Issue 16, 2024

https://doi.org/10.23939/sisn2024.16.035

УДК 004.9

ІНТЕЛЕКТУАЛЬНА СИСТЕМА КОМПЛЕКСНОГО АНАЛІЗУ ВІЙСЬКОВОЇ ІНФОРМАЦІЇ НА ОСНОВІ МАШИННОГО НАВЧАННЯ ТА NLP ДЛЯ ДОПОМОГИ КОМАНДИРАМ ТАКТИЧНИХ ЛАНОК

Віталій Данилик[1], Василь Литвин[2], Вікторія Висоцька[3]

[1, 2, 3] Національний університет "Львівська політехніка",
кафедра інформаційних систем та мереж, Львів, Україна,
[1] E-mail: Vitalii.M.Danylyk@lpnu.ua, ORCID: 0000-0001-5928-7235
[2] E-mail: Vasyl.V.Lytvyn@lpnu.ua, ORCID: 0000-0002-9676-0180
[3] E-mail: Victoria.A.Vysotska@lpnu.ua, ORCID: 0000-0001-6417-3689

У статті описано результати дослідження процесів комплексного аналізу військової інформації на основі машинного навчання та опрацювання природньої мови для допомоги командирам тактичних ланок. Система повинна давати змогу користувачам мати такі можливості: об'єднання словника та інформаційного матеріалу, додавання термінів та абревіатур у словник, класифікація об'єктів для радіотехнічної розвідки, візуалізація повітряних об'єктів, класифікація повітряних об'єктів, користування інформаційними матеріалами, організування інформаційних матеріалів. Розроблена інтелектуальна система складається з чотирьох модулів, а саме: з модуля інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур, модуля класифікації об'єктів для радіотехнічної розвідки, модуля візуалізації та класифікації повітряних об'єктів в реальному часі та модуля структуризації військової інформації. Також у системі розроблено модуль правлення орфографічних та граматичних помилок у тексті на основі алгоритму перебору та словника із 30000 слів українською мовою. В статті подано загальну структуру розробленої системи та відповідно структури, алгоритми функціонування кожного розробленого модуля системи. Наведено також функціональні вимоги до системи та окремо до кожного модуля. Здійснений опис експериментальної апробації розробленого програмного забезпечення.

Ключові слова: військові дані, командир тактичних ланок, структуризація інформації, опрацювання природної мови, класифікація об'єктів, інтелектуальна система, машинне навчання, виправлення помилок, наївний класифікатор Баєса, метод машини опорних векторів, k найближчих сусідів, логістична регресія.

Вступ

Посада командира передбачає опрацювання великої кількості інформації. Будь-які рішення мають бути детально обдумані, підкріплені фактами та раціональними передбаченнями їхніх потенційних результатів [1–2]. Командир має зважати на всі фактори та керуватись уже заздалегідь продуманими й описаними тактиками та стратегіями. Також одним із головних чинників хорошого командування є швидкість. Завдання командира – це швидке та обдумане командування та управління. Але знайти правильне рішення і водночас швидко є складним завданням. Адже щоб це зробити, потрібно швидко аналізувати всю інформацію загалом, а подекуди це неможливо. Сучасні новітні технології розвиваються дуже швидко. Існує безліч сервісів, які полегшують роботу бага-

тьом людям. Вони мають активно використовуватися і у військовій сфері. Новітні інтелектуальні системи (ІС) зможуть покрити можливі проблеми та допоможуть офіцерам приймати швидкі та раціональні рішення. У процесі роботи командири можуть зіткнутися з 3-ма видами проблем: дефіцит знань; повільний аналіз інформації; незрозумілість інформаційних матеріалів. Одна ІС не може вирішити проблеми, з якими може зіткнутися командир тактичних ланок. Відповідно доцільним є створення комплексу ІС. Адже кожна з ІС комплексу сприятиме вирішенню своєї частини проблем і в такий спосіб це допоможе офіцерам приймати швидкі й обдумані рішення.

Аналіз останніх досліджень та публікацій

Станом на 2024 рік можна виділити низку популярних та успішних програмних рішень, які здійснюють діяльність значний період часу і які змогли себе зарекомендувати як системи, однозначно варті ознайомлення (табл. 1).

