ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

WEB SITE RELIABILITY ANALYSIS USING THE PYTHON PARSING METHOD

Dmytro Gurin[1],
Svitlana Maksymova[1],
Vladyslav Yevsieiev[1],
Ahmad Alkhalaileh[2]

[1]Department of Computer-Integrated Technologies, Automation and Robotics, Kharkiv National University of Radio Electronics, Ukraine
[2]Senior Developer Electronic Health Solution, Amman, Jordan

ABSTRACT:

This article discusses methods for analyzing the reliability of a Web site using the parsing method based on the Python language. A generalized reliability analysis algorithm is presented that allows you to assess the quality and stability of the functioning of a website. The algorithm is based on parsing structural elements of the page, such as HTML tags, links , text blocks, and analysis of their state and contents. To implement the algorithm, a Python program was developed that is capable of automatically analyzing a selected website and identifying potential problems. The experiment showed the effectiveness and accuracy of the developed approach, which allows it to be used for systematic monitoring and increasing the reliability of websites.

Key words: Parsing, Website, Structure, Python, Manufacturing Innovation, Industrial Innovation

INTRODUCTION

In the modern world, websites are the main tool of communication and information exchange for organizations and users [1]-[10]. However, for effective use and improvement of web resources, it is necessary to analyze their structure. The development of a website structure analysis program using the parsing method based on the Python language is becoming more and more relevant and necessary.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

The main purpose of such a program is to study the internal structure of the website, its components and the connections between them. This allows you to evaluate the usability of the site, optimize its navigation and increase the overall efficiency of the web project. Analyzing the structure of the website also helps to identify errors and vulnerabilities, improve SEO indicators and increase the general availability of information.

One of the key tools for analyzing the structure of a website is the parsing method, which allows you to extract and analyze data on web pages. Python is used in completely different areas, starting from robotics [11]-[19] and ending with website analysis [20]-[25]. It is one of the most popular programming languages for creating web analytics programs due to its simplicity, flexibility and rich ecosystem of libraries.

Website structure analysis programs can be useful both for owners of web resources and for specialists in web development and SEO optimization. They allow you to systematize information about the structure of the site, identify problems and offer solutions for their elimination. Therefore, various methods and approaches can be used here [26]-[36].

Related works

Many works are devoted to the analysis of the reliability of websites. There are used plenty of methods and instruments to solve such problems. Let's consider several recent works on this topic.

Authors in [20] present an empirical study of 1,396 vulnerability reports affecting 698 Python packages in the Python ecosystem (PyPi). They study a set of 2,224 GitHub Python projects, to better understand the prevalence of vulnerabilities in their dependencies and how fast it takes to update them. Theirr findings show that the discovered vulnerabilities in Python packages are increasing over time, and they take more than 3 years to be discovered.

Utami, I. S., & Setiadi, H. in [21] evaluate and improve website performance by analyzing the quality of the Sebelas Maret University selection of new student admissions (SPMB) website by utilizing a combination of WebQual 4.0 and Importance Performance Analysis (IPA) methods.

The study [22] presents an nalysis of website quality by calculating webqual index to determine the level of quality of the website based on the WQI coefficient interval.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

The research [23] aims to develop and validate a scale to measure website service quality in the hotel industry, namely HWebSQ.

Scientists in [24] note that analyzing public information from social networking sites could produce exciting results and insights on the public opinion of almost any product, service, or behavior. This paper also discusses the sentiment analysis design, gathering data, training the data, and visualizing the data using the Python library.

The paper [25] explore how to develope a crawler with scontrolable and automatic crawling abilitys which can crawl specific target; Then the data that is crawled will be analysised and visualized by using Pandas library and Matplotlib library, the useful information will be extracted from the data analysis and visualization process, so as to complete.

Thus, we see a variety of areas of website analysis. Below in this article we will look at analyzing the reliability of sites using the Python parsing method.

Web site reliability algorithm and program for analyzing development

Analyzing the web page reliability using a Python-based parser has its own characteristics that are important to consider. First, Python provides a wide selection of libraries for working with web pages, such as requests for sending HTTP requests, BeautifulSoup for parsing HTML code, and others. This makes it easy to access and analyze page content. Second, Python has a convenient and understandable syntax that simplifies the development of scripts for analyzing web pages. A third feature is the extensive support of the Python community, which provides many useful packages, documentation, and code examples for a variety of web parsing tasks. In addition, Python can be used to automate routine tasks, such as regularly performing page reliability analysis or automating the process of monitoring connection stability. Another feature is the possibility of integration with other tools and services to obtain a wider range of functionality, for example, saving analysis results in a database or integration with web page monitoring services. Overall, Python is a powerful and easy-to-use tool for web site reliability analysis that makes this process fast, efficient, and convenient.

