DOI 10.31891/2307-5732-2023-319-1-81-83
УДК 004.89

ДАНИЛИК ВІТАЛІЙ
Національний університет "Львівська Політехніка"
https://orcid.org/0000-0001-5928-7235
e-mail: vitalii.danylyk.msaad.2021@lpnu.ua
ЛИТВИН ВАСИЛЬ
Національний університет "Львівська Політехніка"
https://orcid.org/0000-0002-9676-0180
e-mail: vasyl.v.lytvyn@lpnu.ua
МУШАСТА СОЛОМІЯ
Національний університет "Львівська Політехніка"
https://orcid.org/0000-0003-4932-4113
e-mail: solomiyanytrebych@gmail.com

ІНФОРМАЦІЙНА СИСТЕМА ІДЕНТИФІКАЦІЇ ТЕРМІНІВ ТА АБРЕВІАТУР У ТЕКСТОВИХ ДОКУМЕНТАХ

У роботі описано процес побудови та функціонування системи ідентифікації термінів та абревіатур у текстових документах. Така проблема ідентифікації часто виникає у військовій сфері. Окремий термін чи абревіатура може мати кілька пояснень, що враховано у процесі роботи системи. Для пошуку термінів у різних відмінках використано методи опрацювання природномовних текстів. Описано вимоги до системи та обґрунтовано вибір програмних засобів. Наведено приклади функціонування системи.

Ключові слова: інформаційна система, опрацювання природної мови, термін, абревіатура, текстовий документ.

DANYLYK VITALII, LYTVYN VASYL, MUSHASTA SOLOMIYA
Lviv Polytechnic National University

INFORMATION SYSTEM OF IDENTIFICATION OF TERMS AND ABBREVIATIONS IN TEXT DOCUMENTS

The paper examines the process of building and functioning of the system for identifying terms and abbreviations in text documents. The task of developing such a system is urgent, since such an identification problem often arises in the military sphere. During the implementation of the system, it was taken into account that a single term or abbreviation may have several explanations in different regulatory documents. All available explanations are added to the term or abbreviation, which is taken into account during the operation of the system. A feature of the system is the use of natural language processing methods, since terms can be found in different cases. To implement the system, ready-made Python packages were used to cover similar tasks: Tkinter, PyMuPDF Examples of the system's functioning are given. The developed system is used in practice.

In the process of completing the work, the research of problems and the search for solutions for the tasks is carried out, an information system is developed for the processing of documents with the aim of integrating definitions of potentially unknown terms and abbreviations into them, in order to enable officers to use any literature without problems, because all terms and abbreviations will be known. To generalize the documentation, all the necessary requirements for the system are defined, and in order to correctly create the architecture and allocate the functional tasks of the system under development, a system analysis is performed and a conceptual model is built. Using all the specified information, all the necessary diagrams are built using the UML notation. Diagrams depict the relationships between objects and the overall architecture of the system. The architecture of the system is built in such a way that the component systems and the system as a whole can be easily expanded. At the end of the development, testing and implementation of the project is carried out. The process of operation of the components of the system on the part of the end user and the process of deployment by the end user of the information system are described. The object of the study is the presence of slowing factors in the process of command and control carried out by commanders of tactical units, which can slow down decision-making and also affect their correctness. The subject of the study is to solve the problems of the appearance of slowing factors in the process of command and control carried out by commanders of tactical units, by means of work with military data.

Keywords – information system, natural language processing, term, abbreviation, text document.

Постановка проблеми та її рішення

У багатьох документах (D), особливо у військовій сфері, використовуються незнайомі терміни (T) та абревіатури (A). Пояснення цих термінів та абревіатур можна знайти у нормативних документах (ND) [1]. Інколи в різних ND можуть бути різні пояснення. Очевидно, що пам'ятати всі терміни та абревіатури користувачам документів D неможливо. Потрібна автоматизована система, яка б надавала пояснення термінів та абревіатур. Наприклад під час наїзду мишкою на термін чи абревіатуру висвітлюється його пояснення з посиланням на відповідні джерела інформації ND.

Терміни та абревіатури разом формують певний словник $S=T \cup A$. Документи D, що опрацьовуються, як правило знаходяться в pdf форматі.

