МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кваліфікаційна наукова
праця на правах рукопису

СИНЬКО АННА ІВАНІВНА

УДК 004.91+004.773.2+004.912

ДИСЕРТАЦІЯ

МЕТОДИ І ЗАСОБИ ФОРМУВАННЯ СУСПІЛЬНО ОРІЄНТОВАНОЇ ДОКУМЕНТАЦІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ЗА ДОПОМОГОЮ ВІРТУАЛЬНИХ СПІЛЬНОТ

Спеціальність 121 – «Інженерія програмного забезпечення»
12 – «Інформаційні технології»

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

А.І. Синько

Науковий керівник:
Жежнич Павло Іванович
доктор технічних наук, професор

Львів – 2023

АНОТАЦІЯ

Синько А.І. Методи і засоби формування суспільно орієнтованої документації програмного забезпечення за допомогою віртуальних спільнот. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття ступеня доктора філософії за спеціальністю 121 «Інженерія програмного забезпечення» (12 – «Інформаційні технології»). – Національний університет «Львівська політехніка», Львів, 2023.

У дисертаційній роботі розв’язано актуальну науково-прикладну задачу у галузі інженерії програмного забезпечення – розроблення програмного комплексу щодо формування суспільно орієнтованої документації програмного забезпечення за допомогою віртуальних спільнот, який є необхідним для підвищення ефективності створення документації програмного забезпечення та покращення процесів її формування шляхом впровадження відповідних методів і засобів.

Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку літературних джерел та додатків.

В першому розділі «Аналіз особливостей документації програмного забезпечення у віртуальних спільнотах» розглянуто поняття документації, її види, основні функції, сфери застосування. Відображено життєвий цикл розробки програмного забезпечення, кожен етап якого повинен бути задокументований. Описано призначення документації програмного забезпечення, її складові, джерела розміщення та групи споживачів. Наведено аналіз розвитку ІТ діяльності в Україні, який відображає попит щодо створення належної документації.

Розглянуто віртуальні спільноти, як джерела даних, що можуть містити цінне інформаційне наповнення для формування документації програмного забезпечення. Описано їх структуру, види, характеристики та переваги застосування.

Проаналізовано сучасні технології пошуку і аналізу даних, які розміщені у віртуальних спільнотах. Застосування контент-аналізу дозволило охарактеризувати кожну спільноту, з точки зору змісту, тексту та контексту. Через те, що кожна віртуальна спільнота має різнорідну структуру для їх опису застосовано стандарт Dublin Core Metadata Initiative, що уніфіковує метадані, проаналізувавши які, було представлено співвідношення між елементами.

Також, визначено, що сформована документація програмного забезпечення потребує перевірку якості, тому в роботі наведено моделі оцінювання якості, державні та міжнародні стандарти, дослідивши які, було обрано стандарт ISO/IEC-25010. Через те, що документація програмного забезпечення є інформаційним продуктом для аналізу якості застосовано наступні характеристики: функціональна придатність, придатність до використання, супроводжуваність.

У другому розділі наведено опис формальної моделі сховища консолідованих даних, яке побудовано на основі понятійного апарату спільнот бо вони мають подібні, узагальнені структури. Всі дані поділено на такі групи: статичні, що визначені як метадані; та динамічні, які можуть змінюватися з часом.

Формування документації передбачає її математичний опис, в результаті чого побудована формальна модель документації програмного забезпечення, що містить набір класифікаторів, джерел даних, статей та авторів, що їх розміщають.

Для заповнення статтями документації, що має деревовидну структуру термінів, побудовано метод автоматизованого виявлення термінів статей на основі розроблених масок ключових фраз з ваговими коефіцієнтами приналежності до відповідного терміну. В результаті чого для кожного терміну будується дерево прийняття рішення, яке визначає його вагомість щодо змісту статті.

Формування якісного інформаційного наповнення документації програмного забезпечення передбачає перевірку вмісту статей щодо

достовірності розміщених даних, одним із способів яких є аналіз рейтингу. Дослідивши наявні реакції віртуальних спільнот (оцінювання, взаємодії та коментування), побудовано сценарії обчислення рейтингу допису відповідно до їх набору у тій чи іншій спільноті. Реакція коментування визначається за допомогою аналізу тональності текстів.

Також, у розділі наведено опис нормалізації україномовного та англомовного текстів, який реалізовано за допомогою процесу лематизації, що є необхідним для проведення аналізу вмісту допису за різними методами, які наведені в цій роботі.

Забезпечення унікального інформаційного наповнення документації програмного забезпечення передбачає застосування методу визначення подібності між текстами дописів, та виявлення однакового тексту за моделлю N-грам. При виявленні подібних текстів дописів було здійснено процедуру об’єднання дискусій в яких знаходяться ці дописи.

Через те, що користувач може бути зареєстрованим та розміщувати дописи у різних спільнотах розроблено метод виявлення профілів користувача на основі аналізу інформаційного наповнення дописів та персональних даних, опублікованих їм, що дозволило, за допомогою встановлених спільних характеристик побудувати дерево для прийняття рішень та об’єднати профілі, які не відповідають умові унікальності.

У розділі наведено опис застосування різних значень показників порогових оцінок для забезпечення універсальності та адаптивності системи до різних ситуацій, а також розроблено словники термінів, що необхідні для реалізації описаних методів.

В третьому розділі відображено алгоритми пошуку, відбору цінного інформаційного наповнення з віртуальних спільнот для подальшої обробки у сховищі даних. В результаті чого розроблено алгоритм визначення адекватності тем віртуальних спільнот як джерела цінної інформації, що має відповідати

вимогам релевантності і актуальності (вимірюється абсолютним приростом даних).

Через те, що важливим елементом ІТ-сфери є наявність своєчасних та перевірених щодо достовірності даних, тому введено терміни актуальність і авторитетність дискусій, а також описано процедуру їх обчислення.

У розділі розроблено алгоритм виявлення та усунення дублювання дописів та даних в них, що дозволяє забезпечити наявність унікального інформаційного наповнення у документації програмного забезпечення.

Відомо, що вміст дописів віртуальних спільнот може містити небажане інформаційне наповнення, якого не повинно бути у документації програмного забезпечення, тому текстова частина дописів потребує перевірки до того, як він стане статтею документації. В результаті чого було розроблено алгоритми перевірки тексту на наявність заборонених слів, гіперпосилань та файлів на основі сформованого словника.

Так як документація програмного забезпечення має деревовидну структуру тому, для оптимального розподілу інформаційного наповнення, було побудовано алгоритм на основі критеріїв балансування дерев: довжини та ширини (за допомогою апарату теорії B-дерев), ваговий розподіл, що забезпечило виконання показників якості відповідно до характеристик стандарту ISO/IEC- 25010.

У четвертому розділі побудована архітектура програмного комплексу формування документації програмного забезпечення, яка реалізована у вигляді мультиагентної системи, що складається з таких агентів: збору даних, перевірки контенту, структурування інформаційного наповнення та оцінювання якості. Кожен агент є незалежним і виконує завдання відповідно до розроблених методів та алгоритмів, наведених у другому та третьому розділах. Через що було практично відображено реалізацію задач агентів та структури даних з якими вони працюють.

У розділі, також, наведений опис щодо обчислення мір якості за показниками, характеристиками стандарту ISO/IEC-25010, що дозволило оцінити сформовану документацію програмного забезпечення як кінцевий інформаційний продукт.

Ключові слова: програмне забезпечення, документація, віртуальна спільнота, мультиагентна система, сховище консолідованих даних, аналіз подібності текстів, стандартизація тексту, модель N-грам, дерева прийняття рішень, релевантність, авторитетність, рейтинг допису, актуальність даних, профілі користувача, стандарт ISO/IEC-25010, B-дерева.

Список опублікованих праць за темою дисертації

1. Synko A. Software development documenting – documentation types and standards / A. Synko, A. Peleshchyshyn // Scientific Journal of the Ternopil National Technical University, 2020. – №2 (98). – P.120-128. DOI: https://doi.org/10.33108/visnyk_tntu2020.02

2. Марковець О. В. Формування якісної технічної документації до програмного забезпечення / О.В. Марковець, А.І. Синько // Вісник ВПІ, 2021. – Вип. 2. – С. 98–106. – DOI: https://doi.org/10.31649/1997-9266-2021-155-2-98-106

3. Synko A. Application of clusterization for analysis of virtual community users / A. Synko, K. Molodetska // CEUR Workshop Proceedings: Proceedings of the Symposium on information technologies & applied sciences (IT&AS 2021), Bratislava, Slovak Republic, March 5, 2021. – Vol. 2824 – P. 9-19. – Available at: https://ceur-ws.org/Vol-2824/paper2.pdf

4. Синько А. Аналіз архітектури – «моделі представлення 4+1» / А. Синько // Інформація, комунікація, суспільство 2022 [Електронний ресурс] : Матеріали 10-ї Міжнародної наукової конференції ICS-2021. – Львів: Видавництво Львівської політехніки, 2021.

5. Синько А. І. Представлення типів віртуальних спільнот, що можуть бути джерелом документації до програмного забезпечення / А.І. Синько // Інформація

та соціум: збірник матеріалів VII Міжнародної науково-практичної конференції (Вінниця, 3 червня 2022 р.). – 2022. – С. 26–27.

6. Синько А. Побудова формальної моделі документації до програмного забезпечення / А. Синько // Інформація, комунікація, суспільство 2022: Матеріали 11-ї Міжнародної наукової конференції ICS-2022. – Львів: Видавництво Львівської політехніки, 2022.

7. Synko A. The method of trust level of publications hosted in virtual communities / A.Synko // Scientific Journal of TNTU (Tern.), 2022. – Vol. 105. – no 1. P. 68–79. DOI: https://doi.org/10.33108/visnyk_tntu2022.01.068

8. Синько А. Архітектура системи формування документації програмного забезпечення за допомогою віртуальних спільнот / А. Синько // Вісник Хмельницького національного університету. Технічні науки. – 2023. № 1. – С.248-252. – DOI: https://doi.org/10.31891/2307-5732-2023-317-1-248-252

9. Синько А. І. Суспільно орієнтована документація програмного забезпечення // Integration of education, science and business in modern environment: winter debates: IV Міжнародна науково-практична інтернет-конференція, 23–24 лютого, 2023, Дніпро / WayScience. – 2023. – С. 259–260.

10. Синько А. І. Аналіз якості документації програмного забезпечення / А.І. Синько // Scientific research and innovation: Матеріали II-ої Міжнародної науково-практичної інтернет-конференції, 3–4 квітня, Дніпро, / WayScience. – 2023. – С. 344–345.

11. Синько А. Метод автоматизованого виявлення термінів статей за допомогою дерева для прийняття рішень / А. Синько, П. Жежнич // Вісник Хмельницького національного університету. Технічні науки. – 2023. № 2. – С.339-344.

ABSTRACT

Synko A.I. Methods and means of socially oriented documentation of software through virtual communities. – Qualifying scientific work on the rights of the manuscript.

Thesis paper for achievement of the scientific degree Doctor of Philosophy in the specialty 121 «Software Engineering» (12 – «Information technologies»). – Lviv Polytechnic National University, Lviv, 2023.

The thesis paper solved the relevant scientific and applied task in the sphere of software engineering – development of a software complex of socially oriented documentation of software through virtual communities which is necessary to increase the effectiveness of creating a software documentation and improving processes of its formation by implementing methods and means.

The thesis paper consists of introduction, four chapters, conclusions, list of references and appendixes.

In the first chapter entitled «Analysis of features of software documentation in virtual communities», the concept of documentation, its types, main functions, areas of application have been considered. The life cycle of software development has been reflected, each stage of which should be documented. The purpose of software documentation, its components, publication sources and user groups have been described. The analysis of the development of IT activities in Ukraine which reflects the demand for the creation of appropriate documentation, has been provided.

Virtual communities have been considered as sources of data that can contain valuable information content for the formation of software documentation. Their structure, types, characteristics and benefits of application have been described.

Modern technologies for searching and analyzing data placed in virtual communities have been studied. The application of content analysis made it possible to characterize each community in terms of content, text and context. Due to the fact that each virtual community has a heterogeneous structure, the Dublin Core Metadata

Initiative standard, which unifies metadata, has been used to describe them. After analyzing the metadata, the relationship between the elements has been defined.

Moreover, it has been established that the formed software documentation needs a quality check, therefore, the work provides quality assessment models, state and international standards. The ISO/IEC-25010 standard has been chosen to analyze the quality of software documentation as an information product. The following characteristics have been used: functional suitability, usability, maintainability.

The second chapter provides a description of the formal model of the consolidated data warehouse, which has been built on the basis of the conceptual apparatus of the communities because they have similar, generalized structures. All data is divided into the following groups: static, defined as metadata, and dynamic, which can change over time.

The formation of documentation involves its mathematical description, as a result of which a formal model of software documentation has been developed. This model contains a set of classifiers, data sources, articles and authors that published them.

Software documentation has a tree-like structure of terms. In order to fill the documentation articles, the method of automated detection of article terms has been applied on the basis of developed masks of key phrases with weighting coefficients of belonging to the corresponding term. As a result, a decision tree has been created for each term, which determines its importance in relation to the content of the article.

The formation of high-quality information content of software documentation involves checking the content of articles regarding the accuracy of posted data, one of the methods of which is rating analysis. Having studied the existing reactions of virtual communities (ratings, interactions and commenting), scenarios for calculating the rating of a post in accordance with their set in this or that communities have been built. The reaction of commenting has been determined by the analysis of text tonality.

Also, the chapter outlines the normalization of Ukrainian and English texts, which has been implemented using the lemmatization process. The latter is necessary

for the analysis of the content of the post using various methods, which have been implemented in this work. Providing a unique informational content of the software documentation involves the application of the method of determining similarities between the texts of posts, and identifying the same text using the N-gram model. When detecting similar texts of descriptions, the procedure of combining discussions, in which these posts remain, has been introduced.

Due to the fact that a user can be registered and publish posts in different communities, a method for identifying user profiles has been applied. This method is based on the analysis of the informational content of posts and personal data published by a user. Owing to the established common characteristics, a decision tree and merged profiles, which do not meet the uniqueness condition, have been regarded.

The chapter provides a description of the application of different values of threshold evaluation indicators to ensure the universality and adaptability of the system to various situations. Dictionaries of terms necessary for the implementation of the described methods have also been developed.

In the third chapter, the algorithms of search and selection of valuable information content from virtual communities for further processing in the data warehouse have been displayed. As a result, an algorithm of determining the adequacy of themes of virtual communities as a source of valuable information that should meet the requirements of relevance and actuality (measured by the absolute increase in data) has been developed.

Due to the fact that an important element of the IT sphere is the availability of timely and verified data, therefore, the terms of relevance and authority of discussions have been introduced. The procedure for their calculation has been also described.

An algorithm for detecting and eliminating duplication of posts and data in them, which makes it possible to ensure the available of unique information in the software documentation, has been developed in the chapter.

It is known that the content of posts of virtual communities can consist of unwanted information that should not be in the software documentation, so the textual

part of the posts needs to be checked before it becomes an article of the documentation. Correspondingly, the algorithms to check the text for the presence of prohibited words, hyperlinks and files have been provided. These algorithms have been performed on the basis of the developed dictionary.

Since the software documentation has a tree-like structure, for the optimal distribution of information content, an algorithm has been built based on tree balancing criteria: length and width (using the B-tree theory apparatus), weight distribution. The implementation of the structuring algorithm ensured quality performance in accordance with the characteristics of the ISO/IEC-25010 standard.

In the fourth chapter, the architecture of the software complex for the formation of software documentation has been built. The system has been implemented in the form of a multi-agent system consisting of the following agents: data collection, content verification, structuring of information content and quality assessment. Each agent is independent and performs tasks according to the developed methods and algorithms presented in the second and third chapters. Accordingly, the implementation of agents' tasks and the data structure with which they work have been practically reflected.

The section also provides a description of the calculation of quality measures based on the indicators and characteristics of the ISO/IEC-25010 standard, which has made it possible to evaluate the generated software documentation as a final information product.

Keywords: software, documentation, virtual community, multi-agent system, consolidated data warehouse, text similarity analysis, text standardization, N-gram model, decision trees, relevance, authority, post rating, data relevance, user profiles, ISO/IEC-25010 standard, B-trees.

List of publications by the subject of dissertation

1. Synko A. Software development documenting – documentation types and standards / A. Synko, A. Peleshchyshyn // Scientific Journal of the Ternopil National

Technical University, 2020. – №2 (98). – P.120-128. DOI: https://doi.org/10.33108/visnyk_tntu2020.02

2. Markovets O. V. Formation of High-Quality Technical Documentation for Software / O.V. Markovets, A.I. Synko // The journal “Visnyk of Vinnytsia Polytechnical Institute”, 2021. – No. 2. – P. 98–106. – DOI: https://doi.org/10.31649/1997-9266-2021-155-2-98-106

3. Synko A. Application of clusterization for analysis of virtual community users / A. Synko, K. Molodetska // CEUR Workshop Proceedings: Proceedings of the Symposium on information technologies & applied sciences (IT&AS 2021), Bratislava, Slovak Republic, March 5, 2021. – Vol. 2824 – P. 9-19. – Available at: https://ceur-ws.org/Vol-2824/paper2.pdf

4. Synko A. Architecture analysis – «4+1 representation models» / A. Synko // Information, communication, society 2015: Materials of the 10th International sciences conf. ICS-2021. – Lviv: Publishing House of Lviv Polytechnic National University, 2021.

5. Synko A. I. Representation of the types of virtual communities that can be a source of documentation for the software / A.I. Synko // Information and society: Collection of materials of the VII International scientific and practical conference (Vinnitsa, 3 June 2022), 2022. – P. 26–27.

6. Synko A. Construction of a formal model of software documentation / A. Synko // Information, communication, society 2022: Materials of the 11th International sciences conf. ICS-2022. – Lviv: Publishing House of Lviv Polytechnic National University, 2022.

7. Synko A. The method of trust level of publications hosted in virtual communities / A. Synko // Scientific Journal of TNTU (Tern.), 2022. – Vol. 105. – no 1. P. 68–79. DOI: https://doi.org/10.33108/visnyk_tntu2022.01.068

8. Synko A. Architecture of the software documentation system through virtual communities / A. Synko // "Bulletin of the Khmelnitsky National University Series:

Technical Sciences", 2023. № 1. – P.248-252. – DOI: https://doi.org/10.31891/2307-5732-2023-317-1-248-252

9. Synko A. I. Socially oriented software documentation // Integration of education, science and business in modern environment: winter debates: proceedings of the 4th International scientific and practical Internet conference, February 23–24, 2023, Dnipro / WayScience, 2023. – P. 259–260.

10. Synko A. I. Analysis of software documentation quality / A.I. Synko // Scientific research and innovation: proceedings of the 2nd International scientific and practical Internet conference, April 3–4, 2023, Dnipro / WayScience, 2023. – P. 344–345.

11. Synko A. Method of automated detection of article terms using a decision tree / A. Synko, P. Zhezhnych // "Bulletin of the Khmelnitsky National University Series: Technical Sciences", 2023. № 2. – P.339-344.

Зміст

Перелік умовних скорочень

Скорочення, термін, позначення	Пояснення
БД	База даних
ВС	Віртуальна спільнота
ДПЗ	Документація програмного забезпечення
ЖЦ	Життєвий цикл
ЗП	Заборонені посилання
ЗС	Заборонені слова/словосполучення
ЗФ	Заборонені файли
ІП	Інформаційний продукт
ІТ	Інформаційні технології
НІН	Небажане інформаційне наповнення
ПД	Програмна документація
ПЗ	Програмне забезпечення
ПП	Програмний продукт
СКД	Сховище консолідованих даних
API	Application Programming Interface – Прикладний програмний інтерфейс
DOM	Document Object Model – Об’єктна модель документа
ISO	Міжнародна організація із стандартизації
NLTK	Natural Language Toolkit – це набір бібліотек і програм для символьної та статистичної обробки природної мови для англійської мови, написаних мовою програмування Python.
XML	Extensible Markup Language – стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками.
URL	Uniform Resource Locator – стандартизована адреса певного ресурсу
VADER	Valence Aware Dictionary and Sentiment Reasoner – інструменту аналізу настроїв
WWW	World Wide Web – всесвітня павутина

Список рисунків

Список таблиць

Вступ

Актуальність теми. Епоха четвертої промислової революції супроводжується автоматизацією всіх процесів та етапів виробництва. Особливістю розробки програмного забезпечення в Індустрії 4.0 є його адаптивність до потреб, побажань споживача, отримати інформацію про які можна у мережі Інтернет [38, 81]. Більше того, виробники програмного забезпечення (ПЗ) разом з розробкою програмних продуктів (ПП) створюють супровідні віртуальні спільноти (ВС), де користувачі можуть обговорювати ПП у вигляді документування своїх знань та задовольняючи власні потреби [68].

Через високий попит на ІТ-послуги, що перевищує пропозицію на ринку, [10, 31, 67] ІТ-компанії, які займаються розробкою програмних продуктів більшу частину ресурсів приділяють створенню самого ПЗ, в той час як формуванню супровідної документації не надають достатньо уваги [46]. Про це говорить наявність гнучких підходів до управління проектами та розробки ПЗ, які зазвичай не надають перевагу документуванню проекту – наприклад, підхід Agile, що супроводжується документом Agile Manifesto. Цей документ визначає цінності та принципи, якими повинна керуватися команда при розробці ПЗ, в одній із ідей якого описано наступне – «Робоче програмне забезпечення важливіше, ніж повна документація» [102].

Документація ПЗ (ДПЗ) – набір взаємопов’язаних та складених у визначеній формі супровідних документів до ПЗ, що містять інформацію щодо опису загальних положень необхідних для ознайомлення перед його використанням. В багатьох випадках формування ДПЗ на проекті відбувається по залишковому принципу ресурсів. Це зумовлено тим, що створення ДПЗ потребує додаткових затрат (часових, ресурсних та фінансових), які не кожна компанія може собі дозволити або вважає за потрібне. В результаті чого, ДПЗ може бути взагалі відсутня або містити неповні, застарілі дані, які впливають на використання ПЗ споживачами. Відповідно виникає потреба в автоматизованому формуванні ДПЗ, що буде орієнтована на споживачів серед яких виступають:

- розробники, які дороблюють/оновлюють існуюче ПЗ або займаються інтеграцією певного ПЗ щодо інших програмних продуктів;
- кінцеві користувачі, які хочуть придбати ПЗ та/або використовувати його у повній мірі.

Як виробники так і кінцеві користувачі можуть бути учасниками ВС і розміщувати інформацію про ПЗ тим самим документуючи її. Через те, що ІТ-галузь потребує супровід ДПЗ, що містить опис програмного продукту з різних точок зору, тому віртуальні спільноти постають корисним джерелом даних, адже можуть містити як інформацію, що надає виробник, розробник ПП, так і досвід користувачів, які вже придбали та використовували його.

До недоліків розміщеної інформації у віртуальних спільнотах належать [68]:

- швидкий ріст обсягів інформації, пошук та обробка яких займе багато часових витрат;
- перевірка достовірності даних, що розміщують користувачі;
- слабка структурованість, щодо подачі інформації, незважаючи на наявність тем, що групують дані довкола певних тематичних ситуацій у спільнотах.

Відмінність суспільно орієнтованої ДПЗ від документації яку надає виробник полягає у тому, що вона створюється суспільством – учасниками ВС, і передається користувачам, споживачам які потребують її у вигляді інформаційного наповнення спільнот, що можна вважати документацією адже досвід користувачів є зафіксований та може бути переданий у просторі і часі.

Наявність належної ДПЗ є важливою складовою ПП і напряму впливає на його придбання та використання в повній мірі споживачами. З точки зору замовника та розробників ПП, ДПЗ, що базується на відгуках користувачів, допоможе виявити слабкі сторони ПП (проблеми, некоректну роботу тощо) та в подальшому покращувати і оновляти ПП. З точки зору кінцевого користувача ознайомитися із ПП і використовувати його в повній мірі.

Через те, що ІТ галузь здебільш експортно орієнтована, тому ПП і супровідна документація повинна відповідати характеристикам державних та міжнародних стандартів, що дозволять виміряти якість сформованої ДПЗ.

Дослідивши основні проблеми віртуальних спільнот, як джерела інформаційного наповнення та великий попит на розробку програмного забезпечення (ПЗ), яке повинно супроводжуватися належною документацією, що має відповідати комплексу стандартів, виникає необхідність у розробці методів та засобів для автоматизованого пошуку, відбору, структурування та подальшого опрацювання інформації, яка може бути корисною для подальшого формування ДПЗ. ДПЗ формується як певний інформаційний продукт, що містить опис щодо ПЗ, і, метою якого є покриття інформаційних потреб споживачів – кінцевих споживачів, розробників ПП.

При цьому важливим завданням постає оцінка якості сформованої документації для визначення чинників що впливають на її погіршення чи покращення, виміряти яку можна за допомогою метрик міжнародного стандарту ISO/IEC-25010, що містить більший набір показників, аніж національні стандарти. Чим якісніше буде сформована ДПЗ тим швидше і ефективніше споживачі знайдуть відповіді на свої питання заощаджуючи свої ресурси.

Тому актуальною є наукова задача щодо підвищення якості формування суспільно орієнтованої документації програмного забезпечення за допомогою інформаційного наповнення віртуальних спільнот шляхом удосконалення існуючих та розроблення відповідних методів і засобів.

Зв'язок роботи з науковими програмами, планами, темами. Тема дисертаційної роботи відповідає науковому напряму кафедри і виконана в межах зареєстрованої наукової тематики "Управління процесами соціальних комунікацій в глобальному інформаційному просторі" (номер держреєстрації 0119U101870).

Мета і завдання дослідження. Метою дослідження є підвищення якості документації програмного забезпечення на основі побудови нових і

вдосконаленні існуючих методів та засобів консолідації інформаційного наповнення віртуальних спільнот.

Для досягнення поставленої мети необхідно розв’язати такі задачі:

- проаналізувати особливості використання інформаційного наповнення віртуальних спільнот для формування документації програмного забезпечення з урахуванням сучасних тенденцій та перспектив розвитку інженерії програмного забезпечення;
- побудувати формальну модель документації програмного забезпечення, інформаційне наповнення якої формується на основі сховища консолідованих даних з різних віртуальних спільнот, та класифікується відповідно до потреб різних категорій користувачів програмного забезпечення;
- розробити методи формування документації програмного забезпечення з урахуванням особливостей інформаційного наповнення віртуальних спільнот;
- розробити алгоритми виявлення цінного інформаційного наповнення віртуальних спільнот для документації програмного забезпечення;
- розробити метод оцінювання якості документації програмного забезпечення;
- провести експериментальне дослідження шляхом проектування програмно-алгоритмічного комплексу та здійснити апробацію отриманих результатів.

Об’єктом дослідження є процес формування суспільно орієнтованої документації програмного забезпечення.

Предметом дослідження є методи і засоби формування документації програмного забезпечення, джерелом інформаційного наповнення якої виступають віртуальні спільноти.

Методи дослідження. Для аналізу основних складових віртуальних спільнот проаналізовано теорію графів, теорію відношень, апарат теорії реляційних баз даних, теоретико-множинні підходи. Для пошуку джерел, що можуть містити релевантне інформаційне наповнення застосовано метод нормалізації україномовного та англомовного текстів – лематизацію, та метод визначення частоти використання слів – показник TF-IDF. Одним з параметрів обчислення рейтингу допису є дослідження коментарів за допомогою контент-аналізу – аналіз тональності тексту дописів-коментарів. Виявлення термінів та обчислення їх вагових коефіцієнтів описано на основі методу пошуку та прийняття рішень. Для уникнення дублювання даних в документації вміст дописів проаналізовано за методом виявлення косинусоїдної подібності та моделлю N-грам. Для забезпечення оптимального розподілу інформаційного наповнення документації, що має деревовидну структуру, застосовано методи балансування на основі апарату теорії B-дерев. Оцінювання якості сформованої документації програмного забезпечення проведено з використанням методу, в основі якого лежить міжнародний стандарт оцінювання якості програмних продуктів ISO/IEC-25010.

Наукова новизна одержаних результатів полягає в науковому обґрунтуванні та розробленні методів і засобів формування суспільно орієнтованої документації програмного забезпечення за допомогою віртуальних спільнот. При цьому отримано такі наукові результати:

- вперше побудовано формальну модель документації програмного забезпечення, яка враховує особливості інформаційного наповнення віртуальних спільнот та потреби різних категорій користувачів програмного забезпечення через множину класифікаторів, що дозволило побудувати методи та алгоритми формування документації програмного забезпечення, і подальшого оцінювання за показниками якості;

- вдосконалено методи: автоматизованого виявлення вагомих, значущих термінів статей на основі масок ключових фраз з ваговими коефіцієнтами з використанням дерева для прийняття рішень; поєднання методу виявлення подібності та моделі N-грам (без вилучення і з вилученням незначущих, стоп-слів) для виявлення однакового тексту в дописах, що забезпечило формування якісної документації програмного забезпечення;
- набув подальшого розвитку метод оцінювання якості документації програмного забезпечення на основі характеристик та під-характеристик міжнародного стандарту ISO/IEC-25010, який враховує особливості документації програмного забезпечення, що дозволило практично оцінювати результати формування документації програмного забезпечення.

Практичне значення одержаних результатів дисертаційної роботи обумовлено тим, що вони дають змогу автоматизовано формувати інформаційне наповнення документації програмного забезпечення, проводити його перевірку, аналіз та оцінювати її якість враховуючи потреби користувачів. Зокрема, практично цінними є такі результати:

- розроблено алгоритми виявлення цінного інформаційного наповнення віртуальних спільнот, які враховують адекватність джерела за умовами релевантності запиту користувача та актуальності вмісту, структурування інформаційного наповнення документації програмного забезпечення за допомогою апарату Б-дерев, усунення небажаного контенту у дописах шляхом виявлення заборонених слів, гіперпосилань та файлів, що дало можливість автоматизовано формувати документацію програмного забезпечення;
- розроблено метрики оцінювання якості документації програмного забезпечення за допомогою показників – функціональної

придатності, придатності використання та модульності, що дозволило практично оцінити результати виконання формування документації програмного забезпечення;

- побудовано архітектуру програмного комплексу формування документації програмного забезпечення за допомогою інформаційного наповнення віртуальних спільнот у вигляді багатоагентної системи, що забезпечило практичну реалізацію методів та алгоритмів у вигляді завдань агентів.

Особистий внесок здобувача. Усі наукові результати дисертаційної роботи отримані автором самостійно. У друкованих працях, опублікованих у співавторстві, автору належать: [69] – здійснено аналіз віртуальних спільнот в мережі Інтернет; [118-119] – аналіз користувачів віртуальних спільнот; [120] – види документації програмного забезпечення; [46, 66, 120] – стандарти оцінювання якості документації програмного забезпечення як інформаційного продукту; [70] – формальна модель документації; [68] – метод автоматизованого виявлення термінів статей за допомогою дерева для прийняття рішень; [67] – архітектура системи формування документації за допомогою інформаційного наповнення віртуальних спільнот.

Апробація результатів дисертації. Результати дисертаційного дослідження апробовано на міжнародних науково-практичних конференціях та семінарах:

- 10 Міжнародна наукова конференція «Інформація, комунікація, суспільство» (ІКС-2021), Львів, Україна, 2021 р.;
- VII Міжнародна науково-практична конференція «Інформація та соціум», Вінниця, Україна, 2022 р.;
- 11 Міжнародна наукова конференція «Інформація, комунікація, суспільство» (ІКС-2022), Львів, Україна, 2022 р.;

- IV Міжнародна науково-практична інтернет-конференція «Integration of Education, Science and Business in Modern Environment: Winter Debates», Дніпро, Україна, 2023 р.;
- 2nd International Scientific and Practical Internet Conference «Scientific Research and Innovation», Дніпро, Україна, 2023 р.

Результати дисертаційних досліджень регулярно доповідалися на наукових семінарах кафедри соціальних комунікацій та інформаційної діяльності Національного університету «Львівська політехніка» (2020-2023).

Публікації. Основні положення дисертації опубліковано у 11 наукових працях, з яких: 5 статей у наукових фахових виданнях України, 1 стаття у наукових періодичних виданнях інших держав та 5 праць – у матеріалах і тезах конференцій.

Структура та обсяг роботи. Дисертаційна робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел (131 найменувань) і 9 додатків. Основний зміст викладено на 147 сторінках друкованого тексту, містить 66 рисунків, 3 таблиці. Загальний обсяг роботи – 205 сторінок.

Розділ 1. Аналіз особливостей документації програмного забезпечення у віртуальних спільнотах

В умовах глобальної комп’ютеризації та значного розвитку інформаційного суспільства спостерігається збільшення попиту на розробку програмних продуктів та швидкий розвиток індустрії комунікаційних технологій в цілому. Разом з тим зростає кількість інформаційних ресурсів у відкритому доступі, що містять корисний контент про програмні продукти. Одним з видів таких ресурсів постають віртуальні спільноти, де користувачі обмінюються досвідом.

При розробці програмного забезпечення створюється і використовується великий обсяг різноманітної документації, що містить опис певного продукту з різних точок зору. Документація програмного забезпечення потрібна: як спосіб обміну інформацією між розробниками програмного продукту, як засіб управління розробкою програми, а також для передачі користувачам інформації, необхідної для застосування і супроводу програмного забезпечення.

Перевірка та оперативне опрацювання всієї доступної інформації з віртуальних спільнот, як джерела даних для формування документації програмного забезпечення, потребує дослідження структури та основних складових віртуальних спільнот, для подальшого приведення даних до єдиного стандартного структурованого вигляду з метою обробки та використання їх інформаційного наповнення залежно від потреб. Також, необхідно розробити загальну структуру документації, і враховувати як процеси пошуку, відбору даних, так і процеси їх аналізу, перевірки, співставлення та перетворення із застосуванням різних методів і засобів.

Основні результати розділу опубліковані автором у працях [46, 69, 70, 71, 118, 119, 120].

1.1. Аналіз документації програмного забезпечення як інформаційного продукту

Згідно ДСТУ 2732:2004 документ – це інформація, що занесена на матеріальний носій, основною функцією якого є збереження та передавання її в часі і просторі [14]. ДСТУ 2392-94 визначає документацію як набір документів, відібраних зі спеціальною метою [13].

Документація означає створення артефакту: саме документа, який може складатися з електронних файлів, веб-сторінок, знімка дошки або паперу [71]. Документація формується для конкретної аудиторії та зберігається на певному носії (наприклад, на диску, в книзі тощо) в заданому форматі [46].

Створення документації вимагає застосування стандартизованої термінології, використання зазначених вимог та положень відповідно до її змісту і складу, уніфікованих форм документів, крім того, дотримання порядку щодо їх виконання [45].

Роль документації відображається в її функціях:

- інформаційна – початковий, поточний стан системи та її подальший розвиток;
- охоронна – опис вимог безпеки, зниження витрат унаслідок навчання персоналу безпечним методам роботи, захист від матеріальної та нематеріальної шкоди (зломи, втрата даних тощо), спрощене ліцензування програмних продуктів;
- технічна – зниження ризиків при будь-якій некоректній роботи системи та заходи щодо її швидкому відновленню;
- пізнавальна – підвищення загальної компетентності персоналу, внаслідок опрацювання інформаційного опису системи та її вузлів.

Документація може бути різною в залежності від області її застосування: бухгалтерською, конструкторською, нормативною, науковою, технічною, товарною тощо.

Під терміном технічна документація вважають систему графічних та текстових документів, що необхідні та достатні для їх використання на всіх стадіях життєвого циклу (ЖЦ) продукції [78].

Технічна документація може бути наступних видів [17, 120]: конструкторська, технологічна, програмна.

Програмне забезпечення (ПЗ) – це сукупність програмних та документальних засобів призначених для створення і експлуатації систем обробки даних засобами обчислювальної техніки [26, 58].

ЖЦ ПЗ – це період, який містить послідовність діяльності певних процесів, який починається з моменту прийняття рішення щодо необхідності розробки ПЗ і закінчується в момент його повного вилучення з експлуатації [3, 8]. А отже відображає повне, цілісне представлення.

Інформаційні технології (ІТ) – сукупність технологій з використання обчислювальної техніки та/або програмного забезпечення для створення, передачі, зберігання, розповсюдження, обміну інформацією та/або даними [92].

Документація ПЗ (ДПЗ) – набір взаємопов’язаних та складених у визначеній формі супровідних документів до ПЗ, що містять інформацію щодо опису загальних положень необхідних для ознайомлення перед його використанням. Згідно ДСТУ ISO/IEC/IEEE 16326:2015 [23] при розробці систем та ПЗ формуються програмні документи, що містять відомості, необхідні для розробки, встановлення, експлуатації та супроводу програм [12], тобто відповідає ЖЦ ПЗ, що відображено на рис.1.1.

Рис.1.1 Основні етапи ЖЦ ПЗ

Розробка програмних продуктів супроводжується саме документацією програмного забезпечення, яка регулюється комплексом національних, міжнародних та міждержавних стандартів [54], основними складовими яких є:

- змістовна частина – структура, вміст, основні компоненти документів що утворюють програмну документацію (ПД);
- допоміжна частина – умовні позначення, правила оформлення, обліку, оновлення (внесення змін), зберігання, позначення документів тощо.

Документування програмного забезпечення є конкретним завданням – наприклад, створення документа вимог або архітектури ПЗ [88]. Проаналізувавши джерела було виділено наступні види документів згідно ЖЦ ПЗ [23, 120]:

- документування вимог. В результаті чого формується документ специфікації вимог до ПЗ або технічне завдання (приклад документу вимог наведено у Додатку А);
- проектна документація ПЗ, що відображає архітектуру системи;
- документація розробки;
- документація інтеграції;
- документація супроводу та експлуатації ПЗ для різних груп користувачів (приклад документу експлуатації наведено у Додатку В).

Документація програмного забезпечення має такі основні призначення [20, 71]:

- описати та зафіксувати інформацію про систему протягом її ЖЦ;
- надати інформацію про систему різним зацікавленим особам;
- сприяти забезпеченню придатності та практичні до супроводу комп’ютерних програм;
- допомогти контролюванню процесів ЖЦ.

ДПЗ визначається двома складовими: інформаційна та матеріальна [45]. До інформаційної частини належить: мова подання, наявність структурних блоків, прогалин, перевірка на достовірність, наперед визначена чітка структура. Натомість матеріальною вважають: спосіб документування чи запису, форма документа та носія інформації тощо.

ДПЗ може бути опублікованою розробниками програмного продукту, а також у вигляді відгуків користувачів, споживачів [70] на таких майданчиках [71]:

- на офіційних сайтах (наприклад, Microsoft);
- у сховищах даних (наприклад, GitHub);
- у соціальних мережах (віртуальних спільнотах, групах тощо).

ДПЗ може зберігатися як електронно так і на матеріальному носії. Якщо документацію утворюють документи, які містять інформацію зафіксовану у вигляді електронних даних та супроводжуються обов'язковими реквізитами документа, то він є електронним документом, в даному випадку документацію можна вважати електронною [27].

Орієнтованість документації на споживача дозволяє представляти ДПЗ як **інформаційний продукт** (ІП), адже ІП – це задокументована інформація, зібрана та представлена відповідно до вимог для її поширення [25]. Тому якість ДПЗ як ІП впливає на задоволеність та покриття всіх потреб її споживачів.

Термін ІП визначений на законодавчому рівні, зокрема такими стандартами:

- ДСТУ 6096:2009 (Система стандартів з інформації, бібліотечної та видавничої справи) [21];
- Закон України «Про інформацію» [28];
- проект Закону України «Про Концепцію національної інформаційної політики» [59].

Згідно статті 34 Конституції України [37], кожен має право на збір, зберігання, використання та розповсюдження інформації, що розміщена у

вільному доступі на просторах Інтернету. Відповідно збір даних, опублікованих у віртуальних спільнотах для формування документації не є забороненою діяльністю.

Основними характеристиками ІП як послуги постають: динамічність, гнучкість, можливість налаштування під індивідуальні побажання та потреби споживача, незнищуваність (можливість багаторазового використання). ІП представляє собою завершену форму, що є сумою характеристик, які визначають як його функціональне призначення, так і зовнішній вигляд.

Інформаційною потребою користувача ПЗ виступає необхідність (потреба) в отриманні конкретної інформації про ПЗ для виконання поставленого завдання та/або задоволення своїх інтересів [45]. Через вплив отримуваних даних інформаційні потреби користувачів постійно змінюються, трансформуються і модифікуються, тому їх неможливо однозначно визначити та описати. Коли інформаційна потреба відображається у вигляді певної послідовності властивих їй значень у фіксовані моменти часу, та висловлюється природною мовою, її можна вважати інформаційним запитом, з яким користувач звертається до системи для пошуку даних.

ДПЗ як ІП повинна враховувати потреби споживачів.

1.1.1. Аналіз розвитку ІТ діяльності в Україні

За останні роки ІТ індустрія зазнала високих темпів зростання відповідно кількість ДПЗ збільшується щоденно. За даними сервісу Statista ринкові світові витрати на розробку ПЗ більш ніж подвоїлися за десятиріччя між 2012 і 2022 роками [121].

ІТ-сфера в Україні також демонструє стабільне зростання з року в рік. Згідно аналізу фірми PwC, український ринок ІТ нещодавно збільшився у 2,5 рази. У 2011-2015 роках його було збільшено на 150%, і на 2021 рік він перевищив 5,7 мільярдів доларів [51]. Вже зараз Україна вийшла на перше місце за обсягом експорту послуг і принесла понад 4% ВВП. Також, відомо, що наша країна є одним із найбільших експортерів ІТ-послуг в Європі [64, 125].

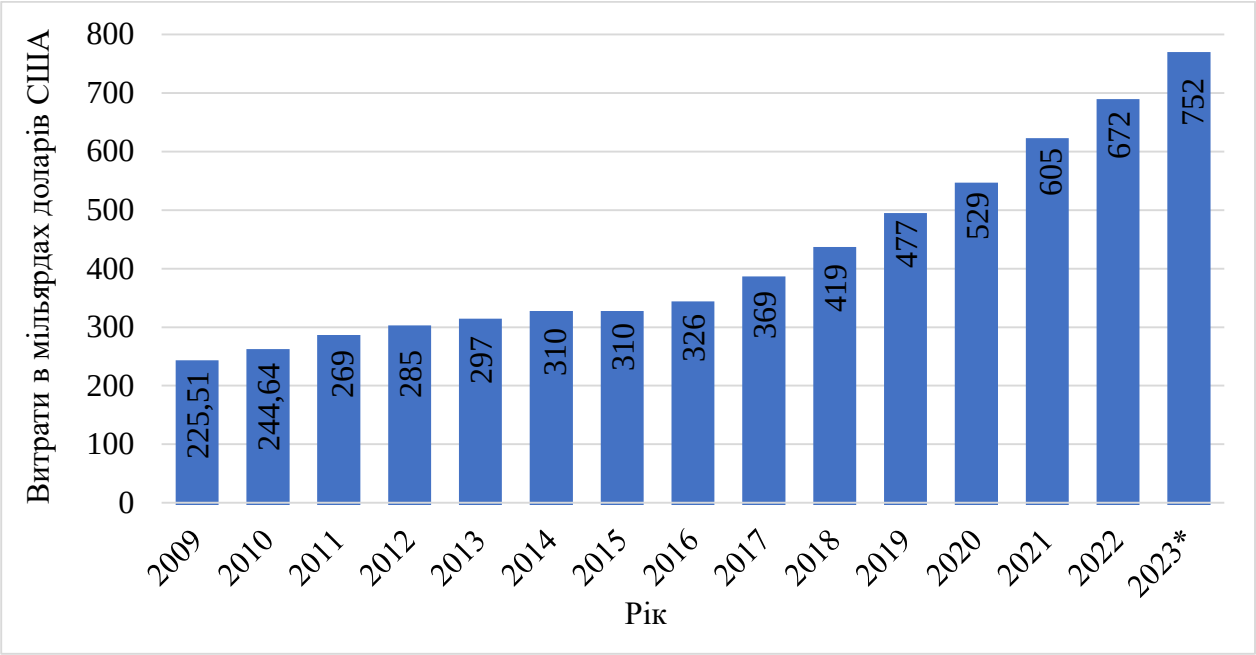


Рис.1.2 Динаміка світових витрат на розробку ПЗ

Українська ІТ-галузь здебільшого є експортно-орієнтована – на рис.1.3. відображено динаміку росту обсягу ІТ-експорту, де спостерігається подвоєння за останні три роки.

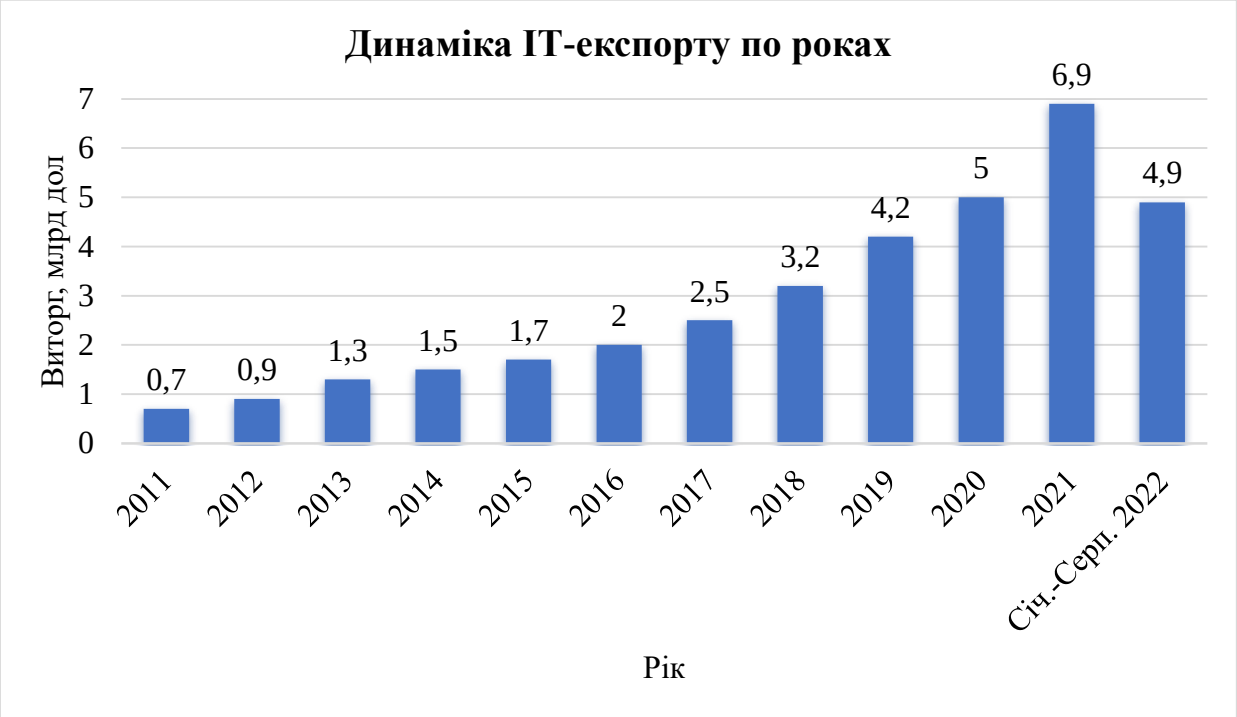


Рис.1.3 Обсяг ІТ-експорту

Слід зауважити, що експорт ІТ-послуг зростає швидше за експорт будь-яких інших товарів чи послуг – статистика за 2015-2020 роки:

Рис.1.4 Обсяг ІТ-експорту відповідно до інших товарів та послуг

Орієнтуючись на дані рис.1.3. та рис.1.4. процес розробки та супроводу програмних продуктів повинен відповідати міжнародним вимогам та стандартам, що передбачають наявність документації до кожного проекту. Тому формування ДПЗ як ІП є актуальним завданням.

В даний час в Україні функціонує понад 1600 компаній, що займаються розробкою ПЗ, і більшість із них надають послуги з розробки ПЗ клієнтам по всьому світу [46].

Українські компанії з розробки ПЗ виділяються серед світової конкуренції завдяки таким чинникам [46]:

- зручному географічному розташуванню;
- культурній близькості;
- якістю надання інформаційних послуг;
- висококваліфікованим фахівцям, які займаються розробкою ПЗ;
- конкурентоспроможним цінам.

Ринок ІТ-послуг – один із секторів ІТ-індустрії, де послуга є об’єктом на ринку інформаційних технологій [6]. ІТ-послуга – комплекс складних послуг організаційного та технічного характеру, спрямованих на підтримку в сфері ІТ [9, 32]. На сьогодні на ринку присутні такі ІТ-послуги: аутсорсинг, ІТ-консультування, обробка та зберігання електронної інформації.

Діяльність у сфері ІТ – господарська діяльність, яка охоплює всі види економічної діяльності під час виконання ЖЦ ПЗ [29]. Результатом

інформаційної діяльності в кожній галузі має бути ІП, який з'являється на ринку у вигляді інформаційних товарів і послуг.

Діяльність у сфері ІТ регламентується на законодавчому рівні, зокрема Закон України «Про стимулювання розвитку сфери інформаційних технологій в Україні» [29], «Про захист інформації в інформаційно-комунікаційних системах», який здійснює регулювання відносин у галузі захисту даних в інформаційних, електронних, комунікаційних та інформаційно-комунікаційних системах [30]. Цей закон можна застосувати до даної сфери, адже галузь ІТ є складовою частиною інформаційно-комунікаційного ринку [6].

Діяльність у сфері ІТ містить відносини між:

- юридичними особами, що є зареєстровані за законодавством України, та їх працівниками чи фізичними особами, які надають послуги та/або виконують роботи для таких суб'єктів господарювання (учасники ПЗ);
- учасниками та споживачами;
- користувачами, споживачами ПЗ.

Учасники ПЗ – фізичні та юридичні особи, які безпосередньо залучені в реалізацію проекту, або чиї інтереси можуть бути порушені при виконанні проекту [11].

Споживачі програмного забезпечення – юридичні та фізичні особи, які є покупцями та користувачами результату проекту, що визначають вимоги до розробленої продукції і послуг, формуючи попит на них [11].

Діяльність у сфері ІТ має такі характерні особливості:

- розроблене ПЗ орієнтовано на вирішення користувацьких завдань – споживача;
- наявність великої кількості суб'єктів діяльності;
- користувачі ПП як правило не є фахівцями у сфері ІТ;
- можливість цілодобового використання ПЗ з будь-якого пристрою;

- наявність ліцензій на ПЗ (безкоштовного, умовно безкоштовного та платного використання);
- більшість ПЗ немає прив’язки до місця використання;
- наявність сервісного обслуговування та належного інформаційного супроводу при використанні.

Вагомим елементом діяльності у сфері ІТ є інформація, яка забезпечує зв’язки, на протязі всього ЖЦ, між учасниками ІТ послуг та їх споживачами у вигляді двосторонньої комунікації на основі взаємного спілкування. Тобто інформація продукується, обробляється, зберігається і передається в межах певної інформаційної системи.

