ATLANTIS PRESS

Advances in Computer Science Research (ACSR), volume 72

IV International Research Conference "Information Technologies in Science, Management, Social Sphere and Medicine" (ITSMSSM 2017)

Parsing of Data on Real Estate Objects from Network Resource*

Vyacheslav Cherkesov, Vitaliy Malikov, Alexey Golubev, Danila Parygin
Department of CAD
Volgograd State Technical University
Volgograd, Russia
chrtyaka@gmail.com, axalter20@gmail.com,
ax.golubev@gmail.com, dparygin@gmail.com

Tatiana Smykovskaya
Department of physics, methods teaching physics and mathematics, information and communication technologies
Volgograd State Socio-Pedagogical University
Volgograd, Russia
smikov_t@mail.ru

***Abstract*—Existing approaches for collecting data from sites on the Internet were considered. A comparative analysis of the solution based on the BeautifulSoup library and the Scrapy framework for parsing the content of network resources was made. Sources of information about real estate objects were analyzed. The method for parsing data on real estate objects was developed based on the results of the conducted studies. In addition, the main problems with the use of parsing technology were identified.**

Keywords*— real estate object; network resource; parsing; data collection; BeautifulSoup; Scrapy

I. INTRODUCTION

The management of a modern city is becoming a high-tech process, provided with technical and information support tools for decision-making [1]. In addition, this process involves an increase in the number of stakeholders interested in the proper assessment of the situation and their opinions. In connection with these necessary be means of conveying information about the situation in the most accessible and open form: via Internet, in a visually clear format in the interactive mode [2].

At the heart of the online system, able to implement all the requirements, lie complex information and analytical solutions [3]. The structure of such systems includes specialized algorithmic tools and implemented with their help software applications [4]. However, their main purpose is to work with the data.

The fact that the data is relevant and how accurately they characterize the situation affects the quality of the conclusions drawn on their basis and subsequent management decisions. The most effective is the analysis of the information from the original source: a resource that generates or aggregates the original data. The optimal speed of data acquisition and processing can be achieved if the resource is available online.

However, the analysis of numerous sources of information on the Internet is a time-consuming process already at the stage of collecting the necessary data. For some sites and tasks, there are standardized by the owners of the resources solutions, distributed in the form API. This is especially useful for tasks big data and widespread for large areas, such as social network [5]. However, conventional solutions do not cover all possible tasks, and for many sites they are not offered at all. In this case, the use of parsers for network resources is one of the most effective solution that allow to automate the process of collecting content in real time.

II. OVERVIEW OF APPROACHES TO THE ANALYSIS AND COLLECTION OF NETWORK RESOURCES CONTENT

Two basic approaches to retrieve the content of network resources can be allocated. The first of them is more focused on the non-professional user, since it does not require special knowledge in the field of programming and IT-technologies. With this approach, the user uses existing systems, plug-ins, which allow to extract content. Such systems include:

- Import.io [6].
- ConvExtra [7].
- Portia [8].

The algorithm for organizing data collection using such systems includes the following steps:

1) Specify the URL of the network resource being examined.

2) Mark the necessary objects for collection (images, headings, links, text blocks, etc.).

3) Start the system (plugin).

Such programs do not require additional skills. In the third stage, they automatically collect, process and provide the data, which can then be stored in different formats (JSON, CSV, etc.).

The main drawback of this approach is that there is no way to automate all the steps of the algorithm described above. This constraint limits the applicability of this approach to scalable projects that require several cycles of data collection.