Після проведення бізнес-аналізу із користувачами такої системи, нами визначено такі вимоги до системи інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур:

- Програма має бути розроблена з використанням Python + Tkinter + PyMuPDF.
- Програмне забезпечення повинне забезпечувати функціонування в режимі 24 години на добу, 7 діб на тиждень, 365 днів на рік (за умови безвідмовного функціонування апаратного забезпечення).
- Програмне забезпечення не має вимагати перерв на регламентне обслуговування та/або резервне копіювання інформаційного змісту.

Система повинна дозволяти користувачам виконувати наступні операції:

- Вибір словника та інформаційного pdf файлу – користувач має мати можливість вибирати як і інформаційний файл формату pdf, так і словник перед інтеграцією.
- CSV словник – користувач має мати можливість використовувати CSV як базу даних всіх термінів, абревіатур та їх визначень.
- Нотатки як визначення – інтегрування в PDF нотаток з визначеннями термінів та абревіатур, які з'являються при наведенні в програмах-переглядачах PDF.
- Формування файлу вхідного розширення – збереження результату як інформаційний файл з тим самим розширенням, що і вхідний інформаційний файл.
- Зберігання останнього використаного словника – при наступному вході в програму останній використаний словник буде вибраний автоматично.
- програма повинна встановлюватися на персональні комп'ютери.

Програма для інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур передбачає роботу одного користувача. Вона використовує файли інших програм. Зокрема PDF для інформаційних файлів в які будуть додаватись невідомі терміни та абревіатури зі словника, а словник в свою чергу має розширення CSV. Для використання словника такого типу потрібно його утворити за допомогою інших програм [2].

Для відображення роботи системи використано діаграму станів (Рис. 1). Дана діаграма відображає всі стани та переходи в яких може перебувати система.

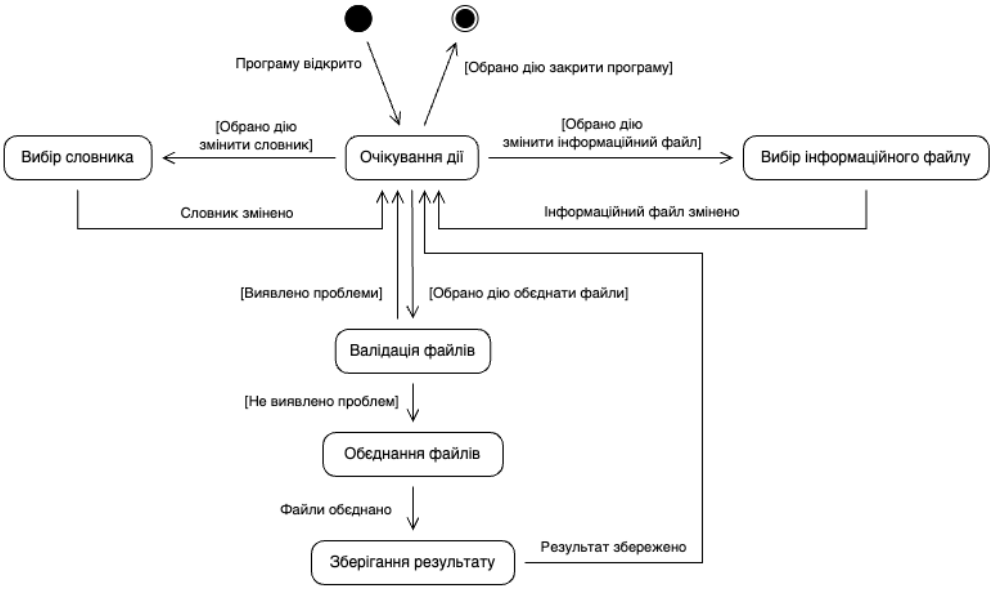

Рис. 1. Діаграма станів інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур

Система містить два зовнішні об'єкти: користувач та програма. Діаграма послідовності (рис. 2) відображає взаємодії цих об'єктів. На діаграмі не показано інші програми які утворюють інформаційний файл чи формують словник так як для роботи з програмою для інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур потрібно мати вже готові файли для об'єднання. Вони використовуються як вхідні дані.