1.1.2. Аналіз інформаційного забезпечення діяльності в сфері ІТ

Важливим чинником діяльності у галузі ІТ є опрацювання та надання актуальної достовірної інформації. В даній галузі існують такі проблеми, що виникають під час роботи з інформацією [45, 70]:

- не достатньо повна інформація про розроблене ПЗ;
- неактуальність або відсутність даних про ПЗ.

Інформаційне забезпечення – це набір документів нормативної бази щодо розміщення, обсяги та форми організації інформації, яка знаходиться в межах певної інформаційної системи [25]. Інформаційне забезпечення має єдину систему класифікації та кодування інформації, схеми інформаційних потоків, уніфіковані системи документації тощо.

Основним середовищем для утворення інформаційного забезпечення у галузі ІТ виступає мережа Інтернет, що містить опис про програмні продукти з різних точок зору. На основі даної інформації, споживачі дізнаються про ПЗ та приймають рішення щодо його придбання і експлуатації.

Зі сторони учасника ПЗ проводяться такі основні види діяльності в мережі Інтернет:

- аналіз ринку ІТ-послуг (виявлення потреб користувачів, дослідження існуючих ІТ-рішень, моніторинг діяльності

конкурентів, відгуків споживачів тощо) через тематичні спільноти, веб-сайти, соціальні мережі, пошукові системи;

- проведення маркетингу (інформування про діяльність ІТ-компаній, PR, реклама тощо) через соціальні мережі, власний сайт, контекстну рекламу тощо;
- супровід та підтримка реалізованих ІТ-проектів через особистий веб-сайт, електронну пошту тощо;
- підтримка ділової комунікації із суспільством;
- отримання ділової та професійної інформації з тематичних сайтів, пошукових систем, конференцій тощо.

Зі сторони споживача ПЗ проводяться такі основні види діяльності в мережі Інтернет:

- отримання інформації, що містить опис про ПЗ з різних точок зору;
- придбання та підключення/завантаження/встановлення ПЗ;
- отримання послуг щодо супроводу, технічної підтримки, консультацій тощо.

Причини придбання ПЗ споживачами є різноманітними в залежності від цілей та потреб, які необхідно задовільнити.

За типом кінцевих споживачів ПЗ виділяють [128]:

- Business-to-Business (B2B) – для підприємств і закупівельних організацій;
- Business-to-Consumer (B2C) – для приватних осіб.

1.2. Аналіз віртуальних спільнот, які можуть бути джерелом інформаційного наповнення документації програмного забезпечення

До найпоширеніших видів веб-сайтів за змістом виділяють: блоги, сайти електронної комерції, краудфандингові платформи, корпоративні сайти, освітні та новинні ресурси, сайти громадського телебачення та соціальні медіа [79].

Майже кожна з цих платформ містить віртуальні спільноти, що надають функціонал комунікації між їх учасниками.

Віртуальна спільнота (ВС), інтернет-спільнота – соціальна група людей, які взаємодіють за допомогою сайтів та спеціалізованих сервісів у WWW [52, 53, 72, 113]. Тому основними складовими ВС є інформаційне наповнення (що основане на досвіді учасників) та аудиторія [118]. Автор [63] надає опис формальної моделі ВС наступним чином:

$$Community = < Content, Member > \quad (1)$$

де $Content$ – інформаційне наповнення;

$Member$ – множина учасників спільноти.

Виділяють такі типи ВС [69]:

- відкриті, де приєднатися до ВС може кожна особа;
- закриті, що потребують отримання дозволу щодо вступу у віртуальну спільноту.

Надалі будемо досліджувати саме відкриті ВС, що регулюються Законом України «Про інформацію» [28] – режими доступу до інформації – відкрита інформація (немає обмежень на отримання, опрацювання, зберігання та поширення інформації всіма зацікавленими особами).

ВС має наступні характеристики [69]:

- **подібність**, щодо структури дописів, профілів користувачів;
- **дискретність**. Відокремленість профілів користувачів та встановлення направлених зв’язків між ними;
- **взаємність**. Симетричний характер при взаємодії учасників ВС, що виступає передумовою для обміну інформацією.

Переваги використання ВС для формування ДПЗ можна представити з таких поглядів [69]:

1) для кінцевих споживачів:
   - отримання цільової інформації – вузькоспеціалізовані спільноти за певною тематикою;

- перегляд коментарів, розміщених іншими користувачами;
- необмежений доступ до матеріалів;
- вибір та налаштування цільової аудиторії;
- відсутність мовного бар’єру;
- доступ з будь-якої точки світу без часових обмежень;
- незначні фінансові затрати.

2) для взаємодії між учасниками (замовниками, розробниками) та користувачами ПЗ:
   - проведення аналізу та формування необхідної статистики, відстеження вподобань та відгуків користувачів;
   - комунікація на відстані;
   - взаємодія в реальному часі;
   - оперативне надання відповіді.

Тому аналіз ВС як джерела даних є необхідним. Слід зауважити, що сторінка ВС є структурованою та поділена на розділи, де користувач (власник, учасник спільноти) може розміщувати інформацію попередньо визначеного характеру. Типова структура ВС складається з таких інформаційних блоків: назва, адреса (URL посилання), множини тем, дискусій та учасників.

Автор [47] виділяє такі типи віртуальних спільнот:

- соціальні мережі;
- дискусійні листи;
- веб-спільноти (організаційні та комунікативні).

Соціальні мережі є найпопулярнішим та найбільш відвідуваним видом спільнот, адже їх структура утворюються індивідами або організаціями. Тому основним об’єктом постають люди, а не інформація. Через це дуже важко здійснювати аналіз щодо її структурованості та збереження [109].

Дискусійні листи не є актуальними для дослідження щодо ДПЗ, адже в їх основі покладено такий засіб комунікації як електронна пошта. Відповідно доступ до дискусій та повідомлень мають тільки залучені учасники.

Організаційні веб-спільноти дають змогу виконувати конкретні специфічні завдання, але при цьому, автор повідомлення є не відомим або прихованим, що унеможливлює проведення повного аналізу щодо достовірності даних, розміщених у публікації. Тому даний вид спільнот також не підходить для формування ДПЗ.

До комунікативних ВС відносять ті спільноти, основною задачею яких є організація спілкування, комунікації учасників інтернет-простору [53, 99, 104, 114]. За способом реалізації їх поділяють на такі типи: блог, форум, гостьова книга, чат.

Чат та гостьова книга не є ресурсами для формування документації, адже інформація подається не тематично та короткочасно зберігається, внаслідок чого її важко індексувати пошуковими машинами тощо.

Проаналізувавши усі типи ВС, що можуть бути корисні для формування ДПЗ обрано наступні:

- комунікаційні веб-спільноти (форум, блог);
- соціальні мережі (групи та сторінки).

Ці типи спільнот є різні з точки зору структури. Наприклад, форум має структуроване подання інформації за певними темами (частинами, розділами), в той час як блоги та соціальні мережі не мають ієрархічної структури відображення інформації, адже дискусії між учасниками виникають спонтанно. Тому доцільно розглядати три типи структур спільнот [69]:

- ті, що мають ієрархічну чітку структуру (рис.1.6);
- ті, які мають ключові слова (теги) за якими їх можна віднести до певної тематики – мережева структура (рис.1.7);
- гібридна структура, що поєднує в собі ієрархічну та мережеву структури (наприклад, форум, що мав ієрархічну структуру додали елементи мережевої структури, де дискусії мають набір ключових слів, дескрипторів що належать певній темі (рис.1.8)).

Рис.1.5 Структура спільнот: а – ієрархічна, б – мережева.

Прикладом ієрархічної структури спільноти є український форум Replace:

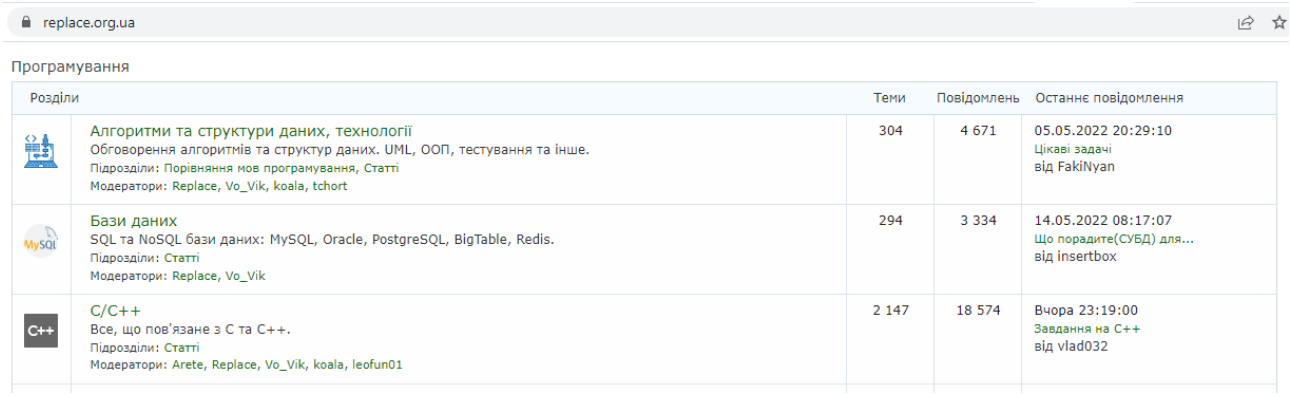


Рис.1.6 Приклад ієрархічної структури (Replace – Український форум програмістів)

Прикладом мережевої структури ВС є групи в соціальній мережі Facebook:

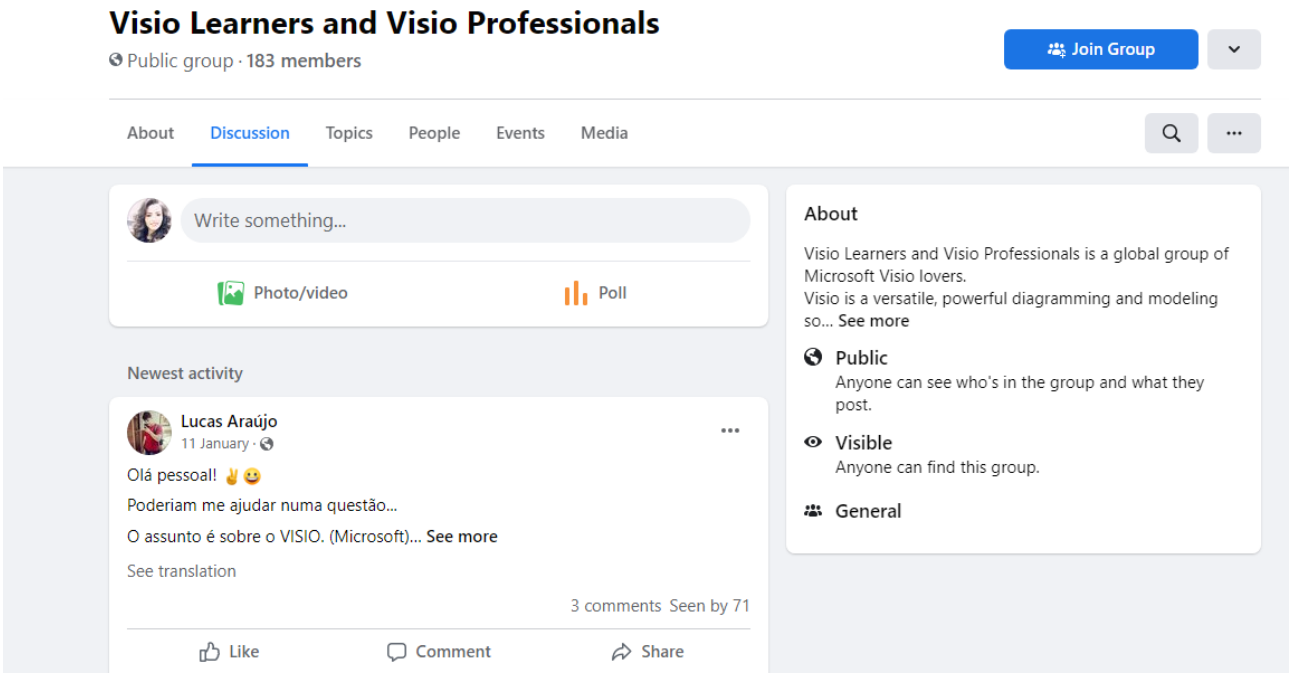


Рис.1.7 Приклад мережевої структури спільноти («Visio Learners and Visio Proffessionals»)

Прикладом гібридної структури спільноти є український вебсайт CodeProject:

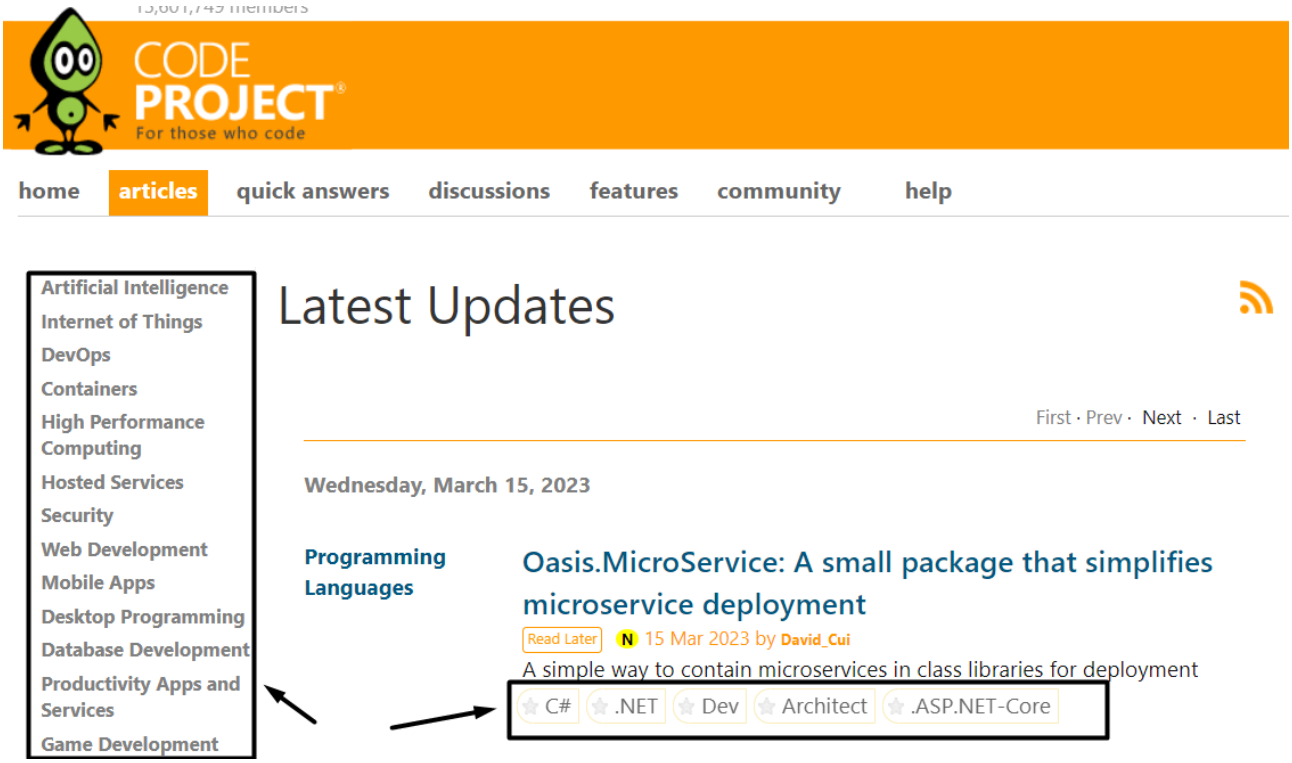


Рис.1.8 Приклад гібридної структури спільноти CodeProject

Не зважаючи на структури ВС, кожна з них має розподіл за темами, розділами і/або тегами, які містять множину дискусій. Загалом дискусії та дописи займають 90% всього наявного контенту ВС.

Дискусія, як форма колективного обговорення певного явища, містить множину повідомлень (дописів), що семантично пов’язані та хронологічно упорядковані [118].

Дискусія може бути таких типів [63]:

- пласка, що має лінійну послідовність повідомлень;
- ієрархічна, яка передбачає, що кожне повідомлення (окрім першого) є відповіддю на одне з попередніх, не обов’язково останнє [53].

Допис є атомарною одиницею та містить наступні складові [119]:

- заголовок повідомлення;
- інформацію про автора;
- дата та місце створення;
- текст;
- тематика, що відображається приналежністю до теми, розділу.

Допис може бути наступних видів:

- допис-дискусія – допис, що розпочинає дискурс (має коментарі), але не має попередніх повідомлень, та відноситься до певної теми;
- допис-коментар є відповіддю на попередній допис, що може бути додатковою інформацією (доповнювати попередній допис, бути зауваженням або твердженням).

Над дописами можна виконувати такі дії: створювати, редагувати, видаляти, поширювати, коментувати, надавати оцінку (формуючи рейтинг).

Текстова частина допису може містити наступне [119]: текст, анімацію, гіпертекст, мультимедіа, графіки.

1.2.1. Дослідження тематики віртуальної спільноти

Для формування ДПЗ необхідно відібрати саме ті ВС, що можуть містити інформацію про ПП.

Відомо, що кожна спільнота в незалежності від її структури має розподіл за темами і/або розділами, що дозволяє відобразити її у вигляді у вигляді дерева, де вершинами виступають розділи (для ієрархічної структури ВС) або тематики (для мережевої структури спільноти) [118].

Тема містить згруповані матеріали за певною тематичною ситуацією. Існують тематичні та багатотематичні ВС. Тому доцільно здійснювати пошук та збір даних не на рівні спільноти, а на рівні теми.

Автор [63] представляє множину тем спільноти таким чином:

$$Theme = \{Theme_i\}_{i=1}^{N^{(Tm)}} \quad (2)$$

де $N^{(Tm)}$ – загальна кількість тем.

Тема описується певною множиною ключових слів, кожне з яких має ваговий коефіцієнт [63]:

$$Theme_i = \{\langle Keyword_j, w_{ij}\rangle\}_{j=1}^{N^{(KwTheme_i)}} \quad (3)$$

де $Keyword_j$ – ключове слово що належить до множини ключових слів *i*-ої теми; w_{ij} – ваговий коефіцієнт для ключового слова $Keyword_j$, що знаходиться в $Theme_i$;

$KwTheme_i$ – кількість $Keyword_j$ в *i*-тій темі.

Кожне повідомлення містить набір ключових слів, які автор призначає [63]: набір ключових слів, що знаходиться в $Text_i$, до певного повідомлення визначатимемо як $KeywordList(Post_i)$, де $KeywordList(Post_i) \subseteq Keyword$.

1.3. Аналіз сучасних технологій виявлення корисного інформаційного наповнення з віртуальних спільнот

Дослідження ВС є багаторічним досвідом серед різних наук (в тому числі науки про комунікацію та суспільство, про математичний аналіз і теорію графів,

про маркетинг, штучний інтелект та програмне забезпечення). Ці галузі відображають поведінку різних верств населення в певний проміжок часу в мережі Інтернет. Поєднання досліджень ВС з аналітичними методами, формують основи методів та засобів соціального аналізу в загалом.

На сьогодні, виділяють такі напрями аналізу ВС [53, 122-124]:

- інформаційна безпека;
- аналіз життєвого циклу ВС;
- пошук взаємозв’язків між користувачами, формування груп;
- виявлення користувацького досвіду;
- дослідження текстово-графічних даних;
- аналіз профілів користувачів;
- Big Data;
- вплив особистості на інформаційне суспільство.

Для виявлення та збору даних застосовують наступні методи:

- ручний;
- автоматизований;
- комбінований (поєднання ручного та автоматизованого).

Ручний збір і аналіз інформації є універсальним та індивідуальним, але є доволі повільним на відміну від інших методів.

Слід зауважити, що ВС є динамічними з точки зору наповнення даними, адже їх інформаційне наповнення залежить від дій учасників. Дану властивість потрібно враховувати при автоматизованому розборі даних, адже код сторінки може бути неповним.

Існують такі техніки аналізу інформації:

- моделювання на основі графів;
- методи автоматизованого збору даних з веб-сторінок (краулінг, парсинг);
- зберігання та управління даними (Big Data);
- інструменти оцінювання та класифікації (скоринг).

Документацію можна представити у вигляді **графу**, що має взаємопов'язані вузли, вершини якого є розділи, а ребра представляють зв'язки між ними [90]. Цей підхід надає словник термінів для подальшого математичного опису ДПЗ, інструментів та методів для проведення кількісного і якісного дослідження, а також для визначення і доведення теорем.

Для автоматизованого виявлення та збору інформаційного наповнення в мережі Інтернет застосовують підхід **краулінг**. Даний спосіб буде корисним для збору наявних дописів у ВС пошуковим агентом, який складається з множини комп'ютерів. Такий агент набагато швидше може знаходити та обирати дописи з різних ВС за заданим пошуковим запитом ніж користувач з використанням веб-браузера.

Метод автоматизованого збору та структурування даних (з англ. **parsing**) дозволяє збирати, систематизувати, оновляти інформацію. До переваг застосування цього методу належать [108, 119, 131]:

- економія ресурсів завдяки швидкому автоматизованому збору даних в будь-якому режимі;
- пошук релевантної інформації;
- автономність – проведення регулярної перевірки за заданим інтервалом часу;
- представлення даних в будь-якому форматі з можливістю конвертації;
- точність збору даних;
- забезпечення рівномірного навантаження на веб-сайт, з якого відбувається збір даних (щоб не створювати ефекту DDOS-атаки).

Програму автоматизованого збору даних можна створити вручну або скористатися існуючими платними та безкоштовними ресурсами (наприклад Netpeak Checker, Import.io, Mozenda тощо).

Важливою рисою застосування процедури автоматизованого збору даних є правильність її налаштування користувачем, який впливає на результат.

Основними етапами збору та структурування даних дописів ВС є наступні [119]:

- постановка задачі (аналіз потрібних атрибутів HTML сторінки ВС);
- збір релевантної інформації з вказаної спільноти;
- сортування отриманих даних;
- представлення інформації у певному структурованому форматі – формування звіту. Зазвичай дені зберігають у БД, текстових файлах тощо.

Даний метод розбору потрібно виконувати для кожної ВС окремо, так як вони мають різні структури.

При розробці програми можна працювати як з кодом сторінки так і з використанням прикладного програмного інтерфейсу (API), що, зазвичай, надають власники платформ на яких розміщені ВС. API, як метод абстрагування між двома різнорівневими ПЗ, надає розробникам повний набір команд для швидкої розробки програм.

Застосування готових API спрощує процедуру автоматизованого збору даних з ВС, адже розробник не має власноруч розбирати HTML код сторінки та витягувати необхідну інформацію, але не всі платформи надають відкриті API для загального користування. Наприклад, платформа Facebook, не дає повну інформацію про користувацькі дані. І, відповідно, отримати дані в повному обсязі з ВС що є на цьому майданчику не є можливим.

Завдяки збільшенню інформаційного наповнення у WWW важливою сферою для досліджень постає робота з **Big Data**. Вивченням великих обсягів даних і проведення їх аналітики, порівняння методів аналізу даних соціальних мереж висвітлено у роботі [112].

Перевагами застосування технік роботи з **Big Data** у ВС є:

- аналіз, обробка, зберігання великих даних в реальному часі по мірі надходження;

- роботі з різнорідними даними (як структурованими так і неструктурованими);
- пошук взаємозв’язків серед всього набору даних.

Однією з найпоширеніших математичних моделей для надання оцінки у вигляді зваженої суми набору характеристик для певної системи є **скоринг** (з англ. scoring) [42, 56]. За допомогою даного методу на основі існуючих даних можна спрогнозувати певні події. В даній роботі цей спосіб є актуальним для обчислення рейтингу користувачів ВС, що розміщують дописи, також для визначення зміни динаміки публікування матеріалів.

1.3.1. Застосування контент-аналізу для дослідження даних у ВС

Вивчення блоків інформації, що містяться на сторінці ВС потребує застосування якісно-кількісний методу – контент-аналізу, головною ідеєю якого є квантифікаційна обробка тексту для представлення об’єктивного, систематичного та кількісного опису з подальшим трактуванням результатів [39]. Даний спосіб характеризується строгістю проведення процедури, об’єктивністю висновків.

Контент-аналіз можна застосувати:

- основний метод дослідження;
- в поєднанні з іншими способами;
- як допоміжний (для прикладу, при проведенні класифікації).

Контент-аналіз містить такі основні складові [39]:

- зміст (все те в тексті, що має значення для учасників комунікації);
- текст (група висловлювань, що утворюють смислову єдність. Може бути у вигляді тексту, зображень, аудіо-, відеозапису тощо);
- контекст (все, що може вплинути на модифікацію сприйняття змісту тексту – час його створення, автор тощо).

Епістемологічною базою контент-аналізу виступають наступні дослідження (рис.1.9):

- «реальність» тексту (співвідношення між символічними об’єктами в межах документу, масивів документів);
- «реальність» контексту («об’єктивна» реальність, яка відтворена в тексті).

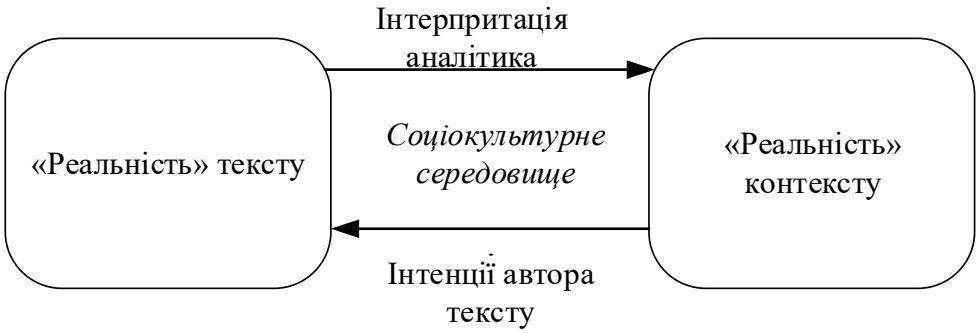


Рис.1.9 Взаємодія між «реальністю» тексту та «реальністю» контексту

В першому випадку досліджується сам текст без соціального контексту (наприклад, обчислюються частота вживання слів, категорії аналізу, які попередньо узгоджені тощо. Внаслідок чого формуються висновки щодо цього масиву документів).

В другому випадку, за допомогою аналізу документів, вивчається соціальна реальність та надаються висновки щодо контексту існування цих документів. Сучасні автоматизовані методи аналізу текстів під контекстом мають на увазі метадані (ті дані, що не містяться безпосередньо в тексті, але представляють структуровану інформацію про нього – зміст даних, їх статус, походження, місцезнаходження, обсяг, форма представлення, автор тощо). Цю інформацію описують метадані, які застосовують у слабоструктурованих системах, при створенні архівів документів. Також вони впливають на ефективність при пошуку інформації.

При проведенні контекст-аналізу виокремлюють такі методи: кількісний і якісний. До найпопулярніших якісних методів дослідження текстів належать традиційний та дискурс аналіз. Серед кількісних методів – текст-майнинг, аналіз природньої мови тощо.

Для опису різнорідних ВС доцільно застосувати стандарт Dublin Core Metadata Initiative (скорочено Dublin Core – Дублінське ядро), яке уніфіковує

метадані [98]. Дублінське ядро містить словник основних понять, що поділений на два рівні – простий (некваліфікований) та компетентний (кваліфікований).

Національна організація з інформаційних стандартів (NISO) пропонує класифікувати метадані таким чином: описові, структурні, адміністративні [110].

Описові метадані відображають вміст документів та файлів для їх подальшого збирання або групування. Наприклад, описові метадані ВС включатимуть такі відомості, як ім’я автора та його місце розташування під час публікування матеріалів, заголовку і тексту допису, ім’я власника спільноти, назви тем, подій тощо.

Структурні метадані відображають інформацію про те, яким чином зберігаються дані, а також відображають склад та організацію ресурсів. Наприклад, цифрову книгу можна завантажити у вигляді зображень окремих сторінок, PDF або HTML. Структурні метадані допису можуть містити наступну інформацію: дата і час публікації, кількість реакцій користувачів на дописи, дата та час реєстрації (тривалість перебування) користувача у спільноті, дата і час певної події (розділ «Події»), розмір та тип файлу (якщо є в дописі) тощо.

Адміністративні метадані використовуються для керування ресурсом або контентом. Приклад: ліцензійні обмеження на розповсюдження інформації, права доступу та їх походження до певних умов використання, а також правила зберігання і видалення даних. Такі метадані є корисними для адміністраторів БД або трафіку, а також для журналів систем безпеки та даних про події.

В таблиці 1.1. представлені співвідношення елементів метаданих Дублінського ядра стосовно ДПЗ та ВС.

Таблиця 1.1

Елементи метаданих Дублінського ядра для ДПЗ та ВС

Аспект опису	**Метадані**	**Зміст**
Вміст ресурсу	Title – Назва	Назва ВС, допису, терміну ДПЗ
	Subject – Предмет	Теми ВС, розділи ДПЗ
	Description – Опис	Розділ «Опис» у ВС, опис термінів ДПЗ
	Type – Тип	Тип допису ВС
	Source – Джерело	Інформація про ВС, допис якого завантажено у СКД
	Relation – відношення	Зв’язки між термінами ДПЗ, відношення між користувачами ВС у вигляді профілів користувачів
	Audience – аудиторія	Учасники ВС, користувачі ДПЗ
Інтелектуальна власність	Creator – Автор	Автор допису ВС або статті ДПЗ
	Publisher – Публікатор	Власник ВС
	Contributor – Співавтор	Експерт (модератор), який редагує дописи/дискусії
	Rights – права	Дані про авторські права
Стан	Data – Дата	Дата створення/редагування допису
	Format – Формат	Формат ВС
	Identifier – Ідентифікатор	URL адреса сторінки звідки взято статтю ДПЗ
	Language – Мова	Мова дописів ВС, статей ДПЗ

1.4. Вибір методу оцінювання якості документації програмного забезпечення

Формування документації програмного забезпечення відповідно до міжнародних стандартів потребує постійного спостереження та проведення аналізу щодо її відповідності. Також постає необхідним формування сучасних та суспільно орієнтованих вимог до її якості, щоб документація була зручною, доступною та інформативною для широкого кола користувачів [45]. Через це задача побудови якісної ДПЗ є нагальною та актуальною для представників ІТ бізнесу, оскільки її вирішення впливає не тільки на покращення якості створюваного продукту [46], а також на послуги, які користувачі отримують внаслідок користування ПЗ.

Якість ДПЗ як ІП відображає міру відповідності властивостей і характеристик щодо очікувань, вимог та потреб її споживачів.

На сьогодні існують багато моделей якості програмного забезпечення, структура яких описується ієрархією, елементами якої є множина характеристик [45], під-характеристик та відношень підпорядкованості між ними [46].

Професійним підходом до визначення якості є моделі [46]:

- модель МакКолла (1977), в якій характеристики якості поділені на три групи — фактори (задаються вимогами), критерії (визначають як цілі) та метрики (використовують для кількісного опису та вимірювання якості) [103];
- модель Боема (1978) [84], яка є розширенням моделі МакКолла [103] (визначено дев’ятнадцять проміжних атрибутів, які доповнюють усі одинадцять факторів якості за моделлю МакКолла);
- модель Гезі (1991), що описує вісім характеристик якості [87];
- міжнародний стандарт IEEE [91];
- модель Дромера (1994, 1998) [85];
- QMOOD (2002) — містить шість характеристик якості [83];

- міжнародний стандарт ISO/IEC-25010 [94], що є продовженням стандарту ISO/IEC-9126 [93]

Згідно з дослідженнями, що подані у статті [57], моделі якості ПЗ поділяються на дві групи [46]:

- основоположні або базові моделі якості ПЗ, що виникли в результаті роботи авторських колективів міжнародних авторитетних організацій, наприклад, ISO (9126-1, 25010) та IEEE;
- корпоративні моделі якості ПЗ, які суттєво поступаються базовим. До них належать: модель МакКолла, Боема, Дромера, Гезі, QMOOD.

Автор [45] виділяє наступні аспекти аналізу якості документації: інформаційного наповнення, оформлення [19] та подання даних. В цій роботі досліджуватимемо саме якість інформаційного наповнення ДПЗ.

Наведені вище моделі якості та міжнародний стандарт можуть бути застосовані для ПЗ як до програмного продукту. Документація ПЗ є ІП, оскільки ІП — це задокументована інформація, підготовлена відповідно до вимог для її поширення [46]. Але необхідно звернути увагу на те, що не всі характеристики та під-характеристики, які містять ці моделі якості придатні саме до ДПЗ, що орієнтована на задоволення потреб суспільства. Тому виникає потреба у дослідженні та відборі характеристик та під-характеристик якості щодо формування ДПЗ як ІП.

В Україні для оцінювання якості ПЗ представлені такі стандарти [15, 16, 22]:

- ДСТУ 2844-94;
- ДСТУ 2850-94;
- ДСТУ ISO/IEC 12119:2003.

Але ці державні стандарти не містять весь набір показників, що можна застосувати для оцінювання якості ДПЗ. Натомість, міжнародний стандарт ISO/IEC-25010 [94] дозволяє оцінити якість ДПЗ за більшим набором показників,

аніж вище наведені національні стандарти, що містять опис показників якості, які еквівалентні характеристикам міжнародного стандарту [61].

Характеристики якості стандарту ISO/IEC-25010 наведені рис.1.10

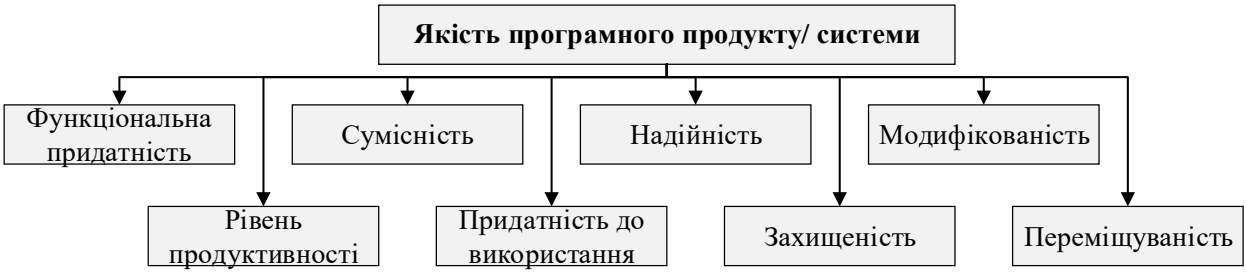


Рис.1.10 Характеристики якості стандарту ISO/IEC-25010

Показники, характеристики, наведені стандартом ISO/IEC-25010 оцінювання можна проводити за такими поглядами:

- якість для інформаційних продуктів;
- якість для програмних продуктів.

Через те, що ДПЗ є ІП, тому характеристики якості, з точки зору програмного продукту не будуть розглянуті.

Стосовно якості ДПЗ повинна відповідати таким характеристикам:

- містити актуальні та достовірні дані;
- забезпечувати точність та повноту – не містити прогалин щодо інформаційного наповнення документації;
- орієнтованість на потреби споживача;
- бути зрозумілою широкому колу споживачів та зручною у використанні;
- доступною та адаптивною до особливостей середовищ використання.

В роботі [45] наведено опис показників якості стандарту ISO/IEC-25010, дослідивши які для оцінювання ДПЗ обрано наступні характеристики та під-характеристики [66, 94]:

1) Функціональна придатність (Functional suitability), яка визначається під-характеристикою:

- відповідність (completeness) – ступінь, відповідності набору функцій ПЗ до вимог та цілей користувача.

2) Придатність до використання (usability), має такі під-характеристики:
   - відповідність розпізнавання (appropriateness recognisability) – міра можливостей користувача розпізнати чи продукт, описаний в ДПЗ, підходить до його цілей та потреб.
3) Супроводжуваність (maintainability):
   - модульність (modularity) – ступінь, що відображає на скільки ДПЗ є скомпонована з окремих, достатньо незалежних компонент.

Оцінка кожної під-характеристики визначається інтервалом [0, 1].

Висновки до розділу

В першому розділі здійснено огляд літературних джерел відповідно до теми обраного дисертаційного дослідження, що дозволило надати визначення терміну документації програмного забезпечення, як одному з видів документації, а також описати її функції, призначення, сфери застосування, користувачів та їх види діяльності.

Здійснено аналіз тенденцій розвитку галузі інформаційних технологій в Україні та основні етапи розробки, які повинні супроводжуватися належною документацією програмного забезпечення, що і обґрунтувало актуальність даної роботи та дозволило визначити прогалини в інформаційному забезпеченні.

Встановлено, що віртуальні спільноти можуть бути цінним, корисним джерелом даних для формування інформаційного наповнення документації, яке створюється суспільством, щодня оновлюється та доповнюється, тому проведено аналіз їх структур, основних складових, наведено характеристики та переваги використання. Проаналізовано особливості використання інформаційного наповнення віртуальних спільнот для формування документації програмного забезпечення з урахуванням сучасних тенденцій та перспектив розвитку інженерії програмного забезпечення.

Орієнтованість документації програмного забезпечення на потреби споживача дозволило представляти її як інформаційний продукт, який відповідає певним вимогам, що описані в міжнародних та державних стандартах. Для проведення аналізу обрано стандарт ISO/IEC-25010, що забезпечив визначення набору характеристика та під-характеристик для оцінювання якості сформованої документації програмного забезпечення.

Розділ 2. Побудова формальної моделі документації програмного забезпечення та методів її наповнення

Основною властивістю формування ДПЗ є необхідність представлення користувачеві структурованої, компактної та достовірної інформації, що містить опис про ПЗ з різних точок зору.

Так як джерелами інформаційного наповнення документації є різнорідні ВС, тому необхідно навести опис наступних моделей:

1. Модель віртуальної спільноти.
2. Модель сховища консолідованих даних (СКД), що побудована на основі понятійного апарату спільнот бо вони мають подібні, узагальнені структури.
3. Модель ДПЗ у вигляді кінцевого інформаційного продукту.

Через чисельні дослідження інших науковців [63, 72] частину формальної моделі ВС наведено в розділі 1.2 «Аналіз віртуальних спільнот, які можуть бути джерелом інформаційного наповнення документації програмного забезпечення».

Для опису моделі СКД проаналізовано структуру джерела даних – віртуальних спільнот, визначено набір метаданих, які потрібні для подальшого формування ДПЗ. Виокремлено поняття теми як цілісного логічного джерела, що містить інформацію з певної тематичної ситуації. Висвітлено основні складові ДПЗ, кожна з яких виступає теоретичним підґрунтям для побудови методів та засобів її формування.

Основні результати розділу опубліковані автором у працях [68, 70].

2.1. Побудова формальної моделі сховища консолідованих даних

Для збереження інформації з певної предметної області застосовують сховища даних, які в подальшому використовуються для підтримки прийняття рішень. Джерела інформації (віртуальні спільноти) зазвичай є однотипними, але при цьому можуть мати різну структуру. Тому для роботи з даними потрібно їх

привести єдиного вигляду та, надалі, завантажити до сховища даних. Після цього отриману інформацію можна опрацьовувати для формування ДПЗ як готового ІП. Отже сховище даних є важливою складовою для генерування ДПЗ джерелами якої виступають ВС [67].

Процес збору, поєднання та упорядкування інформації з ВС виконують за допомогою методів інтеграції даних: тиражування, федералізації, консолідації [24, 43, 76, 82, 126]. Ці методи допомагають отримати цілісне уявлення щодо обраної предметної області [45].

Для роботи з однорідними джерелами доцільно застосовувати тиражування даних. Але, зазвичай, ВС мають різну структуру даних тому даний метод не є актуальним для формування ДПЗ за допомогою ВС.

Метод федералізації даних можна застосовувати для роботи з неоднорідними спільнотами. Але при роботі з великою кількістю джерел можуть виникнути істотні проблеми з продуктивністю опрацювання матеріалів.

Консолідація даних полягає у формуванні єдиного сховища документації з різнорідних спільнот, що дозволить скласти адекватну інформаційну модель проблемної області з метою її аналізу, опрацювання та подальшого ефективного використання в процесах підтримки прийняття рішень [40, 45, 67, 82]. Консолідація даних – зручний спосіб об’єднання даних із різних джерел в одному звіті.

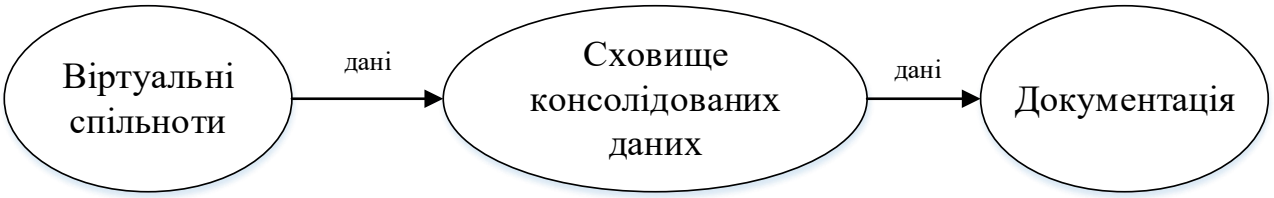


Рис.2.1 Загальна схема потоку даних для формування ДПЗ

Проаналізувавши методи збору даних з різнорідних джерел для формування ДПЗ обрано метод консолідації даних. На основі цього введено поняття – СКД, що містить набір структурованих, представлених в одному вигляді даних завантажених з ВС, за допомогою методу консолідації.

Процес консолідації даних відбувається під час етапу обробки даних [82].

Одним із складових етапу консолідації є процес ETL (Extract, Transform, Load – з англ. Витяг, Перетворення та Завантаження), що зображений на рис.2.2 Процес консолідації даних полягає у наступному [115]:

1. Отримання, витяг даних з різнорідних джерел.
2. Перетворення даних та збереження їх у певній структурі або форматі, з метою подальшого аналізу (інтеграція).
3. Завантаження перетворених, очищених даних у кінцеву базу даних (БД) – СКД.

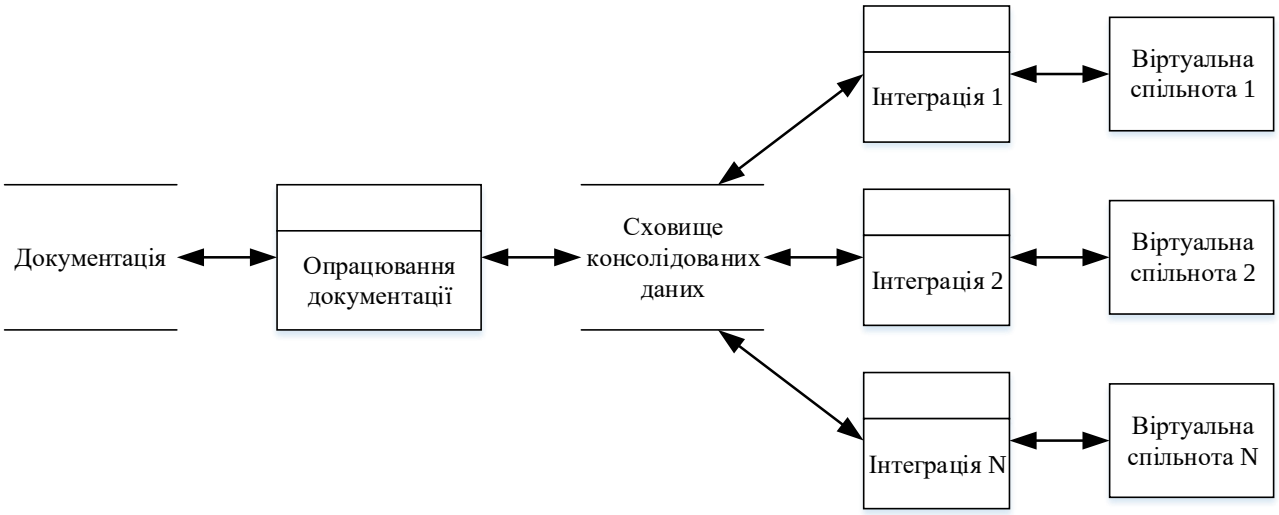


Рис.2.2 Схема формування ДПЗ за допомогою методу консолідації даних

СКД отримує дані за допомогою їх експорту з різних ВС, тому формальна модель СКД базується на основі атрибутів ВС, що необхідні для подальшої обробки та формування ДПЗ.

Формальна модель СКД відобразимо кортежем наступного вигляду:

$$DocDataWarehouse = \langle SourceVC, Theme, Post, User \rangle, \quad (2.1)$$

де $SourceVC = \{SourceVC_i\}_{i=1}^{N^{(SourceVC)}}$ – множина (список) ВС;

$N^{(VC)}$ – загальна кількість спільнот.

$Theme = \{Theme_i\}_{i=1}^{N^{(Theme)}}$ – множина завантажених джерел (тем) інформації;

$N^{(Theme)}$ – загальна кількість тем;

$Post = \{Post_i\}_{i=1}^{N^{(Post)}}$ – множина дописів;

$N^{(Post)}$ – кількість дописів;

$User = \{User_i\}_{i=1}^{N^{(User)}}$ – множина користувачів;

$N^{(User)}$ – кількість завантажених користувачів.

Деталізуємо модель СКД. Дані СКД можуть бути наступних типів:

- статичні, сталі – значення яких не змінюється протягом певного тривалого часу;
- динамічні – значення яких може змінюватися, оновлятися в результаті звернень системи або роботи агенту з ними.

Тому зазначені масиви даних у формальній моделі СКД визначається двома складовими:

- метадані (атрибути дублінського ядра опис яких наведено в розділі 1.3.1) – статичні дані;
- історія – динамічні дані:
  - додатковий змістовний контент про відповідні поняття;
  - атрибути роботи агентів. Для реалізації роботи системи було прийнято рішення залучити агентів, що дозволить провести декомпозицію шляхом поділу одного завдання рішенням набору менших завдань, що є простішими. Основними атрибутами агентів є (статус та час звернення до системи).

Кожна ВС, дані якої зберігаються в СКД, має наступні параметри:

$$SourceVC_i = \langle DC(SourceVC_i), History(SourceVC_i) \rangle, \quad (2.2)$$

де $DC(SourceVC_i)$ – метадані про *i*-ту ВС;

$History(SourceVC_i)$ – історія змін при роботі з *i*-тою ВС.

Атрибути дублінського ядра для *i*-ої ВС, що необхідні для СКД, відобразимо у такому вигляді:

$$DC(SourceVC_i) = \left\langle \begin{matrix} Title(SourceVC_i), URL(SourceVC_i), \\ Date(SourceVC_i), Format(SourceVC_i), \\ Description(SourceVC_i) \end{matrix} \right\rangle, \quad (2.3)$$

де $Title(SourceVC_i)$ – назва *i*-тої спільноти;

$URL(SourceVC_i)$ – ідентифікатор (посилання) *i*-тої спільноти;

$Date(SourceVC_i)$ – дата створення *i*-тої спільноти;

$Format(SourceVC_i)$ – формат (шаблон завантаження) *i*-тої спільноти;

$Description(SourceVC_i)$ – описові дані *i*-тої спільноти;

Історія змін при роботі з даними *i*-тої ВС описується такими параметрами:

$$History(SourceVC_i) = \langle AddInfo(SourceVC_i), Agent(SourceVC_i)\rangle, \quad (2.4)$$

де $AddInfo(SourceVC_i)$ – додаткова інформація про *i*-ту ВС;

$Agent(SourceVC_i)$ – інформація щодо роботи агенту з *i*-тою ВС.

Допоміжні атрибути для *i*-ої ВС характеризуються таким кортежем:

$$AddInfo(SourceVC_i) = \left\langle \begin{matrix} CountOfTheme(SourceVC_i), \\ Rate(SourceVC_i), Status(SourceVC_i) \end{matrix} \right\rangle, \quad (2.5)$$

де $CountOfTheme(SourceVC_i)$ – кількість тем що є завантажено до СКД з *i*-тої спільноти;

$Rate(SourceVC_i)$ – інформація про рейтинг *i*-тої спільноти;

$Status(SourceVC_i)$ – статус *i*-тої спільноти;

$AddInfoTime(SourceVC_i)$ – часовий атрибут для роботи із завантаженою інформацією з *i*-тої ВС.

Робота агенту для *i*-ої ВС містить такі атрибути:

$$Agent(SourceVC_i) = \left\langle \begin{matrix} ActivityTitle(SourceVC_i), \\ AgentDate(SourceVC_i), Estimate(SourceVC_i), \\ StatusAgent(SourceVC_i) \end{matrix} \right\rangle, \quad (2.6)$$

де $ActivityTitle(SourceVC_i)$ – назва роботи, дії агенту що працює з *i*-тою спільнотою (бо один агент може здійснювати різні дії);

$AgentDate(SourceVC_i)$ – дата роботи агенту з *i*-тою спільнотою;

$Estimate(SourceVC_i)$ – оцінка роботи агенту з *i*-тою спільнотою;

$StatusAgent(SourceVC_i)$ – статус роботи агенту з *i*-тою спільнотою.

$$ActivityTitle = \langle AgentTitle, AgentActivity\rangle, \quad (2.7)$$

де $AgentTitle$ – назва агенту;

$AgentActivity$ – дія агенту.

Тема, як джерело інформації, описується наступними складовими:

$$Theme_i = \langle DC(Theme_i), History(Theme_i) \rangle, \quad (2.8)$$

де $DC(Theme_i)$ – метадані про *i*-ту тему;

$History(Theme_i)$ – історія змін при роботі з *i*-тою темою.

Надалі деталізуємо кожен набір даних для *i*-ої теми.

Атрибути дублінського ядра для *i*-ого джерела (теми) представимо у такому вигляді:

$$DC(Theme_i) = \left\langle \begin{matrix} Title(Theme_i), URL(Theme_i), Keyword(Theme_i), \\ Description(Theme_i), DateCreate(Theme_i) \end{matrix} \right\rangle, \quad (2.9)$$

де $Title(Theme_i)$ – назва *i*-того джерела (теми) для ієрархічної структури; назва тегу для мережевої структури; об'єднання назви теми і тегу для гібридної структури спільноти.

$URL(Theme_i)$ – ідентифікатор (посилання) *i*-того джерела;

$Keyword(Theme_i) = \{\langle Keyword_j, w_{ij} \rangle\}_{j=1}^{N^{(Keyword(Theme_i))}}$ – множина ключових слів що визначають *i*-те джерело;

$N^{(Keyword(Theme_i))}$ – загальна кількість ключових слів в *i*-тому джерелі (темі).

$Description(Theme_i)$ – описові дані *i*-тої теми.

$DateCreate(Theme_i)$ – дата створення *i*-того джерела (теми).

