Editors
Adam Wójtowicz, Daniel Wilusz,
Olena Kryvoruchko, Volodymyr Tokar

Challenges and Reality of the IT-space:

Software Engineering and Cybersecurity

International Conference SECS-2022
October 25–26th, 2022

Proceedings Book

Institute of Bioorganic Chemistry, Polish Academy of Sciences
Scientific Publishers OWN

Editors

ADAM WÓJTOWICZ
Poznań University of Economics and Business
Department of Information Technology
al. Niepodległości 10, 61-875 Poznań, Poland
e-mail: awojtow@kti.ue.poznan.pl

DANIEL WILUSZ
Poznań University of Economics and Business
Department of Information Technology
al. Niepodległości 10, 61-875 Poznań, Poland
e-mail: wilusz@kti.ue.poznan.pl

OLENA KRYVORUCHKO
State University of Trade and Economics
Department of Software Engineering and Cybersecurity
19 Kyoto Street, Kyiv, Ukraine, 02156
e-mail: kryvoruchko_ev@knute.edu.ua

VOLODYMYR TOKAR
State University of Trade and Economics
Department of Software Engineering and Cybersecurity
19 Kyoto Street, Kyiv, Ukraine, 02156
e-mail: v.tokar@knute.edu.ua

ISBN 978-83-7712-049-1

Institute of Bioorganic Chemistry
Polish Academy of Sciences
Scientific Publishers OWN

Printed in Poland

Preface

The present book includes a set of selected papers from the conference "Challenges and Reality of the IT-space: Software Engineering and Cybersecurity" (SECS 2022) that was held online on October 25–26th 2022. The conference has been organized by Department of Software Engineering and Cybersecurity (State University of Trade and Economics, Ukraine) and Department of Information Technology (Poznań University of Economics and Business, Poland). The conference committee has received 33 articles, out of which 16 have been accepted for the conference presentation based on peer-review process and the best 10 articles have been selected for the publishing in this book. It gives the acceptance rate of approximately 0.3. The decisions to organize the conference in online mode and to make the conference free of charge have been intended to popularize the content of these valuable works.

It has to be noted that conference participants from all over Ukraine have managed to contribute to the conference despite the ongoing hybrid war. Not only the external conditions, also the topic of the conference has been challenging. The state-of-the-art achievements in the fields of cybersecurity and reliable software engineering are of significant importance in the circumstances of the foreign state aggression. Obviously the war takes place also in the cyberspace. In this challenging situation, it is a proper decision to stimulate research in this field by organizing a meeting place for researchers willing to exchange their ideas and results for achieving solid IT infrastructure.

The successful collaboration between Ukrainian and Polish researchers on this new conference is the first step towards further collaboration on joint research and educational projects as well as on subsequent editions of the conference.

Adam Wójtowicz
Daniel Wilusz
Olena Kryvoruchko
Volodymyr Tokar

January 2023

Conference Co-chairs and Program Committee

Conference Co-chairs

Anatolii Mazaraki
Olena Kryvoruchko
Adam Wójtowicz
Daniel Wilusz
Natalia Kotenko

Program Committee

Wojciech Cellary	WSB University in Poznań, Poland
Jakub Flotyński	Poznań University of Economics and Business, Poland
Benjamin Gateau	Luxembourg Institute of Science and Technology, Luxembourg
Omiros Iatrellis	University of Thessaly, Greece
Piotr Kałużny	Poznań University of Economics and Business, Poland
Oleksandr Korneiko	National Academy of Internal Affairs of Ukraine, Ukraine
Valerii Kraskevych	State University of Trade and Economics, Ukraine
Olena Kryvoruchko	State University of Trade and Economics, Ukraine
Valerii Lakhno	National University of Life and Environmental Sciences of Ukraine, Ukraine
Natalia Lukova-Chuiko	Taras Shevchenko National University of Kyiv, Ukraine
Serhii Ostapov	Yuriy Fedkovych Cherinivtsi National University, Ukraine
Jarogniew Rykowski	Poznań University of Economics and Business, Poland
Ihor Tereikovskyi	National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, Ukraine
Mykola Tsutsura	Kyiv National University of Construction and Architecture, Ukraine
Svitlana Tsutsura	Kyiv National University of Construction and Architecture, Ukraine
Yurii Ushenko	Yuriy Fedkovych Cherinivtsi National University, Ukraine
Paweł Weichbroth	Gdańsk University of Technology, Poland
Daniel Wilusz	Poznań University of Economics and Business, Poland
Adam Wójtowicz	Poznań University of Economics and Business, Poland

Contents

Cybersecurity

Software Engineering

Cybersecurity

Long Tail of Security Vulnerabilities and Nation State APT Actors

Daniel Wilusz[1] [0000-0001-8678-6312], Dominik Sadowczyk[2],
Adam Wójtowicz[1] [0000-0003-1276-0915], and Leszek Tasiemski[2]

[1] Department of Information Technology, Poznan University of Economics and Business
{wilusz, awojtow}@kti.ue.poznan.pl
[2] WithSecure Corporation
{dominik.sadowczyk, leszek.tasiemski}@withsecure.com

Abstract. Recently numerous Advanced Persistent Threat groups originating from various countries have been identified, carrying out a wide range of attacks from spear phishing to exploits focused on various entities both commercial and governmental and even military. Many of them exploit zero-day unknown vulnerabilities for which no patch is available, however there are also many cases in which the patch is publicly known and perfectly accessible to the software administrator, but still it is not applied to vulnerable software. This phenomenon is analyzed in the presented work. The list of most commonly exploited vulnerabilities has been cross-referenced with commonly available reports of APT actors' activity, and checked against the raw data from a massively used vulnerability management solution. The authors postulate that APT groups successfully exploit the "long tail" of security vulnerabilities that remain unpatched for months and even years, despite the availability of a fix.

Keywords: cybersecurity, software vulnerabilities, security patch, unpatched vulnerability, vulnerability long tail, advanced persistent threat

1 Introduction

Advanced Persistent Threat (APT) is an adversary with sophisticated levels of expertise, motivation and significant resources often provided by nation states, which allow it to create opportunities to achieve its objectives by using multiple attack vectors [9], [21], [22]. APT is characterized by a repeated and prolonged pursuit of its objectives, adaptation to the target's defense mechanisms, and determination to maintain the level of attack intensity to achieve its objectives [22]. Numerous APTs have been identified originating from various countries [11], [14], [18], [19], [29], [31], carrying out a wide range of attacks from spear phishing to exploits focused on various entities both commercial and governmental and even military [5], [17].

Nation state actors take advantage of vulnerable software to conduct espionage or sabotage operations in cyberspace. Many of them exploit unknown vulnerabilities (so called zero-days), for which no patch is available as even the vendor is not aware of their existence. The serious challenge of zero-days is extensively researched in literature [2], [3], [16], [27], [28]. However, in practice there are many cases in which the patch is publicly known and perfectly accessible to the software administrator, but despite those circumstances it is not applied to vulnerable software [15], [24]. This phenomenon is analyzed in the presented work. The authors of this article postulate that APT groups successfully exploit the "long tail" of security vulnerabilities in systems that remain unpatched for months and even years [13], despite the availability of a fix. In this paper the long tail of security vulnerabilities is defined as a set of well-known vulnerabilities that have not been patched by entities that should mitigate them within the average time taken to fix critical cybersecurity vulnerabilities. The average time taken to fix cybersecurity vulnerability that has been reported in recent sources varies from 200 days [25] to 205 days [33] (256 days for high severity vulnerabilities) and the value of 205 days has been applied to the presented analysis.

While analyzing the phenomenon of long-tail vulnerabilities the following research questions arise:

- What is a specificity of the vulnerabilities that are frequently unpatched in software despite the availability of the patch?
- What is the risk level related to the existence of long-tail vulnerabilities in the context of known APT attacks (CVE severity)?
- What are the main groups of factors that build barriers to applying the patch in time for software administrators?
- What are possible countermeasures or recommendations for various stakeholders that could mitigate the risks?

The rest of the paper is organized as follows. In Section 2 the background for the presented work is described. The research method used in the data analysis of the long tail vulnerabilities and corresponding APTs is presented in Section 3. Section 4 presents and explains the results of the analysis. A discussion of the findings and conclusions are presented in Section 5.

2 Background

In the vulnerability timeline the following key points can be distinguished:

1. Vulnerability is introduced by a vendor or open-source contributor.
2. The vulnerability is detected and confidentially reported to the security organization or to the software vendor.
3. The vulnerability description is published.
4. The vulnerability is patched and the patch is released (in some cases order of the activities 3 and 4 is changed).
5. The patch is deployed by the end user.

Vulnerabilities are introduced into software during the development process as a result of various intentional or unintentional events: bugs in source code, bugs in software libraries, and errors in configuration or testing procedures. Users introduce vulnerabilities to their systems as a result of third-party software installation, its upgrade (new features – new bugs) or installation of custom extensions. Sometimes a vulnerability may stem from improper configuration – like the use of unsafe protocols or weak cryptographic ciphers. Vulnerability identification can occur as a result of testing performed by the testing team, analysis of the source code by the product community, or exploitation attempts performed by external or internal adversaries. It may be a matter of discovering a weakness in a cryptographic procedure, rendering it vulnerable. Once a vulnerability is discovered, it is described and published by security organizations such as National Vulnerability Database (NVD) [23] or MITRE's CVE [20]. Once the specification of the vulnerability is known, software vendors release a patch, and security organizations make preliminary security recommendations. Once the patch is released, system administrators install it on their systems.

The majority of software vendors and security researchers respect the code of "responsible disclosure", where a memorandum on releasing the information is mutually agreed between the vendor and researcher. Thanks to this, the information about the vulnerability is published simultaneously with the fix, minimizing the time when systems are exposed to an attack. In current-day bug bounty programs, responsible disclosure behavior is also motivated by a financial reward to the researcher.

The area of research on security vulnerabilities is dominated by numerous zero-day vulnerability studies [26]. Researchers are working on detecting and preventing zero-day attacks, proposing a range of techniques and frameworks [2], [3], [16], [27], [28]. Kotzias et. al examined the issue of patching delays across tens of millions of client hosts for dozen client-side and over one hundred server-side applications and noted that up to nine months is required to patch 90% of server-side hosts [15]. Sarabi et. al. studied the patching behavior of more than 400 thousand users and found that many hosts stay unpatched even with known and exploited vulnerabilities [24]. Allodi et. al. have developed a theoretical model that proves the following theses: an attacker exploits only one vulnerability for a given software version, frequently chooses vulnerabilities that require low attack complexity, and prepares the exploitation of new vulnerabilities in a slow manner [1].

Despite numerous studies on new vulnerabilities and the problem of patching known vulnerabilities, to the best of our knowledge, there are no studies on the matching of known, often unpatched vulnerabilities (named in this paper as "long tail"), with known APT groups for which history of exploitation of these vulnerabilities has been officially proven.

3 Method

In order to analyze the phenomenon of the long tail of security vulnerabilities, the authors took a list of most commonly exploited vulnerabilities, maintained and published officially by the CISA organization [6], and cross-referenced it with commonly available reports of APT actors' activity [4], [12], [18], [19], [29]. That list of CVEs was then checked against the raw data from the vulnerability management solution [32]. Because of anonymization requirements, the results are shown as a percentage of vulnerable assets out of total active assets known to the product. The asset is defined here as a host or other type of device that has an IP address and is connected to a corporate network. It can be a server, desktop, workstation, network device, printer, mobile (rare case), or IoT device. It is also indicated when the vulnerability was first reported and when it was seen most recently to give insight into a time span and the long tail effect of vulnerabilities.

Because of confidentiality reasons, absolute numbers cannot be used in this analysis, and it operates on percentage values of affected assets in the whole asset pool of the database. It has to be stressed that even if percentage values are low, it means thousands of instances in absolute numbers. Also, the percentage values are calculated in relation to all assets, all architectures, and all operating systems. For instance, missing security updates for Windows affect only assets running this operating system, but the percentage is calculated relative to the size of the full asset base, which obviously includes many non-Windows machines. Lastly, presented data come from the vulnerability management product database [32] which tends to be used by mature organizations, that actively manage their vulnerabilities (SMB and SME organizations; the product is being sold through a channel where partners are usually MSP/MSSP IT companies providing services to their end customers). It is therefore safe to assume that the long tail percentages would look much worse in the general population of global assets. On the other hand, APT actors tend to target mature organizations, hence the target group of APT actors can be compared to a representative sample of vulnerability management product customers.

4 Results

The 200,000 raw data sample obtained on 29. August 2022 from WithSecure(TM) Elements Vulnerability Management solution [32] was filtered to include vulnerabilities that were first reported before 205 days of the survey (which is the average time taken to fix cybersecurity vulnerability [33]). The data was sorted by the percentage of assets affected by the vulnerability. A CVE code or set of codes was then identified for a particular vulnerability (many CVEs can be related to one vulnerability). Then, based on the CISA Known Exploited Vulnerabilities Catalog [6], vulnerabilities have been limited to only those for which an exploit has been reported and finally the list of vulnerabilities has been limited to the top twenty-five. Vulnerabilities with an assigned percentage of affected assets and the time of first reporting are shown in Table 1. Analysis of the data in Table 1 reveals that the top 25 list is dominated by vulnerabilities reported earlier than 3 years since the time of the research. Figure 1 shows a slow decline in the number of vulnerabilities in the top 25 list relative to their first report time.

Table 1. Long tail vulnerabilities based on raw data obtained from solution [32], first reported before the average time taken to fix cybersecurity vulnerability, sorted by the percentage of affected assets (top 25), limited to those for which an exploit has been reported [6], with corresponding CVE codes

VID	Vulnerability	Percentage affected [%]	First time reported	CVE
1	Teamviewer through 14.7.1965 Improper Authentication Vulnerability	1.71	18.02.2020	2019–18988
2	Intel Management Engine Components Privilege Escalation Vulnerability	0.91	22.11.2017	2017–5689
3	Apache Log4j2 Remote Code Execution Vulnerability (Authenticated Check for Windows)	0.82	15.12.2021	2021–44228
4	January 2022 Security Updates	0.65	17.01.2022	2022–21882
5	Remote Desktop Services Remote Code Execution Vulnerability BlueKeep	0.60	01.06.2019	2019–0708
6	Apache Tomcat before 7.0.100, 8.5.51 and 9.0.31 File Inclusion Vulnerability	0.54	27.02.2020	2020–1938
7	October 2021 Security Updates (including remote code execution in MS Office)	0.54	15.10.2021	2021–37976
				2021–37975
8	Samba before 4.10.18, 4.11.13, 4.12.7 Netlogon Elevation of Privilege Vulnerability	0.47	22.09.2020	2020–1472
9	November 2021 Security Updates	0.41	12.11.2021	2021–41379
				2021–42278
				2021–42287
				2021–42292
				2021–42321
10	September 2021 Security Updates	0.41	27.09.2021	2021–38647
				2021–38645
				2021–38648
				2021–38649
				2021–38646
				2021–40444
11	March 2021 Security Updates	0.39	05.03.2021	2021–21193
				2021–27059
				2021–26855
				2021–26857
				2021–26858
				2021–27065
				2021–26411
				2021–21166
12	May 2021 Security Updates	0.38	17.05.2021	2021–31207
				2021–31166
13	October 2017 Security Updates	0.38	24.10.2017	2017–11774
				2017–11826
14	July 2020 Security Updates	0.37	20.07.2020	2020–1350
				2020–1040
				2020–1147

Table 1 (continued). Long tail vulnerabilities based on raw data obtained from solution [32], first reported before the average time taken to fix cybersecurity vulnerability, sorted by the percentage of affected assets (top 25), limited to those for which an exploit has been reported [6], with corresponding CVE codes

VID	Vulnerability	Percentage affected [%]	First time reported	CVE
15	MS17-010: Microsoft Windows SMB remote code execution (WannaCry)	0.33	19.05.2017	2017–0143
16	Apache Tomcat before 7.0.82, 8.0.47, 8.5.23 and 9.0.1 Remote Code Execution Vulnerability	0.32	30.10.2017	2017–12617
17	September 2017 Security Updates	0.31	14.09.2017	2017–8759
18	Oracle Java is Missing June 2013 Critical Patch	0.31	03.06.2014	2013–2465
19	January 2021 Security Updates	0.27	19.01.2021	2021–1647
20	January 2018 Security Updates	0.27	16.01.2018	2018–0798
				2018–0802
21	MS15-081: Vulnerabilities in Microsoft Office Could Allow Remote Code Execution	0.27	14.12.2015	2015–1642
22	Oracle Java is Missing April 2013 Critical Patch	0.26	03.06.2014	2013–2423
23	Oracle Java is Missing February 2013 Critical Patch	0.25	03.06.2014	2013–0431
24	Adobe Reader is Missing APSB13-15 Security Update	0.25	14.12.2015	2013–2729
				2013–3346
25	Adobe Reader is Missing APSB13-07 Security Update	0.25	02.11.2016	2013–0640
				2013–0641

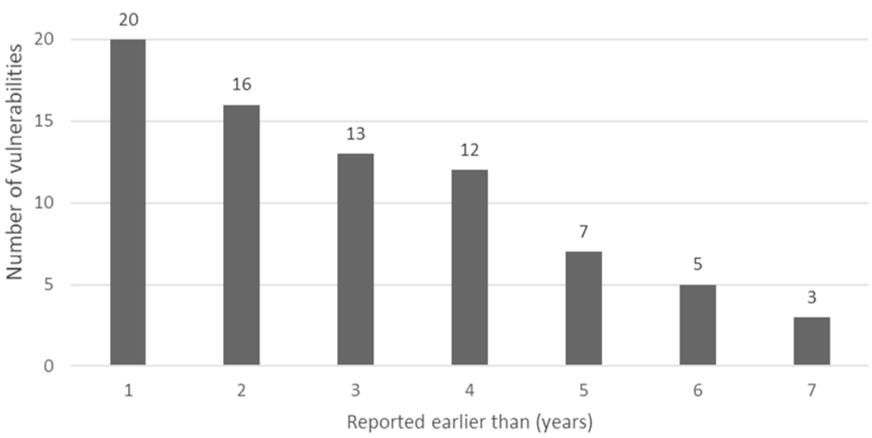


Fig. 1. Number of vulnerabilities older than given number of years

In the next step a base score and a CVSS 2.0 vector have been assigned to each CVE code based on the NIST National Vulnerability Database to facilitate comparative analysis of attributes of long tail vulnerabilities. These data are presented in Table 2. It is worth noting that 24 of the 49 CVE's have a base score greater than 7, which denotes high severity. Moreover, none of the CVEs in Table 2 has a base score indicating low severity. Further analysis of the data in Table 2 shows that for as many as 41 of the 49

CVEs a network is the access vector, 22 CVEs are characterized by low access complexity and as many as 43 CVEs require no authentication. In terms of impact on the attributes of confidentiality, integrity, and availability, the 20 CVEs are characterized by an impact on all three.

Table 2. Top 25 long tail vulnerabilities with CVSS 2.0 vectors assigned [23]

VID	CVE	Base Score	Access Vector	Access Complexity	Authentication	Confidentiality impact	Integrity impact	Availability Impact
1	2019-18988	4,4	Local	Medium	None	Partial	Partial	Partial
2	2017-5689	10	Network	Low	None	Complete	Complete	Complete
3	2021-44228	9,3	Network	Medium	None	Complete	Complete	Complete
4	2022-21882	7,2	Local	Low	None	Complete	Complete	Complete
5	2019-0708	10	Network	Low	None	Complete	Complete	Complete
6	2020-1938	7,5	Network	Low	None	Partial	Partial	Partial
7	2021-37976	4,3	Network	Medium	None	Partial	None	None
	2021-37975	6,8	Network	Medium	None	Partial	Partial	Partial
8	2020-1472	9,3	Network	Medium	None	Complete	Complete	Complete
9	2021-41379	4,6	Local	Low	None	Partial	Partial	Partial
	2021-42278	6,5	Network	Low	Single	Partial	Partial	Partial
	2021-42287	6,5	Network	Low	Single	Partial	Partial	Partial
	2021-42292	6,8	Network	Medium	None	Partial	Partial	Partial
	2021-42321	6,5	Network	Low	Single	Partial	Partial	Partial
10	2021-38647	7,5	Network	Low	None	Partial	Partial	Partial
	2021-38645	4,6	Local	Low	None	Partial	Partial	Partial
	2021-38648	4,6	Local	Low	None	Partial	Partial	Partial
	2021-38649	4,6	Local	Low	None	Partial	Partial	Partial
	2021-38646	6,8	Network	Medium	None	Partial	Partial	Partial
	2021-40444	6,8	Network	Medium	None	Partial	Partial	Partial

Table 2 (continued). Top 25 long tail vulnerabilities with CVSS 2.0 vectors assigned [23]

VID	CVE	Base Score	Access Vector	Access Complexity	Authentication	Confidentiality impact	Integrity impact	Availability Impact
11	2021-21193	6,8	Network	Medium	None	Partial	Partial	Partial
	2021-27059	8,5	Network	Medium	Single	Complete	Complete	Complete
	2021-26855	7,5	Network	Low	None	Partial	Partial	Partial
	2021-26857	6,8	Network	Medium	None	Partial	Partial	Partial
	2021-26858	6,8	Network	Medium	None	Partial	Partial	Partial
	2021-27065	6,8	Network	Medium	None	Partial	Partial	Partial
	2021-26411	5,1	Network	High	None	Partial	Partial	Partial
	2021-21166	6,8	Network	Medium	None	Partial	Partial	Partial
12	2021-31207	6,5	Network	Low	Single	Partial	Partial	Partial
	2021-31166	7,5	Network	Low	None	Partial	Partial	Partial
13	2017-11774	6,8	Network	Medium	None	Partial	Partial	Partial
	2017-11826	9,3	Network	Medium	None	Complete	Complete	Complete
14	2020-1350	10	Network	Low	None	Complete	Complete	Complete
	2020-1040	7,7	Adjacent	Low	Single	Complete	Complete	Complete
	2020-1147	6,8	Network	Medium	None	Partial	Partial	Partial
15	2017-0143	9,3	Network	Medium	None	Complete	Complete	Complete
16	2017-12617	6,8	Network	Medium	None	Partial	Partial	Partial
17	2017-8759	9,3	Network	Medium	None	Complete	Complete	Complete
18	2013-2465	10	Network	Low	None	Complete	Complete	Complete
19	2021-1647	7,2	Local	Low	None	Complete	Complete	Complete
20	2018-0798	9,3	Network	Medium	None	Complete	Complete	Complete
	2018-0802	9,3	Network	Medium	None	Complete	Complete	Complete
21	2015-1642	9,3	Network	Medium	None	Complete	Complete	Complete
22	2013-2423	4,3	Network	Medium	None	None	Partial	None
23	2013-0431	5	Network	Low	None	None	Partial	None
24	2013-2729	10	Network	Low	None	Complete	Complete	Complete
	2013-3346	10	Network	Low	None	Complete	Complete	Complete
25	2013-0640	9,3	Network	Medium	None	Complete	Complete	Complete
	2013-0641	9,3	Network	Medium	None	Complete	Complete	Complete

Table 3. Long tail vulnerabilities cross-referenced with data on APT actors' activity [18], [19], [29], [12], [4]

VID	CVE	CVE category name	CISA TOP	APT	Suspected origin
3	2021-44228	Apache Log4j2 JNDI configuration, log messaging, and parameterization features not protecting against attacker-controlled LDAP and other endpoints	2021	Magic Hound	Iran
				Aquatic Panda	China
				Lazarus	North Korea
				Mercury	Iran
8	2020-1472	Netlogon Elevation of Privilege Vulnerability	2020 2021	FIN7	Russia
				DragonFly	Russia
				Wizard Spider	Russia
				menuPass	China
11	2021-26855	Microsoft Exchange Server Remote Code Execution Vulnerability	2021	HAFNIUM	China
				Threat Group-3390	China
	2021-26857	Microsoft Exchange Server Remote Code Execution Vulnerability	2021	HAFNIUM	China
				Threat Group-3390	China
	2021-26858	Microsoft Exchange Server Remote Code Execution Vulnerability	2021	HAFNIUM	China
				Threat Group-3390	China
	2021-27065	Windows Container Execution Agent Elevation of Privilege Vulnerability	2021	HAFNIUM	China
				Threat Group-3390	China
	2021-26411	Internet Explorer Memory Corruption Vulnerability	–	APT37	North Korea
13	2017-11774	Microsoft Outlook Security Feature Bypass Vulnerability	–	APT 33	Iran
				APT 34	Iran
14	2020-1040	Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability	–	Sandworm Team	Russia
				Kimsuky	North Korea
				APT 28	Russia
				ATP 33	Iran
16	2017-12617	Apache Tomcat JSP Code Injection and Remote Execution Vulnerability	–	Sea Turtle	Iran
17	2017-8759	.NET Framework Remote Code Execution Vulnerability	–	APT-C-01	China
				BlackOasis	Middle East
				Cobalt Group	ND
				Leviathan	China
20	2018-0798	Microsoft Office Memory Corruption Vulnerability	–	Tonto Team	China
				Higaisa	South Korea
				BRONZE BUTLER	China
				Threat Group-3390	China
	2018-0802	Microsoft Office Memory Corruption Vulnerability	–	APT 37	North Korea
				Tonto Team	China
				Confucius	India
				Tropic Trooper	China
				Inception	Russia
				BRONZE BUTLER	China
25	2013-0640	Adobe Reader and Acrobat via a Crafted PDF Document	–	DarkUniverse	ND

Finally, the long tail vulnerability list has been cross-referenced with commonly available reports of APT actors' activity [4], [12], [18], [19], [29] and CISA Top Routinely Exploited Vulnerabilities [7], [8]. The matching is presented in Table 3. It is worth noting that the first 6 CVEs listed in Table 3, are included in the Top Routinely Exploited Vulnerabilities list prepared by CISA. This shows that common, known, unpatched vulnerabilities have not only been actively exploited by APTs, but are directly listed by security organizations as frequently exploited. In other words, organizations are not patching known vulnerabilities, despite widely available warnings of their active exploitation.

Interesting relationships become apparent when the data in Tables 2 and 3 are combined. As many as 13 of the 14 vulnerabilities listed in Table 3 do not require authentication. In addition, 7 of the 14 CVEs have a base score exceeding 9, meaning that they allow all three information security attributes (confidentiality, integrity, availability) to be completely compromised with at least a medium attack complexity. CVEs with a mean base score were also used by APT groups, e.g., CVE-2021-26857, CVE-2021-26858, CVE-2021-27065, CVE-2021-26411, CVE-2017-11774, and CVE-2017-12617. The aforementioned CVEs are characterized by a network attack vector, lack of authentication, and partial violation of confidentiality, integrity, and availability. Thus, even CVEs having only a medium base score can pose a threat.

5 Discussion and Conclusions

The presented analysis shows several widespread vulnerabilities that are a few years old and still run unpatched, despite a fix from the vendor being available. After initial hype and interest, even the most prominent vulnerabilities, such as Log4j (CVE-2021-44228) or ProxyShell (CVE-2021-34473, CVE-2021-34523, and CVE-2021-31207), fade, get forgotten, while vulnerable systems invite attackers to take advantage of unpatched security holes.

The general findings from this work are as follows: (1) based on raw real-world data, a non-negligible percentage of active servers is still affected by long-tail vulnerabilities despite the availability of the patches; (2) these identified long-tail vulnerabilities have a historical record of exploitation by foreign APTs that are described in cybersecurity professional sources. Therefore, (3) there is a significant risk that APTs will take advantage of their expertise and materialize the vulnerabilities into attacks on systems, data or users whenever it becomes beneficial for them. Particularly, APT Groups commonly exploit CVEs that do not require authentication and are accessible remotely from the network. There is one exemption from that rule, namely CVE-2020-1040, which requires access from an adjacent network and authentication with a single factor.

There are several reasons for the long tail of unpatched vulnerabilities. Corporate inertia plays a role. For some organizations, the process of introducing a software patch, especially in a server infrastructure or OT installations is a time consuming and complex endeavor. The reasons are frequently procedural or even compliance related. In this case it is often a "stability over security" imbalance. Some organizations and individuals use software (or hardware-software) solutions that have reached the end of life (EoL), thus they are not supported by the vendor anymore (e.g., Microsoft Windows XP). EoL software continues to function, but does not receive any regular security updates, hence quickly becomes permanently vulnerable. Some vendors may not be aware that their product embeds third-party modules that are vulnerable, especially when it comes to open-source libraries, like Log4j [14]. Without a specialized software updating product, some organizations, especially small ones, may lack visibility of critical vulnerabilities in their environment and lack tools for monitoring the patching progress. Lastly, introducing a patch, unless done fully automatically, introduces additional work, usually for IT staff. It may be forgotten, or other tasks may take priority.

Patching should become a fully automated process, without introducing unnecessary delays for human input (with some exceptions, such as critical infrastructure). The immediate nature of the patching process should correspond with the rapid exploitation of published vulnerabilities. Currently, within hours from releasing the information, cyber criminals start scanning for exposed targets. Protection must catch up with that pace. Nation states, on the other hand, need to be assumed to have knowledge of vulnerabilities months if not years ahead of the general public (zero-days) and ability to exploit them even before the vendor is aware of the bug and able to release a patch.

Governments and policymakers could consider introducing regulations that would mandate vendors to implement automatic patching mechanisms for all internet connected devices. There could be a time span within which critical vulnerabilities in vendor software or any third-party modules employed would have to be patched. Another step could be that the automatic patching mechanism could be impossible to disable for the user. Recent plans [10] made by the EU (Cyber Resilience Act) are a significant step in that direction toward embedded device manufacturers, among other things, requiring them to keep publishing security patches for a period of either five years, or the product expected lifetime, as well as to disclose incident within twenty-four hours of becoming aware of it.

Future work includes a more detailed vulnerability analysis:

- number of vulnerabilities on particular assets;
- coexistence of vulnerabilities on particular assets;
- type and severity of vulnerabilities occurring on particular assets;
- estimation of APT probability and impact.

References

[1] Allodi, L., Massacci, F., & Williams, J. (2021). The work-averse cyberattacker model: Theory and evidence from two million attack signatures. Risk Analysis, 42(8), 1623–1642. https://doi.org/10.1111/risa.13732

[2] Bilge, L., & Dumitras, T. (2012). Before we knew it: An Empirical Study of Zero-Day Attacks In The Real World. Proceedings of the 2012 ACM Conference on Computer and Communications Security – CCS'12. https://doi.org/10.1145/2382196.2382284

[3] Blaise, A., Bouet, M., Conan, V., & Secci, S. (2020). Detection of zero-day attacks: An unsupervised port-based approach. Computer Networks, 180. https://doi.org/10.1016/j.comnet.2020.107391

[4] Bussoletti, F. (2019). Is Iran's apt34 behind the Sea Turtle Cyber Espionage Campaign? Difesa e Sicurezza, https://www.difesaesicurezza.com/en/cyber-en/is-irans-apt34-behind-the-sea-turtle-cyber-espionage-campaign/
[5] Chen, P., Desmet, L., & Huygens, C. (2014). A study on Advanced persistent threats. Advanced Information Systems Engineering, 63–72. https://doi.org/10.1007/978-3-662-44885-4_5
[6] CISA (2022), Known Exploited Vulnerabilities Catalog, https://www.cisa.gov/known-exploited-vulnerabilities-catalog
[7] CISA. (2021). Alert (AA21-209A) Top Routinely Exploited Vulnerabilities. CISA. Retrieved from https://www.cisa.gov/uscert/ncas/alerts/aa21-209a
[8] CISA. (2022). Alert (AA22-117A) 2021 Top Routinely Exploited Vulnerabilities. CISA. Retrieved from https://www.cisa.gov/uscert/ncas/alerts/aa22-117a
[9] Cole, E. (2013). The changing threat. Advanced Persistent Threat, 3–26. https://doi.org/10.1016/b978-1-59-749949-1.00001-2
[10] Dobberstein, L. (2022). EU puts manufacturers on Hook for Smart Device Security. The Register – Biting the hand that feeds IT, https://www.theregister.com/2022/09/16/eu_cyber_resilience_act/
[11] Flashpoint. (2022). Russian apt and ransomware groups: Vulnerabilities and threat actors who exploit them, https://flashpoint.io/blog/vulnerabilities-exploited-by-russian-apts-ransomware-groups/
[12] Fraunhofer-Institut für Kommunikation, Informationsverarbeitung und Ergonomie FKIE. (2022). malpedia. Malpedia, https://malpedia.caad.fkie.fraunhofer.de/
[13] Kaspersky Lab (2022). Targeted attack on industrial enterprises and public institutions, https://ics-cert.kaspersky.com/publications/reports/2022/08/08/targeted-attack-on-industrial-enterprises-and-public-institutions/
[14] Knapczyk, P. (2022). Overview of the cyber weapons used in the Ukraine – russia war. Trustwave, https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/overview-of-the-cyber-weapons-used-in-the-ukraine-russia-war/
[15] Kotzias, P., Bilge, L., Vervier, P.-A., & Caballero, J. (2019). Mind your own business: A longitudinal study of threats and vulnerabilities in enterprises. Proceedings 2019 Network and Distributed System Security Symposium. https://doi.org/10.14722/ndss.2019.23522
[16] Kumar, V., & Sinha, D. (2021). A robust intelligent zero-day cyber-attack detection technique. Complex & Intelligent Systems, 7(5), 2211–2234. https://doi.org/10.1007/s40747-021-00396-9
[17] Lemay, A., Calvet, J., Menet, F., & Fernandez, J. M. (2018). Survey of publicly available reports on Advanced Persistent Threat Actors. Computers & Security, 72, 26–59. https://doi.org/10.1016/j.cose.2017.08.005
[18] Madiant (2022), Advanced Persistent Threats (APTs), https://www.mandiant.com/resources/insights/apt-groups
[19] Mitre (2022), Groups, https://attack.mitre.org/groups/
[20] MITRE. (2022). CVE. Retrieved from https://cve.mitre.org/
[21] Nist (2012). NIST Special Publication 800-30 Revision 1, Guide for Conducting Risk Assessments. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf
[22] Nist (2012). NIST Special Publication 800-39 Managing Information Security Risk. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-39.pdf
[23] NIST. (2022). National Vulnerability Database. NVD, https://nvd.nist.gov/

[24] Sarabi, A., Zhu, Z., Xiao, C., Liu, M., & Dumitraş, T. (2017). Patch Me if you can: A study on the effects of individual user behavior on the end-host Vulnerability State. International Conference on Passive and Active Network Measurement, 113–125. https://doi.org/10.1007/978-3-319-54328-4_9
[25] Security Magazine (2021), Average time to fix severe vulnerabilities is 256 days, https://www.securitymagazine.com/articles/95929-average-time-to-fix-severe-vulnerabilities-is-256-days
[26] Singh, A.P. (2017). A study on Zero Day malware attack. International Journal of Advanced Research in Computer and Communication Engineering, 6(1), 391–392. https://doi.org/10.17148/ijarcce.2017.6179
[27] Singh, U.K., Joshi, C., & Kanellopoulos, D. (2019). A framework for Zero-day vulnerabilities detection and prioritization. Journal of Information Security and Applications, 46, 164–172. https://doi.org/10.1016/j.jisa.2019.03.011
[28] Singh, U.K., Joshi, C., & Singh, S.K. (2017). Zero day attacks defense technique for protecting system against unknown vulnerabilities. International Journal of Scientific Research in Computer Science and Engineering, 5(1), 13–18. https://isroset.org/pub_paper/IJSRCSE/3-IJSRCSE-VC000412.pdf
[29] Stirparo, P., Bizeul, D., Bell, B., Chang, Z., Esler, J., Bleich, K., ... & Egloff, F. (2019). APT Groups and Operations, https://apt.threattracking.com
[30] Toulas, B. (2022). Microsoft: Iranian hackers still exploiting Log4j Bugs against Israel. BleepingComputer, https://www.bleepingcomputer.com/news/security/microsoft-iranian-hackers-still-exploiting-log4j-bugs-against-israel/
[31] Weedon J. (2015) Beyond ‘Cyber War’: Russia’s use of strategic cyber espionage and information operations in Ukraine. In: Cyber war in perspective: Russian aggression against Ukraine. Talinn: NATO CCD COE Publications. p. 67–77. https://ccdcoe.org/uploads/2018/10/Ch08_CyberWarinPerspective_Weedon.pdf
[32] WithSecure (2022), WithSecure Elements Vulnerability Management, https://www.withsecure.com/us-en/solutions/software-and-services/elements-vulnerability-management
[33] ZDNet (2021), Average time to fix critical cybersecurity vulnerabilities is 205 days: report, https://www.zdnet.com/article/average-time-to-fix-critical-cybersecurity-vulnerabilities-is-205-days-report/

Modeling the Security of Critical Infrastructure Objects Using a Drone System

Viktoriia Klym [1] [0000-0002-5887-1955], Yuri Tarasenko [1] [0000-0002-4226-5707], and Volodymyr Katan [2] [0000-0001-9167-2619]

[1] University of Customs and Finance, Volodymyr Vernadskyi str. 2/4, 49000 Dnipro, Ukraine
[2] Oles Honchar Dnipro National University, Gagarin av. 72, 49010 Dnipro,Ukraine
v0123klim@gmail.com

Abstract. Often the object of critical infrastructure has a very large area, then it is more human resources are involved. The protecting a large area with traditional methods requires the involvement of a large number of guards. In the paper a model of a multi-system of drone of access control and protection of a critical infrastructure object (CIO) that occupies a large area is presented. A proposed model consists of the physical and mathematical space-time model, in which the description of the actual components of the system, their dynamic characteristics and functioning parameters is presented. The model focuses on a multi-level approach. With this approach, the large territory is controlled by multi-system of drones that move in horizontal planes at different heights simultaneously. The new formulation is called the multiple drone-access control problem. This control structure is especially important for potentially vulnerable areas in critical infrastructure. The paper proposes to use a greedy algorithm to solve a large class of CIO security problems. An analysis of the proposed model and the directions for the development of the work are discussed.

Keywords: access control, vulnerable areas, greedy algorithm, coverage problem

1 Introduction

Recently aspects of the security of national critical infrastructure have taken the first positions in the state strategic programs for the development of countries. Intensive promotion and assistance are received by projects on the organization and implementation of the protection of critical infrastructure at the interethnic and interstate level. For example The European Programme for Critical Infrastructure Protection (EPCIP) is a framework under which various measures together aim to improve the protection of critical infrastructure in the EU. Also these measures include the establishment of the European Reference Network for Critical Infrastructure Protection (ERNCIP) [1].