We will use the store's website at the address 'https://epicentrk.ua/ua/shop/' as a base site for the research of parsing methods for analyzing the reliability of the Web site.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

Before the development of the parsing program for analyzing the reliability of the Web site 'https://epicentrk.ua/ua/shop/ not based on the Python language, the following algorithm was developed, which is presented in Figure 1.

As you can see from Figure 1, the following libraries will be used in the developed program:

– requests is used to implement HTTP requests using the Python programming language. Using requests allows you to retrieve web page content, perform POST requests, pass parameters, manage cookies, and other aspects of HTTP requests. It is convenient for working with website APIs, receiving data for parsing or interacting with web servers.

– BeautifulSoup is a tool for parsing HTML and XML documents in Python. It allows you to parse incoming data from web pages and conveniently extract the necessary elements using CSS selectors, XPath or other methods. BeautifulSoup makes parsing HTML code simple and effective, and is used in many projects to retrieve data from the Internet.

pip install requests

pip install beautifulsoup4

The first command will install the requests library, and the second - beautifulsoup4. After installing these libraries, you can import them into your Python programs and use their functionality.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

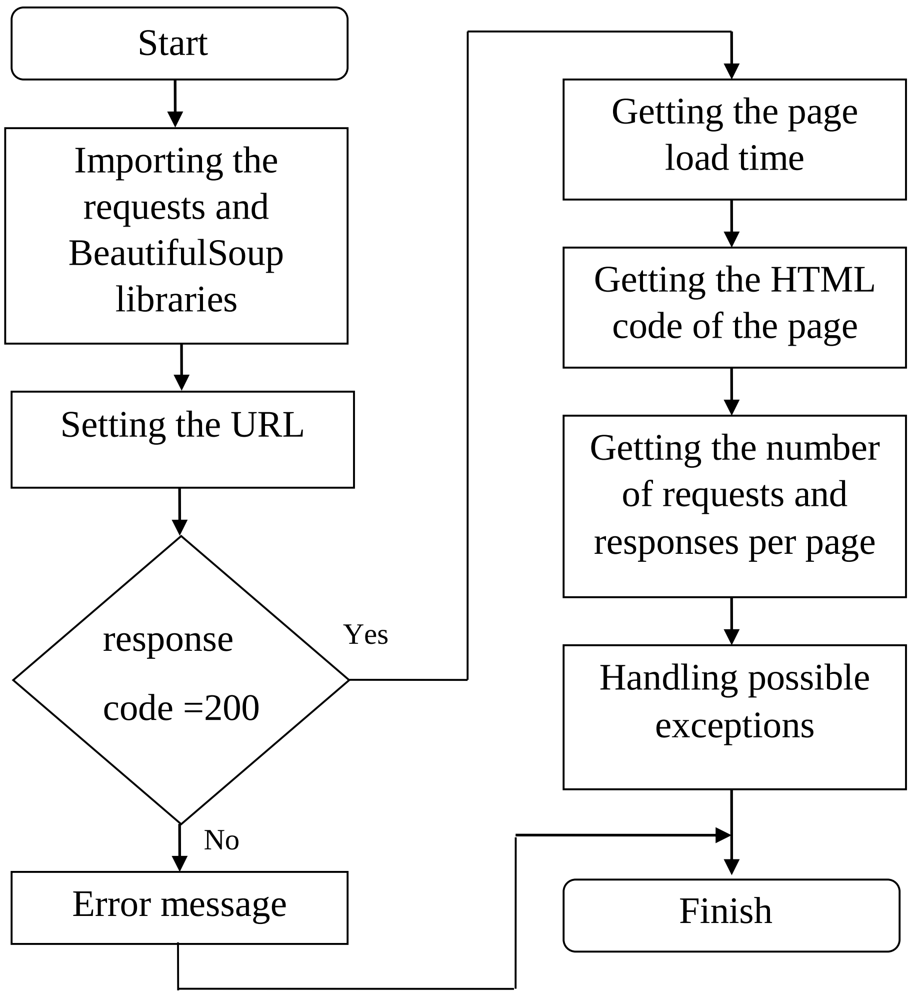

Start
Importing the requests and BeautifulSoup libraries
Setting the URL
response code =200
Yes
No
Error message
Getting the page load time
Getting the HTML code of the page
Getting the number of requests and responses per page
Handling possible exceptions
Finish