Історія змін при роботі з даними *i*-тої теми описується таким кортежем:

$$History(Theme_i) = \langle AddInfo(Theme_i), Agent(Theme_i) \rangle, \quad (2.10)$$

де $AddInfo(Theme_i)$ – додаткова інформація про *i*-ту тему;

$Agent(Theme_i)$ – інформація щодо роботи агенту з *i*-тою тему.

Допоміжні атрибути для *i*-ого джерела (теми) містить наступні характеристики:

$$AddInfo(Theme_i) = \left\langle \begin{matrix} CountOfPostDis(Theme_i), Rate(Theme_i), \\ Status(Theme_i), HistoryChange(Theme_i) \end{matrix} \right\rangle, \quad (2.11)$$

де $CountOfPostDis(Theme_i)$ – кількість дописів-дискусій, що є завантажено до СКД з *i*-тої теми;

$Rate(Theme_i)$ – інформація про рейтинг *i*-тої теми;

$Status(Theme_i)$ – статус *i*-тої теми,

$HistoryChange(Theme_i)$ – історія змін i-тої теми після роботи експерта (наприклад щодо зміни вагових коефіцієнтів ключових слів).

Робота агенту для i-ого джерела (теми) характеризується такими атрибутами:

$$Agent(Theme_i) = \left\langle \begin{matrix} ActivityTitle(Theme_i), DateQuery(Theme_i), \\ Estimate(Theme_i), StatusAgent(Theme_i) \end{matrix} \right\rangle, \quad (2.12)$$

де $ActivityTitle(Theme_i)$ – назва роботи, дії агенту що працює з i-ою темою;

$DateQuery(Theme_i)$ – дата запиту роботи агенту з i-тою темою;

$Estimate(Theme_i)$ – оцінка роботи агенту з i-тою темою;

$StatusAgent(Theme_i)$ – статус роботи агенту з i-тою темою.

Допис – неподільна інформаційна одиниця, що визначається кортежем наступного виду:

$$Post_i = \langle DC(Post_i), AdditionalInf(Post_i), Agent(Post_i) \rangle, \quad (2.13)$$

де $DC(Post_i)$ – метадані про i-тий допис;

$AdditionalInf(Post_i)$ – додаткова інформація про i-тий допис;

$Agent(Post_i)$ – інформація роботи агента з i-тим дописом.

Надалі деталізуємо кожен набір даних для i-ого допису.

Атрибути дублінського ядра для i-ого допису подамо у такому вигляді:

$$DC(Post_i) = \left\langle \begin{matrix} Title(Post_i), URL(Post_i), Text(Post_i), \\ Keyword(Post_i), Multimedia(Post_i), \\ Files(Post_i), Hypertext(Post_i), Type(Post_i), \\ LevelDiscuss(Post_i), Author(Post_i), \\ Language(Post_i), DateCreate(Post_i) \end{matrix} \right\rangle, \quad (2.14)$$

де $Title(Post_i)$ – назва i-го допису;

$URL(Post_i)$ – ідентифікатор (посилання) i-тий допис;

$Text(Post_i)$ – текстові дані i-го допису;

$Keyword(Post_i) = \{\langle Keyword_j, w_{ij} \rangle\}_{j=1}^{N^{(Keyword(Post_i))}}$ – множина ключових слів що належать i-тому допису;

$N^{(Keyword(Post_i))}$ – кількість ключових слів що є в дописі;

$N^{(Keyword(Post_i))} \in Keyword(Theme)$ – всі ключові слова $Post_i$ належать темі;

$Multimedia(Post_i)$ – мультимедійні дані *i*-го допису;

$Files(Post_i)$ – файлові дані *i*-го допису;

$Hypertext(Post_i)$ – гіпертекстові дані *i*-го допису;

$Type(Post_i)$ – тип *i*-го допису (допис-дискусія або допис-коментар);

$Author(Post_i)$ – автор *i*-го допису;

$Language(Post_i)$ – мова *i*-го допису;

$DateCreate(Post_i)$ – дата створення *i*-того допису.

Допоміжний масив даних для *i*-того допису містить наступні характеристики:

$$AdditionalInf(Post_i) = \left\langle \begin{matrix} LevelDiscuss(Post_i), Rate(Post_i), \\ Status(Post_i), HistoryChange(Post_i) \end{matrix} \right\rangle, \quad (2.15)$$

де $LevelDiscuss(Post_i)$ – рівень розташування *i*-того допису в дискусії;

$Rate(Post_i)$ – інформація про рейтинг *i*-того допису;

$Status(Post_i)$ – статус *i*-того допису (активний, скритий тощо);

$HistoryChange(Post_i)$ – історія змін *i*-того допису (наприклад, при повторному завантаженні допис може бути редагований, тому необхідно зберігати всю історію існування та редагування допису).

Робота агенту для *i*-ого допису характеризується такими атрибутами:

$$Agent(Post_i) = \left\langle \begin{matrix} AgentTitle(Post_i), DateQuery(Post_i), \\ Estimate(Post_i), StatusAgent(Post_i) \end{matrix} \right\rangle, \quad (2.16)$$

де $AgentTitle(Post_i)$ – назва агенту що працює з *i*-тим дописом;

$DateQuery(Post_i)$ – дата запиту роботи агенту з *i*-тим дописом;

$Estimate(Post_i)$ – оцінка роботи агенту з *i*-тим дописом;

$StatusAgent(Post_i)$ – статус роботи агенту з *i*-тим дописом.

Користувач – особа, яка є учасником ВС, що визначається кортежем наступного виду [114]:

$$User_i = \langle DC(User_i), AdditionalInf(User_i), Agent(User_i) \rangle, \quad (2.17)$$

де $DC(User_i)$ – метадані про *i*-го користувача;

$AdditionalInf(User_i)$ – додаткова інформація про *i*-го користувача;

$Agent(User_i)$ – інформація роботи агента з *i*-тим користувачем.

Надалі деталізуємо кожен набір даних для *i*-го користувача.

Атрибути дублінського ядра для *i*-го користувача подамо у такому вигляді:

$$DC(User_i) = \left\langle \begin{matrix} URL(User_i), PersonalData(User_i), \\ Email(User_i), DateCreateAccount(User_i) \end{matrix} \right\rangle, \quad (2.18)$$

де $URL(User_i)$ – ідентифікатор (посилання) на *i*-го користувача;

$PersonalData(User_i)$ – особисті дані *i*-го користувача;

$Email(User_i)$ – електронна пошта *i*-го користувача;

$DateCreateAccount(User_i)$ – дата створення *i*-тим користувачем профілю у ВС.

Допоміжний масив даних для *i*-го користувача містить наступні характеристики:

$$AdditionalInf(User_i) = \left\langle \begin{matrix} CountOfPost(User_i), \\ Rate(User_i), Status(User_i) \end{matrix} \right\rangle, \quad (2.19)$$

де $CountOfPost(User_i)$ – кількість дописів створених *i*-тим користувачем;

$Rate(User_i)$ – інформація про рейтинг *i*-того користувача;

$Status(User_i)$ – статус *i*-того користувача (активний, неактивний тощо).

Робота агенту для *i*-го користувача характеризується такими атрибутами:

$$Agent(User_i) = \left\langle \begin{matrix} AgentTitle(User_i), DateQuery(User_i), \\ Estimate(User_i), StatusAgent(User_i) \end{matrix} \right\rangle, \quad (2.20)$$

де $AgentTitle(User_i)$ – назва агенту що працює з *i*-тим користувачем;

$DateQuery(User_i)$ – дата запиту роботи агенту з *i*-тим користувачем;

$Estimate(User_i)$ – оцінка роботи агенту щодо *i*-го користувача;

$StatusAgent(Post_i)$ – статус роботи агенту щодо *i*-го користувача.

Кожне поле, що містить особисті дані користувача [113, 114], відобразимо у вигляді кортежу:

$$PersonalData_{j_m} = \langle Characteristic_{j_m}, Datatype_{j_m}, Value_{j_m} \rangle, \quad (2.21)$$

де $Characteristic_{j_m}$ – характеристика *j*-го автора *m*-го профілю;

$Datatype_{j_m}$ – тип даних характеристики *j*-го автора *m*-го профілю;

$Value_{j_m}$ – значення характеристики *j*-го автора *m*-го профілю.

2.2. Побудова формальної моделі документації програмного забезпечення

ДПЗ має інформаційне наповнення (що представлене у вигляді статей впорядкованих за розділами), набір класифікаторів (факти та події щодо ПЗ) і взаємозв'язки між ними. Джерелом інформаційного наповнення виступає СКД. Кожна стаття містить змістовну частину, а також інформацію про авторів та джерело (звідки вона була отримана).

Тому документацію можна подати кортежем у такому вигляді [70]:

$$Doc = \langle Classifier, SourceVC, Creator, Article \rangle, \quad (2.22)$$

де $Classifier = \{Classifier_i\}_{i=1}^{N^{(Classifier)}}$ – множина класифікаторів;

$N^{(Classifier)}$ – всі класифікатори що зустрічаються в документації до ПЗ;

$SourceVC = \{\text{C}ommunity_i\}_{i=1}^{N^{(Community)}}$ – множина джерел;

$N^{(Source)}$ – джерела з яких були стягнуті повідомлення для формування документації;

$Creator = \{Creator_i\}_{i=1}^{N^{(Creator)}}$ – множина авторів;

$N^{(Creator)}$ – автори статей що є в документації;

$Article = \{Article_i\}_{i=1}^{N^{(Article)}}$ – множина статей, що утворюють документацію;

$N^{(Article)}$ – статті що утворюють документацію.

Класифікатор можна представити у вигляді графу, вершинами якого виступають терміни, а дуги відображають зв'язки між термінами. Зазвичай класифікатор представляють у вигляді лісу або дерева.

Класифікатор ДПЗ характеризується такими параметрами:

$$Classifier_i = \langle TitleClassifier_i, Term(Classifier_i), TermRel_i \rangle, \quad (2.23)$$

де $TitleClassifier_i$ – назва *i*-го класифікатору;

$Term(Classifier_i) = \{Term_{k_j^{(TC_i)}}\}_{j=1}^{N^{(TC_i)}}$ – множина атомарних, неподільних термінів для *i*-го класифікатору;

$k_j^{(TC_i)}$ – множина номерів термінів, що належать *i*-му класифікатору;

$N^{(TC_i)}$ – загальна кількість термінів для i-го класифікатору;

$TermRel_i \equiv Term_i \times Term_i$ – множина взаємозв'язків між термінами i-го класифікатору, що виступає підмножиною декартового добутку двох множин термінів.

Спільноту опишемо у вигляді множини тем, що завантажені до СКД:

$$Community = \{Theme_k\}_{k=1}^{N^{(Theme)}}, \quad (2.24)$$

де $N^{(Theme)}$ – кількість тем,

$Theme_k \subset \boldsymbol{Theme}$ – тема з множини тем СКД.

ДПЗ має повідомлення, які у ВС виступають дописами. Стаття ($Article$) ДПЗ може містити як один так і множину повідомлень, що для ВС є дискусією.

Стаття визначається множиною повідомлень що їй належать:

$$Article = \{Message_i\}_{i=1}^{N^{(Message)}}, \quad (2.25)$$

де $Message_i$ – i-те повідомлення, що є в статті;

$N^{(Message)}$ – кількість повідомлень, які містяться у статті.

Зазвичай перше повідомлення в статті задає тему обговорення – повідомлення-дискусія ($MessageDiscuss$), а всі наступні є відповідями (повідомлення-коментар – ($MessageCom$), що певною мірою належать до першого допису.

В залежності від типу статті повідомлення-коментарі можуть утворювати таку структуру:

- ієрархічну;
- пласку.

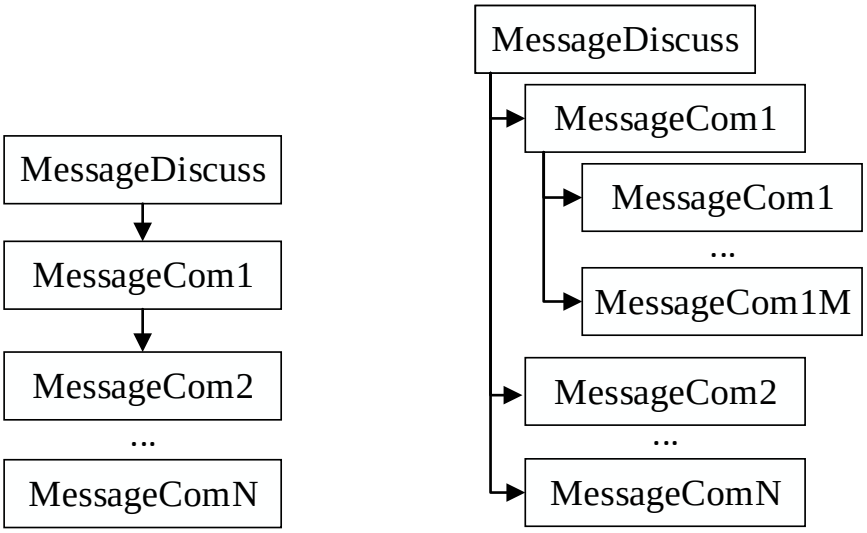


Рис.2.3 Структури статті: а – ієрархічна; б – пласка.

Через те, що одна особа (користувач) може мати декілька профілів у різних спільнотах введено поняття:

- автора повідомлення документації ($Creator_i$) – особа, яка ідентифікується в межах однієї або декількох спільнот, та публікує матеріали, які в подальшому стають контентом документації;
- автора допису ($Author_j$) є зареєстрований користувач спільноти, що розміщує матеріали в конкретній спільноті.

Відповідно i-го автора повідомлення ДПЗ можна подати у такому вигляді:

$$Creator_i = \langle Author_i, CreatorRate_i \rangle, \quad (2.26)$$

де $Author_i \subset User$ – множина профілів i-го автора, дописи якого були завантажені для формування ДПЗ;

$CreatorRate_i$ – рейтинг i-го автора.

2.2.1. Опис структури класифікаторів документації програмного забезпечення

Класифікатори ДПЗ можна поділити на такі погляди:

- приналежності до конкретного ПЗ ($Classifier_1$);
- тематики відповідно до ЖЦ ПЗ ($Classifier_2$);
- класи задач споживачів ПЗ, що відображають функції, вид ПЗ тощо ($Classifier_3$);

- архітектурні питання ($Classifier_4$);
- найменування (поєднання) всіх задіяних ПЗ ($Classifier_5$): основне ПЗ на яке формується ДПЗ; залежні ПЗ (які задіяні для успішного функціонування програми); ті програмні продукти, що використовують описане ПЗ;
- словник термінів ($Classifier_6$).

Класифікатори мають множину термінів деревовидної структури, що визначають його.

Класифікатор приналежності до конкретного ПЗ ($Classifier_1$) може мати структуру термінів що представлено на рис. 2.4.

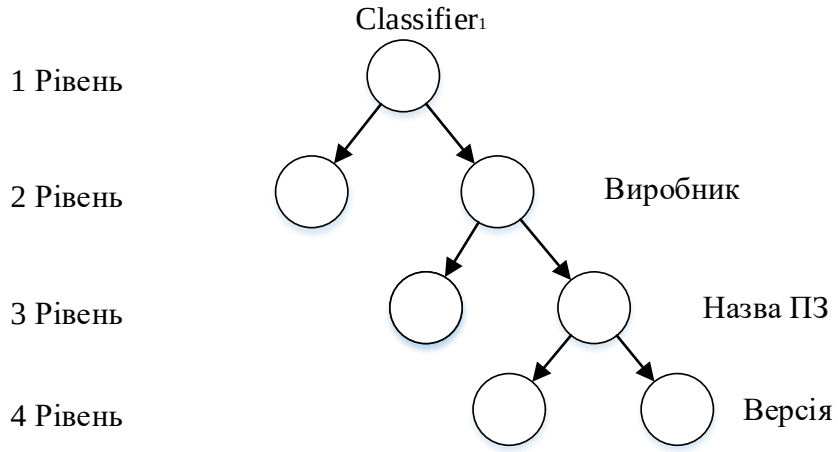


Рис.2.4 Структура класифікатору приналежності до ПЗ

Класифікатор приналежності до ПЗ має такі рівні:

- перший рівень – термін назви класифікатору;
- другий рівень – терміни назви виробників ПЗ (наприклад, Intel, Microsoft тощо);
- третій рівень – терміни назв ПЗ (наприклад, Pentium, Azure, Visual Studio Code тощо);
- четвертий рівень – терміни, що визначають версії ПЗ – номер або назва (Lite, Pro тощо).

Класифікатор ЖЦ ПЗ ($Classifier_2$) має структуру термінів що представлено на рис. 2.5.

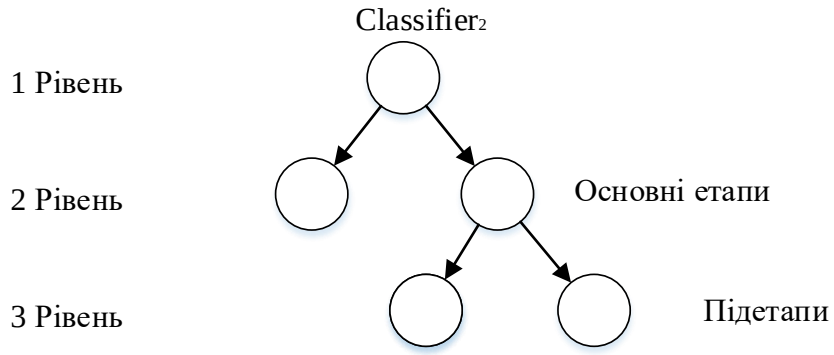


Рис.2.5 Структура класифікатору ЖЦ ПЗ

Класифікатор ЖЦ ПЗ має такі рівні:

- перший рівень – термін назви класифікатору;
- другий рівень – основні, загальні етапи ЖЦ ПЗ (вимоги, аналіз, розробка, тестування, експлуатація);
- третій рівень – деталізація термінів попереднього рівню.

Класифікатор класів задач споживачів ПЗ ($Classifier_3$) має структуру термінів що представлено на рис. 2.6.

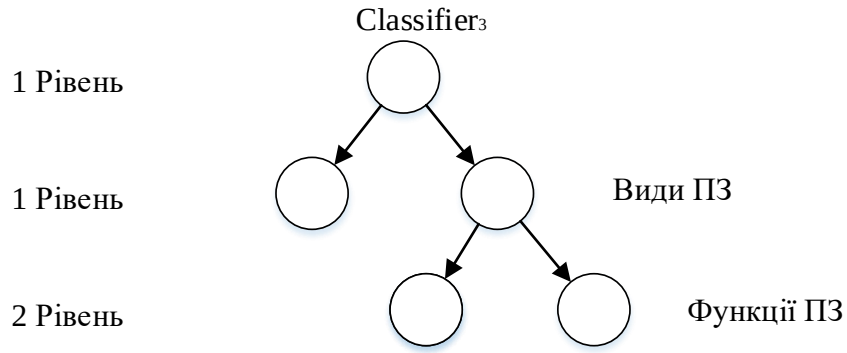


Рис.2.6 Структура класів задач споживачів ПЗ

Класифікатор задач споживачів ПЗ має такі рівні:

- перший рівень – термін назви класифікатору;
- другий рівень – види ПЗ (наприклад, база даних (БД), текстовий редактор, браузер тощо);
- третій рівень – назви функцій ПЗ.

Класифікатор, який містить архітектурні питання щодо ПЗ ($Classifier_4$) може мати структуру відображену на рис.2.7.

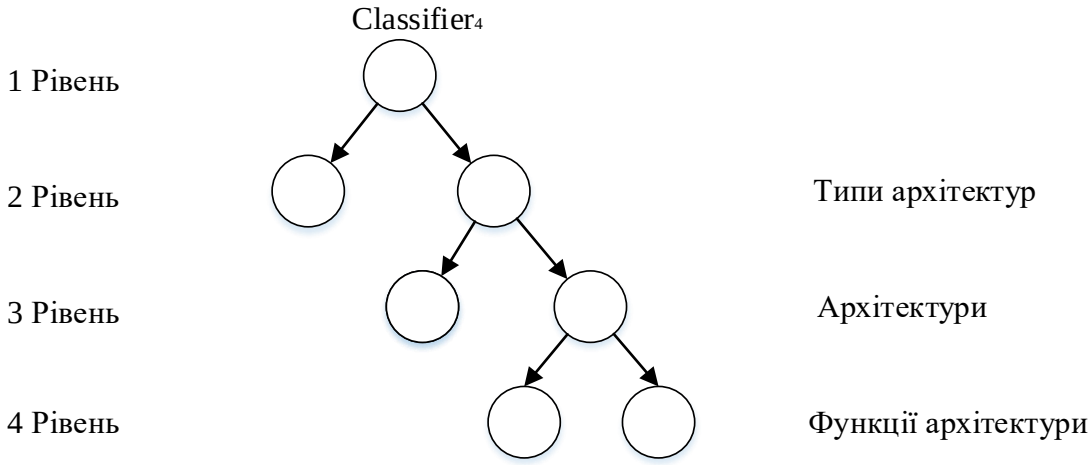


Рис.2.7 Структура класифікатору архітектурних питань ПЗ

Класифікатор архітектурних питань ПЗ має такі рівні:

- перший рівень – термін назви класифікатору;
- другий рівень – типи архітектур (наприклад, розподілена, архітектура «клієнт-сервер», розподілена архітектура «файл-сервер», розподілена архітектура веб-додатків, сервіс-орієнтована архітектура (від англ. SOA) тощо).
- третій рівень – архітектури (MVVM (Model-View-ViewModel), мікросервісна архітектура (microservices або microservice architecture));
- четвертий рівень – назви функцій, інтерфейсів, стилів, шаблонів, методів.

Класифікатор словнику термінів ($Classifier_6$) містить в собі всі вагомі терміни (без повторень), що зустрічаються у статтях ДПЗ.

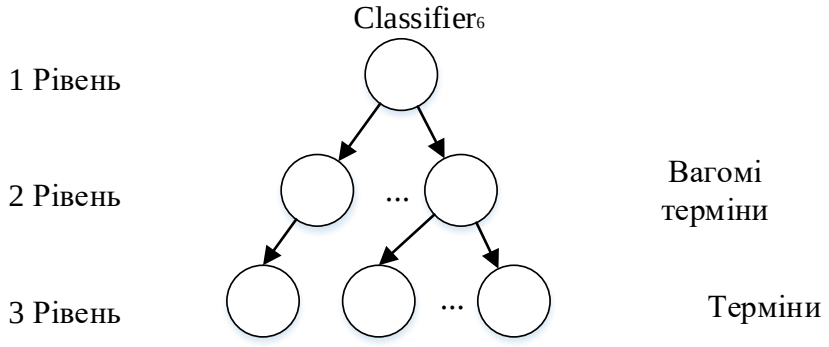


Рис.2.8 Структура класифікатору словнику термінів

Класифікатор словнику термінів, що були виявлені у статтях ДПЗ має такі рівні:

- перший рівень – термін назви класифікатору;
- другий рівень – вагомі терміни, що задовольняють умові 2.42 наведеній у розділі 2.5;
- третій рівень – терміни за допомогою яких утворено вагомі терміни.

Слід зазначити, що кількість рівнів наведених дерев термінів може бути більшою завдяки деталізації останнього рівня.

2.2.2. Побудова словників термінів для формування документації програмного забезпечення

Для опису класифікаторів, проведення аналізу тональності тексту, застосування методів обчислення релевантності запиту до тексту, виявлення подібності серед текстів, усунення небажаного інформаційного наповнення ДПЗ, потрібно розробити та наповнити відповідні словники. Словник містить колекцію елементів, де кожен з них має унікальний ключ, який є константою, та асоційоване з ним певне значення [117].

Словники потрібно використовувати у таких випадках:

- зберігання будь-яких даних, пов’язаних з об’єктом. Ключі – об’єкти, значення – пов’язані з ними дані;
- встановлення відповідності між об’єктами (наприклад, «батько-нащадок»). Ключ – об’єкт, значення – відповідний йому об’єкт;
- підрахунку кількості якихось об’єктів. В такому випадку потрібно сформувати словник, у якому ключами є об’єкти, а значеннями — їх кількість.

Для зберігання та аналізу великих обсягів даних доцільно застосовувати словники [130]. Переваги застосування словників даних для формування ДПЗ:

- масштабованість (здатність БД зберігати великі обсяги даних без втрати продуктивності при обробці інформації);

- швидкість сприйняття даних експертом, що формує словники завдяки їх зручному представленню;
- гнучкість, адаптивність до зміни структур баз даних, що може виникнути при обробці інформації.

Формування ДПЗ передбачає наявність таких словників термінів:

- словник «stop-word» (англійською та українською мовами) – слова, що не несуть смислового навантаження щодо основного змісту тексту;
- «сленговий» словник що притаманний для ІТ-сфери. Зазвичай це є англомовні терміни або скорочені форми англійський слів які говорять українською мовою (приклад слів наведено у Додатку Ж);
- словник «небажаного інформаційного наповнення» (терміни, посилання які є неприйнятними для інформаційного наповнення ДПЗ) англійською та українською мовами;
- словник «абревіатур, скорочень»;
- словник «помилок» написання слів авторами при розміщенні дописів;
- «словник емоційного забарвлення» україномовного [73] та англомовного тексту [89];
- словник «класифікаторів», що місить набір регулярних виразів і термінів (до яких належать ці вирази) з ваговими коефіцієнтами.

2.3. Сценарії обчислення рейтингу допису розміщеному у віртуальних спільнотах

Всі опубліковані дописи у ВС основані виключно на досвіді авторів, що їх опублікували. Відомо, що досвід є сукупністю практично опанованих знань, навичок, основаних на пережитому, випробуваному людиною. Досвід користувача не завжди може відповідати дійсності подій що відбуваються, тому дані, розміщені у дописі, потребують перевірки. Більшість спільнот надають

можливість користувачам оцінювати дописи залишаючи реакції, тим самим формуючи його рейтинг. Чим більше рейтинг допису тим більше є рівень довіри до даних розміщених в ньому. В той час як низький рейтинг потребує більшої уваги щодо аналізу достовірності інформаційного наповнення допису.

На процедуру аналізу рівня довіри розміщених даних впливають наступні чинники:

1. Аналіз рейтингу допису на основі реакцій, що залишили інші користувачі до нього.
2. Результати застосування різноманітних лінгвістичних методів аналізу тексту допису та коментарів.

Дослідивши більше десятка тематичних ВС (Таблиця 1) було виявлено наступні реакції на допис, що можуть залишати користувачі:

1. «Оцінювання» (вподобання, несхвалення тощо) описує позитивні або негативні емоції від контенту;
2. «Взаємодії» (збереження, поширення, перегляд допису);
3. «Коментування» для вираження своєї думки щодо інформаційного наповнення опублікованого допису. Коментар може мати позитивні так і негативні емоції, для розпізнавання яких застосовують аналіз тональності тексту [77, 127].

Таблиця 2.1

Порівняльна характеристика щодо наявності реакцій у ВС

Назва віртуальної спільноти	**Реакції**												
	Оцінювання									**Взаємодії**			**Commenting**
	Like (upvotes)	**Dislike (downvote)**	**Love**	**Care**	**Haha**	**Wow**	**Sad**	**Angry**	**In the range**	**Save**	**Share**	**View**	
Facebook.com	+	–	+	+	+	+	+	+	–	+	+	–	+
Dou.ua	+	–	–	–	–	–	–	–	–	+	+	+	+
Reddit	+	+	–	–	–	–	–	–	–	+	+	–	+
Replace	–	–	–	–	–	–	–	–	–	–	–	–	+
Senior.ua	–	–	–	–	–	–	–	–	–	–	+	+	+
CodeProject	–	–	–	–	–	–	–	–	+	+	+	+	+
Stack Overflow	+	+	–	–	–	–	–	–	–	+	+	+	+
XDA Developers	+	–	+	–	+	+	+	+	–	+	+	–	+
thescriptforum.proboards.com	+	–	–	–	–	–	–	–	–	+	+	–	+
CodeGuru Forums	–	–	–	–	–	–	–	–	+	+	+	–	+
Hot Scripts	–	–	–	–	–	–	–	–	+	+	–	+	+
SitePoint Forum	+	–	–	–	–	–	–	–	–	–	+	+	+
DZone	+	–	–	–	–	–	–	–	–	+	+	+	+
BYTES	–	–	–	–	–	–	–	–	–	–	–	+	+
Quora	+	+	–	–	–	–	–	–	–	+	+	+	+
Lobsters	+	+	–	–	–	–	–	–	–	–	+	–	+
StackExchange	+	+	–	–	–	–	–	–	–	+	+	+	+
Coderanch	+	+	–	–	–	–	–	–	–	–	+	–	+
FindNerd	+	+	–	–	–	–	–	–	–	+	+	+	+
Digital Ocean Community	–	–	–	–	–	–	–	–	–	–	+	–	+

Більшість різновидів реакції «Оцінювання» притаманні спільнотам, що знаходяться у соціальній мережі Facebook:

- «Вподобання» (Like) викликає приємні враження від розміщеної інформації в дописі завдяки тому, що контент відповідає смакам, бажанням, настроям користувача [74];
- «Несхвалення» (Dislike) виражає осуд, заперечення та невдоволення щодо інформаційного наповнення допису [50];
- «Любов» (Love) викликає почуття глибокої прихильності, що охоплює низку сильних і позитивних емоцій [44] (від розміщеної інформації автором в дописі);
- «Турбота» (Care) є символом підтримки та піклування (щодо опублікованого контенту);
- «Сміх» (Haha) викликають двоякі почуття бо сміх може бути позитивний та саркастичний від прочитаної інформації в дописі;
- «Здивування» (Wow) почуття, що виникає від отримання несподіваної, незрозумілої інформації, яка вражає;
- «Пригніченість, сум» (Sad) викликає негативні емоції від отриманої інформації;
- «Злість» (Angry) викликає дуже негативні, недоброзичливі, обурливі емоції у користувачів.

Існують ВС, в яких оцінка дописів знаходиться в певному діапазоні. Наприклад, спільноти CodeProject, CodeGuru Forums та Hot Scripts здійснюють оцінювання дописів значеннями від одиниці (найменший бал) до п’яти «зірочок» (найвищий бал). Тому для обчислення рейтингу допису необхідно надати вагові коефіцієнти кожній реакції на нього. Природніми обмеженнями оцінки рейтингу допису вважатимемо інтервал значень [-1, 1].

Так як, майже кожна ВС має свій набір оцінок (що частково перетинаються), тому обчислення рейтингу допису буде відрізнятися для кожної з них. Провівши аналіз реакцій користувачів представлених в табл. 2.1. встановлено, що оцінювання допису відбувається одним з наступних випадків:

1. «в діапазоні» зазначеному (обчислення однієї позначки наведено в формулі 2.28);
2. емоції значеннями «Like» (для вираження позитивної реакції) та «Dislike» (для вираження негативної реакції);
3. емоції значеннями: «Like», «Love», «Care», «Haha», «Wow», «Sad», «Angry».

Для більш точного обчислення рейтингу дописів різних ВС для кожного випадку застосуємо різні коефіцієнти щодо емоцій. Адже чим більша кількість емоцій тим точніше можна визначити рейтинг допису (випадок 1, 3). В той час як емоції Like/Dislike (Випадок 2) не надають повного опису емоцій користувача.

Випадок 1. При аналізі ВС, що мають такий вид оцінювання «В діапазоні» (Таблиця 1) був представлений наступний діапазон оцінок – від 1 до 5-ти зірочок, де п’ять зірочок є найвищим показником, а одна зірочка – найнижчою оцінкою. Але значення діапазону оцінок для кожної спільноти може бути різною, тому при виникненні такої ситуації необхідно рівномірно поділити значення оцінок між всіма в інтервалі [-1, 1]. Крок, коефіцієнт для однієї позначки (*OneEst*) обчислюватиметься за формулою:

$$OneEst = \frac{|Diapason|}{MaxEst - MinEst} = \frac{|-1-1|}{MaxEst - MinEst} = \frac{2}{MaxEst - MinEst}, \qquad (2.27)$$

де |*Diapason*| – значення діапазону вимірювання оцінок по модулю;
MaxEst – максимальна запропонована оцінка допису;
MinEst – мінімальна запропонована оцінка допису.

Наприклад, згідно з формулою 2.27 при інтервалі [-1, 1] оцінювання допису спільноти «CodeProject» (рис.2.11.) розподіл балів буде наступний:

$$OneEst = \frac{2}{MaxEst - MinEst} = \frac{2}{5-1} = 0.5, \qquad (2.28)$$

Отже, п’ять зірок оцінка становить 1; чотири зірки – значення 0.5; три зірки – значення 0; дві зірки – оцінка -0.5; одна зірка – оцінка -1.

Випадок 2. Для ВС що мають оцінювання допису такими реакціями *Like* та *Dislike*. Коефіцієнти мають бути наступні: $k_{like} = 1$ і $k_{dislike} = -1$.

Випадок 3. Емоції, що представлені в даному випадку мають наступні коефіцієнти:

- вподобання $k_{like} = 0.5$;
- любов $k_{love} = 1$;
- турбота $k_{care} = 0.7$;
- сміх $k_{haha} = 0.35$;
- здивування $k_{wow} = 0.4$;
- пригніченість, сум $k_{sad} = -0.5$;
- злість $k_{angry} = -1$.

Значення реакції оцінювання допису визначатимемо:

$$R_{\text{оц}} = \frac{k_{like}*C_{like}+k_{love}*C_{love}+k_{care}*C_{care}+k_{haha}*C_{haha}+k_{wow}*C_{wow}+k_{sad}*C_{sad}+k_{angry}*C_{angry}}{CountOfEstimate}, \quad (2.29)$$

де k_{like} – значення коефіцієнту показника реакції «Вподобання» (відносно випадку що наведені вище);

C_{like} – кількість користувачів що залишили реакцію «Вподобання»;

k_{love} – значення коефіцієнту показника реакції «Любов»;

C_{love} – кількість користувачів що залишили реакцію «Любов»;

k_{care} – значення коефіцієнту показника реакції «Турбота»;

C_{care} – кількість користувачів що залишили реакцію «Турбота»;

k_{haha} – значення коефіцієнту показника реакції «Сміх»;

C_{haha} – кількість користувачів що залишили реакцію «Сміх»;

k_{wow} – значення коефіцієнту показника реакції «Здивування»;

C_{wow} – кількість користувачів що залишили реакцію «Здивування»;

k_{sad} – значення коефіцієнту показника реакції «Пригніченість»;

C_{sad} – кількість користувачів що залишили реакцію «Пригніченість»;

k_{angry} – значення коефіцієнту показника реакції «Злість»;

C_{angry} – кількість користувачів що залишили реакцію «Злість»;

$CountOfEstimate$ – кількість оцінок допису.

Реакція «Взаємодії» з дописом описується такими показниками:

- «Збереження» (Save) допису відображає цінність для користувача контенту розміщеного в ньому для подальшого звернення та його опрацювання;
- «Поширення» (Share) допису іншим користувачам або на своїй сторінці відображає про те, що користувач рекомендує його інформаційне наповнення іншим. Рекомендації, поради, є позитивною, схвальною характеристикою кого, чого-небудь;
- «Перегляди» (View) допису відображає загальну кількість користувачів, які здійснили огляд допису.

Оцінки реакцій «Поширення» та «Збереження» потрібно обчислювати відносно показника «Перегляди» допису, адже вони не є унікальними як реакції «Оцінювання», бо один користувач може здійснити декілька реакцій (наприклад, зберегти і поширити допис).

Згідно проаналізованих даних, що наведені у Таблиці 1, є спільноти що не надають інформацію щодо переглядів дописів. В такому випадку необхідно здійснити запит до адміністратора чи модератора спільноти. На основі даних про перегляди можна обчислити оцінку реакції «Взаємодії» з дописом. Якщо дані про кількість переглядів отримати не можливо, значення реакції «Взаємодії» не обчислюється і рейтинг допису визначатиметься виключно за допомогою реакцій «Оцінювання» і «Коментування».

Для всіх вище наведених випадків реакція «Взаємодії» матиме коефіцієнти наведені в таблиці 2.2, що в сумі дорівнюють одиниці.

Таблиця 2.2

Значення коефіцієнтів в залежності від набору реакції «Взаємодії»

№	**Назва реакції взаємодії**	**Значення коефіцієнтів**
1	«Поширення», «Збереження»	$k_{share} = 0.6, k_{share} = 0.4$
2	«Поширення»,	$k_{share} = 1$
3	«Збереження»	$k_{save} = 1$

Коефіцієнт показника «Поширення» є рекомендаційною реакцією та охоплює більше коло користувачів ніж реакція «Збереження» (для повторного звернення користувача до його контенту), тому значення коефіцієнту має бути більшим.

Для оцінювання реакції «Коментування» доцільно провести аналіз тональності тексту дописів-коментарів, що дозволить автоматизовано виявити емоційно забарвлену лексику і оцінки авторів (думок) в текстах по відношенню до об'єктів, мова про які йде в тексті [77, 127]. Даний аналіз дає можливість провести класифікацію коментарів до допису за наступними класами оцінок [107]: негативна, позитивна і нейтральна. Оцінювання реакції «Коментування» здійснюється на основі значення кількості позитивних і негативних емоцій. Наявність нейтрального класу відображає, що допис містить не тільки «емоційний» текст, але і інше інформаційне наповнення (наприклад, факти про ПЗ), що може бути корисним для ДПЗ.

Значення реакції взаємодії користувачів з дописом визначатимемо:

$$R_{\text{взаєм}} = \frac{k_{share} * CountOfShare}{CountOfView} + \frac{k_{save} * CountOfSave}{CountOfView}, \quad (2.30)$$

де k_{share} – значення коефіцієнту показника «Поширення» (в залежності від кількості реакцій що надає ВС для оцінювання допису – Таблиця 2);

$CountOfShare$ – кількість «Поширення» допису;

$CountOfView$ – кількість «Переглядів» допису;

k_{save} – значення коефіцієнту показника «Збереження»;

$CountOfSave$ – кількість «Збереження» допису.

Значення реакції «Коментування» допису на основі оцінок аналізу тональності тексту коментарів до нього визначатимемо:

$$R_{\text{ком}} = \frac{\sum_{j=1}^{CountOfCom} \left({pos}/{(pos+neg)} \right)}{CountOfCom}, \quad (2.31)$$

де pos – значення позитивного забарвлення тексту допису;

neg – значення негативного забарвлення тексту допису;

$CountOfCom$ – кількість коментарів до допису.

Отже, рейтинг допису ВС що мають реакції «Оцінювання», «Взаємодії» та «Коментування», визначатимемо наступним чином:

$$Rate(Post) = k_{\text{оц}} * R_{\text{оц}} + k_{\text{взаєм}} * R_{\text{взаєм}} + k_{\text{ком}} * R_{\text{ком}}, \qquad (2.32)$$

де $k_{\text{оц}}$ – значення коефіцієнту показника «Оцінювання» (рекомендовано $k_{\text{оц}} = 0.7$);

$R_{\text{оц}}$ – значення оцінки показника «Оцінювання»;

$k_{\text{взаєм}}$ – значення коефіцієнту показника «Взаємодії» (рекомендовано $k_{\text{взаєм}} = 0.1$);

$R_{\text{взаєм}}$ – значення оцінки показника «Взаємодії»;

$k_{\text{ком}}$ – значення коефіцієнту показника «Коментування» (рекомендовано $k_{\text{ком}} = 0.2$);

$R_{\text{ком}}$ – значення оцінки показника «Коментування».

Дослідивши значення реакцій «Оцінювання» та «Коментування» для спільнот, які містять їх, було визначено, що оцінки значень реакцій є пропорційними, тобто якщо допис має значення реакції «Оцінювання» то і коментарі до нього були позитивними та, згідно аналізу настроїв, давали високі значення. Існують ВС, що не мають реакції «Оцінювання», але мають такі реакції:

- «Взаємодії» та «Коментування» – (наприклад, ВС Senior.ua);
- «Коментування» – спільнота Replace.

Враховуючи вище сказане, оцінку рейтингу дописів можна здійснювати для ВС, що не мають реакції «Оцінювання», але обов’язковою наявною реакцією має бути «Коментування», але для дописів (нових) які не мають коментарів обчислити рейтинг буде не можливим. При цьому, за наявності у ВС реакцій «Взаємодії» та «Коментування» розподіл коефіцієнтів для обчислення за формулою 2.32 буде наступним: $k_{\text{взаєм}} = 0.25$, $k_{\text{ком}} = 0.75$. Відповідно для ВС, що

мають тільки реакцію «Коментування» значення коефіцієнту буде дорівнювати одиниці.

Порогове значення щодо рівня достовірності даних розміщених у дописів визначається виконанням умови:

$$Rate(Post) \geq \alpha, \tag{2.33}$$

де α – мінімальне порогове значення рейтингу допису для прийняття до розгляду інформаційного наповнення допису.

Схема обчислення допису згідно наявних реакцій в тій чи іншій спільноті наведено схемою у Додатку Г.

2.4.1. Обчислення рейтингу автору допису

Обчислення рейтингу автору є доцільним у таких випадках:

- для нових дописів, що не набрали достатню кількість часу та реакцій для обчислення рейтингу, але їх інформаційне наповнення потрібно брати до уваги для формування ДПЗ;
- для дописів, які з певних причин не мають значень реакцій «Оцінювання», але можуть мати рейтинг за іншими дописами або оцінити рейтинг допису можна за реакціями «Взаємодії» та «Коментування» наприклад:

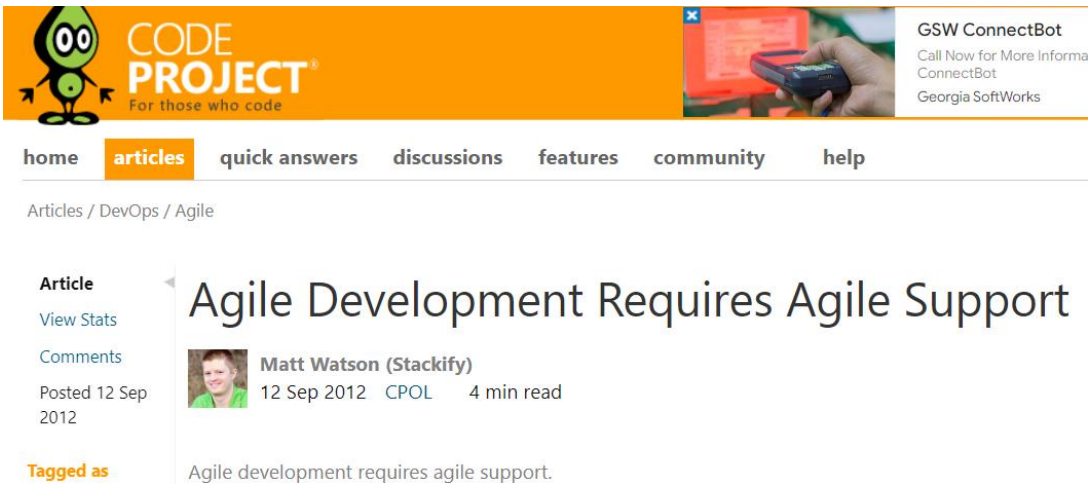


Рис.2.9 Приклад допису спільноти CodeProject що не має оцінок реакцій користувачів

Допис спільноти CodeProject, що має оцінку реакцій користувачів наведено на рис. 2.18.

Рейтинг *i*-того автору певної теми обчислюватимемо як середнє зважене рейтингів всіх дописів, що він розмістив:

$$Rate(Author_i) = \frac{\sum_{j=1}^{CountOfPost(Author_i)}(Rate(Post_j(Author_i)))}{CountOfPost(Author_i)}, \quad (2.34)$$

де $CountOfPost(Author_i)$ – кількість дописів, що опублікував *i*-тий автор в певній темі, для яких відомий рейтинг;

$Rate\left(Post_j(Author_i)\right)$ – рейтинг *j*-го допису що розмістив *i*-тий автор.

Обчислення рейтингу допису наведено формулою 2.32.

2.4. Метод виявлення подібності між текстами дописів

Важливою характеристикою ДПЗ є наявність унікального інформаційного наповнення, однією з властивостей якої є компактність представлення даних. Для забезпечення характеристики унікальності потрібно після процедури завантаження дописів до СКД перевіряти їх вміст. Дану процедуру можна виконати застосувавши різні методи та моделі, що допомагають встановити унікальність тексту або визначити значення міри подібності [106, 107, 128].

Встановлення подібності між двома одномовними текстами дописів відбувається наступним чином:

1. Нормалізація текстів дописів [95].
2. Формування спільного словнику для всіх дописів.
3. Застосування показника TF-IDF для кожного тексту допису.
4. Перетворення кожного тексту допису у вектори.
5. Обчислення косинусу подібності між векторами.

2.4.1. Нормалізація тексту допису

Для проведення аналізу тексту допису за різними методами попередньо необхідно його нормалізувати. Нормалізація – це метод, у якому кожне слово тексту приводиться до його словникової форми, щоб скоротити час пошуку [95].

Слова, які мають те саме значення, але мають деякі відмінності в залежності від контексту чи речення, нормалізуються.

Для нормалізації тексту зазвичай застосовують операцію стемінгу, результатом виконання якої є знаходження основи слова (стеми) за допомогою усування допоміжних частин, таких, як закінчення та суфікс. Результати стемінгу інколи є схожими на визначення кореня слова, але, при цьому, його алгоритми ґрунтується на інших принципах [62]. Тому слово після обробки алгоритмом стемінгу (стематизації) може відрізнятися від морфологічного кореня слова [62]. Для усунення цього питання необхідно застосовувати алгоритм лематизації.

Алгоритмічний процес лематизації полягає у знаходженні леми слова в залежності з його значенням. Лематизація зазвичай відноситься до морфологічного аналізу слів, метою якого є видалення флективних закінчень. Тому даний підхід вважається кращим за інші алгоритми стемінгу бо результатом є знаходження леми, яка є первісною, базовою формою її флективних форм. Недоліком застосування лематизації є залежність від правильності розпізнавання частин мови.

Застосування процесу лематизації для англомовного або україномовного тексту, попередньо передбачає нормалізацію текстів дописів.

Нормалізація тексту полягає у виконанні наступних кроків:

1. Стандартизація тексту:

   1.1. Розбиття тексту на лексеми (слова).

   1.2. Очищення слів від апострофів, знаків пунктуації – токенизація тексту [7].

2. Вилучання слів, які не беруть до уваги при проведенні лінгвістичного аналізу адже вони не несуть важливу інформацію для тексту (займенники, артиклі, прийменники, сполучники) – стоп-слова (з англ. stop words) [77].

Результатом проходження процесу лематизації є нормалізація слів, що містяться в тексті:

- для англомовного слова – лема;
- для україномовного слова – множина лем (частини мови та відмінки слова у тексті)

Існують випадки, коли слова україномовного тексту можуть не отримати лему, наприклад англомовні слова (назви ПЗ тощо) або яких немає у словнику (наприклад слова з помилками). В таких випадках слова належать до наступним класам:

- «unclass» – для слів інших мов;
- «unknown» – для слів, яких немає у словнику.

```
До нормалізації: Get daily status reports for successful and failed backups
Після нормалізації: get daily status report successful failed backup
```

Рис.2.10 Нормалізація англомовного тексту

```
До нормалізації: У жодному разі не можна давати можливість користувачу встановлювати прості паролі.
                 З цими вимогами допоможуть pam rules: pluggable authentication modules.
Після нормалізації: можливість користувач встановлювати простий пароль
                    вимога допомогти pam rule pluggable authentication module
```

Рис.2.11 Нормалізація україномовного тексту

Слова, які не є україномовними належать класу «unclass» та автоматично нормалізуються відповідно до англійської мови.

2.4.2. Обчислення показнику TF-IDF для тексту допису

Статистичний показник TF-IDF (від англ. TF — term frequency, IDF — inverse document frequency) [48] відображає оцінку важливості слів у документі на основі частоти використання їх у тексті [77]. Тобто чим частіше слово зустрічається в тексті тим більшу вагу, внесок воно має щодо його змісту. Значимість (вага) слова є пропорційною щодо кількості його вживань у документі, та обернено пропорційною частоті вживання цього слова в інших документах колекції [48].

Статистичний показник TF-IDF визначається наступним чином [48]:

$$TF - IDF = TF \times IDF = \frac{n_l}{\sum_{i=1}^{k} n_i} \times \ln(\frac{n_c}{\sum_{j=1}^{m} n_j}), \quad (2.35)$$

де n_l – число входжень l-го слова щодо тексту допису;

n_i – загальна кількість слів у дописі;

n_c – кількість дописів за якими сформовано словник;

n_j – кількість дописів в яких виявлено l-те слово.

Показник TF-IDF застосовують в задачах аналізу текстів та інформаційного пошуку.

2.4.3. Виявлення подібності між текстами дописів

Для побудови вектору до кожного допису необхідно його текст перетворити у числа, де координати вектору будуть утворювати слова тексту. Формування векторів ґрунтується на основі обчисленого TF-IDF показника. Важливо врахувати, що значення кожного елемента вектору має відображати те саме слово у векторах для всіх текстів дописів. Також, всі вектори повинні бути приведені до єдиної довжини (розміру).

Виявлення подібності векторів передбачає визначення відстані між ними за допомогою обчислення косинусу кута, що представляється за допомогою скалярного добутку та норми кожного вектору. Косинус кута між двома векторами обчислюється наступним чином:

$$\cos(\theta) = \frac{Vector_1 \cdot Vector_2}{\|Vector_1\| \|Vector_2\|}, \quad (2.36)$$

де $Vector_1$ – координати першого вектору (тексту допису);

$Vector_2$ – координати другого вектору;

$\|Vector_1\|$ – норма першого вектору;

$\|Vector_2\|$ – норма другого вектору.

Вектори вважаються подібними, якщо вони мають однаковий напрямок [101]. При чому довжина тексту допису при оцінці подібності не є важливою. Значення косинусу кута між двома текстами дописів може знаходитися в інтервалі [0, 1]. Якщо дописи мають однакове інформаційне наповнення то

значення метрики буде дорівнювати одиниці. Якщо в текстах дописів не виявлено подібності тоді значення буде дорівнювати нулю.

Відображення подібності між текстами дописів доцільно представити у вигляді матриці, яка є симетричною до головної діагоналі (рис.2.12). Для проведення дослідження вибірка, словник склав 180 дописів. Кількість слів 10 тисяч.

