекурсивне рівняння 1:

$$D_{0,0} = 0$$
$$D_{i,j} = min \begin{cases} D_{i-1,j-1} + 0\ (equals, nochange) \\ D_{i-1,j-1} + 1\ (replace) \\ D_{i-1,j} + 1\ (insert) \\ D_{i,j-1} + 1\ (delete) \end{cases} \quad (1),$$

де *D* – таблиця з деякою кількістю рядків і стовпців, *i* та *j* – індексування позицій у рядках і стовпцях.

Побудова матриці дистанцій схоже на прокладання маршруту через лабіринт: починаємо з лівого верхнього кута матриці-карти і повинні потрапити в правий нижній. Частина матриці можна заповнити без обчислень: стовпець і рядок з нульовими індексами заповнюються числами по порядку, починаючи з нуля. Це просто пояснити тим, що для того, щоб з порожнього рядка отримати якийсь рядок *T* (довжиною *k*), потрібно рівно k вставок – по одній на кожен символ. Аналогічно і в зворотний бік: для того, щоб з рядка S довжиною l отримати порожній рядок, потрібно рівно l вилучень. Таким чином, числа в нульовий рядку і колонці не залежить від вмісту порівнюваних рядків.

Редакційний припис – послідовність дій, необхідних для отримання з першого рядка другий найкоротшим шляхом. Позначимо дії наступним чином: D (англ. Delete) – видалити, I (англ. Insert) – вставити, R (replace) – замінити, M (match) – збіг.

Для рядків ABC і ABF редакційний припис буде виглядати як у фомулі 2:

$$\begin{Bmatrix} M\ M\ R \\ A\ B\ C \\ A\ B\ F \end{Bmatrix} \quad (2)$$

Для відстані Левенштейна існують такі верхня і нижня межі:

- дистанція Левенштейна не є меншою за різницю довжини рядків, що порівнюються;
- вона не є більшою довжини найдовшого рядка;
- вона дорівнює 0 тоді і тільки тоді, коли рядки однакові (однакові символи на однакових позиціях).

Між відстанню Левенштайна та відстанню Гемінга [8] існують такі взаємозв'язки:

- для рядків однакової довжини відстань Левенштайна рівна відстані Гемінга, оскільки відстань Гемінга користується лише операцією заміни одного символу на інший і не дозволяє вставки та вилучення символів;

– якщо рядки різної довжини, то верхньою межею є відстань Гемінга плюс різниця довжини рядків.

На наступному етапі дослідження розроблено складові проекту web-системи – три UML-діаграми для web-додатку [9]. Діаграма використання [10] (діаграма прецедентів) – це найбільш загальне уявлення функціонального призначення системи.

На діаграмі використання (рис. 2) застосовано два типи основних сутностей: варіанти використання і діючі особи, між якими встановлюються наступні типи відносин:

– асоціація між діючою особою і варіантом використання;
– узагальнення між діючими особами;
– узагальнення між варіантами використання;
– залежності (різних типів) між варіантами використання.

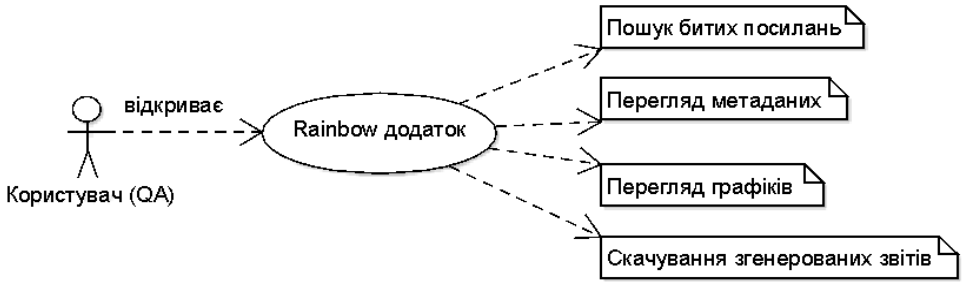

Рис. 2. Діаграма використання розроблюваного додатку

Діаграма діяльності [10] (діаграма діяльності) – спосіб опису поведінки на основі вказівки потоків управління і потоків даних.