Figure 1: Web parsing program algorithm - site 'https://epicentrk.ua/ua/shop/

The developed software parsing code of the Web site with an explanation, is presented below:

```
import requests
from bs4 import BeautifulSoup
```

This piece of code is used to work with web pages in the Python programming language. Let's look at each part separately:

import requests: This line imports the requests module, which allows you to interact with HTTP requests. With its help, you can send HTTP requests to a web server and receive responses. For example, you can send a GET request to retrieve the content of a web page;

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

from bs4 import BeautifulSoup: This line imports the BeautifulSoup class from the bs4 module. BeautifulSoup is a library for working with HTML code. It allows you to easily parse HTML documents, access their elements and extract the necessary information. When a web page is submitted, you can use BeautifulSoup to parse its HTML code and extract the necessary data such as headers, links, text, etc.

So, these two import instructions allow you to make HTTP requests to web servers and parse the HTML code of web pages in a Python environment, making it easy to analyze and retrieve information from websites.

```
def analyze_webpage(url):
    try:
        # We send a request to the server and receive a response
        response = requests.get(url)
        # We check the status code of the server response
        if response.status_code == 200:
            print("Server response status code: 200 (Success)")
        else:
            print(f" Server response status code: {response.status_code}")
```

This piece of code is the analyze_webpage function, which sends a request to the web server at the specified URL and checks the status code of the server's response. Let's examine each line of code:

def analyze_webpage(url):: This is the declaration of the analyze_webpage function, which takes a single url parameter. This function is designed to parse the page from the specified URL.

try:: Start of the try-except block used to handle exceptions. In this case, we'll try to execute some code, and if an error occurs, we'll handle it.

response = requests.get(url): Uses the requests library to send an HTTP request to the specified URL. requests.get(url) makes a GET request for this URL and returns a response object that contains information about the server's response.

if response.status_code == 200:: We check the status code of the server's response. If the status code is 200, it means the server responded successfully, and we output the message "Server Response Status Code: 200 (Success)".

else:: In case the status code is not equal to 200, we print a message with the actual value of the status code.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

This code snippet demonstrates checking the status code of the server's response after sending a request to the specified URL, allowing you to determine whether the request was successful.

```
# We get the page load time
    load_time = response.elapsed.total_seconds()
    print(f"Page load time: {load_time} s")
```

This piece of code calculates the time it took to load the response page from the server. Let's take a closer look at this line:

response.elapsed is an attribute of the response object that represents the time it took to download the response.

.total_seconds() is a method that returns the total number of seconds in a timedelta object.

So this line of code calculates the time it took to load the response page from the server in seconds and assigns it to the load_time variable.

After this time is calculated, it is printed on the screen using the print function, which uses a formatted string to print the value of the load_time variable along with the text message "Page Load Time:"

```
# We get the HTML code of the page
    html_content = response.content
    soup = BeautifulSoup(html_content, 'html.parser')
```

This code snippet retrieves the HTML code of the response page from the server and parses it using the BeautifulSoup library. Let's break down each line:

response.content is an attribute of the response object that contains the content of the web page in bytes.

html_content is a variable to which the content of the page is written in byte form.

BeautifulSoup(html_content, 'html.parser') is a constructor for a BeautifulSoup object that creates a BeautifulSoup object for further HTML parsing.

html_content is a string with the byte content of the page we want to parse.

'html.parser' is a string that tells BeautifulSoup to use the built-in HTML parser to process HTML code.

So this piece of code allows you to get the HTML code of a web page and create a BeautifulSoup object that can be used to further parse and analyze the HTML structure of the page.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

```
# We get the number of requests and responses on the page
requests_count = len(soup.find_all())
print(f"The number of requests and responses on the page: {requests_count}")
```

This piece of code defines the number of requests and responses per page. Let us consider each line:

soup.find_all() - This BeautifulSoup method is used to find all elements on the page that match the specified criteria. With no arguments in the find_all() method, it returns a list of all elements on the page.

len() - this function is used to calculate the number of elements in a list.

requests_count is a variable that records the number of items found, which reflects the number of requests and responses on the page.

print() - this function displays a text message on the screen.

f"Number of requests and responses per page: {requests_count}" is a formatted string in which the value of the requests_count variable is substituted as a number. This message is displayed on the screen and shows the number of requests and responses on the page.