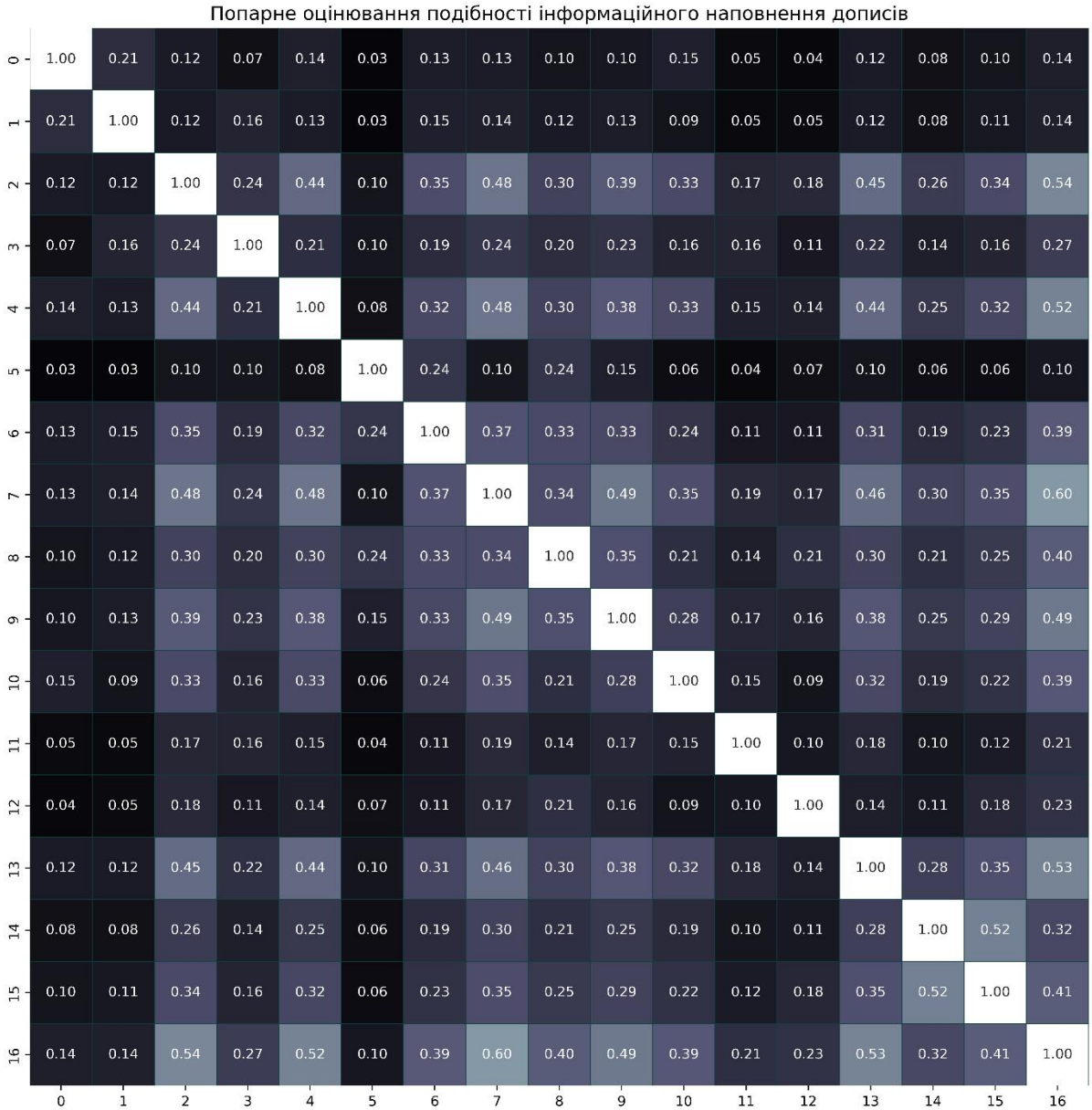


Рис.2.12 Матриця попарної косинусоїдної подібності дописів

Надалі, для отриманих значень коефіцієнтів кореляції між дописами необхідно здійснити перевірку міри подібності:

$$\mu_{similarity} \geq \alpha, \quad (2.37)$$

де α – показник, що визначає порогове значення для оцінки міри подібності тексту допису.

Природніми обмеженнями для $\mu_{similarity}$ є інтервал [0, 1]. Дописи, які задовольняють умові наведеній формулою 2.37 заносять у список для подальшої перевірки за допомогою моделі N-грам (розділ 2.4.4).

2.4.4. Виявлення однакового тексту дописів за допомогою моделі N-грам

Основним недоліком виконання методу виявлення подібності між текстами дописів є те, що при здійсненні аналізу, до уваги не береться довжина тексту допису. Тому, наприклад, якщо один допис є частиною іншого, але має більшу довжину, метод подібності не виявить дану характеристику і результатом буде низьке значення подібності. Хоча такі дописи доцільно об’єднати для уникнення повторень, надлишковості даних в ДПЗ.

Для врахування даної характеристики потрібно застосувати послідовну перевірку слів тексту за допомогою моделі N-грам [100], яку, зазвичай, використовують для виявлення плагіату.

Обравши ту саму вибірку дописів, як і для методу виявлення подібності, було визначено, що знаходження однакового тексту дописів за допомогою N- грам, надає краще результати – вищі значення щодо подібності між текстами (рис.2.13).

Модель N-грам доцільно застосовувати після виконання методу виявлення подібності адже вона передбачає [100]:

- застосування складніших алгоритмів для перевірки текстів, відповідно працює повільніше;
- формування словника для кожного тексту допису з врахуванням N- грам відповідних порядків.
- аналіз здійснюється не між всією вибіркою дописів, а серед пар, значення яких відповідають умові подібності (формула 2.37).

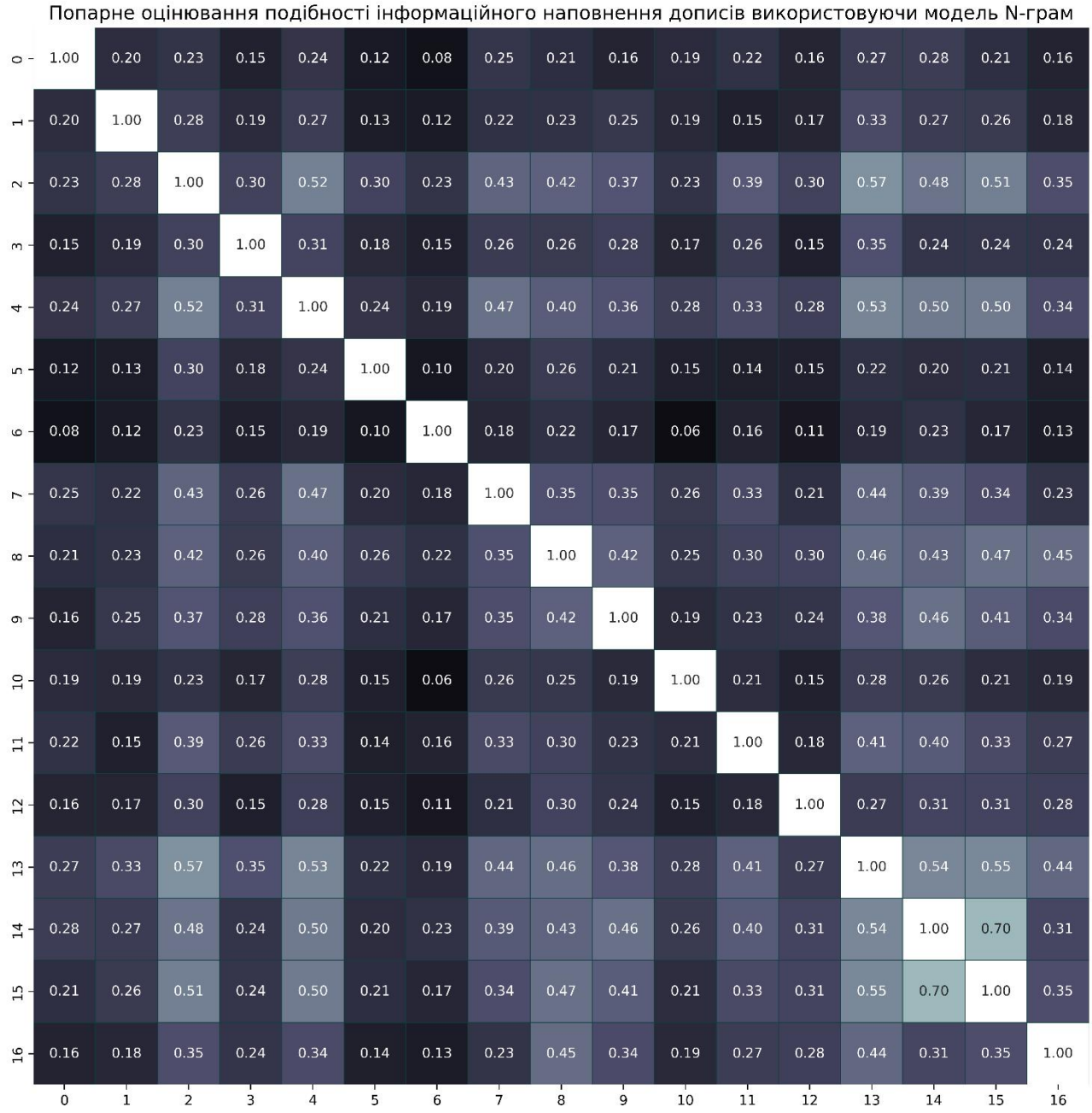


Рис.2.13 Матриця попарної подібності дописів за моделлю N-грам

Врахувавши вище наведене, описано послідовність щодо порівняння для двох текстів дописів за моделлю N-грам, що полягає у наступному:

1. Стандартизація текстів дописів.
2. Формування спільного словника з N-грамами відповідних порядків.
3. Отримання векторного представлення текстів відповідно до сформованого TF-IDF словника.
4. Обчислення косинусоїдної подібності векторів.

Здійснивши пошук двох дописів однієї тематики та перевіривши їх за методами подібності та N-грам (5-го порядку [36]), було отримано наступні результати – рис.2.14

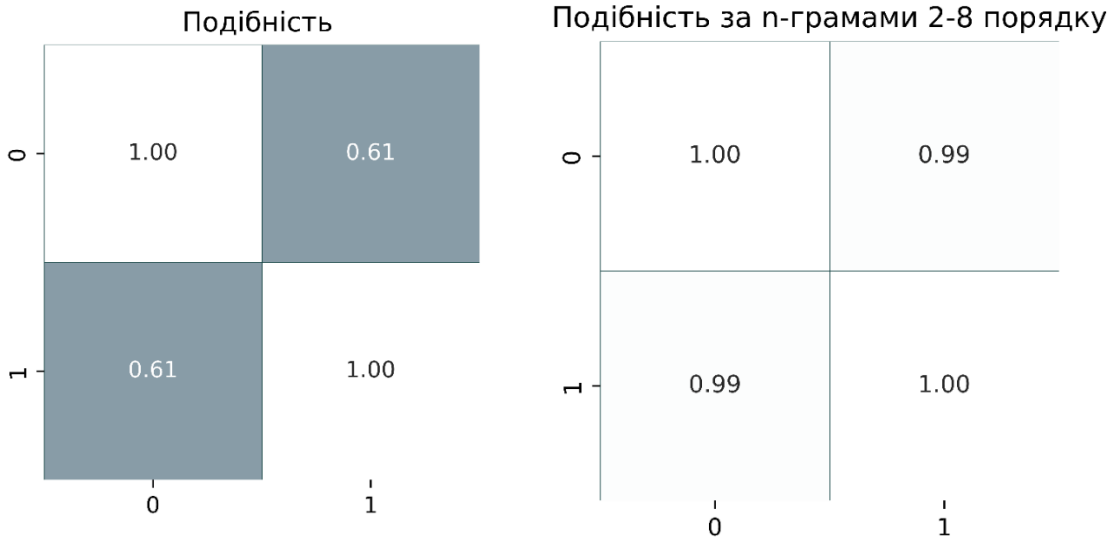


Рис. 2.14 Результат аналізу текстів дописів: а) за методом косинусоїдної подібності; б) за моделлю N-грам (із стоп-словами)

Проаналізувавши вміст цих дописів було виявлено, що один має високу подібність щодо певної частини іншого допису.

Слід зазначити, що при виявленні однакового тексту дописів за моделлю N-грам, незначущі слова (стоп-слова або шумові слова) не були видалені з тексту, але спостерігається тенденція, що для уникнення публікування однакових текстів, користувачі можуть використовувати рерайтинг, основними техніками якого є: заміна фраз та словосполучень паралельними конструкціями, застосування синонімів, спрощення тексту завдяки видаленню слів, які не несуть смислового навантаження тощо. Тому виявлення подібності дописів за допомогою моделі N-грам можна застосовувати із вилученням незначущих, допоміжних слів (stop words). Здійснивши перевірку за моделлю N-грам для двох дописів із вилученням стоп-слів було отримано наступне:

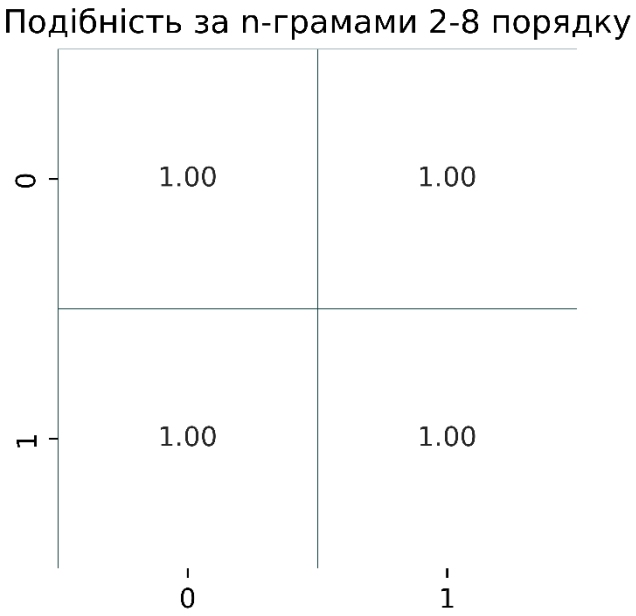


Рис.2.15 Результат аналізу текстів дописів за моделлю N-грам (без стоп-слів)

Отже, порівнювальні частини текстів виявились однаковими. Дослідивши вміст порівнювальних дописів було виявлено, що один допис є частиною іншого. При чому, певні незначущі слова відрізнялись.

Так як метод виявлення подібності передбачає роботу з множиною дописів, а встановлення однакового тексту за моделлю N-грам між двома дописами, тому доцільно спочатку провести аналіз інформаційного наповнення за методом подібності, і, надалі, для дописів, тексти яких задовольняють умову наведену формулою 2.37 провести аналіз за моделлю N-грам. Виконання першого методу подібності дозволить визначити позиції розташування дописів серед всієї вибірки для подальшого їх аналізу щодо однакового тексту за моделлю N-грам.

Результатом проходження алгоритму є встановлення унікальності вмісту серед подібних дописів за допомогою виявлення значення унікальності інформаційного наповнення в них.

Отже, міра унікальності дописів визначається наступним критерієм:

$$\mu_{unique}(Post_i, Post_k) \geq \alpha, \tag{2.38}$$

де α – показник, що визначає порогове значення для оцінки міри унікальності допису.

2.5. Метод автоматизованого виявлення термінів статей за допомогою дерева для прийняття рішень

Інформаційне наповнення або текст статті має ключові фрази, що зберігаються у вигляді символів масок або регулярних виразів (має складну побудову), які відповідають певним термінам ДПЗ.

Термін – це здебільшого однозначне слово чи словосполучення, що виступає назвою поняття [5]. Терміни мають деревовидну структуру, де коренем виступає сам термін, а листями ключові фрази, які йому належать. Ключова фраза – слово або вислів, що містить змістовне навантаження до певного терміну. Один термін може стосуватися множини ключових фраз з певними ваговими коефіцієнтами. В свою чергу одна ключова фраза може відноситися до декількох термінів з різними ваговими коефіцієнтами. Деякі терміни можуть мати однаковий набір ключових фраз, але з різними ваговими коефіцієнтами [68]. Відповідність між термінами і ключовими фразами з ваговими коефіцієнтами задає експерт. Всі терміни та відповідні їм ключові фрази зберігаються у словнику класифікаторів. Приклад відповідності ключових фраз до терміну наведений у Додаток Ж.

Схема обміну даних – виявлення термінів статті на основі ключових фраз наведено на рис.2.16.

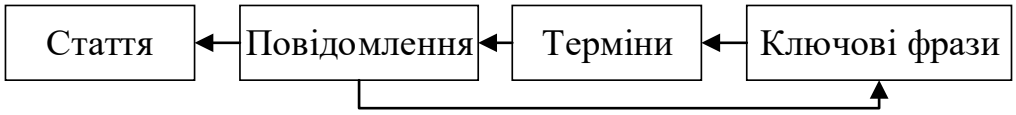


Рис.2.16 Загальна схема передачі даних для виконання даного методу

Для визначення ваги термінів статті потрібно побудувати дерево для прийняття рішення [35]. Дерево прийняття рішень є графічним методом, який пов’язує вузли прийняття рішень (терміни), можливі стратегії (ключові фрази) та наслідки їх застосування враховуючи зовнішні фактори [4] – наявність ключових фраз які належать до терміну для обчислення міри відповідності його приналежності до статті [68].

Побудова дерева для прийняття рішення щодо терміну полягає в наступному. Виявлені в тексті допису ключові фрази що відносяться до певного терміну з ваговими коефіцієнтами необхідно впорядкувати за спаданням ваг. Відношення вагових коефіцієнтів ключової фрази до терміну наведено в словнику. На кожному рівні дерева загальна вага ключових фраз повинна дорівнювати одиниці [68]. Обчислити значення ваги ключової фрази що не належить терміну на кожному рівні можна наступним чином:

$$\overline{w}(n) = 1 - w(n), \tag{2.39}$$

де $w(n)$ – вага ключової фрази що належить терміну.

На основі вагових значень ключових фраз побудувати дерево що зображено на рис.2.17.

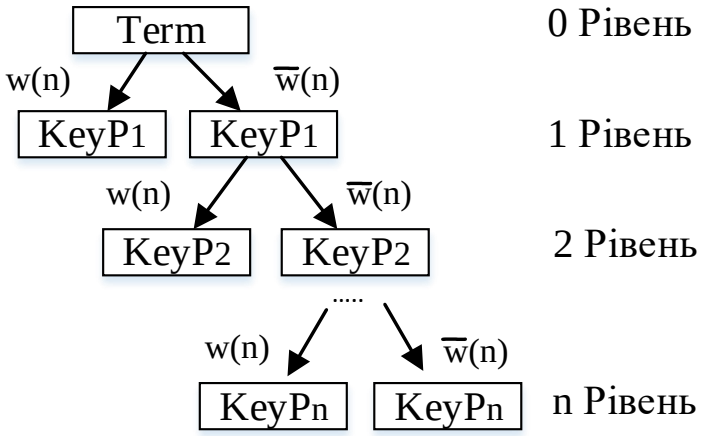


Рис.2.17 Схема побудови дерева для визначення вагового коефіцієнту терміну

Обчислення ваги терміну до тексту допису відбувається наступним чином:

$$W(Term_i) = \left(w(n) \cdot \overline{w}(n-1) + w(n-1)\right) \cdot \overline{w}(n-2) + \ \ldots, \tag{2.40}$$

Обчислення відбуватиметься доки, доти не буде досягнуто кореню дерева (нульовий рівень).

Стаття ДПЗ має форму дискусії, структура якої наведена рис.2.11. Повідомлення які знаходяться на вищому рівні є вагомішими ніж ті, які розташовані на нижчих рівнях. Тому при обчисленні ваги термінів доцільно враховувати рівень розташування повідомлення у дискусії наступним чином:

$$\mu_{validity}\big(Term, TextPart(Article)\big) = (Term_i) \times k_{ij}\,, \qquad (2.41)$$

$k_{ij} = \frac{1}{\log_2 level(Message_m)}$ – коефіцієнт m-го повідомлення,

$level(Message_i)$ – рівень розташування i-го повідомлення у статті.

Слід зазначити, що обчислення за формулою 2.41 потрібно проводити для повідомлень, які розташовані починаючи з другого рівня, бо $\log_2 1 = 0$. Тобто для першого повідомлення ($MessageDiscuss$) і повідомлень другого рівня коефіцієнт буде дорівнювати 1, а для всіх інших коефіцієнт є меншим.

Застосування такого розподілу оцінок коефіцієнтів забезпечує покриття наступної ситуації: якщо перше повідомлення є запитанням (коли користувач цікавиться чимось), а інші є відповідями на нього. Тобто перше повідомлення може не містити багато значущої інформації. Натомість повідомлення другого рівня вже може містити потрібні, важливі дані, зміст яких потребує аналізу.

Таким же чином до кожного терміну необхідно побудувати відповідні дерева та обчислити міру вагомості враховуючи коефіцієнти. Після чого буде отримано список термінів (з обчисленими мірами вагомості, значення кожної з яких знаходитиметься в інтервалі [0, 1]). Надалі потрібно відібрати значущі, вагомі терміни до вмісту статті за допомогою виконання наступної умови:

$$\mu_{validity}\big(Term, TextPart(Article)\big) \geq \alpha, \qquad (2.42)$$

де α – показник, що визначає порогове значення для оцінки міри вагомості терміну.

Наведемо приклад побудови дерева для прийняття рішення щодо терміну на основі частини тексту допису спільноти CodeProject що став статтею ДПЗ:

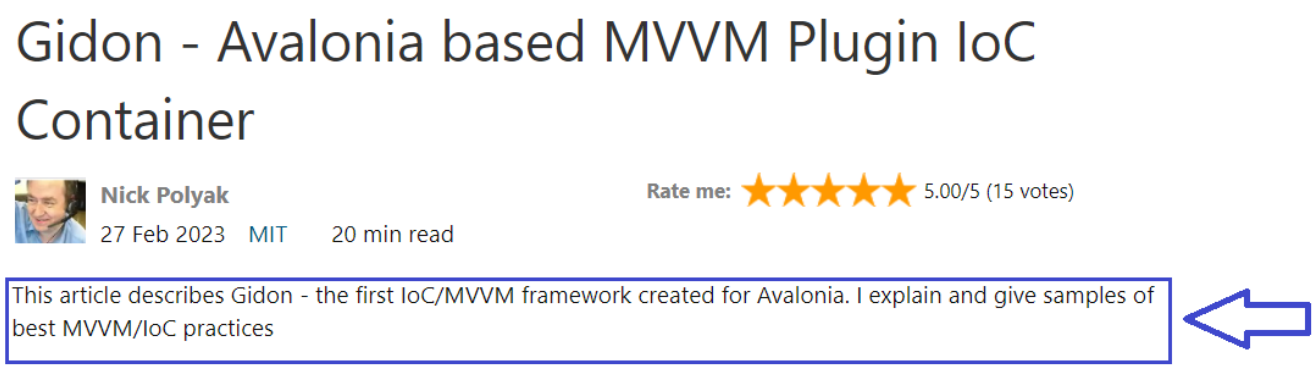


Рис.2.18. Частина тексту допису спільноти CodeProject

Наведена частина тексту повідомлення, що може бути інформаційним наповненням статті містить багато ключових фраз до різних термінів. Наприклад, термін «framework MVVM» має ключові фрази з відповідними ваговими коефіцієнтами приналежності: [«Gidon»: 0.2, «first IoC»: 0.4, «MVVM framework»: 0.7, «Avalonia»: 0.2].

Надалі, згідно схеми, що представлена на рис. 2.17, будуємо дерево для визначення ваги терміну «framework MVVM» (рис. 2.19).

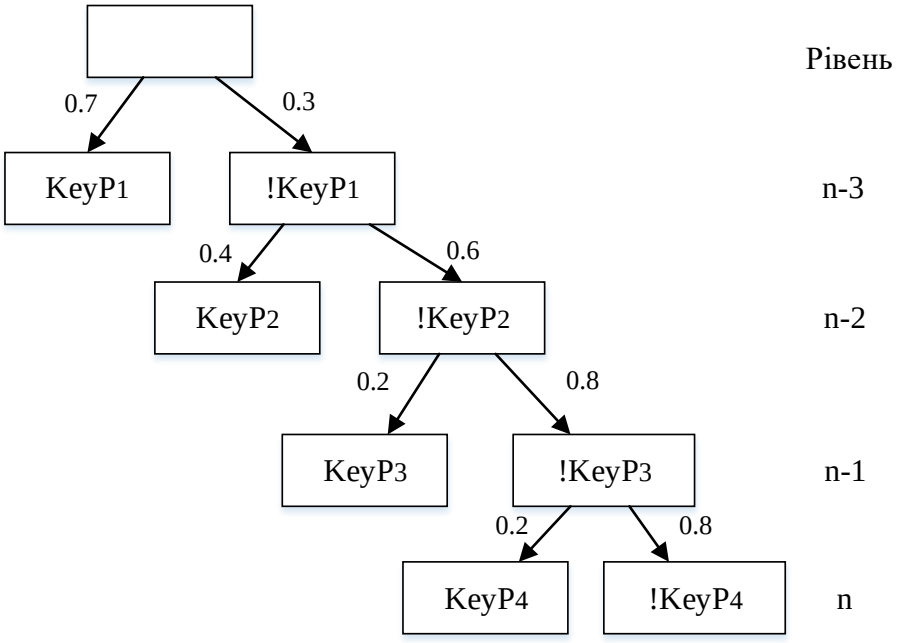


Рис.2.19 Розподіл ключових фраз терміну

Після чого необхідно обчислити міру вагомості терміну до тексту повідомлення на основі виявлених ключових фраз таким чином:

$$W(Term_1) = \begin{matrix}((w(n) \cdot \overline{w}(n-1) + w(n-1)) \cdot \overline{w}(n-2) + \\ \overline{w}(n-2)) \cdot \overline{w}(n-3) + \overline{w}(n-3)\end{matrix}, \qquad (2.43)$$

де $w(n)$ – вага ключової фрази, що розташований на n-му рівні;

$\overline{w}(n-1)$ – обчислена вага ключової фрази (за формулою 2.41), що розташована на рівні $n-1$.

Через те, що повідомлення розташоване на першому рівні тому значення коефіцієнту дорівнюватиме одиниці. Провівши обчислення терміну «framework MVVM» до тексту статті $\mu_{validity}(Term_1, Article) = 0.885$. При пороговому

значенні $\alpha > 0.75$ для оцінки міри вагомості даний термін є значущим до тексту статті $\left(Article(ValidTerm)\right)$ відповідно заносимо його у список термінів статті.

Отже, метод автоматизованого формування термінів певної статті полягає у наступному:

1. Для певного повідомлення статті здійснюється пошук ключових фраз в тексті на основі розробленого експертом словнику.
2. Виявлення відповідності термінів ключових фраз до термінів (слід зауважити, що кількість термінів може бути більшою за кількість ключових фраз, адже одна фраза може належати декільком термінам з різними ваговими коефіцієнтами).
3. Групування за термінами ключових фраз і побудова для кожного з них дерева для прийняття рішень (рис.2.17).
4. Обчислення вагового значення терміну (формула 2.41); перевірка виконання умови вагомості терміну до допису (формула 2.42); формування списку термінів, значення вагових коефіцієнтів яких задовольняють умові вагомості.
5. Застосування кроків 1-4 для всіх повідомлень статті.

В результаті виконання методу буде отримано список вагомих, значущих до змісту статті термінів з ваговими коефіцієнтами.

2.6. Метод виявлення профілів користувача

Зазвичай користувачі мережі Інтернет при реєстрації у ВС залишають особисті дані: ПІБ, стать, вік, номер телефону, країна, місце роботи і/або навчання, уподобання та інтереси тощо [114]. Особливістю більшості ВС є достовірність опублікованих даних профілю учасника до його особистих даних. Це означає, що люди реєструються під своїми власними іменами, додають особисті фото та відео матеріали, і спілкуються онлайн з іншими людьми, яких знають особисто. Розміщені особисті дані можна поділити на дві категорії:

- ті, які потрібні для виявлення автора як особи ($Creator_j$), що може бути користувачем різних ВС. Необхідними атрибутами постають [113] – ім'я, дата народження, місто народження, проживання тощо;
- ті, що необхідні для виявлення професійних навичок автора, який публікує дописи, інформаційне наповнення яких може бути корисним для формування ДПЗ. Такими даними постають – посада, досвід роботи, інтереси автора тощо.

Застосувавши метод подібності текстів дописів (розділ 2.4.1), які розміщені в одній ВС, було виявлено, що дописи з високою мірою подібності належали одному автору.

Також, спостерігається тенденція, якщо користувача цікавить якесь питання і він є зареєстрованим у декількох ВС, відповідно він публікує його у різних спільнотах, при цьому не змінюючи текст допису. Такий підхід дозволить охопити більшу аудиторію – отримати більше відповідей і оцінок від користувачів щодо розміщеного допису за короткий проміжок часу.

При виявленні подібності між текстами дописів необхідно ідентифікувати їх авторів. Якщо автор в одній спільноті опублікував два подібних дописи проводити ідентифікацію не має потреби. Натомість, якщо дописи опубліковані у різних ВС виникає потреба у визначенні автору завдяки перевірці атрибутів що йому належать. Алгоритм випадків дублювання наведено на рис. 3.4. розділ 3.2.

Так як кожна ВС має свою особливу структуру і користувачі можуть залишати різний набір даних про себе, тому необхідно збирати всю інформацію, яка є доступною за формулою 2.21. Процедура ідентифікації користувача у різних ВС полягає у наступному:

1. Збір даних що визначають профіль користувача.
2. Нормалізація даних.
3. Присвоєння вагових коефіцієнтів, в залежності від їх типу.
4. Для виявлених ознак побудувати дерево для прийняття рішень (приклад побудови наведено у розділі 2.5).

Значення вагових коефіцієнтів характеристик профілю користувача можуть бути наступними:

- подібність текстів $k_{text} = 0.5$;
- персональні фотографії $k_{persphoto} = 0.3$, інші зображення $k_{photo} = 0.1$;
- персональні дані: ім'я $k_{name} = 0.2$, $k_{sity} = 0.2$, $k_{age} = 0.1$;
- інтереси користувача та інші дані $k_{other} = 0.1$.

На основі побудованого дерева рішень обчислити міру ідентифікації ($\mu_{identity}$) профілю користувача та здійснити перевірку виконання умови:

$$\mu_{identity} \geq \alpha, \tag{2.44}$$

де α – показник, що визначає порогове значення для приналежності профілів одному користувачеві при здійсненні ідентифікації.

Рекомендованим значенням є $\mu_{identity} = 0.75$, тобто принаймні за трьома характеристиками профілів користувачів мають бути виявлений збіг даних (як за типом даних так і за значенням).

Виявлені профілі користувача за допомогою цього методу дозволить передбачити його поведінку:

- активність щодо створення інформаційного наповнення;
- креативність щодо публікування якісного (як за обсягом, так і за змістом) [62], оригінального контенту;
- отримати більше інформації щодо його інтересів, досвіду роботи завдяки більшому набору ознак;
- формування рейтингу автору статті документації ($Creator_i$).

2.6.1. Аналіз показників порогового значення оцінок

Природніми обмеженнями оцінювання будь-якого явища є інтервал [0, 1]. Для визначення порогового значення оцінки α, яка буде схвалювати рішення при виконанні умови, автором запропоновано застосувати декілька методів що зробить дане значення адаптивним для роботи системи з різними даними.

1. Застосування «високого» порогового значення на основі закону Парето (принцип «80/20») адже він є одним із найпоширеніших способів оцінювання ефективності будь-якої діяльності. Його зміст полягає у тому, що всього 20% зусиль дають 80% результату, а інші 80% зусиль — лише 20% результату [96].
2. Застосування «середнього» показника ¾=0.75. Принцип його полягає у тому, що для виконання певної умови, значення оцінки будь-якого показника має бути 0.75 і вище. Якщо значення є нижчим умова вважається не виконаною.
3. Використання «низького» показника 2/3=0.66. Суть є тією ж самою як і в пункті 2, але порогове значення оцінки для виконання умови має бути 0.66 і вище.

Застосування «високого» порогового значення можна застосувати, наприклад, для оцінки авторитетності дискусій, що містяться в темі. Відповідно дискусії з високим рейтингом (від 0.8 до 1 що становить 20% із всього інформаційного наповнення теми) містять 80% корисної інформації, а дискусії з меншим рейтингом містять відповідно 20% потрібного контенту.

Цей самий принцип можна віднести і до авторів дискусій – автори з високим рейтингом (від 0,8 до 1 що становить 20% із всього інформаційного наповнення) публікують 80% корисної інформації, і, автори з меншим рейтингом публікують всього 20% контенту, що може бути розміщений у документації.

Отже, застосування різних значень порогової оцінки α для обчислення релевантності, авторитетності, подібності тощо, робить систему універсальною та адаптивною до різних ситуацій, що можуть виникнути при роботі з даними.

Висновки до розділу

У другому розділі побудовано формальні моделі, які є основою для подальших досліджень щодо формування методів та алгоритмів для роботи з даними. Зокрема, отримано такі результати:

- побудовано формальну модель сховища консолідованих даних на основі понятійного апарату віртуальних спільнот, яка містить чотири складові (джерела – спільноти, теми, дописи та користувачів), що дозволило завантажувати та зберігати дані з віртуальних спільнот в єдиному місці;
- побудовано формальну модель документації програмного забезпечення, яка враховує особливості інформаційного наповнення віртуальних спільнот та потреби різних категорій користувачів програмного забезпечення через множину класифікаторів, що дозволило побудувати методи для її формування;
- наведено терміни кожного класифікатору на основі попередньо розроблених словників, що дозволило побудувати структуру документації програмного забезпечення;
- побудовано метод автоматизованого виявлення вагомих, значущих термінів статей на основі масок ключових фраз з ваговими коефіцієнтами з використанням дерева для прийняття рішень, що забезпечило заповненню інформаційного наповнення документації статтями;
- розроблено сценарії обчислення рейтингу допису, на основі наявних реакцій користувачів у віртуальних спільнотах, що дозволило визначити рівень довіри до розміщених даних у дописах спільнот;
- описано поєднання методу виявлення подібності та моделі N-грам (без вилучення і з вилученням незначущих, стоп-слів) для виявлення однакового тексту в дописах, що забезпечило формування якісної документації програмного забезпечення;
- описано метод виявлення профілів користувача на основі даних розміщених ним у віртуальних спільнотах, що дозволило виявити унікальних авторів інформаційного наповнення документації;
- наведено значення показників порогових оцінок за допомогою показників які застосовують при розрахунках, що дозволило забезпечити універсальність та адаптивність системи до різних ситуацій.

Розділ 3. Розробка алгоритмів виявлення цінного інформаційного наповнення для формування документації програмного забезпечення

Процес формування документації програмного забезпечення є складним та потребує: залучення трудових, часових і фінансових ресурсів; використання різноманітних технік пошуку та відбору джерел, що можуть містити цінне інформаційне наповнення; проведення логічного аналізу і обробки великих обсягів даних для подальшого їх приведення до єдиного представлення кінцевому користувачеві. Тому виникає необхідність у створенні відповідних програмних компонент для роботи та підтримки програмно-алгоритмічного комплексу формування супровідної документації програмного забезпечення, яка широко використовується в IT сфері.

В даному розділі наведено методи та алгоритми, що потрібні для реалізації наступних програмних компонент: відбору адекватних джерел даних, визначення та завантаження авторитетних дискусій, виявлення дублювання інформаційного наповнення, усунення небажаного контенту, формування розділів для відображення документації програмного забезпечення як готового інформаційного продукту.

Цінність інформаційного наповнення характеризується його значимістю для споживачів — дані повинні міститися у адекватному джерелі, бути актуальними і достовірними.

Основні результати розділу опубліковані автором у працях [65].

3.1. Розроблення алгоритму визначення адекватності тем віртуальних спільнот як джерела цінної інформації

Адекватність є широким терміном, який застосовують у багатьох галузях наук (філософії, психології, правознавстві, математиці тощо). У теорії ймовірності адекватність вважають відповідністю фактичних результатів

(наприклад, результати дослідження), результатам, отриманим за допомогою розрахунків. У техніці — відповідність певних основних характеристик моделі характеристикам об’єкта, що моделюється [1].

Адекватним джерелом інформації для подальшого формування документації, можна вважати таке джерело, інформаційне наповнення якого задовольняє вимоги та потреби споживачів, а також містить достовірну (перевірену іншими учасниками спільноти) і актуальну інформацію.

Як було зазначено у розділі 1.2 інформаційне наповнення ВС є структурованим за темами (тематичними ситуаціями), кожна з яких містить певний набір ключових слів (що і характеризують її) та множину дискусій. Тому кожну тему можна представити як логічну одиницю джерела інформації, яка потребує детального розгляду щодо її адекватності для подальшого виявлення та завантаження авторитетних дискусій.

Адекватна тема повинна задовольняти наступним вимогам: релевантність і актуальність.

Релевантність – міра відповідності отриманого результату бажаному [60]. Релевантність пошуку – міра відповідності результатів пошуку щодо завдання, яке було поставлено в пошук [75]. Відомо, що кожна пошукова система задає свої пріоритети пошуку за індексами. Тому один і той самий запит в різних системах відображає різні результати.

Показник *релевантності* ($\mu_{relevance}$) визначає чи містять теми ВС інформацію до певного ПЗ за допомогою визначення міри відповідності запиту користувача до множини ключових слів, що характеризують тему. Природніми обмеженнями для $\mu_{relevance}$ є інтервал $[0, 1]$. Джерело можна вважати релевантним якщо виконується наступний критерій:

$$\mu_{relevance} \geq \alpha, \tag{3.1}$$

де α – показник, що визначає порогове значення для оцінки міри релевантності джерела.

Актуальність – суб’єктивна цінність, яку люди надають інформації, якщо вона відповідає їхнім потребам та інтересам [5]. Актуальність даних — властивість, яка відображає їхню значимість в сучасній дійсності [2].

Відповідно актуальне повідомлення в межах дискусії спільноти – це допис, що містить актуальні дані. Актуальність дискусії як елемент теми, визначається наявністю актуальних дописів в ній. Показник *актуальності* тем (μ_{actual}) визначається рівнем приросту інформаційного наповнення за певний проміжок часу. Природніми обмеженнями для μ_{actual} є інтервал [0, 1]. Джерело можна вважати авторитетним якщо виконується наступний критерій:

$$\mu_{actual}(\mathrm{A}) \geq \beta, \tag{3.2}$$

де β – показник, що визначає порогове значення для оцінки міри актуальності інформаційного наповнення в темі.

Наведені вимоги, які визначають адекватність теми як джерела даних, виконуються послідовно. Якщо, при перевірці, умова релевантності не є виконаною, відповідно вимога актуальності не буде розглянута і джерело вважатимемо не адекватним для формування інформаційного наповнення ДПЗ.

Покрокове виконання алгоритму визначення адекватності джерела (теми) представлено за допомогою UML-діаграми дій (рис.3.1).

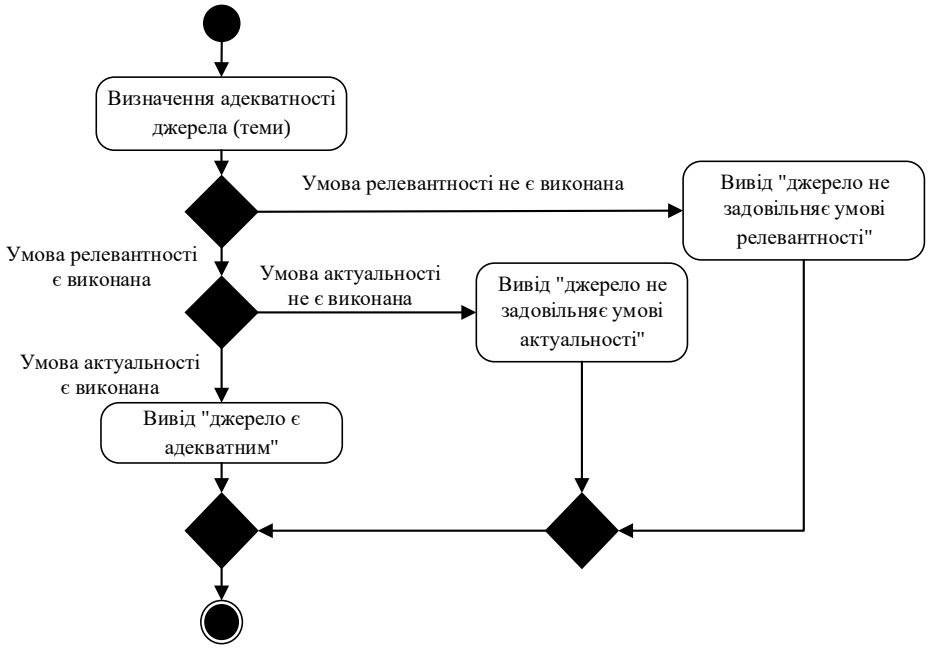


Рис.3.1 UML-діаграма дій щодо визначення адекватності джерела

Результатом проходження алгоритму є сформована БД адекватних тем для подальшого аналізу даних в них. При цьому теми отримують наступні мітки:

- «Адекватна» – для тем які задовольняють вимогам релевантності та актуальності;
- «Неадекватна» – якщо не виконана одна або дві вимоги адекватності.

3.1.1. Визначення міри релевантності тем віртуальних спільнот

Зазвичай, релевантність визначається статистичними методами досліджень основними функціями яких є збір, зведення, групування, узагальнення, аналіз та оцінка даних.

Проаналізувавши критерії підвищення релевантності пошуку інформаційного наповнення [111] з ВС було виділено наступні:

- назва спільноти, теми та допису повинна містити ключові слова та логічну структуру щодо назви ПЗ;
- відсутність помилок в тексті для коректного індексування текстових частин сайту;
- наявність заповнених мета тегів, що можуть містити корисні дані (Title, Description, Keywords, заголовки), ключових слів в URL адресі та у внутрішніх посиланнях на сторінку ВС тощо;
- унікальність вмісту дописів;
- час завантаження сторінок спільноти;
- візуальне представлення інформаційного наповнення ВС (розмір тексту).

Релевантна тема – тема, що розміщена у ВС та з певною мірою відповідності містить інформаційне наповнення відповідно до запиту користувача у вигляді множини дискусій.

Для виконання процедури відбору і подальшого оцінювання тем ВС експерт попередньо повинен сформувати та наповнити БД відповідними атрибутами, отримавши які можна здійснити процедуру автоматизованого збору

даних. Після чого необхідно відібрати саме ті теми, що можуть містити інформацію до конкретного ПЗ.

Відомо, що ВС, які містять інформаційне наповнення до певного ПЗ вказують ці дані в назві спільноти (якщо це джерело повністю орієнтовано на вказане ПЗ), або в темі(-ах). Якщо назва ПЗ не згадується в темах спільноти то вважатимемо, що таке джерело не містить інформації до нього. Для співставлення назв автором введено поняття міри релевантності назви ПЗ до назв тем ВС $\mu_{relevance}(SW(Title), Theme(Title))$.

Для визначення міри релевантності попередньо необхідно нормалізувати назви тем та сформувати словник з показниками IDF, що визначає коефіцієнти слів. При наявності словника обчислення міри релевантності до запиту користувача полягає у наступному:

1. Нормалізація запиту (розділ 2.4.1).
2. Застосування показника частотності слів TF-IDF (опис якого наведено у розділі 2.4.2) для обчислення міри релевантності.

В результаті буде отримано вектор релевантності між назвами тем та запитом, нормалізовані значення якого знаходитимуться в інтервалі [0, 1].

Даний алгоритм може застосувати як експерт, при відборі релевантних джерел даних, так і користувач, при пошуку релевантних тем для формування ДПЗ відповідно до запиту.

Пошуковий запит користувача для генерування ДПЗ може бути таких видів:

- звичайний запит;
- доповнений, розширений запит.

Звичайний запит відповідає запиту пошукових системам, що основані на релевантності ключових слів, які задав користувач.

Розширений запит, передбачає, що ключових слів, які задав користувач може бути недостатньо для відбору тем, тому його потрібно доповнити словами (зі словнику термінів), що мають високий ваговий коефіцієнт відповідності.

Тому виникає потреба у об’єднанні запиту користувача та відповідних вагомих, значущих термінів (що задовольняють умові наведеній у формулі 2.42) зі словника наступним чином у вираз ($Expression$), що містить множину значущих, унікальних слів:

$$Expression = Query \cup ValidTerm, \tag{3.3}$$

де $Query$ – множина слів що є в запиті користувача;

$ValidTerm$ – множина вагомих термінів до запиту користувача.

Даний підхід дозволить охопити більшу кількість тем.

Алгоритм визначення релевантності тем до розширеного запиту користувача наведено на рис.3.2.

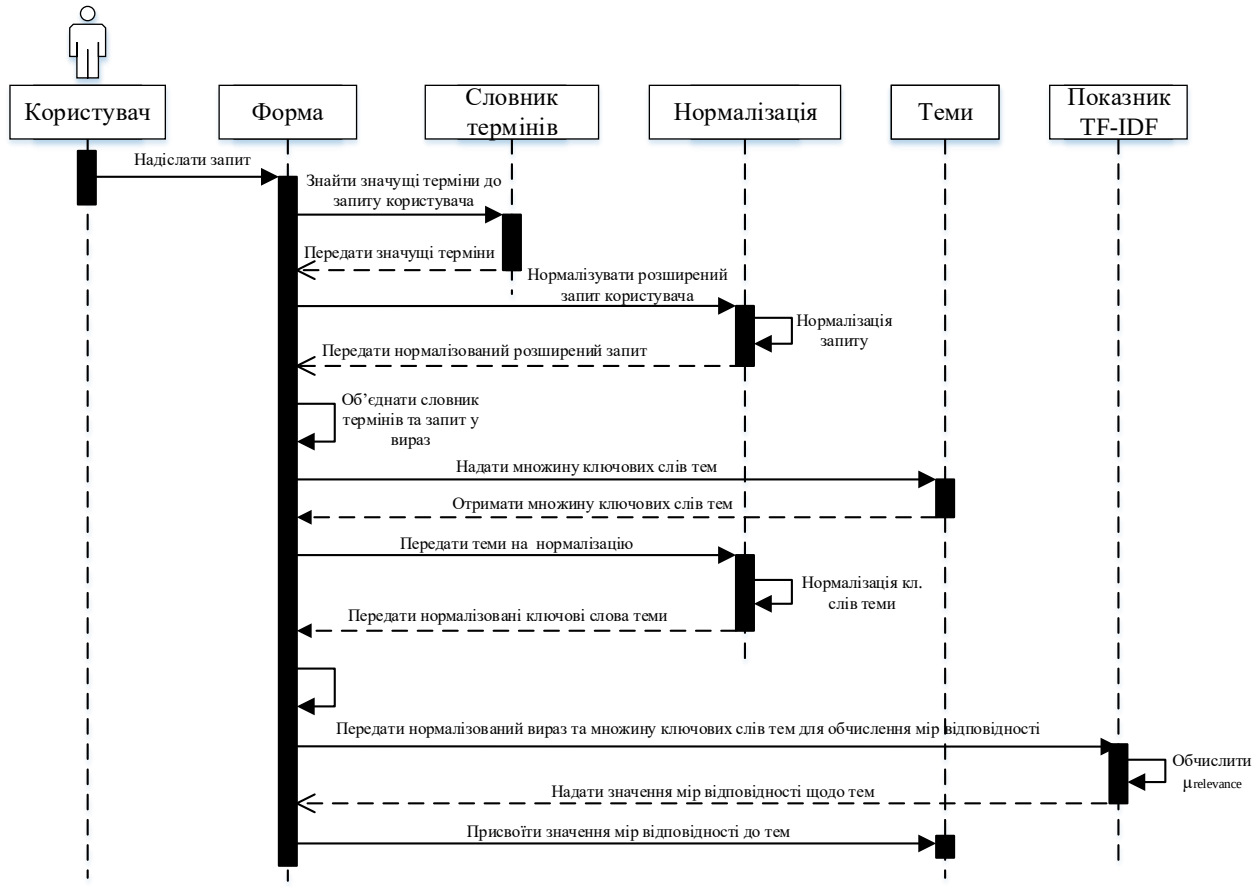


Рис.3.2 UML-діаграма послідовності визначення релевантності тем до розширеного запиту користувача

Алгоритм визначення релевантності тем до розширеного запиту користувача полягає у виконанні наступних кроків:

1. Розширення запиту користувача за допомогою словника термінів.

2. Нормалізація запиту користувача.
3. Нормалізація назв тем та формування словнику з показниками IDF.
4. Застосування показника TF-IDF розширеного запиту користувача до тем.

Для звичайного запиту користувача виконання алгоритму визначення релевантності тем починається з другого пункту.

Надалі, після того як було визначено значення мір релевантності тем $\mu_{relevance}(User(Query), VC(Theme))$, потрібно перевірити виконання умови, наведеної у формулі 3.1. Теми, які задовольняють показнику порогового значення оцінки релевантності стають джерелами для формування ДПЗ. В іншому випадку вважатимемо що джерело (тема) не є релевантним.

Процедуру перевірки тем на виконання умови релевантності наведено на рис.3.3.

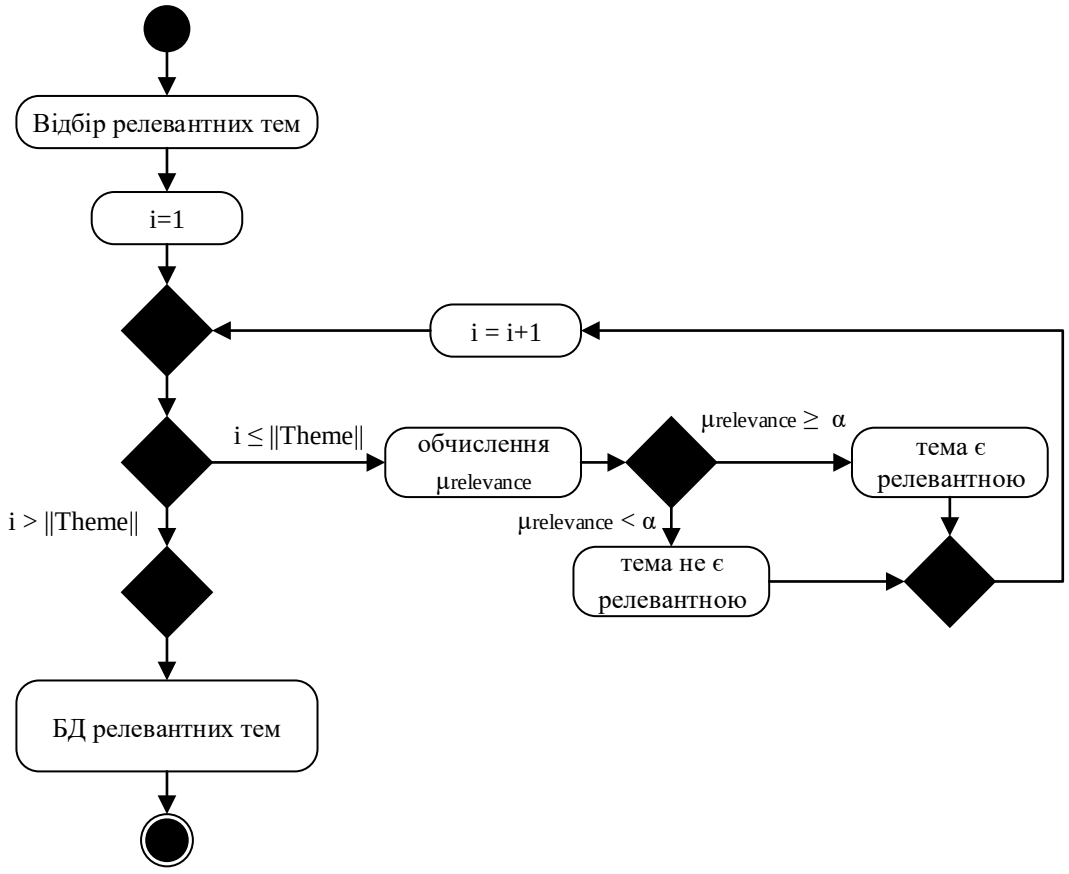


Рис.3.3 UML діаграма дій щодо відбору релевантних тем ВС відповідно до запиту користувача

Отже показник релевантності є досліджений. Надалі, релевантні теми потрібно проаналізувати за показником актуальності інформаційного наповнення. Адже саме він впливає на подальше рішення щодо призначення темі мітки «Адекватна».