The events of 2022 taking place in Ukraine are proof of the relevance of the tasks of creating and improving methods and tools aimed at avoiding specific cognitive, legal and technical problems with the security and protection of critical infrastructure objects (CIO) [2].

The CIO are characterized by their social, economic, industrial, defense, environmental significance for ensuring the safe life of the population and the country as a whole. At the same time the performance of vital functions and services by CIO is associated with the existence of threats, the possibility of crisis situations due to unauthorized interference in their activities. The sense of protecting and ensuring the safety of CIO is the impossibility of harming the regular functioning and properties of these objects or their structural components. At the same time, the spectrum of information security aspects in the form of availability, integrity and confidentiality of information completely depends on the progress of the technologies used, their social component and the probabilistic nature of the possible external action at the research object [2].

The main tasks of complex modeling of the process of protection and safety of objects are the following [3]:

- ensuring the protection of CIO, through the functioning of an effective system of physical security, security of operating systems and cyber security;
- performing a risk assessment at CIO and exchanging information about risks and threats with other subjects of the national critical infrastructure protection system;
- operational response to illegal actions, physical attacks aimed at disabling or damaging the operation of operating systems or systems for ensuring the physical security of a critical infrastructure object.

In accordance with the declared tasks the model of the process of protection and security of critical infrastructure objects has a segmental structure, which includes subsystems for physical security and cybersecurity, a control and communication system, and the presence of a decision support subsystem. Also it is additionally possible to introduce a recognition subsystem: cybercrime, cyberespionage, cyberterrorism and other cyber incidents [2]. It shows that the safety of the CIO depends on multidisciplinary aspects of operation, including the uncertainty of the environment [4]. It is the probabilistic nature of the external action that prevents the choice of the optimal variant for the protection of CIO. Therefore, for the task of complex protection and security of CIO, the potential possibility of rapid change over time of the main input data, taking into account the random values of the external influence, determines the features of the formulation and the solution method.

The use of drones has wide practical applications in various spheres of human activity. This is due to the small size and mass, as well as due to the high mobility and the advantage of moving through airspace at low altitudes. With the help of drones, a large class of applied problems is solved, in which it is necessary to conduct panoramic and dynamic video shooting, to transport compact and light loads [5, 6]. At the moment, the tasks of protecting objects are the most relevant class of tasks performed by drones [7].

Often the object of critical infrastructure has a very large area. If the industrial, logistics or agricultural complex have the large area, then more human resources are involved. The protecting a large area with traditional methods requires the involvement of a large number of guards. Such a protection system is quite effective, but very expensive. Not every organization, government institute, enterprise is ready to bear such costs.

Protecting the territory of the object with the help of drones allows the enterprise to significantly reduce costs without worsening the level of its security. Instead of human patrols over the territory of the protected object, a drone is launched, which is difficult to see during the day and almost impossible to detect at night. the drone constantly records video. Modern video cameras have sufficient resolution to record the illegal movement of even one person in the protected area from a height of 200–300 meters. Modern drones are able to maintain reliable communication from 5 to 10 km, depending on the specific model.

The paper considers a mathematical model of a system of access control and protection of a critical infrastructure object that occupies a large area, in which the movement of drones moves automatically, the system control is based on the execution of a greedy algorithm. Section 2 provides an overview of papers that show current issues of cybersecurity, the main possibilities of using drones in the field of security, the use of a greedy algorithm in mathematical models of various applied problems. Section 3 describes the formulation of the problem for the model of a system of access control and protection of a critical infrastructure object, in which the description of the actual components of the system, their dynamic characteristics and functioning parameters, interactions, description of the main conditions and relationships for the greedy algorithm are presented. The analysis of the proposed mathematical model and the directions for the development of the work are discussed in Section 4.

2 Literature review

The use of drones in ensuring cybersecurity in terms of general approaches, methods, types of probable attacks is considered in the works [8], [9]: the research [8] is devoted to complex analysis of the drones security vulnerabilities and the attack life cycle, which conducts a comprehensive review of the different aspects of drones' cybersecurity including two main aspects: drones' security vulnerabilities, and the security concerns associated with compromised drones; the paper [9] addresses a multicriteria analysis based on the drone assessment to support public security, structures the preferences and exposes the set of the most favorable solution in the light of quantitative and qualitative data, upon inter-criterion and intra-criterion analysis, uses the known web platform for modeling, and exploring the results by numerical and graphical analysis.

The research [10] demonstrates the benefits of a web-based cyber-security decision support tool for healthcare, where it is finding an optimal security control set for health organizations. A prototype personal protective equipment (PPE) access monitoring system which combines smart PPE and an indoor/outdoor personnel location monitoring system is proposed in [11].

Greedy algorithm is used for the optimization of problems decide in different applied tasks. Greedy algorithm is used by class of delivery problems. The initial problem [12] which is called the Maximum Coverage Facility Location Problem with Drones is connected with serving the demands and decides with greedy algorithm, formulates the objective of maximizing coverage while explicitly incorporating the drone energy consumption and range constraints. In the paper [13], a heuristic named the maximum coverage greedy randomized heuristic (MCGRH) is developed, that improves selecting the locations of drone launching centers, which maximizes patient service coverage within certain drone range constraints. Such that in the research [14] it decides the some delivery problem with the symbiosis among a truck and multiple drones in a last-mile package delivery scenario. It is called the Multiple Drone-Delivery Scheduling Problem.

The paper [15] describing an applied system of drones considers a multi-drone enabled data collection system for smart cities, where there are two kinds of drones, and proposes an efficient and novel search algorithm named Drones Traveling Algorithm (DTA) to obtain a near-optimal solution.

The rapid development of the drone industry is expanding the field of cybersecurity research. At the same time, two main trends of research are distinguished: tasks where drones ensure the cybersecurity of an object and tasks where drones themselves are an object of cybersecurity.

The problem of the attacks on the critical infrastructure network is considered in the paper [16], such that a correlation analysis approach based on the greedy algorithm, which optimizes event analysis steps and significantly improves the performance is realized, it is also the real-time correlation analysis technology for security events is discussed.

Multi-robot system [17] is implemented through decentralized control algorithms, where the decision-making and navigation processes are localized to sensor-equipped drones. The quantitatively compare algorithms for such decentralized systems through simulation is described.

This research [18] is devoted to detection and identification of type and flight mode of single and multiple drones in radio frequency domain applying supervised deep learning algorithms, particularly fully-connected deep neural network models that use raw drone signals rather than features.

It is new generic certificate based access control scheme to provide inter-drone is designed [19] which provides the drones' security and privacy, especially provably secures against the known attacks and provides anonymity.

3 Proposed Model

We have considered a scenario in which a system of drones is used for organizing of access control, protection and safety on an object of critical infrastructure with large area. For simulation of access controls and protection system we redefine the simulation of a multi-agent system with the greedy algorithm, which is commonly used for to decide the coverage problem [13], [14], [16], [17].

3.1 Physical and mathematical space-time model. Multi system consists of *M* drones d_1, d_2,…, d_m with the same capabilities. The drones are characterized by decision-making agent with a control radius *R* and a movement radius ρ.

Trajectory Model. Let *S* is 2-D area of critical infrastructure represents our access control, protection scenario. We consider a discrete map *S* of an area that needs to be protected (covered). This map is an *p* by *q* matrix of cells. Each cell has pair of coordinates $(x_s, y_s) \in X \times Y$, where $X = \{1,\dots, p\}$ $Y = \{1,\dots, q\}$. The coordinates of each cell are located in the center of the square or the rectangle.

Such covered area comprises facility location (drone launching center) positions $F = \{1,\dots, k\}$ for start or recharge of drones within coordinates $(x_f, y_f) \in S$.

The position of drone *d* is defined with $(x_d, y_d) \in S$ also. It illustrates the coordinates of cell of map S (x_s, y_s), the position of drone (x_d, y_d) and the coordinates of some facility location (see Fig. 1).

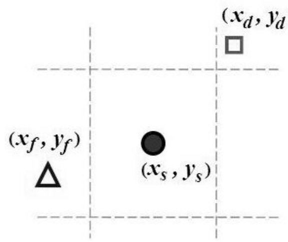


Fig. 1. It illustrates the coordinates of cell of map S with drone and facility location

Let $d_i(t) = (x_{di}, y_{di})$ represent the position of drone *i* at time-step *t*. We note that a starting position $d_i(0)$ is determined with position of some facility location when $t = 0$. When drone chooses its next position, it depends of several conditions terms.

We assume that the movement of the drone in the vertical component is negligible compared to the distance to the facility location. Ideal case is that the drone moves along a horizontal plane. In this case, the physical model does not take into account the effects of wind, raindrops or temperature. Additionally, we note the absence of any obstacles when moving the drone along a horizontal plane.

Also the problem assumes that more than one facility location is used for the drone system, and they are located symmetrically and uniformly across the area.

If the position of the drone or station is on the boundary line between the cells or at its corner points, then the algorithm then imposes additional conditions for the next stage.

Let l_i (t) be the distance from drones position $d_i(t)$ at facility location f_j. Than $\rho = l_i(t) - l_i(t\text{-}1)$ be distance between two positions of drone: at time-step *t* and *t*-1 (see Fig. 2).

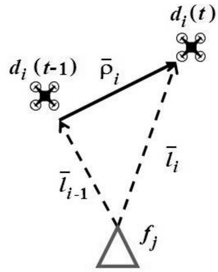


Fig. 2. It illustrates the two positions of drone: at time-step t-1and t relative to the facility location f_j

Each drone d_i is moving from position $d_i(t\text{-}1)$ to next position $d_i(t)$ according to conditions $|l_i\ (t)| \le$ r and $d_i(t) \notin d_{\text{-}i}(t\text{-}1)$, where $d_{\text{-}i}(t\text{-}1)$ represent the positions of every drone except for i.

For each time-step the coverage of map S is defined by each cell, for which condition is performing $\rho_c \le R$, where ρ_c is distance from cell (x_s, y_s) to position of drone $d_i(t)$. Let $C(d_i)$ be the coverage set of cells for each drone.

Local area coverage model. For every action of drone $d_i\ (t)$ the utility function U_i is defined as follows:

$$U_i(d_i, d_{\text{-}i}) = \Sigma_{(x,\, y)\, \in\, C(d_i)}\, S_{x,y} \quad (1)$$

Hence, in order to satisfy the total coverage of map S, the utility function U_t is defined as follows, which is the sum of the values of every cell in S covered by every drone:

$$U_t(d) = \Sigma_{(x,\, y)\, \in\, C(d)}\, S_{x,y} \quad (2)$$

The formulas (1) and (2) shows that global utility of coverage system consists of sum with utilities of every cell that have control radius and is not covered by another drone.

To obtain the norm of the utility function, let compose the following expression:

$$U^*_t(d) = U_t(d)/\, \Sigma_{(x,\, y)\, \in\, \mathrm{X} \times \mathrm{Y}}\, S_{x,y} \quad (3)$$

In the ideal case when the map is covered with all cell $\mathrm{U}^*_t(d) = 1$.

The overall objective is to maximize the total value of the covered cells, utility $U_t(d)$, while minimizing the time it takes for this value to converge.

Proposed multi-level approach. In order to improve the efficiency of the access control and management system at a critical infrastructure facility using a multi-system of drones, we propose a multi-level approach. With this approach, the large territory is controlled by multi-system of drones at several levels simultaneously.

Scenario 1. Let level 1 consist of M1 drones $d_1, d_2, \ldots, d_{m1}$ that move at height $h1$ with the control radius $R1$. Then the area S is discretely divided into cells. A map S1 is an $p1$ by $q1$ matrix of cells. Each cell has pair of coordinates $(x_{s1}, y_{s1}) \in X1 \times Y1$, where $X1=\{1,\ldots, p1\}$, $Y1 = \{1,\ldots, q1\}$.

Similarly, at height $h2$, the system has a second flight level M2 of drones $d_1, d_2, \ldots d_{m2}$, that move with the control radius $R2$. At this level the area S is discretely divided into cells with map S2. A map S2 is an $p2$ by $q2$ matrix of cells. Each cell has pair of coordinates $(x_{s2}, y_{s2}) \in X2 \times Y2$, where $X2 = \{1,\ldots, p2\}$, $Y2 = \{1,\ldots, q2\}$.

If a multi-system of drones have N levels for any level we have a similar describing of initial value of characteristics.

Such a multilevel system is characterized by the relations:

– for heights $h1< h2<..hN$,

– for control radius $R1< R2<..RN$,

– for size of matrix S, such as $p1 > p2>.. pN$ and $q1 > q2>..qN$.

Note that the ratios presented above will be correct in the case when all drones of the multisystem have the same characteristics. Then, at the lowest flight level $h1$of the drones, the matrix S1 will have the largest number of cells $p1 \times q1$, and at the level with the maximum flight altitude hN, the matrix will have the smallest number of cells $pN \times qN$ (see Fig. 3).

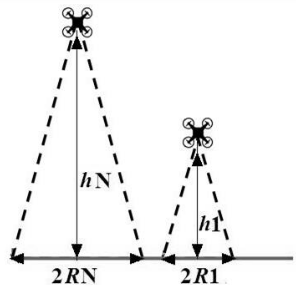


Fig. 3. The proposed a multi-level approach

Scenario 2. The total area S is discretely divided into several subarea of number N, such that $S1\cup S2\cup S3\ldots\cup SN = S$. Let level 1 consists of M1 drones $d_1, d_2,\ldots, d_{m1}$ that move at height $h1$ and monitor the own subarea S1 with the control radius $R1$. Then the subarea S1 is discretely divided into cells. A map S1 is an $p1$ by $q1$ matrix of cells. Each cell has pair of coordinates $(x_{s1}, y_{s1}) \in X1 \times Y1$, where $X1 = \{1,\ldots, p1\}$, $Y1 = \{1,\ldots, q1\}$. Note that the size of the cells should be chosen in accordance with the control radius $R1$.

Similarly, a height $h2$ is the second flight level M2 of drones $d_1,d_2,\ldots d_{m2}$, which have the control radius $R2$. At this level the subarea S2 is discretely divided into cells with map S2. A map S2 is an $p2$ by $q2$ matrix of cells. Each cell has pair of coordinates $(x_{s2}, y_{s2}) \in X2 \times Y2$, where $X2 = \{1,\ldots, p2\}$, $Y2=\{1,\ldots, q2\}$.

In this scenario it assumed the same relations for heights $h1< h2<..hN$ and for control radius $R1< R2<..RN$. Each level of drone system monitors its own subarea of total area S. The main difference from the previous scenario is the division of the total area S into subareas. Each subarea is characterized by a cell size. This scenario ensures the division of the total area into subarea and their coverage with cells of different sizes in accordance with their control radius.

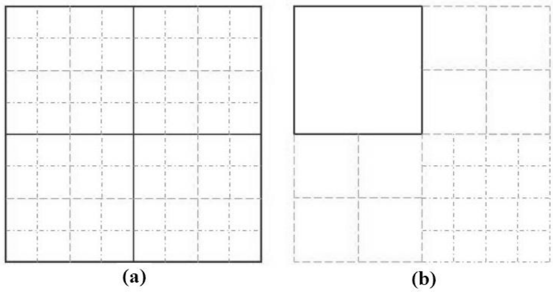


Fig. 4. Example of area coverage according to scenario 1 (a) and scenario 2 (b)

Total area coverage model. In order to describe a multi-level approach the utility function $U^1{}_t$ at height $h1$ is defined as follows:

$$U^1{}_t = \sum_{(x,\, y)\, \in\, C^1(d)} S_{x,y} \tag{5}$$

According to proposal model for the height hN the utility function $U^N{}_t$ is obtained as follows:

$$U^N{}_t = \sum_{(x,\, y)\, \in\, C^N(d)} S_{x,y} \tag{6}$$

Then our model determines total utility function $U^{total}{}_t$:

$$U^{total}{}_t = \sum U^i{}_t \tag{7}$$

Finally the norm of total utility function is given as:

$$\mathrm{U}^*{}_t(d) = U^{total}{}_t \,/ \sum_{(x,\, y)\, \in\, \mathrm{X} \times \mathrm{Y}} S_{x,y} \tag{8}$$

3.2 Problem definition. The new formulation is called multiple drone-access control problem. Let S be the matrix ($p \times q$) of controlling area's map with cells $S_1, S_2, .. S_{p \times q}$, m the number of drones, and k number of facilities location. The objective of this problem is to find a best action of drone $d_i(t)$ for covering the maximum number cells of matrix S consist at value $C(d_i)$, such that $(x_s, y_s) \in C(d_i)$ and $C(d_i) \cap C(d_j) = 0$ for i≠j, at each time-step t, while minimizing the total time it can be achieved.

3.3 Greedy algorithm. The greedy algorithm has the following steps: (i) determining the position of the drone with the maximum number of controlled matrix cells: determining the local utility function, (ii) sum and combining the total number of controlled matrix cells: determining the coverage of the matrix, (iii) definition of a general utility function and its normalization.

In accordance with the principle of the greedy algorithm at each time-step t, any drone $d_i\,(t)$ choses its position as follows:

$$d_i\,(t+1) = \arg\max\, U_i(d_i, d_{-i}) \tag{9}$$

It is important that the greedy algorithm will always choose to make a move that maximizes its utility, and terminates when $d_i(t) = d_i(t+1)$. Because of these characteristics, the greedy algorithm will get stuck in local optima, because it only takes an action that immediately benefits the utility function [16, 17].

Its known that the greedy algorithm has the not good performance regardless of starting initial values due to looking for the local optima. However, the greedy algorithm has one consistent advantage: the number of time-steps that it takes for to stabilize of process, which is defined as the point after which value of utility function stays within 10% of the final utility [17]. That's why the starting position of each drone in multiple system will more homogeneously distribute on the territory. In this case the drones are much more likely to encounter local optima that by definition they are unable to cross, because the greedy algorithm only makes moves that provide immediate benefit. This fact exposes the NP-hardness of that problem, even for the simple case with several drones. Therefore it is no absolute time solution available for this problem as the problem is a known NP-hard problem. The greedy algorithm provides a Log-n approximate algorithm [12], [13], [17], [18].

A modification of the algorithm with a distinctive feature characteristic only for the tasks of ensuring the security of the open territory of the CIO covering a large area is proposed.

Vulnerability zones are determined on the map S of the territory of the CIO depending on the degree of risk of unauthorized ground penetration. For example, the perimeter of the territory always has the highest degree of vulnerability; as you approach the center, the degree of vulnerability decreases. When the matrix S is divided into cells, each cell is assigned a coefficient (indicator) of vulnerability. Next, neighboring cells of the matrix with the same coefficient of vulnerability, forming continuous arrays, are combined into one cell of a complex configuration. It is rational to create such composite cells in areas with the lowest coefficients of vulnerability. Moreover, unification is possible until the cell exceeds the certain maximum linear size.

Thus, the number of cells in the matrix S is significantly reduced, which in turn leads to a faster search for the local optima solution for covering the cells of the matrix.

4 Results and discussion

The proposed model shows the principle of operation of the access control and management system at a critical infrastructure facility that occupies a large area with the multiple drone system, the control of which is realized through the use of a greedy algorithm. Such system has a number of operational advantages: the implementation of control does not depend on the human factor, the minimum of the human resource involved in the control of most of the critical infrastructure object.

Due to the problem belongs to optimization problems in time and space, the process of finding the optimal solution requires a large amount of computation, which does not always have good convergence. The use of the greedy algorithm in the proposed model implies obtaining a solution for the maximum coverage in the case of a uniform location of the points of the initial movement of the drones. In addition, the presence in the system of several levels (heights) of the flight of drones helps to divide the task into several: at the level that controls the smallest number of cells, the convergence of the solution will be the greatest.

When covering the area according to scenario 2 for subareas, it is expected a situation in which $SI \cap SJ \neq 0$, $I \neq J$. In this case the collision of drones is eliminated due to the multi-level approach. Also, when covering the area S according to any of the proposed scenarios, the number of ”blind zones” decreases due to the intersection of sections and simultaneous control at different levels of the system.

In order to increase the efficiency of the proposed model, it is recommended to use a matrix with a high density of areas map for vulnerabilities.

The presented model has prospects for its development in further work on the following tasks: placement of a facility location at each given flight level of drones, tracking battery charge and weight of each drone, determination of the optimal number of facility locations, priority control of the most vulnerable places of the controlled area.

In order to overcome the probabilistic nature of external influence it seems possible and necessary in this problem to use the theory of random sets to assess the most dangerous direction and time interval of external influence such as unauthorized access. The use of game theory would improve the reliability of such a control and access control system at a critical infrastructure facility that occupies a large area.

5 Conclusions

The proposed multi-level model of a multi-drone system performs the functions of continuous video observing of the open territory of the CIO which covering a large area with located buildings, also in case of inaccessibility of some areas of the territory due to the complex location of ground structures, their partial or complete destruction or potential explosion hazard (for example, gas leakage or mining).

The proposed mathematical model of the access control and management system shows the possibility of using a greedy algorithm to implement effective access control and protection of a critical infrastructure facility that occupies a large area. Thus, the paper shows the use of a greedy algorithm for a wide class of applied cybersecurity problems using drone or drone systems. The some special addition in the algorithm only for the tasks of ensuring the security of the open territory of the CIO covering a large area is considered.

For the first time, a multi-level drone system has been proposed, in which drones move in horizontal planes at different heights. This control structure is especially important for potentially vulnerable areas in critical infrastructure. Location of the facility location at a height corresponding to the flight level of the drones (for example, on the roof of a ground structure) will reduce interruptions in the operating mode of each of them. In addition, facility locations powered by solar energy solve the problem of autonomous power supply for the drone system as a whole. At the same time, the simultaneous crossing of control zones, on the one hand, increases the reliability of the information received, on the other hand, reduces the number of "blind zones". Also, the proposed multi-level approach reduces the risk of drone collisions, but improves the control of vulnerabilities due to the possibility of covering them with a matrix with the smallest cell size.

Besides some drones (or levels of drones) can perform the protective function of the CIO if they have a useful mass. At the same time, the vulnerability from hitting drones as air targets from the ground decreases with increasing flight level.

The use of such a model in simulation will allow to choose a scenario with the optimal number of drones in the system, depending on the area of the territory, to determine the number and location of vulnerabilities, defense lines, to work out the necessary accuracy in identifying external influences by setting several levels of drone flight.

References

[1] Critical infrastructure protection. EU Science Hub, https://joint-research-centre.ec.europa.eu/scientific-activities-z/critical-infrastructure-protection_en, last accesed 2022/08/27.

[2] Tarasenko, Yu., Klym, V.: The methodology of building the cognitive model of critical infrastructure's security. European Science, 1(sge11-01), 38–50 (2022). https://doi.org/10.30890/2709-2313.2022-11-01-009.

[3] Verkhovna Rada of Ukraine (2017). Zakon Ukrainy № 1882-IX «Pro krytychnu infrastrukturu». Holos Ukrainy – Voice of Ukraine, 236 (2021).

[4] Barchenko, N., Obodiak, V.: Systema standartiv kompleksnykh system zakhystu informatsii ta upravlinnia informatsiinoiu bezpekoiu. In monograph: Suchasni informatsiini tekhnolohii v kiberbezpetsi, pp. 176–193. Sumy State University (2021).

[5] UgCS Commander: Manage multiple UAVs, https://www.ugcs.com/page/ugcs-commander-manage-multiple-uavs, last accesed 2022/08/29.

[6] Norasma, C., Fadzilah, M.A., Roslin N., Zanariah , Z.W.N., Tarmidi, Z., Candra FS.: Unmanned Aerial Vehicle Applications In Agriculture. 1st South Aceh International Conference on Engineering and Technology. IOP Conf. Series: Materials Science and Engineering 506, 012063 IOP Publishing. (2019). doi:10.1088/1757-899X/506/1/012063.

[7] Povitriani viiny. Bezpilotni litalni aparaty zakhopyly nebo, https://texty.org.ua/projects/107398/povitryani-vijny-bezpilotni-litalni-aparaty-zahopyly-nebo/, last accesed 2022/08/29.

[8] Yaacoub, J.-P., Noura, H., Salman, O. et al.: Security analysis of drones systems: Attacks, limitations, and recommendations.Internet of Things 11, 100218 (2020). https://doi.org/10.1016/j.iot.2020.100218.

[9] Moreira, L., Angelo M., et al.: Evaluation of drones for public security: a multicriteria approach by the PROMETHEE-SAPEVO-M1 systematic. Procedia Computer Science 199, 125–133. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021). (2022). https://doi.org/ 10.1016/j.procs.2022.01.016.

[10] Ozdemir Sonmez, F., Hankin, C., Malacaria, P.: Decision support for healthcare cyber security. Computers & Security 122, 102865 (2022). https://doi.org/10.1016/j.cose.2022.102865.

[11] Hayward, St., Katherine van Lopik, West, A.: A holistic approach to health and safety monitoring: Framework and technology perspective. Internet of Things 20, 100606 (2022). https://doi.org/10.1016/j.iot.2022.100606
[12] Chauhan, D., Unnikrishnan, A., Figliozzi, M.: Maximum coverage capacitated facility location problem with range constrained drones. Transportation Research Part C 99, 1–18 (2019). https://doi.org/10.1016/j.trc.2018.12.001.
[13] Al-Rabiaah, S.; Hosny, M., AlMuhaideb, S.: An Efficient Greedy Randomized Heuristic for the Maximum Coverage Facility Location Problem with Drones in Healthcare. Appl. Sci. 12, 1403 (2022). https://doi.org/10.3390/app12031403 .
[14] Sorbelli, F., Coro, F., Das, S., Palazzetti, L., Pinotti C.: Greedy Algorithms for Scheduling Package Delivery with Multiple Drones. New Delhi, India Conference Paper, September 2021. ICDCN 2022, January 4–7 (2022). DOI: 10.1145/3491003.3491028.
[15] Du, Yao, Wangy, K., Yangzx, K., Zhang G.: Trajectory Design of Laser-Powered Multi-Drone Enabled Data Collection System for Smart Cities. 2019 IEEE Global Communications Conference (GLOBECOM), 09–13 December 2019. Waikoloa, HI, USA (2019). DOI: 10.1109/GLOBECOM38437.2019.9013552.
[16] Lu, P., Hu, T., Wang, H., Zhang, R., Wu, G.: G-CAS: Greedy Algorithm-Based Security Event Correlation System for Critical Infrastructure. Network Security and Communication Networks, vol. 2021, article ID 3566360, 13 pages (2021). https://doi.org/10.1155/2021/3566360.
[17] Shah, A., Ferguson, B., Marden, J.: Improving the Efficiency and Scalability of Multi-Drone Coverage Systems with Decentralized Control. University of California, Santa Barbara, Research Mentorship Program (UCSB RMP), (2019). https://web.ece.ucsb.edu/~blf/publications/RMP/Shah2019.pdf.
[18] Sazdić-Jotić, B., Pokrajac, I., Bajčetić, J., Bondžulić, B., Obradović, D.: Single and multiple drones detection and identification using RF based deep learning algorithm. Expert Systems with Applications, vol. 187, 115928, (2022). https://doi.org/10.1016/j.eswa.2021.115928.
[19] Chaudhry, SA, Yahya, K,, Karuppiah, M., Kharel, R., Bashir, AK., and Zikria, YB.: GCACS-IoD: A certificate based generic access control scheme for Internet of drones. Computer Networks, 191. (2021) DOI: https://doi.org/10.1016/j.comnet.2021.107999.

Cattle Identification System Based on Dermatoglyphs

Iryna Borodkina[1, 2] [0000-0003-3667-3728], and Heorhii Borodkin[2] [0000-0002-6488-6512]

[1] Kyiv National University of Culture and Arts,
Yevhen Konovalets Street, 36, Kyiv, 01601, Ukraine
borir@ukr.net

[2] National University of Life and Environmental Sciences of Ukraine,
Heroiv Oborony Str.15, building 3, Kyiv, 03041, Ukraine
george.borodkin@gmail.com

Abstract. The article considers the process of designing an information system for accounting and identification of cattle based on dermatoglyphs. Current issues of creation and use of such information systems in farms are identified. The tasks to be solved by the information system of accounting and identification of cattle based on dermatoglyphs are defined. The general concept of building such an information system is considered. The methods, approaches and algorithms that should be used when processing dermatoglyphs in the system are defined. The main components of the system are highlighted. Requirements for the functionality of the system have been defined. They ensure the effective use of all available sources of information. The results of system modeling by the means of UML and the several results of system testing are given.

Keywords: cattle identification, pattern recognition, cattle accounting, nasolabial mirror, dermatoglyph

1 Introduction

To date, it has been confirmed [1] that despite the large number of farms, their place in the agricultural economy of Ukraine still remains insignificant. In 2017, they produced only 8.7% of agricultural products (at constant 2010 prices) of all categories of farms and 15.5% – gross output of agricultural enterprises. The share of gross livestock production in 1990–2017 is low and generally does not exceed 2.0% in value terms and 3.5% in kind [1]. Therefore, information support of business activities of these farms is an important task, the solution of which will significantly improve the infrastructure of agribusiness. One of the problems of farmers is the registration of livestock, monitoring of diet, health, vaccination and other zootechnical measures, sexual maturity, number of livestock and other parameters. The current state of the level of informatization allows to automate these processes by developing and implementing appropriate information and management systems. The creation of such a system will automate the above processes, which will save time, energy and increase the productivity of farmers.

The aim of the study is to create an information management system for accounting and identification of cattle by dermatoglyphs, which will speed up and simplify the process of accounting, viewing and editing data on cattle. This can not only significantly improve the work of farms involved in animal husbandry, but also create a single regional database of cattle. The latter will help to increase control over the breeding and movement of cows in the region and the registration of animals in the event of their delivery to meat plants.

2 Cattle identification

2.1 Methods of cattle identification

Identification is a system of animal registration, which includes assigning an individual number to an animal in different ways by marking [2]. The existing system of identification and registration of cattle is based on the Directive of the European Parliament and the Council of the EU No. 1760/2000 [3], which establishes a system of identification and registration of cattle.

After the analysis of methods of accounting and identification of cattle [4], we identified the following main methods:

- double ear tags with individual code,
- farm registration,
- passport for cattle,
- computerized database for individual animals.

Today the following methods of identification are widespread:

- Tattoos (made with special forceps with sharp protrusions; numbers are placed on the inner surface of the ear). [5]
- Ear plucking [10], [27] (a classic method of marking that has long been used in practice. It provides good visibility of data, operational reliability of labels, their preservation, because the puncture is made through the entire thickness of the ear).
- Hot and cold branding [4], [6] (a procedure that changes the color of an animal's fur from black to white and thus provides an effective distinguishing mark. In cases where the animal has a light complexion, branding completely freezes hair follicles, leaving a “bald” brand).
- Marking: metal earrings, synthetic tags [9] (marking is carried out by attaching a tag to the ear of the animal, placing it in the middle in the inner part of the ear, if possible, closer to its base; this method is the most common and accessible).
- Collars, leg and tail bracelets [8], [9] (modern method of identification and control of animals, which is suitable for modern complexes; collar or bracelet is equipped with special sensors).
- Chipping [10] (modern method of marking animals with special chips, easy to use, painless and safe; electronic identification of animals is performed by implanting a subcutaneous microchip containing a unique individual fifteen-digit number that remains throughout the life of the animal).
- Impressions of the nasolabial mirror [14] (dermatoglyph).

2.2 Selection of the criterion for cattle identification

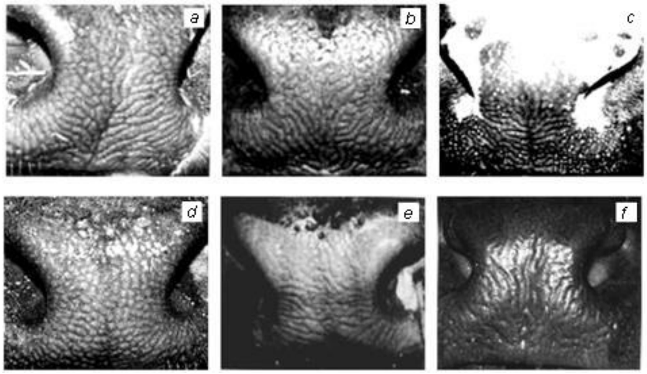


Fig. 1. Examples of the main types of cattle dermatoglyph patterns

To identify cattle in the developed information system, the method of nasolabial mirror imaging was chosen [7], [15]. This method is simple, humane, bloodless and non-labor-intensive [17]. In addition to the identification of livestock, this method also can be used to establish the species of the animal and to identify certain anatomical and physiological features of the animal. The main types of patterns of the cow nasolabial mirror are shown in Fig.1 (a – “ear”, b – “crown”, c – “branch”, d – “grain”, e,f – “combi”) [19, 21].

3 Applying of pattern recognition technologies for cattle identification

The basis of animal identification in the developed system is dermatoglyphics – a method of identifying individual species of animals based on the features of skin patterns. [11] In humans, the skin of the palmar side of the hand has a complex pattern, which is formed by combs, so this skin is called comb. The uniqueness and immutability of this pattern allowed William Herschel in the late 19th century to put forward the idea of human identification by fingerprints, which formed the basis for the fingerprinting development [13].

Dermatoglyphic pattern of the dog nasal mirror, its qualitative and quantitative indicators are strictly individual and characterize each individual like human fingerprints [6], [13]. In Canada, the United States, Korea and China, several studies of papillary patterns of the nasal mirror of service dogs (German Shepherd, Labrador, Retriever) have been conducted by computer programs, which, however, have not gained much popularity and widespread [12]. They still remain at the level of scientific research in forensic veterinary examination.

This method of identification has interested farmers and cattle owners, because it differs from branding and chipping by non-invasiveness, safety and practical lack of material costs. Studies have shown that for cows, the papillary patterns of the nasal mirror are also a characteristic and unique feature of each individual.

Thus, the use of technologies for establishing the identity of an animal's nasolabial mirror imprint with a dermatoglyph image stored in a system database is the main painless technological approach for identifying cattle [16]. The comparison procedure can be performed for all stored in the database dermatoglyphs, both sequentially and parallel-sequentially. If an identical pair of images is found with the specified confidence, the procedure ends [20]. To compare images in the system the generally accepted methods of the theory of image recognition are used.

The cattle dermatoglyph image's processing requires performing the following actions: image noise reduction in, image border analysis, image binarization, creation of image element contours, search for certain elements and obtained image characteristic's calculation.

One or more actions are sequentially performed with each image that is: applying filters for smoothing, highlighting borders, changing color, adjusting brightness, contrast, etc. The result of such processing is an dermatoglyph image with special points highlighted on it. To improve the quality of the image and reduce the level of noise in the system, a Gabor filter is used [31], [32].

The Sobel operator, which is a discrete differential operator that calculates the approximate value of the gradient or the norm of the gradient for the brightness of the image, is used to highlight the boundaries in the system. The Sobel operator is based on convolution of the image with small separable integer filters in the vertical and horizontal directions [33], [34].

In addition to the Sobel operator, Kenny's algorithm [35] can be used to select image boundaries in the system. It show good results and provide the following criteria are met: a significant signal/noise ratio, accurate definition of the boundary position, one response per boundary.

Kenny's algorithm consists of the following stages:

1. Smoothing. At this stage, the image is blurred to remove noise using a Gaussian filter.

2. Search for gradients. Boundaries are defined where the gradient of the image reaches its maximum value. Gradients can have different directions, so the Kenny algorithm uses four filters to detect horizontal, vertical and diagonal edges in a blurred image.

3. Filtering of non-maximums. Only local maxima are considered as boundaries.

4. Double threshold filtering. Potential limits are determined by thresholds.

5. Tracing the area of ambiguity. Boundaries are determined by suppressing all edges that are not connected to certain (strong) boundaries. Before applying the detector, usually the images are converted to grayscale to reduce computational costs.

The boundary detection algorithm is not limited to calculating the gradient of the smoothed image. Only the maximum points of the image gradient remain in the contour of the border, and the non-maximum points lying next to the border are removed. Information about the direction of the border is also taken into account here in order to remove point right next to the border and not to break the border itself near the local maximums of the gradient. Then, with the help of two thresholds, weak boundaries are removed. At the same time, the border fragment is processed as a whole. If the value of the gradient somewhere on the studied fragment exceeds the upper threshold, then this fragment is also considered an "acceptable" boundary in those places where the value of the gradient is below this threshold, until it becomes below the lower threshold. If there is no point in the entire fragment with a value greater than the upper threshold value, then it is removed. This approach makes it possible to reduce the number of gaps in the original borders.

Image binarization is an image processing method that converts a color or grayscale image into a monochrome image, that is, an image that uses only two types of pixels (black and white).

To perform threshold binarization of halftone images, the system uses based on the Otsu method algorithm [36].

The next stage of identification is image skeletonization. A skeleton in computer graphics is a set of points that are equidistant from the borders of the figure. In fact, the skeleton is a representation of the shape of the figure. It simplifies its further analysis. The skeletonization procedure has the following general requirements: selected lines must have a thickness of one element (pixel), the lines and nodes of the input image must correspond to the lines and nodes of the processed image, and the shape of the object composed of lines must not be strongly distorted.

Using the skeletonization process allows you to create easy-to-implement and effective selection algorithms, as well as to thin the lines in the image. Image skeletonization is a very important component of biometric systems that rely on the comparison of different biometric characteristics. As algorithms for skeletonization of a binary image, a wave algorithm, an algorithm for thinning regions or a template method can be used. Choosing an algorithm that is optimal in a certain sense is one of the areas of further research.

4 Design of the recognition system

4.1 Analysis of functional and non-functional system requirements

The proposed identification system can be used not only to identify a specific animal – it is only one of the functions of the system [24], [25]. In fact, the process of accounting for the condition of livestock throughout life is no less important. This may include the dynamics of changes in weight and milk yield of cows, the date of vaccination, the date of mating, the date of calving, the pedigree of each animal and so on.

To determine the functional requirements for the system, a comparative analysis of currently existing software for accounting for cattle was conducted. "Cattle accounting program" [24] offers the following functionality: accounting for growth and milk yield for each animal with the display of data in graphical and tabular form, statistics of slaughter and death, automatic construction of pedigrees for each animal, accounting for the reproduction of each animal (calving , coverage), its diseases and vaccinations [30]. The system has built-in directories of diseases and veterinary drugs. Vaccination records contain data with the date of vaccination, dosage of the drug and the name of veterinarians who performed the vaccination [23]. The system provides the possibility of grouping by territorial division of places for keeping animals, keeps records of farm staff, determines, if necessary, those responsible for each animal, provides delimitation of access rights. Identification of animals in the system is through the use of ear tags with an individual code. There is no mobile version, no internet access required.