Діаграма діяльності (рис. 3) – ще один спосіб опису поведінки, який візуально нагадує блок-схему алгоритму. Однак за рахунок модернізованих позначень, узгоджених з об'єктно-орієнтованим підходом, а головне, за рахунок нової семантичної складової (вільна інтерпретація мереж Петрі), діаграма діяльності UML є потужним засобом для опису поведінки системи.

На діаграмі діяльності застосовано один основний тип сутностей – дія, і один тип відносин – переходи (передачі управління і даних). Також використовуються такі конструкції як розгалудження, злиття, з'єднання.

Для розроблюваного додатку створена така діаграма, яка представляє усі дії, які користувач може зробити, та деякі розгалудження, при яких неможливо пройти далі.

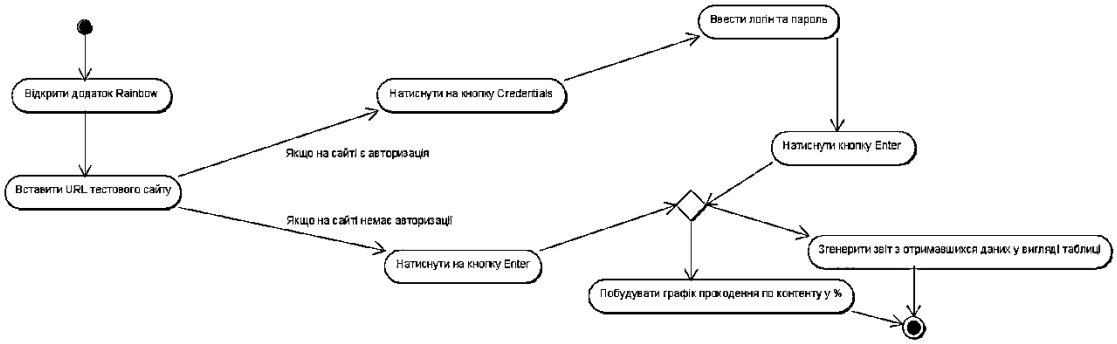

Рис. 3. Діаграма діяльності розроблюваного додатку

Діаграма автомата [10] (схема кінцевого автомата) – це один із способів детального опису поведінки в UML на основі явного виділення станів і опису переходів між станами.

По суті, діаграми автомата, як це випливає з назви, є граф переходів станів, навантажений додатковими деталями. На діаграмі автомата застосовано один основний тип сутностей – стану, і один тип відносин – переходи, і для тих і для інших визначено безліч різновидів, спеціальних випадків і додаткових позначень. Дана діаграма теж має свою роль у представлені додатку, за її допомогою можна легко додати відображення процесу авторизації (рис. 4).

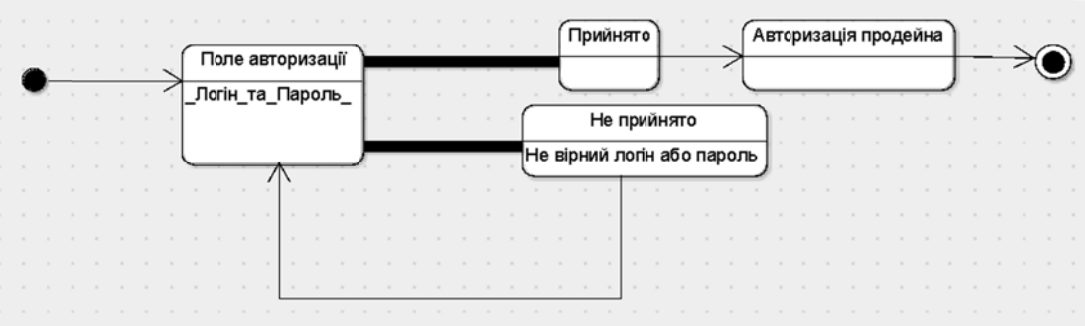

Рис. 4. Діаграма функціоналу авторизації на сайті

Для знаходження усіх посилань та метаданих на сайті потрібно продивитися усі сторінки. Саме для цього і використовується алгоритм пошуку у ширину. Алгоритм дозволяє визначити першу головну сторінку і від неї йти до посилань в меню, а вже після цього на всі підсторінки сайту.