3.1.2. Визначення показника актуальності тем віртуальних спільнот

Для обчислення рівня актуальності необхідно застосувати поняття активності (А), що визначає темп приросту інформаційного наповнення в певній темі. Звідси випливає наступне твердження: чим більше дописів було створено за визначений термін часу (Т) у певній темі, тим більше є значення рівня актуальності, природніми обмеженнями якого становить інтервал [0, 1].

Для обчислення активності теми застосуємо логарифмічну функцію, що має наступні переваги [34, 41]:

1. Множина визначень логарифмічної функції становить $D = (0, \infty)$, а діапазон отримуваних результатів для теми знаходитиметься в природніх обмеженнях [0, 1].
2. Логарифмічна функція є монотонною, та зростаючою (для обчислення активності) при $b > 1$ (приріст повідомлень в темі має бути більше ніж на одне повідомлення):
    2.1. якщо $b = 1$ – приріст в темі на одне повідомлення, що не є хорошим показником для ствердження активності. Відповідно тему вважатимемо не актуальною;
    2.2. якщо $b = 0$, тобто інформаційного приросту за час порівнювального періоду не відбулося або тема є новою і не має на даний момент часу дописів відповідно логарифм буде невизначеним (що не відповідає умові обчислення логарифму). Таку тему до уваги не беремо.
3. Пропорційність застосування логарифмічні функції за різними основами.
4. Простота при роботі з великою кількістю даних (повідомлень), оскільки функція логарифму зростає повільніше зі збільшенням чисел.

Враховуючи вище наведені умови застосування логарифму автором розроблено формулу обчислення активності теми із застосуванням натурального логарифму:

$$A = \frac{ln(y_i)}{ln(y_n)}, \quad (3.4)$$

де $ln(y_i)$ – логарифмічне значення кількості повідомлень поточного рівня (порівнюваного періоду);

$ln(y_n)$ – логарифмічне значення загальної кількості повідомлень.

Тема є актуальною якщо виконується умова наведена формулою 3.2.

Отже, застосування даного методу дозволить усунути незначущі теми, що мають малий приріст щодо кількості повідомлень за визначений період часу, а також позбутися від джерел, які не мають дописів (нові теми) або мають тільки один допис.

3.1.3 Визначення актуальних дискусій в темі

Теми ВС містять дискусії, що опубліковані у різний час. Важливим елементом сфери ІТ є наявність актуальних даних. Актуальність розміщених даних можна перевіряти за допомогою аналізу текстової частини допису, та коментарів до нього, а також перевіряти дату розміщення цих публікацій, адже актуальність – це властивість інформації, що може бути втрачена з часом [33]. Актуальність дискусій, можна визначати часом опублікування або редагування допису-дискусії та дописів-коментарів. Допис-дискусія визначає час започаткування дискусії. Допис-коментар характеризує, що дане обговорення є активним і актуальним на час його розміщення.

Нехай $PostDiscTime(Post_i)$ – час публікування/редагування допису, що розпочинає дискурс, а $PostComTime(Post_i)$ – час публікування/редагування допису-коментаря в дискусії. При чому дата розміщення $PostDiscTime(Post_i)$ є старшою від дати опублікування $PostComTime(Post_i)$. Дата редагування допису-дискусії може бути як раніше так і пізніше від дати публікування або редагування допису-коментаря.

Початкове значення моменту часу щодо відліку актуальності для дописів у дискусії визначатимемо:

$$T_0 = T_{now} - T_{period}, \tag{3.5}$$

де T_{now} – теперішнє значення моменту часу;

T_{period} – період актуальності повідомлень.

Отже, дискусія є актуальною, якщо час публікування/редагування допису-дискусії чи дописів-коментарів є меншим за T_0 значення:

$$Actual(Disc) = \begin{cases} T_0 < PostDiscTime(Post) \ \ OR \\ T_0 < PostComTime(Post) \end{cases}, \tag{3.6}$$

де T_0 – початкове значення моменту часу щодо відліку актуальності для дописів; $PostDiscTime(Post)$ – момент часу публікування/редагування допису-дискусії; $PostComTime(Post)$ – момент часу публікування/редагування допису-коментаря.

Розміщення дописів-коментарів відображає актуальність дискусії.

Отже, якщо допис-дискусія або допис-коментар, що належить цій дискусії мають пізнішу дату публікації ніж значення T_0, тоді дана дискусія є актуальною.

3.1.4. Визначення авторитетних дискусій в темі

Формування ДПЗ потребує перевірки не тільки актуальності за часом публікування/редагування розміщених даних, але також у визначенні авторитетності дискусій, що знаходяться у адекватних темах.

Показник *авторитетності* ($\mu_{authority}$) визначає рівень довіри до інформаційного наповнення, що розміщене в дискусії, на основі середньої оцінки рейтингів дописів що належать їй. До уваги беруться дискусії лише тих тем, які задовольняють умову адекватності. Природніми обмеженнями для $\mu_{authority}$ є інтервал [0,1].

Середню оцінку рейтингу допису *i*-тої дискусії обчислюватимемо так:

$$PostRate(Disc_i) = \frac{\sum_{j=1}^{CountOfPost(Disc_i)} Rate(Post_{j_i})}{CountOfPost(Disc_i)}, \tag{3.7}$$

де $CountOfPost(Disc_i)$ – кількість дописів в *i*-тій дискусії;

$Rate(Post_{j_i})$ – рейтинг j-го допису, що знаходиться в i-тій дискусії.

Дискусія є авторитетною якщо виконується наступна умова:

$$\mu_{authority}(PostRate(Disc_i)) \geq \alpha, \quad (3.8)$$

де α – показник, що визначає порогове значення для оцінки міри авторитетності дискусії.

Ті дискусії, що задовольняють умові авторитетності можуть бути статтями ДПЗ, а, відповідно, інші дискусії не будуть розглянуті.

3.2. Розроблення алгоритму усунення дублювання інформаційного наповнення в документації

Однією з властивостей інформації є її компактність, стислість представлення [45]. Для забезпечення цієї властивості інформація в документації повинна містити унікальне інформаційне наповнення. Для того щоб статті ДПЗ не містили однакового вмісту потрібно розробити спосіб усунення дублювання даних, що ґрунтується на послідовному застосуванні алгоритмів виявлення однакових ідентифікаторів та вмісту серед: завантажених дописів, дописів і наявних статей. При виявленні однакового або подібного інформаційного наповнення дописів провести процедуру об’єднання даних.

Дублювання публікацій може виникнути в таких випадках:

1. У спільноті різні автори опублікували одну і ту ж саму інформацію (наприклад, один автор скопіював вміст допису іншого або два автори взяли інформаційне наповнення якогось сайту і розмістили ці дані);
2. У спільноті один автор опублікував декілька раз однакові дані або розміщений допис може відноситися та знаходитися в різних розділах спільноти (наприклад мати декілька тегів – для мережевої та гібридної структур ВС).
3. В різних спільнотах автор опублікував однакову інформацію.
4. В декількох спільнотах різні автори опублікували одні і ті ж самі дані (наприклад, отримавши їх з якогось джерела).

Випадки дублювання даних у ВС відображено UML діаграмою дій рис.3.4.

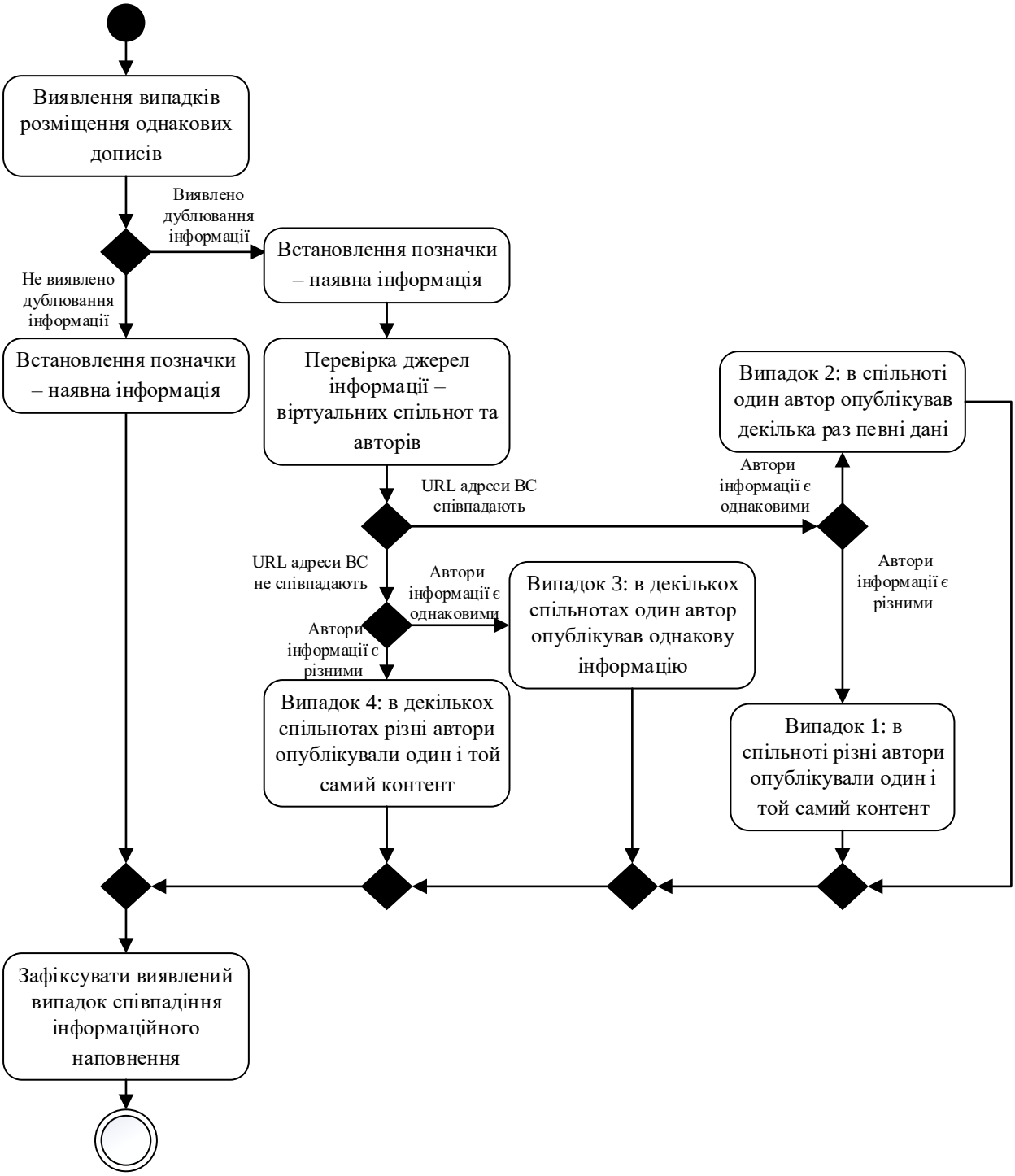


Рис.3.4 UML діаграма дій – випадки дублювання публікацій у ВС

Для уникнення дублювання інформаційного наповнення в ДПЗ дописи можна перевіряти за унікальним номером (ідентифікатором) та аналізуючи вміст за допомогою методів виявлення подібності (розділ 2.4.3) і моделлю N-грам (розділ 2.4.4). В цьому розділі наведено алгоритми перевірки унікальності дописів за ідентифікаторами при завантаженні та оновленні даних:

1. Серед дописів тем ВС та СКД.
2. Серед перевірених унікальних дописів СКД та статей ДПЗ.

3.2.1. Алгоритм визначення унікальних дописів для подальшого їх завантаження

При повторному завантаженні дописів у СКД виникає потреба у їх перевірці за атрибутами – URL адреса та унікальний номер, і, відповідно встановленні відповідних позначок (міток) для подальшої роботи з ними:

- «новий» допис, який раніше не завантажували;
- наявний допис:
    - «редагований» – зміст допису було змінено після його завантаження у систему;
    - «існуючий» допис, який залишився без змін після останнього його завантаження;
    - «недопустимий» – для дописів зміст яких містить недопустимі дані для ДПЗ, що були виявлені при попередній перевірці.

На основі цього було побудовано алгоритм представлений на рис.3.5, який відображає покрокову перевірку дописів в циклі. Результатом проходження алгоритму є встановлення міток для кожного допису, що відображають вказівки для подальших дій.

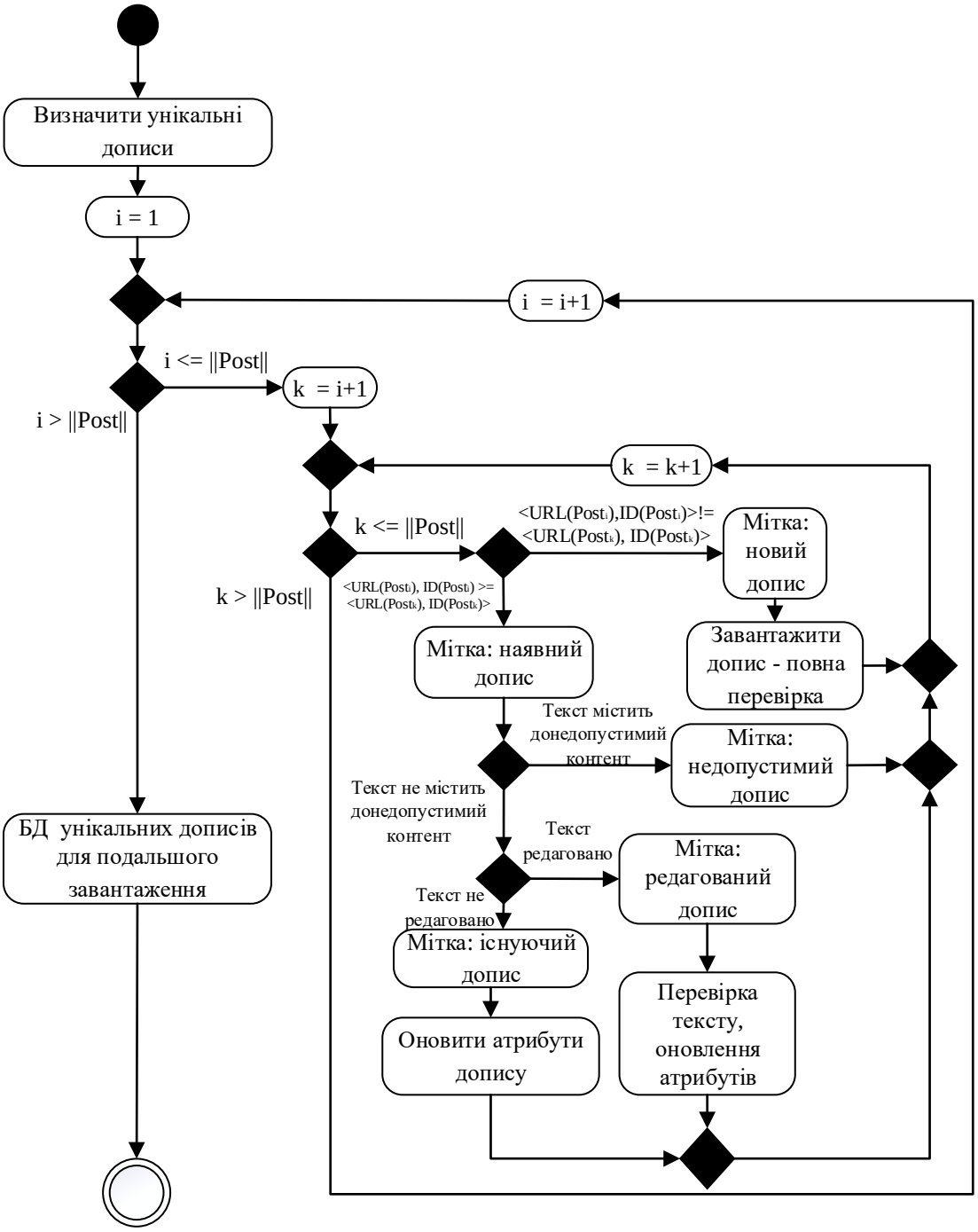


Рис.3.5 UML діаграма дій – визначення та завантаження унікальних дописів

Значення «Оновити атрибути допису» передбачає, що допис вже є у СКД і потребує оновлення рейтингу допису (сценарії обчислення наведено у розділі 2.3), рейтингу автору та дати редагування.

Перевірка текстової частини передбачає виконання наступного:

- аналіз тексту на унікальність за методом наведеним у розділі 2.4;
- перевірка щодо наявності НІН (розділ 3.3).

Значення «Перевірка тексту, оновлення атрибутів» передбачає перевірку текстової частини, оновлення атрибутів (рейтингу допису, дати редагування тощо). При виявленні НІН потрібно залишити попередній вміст допису.

Значення «Завантажити допис – повна перевірка» передбачає здійснення повної перевірки допису – текстової частини, визначення рейтингу допису, запис автору та формування його рейтингу тощо. У випадку виявлення НІН зафіксувати цю інформацію у вигляді проставлення відповідної мітки – «Недопустимий».

Результатом реалізації даного алгоритму є завантажені нові та оновлені існуючі дописи з ВС у СКД.

3.2.2. Алгоритм перевірки унікальності вмісту серед завантажених дописів

Дописи, що мають різні ідентифікатори та URL адреси потребують перевірки текстової частини щодо наявності унікального інформаційного наповнення. Адже користувачі мережі Інтернет можуть розміщувати дописи однакового вмісту як в одній так і в декількох ВС. Так як ДПЗ повинна мати статті, що містять унікальний контент відповідно після завантаження необхідно здійснювати перевірку вмісту всіх дописів що є у СКД. Для виявлення дописів, що мають схожий або однаковий вміст, застосовано метод виявлення подібності (розділ 2.4.3) та модель N-грам (розділ 2.4.4).

Алгоритм усунення дублів серед завантажених дописів полягає у виконанні послідовності таких кроків:

1. Здійснюється перевірка текстової частини множини дописів з використанням методу виявлення подібності:
    - якщо значення подібності текстів задовольняє умові наведеній формулою 2.37 тоді тексти дописів вважатимемо унікальними;
    - дописи для яких умова 2.37 не виконується, тобто значення міри подібності між їх текстами є великою, тоді застосовуємо модель N-грам. Вважатимемо, що

дописи мають унікальні тексти при виконанні умови 2.38. В іншому випадку необхідно застосувати алгоритм об’єднання дописів, що описаний у розділі 3.2.3.

2. Алгоритм виконується доки, доти не будуть перевірені всі дописи.

Результатом виконання алгоритму будуть відібрані унікальні за вмістом дописи які зберігаються у СКД та проставлені відповідні мітки:

- «унікальний» допис;
- «об’єднаний» допис.

3.2.3. Алгоритм об’єднання дописів що мають однакове інформаційне наповнення

При виявленні дописів, що містять однакове інформаційне наповнення або міра унікальності яких приймає низьке значення (не задовольняє вище наведеному критерію за формулою 2.37), виникає потреба у проведенні процедури їх об’єднання.

Алгоритм об’єднання дописів, в текстових частинах яких виявлено однакове інформаційне наповнення відображено на рис.3.6. Слід зазначити, що перед виконанням даного алгоритму всі дописи-коментарі мають бути перевірені щодо наявності НІН (розділ 3.3). Якщо допис-коментар містить НІН йому потрібно призначити мітку «Недопустимий» та вилучити за алгоритмом, наведеним у розділі 3.3.1.

Виконання алгоритму представлено за допомогою UML-діаграми дій на рис.3.6.

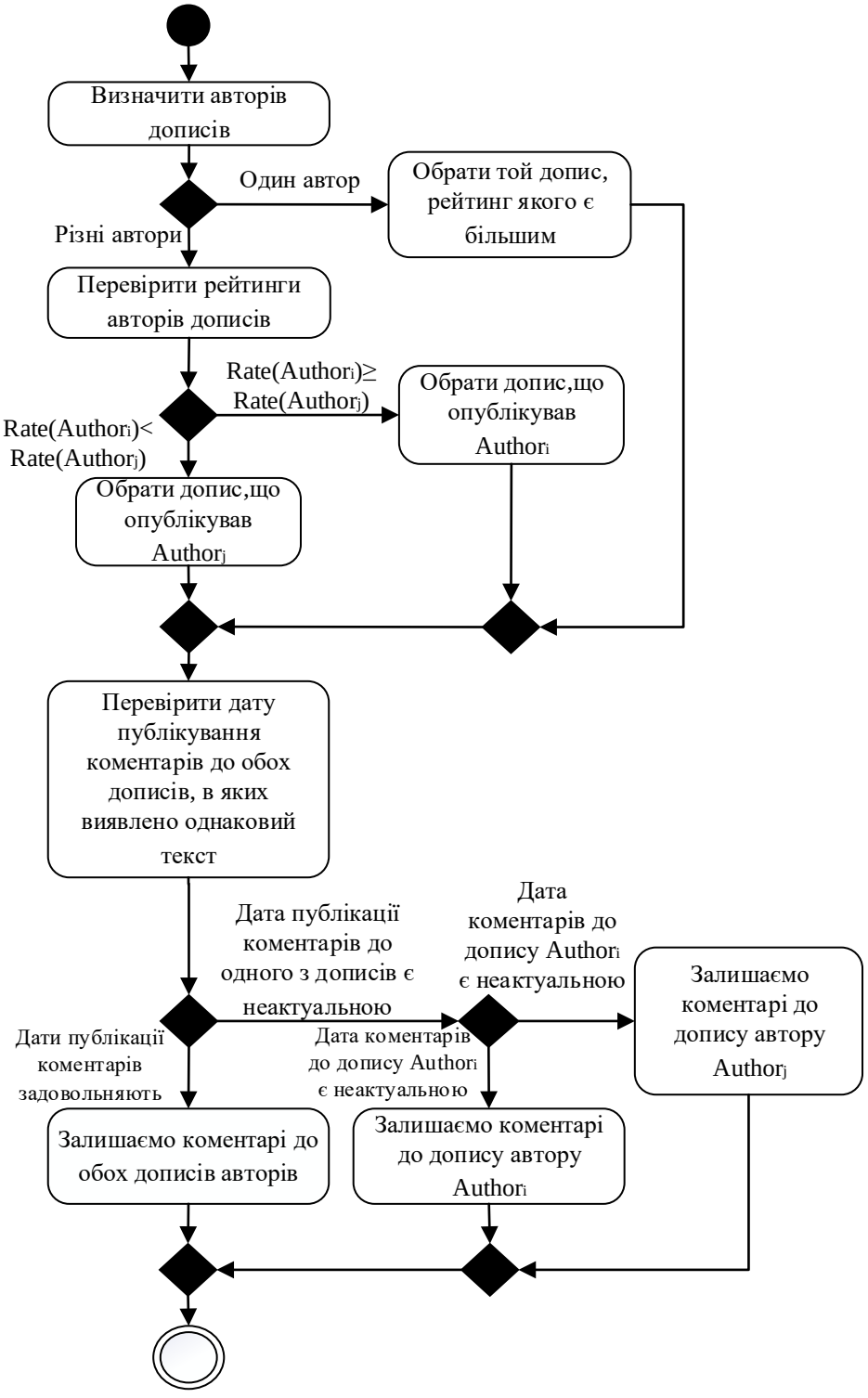


Рис.3.6 UML діаграма дій – об’єднання дописів, що мають однакове інформаційне наповнення

Алгоритм об’єднання дописів що мають однакове інформаційне наповнення полягає у наступному:

1. Визначення чи належать дописи одному автору:

- якщо дописи розміщені одним автором, тоді обрати той допис, що має більший рейтинг;
- якщо дописи розміщені різними авторами, тоді здійснити перевірку рейтингів авторів. Надалі обрати допису рейтинг автору якого є більшим.

2. Перевірка коментарів дописів за датою публікації.
   - якщо дати публікацій коментарів обох дописів є актуальними, тоді зберігаємо дві гілки множин дописів-коментарів;
   - збереження однієї гілки коментарів, дата розміщення яких є актуальною.

Результатом виконання наведеного алгоритму є забезпечення унікальності вмісту дописів що є у СКД, завдяки об'єднанню тих дописів, які містять однакові дані в текстовій частині.

3.2.4. Алгоритм наповнення документації програмного забезпечення статтями

При оновленні інформаційного наповнення ДПЗ виникає потреба у перевірці тих статей, що вже є в документації і тих які можуть їми стати, для забезпечення актуальності наявної інформації та уникнення дублювання даних.

Для завантаження дописів з СКД у ДПЗ передбачається, що вони задовольняють наступним вимогам:

- дописи отримано із адекватних тем;
- дописи знаходяться в актуальних (розділ 3.1.3) та авторитетних (розділ 3.1.4) дискусіях;
- дописи містять унікальне інформаційне наповнення (розділ 3.2.2);
- дописи не містять НІН (розділ 3.3.).

ДПЗ містить статті ($Article$) які відповідають дописам ($Post$) у СКД – $Post(Article)$.

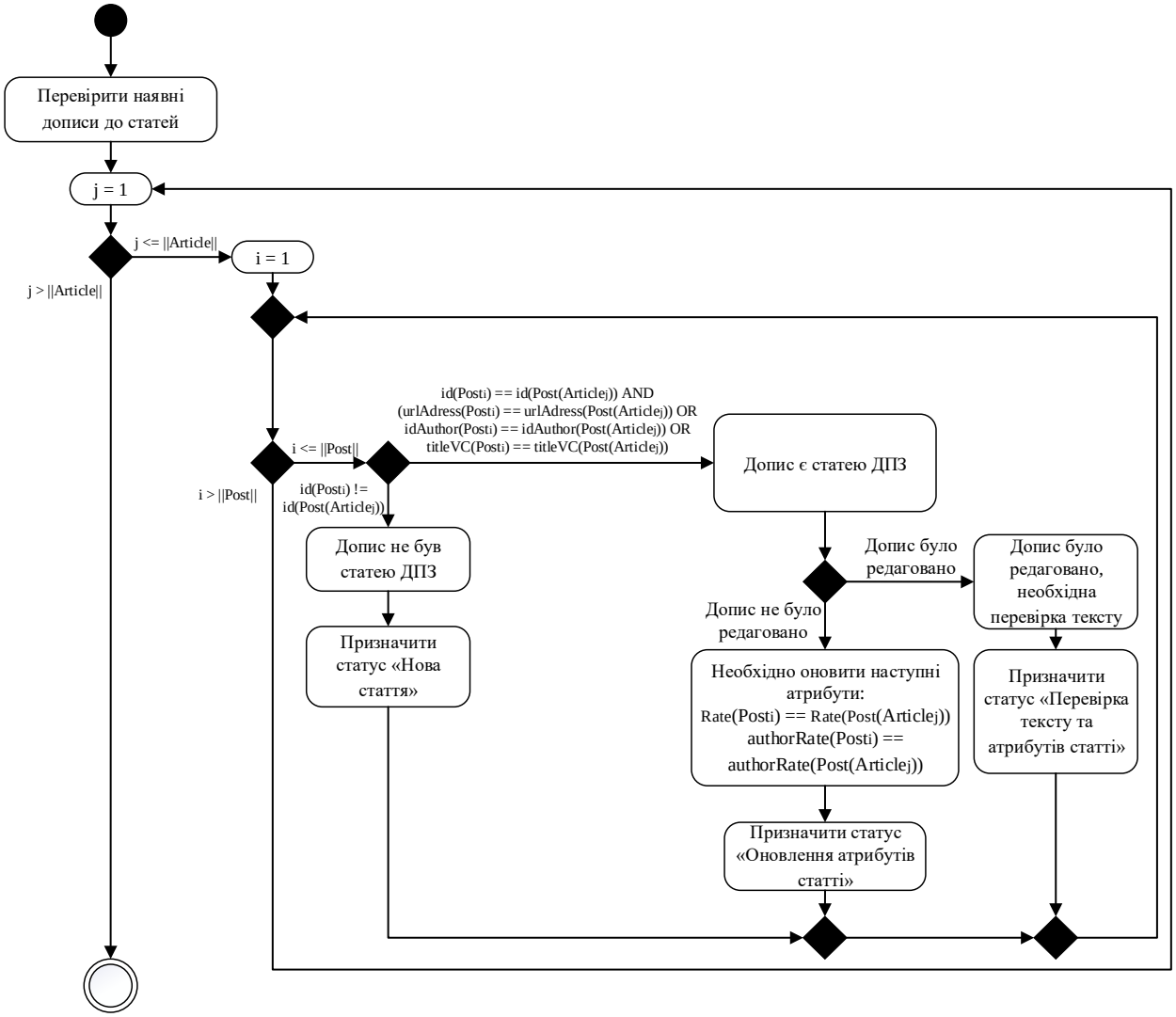


Рис.3.7 UML діаграма діяльності доповнення та оновлення ДПЗ статтями

Алгоритм перевірки статей ДПЗ і дописів СКД полягає у виконанні послідовності таких кроків:

1. Перевірка атрибутів, за якими можна визначити відповідність статті допису: ідентифікатор і URL адреса. Замість останнього можуть бути такі атрибути – ідентифікатор автору або назва спільноти .
2. При виявленні збігу можна вважати, що допис є статтею ДПЗ та потребує перевірки атрибутів, які могли зазнати змін: редагування текстової частини і її дата зміни, рейтинг статті, рейтинг автору статті.
    - Якщо виявлено редагування текстової частини допису відповідно потрібно перевіри його вміст та призначити статус «Перевірка тексту та оновлення атрибутів статті».

- Якщо допис не був редагований, тоді призначити статус «Оновлення атрибутів статті»

3. Якщо збігу не знайдено вважатимемо, що допис раніше не був статтею ДПЗ тоді призначити статус «Нова стаття», який потребує повної перевірки.

Застосування даного алгоритму допоможе зекономити часові та системні ресурси при оновленні та доповненні ДПЗ новими статтями.

Результатом проходження даного алгоритму є відібрані дописи що отримали наступні статуси:

1. «Перевірка тексту та оновлення атрибутів статті» передбачає перегляд/оновлення термінів (розділ 2.5), оновлення рейтингу статті та автору, час редагування допису;
2. «Оновлення атрибутів статті» щодо рейтингу статті та рейтингу автору що опублікував її;
3. «Нова стаття», яка не була раніше у ДПЗ потребує визначення термінів, для встановлення приналежності до розділів ДПЗ; занесення або оновлення у БД інформації щодо автору статті та його рейтингу, рейтингу статті тощо.

3.3. Розроблення алгоритмів визначення та усунення небажаного інформаційного наповнення в дописах

На сьогодні доволі часто можна стикнутися із проблемою небажаного інформаційного наповнення (НІН) у ВС: наявність ненормативної/нецензурної лексики, дані що порушують авторські права, посилання на заборонені веб-сайти, віруси що передаються у вигляді файлів тощо [63]. Розміщене НІН маже бути образливим, неприйнятним для інших учасників ВС і спричиняти конфлікти. НІН не є корисним для контенту ДПЗ, яка має відповідати державним і міжнародним стандартам. Задля запобігання цієї проблеми було розроблено алгоритми, що дозволяють виявити та усунути (замінити або видалити)

небажаний контент, який було виявлено після завантаження дописів з ВС до СКД завдяки словнику «небажаного інформаційного наповнення».

Після завантаження дописів до СКД, які є унікальні за атрибутами – ідентифікатор та URL адреса, необхідно перевірити їх вміст щодо наявності НІН за допомогою заздалегідь розробленого експертом словнику, який містить заборонені слова/вислови, посилання на веб-ресурси та постійно доповнюється.

Процедура виявлення та фільтрації заборонених слів у дописах відбувається за схемою, представленою на рис.3.8.

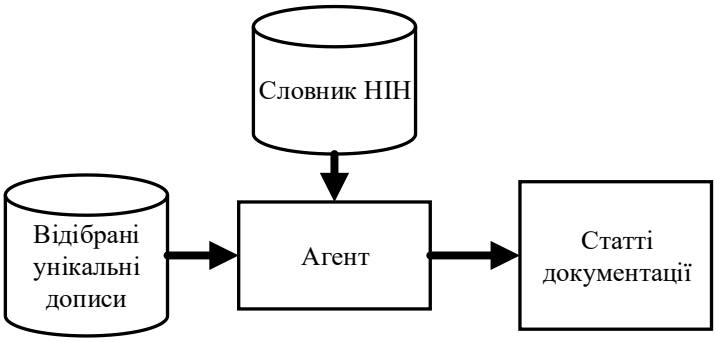


Рис.3.8 Загальна схема перевірки забороненого контенту

Виконання алгоритму вилучення НІН не є повністю автоматизованим та потребує залучення експерта, адже заборонені слова/словосполучення (ЗС) можуть мати різні значення в залежності від контексту заміна або видалення яких може привести до втрати основного змісту. Так само із забороненими посиланнями (ЗП) та вкладеними забороненими файлами (ЗФ). Наведені нижче алгоритми відображають дії поведінки системи в різних ситуаціях (при виявленні ЗС – рис.3.9, при виявленні ЗП або ЗФ – рис.3.10), а також присвоєння відповідних статусів.

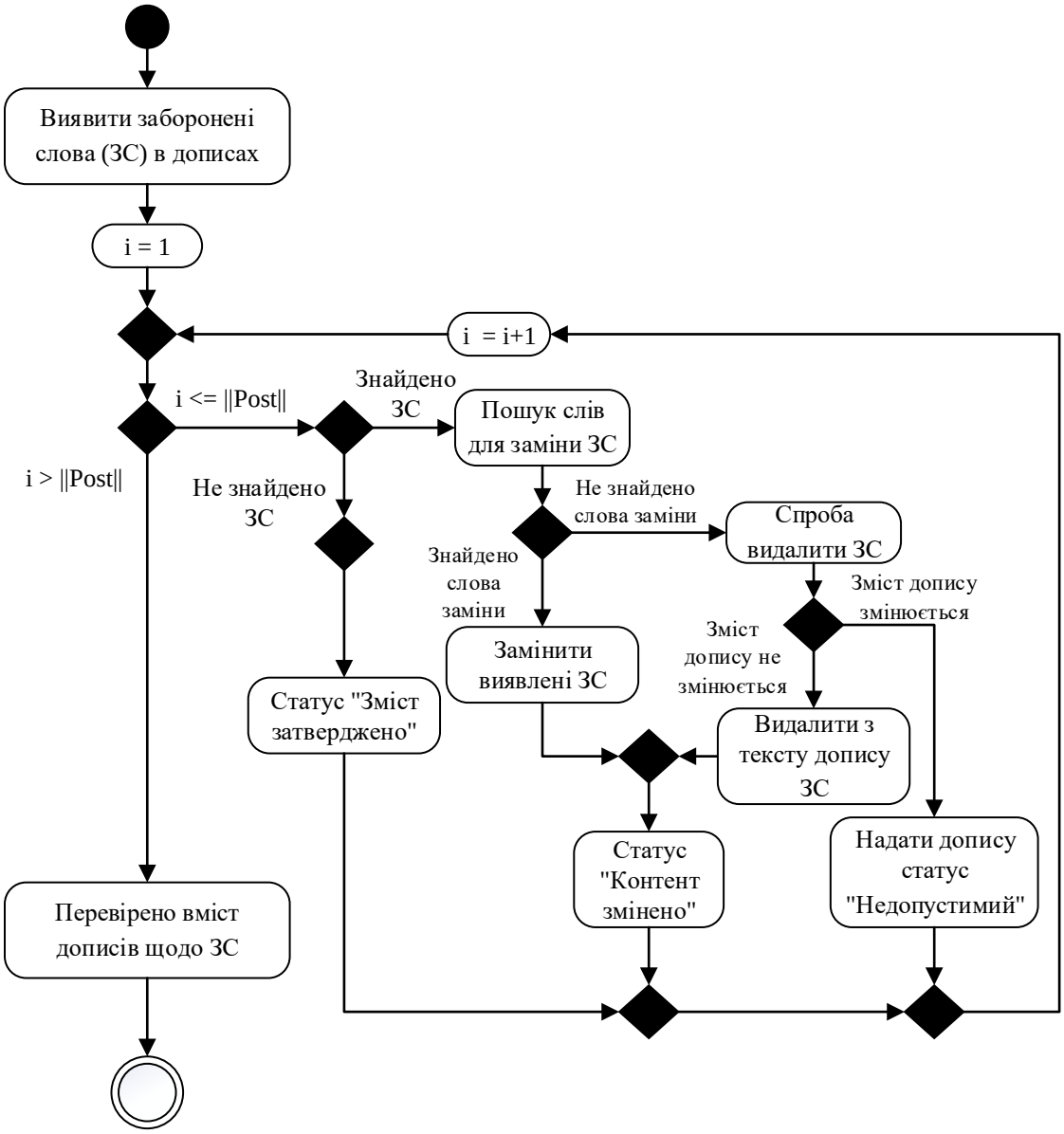


Рис. 3.9 UML діаграма дій щодо виявлення та усунення ЗС в дописах

Виявлені дописи зі статусом «Недопустимий» потребують огляду і перевірки експертом для покращення роботи системи.

Видалення виявлених ЗП або ЗФ у тексті допису, що містить опис про ПЗ може змінювати його зміст. Тому потрібно, щоб система намагалась замінити ЗП і ЗФ. Наприклад, при виявленні ЗП на сайт щодо завантаження неліцензійного ПЗ або ЗФ що містить його, направляти на офіційні сайти виробників.

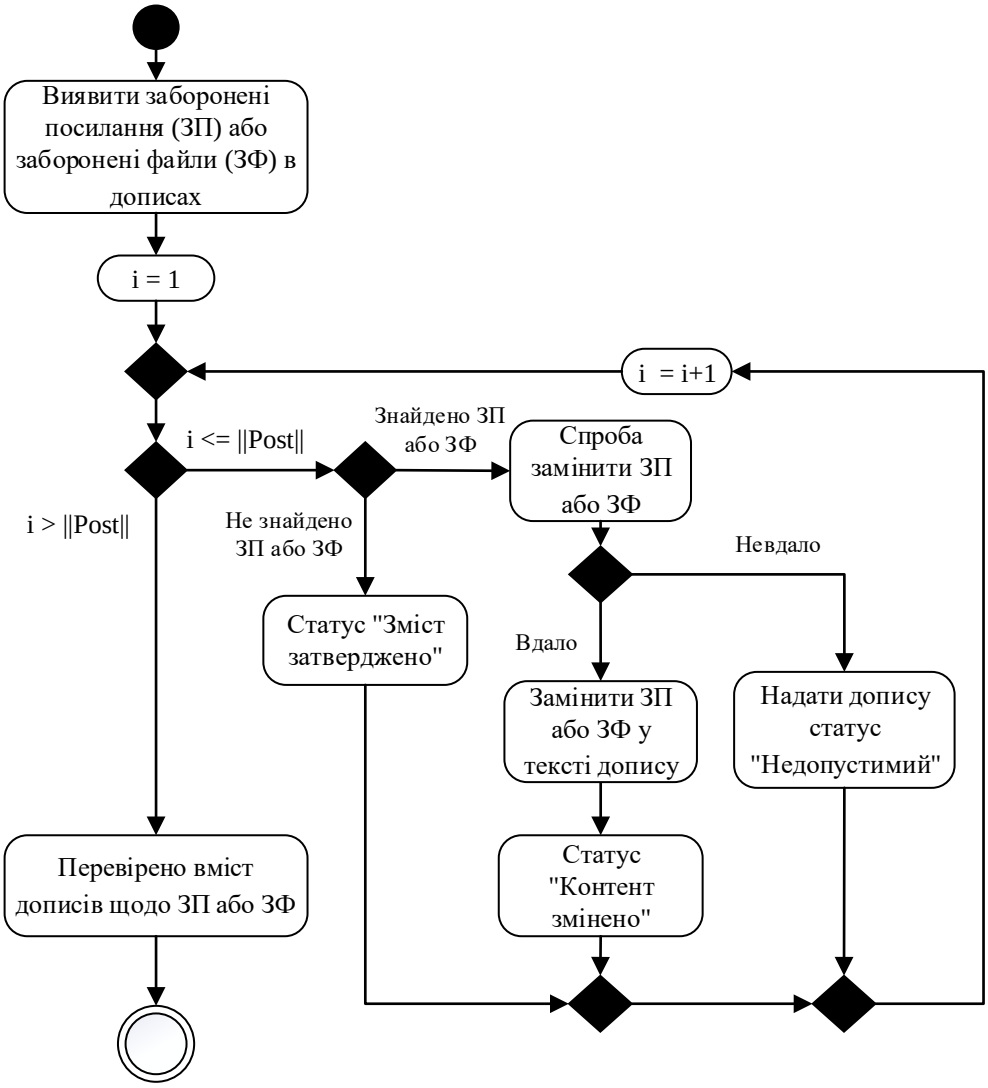


Рис.3.10 UML діаграма дій щодо виявлення та усунення НІН

Результатом проходження алгоритмів що зображено на рис.3.9 та рис.3.10. є дописи, які пройшли перевірку щодо виявлення НІН та мають наступні статуси:

- «Зміст затверджено» – дописи в яких не виявлено НІН щодо ЗС, ЗП та ЗФ;
- «Контент змінено» – дописи в яких було виявлено НІН та успішно замінено;
- «Недопустимий» – дописи, що містять НІН яке неможна замінити (наприклад, видалення НІН змінює основний зміст інформаційного наповнення допису).

Під час перевірки експертом дописів можливі наступні сценарії розвитку подій:

- заміна НІН словами, що мають близьке або тотожне лексичне значення (синоніми) і відповідно поміняти статус допису та занести ці дані в словник;
- видалення НІН без втрати змісту допису – заміна статусу;
- затвердження наданого статусу допису через неможливість заміни/ видалення НІН.

Дописи, які успішно пройшли перевірку щодо НІН та отримали статус «Зміст затверджено» або «Контент змінено», можуть бути статтями ДПЗ.

3.3.1. Алгоритм вилучення допису з дискусії що отримав статус «Недопустимий»

Дописи, що мають статус «Недопустимий», можуть розміщуватися в дискусії наступним чином:

- на першому рівні – допис що задає дискурс (допис-дискусія);
- на другому або нижчих рівнях – бути коментарем на попередній допис (допис-коментар).

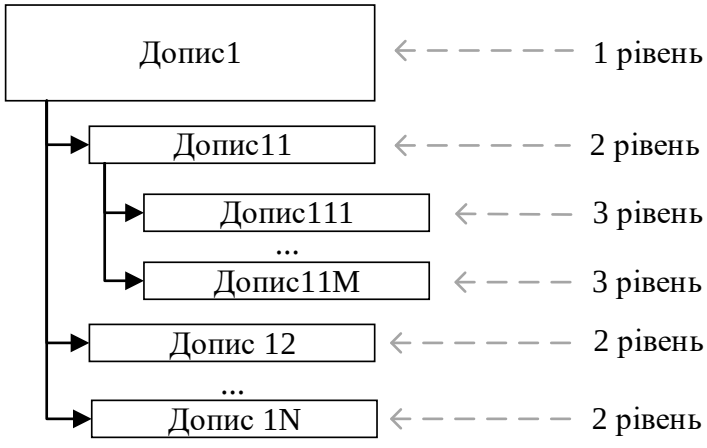


Рис. 3.11 Представлення рівнів розташування дописів в дискусії

Відповідно, якщо допис на першому рівні має статус «Недопустимий» доцільно вилучати всю дискусію, адже всі наступні дописи напряму пов’язані із ним.

Якщо допис зі статусом «Недопустимий» знаходиться на другому або нижчих рівнях дискусії, тоді потрібно вилучити всі коментарі до нього (дописи що розташовані на рівень або більше рівнів нижче та є відповідями на нього). Наприклад, Допис11 має статус «Недопустимий» відповідно всі наступні дописи-коментарі до нього потрібно ігнорувати (рис.3.12).

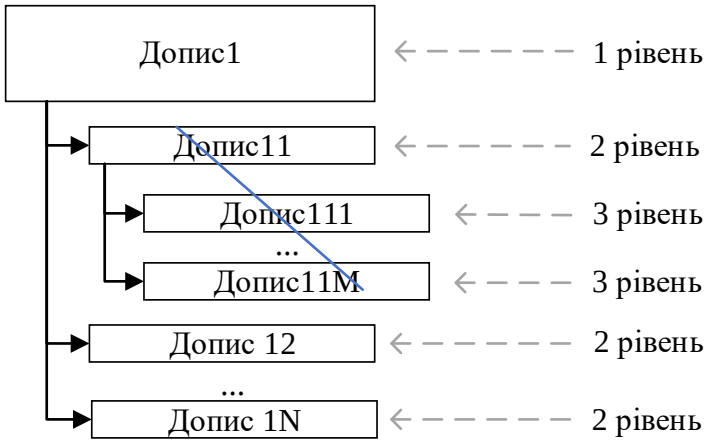


Рис.3.12 Приклад вилучення частини дискусії

3.4. Розроблення алгоритму структурування інформаційного наповнення документації програмного забезпечення

У процесі формування ДПЗ виникає ряд питань щодо структурування інформаційного наповнення. Структурована подача даних має такі переваги:

- дозволяє систематизувати та впорядкувати дані;
- можливість перевірити розділи згідно характеристиками якості, що наведені у розділі 4.1.

Для упорядкування інформаційного наповнення ДПЗ автором розроблено наступний алгоритм який передбачає:

- формування оптимальної структури ДПЗ за допомогою апарату Б - дерев;
- рівномірний розподіл кількості статей для кожної гілки ДПЗ з метою дотримання якості представлення інформаційного наповнення.

3.4.1. Представлення структури документації програмного забезпечення

Структуру ДПЗ можна подати у вигляді дерева (ієрархії), де коренем виступає назва ДПЗ, а всі інші вузли (крім кореня – нульовий рівень) є розділами та статтями.

Перший рівень розділів ДПЗ утворюють класифікатори, опис яких наведено у розділі 2.2.1. Також у СКД є множина унікальних за змістом, без НІН, авторитетних дискусій. Відповідно такі дискусії можуть бути статтями ДПЗ, тому для них необхідно застосувати метод автоматизованого виявлення термінів з ваговими коефіцієнтами, який описано у розділі 2.5.

Вагомі, значущі терміни (які задовольняють умові описаній формулою 2.42) однієї статті ($Article_i$) можуть належати декільком класифікаторам:

$$Article_i \in \left\{\langle Term_j, w_{ij}\rangle\right\}_{j=1}^{N^{(TermArticle_i)}} \in \{Classifier_k\}_{k=1}^{6}, \qquad (3.9)$$

де $Term_j \in \boldsymbol{Term}$ – термін з множини термінів, що зберігаються у словнику;

w_{ij} – вага терміну $Term_j$ щодо статті $Article_i$;

$N^{(TermArticle_i)}$ – кількість вагомих термінів у статті $Article_i$;

$Classifier_k$ – k-тий класифікатор.

На першому рівні при відображенні всіх розділів, класифікаторів, можуть існувати повтори статей, бо стаття містить множину вагомих термінів, які можуть належати декільком класифікаторам. Але при відображенні конкретного класифікатору повторень статей немає.

Перший рівень розділів є сформований і містить назви класифікаторів. Надалі необхідно заповнити другий і наступні рівні розділів здійснивши розподіл згідно структури термінів приклад яких наведено у розділі 2.2.1. Таким чином, класифікатор приналежності до конкретного ПЗ ($Classifier_1$) на другому рівні має терміни щодо виробника ПЗ, на третьому рівні – назва ПЗ і т.д.

Так як кожен термін напряму або через інші терміни належить класифікаторам з певними ваговими коефіцієнтами, тому не існуватиме термінів

статей за якими не можна встановити приналежність до розділу ДПЗ першого рівня.

Після заповнення ДПЗ статтями, виникає потреба у перевірці структури – розділів на кожному рівні, адже може виникнути дисбаланс, що викликаний кількісним розподілом розділів та/або статей.

3.4.2. Розроблення алгоритму оптимального розподілу інформаційного наповнення документації

Однією із важливих задач щодо структурування ДПЗ постає задача балансування (рівномірного розподілу) інформаційного наповнення щодо кожного її розділу. Адже кількість статей в одному розділі може значно перевищувати кількість іншого, що знижує якість структури документації та ускладнює пошук необхідних даних для кінцевих користувачів. Тому виникає потреба у переструктуруванні розділів ДПЗ.

Наявність дисбалансу серед розділів ДПЗ може бути викликано такими чинниками:

- наявність різної кількості статей щодо певних тем (кількість статей на одну тематику суттєво переважає кількість статей щодо іншої);
- через неправильне визначення вагових коефіцієнтів та присвоєння їх термінам;
- словник ключових фраз є неповним, що спричиняє брак термінів та/або точність визначення їх вагових коефіцієнтів.

Так як ДПЗ має деревовидну структуру тому до неї можна застосувати певні критерії методів балансування дерев. Метод оптимального розподілу інформаційного наповнення ДПЗ полягає у забезпеченні виконання наступних критеріїв збалансованості:

1. довжина та ширина (кількість вершин на кожному рівні) гілок ДПЗ за допомогою апарату теорії B-дерев [63];
2. вага між сусідніми вершинами, що визначається кількістю статей які їм належать.

Структурування інформаційного наповнення ДПЗ стосовно критеріїв ширини та вагового розподілу доцільно робити починаючи з третього рівня розділів. Адже на вищих рівнях розташовані значущі терміни над якими не можна здійснювати операції об'єднання або розділення розділів, які, зазвичай, застосовують при виявленні дисбалансу. Наприклад, на другому рівні класифікатору приналежності до конкретного ПЗ розташовані дві вершини, які містять терміни виробників «Корпорація Intel» та «Фірма Microsoft Windows», відповідно статті, що належать до одної вершини, не можна переносити до іншої або створювати сусідню додаткову вершину (при розділенні).

Критерій довжини піддерев. Згідно структури Б-дерев дерево ДПЗ є збалансованим, якщо висоти всіх піддерев класифікаторів відрізняються не більше, ніж на одиницю.