Another system is the cattle accounting program "Farm + 1F + 5.0" [27]. It differs from the previous analogue by having information about the manufacturer of the vaccine and the ability to import all information into standard Excel files. Identification in this system is due to the earlobe with an individual code. The disadvantages of the system include the lack of a mobile version and the ability to work with it exclusively offline.

Analysis of analogues showed that the method of identification in these systems is an ear tag with an individual code, and the process of putting it on is painful and inhumane [18], [22]. It is desirable to replace the method of identification with one that does not cause pain to cattle, namely the identification of the image of the nasolabial mirror. In contrast to this system, the system considered in the article should work on mobile devices, which significantly expands the possibilities for users and provides the ability to quickly update data on animals.

4.2 Description of the image adaptation process for system identification

The system consists of desktop software and a database. The software is responsible for identifying users, viewing user data, downloading, scanning, and viewing images, and connecting to the database.

The work of the program begins with the user registration or identification, after which you can either view your data (regular user), or the data of all users (administrator). It is possible to download at testing image. Than the software proceeds to process this image and check for matches with images that are already in the database.

The adaptation process consists of four stages.

The first stage – the image is processed to a black and white state, this is done by checking the brightness of an individual pixel, after which its color changes to black, white or gray.

The second stage is skeletalization of the image, i.e. noise removal [26], [28]. This stage has two substages, in both cases a pixel matrix of three by three pixels is processed, with the central pixel always being the main one. This matrix is compared to a set of template matrices. Each of the two sub-steps has its own set of templates. Skeleton lines are built on the first substage. Lines must be one pixel thick. The lines and nodes of the original image must correspond to the lines and nodes on the transformed image. The shape of the object, which is composed of lines, should not be too repetitive. In the second sub-step, after the pattern has been found, the gray pixels are removed. Deleting is repeated until all such pixels are deleted.

The third stage is the stage at which points, lines and nodes are distinguished [29]. A neighborhood of eight points is processed, if only one point is black, it is an endpoint; if there are two, it is a line point, and if there are three, it is a branch.

The fourth stage is the search and comparison of points. The points are selected on both images. Then by the method of their correlation comparison on the neighborhood of thirty by thirty pixels is checked to see if they match.

4.3 System design

The UML language was used to describe the design of the cattle registration and identification system. With its help the basic design diagrams were constructed.

Use case diagram. The use case diagram describes the subject area related to cattle accounting (Fig. 2). This diagram identifies 2 actors who, within their functional responsibilities, take an active part in the operation of the system. The administrator has the ability to use administration tools to create and edit databases. These databases will store all data about users, their livestock, farms, and so on. The administrator has access absolutely to all DB data. He can view, edit and delete them if necessary. The administrator provides access to the database to regular users, is responsible for database recovery, data integrity and logging.

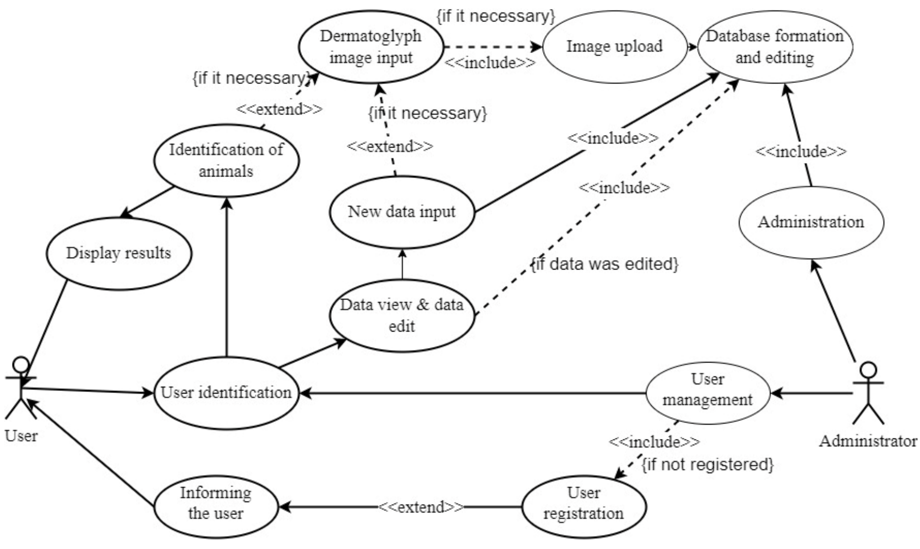


Fig. 2. Use case diagram of the cattle identification system

The user must register in the system. In the future, to log in, he must be identified by his login and password (for today), or by any other designed means of authentication. Registered users can work with the database, and they have access only to information that relates to data about his cattle. If the user first accesses the database, he must first download the data about his cattle, including images of their dermatoglyphs. The user may not store all the information about his livestock if for some reason he does not need it.

Class diagram. The class diagram is based on the subject area of the system and describes all classes used by the system to cattle account and their based on dermatoglyph identification.

The user either registers or authenticates (enters his login and password). After identification, the user is given the opportunity to search for data by identifier or by dermatoglyph image. The user can upload a dermatoglyph image and then initialize the match search process: the match will display animal data, if no match is found, the user is asked to add a dermatoglyph image and a description of the animal to the database. Adding a new image is optional.

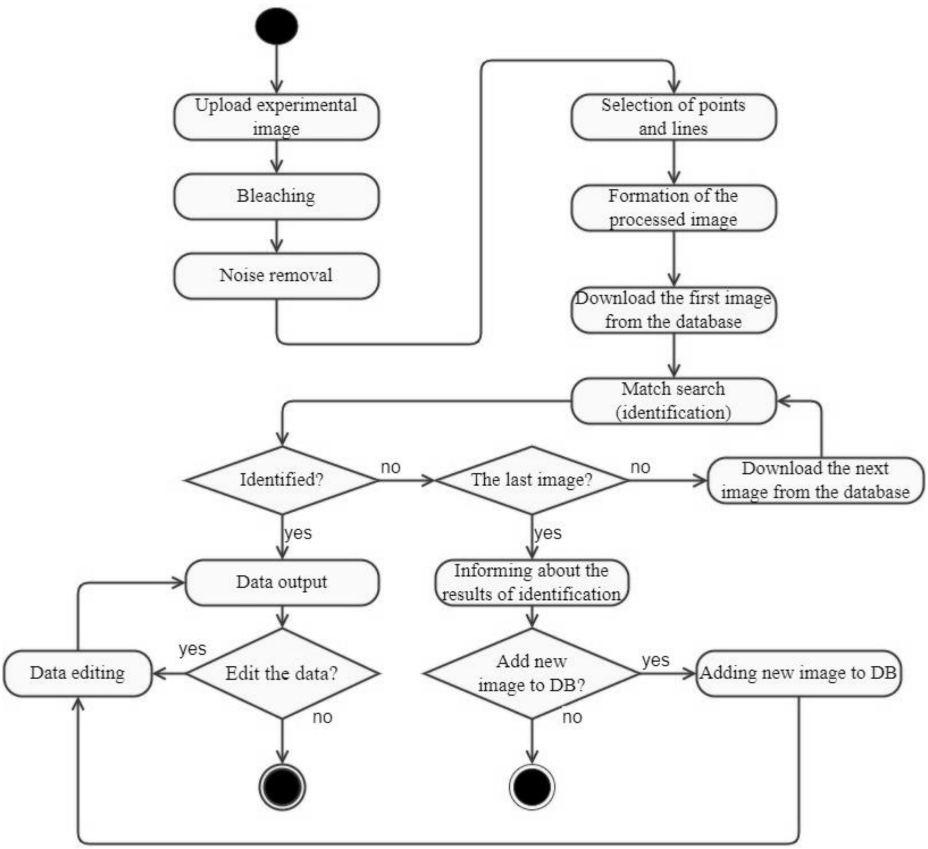


Fig. 3. Activity diagram of the identification process

Activity diagram. The activity diagram (Fig. 3) allows you to specify the behavior of the system in the form of coordinated sequential and parallel execution of subordinate nested activities or individual actions depending on the processes implemented by the system.

The matching process begins with the user uploading a dermatoglyph image. Then this image is discolored, i.e. simplified to three colors (black, white and gray). After that, all gray pixels are removed, then lines and dots on the experimental image are selected. After second step the images are loaded from the database one by one, and each of them is compared with the experimental image. This continues until matches are found or the images in the database run out. If no matches are found, the user can either add a new image to the database or quit. If there is a match, a result is displayed that can be either edited or not edited and finished.

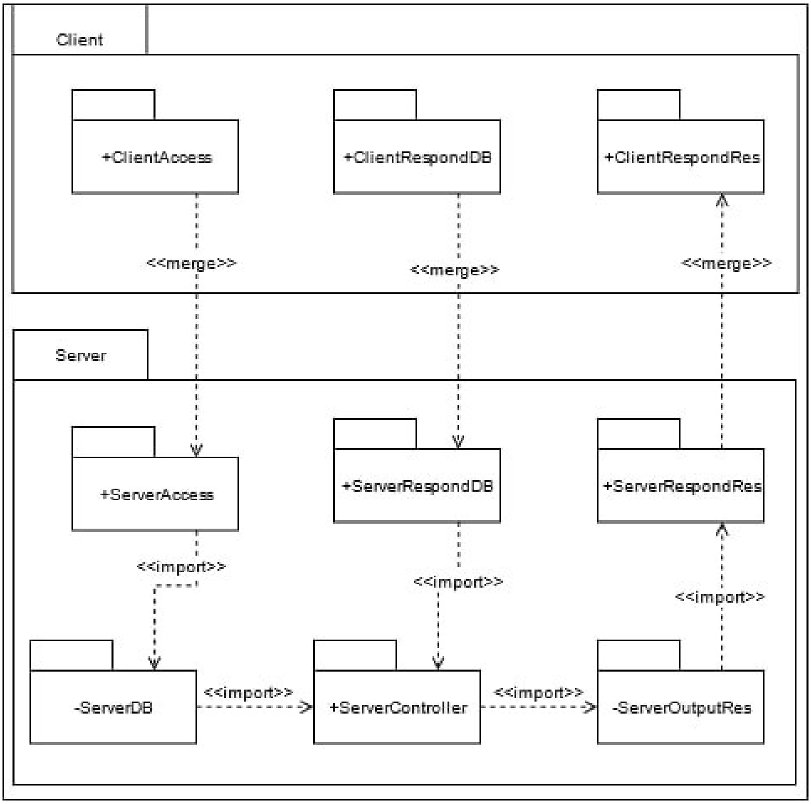


Fig. 4. Package diagram of the identification system

Package diagram. Fig. 4 shows the package diagram of the designed system, which composes the whole system in the form of two installation applications: Client & Server. Each application corresponds to one of two packages: Client or Server.

Client is a dependent package and Server is a basic package. There are three packages in the Client package:

– ClientAccess allows a user to go through the authentication process,
– ClientRespondDB is responsible for communicating with the database,
– ClientRespondRes is responsible for outputting the result.

The Server package includes such packages as:
– ServerAccess provides user access to the database,
– ServerDB implements interaction with the database,
– ServerController responsible for data processing,
– ServerRespondDB provides communication between the backend and the database,
– ServerOutputRes responsible for sending the work of the backend to the client part.

System database project. The logical model of the system database is developed using ER-diagram technology. The main tables in the database are the Owner table and the Cow table. The “Owner” table stores all the information about the user, and the “Cow” table stores all the information about an individual animal. The Cow table is linked to the CowDetails table, which contains all the details about the animal.

5 System implementation

The user interface was developed in the PyCharm development environment, in the Python programming language. The system backend was developed in the Python programming language, in the PyCharm development environment. The main reason for choosing Python is access to the PIL library (Python Imaging Library), which provides extensive support for graphic file formats, efficient internal representation and quite powerful image processing capabilities. The main image library is designed for quick access to data stored in several basic pixel formats.

The database was implemented on a MySQL 2014 server, and PhpMyAdmin connected via an XAMPP web server was used as a navigator.

The desktop software and database are responsible for the backend. The software registers and identifies the user, after successful identification the user will be able to proceed to the accounting and viewing of data, after which he will be able to add new data, including images of dermatoglyphs. The user can either simply add an image of the new dermatoglyph or download and scan the image and use it to identify the animal. The database stores detailed information about both users and livestock.

6 System testing

The program was tested on a 64-bit Windows 10. The test was to test the system’s ability to identify the image and relate it to that already in the database. To test the identification process, a repository of graphic images was created, shown in Fig. 5.

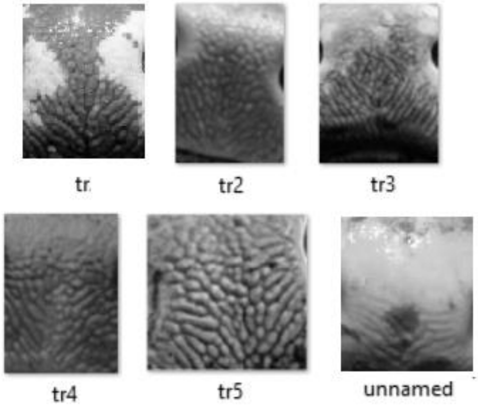
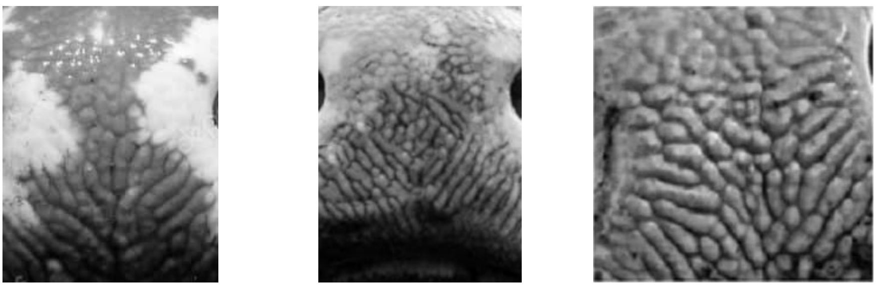


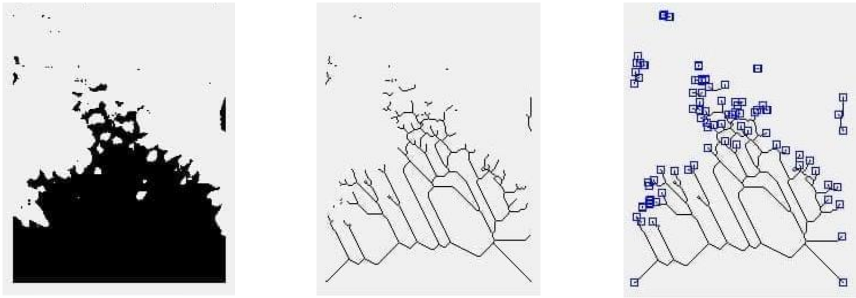
Fig. 5. Graphic object repository for system testing

To obtain a positive test result, the software must find a dermatoglyph image and all information about the animal associated with it in the database, or report the absence of the specified dermatoglyph in the database. Thus, the task of testing is to find the dermatoglyph in the database system. To do this, an image of the animal's dermatoglyphis selected from the image repository (note that the information about the animal has already been entered into the database). The software must identify the animal by dermatoglyph and display its description in the program's working window. Three different images of dermatoglyphs, shown in Figure 6, were used for testing. This choice of test images is explained by the need to test the program's ability to identify objects of different types of patterns of the nasolabial mirror of animals. The identification process consists of three main stages – discoloration, noise removal and selection of special points. The results of step-by-step identification for each dermatoglyph are shown in Fig. 7–9).



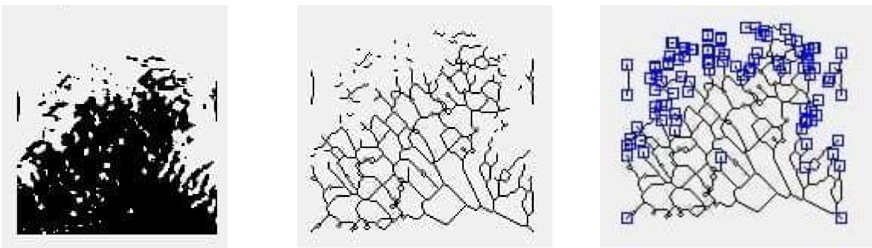
Dermatoglyph1 Dermatoglyph2 Dermatoglyph3

Fig. 6. Images of dermatoglyphs for system testing



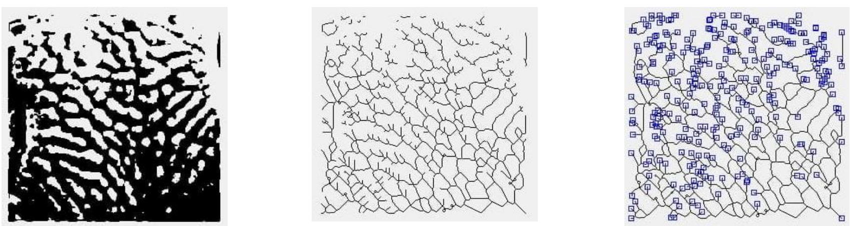
Step 1. Discoloration | Step 2. Noise removal | Step 3. Selection of special points

Fig. 7. Stages of image processing of dermatoglyph 1 to identify the animal



Step 1. Discoloration | Step 2. Noise removal | Step 3. Selection of special points

Fig. 8. Stages of image processing of dermatoglyph2 to identify the animal



Step 1. Discoloration | Step 2. Discoloration | Step 3. Selection of special points

Fig. 9. Stages of image processing of dermatoglyph3 to identify the animal

In all three cases, dermatoglyphs were identified and information about the animal was displayed on the user's screen from the database.

7 Conclusion

The paper considers the problem of building an information system for accounting and identification of cattle by dermatoglyphs. In the aggregate analysis of the data of quantitative and qualitative characteristics of dermatoglyphs of the nasal mirror of cattle and the presence of its graphic image, the possible identification of the animal can be carried out with a high degree of probability. From an economic and ethical point of view, the replacement of chipping and branding by the definition of dermatoglyphs of the nasal mirror is very important. The disadvantage of this procedure can only be a physiological impossibility to perform it – significant damage to the nasal mirror. Modern information technologies and experience of dactyloscopic examination allow to create a computer system designed for accounting and identification of cattle using dermatoglyphs.

Prospects for further research are to develop software and algorithmic methods that will improve the time characteristics of the system, to study the dependence of speed and accuracy (error) of identification on the sample size and the number of features.
In the future, special attention should be paid to the possibilities of modern methods of identification of cattle using simplified methods of image input based on paper technology, identification of dermatoglyphs based on special schemes, and some other approaches.

References

[1] Groshev S.V. The current state of development of farms in Ukraine, Ukrainian Journal of Applied Economics. Volume 3(3), 186–194 (2018). http://ujae.wunu.edu.ua/index.php/ujae/article/view/303, last accessed 2022/09/21
[2] Legal requirements for the identification and registration of cattle, pigs, sheep and goats in the European Union, http://www.icar.org/wp-content/uploads/2015/09/Sprenger_RU.pdf, last accessed 2022/09/20.
[3] Regulation (EC) No 1760/2000 of the European Parliament and of the Council of 17 July 2000 establishing a system for the identification and registration of bovine animals and regarding the labelling of beef and beef products and repealing Council Regulation (EC) No 820/97, https://eur-lex.europa.eu/eli/reg/2000/1760/2021-04-21, last accessed 2022/09/20.
[4] Bello R.W., Olubummo D.A, Seiyaboh Z., Enuma O.C., Talib A. Z., Mohamed A.S.A.. Cattle identification: the history of nose prints approach in brief. Proceedings of the 6th International Conference on Agricultural and Biological Sciences (ABS 2020). IOP Conf. Series: Earth and Environmental Science 594 (2020). DOI:10.1088/1755-1315/594/1/012026
[5] Bello R.W., Talib A,Z., Mohamed A.S.A., Olubummo D.A., Otobo F.N. Image-based Individual Cow Recognition using Body Patterns. International Journal of Advanced Computer Science and Applications, 11(3), 92–98 (2020). DOI:10.14569/ijacsa.2020.0110311
[6] Mougeot G., Li D., Jia S.A deep learning approach for dog face verification and recognition. In: Proc. Pacific Rim Int. Conf. Artif. Intell. pp. 418–430. Springer (2019).
[7] Kumar S., Singh S.K., Singh R.S., Singh A.K., Tiwari S. Real-Time Recognition of Cattle using Animal Biometrics. Journal of Real-Time Image Processing 13(3) 2017, pp. 505–526. DOI:10.1007/s11554-016-0645-4

[8] Giot R., El-Abed M., Rosenberger C. Fast computation of the performance evaluation of biometric systems: Application to multibiometrics. Future Generation Computer Systems 29(3), 788–799 (2013). DOI: 10.1016/j.future.2012.02.003

[9] Awad A.I. From classical methods to animal biometrics. Computers and Electronics in Agriculture, 123(C), 423–435 (2016). DOI:10.1016/j.compag.2016.03.014

[10] Lu Y., He X., Wen Y., Wang P.S.P. A new cow identification system based on iris analysis and recognition. International Journal of Biometrics, 6(1), 18-32 (2014). DOI:10.1504/IJBM.2014.059639

[11] Kumar S., Singh S.K. Cattle Recognition: A New Frontier in Visual Animal Biometrics Research. Proceedings of the National Academy of Sciences, India, Physical Sciences, Section A, 90(7), 689–708 (2019). DOI:10.1007/s40010-019-00610-x

[12] Bello R.W., Abubakar S., Development of a Software Package for Cattle Identification in Nigeria. Journal of Applied Sciences and Environmental Management 23(10), 1825–1828 (2019). DOI:10.4314/jasem.v23i10.9.

[13] Sarfraz N.. Adermatoglyphia: Barriers to Biometric Identification and the Need for a Standardized Alternative Cureus. 11(2): e4040, 2019. DOI:10.7759/cureus.4040

[14] Sathya J.L., Kumar S. Beef Cattle Identification Using PCA and Other Cattle Identification Methods. International Journal of Engineering Sciences & Research Technology, 6(11), 177–185 (2017). DOI:10.5281/zenodo.1042110

[15] Kumar S., Singh S.K., Singh A.K. Muzzle point pattern based techniques for individual cattle identification. IET Image Processing. 11(10), 805–814 (2017). DOI:10.1049/iet-ipr.2016.0799

[16] Qiao Y., Kong H., Clark C., Lomax S., Suc D., Eiffert S., Sukkarieh S. Intelligent perception for cattle monitoring: A review for cattle identification, body condition score evaluation, and weight estimation. Computers and Electronics in Agriculture, 185(6), 106–143 (2021). DOI:10.1016/j.compag.2021.106143

[17] Kusakunniran W., Wiratsudakul A., Chuachan U., Kanchanapreechakorn S., Imaromkul T., Suksriupatham N., Thongkanchorn K. Biometric for Cattle Identification using Muzzle Patterns. International Journal of Pattern Recognition and Artificial Intelligence 34(12), 2020. DOI:10.1142/S0218001420560078

[18] Volnistyj G.E. Belarusian information system for identification and traceability of animals and products of animal origin. Systems approach. 2016. http://www.eurasiancommission.org/ru/act/dmi/workgroup/Documents/Question_3.2(presentation).pdf

[19] Bae H.B., Pak D., Lee S. Dog Nose-Print Identification Using Deep Neural Networks. IEEE Access, 9, 49141–49153 (2021). DOI:10.1109/ACCESS.2021.3068517

[20] Mahmoud H.A., El Hadad R.H.M. Automatic cattle muzzle print classification system using multiclass support vector machine. International Journal of Image Mining, 1(1), 126–140 (2015). DOI:10.1504/IJIM.2015.070022

[21] Zin T.T., Phyo C.N., Tin P., Hama H., Kobayashi I. Image technology based cow identification system using deep learning. In: Proceedings of the International Multi-Conference of Engineers and Computer Scientists Vol. 1, pp. 320-323 (2018). Hong Kong. http://www.iaeng.org/publication/IMECS2018/IMECS2018_pp320-323.pdf, last accessed 2022/09/25.

[22] Noviyanto A., Arymurthy A.M.. Beef cattle identification based on muzzle pattern using a matching refinement technique in the sift method. Computers and Electronics in Agriculture, 99, 77–84 (2013). DOI:10.1016/j.compag.2013.09.002

[23] Bello R.W., Abubakar S. Development of a Software Package for Cattle Identification in Nigeria. Journal of Applied Sciences and Environmental Management, 23(10), 1825–1828 (2019). DOI:10.4314/jasem.v23i10.9.

[24] Cattle census program 5.1. Version 5.1. http://www.softwr.ru/prg/programma_ucheta_krupnogo_rogatogo_skota/, last accessed 2021/11/20.

[25] Cattle census program (Farm + 1F + 5.0). URL: https://agroserver.ru/b/programma-ucheta-krupnogo-rogatogo-skota-ferma-1f-5-0-309828.htm, last accessed 2021/11/15.

[26] Nemchenko T.A., Lupyan E.A., Zakharov M.Yu., Krasheninnikova Yu.S., Fleitman E.V. Using the "skeletonization" procedure to highlight lines in satellite images. http://smiswww.iki.rssi.ru/files/publications/loupian/earth_res_1994_06.pdf, last accessed 2021/11/20.

[27] Lu Y., He X., Wen Y., Wang P.S. A new cow identification system based on iris analysis and recognition. International Journal of Biometrics, 6(1), 18–32 (2014). DOI:10.1504/IJBM.2014.059639

[28] Applying the Wave Algorithm to Find the Skeleton of a Bitmap. http://ocrai.narod.ru/vectory.html, last accessed 2021/10/25.

[29] Minagawa H., Fujimura T., Ichiyanagi M., Tanaka K., Fangquan M. Identification of beef cattle by analyzing images of their muzzle patterns lifted on paper. In: Proceedings of the 3rd Asian Conference for Information Technology in Asian agricultural information technology & management, 8, pp. 596–600. Publications of the Japanese Society of Agricultural Informatics, (2002).

[30] Awad A.I., H.M. Zawbaa, H.A. Mahmoud, E.H.H.A. Nabi, R.H. Fayed,A.E. Hassanien. A Robust Cattle Identification Scheme Using Muzzle Print Images. Proceedings of the 2013 Federated Conference on Computer Science and Information Systems, pp. 529–534 (2013).

[31] Palm Ch., Lehmann Th.M. Classification of color textures by gabor filtering. Machine GRAPHICS & VISION 11(2/3), pp. 195–219 (2002).

[32] Kroon D. Numerical optimization of kernel based image derivatives. http://www.k-zone.nl/Kroon_DerivativePaper.pdf/, last accessed 2019/04/25

[33] Asif Fuad A., S.M. Rizvi. Hardware Software Co-Simulation of Canny Edge Detection Algorithm. International Journal of Computer Applications 122 (19), pp. 7–12, (2015).

[34] Jain A.K., Nandakumar K., Ross A. 50 years of biometric research: Accomplishments, challenges and opportunities. Pattern Recognition Letters, 79, pp. 80–105 (2016). DOI: 10.1016/j.patrec.2015.12.013

[35] Vala H.J., Baxi A. A Review on Otsu image segmentation algorithm. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) 2(2): 387. (2013).

Software Engineering

Implementation of Procedure for the Identification of Dynamic Systems Based on Neural Networks

Olena Kryvoruchko[1] [0000-0002-7661-9227], Yuliia Kostiuk[2] [0000-0001-5423-0985], Tetyana Savchenko[3] [0000-0002-8884-5360], and Dmytro Hnatchenko[4] [0000-0002-6584-4525]

State University of Trade and Economics, Kyiv, Ukraine
[1]ev_kryvoruchko@ukr.net, [2]sv_t@ukr.net,
[3]kostyuk.yu@ukr.net, [4]dmitriy.gnatchenko@gmail.com

Abstract. One of the promising directions in the field of complex dynamic objects management is the application of intelligent control systems based on artificial neural networks. The main advantage of these control systems is the use of such properties of neural networks as the ability to approximate arbitrary nonlinear dependencies, the ability to learn, potentially high resistance to interference and fault tolerance. Artificial neural networks are an effective means of solving complex problems that are poorly formalized. This class traditionally includes the tasks of classification, clustering, multidimensional images approximation, time series forecasting, nonlinear filtering, identification, as well as complex technological objects management. Currently, the methods of neural network technology are actively used for the identification and management of nonlinear dynamic objects in real time. For the effective use of artificial neural networks in order to solve the problems of identifying nonlinear dynamic objects and designing automatic control systems, it is necessary to solve the problem of optimizing the parameters and structure of the neural network.

Keywords: identification, optimization, dynamic system, neural network, mathematical model, intelligent system, regressor, predictive model

1 Introduction

In connection with today's new challenges to the requirements in management processes in various fields of technology, the problem of building mathematical models of dynamic objects becomes extremely important. Both theoretical and experimental methods can be used to build mathematical models. The experience accumulated during the control systems design convincingly shows that it is not possible to build an adequate model to the real system taking into account only the theoretical study of physical processes in the system [1], [4], [8]. The methods of dynamic systems mathematical models building based on the data of observations of their behavior are the subject of the theory of identification. From the point of view of the theory of identification, a system is an object in which the interaction between different types of variables takes place and the observed signals are formed. Dynamic systems are systems in which the current value of the output signal depends not only on the current, but also on the previous values of external influences [1], [2], [5].

Numerical tables and graphs are used to describe the properties of some systems. Such descriptions are called graphic models. For example, linear systems may be represented only by their impulse responses, unit step responses, or frequency responses. Appropriate graphical representations are widely used in various design tasks [1], [3], [13].

In more complex cases, such models may be needed, in which the relations between system variables are given in the form of difference and differential equations. Such models are called mathematical (or analytical) models [9], [14], [17], [18]. Mathematical models are also an instrumental means of solving the problems of simulation modeling and forecasting, which often arise not only in engineering, but also in economics, ecology, biology and other fields [1], [4], [6], [11].

The application of neural networks (NN) to solve the problems of managing complex dynamic systems, namely systems with uncertainties, non-stationary systems, which is associated with the development of high technologies in various fields of science and technology, is the most common.

One of the approaches that is attracting more and more interest is the use of artificial neural networks (ANNs) to solve the problems of both identification (construction of neural network models) of nonlinear dynamic objects, and for the design of automatic control systems in general.

A huge number of works by Werbos P., Narendra K., Lewin A., Chen L., Omatu S. are devoted to the problems of the use of NN. This is due to the fact that neural network models are considered as a natural development of the traditional theory of linear systems, methods of optimizing the function of many variables, statistical methods [7], [8], [9], [10], [11]. An important class of ANNs was developed by the Finnish scientist Teivo Kohonen in 1982 [19], [20], [21]. Self-organizing Kohonen maps are a powerful neural network tool for multidimensional data analysis and visualization.

Accordingly, many issues related to the development of algorithms and methods for identification of nonlinear objects based on neural network models, synthesis of the structure and algorithms for adaptation (learning) parameters of neural network regulators, features of their implementation in multi-mode control systems for nonlinear dynamic objects have not yet been resolved.

The creation of neural network models that are significantly different from their biological prototypes, but which have such characteristic features as the ability to learn, generalize and abstract – these are the properties of NNs that allow to solve effectively the problems of building formal (mathematical) models of complex nonlinear dynamic objects according to absence of a priori information necessary for the use of traditional methods of identification.

It is clear that technical systems, including artificial neural networks as predictive models, adaptive regulators or universal computers, can be classified as intelligent systems [2], [3], [27], [16], capable of synthesizing a goal and finding rational ways to achieve it based on the use of information and knowledge in the presence of motivation.

2 Aim and Method

The article is devoted to applied aspects of the use of neural networks to solve the problems of dynamic systems identification. The main goal is to create a formalized approach to the neural network implementation of the identification procedure. In accordance with the goal, the main tasks are: substantiating the possibility of using neural networks as formal models of dynamic objects; development of basic neural network model structures; development, analysis and algorithmization of a multi-stage identification procedure based on neural network model structures, including planning and conducting an experiment; choosing the structure of the neural network; neural network learning; testing (establishing the adequacy) of the model; optimization of the neural network structure; creation of a hardware and software complex for technological process monitoring, based on the application of predictive neural network models.

Methods of optimization theory, discrete systems theory, probability theory, and mathematical statistics are used to solve the tasks. Digital modeling is used in experimental studies.

3 Results

Intelligent systems based on ANNs make it possible to successfully solve the problems of identification and management, forecasting, pattern recognition, and optimization. Other, more traditional approaches to solving these problems are also known, but they lack the necessary flexibility and have significant limitations on the operating environment. The use of ANNs provides more promising alternative solutions [2], [3], [9].

Artificial neural networks can change their behavior depending on the conditions of the external environment, that is, they can adapt. In order to provide the desired reaction, after the presentation of input signals (perhaps with corresponding outputs), neural networks self-adjust. The response of the network after training may be somewhat insensitive to small changes in the input signals. This feature of extracting an image through noise and distortion overcomes strict accuracy requirements [24]. Neural networks also have the ability to operate on data generated in the learning process. The listed properties make it possible to effectively use ANNs when solving many tasks [1], [8], [27].

Approximation of functions/modeling. There is a training sample $((x1, y1), (x2, y2), \dots, (xn, yn))$ (input-output data pairs), which is generated by an unknown function $y = f(x)$, distorted by noise. The task of approximation is to find the estimate of the unknown function $y = f(x)$. Approximation of functions is necessary when solving many engineering and scientific modeling problems [8].

Identification/prediction. Given n discrete readings of the output signals of the system $\{y(t1), y(t2), \dots, y(tn)\}$ (possibly with corresponding inputs $\{u(t1), u(t2), \dots, u(tn)\}$) in consecutive moments of time $t1, t2, \dots, tn$. The task is to build a model that predicts the value of $y(tn + 1)$ at the moment of time $tn + 1$. Predictive models can be used both in management systems [1], [9], and in nontechnical areas, for example, for analyzing stock exchange prices and weather forecasting.

Management. Consider a dynamic system given by the set $\{u(t), y(t)\}$, where $u(t)$ is the input control influence, and $y(t)$ is the output of the system at time t. In control systems with a reference model, the goal of control is to calculate such an input influence $u(t)$, in which the system follows the desired trajectory dictated by the reference model. A neural network is chosen as a model, and the dynamic process of its adjustment is a solution to the management problem [9], [10], [23], [28].

As for the classification of images, the task consists in determining the belongingness of the input image (for example, a speech signal or a handwritten symbol), represented by a vector of features, to one or several predefined classes. For example, recognition of printed and handwritten texts, speech recognition, classification of objects by their images and scene analysis [11], [18].

When solving the clustering problem, there is no training sample with class labels. The clustering algorithm is based on identifying the similarity of images through the selected metric and placing close images into one cluster. Clustering is used to obtain knowledge, compress data, and model complex technological processes [12], [25].

The task of the optimization algorithm is to find a solution that satisfies the system of constraints and maximizes or minimizes the objective function [13]. This is a classic example of an optimization problem successfully solved by ANN.

The task of identification, according to the general statement, consists in building an optimal model (formalized description) based on some criterion based on the results of observations of the input and output variables of the system. In practice, the implementation of the identification procedure (Fig. 1) requires the solution of a number of auxiliary tasks, the main of which are the planning/conducting of the experiment and the preliminary processing of experimental data, the selection of the model structure, the evaluation (optimization of the parameters) of the model, and the decision-making regarding the adequacy of the model.

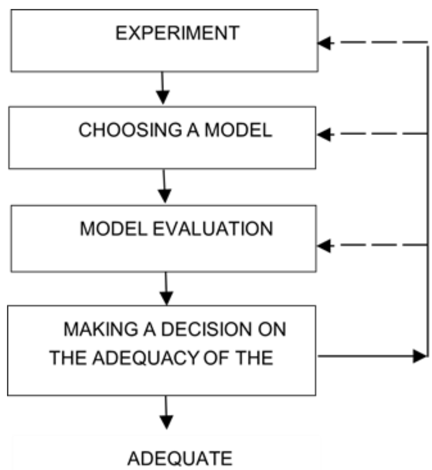


Fig. 1. Generalized scheme of the identification procedure implementation

The main task at this stage of identification is to collect the necessary amount of data in the entire working range of the system $Z^N = \{[u(t), y(t)], t = 1, \dots, N\}$ (Fig. 2). Completeness and reliability of data largely determine the quality of identification. The values of input and output signals can be recorded in the process of conducting targeted identification experiments, where the user can define the list and moments of measurement of signals, and some of the input signals can be controlled. The task of planning experiments, thus, is to choose the most informative data about the system signals, given the possible limitations. In some cases, the user may be deprived of the opportunity to influence the course of the experiment and must rely on the data of observations in the mode of normal operation. The main points at the stage of planning/conducting the experiment are the rational choice of the sampling frequency; input (test) signal synthesis; filtering, removal of unwanted effects from experimental data (trends, random (unfounded) signal "bursts"); testing for non-linearity.

The set of candidate models is established by fixing the group of models that are suitable for describing the studied system (for example, ARX or ARMAX models). Then the general form of representation of the model structure should be specified on the basis of information about the general dynamics of the system (for example, $Z^N = \{[u(t), y(t)], t = 1, \dots, N\}$ at the stage of the ARX experiment (2,2,1), where the description (2,2,1) defines the time delay for one sampling and the dependence of the current output signal of the system on two previous inputs and two previous outputs). That is, it is necessary to determine the input vector (regressor) for the selected family of candidate models. When using neural network models, a significant role is played not only by the choice of regressor, but also by the task of the internal structure of the NM – the number of hidden layers and the number of neurons in each hidden layer.

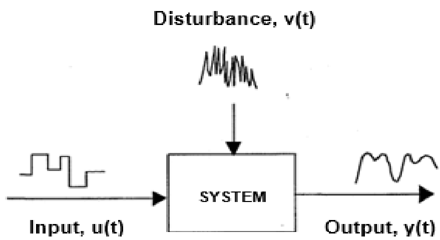


Fig. 2. Scheme of obtaining a set of experimental data

Choosing a model structure is the most important and at the same time the most time-consuming stage of the identification procedure. At this stage, knowledge about the formal properties of models must be combined with a priori knowledge of the object, engineering art and intuition.

After deciding on the use of a certain model structure, it is necessary to choose a certain model that best satisfies a certain criterion. The traditional strategy for evaluating the model is to evaluate the results of one-step forecasting using the root mean square error criterion [1], [3], [4].