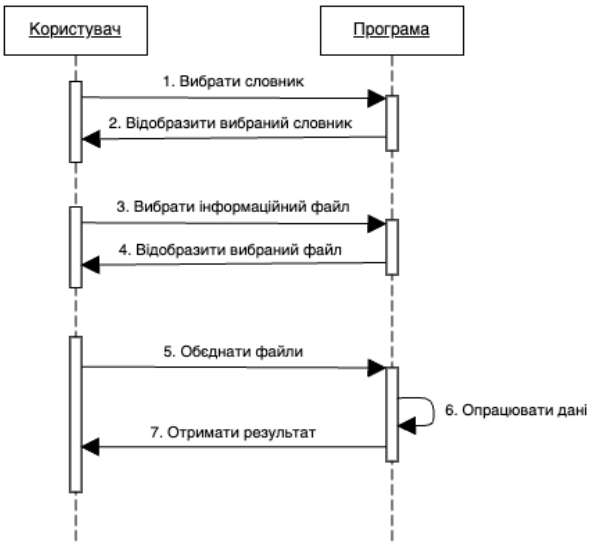

Рис. 2. Діаграма послідовності інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур

Моделювання об'єктів предметної області

Система інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абривіатур передбачає роботу з інформаційними файлами та словниками. Ці файли можуть мати різні розширення, відповідно для кожного з розширень буде своя реалізація. Для вирішення цього основного завдання на базі інтерфейсів була розроблена відповідна діаграма класів (рис. 3) [3]. Можна дописувати різну реалізацію інтерфейсів IResource та Idictionary, що дає можливість в майбутньому розширювати програму для багатьох інших типів файлів [4].

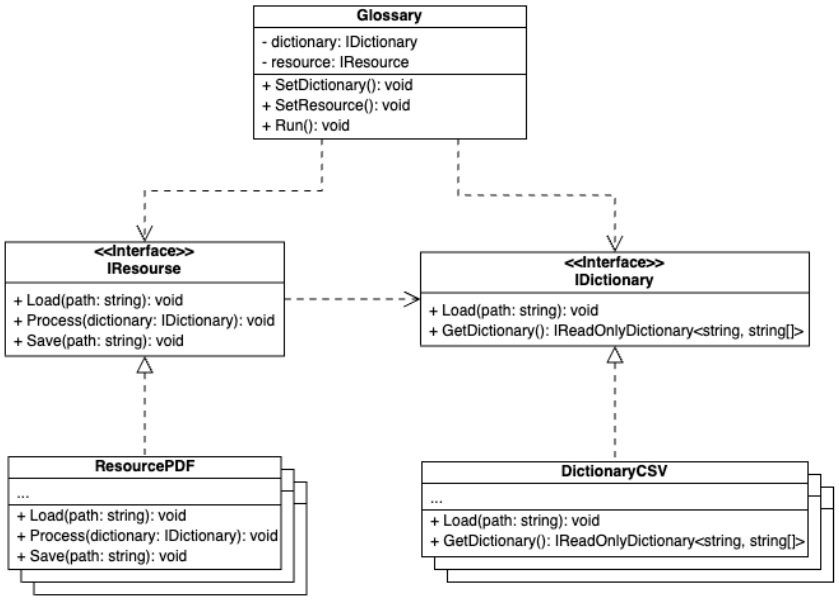

Рис. 3. Діаграма класів інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур

Вибір та обґрунтування методів розв'язання задачі

Особливістю системи є опрацювання природної мови (NLP), оскільки терміни можуть зустрічатися у різних відмінках тощо. NLP поєднує комп’ютерну лінгвістику – моделювання людської мови на основі правил – зі статистичними моделями, моделями машинного та глибокого навчання [5]. Разом ці технології дозволяють комп’ютерам обробляти людську мову у формі тексту або голосових даних і «розуміти» її. Обробка природної мови включає два підходи: статистичний та методи нейронних мереж [6]. Багато досліджень обробки природної мови значною мірою покладаються на машинне навчання [7]. Обробка природної мови є необхідним функціоналом для інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур. У системі розроблено методи обробки певних слів та словосполучень для їх пошуку, а не методи для усвідомлення сенсу речень чи подібних завдань з якими безперечно краще справиться нейронна мережа. Завдання що необхідно виконати відносяться до морфологічного аналізу. Морфологічний аналіз включає лематизацію, морфологічну сегментацію, позначення частини мови та стемінг.