Критерій ширини піддерев. Ґрунтуючись на методі балансування – теорії Б-дерев, дерево ДПЗ є збалансованим, якщо, на кожному рівні кількість вершин, розділів або статей кожного піддерева задовольняє умові [63]:

$$N \leq Part_l \leq 2N, \tag{3.10}$$

де N – мінімальна кількість вершин, розділів або статей на l-му рівні;

$2N$ – максимальна кількість вершин розділів або статей на l-му рівні.

Значення мінімальної кількості N задає експерт, адже дане число залежить від обсягів даних стосовно розділів та статей ДПЗ.

Ваговий критерій піддерев. Дерево ДПЗ є збалансованим, якщо ваги кожної вершини піддерева – сусідніх вузлів, що мають спільну вершину, та вузлів, які мають спільне ребро відповідають умові:

$$M \leq CountOfArticles \leq 3M, \tag{3.11}$$

де M – мінімальна кількість статей, що належить вузлу/вершині;

$3M$ – максимальна кількість статей, що належить вузлу/вершині.

Вага дерева визначається як сума кількості статей цілого піддерева, що йому належить. Приклад дерева на якому відображено кількісний розподіл статей на кожну вершину наведено на рис.3.13. Ліве піддерево задовольняє умові

(як на одному рівні так і серед різних рівнів), наведеній формулою 3.11, в той час як праве піддерево має наступний ваговий розподіл:

- на одному рівні (четвертий рівень) – кількість статей однієї вершини в п’ять разів переважає кількість статей іншої;
- на різних рівнях – кількість статей верхньої вершини в десять разів переважає кількість статей нижньої вершини.

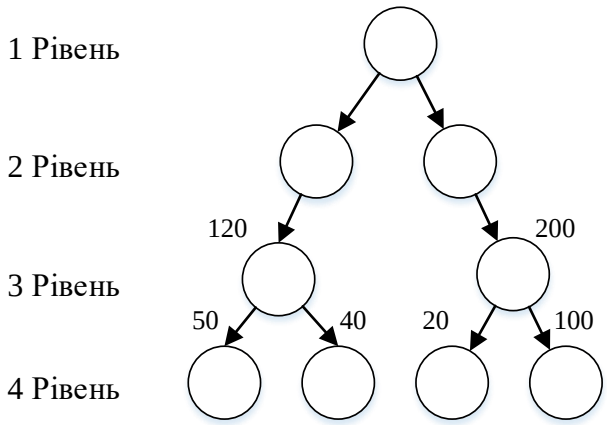


Рис.3.13 Можливий ваговий дисбаланс дерева ДПЗ

При виявленні дисбалансу виконують операції об’єднання або розділення вершин:

- як на одному рівні;
- між різними сусідніми рівнями.

Необхідність об’єднання вершин говорить про те, що терміни кожної з них можуть бути занадто деталізовані і потребують узагальнення. В той час, як поділ вершин передбачає деталізацію термінів.

В роботі автором розроблено алгоритм структурування інформаційного наповнення ДПЗ за вище зазначеними критеріями таким чином:

1. Перевірка структури ДПЗ за критеріями балансування: вглиб, вшир та вагового розподілу:
    1.1. якщо сформована ДПЗ задовольняє всім критеріям, тоді вважатимемо структуру оптимальною;

1.2. якщо сформована ДПЗ не задовольняє критеріям, тоді потрібно визначити який саме критерій є не виконаний та здійснити об’єднання/розділ вершин в залежності від потреби.

2. Застосувати крок 1 доти, доки не буде виконаний пункт 1.1.

Існує випадок коли усунути дисбаланс стосовно структури ДПЗ неможливо – терміни, що утворюють розділи, не можна деталізувати або узагальнити (бо вони, наприклад, є не сумісні). Тому критерій(ї) збалансованості можуть бути не виконані. В такому разі необхідно зафіксувати інформацію про виявлений дисбаланс і передати експертові для розгляду.

3.4.3. Формування оптимальної глибини класифікаторів документації програмного забезпечення

ДПЗ мітить класифікатори, структуру якої утворюють терміни (розділ 2.2.1).

Існує неофіційне правило навігації на сайті – правило «трьох кліків». Воно передбачає, що користувач повинен мати можливість знайти будь-яку інформацію не більше ніж за три кліки мишею.

Передбачається що користувач отримує ДПЗ починаючи з першого рівня на якому розташовані класифікатори.

Слідуючи цьому правилу, класифікатори, що представлені у вигляді дерева, повинні мати не більше чотирьох рівнів (перший, другий, третій та четвертий), а висота дерева – число ребер між коренем та листками не повинна перевищувати значення три. Тобто на першому рівні – назва класифікатору, другий – терміни, третій – терміни, четвертий – статті.

ДПЗ має оптимальну глибину якщо кожен класифікатор задовольняє даній умові.

3.4.4. Розробка алгоритму перевірки актуальності інформаційного наповнення документації програмного забезпечення

Як було зазначено раніше ДПЗ повинна містити актуальні дані. Тому час від часу доцільно здійснювати перегляд дати розміщення статей наявних у ДПЗ. Актуальність ДПЗ визначатимемо датою розміщення повідомлень у статтях.

Так, як стаття ДПЗ є еквівалентною щодо структури дискусії СКД тоді для перевірки актуальності можна застосувати формули 3.5 і 3.6. Ті статті, які не задовольняють умові 3.6 доцільно вилучати з ДПЗ.

Висновки до розділу

В третьому розділі розроблені алгоритми для пошуку, відбору та перевірки дописів сховища консолідованих даних та статей документації. Зокрема:

- розроблено алгоритм визначення адекватності джерела за умовами релевантності запиту користувача та актуальності вмісту, що дозволило відібрати джерела які містять актуальні дані про програмне забезпечення;
- побудовано алгоритм виявлення та усунення дублювання інформаційного наповнення для завантаження/оновлення нових/існуючих дописів/статей до сховища консолідованих даних/ документації програмного забезпечення, що забезпечило наявність унікального контенту;
- описано алгоритм виявлення та усунення небажаного інформаційного наповнення серед унікальних дописів, що дозволило забезпечити якість вмісту документації програмного забезпечення;
- розроблено алгоритм структурування інформаційного наповнення документації програмного забезпечення на основі критеріїв балансування дерев, що забезпечило якість відображення документації програмного забезпечення.

Розділ 4. Розроблення програмного комплексу формування документації за допомогою віртуальних спільнот

Реалізація описаних методів та алгоритмів щодо формування документації програмного забезпечення джерелом якої будуть віртуальні спільноти є складним та громіздким завданням, а отже потребує розробки відповідного програмного комплексу. Основними задачами постають пошук, збір даних з віртуальних спільнот, для подальшого їх опрацювання та збереження в документації програмного забезпечення, а також її періодичне оновлення, доповнення новою інформацією, аналіз якості відповідно до міжнародних стандартів. Це пов’язано з тим, що віртуальні спільноти є динамічними – постійно доповнюються новою інформацією, з іншого боку документація програмного забезпечення повинна містити актуальні дані так як індустрія інформаційних технологій швидко розвивається.

Для створення програмного засобу формування ДПЗ спроектовано реляційні бази даних.

Для оцінювання якості документації програмного забезпечення як сформованого інформаційного продукту застосовано характеристики та під-характеристики стандарту ISO/IEC-25010.

Основні результати розділу опубліковані автором у працях [46, 66].

4.1. Розробка показників якості документації програмного забезпечення

Основним завданням ДПЗ є надання достовірної, актуальної та представленої у структурованому вигляді інформації споживачеві, яка б задовольняла всі його потреби і відповідала вимогам та стандартам. Тому виникає потреба у проведенні аналізу сформованої ДПЗ за показниками якості, що надають державні та міжнародні стандарти.

Так як стандарт ISO/IEC-25010 містить набір показників якості, частиною яких виступають державні стандарти наведені у розділі 1.4, тому якість ДПЗ будемо оцінювати саме за цим міжнародним стандартом.

Деревовидна структура ДПЗ у вигляді ієрархії термінів дозволяє досліджувати всі показники оперуючи термінами. Тому всі характеристики і під-характеристики описані саме з цього погляду, значення мір яких визначається інтервалом [0, 1].

Характеристика функціональна придатність містить під-характеристику – **відповідність**, яку можна визначати за функціональною повнотою [45], що відображає структуру ДПЗ наявністю термінів класифікаторів.

Функціональну повноту визначатимемо наступним чином:

$$\mu(Quality^{(FS)}) = 1 - \frac{A^{(FS)}}{B^{(FS)}}, \tag{4.1}$$

де $A^{(FS)}$ – кількість термінів, яких немає в структурі ДПЗ.

$B^{(FS)}$ – загальна кількість термінів які визначають структуру ДПЗ.

Отже, відповідність за функціональною повнотою відображає міру наявності в ДПЗ структурних прогалин за термінами які належать класифікаторам.

Характеристика придатність до використання містить під-характеристику – **відповідність розпізнавання**, яку визначатимемо такими метриками [45]:

- повнота описів (completeness of descriptions);
- достовірність (reliability) інформаційного наповнення ДПЗ;
- актуальність (actuality) інформаційного наповнення ДПЗ.

Відповідність за функціональною повнотою описів відображає прогалини вмісту, інформаційного наповнення ДПЗ через терміни що не мають приналежності до наявних статей.

Метрика повнота описів визначається наступним чином:

$$\mu(Quality^{(O.CD)}) = 1 - \frac{A^{(O.CD)}}{B^{(O.CD)}}, \tag{4.2}$$

де $A^{(O.CD)}$ – кількість термінів, які не мають статей в ДПЗ.

$B^{(O.CD)}$ – загальна кількість термінів ДПЗ.

Метрику достовірності інформаційного наповнення ДПЗ визначатимемо через рівень довіри до розміщених даних у статтях. Рівень довіри формується на основі рейтингу наявних статей у ДПЗ або авторів, що їх опублікували. Статті містять множину термінів, тому метрику достовірності визначатимемо:

$$\mu(Quality^{(O.R)}) = 1 - \frac{A^{(O.R)}}{B^{(O.R)}}, \quad (4.3)$$

де $A^{(O.R)}$ – кількість термінів, рівень довіри до яких є низьким.

$B^{(O.R)}$ – загальна кількість термінів ДПЗ.

Метрику актуальності інформаційного наповнення ДПЗ визначатимемо датою публікації/оновлення статей, які містять терміни:

$$\mu(Quality^{(O.A)}) = 1 - \frac{A^{(O.A)}}{B^{(O.A)}}, \quad (4.4)$$

де $A^{(O.A)}$ – кількість термінів, що стосуються неактуальних статей.

$B^{(O.A)}$ – загальна кількість термінів ДПЗ.

Слід зазначити, що актуальність та достовірність може змінюватися в часі.

Характеристика супроводжуваність містить під-характеристику – **модульність,** що впливає на швидкість пошуку даних та їх сприйняття. Якість ДПЗ за даною метрикою визначатимемо критеріями щодо структурування інформаційного наповнення, які наведено у розділі 3.6:

1. довжина (length) кожного з піддерев ДПЗ на основі понятійного апарату B-дерев, що визначається рівнями розташування термінів наступним чином:

$$\mu(Quality^{(M.L)}) = 1 - \frac{A^{(M.L)}}{B^{(M.L)}}, \quad (4.5)$$

де $A^{(M.L)}$ – кількість термінів, в яких виявлено дисбаланс щодо довжини дерева термінів.

$B^{(M.L)}$ – загальна кількість термінів ДПЗ.

2. ширина (width) визначається кількістю вершин на кожному рівні за допомогою теорії B-дерев наступним чином:

$$\mu(Quality^{(M.W)}) = 1 - \frac{A^{(M.W)}}{B^{(M.W)}}, \quad (4.6)$$

де $A^{(M.W)}$ – кількість термінів, в яких виявлено дисбаланс щодо ширини дерева термінів.

$B^{(M.W)}$ – загальна кількість термінів ДПЗ.

3. вага (weight) між сусідніми вершинами, що визначається кількістю статей що їх належать:

$$\mu(Quality^{(M.Weig)}) = 1 - \frac{A^{(M.Weig)}}{B^{(M.Weig)}}, \quad (4.7)$$

де $A^{(M.Weig)}$ – кількість термінів статей, в яких виявлено дисбаланс щодо ваги між сусідніми вершинами дерева термінів.

$B^{(M.Weig)}$ – загальна кількість термінів ДПЗ.

Супроводжуваність ДПЗ за під-характеристикою модульність відображає рівень структурованості, дотримання балансу інформаційного наповнення.

4.2. Розробка архітектури системи формування документації програмного забезпечення

Наведені у третьому розділі алгоритми та методи для формування ДПЗ за допомогою ВС є підставою для створення програмно-алгоритмічного комплексу, розробка якого потребує проведення декомпозиції шляхом поділу одного завдання рішенням набору менших завдань – компонент, що є простішими. Для реалізації поставлених завдань застосовано багатоагентну систему, що складається з множини взаємодіючих агентів, як автономних та інтерактивних програмних систем, які вирішують певні окремі задачі, самостійно активізуються та можуть взаємодіяти з іншими агентами і середовищем, у якому функціонують в різні моменти часу [97]. Агент, як комп'ютерна програма, має такі властивості: ефективність, коректність, повнота, надійність [80].

Переваги застосування мультиагентної системи для досягнення основної мети полягають у наступному [66, 97]:

- розподілена робота агентів на різних обчислюваних ресурсах;

- зменшення часових затрат та навантаження на процесор комп’ютера;
- контроль та ефективність роботи кожного агенту;
- можливість розподілу завдань для розробників (груп розробників) агентів;
- декомпозиція одної складної задачі на простіші окремі задачі;
- автономна робота кожного агенту.

Агенти напряму між собою не взаємодіють, а тільки через СКД або документацію. Архітектуру системи формування ДПЗ із залученням агентів наведено на рис.4.1.

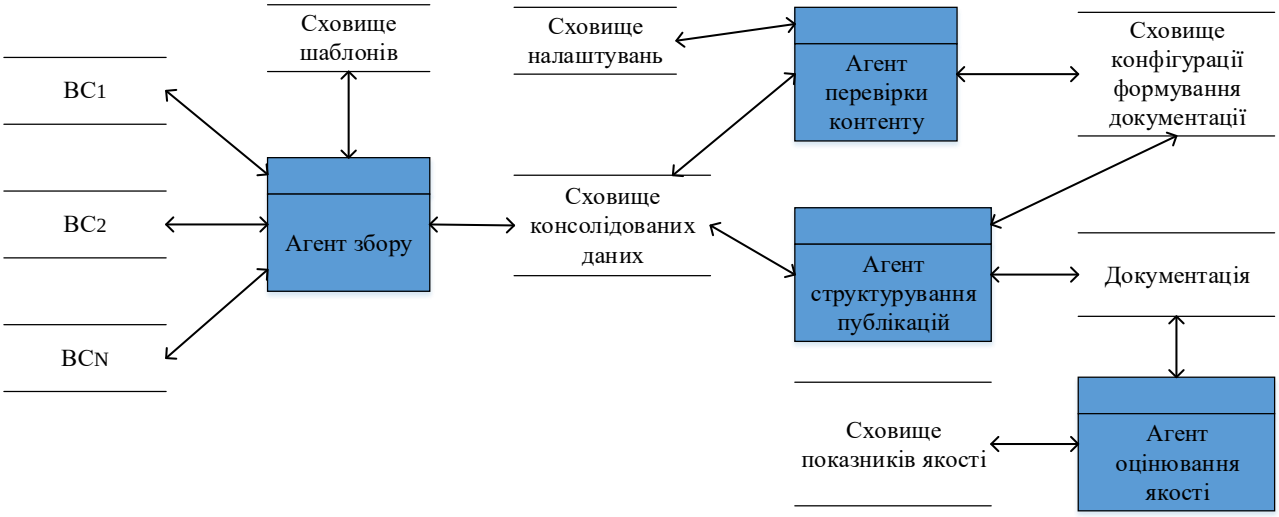


Рис.4.1 Архітектура системи формування ДПЗ

Архітектура системи формування ДПЗ складається з наступних агентів [66]:

- **агент збору даних.** З попередньо відібраних експертом ВС та розроблених відповідних шаблонів агент збирає наступні дані: теми, дописи та авторів у СКД для подальшої їх обробки іншими агентами;
- **агент перевірки контенту.** На основі наданої інформації агентом збору даних (занесених у СКД) та за допомогою сховища налаштувань, визначає адекватні теми до запиту користувача серед загального списку тем; проводить аналіз рейтингу: дописів, дискусій, авторів, спільнот. Також перевіряє нові та існуючі дописи щодо унікальності вмісту для уникнення дублювання даних, а також

наявності небажаного інформаційного наповнення, в результаті чого проставляє відповідні мітки;

- **агент структурування інформаційного наповнення.** Агент визначає приналежність статті до термінів ДПЗ, що утворюють розділи документації. Надалі вирішує задачу оптимального розподілу інформаційного наповнення за критеріями балансування: довжини, ширини та вагового навантаження для кожного піддерева;
- **агент якості.** Агент перевіряє сформовану ДПЗ за показниками якості, що наведені в розділі 4.1. на основі показників стандарту ISO/IEC-25010 та рейтингування даних.

Статуси роботи, виконання завдань будь-якого агенту можуть бути наступні:

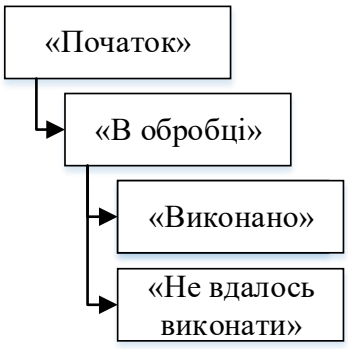


Рис.4.2 Статуси роботи будь-якого агенту

4.3. Аналіз результатів експерименту

4.3.1. Робота агенту збору даних.

Основними задачами агенту збору даних є:

1. Завантаження у СКД нових тем з ВС у вигляді списку.

2. Завантаження нових та оновлення існуючих дописів з адекватних тем, що можуть становити інформаційне наповнення ДПЗ.

Пошук віртуальних спільнот здійснює експерт, тим самим формуючи список назв ВС та посилань на них у вигляді URL адрес. При завантаженні тем з різних ВС може виникнути наступні ситуації:

- теми мають однакові назви;
- теми є близькими за семантичним значенням.

Такі теми доцільно групувати зважаючи на формат спільнот, яким вони належать.

Через те що, кожна ВС має свою об’єктну модель документу (з англ. Document Object Mode – DOM), то перед завантаженням інформації потрібно для кожної ВС розробити загальний шаблон, який буде містити необхідні атрибути для проведення процедури збору даних. Розроблені шаблони зберігаються у *«Сховище шаблонів»*. Задачу збору даних із запропонованого списку ВС за допомогою наданих шаблонів ВС здійснює агент збору даних.

Обмеження в роботі цього агенту є обсяг даних, який можна отримати в результаті їх автоматизованого завантаження з ВС за певними параметрами до уникнення блокування.

Моделювання БД для мультиагентної системи здійснюється на основі реляційних БД, де вся інформація зберігається в таблицях, пов’язаних один з одним спеціальними відносинами [49].

Для представлення даних з якими працює агент збору даних наведена інформаційна модель рис.4.3, що відображає сутності для збереження інформації отриманої в результаті роботи цього агенту.

Даний агент здійснює збір даних з наступних сутностей (рис.4.3):

- ВС: спільноти – сутність «SourceVC», розділи – сутності «Part» і «Tag», дописи – «Post_VC»;
- СКД: відібрані спільноти – сутність «VC», теми – сутність «Theme», дописи – сутність «Post», шаблони ВС – сутність «Format», сутність «History» містить посилання на динамічні сутності спільнот («Add_info_VC»), тем (Add_info_theme) і дописів (Add_info_post), а також посилання агенту – сутність «Agent» та його діяльності «Activity_title».

Рис.4.3 Схема даних для роботи агенту збору даних

Наведемо розв’язок вище наведених завдань агенту збору.

Виконання завдання 1. Новими джерелами даних – темами (для ієрархічної структури), тегами (для мережевої структури) вважатимемо ті, які є у ВС, але немає у СКД:

```
SELECT id_part as ID, title_part as Title, URL_part as URL,
id_sourceVC
FROM Part
INNER JOIN Source_VC ON Part.id_sourceVC = Source_VC.id_sourceVC
INNER JOIN Tag ON Source_VC.id_tag = Tag.id_tag
INNER JOIN VC ON Source_VC.URL_sourceVC = VC.URL_VC
INNER JOIN Theme ON VC.id_VC = Theme.id_VC
WHERE (URL_part <> URL_theme OR URL_tag <> URL_theme)
AND URL_VC = URL_sourceVC
```

Результатом виконання запиту буде отримано теми яких немає у СКД.

Виконання завдання 2. Завантаження нових та оновлення існуючих дописів СКД з адекватних джерел (розділ 3.1) – тем ВС.

Процес завантаження допису з ВС складається з наступних кроків:

Крок 1. Знаходження шляху до допису у відібраній темі із загального списку дописів, у вигляді HTML сторінки.

Крок 2. Перевірка дати публікування/редагування допису для уникнення завантаження дописів, які були опубліковані/редаговані раніше вказаної дати. Для мережевої та гібридної структур ВС, де теги виступають розділами, які необхідно перед завантаженням допису здійснювати перевірку його URL адреси з метою уникнення дублювання. Адже один допис може мати декілька тегів, тобто відноситися до декількох тем.

Крок 3. Завантаження атрибутів допису згідно шаблону.

Приклад шаблону ВС з необхідними атрибутами для розв’язку завдання завантаження назв тем наведено в додатку Д.

Запит щодо завантаження нових дописів з ВС до СКД:

```
SELECT id_VC, title_VC, id_theme, id_post, URL_post
FROM Post_VC
```

```
INNER JOIN Part ON Post_VC.id_part = Part.id_part
INNER JOIN Tag ON Post_VC.id_tag = Tag.id_tag
INNER JOIN Source_VC ON Part.id_sourceVC = Source_VC.id_sourceVC
INNER JOIN VC ON Source_VC.URL_sourceVC = VC.URL_VC
INNER JOIN Theme ON VC.id_VC = Theme.id_VC
INNER JOIN Post ON Theme.id_theme = Post.id_post
INNER JOIN History ON
Post.id_add_info_post = History.id_add_info_post
INNER JOIN Add_info_theme ON
Post.id_add_info_theme = Add_info_theme.id_add_info_theme
WHERE status_theme = “Adequate” AND
AND id_post_VC <> id_post AND
(URL_part = URL_theme OR URL_tag = URL_theme) AND
AND (Post.date_create BETWEEN “YYYY-MM-DD” AND “YYYY-MM-DD”)
```

Допис є актуальним, якщо знаходиться в часовому проміжку зазначеному в запиті. При завантаженні дописів до СКД розроблений запит передбачає перевірку за унікальними номерами дописів, що описано UML діаграмою дій рис.3.5.

Запит на оновлення існуючих дописів з ВС до СКД, текстова частина (назва і текст) яких не була редагована, що мають мітку «існуючий» (згідно алгоритму розділу 3.2.1):

```
SELECT URL_post_VC AS URL, Post_VC.rate_post AS Rate
FROM Post_VC
INNER JOIN Part ON Post_VC.id_part = Part.id_part
INNER JOIN Tag ON Post_VC.id_tag = Tag.id_tag
INNER JOIN Source_VC ON Part.id_sourceVC = Source_VC.id_sourceVC
INNER JOIN VC ON Source_VC.URL_sourceVC = VC.URL_VC
INNER JOIN Theme ON VC.id_VC = Theme.id_VC
INNER JOIN Post ON Theme.id_theme = Post.id_post
INNER JOIN History ON
Post.id_add_info_post = History.id_add_info_post
INNER JOIN Add_info_post ON
History.id_add_info_post = Add_info_post.id_add_info_post
```

```
WHERE id_post_VC = id_post
AND (URL_part = URL_theme OR URL_tag = URL.theme)
AND title_post_VC = title_post AND text_post = text
AND Post_VC.rate_post <> Add_info_post.rate_post
```

Запит на оновлення існуючих дописів з ВС до СКД, текстова частина яких була редагована, що мають мітку «редагований» (згідно алгоритму розділу 3.2.1):

```
SELECT id_post AS ID, URL_post_VC AS URL, rate_post AS Rate
FROM Post_VC
INNER JOIN Part ON Post_VC.id_part = Part.id_part
INNER JOIN Tag ON Post_VC.id_tag = Tag.id_tag
INNER JOIN Source_VC ON Part.id_sourceVC = Source_VC.id_sourceVC
INNER JOIN VC ON Source_VC.URL_sourceVC = VC.URL_VC
INNER JOIN Theme ON VC.id_VC = Theme.id_VC
INNER JOIN Post ON Theme.id_theme = Post.id_post
WHERE id_post_VC = id_post
AND (URL_part = URL_theme OR URL_tag = URL_theme) AND
AND title_post_VC <> title_post AND text_post <> text
```

4.3.2. Робота агенту перевірки контенту.

Основними задачами агенту є перевірка та відбір інформаційного наповнення що є у СКД за методами, техніками та алгоритмами наведеними в другому і третьому розділах. До них належать:

1. Визначення адекватності теми (розділ 3.1).
2. Обчислення рейтингу допису на основі сценаріїв, що наведені в розділі 2.3.
3. Формування рейтингу автору (розділ 2.3.1).
4. Визначення авторитетних дискусій, на основі рейтингу дописів (розділ 3.1.4).
5. Встановлення актуальних дискусій тем (розділ 3.1.3).
6. Формування рейтингів тем та ВС на основі рейтингу дописів, які завантажено до СКД.

7. Виявлення профілів користувача на основі методу наведеному у розділі 2.6.
8. Виявлення на усунення дублювання інформаційного наповнення у дописах СКД за допомогою методу виявлення подібності між текстами дописів, який наведено у розділі 2.4.
9. Перевірка тексту на наявність НІН на основі методу описаного в розділі 3.3.

Після успішного виконання кожного із завдань (результатом роботи агеннту є статус «Виконано» рис.4.2) агент перевірки контенту проставляє на дописи відповідні мітки. *Сховище налаштувань* містить оцінки порогових значень наведеними формулами 3.1, 3.2, 2.33, 3.6, 3.8, 2.37, 2.38 і 2.44.

Виконання завдання 1. Визначення адекватних тем на основі методу описаного в розділі 3.1 передбачає виконання умови релевантності та актуальності. Кількість та список релеватних тем, що є в СКД, до запиту користувача наведено на рис. 4.4.

```
query = "Навчання front-end спеціаліста з подальшим працевлаштуванням"
relevance_themes = process_query_ua(query, extended=False)
print(f"Знайдено {relevance_themes.shape[0]} релевантних(-і) тем(-и):\n{relevance_themes}")
```

```
Знайдено 3 релевантних(-і) тем(-и):
працевлаштування    1.0
навчання            1.0
front-end           1.0
```

Рис. 4.4 Визначення релевантних тем до звичайного запиту користувача

Кількість та список тем з ваговими коефіцієнтами до розширеного запиту користувача на основі словнику термінів зображено на рис. 4.5.

```
query = "Навчання front-end спеціаліста з подальшим працевлаштуванням"
relevance_themes = process_query_ua(query, extended=True)
print(f"Знайдено {relevance_themes.shape[0]} релевантних(-і) тем(-и):\n{relevance_themes}")
```

```
Знайдено 11 релевантних(-і) тем(-и):
it-спеціалісти       1.0
front-end            1.0
працевлаштування     1.0
навчання             1.0
робота               1.0
frontend             1.0
ринок праці          0.9
job                  0.9
кар'єра              0.7
frontend дайджест    0.6
ринок                0.4
```

Рис.4.5 Визначення релевантних тем до розширеного запиту користувача

Як видно з результатів, розширений запит із застосуванням словнику термінів дозволяє виявити більшу кількість релевантних тем, аніж звичайний запит.

Визначення актуальності тем формується на основі активності абсолютного приросту даних у темах ВС. Для обчислення умови адекватності відібрано теми, що можуть містити дані про ПЗ зі спільнот Dou.ua та CodeProject. Обчислення показника активності передбачає спочатку визначення загальної кількості унікальних дописів-дискусій в темах, а потім приріст в кожній за певний інтервал часу.

Загальна кількість дописів-дискусій щодо завантажених тем спільнот Dou.ua та CodeProject за 2018-2022 роки відображено на рис. 4.6.

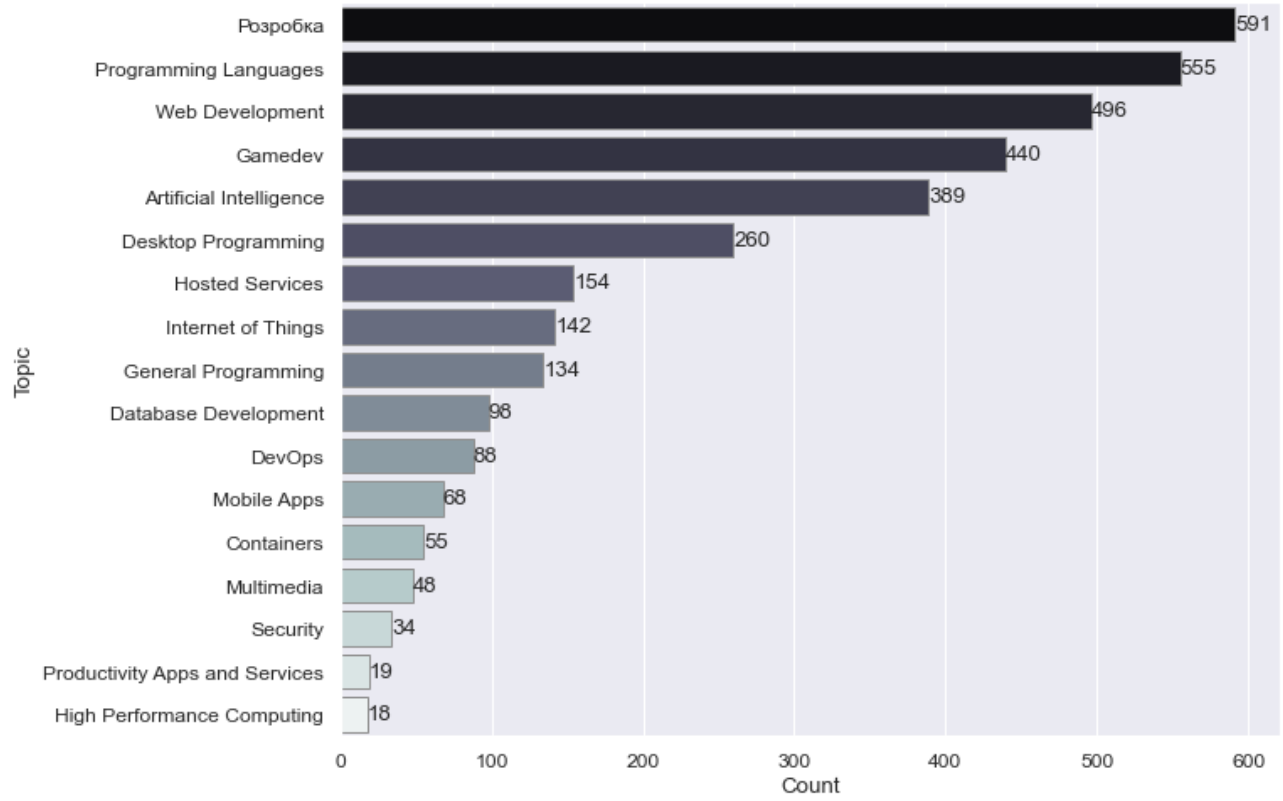


Рис.4.6 Кількість дописів-дискусій для обраних тем.

Надалі відображено щоквартальний приріст (період часу) дописів в темі «Artificial Intelligence» з обчисленим значенням активності згідно формули 3.4 (рис. 4.7).

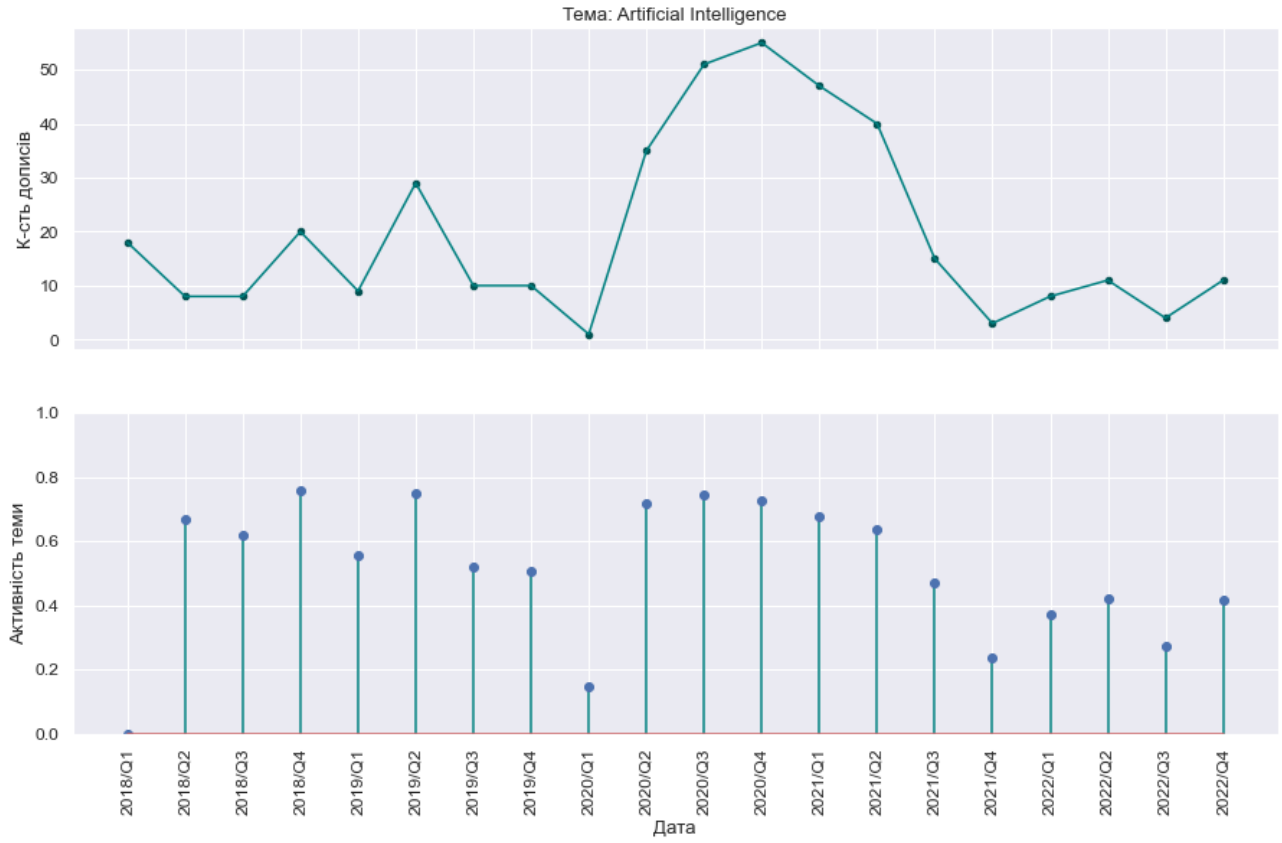


Рис.4.7 Значення активності теми «Artificial Intelligence» за період 2018-2022 роки

Як видно із дослідження, існували періоди коли умова актуальності теми «Artificial Intelligence» не була виконана (наприклад перший квартал 2020 року).

Результатом успішної реалізації (статус роботи агенту «Виконано» (рис.4.2)) завдання щодо визначення адекватності тем є проставлення міток:

- «Адекватна», що задовольняє умовам адекватності наведеним у розділі 3.1;
- «Неадекватна».

Виконання завдання 2. Обчислення рейтингу допису на основі сценаріїв, в залежності від наявних реакцій ВС. Як зазначено у розділі 2.3. реакції на допис можуть бути наступними: оцінювання, взаємодії та коментування.

Реалізація даного завдання передбачає застосування набору інструментів природньої мови (з англ. Natural Language Toolkit – NLTK) – програми та бібліотеки для символьної і статистичної обробки природної мови, реалізованих

мовою програмування Python [55]. Основними компонентами, що були застосовані для роботи агенту перевірки контенту бібліотеки NLTK є [105]:

- лексичний аналіз;
- виявлення подібності та однакового інформаційного наповнення між текстами дописів;
- аналіз тональності тексту (англ. sentiment analysis) дописів-коментарів.

Однією з характеристик обчислення рейтингу допису є реакція «Коментування», що формується на основі використання аналізу тональності тексту дописів-коментарів, проведення якого потребує розпізнавання мови для застосування відповідних словників. Визначення тональності тексту дописів передбачає застосування інструменту аналізу настроїв, що адаптований до настроїв віртуальних спільнот – VADER (Valence Aware Dictionary and Sentiment Reasoner) [88, 86].

Інструмент VADER передбачає розпізнавання слів тексту наступного характеру:

- типові заперечення, схвалення;
- скорочення;
- пунктуацію (знак оклику, кома тощо);
- сленгових слів;
- спеціальних символів що відображають емоції (емограм).

Інструмент VADER містить метод polarity_scores що повертає такі значення (рис. 4.8):

- нормалізовані оцінки тональності тексту (позитивна, негативна та нейтральна оцінки, сума яких знаходиться в інтервалі значень [0, 1]);
- узагальнена об’єднана оцінка «compound» (інтервал значень [-1, 1]).

```
[{'neg': 0.0, 'neu': 0.962, 'pos': 0.038, 'compound': 0.25},
 {'neg': 1.0, 'neu': 0.0, 'pos': 0.0, 'compound': -0.4588}]
```

Рис. 4.8 Результат аналізу допису застосовуючи інструмент VADER

В розроблених сценаріях обчислення рейтингу допису – реакції «Коментування» застосовано оцінки – позитивна та негативна, адже саме вони відображають емоційне забарвлення тексту, що може містити досвід користувачів щодо використання ПЗ. Водночас наявність нейтрального тексту відображає інформацію, що стосується безпосередньо фактів та подій про ПЗ.

Обчислення реакції «Коментування» на основі формули 2.31 здійснюється наступним чином:

```
from nltk.sentiment.vader import SentimentIntensityAnalyzer
epsilon = 1e-5
sid = SentimentIntensityAnalyzer()
def expected_tonality_score_of_comments(texts: list[str]):
    scores = [sid.polarity_scores(text.replace('\n', ' ')) for text in texts]
    return np.mean([ score['pos'] / (score['neg'] + score['pos'] + epsilon) for score in scores ])
```

Рис.4.9 Обчислення реакції «Коментування» застосовуючи інструмент аналізу настроїв VADER

В залежності від мови дописів-коментарів застосовано розроблені словники:

- для україномовного тексту – Український тональний словник [73], слова якого представлено оцінками (-2, - 1, 1, 2) в залежності від ступеню емоційного забарвлення;
- для англомовного тексту [88].

Обчислення рейтингу англомовного допису ВС CodeProject на основі реакцій «Оцінювання» (в діапазоні), «Взаємодії» та «Коментування» згідно формули 2.32. наведено на рис. 4.10.

```
def compute_r_scores(data: pd.DataFrame, coefficients: list = [0.7, 0.1, 0.2]):
    data = data.copy()
    data['DownloadedCount'].replace({np.nan: 0.0}, inplace=True)
    data['StarsCount'].replace({np.nan: 0.0}, inplace=True)
    data['Rating'].replace({np.nan: 0.0}, inplace=True)

    data['R_оцінювання'] = data['Rating'].transform(lambda x: x / 5.0)
    data['R_взаємодії'] = (data['DownloadedCount']*0.4 + data['StarsCount']*0.6) / data['PageViews']
    data['R_тональності'] = [expected_tonality_score_of_comments(texts) for texts in data['Comments'].values]
    data['R'] = data['R_оцінювання']*coefficients[0] + data['R_взаємодії']*coefficients[1] + data['R_тональності']*coefficients[2]
    return data
```

Рис. 4.10 Обчислення рейтингу допису

Результат формування рейтингу дописів згідно формули 2.32 із запропонованими ваговими коефіцієнтами реакцій наведено у вигляді таблиці, де обчислене значення рейтингу дописів (поле, атрибут «R») відсортоване за зростанням (рис. 4.11).

```
rated_data.sort_values('R').loc[:, ['R', 'ArticleUrl', 'R_оцінювання', 'R_взаємодії', 'R_тональності']]
```

	R	ArticleUrl	R_оцінювання	R_взаємодії	R_тональності
14	0.619284	https://www.codeproject.com/Articles/1107480/D...	0.8	0.000840	0.295998
16	0.661933	https://www.codeproject.com/Articles/868653/EX...	0.8	0.000738	0.509294
12	0.669349	https://www.codeproject.com/Articles/671407/Ca...	0.8	0.000602	0.546442
4	0.682092	https://www.codeproject.com/Articles/732679/HT...	0.8	0.000392	0.610266
5	0.687321	https://www.codeproject.com/Articles/5061604/D...	0.8	0.000541	0.636334
9	0.691335	https://www.codeproject.com/Articles/206507/Du...	0.8	0.000360	0.656494
8	0.694869	https://www.codeproject.com/Articles/602794/Bu...	0.8	0.000792	0.673948
15	0.697563	https://www.codeproject.com/Articles/1254918/A...	0.8	0.001396	0.687119
6	0.702588	https://www.codeproject.com/Articles/5061469/C...	0.8	0.000915	0.712483
10	0.708574	https://www.codeproject.com/Articles/5292782/W...	0.8	0.001430	0.742156
7	0.711919	https://www.codeproject.com/Articles/614028/Pe...	0.8	0.000863	0.759161
13	0.720543	https://www.codeproject.com/Articles/1137299/O...	0.8	0.000857	0.802287
1	0.723761	https://www.codeproject.com/Articles/1096861/D...	0.8	0.000326	0.818644
2	0.735098	https://www.codeproject.com/Articles/5324952/H...	0.8	0.001006	0.874987
11	0.744990	https://www.codeproject.com/Articles/1217419/A...	1.0	0.000487	0.224706
17	0.787418	https://www.codeproject.com/Articles/5341837/A...	1.0	0.002228	0.435976
19	0.815000	https://www.codeproject.com/Articles/1274712/S...	1.0	0.000716	0.574640
0	0.820172	https://www.codeproject.com/Articles/5160398/A...	1.0	0.000661	0.600530
18	0.843944	https://www.codeproject.com/Articles/4042463/P...	1.0	0.000729	0.719356
3	0.857963	https://www.codeproject.com/Articles/1224689/S...	1.0	0.001091	0.789267

Рис.4.11 Список дописів спільноти CodeProject відсортованих за зростанням рейтингу

Обчислене значення рейтингу допису зберігається у БД – сутність «Add_info_post» атрибут «rate_post» (рис.4.2).

Результатом успішної реалізації (статус роботи агенту «Виконано» (рис.4.2)) завдання щодо визначення рейтингу дописів є проставлення міток:

- «Прийнятний рейтинг», що задовольняє умові наведеній у формулі 2.33. розділ 2.3;
- «Низький рейтинг».

Виконання завдання 3. Обчислення рейтингу автору описано у розділі 2.3.1. та визначається як середньозважене всіх дописів що було завантажено до СКД (за формулою 2.34).

Обравши унікальних авторів дописів СКД було обчислено їх рейтинг та відображено стовпчиковою діаграмою:

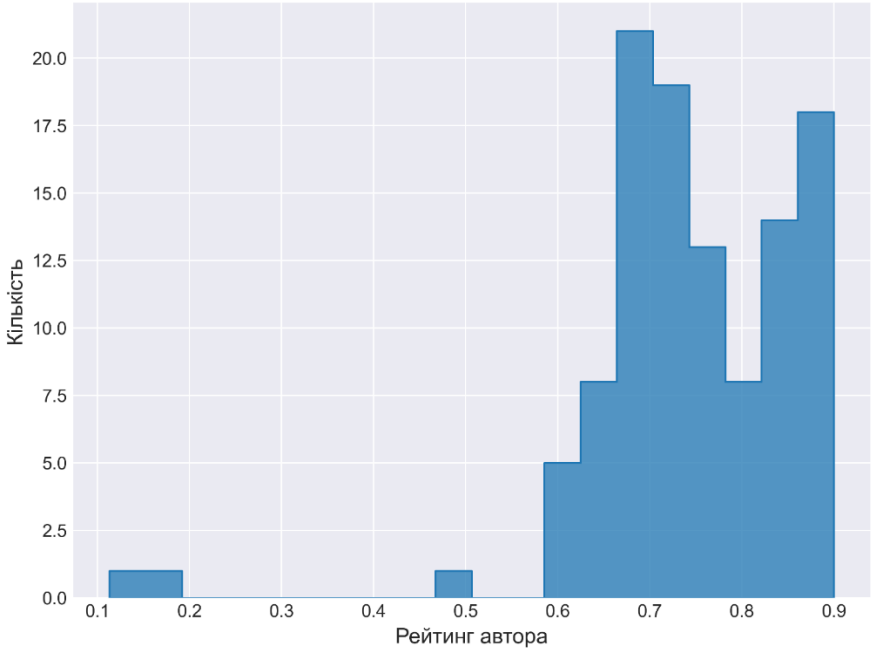


Рис. 4.12 Стовпчикова діаграма рейтингу авторів у СКД

Виконання завдання 4. Визначення актуальності дискусії згідно формули 3.6, яка наведена в розділі 3.1.3, передбачає перевірку дат публікування допису-дискусії та дописів-коментарів, що належать цій дискусії.

Виконання цього завдання потребує представлення даних у вигляді інформаційної моделі рис. 4.13.

Даний агент здійснює збір даних з наступних сутностей:

- СКД: сутність «Post» містить дані допису, «Language» – інформація про мову допису, «Type» зберігає тип допису (допис-дискусія чи допис-коментар), «Author» – дані про автора допису, «Personal_data» – особисті дані;
- ДПЗ: сутність «Creator» містить дані автору статті.

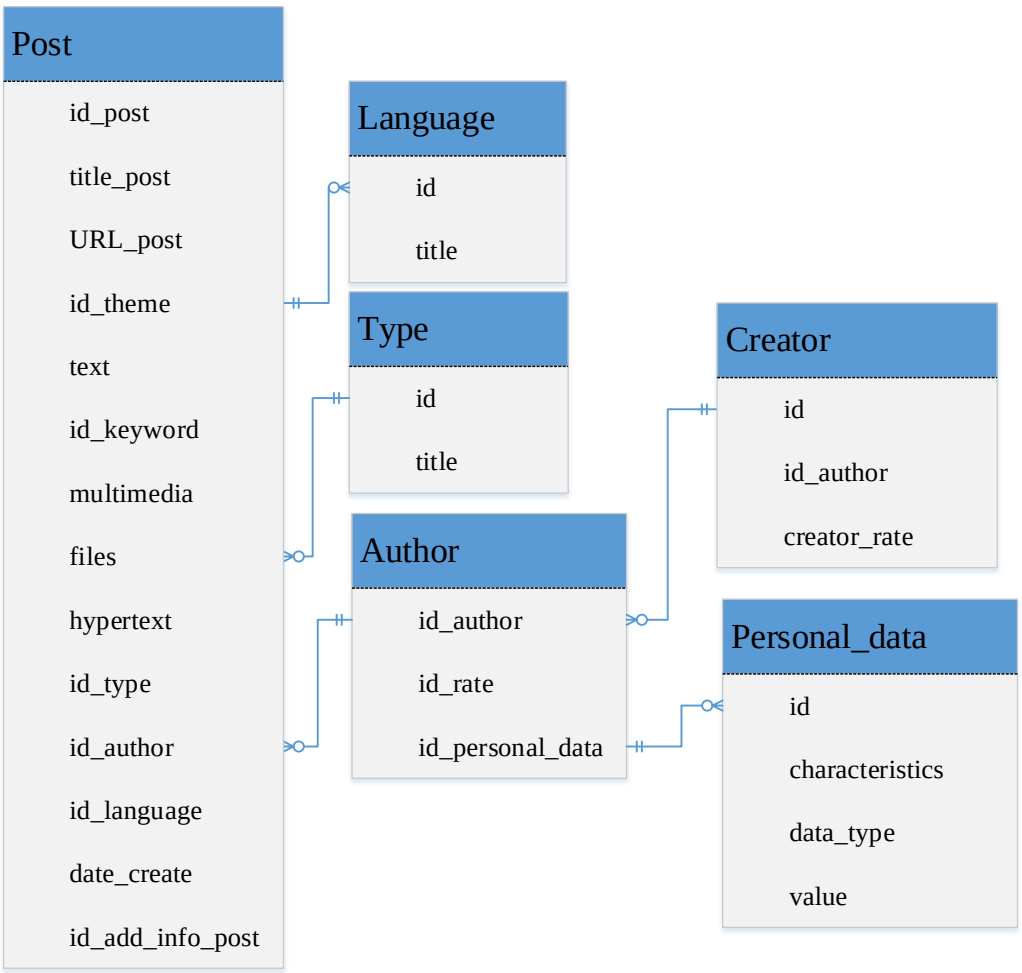


Рис. 4.13 Схема даних для роботи агенту збору даних

Запит на завантаження нових дописів з ВС до СКД:

```
SELECT id_post AS ID, URL_post AS URL, date_create AS Date
FROM Post
INNER JOIN Type ON Post.id_type = Type.id
WHERE (Type.title = “Post-discuss”
AND(Post.date_create BETWEEN “YYYY-MM-DD” AND “YYYY-MM-DD”))
OR(Type.title = “Post-comment”
AND(Post.date_create BETWEEN “YYYY-MM-DD” AND “YYYY-MM-DD”))
```

Результатом успішної реалізації (статус роботи агенту «Виконано» (рис. 4.2)) завдання щодо визначення актуальності дискусії є проставлення міток:

- «Актуальна дискусія», що задовольняє умові актуальності наведеної у розділі 3.2;

- «Неактуальна дискусія».