The built model should be adequate to the real system and the conditions in which it is supposed to be used. The stage of establishing the adequacy (confirmation) of the model requires the direct participation of the development engineer. If the obtained model does not satisfy any criterion, then the previous stages of the identification procedure are repeated until a new series of experiments is conducted.

The algorithm for implementing the identification procedure generates a natural logic of action: collect data; choose many models; choose the best model in this set; make a decision about the possibility (impossibility) of using the model.

The result of the first three stages of the identification procedure implementation is a specific model – one of the set determined by the choice of the model structure, and the one that, according to the selected criterion, best reproduces the observation data.

It is likely that the first of the models found will not pass verification at the confirmation stage. In this case, one (or several) previous stages of the identification procedure are repeated.

Prototypes of neural network models are linear representations of dynamic systems. However, considerable attention is paid to the choice of the external and internal structure of neural network models.

Let's introduce strict definitions of system, model structure, and model according to the terminology adopted in [1], [17], [19], [25], [26]. A real system S can be represented as follows:

$$S\colon yt{=}g0\varphi t{+}e0(t) \tag{1}$$

where $g_0(\varphi(t))$ is some non-linear mapping implemented by the system;

$\varphi(t)$ – regression vector;

$e_0(t)$ – a "white noise" type signal that does not depend on system inputs.

The model structure (M) is a parameterized set of candidate models:

$$\mathrm{M}{:}g\varphi t,\theta,\theta \,\big|\, \theta \in DM \tag{2}$$

$$yt{=}\ g\varphi t,\theta,\theta{+}\mathrm{e(t)} \tag{3}$$

where θ defines a set of p adjustable model parameters;

D_M – some subset of the R^p space on which a specific model is searched.

The predictive model structure can be presented in the following form:

$$\hat{\mathrm{y}}t \,\big|\, \theta{=}\ g\varphi t,\theta,\theta \tag{4}$$

The main requirement for the model structure is that the real structure belongs to the set of M models:

$$\mathrm{S}{\in}\mathrm{M} \tag{5}$$

Model M* is a description of type (4) under the condition of a specific vector $\theta = \theta*$ task:

$$\mathrm{M}* = \ \mathrm{M}\ (\theta\ *);\ \theta * \in D_M \tag{6}$$

Thus, the task of identification consists in constructing some (generally non-linear) function $\mathrm{g}(\varphi(\mathrm{t},\theta),\theta)$ $\mathrm{w}here$ $\varphi(\mathrm{t})$ is a regression vector, and θ is a vector of parameters adjusted during the implementation of the identification algorithm.

The problem of nonlinear dynamic systems identification is related to the extreme complexity of choosing the model structure. The ability of multilayer neural networks to model arbitrary non-linear continuous functions as a result of training on a large number of examples allows to solve effectively this problem.

The implementation of the model structure on a two-layer neural network (taking into account (2 and 3)) has the following mathematical representation:

$$g(\varphi(t,\theta),\theta) = \hat{y}(t \mid \theta) = \hat{y}_i(t \mid (w,W)) = F\left(\sum_{j=1}^{n_h} W_i f\left(\sum_{l=1}^{n_\varphi} w_{jl\varphi l} + w_{j0}\right) + W_0\right) \quad (7)$$

where

$$f(x) = th(x) \quad (8)$$

activation function of neurons of the hidden layer;

$$Fx = kx; \ k = const \quad (9)$$

activation function of neurons of the output layer;

n_φ – dimensionality of the regression vector (the number of inputs of the NN);

n_h – number of neurons of the hidden layer;

θ – vector of tunable parameters of the neural network, which includes weights and neural shifts (w_{jl}, W_{ij}).

The use of NNs as model structures involves the solution of two main problems – the selection of the input vector (regressor) of the neural network model and the selection of the internal structure of the neural network.

A completely natural way to build neural network model structures is to use identification methods based on linear models. This approach has the following advantages: the definition of the regression vector is based on well-studied methods of building linear structures; the internal structure of the NN can expand depending on the degree of complexity of nonlinear mappings (4); the level of complexity of choosing a model structure can be significantly reduced, which is essential when using the method by end users (technologists); the obtained models can be used for the synthesis of control systems.

Neural network model structures can be represented by a vector of inputs (regressor) and a generalized form of description of the predictive model according to expression (4). The following modifications of linear regression models can be used as basic nonlinear neural network model structures:

NNARX (Neural Network − based AutoRegressive exogenous signal) − neural network autoregressive model.

Regressor:

$$\varphi(t,\theta) = [y(t-1) \dots y(t-n_a) \quad u(t-n_k) \dots u(t-n_b-n_k+1)]^T \quad (10)$$

Predictive model:

$$\hat{y}(t \mid \theta) = \hat{y}(t \mid t-1,\theta) = g(\varphi(t),\theta) \quad (11)$$

NNARX models, like their linear prototypes, are robust because there is a simple algebraic relationship between the predicted output and the previous values of the system's inputs and outputs. This property is especially important in the case of modeling nonlinear systems, determines the preference given to NNARX models in case of identification of deterministic objects with a low level of measurement noise.

NNARMAX1 (Neural Network − based AutoRegressive, Moving Average, exogenous signal, version 1) − neural network autoregressive moving average model, exogenous type of signals; version 1.

Regressor:

$$\begin{aligned}&\varphi(t,\theta) = \\ &= [y(t-1) \dots y(t-n_a) \quad u(t-n_k) \dots u(t-n_b-n_k+1) \quad g(t-1) \dots g(t-n_c)]^T = \\ &\quad = [\varphi_l^T(t) \quad g(t-1) \dots\ g(t-n_c)]^T \end{aligned} \tag{12}$$

where $g(t) = y(t) - \hat{y}(t \,|\, \theta)$ is prediction error.

Predictive model:

$$\hat{y}(t \,|\, \theta) = g(\varphi_1(t), \theta) + (\mathrm{C}(q^{-1}) - 1)\varepsilon(t) \tag{13}$$

where $\mathrm{C}(q^{-1}) = 1 + \mathrm{c}_1 q^{-1} + \dots + \mathrm{c}_n q^{-n}$ polynomial from the delay operator q.

Despite the fact that the function g is implemented on a direct-acting NN, the predictive model (13) has feedback: the prediction error depends on the output signal NN y̑. For the linear case (ARMAX model), the stability of the model can be established by analyzing the roots of the polynomial C, in the case of a neural network implementation, it is much more difficult to analyze the stability of the model. The NNARMAX model can be stable in one operating mode and unstable in another, which is a significant disadvantage in case of practical application.

NNARMAX2 (Neural Network − based, AutoRegressive, Moving Average, exogenous signal, version 2) − neural network autoregressive moving average model, exogenous type of signals; version 2.

Regressor:

$$\begin{aligned}&\varphi(t,\theta) = \\ &= [y(t-1) \dots y(t-n_a) \quad u(t-n_k) \dots u(t-n_b-n_k+1) \quad \varepsilon(t-1) \dots \varepsilon(t-n_c)]^T = \\ &\quad = [\varphi_l^T(t) \quad \varepsilon(t-1) \dots\ \varepsilon(t-n_c)]^T \end{aligned} \tag{14}$$

Predictive model:

$$\hat{y}(t \,|\, \theta) = g(\varphi(t), \theta) \tag{15}$$

This version of the NNARMAX model has the same advantages and disadvantages as the NNARMAX1. The difference is only in the presentation of the moving average directly by the neural network model (without using polynomial C).

NNSSIF (Neural Network – based, State Space Innovations form) – neural network model of the "state space updates" type.

Regressor:

$$\varphi(t) = \left[\dot{x}^T(t\,|\,\theta) \quad u^T(t) \quad \varepsilon^T(t\,|\,\theta)\right]^{\mathrm{T}} \tag{16}$$

Predictive model:

$$\dot{x}(t+1\,|\,\theta) = g(\varphi(t),\theta), \hat{y}(t\,|\,\theta) = C(\theta)\,\dot{x}(t\,|\,\theta) \tag{17}$$

Extending the state space update model [1], [18], [19], [20] in the case of nonlinear model structures is much more difficult than when modifying linear input-output descriptions of dynamic systems. As in the case of NNARMAX models, the problem of stability analysis plays an important role. Moreover, the question of establishing identity becomes much more complicated. In some cases, the problems can be solved by introducing several neural network structures to predict individual parts of the state vector [1], [19].

When using neural network models to implement the procedure for identifying non-linear dynamic systems, it is necessary to solve the problem of selecting the input vector of the NN (regressor) and determine the internal structure of the neural network.

The method of selecting a regressor is based on the availability of a priori knowledge about the system (process). Determining the internal structure of a neural network model is a more complex and ambiguous task.

Let's consider a regressor:

$$\varphi(t) = [\varphi_1 \ldots \varphi_d]^T = [\varphi_1(t-1) \ldots \varphi_1(t-n) \ldots \varphi_d(t-1) \ldots \varphi_d(t-n)]^T \tag{18}$$

where φ_i is i component of the regressor;

n – regression "depth".

The selection of the regressor involves the determination of the components of the regressor φ_{i} and the depth of the regression, that is, the number of n values of the regressor components at previous moments of time. The parameters of the system (process) that can be directly measured (or estimated) in the operating mode are usually used as regressor components. The choice of regression depth is determined by the dynamics of the system, therefore, in the absence of the necessary a priori information, it can be carried out by successively increasing n and checking the adequacy of the model. Another way is to choose a deliberately large value of n and carry out further structural optimization of the model.

The "external" structure of the neural network model is completely determined by the regressor and the set of parameters which values need to be predicted, that is, the number of inputs (the number of neurons in the input layer of the NN) is determined by the number of regressor elements, the number of outputs (the number of neurons in the output layer) is determined by the number of predicted values.

Minor changes in the internal structure of the neural network model usually do not have a significant impact on its quality, then the choice of regression depth, i.e., the number of signal readings at previous time points, plays a decisive role. Insufficient depth of regression leads to a model that does not take into account a significant part of dynamic properties, excessive depth of regression also causes a number of problems [1], [22], [23]. Initializing a neural network model structure by trial and error is a time-consuming procedure, which leads to a natural desire to get any recommendations on the initial choice of both the regression depth and the internal structure of the NN. Therefore, the correct choice of regressor significantly simplifies the procedure of initialization of the neural network model.

Let's suppose that as a result of conducting an experiment and pre-processing the data, some set is obtained:

$$Z^N = \{[u(t), y(t)], \quad \mathrm{t} = 1, \dots, \mathrm{N} \tag{19}$$

where $u(t), y(t)$ are system inputs and outputs respectively, N – number of discrete counts. Let's also assume that some model structure is selected:

$$\mathrm{y}(t) = \hat{\mathrm{y}}(t \mid \theta) + e(t) = g(t, \theta) + e(t) \tag{20}$$

According to the general scheme of the identification procedure implementation, the next stage is the evaluation of selected model structure parameters. When using neural network model structures, this stage is the adjustment of the weight coefficients of the network as a result of the training procedure implementation on many examples. Training is a mapping of a set of experimental data to a set of parameters of a neural network model

$$Z^N \to \hat{\theta} \tag{21}$$

with the aim of obtaining an optimal forecast of the output signal based on some criterion $\hat{y}$. The traditionally used criterion [1], [3], [4], [5] is the root mean square error of prediction:

$$V_N(\theta, Z^N) = \frac{1}{2N} \sum_{t=1}^{N} (y(t) - \hat{\mathrm{y}}(t \mid \theta))^2 = \frac{1}{2N}\, \varepsilon^2(t, \theta) \tag{22}$$

This approach belongs to the class of prediction error methods [1], since the main task is to minimize the total norm of the prediction error $\varepsilon = y(t) - \hat{y}\,(t|\theta)$. In some cases, norms other than quadratic are considered, which are optimal for a non-Gaussian distribution of disturbances $e(t)$. When using the criterion in the form of (22), the forecasting error method corresponds to the estimation by the maximum likelihood method under the condition of normal distribution of disturbances $e(t)$. The most attractive feature of the method is a fairly simple algorithm for estimating parameters (weighting coefficients) of NN and independence from disturbances (provided they are normally distributed) [1]. In some cases, this criterion is not absolutely optimal [1], but in practice, it usually leads to a better model.

At the stage of parameter optimization, the selection of the "best" (due to some criterion) model within a fixed model structure is implemented. According to the identification procedure (Fig. 1), the next step is to decide on adequacy, or to confirm the model. The main task at this stage is to get an answer to the question "how good" the optimized model is.

The task of building an optimal neural network model can be solved in two stages: first, the regressor structure is selected, which is determined by the type of model structure (NNARX, NNARMAX) (Fig. 3), as well as the regression depth on the input m and output n; then the internal architecture of the NN is optimized for the given regression vector. If the amount of experimental data (training set) is not limited (quite large), the problem of structural optimization becomes less significant [20], and a fully connected NN can be chosen as a model structure. For a complete network, the optimization task is reduced to determining the number of hidden layers and the number of neurons in the hidden layer. The easiest way to solve the problem is to sequentially increase the number of neurons in the hidden layers with a parallel check of the adequacy of the model on the test set.

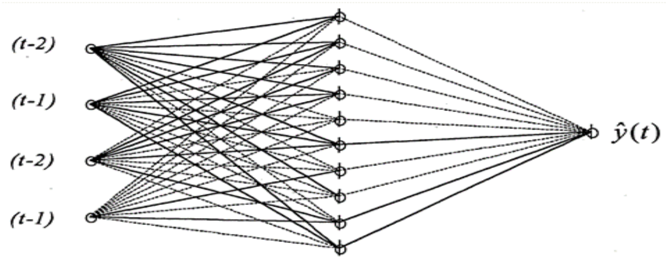


Fig. 3. The architecture of a fully connected neural network model (NNARX (2,2,1), 61 adjustable parameter) before structural optimization

Obviously, sequentially removing connections based on the results of testing a trained neural network model is not as intelligent as incrementing the number of neurons. In addition, working out all possible structural combinations requires significant computational costs. Therefore, it is necessary to build a method for ranking connections (weighting coefficients) of the neural network model in order to determine the best "candidates" for removal (Fig. 4).

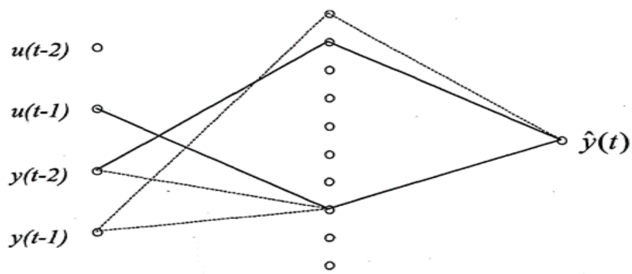


Fig. 4. The architecture of optimized neural network model after removing 53 structural connections (weights and neural shifts)

The approach to solving the problem of dynamic systems based on neural network model structures identification is a multi-stage procedure, at each stage of which a number of conceptually significant sub-problems are solved. The consistent solution of these subtasks using the considered recommendations determines the effectiveness of the method. When solving the problem of nonlinear systems identification, the entire amplitude-frequency operating range of the system should be uniformly represented in the set of experimental data. Data preprocessing issues such as filtering and zero mean and unit variance play an important role. Usually, the experimental data are divided into a training set and a test set, which are used, respectively, to adjust the parameters and confirm the adequacy of the model. The proportions in which this division is to be made are determined by the size of the experimental set; the more data, the larger the size of the test set can be. It is recommended to use model structures of the NNARX type, implemented on fully connected neural networks, and the ratio of the number of training pairs to the adjustable parameters of the NN should be in range from 3 to 10. Training should be carried out on the basis of a non-regularized criterion.

With a significant size of the set of experimental data, the effectiveness of regularization methods is insignificant. Therefore, it is recommended to use fully connected NNs, which are trained without regularization, as model structures. At a high level of noise, structures of the NNARMAX type can be used.

At the stage of evaluation (optimization of parameters) of the model, it is necessary to choose a criterion, taking into account which the optimization procedure is implemented. In addition, one of the considered methods of optimization (training) of the neural network model should be selected, the traditional least squares criterion and its modification, which uses the concept of regularization by introducing the damping of weight coefficients, should be investigated. To implement the neural network model training procedure, it is recommended to use the Levenberg-Marcard method. This method, developed specifically for solving the problems of unconditional minimization of the mean square criterion, has a high speed of convergence, computational robustness and ease of application. Since the function to be minimized generally has several local minima, it is recommended to train NN 5–7 times, changing the initial values of the weighting coefficients.

Making a decision about the adequacy of the model largely depends on the specifics of the task and the expected practical application of the model. In general, it is desirable that the performance of the model is confirmed on the test set. The neural network identification procedure practically ensures obtaining adequate forecasting models intended for modeling the behavior of dynamic systems.

4 Conclusion

Based on the analysis, systematization and generalization of scientific achievements in such fields as computer science, mathematics, statistics and neurophysiology, a complex formalized approach to solving the problem of dynamic object identification using neural network models was formed.

It is appropriate to use neural network algorithms for the identification of dynamic objects during the creation of software and technical means of information and computing complexes for modeling nonlinear process control systems in the absence of a sufficient amount of a priori information. The practical implementation of the developed algorithms is a software product, which is a composition of modules that correspond to each stage of the identification procedure and are combined in a single shell. Intuitive user interface, speed and computational robustness of the algorithms used allow obtaining adequate neural network models of dynamic objects in a short time and with minimal user involvement.

The unique capabilities of artificial neural networks (learning, generalization, abstraction), inherited by them from biological prototypes, make it possible to effectively use neural network structures to solve a large number of technical problems, in particular, the identification of dynamic objects. The task of identification consists in building a model (formal, mathematical description) based on experimental data.

Universal approximating properties of multilayer neural networks determine the possibility of their application as nonlinear dynamic objects.

ANNs (in the case of hardware implementation) are ideal for solving problems of real-time identification and control, as they provide extremely high speed of operations.

References

[1] Smyrnov, S., Shalashenko, M.: Neural networks for forecasting problems of financial time series application. Ekonomika: Visnyk DNU im. Olesya Gonchara 2(10), 120–127 (2008) [in Ukrainian].

[2] Swingler, K.: Applying Neural Networks: A Practical Guide. Academic Press, London (1996).

[3] Sharov, S., Lubko, D., Osadchyj, V.: Intelligent information systems. Vydavnytstvo MDPU im. Bogdana Khmel'nyts'kogo, Melitopol (2015) [in Ukrainian].

[4] Haykin, S.: Neural Networks: A Comprehensive Foundation. 2nd ed. Pearson Education, Singapore (2005).

[5] Hopfield, J.: Neural networks and physical systems with emergent collective computational abilities. In: Proceedings of the National Academy of Sciences, vol. 79, pp. 2554–2558 (1982).

[6] Lakhno, V., Malyukov, V., Kryvoruchko, O., Desiatko, A., Shestak, Y.: Smart city technology investment solution support system accounting multi-factories (2020). doi: 10.1007/978-3-030-63322-6_1.

[7] He, X., Asada, H.: A new method for identifying orders of input-output models for nonlinear dynamic systems. In: Proc. of the American Control Conference, San Francisco, pp. 67– 83. (1993).

[8] Peterson, C.: Determining dependency structures and estimating nonlinear regression errors without doing regression. International Journal of Modern Physics, vol. 611, pp. 18–31 (1995).

[9] Larsen, J., Hansen, L.: Generalization performance of regularized neural network models. In: Neural Networks for Signal Processing: Proc. of the IEEE Workshop IV, Brussel (Belgium), pp. 42–51 (1994).

[10] LeCun, Y., Kanter, I.: Eigenvalues of covariance matrices: application to neural- network learning. Physical Review Letters, vol. 66, pp. 2396–2399 (1991).

[11] Billings, S., Zhu, Q.: Nonlinear model validation using correlation tests. Int. Journal of Control 60(6), 466–470 (1994).

[12] Hassibi, B., Stork, D.: Second order derivatives for network pruning: optimal brain surgeon. In: Proceedings of the NIPS5, San Mateo (California), pp. 164–172 (1993).

[13] Pederson, M., Hansen, L.: Recurrent networks: second order properties and pruning. In: Neural Information Processing Systems: Proc. of th 7-th Conference, Vienna (Austria), pp. 673–680 (1994).

[14] Lakhno, V., Matus, Y., Malyukov, V., Desyatko, A., Hnatchenko, T.: Smart city cybersecurity projects financing model in case of description of investors' resources with fuzzy sets. In: IEEE International Conference on Advanced Trends in Information Theory, ATIT 2019 – Proceedings, pp. 249–252 (2019). doi: 10.1109/ATIT49449.2019.9030499.

[15] Bebeshko, B., Khorolska, K., Kotenko, N., Desiatko, A., Sauanova, K., Sagyndykova, S., Tyshchenko, D.: 3D modelling by means of artificial intelligence. Journal of Theoretical and Applied Information Technology 99(6), 1296–1308 (2021).

[16] Lakhno, V., Akhmetov, B., Mazaraki, A., Kryvoruchko, O., Chubaievskyi, V., Desiatko, A.: Methodology for assessing the effectiveness of measures aimed at ensuring information security of the object of informatization. Journal of Theoretical and Applied Information Technology 99(14), 3417–3427 (2021).

[17] Tereikovskyi, I., Tereykovska, L., Mussiraliyeva, S., Tsiutsiura, M., Achkoski, J.: Markov model of unsteady profile of normal behavior of network objects of computer systems. In: CEUR Workshop Proceedings, p. 2588 (2019).

[18] Terentyev, O., Tsiutsiura, S., Honcharenko, T., Lyashchenko, T.: Multidimensional space structure for adaptable data model. International Journal of Recent Technology and Engineering 8(3), 7753–7758 (2019).

[19] Lakhno, V., Kasatkin, D., Kartbayev, T., Togzhanova, K., Alimseitova, Z., Tussupova, B.: Analysis of methods and information technologies for dynamic planning of smart city development. International journal of advanced trends in computer science and engineering 9(5), 7496–7505 (2020).

[20] Dudnyk, A., Hachkovska, M., Zaiets, N., Lendiel, T., Yakymenko, I.: Managing a greenhouse complex using the synergetic approach and neural networks. Eastern-European Journal of Enterprise Technologies 4(2), 72–78 (2019).

[21] Lutska, N., Zaiets, N., Vlasenko, L., Shtepa, V., Savchuk, O.: Forecasting the Efficiency of the Control System of the Technological Object on the Basis of Neural Networks. In: Proceedings of the 20th IEEE International Conference on Modern Electrical and Energy Systems, MEES 2021, pp. 1–6 (2021). DOI: 10.1109/MEES52427.2021.9598540

[22] Savchuk, O., Ladanyuk, A.: Research of possibilities of using neural networks in the decision support system. Eastern-European Journal of Enterprise Technologies 4(4), 15–19 (2015).

[23] Korobiichuk, I., Smityuh, Y., Kishenko, V., Boyko, R., Elperin, I.: Identification of techno–logical objects on the basis of intellectual data analysis. Advances in Intelligent Systems and Computing, 487–495 (2020).

[24] Korobiichuk, I., Shevchuk, D., Prokhorenko, I., Smityuh, Y., Boyko, R.: Synthesis of an intelligent uav control system based on fuzzy logic in external disturbance conditions. Journal of Automation, Mobile Robotics and Intelligent Systems 14(3), 3–9 (2020).

[25] Ladanyuk, A., Shumygai, D., Boiko, R.: Situational coordination of continuous technological complexes subsystems. Journal of Automation and Information Sciences 45(8), 68–74 (2013).

[26] Kharlamenko, V., Ruban, S., Korobiichuk, I., Petruk, O.: Adaptive control of dynamic load in blooming mill with online estimation of process parameters based on the modified kaczmarz algorithm. Advances in Intelligent Systems and Computing, 227–233 (2017).

[27] Ngugen, D., Widrow, N.: Neural Networks for selflearning control systems. In: IEEE Control Systems Magazine, vol. 10, pp. 18–23 (1990).

[28] Adranova, A., Yona, L., Kryvoruchko, O., Desiatko, A., Palahuta, K., Blozva, A., Gusev, B.: Methodology forming for the approaches to the cyber security of information. Journal of Theoretical and Applied Information Technology 98(12), 1993–2005 (2020).

A Method for Developing a Digital Twin with a Web Service Access Software Interface Based on Simulink Coder Generated Code

Andrii Stopakevych[1] [0000-0003-1719-9071], and Oleksii Stopakevych[2] [0000-0002-8318-6853]

[1] State University of Intellectual Technologies and Communications, Odessa, Ukraine
[2] Odessa Polytechnic National University, Odessa, Ukraine
stopakevich@gmail.com

Abstract. A common approach to the development of digital twins of dynamically functioning objects in the form of real-time software systems is the generation of these twins according to existing computer models, which is even recommended by some standards for the safety of industrial systems. MATLAB Simulink Coder toolbox generates a program code in C language. This generated code is a program module that is intended to be integrated into a program in the same language. But, if we want the interface of communication with the dynamic model to be implemented by RESTful web service technology, the development of such a solution in C language will be long enough in time. This paper proposes a simpler alternative method including three steps. First, the generated C program should be implemented in an executable file with a command-line interface. Second, the whole part related to the implementation of the web service should be implemented using a Python program based on the Flask library. Third, the interaction between the Python program and the executable with the command line interface should be implemented through the “expect” interface.

Keywords: web, service, python, simulink, coder, model

1 Introduction

Simulink is one of the most well-known MATLAB toolboxes, which allows users to model dynamic systems in a form of block diagrams and conduct their simulations. According to the original idea, Simulink is a tool to investigate dynamic systems properties (e.g., compliance of control system models to robustness requirements), but not to implement them. For implementation, it was developed a separate toolbox – MATLAB Simulink Coder, which allows users to generate program code in several languages (e.g. C, C++) for different platforms. The aim of generated code is its execution in real-time environments. Note that a generated code is not ready for execution as is, it should be included in other software, which provides desired communication interfaces. Alternatively, we can run Simulink models on the Desktop in real-time mode. This feature is also implemented in modern MATLAB. It allows running models in near real-time mode and communicating with other systems via OPC protocol, ADC interfaces, and some other interfaces, which are commonly implemented with the usage of additional dynamic link libraries. Running the MATLAB Simulink model in real-time mode has some disadvantages: it needs MATLAB or huge runtime time step; execution is single-threaded, so only one model can be run; and some program interfaces in MATLAB (for example, .NET) are not fast, so real-time violation problems may arise.

These circumstances force developers to look for ways out of the situation. This paper proposes a technology that allows to quickly transform the Simulink model into an autonomous dynamic system operating in real-time as a web service in a large number of software environments, including environments with limited re-sources. Such systems are useful for digital twins' realization (to debug control and information systems), as well as for educational and research purposes – for the realization of laboratory works with the possibility of their parallel execution by students remotely.

2 Literature review

One of the modern representative papers that describe the development of digital twins with real-time Simulink models without code generation is [1]. All controlled plant dynamics (generator, turbine-governor, etc.) and control systems (voltage regulators and power systems stabilizers) are implemented in the Matlab Simulink model. The model runs in real-time mode and is connected with the PXIe-8133 controller using special DLLs. A user can work with the digital twin of a controlled power plant using LabView.

The possibilities and problems of publicly available Simulink models' development are analyzed in the detailed review [2]. The scope of the review is 1,734 Simulink models from 194 projects. Some gained statistics show that the median model has about 70 blocks, a typical industrial model consists of an average of 200–2,000 blocks, up to 20,000. Manual translation of such models to program code can lead to errors that require additional time to find. Some base standards in automotive, medical, and other high-risk industries recommend using program code generation tools to minimize error probability [3]. The use of Simulink to construct models for high-risk systems is justified not only by the variety of build-in blocks which allows modeling a system of very high complexity but also by features for a detailed study of the adequacy and accuracy of the models

The book [4] is fully devoted to the technical aspect of code generation in MATLAB and Simulink coder. Proper tuning of the generator allows us to find a balance between execution time, resource consumption, and code clearness. Also, some differences in code generation tasks in C/C++ are considered.

The papers [5], [6] describe in particular the problem of using additional tools with unique metrics to analyze the complexity of models. Some guidelines to construct more effective models using modularity principles are given. Some differences between C language and Simulink structures are shown.

Generally, we can say that existent scientific papers investigate two separate tasks: 1) realization of a connection interface to the real-time Simulink model; 2) optimizing

Simulink model structure for more efficient code generation. The idea of this work is to develop new technology to realize a connection interface in executable based on generated code of a real-time model. We'll use web service technology as a base for connection realization.

3 Proposed method

3.1 Aim of the proposed method and requirements that it must meet

This paper aims to develop digital twins with a web service interface on a base of code, generated by Simulink Coder. The software should not require MATLAB or MATLAB Runtime setup. In addition, the digital twin software development method must meet the following requirements: 1) fast development; 2) portability, i.e., the ability to work in both general-purpose OS, real-time OS, and OS for single-board computers; 3) low resource requirements; 4) a standard remote access software interface.

3.2 Selection of tools and strategies to implement the requirements

Simulink Coder allows the conversion of most Simulink models into C and C++ code. Therefore, the first step is to generate code in one of these languages. Though C and C++ are formally cross-platform, in practice programs that implement threads handling, network communication and drivers will differ greatly due to incompatible program interfaces of different OSes. That's why the completion of the generated program code up to a full-fledged web service in C/C++ languages is a rather expensive task and consequently does not meet requirement 1 (fast development). Implementation of an additional library or a module for Apache or Nginx web servers is also possible and is simpler than the previous alternative, but does not meet requirement 2 (portability). We'll also notice that the software interfaces of web servers are constantly changing and it's required to periodically update them due to security reasons. Therefore, to implement the functionality of the web service, it is proposed to write an additional program in a programming language that is optimized for this purpose
Among the languages optimized for the development of web services we'll notice C#, Java, Go, and Python. The first two languages require a virtual machine, which is not available for all platforms and does not meet requirement 2 (portability). In terms of greater compliance with requirement 1 (fast development), the language Python wins. Python is simpler than Go, functional, and extensible due to a large number of free libraries. In addition, Python programs do not require compilation to run. Go language is roughly speaking a more developed version of C language, which is optimized not only for system programming but also for network and web applications. Although the difference in speed between Python and Go programs can differ approximately ten of times, in the considered case the "bottleneck" is the program, which runs the model in real-time. Therefore, at high loads, the system will fail not because of the slow web service, but because of the inability to parallelly execute multiple dynamic models in real-time. Consequently, we choose Python.

As a mechanism of interaction between the program on the Python and C / C ++ program can be considered the following mechanisms: 1 – implementation of a library in C / C++ with the subsequent call it from the program on the Python; 2 – ex-change, using sockets between applications; 3 – unidirectional information exchange with the command line interface. In terms of requirement 2 (portability), it is worth choosing the command line interface. Although the first mechanism is generally the most correct, for Python it is impractical because of Python's use of GIL locks in multithreaded programs, which causes problems with the speed and reliability of the program. The second mechanism makes the C program highly platform-dependent. Therefore, the third mechanism should be chosen, because Python has libraries for command line interactive applications, both for POSIX-compatible systems and for Windows (there is no such library for Go, for example, at the moment).

Since object-oriented functionality is not required in this case and reduces the speed of operation, we will choose the C language for code generation. All program codes should be written according to the C99 specification.

There are two main specifications of web services: REST and SOAP. The first one is more popular lately and is more in line with requirements 3 (low resource requirement) and 4 (standard interface). RESTful services use JSON data schemas transferred via HTTP protocol as a message language.

The structural diagram of the proposed software solution to implement an autonomous digital twin with web service API is shown in Fig. 1.

3.3 Steps of digital twin development by the proposed method

The method we propose is performed in 6 steps:

1. Selection of blocks that can be transformed by Simulink Coder;
2. Manual or automated Simulink model assembly;
3. Real-time implementation of the dynamic system solver as a console application based on the generated code;
4. Implementation of RESTful web service in Python;
5. Implementation of interface for remote access to web service.
6. Integration of RESTful web service into the existing software and hardware environment.

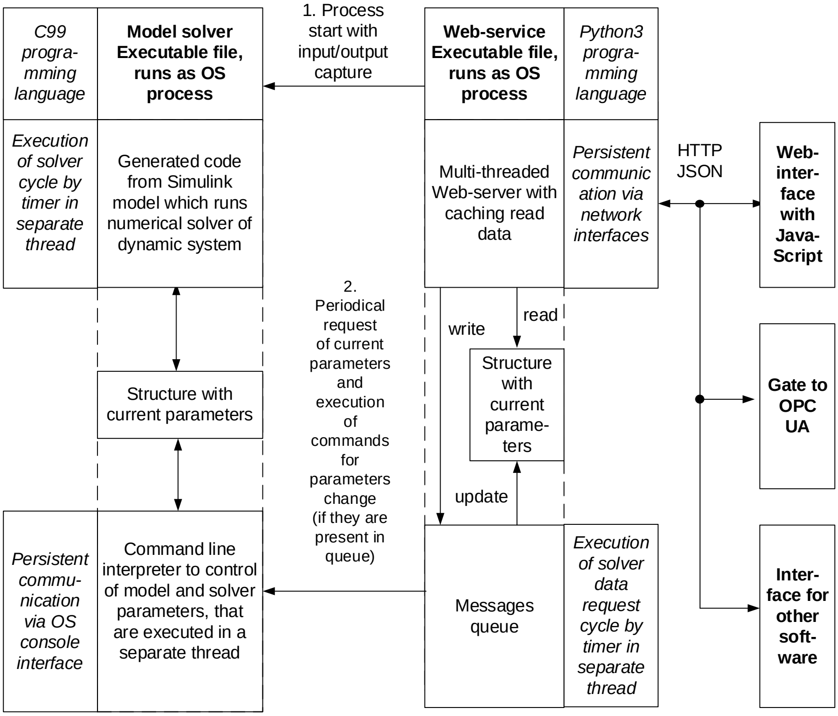


Fig. 1. Software solution scheme to implement an autonomous digital twin with web service API

4 Illustrative example of the proposed method

4.1 Simple dynamic system

Let us consider the problem of modeling the linear dynamic system

$$W(s) = \begin{bmatrix} W_{11}(s) & W_{12}(s) \\ W_{21}(s) & W_{22}(s) \end{bmatrix} = \begin{bmatrix} \frac{11}{111s+1}e^{-100s} & \frac{12}{333+1}e^{-200s} \\ \frac{21}{222s+1}e^{-300s} & \frac{222}{444s+1}e^{-400s} \end{bmatrix}$$

where s is the Laplace operator.

4.2 Step 1 – Selection of blocks that can be transformed by Simulink Coder

The main requirements for models in Simulink Coder are the following:

1. The model must be implemented with one of the solvers in discrete time. This means that some blocks are not applicable, in particular all blocks of the Simscape toolbox.

2. A model can only contain MATLAB code from a subset of Embedded MATLAB.
3. The model must not contain MEX and other platform-dependent blocks, including those responsible for accessing basic programming interfaces.

All model parameters that the user must change or monitor must be distinctly identified since MATLAB tends to reduce the code responsible for calculations when generating code.

It is also desirable, if possible, to replace blocks with a set of parameters with elementary Simulink blocks to increase the clarity of the model.

To make the model structure clearer, let us represent the transfer functions (TF) in the form of differential equations:

$$\dot{x}(t) = \frac{-1}{T} \cdot x(t) + \frac{k}{T} \cdot u(t-\tau),$$

where x – state variable, T – time constant of TF, k – gain of TF, τ – transport delay.

Such form can be realized with several Simulink blocks (sum, integrator, transport delay, gain, constant, in, out, scope). And all of them can be transformed to code by Simulink Coder.

To make the model more interactive we add the requirement that a user should see the values of the dynamic system outputs (y_1, y_2), as well as the outputs of the main model blocks (integrators, lags, and blocks for calculating the k/T ratio). We choose ode5 as a discrete solver. It's a discrete analog of the default Matlab solver ode45. The sample time of the discrete solver is 0.2.

4.3 Step 2 – Manual or automated Simulink model assembly

The standard way to develop a mathematical model scheme in Simulink is to assemble it manually. For a further generation, each block must be given a clear identifier in Latin letters in order to localize the code responsible for it.

Features of Simulink models automated assembly in MATLAB are not much developed. The `add_block` function is used to add blocks to the diagram with given parameters, and the `add_line` function is used to link inputs between blocks.

An interesting solution to this problem is found in the Matsim library [7]. This library has two unique features: the construction of a Simulink model from equations and the automatic placement of blocks. The placement algorithm is implemented in such way: 1) determine a block for every operation and measure all block sizes; 2) describe a layout problem using DOT language, which is solved by finding the optimal placement of blocks with a minimum of intersections taking into account their order and priority; 3) translate a solution to the real coordinates of Simulink blocks and construct a model with them. The schematic representation of the solution for the element (1,1) of the transfer function matrix, performed with the GraphViz interpreter of DOT language is shown in Fig. 2.

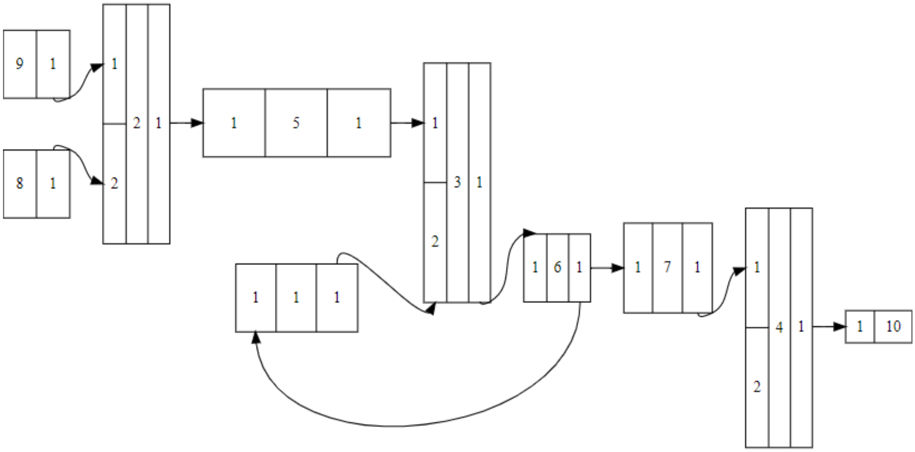


Fig. 2. GraphViz representation of the DOT language interpreter solution

The control signal (block 8) is summed (block 2) with the disturbance signal to the input (block 9). The value is then multiplied by the result of the *k*/*T* calculation (block 5). The calculation result of block 5 is summed in block 1 with the calculation result of block 1 (-1/*T*). The signal then passes through the integrator (block 6), the lag-link (block 7), is summed in block (4), and is written to the output variable y_1 (block 10).