Лематизація – завдання видалення лише флективних закінчень і повернення базової словникової форми слова, яка також відома як лема. Це ще один прийом редукції слів до їх нормалізованої форми. Але в цьому випадку перетворення фактично використовує словник для відображення слів у їх фактичній формі.

Морфологічна сегментація – розділення слова на окремі морфеми та визначення класу морфем. Складність цього завдання багато в чому залежить від складності морфології (тобто структури слів) мови, що розглядається. Англійська мова має досить просту морфологію, особливо флексійну, і тому часто можна повністю ігнорувати це завдання і просто моделювати всі можливі форми слова (наприклад, «open, opens, opened, opening») як окремі слова. Однак у багатьох мовах такий підхід неможливий, оскільки кожна стаття словника містить тисячі можливих словоформ.

Позначення частини мови – визначення частини мови для кожного слова за поданим реченням. Багато слів, особливо загальновживані, можуть виконувати функції кількох частин мови. Наприклад, «book» може бути іменником («the book on the table») або дієсловом («to book a flight»); «set» може бути іменником, дієсловом або прикметником; і «out» може бути будь-якою з принаймні п'яти різних частин мови.

Стемінг – процес скорочення відмінюваних (або іноді похідних) слів до основної форми (наприклад, «close» буде коренем для «closed», «closing», «closed», «closer» тощо). Створення коренів дає такі ж результати, як і лематизація, але робить це на основі правил, а не словника.

Ці всі методи будуть грати ключову роль в алгоритмах опрацювання природної мови, яка буде міститись в інформаційних файлах pdf. Ці методи допоможуть зв'язати словник з текстом навіть тоді, якщо слова будуть стояти в різних відмінках чи словосполученнях.

Програма інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур передбачає опрацювання алгоритму формування структури даних словника яка з'єднує всі слова з усіма визначеннями (рис. 4) [8].