So, this piece of code allows you to find and display the number of requests and responses on a website page.

```
# We study the metadata of the page
title = soup.title.text if soup.title else " The page title is missing "
meta_description = soup.find('meta', {'name': 'description'})
description = meta_description['content'] if meta_description else " There is no meta description "
print(f"Page title: {title}")
print(f"Page meta description: {description}")
```

This piece of code parses the page's metadata, such as the title and meta description. Let us consider each line:

soup.title - this expression is used to find the <title> tag in the HTML structure of the page, if such a tag exists.

.text is an attribute that gets the textual content of the tag. If the <title> tag exists, .text will return its text content.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

if soup.title else "Page title is missing" is a Python ternary operator that checks if the <title> tag exists. If the <title> tag exists, the value of the title variable will be equal to the text content of the <title> tag. Otherwise, the string "Page name is missing" is set.

soup.find('meta', {'name': 'description'}) - This BeautifulSoup method is used to find the <meta> tag with the name='description' attribute. The <meta name='description'> tag is often used to define the meta description of a page.

meta_description['content'] - this expression receives the value of the content attribute of the <meta> tag, if such a tag exists and contains the content attribute.

if meta_description else "No meta description" - this ternary operator checks if the <meta name='description'> tag exists. If such a tag exists and has a content attribute, the value of the description variable will be equal to the value of the content attribute. Otherwise, the line "Meta-description is missing" is set.

So, this piece of code allows you to get and display the title and meta description of the page.

```
except Exception as e:
    print(f"Error: {e}")
```

This code snippet handles exception handling using the try-except construct. Let's take a look at this snippet:

except Exception as e: - this construct is used to handle any exception that occurs during the execution of the code in the try block. Exception is the base class for all exceptions in Python.

as e - tells Python that the exception that is thrown will be accessed through the e variable.

print() - this function is used to display an error message on the screen.

f"Error: {e}" is a formatted string in which the value of the variable e (the exception) is inserted into the message text. Thus, an error message is displayed along with the error text itself.

This piece of code allows you to catch any exceptions that occur during the execution of the try block and print an error message to the screen describing the error.

```
# Call the function to analyze the web page
url = 'https://epicentrk.ua/ua/shop/' # Replace the URL with the desired one
analyze_webpage(url)
```

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

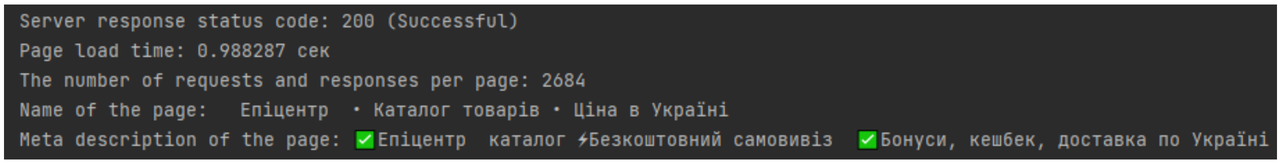
This code snippet calls the analyze_webpage(url) function to analyze the web page at the specified URL. Let's take a look at this snippet:

url = 'https://epicentrk.ua/ua/shop/' - the URL of the page to be analyzed is set. In this case, this is the page of the online store "Epicenter K", but you can replace this URL with any other that you want to analyze.

analyze_webpage(url) - This command calls the analyze_webpage() function with the specified URL as an argument. The function will parse the page for this URL.

So this piece of code calls a function to parse the web page at the specified URL.

The results of parsing the website https://epicentrk.ua/ua/shop/ based on the Python language are presented in Figure 2.



```
Server response status code: 200 (Successful)
Page load time: 0.988287 сек
The number of requests and responses per page: 2684
Name of the page:   Епіцентр  • Каталог товарів • Ціна в Україні
Meta description of the page: ✅Епіцентр  каталог ⚡Безкоштовний самовивіз  ✅Бонуси, кешбек, доставка по Україні
```

Figure 2: The results of parsing the website https://epicentrk.ua/ua/shop/

Conclusion

As a result of the conducted research, a generalized algorithm for analyzing the websites reliability was developed. It was based on the methods of parsing web pages using the Python programming language. This algorithm allows you to effectively identify potential problems in the functioning of websites, such as unavailability of pages, errors in the content or structure of the site.