The manually or automatically assembled Simulink model must be tested for compliance with requirements. This procedure can be automated with standard Simulink functionality, special scripts, and additional toolboxes.

4.4 Step 3 – Real-time implementation of the dynamic system solver as a console application based on the generated code

First, we need to select the target and programming language in Simulink Coder. In the proposed algorithm, the target ert (embedded systems) and the C language should be selected. A generated code can be compiled with any modern compiler for any OS.

Required settings: generate code only, dynamic memory allocation for model initialization, represent root level inputs and outputs as model data structure, parameter behavior – configurable, leave operand order in expression unchanged. It is also recommended to set goals: efficiency and traceability.

After generation, Simulink Coder forms a folder with the program that runs an infinite computational loop without a timer. Unlike programs generated by the usual Matlab Coder, the program contains loop handlers that allow modification of the code for real-time simulation.

The key file of the generated code is a file with the Simulink model name and a .c extension. This file (for this example it is about 30 kb of code) implements numerical differentiation using the ode5 algorithm and does not need modification. The file with the ending _data.c contains the data structure, which contains the parameters of the model, which can be modified at runtime.

The file that needs to be modified is called ert_main.c. There are two tasks to be implemented in this file: 1) the ability to modify model parameters; 2) execution in real-time or with a given acceleration.

To solve the first problem, we should realize a simple command line interpreter (CLI) which is a trivial task and has many examples of realization. A minimal command line interface must have the following commands: 1) start – start the simulation; 2) pause – pause; 3) restart – reset and start the simulation; 4) time – display the current simulation time / move to a given time; 5) acc – simulation acceleration factor; 6) set – set parameter 7) get – get parameter value, parameter group or all blocks; 8) status – current state (pause, execution, error); 9) stepsize – get simulation step; 10) quit – stop the program.

As in classic CLI after the program boot command prompt should appear. To execute a command the ENTER key must be pressed so that a result of a command can be shown. To increase the reliability of working with the "expect" tool, it is recommended to mark the beginning and the end of the result of the command with special ASCII characters (#2 and #3 in the decimal system). In addition, the result should have a character that indicates an error (#21) and a successful result of execution (#6).

The solution to the second problem depends largely on the capabilities of the platform. For a cross-platform solution, we need to implement at least POSIX threading as Windows threading.

4.5 Step 4 – Implementation of a RESTful web service in Python

API Design. First of all, to implement any web service it is necessary to define a programming interface. In our case we'll only 2 types of commands: GET (read parameters) and PUT (change parameters). Data serialization follows JSON rules.

GET commands – commands for reading data

1. /get/acc – current simulation acceleration factor. Example output: 1.000000.
2. /get/blocks – current block states. Example output: {"A11": –0.045103, "A12": –0.0, "A21": –0.024769, "A22": –0.0, "Add": 1.0, "Add1": 0.0, "Add4": 0.053996, "Add5": 0. 069825, "Add6": 0.0, "Add7": 0.0, "G11": 0.099099, "G12": 0.0, "G21": 0.094595, "G22": 0.0, "Time": "67.800000", "W11": 5.006425, "W11delay": 0.0, "W12": 0.0, "W12delay": 0.0, "W21": 5.498778, "W21delay": 0.0, "W22": 0.0, "W22delay": 0.0}
3. /get/model/[k, T or tau] – current block coefficients. Example output for k: {"k11": 11.0, "k12": 12.0, "k21": 21.0, "k22": 22.0}.
4. /get/status – simulation state. Examples of output: RUNNING, PAUSED, ERROR.
5. /get/stepsize – simulation step. Example output: 0.2.
6. /get/time – simulation time. Example output: 789.200000.
7. /get/variable/[u, f or y] – the current values of the main variables with the simulation time. Example output for y: {"y1": 10.990272, "y2": 19.569536, "Time": "880.400000"}.

PUT commands are commands to write data

1. /set/acc – new simulation acceleration factor. Example request: {“new_acc”:10}
2. /set/model – new model parameters. Example query: {“k11”:2, “T12”:4}
3. /set/status – change of state. Example query: {“new_status”: “pause”}
4. /set/time – accelerate to new time. Example query: {“new_time”: “500”}
5. /set/time – new variables. Example query {“u1”:1, “f2”:5}

Configuration file format selection. We suggest storing the configuration as an INI text file. This form is optimal for storing a small number of settings and convenient for manual editing by the user. The configuration must contain the following parameters:

1. network – network address and port,
2. name and encoding of the executable file of the solver
3. default start mode – background program or program with command line interface, the requirement of automatic start of the simulation
4. security parameters (presence of guest access, necessity of authorization for data recording, permission for remote reboot, and suspension of simulation)
5. reporting parameters – file name for the report, degree of report detail (debug, normal, errors only)
6. model parameters, entered before starting.

Selection of software libraries. The Python language is the leader in the number of quality and free libraries for many typical programming tasks.

The most popular libraries for implementing web services and applications in Python are “Django” and “Flask”. The “Flask” library is minimalist while sufficient for the proposed procedure, so we recommend using it. Implementing a public programming interface requires libraries to control access from other resources – “Swagger” and “flask-cors”. We recommend using “flask-login” library for authorization functionality and “lru-cache” library for caching. Since simulation is discrete, caching GET requests can significantly reduce the load.

The “wexpect” library for Windows and “pexpect” for POSIX-compatible platforms are recommended to use for interaction with the solver's executable program CLI. The API of these libraries is almost identical, but the internal implementation is completely different.

To implement multithreading, it is necessary to use the functions of the “os”, “threading” and “signal” libraries.

To form the priority queue it is recommended to use the functionality implemented with the help of “queue”, “itertools” and “TypeVar” libraries.

To process configuration files, it is recommended to use the “configparser” library (the advantage of this library is also the possibility to change configuration parameters by passing command line arguments to the program), to maintain reports use “logger” library, to record the current date and time in the report by format use “datetime” library.

Proposed software realization strategy. The program should run four parallel processes: the main program process, the Flask server process, the AppThread, and the TimerThread processes.

The main program process initiates the other three processes and handles I/O and standard OS signals (SIGTERM, SIGINT, etc.).

The Flask server process runs in multithreaded mode and handles the described JSON API addresses (routes). Using a decorator, which contains an address template, a handler function is bound to the address. In the case of a "GET" request, the handler functions have to return the data from the cache which is implemented using the functions with the "lru_cache" decorator specifying the period of the cache update. In the case of a "PUT" request, the input data is checked and if it is correct then it is added to the queue with priority 1.

A process realized with the TimerThread class periodically adds commands for updating variables (priority 2) and updating block parameters (priority 3) to the queue. The process also adapts the time of cycle adding to the queue based on the results of tracking the time of returning the result of commands from the solver, to avoid overflow of the queue when the computer load is high.

The process realized with the AppThread class provides continuous operation with the solver using expect-like control: it selects commands from the queue, sends commands to the solver (sendline), waits for the return of the result (expect) and reads the results (readline). The AppThread class structure and main algorithms activity diagrams are shown in Fig. 3.

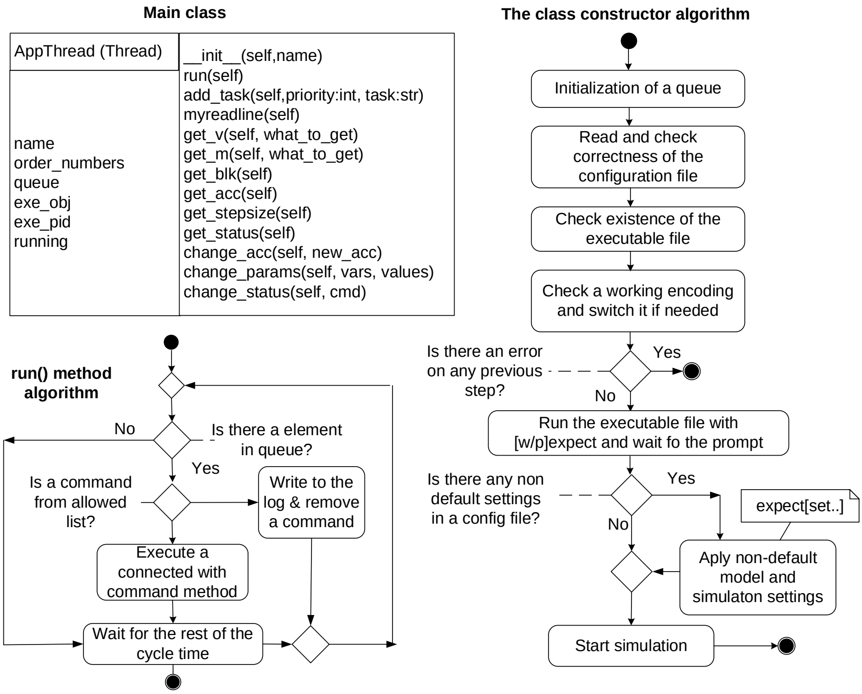


Fig. 3. The main class structure and algorithms

Portable distribution. The executable file of the solver program when built with the MINGW64 compiler takes ~1.4 MB of disk space, in working mode, the process takes ~14 MB + library memory. Portable Python program builds for different platforms can be implemented with the "cx_Freeze" utility. Under Windows, by fine-tuning it was possible to achieve the size of the executable ~13 MB + ~24 MB of libraries.

4.6 Step 5 – Implementation of an interface for working remotely with a web service

The easiest way to communicate with RESTful web service is the console client curl. For some tasks, this interface can be sufficient.

One of the most common user interfaces for web services is a web interface. The technology for developing such an interface is as follows 1) Using the Matlab `print` command, the Simulink model is exported into SVG format. 2) Based on the coordinates calculated with the DOT language, the necessary text elements are superimposed on the diagram. 3) Inscriptions are timed with the function of periodic exchange with the web service (when using the library JQuery, for example, the exchange of messages is implemented using the function `$.getJSON`) 4) functional elements are created to change parameters using put calls (in jQuery function `$.ajax`). 5) Plots are built, if necessary.

An example of a working interface is shown in Fig. 4.

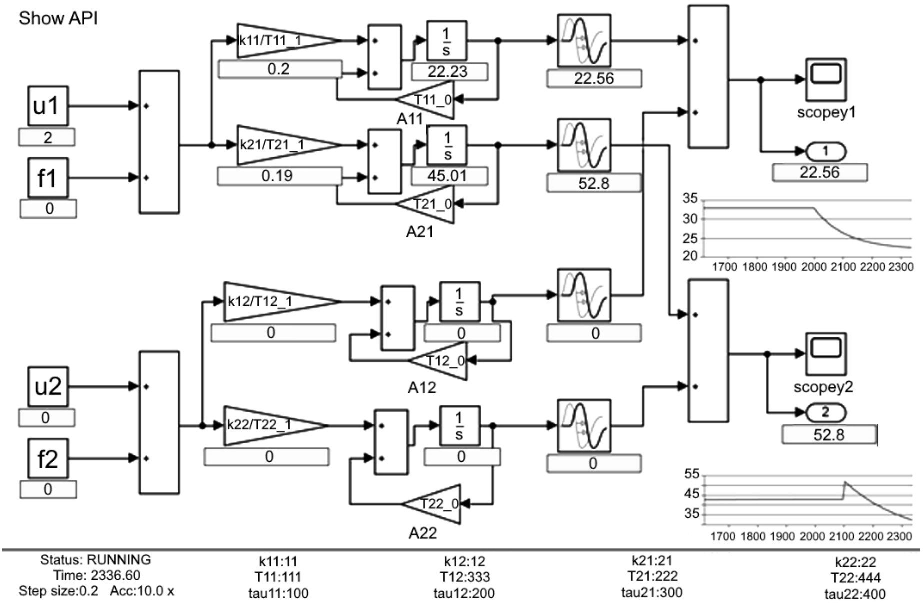


Fig. 4. The working web interface GUI demonstration

The figure shows the moment after changing the value from u_1 from 3 to 2. When you click on a parameter that can be changed, a dialog box appears in the interface with a text field. On confirmation, the data is transmitted to the web service. Real-time plotting is done using the CanvasJs library. The yellow rectangular frames show the output signals of the blocks.

4.7 Step 6 – Integration of RESTful web service into the existing software and hardware environment

This step is difficult to formalize because much depends on the task. For integration with MQTT is possible to use the functionality of IoT platforms. For integration with a typical modern industrial automation system, it is necessary to use JSON-OPC UA gateway. Such a gateway can be developed separately, or use the standard one, for example, implemented in the OPC Router package. For educational purposes, it is reasonable to develop an additional program, which will authorize users to perform the specified work. After authorization on the server, a separate web service should start for a particular user with a network port allocated to it. After the work is done, the service must stop working and release the port and save a report with the user's actions.

5 Conclusions

The paper proposes a method for integrating dynamic systems running in real-time with software using web service technology. Dynamic systems implemented in the MATLAB Simulink are considered. The method is supposed to adapt the program code of a dynamic model generated in the C language in such a way that it works with an application in Python language, which implements the interaction with web service technology.

The proposed method meets the following criteria: 1) fast development; 2) portability, i.e., the ability to work in both general-purpose OS, real-time OS, and OS for single-board computers; 3) low resource requirements; 4) a standard remote access software interface.

The proposed method has no complete analogs. The most similar result was obtained in the paper [8]. This paper executes a web service for a MATLAB Simulink model running in real-time using the additional adapter. The Java programming language is used to develop the service side of the application. In our method, web service does not require MATLAB Simulink to run permanently. So, a developed using the proposed method web service is more portable and makes it possible to run multiple web services for different purposes on a single computer.

References

[1] Parizad, A., Baghaee, H.R., Iranian, M.E., . Gharehpetian, G.B., Guerrero, J.M.: Real-time simulator and offline/online closed-loop test bed for power system modeling and development. International Journal of Electrical Power & Energy Systems (122) (2020). https://doi.org/10.1016/j.ijepes.2020.106203.

[2] Boll, A., Brokhausen, F., Amorim, T., Kehrer, Timo, Vogelsang, A. Characteristics, potentials, and limitations of open-source Simulink projects for empirical research. Software and Systems Modeling 6 (20), 2111–2130 (2021). https://doi.org/10.1007/s10270-021-00883-0
[3] Industry Standards – MATLAB & Simulink, https://www.mathworks.com/help/ecoder/ug/industry-standards-ecoder.html, last accessed 2022/09/18.
[4] Marquez, F., MATLAB and Simulink Code Generation. Independently Published (2019) ISBN: 1093112301, 9781093112306
[5] Jaskolka, M., Scott, S., Pantelic, V., Wassyng, A., Lawford, M. Applying Modular Decomposition in Simulink. In: 2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), pp. 31-36. IEEE, Coimbra, Portugal (2020) https://doi.org/ 10.1109/ISSREW51248.2020.00033
[6] Jaskolka, M., Pantelic, V. Wassyng, A. Lawford, M. Supporting Modularity in Simulink Models. arXiv (2020). https://doi.org/10.48550/ARXIV.2007.10120
[7] Matsim: A sleek, intuitive interface for building Simulink models from a Matlab script, https://github.com/gave92/Matsim , last accessed 2022/09/18.
[8] Gula, M., Zakova, K. Matlab Adapter – Online Access to Matlab/Simulink Based on REST Web Services. In: Proceedings of the 4th Computer Science On-line Conference 2015 (CSOC2015), Vol 2: Intelligent Systems in Cybernetics and Automation Theory, pp. 199–206. https://doi.org/10.1007/978-3-319-18503-3_20

Motion Control of the Angular Type Educational Manipulator Using Visual Components

Igor Nevliudov[1] [0000-0002-9837-2309], Sergii Novoselov[1] [0000-0002-3190-0592], and Oksana Sychova[1] [0000-0002-0651-557X]

[1] Kharkiv National University of Radio Electronics, Kharkiv, 61166, Ukraine
sergiy.novoselov@nure.ua

Abstract. At present, an urgent task is the introduction into the educational process of automation technical means digital twins for debugging and researching the main properties of the real devices behavior in various environmental situations. In the process of creating and debugging programs for controlling the manipulator, it is necessary to be able to control the correctness of the execution of the given instructions by the manipulator. For this, the manipulator itself is most often used directly, loading a test program into it and checking the trajectory of its links. But it is more promising to use a digital twin of the manipulator with the possibility of simulating all the movements of its links for visual control of the movement of the working tool in the given coordinates. Usually, the task of 3D visualization requires the use of additional libraries and computing power. This paper proposes a method of software calculation and visualization of manipulator operation, which can be implemented in any visual programming language. The peculiarity of the method is that no additional libraries are used, and the calculation of nodal points of the manipulator is carried out by means of the chosen programming language in real time, using the law of inverse kinematics.

Keywords: G-code, inverse kinematics, stepper motor, microcontroller, manipulator, control software, protocol

1 Introduction

Manipulator motion control software systems are designed to create programs for controlling the movement of manipulator links, remote control of the device, and visualization of the current state of moving mechanisms [1], [2], [3], [4]. The main task of the software tool is to facilitate the process of creating control programs and increase productivity by visualizing the movement of mechanical moving parts of the manipulator.

When creating control programs, the characteristics of the specific type of manipulator for which they are created are taken into account. The three basic functions of data transformations are aimed at solving three standard configuration problems of manipulator kinematics with the protection of their solutions against dangerous movements of the manipulator:

- Conversion of the angular configuration of the manipulator links into the Cartesian coordinates of the selected point on the gripper axis.
- Conversion of the target coordinates of the manipulator with the target parameters of the gripper into the angular configuration of the manipulator links at the target point.
- Linear interpolation of the movement in the Cartesian coordinates of the target vector according to the given values of the angular configurations in the current and target points of the planned movement of the manipulator gripper.

The purpose of the work is to create a software tool for controlling the movement and simulating the operation of angular type manipulators using visual components.

2 Analysis of Manipulators Designs

The training model of the manipulator robot serves as the control object. The manipulator contains two movable joints and can rotate around a vertical axis. Also, the manipulator has a gripper for grabbing and moving parts within its working area.

The design is based on three stepper motors. Each stepper motor implements a certain degree of freedom. The motors are controlled by a control module based on the Arduino Mega controller.

The manipulator has end sensors, one for each degree of freedom. At the beginning of work, the initial initialization of the control system is performed. At the same time, a test start of each stepper motor is performed and the operation of the corresponding end sensor is monitored. If all the sensors have worked, the device goes into standby mode for commands from the user. Fig. 1 shows the kinematic diagram of the manipulator.

The design is based on three stepper motors. Each stepper motor implements a certain degree of freedom. The motors are controlled by a control module based on the Arduino Mega controller.

The manipulator has two main links *L1* and *L2*, as well as two rotary-type kinematic pairs operating in the same plane. By rotation the angles $\varphi 1$ and $\varphi 2$, the kinematic pairs are moved.

By rotating the manipulator by an angle $\varphi 3$ relative to the *Y* axis, movement in three-dimensional space is performed.

The first link L1 is attached to the base and rotates by an angle φ1, and the second link *L2* is attached to the end of the first and rotates relative to it by an angle $\varphi 2$. The working body of the manipulator is located at the end of the second link. Thanks to this design, heavy servo modules can be placed in the lower part of the manipulator. This will reduce the weight of the mechanism itself and, due to this, increase the weight of objects that can be moved.

Support bearings are used to hold the platform in the horizontal plane. The main gear rests on the bearings, which is driven by a stepper motor using a belt transmission.

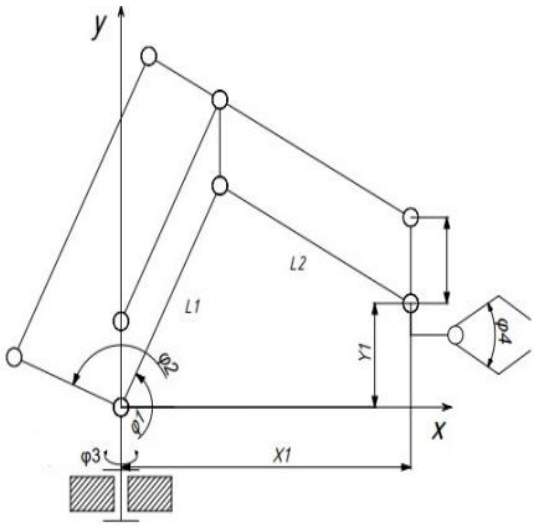


Fig. 1. Kinematic scheme of the manipulator

The simplified functional scheme of an industrial robot is shown in **Fig. 2**. An industrial robot consists of a mechanical part (which contains one or more manipulators) and a control system for this mechanical part. In addition, the robot may have means of feeling (which together form an information-sensory system), the signals of which are sent to the control system [5], [6].

A manipulator is a controlled mechanism (or machine) that is designed to perform motor functions similar to the functions of a human hand when moving objects in space, and is equipped with a working body. In some cases, an industrial robot may include two (or more) manipulators [7].

The executive mechanism of the manipulator is, as a rule, an open kinematic chain, the links of which are connected in series by joints of various types. The combination and mutual arrangement of links and joints determines the number of degrees of freedom, as well as the scope of the robot's manipulation system. The executive mechanism of the manipulator is, as a rule, an open kinematic chain, the links of which are connected in series by joints of various types. The combination and mutual arrangement of links and joints determines the number of degrees of freedom, as well as the scope of the robot's manipulation system:

- Robots operating in a Cartesian coordinate system, i.e. robots in which all three initial joints are translational (for example, the IBM RS-1 robot [8]).
- Robots that work in a cylindrical coordinate system, that is, robots in which among the initial articulations there are two translational and one rotational joints (for example, the Prab Versatrans 600 robot [9]).
- Robots that work in a spherical coordinate system, i.e. robots that have one translational and two rotational joints among the initial joints (for example, the Unimate 2000B robot from the Unimation company [10], [11]).

- Robots that work in an angular or rotational coordinate system, that is, robots in which all three initial articulations are rotational (for example, PUMA robots from Unimation or T3 from Cincinnati Milacron [11]).

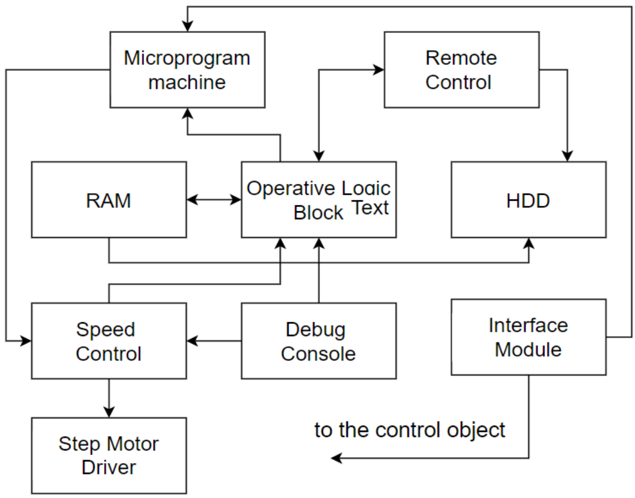


Fig. 2. Functional scheme of industrial robot

In some cases, the manipulator of an industrial robot is installed on a movable base, which means providing it with additional degrees of freedom. For example, the manipulator can be installed on rails or on a movable carriage that moves along a floor track or along suspended guides.

There are also industrial robots with closed kinematic chains. An example can be parallel robot – manipulation robot in which the working body is connected to the base by at least two independent kinematic chains. This class of manipulation robots includes, in particular, the Hugh-Stewart platform and delta robots [12], [13], [14].

There are also industrial robots with closed kinematic chains. An example can be parallel robots – manipulation robots in which the working body is connected to the base by at least two independent kinematic chains. This class of manipulation robots includes, in particular, the Hugh-Stewart platform and delta robots.

3 Discussion of similar decisions

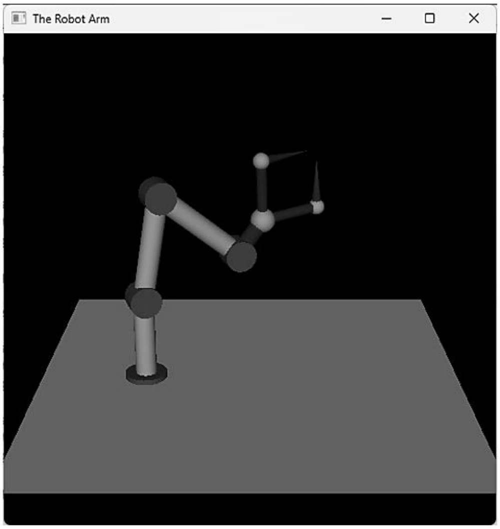
The task of controlling the movement links manipulator with simultaneous visualization of movement is of great importance. This task is especially acute at the stage of debugging the technological program. Several solutions can be used to visualize the movement of manipulator links. Currently, the most common graphics tools for this are OpenGL and Unity.

In work [15], a comparative analysis of the performance of these two tools is carried out to solve the problem of simulating the processes of transformation of 3D graphic objects of different complexity. The simulation took place in real time. Modeling was carried out using a simplified mass-spring method to realize the deformation of objects.

The work presents the results of experimental studies for various conditions of use. The number of simulations is 20 times, and performance was calculated by averaging the time spent on the task. A value of 200 frames was chosen as the boundary for simulation.

Objects such as a sphere, the letter “A” and the letter “L” were used for modeling. The result of experimental studies showed that the usual OpenGL and shaders, which were written using GLSL, are 12 times faster than Unity. The authors explain these results by the fact that Unity not only models and performs calculations, but also provides development environment functions for programmers and users. Thus, the authors recommend using OpenGL to perform the tasks of modeling 3D objects in visualization programs.

Another work that was analyzed is [16], which is an example of the implementation of the task of modeling an industrial manipulator with a grip by means of OpenGL using the C# programming language (**Fig. 3**).




```
static void DrawRobotArm(int numSegs)
{
    GL.glMatrixMode(GL.GL_MODELVIEW);

    GL.glTranslatef(baseTransX, 0, baseTransZ);
    GL.glRotatef(baseSpin, 0f, 360f, 0f);
    DrawBase(64);

    GL.glTranslatef(0, 0.4f, 0);
    GL.glRotatef(shoulderAng, 0f, 0f, 90f);
    DrawArmSegment(64);

    GL.glTranslatef(0, 0.5f, 0);
    GL.glRotatef(elbowAng, 0f, 0f, 90f);
    DrawArmSegment(64);

    GL.glTranslatef(0, 0.5f, 0);
    GL.glRotatef(wristAng, 0.0f, 0f, 90f);
    DrawWrist(16);

    GL.glRotatef(wristTwistAng, 0.0f, 180f, 0f);

    GL.glPushMatrix();

    GL.glTranslatef(0f, 0.2f, 0f);
    GL.glRotatef(fingerAng1, 0f, 0f, -180f);
    DrawFingerBase(16);
```

Fig. 3. An example of implementing a 3D manipulator model using OpenGL

The peculiarity of this implementation is that the visualization takes place for the already calculated coordinates of the placement of nodes of the manipulator links. Added a function to the source code to determine the metric of the time it takes to perform the rendering. On average, this time is 0.4 ms for the test computer.

4 Development of the Software Structure Diagram

The structure diagram of the software, presented in **Fig. 4**. The main modules of the program are: the main module, the program execution module, the command transfer module, the database module, and the visualization module. Also, six timers are used in the program, which allow to implement independent flows of controlling the progress of the program.

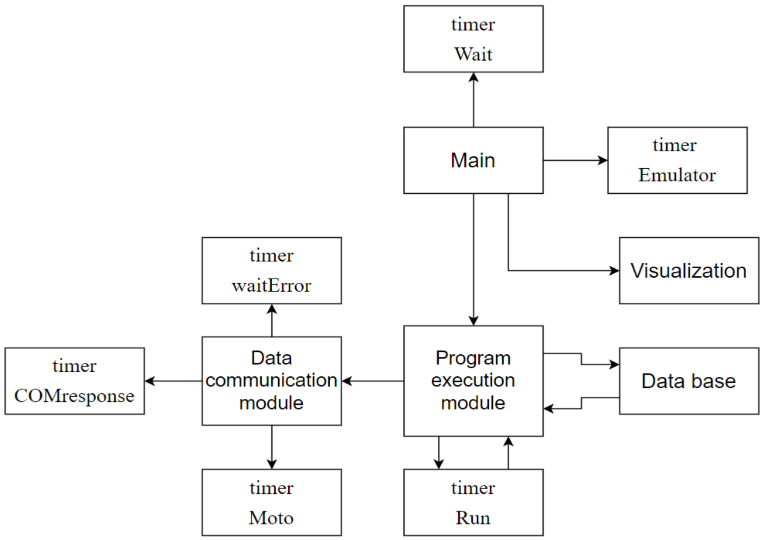


Fig. 4. Structure diagram of the software

The program execution module organizes work with instructions stored in the database. The corresponding module is used to receive the next command from the database. Each team is selected sequentially. The queue is controlled by an independent flow, which is responsible for the timer_Run timer. After receiving the next instruction, a command in *G*-Code format is prepared. The following commands are involved in the program: position the tool *G0*, turn on the gripper *M5*, turn off the gripper *M3*, turn on the pump *M1*, turn off the pump *M2*, turn on the laser *M6*, turn off the laser *M7*, turn on the power of the motors *M17*, turn off the power of the motors *M18*, calibrate the manipulator *G28*.

The visualization module is used in the emulation mode of the work program to visually control the position of the manipulator links to facilitate the debugging of the control program.

Part of the program interface is intended for placing manipulator control commands. Each command is represented by a separate line that can be dragged within a program segment, thus changing the flow of program execution. In the other part, you can edit the parameters of each command. The content of this interface area depends on the specific instruction. In total, the program provides six different types of commands: Move To, While, Repeat, IF, GoTo, WaitInput.

The “Move To” command is used to move the working tool of the manipulator to a given point in space. This command requires the assignment of the following parameters: X movement coordinate, Y movement coordinate, Z movement coordinate, R movement coordinate, S movement speed. XYZ coordinates specify the position of the end point of the tool movement in 3D space. The R coordinate is a parameter that determines the displacement of the manipulator along the rail to which it is attached (if available).

In the emulation mode, a list of nodal points is provided. This list allows you to remember the key positions of the manipulator, which are often repeated during its operation. This list can be updated during the program operation. Points can be added or deleted from the list. All of them are stored in the corresponding database table. Each point can be given a unique name so that it can be conveniently identified during work.

Also, in this mode, you can move the manipulator to a given point without executing the full program. Thus, the operator has the opportunity to debug each step of the program in the process of its creation in real time.

In order to read data from the manipulator's memory about the current state of its links and automatically fill the data fields with the corresponding real coordinates, a special button “Get current position” is provided.

The program also has a “Label for transition” item. This parameter is used to assign a unique identifier to the command, which is used to jump to it unconditionally from any line of the program when using the “GoTo” statement.

5 Simulation of the Graphic Image of the Manipulator Position in Two Projections

To model the position of the manipulator links and visualize their movement, two views of the device are used: a top view and a side view (**Fig. 5**).

The two specified views make it possible to evaluate the movement of the manipulator without using the isometric view of the coordinate system. The difficulty of solving the problem lies in the fact that the manipulator can rotate around a vertical axis, so the side view will represent a transformed view in the vertical plane of vision.

The calculation of the position of the manipulator links, as well as the coordinates of the joints, is calculated in several stages.

The initial data for the calculation are the location coordinates of the manipulator base B(x, y, z) and the desired final coordinates of the tool T(x, y, z). At the first stage, we present the location of the manipulator in the XOY plane, which is a top view (**Fig. 6**).

In this view, the links of the manipulator form one line, from the base point *B(x, y)* to the end point *T(x, y)*. The center of the base of the manipulator (point *B*) also has an offset of ΔB_x, ΔB_y relative to the center of the graphic system coordinates. This should be taken into account when building a graphic image on the program form.

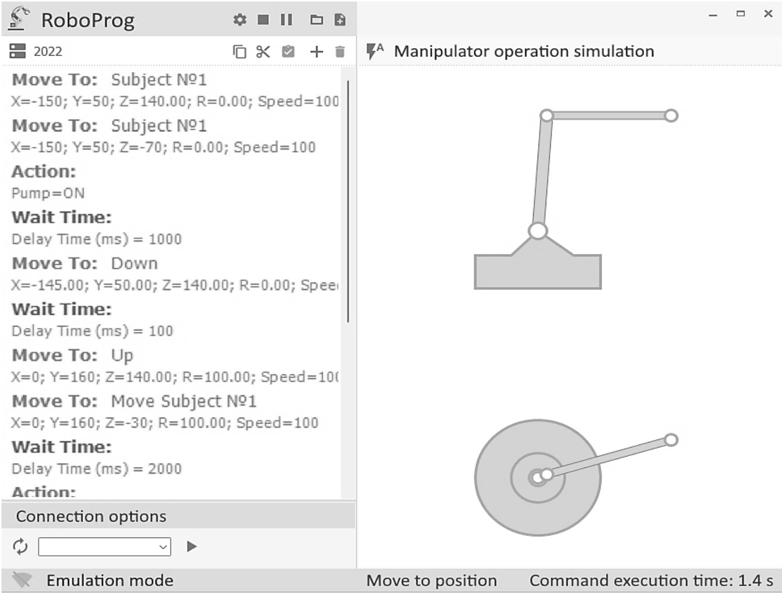


Fig. 5. The program interface is in the manipulator simulation mode

At this stage, we do not know the coordinates of the joint *M*, which is located between the base of the manipulator *B* and the end point *T*. To find these coordinates, it is necessary to first find the height of the location of the joint *M* in the *ZBX* plane (coordinate M_z), and then find the projection of the point *M* on the *X* axis, and this will be the M_x coordinate.

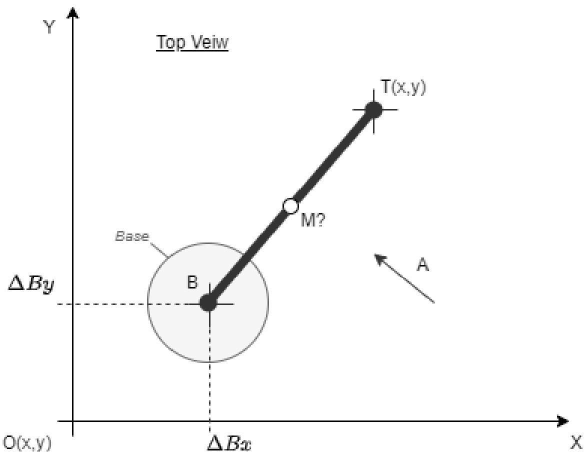


Fig. 6. Manipulator location in the *XOY* plane (top view)

To find the M_z coordinate, it is necessary to solve the problem of inverse kinematics. The task of inverse kinematics is to find such angles so that the end link reaches the desired target point. As part of our task, we will consider the location of the links of the manipulator on view A (**Fig. 7**).

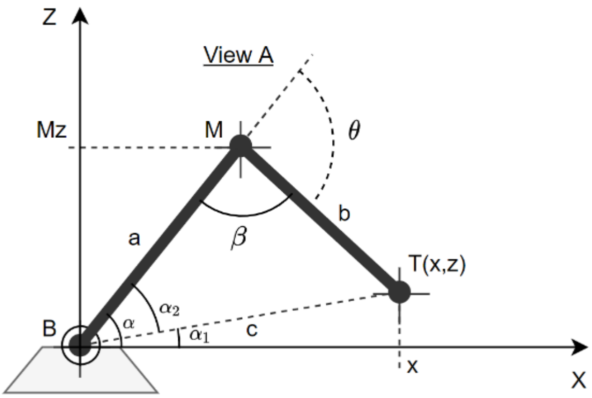


Fig. 7. Location of manipulator links in the *ZBX* plane

To solve this problem, we need to know the length of two links $BM = a$, $MT = b$. Let's draw a straight line c connecting the points of origin of coordinates *B* and the given *T(x, y)*. The length of the straight line c is determined:

$$c = x^2 + z^2 \tag{1}$$

Then the coordinates of the point *T(x, y)* can be written as follows:

$$T_x = c\cos(\alpha_1) \tag{2}$$

$$T_z = c\sin(\alpha_1) \tag{3}$$

where α_1 is the angle between line *c* and *x*-axis.
Angle α consists of two angles α_1 and α_2:

$$\alpha = \alpha_1 + \alpha_2 \tag{4}$$

We find the angle α_1:

$$\alpha_1 = arctg(y/x) \tag{5}$$

We find α_2 using the theorem of cosines (for a triangle with sides *a*, *b*, *c* and an angle opposite to side α):

$$\alpha_2 = b^2 + c^2 - 2bc \cdot \cos(\alpha) \tag{6}$$

From (6), you can write the expression for finding the angle α_2 :

$$\alpha_2 = \cos^{-1}\left(\left(a^2 - b^2 + c\right)/2ca\right) \tag{7}$$

where *a* is the shoulder of *BM* (**Fig. 7**), *b* is the shoulder of *MT*; *c* is the *BT* arm, which is calculated according to (1).
Similarly, we find the angle β :

$$\beta = \cos^{-1}\left(\left(a^2 + b^2 - c\right)/2ab\right) \tag{8}$$

The angle θ is defined as:

$$\theta = 180 - \beta \tag{9}$$

Knowing all the current angles of the manipulator kinematics, it is possible to determine the M_x coordinate of the middle joint:

$$M_x = B_x - BM \cdot \cos\left(\pi\alpha/180\right) \tag{10}$$

The M_z coordinate is calculated similarly:

$$M_z = B_z - BM \cdot \sin\left(\pi\alpha/180\right) \tag{11}$$

The next step is to determine the angle of displacement of the manipulator links when rotating around the vertical axis *Z* in the top view in order to further obtain the projection of point M on the X coordinate axis (**Fig. 8**).

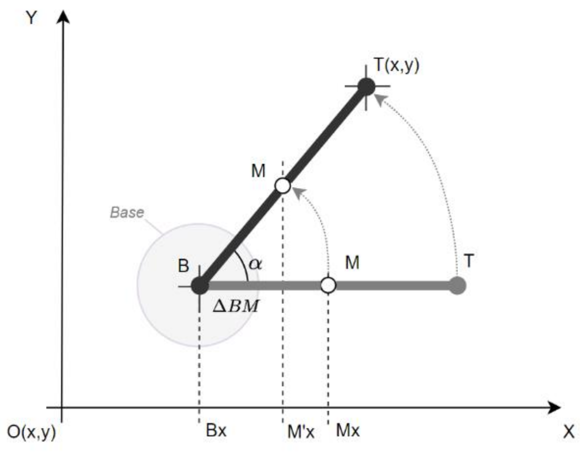


Fig. 8. The principle of displacement of point *M* when rotating the manipulator around the *Z* axis

To solve this problem, let's imagine that the line *BT* together with the known displacements of the end point in coordinates *(x, y)* forms a right triangle *BT'T* (**Fig. 9**).