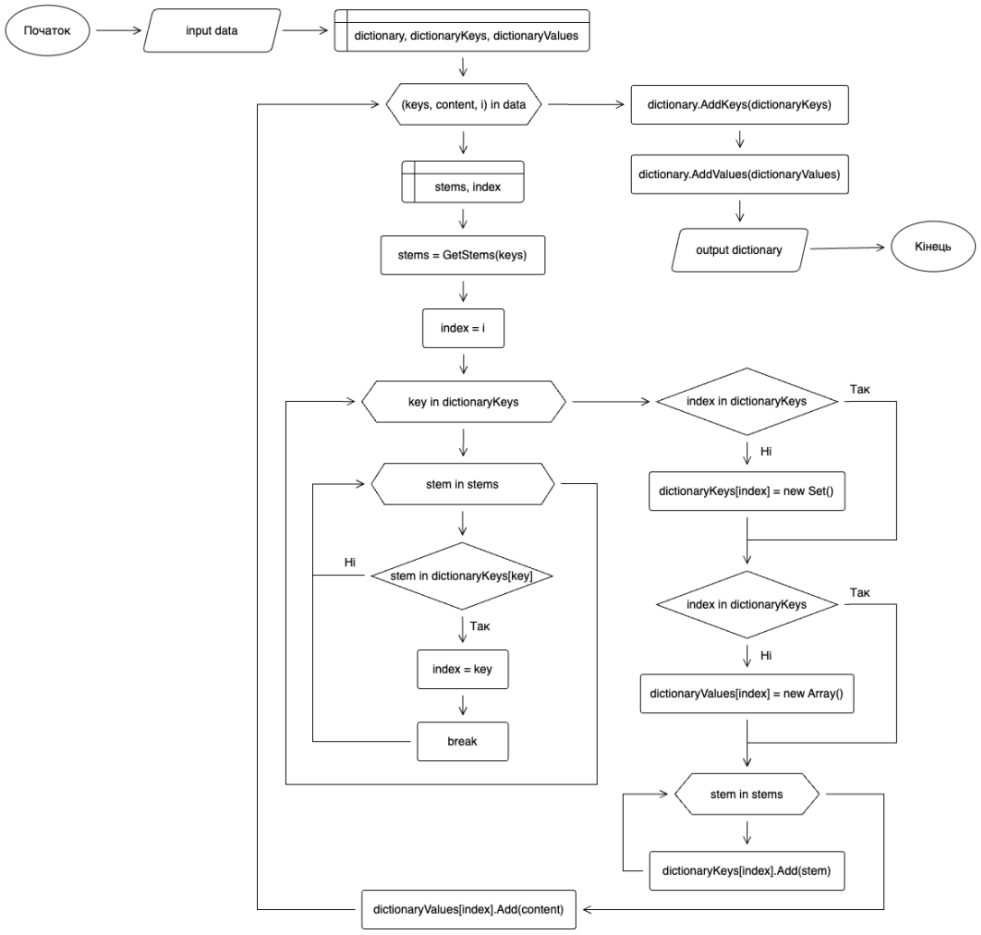

Рис. 4. Блок-схема алгоритму інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур

Алгоритм отримує на вхід дані словника в текстовій формі з метою перетворити ці дані на структурований об'єкт, який містить модель словника з яким буде працювати програма. Алгоритм формує дві структури даних на основі хеш-таблиці для ключів (слів словника) та визначень (контенту який пов'язаний з словом в словнику). У словнику може бути декілька визначень для декількох термінів. Даний алгоритм визначає групи та об'єднує сутності для укомплектування декількох визначень одного терміну чи словосполучення. У результаті ці об'єкти об'єднуються в єдину структуру даних що повертається.

Вибір і обґрунтування засобів розроблення проекту

Програма інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур передбачає роботу на персональному комп'ютері. Також для роботи даної програми потрібно взаємодіяти з форматами файлів PDF та CSV. Відповідно найкращим підходом буде використовувати готові пакети Python для покриття даних задач: Python + Tkinter + PyMuPDF. Python також має низку бібліотек, які дозволяють програмістам писати програми для аналізу даних і машинного навчання швидше і ефективніше, наприклад TensorFlow і Keras. Python має багато фреймворків графічного інтерфейсу, але Tkinter є єдиним фреймворком, вбудованим у стандартну бібліотеку Python. Tkinter має кілька сильних сторін. Він кросплатформний, тому той самий код працює в Windows, macOS і Linux. Візуальні елементи відображаються за допомогою власних елементів операційної системи, тому програми, створені за допомогою Tkinter, виглядають так, ніби належать платформі, на якій вони запускаються.

Цей фреймворк Python надає інтерфейс до набору інструментів Tk і працює як тонкий об’єктно-орієнтований рівень зверху Tk [9]. Набір інструментів Tk – це кросплатформна колекція «графічних елементів керування», або віджетів, для створення інтерфейсів програм. PyMuPDF — це прив’язка Python для MuPDF —засіб перегляду PDF, XPS і електронних книг, який підтримується та розробляється Artifex Software, Inc. MuPDF може отримати доступ до файлів у форматах PDF, XPS, OpenXPS, CBZ, EPUB і FB2 (електронні книги), він відомий своєю високою продуктивністю та високою якістю візуалізації. Для зручної розробки був використаний редактор коду Visual Studio Code де можна було підібрати різноманітні додаткові інструменти в вигляді розширень [10].

Розробка програми

Перед розробкою проекту було організовано робочий простір та визначено основний об'єктно орієнтований підхід який добре висвітлений в різноманітних програмах розроблених на Java [11]. Також було опрацьовано завдання клієнта та сервера [12]. Роль клієнта і сервера було об'єднано в єдину програму з графічним інтерфейсом та роботою з даними яка зазвичай опрацьовується сервером [13]. Для зручної роботи з програмою було використано елементи синхронних методів та функцій які використовуються при розробці програм що опрацьовують дані в реальному часі [14]. Було встановлено Python та імпортовано необхідні пакети такі як Tkinter, Fitz та інші. Також перед розробкою коду був розроблений прототип в інструменті для створення дизайну Figma (рис. 5). Таким чином було підібрано зручний і красивий графічний інтерфейс програми.