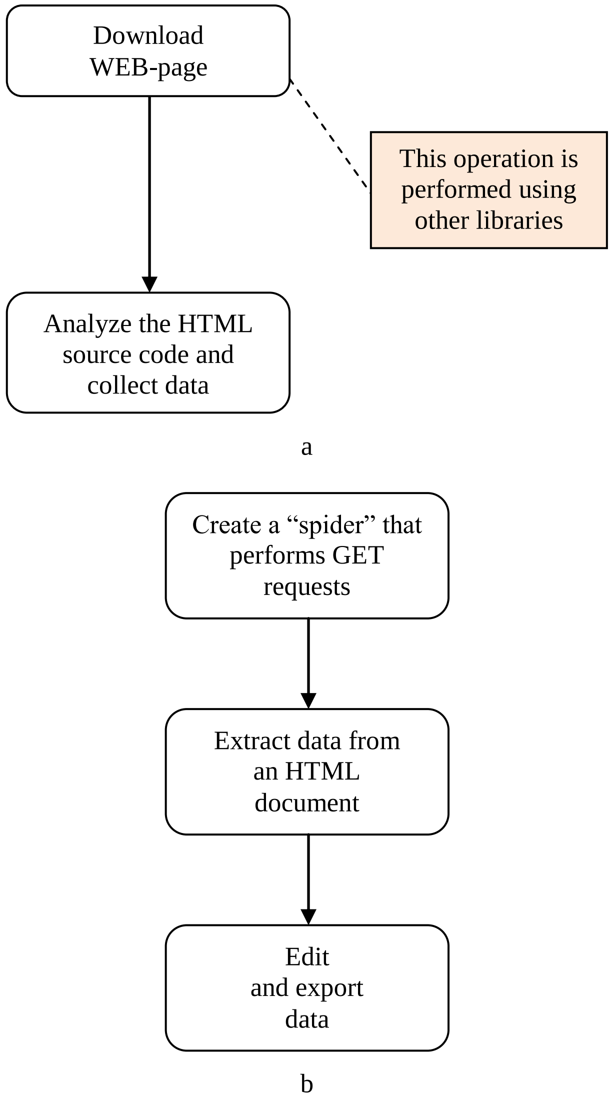
The second approach is to write your own parser for network resources. For its implementation there are also auxiliary software tools – specialized frameworks and libraries, such as Scrapy [9], Beautifulsoup [10] or Grab [11].

The reported study was funded by RFBR according to the research projects No. 16-07- 00388_a, No. 16-07- 00353_a, No. 17-37-50033_mol_nr.

This approach allows to automate the process of parsing the network resources, aggregate content in real time and constant updates. However, be aware that there are some limitations and difficulties imposed directly by the network resource: limit the number of requests, the function of reCAPTCHA and other methods of protection. In General, the second approach is able to provide the above expansion needs and automate the process of parsing of sites.

III. COMPARATIVE ANALYSIS OF TECHNOLOGIES FOR BUILDING SELF-WRITTEN SOFTWARE SOLUTIONS FOR PROCESSING OPEN DATA SITES

In the conducted study, it was decided in practice to explore the possibilities of the library BeautifulSoup (Fig. 1, a) and Scrapy framework (Fig. 1, b). Both of these solutions contribute to the realization of a full cycle of data collection of network resources, however, have a number of features of the implementation developed on the basis of their software modules.


Download WEB-page
This operation is performed using other libraries
Analyze the HTML source code and collect data
Create a "spider" that performs GET requests
Extract data from an HTML document
Edit and export data

a

b

Fig. 1. The procedure of configuring for work with web pages data: a. Library BeautifulSoup; b. Framework Scrapy

BeautifulSoup is a library for extracting data from web pages. However, in most cases, using BeautifulSoup should be accompanied by auxiliary software packages such as "urlib2" or "requests" [12]. These libraries provide a download of web pages for later use of BeautifulSoup in the analysis phase of the source HTML code. At the same time, due to its relative simplicity, the BeautifulSoup library is suitable for novice developers.

Scrapy is a framework for scraping web pages. How to work with this framework involves the creation code for the "spider", which prescribes how to process a page or group of pages. The main feature of this framework is that it is built on the asynchronous network library Twisted. Therefore, Scrapy is implemented using asynchronous code that provides parallelism. This approach at times improves the performance of the "spider".