To determine the angle α, it is necessary to calculate the length of the segment *BT*. For this, we need to know the coordinates of points *B* and *T*. We know the coordinate *B*, it is the coordinate of the location of the base of the manipulator. The coordinate of point *T* is also known to us as the end coordinate of the tool movement in the *XY* plane.

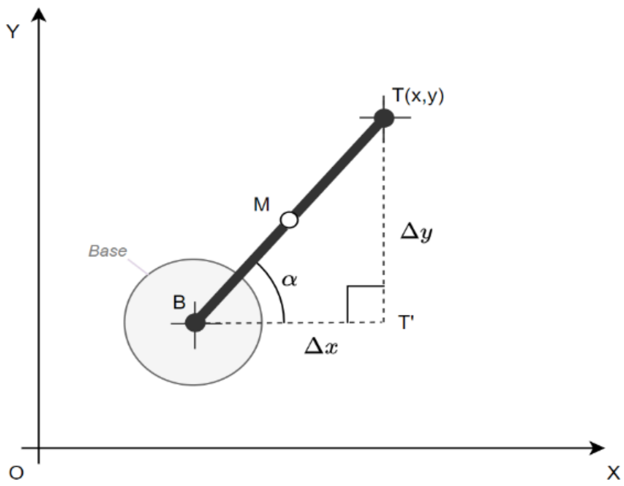


Fig. 9. Determination of angle *α*

Thus, the length of the segment *BT*, taking into account the coordinates of the two points, will be equal:

$$BT = \sqrt{(T_x - B_x)^2 - (T_y - B_y)^2} \tag{12}$$

Now let's calculate the rotation angle α :

$$\alpha = \sin(\Delta y / BE) \tag{13}$$

We find the projection of point *M* on the *X* axis (**Fig. 8**):

$$B_x T_x^{'} = (T_x - B_x)\sin(\alpha) \tag{14}$$

Now you can display the position of the manipulator links in the side view (**Fig. 10**).

The last step is to determine the position of point *M* on the segment *BT* in *XY* coordinates in the top view. To do this, you need to find the point of intersection of two straight lines:

- the first line passes through points *B* and *T* in the top view;
- the second line passes through points *M* and *T'* in the side view.

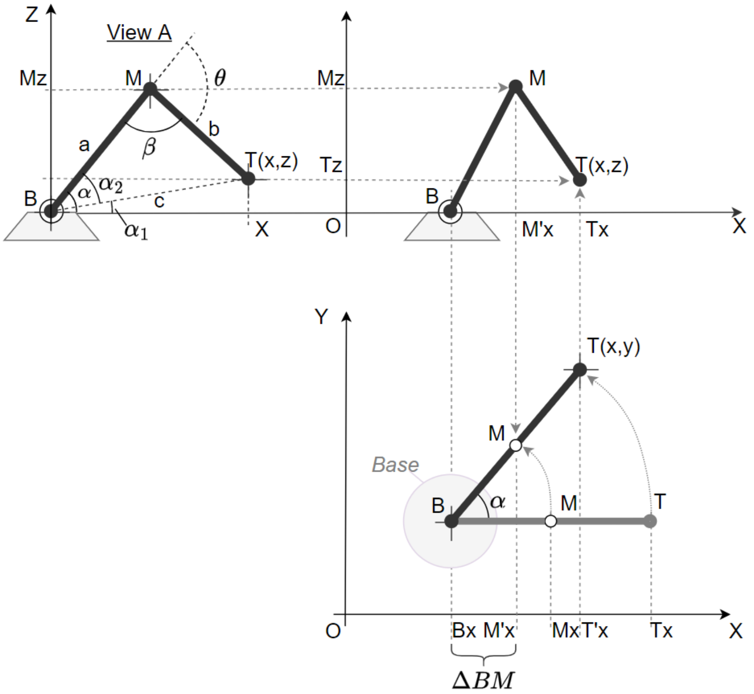


Fig. 10. The stages of determining the coordinates of the manipulator joints in two projections

Knowing the coordinates of the segments *BT* and *MT'* from the classical notation of the equation of lines

$$(x-x_1)/(x_2-x_1)=(y-y_1)/(y_2-y_1) \tag{15}$$

it is necessary to go to the equation in one row:

$$x(y_2-y_1)+y(x_1-x_2)-x_1y_2+y_1x_2=0 \tag{16}$$

or

$$ax+by+c=0 \tag{17}$$

where $a=y_2-y_1$, $b=x_1-x_2$, $c=-x_1y_2+y_1x_2$.
From here, the *x*-coordinate of point *M* will be written as follows:

$$x=(b_1c_2-c_1b_2)/(a_1b_2-a_2b_1) \tag{18}$$

The *y*-coordinate of the point *M* is determined similarly:

$$y=(a_2c_1-c_2a_1)/(a_1b_2-a_2b_1) \tag{19}$$

6 Analysis of test results

As a result of performance testing of the proposed method of simulating the movement of manipulator links, the average value of the operation execution time was obtained – 0.4 ms. For comparison, the method [16] based on the use of the OpenGL library was taken. For this method, the average rendering time is 0.395 ms. **Fig. 11** shown the results of the testing performed for 200 frames.

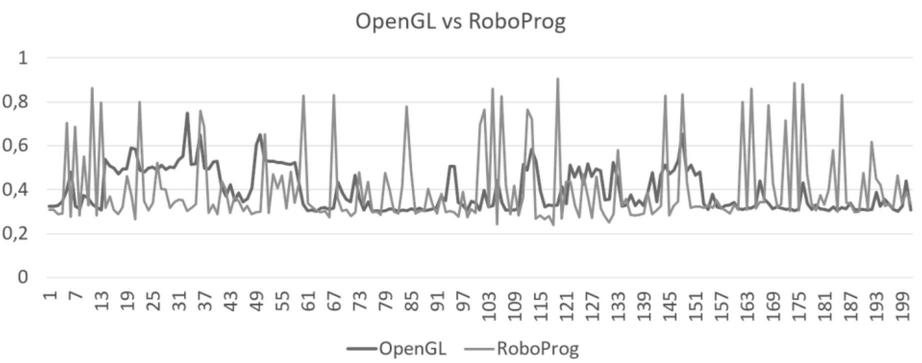


Fig. 11. The result of comparing the performance of two visualization methods

It can be seen from the given figure that the studied visualization method has a number of drops of measured data, which is clearly visible in **Fig. 11**. This is due to the simultaneous parallel operation of several processes in the manipulator motion control program: the process of determining the current position of the manipulator links with the solution of the inverse kinematics task, the process of processing the program counter, the dues of program cycles, and others.

7 Conclusions

This work describes the creating method an industrial manipulator digital twin for visualizing movement of the links angular manipulator using visual components. The object of control is the training model of the manipulator robot. The manipulator has two movable joints and can rotate around a vertical axis. Also, the manipulator has a gripper for grabbing and moving parts within its working area. A method of calculating the position of the manipulator links and determining the coordinates of the joints is proposed. The peculiarity of the developed method is that no additional libraries are used, and the calculation of nodal points of the manipulator is carried out by means of the selected programming language in real time, using the law of inverse kinematics.

The obtained results show sufficient performance of the proposed visualization method, which is within the speed of image processing using the OpenGL library, and in some cases, even exceeds it.

References

[1] Nair S., Rajeswaran A., Kumar V., Finn C., Gupta A. R3M: A Universal Visual Representation for Robot Manipulation, arXiv:2203.12601v2 [cs.RO], 18 Apr. 2022. https://arxiv.org/abs/2203.12601.

[2] Novoselov S., Sychova O. Using Wireless Technology for Managing Distributed Industrial Automation Objects within the Concept of Industry 4.0, 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), 2019, pp. 580–584, doi: 10.1109/PICST47496.2019.9061333.

[3] Novoselov S., Sychova O., Tesliuk S. Development of the Method Local Navigation of Mobile Robot a Based on the Tags with QR Code and Wireless Sensor Network, 2019 IEEE XVth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH), 2019, pp. 46–51, doi: 10.1109/MEMSTECH.2019.8817405.

[4] Nevludov I., Sychova O., Andrusevich A., Novoselov S., Mospan D., Mospan V. Simulation of the Sensor Network of Base Stations in a Local Positioning System in Intelligent Industries, 2020 IEEE Problems of Automated Electrodrive. Theory and Practice (PAEP), 2020, pp. 1–6, doi: 10.1109/PAEP49887.2020.9240842.

[5] Ostanin M., Mikhel S., Evlampiev A., Skvortsova V., & Klimchik A. Human-robot interaction for robotic manipulator programming in Mixed Reality, 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 2805–2811, doi: 10.1109/ ICRA40945.2020.9196965.

[6] Inoue K., Nishihama Y., Arai T., Mae Y. Mobile manipulation of humanoid robots-body and leg control for dual arm manipulation, Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292), 2002, pp. 2259–2264 vol.3, doi: 10.1109/ROBOT.2002.1013568.

[7] Yi S.J. Software Framework for an Intelligent Mobile Manipulation Robot, 2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT), 2018, pp. 1–4, doi: 10.1109/ICT-ROBOT.2018.8549895.

[8] IBM Archives. IBM RS 1 robotic system. IBM – Deutschland | IBM. URL: https://www.ibm.com/ibm/history/exhibits/robotics/robotics_5. html (date of access: 28.07.2022).

[9] Cyberne1. 1958-62 – “VERSATRAN” Industrial Robot – Harry Johnson & Veljko Milenkovic – cyberneticzoo.com. URL: http://cyberneticzoo.com/early-industrial-robots/1958-62-versatran-industrial-robot-harry-johnson-veljko-milenkovic/ (date of access: 28.07.2022).

[10] Automate. Unimate – The First Industrial Robot. Automate. URL: https:// www.automate. org/ a3-content/joseph-engelberger-unimate (date of access: 28.07.2022).

[11] Gasparetto A., Scalera L. From the Unimate to the Delta Robot: The Early Decades of Industrial Robotics. In: Zhang, B., Ceccarelli, M. (eds) Explorations in the History and Heritage of Machines and Mechanisms. History of Mechanism and Machine Science, vol 37. Springer, Cham. 2019, https://doi.org/10.1007/978-3-030-03538-9_23.

[12] Becerra-Vargas M., Morgado Belo E., Application of H∞ theory to a 6 DOF flight simulator motion base. Journal of the Brazilian Society of Mechanical Sciences and Engineering. 2012, Volume 34, pp. 193–204. https://doi.org/10.1590/S1678-58782012000200011.

[13] Wu L., Zhao R., Li Y., Chen Y.H. Optimal Design of Adaptive Robust Control for the Delta Robot with Uncertainty: Fuzzy Set-Based Approach. Appl. Sci. 2020, 10, 3472. https://doi.org/10.3390/app10103472.

[14] Romano D., Bloemberg J., Tannous M., Stefanini S., Impact of Aging and Cognitive Mechanisms on High-Speed Motor Activation Patterns: Evidence From an Orthoptera-Robot Interaction, in IEEE Transactions on Medical Robotics and Bionics, vol. 2, no. 2, pp. 292–296, May 2020, doi: 10.1109/TMRB.2020.2977003.
[15] Choi M.H., Kim M.S., Sung N.J., Choi Y.J., Hong M. Comparison in Performance of Parallel Deformable Object Simulation between OpenGL and Unity, KSII The 9th International Conference on Internet (ICONI) 2017 Symposium, pp. 213–215, 2017.
[16] Maccaferri L., RobotArmOpenGLCSharp, URL: https://github.com/leniel/leniel.net (date of access: 12.11.2022).

Method of Restructuring Use Cases in the Process of the Information System Design

Oleksii Kungurtsev[1] [0000-0002-3207-7315], and Svitlana Zinovatna[1] [0000-0002-9190-6486]

[1]Odessa National Polytechnic University, 1 Shevchenko av., Odesa, 65044, Ukraine

akungurtsev19@gmail.com
zinovatnaya.svetlana@op.edu.ua

Abstract. Use cases are widely used as a means of formulating requirements in the development of information systems. All subsequent design stages depend on the quality of their presentation. Structuring use cases can significantly increase their understanding and maintenance in the face of changing requirements. Flexible technologies involve working in small teams. The existing communication between teams is not sufficient to highlight sub use cases at the project level. There is a need for automated analysis of the corpus of all use cases. A mathematical model of a use case which makes it possible to define the criteria for comparing scenarios and eliminate the redundancy of descriptions is proposed. A four-step method for restructuring use cases has been developed. At the first stage, use cases are presented in a formalized form. At the second, they are storedin the repository, which ensures their quick search and placement. At the third stage, procedures of scenario comparison are performed. Scenario similarity criteria are proposed. At the fourth stage, the formation of subordinate use cases is carried out, their texts are coordinated with all interested teams, and the use casesthat cause subordinate use cases are corrected. Experiments providing the formalized compilation of use cases by several development teams followed by automated restructuring were carried out to test the proposed solutions. As a result,new subordinate use cases were correctly identified and the scope of use of previously formed ones was expanded. There was a significant reduction in the timefor restructuring. The proposed method of restructuring use cases improves the clarity and consistency of requirements, the possibility of their adjustment and maintenance, and reduces the compilation time. The method can be used in the design of any information system, where the requirements are presented in the form of use cases.

Keywords: use case, subordinate use case, scenario, information system de-sign

1 Introduction

Use cases (UC) are the main way to represent functional requirements [1], and partially non-functional requirements [2] to the designed information system (IS). The quality of the entire project largely depends on the quality of UC writing. There are a number of recommendations for compiling UCs that relate to general issues of selecting UCs, ways of scenario recording in relation to the tasks solved, and formats for presenting

UCs. In [3], the concept of “subordinate UC” (SUC) is introduced to define UC, which is called from some step in the scenario of the main UC. Usually, SUCs are formed from extensions of the main UC. There are at least two reasons for this:

– the extension is used in several places. The formation of a SUC from it will simplify the maintenance of requirements and the code that implements them;

– expansion makes the main UC difficult to understand.

The SUC must be linked to the main UC by an include or extend relationship. Fig. 1 shows examples of such relationships.

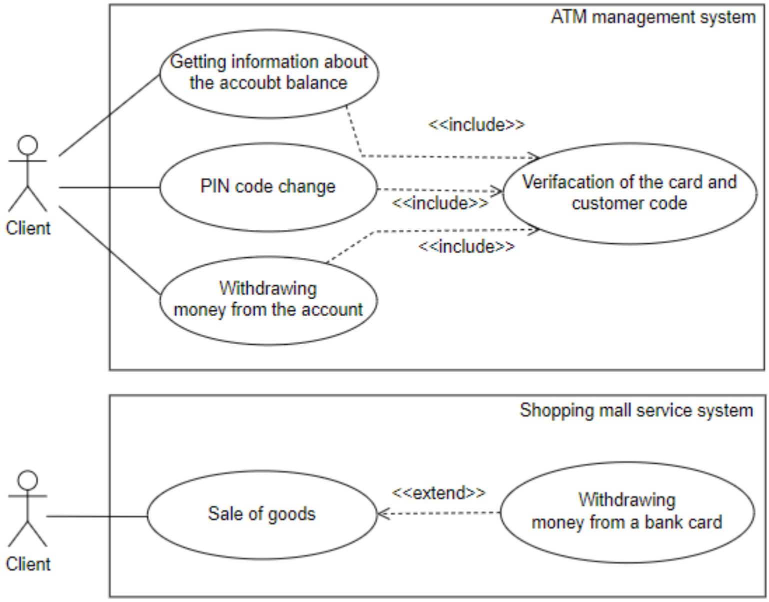


Fig. 1. Relationships between master and subordinate use cases

Within the framework of a large project, dozens or hundreds of UCs are formed by different development teams. Under such conditions, determining the identity of subordinate UCs selected in various subsystems, and, furthermore, finding repeating fragments of scenarios, is a very complex and time-consuming task. The problem becomeseven more difficult in the context of global software development [4].

In this paper, it is proposed to consider the problem associated with the allocation of SUC in a broader sense – the elimination of repetitive requirements and the code corresponding to them.

In a simplified form, the process of UC formation in the design of IS is shown in Fig. 2. At the system analyst level, a list of the main UCs can be generated. If several development teams are working on the project, then the allocation of SUC becomes possible only within some parts of the project.

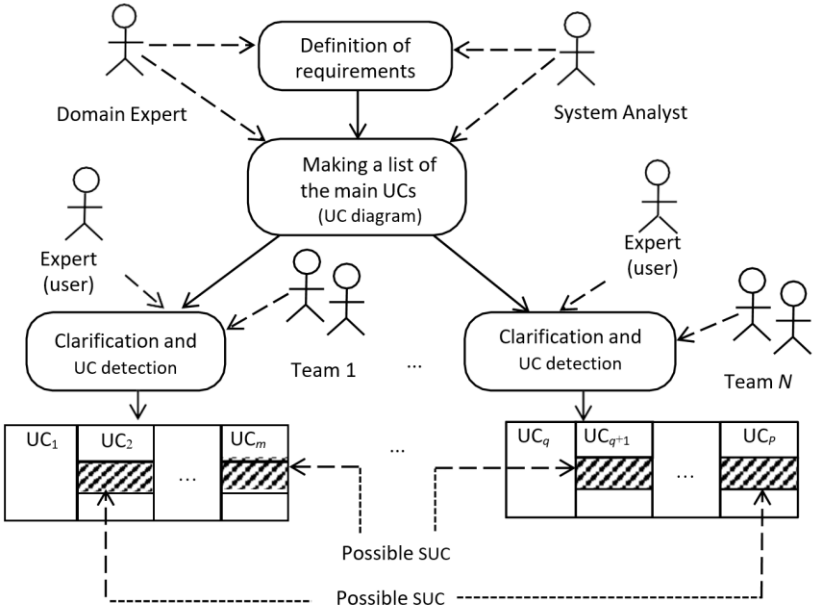


Fig. 2. The process of generating use cases

The purpose of the work is to improve the quality of presentation of functional requirements in the form of use cases by eliminating the redundancy of descriptions and introducing UC structuring.

To achieve the said goal, it is proposed to solve the following tasks:

1. to create a mathematical model of UC that makes it possible to compare fragments of their scenarios;
2. to develop a method for UC restructuring;
3. to test the study results.

2 Review of the literature

The need to improve the description of UC for their use at various design stages is indicated in [5].

In [6], it is noted that UCs should be the main tool for communication and verification of requirements by the user. However, the authors believe that expanded class diagrams can be a good mechanism for communicating and checking requirements. In our opinion, first, it is necessary to perform structuring of the UC, which in the future will ensure the construction of high-quality class diagrams.

Studies carried out within the framework of the energy project [7] have shown the effectiveness of creating a repository for more than 50 UCs. The authors note that this created the conditions for solving a number of tasks of project progress control, documentation management, profitability and safety improvement. In our opinion, the functions of the repository can be extended with the tasks of analysis and UC restructuring. The authors propose a formal model of the UC diagram followed by a multi-view consistency check. However, the authors do not consider the distribution of functions between UCs.

The need for further formalization of requirements was noted in [9]. It is proposed to use a structured natural language and the corresponding FRET tool. In our opinion, formalization should be introduced wherever there are conditions for this, for example, when forming UC. The problem of presenting UCs with varying degrees of detail is considered in [10]. The complexity of this task is noted and a number of recommendations for "slicing" UC in Agile technologies are given. However, the authors do not propose a formal model for the UC refinement process.

In [11], the problem of low-quality specification of requirements is noted. As a solution, it is proposed to use document templates compiled on the basis of the experience of successfully completed projects. We believe that the use of templates will be especially effective in the formation of UC and SUC. Such templates were proposed in [12] as part of building a model of conceptual classes based on an automated description of UC.

In [13], the influence of natural language on the quality of work with requirement specifications is noted and it is recommended to use natural language processing tools. In [14], it is proposed to improve the quality of project documentation by expanding the use of formal methods for its presentation. This problem extends to a large extent to the description of UC. Both natural language processing tools and the use of special templates that reduce the ambiguity of text fragments can be its solution. Models of UC scenario steps [15] can be a solution to this problem.

3 Materials and methods

Selection of SUC requires comparison of UC scenarios. It is possible to organize a comparison of scenarios of all UCs to select matching sequences of steps. However, since UCs are compiled by different developers, the probability of finding matching texts is negligible. We can use fuzzy string comparison [16] by introducing certain matching coefficients. However, a very low signal-to-noise ratio is expected in this case too. To solve the problem, it is proposed to introduce certain formalization in the description of UC, expressed in the following steps.

7. When describing all UCs, use a unified classification of scenario steps [12].
8. When describing all UCs, use a unified system of generalized data typing [15]. Let us represent UC in the form of a tuple:

$$UC = \langle id, ucType, mP, mES, mRef, refT \rangle, \quad (1)$$

where id is the UC identifier; ucType – a UC type, can take two values: main (for main UCs) or subordinate (for subordinate UCs); mP = (p1, p2, ... , pn) – a set of steps of the main scenario ordered by numbers; *mES* – a set of alternative scenarios; *mRef* is a set of references of the subordinate UC to the UC that access it; *refT* is a reference to the command that accompanies the UC.

The mRef parameter makes it possible to determine the efficiency of using the subordinate UC. For main UC $mRef = \varnothing$. In the main UC links to subordinate UCs are provided by a special scenario clause.

Let us represent each alternative scenario as a tuple:

$$UC = <id, ucType, mP, mES, mRef, refT>, \quad (2)$$

where ep_j is the number of the main scenario step from which the transition to the alternative scenario takes place; rp_k – the number of the main scenario step, to which thereturn from the alternative scenario takes place; cA – condition of transition to an alternative scenario; $mAP = (p_1, p_2, ... , p_k)$– a numbered set of extension scenario steps.

If $rp_k = 0$, then UC ends.

Use case restructuring method provides for the following steps.

Stage 1 – Formalized presentation of UC scenarios

In the representation of the step of the main scenario, extension scenario or SUC, we will indicate its type. The following types of UC steps are proposed in [12]: Create, Enter data, Request a value, Request a list of values, Select from a list, Request a service, Request with a value, Repeat actions, Successful completion of the UC, Failure of the UC, Call of the UC.

For each step type, a model has been compiled that makes it possible to formalize and automate the formation of the step. As an example, the model of the "Value Request" step is given. The user asks the system for some data. This is usually followed by an evaluation of the obtained data by the user. The step description template looks like this:

$$requestValue = <nP [, Client, tp_1, tu_1], Actor, tp_2, tu_2, data_1$$
$$[, tp_3, tu_3, data_2][, tp_4, tu_4, data_3, tp_5], \quad (3)$$
$$tp_6, data_1[, tp_7]>$$

where nP is the number of the scenario step; *Client* – an optional element (introduced when it is necessary to specify who the initiative comes from); *Actor* – the system user who will perform the said UC step. For them, the developer specifies the position.

The elements of the template can be pre-composed pieces of text (tp_i), pieces of text formulated by the developer (tu_j), and data that is input and/or output from the $data_k$ system. The $data_k$ element is formed from two components: the name of the data and its type. This information is used only by the developer and is not visible to the user. Square brackets enclose optional template elements. Below are the values of the template elements:

– tp_{one} = "wishes to receive";
– tu_{one} – is formed by the developer, for example, "repair cost ...";
– tp_2 = "requests the system";
– tu_2 – is formed by the developer, for example, "repair cost ...";
– $data_{one}$ – data that is requested from the system;

– *tp*$_3$ = “based on”;
– *tu*$_3$ – is formed by the developer, for example, “car brands ...”;
– *data*$_2$ – data on the basis of which the requested value is determined;
– *tp*$_{four}$ = “subject to”;
– *tu*$_{four}$ – is formed by the developer, for example, “availability of spare parts ...”;
– *data*$_3$ – data on the basis of which the fulfillment of the condition is checked;
– *tp*$_5$ = “The system confirms the fulfillment of the condition” – an optional element;
– *tp*$_6$ = “The system outputs”;
– *tp*$_7$ = “Client/Actor agrees”.

Service request step template:

reqService = <nP [, Client, tp_1, tu_1], Actor, tp_3, tu_2, {$data_1$},tp_4 [, tp_5, tu_3]>,

where tp = “wishes”; tu1 is a text that identifies the service (e.g. “undercarriage overview”) or document (e.g. “application for a reduced rate”); tp3 = “enters”; tu2 – a phrase that is formed by the user, the name of the service or document; data1 – service or document representation in the project; tp4 = “The system confirms the possibility of performing the service (document)”; tp5 = “Transfer of the control to scenario step”; tu3 – scenario step number.

To formalize the representation of input and output data, the following set of generic types is proposed:
– *List* – list (can represent a linear list, an array, a set, etc.);
– *Struct* – the structure (in the general case it contains fields of different types),must contain the numbering of the fields;
– *Text* – any text;
– *Numb* – any number format;
– *Bool* – boolean value;
– *Void* – the function does not return the value;
– *PClass* – a reference to a class object;

From the point of view of SUC selection, the step of the type “Request for a service”has a special meaning. It may be followed by steps (subordinate), revealing the mechanism for providing the service. It is this sequence of steps that can be a candidate fora SUC formation. To formalize the semantic relationship between steps, it is proposedto introduce link indicators in the form of step numbers into the texts of steps belongingto the “Service Request” group.

Example 1. A scenario fragment that implements registration will look like this:

N. [0] The client wishes to register in the system. The system confirms the possibility of registration.

N + 1. [N] The system displays the registration conditions. The client agrees.

N + 2. [N] The system suggests entering an email. The client enters. The system confirms the completion of the registration.

Taking into account the considered types of scenario steps and data, we will represent the scenario step in the form:

$$p = \langle nP, pH, pType, pText, mData \rangle, \quad (4)$$

where *pH* is the head step number, possible values: empty string, integer; *pType* – step type; *pText* – step text; *mData* is a set of data entered into the system or received from it. In accordance with the accepted classification, a set can contain up to three data. Each data has the form

$$mData_i = \langle dName, dType \rangle, \quad (5)$$

where dName is the name of the data; dType is the type of the data.

Stage 2 – Placement of the UC in the repository

To simplify the performance of operations with UC (storage, structuring, tracking changes), a repository is created.

Using queries to the repository database (Fig. 3), it is possible to organize the set of all UCs, divided into subsets depending on the commands that work with individual UCs. The common part containing SUC is also selected.

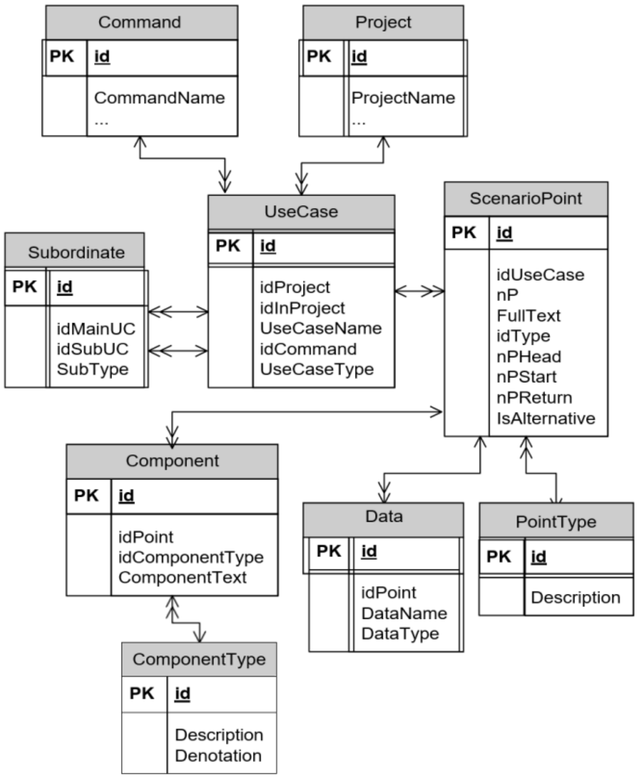


Fig. 3. Repository data model

Stage 3 – Comparison of UC scenarios

Selection of subordinate UCs can be performed at the level of the development team and at the project level.At the level of the development team, it is possible to select subordinate UCs fromthe set of UCs that this team is engaged in. We call such a SUC local. Verification andapproval of such a selection should occur within the team itself.

At the project level, it is necessary to determine the possibility of using the local SUC of a certain team for UCs developed in other teams. At the same time, the analysis of the possibility of using the SUC should be determined by the team that is proposed to use the SUC. In addition, considering the entire corpus of UCs, it is necessary to ensure the possibility of identifying SUCs that have not previously been created at the level of individual teams.

If a team has selected a SUC within their part of the project and it turns out that the SUC can be used in other parts, then the respective teams must confirm its use in their parts. If a new SUC is selected, then all teams where it will be used must confirm the possibility of its application within their part of the project. Each SUC must have a link to the team (developer). The main operation of the process is finding the occurrence of one scenario into another. Let us determine the main options for comparison:

– the identity of two subordinate UCs is established when the conditions for their call coincide and their sequences of steps coincide;
– a subordinate UC can be selected from two main UCs if a certain common sequence of steps in the scenarios of two main UCs with a length of at least 2 points is determined;
– the entry of a subordinate UC into the UC is fixed if all the points of the subordinate UC coincide with a part of the sequence of UC steps.

A group of semantically related steps should not be split into parts when compared. However, the order of substeps in a group can be arbitrary.

Example 2: A scenario fragment that implements registration in a sequence different from the one in Example 1:

N. [0] The client wishes to register in the system. The system confirms the possibility of registration.

N + 1. [N] The system suggests entering an email. The client enters. The system confirms its correctness.

N + 2. [N] The system displays the registration conditions. The client agrees. The system confirms the completion of the registration.

Let us formulate the conditions for the coincidence of two steps from different scenarios.

1. Step types p_i and p_j must match ($pType_i = pType_j$).
2. The actual values of Client and Actor are not compared (the same use case can be performed by different executors in different subsystems of the same project).
3. It follows from the scenario step model (3) that the text of a step of a certain typecan have different spellings due to optional elements and elements formed by the developer. Since these elements are important for the specific implementation of the steps,the necessary condition for the steps to match is the identity of their structures.

4. Any step of the scenario, except for the points of repeating actions and calling the SUC, provides for the performance of certain operations in the system of the form: creating an object, entering or receiving data, possibly, if certain conditions are met. Therefore, it is necessary to compare all text fragments formulated by the developer (tu_j).
5. The data that is input, output or created within the framework of the scenario step,in accordance with the template, must have a name and type. Both of these parameters are subject to comparison.

To determine the coincidence of two scenario items, it may be necessary that some elements are identical (we denote this operation as≡) and incomplete or fuzzy (we denote this operation as ≅). The result of a fuzzy match is the value of the similarity coefficient K. In what follows, we will consider the elements similar if their similarity coefficient is not less than a certain threshold value ($K \geq K_{min}$). Thus, we obtain the condition for the coincidence of points p_i and p_j, which belong to scenarios S_1 and S_2, respectively. Here $\in_s$ denotes the operation of an item belonging to a UC scenario.

$$(p_i \in_s S_1) = (p_j \in_s S_2) \text{ if } ((pType_i = pType_j) \\ \wedge (eList_i \equiv eList_j) \wedge (editText_i \cong editText_j) \\ \wedge \exists((dType_{i,k} = dType_{j,k})\ k = 1,n)) \tag{6}$$

where eList is a list of template elements of a scenario step; editText – texts of scenario step edited by the developer.

One step of the main scenario, possibly an alternative one, can have several extension scenarios. In order for the compared items to match, their alternative scenarios must also match. The number of alternative scenarios for the compared steps must match, but the order in which they are written can be arbitrary.

Let us formulate the conditions for the coincidence of semantic groups of steps. If a step pi is found for which pH = “[0]”, then an ordered set of steps of the subordinate group gi = (pk | pk · pH = “[i]”)should be formed.

If a step pj is found in some other UC, such that pj ≅ pi ∧ pj· pH = “[0]”, then a subordinate group gj = (pq | pq · pH = “[j]”) is created for it.

To forma SUC based on steps pi and pj, it is necessary for the number of elements in the groups to be the same | gi | = | gj | = n and for each step from the set gi, a matching step was found in the set gj $pk \cong pq$; $k = \overline{1,n}$; $q = \overline{1,n}$.

Let us formulate the conditions for the coincidence of extensions for points pi and pj with ni and nj alternative scenarios, respectively. First of all, the conditions for switching to an alternative scenario must match:

$$cA_{i,p}: ((cA_{i,p} \cong cA_{j,k}), p=1, n_i; k=1, n_j)) \wedge (n_i = n_j) \\ \exists cA_{i,p}: ((cA_{i,p} \cong cA_{j,k}), \\ p = \overline{1, n_i}; k = \overline{1, n_j}) \wedge (n_i = n_j) \tag{7}$$

Further, in accordance with (6), the steps of alternative scenarios are compared in pairs,for which condition (7) is satisfied.

If condition (6) is met the first time for scenarios S_1 and S_2, then a new scenario $S_{1,2}$ is created, the first step of which is $pS12_1 = p_i$.

If condition (6) is also satisfied for the next pair of steps $p_{i+1} \equiv p_{j+1}$, then step p_{i+1} is added to scenario $S_{1,2}$ and the process of scenario formation continues. Otherwise, the scenario $S_{1,2}$ is destroyed (there is only 1 step in the scenario).

It is proposed to evaluate the degree of coincidence $K_{i,j}$ of two scenarios S_1 and S_2 as an average coefficient of similarity of the steps included into them:

$$K_{i,j} = \frac{\sum k}{n} \tag{8}$$

Stage 4 – Selection of SUC and UC restructuring

The execution of the stage involves the following sequence of actions.

1. For each local SUC, its comparison with other local SUC is performed. In case of a match, the SUC is defined as global and a link to the support command is set in it. Local links are replaced with a global one.
2. For each global SUC, the possibility of its inclusion in the UC scenarios is determined. If possible, the UC scenario is edited accordingly. A link to the SUC is set in it.
3. For each UC, a comparison with other UCs (of different localization) is performed. If common parts are selected, then a global SUC is formed, a link to the support command is set in it, scenarios and links in the UC that have a common fragment are edited.

All operations for selecting a new UC or expanding the scope of its use are coordinated with the developer teams, which must introduce changes into the UC descriptions.

4 Results

To carry out the experiments, a document “Vision” for the development of an information system on the topic “Automation of the work of a clinic”, a list of users of the designed system and a list of UCs of 16 names were compiled. The developers were represented by 4 teams of 2 people. Each group received tasks to form 4 UCs in the UseCaseEditor. The groups were asked, if possible, to form a SUC in addition to the UC.

To test the results of the study, a software product that makes it possible to select SUC on the basis of the entire UC corpus in accordance with the developed methodology was developed.

The results were introduced into Table 1 after the discussion with all participants of the experiment. Symbols for UC and SUC were introduced in the table. For example, SUC (1) indicates that it was selected from UC 1 and is not used anywhere else. SUC (1-10-15)s indicates that in terms of content it is SUC (1)s, however, it was found out that it is a part of UC 10 and UC 15. A record of the form (2-6-10*)s means that SUC can be used for UC 2 and UC 6, but its use for UC 10 is a mistake.

The analysis of table data shows that at $K_{min} = 0.5$ the best results were obtained: the scope of SUC 1 and 7 was expanded by three UCs, and 2 new UCs were found.

During the experiments, the time spent on compiling UC and SUC was estimated. On average, 3.5 hours were spent on compiling 1 UC. It took 1.5 hours to select and compile one SUC, as well as adjust the UC within one team. The same work, but with unfamiliar UCs (4 UCs from another team) took 3.7 hours. The calculation of the time spent for the given example in the "manual" search for SUC increased the total time of UC formation by 58%.

Table 1. Results of selection of SUC in "manual" and automated modes

Developer teams	**1**	**2**	**3**	**4**
UC	12, 3, 4	5, 6, 7, 8	9, 10, 11, 12	13, 14, 15, 16
Selection of SUC				
Manual mode				
	(1)s	(7)s	(9-11)s	
Auto mode				
***Kmin*=0.2**	(1-10-15)s, (2-6-10*)s	(7-13)s, (5-9-16*)s	(12-14)s	(13-15*)s
***Kmin*=0. 5**	(1-10-15)s, (2-6)s	(7-13)s, (5-9)s	(12-14)	
***Kmin*=0.8**	(1-15)			

Automation of SUC selection became possible due to the use of UC step models. Further formalization of the UC definition, for example, by using a formalized natural language, is undesirable, since it will create inconvenience for the developer. It follows from the experiment results that the quality of SUC selection significantly depends on the value of the similarity coefficient K_{min}.

There is no guarantee that K_{min} = 0.5 value will always bethe best. The solution could be to use a domain dictionary to define an additional semantic relationship between compared texts, and as minimum, to use synonyms.

The effectiveness of the proposed method of UC restructuring depends on the specific subject area. In the conducted experiment, the tasks for the development of UC were selected taking into account the possibility of selecting SUC. In real conditions, it can be expected that the proportion of SUC in the UC corpus will be 2–3 times lower [3]. This will reduce the time to search for the use of SUC in UC, but not the selection of new SUCs. Therefore, in this case, we can expect a reduction in the time for restructuring by about 30%. – 40% as well.

5 Conclusions

The analysis of existing technologies for compiling UC was carried out. It was established that working in small teams on projects of medium and high complexity does not allow presenting the UC corpus in a well-structured form.

A mathematical model of the use case characterized by the introduction of the concept of the UC type, references to other UCs and the development team was proposed, which made it possible to further organize the process of comparing the UC and selecting the UC.

For the first time, a method of automated UC restructuring which allows comparing UC scenarios, selecting SUC, correcting the links between UC and SUC was developed. Application of the method makes it possible to improve the structure of the UC corpus, which increases the degree of understanding of the requirements, reduces the time and errors for maintaining requirements due to the elimination of duplication.

The experiments conducted showed the selection of all repeating fragments of scenarios, the correct selection of the SUC and a significant reduction in the time for UC restructuring (about 35%).

The proposed method can be used in any IS project where the functional requirements are presented in the form of UC.

References

[1] Wazlawick, R.S.: Object-Oriented Analysis and Design for Information Systems: Modeling with UML, OCL, and IFML. Morgan Kaufman, San Francisco (2014).

[2] Nilsen, A.F, Muller, G.: Use Cases and Non-functional Requirements Presented in Compact System Description A3s. In: INCOSE International Symposium, vol. 24, pp. 1–15 (2014).