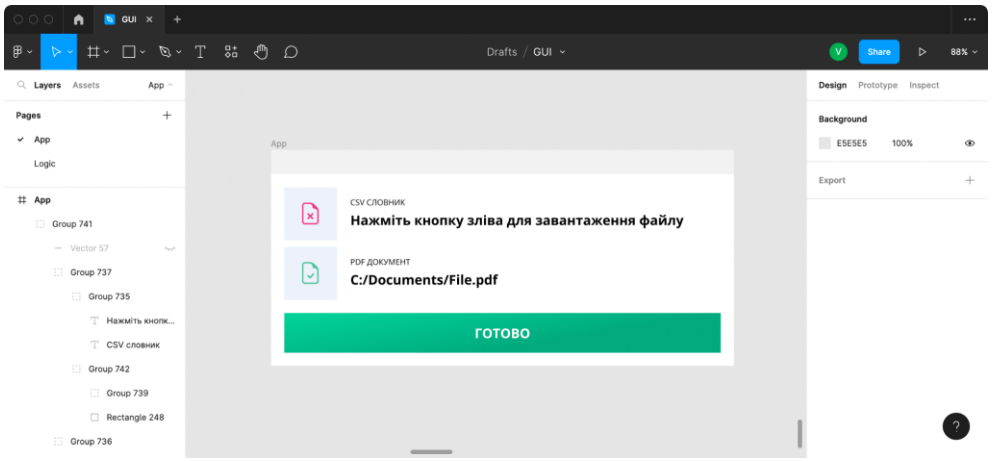

Рис. 5. Дизайн графічного інтерфейсу

Після створення дизайну було розроблено верстку графічного інтерфейсу використовуючи бібліотеку Tkinter. Таким чином стало можливим відкривати програму і користуватись нею, використовуючи графічний інтерфейс.

Програма інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур має одне вікно для взаємодії користувача з програмою (рис. 6) та передбачає формування словника в любій зручній програмі яка дозволяє експортувати дані з типом CSV (рис. 7). Даний файл, як і інформаційний файл з типом PDF, буде завантажуватись в програму для їх об'єднання.

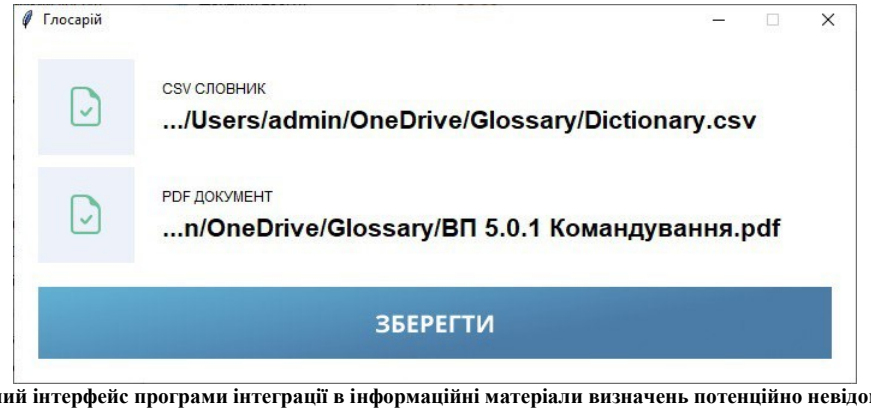

Рис. 6. Графічний інтерфейс програми інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур

A1 [CCIR]

	A	B	C	D	E	F	G
1	[CCIR]	Critical commander's information requirements					
2	[FRAGO]	Fragmentary order					
3	[IR]	Information requirements					
4	[CCIR]	Вимоги командира до критичної інформації					
5	[FRAGO]	Фрагментальний/коригуючий наказ					
6	[IR]	Інформаційні вимоги					
7							

Рис. 7. Словник термінів та абревіатур

У результаті в користувача на комп'ютері з'являється копія інформаційного файлу з додатковими даними, які відображаються в переглядачі як нотатки (рис. 8) [15]. Таким чином користувач може переглядати документ без зайвої інформації, а коли ця інформація знадобиться, він може знайти її, навівши курсор на відповідний термін. Розроблена система використовується на практиці [16].