Comparison of the two concepts is presented in the table (Table 1). In General we can say that the relative ease of implementing solutions based on BeautifulSoup have to pay limited functionality. In fact, this library only allows to analyze the downloaded HTML-code and extract information from it. While Scrapy, a complete and powerful framework with many advanced features. For example, Scrapy has its own library "scrapy-proxies", which allows you to send HTTP requests using the random proxy from the list [13].

In the framework of the studies apply one of the studied solutions was offered to the second-year students of the direction "Informatics and computer facilities" of the VSTU. Twelve of the fourteen people chose to implement the approach based on the BeautifulSoup library, which was finally justified, and in the allotted time, when working with different sites, everyone got working prototypes of programs for data collection.

TABLE I. COMPARATIVE ANALYSIS OF BEAUTIFULSOUP AND SCRAPY

Figure	BeautifulSoup	Scrapy
Study	Relatively easy to learn, suitable for beginners	It is necessary to study a large amount of documentation
Community	Practically absent	A lot of projects, plugins, open source code, many discussions in various forums of the developers
Extensibility	Practically absent	You can easily develop your own middleware or pipeline to add custom functions, easy to maintain
Performance	Needed to import additional libraries "multiprocessing" to improve performance	High performance. Web pages are processed in a short period of time, in many cases it is necessary to set the delay on boot to avoid blocking the "spider"

IV. Discussion of Examples of Practical Implementation of Data Collection Technology

The popular resources of the Internet, which contain information in the format of advertisements for real estate being sold or leased out, such as Avito, Yandex.Realty, CIAN, etc. [14-23], were analyzed for investigation the applicability of approaches to collecting open data. In addition, Russia's leading electronic trading platforms with information on auctioned objects such as Sberbank-AST, RTS-Tender and others were considered separately [24-27], and also a federal register of data on objects real estate Rosreestr [28].

The structure of the data that contain in the announcements and lots was developed for each of them after researching the selected sources. The main activities with real estate (buying or selling, renting or hiring) and types of real estate objects were considered.

Full detailing has been done from the received ad types. This process was carried out in order to harmonize the procedure for collecting information from source sites.

The comparison showed that for all sites, different data structures for objects were obtained. Therefore, it was decided for each of the sources to develop a separate module for the collection of data on real estate.

For the development of modules, Python [29] (version 3.6) was chosen. The modules were implemented on the basis of one of the solutions described above for gathering information from sites, the BeautifulSoup library or the Scrapy framework to the artist's choice.

The results of the research and the development of modules have made it possible to form a unified procedure for working with network resources. The resulting procedure for data collection can be summarized as a method of parsing network resources for real estate objects, which consists of the following stages:

1) To define a "heavy" link, by clicking on which you can get links to ads from all regions of the country.

2) Using the resource index, links to advertisements by types of real estate are compiled.

So, if you are interested in apartments, private houses and land that are subject to sale, the result of compiling the URL of these types of real estate for the "Avito" site is presented in the table (Table 2). Thus, the parser receives links for further actions to collect data.

TABLE II. References to announcements by kinds of objects of the real estate in the Volgograd area from a site "Avito"

Object of the real estate	Link
Apartment	https://www.avito.ru/volgogradskaya_oblast/kvartiry/prodam
Houses, dachas, cottages	https://www.avito.ru/volgogradskaya_oblast/doma_dachi_kottedzhi/prodam
Land plot	https://www.avito.ru/volgogradskaya_oblast/zemelnye_uchastki/prodam

3) To determine the number of pages where the links to ads are located.

This is necessary to determine the number of iterations for obtaining links to the pages of objects. The DOM-tree defines a pointer that contains enumerations of page numbers. The last page number is determined.

4) After receiving the number of pages with the list of site ads, the parser starts a page-by-page collection of links to real estate objects.

To do this, the parser determines the title of the declaration in the DOM-tree and in the <a> tag defines the value of the link for the "href" attribute.