The program developed in Python successfully implements the proposed algorithm and is capable of automatically analyzing selected websites. In the process of experimental research, the effectiveness and accuracy of the developed approach was verified. The results showed that the algorithm and the program are able to detect various types of problems, which allows them to be used for systematic monitoring and increasing the reliability of websites.

This approach to the websites reliability analysis has practical significance for owners and administrators of web resources, as it allows to quickly identify and eliminate problems that can lead to a decrease in the quality of user service. Also, this approach can be used to automate the process of monitoring and increase the reliability

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

of websites, which will save time and resources in the maintenance and support of web resources.

REFERENCES:

1. Omarov, M., Tikhaya, T., & Lyashenko, V. (2019). Internet marketing metrics visualization methodology for related search queries. International Journal of Advanced Trends in Computer Science and Engineering, 8(5), 2277-2281.

2. Dadkhah, M., Lyashenko, V. V., Deineko, Z. V., Shamshirband, S., & Jazi, M. D. (2019). Methodology of wavelet analysis in research of dynamics of phishing attacks. International Journal of Advanced Intelligence Paradigms, 12(3-4), 220-238.

3. Vasiurenko, O., Lyashenko, V., Baranova, V., & Deineko, Z. (2020). Spatial-Temporal Analysis the Dynamics of Changes on the Foreign Exchange Market: an Empirical Estimates from Ukraine. Journal of Asian Multicultural Research for Economy and Management Study, 1(2), 1-6.

4. Deineko, Zh., & et al.. (2021). Features of Database Types. International Journal of Engineering and Information Systems (IJEAIS), 5(10), 73-80.

5. Sotnik, S., & et al.. (2023). Development Features Web-Applications. International Journal of Academic and Applied Research (IJAAR), 7(1), 79-85.

6. Sotnik, S. Overview: PHP and MySQL Features for Creating Modern Web Projects/ S Sotnik, V. Manakov, V. Lyashenko //International Journal of Academic Information Systems Research (IJAISR). – 2023. – Vol. 7, Issue 1. – P. 11-17.

7. Lyashenko, V., Kobylin, O., & Baranchykov, Y. (2018, October). Ideology of Image Processing in Infocommunication Systems. In 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T) (pp. 47-50). IEEE.

8. Z. Deineko, S. Sotnik, V. Lyashenko, "Multimedia Systems in Education," International Journal of Academic Information Systems Research (IJAISR). 2022, vol. 6 issue 7, pp. 23-28.

9. Kuzomin, O., & et al.. (2020). Mobile Expert System for Diagnostic Human State in Emergency Situations. International Journal of Advanced Trends in Computer Science and Engineering, 9(4), 6485-6489.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

10. Omarov, M., Tykha, T., & Lyashenko, V. (2019). Use of Wavelet Techniques in the Study of Internet Marketing Metrics. Eskişehir Technical University Journal of Science and Technology A-Applied Sciences and Engineering, 20, 157-163.

11. Stetsenko, K., & et al. (2023). Exploring BEAM Robotics for Adaptive and Energy-Efficient Solutions. Multidisciplinary Journal of Science and Technology, 3(4), 193-199.

12. Yevsieiev, V., & et al. (2024). Using Contouring Algorithms to Select Objects in the Robots' Workspace. Technical Science Research In Uzbekistan, 2(2), 32-42.

13. Shcherbyna, V., & et al. (2023). Mobile Robot for Fires Detection Development. Journal of Universal Science Research, 1(11), 17-27.

14. Yevsieiev, V., & et al. (2024). Active Contours Method Implementation for Objects Selection in the Mobile Robot's Workspace. Journal of Universal Science Research, 2(2), 135-145.

15. Nevliudov, I., & et al. (2023). Mobile Robot Navigation System Based on Ultrasonic Sensors. In2023 IEEE XXVIII International Seminar/Workshop on Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory (DIPED),IEEE, 1, 247-251.

16. Yevsieiev, V., & et al. (2024). Object Recognition and Tracking Method in the Mobile Robot's Workspace in Real Time. Technical Science Research In Uzbekistan, 2(2), 115-124.

17. Basiuk, V., & et al. (2023). Mobile Robot Position Determining Using Odometry Method. Multidisciplinary Journal of Science and Technology, 3(3), 227-234.

18. Yevsieiev, V., & et al. (2024). The Canny Algorithm Implementation For Obtaining the Object Contour in a Mobile Robot's Workspace in Real Time. Journal of Universal Science Research, 2(3), 7-19.