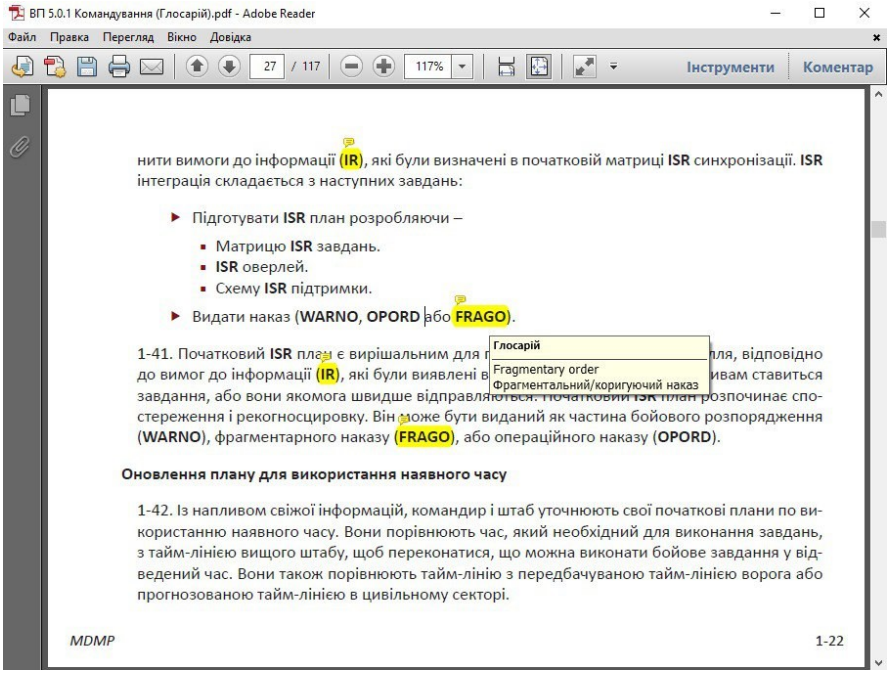

Рис. 8. Інформаційний файл, що генерує система

Висновки

У роботі досліджено процес побудови та функціонування системи ідентифікації термінів та абревіатур у текстових документах. Завдання з розроблення такої системи є актуальною, оскільки така проблема ідентифікації часто виникає у військовій сфері. Під час реалізації системи враховано те, що окремий термін чи абревіатура може мати кілька пояснень в різних нормативних документах. Всі наявні пояснення доєднуються до терміну або абревіатури, що враховано у процесі роботи системи. Особливістю системи є використання методів опрацювання природної мови, оскільки терміни можуть зустрічатися у різних відмінках. Для реалізації системи використано готові пакети Python для покриття подібних задач: Tkinter, PyMuPDF Наведено приклади функціонування системи. Розроблена система використовується на практиці.

Література

1. Klymovych O., Hrabchak V., Lavrut O., Lavrut T., Lytvyn V., Vysotska V. The diagnostics methods for modern communication tools in the Armed Forces of Ukraine based on neural network Approach. CEUR Workshop Proceedings. 2020. Vol. 2631. P. 198–208.

2. Pashchetnyk O., Lytvyn V., Zhyvchuk V., Polishchuk L., Vysotska V., Rybchak Z., Pukach Y. The ontological decision support system composition and structure determination for commanders of Land Forces formations and units in Ukrainian Armed Force. CEUR Workshop Proceedings. 2021. Vol. 2870. Kharkiv, Ukraine, April 22-23, 2021. P. 1077–1086.

3. Buschmann F., Meunier R., Rohnert H., Sommerlad P., Stal M. Pattern-Oriented Software Architecture. 2002.

4. Bass L., Clements P., Kazman R. Software Architecture in Practice. Addison-Wesley, 2003.

5. Gyansetu. Intro to NLP in Machine Learning. 12, 2020. URL: https://gyansetu.in/blogs/what-is-natural-language-processing/.

6. Trevor Hastie, Robert Tibshirani, Jerome H. Friedman The Elements of Statistical Learning. Trevor Hastie. Springer, 2001.

7. Stuart Russell, Peter Norvig Artificial Intelligence – A Modern Approach. Prentice Hall. Stuart Russell. 2002.

8. Evergreen. UML: для чого потрібні діаграми процесів. Лютий 2021. URL: https://evergreens.com.ua/ua/articles/uml-diagrams.html.