На рисунку 5 зображено вигляд результатів пошуку посилань на web-сайті http://samsung.com.

Rainbow Перевірка Звіт .xls Візуальний графік

Результати для http://www.samsung.com

№	Посилання	Статус	Content Type	Title	Description	Зміна посилання
1	http://www.samsung.com/ua/promotions/wm/addwash/	200	text/html	AddWash Samsung | Ukraine	Новые стиральные машины Samsung AddWash позволяют добавлять вещи или кондиционер для белья прямо во время стрики!	
2	http://www.samsung.com/us/aboutsamsungir/newsMain.do	200	text/html; charset=UTF-8	About Samsung - Samsung	Samsung Group - About Aamsung	
3	http://www.samsung.com/ua/official_partners/#corperation	200	text/html	Official_partners		
4	https://press.samsung.ua/	200	text/html;charset=utf-8	Пресс-центр Samsung Electronics Украина	Samsung Electronics Co., Ltd. — мировой лидер в области производства полупроводников, телекоммуникационного и цифрового медиаоборудования, а также в сфере технологий цифровой конвергенции.	
5	http://www.samsung.com/ua/brandshops/	200	text/html	Title		
6	http://livechat.support.samsung.com/Customer_new/UA	200	text/html; charset=utf-8	Категорія | Samsung Live Chat		
7	http://www.samsung.com/ua/support/erms1	200	text/html; charset=UTF-8	Запит технічному спеціалісту | Samsung UA	Email Technical Support	
8	https://www.facebook.com/SamsungMobileUkraine/	200	text/html	Samsung Mobile Ukraine	Samsung Mobile Ukraine. 235 757 вподобань · 2 291 особа обговорює це. Ласкаво просимо до офіційної сторінки Samsung Mobile Ukraine! Приєднуйтесь до...	
9	https://twitter.com/samsungukrain	404	text/html;charset=utf-8	Twitter / ?		https://twitter.com/samsungukraine
10	https://www.instagram.com/samsungua/	200	text/html	Samsung Ukraine (@samsungua) • Instagram photos and videos		

Рис. 5. Результат пошуку посилань на сайті http://samsung.com

Алгоритм Левенштейна призначений для редагування орфографії слів (у розроблюваному додатку – це редагування посилань).

Після використання алгоритму пошуку у ширину та отримання всіх посилань на сайті, їх статус-коди, алгоритм Левенштейна потрібен для посилань, які мають статус-коди «404» – повідомлення про помилку в списку кодів стандартних відповідей HTTP-сервера, яке означає, що сервер не може знайти того, чого вимагає запит від клієнта. Розроблено окрему колонку в таблиці результатів, в яку можна завантажити Excel-файл з переліком правильних посилань та сторінок сайту. І роботою алгоритму є перевірка та зіставлення списку правильних посилань з непрацюючими. На рисунку 5 показано колонку, в якій реалізовано алгоритм Левенштейна.

Так як було зазначено раніше, за допомогою візуалізації даних, людина краще розуміє та сприймає подану інформацію. Саме тому, візуалізація результатів пошуку непрацюючих посилань та знайдених метаданих на сайті є корисним моментом у розробці web-системи.

На рисунку 6 представлено візуалізацію результатів пошуку та відображення метаданих у вигляді кругових діаграм.