Виконання завдання 5. Визначення авторитетних дискусій, на основі рейтингу дописів. Для відображення співвідношення обрано 150-ть дискусій з трьох тем (з кожної теми по 50-ть дискусій) та обчислено авторитетність згідно формули 3.7 розділ 3.1.4 Результат рейтингів дискусій представлено трьома діапазонами значень та побудовано стовпчикову діаграму:

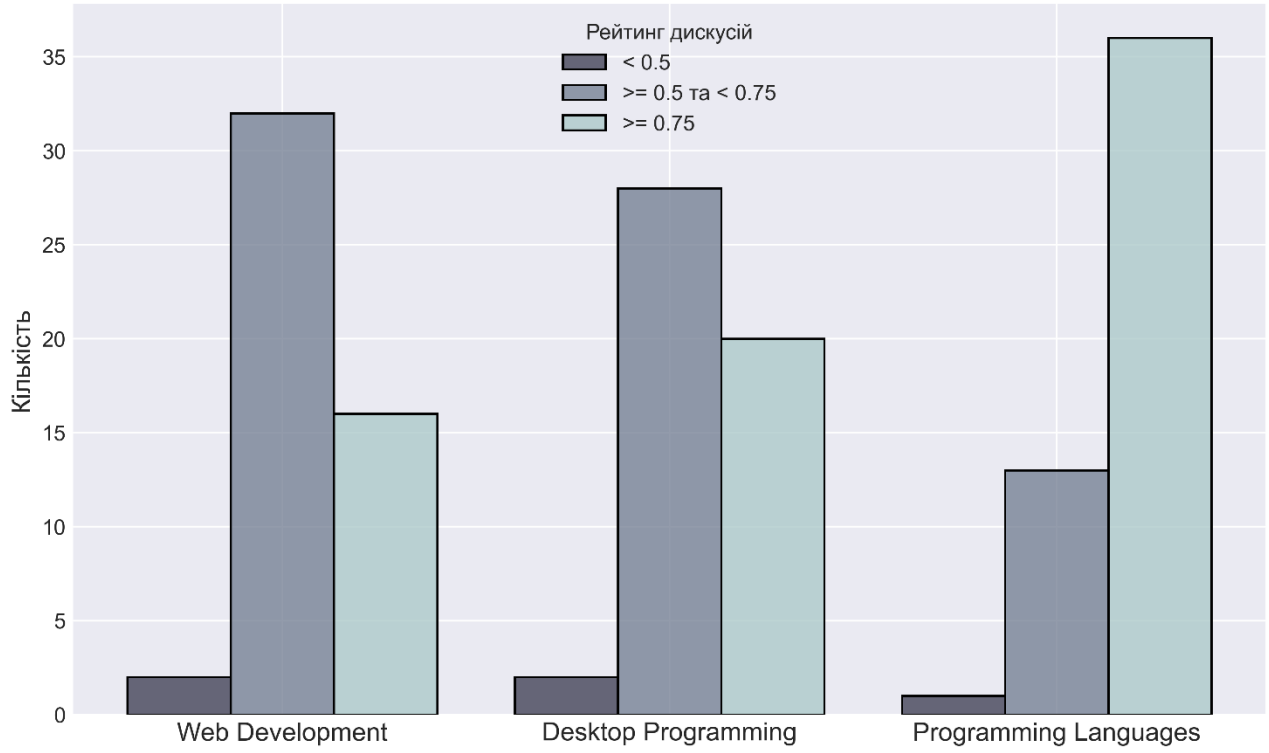


Рис.4.14 Згруповані за значенням рейтингу дискусії тем спільноти CodeProject

Результатом успішної реалізації (статус роботи агенту «Виконано» (рис.4.2)) завдання щодо визначення актуальності дискусії є проставлення міток:

- «Авторитетна дискусія»;
- «Неавторитетна дискусія».

Виконання завдання 6. Теми, які існують у СКД потребують визначення рейтингу за допомогою дописів, які наявні в ній. Так само і для ВС необхідно обчислювати рейтинг на основі рейтингу тем. Це дозволить сформувати певну статистику для експерту. Якщо на протязі декількох звернень спостерігається

низький рейтинг ВС або теми, тоді її необхідно розглянути та прийняти рішення щодо доцільності подальшого завантаження даних з цього джерела.

Обчислення рейтингу теми можна визначати як середнє рейтингів дискусій, які містяться в ній. Запит на обчислення рейтингу всіх тем СКД впорядкованих за зростанням згідно схеми даних наведеної рис. 4.3:

```
SELECT id_theme AS ID, title_theme AS Title, AVG(rate_post) AS Rate
FROM Theme
INNER JOIN Post AS Theme.id_theme = Post.id_theme
INNER JOIN History AS
Post.id_add_info_post = History.id_add_info_post
INNER JOIN Add_info_post AS
History.id_add_info_post = Add_info_post.id_add_info_post
GROUP BY id_theme
ORDER BY AVG(rate_post) ASC
```

Обчислення рейтингу ВС можна визначати як середнє рейтингів тем, які містяться в ній.

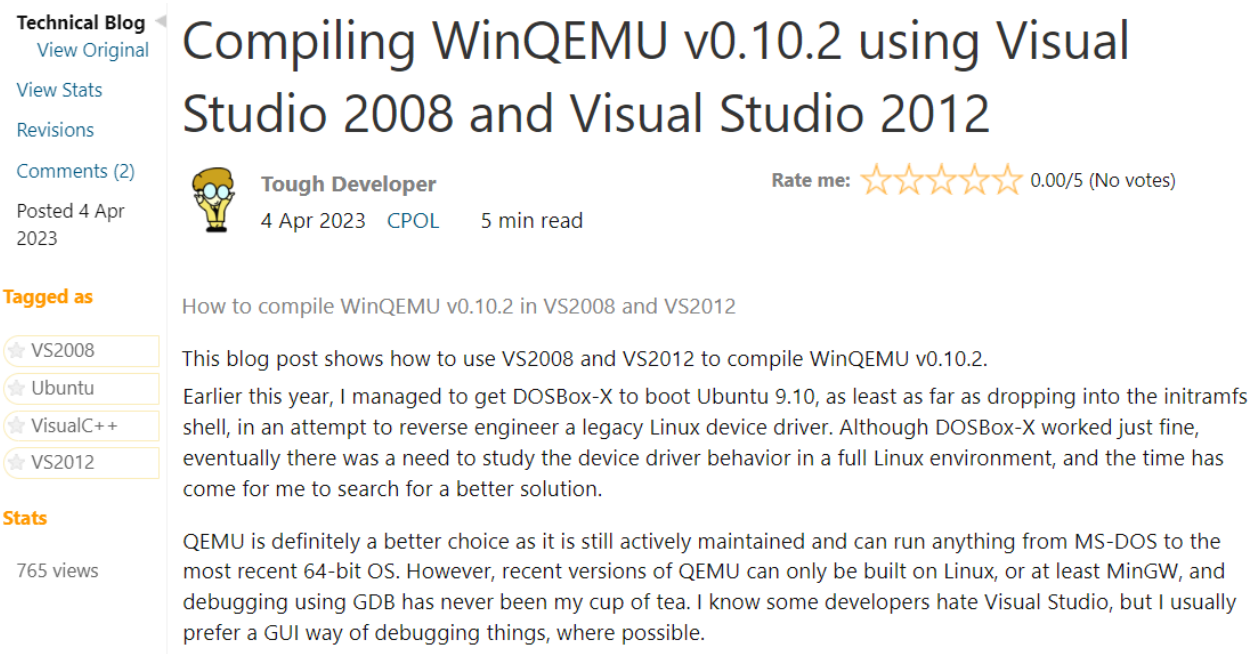
4.3.3. Робота агенту структурування інформаційного наповнення.

Основними задачами агенту є:

1. Виявлення ключових фраз текстової частини статей за допомогою словнику.
2. Наповнення ДПЗ статтями – завантаження нових та оновлення існуючих згідно алгоритму наведеному у розділі 3.2.4.
3. Аналіз структури ДПЗ за методом структурування інформаційного наповнення, що наведений у розділі 3.4.

Сховище конфігурації формування документації містить порогове значення вагомості терміну – формула 2.42 та оцінки, що забезпечують мінімальну кількість вершин та статей для формування оптимальної структури ДПЗ.

Виконання завдання 1. Виявлення ключових фраз з ваговими коефіцієнтами у статті за допомогою розробленого словнику.



Technical Blog
View Original
View Stats
Revisions
Comments (2)
Posted 4 Apr 2023

Tagged as
VS2008
Ubuntu
VisualC++
VS2012

Stats
765 views

Compiling WinQEMU v0.10.2 using Visual Studio 2008 and Visual Studio 2012

Tough Developer
4 Apr 2023 CPOL 5 min read

Rate me: 0.00/5 (No votes)

How to compile WinQEMU v0.10.2 in VS2008 and VS2012

This blog post shows how to use VS2008 and VS2012 to compile WinQEMU v0.10.2.

Earlier this year, I managed to get DOSBox-X to boot Ubuntu 9.10, as least as far as dropping into the initramfs shell, in an attempt to reverse engineer a legacy Linux device driver. Although DOSBox-X worked just fine, eventually there was a need to study the device driver behavior in a full Linux environment, and the time has come for me to search for a better solution.

QEMU is definitely a better choice as it is still actively maintained and can run anything from MS-DOS to the most recent 64-bit OS. However, recent versions of QEMU can only be built on Linux, or at least MinGW, and debugging using GDB has never been my cup of tea. I know some developers hate Visual Studio, but I usually prefer a GUI way of debugging things, where possible.

Рис.4.15 Текст допису спільноти CodeProject

Результати знайдених ключових фраз статті представлено стовпчиковою діаграмою:

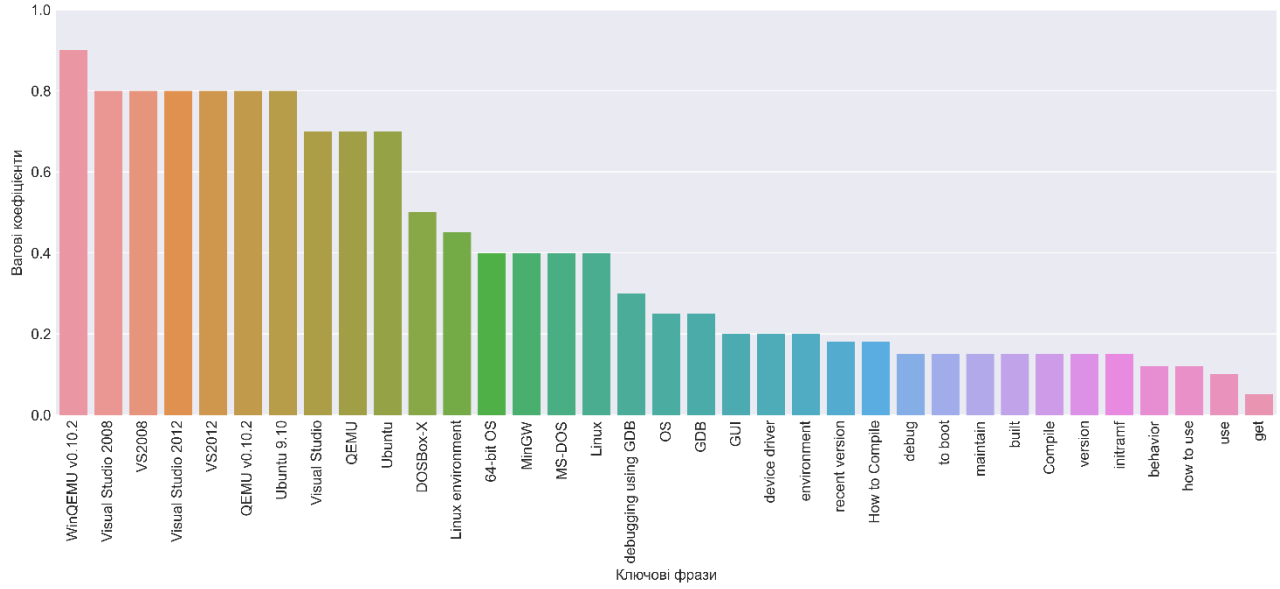


Рис.4.16 Список виявлених ключових фраз зі статті з ваговими коефіцієнтами

Список виявлених ключових фраз зі статті з ваговими коефіцієнтами у вигляді діаграми для яких, надалі можна будувати дерева прийняття рішень (розділ 2.5). Приклад терміну з ваговими коефіцієнтами наведено у словнику додатку – Додаток Ж.

Основні сутності для виконання методу автоматизованого генерування термінів допису зображено на рис. 4.17. До них відносяться:

- стаття;
- мова (тексту статті);
- ключові фрази (які належіть допису);
- терміни (отримані після виконання методу наведено в цьому розділі);
- відповідність між термінами та ключовими фразами;
- відповідність між статтею і термінами.

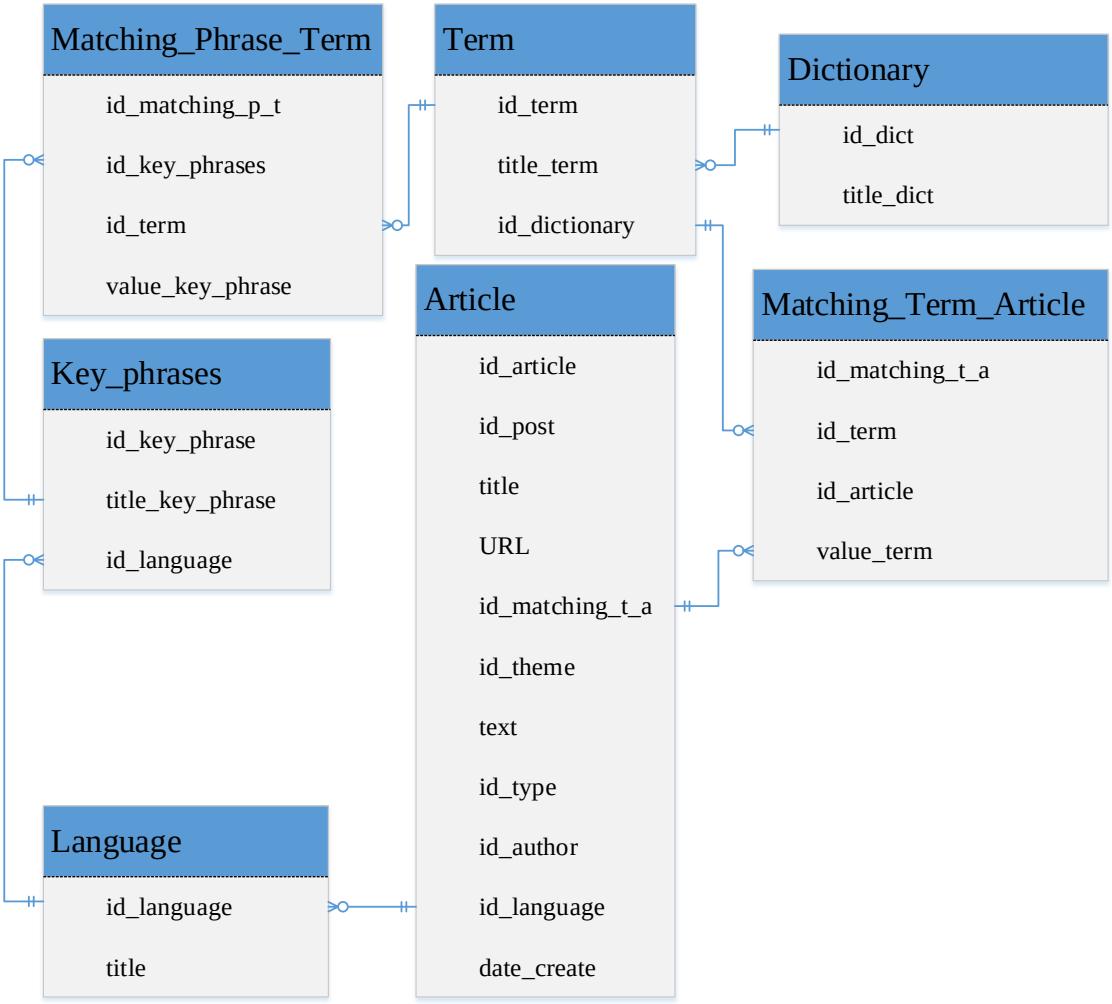


Рис.4.17 Схема даних для реалізації методу автоматизованого формування термінів дописів

Виконання завдання 2. Завантаження нових та оновлення існуючих статей у ДПЗ є подібними до завдань, коли агент збору даних здійснював завантаження

та оновлення дописів з ВС у СКД (наведено у завданні 2). Різниця полягає в сутностях та статусах з якими потрібно здійснювати роботу.

Дискусія, яка знаходиться у СКД може бути статтею ДПЗ якщо:

1. Тема, як джерело даних, є адекватною.
2. Дискусія є актуальною та авторитетною.
3. Вміст дописів дискусії є унікальним та не містить НІН.

4.3.4. Робота агенту оцінювання якості.

Даний агент перевіряє якість сформованої ДПЗ за показниками наведеними у розділі 4.1: функціональної повноти, відповідності розпізнавання та модульності. Для проведення дослідження обрано дописи спільноти CodeProject за 2018 – 1 квартал 2023 рр. (всього 21 квартал). Структуру термінів побудовано згідно класифікаторів описаних у розділі 2.2.1. Множину термінів визначено завдяки аналізу типів документів, що виникають під час ЖЦ ПЗ [120]: документу вимог (Додаток А), архітектури, експлуатації (Додаток В), тощо. Надалі сформовано терміни та ключові фрази приналежності до них (приклад терміну наведено у Додаток Ж Таблиця Ж.3).

Функціональна повнота відображає структурні прогалини ДПЗ відповідно до структури класифікаторів.

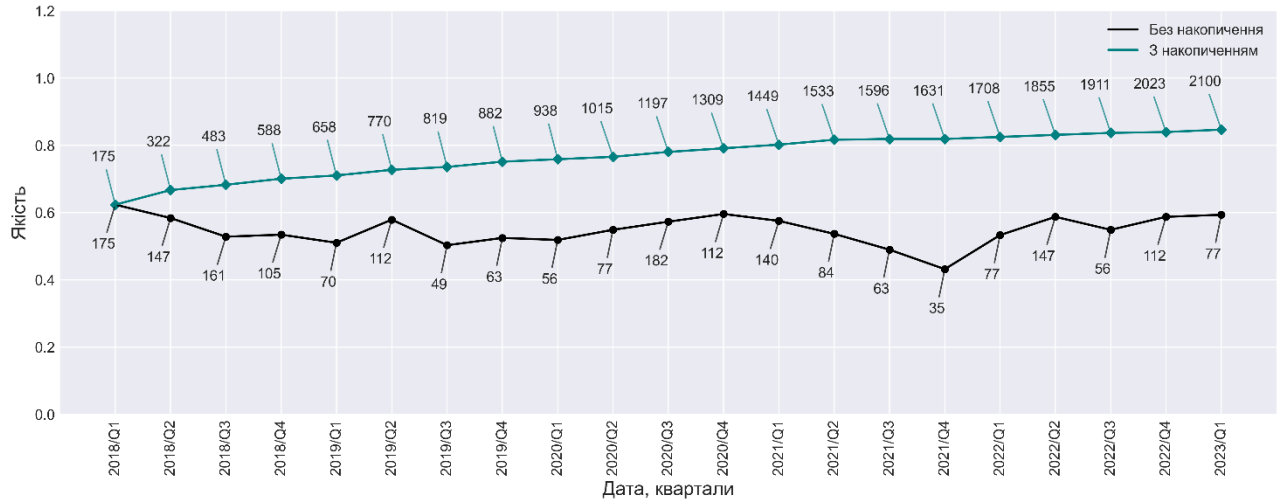


Рис.4.18 Динаміка змін під-характеристики якості «Функціональна повнота» в документації програмного забезпечення

Рис. 4.18. відображає ступінь заповнення статтями структури ДПЗ за певні періоди часу (щоквартально). На графіку відображено дві залежності:

- «з накопиченням» – враховуючи статті попередніх кварталів;
- «без накопичення» – до уваги беруться лише статті певного кварталу.

Повнота описів відображає кількісне співвідношення заповнення термінів ДПЗ статтями за певні періоди часу (щоквартально) – рис. 4.19. Терміни, що мають одну статтю переважають на протязі всіх досліджуваних кварталів.

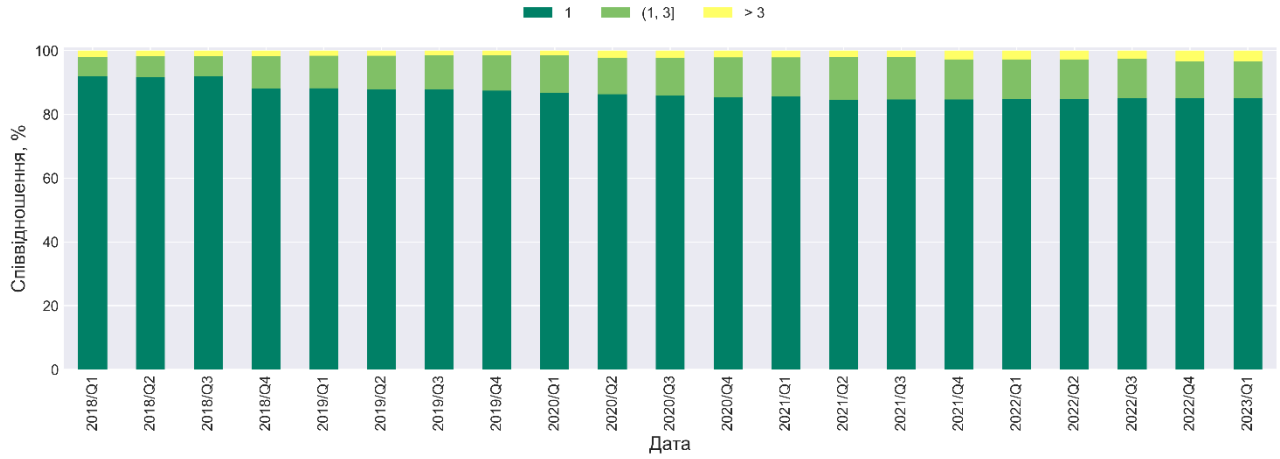


Рис.4.19 Динаміка змін під-характеристики якості «Повнота описів» в документації програмного забезпечення

Дослідження метрики «Повнота описів» проведено в залежності «з накопиченням». Отже, зі збільшенням інформаційного наповнення кількість повторень термінів збільшувалась.

Метрика достовірності відображає кількісне співвідношення термінів рівень довіри до яких є високим та низьким відповідно до рейтингу статей що їм належать. Порогове значення оцінки термінів рівень довіри до яких є низьким визначається умовою: $\alpha > 0.7$. Результат обчислення метрики достовірності відображено на рис. 4.20.

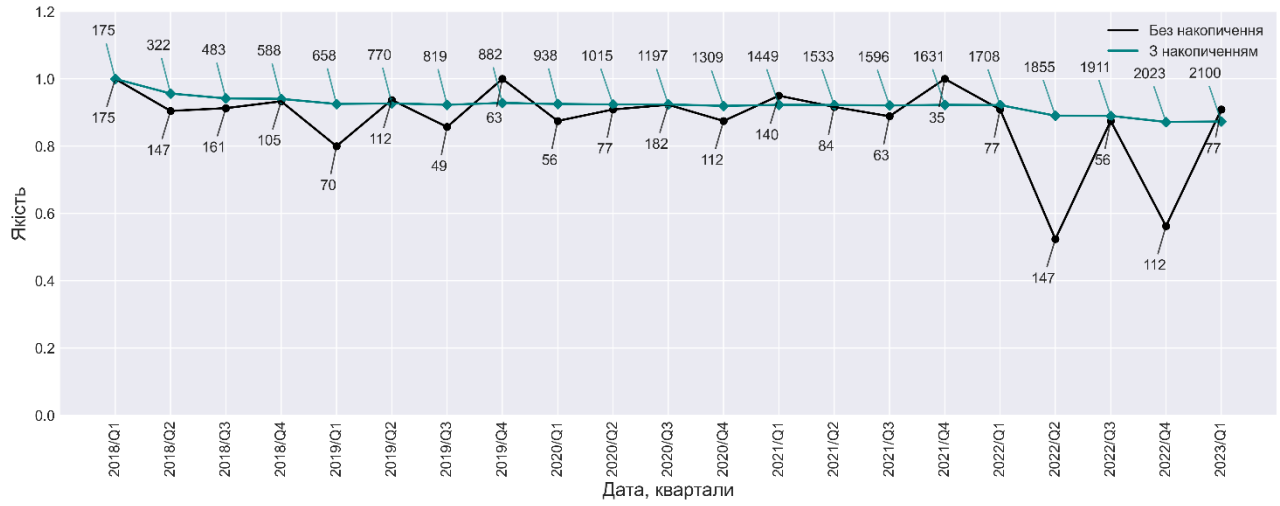


Рис.4.20 Динаміка змін під-характеристики якості «Достовірність» в документації програмного забезпечення

Зі збільшенням інформаційного наповнення – як кількості термінів, так і кількості статей в яких вони виявлені, спостерігалась тенденція зниження якості. Зниження показнику якості в певному кварталі говорить про те, що в попередньому кварталі рейтинг наявних термінів, який залежить від рейтингу статті в якому він виявлений, був більшим. Навіть при тенденції збільшення показника якості з першого по другий квартал 2019 року (залежність «без накопичення») все одно це не вплинуло на глобальне значення, а лише вдалось втримати на одному рівні (залежність «з накопиченням»).

Метрика актуальності відображає кількісне співвідношення термінів які є актуальними та неактуальними щодо досліджуваного кварталу часу. Термін актуальності статей визначається 2-ма роками або 8-ма кварталами. При чому на графіку представлено три розвитку подій:

- «без оновлення» – оновлення даних не відбувалось на протязі всього періоду;
- здійснено «перше оновлення» – перший квартал 2019 року;
- здійснено «друге оновлення» статей ДПЗ – перший квартал 2020 року.

Результат обчислення метрики актуальності відображено на рис. 4.21.

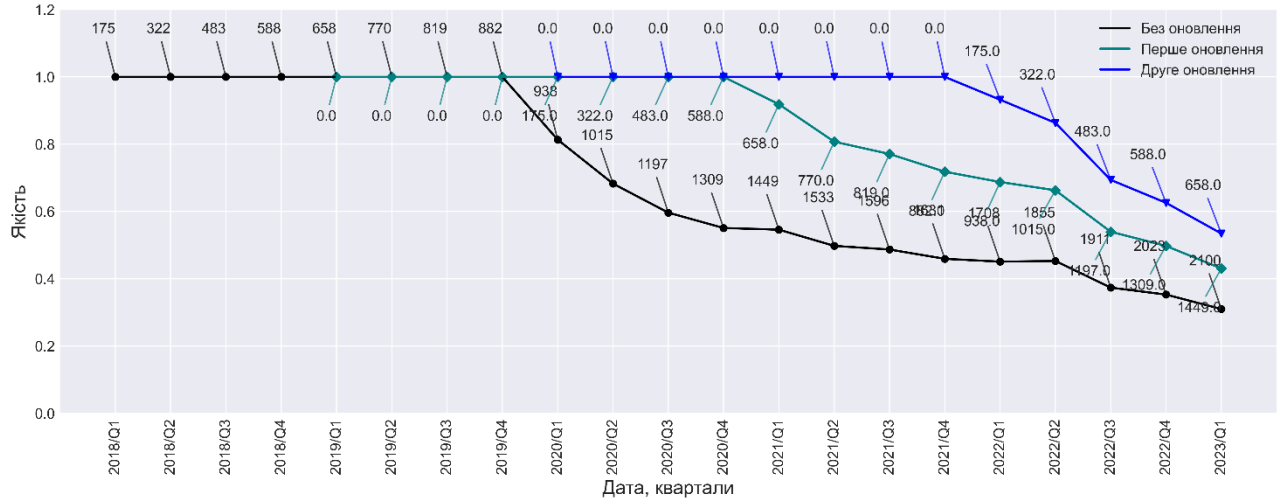


Рис.4.21 Динаміка змін під-характеристики якості «Актуальність» в документації програмного забезпечення

Отримана динаміка змін під-характеристики дозволяє визначити оптимальний час для оновлення статей ДПЗ відповідно до порогового значення актуальності, яке є адаптивним і встановлюється експертом. Для кожного разу оптимальний час оновлення інформаційного наповнення ДПЗ може змінюватися адже кількість статей є різною.

Надалі, проаналізовано якість ДПЗ за характеристикою супроводжуваність.

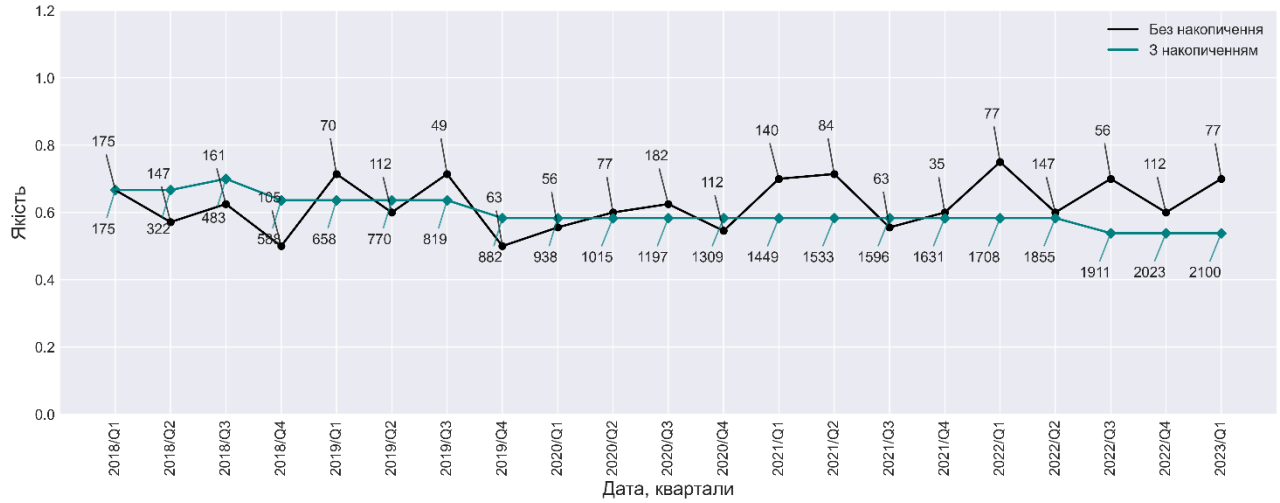


Рис.4.22 Динаміка змін під-характеристики якості «Модульність» стосовно довжини гілок документації програмного забезпечення

Аналіз якості за під-характеристикою модульність стосовно довжини проводився дотримуючись правила «трьох-кліків» що передбачає оптимальну

глибину – три або чотири. Відповідно всі гілки ДПЗ що не відповідають цьому критерію створюють дисбаланс.

Рис.4.22. показує, що структура ДПЗ стосовно глибини на протязі 21 кварталів мала достатньо хороший рівень (значення якості є більшим 0.5), але в загальному з часом і зі збільшенням інформаційного наповнення (залежність «з накопиченням») погіршувалась, не зважаючи на те, що в певні квартали в залежності «без накопичення» спостерігалось підвищення значення якості.

Надалі, здійснено аналіз щодо оптимальної ширини кожної гілки ДПЗ відповідно до умови наведеної формулою 3.10.

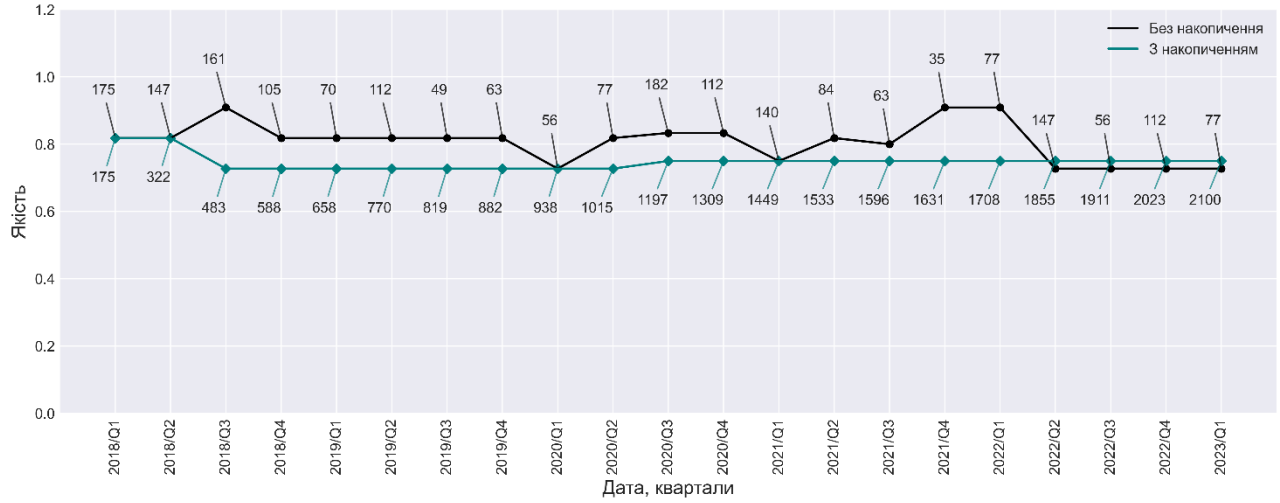


Рис.4.23 Динаміка змін під-характеристики якості «Модульність» в документації програмного забезпечення

Рис.4.23. показує, що структура ДПЗ стосовно ширини на протязі 21 кварталів мала коливання в діапазоні [0.72, 0.9] – залежність «без накопичення», але на загальну структуру це не сильно плинуло (залежність «з накопиченням»). Отже, нові статті, що з’являлись кожного кварталу не сильно впливали на попередньо визначену структуру стосовно ширини.

Надалі, здійснено аналіз щодо оптимального вагового навантаження серед гілок ДПЗ (починаючи з третього рівня) відповідно до умови наведеної формулою 3.11.

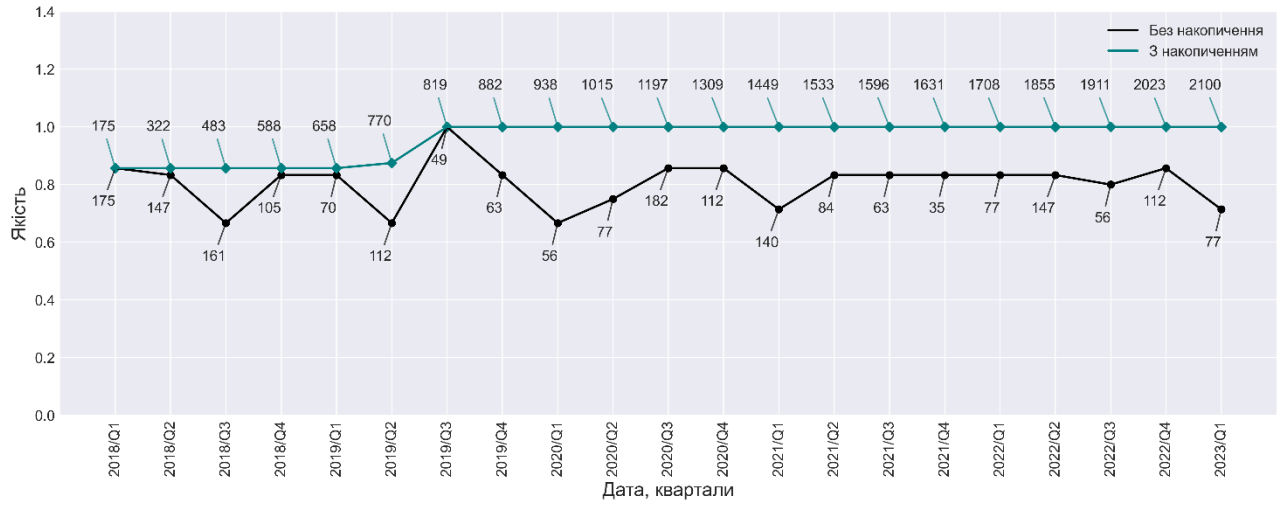


Рис.4.24 Динаміка змін під-характеристики якості «Модульність» в документації програмного забезпечення

Рис.4.24. показує, що зі збільшенням інформаційного наповнення у ДПЗ (залежність «з накопиченням») показник якості стосовно збалансованості за ваговим критерієм покращувався, і, за перші сім кварталів, досягнув значення одиниці – всі гілки були збалансовані.

Отже, дослідивши якість ДПЗ за сьома під-характеристиками майже всі з них вдалось тримати на достатньо високому рівні якості, а деякі навіть покращувати.

Окрім наведених оцінок якості, ДПЗ як інформаційний продукт можна дослідити та сформувати статистику за такими метриками:

- найавторитетніший розділ – розділ, що містить статті, автори яких отримали найвищі оцінки/рейтинг;
- найсучасніший розділ – розділ, який містить найбільшу кількість нових статей;
- найактуальніший розділ – розділ що містить найбільшу кількість статей. Відповідно такий розділ відображає потреби споживачів.
- найрейтингованіший розділ – розділ, який містить статті рейтинг яких є найбільший.

Отримана статистика стосовно розділів ДПЗ допоможе експерту прийняти рішення щодо розподілу вагових коефіцієнтів для термінів, перегляду

показників за якими формується структура документації, а також встановлення порогових значень для розроблених методів та алгоритмів.

Сховище показників якості містить порогові значення мір щодо оцінювання ДПЗ за характеристиками та під-характеристиками, наведеними у розділі 4.1, та зберігає результат статистики за метриками що призначені для експерта.

Висновки до розділу

У четвертому розділі розроблено метод оцінювання якості документації програмного забезпечення на основі метрик міжнародного стандарту ISO/IEC-25010 (функціональної придатності, придатності використання та модульності), що дозволило практично оцінити результати виконання формування документації програмного забезпечення.

Побудовано архітектуру системи формування документації програмного забезпечення за допомогою інформаційного наповнення віртуальних спільнот у вигляді багатоагентної системи, що забезпечило практичну реалізацію методів та алгоритмів у вигляді завдань агентів. Описано переваги вибору такого підходу і взаємодію агентів, як автономних та інтерактивних програмних систем, які напряму не взаємодіють, а тільки через сховище консолідованих даних та документацію. Наведено задачі кожного агенту та їх реалізація, що дозволило відобразити результати практичного впровадження запропонованих методів і алгоритмів, наведених у другому та третьому розділах.

Завдання агентів виконані за допомогою сучасних технологій автоматизованого збору та аналізу інформації, реалізовані мовою Python, що містить набір бібліотек, які дозволяють здійснювати символьну та статистичну обробку даних. Для аналізу природньої мови за наведеними методами було застосовано існуючі та розроблено нові словники для україномовного та англомовного текстів.

Розроблена архітектура системи відображає автоматизований процес формування суспільно орієнтованої документації програмного забезпечення джерелом якої постають віртуальні спільноти завдяки реалізації кожним незалежним агентом певного набору задач, а також залучення експерта, що розробляє шаблони для завантаження даних, формує словників термінів, визначає оптимальні значення оцінок та порогових значень мір, здійснює аналіз якості документації як завершеного інформаційного продукту.

Аналіз результатів досліджень показав на практиці ефективність розроблених методів і засобів формування документації програмного забезпечення за допомогою інформаційного наповнення віртуальних спільнот, а також дозволив визначити оптимальні значення для формування показників порогових значень для роботи системи.

Висновки

У дисертаційній роботі вирішено важливу наукову задачу підвищення якості формування суспільно орієнтованої документації програмного забезпечення шляхом розроблення нових та вдосконалення існуючих методів і засобів аналізу даних. При цьому отримано такі результати:

1. Проаналізовано особливості використання інформаційного наповнення віртуальних спільнот для формування документації програмного забезпечення з урахуванням сучасних тенденцій та перспектив розвитку інженерії програмного забезпечення, що підтвердило актуальність наукової задачі підвищення якості формування суспільно орієнтованої документації програмного забезпечення.
2. Вперше побудовано формальну модель документації програмного забезпечення, яка враховує особливості інформаційного наповнення віртуальних спільнот та потреби різних категорій користувачів програмного забезпечення через множину класифікаторів, що дозволило побудувати методи та алгоритми формування документації програмного забезпечення, і подальшого оцінювання за показниками якості.
3. Вдосконалено методи: автоматизованого виявлення вагомих, значущих термінів статей на основі масок ключових фраз з ваговими коефіцієнтами з використанням дерева для прийняття рішень; поєднання методу виявлення подібності та моделі N-грам (без вилучення і з вилученням незначущих, стоп-слів) для виявлення однакового тексту в дописах, що забезпечило формування якісної документації програмного забезпечення.
4. Розроблено алгоритми виявлення цінного інформаційного наповнення віртуальних спільнот, які враховують адекватність джерела за умовами релевантності запиту користувача та актуальності вмісту, структурування інформаційного наповнення документації програмного забезпечення за допомогою апарату Б-дерев, усунення небажаного контенту у дописах шляхом виявлення заборонених слів, гіперпосилань та файлів, що дало

можливість автоматизовано формувати документацію програмного забезпечення.

5. Набув подальшого розвитку метод оцінювання якості документації програмного забезпечення на основі метрик міжнародного стандарту ISO/IEC-25010 (функціональної придатності, придатності використання та модульності), що дозволило практично оцінити результати виконання формування документації програмного забезпечення.
6. Побудовано архітектуру програмного комплексу формування документації програмного забезпечення за допомогою інформаційного наповнення віртуальних спільнот у вигляді багатоагентної системи, що забезпечило практичну реалізацію методів та алгоритмів у вигляді завдань агентів.

Література

1. Адекватність [Електронний ресурс]. Редакція «Великої української енциклопедії».
2. Актуальність // Великий тлумачний словник сучасної української мови (з дод. і допов.) / уклад. і гол. ред. В. Т. Бусел. – 5-те вид. – К. ; Ірпінь : Перун, 2005. – ISBN 966-569-013-2.
3. Алексенко О. В. Технології програмування та створення програмних продуктів: конспект лекцій / О. В. Алексенко – Суми : Сумський державний університет, 2013. – С. 133.
4. Бідюк П. І. Системи і методи підтримки прийняття рішень [Електронний ресурс]: підручник для здобувачів ступеня магістра за спеціальністю 124 Системний аналіз / П. І. Бідюк, О. Л. Тимощук, А. Є. Коваленко, Л. О. Коршевнюк ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані – Київ : КПІ ім. Ігоря Сікорського 2022. – С. 610.
5. Васильченко В. Термін, термінологія / В. Васильченко // [Електронний ресурс] Цікаві лінгвістичні терміни. – Режим доступу: https://uain.press/blogs/termin-terminologiya-tsikavi-lingvistichni-termini-923680
6. Вікулова А. О. Перспективи розвитку ІТ-послуг в Україні / А. О. Вікулова, В. В. Савчук // Причорноморські економічні студії, 2020. – Вип. 51. – С. 27-32.
7. Глибовець А. М. Алгоритм токенізації та стемінгу для текстів українською мовою / А. М. Глибовець, В. В. Точицький // НАУКОВІ ЗАПИСКИ НаУКМА, 2017. – Том 198. – С. 4-8.
8. Гнатовська Г.А. Конспект лекцій: «Технологія створення програмних продуктів», Одеса, 2015. – С. 97.
9. Голидьбіна А. В. Особливості сучасного ринку ІТ-послуг та специфіка просування на ньому / А. В. Голидьбіна, Н. В. Язвінська // Економічний вісник Національного технічного університету України "Київський

політехнічний інститут", 2017. – № 14. – С. 291-298. – Режим доступу: http://nbuv.gov.ua/UJRN/evntukpi_2017_14_47

10. Дослідження Do IT Like Ukraine: IT-індустрія зростає попри все. [Електронний ресурс]. IT Ukraine Association, 2022. – Режим доступу: https://itukraine.org.ua/it-reports-do-it-like-ukraine.html
11. Дранишников Л.В. Менеджмент проектів програмного забезпечення: конспект лекцій для студентів здобувачів вищої освіти першого (бакалаврського) рівня зі спеціальності 121 Інженерія програмного забезпечення. спец. усіх форм навчання / Л.В. Дранишников // Дніпровський державний технічний університет – Кам'янське, 2019. – С.123.
12. ДСТУ 2.610:2006. Єдина система конструкторської документації. Правила виконання експлуатаційних документів.
13. ДСТУ 2392-94 (Інформація та документація. Базові поняття. Терміни та визначення): Національний стандарт України. – К. : Держстандарт України, 1994. – С. 25.
14. ДСТУ 2732:2004 (Діловодство й архівна справа. Терміни та визначення понять): Національний стандарт України. – К. : Держстандарт України, 2005. – С. 36.
15. ДСТУ 2844-94 (Програмні засоби ЕОМ. Забезпечення якості. Терміни та визначення): Національний стандарт України. – К. : Держстандарт України, 1995. – С. 22.
16. ДСТУ 2850-94 (Програмні засоби ЕОМ. Забезпечення якості. Показники та методи оцінювання якості програмного забезпечення): Національний стандарт України. – К. : Держстандарт України, 1996. – С. 42.
17. ДСТУ 3278-95 (Система розроблення та поставлення продукції на виробництво, основні терміни та визначення). – Держстандарт України, 1995.

18. ДСТУ 3973-2000. Система розроблення та поставлення продукції на виробництво. Правила виконання науково-дослідних робіт. Загальні положення.
19. ДСТУ 4163:2020 Уніфікована система організаційно-розпорядчої документації. Вимоги до оформлення документів. [Електронний ресурс]. – Режим доступу: https://zakon.help/files/article/11494/%D0%94%D0%A1%D0%A2%D0%A3%204163_2020.pdf
20. ДСТУ 4302:2004 Інформаційні технології. Настанови щодо документування комп'ютерних програм (ISO/IEC 6592:2000 МОД), 2004.
21. ДСТУ 6096:2009 Система стандартів з інформації, бібліотечної та видавничої справи. Універсальна десяткова класифікація. Структура, правила ведення та індексування, 2009.
22. ДСТУ ISO/IEC 12119-2003 (Інформаційні технології. Пакети програм. Тестування і вимоги до якості. ISO/IEC 12119:1994, IDT): Національний стандарт України. – К. : Держспоживстандарт України, 2004. – С. 26.
23. ДСТУ ISO/IEC/IEEE 16326:2015 Розроблення систем та програмного забезпечення. Процеси життєвого циклу. Керування проектами (ISO/IEC/IEEE 16326:2009, IDT).
24. Жежнич П. І. Консолідовані інформаційні ресурси баз даних та знань: навчальний посібник / П.І. Жежнич. – Львів: видавництво Національного університету "Львівська політехніка", 2010. – С. 212.
25. Жежнич П.І. Технології інформаційного менеджменту: Навчальний посібник – Львів: Видавництво Національного університету "Львівська політехніка", 2010. – С. 260.
26. Загальна характеристика і класифікація програмного забезпечення і базових технологій управління інформаційними [Електронний ресурс].
27. Закон України "Про електронні документи та електронний документообіг": [Електронний ресурс] // Відомості Верховної Ради України. – №851-IV від

22.05.2003 у редакції від 06.11.2014. – https://zakon.rada.gov.ua/laws/show/851-15#Text

28. Закон України "Про інформацію": [Електронний ресурс] // Відомості Верховної Ради України. – №2658-XII від 02.10.92 у редакції від 02.06.2016. – http://zakon3.rada.gov.ua/laws/show/2657-12.
29. Закон України «Про стимулювання розвитку сфери інформаційних технологій в Україні»: [Електронний ресурс] // Відомості Верховної Ради України. — http://search.ligazakon.ua/l_doc2.nsf/link1/JI03715A.html
30. Закон України «Про захист інформації в інформаційно-комунікаційних системах»: [Електронний ресурс] // Відомості Верховної Ради України. – №80/94-ВР від 22.05.2003 у редакції від 05.07.1994. – https://zakon.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80#Text
31. За рік ІТ-ФОПів стало на 16% більше – дослідження. [Електронний ресурс] Українська правда, 2021. – Режим доступу: https://www.epravda.com.ua/news/2021/04/12/672899/
32. Замула І. В. ІТ-послуга: поняття та види для облікових цілей / І. В. Замула, Л. В. Чижевська, І. Л. Грабчук // Проблеми теорії та методології бухгалтерського обліку, контролю і аналізу, 2021. – №2(49), – С. 29–33. – DOI: https://doi.org/10.26642/pbo-2021-2(49)-29-33
33. Інформація: поняття, властивості, форми, безпека та конфіденційність [Електронний ресурс]. – Режим доступу: https://www.kursak.com/informatsiia-poniattia-vlastyvosti-formy-bezpeka-ta-konfidentsiynist/#22
34. Карена М. Доуніверситетський посібник з математики / М. Карена // Національний університет Літораль, 2019.
35. Катренко А. В. Теорія прийняття рішень : підручник з грифом МОН / А. В. Катренко, В. В. Пасічник, В. П. Пасько – К. : Видавнича група BHV, 2009. – С.448.

36. Колєчкіна Л. М. Розробка методу і алгоритму перевірки тексту на унікальність / Л. М. Колєчкіна, О. П. Пухтєєва // Науковий вісник КУЕІТУ, Нові технології, 2013. – № 1-2. – С. 58-62. – Режим доступу: http://dspace.puet.edu.ua/bitstream/123456789/4623/1/Rozr1707.pdf
37. Конституція України: [Електронний ресурс] Відомості Верховної Ради України (ВВР), 1996. – № 30. – С. 141. – Режим доступу: http://zakon5.rada.gov.ua/laws/show/254%D0%BA/96-%D0%B2%D1%80
38. Користування інтернетом серед українців: результати телефонного опитування, проведеного 13-18 травня 2022 року. [Електронний ресурс]. Режим доступу: https://kiis.com.ua/?lang=ukr&cat=reports&id=1115&page=12
39. Костенко Н. Досвід контент-аналізу: моделі та практики / Н. Костенко, В. Іванов // Монографія. – К.: Центр вільної преси, 2003. – С. 44.
40. Кунанець Н.Е. Вступ до фаху “Консолідована інформація” / Н.Е. кунанець, В.В. Пасічник // Навчальний посібник. Друге видання. Львів: Видавництво Львівської політехніки, 2013. – С.196.
41. Логарифмічна функція: властивості, приклади, вправи [Електронний ресурс]. WARBLETONCOUNCIL, 2019.
42. Лункіна Т. І. Методи управління ризиками споживчого кредитивання / Т. І. Лункіна, К.О. Вельховацька // Миколаївський національний аграрний університет, «Young Scientist», 2015. – № 2 (17). – С.157-160.
43. Литвин В.В.. Проектування інформаційних систем: навч. посіб. / В. В. Литвин, Н.Б. Шаховська. – Львів: Вид-во Нац. ун-ту “Львівська політехніка”, 2010. – 210 с.
44. Любов. [Електронний ресурс] Академічний тлумачний словник української мови. Режим доступу: http://sum.in.ua/s/ljubov
45. Марків О.О. Лінгвістичне забезпечення формування туристичної документації на основі відкритих веб-ресурсів: Дисертація на здобуття наукового ступеня к.т.н. – Львів: Національний університет "Львівська політехніка", 2017. – С. 188.

