УДК 004.624

С.І. Шаповалова, Д.М. Хохлов

ЕКСТРАКЦІЯ СТРУКТУРОВАНИХ ДАНИХ З WEB-РЕСУРСІВ

Анотація: В статті запропоновано опис і обґрунтування програмного інструментарію на основі асинхронних можливостей платформи .NET для екстракції даних з Web-ресурсів в форматі HTML. Інструментарій може використовуватися в корпоративних системах агрегування даних.

Ключові слова: Web-системи, екстракція даних, асинхронні запити, MongoDb.

Введення

У сучасному світі все більше аналітичних центрів використовують інформацію з мережі Інтернет у різних галузях діяльності. Зазвичай, необхідна інформація знаходиться у неструктурованому вигляді на декількох Web-ресурсах. Для її отримання доцільно використовувати спеціальний програмний інструментарій, який аналізує сторінку і зберігає структуровані дані в базі даних для подальшого їх використання. Проблема полягає в тому, що програмне забезпечення екстракції даних необхідно розробляти кожен раз для кожного ресурсу. Тому створення універсального програмного інструментарію для отримання структурованих даних з Web-ресурсів є актуальною задачею з вагомим практичним застосуванням.

Аналіз останніх досліджень

В останні роки з'явилось комерційне програмне забезпечення для часткового розв'язання поставленої задачі. Найбільш розповсюдженим і апробованим є WebHarvy [1], який може автоматично отримувати дані (текст, посилання та зображення) з Web-сторінок і зберігати зміст в обмеженій кількості форматів. Більш досконалим є програмне забезпечення WebSundew [2] призначене для екстракції даних з цільових Web-сайтів і їх зберігання в заданому форматі. Цей інструмент розрахований для застосування до окремого сайту. Його використовують практично у всіх сферах бізнесу. Однак існуючі програмні рішення екстракції та агрегування даних мають наступні загальні недоліки:

- необхідність інсталяції на сервері, що вимагає високої швидкості передачі даних в мережі Інтернет та додаткових комп'ютерних потужностей;
- складність у конфігурації;
- необхідність достатньо великого часу для вивчення та апробування програмного засобу.

На теперішній час ведуться дослідження для позбавлення програмного забезпечення зазначених недоліків. Так, в даній роботі запропоновано універсальний інструментарій екстракції даних з Web-ресурсів ScraperPlus.

 ISSN 1560-8956

Мета статті

Метою статті є опис і обґрунтування програмного інструментарію для екстракції даних з Web-ресурсів в форматі HTML, його переваг.

Задачі екстракції Web-контенту:

1. Визначення структур представлення Web-сторінки, яка аналізується.
2. Завантаження сторінки для парсингу.
3. Парсинг та обробка інформації на сторінці.
4. Зберігання результату в базі даних.
5. Відкриття доступу до API, через яке користувач завантажить результати роботи.

Першу задачу користувач вирішує за допомогою Web-інтерфейсу ScraperPlus, виконуючи наступні дії:

1. Введення назви завдання обробки сайту.
2. Визначення структури даних, яку необхідно отримати, та задання селекторів html контейнерів, з яких будуть діставатись дані.
3. Введення адресу сторінки, з якої необхідно починати обхід сайту.

Для розв’язання першої задачі були використані наступні технології: asp.net mvc4 - технологія для Web розробки; JavaScript бібліотека jquery для спрощення управління DOM (Document Object Model) елементами; JavaScript бібліотека KnockoutJs для створення інтерактивного інтерфейсу.

Для завантаження сторінки можна використати декілька підходів:

- синхронне послідовне отримання сторінок;
- розбиття процесу отримання сторінок на декілька потоків;
- асинхронне отримання сторінок.

Для ScraperPlus було обрано асинхронний метод отримання сторінок з використанням нових можливостей C# 5.0 і класу HttpClient. Цей метод дає змогу отримувати декілька сторінок паралельно, що збільшує ефективність роботи системи в цілому.

Для визначення оптимального методу парсингу HTML-документів в даній роботі було проведено дослідження, результати якого зведено в табл. 1.

Для реалізації парсингу в інструментарії ScraperPlus використана .Net бібліотека CsQuery, яка працює на основі методу аналізу DOM-дерева. Тільки цей метод надає можливість автоматизовано генерувати шлях до позначеного користувачем елемента Web-сторінки.