19. Yevsieiev, V., & et al. (2022). A robotic prosthetic a control system and a structural diagram development. Collection of scientific papers «ΛΌΓΟΣ», Zurich, Switzerland, 113-114.

20. Alfadel, M., & et al. (2023). Empirical analysis of security vulnerabilities in python packages. Empirical Software Engineering, 28(3), 59.

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

21. Utami, I. S., & Setiadi, H. (2021). Analysis the effect of website quality on user satisfaction with the WebQual 4.0 method and importance-performance analysis (IPA)(case study: SPMB Sebelas Maret University's Website). In Journal of Physics: Conference Seriesб IOP Publishing, 1842(1), p. 012003.

22. Syahputri, K., & et al. (2021). Analysis of website service quality with webqual 4.0 integration method. In IOP Conference Series: Materials Science and Engineering, IOP Publishing, 1122(1), p. 012035.

23. Nguyen, H. T. T., & et al. (2020). Development and validation of a scale measuring hotel website service quality (HWebSQ). Tourism Management Perspectives, 35, 100697.

24. Gunawan, T. S., & et al. (2020). Social network analysis using python data mining. In 2020 8th international conference on cyber and IT service management (CITSM), IEEE, 1-6.

25. Wu, H., & et al. (2020). Data analysis and crawler application implementation based on python. In 2020 International Conference on Computer Network, Electronic and Automation (ICCNEA), IEEE, 389-393.

26. Nevliudov, I., Yevsieiev, V., Baker, J. H., Ahmad, M. A., & Lyashenko, V. (2020). Development of a cyber design modeling declarative Language for cyber physical production systems. J. Math. Comput. Sci., 11(1), 520-542.

27. Nevliudov, I., & et al.. (2020). Method of Algorithms for Cyber-Physical Production Systems Functioning Synthesis. International Journal of Emerging Trends in Engineering Research, 8(10), 7465-7473.

28. Abu-Jassar, A. T., Attar, H., Yevsieiev, V., Amer, A., Demska, N., Luhach, A. K., & Lyashenko, V. (2022). Electronic User Authentication Key for Access to HMI/SCADA via Unsecured Internet Networks. Computational intelligence and neuroscience, 2022, 5866922.

29. Al-Sherrawi, M. H., Lyashenko, V., Edaan, E. M., & Sotnik, S. (2018). Corrosion as a source of destruction in construction. International Journal of Civil Engineering and Technology, 9(5), 306-314.

30. Lyashenko, V. V., Deineko, Z. V., & Ahmad, M. A. Properties of wavelet coefficients of self-similar time series. In other words, 9, 16.

31. Nevliudov, I., Yevsieiev, V., Lyashenko, V., & Ahmad, M. A. (2021). GUI Elements and Windows Form Formalization Parameters and Events Method to

ISSN (E): 2181-4570 ResearchBib Impact Factor: 6,4 / 2023 SJIF 2024 = 5.073/Volume-2, Issue-5

Automate the Process of Additive Cyber-Design CPPS Development. Advances in Dynamical Systems and Applications, 16(2), 441-455.

32. Mustafa, S. K., Yevsieiev, V., Nevliudov, I., & Lyashenko, V. (2022). HMI Development Automation with GUI Elements for Object-Oriented Programming Languages Implementation. SSRG International Journal of Engineering Trends and Technology, 70(1), 139-145.

33. Kuzomin, O., Ahmad, M. A., Kots, H., Lyashenko, V., & Tkachenko, M. (2016). Preventing of technogenic risks in the functioning of an industrial enterprise. International Journal of Civil Engineering and Technology, 7(3), 262-270.

34. Al-Sharo, Y. M., Abu-Jassar, A. T., Sotnik, S., & Lyashenko, V. (2023). Generalized Procedure for Determining the Collision-Free Trajectory for a Robotic Arm. Tikrit Journal of Engineering Sciences, 30(2), 142-151.

35. Lyashenko, V., Laariedh, F., Ayaz, A. M., & Sotnik, S. (2021). Recognition of Voice Commands Based on Neural Network. TEM Journal: Technology, Education, Management, Informatics, 10(2), 583-591.

36. Maksymova, S., Matarneh, R., & Lyashenko, V. V. (2017). Software for Voice Control Robot: Example of Implementation. Open Access Library Journal, 4, e3848.