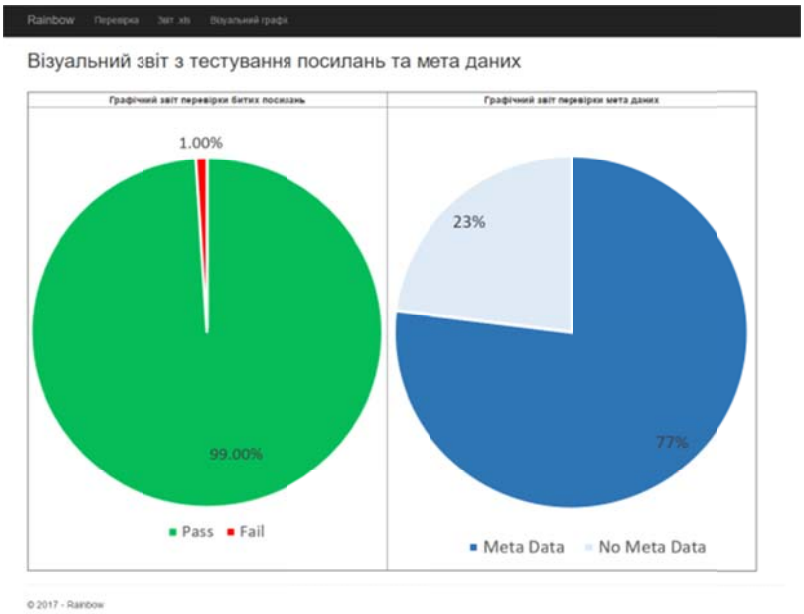

Рис. 6. Кругова діаграма відображення результатів пошуку та метаданих

В результаті розробки програмного забезпечення удосконалено підходи щодо пошуку непрацюючих посилань та виведення мета даних. Цього було досягнуто шляхом створення web-системи, що підвищує продуктивність користувача та скорочує час на тестування.

В роботі розв'язані наступні поставлені задачі:

- проведено аналіз підходів до автоматизації тестування програмного забезпечення web-сайтів;
- досліджено вже існуючі інструменти та програми, що досягають поставленої мети;
- проведено аналіз варіантів пошуку битих посилань на сторінках web-сайтів;
- проведено аналіз можливості побудови графіків та створення звітів на основі отриманих даних;
- розроблено власний додаток для пошуку битих посилань та знаходження метаданих сторінок сайту.

Результатом виконання роботи є робочий web-додаток, що дозволить швидко та зручно шукати посилання на сайті, виправляти посилання зі статус-кодами 404, та відображати метадані, а також будувати кругові діаграми для більшого сприйняття результатів. І останньою функцією є завантаження результатів у Excel-файл для збереження на робочому комп'ютері.

Розроблена програма має особливе значення у сфері IT, так як дозволяє скоротити час тестування, а

отже є можливість зробити й інші перевірки. Розроблена програма безкоштовна та з відкритим кодом, і такі функції, як генерування звіту, побудова графіків та проходження авторизації на сайті буде представлена для всіх користувачів.

ЛИТЕРАТУРА

1. Testirovanie API [Elektronnyj resurs] – Rezhim dostupa: http://software-testing.ru/forum/index.php?/topic/28679-testirovanie-web-api/
2. Yakobson A., Buch G., Rambo Dzh. Unificirovannyj process razrabotki programmnogo obespecheniya. – SPb.: Piter, 2002. – 496 s.
3. Gluhih M. I., Icykson V. M. Programmnaya inzheneriya. Obespechenie kachestva programmnyh sredstv metodami staticheskogo analiza. Uchebnoe posobie. SPb: Izd-vo Politekhn. un-ta. 2011, 150 s.
4. Uchebnik po SEO [Elektronij resurs] – Rezhim dostupu: http://ksw.net.ua/interesnoe/interesnye-statii/187-bitye-ssylki
5. Meht'yu D. HTML5. Razrabotka veb-prilozhenij. M.: Rid Grupp, 2012 – 320 s.
6. Lekciya №6 Metodi vizualizacii interfejsu koristuvacha pri proektuvanni. Ob'єktno-orієntovanij interfejs koristuvacha (OOIK) [Elektronij resurs] – Rezhim dostupu: https://research.facebook.com/publications/deepface-closing-the-gap-to-human-level-performance-in-face-verification/
7. Vidstan' Levenshtejna [Elektronij resurs] – Rezhim dostupu: https://uk.wikipedia.org/wiki/Відстань_Левенштейна
8. Vidstan' Gemminga [Elektronij resurs] – Rezhim dostupu: https://uk.wikipedia.org/wiki/Відстань_Геммінга
9. Rational Unified Process. Best Practices for Software Development Teams, A Rational Software whitepaper from IBM, No. 11/01, p. 21.
10. Hassan Goma. UML-proektirovanie sistem real'nogo vremeni parallel'nyh i raspredelennyh prilozhenij – DMK Press: Piter, 2011. – 704s.

Рецензент: *д.т.н., проф. Сис В. Б.*
Херсонський національний технічний університет