In addition, the title itself contains textual information about the object. For example, "3-room apartment, 56 m², 2/4 floor in Volgograd". From this title we can determine that the object of sale is a three-room apartment with an area of 56 square meters, located on the second floor of a four-story building. Therefore this title should also be saved.

Under the heading of the ad, as a rule, is the price of the object. To save it in the DOM-tree, you define the corresponding <div> tag with the following class description: ".js-catalog_after-ads .description .about".

5) After receiving links to all properties, the parser starts collecting the remaining data directly from the pages of the ads.

This data is in the DOM tree in the corresponding <ul> tag with the following class description: ".item-view-block .item-params .item-params-list". The main characteristics of the object are in each HTML element under the <li> tag. They are read depending on the type and stored in temporary variables.

The same pages also contain information about the address of the object. It is located in the DOM tree under the <div> tag with the class description ".item-map.js-item-map .item-map-location". Presented by a number of <span> tags that contain the full address of the object.

6) All found data on the ad page is entered to the dictionary, the fields of which correspond to the basic characteristics of the property.

Further, this data is saved in a JSON file. This action is necessary for the subsequent processing of the received data.

V. Problems of Application of the Developed Solutions

The developed methods were applied to the pool of the investigated network resources and showed its practical applicability. However, when developing and testing the launch of parsers, a number of problems were identified. The main problems that hinder the operation of programs and the implementation of the technology described above in pure form can be grouped as follows:

1) The main problem in the processing of most resources – sites have protection from bot programs, i.e. from programs that simulate the actions of the average user when working with the site.

For many sites [14, 15, 25] it was enough to introduce an additional delay of 5 seconds, and continue the module. However, for some sources [16, 22, 23], this method did not work, and they began blocking access to the site by IP-addresses. To work around this problem, we used a list of proxy servers and their periodic change, as well as a user-agent field for the parsing script. In addition, some sources [16, 18] use the CAPTCHA function to access site data, complex JavaScript logic, or display important information on images. But solutions to overcome this kind of protection (using machine learning to recognize CAPTCHA and image segmentation, using Selenium to perform JavaScript logic on the loaded page) went beyond the scope of the study.

2) In addition, for correct operation of the modules, one had to take into account the absence of some of the processed fields (tags or their attributes) in some declarations.

3) Also, during the development, it was revealed that the search and collection of information on the site from specially structured sections, such as tables, does not always give complete information about the object.

4) Much of the important information is stored in the text of the description, which is written by a person and can not be directly formalized. Thus, its recognition requires the development of additional algorithmic and software solutions.

VI. Conclusion

Testing the developed modules for parsing allowed to identify a number of common problems for data collection from network resources. Nevertheless, the conducted research has shown the effectiveness of using such means of working with content to automate data collection. The technologies considered and the proposed parsing method allow to work systematically with data sources on the network, scale the task, including handling an unlimited amount of additional resources.

However, from the point of view of organizing an integrated procedure for collecting data from multiple sources, a new task arises that can be the subject of another study. Since the structure of data obtained from different parsers no, you need to implement the adaptation of the modules according to a single pattern. Moreover, given the lay time delays of requests to sites, required the development of a mechanism in a multithreaded retrieve data from several modules of the parser. The prospect of accomplishing these and other possible tasks will allow creating conditions for the formation of an extensive thematic database for solving specific problems of urban development.

References

[1] D. S. Parygin, N. P. Sadovnikova, O. A. Shabalina, “Informational and analytical support for the management of a city: monograph”, Volgograd, 2017, 116 p.

[2] D. Parygin, N. Sadovnikova, M. Kalinkina, T. Potapova, A. Finogeev, “Visualization of data about events in the urban environment for the decision support of the city services actions coordination”, SMART 2016, Proc. of the 5th International Conference on System Modeling & Advancement in Research Trends, Moradabad, India, 25–27 November 2016, IEEE, 2016, pp. 283–290

[3] S. Ustugova, D. Parygin, N. Sadovnikova, V. Yadav, I. Prikhodkova, “Geoanalytical system for support of urban processes management tasks”, CIT&DS 2017, Proc. of the Second International Conference on Creativity in Intelligent Technologies & Data Science, Volgograd, Russia, 12–14 September 2017, Springer IPS, 2017, CCIS 754., pp. 430–440.