[3] Cockburn, A.: Writing Effective Use Cases. Addison-Wesley, USA (2001).

[4] Mighetti, J.P., Hadad, G.D.S.: A Requirements Engineering Process Adapted to Global Software Development. CLEI Electronic Journal 19(3), 1–21 (2016).

[5] Russell, M.: Supporting Decision Makers with Use Cases; case study result. Procedia Computer Science 153, 294–300 (2019).

[6] Dobing, B., Parsons, J.: Understanding the Role of Use Cases in UML: A ReUCew and Research Agenda. Journal of Database Management (JDM) 11(4), 28–36 (2000).

[7] Clausen, M., Apel, R., Dorchain, M.: Use case methodology: a progress report. Energy Informatics 1(1), 274–283 (2018).

[8] El Miloudi, K., Ettouhami, A.: A Multiview Formal Model of Use Case Diagrams Using Z Notation: Towards Improving Functional Requirements Quality. Journal of Engineering 2018(3), 1–9 (2018).

[9] Giannakopoulou, D., Pressburger, T., Mavridou, A., Schumann, J.: Automated formalization of structured natural language requirements. Information and Software Technology 137, 106590 (2021).

[10] Linders, B.: Applying Use Cases in Agile: Use Case 2.0, Slicing and Laminating, https://www.infoq.com/news/2014/02/use-cases-agile, last accessed 2022/09/15.

[11] Barcelos, L.V., Penteado, R.D.: Elaboration of software requirements documents by means of patterns instantiation. Journal of Software Engineering Research and Development 5, 3 (2017).

[12] Kungurtsev, O., Novikova, N., Reshetnyak, M., Cherepinina, Ya., Gromaszek, K., Jarykbassov, D.: Method for defining conceptual classes in the description of use cases. In: Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2019, 1117624. Wilga, Poland (2019).

[13] Ahmed, H., Hussain, A., Baharom, F.: The Role of Natural Language Processing in Requirement Engineering. International Journal of Engineering & Technology 7(4), 168–171 (2018).

[14] Shah, U.S., Jinwala, D.C.: Resolving Ambiguities in Natural Language Software Requirements: A Comprehensive Survey. ACM SIGSOFT Software Engineering Notes 40(5), 1–7 (2015).
[15] Kungurtsev, O.B., Novikova, N.O., Zinovatna, S.L., Komleva, N.O.: Automated object-oriented for software module development. Applied Aspects of Information Technology 4(4), 338–353 (2021).
[16] Kalyanathaya, K.P., Akila, D., Suseendren, G.: A Fuzzy Approach to Approximate String Matching for Text Retrieval in NLP. Journal of Computational Information Systems 15(3), 26–32 (2019).

Method of Converting the Monolithic Architecture of a Front-End Application to Microfrontends

Olexandr Varenyk[1 [0000-1111-2222-3333]], Nataliia Khatsko[1, 2 [0000-0002-2543-0280]], and Kyrylo Khatsko[1 [0000-0003-3315-1553]]

[1] National Technical University «Kharkiv Polytechnic Institute», Kharkiv, Ukraine, 61002, Kyrpychova str., 2

[2] National University of Radio Electronics, Kharkiv, Ukraine, 61166, Nauky Ave. 14

n.khatzko@gmail.com

Abstract. Increasing software requirements lead to the need to review existing architectures and approaches to information systems design. A lot of shortcomings of the old server application with the monolithic architecture could be eliminated by dividing the application into smaller software components by using microservice architecture (MSA). Applications with MSA are easy to support, scale and test. The same advantages are becoming increasingly important for the large client single page applications (SPA). While the process of migration of the monolithic server applications to MSA is already described well enough, the dividing of the client application to the micro-frontends is still a tricky issue. This study aims to develop a method of applying micro-frontends approach for monolithic SPA. First, the authors studied the existing research of architecture reengineering and benefits of the microservice approach. Then they explained the process of converting a monolithic architecture into a microservice one, which was proposed earlier. After that, a new transformation method is proposed to apply such process for front-end application by migration to intermediate stage, namely the modular architecture. Provided architecture transformation algorithm allows code owners to get a guaranteed better result by taking into account the limitations of the front-end SPA which was not covered by authors of previous articles. Experiments with typical front-end SPA proved that proposed method of converting could be applied.

Keywords: software migration, microservice architecture, microfrontends

1 Introduction

To create a scalable, future-oriented software systems in modern industrial programming, the microservice architectural approach is increasingly used [1], [2]. Microservices break traditional monolithic applications into a set of smaller services that could be independently developed, tested, and deployed [3]. Due to highly decoupled software modules, microservice applications are easy to debug, update, use a third-party code, therefore, the dominant belief in a professional environment, that the future is in MSA [4]. Nevertheless, there are opposite points of view, especially in security field of research, which only reaffirm actuality of this field [5].

However, many applications have already been developed as a monolithic or modular, so in order to improve these applications, it is necessary to migrate them to the microservice architecture. Such actions have become the preferred solution for software upgrades than a new development [6].

With the development of browser-based client applications, as well as the requirements for them, the same problems that occur in monolithic backend applications become more and more relevant, this is especially acute in single page applications (SPA), which were originally conceived as a single monolith.

The monolithic SPA application has been broken into separate microfrontends using the microservice approach. The topic of this paper is the method of converting the monolithic architecture of front-end applications to microfrontends.

The motivation of the topic of the current work is in ability to handle all the issues of the monolithic architecture by applying similar strategies as for monolithic back-end applications to convert them into microservice-based. Applying this architecture allows services to be developed by separate team, services could be tested and deployed in isolation. But in most of the cases on front-end side there is the typical SPA application. It could be designed using modern frameworks, but it is still the monolithic by its nature with all the disadvantages of this architecture. Therefore, applying the microfrontend architecture could help in this situation.

The second section provides an analysis of existing articles and papers related to strategies of the migration to the microservice architecture. Existing problems that could be occurred in applications with the monolithic architecture approach could be successfully solved with application of this migration. Despite of the existence of the fact that all the problems of the monolithic backend applications are inherent in front-end applications, approach with dividing has been less reflected in front-end development.

The next section describes the exiting methods of migration to the microservices. The usage of the microservice approach to break a monolithic SPA application into separate microfrontends is proposed. This section also highlights the limitations of the front-end SPA applications that could not allow to apply existing methods directly for converting its monolithic architecture to separate independent units similar to microservices. The additional motivation of such migration is described here. Finally, in the main partof the section the new determination of the existing steps is proposed and all the changes to be done on every step are described.

The next sections the existing technical approaches to organize microfrontends is described. The main advantages and disadvantages of these approaches and the ability to be used for SPA are listed here.

The results section contains a description of the experiment to prove the proposed method. The requirements to the application that should be the subject of the experiment of the architectural transformation to the microfrontends are given here. Changes in code and architecture are described in detail according to the previously explained method steps. In this section the choice of a technical solution for the organization of microfrontends is justified. New microfrontends are hosted and the final application is evaluated.

The last section contains the questions that are still not covered in the current experiment or could be improved in future works.

2 Analysis of Migration Strategies

Many business applications have been in use for many years, their development does not stop, and a lot of unsuccessfully fixed bugs have accumulated [7]. It would be useful for such applications to get a second life with a new architecture without this accumulated set of bugs. There is the reason to believe that migrating to the microservice architecture will help overcome the existing problems. Particular reasons for migrating older applications are the fact that microservices improve maintainability over traditional monoliths due to a smaller code base, a strong isolation of components, and an organization of microservices around a business functionality. In addition, the development company has the ability to create autonomous teams of employees, which should reduce coordination efforts and increase team productivity.

However, the introduction of microservices can complicate the quality assurance of systems [8]. From an architectural point of view, quality assurance is considered a key issue when migrating or developing systems based on microservices [9]. Most of the existing research on microservices are focused on architectural principles and the architectural patterns [10], [11], [12] in microservice migration practices, which could provide an analytical view of the common patterns and methods used for MSA and could be considered the starting point of the authors' work. Many researchers have contributed to the development and quality improvement of systems based on microservices [2], [3], [13].

As a result, MSA has also become the preferred path for software upgrades based on the architecture [14].

There are many examples of a successful rewriting of applications based on microservices [15], when applications are made immediately in the execution of microservices next to the original application [14], [16], [17].

While MSA has gained a lot of popularity as an architectural style for back-end development of web applications, this architecture has been less reflected in front-end development. There are a lot of published research related to the transformation of the back-end from a monolithic or modular architecture to a microservice one [4], [10], [11], [12], [18], [19].

3 Migration Method

Since the microservice architecture primarily touched server applications, the published methods for migrating to MSA could considered.

The process of moving from an existing system to microservices, based on earlier work on systems reengineering [10], [11], is described in three steps: reverse engineering, architecture transformation, and forward engineering [20].

The described migrations were motivated by the need to partially or completely modernize the system, to some extent such a system was considered legacy, so the system that existed before the migration was called pre-existing, and the target micro-service system was called new system. At the reverse engineering step, the system was analyzed to identify obsolete code, which became a candidate for transferring it to services. Further, this transformation was a restructuring of the code with the transformation of the current architecture to a microservice one but maintaining the same level of abstraction. At this step, the architecture, business model and business strategy are changed [20]. At the stage of backward engineering, the system is being finalized, implemented and deployed.

However, the client part of the system, the so-called front-end, has a number of limitations that make such a conversion a difficult task. Such limitations include the need to work with a single environment. It is executed on the client side, so within one application there is always only one address bar, one global BOM object, and, accordingly, the DOM that is part of it. It is the problem that the main limitations of microfrontends are built around.

The authors presented the adapted process of transition of a monolithic SPA application to microfrontends as follows (see Fig. 1).

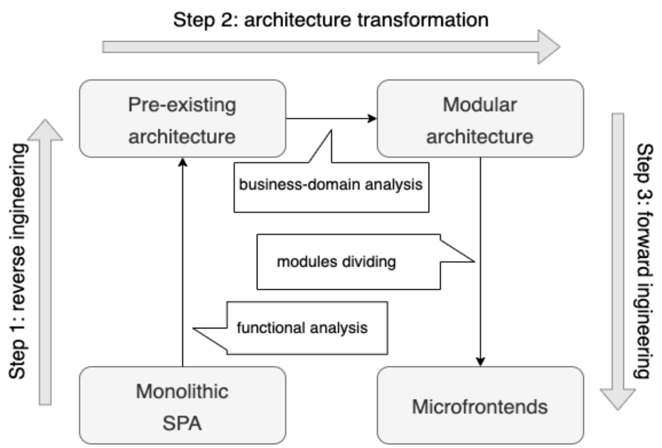


Fig. 1. Migration to microfrontends

The SPA approach has become popular relatively recently, so the motivations for migrating to microfrontends are caused not so much by outdated architecture, but by the non-functional benefits that microfrontends can provide. That is why the authors propose to revise the above transition steps and specify them as more appropriate in the context of working with client applications. So, for example, at the stage of reverse engineering, it is proposed to shift the focus from the search for legacy code to the functional analysis of the application as such. At this step, functions are grouped, or unified, large functions are divided. At the end of this step, the current architecture (pre-existing) is still a monolithic and requires further analysis before moving on to the next step.

The purpose of the stage of the architecture transformation is to analyze the current application to determine the main business functions of the application and, based on them, to identify potentially separate parts of the application that should not depend on each other as much as possible. For these purposes, the Strategic Design Domain-driven design approach could be used [18]. In the context of DDD, the main application domains are identified. To successfully solve this problem, all stakeholders could be involved: developers, architects, product owners; the project documentation is studied, compared with the main business requirements.

Based on the allocated domains, the application modules are created, which allows code owners to move to the modular architecture stage. It should be noted that at this stage the authors have already solved some problems inherent in monoliths: code is more structured andless coupled. And although one application with bundles hosted on the one server, this architecture allows code owners to organize lazy loading of modules. This leads to a decrease in the size of the main-bundle, and hence to a decrease in the initial load time of the application.

On the forward engineering step, the coupling between the components of the different modules is finally broken. At this stage, a technical solution for organizing microfrontends should be chosen. Due to the limitations of client applications, in any division into separate parts code owners still need one main application to manage other microservices. All the existing technical solutions come down to solve the problem of how individual microfrontends connect to the main application and how it orchestrates them.

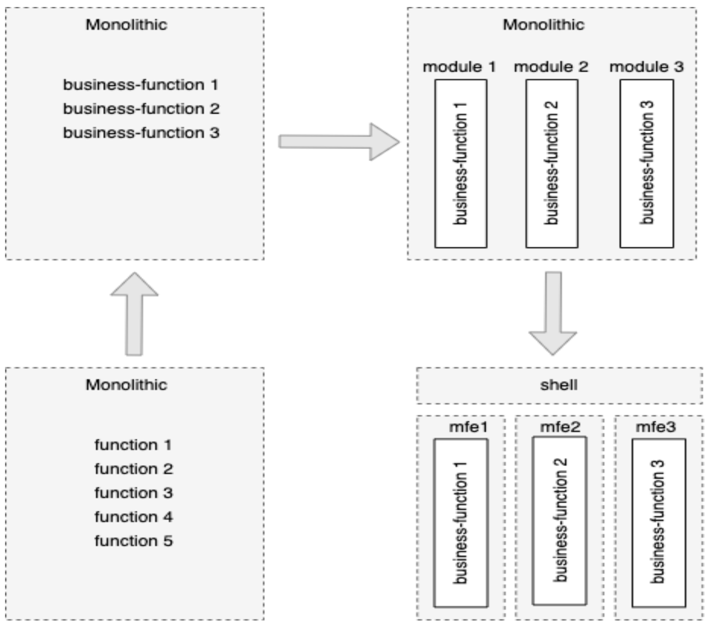


Fig. 2. Architecture transformation during migration to microfrontends

Step by step architecture changes are described on Fig. 2. On the step of the backward engineering step there is no architecture changes. Here changes on low level can be seen – grouping, unifying or removing unused functions. On this step code owners still have the same business models as in preexisting system. On the next step the transformation of the architecture finally takes place. By the end of the step code owners have the modular monolithic architecture. Business models could be changed to ensure a better isolation of the modules. These modules are candidates to be moved into separate microfrontends on the last step of the forward engineering step. By the end this step code owners have several separate applications (microfrontends) that are connected to the main application (shell).

4 Technical solutions for microfrontends

Currently there are several variants how to provide microfrontend architecture. The simplest way to organize microfrontends is to create several independent applications. Code owners need to have one main application with hyperlinks to other micro-frontend applications. Clicking on such hyperlink the user is navigated to the other application with other URL. The only benefit of this approach is its simplicity, but bad user experience is the biggest price for this advantage.

Another commonly used variant is applying single-spa framework. The idea is to create framework-specific wrapper for every microfrontend application to integrate them in one single-spa application. The main disadvantage of this approach is the need to follow the strict single-spa framework rules for every microfrontend to organize integration with other microfrontends. If there is a ready-made application, then it is a bug risk that it should be rewritten taking into account the single-spa rules.

One of the most popular mechanisms is to apply i-frames. All necessary widgets should be placed in i-frames that load the corresponding microfrontend hosted on a separate host. Data is exchanged between them using POST messages. The main disadvantage of the approach is the necessary of the loading full bundle of the micro-frontend.This fact limits ability to use i-frames only for good isolated applications. Another downside is the risk of the additional unnecessary libraries reloading with the microfrontend bundles.

The most modern way to work with microfrontends is to apply the Module Federation feature of the Webpack module bundler. This approach allows both the good communication of the microfrontends and the ability to avoid code duplication. The main idea of the approach is configure the shell application to import just the necessary module from a microfrontend application.

List the most popular variants additionally contains Tailor.js library. This library allows to create the microfrontend architecture based on the server rendering. This fact makes it unsuitable to use for SPA application because of the necessary of having a server Node.js application. Comparison of the technical solutions for microfrontends are displayed on Table 1.

Table 1. Microfrontend technical solutions comparison

Feature	Hyperlinks	single-spa	i-frames	Module Federation	Tailor.js
One URL for all application	–	+	+	+	+
Lack of framework – specific requirements for microfrontends	+	–	+	+	+
Avoiding code duplication	–	+	–	+	+
Lack of necessary to add backend server	+	+	+	+	–
Applicable to use with SPA	+	+	+	+	–

At this moment it seems, that Module Federation is preferable option, but further research could discover that more features required for breaking front-end monolith.

5 Results

The authors took for consideration the previously created Chess Tutorials application, on the client part of which experiments were carried out. When writing, the authors, recognizing the problem, chose an application that has a large number of internal communications, in order to maximally reflect the problems that developers face in the process of solving real problems. The client part is typical monolithic SPA created on theAngular framework with a state management organized with NgRx. The application is an educational platform for learning the game of chess. The application is designed for two types of clients – teachers and students.

To prove the ability to apply the proposed method of converting the pre-existing monolithic SPA has been refactored according to all the necessary steps of migration: reverse engineering, architecture transformation, and forward engineering.

At the stage of the backward engineering, the interface elements have been unified, large functionality has been analyzed and has been divided into smaller reusable parts, reused components have been identified. Functions related to authorization and student entities have been separated; an application routing has been changed by adding new route for home page, Angular SDK has been upgraded to higher version etc.

At the next stage, an analysis of the business functions of the application has been carried out. Business requirements have been defined the following abilities for different types of users: tutors could invite students to the system, create lessons and manage study groups, including tracking learning statistics for students; students could complete tasks from the lessons available to them, view the study groups. All available business documentation has been studied including vision and existing prototypes. The example of the documentation that could be applied to identify business domains is application user path is displayed on Fig. 3.

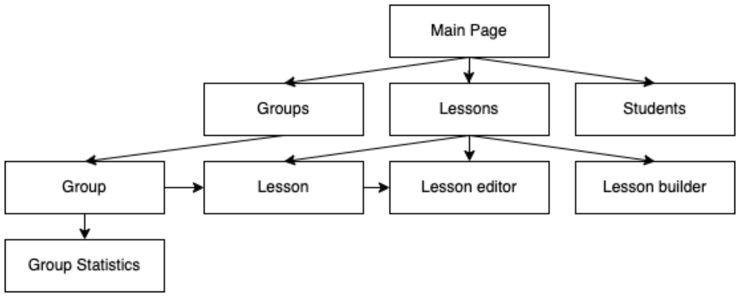


Fig. 3. Chess Tutorials documentation: path of the user with role tutor

Summarizing all the abilities and data from documentation, the authors have identified such basic business functions as managing students, working with lessons, working with study groups. Based on the selected business functions, the following domains have been determined: Lessons, Students, Groups and subdomain for Groups – Group Statistics. The schematic view of the modular application is displayed on Fig. 4. Since domains should have a separate model valid only within their bounded context [12], at this stage, the application state (store) has been restructured. All support functions of the centralized storage have been broken into separate modules according to belonging to a certain domain. The models of User and Student have also been separated, since they belonged to different domains. In reality, it is not always possible to achieve complete isolation of domains. This is exactly the situation that the authors have been faced with. To solve this problem, the data obtained during functional analysis have been used - the identified reusable components have been taken out into separate shared modules. It should be noted that creating a single shared module is a bad solution for large systems. Since not all functions could be reused in each of the above domains, it is recommended to organize several shared modules to prevent unnecessary functionality from being imported.

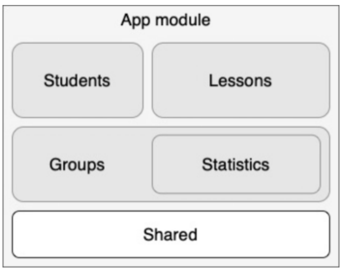


Fig. 4. Application structure by the end of the architecture transformation step

All the domain logic has been moved to separate independent modules. All smart and dump components and support services responsible for working with students have been collected in the Students module; the components of the lesson builder, view and list of lessons (support services and other structures) have been moved to the Lessons module; everything related to group management, including a separate module of Statistics, have been moved to the Groups module. After the necessary transformations, the Chess Tutorials application still is a monolithic application, but consisting of as much as possible separated domain modules, as well as shared modules. At this stage a lazy loading of the modules has been applied. Domain modules also have had their own internal routing. The above domain modules are the candidates for separate microfrontends. Module (sub-domain) Statistics at this stage do not look independent enough to be moved to a separate domain, and, accordingly, are not a candidate for moving to a separate microfrontend. With the development of the application and with the addition of new functions, such a transfer may become relevant in the future, so such a decision could be postponed at this stage.

On the forward engineering step, all possible links between domains have been broken down as much as possible, since each domain would be placed in a separate application. Orchestration by these applications has been done by a shell application that has been built from the main application module. Orchestration itself has been done using routing. On the last the microfrontends have finally been created. One of the problems that has been occurred at this stage is the correct technical solution for implementing the individual microfrontends. Since they should be separate independent applications, duplication of a large amount of code, at least the framework itself and styles, cannot be avoided. These domain modules still have had some common functions placed in shared modules. Code owners also have had state manager and common data used in several modules. All these have led to the fact that duplicate parts of the code would be loaded several times, for example, when using the i-frame approach. To avoid code duplication in bundles of the future microfrontends the authors have applied Webpack Module Federation approach. Microfrontends is still separate independent applications with duplicated code, but Webpack has allowed to load only the necessary (declared) modules of microfrontend applications in the resulting application on the client side. Shared module has been divided into separate shared libraries. New microfrontend applications have been created and main domains have been moved from main application to them. After that the authors have set up shell application to import just domain modules from microfrontends and have deployed all the applications separately. As a result, code owners have had shell application, mfe1 (Students), mfe2 (Lessons) and mfe3 (Groups).

Comparison of the pre-existing system and refactored system is displayed on Table 2.

Table 2. Pre-existing and target systems comparison

Measurable indicator	Pre-existing system (monolithic SPA)	Target system (Microfrontends)
Production builds building time, ms*	20290	15363**
Size of the main bundle, KB ***	540.5	82.7
First page average load time, ms***	644	269

* Average time on developer machine.

** Average build time of the mfe2.

*** Pingdom speed test tool.

Because of the ability to run build process of every single microfrontend and shell application in parallel, the result build time could be equal to the build time of the largest application – mfe2. The size of the final bundle of the target system (microfrontends) has become smaller, which reduced the load time of the first page of the application.

6 Conclusions and Future Work

In current work existing methods of migration to microservices have been adopted to be acceptable for microfrontends. Steps of converting and new states of the application have been defined and described to take into account the limitations of the front-end SPA which was not covered by authors of previous articles. Experiments with typical front-end SPA Chess Tutorials have proved that proposed method of converting could be applied. Authors are going to explore the concept of DDD for more efficient domain identification and microfrontend separation. Better understanding or conceptions of The Strategic design and the boundary context will allow to design better domain models and as a result better isolated microfrontends. Authors are also interested in improving of the proposed method of converting by formalizing and describing the process of working with multiframework SPA microfrontends (Angular, React, Vue) which will be the topic of future works.

References

[1] Furrer, F.J.: Future-Proof Software-Systems. Springer, Wiesbaden. (2019). 376p. https://doi.org/10.1007/978-3-658-19938-8.

[2] Gidey, H.K., Marmsoler, D., Eckhardt, J.: Grounded Architectures: Using Grounded Theory for the Design of Software Architectures. In: 2017 IEEE International Conference on Software Architecture Workshops (ICSAW), pp. 141–148. (2017), DOI:10.1109/ICSAW.2017.41.

[3] Terdal, S.: Microservices Enabled E-Commerce Web Application. International Journal for Research in Applied Science and Engineering Technology 10, 3548–3555 (2022). DOI:10.22214/ijraset.2022.45791.

[4] Francesco, P.D.; Lago, P.; Malavolta, I. Migrating towards microservice architectures: An industrial survey. In Proceedings of the International Conference on Software Architecture (IEEE), Seattle, WA, USA, pp. 29–38. (2018) DOI: 10.1109/ICSA.2018.00012.

[5] Mateus-Coelho, N., Cruz-Cunha, M., Ferreira, L.: Security in Microservices Architectures. Procedia Computer Science. 1819 (6). pp. 1225–1236. (2021). https://doi.org/10.1016/j.procs.2021.01.320.

[6] Cruz, P.; Astudillo, H.; Hilliard, R.; Collado, M. Assessing Migration of a 20-Year-Old System to a Micro-Service Platform Using ATAM. In Proceedings of the 2019 IEEE International Conference on Software Architecture Companion (ICSA-C), Hamburg, Germany, 25–26 March 2019; pp. 174–181. (2019) DOI:10.1109/ICSA-C.2019.00039.

[7] Auer, F., Lenarduzzi, V., Felderer, M., Taibi, D.: From monolithic systems to microservices: An assessment framework. Information and Software Technology, vol. 137, pp. 106600, (2021). https://doi.org/10.1016/j.infsof.2021.106600.

[8] Li, S et al.: Understanding and addressing quality attributes of microservices architecture: A Systematic literature review. Information and Software Technology, vol. 131, 106449. (2021). https://doi.org/10.1016/j.infsof.2020.106449.

[9] Soldani, J., Tamburri, D.A., Van Den Heuvel, W.J. The pains and gains of microservices: a systematic grey literature review Journal of Systems and Software. vol. 146, 215–232 (2018) https://doi.org/10.1016/j.jss.2018.09.082.

[10] Kazman, R., Woods, S.G., Carrière, S.J. Requirements for integrating software architecture and reengineering models: Corum ii. In Reverse Engineering, 1998. http://doi.org/10.1109/WCRE.1998.723185.

[11] Razavian, M., Lago, P. Understanding SOA migration using a conceptual framework. Journal of Systems Integration, 1(3), 2010. https://core.ac.uk/download/pdf/15455794.pdf.

[12] Manfred, S. Enterprise Angular: Micro Frontends and Moduliths with Angular. https://www.angulararchitects.io/en/book/.

[13] Homay, A., Zoitl, A., de Sousa, M., Wollschlaeger, M.: A Survey: Microservices Architecture in Advanced Manufacturing Systems. In: 2019 IEEE 17th International Conference onIndustrial Informatics (INDIN), pp. 1165–1168. (2019). Doi: 10.1109/INDIN41052.2019.8972079.

[14] Abdellatif, M. et al: A Taxonomy of Service Identification Approaches for Legacy Software Systems Modernization. Journal of Systems and Software. 173. 110868. (2020). https://doi.org/10.1016/j.jss.2020.110868.

[15] Hasselbring, W., Steinacker, G.: Microservice Architectures for Scalability, Agility and Reliability in E-Commerce. In 2017 IEEE International Conference on Software Architecture Workshops (ICSAW) pp. 243–246. Gothenburg, Sweden (2017). DOI: 10.1109/ICSAW.2017.11.

[16] Patil, M., Prajapati, S.: Microservice Architecture for Scalability and Reliability in E-Commerce. International Journal of Advanced Research in Science, Communication and Technology. pp. 398–402. (2021). DOI: 10.48175/IJARSCT-2050.

[17] Asrowardi, I., Putra, S., Subyantoro, E. Designing microservice architectures for scalability and reliability in e-commerce. Journal of Physics: Conference Series. 1450(1). 012077. (2020). DOI: 10.1088/1742-6596/1450/1/012077.

[18] Evans, E. Domain-Driven Design: Tackling Complexity in the Heart of Software. https://www.amazon.com/Domain-Driven-Design-Tackling-Complexity-Software/dp/0321125215.

[19] Blinowski, G., Ojdowska, A., Przybylek, A. (2022). Monolithic vs. Microservice Architecture: A Performance and Scalability Evaluation. IEEE Access. 10. 1–1. https://doi.org/10.1109/ACCESS.2022.3152803.

[20] Di Francesco, P., Lago, P., Malavolta, I. Migrating Towards Microservice Architectures: An Industrial Survey. Conference Pape, 2018 https://researchgate.net/publication/326566575.

Automated Software Development with Finite-State Machine Based Structures

Dmytro Mykhailovych Nikitin [0000-0003-4388-4996],
Vira Volodymyrivna Golian [0000-0001-5981-4760],
Nataliia Viktorivna Golian [0000-0002-1390-3116],
and Zoia Volodymyrivna Dudar [0000-0001-5728-9253]

Kharkiv National University of Radio Electronics, Ukraine
info@nure.ua

Abstract. This paper introduces the models and methods of automated software development with finite-state machine-based structures. The purpose is to provide and analyze the main implementations of finite-state machines in object-oriented design and development. The frameworks and components currently available for use are thoroughly examined and investigated with providing the corresponding analysis results. One of the main goals is to create a software component that is based on abstract automata which can be used to implement the behavior of the module which provides different outcome or output signals with the same initial input data. The component is presented as part of the class diagram built to highlight the main points of integrating it into the system. A comprehensive and detailed approach of converting the existing system to use the proposed State Engine pattern in flexible sub-modules is outlined.

Keywords: abstract automata, state engine, output state, finite state machine, transition manager, system refactoring

1 Introduction

Creating any type of software requires building on previous work. Creating a successful product requires completing several stages. These stages are necessary for any type of software development, from a small program preparing information to a large-scale product.

Management of such large projects requires careful analysis of existing approaches, methods, and tools, which will allow to adapt to emerging requirements without unnecessary problems.

Creating good software requires first designing its features, timeline, and sources of data. Completing the first fundamental stage in software development can be achieved by planning these parts out.

Modern software development involves regular refactoring processes. This involves restructuring the program code, which is independent of any changes to functionality. This is done to improve support for changes and facilitate code understanding. Many patterns and techniques refactoring developed over the years.

Quite often there is a need to design a system component, which is an object of a selected domain area, which can change its properties or behavior depending on the current state. In this case, it makes sense to use an algorithm or pattern that works on the basis of a finite state machine or its object-oriented representation.

2 Aim and Materials

This study examines modernized software design and implementation by identifying existing flaws and replacing them with new finite state machine based software. It also aims to analyze existing software solutions and algorithms to determine any problems with their implementation.

The main approaches used in the work are:

– the automata theory (theoretical data on finite automata) [1];
– discrete mathematics (finite automata as a model of a discrete system) [2];
– the Finite State Machine as a model represented by the “State” design pattern.
– abstraction (choosing only the most basic properties of a software system: abstraction from the mathematical properties of the finite state machine when considering pattern design) [3];
– design patterns for OOP systems;
– formalization (a set of elements that represents a model as a finite state machine) [4];
– methods of algorithm building and analyzing.

3 Results and Discussion

3.1 Subject area analysis

One of the most popular and effective methods of building components that can change their behavior depending on their state is implementation using an algorithm built on the basis of a finite state machine.

A finite state machine is an abstract automaton whose number of possible internal states is finite. At the same time, it has a finite set of input symbols that lead to the formation of output words. It is also important that each input symbol is able to transfer the automaton to a new state. This is a fairly convenient abstraction that allows you to encapsulate a complex algorithm. In addition to this finite state machines are very efficient.

The abstract automata can be explained as a 6-element mathematical scheme G. This scheme has a finite set I of input signals; a finite set O of output signals; a finite set S of internal states; an initial state s0; a transition function x(s, i) and an output function y(s, i) [5].

The automaton is defined by the scheme G:

$$G = | I, O, S, s_0, x, y | \tag{1}$$

The finite state machine works according to the next scheme: first, input I(n) is received on each n-th cycle of the machine that is in the S(n) state. Next, the machine transitions to a new state S(n + 1) during the (n + 1)-th cycle with the release of the output signal O(n). These formulas can be written as [6]:

$$S(n+1) = x[S(n), I(n)],\ n = 1, 2, \dots \quad (2)$$

$$O(n) = y[S(n), I(n)],\ n = 1, 2, \dots \quad (3)$$

The graph of a finite state machine is presented in Figure 1.

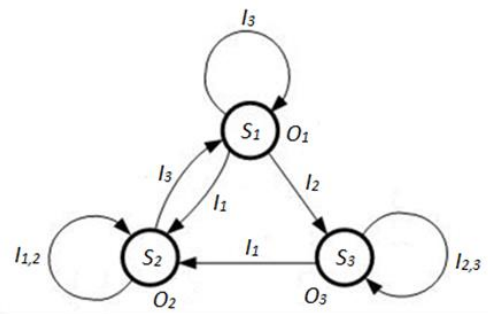


Fig. 1. The graph of a finite state machine

Changing an object's behavior depending on its state is achieved with the help of the State pattern. This can be seen as an integration of the finite state machine on top of the OOP-design world.

When designing modules, or classes, using the State pattern, a particular group of internal states is expected that manipulate different operations based on the current state. In order to achieve a consistent glance across all states, a shared interaction interface is required [7].

3.2 Existing algorithms and patterns based on finite state machines

Change in behavior occurs when an object’s state changes. This is a design pattern called the State pattern.

Typically, “if” statements are used to check the current state of an object and determine the appropriate behavior for the next execution. These states are mainly defined with a state machine.

The concept involves many components that change each other's states. The specific set of possibilities for the module’s states and the transitions between them is determinate and finite. Various states the module may be in can result from different events that transpire to it [8].

By understanding this concept, developers can apply it to individual software objects. E.g., “Document” objects in a software system change appearances depending on their stage in the document publishing process. For example, an unfinished draft,

a pending review or a published version can be handled differently via the publish method:
– In “Draft” mode, the app sends the document to “Moderation”;
– by an administrator, any work can go from “Moderation” to “Published”;
– when the object belongs to the “Published” group, the method does nothing.

The State pattern describes multiple classes specific to individual states. It uses these classes to assign context components' intended behaviors while running the program to dynamically change it [9].

A native component makes reference to a specific state object by using a native component context. Once assigned to the state, the context delegates work to it changing it in the runtime.

The class diagram of the “State” pattern including the main modules is shown in Figure 2.

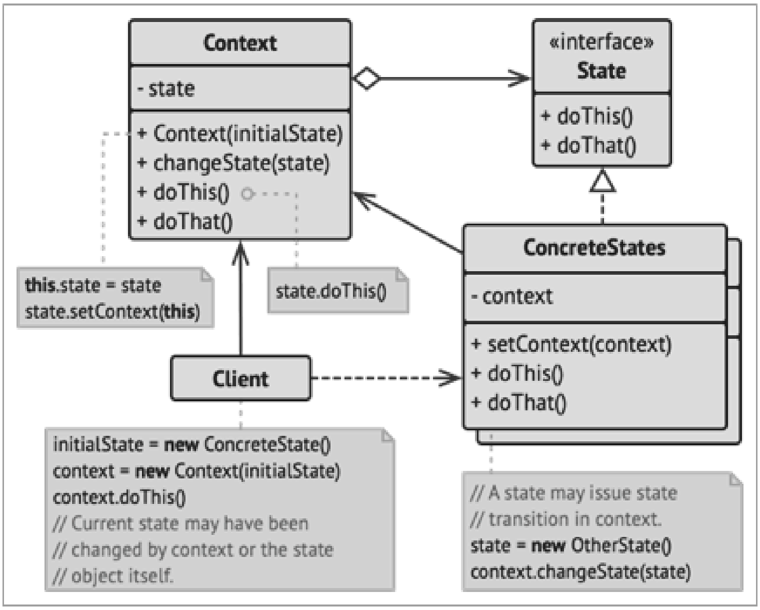


Fig. 2. Class diagram of the “State” pattern

There are a few side effects to this pattern:
– the different classes implementing states conceptually distinct from one another results in the transition of logic traversing between them being distributed;
– the state’s classes not being linked to each other is not guaranteed;
– designing a state classes hierarchy is difficult.

3.3 Problem statement

Based on the analysis of the selected subject area, it is possible to identify the main problems, the solution of which is the main focus of the research:

– reduction of the number of internal states and transitions between them when performing asynchronous operations;
– the need to create new objects for the finite state machine makes it difficult to reuse the state code in object-oriented projects;
– transitioning between object states requires moving away from centralization (the process of changing the current state);
– the requirement that all OOP state classes depend on each other causes problems for developers (impossibility of using them to describe other models).

Because of the described problems, it is possible to single out the tasks, the solution of which is the goal of the conducted research:

– analysis of the "State" design pattern, which is based on the object-oriented abstraction of the finite state machine;
– development of a design pattern that allows objects to change their behavior depending on their state, with the elimination of the main shortcomings of the existing "State" pattern;
– development of a refactoring method that will allow moving from the use of the "State" pattern to the use of the developed pattern with the minimization of the chances of errors.

3.4 "State Engine" design pattern overview

This paper proposes a new "State Engine" pattern, which combines the advantages of finite state machines and the "State" pattern.

In order to provide the ability to reuse state classes (for other domain models or in the context of any other component), this pattern uses an event dispatch mechanism with events that are used and sent by state machine classes to notify of operations that have been performed. The state machine object, in turn, reacts to these events by changing its current state, which can be performed both by the automaton itself and by a separate object that will be responsible for it. This scheme is somewhat similar to the "Observer" / "Publish-Subscribe" patterns, which are used to exchange messages between system components or between subsystems.

The described changes make it possible to centralize the logic of the transition between states of the automaton in one place, eliminating the need for classes of states to know about each other (which can be seen in the classic implementation of the pattern, where each specific state is responsible for changing the state of the automaton).

At the same time, the logic of transition between states of the automaton can be implemented by various methods. For example, you can build a table of transitions between states, which will indicate which state to go to, given the current state and the event that was received from it. Another way is to use standard tools of the programming language, such as switch (if-else) instructions, which will generate a new state of the state machine object for each variant of the state-event pair [10].

The main difference of using the "State Engine" pattern is that the existing states do not know about the existence of other states, which allows you to dynamically change the set of states without changing the existing logic of transitions between them.

This modification requires a state manager class to represent each possible state. Initially, this state class decides which new state to enter. In the new version the state only hands over the event to a state manager by providing a specific event object, which then determines alternatives on how to transition.

The class diagram of the modernized pattern is shown in Figure 3.