Зберігання результатів в базу даних здійснюється за допомогою нереляційної бази даних MongoDb [3], завдяки можливості динамічно зберігати дані різної структури та великій швидкості запису у порівнянні з іншими відомими базами даних.

Таблиця 1

Базові методі парсингу

Назва методу	Переваги	Недоліки
Аналіз DOM дерева (Document Object Model)	• можливість отримання даних будь-якого типу і будь-якого рівня складності; • можливість отримання значення на основі прописаного шляху.	• складність і неоднозначність DOM-шляху.
Парсинг рядків	• більша ефективність в порівнянні з аналізом DOM-дерева та XPath у простих шаблонних випадках.	• необхідність задання виключно жорсткого паттерну для кожного елементу пошуку.
Використання регулярних виразів	• використання патерну у вигляді регулярного виразу для екстракції даних, які мають строгий формат (наприклад, електронні адреси, телефони).	• складність отримання доступу до не строго форматованих даних.
XML парсинг (на базі мови запитів XPath)	• можливість вибору елемента на основі мові запитів XPath.	• великі затрати часу для перетворення формату HTML в XML; • низька валідність отриманого HTML-документа.

ISSN 1560-8956

Відкриття доступу до API реалізовано на основі можливостей ASP.NET MVC Web API. Дані можуть бути отримані у форматі json або xml, в залежності від запитуваного типу контенту.

Етапи екстракції контенту

На рисунку 2 приведені задачі, які виконує програміст для екстракції структурованих даних з Web-ресурсу без використання інструментарію ScraperPlus (рис. 2а), і задачі – з використанням (рис. 2б).

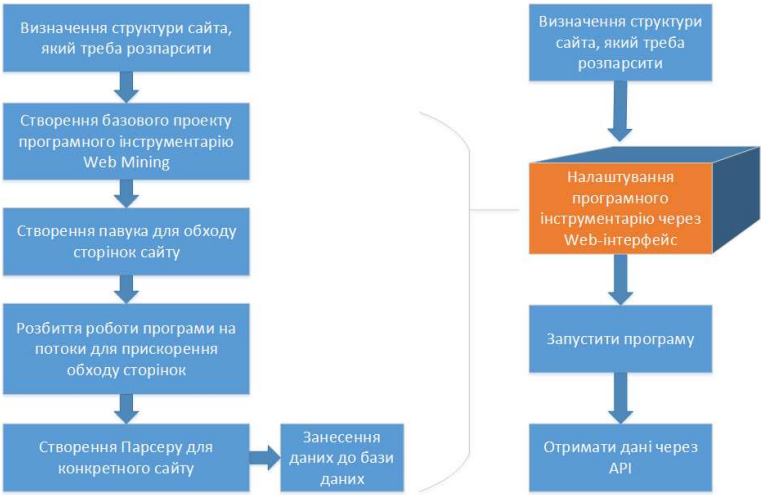

Рис. 1 – Етапи екстракції даних: а) без використання інструментарію ScraperPlus; б) з використанням

Екстракція даних з HTML документа складається з декількох етапів. Перший етап - визначення структури сайту, тобто аналіз HTML-документа, інформації, яка там знаходиться, та визначення структури даних, яка потрібна для вирішення задачі. Після цього створюється базовий проект програми. Створюється базова архітектура проекта, підключаються всі потрібні залежності. Наступним етапом є розробка павука для обходу сторінок. Для більш швидкого обходу необхідно розбити загрузку сторінок на потоки. Після завантаження хоча б однієї сторінки, модуль парсингу починає обробку. На основі інформації, яка була занесена в програму на етапі аналізу, парсер отримує дані зі сторінки в необхідному форматі. Після закінчення роботи павука і парсера дані записуються у базу даних (рис. 1а).

Програмний інструментарій ScraperPlus спрощує цю схему. Першим етапом є аналіз структури сайту, який є подібним відповідному етапу традиційної екстракції даних. Наступним етапом є настройка програми через Web-інтерфейс. В даному модулі об’єднуються всі етапи традиційної екстракції, окрім першого. Після цього користувач запускає

ScraperPlus, який після завершення своєї роботи надає інформацію через API.