[4] A. Golubev, I. Chechetkin, D. Parygin, A. Sokolov, M. Shcherbakov, “Geospatial data generation and preprocessing tools for urban computing system development”, Procedia Computer Science, Proc. of the 5th International Young Scientist Conference on Computational Science, YSC 2016, Krakow, Poland, 26–28 October 2016, Elsevier, 2016., vol. 101, pp. 217–226.

[5] D. Donchenko, N. Sadovnikova, D. Parygin, O. Shabalina, “Promoting urban projects through social networks using analysis of users influence in social graph”, Advances in Comptuer Science Research, Proc. of the 2016 Conference on Information Technologies in Science, Management, Social Sphere and Medicine (ITSMSSM), Tomsk, Russia, 23–26 May 2016, Paris : Atlantis Press, 2016, vol. 51, pp. 162–165.

[6] Import.io – extract data from the web. https://www.import.io/ (date of access: 24.10.2017)

[7] ConvExtra. http://convextra.com/ (date of access: 24.10.2017)

[8] Portia – scraping platfrom. https://scrapinghub.com/portia/ (date of access: 24.10.2017)

[9] Scrapy – an open source and collaborative framework for extracting the data you need from websites. https://scrapy.org/ (date of access: 24.10.2017)

[10] Beautiful Soup Documentation. https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (date of access: 02.07.2017)

[11] Grab – python framework for building web scrapers. http://grablib.org/en/latest/ (date of access: 24.10.2017)

[12] Requests: HTTP for Humans. http://docs.python-requests.org/en/master/ (date of access: 01.07.2017)

[13] Random proxy middleware for Scrapy. https://github.com/aivarsk/scrapy-proxies (date of access: 24.10.2017)

[14] The site of free ads "Iz Ruk v Ruki". http://irr.ru/ (date of access: 24.10.2017)

[15] Announcements about sale, purchase, lease of real estate. http://realty.dmir.ru/ (date of access: 24.10.2017)

[16] Purchase, sale and rent of real estate "Mir Kvartir". http://mirkvartir.ru/ (date of access: 24.10.2017)

[17] CIAN – a database of real estate. https://www.cian.ru/ (date of access: 24.10.2017)

[18] Yandex.Realty https://realty.yandex.ru/ (date of access: 24.10.2017)

[19] Buying, selling and leasing of real estate "Kvadroom". https://kvadroom.ru/ (date of access: 24.10.2017)

[20] Sales and rentals – Real estate Mail.Ru. https://realty.mail.ru/ (date of access: 24.10.2017)

[21] Real estate – sale and rent of apartments, houses, rooms, offices, property prices, Domino. http://domino-rf.ru/nedvizimost/ (date of access: 24.10.2017)

[22] Restate.ru – real estate portal of Moscow and St. Petersburg, Leningrad and Moscow regions. http://www.restate.ru/ (date of access: 24.10.2017)

[23] Bulletin board from individuals and companies on Avito. https://www.avito.ru/ (date of access: 24.10.2017)

[24] Register of property trades. https://www.etp-torgi.ru/market/ (date of access: 24.10.2017)

[25] Bidding for privatization, lease and sale of property. https://i.rts-tender.ru/main/auction/Trade/Search.aspx (date of access: 24.10.2017)

[26] Investment portal of the city of Moscow. https://investmoscow.ru/tenders/ (date of access: 24.10.2017)

[27] Sberbank-AST. Automated bidding system. http://www.sberbank-ast.ru/ (date of access: 24.10.2017)

[28] Portal of Services of the Federal Service of State Registration, Cadastre and Cartography. https://portal.rosreestr.ru/ (date of access: 24.10.2017)

[29] Python Software Foundation. https://www.python.org/ (date of access: 26.06.2017)