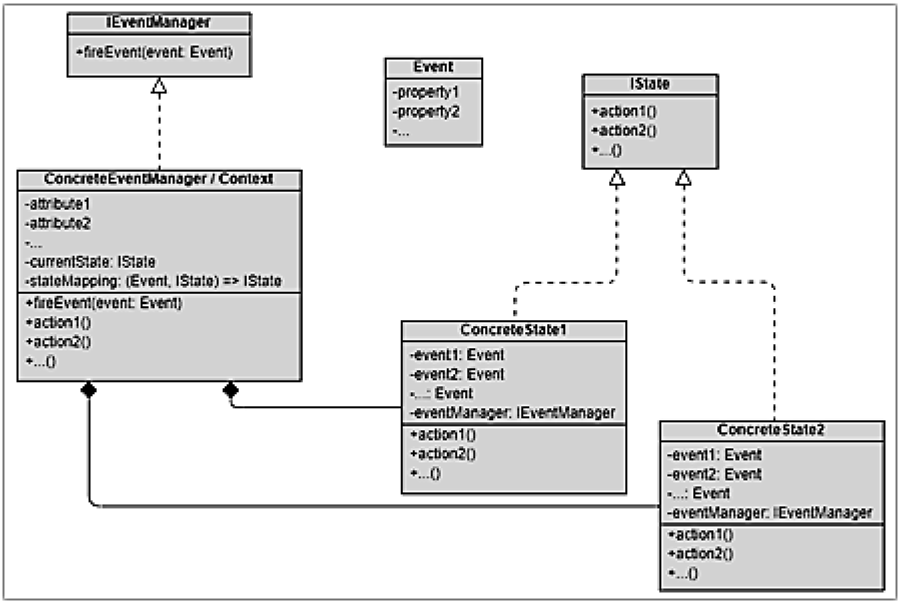


Fig. 3. Modified pattern class diagram

The following participants are present in this diagram:

– *Event* is the base class for the event that was generated by the state classes. Contains a set of attributes "property1", "property2" that can be used by the class that receives events;
– *IState* is a basic interface of state types that defines a set of operations ("action1", "action2") that all states of a certain object must support;
– *ConcreteState1*, *ConcreteState2* are concrete state classes that implement the "IState" interface. These classes are responsible for creating events described by the "Event" class;
– *IEventManager* is a basic component interface that can receive events from specific states, and which is passed to states when they are created. This interface is the only contract that allows classes of states and the finite state machine to interact;

– *ConcreteEventManager*/*Context* is a concrete class of a state machine that has a current state that can be changed during its life cycle, and implements the "IEventManager" interface to respond to events that have occurred and change its state.

When creating an abstraction object of a finite state machine ("ConcreteEvent-Manager" class), all existing states of the component are initialized and a graph of transitions between the states is built upon the occurrence of a certain event. All created states receive an object to notify of events and a context object, which in this case is of the same type "ConcreteEventManager" but can be split into different classes as needed.

During the operation, the automaton object delegates operations to the current state, which can modify the stored context or send an event that will change the current state of the automaton to a new one defined in the transition table. That is, the decision to change the state is made by the state machine itself, not by its states.

Because of the state manager's choice of the next state's value based on current conditions, classes that exhibit state values don't have to communicate with each other.

3.5 Use of the "State Engine" design pattern

The "State Engine" pattern can be used in the following cases:

– if there is a component that drastically changes its behavior depending on its internal state, and there are many types of such states (and they can change), and their code often changes;
– if the program code of the module contains many complex, similar to each other conditional statements that determine the behavior depending on the current values of the properties of the module;
– if a tabular state machine built on conditional operators is used;
– if there is a need to reuse state classes to implement a state machine of a different type that has nothing to do with the original class.

"State Engine" pattern makes it quite easy to implement the abstraction of a finite state machine to describe the behavior of a model of a certain domain. Then, if necessary, it is possible to adjust and modify the created component to meet new requirements.

The pattern presented in the paper is flexible and dynamic, which is reflected in the ability to change and adjust it depending on the needs of a particular software system. "State Engine" pattern allows you to make the following modifications:

– data model storage. A context can be implemented in such a way that the data model that is passed to the state classes for use during operations is separated from the event handler and placed in a separate class. This approach complicates the implementation of the pattern, but reduces the dependency between state classes and the context class;
– stateless and stateful state classes. In the "State Engine" pattern, the context and state classes contain references to the data model and event manager. Thus, state classes

are stateful (keep their internal state during their operation). This approach is not always acceptable, because in some cases memory consumption is critical. In this case, the pattern can be changed so that the state classes are stateless. At the same time, state classes will have to pass parameters to the data model and event manager on each method call. In addition to saving memory, this will allow using the state object for different data models, since they will not be bound to each other;
- definition of transitions between states. In the initial version, transitions between states are specified in the context (event manager) using a transition graph or special constructions of the programming language. But for certain cases, it is possible to implement this logic in a separate class, which will allow the context class not to depend on a specific table of transitions and to be able to change it if necessary. This can be useful if adjacent states can change depending on certain conditions.

3.6 Transition to "State Engine" pattern

For software that contains system modules that implement the "State" pattern, a very important criterion when switching to the "State Engine" pattern is the ability to carry out modernization with a minimum number of actions and without significant changes in the existing code base to reduce the risk of errors [11].

During the implementation of this work, an instruction was developed containing a sequence of stages necessary for refactoring system components that are planned to be modernized. The sequence of performed actions is as follows:
- decide on a class that will play the role of the context (event manager). It can be both an existing class that already depends on the state, and a new class, if the state code is spread over several classes;
- create a common state contract (interface, if possible, use an existing one). It should describe methods that are present in all states that are directly relevant to the context. An important point is that not all context behavior needs to be placed in new classes, but only that which is state-dependent;
- create a class (use existing ones if possible) for each actual state that implements the state contract. Implement the specific classes of required states with the code of the described behavior of the component. After the operations, all methods of the state interface must be implemented in the created state classes. It is also necessary to pass an object of the event manager as a parameter to the classes of states in order to be able to send a notification about the occurrence of an event;
- when extracting behavior from the context, there is a probability of encountering the fact that this behavior depends on private properties or methods of the context, which are not accessible from the state object. There are several ways to work around this problem. The simplest is to leave the behavior inside the context, calling it from the state object. Another way is to implement state classes as nested components of the context class, which will allow them to access all the private members of the context. However, this method is supported only by some programming languages (for example, Java, C#) and leads to increased coupling of state classes;

- create a base event class that will occur in state classes when certain operations are performed. Add the necessary fields to the class to pass data from the state class to the event manager;
- implement event-occurrence logic in state classes, paying attention to the fact that the occurrence of an event can lead to a change in the state of the context. Each occurrence of an event must be accompanied by sending a corresponding message to the event manager;
- create a field in the context for storing states, as well as a table of transitions between states, which will transfer the context to a new state based on the event-state pair;
- remove the code that switches the context state from the specific state classes, as it will now be the responsibility of the state manager.

Performing the actions described above will allow you to flexibly configure system components that have their own state and reuse state classes between objects of different types.

3.7 Relationship with other patterns

The primary task of patterns is to solve a problem that occurs quite often and with the wrong approach to the solution can lead to the emergence of new problems or make it impossible to scale the module. That is, patterns are, in addition to a way of building the software code in a certain way, also a means of describing the problems that lead to such a solution.

Design pattern “State Engine” shares some features with the following patterns:

- “Bridge”, “Strategy” and “State Engine” (as well as a bit of “Adapter”) have similar class structures – they all have at their core the principle of “composition”, that is, delegating work to other components, whose lifetime is determined by the main module. They are distinguished by the fact that they are used to solve different problems;
- The “State Engine” can be considered as a superstructure on top of the “Strategy” pattern. Composition is used in both patterns, allowing you to change the behavior of the main component by delegating work to nested helper objects. However, in “Strategy” these objects do not know about each other and are not related in any way. In the “State” pattern, specific states can independently switch the context of the main object, determining its future state [12].

3.8 “State” and “State Engine” design patterns comparison

The “State Engine” pattern is free of the main disadvantages of the “State” pattern:

- complicated reuse of state classes, as they know about each other and are highly dependent on each other, which makes it impossible to use them in a situation where the order of the transition between states is different under the same states;
- managing the states of the context class, because they determine the next state of the main model and explicitly indicate it, depending on the conditions of the business

logic, which exposes the details of the implementation of the context to the state classes;
– distribution of the logic of transition between context states to specific classes of states, which complicates the state change logic and divides it into different modules of the system.

The modified version of the pattern centralizes the logic of the transition between states in the context class (or a separate class that is responsible for building the transition graph and is associated with the context class through composition), which provides a better understanding of the system and the ability to make the necessary changes in one place.

In the new version of the pattern, state classes are no longer dependent on and unaware of other states, which allows them to be reused to design new components that may have the same states, but transition between them is performed in a completely different order. This is achieved by exposing a separate EventManager interface (event manager), which allows state classes to pass information about an event that has occurred, which will cause the context state to change depending on its current state. The implementation of the EventManager can be both in the context class itself and in a separate class.

4 Conclusions

Because of the State pattern's negative characteristics, a design idea emerged that solved many of its shortcomings. Called the State Engine pattern, this new idea improved on the State pattern by allowing changes in behavior depending on the object's current condition.

Using proposed design methods, developers can create reusable state classes while independently implementing changing states throughout various modularized systems. Furthermore, these systems can maintain single focus of operations while ensuring state transitioning logic centralization.

The practical use of this discovery is that it makes designing and developing software much easier. The results have significance in the real world, as they can be used to shape the behavior of software systems.

The obtained results make it possible to significantly speed up the process of performing operations in the program code that use the implementation of a state machine to control a set of instructions, and to facilitate the process of creating an object-oriented model of a component that can change its behavior depending on its current state.

References

[1] Grami A.. 2022. Finite-State Machines. In: Discrete Mathematics: Essentials and Applications. Academic Press, pp. 373–388. ISBN 9780128206560. DOI 10.1016/B978-0-12-820656-0.00020-4.

[2] Ibias A., Nunez M., Robert M. Hierons. 2021. Using mutual information to test from Finite State Machines: Test suite selection, volume 132. ISSN 0950-5849. DOI 10.1016 /j.infsof. 2020.106498.

[3] Barbanera F, de'Liguoro U., Hennicker R. 2019. Connecting open systems of communicating finite state machines. Journal of Logical and Algebraic Methods in Programming, volume 109. ISSN 2352-2208. DOI 10.1016/j.jlamp.2019.07.004.

[4] Engler E., Baldauf M., Sill Torres F., 2020. Finite State Machine Modelling to Facilitate the Resilience of Infrastructures: Reflections. Infrastructures 5, no. 3(24). DOI 10.3390/infrastructures5030024.

[5] Wang J. 2019. Formal Methods in Computer Science. CRC Press. p. 34.

[6] Al-Fedaghi S., 2020. Modeling the Semantics of States and State Machines. Journal of Computer Science 16(7). DOI 10.3844/jcssp.2020.891.905.

[7] Vivas A., Sanabria J. 2020. A Microservice Approach for a Cellular Automata Parallel Programming Environment. Electronic Notes in Theoretical Computer Science, volume 349, pp. 119-134. ISSN 1571-0661. DOI 10.1016/j.entcs.2020.02.016.

[8] Nazar N., Aleti A., Zheng Y. 2022. Feature-based software design pattern detection. Journal of Systems and Software, volume 185. ISSN 0164-1212. DOI 10.1016/j.jss.2021.111179.

[9] Bozorgvar N., Rasoolzadegan A, Harati A. 2022. Probabilistic detection of GoF design patterns. DOI 10.1007/s11227-022-04718-7.

[10] Schulz, S., Pease, A. 2020. Teaching Automated Theorem Proving by Example: PyRes 1.2. Automated Reasoning, IJCAR 2020, Lecture Notes in Computer Science, volume 12167. DOI 10.1007/978-3-030-51054-1_9.

[11] Alcocer J. P. S., Antezana A. S., Santos G., Bergel A. 2020. Improving the success rate of applying the extract method refactoring. Science of Computer Programming, volume 195. ISSN 0167-6423. DOI 10.1016/j.scico.2020.102475.

[12] Kaur S., Singh P. 2019. How does object-oriented code refactoring influence software quality? Research landscape and challenges. Journal of Systems and Software, volume 157. ISSN 0164-1212. DOI 10.1016/j.jss.2019.110394.

[13] Lemus V., Acuña-Yeomans E., Zamora V., Hernández-Quiroz F., Zenil H. 2022. Computable model discovery and high-level-programming approximations to algorithmic complexity. Theoretical Computer Science, volume 927, pp. 31–41. ISSN 0304-3975. DOI 10.1016/j.tcs.2022.05.026.

Approaches for Building an Agent-Oriented Model of the Competence Formation System

Tetiana Popenko[1] [0000-0002-4918-9866], Olena Kononova[1;2] [0000-0003-0403-7292], Alona Yurzhenko[3;4] [0000-0002-6560-4601], and Alla Paziak[3;5] [0000-0002-7699-6428]

[1] Maritime Applied College of Kherson State Maritime Academy 14, Ushakova ave., Kherson, 73000, Ukraine
[2] konon2017@ukr.net
[3] Kherson State Maritime Academy 20, Ushakova ave., Kherson, 73000, Ukraine
[4] helen18@online.ua
[5] pazyakalla@gmail.com

Abstract. The paper describes intellectual system of the formation of professional competencies and the approaches to building an agent-oriented model of competence formation system. The diagram of the functioning of an intelligent system is shown in the article. The main functions and executive mechanisms of agents are summarized. It was concluded that agent-oriented approach has not yet become the leading paradigm for building corporate industrial applications and information systems, and requires the development of efficient and understandable and practical AOS programming architectures and methodologies, needs to be promoted among system engineers and software developers. An analysis of the scientific literature showed that the solution of the listed problems is carried out, as a rule, by non-formal methods, which reduces their practical value. The prospects of further research can be seen in the use of VR and AR to create models of complex distributed and heterogeneous systems.

Keywords: competency, education, agent-oriented model, competency-based approach, LMS moodle

1 Introduction

Modern approach to education dictates high-level requirements for the content of education materials, and the ability to apply received knowledge, skills and personal qualities in future professional activities.

This is ensured by competency-based training. The development of information technologies allows to organize individual trajectory in learning and provide adaptive educational content, to ensure a competency-based approach. Individual trajectory in learning helps student to choose educational tracks on the topics of content by their own, same as adaptive educational content tries to predict and choose the most suitable types of exercises, the complexity of the programs according to the student's level of knowledge. And competency-based approach is aimed at the formation of competencies and skills needed in future profession [1].

There are a lot of training computer systems which combine the properties of intelligent systems and adaptive control systems.

Adaptation technologies in these systems are borrowed either from the field of artificial intelligence (AI): adaptive planning, intelligent analysis of student's decisions, support for interactive problem solving, support for problem solving by examples and support for collaboration or from the field of adaptive hypermedia: support for adaptive presentation and adaptive navigation [2], [3].

2 Analysis of recent research

A general idea of intelligent learning systems was formulated back in 1970 by D. Won, P. Jansen and J. Carbonell [4], but real research and commercial ITS appeared already in the 80s of the XX century.

While studying Agent-based computing R. Abdalla and A. Mishra investigated the agent concepts, techniques, methods, and tools used in evolving Internet of Things systems. In their research scientists conducted a comprehensive analysis of selected agent-oriented software engineering methodologies. They have also provided a proposal of a draft unified approach that drives benefits of these methodologies towards advancement in the studied area [5].

Multi-Agent Programming was studied both by A.R. Panisson, P. McBurney, R.H. Bordini and by T. Ahlbrecht, J. Dix, N. Fiekas, T. Krausburg groups of scientists. The first one investigated mostly the advantages of using argumentation-based techniques in multi-agent systems. They also proposed an argumentation framework using the particular structure of argumentation schemes at its core [6], [7]. While the second group of researchers tried to find scenarios, where it pays off to use the tools of agent-oriented software engineering, and tried to encourage people to learn about those tools and instruments [8].

A. Croatti and A. Ricci studied programming agent-based mobile applications, namely the JaCa-Android framework. They described JaCa-Android framework and benefits of its use. They also highlighted that the impressive progress of technologies makes it possible to explore the use of agent-oriented programming languages and frameworks based on cognitive architectures [9].

E. Iotti, G. Petrosino, S. Monica, and F. Bergenti discussed two approaches to agent-oriented programming and compared them from a practical point of view. Thus, two considered languages were used to solve the same coordination problem, and obtained implementations were compared to discuss pluses and minuses of both approaches (Jadescript and Jason) [10].

The analyze on agent programming was conducted by R.H. Bordini, A. Fallah Seghrouchni, K. Hindriks, B. Logan, and A. Ricci. The researchers reviewed the state of the art in agent programming, focussing particularly on BDI-based agent programming languages [11].

Agent-oriented modeling was also the subject of research of Y. Ivashkin and M. Nikitina scientists. Their proposal was the agent-oriented simulation model of the logistic system of the material flows. The results of their research led to optimization of material flows [12].

Agent-oriented methodology became a subject of scientific research of the group of scientists: C.W. Shiang, S.Y. Wai, N. Jalia and M. Bin Khairuddin.The adoption of such methodology was investigated. The researchers introduced a systematic way to model crime simulation in detail [13].

Research of S. François, J. Ferber, T. Stratulat, F. Michel was devoted to the original methods for describing reactive agents. Thus, the "eco-resolution" method proposed by them is based on solving a problem by a set of agents that communicate by exchanging messages. Here, the solution of the problem is understood as the evolution of a dynamical system until it reaches stable stationary states. These stationary states correspond to the satisfaction of the goals of various agents.

It is also actively conducting researches on the development of Internet education. The informational systems are a software shell that not only provides distance learning and testing of student's, but also allows to manage the activities of a virtual educational institution.

Meanwhile, the disadvantage of these systems is the processing of limited individual knowledge about the student [14].

3 Results

For example, such automated learning systems as ELM-ART-II, AST, ADI, ART-Web, ACE, KBS-Hyperbook and ILESA allow to develop intelligently a learning sequence in the form of a frame script. The interactive learning system "Learning Space" (of the firm LOTUS) provides an opportunity for a tutor to place educational material on a server, to create discussion seminars for discussing individual topics with students. Students can study the material, participate in discussions on topics.

Connection between participants of the training (teachers and student's) is carried out using e-mail, which is a part of the DOMINO services. However, the key role in this system is played by the human factor, which does not allow to classify it as intellectual.

Agent-based systems are being developed and implemented. They are the following:

- Math Tutor, which provides dynamic, interactive teaching aids for learning mathematics;
- OLAT web application which implements a learning management system that supports any kind of online learning, teaching and learning with few educational limitations.

However, despite the variety of information systems and software, intelligent learning systems require improvement in the following directions:

1. Diagnostics and expert assessment of the student's competencies.
2. Planning and analysis of the learning trajectory, the ability to adapt to learning goals.
3. Development and implementation of the programs' interface that correspond to the personal qualities of the student and his emotionality.

The intellectual system of the formation of professional competencies should provide a full cycle of management (reaction to events; dynamic planning; coordination and revision of plans; monitoring and control).

Let define the main functions of such system:

– Administration;
– Formation of an individual training program based on the results of primary control and questioning;
– Implementation of diagnostics by means of testing at various stages of training;
– Formation of the current assessment of the structure`s state and parameters of professional competencies;
– Analysis of the results of diagnostics to form an adequate control action;
– Decision making on the formation of control actions at various stages of education;
– Providing educational content which adapted to the individual abilities and needs of learners;
– Support functions for the formation of educational materials;
– Communication functions.

Figure 1 shows a diagram of the functioning of an intelligent system, a feature of which is the presence of two correction blocks that allows to form control actions at various stages of training.

Automation of the intellectual system will allow not only to exercise control at various stages of education, but also to modify educational programs based on the results of knowledge control. At the same time, individual sets of modules are determined not only by the standard of the specialty, but also by the student's choice. And the set of educational elements of the module program may vary, depending on the individual trainees' abilities and the need for an intensity of study of the material in further education.

The process of forming of professional competencies is a complex action that includes several stages.

The solution of the problem of managing a complex deterministic process is proposed to be considered as the result of the interaction of many independent purposeful program modules – the so-called agents, while the agent can act on behalf of and in the interests of a person.

The field of multi-agent systems is described in [15].

These systems include:

1. open systems – systems whose structure can change in the course of their operation;
2. complex systems – systems that include many modules (subsystems);
3. interactive systems – systems that accept user commands and interact intelligently with him.

The use of multi-agent systems is most common in the automation of complex systems management, in systems for collecting and processing information; games [16].

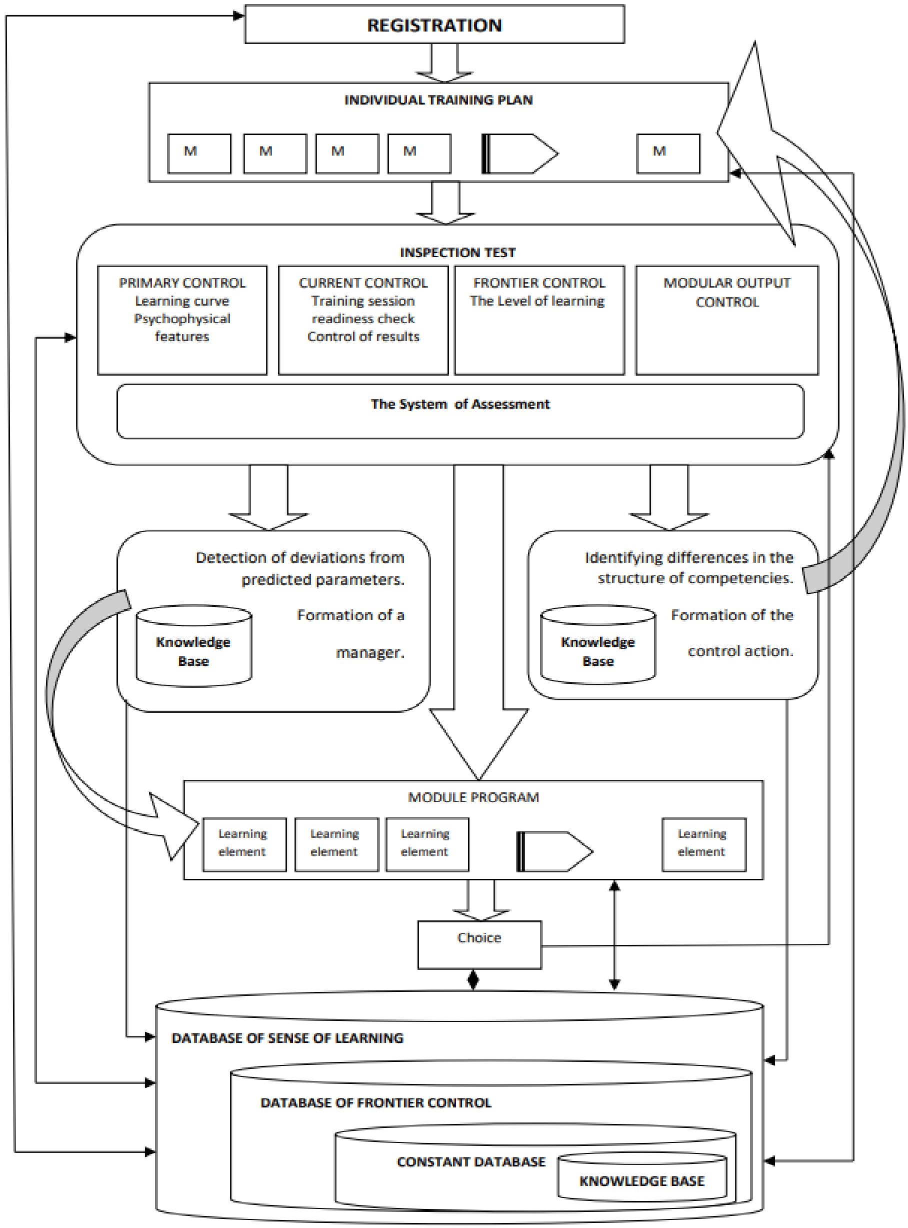


Fig. 1. Functional diagram of an intellectual system for the formation of professional competencies

According to the proposed scheme, the functions of the system are divided into separate modules (control and diagnostics, planning, analysis, blocks that provide communication channels) which, interacting, provide a flexible system of managing for the formation of professional competencies [17].

The operation of each module is provided by a software agent, the operation algorithm of which is determined by the script [18]. The relationship of the set of agents of the system is represented by an agent-oriented model in Figure 2.

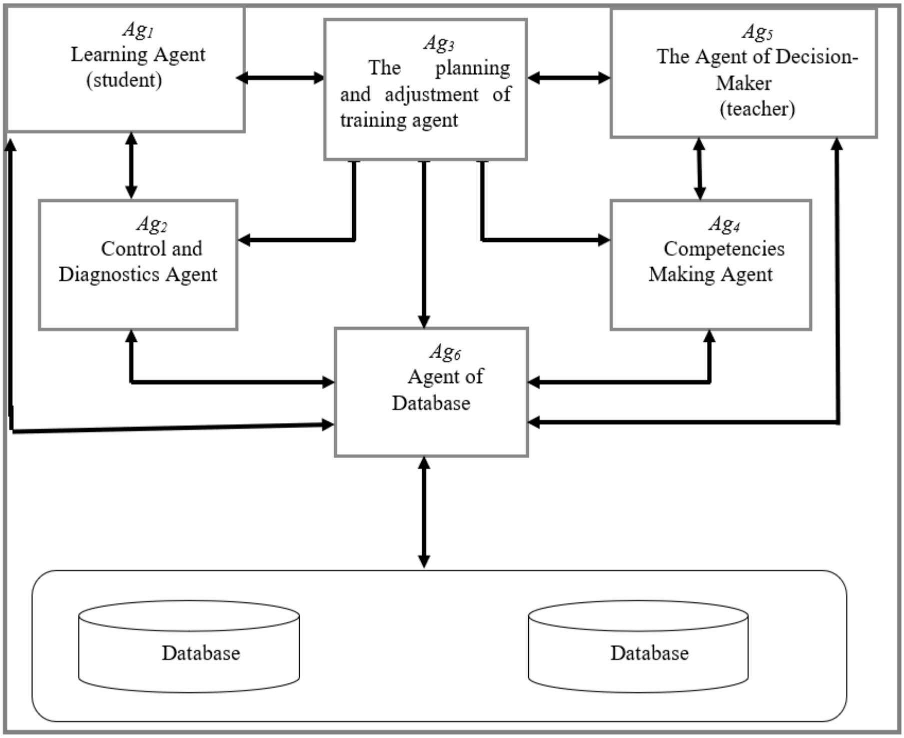


Fig. 2. Agent-based model of the system for the formation of professional competences

At each step, agents consider system inputs and react to unpredictable events (delays, failures, changes). The reaction can be independent, or carried out in cooperation with the trainee.

Databases contain a formalized representation of objects in the subject area and their relationships, as well as actions on objects. The main functions and executive mechanisms of agents are summarized in Table 1.

Table 1. The main functions and executive mechanisms of agents

	Main functions	Environment	Actuating Mechanism	Detectors
Learning agent (student)	Organization of procedures for issuing educational material, recording events in the process.	Student, data channels between agents.	Display of dialog boxes.	Keyboard input
Control and diagnostics agent	Organization of diagnostics and control of student's knowledge. Recording events during diagnostics.	Student, data channels between agents.	Formation of data. Transfer of data to agents at their request.	Agent Requests
The planning and adjustment of training agent	Implementation of control. Formation of control actions. Optimization of the educational program. Fixing events.	Data transmission channels between agents, taxonomy of the educational program, methodological and didactic technologies.	Deciding on the form of control action. Organization of decision implementation.	Agent requests
The agent of Decision-Maker (teacher)	Organization of procedures for the formation of a reference model of professional competencies.	User, data transmission channels between agents.	Display of dialog boxes.	Keyboard input. Agent requests
Competencies making agent	Formation of the current model of the state of professional competencies.	Data transmission channels between agents, reference models of professional competencies.	Formation and display of a histogram of training and a diagram of the formation of competencies.	Agent Requests
Agent without database	Providing a communication channel.	Data transfer channels between agents, knowledge bases, databases.	Single-tier and multi-tier database applications.	Agent Requests

4 Discussion

The analysis of scientific knowledge and pedagogical practice demonstrates that the student's tests can be effective. Pedagogical experiment was conducted at Maritime Professional College of Kherson State Maritime Academy (Ukraine) to check the results on implementation of tests in the agent-based model of the system for the formation of professional competences in subject "Ship electric power systems" [19]. 75 cadets participated in the experiment. They all study at ship engineering department, male, aged 17–18, 3rd year of study. Participants were divided into two groups: control group (38 cadets) and experimental group (37 cadets).

By analyzing the data after the experiment, namely – the test on LMS MOODLE of education establishment, one can observe that the current state of formation of the professional competence of future ship engineers changed positively in experimental group mainly.

During the work in two groups the teacher monitored each group dynamic and individual activity of each student. After the tests, by the end of each module, a considerable increase in the high level of professional competence was shown in experimental group, while the level of professional competence in control group changed slightly [20].

The part of e-course "Ship electric power systems" on LMS Moodle with test by the end of the module is shown in figure 3.

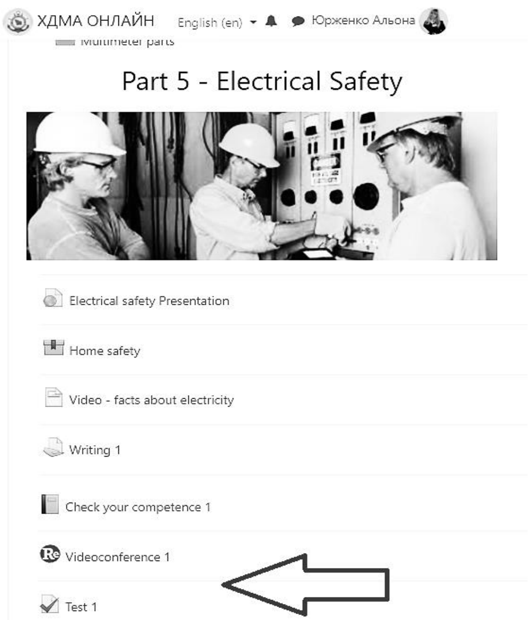


Fig. 3. Part of e-course "Ship electric power systems" with the test by the end of module

Results of five test according to number of modules per semester were taken to find out each student's level of professional competence. According to results after pedagogical experiment control group participants have following levels: low – 16 cadets (37.5 %); medium – 13 cadets (34%); sufficient – 9 cadets (24.5%). While experimental group participants have: low level – 5 cadets (13.5%); medium level – 20 cadets (54%); sufficient level – 12 cadets (32.5%).

Detailed graphical representation of results before and after pedagogical experiment can be seen in figure 4.

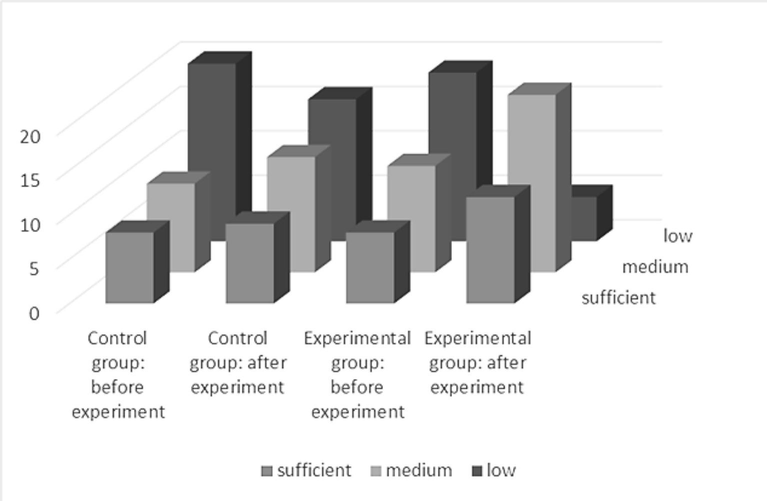


Fig. 4. Results on the levels of professional competence of future ship engineers before and after pedagogical experiment

In order to get feedback from the participants of experimental group LMS MOODLE Module Survey was created. The results of survey have shown student's positive assessment of studying. Gathered data reflected student's attitude to experimental learning which they found stimulating.

5 Conclusions

The development of information and communication technologies goes not only along the way of expanding the field and diversity of their application, but also leads to the emergence of a new multi-level hierarchy of hardware, software, conceptual entities that require new concepts, methods and tools for understanding and managing in the heterogeneous structure of the global information space. Agent-oriented approach allows to rise to a new level of conceptualization and intellectualization of modern information and communication systems. The achievements of the agent-oriented approach are various mathematical models of agents and MAS, concepts and methodologies of multi-agent design and programming, agent programming languages, and sufficiently developed tools and platforms for implementing multi-agent applications.

At the same time, the agent-oriented approach has not yet become the leading paradigm for building corporate industrial applications and information systems, and requires the development of efficient and understandable and practical AOS programming architectures and methodologies, needs to be promoted among system engineers and software developers.

Forecasting learning outcomes will make it possible to analyze the quality of education, to see the degree of assimilation of educational material by students, to detect a discrepancy in competencies between the discipline and the requirements of firms, and to assess the possibility of employment of graduates. An analysis of the literature shows that the solution of the listed problems is carried out, as a rule, by non-formal methods, which reduces their practical value. The modern level of information technology makes it possible to develop qualitatively new models that combine the advantages of mathematical methods, statistics, theory of neural networks, and programming. With the advent of the theory of multi-agent systems, it became possible to create models of complex distributed and heterogeneous systems, the class of which is the object under the study.

References

[1] Diahyleva, O., Gritsuk, I., Kononova, O., Yurzhenko, A.: Computerized adaptive testing in educational electronic environment of maritime higher education institutions. CEUR Workshop Proceedings, 8th Workshop on Cloud Technologies in Education, 2879, pp. 411–422. CTE 2020, Kryvyi Rih (2020). URL: http://ceur-ws.org/Vol-2879/paper23.pdf

[2] Zaytseva, T., Kravtsova, L., Tereshchenkova, O., Yurzhenko, A.: Simulation modeling as a means of solving professionally-oriented problems in maritime industry. CONFERENCE 2021, ISDMCI, vol. 77, pp. 94–106. Zalizny Port, Kherson oblast, Ukraine (2021). doi:10.1007/978-3-030-82014-5_7

[3] Leshchenko, A., Paziak, A., Diahyleva, O., Masonkova, M., Yurzhenko, A.: Advanced teachers training in the remote mode. CEUR Workshop Proceedings, 2nd Workshop on Technology Enhanced Learning Environments for Blended Education – The Italian E-Learning Conference, 3025, pp. 1–5. teleXbe, Foggia, Italy (2021). URL: http://ceur-ws.org/Vol-3025/ paper1.pdf

[4] Won, D., Jansen, P., Carbonell, J.: Minimizing and recovering from the effect of concept drift via feature selection. CONFERENCE 2020, ECAI, vol. 325, pp. 1–7. Santiago de Compostela, Spain (2020). URL: https://ecai2020.eu/papers/1661_paper.pdf

[5] Abdalla, R., Mishra, A.: Agent-oriented software engineering methodologies: Analysis and future directions". Complexity 2021, 1–21 (2021). URL: https://www.hindawi.com/journals/complexity/2021/1629419/

[6] Panisson, A., McBurney, P., Bordini, R.: A computational model of argumentation schemes for multi-agent systems. Argument and Computation 12(3), 357–395 (2021). doi:10.3233/AAC-210555

[7] Panisson, A., Bordini, R.: Towards a computational model of argumentation schemes in agent-oriented programming languages. CONFERENCE 2020, WI-IAT, pp. 9–16. Melbourne, Australia (2020) doi:10.1109/WIIAT50758.2020.00007
[8] Ahlbrecht, T., Dix, J., Fiekas, N., Krausburg, T.: The 15th Multi-Agent Programming Contest: Assemble with Care. Lecture Notes in Computer Science 12947, 3–20 (2021). doi:10.1007/978-3-030-88549-6_1.
[9] Croatti, A., Ricci, A.: Autonomous Agents and Multiagent Systems. CONFERENCE 2021, AAMAS, vol. 3, pp. 1712–1714 (2021)
[10] Iotti, E., Petrosino, G., Monica, S., Bergenti, F.: Two agent-oriented programming approaches checked against a coordination problem. CONFERENCE 2021, DCAI, vol. 1237, pp. 60–70 (2021). doi:10.1007/978-3-030-53036-5_7
[11] Bordini, R., Fallah Seghrouchni, A., Hindriks, K., Logan, B., Ricci, A.: Agent programming in the cognitive era. Autonomous Agents and Multi-Agent Systems. CONFERENCE 2020, AAMAS, vol 34(2), pp. 1–4 (2020). doi:10.1007/s10458-020-09453-y
[12] Ivashkin, Y., Nikitina, M.: Agent-oriented modeling and optimization of material flows of multi-assortment production. CONFERENCE 2019, MMMC, vol. 52(13), pp. 660–664. Berlin, Germany (2019). doi:10.1016/j.ifacol.2019.11.109
[13] Shiang, C., Wai, S., Jalia, N., Bin Khairuddin, M.: Agent oriented methodology for crime modelling and simulation. International Journal of Recent Technology and Engineering 8 (2), 1359–1365 (2019) doi:10.35940/ijrte.B1067.0882S819
[14] François, S., Ferber, J., Stratulat, T., Michel, F.: A hierarchical representation of behaviour supporting open ended development and progressive learning for artificial agents. Autonomous Robots 45, 245–264 (2021). doi:10.1007/s10514-020-09960-7
[15] Shoham, Y.: Agent-oriented programming. Artificial Intelligence 60 (1), 51–92 (1993). doi: 10.1016/0004-3702(93)90034-9
[16] Jennings, N., Wooldridge, V.: Applications of Intelligent Agents. Queen Mary West. eld College, University of London 27, 1–25 (2000).
URL: https://www.academia.edu/14215931/1_Applications_of_Intelligent_Agents
[17] Parunak., H.: Applications of Distributed Artificial Intelligence in Industry". Foundations of Distributed Artificial Intelligence 1–18 (1994).
URL: https://www.emse.fr/~boissier/enseignement/atelier00/jennings.pdf
[18] Dyagileva, O., Masonkova, M., Paziak, A., Striuk, A., Yurzhenko, A.: Intelligent information technology in the system of teachers' advanced training. CEUR Workshop Proceedings, 1nd International Workshop on Information Technologies: Theoretical and Applied Problems, 3039, 306–313. ITTAP, Ternopil, Ukraine (2021).
[19] Xiao, Z., Zheng, X., He, Y., Tai, N., Cheng, H., Shen, J.: An additional bus inductance-based protection scheme for shipboard DC zonal electric distribution systems. International Journal of Electrical Power and Energy Systems 138, pp. 107910 (2022).
doi:10.1016/j.ijepes.2021.107910
[20] Zinchenko, S., Tovstokoryi, O., Ben, A., Nosov, P., Popovych, I., Nahrybelnyi, Y.: Automatic optimal control of a vessel with redundant structure of executive devices. CONFERENCE 2021, ISDMCI, vol. 77, pp. 1–16 (2022). doi:10.1007/978-3-030-82014-5_18