Архітектура інструментарію ScraperPlus

ScraperPlus складається з 4-х базових модулів для розв’язання відповідних задач (рис. 2).

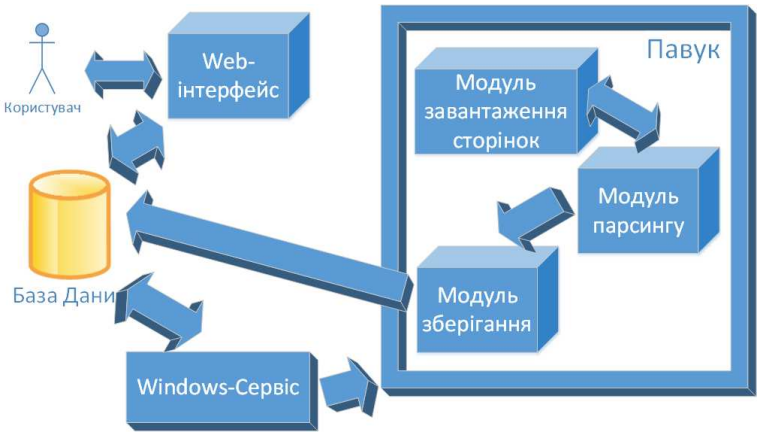

Рис. 2 – Архітектура ScraperPlus

Web-інтерфейс використовує бібліотеку KnonkoutJs та Ajax технології для того, щоб користувач працював з інтерфейсом без перерв на перезавантаження сторінок при надходженні нового запиту на роботу. Для роботи з DOM використано JavaScript бібліотеку jQuery [4]. Задачами даного модуля являється:

- формування завдання екстракції даних;
- розміщення завдання до черги у базу даних.

За допомогою даного модулю користувач має можливість створити нове завдання на парсинг, перезапустити вже завершене та отримати результати парсингу.

Windows-сервіс, представлений у вигляді Windows-застосунку, працює за наступною схемою:

1. Чекає завдання з бази даних.
2. Запускає павука для екстракції даних.
3. Змінює статус завдання на “Виконано”.
4. Надає користувачу можливість переглянути дані через API.

Павук складається з трьох модулів:

- модуль загрузки сторінок;

 ISSN 1560-8956

- модуль парсингу;
- модуль зберігання даних.

Модуль завантаження сторінок займається створенням запитів до інших Web-ресурсів та записом до черги для парсингу. Він реалізований за допомогою нового класу у C# 5.0 HttpClient, який дає змогу робити асинхронні запити, тим самим пришвидшуючи роботу в кілька разів в залежності від комп’ютера [5]. Спочатку цей модуль отримує посилання з черги, яке потрібно загрузити, і створює асинхронну задачу [6]. Ці дії мають повторюватись N разів для збільшення швидкості завантаження сторінок. Проведені дослідження показали, що доцільно використовувати N = 20, оскільки зменшення призводить до пониження швидкості завантаження, а збільшення - до перенавантаження серверу, на якому знаходиться Web-ресурс. Після отримання сторінки вона розміщується в черзі для парсингу. Модуль завантаження працює в окремому потоці, щоб не заважати іншим.

Модуль парсингу працює в основному потоці. Він використовує бібліотеку CsQuery, за допомогою якої виконуються запити до HTML сторінки, використовуючи синтаксис jQuery. Це дає змогу коректно отримувати дані з різних елементів DOM. Модуль парсингу, за допомогою селекторів, які налаштував користувач, перевіряє, до якого типу відноситься дана Web-сторінка: містить контент або посилання. В першому випадку отримуються дані і записуються у об’єкт, який має тип Dictionary. Якщо сторінка містить посилання, то вони заносяться до черги на завантаження, яка з частотою 50мс перевіряється модулем завантаження. Коли дані розпарсені, відповідні параметри передаються для формування події, яка сигналізує про нові дані. Ця подія використовується модулем зберігання даних.