9. Jacobsen Ivar, Magnus Christerson, Patrik Jonsson, Gunnar Overgaard. Object Oriented Software Engineering. Jacobsen, 1992.

10. Visual Studio Code. Why Visual Studio Code? June, 2021. URL: https://code.visualstudio.com/docs/editor/whyvscode.

11. Simplilearn. Node.js vs. Java: Differences, Applications, and Why You Should Learn Them. April 12, 2021. URL: https://www.simplilearn.com/node-js-vs-java-article/.

12. Coursera. What Does a Front-End Developer Do? September 30, 2022. URL: https://www.coursera.org/articles/front-end-developer/.

13. Coursera. What Does a Back-End Developer Do? September 30, 2022. URL: https://www.coursera.org/articles/back-end-developer/.

14. Search Unified Communications. What is a real-time application (RTA)? Definition and Examples. May 2022. URL: https://www.techtarget.com/searchunifiedcommunications/definition/real-time-application-RTA/.

15. Берко А. Ю. Організація баз даних / А. Ю. Берко, О. М. Верес. – Університет «Львів. політехніка», 2003.

16. Smashing magazine. What Is User Experience Design? October 2010. URL: https://www.smashingmagazine.com/2010/10/what-is-user-experience-design-overview-tools-and-resources/.

References

1. Klymovych O., Hrabchak V., Lavrut O., Lavrut T., Lytvyn V., Vysotska V. The diagnostics methods for modern communication tools in the Armed Forces of Ukraine based on neural network Approach. CEUR Workshop Proceedings. 2020. Vol. 2631. P. 198–208.

2. Pashchetnyk O., Lytvyn V., Zhyvchuk V., Polishchuk L., Vysotska V., Rybchak Z., Pukach Y. The ontological decision support system composition and structure determination for commanders of Land Forces formations and units in Ukrainian Armed Force. CEUR Workshop Proceedings. 2021. Vol. 2870. Kharkiv, Ukraine, April 22-23, 2021. R. 1077–1086.

3. Buschmann F., Meunier R., Rohnert H., Sommerlad P., Stal M. Pattern-Oriented Software Architecture. 2002.

4. Bass L., Clements P., Kazman R. Software Architecture in Practice. Addison-Wesley, 2003.

5. Gyansetu. Intro to NLP in Machine Learning. 12, 2020. URL: https://gyansetu.in/blogs/what-is-natural-language-processing/.

6. Trevor Hastie, Robert Tibshirani, Jerome H. Friedman The Elements of Statistical Learning. Trevor Hastie. Springer, 2001.

7. Stuart Russell, Peter Norvig Artificial Intelligence – A Modern Approach. Prentice Hall. Stuart Russell. 2002.

8. Evergreen. UML: dlia choho potribni diahramy protsesiv. Liutyi 2021. URL: https://evergreens.com.ua/ua/articles/uml-diagrams.html.

9. Jacobsen Ivar, Magnus Christerson, Patrik Jonsson, Gunnar Overgaard. Object Oriented Software Engineering. Jacobsen, 1992.

10. Visual Studio Code. Why Visual Studio Code? June, 2021. URL: https://code.visualstudio.com/docs/editor/whyvscode.

11. Simplilearn. Node.js vs. Java: Differences, Applications, and Why You Should Learn Them. April 12, 2021. URL: https://www.simplilearn.com/node-js-vs-java-article/.

12. Coursera. What Does a Front-End Developer Do? September 30, 2022. URL: https://www.coursera.org/articles/front-end-developer/.

13. Coursera. What Does a Back-End Developer Do? September 30, 2022. URL: https://www.coursera.org/articles/back-end-developer/.

14. Search Unified Communications. What is a real-time application (RTA)? Definition and Examples. May 2022. URL: https://www.techtarget.com/searchunifiedcommunications/definition/real-time-application-RTA/.

15. Berko A. Yu. Orhanizatsiia baz danykh / A. Yu. Berko, O. M. Veres. – Universytet «Lviv. politekhnika», 2003.

16. Smashing magazine. What Is User Experience Design? October 2010. URL: https://www.smashingmagazine.com/2010/10/what-is-user-experience-design-overview-tools-and-resources/.