46. Марковець О.В. Формування якісної технічної документації до програмного забезпечення / О.В. Марковець, А.І. Синько // Вісник ВПІ, 2021. – Вип. 2. – С. 98–106. – DOI: https://doi.org/10.31649/1997-9266-2021-155-2-98-106
47. Мастикаш О.В. Особливості класифікації та аналізу віртуальних спільнот / О.В Мастикаш // Вісник Хмельницького національного університету. Технічні науки. 2017. №6. С.165-168. – Режим доступу: http://nbuv. gov. ua/UJRN/Vchnu_tekh_2017_6_28.
48. Метрика TF-IDF (Term frequency–inverse document frequency) [Електронний ресурс]. Loginom. Режим доступу: https://wiki.loginom.ru/articles/tf-idf.html
49. Мулеса О.Ю. Інформаційні системи та реляційні бази даних / О.Ю. Мулеса, Я.В. Варга, навч. посібник. – Ужгород, 2023. – С.132.
50. Несхвалення. [Електронний ресурс] Академічний тлумачний словник української мови. Режим доступу: http://sum.in.ua/s/neskhvalennja
51. Обсяг українського IT експорту вперше перетнув позначку $5 млрд за рік: [Електронний ресурс] // IT Ukraine Association, 2021. – Режим доступу: https://itukraine.org.ua/ukrainian-it-exports-exceed-$5-billion-in-a-year-for-the-first-time.html
52. Паздерська Р.С. Класифікація віртуальних спільнот / Р.С. Паздерська, О.В. Марковець // Вісник Хмельницького національного університету. Технічні науки. – 2021. № 1. – С.37-44. – Режим доступу: http://journals.khnu.km.ua/vestnik/wp-content/uploads/2021/08/9-1.pdf
53. Пелещишин А. М. Аналіз існуючих типів віртуальних спільнот у мережі Інтернет та побудова моделі віртуальної спільноти на основі веб-форуму / А. М. Пелещишин, Р. Б. Кравець, Ю. О. Сєров // Вісник Національного університету "Львівська політехніка". – 2011. – № 699 : Інформаційні системи та мережі. – С. 212-221.
54. Перелік Національних стандартів України для створення, впровадження та супроводження автоматизованих і інформаційних систем [Електронний

ресурс]. Національна бібліотека України імені В.І. Вернадського, 2014. – Режим доступу: http://nbuv.gov.ua/node/1469

55. Перші кроки в NLP: розглядаємо Python-бібліотеку NLTK в реальному завданні [Електронний ресурс] – Режим доступу: https://dou.ua/lenta/articles/first-steps-in-nlp-nltk/
56. Поляруш І.М. Скоринг, як вдосконалений механізм оцінки потенційного позичальника банком – демонстрація процесу обробки даних / І.М. Поляруш // Ефективна економіка, 2015. – № 11.
57. Поморова О.В. Аналіз методів та засобів оцінки якості програмних систем / О.В. Поморова, Т.О. Говорущенко // Радіоелектронні і комп'ютерні системи, 2009. – № 6 (40). – С. 148-158.
58. Прикладне програмне забезпечення: Навч.-метод. посібник для самост. вивч. дисц. Молодцова О. П. – К.: КНЕУ, 2000. – С. 264 –ISBN 966-574-098-9.
59. Проект Закону України «Про концепцію національної інформаційної політики» [Електронний ресурс] // Відомості Верховної Ради України. – №687-IV від 03.04.2003 – https://zakon.rada.gov.ua/laws/show/687-15#Text
60. Релевантність та пертинентність [Електронний ресурс]. Територія методиста, 2019.
61. Ришковець Ю. В. Метод оцінювання якості мультимедійних веб-систем / Ю. В. Ришковець // Радиоэлектроника и информатика, 2014. – № 1. – С. 53-57. – Режим доступу: http://nbuv.gov.ua/UJRN/reii_2014_1_13
62. Рожко Р. Автоматизований аналіз мотиваційних листів на основі сентимент аналізу [Електронний ресурс] / Р. Рожко // Курсова робота. – Київ: Національний університет «Києво-могилянська академія», 2020. – С. 34. Режим доступу: https://ekmair.ukma.edu.ua/handle/123456789/18229
63. Сєров Ю.О. Методи та засоби побудови ефективних віртуальних спільнот на основі Веб-форумів: Дисертація на здобуття наукового ступеня к.т.н. – Львів: Національний університет "Львівська політехніка", 2010. – 166 с.

64. Скрипін В. Експорт IT-послуг в Україні з початку року зріс на 16% (до $4,9 млрд) — в серпні на IT припало 48% загального обсягу експорту / В. Скрипін // [Електронний ресурс], 2022. – Режим доступу: https://itc.ua/ua/novini/eksport-it-poslug-v-ukrayini-z-pochatku-roku-zris-na-16-do-4-9-mlrd-v-serpni-na-it-pripalo-48-zagalnogo-obsyagu-eksportu/
65. Синько А. Аналіз архітектури – «моделі представлення 4+1» / А. Синько // Інформація, комунікація, суспільство 2021 [Електронний ресурс] : Матеріали 10-ї Міжнародної наукової конференції ICS-2021. – Львів: Видавництво Львівської політехніки, 2021. – С.19-20. – Режим доступу: http://ics.skid-lp.info/ics_2021.pdf. ISBN 978-966-941-592-9
66. Синько А. І. Аналіз якості документації програмного забезпечення / А.І. Синько // Scientific research and innovation: Матеріали II-ої Міжнародної науково-практичної інтернет-конференції, 3–4 квітня, Дніпро, WayScience. –2023. – С. 344–345.
67. Синько А. Архітектура системи формування документації програмного забезпечення за допомогою віртуальних спільнот/ А.Синько // Вісник Хмельницького національного університету. Технічні науки. – 2023. № 1. – С.248-252. – DOI: https://doi.org/10.31891/2307-5732-2023-317-1-248-252
68. Синько А. Метод автоматизованого виявлення термінів статей за допомогою дерева для прийняття рішень / А.Синько, П. Жежнич // Вісник Хмельницького національного університету. Технічні науки. – 2023. № 2. – С.339-344.
69. Синько А. І. Представлення типів віртуальних спільнот, що можуть бути джерелом документації до програмного забезпечення / А.І. Синько // Інформація та соціум: збірник матеріалів VII Міжнародної науково-практичної конференції (Вінниця, 3 червня 2022 р.), 2022. – С. 26–27.
70. Синько А. Побудова формальної моделі документації до програмного забезпечення / А. Синько // Інформація, комунікація, суспільство 2022:

Матеріали 11-ї Міжнародної наукової конференції ICS-2022. – Львів: Видавництво Львівської політехніки, 2022. – С.47-48.

71. Синько А. І. Суспільно орієнтована документація програмного забезпечення // Integration of education, science and business in modern environment: winter debates: IV Міжнародна науково-практична інтернет-конференція, 23–24 лютого, 2023, Дніпро / WayScience. – 2023. – С. 259–260.
72. Трач О.Р. Математичне та програмне забезпечення організації життєвого циклу віртуальних спільнот: Дисертація на здобуття наукового ступеня к.т.н. – Львів: Національний університет "Львівська політехніка", 2018. – 172с.
73. Український тональний словник. [Електронний ресурс] Режим доступу: https://github.com/lang-uk/tone-dict-uk
74. Уподобання. [Електронний ресурс] Академічний тлумачний словник української мови. Режим доступу: http://sum.in.ua/s/upodobannja
75. Шамрай В. Релевантність існуючих пошукових систем інформації / В. Шамрай // XX Міжнародна науково-технічна конференція “Фізичні процеси та поля технічних і біологічних об’єктів”: матеріали конференції. – Кременчук: КрНУ, 2021. – С.96-98
76. Шаховська Н.Б. Аналіз інформаційних систем оброблення даних туристичної сфери / Н.Б. Шаховська, Д.І. Угрин // Науковий вісник НЛТУ України: зб. наук.-техн. праць. – Львів : НЛТУ України. – 2008. – Вип. 18.10. – С. 258–263.
77. Шаховська Н. Б. Метод аналізу відгуків клієнтів з природномовних текстів / Н. Б. Шаховська, Х. Р. Шаховська // Штучний інтелект, 2018. – № 3. – С. 18-26. – Режим доступу: http://nbuv.gov.ua/UJRN/II_2018_3_4
78. Шмиг Р. А. Технічна документація // Термінологічний словник-довідник з будівництва та архітектури / Р. А. Шмиг, В. М. Боярчук, І. М. Добрянський, В. М. Барабаш; за заг. ред. Р. А. Шмига. – Львів, 2010. – С. 192. – ISBN 978-966-7407-83-4.

79. Які бувають типи сайтів. [Електронний ресурс]. ИнтерТеам, 2022. Режим доступу: https://interteam.com.ua/uk/jaki-buvajut-tipi-sajtiv/
80. Яремко В.С. Огляд наявних мультиагентних систем для задач інтелектуального аналізу даних / В.С. Яремко // Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського». Інформатика, обчислювальна техніка та автоматизація, 2018. – Том 29 (68) ч.2. №3. – С.47-55. – Режим доступу: https://journals.indexcopernicus.com/api/file/viewByFileId/633605.pdf
81. 63% людей зараз онлайн. [Електронний ресурс]. Великий звіт Digital 2022 про користувачів інтернету, 2022. – Режим доступу: https://ain.ua/2022/04/30/zvit-digital-2022/
82. Antonyuk N. Consolidated Information Web Resource for Online Tourism Based on Data Integration and Geolocation / N. Antonyuk, A. Vysotsky, V. Vysotska, V. Lytvyn, Y. Burov, A. Demchuk, I. Lyudkevych, L. Chyrun, S. Chyrun, I. Bobyk // Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2019. – P. 15-20. DOI: 10.1109/STCCSIT.2019.8929790
83. Bansiya J. Hierarchical Model for Object-Oriented Quality Assessment / J. Bansiya, C. Davis // IEEE Transactions on Software Engineering, 2002. – Vol. 28, issue 1. – P. 4-17.
84. Boehm B. W. Characteristics of Software Quality / B. W. Boehm, J. R. Brown, H. Kaspar, M. Lipow, G. McLeod, M. Merritt // North Holland, 1978.
85. Dromey G. R. A model for software product quality / G. R. Dromey // IEEE Trans. on software Eng., 1995. – Vol. 21, no. 2. – P. 146-162.
86. Elbagir S. Twitter Sentiment Analysis Using Natural Language Toolkit and VADER Sentiment / S. Elbagir, J. Yang// Proceedings of the International MultiConference of Engineers and Computer Scientists 2019 (IMECS 2019), Hong Kong, March 13-15, 2019. – P 12-16. ISBN: 978-988-14048-5-5.

Available: https://www.iaeng.org/publication/IMECS2019/IMECS2019_pp12-16.pdf

87. Ghezzi C. Fundamental of software Engineering / C. Ghezzi, M. Jazayeri, D. Mandrioli // Prentice Hall, 1991.
88. Guide to the Software Engineering Body of Knowledge [Electronic resource] Version 3.0, IEEE Computer Society, 2014. Available at: www.swebok.org.
89. Hutto C.J. VADER Sentiment Analysis, 2022 [Electronic resource] Available: https://github.com/cjhutto/vaderSentiment
90. Jiang W. Understanding graph–based trust evaluation in online social networks: Methodologies and challenges / W. jiang, G. Wang, Md Z.A. Bhuiyan, J.Wu // ACM Computing Surveys (CSUR), 2016. – Vol 49 (1). – P. 1-35. DOI: https://doi.org/10.1145/2906151
91. International standard IEEE 1219-1993. Standard for Software Maintenance, Software Engineering Standards Subcom-mittee of the IEEE Computer Society, 1993. – P. 45.
92. Information and communication technologies (ICT). [Electronic resource]. Access mode: http://uis.unesco.org/en/glossary-term/ information-and-communication-technologies-ict
93. International Standard ISO 9126-1. Software engineering – Product quality, part 1: Quality, 2001. – P. 32.
94. ISO/IEC 25010:2011 Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models, 2011. – P.34.
95. Izonin I. A Two-Step Data Normalization Approach for Improving Classification Accuracy in the Medical Diagnosis Domain / I. Izonin; R.Tkachenko; N. Shakhovska; B. Ilchyshyn; K.K. Singh // Mathematics 2022. – Vol. 10. – P.19-42. Available at: https://doi.org/10.3390/math10111942
96. Koch R. The 80/20 Principle: The Secret to Achieving More with Less / R. Koch // Paperback, 1999. – P. 336.

97. Kravets P. Promoting training of multi-agent systems / Kravets P., Lytvyn V., Vysotska V., Burov Y. // CEUR Workshop Proceedings, 2020. – Vol.2608. – P. 364-378. Available at: https://ceur-ws.org/Vol-2608/paper28.pdf
98. Lawrenz S. The Significant Role of Metadata for Data Marketplaces / S. Lawrenz, P. Sharma, A. Rausch // International Conference on Dublin Core and Metadata Applications, 2020. – P. 95–101. Available at: https://dcpapers.dublincore.org/pubs/article/view/4249
99. Littlejohn S. W. Theories of human communication / S. W. Littlejohn. – Belmont, CA: Wadsworth, 2002.
100. Lytvyn V. Development of the quantitative method for automated text content authorship attribution based on the statistical analysis of N-grams distribution / V. Lytvyn, V. Vysotska, I. Budz, Y. Pelekh, N. Sokulska, R. Kovalchuk, L. Dzyubyk, O. Tereshchuk, M. Komar, // Eastern-European Journal of Enterprise Technologies, 2019. – Vol. 6(2-102). – P. 28-51. DOI: 10.15587/1729-4061.2019.186834
101. Lytvyn V. Identifying Textual Content Based on Thematic Analysis of Similar Texts in Big Data / V. Lytvyn, V. Vysotska, I. Peleshchak, T. Basyuk, V. Kovalchuk, S. Kubinska, L. Chyrun, B. Rusyn, L. Pohreliuk, T. Salo // Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2019. – P. 84-91. DOI: 10.1109/STC-CSIT.2019.8929808
102. Manifesto for Agile Software Development (2011). [Electronic resource]. Available at: http://agilemanifesto.org/
103. McCall J. A. Factors in Software Quality / J. A. McCall, P. K. Richards, G. F. Walters, // Nat'l Tech. Information Service, 1977. – Vol. 1, 2, 3.
104. Miller K. Communication Theories: Perspectives, processes, and contexts / K. Miller. – New York: McGraw-Hill, 2005.
105. Natural Language Toolkit [Electronic resource]. Available at: https://www.nltk.org/

106. Orkphol K. Word Sense Disambiguation Using Cosine Similarity Collaborates with Word2vec and WordNet / K. Orkphol, W. Yang // Information Security Research Center, College of Computer Science and Technology, Harbin Engineering, 2019. - №11(5). DOI: https://doi.org/10.3390/fi11050114
107. Paulo-Santos A. Determining the Polarity of Words through a Common Online Dictionary / A. Paulo-Santos, C. Ramos, C. N. Marques // Proceedings by 15th Portuguese Conference on Artificial Intelligence, EPIA. – Lisbon, Portugal, 2011. – P. 649-663.
108. Reddy S. Universal semantic parsing / S. Reddy, O. Tackstrom, M. Steedman, M.lapata // arXiv preprint, 2017. DOI: https://doi.org/10.48550/arXiv.1702.03196
109. Renninger K. A. Building Virtual Communities: Learning and Change in Cyberspace (Learning in Doing: Social, Cognitive and Computational Perspectives) / K. A. Renninger, W. Shumar. – 2002. – P. 414.
110. Riley J. Understanding metadata what is metadata, and what is it for? / J.Riley // National Information Standards Organization (NISO), 2004
111. Schütze H. Introduction to information retrieval / H. Schütze, C. D. Manning; P. Raghavan// Cambridge, UK: Cambridge University Press, 2008. ISBN 0-521-86571-9.
112. Sivarajah U. Critical analysis of Big Data challenges and analytical methods / U. Sivarajah, M. M. Kamal, Z.Irani, V.Weerakkody // Journal of Business Research, 2017. – Vol. 70. – P. 263-286. DOI: https://doi.org/10.1016/j.jbusres.2016.08.001
113. Shakhovska N. Online Community Information Model for Use in Marketing Activities / N. Shakhovska; O. Peleshchyshyn; Z. Myna, T. Bilushchak // In Proceedings of the COAPSN, Lviv, Ukraine, 16–17 May 2019. – Vol. 2392. – P.263-272.
114. Schoberth T. Online Communities: A Longitudinal Analysis of Communication Activities [Electronic resourse] / T. Schoberth, J. Preece, A. Heinzl // Proc HICSS.

– 2003. – Mode of access: http://www.hicss.hawaii.edu/ HICSS36/HICSSpapers/IN. – Last access: 2009 – Title from the screen.

115. Sonali V. A comparative study of various ETL process and their testing techniques in data warehouse / V. Sonali, P. Vaishnav // Journal of Statistics and ManagementSystems, 2017. – Vol. 20(4) – P. 753-763. DOI: 10.1080/09720510.2017.1395194
116. Sommerville I. Software Engineering / I. Sommerville // Pearson, 10th Edition, 2015. – P. 811.
117. Stephenson B. Dictionaries / B. Stephenson // In: The Python Workbook. Texts in Computer Science. Springer, Cham, 2019. DOI: https://doi.org/10.1007/978-3-030-18873-3_6
118. Synko A. Application of clusterization for analysis of virtual community users / A. Synko, K. Molodetska // CEUR Workshop Proceedings: Proceedings of the Symposium on information technologies & applied sciences (IT&AS 2021), Bratislava, Slovak Republic, March 5, 2021. – Vol. 2824 – P. 9-19. – Available at: https://ceur-ws.org/Vol-2824/paper2.pdf
119. Synko A. The method of trust level of publications hosted in virtual communities. / A.Synko // Scientific Journal of TNTU (Tern.), 2022. – Vol. 105. – no 1. P. 68–79. DOI: https://doi.org/10.33108/visnyk_tntu2022.01.068
120. Synko A. Software development documenting – documentation types and standards / A. Synko, A. Peleshchyshyn // Scientific Journal of the Ternopil National Technical University, 2020. – №2 (98). – P.120-128. DOI: https://doi.org/10.33108/visnyk_tntu2020.02
121. Statista. Information technology (IT) spending on enterprise software worldwide, from 2009 to 2023: [Electronic resource]. Access mode: https://www.statista.com/statistics/203428/total-enterprise-software-revenue-forecast/
122. Fedushko S. Complex Model for Personal Data Management of Online Project Users / S. Fedushko, O. Mastykash, Y. Syerov, Y. Kalambet // In: Hu Z.,

Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education IV. ICCSEEA 2021. Lecture Notes on Data Engineering and Communications Technologies, Springer, 2021. – Vol. 83. – P. 256-269.

123. Fedushko S. Determination of the account personal data adequacy of web-community member / S. Fedushko, Yu. Syerov, A. Peleschyshyn, R. Korzh // International Journal of Computer Science and Business Informatics (IJCSBI), 2015. – Vol. 15, No. 1. – P. 1-12.
124. Fedushko S., Modelling the Behavior Classification of Social News Aggregations Users / S. Fedushko, O. Trach, Z. Kunch, Y. Turchyn, U. Yarka // CEUR Workshop Proceedings: Proceedings of the 1st International Workshop on Control, Optimisation and Analytical Processing of Social Networks (COAPSN-2019), 2019. – Vol 2392. – P. 95–110.
125. Ukraine IT report 2021: [Електронний ресурс] // IT Ukraine Association, 2021. – Режим доступу: https://reports.itukraine.org.ua/
126. Vysotsky A. Online Tourism System for Proposals Formation to User Based on Data Integration from Various Sources / A. Vysotsky, V. Lytvyn, V. Vysotska, D. Dosyn, I. Lyudkevych, N. Antonyuk, O. Naum, A. Vysotskyi, L. Chyrun, O. Slyusarchuk // Proceedings of the International Conference on Computer Sciences and Information Technologies, CSIT, 2019. – P. 92-97. DOI: 10.1109/STCCSIT.2019.8929849
127. Wankhade M. A survey on sentiment analysis methods, applications, and challenges / M. Wankhade, A. C. S. Rao, C. Kulkarni // Artificial Intelligence Review, 202. – C. 1–50.
128. Werder K. Towards a Software Product Industry Classification / K.Werder, H. Wang // New Trends in Software Methodologies, Tools and Techniques. H. Fujita, G. A. Papadopoulos, IOS Press, 2016. – ISBN 978-1-61499-674-3. DOI: 10.3233/978-1-61499-674-3-27

129. Wigers K. Software Requirements Third Edition / K. Wigers, J. Beatty // International Institute of Business Analysis, 2013. – P.672.
130. Zafirópulos Y. Dictionaries / Y.Zafirópulos // In: Swift 4 Recipes. Apress, Berkeley, CA, 2019. DOI: https://doi.org/10.1007/978-1-4842-4182-0_9
131. Zhao H. Pyramid scene parsing network / H. Zhao, J. Shi, X. Qi, X. Wang, J.Jia // Proceedings of the IEEE conference on computer vision and pattern recognition, 2017. – P. 2881–2890. DOI: https://doi.org/10.48550/arXiv.1612.01105

Додаток А. Приклад змісту документу вимог ПЗ

Область знань щодо вимог до програмного забезпечення стосується виявлення, аналізу, специфікації та перевірки вимог до програмного забезпечення, а також управління вимогами протягом усього життєвого циклу програмного продукту. Серед дослідників та практиків галузі широко визнано, що проекти програмного забезпечення є критично вразливими, коли діяльність, що пов’язана з вимогами, виконується погано, тому формування документу вимог є важливим .

У загальному випадку під вимогами до ПЗ розуміють властивості та специфікації, які повинна мати система для виконання запропонованих замовником функцій [88]. Прикладами таких функцій можуть бути бізнес-функції, документообіг, керування даними і структурою інформації, що необхідна для прийняття системних рішень, та ін.

До основних типів вимог належать [129]:

- вимоги до продукту (охоплюють умови зацікавлених користувачів щодо зовнішнього поводження системи і погляди розробників на деякі параметри системи);
- вимоги до ПЗ (системні, функціональні і нефункціональні вимоги);
- вимоги користувачів (задаються умовами досягнення цілей і задач у вигляді сценаріїв, прецедентів та містять вимоги споживачів до спектра розв’язуваних майбутньою системою задач);
- вимоги до атрибутів якості (це деякі обмеження на властивості функцій або системи, важливі для користувачів або розробників);
- специфікація функціональних вимог (опис функцій та їхніх властивостей, які не містять у собі протиріч і виключень).

Всі ці типи вимог утворюють документацію вимог до ПЗ та реалізуються на трьох рівнях (рис.А.1) :

- рівень бізнес-вимог;
- вимог користувачів;

- функціональних вимог.

Структуру документу вимог до ПЗ наведено у таблиця 1.

Таблиця А.1 Структура документу вимог ПЗ

Резюме (до 250 слів)
Зміст
Статус документу (дати, автори, компанії і т. д)
Зміни, які зроблені після останньої версії програми
1. Вступ
1.1. Ціль (мета)
1.2. Контекст
1.3. Терміни, скорочення, абревіатури
1.4. Посилання
1.5. Короткий огляд програми
2. Загальний опис
2.1. Зручність роботи з програмою
2.2. Загальні характеристики
2.3. Правила користувача
2.4. Умови роботи над проектом
2.5. Припущення та взаємозв’язки
3. Вимоги (модель системи)
3.1. Функціональні вимоги
3.2. Нефункціональні вимоги
4. Доповнення

Функціональні вимоги пов’язані з функціональним аспектом програмного забезпечення. Вони визначають функції та функції в системі програмного забезпечення та з неї.

Нефункціональні вимоги – це неявні або очікувані характеристики програмного забезпечення, про які користувачі роблять припущення. До нефункціональних вимог належать: безпека, вартість, конфігурація, зберігання, продуктивність, гнучкість, сумісність, доступність тощо.

Додаток Б. Приклад змісту документу на етапі аналізу

Побудова архітектури системи здійснюється шляхом визначення цілей системи, її вхідних і вихідних даних, декомпозиції системи на підсистеми, компоненти або модулі та розроблення її загальної структури. Проектування архітектури системи може проводитися різними методами (стандартизованим, об’єктно-орієнтованим, компонентним і ін.), кожний з яких пропонує свій шлях побудови архітектури, а саме, визначення концептуальної, об’єктної й інших моделей за допомогою відповідних конструктивних елементів (блок-схем, графів, структурних діаграм тощо).

Ian Sommerville пропонує розділяти архітектуру ПЗ на два рівня абстракцій [116]:

1. Архітектура, що стосується певної програми, яку потрібно розкласти на компоненти.

2. Архітектура складних корпоративних систем, що містять інші системи, програми та програмні компоненти.

До складу архітектурного проекту ПЗ входять:

- опис елементів, з яких складається дана система;
- схеми взаємодій між цими елементами;
- документація зразків (patterns), на основі яких здійснюється їх компоновка;
- список і зміст обмежень (вимог), характерних для цих зразків.

Ілюстративними засобами вираження характеристик ПЗ, в архітектурному проекті використовуються різні нотації:

- блок-схеми (схеми алгоритмів);
- діаграми (UML, DFD);
- макети.

Приклад структури документу на етапі аналізу наведено у таблиці Б.1.

Таблиця Б.1 Структура документа на етапі аналізу

Резюме (до 250 слів)
Зміст
Статус документу (дати, автори, компанії тощо)
Зміни, які внесені після останньої версії програми

1. Вступ
   1.1. Ціль (мета)
   1.2. Контекст
   1.3. Терміни, скорочення, абревіатури
   1.4. Посилання
   1.5. Короткий огляд програми
2. Загальний опис
   2.1. Спільне з минулими і майбутніми проектами
   2.2. Спільне з існуючими проектами
   2.3. Функції та цілі
   2.4. Параметри середовища
   2.5. Загальні обмеження
   2.6. Відношення до зовнішніх програм
   2.7. Опис моделі
3. Вимоги (модель системи – функціональна декомпозиція)
   3.1. Функціональні вимоги
   3.2. Вимоги відношень з зовнішніми програмами
   3.3. Вимоги продуктивності
   3.4. Вимоги щодо ресурсів
   3.5. Вимоги тестування
   3.6. Вимоги перевірки
   3.7. Вимоги до документації проекту
   3.8. Вимоги до безпечності, переносимості
   3.9. Вимоги якості, надійності, підтримки, захисту
4. Доповнення
   4.1. Первинні ефекти аналізу
   4.2. Покращуються: документ з вимогами
   4.3. Словник даних

Створюються: документи, які містять опис створеної моделі: UML діаграми, звіт, який містить визначення методів, класів, ознак, алгоритмів, відносин тощо; графік етапу проектування.

Додаток В. Приклад змісту документу експлуатації

Документація експлуатації формується під час етапу аналізу і проектування ПЗ, а також є документованим результатом розробленого програмного продукту.

Згідно ДСТУ 2.610-2006 **експлуатаційний документ** – конструкторський документ, який окремо або в сукупності з іншими документами визначає правила експлуатації (наприклад, посібник з експлуатації) ПЗ та/або відображає відомості, що засвідчують гарантовані виробником значення основних параметрів та характеристик (властивостей) ПЗ, гарантії та відомості щодо нього експлуатації протягом встановленого терміну служби (наприклад, формуляр) [11].

Документація експлуатації формується за такими стандартами:

- ЕСКД, Єдина система конструкторської документації;
- ЕСПД, Єдина система програмної документації;
- КСАС, Комплекс стандартів на автоматизовані системи.

Для побудови структури документу експлуатації за основу взято міждержавні стандарти ДСТУ 2.610-2006 («Правила виконання експлуатаційних документів») [11], ДСТУ 3973-2000 («Система розроблення та поставлення продукції на виробництво. Правила виконання науково-дослідних робіт. Загальні положення») [18], внаслідок чого виділено наступні частини (рис.В.1):

- опис і робота;
- використання за призначенням;
- технічне обслуговування (супровід, підтримка).

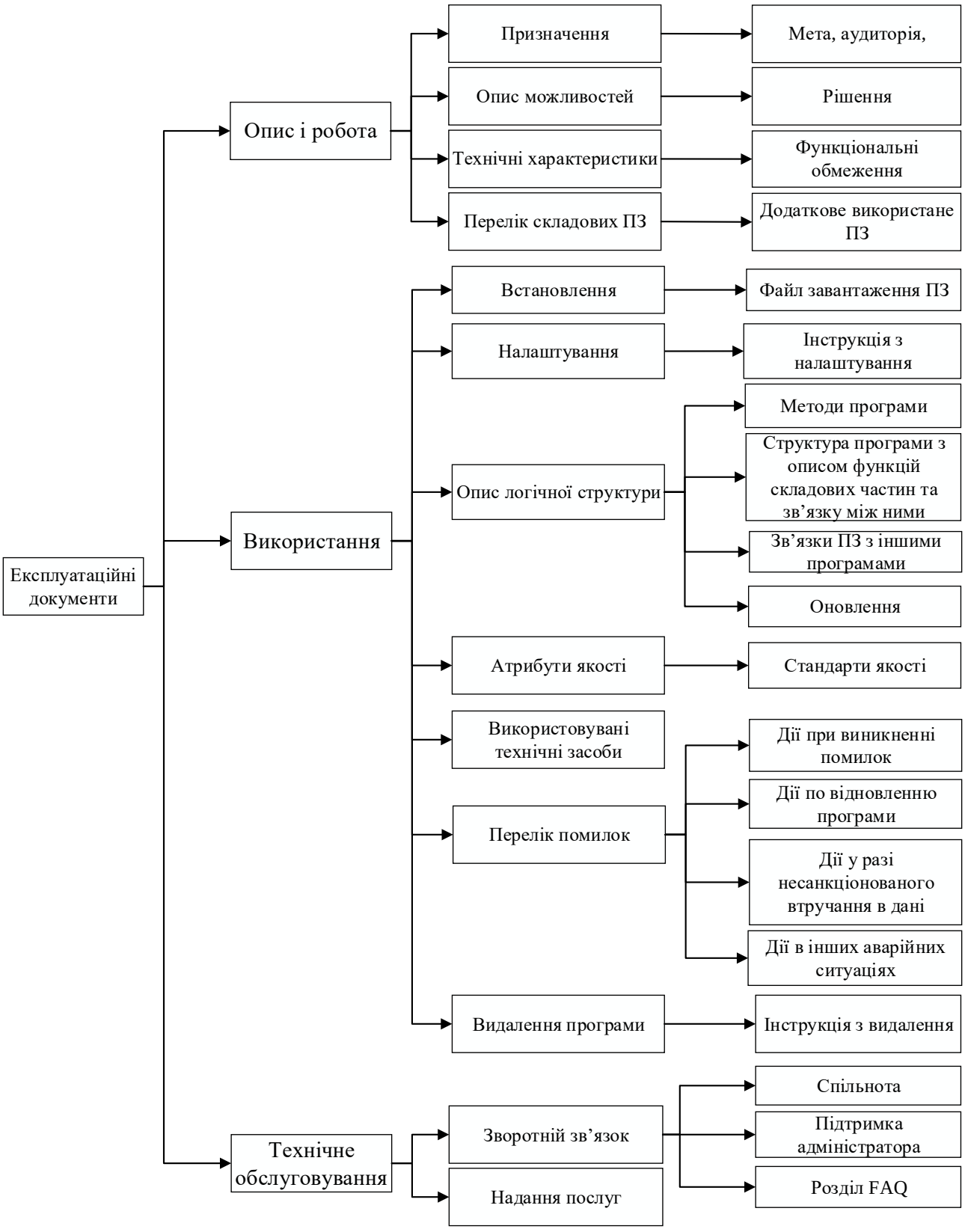


Рис.В.1. Структура документу експлуатації ПЗ

Розділ «Опис і робота» містить такі складові:

- призначення програми, область застосування, параметри, класи користувачів;
- опис можливостей – рішення, класи задач, яке надає ПЗ;
- технічні характеристики – містить дані, основні параметри, властивості системи, необхідні для встановлення та використання програми;
- складові програмного продукту – перелік додаткових програм за допомогою яких функціонує розроблене ПЗ.

Розділ «Використання за призначенням» поділяється на такі частини:

- порядок встановлення (файл завантаження) програми;
- налаштування ПЗ, а також інших допоміжних програм, які потрібно застосувати для коректної роботи ПЗ;
- опис логічної структури ПЗ;
- алгоритм програми;
- Використовувані методи;
- структура програми з описом всіх виконуваних функцій вищого та нижчого рівнів, завдань, комплексів задач, процедур, версій системи (найменування; умови для виконання операцій; підготовчі дії; основні дії в необхідній послідовності; заключні дії; ресурси, що витрачаються на операцію);
- зв’язки ПЗ з іншими програмами;
- атрибути якості (можна застосувати міжнародний стандарт ISO- 25010);
- інші застосовані технічні засоби для виконання певних дій в системі;
- перелік можливих несправностей/помилок в процесі використання:
- дії при відмовах роботи ПЗ або при виникненні помилок (клас, назва, опис помилки та дії користувача);
- дії по відновленню програм;

- дії у випадках виявлення несанкціонованого втручання в дані;
- дії в інших аварійних ситуаціях;
- порядок видалення програми.

Розділ «Технічне обслуговування ПЗ»:

- зворотній зв’язок з відділом технічної підтримки (наприклад, при виникненні помилок, які не описані в розділі «Аварійні ситуації»):
- спільнота для обговорення користувачів та зацікавлених осіб ПЗ;
- підтримка адміністратора;
- розділ FAQ.
- надання послуг.

Проведення деталізації основних розділів на підрозділи необхідно для чіткого визначення наповнення документу експлуатації як ІП інформацією. Кожен розділ відображає дані у єдиному вигляді із упорядкованими блоками інформації та формою подання.

Додаток Г. Схема обчислення рейтингу допису розміщеному у ВС

Рис.Г.1 Схема обчислення рейтингу допису розміщеному у ВС

Додаток Д. Шаблон завантаження даних з віртуальної спільноти CodeProject

```
code_project_forum_path =
Template('https://www.codeproject.com/script/Content/Tag.aspx?tags=${page}')
def find_articles_code_project(content, articles_storage: list, already_watched:
set):
    """Scrap articles from content into an article_list"""
    # блок 1
    bs = BeautifulSoup(content.text, 'html.parser')
    articles = bs.find('div', class_='article-list').find('div',
class_='content-list').findAll('div', class_='content-list-item')

    # блок 2
    for article in articles:
        # check if article is fresh enough
        date = article.find('div', class_ = 'entry').text.strip()
        try:
            if int(date.split(' ')[2]) < year:
                continue
        except:
            pass

        # блок 3

        # article url
        article_url = 'https://www.codeproject.com' + article.find('div',
class_='title').find('a')['href']
        if article_url in already_watched:
            continue

        # article parser
        bs_article = BeautifulSoup(requests.get(article_url).text,
'html.parser')
        article_element = bs_article.find('div', class_='article')

        # блок 4
        art_info = {
            'Title': article_element.find('h1').text.strip(),
            'ArticleUrl': article_url,
            'Lang': 'en',
            'Date': date
        }

        description = article_element.find('div', id="ctl00_DescriptionSpot")
        art_info['Description'] = description.text if description else np.nan

        # блок 5
        # author and number of views from article url
        author_div_a = bs_article.find('span', id="ctl00_Authors").find('a')
        art_info.update({
            'Author': author_div_a.text.strip(),
            'AuthorUrl': author_div_a['href'],
        })

        #rating start
        rating_element = article_element.find('div',
id='ctl00_RateArticle_VoteCountNoHist')
        art_info['Rating'] = rating_element.text.strip('/')[0] if rating_element
else np.nan
```

```
        art_info['RatingVotes'] = rating_element.text.strip(' ')[1].strip('
').split(' ')[0] if rating_element else np.nan

        # блок 6
        # Tabs with comments
        tabs = bs_article.find('div', class_='tabs')
        tabs = tabs.findAll('div') if tabs else []
        try:
            art info['RevisionsCount'] = tabs[3].find('a').text.split('
')[1].strip('()')
        except:
            art_info['RevisionsCount'] = np.nan

        try:
            art_info['CommentsCount'] = tabs[4].find('a').text.split('
')[1].strip('()')
        except:
            art_info['CommentsCount'] = np.nan

        try:
            art_info['PostingDate'] = ' '.join(tabs[5].find('div').text.split('
')[1:])
        except:
            art_info['PostingDate'] = np.nan

        # Tags
        tags = bs_article.find('span', id="ctl00_TagList_VisibleTags")
        tags = tags.findAll('div', class_='t') if tags else []

        if len(tags) > 1:
            art_info['Tags'] = [tag.find('a').text.lower() for tag in tags[1:]]
        else:
            art_info['Tags'] = []

        # Stats
        stats = bs_article.find('div', class_='stats')
        stats = stats.findAll('div') if stats else []
        for stat in stats:
            if stat.text.split(' ')[-1] == 'views':
                art_info['PageViews'] = stat.text.split(' ')[0]
            if stat.text.split(' ')[-1] == 'bookmarked':
                art_info['StarsCount'] = stat.text.split(' ')[0]
            if stat.text.split(' ')[-1] == 'downloads':
                art_info['DownloadedCount'] = stat.text.split(' ')[0]

        # topic
        topic = bs_article.find('div', class_="container-breadcrumb")
        art_info['Topic'] = topic.findAll('div')[-1].text.strip(' ') if topic
else np.nan

        articles_storage.append(art_info)
        already_watched.add(article_url)
```

Додаток Е. Лематизація текстів дописів

```
def construct_en_standardize_func(lemmatize: bool = True, clean_stopwords: bool =
True):
    """Returns the standardizing function for English text"""

    if lemmatize:
        lemmatizer = WordNetLemmatizer()
        process = lambda s: lemmatizer.lemmatize(s)
    else:
        process = lambda s: s

    stop_words = stopwords.words('english') if clean_stopwords else []
    stop_words.append(" ")

    punct_pattern = re.compile(f"[{punctuation}]")

    def standardize(text: str):
        text = text.lower()
        text = text.replace('\\n', ' ')
        text = re.sub(punct_pattern, ' ', text)
        text = " ".join([process(word) for word in text.split(' ') if word not in
                         stop_words])
        return text

    return standardize

def process_lemma_xml(xml_text: str, stopwords: list):
    lemma_tokens = []
    en_lemma = WordNetLemmatizer()

    for sentence in BeautifulSoup(xml_text, 'xml').findAll('sentence'):
        for tok_reading in sentence.findAll('tokenReading'):
            token = tok_reading.find('token')
            if token['tags'] == 'unclass':
                lemma_tokens.append(en_lemma.lemmatize(token['value']))
            elif token['tags'] == 'punct':
                continue
            elif token['tags'] == 'unknown' and token['lemma'] not in stopwords:
                lemma_tokens.append(token['value'])
            elif token['lemma'] not in stopwords:
                lemma_tokens.append(token['lemma'])

    return " ".join(lemma_tokens)

def standardize_ua_text(text, stopwords: list):
    # text = text.lower()

    with open('temp_input.txt', 'x', encoding='utf-8') as temp_input_file:
```

```
            temp_input_file.write(text)
        lemma_process_output = subprocess.run(
            ["python",
            "D:/projects/lpnu/src/main/python/tag_text.py",
            "temp_input.txt"], capture_output=True
        )
        os.remove('temp_input.txt')

        lemma_okens_xml, _ = utf_8.decode(lemma_process_output.stdout[9:])
        std_text = process_lemma_xml(lemma_okens_xml, stopwords)

        return std_text

    def construct_ua_standardize_func(lemmatize: bool = True, clean_stopwords: bool =
    True):
        if clean_stopwords:
            stopwords = ua_stopwords
        else:
            stopwords = []

        stopwords.append(" ")

        def standardize(text: str):
            text = standardize_ua_text(text, stopwords)
            return text

        return standardize

        #  Вибір лематизації текстів в залежності від мови
    def standardize_text(df: pd.DataFrame, lemmatize: bool = True, clean_stopwords:
    bool = True):
        en_standardizer = construct_en_standardize_func(lemmatize=lemmatize,
    clean_stopwords=clean_stopwords)
        ua_standardizer = construct_ua_standardize_func(lemmatize=lemmatize,
    clean_stopwords=clean_stopwords)

        if 'Lang' in df.columns:
            ua_text_mask = df['Lang'] == 'ua'
        else:
            ua_text_mask = np.full(df.shape[0], False)

        cleared_text = np.empty(df.shape[:1], dtype='object')

        cleared_text[~ua_text_mask] = df[~ua_text_mask]['Content'].map(en_standardizer)
        cleared_text[ua_text_mask] = df[ua_text_mask]['Content'].map(ua_standardizer)

        return cleared_text
```

Додаток Є. Визначення релевантності тем віртуальних спільнот

```
def process_query(query: str, extended: bool, tfidf_themes=tfidf_themes,
additional_themes=additional_themes):
    query = standardize_text(query, [])
    tfidf_vector = tfidf_themes.transform([query]).A[0]

    scores = {k: 0.0 for k in tfidf_themes.vocabulary_.keys()}

    for theme, index in tfidf_themes.vocabulary_.items():
        scores[theme] += tfidf_vector[index]
        if extended and theme in additional_themes.keys():
            for extra_theme, extra_score in additional_themes[theme].items():
                scores[extra_theme] += tfidf_vector[index] * extra_score


    scores = pd.Series(scores).sort_values(ascending=False)
    return scores if scores.max() == 0 else scores / scores.max()
```

Додаток Ж. Словник термінів

Таблиця Ж.1. – Список англомовних термінів які вживають українською мовою.

Україномовне застосування	**Термін англійською мовою**	**Тлумачення**
Адмін	Admin	Адміністратор
Айді	Identificator	Ідентифікатор
Айпі	Internet Protocol Address	IP-адреса, унікальна адреса комп'ютера в мережі
Айті	IT, Information Technologies	Інформаційні технології
Апгрейд	Upgrade	Оновлення, модернізація апаратного забезпечення
Апдейт	Update	Модернізація програмного забезпечення
Апрув	Approve	Підтвердження, згода, схвалення чого-небудь
Аттачнути	Attach	Прикріплювати
Баг	Bug	Помилка
Багрепорт	Bug report	Повідомлення або звіт про помилку
Бан	Ban	Обмеження чи позбавлення будь-яких прав користувача
Бенефіт	Benefit	Винагорода
Браузер	Browser	Програма для перегляду веб-сторінки
Бекап	Backup	Резервне копіювання
Бекенд	Back-end	Розробка «внутрішньої частини» сайтів, додатків тощо
Валідний	Valid	Коректний, дійсний
Вебінар	Web based seminar	Онлайн семінар або презентація, яка проводиться в режимі реального часу через Інтернет
Віндоус, Вінда	Windows	Операційна система Windows
Воркшоп	Workshop	Навчальний захід для команди з певній галуз
Геймдев	Game development	Розробка, створення ігор
Геймер	Gamer	Гравець
Дебажити	debug / debugging	Пошук та виправлення помилок в коді програми
Девайс	Device	Технічний пристрій
Девелопер	Developer	Розробник
Девелопмент	Development	Розробка чогось (програм, ігор, додатків тощо)

Дедлайн	Deadline	Крайній термін виконання завдання чи роботи
Демка	Demo	Демонстраційна версія чогось
Деплой	Deploy	Розгортання, перенесення ПЗ
Джуніор	Junior Developer	Розробник-початківець у певній галузі технологій
Домен	Domain	Унікальна адреса сайту в Інтернеті
Законектити, підконектитись	Connect	Приєднатися до чого-небудь
Інсталювати	Install	Встановлювати
Кастомер	Customer	Клієнт, замовник, покупець
Кейс	Case	реальна ситуація, випадок.
Кодер	Coder	Програміст
Комітнути, закомітити, зробити коміт	Commit	Зафіксувати зміни коду в сховищі, репозиторії
Копіпаст	Copy / paste	Скопіювати та надалі вставити
Кряк	Crack	Програма для злому іншого ПЗ
Лайфхак	Lifehack	Корисна порада
Левел	Level	Рівень
Лінк	Link	Посилання на веб-ресурс
Локейшн	Location	Місце розташування
Лептоп	Laptop	ноутбук
Мануал	Manual	Інструкція використання
Ментор	Mentor	Наставник
Мідл	Middle Developer	Середній рівень знань розробника
Мітап	Meetup	Збори, зустріч
Овертайм	Overtime	Позаробочі години
Оупенспейс	Open space	Офіс відкритого типу
Рандом, рандомний	Random	Довільний, випадковий
Рісерч, ресерч	Research	Дослідження якоїсь теми
Саппорт	Support	Служба підтримки
Сейв	Save	Зберегти зміни
Скіл(и)	Skill	Навички, вміння
Скрін, скріншот	Screenshot	Знімок екрана
Скрипт	Script	Програмний файл сценарію, що автоматизує певну задачу
Софт	Software	Програма, програмне забезпечення
Утиліта	Utility	Допоміжне програмне забезпечення
Факап	Fuck up	Провал, невдача
Фіксити	Fix	Виправляти помилки

Фреймворк	Framework	Програмний продукт / платформа / система, що полегшує створення та підтримку проектів
Фріланс	Freelance	Робота з дому
Чекнути	Check	Щось перевірити.
Юзати	Use	Використовувати
Юзер	User	Користувач

Таблиця Ж.2. – Список «сленгових» термінів ІТ-галузі.

Україномовне застосування	**Тлумачення**
Гіг	Гігабайт
Глюк	Незрозумілий, неочікуваний збій
Гавнокод	Поганий, нечитабельний код
Гуглити	Здійснювати пошук через систему Google.
Айтішник	Фахівець з інформаційних технологій
Апрувнути	Підтверджувати щось
Варєзник	Веб-ресурс, що поширює варез (нелегальне програмне забезпечення)
Галера	Організація, компанія, де працюють люди сфери іт
Веслярі	Працівники ІТ-компаній
Залізо	Апаратне забезпечення, комплектуючі комп'ютерів
залити	Завантажити, виставити на загал щось
Зафакапити, нафакапити	Не впоратися з чимось
Інет	Інтернет
Клава	клавіатура
Кодити	Програмувати
Компілятор	Програма, що здійснює компіляцію
Конфа	Конференція
Лагати, лаги	Затримка при виконанні програми
Материнка	Материнська плата
Моніторити	Спостерігати за чим-небудь
Смітник	Корзина
Ноут	Ноутбук
Оперативка	Оперативна пам'ять
Піратка	Неліцензійне ПЗ або ОС
Плюшки	Бонуси
Прога	Програма
Сервак	Сервер
Сисадмін	Системний адміністратор
Системник	Системний блок
Чайник	Недосвідчений користувач, новачок
Шарпи	мова програмування C #

Таблиця Ж.3. – Відповідність ключових фраз до терміну, що побудована за допомогою регулярних виразів.

Термін	Ключова фраза	Міра відповідності фрази до терміну
Operating system Microsoft Windows 10	OS | Operating system | Операційна система | ОС Microsoft | MS | (M)??крософт Windows | Віндо?с в | верс?? | v | version 10	1
	OS | Operating system | Операційна система | ОС Microsoft | MS | (M)??крософт Windows | Віндо?с 10	1
	Microsoft | MS | (M)??крософт Windows | Віндо?с v | version | в | верс?? 10	0.85
	Microsoft | MS | (M)??крософт Windows | Віндо?с 10	0.85
	Microsoft | MS| (M)??крософт Win v | version | в | верс?? 10	0.7
	Microsoft | MS | (M)??крософт Win 10	0.7
	Windows | Віндо?с v | version | в | верс?? 10	0.6
	Windows | Віндо?с 10	0.6
	Win v | version | в | верс?? 10	0.5
	Win 10	0.45
	Windows | Віндо?с	0.35
	Win	0.3

Додаток 3. Акти про впровадження результатів дисертаційних досліджень

ЗАТВЕРДЖУЮ
Проректор з наукової роботи
НУ «Львівська політехніка»
_____ д.т.н., професор Демидов І.В.
"21" квітня 2023 р.

Акт

про використання результатів дисертаційної роботи

Синько Анни Іванівни

«Методи і засоби формування суспільно орієнтованої документації програмного забезпечення за допомогою віртуальних спільнот»

у навчальному процесі кафедри «Інформаційних систем та мереж»

Даний акт складений комісією у складі:

д.т.н., проф. Литвин В.В. – завідувач кафедри інформаційних систем та мереж;

к.т.н., доц. Андруник В.А. – доцент кафедри інформаційних систем та мереж;

к.т.н., доц. Ришковець Ю.В. – доцент кафедри інформаційних систем та мереж, лектор дисципліни «Алгоритмізація і програмування», про те що в навчальному процесі кафедри інформаційних систем та мереж використано результати дисертаційної роботи «Методи і засоби формування суспільно орієнтованої документації програмного забезпечення за допомогою віртуальних спільнот» для студентів спеціальності 124 «Системний аналіз».

Методи: відбору адекватних джерел інформації, визначення авторитетних дописів, виявлення та усунення небажаного інформаційного наповнення використовуються з метою формування документації програмного забезпечення впроваджено для виконання лабораторних робіт дисципліни «Алгоритмізація і програмування». Метод структурування інформаційного наповнення застосовано для розробки навчально-методичних матеріалів дисципліни «Програмні засоби бізнес-аналізу», що містить необхідні теоретичні та практичні відомості для засвоєння предмету. З результатів дослідження розроблені навчально-методичні матеріали з дисципліни «Програмні засоби бізнес-аналізу» отримали сертифікацію у Віртуальному навчальному середовищі Національного університету «Львівська політехніка».

Члени комісії: _____ Литвин В.В.

_____ Андруник В.А.

_____ Ришковець Ю.В.

ЗАТВЕРДЖУЮ

Директор ТОВ «Інтелектуальні Вендінгові Системи»

_____ Тарас КОЗАК

"5" квітня 2023 р.

АКТ

про дослідне використання результатів
дисертаційного дослідження аспіранта кафедри соціальних комунікацій та інформаційної діяльності
Національного університету «Львівська політехніка»
Синько Анни Іванівни

Даний акт свідчить про те, що методи (визначення адекватності теми віртуальної спільноти як джерела цінного інформаційного наповнення; обчислення рейтингу допису на основі реакцій користувачів та аналізу тональності тексту коментарів до нього; аналізу автора допису) та словники термінів розроблені під час виконання дисертаційної роботи Синько Анни Іванівни на тему «Методи і засоби формування суспільно орієнтованої документації програмного забезпечення за допомогою віртуальних спільнот» пройшло дослідне випробовування на ТОВ «Інтелектуальні Вендингові Системи» для збору та аналізу даних з віртуальних спільнот щодо впровадженого програмного забезпечення платіжних терміналів.

Впроваджені методи дозволяють визначити джерела – віртуальні спільноти та теми, що містять інформацію про конкретне програмне забезпечення; отримати відгуки, що є оцінені іншими користувачами, споживачами у вигляді реакцій (оцінювання, взаємодії та аналіз тональності тексту коментарів), які є наявні у тій чи іншій спільноті; відібрати дані за становлений проміжок часу; провести аналіз автору, що розмістив допис для подальшого формування його рейтингу. Обчислення адекватності теми та аналіз тональності тексту реалізовано за допомогою словників термінів.

Даний підхід дозволяє швидко отримати інформацію про програмне забезпечення від споживачів з будь-якої віртуальної спільноти та оцінити розміщені дані за допомогою наявних реакцій спільноти або на основі рейтингу автору допису.

Даний акт не є підставою для взаємних фінансових розрахунків.