Модуль зберігання працює одному потоці з парсером. Він підписується на подію отримання даних. Додає розпарсені дані у список для збереження. Кожні M записів зберігає у базу даних. Проведені дослідження показали, що доцільно використати M = 100 тому, що зменшення призводить до збільшення кількості запитів до БД, а збільшення - до ризику втрати великої кількості даних під час збою програми. Дані зберігаються в одну колекцію. У розробці були використані:

Шаблон проектування “Репозиторій”, який дає змогу абстрагувати застосунок від технології доступу до даних та полегшити Unit тестування [7];

Паттерн “Singleton”, який гарантує, що клас має тільки один екземпляр, та надає глобальну точку доступу до нього.

База даних реалізована на основі MongoDb. Для роботи з нею використано офіційний C# драйвер від компанії 10gen.

Апробація програмного інструментарію екстракції структурованих даних ScraperPlus

Для визначення ефективності використання ScraperPlus перед професійним розробником були поставлені задачі провести екстракцію да-

них з заданого сайту і з того самого сайту після модифікації його структури. Кожна задача вирішувалась двома способами:

з використанням ScraperPlus;

на базі існуючих технологій, які містять такі самі модулі і засоби завантаження сторінки, парсингу, збереження БД та відкриття доступу до API.

Для виконання обох завдань було створено програмне забезпечення, яке отримує дані з обраних Web-ресурсів в заданому користувачем (програмістом) форматі для подальшої обробки.

Час виконання задачі обома способами представлено у таблиці 2.

Таблиця 2

Час виконання задачі екстракції структурованих даних

Назва дії	Традиційні технології екстракції даних, годин	ScraperPlus, годин
Екстракція даних з сайту	40	0.2
Екстракція даних з того самого сайту зі зміненою структурою	2	0.2

Запропонований програмний інструментарій був використаний в прикладних задачах:

1. Пошук вакансій. В Україні існує більше десяти сайтів, які пропонують розміщення вакансій роботи на їх ресурсі. Для пошуку роботи, користувачеві необхідно реєструватися і провести пошук на кожному сайті окремо. Для апробації програмного інструментарію ScraperPlus було обрано наступні сайти: www.work.ua, www.hh.ru, тому що вони мають різну структуру і html-документів, які потрібно обробити.
2. Агрегування метеоумов для АЕС. Для апробації було обрано 5 сайтів, які надають найбільш актуальну, точну та достовірну інформацію зміни метеоумов: http://rp5.ua/, http://www.gismeteo.ua/, http://www.weather.ua/, http://sinoptik.ua/, http://www.meteonova.ua/. Аналіз цієї інформації дає змогу своєчасно реагувати на критичні зміни метеоумов.

Висновки

1. Проведено аналіз останніх досліджень з розробки програмного забезпечення екстракції даних з Web-ресурсів. Обґрунтовано доцільність розробки програмного інструментарію для розв’язання цієї задачі.
2. Проведено аналіз методів і засобів вирішення задачі екстракції Web-контенту. Обґрунтовано використання ASP.NET MVC для вирішення завдання конфігурування програмного інструментарію, методу

аналізу DOM дерева для завдання парсингу, MongoDB для вирішення завдання збереження даних.

3. Створено програмну реалізацію інструментарію ScraperPlus. Доведено ефективність його використання.

Література

1. WebHarvy // WebHarvy. [2011—2013]. Дата оновлення: 17.01.2013. URL: http://www.webharvy.com (дата звернення: 27. 03.2013).
2. WebSundew // WebSundew. [2005—2013]. Дата оновлення: 24.03.2013. URL: http://www.websundew.com (дата звернення: 27.03.2013).
3. Бэнкер К. MongoDB в действии. / Пер. с англ. Слинкина А. А. - М.: ДМК Пресс, 2012. - 394с. – Москва, 2012 – С. 394.
4. Bibeault B., Katz Y. jQuery in Action, Second Edition // O’Reilly Media, Inc. — 2010. — CT 06901. — P. 488.
5. Рихтер Д. CLR via C#. Программирование на платформе Microsot .NET Framework 4.0 на языке C#. 3-е изд // Джеффри Рихтер — СПб.: Питер, 2012. —928 с.
6. Freeman A. Pro .NET 4 Parallel Programming in C# // Adam Freeman. – 2010. — P. 295.
7. Фримен Э., Сьерра К., Бейтс Б. Паттерны проектирования // Фримен Э., Сьерра К., Бейтс Б. — СПб.: Питер, 2011. — 656 е.: ил.

Отримано 30.11.2012 р.