Таблиця 1

Порівняльна таблиця характеристик для МКР та найближчих аналогів

Характеристики	МКР	Найближчі аналоги		
		Fandom	Klassify	Dataedo
Платформа	4	5	4	5
Операційна система	5	5	5	5
Мова програмування	Краще	Гірше	Краще	Гірше
Необхідні ресурси	менше	більше	більше	більше
Інтероперабельність	Так	Ні	Ні	Ні
Мобільність	Ні	Так	Так	Так
Масштабованість	Так	Так	Ні	Так
Взаємодія з користувачем	5	4	4	3
Функціональність	5	4	4	4
Придатність до використання	5	5	4	3
Надійність	5	5	4	4
Продуктивність	5	5	4	5
Експлуатаційна придатність	4	5	5	5

Dataedo – один із найпопулярніших інструментів словника даних. Klassify, Data Classification Suite – це платформа класифікації даних, яка допомагає організаціям донести політику класифікації даних до своїх користувачів і змусити їх застосовувати відповідну класифікацію на основі чутливості даних, для обміну ними та зберігання. Fandom – це американський вікі-сервіс, відомий як Wikia – вікі-програмне забезпечення з відкритим кодом.

При макетуванні текстових блоків актуальним залишається проблема швидкої, якісної, а головне правильної операції правлення текстів [3]. Тому створення інформаційної технології макетування з вбудованим словником термінів українською мовою, опціями виправлення орфографії текстів українською мовою та можливостями збереження результату роботи програми у текстовому форматі або у форматі Pdf є важливим завданням [4]. У [3–4] проаналізовано алгоритм виправлення орфографічних помилок у текстах українською мовою з допомогою словника, що містить більш ніж 30000 слів української мови, і поданого окремим текстовим файлом. Наповнення словника проведено на основі вмісту сучасних словників української мови. Дослідники проаналізували відомі алгоритми правлення текстів. Завдання пошуку зводиться до аналізу рядків текстів під час опрацювання текстової інформації. Це означає виділення збігу підрядків двох рядків. В узагальненому завданні відбувається локалізація всіх операцій порівняння тексту. У статті відзначено, що, незважаючи на існування різних способів вирішення цієї проблеми, здебільшого вони зводяться до завдань текстового пошуку й асоціативного мислення. У процесі роботи розглянуто алгоритми

перебору та Кнута-Моріса-Прата. Розроблена програма надає можливість виправляти помилки набору та адаптовує текстову інформацію до друку. Проаналізовано програми-аналоги, які займаються опрацюванням тексту і проведенням коректури в автоматичному режимі. Розглянуто програми, як-от: “Типограф” (розроблення студії Артемія Лебєдєва), “Українська типографія і переноси” (розроблення Олександра Жиденка), “Автотипографіка” (розробник Webcode Group), “Типограф” (розробник Eugene Spearance). Показано, що наявні аналоги не вирішують питання проведення коректури українського тексту в автоматизованому режимі. Створено перелік правил набору тексту, який використовується на етапі коректури. Побудовано алгоритми виправлення помилок згідно з переліком правил набору тексту, що дає змогу виправляти помилки в автоматичному режимі. В [3–4] проаналізовано функції розробленого програмного продукту, в якому реалізовано всі вищеописані правила коректури, і результатом його роботи є відкоректована текстова інформація [5]. На рис. 1 показано панель управління програми-редактора, що надає можливість редагувати тексти українською мовою. Головне меню програми представлене у вигляді набору спадних меню. Натисканням опції File відкривається спадне меню, яке дає змогу відкрити новий документ, зберегти наявний, відправити документ на друк чи вийти з програми. Опція Edit містить функції для роботи з текстом: копіювання, вирізання, вставляння чи видалення тексту. Опція Format відповідає за основні функції форматування: накреслення та вирівнювання шрифту. Опція Search дає можливість скористатися знаходженням потрібної частини слова чи цілого слова у тексті. Опція About представляє інформацію про авторів програми. Кнопки “Зберегти *.txt” та “Зберегти *.ps” сприяють збереженню документу у відповідному форматі. Натисканням на кнопку “Правити текст” розпочинається перевірка орфографії. Алгоритм виконується у такій послідовності: програма виділяє окреме слово з уведеного користувачем тексту. Воно порівнюється з усіма словами у словнику. Якщо знайдено таке саме слово або слово відсутнє у словнику, то воно лишається без змін. У разі, коли у слові допущено одну чи дві помилки, програма виділяє слово червоним кольором, після чого відкривається діалогове вікно, яке представлене на рис. 2.

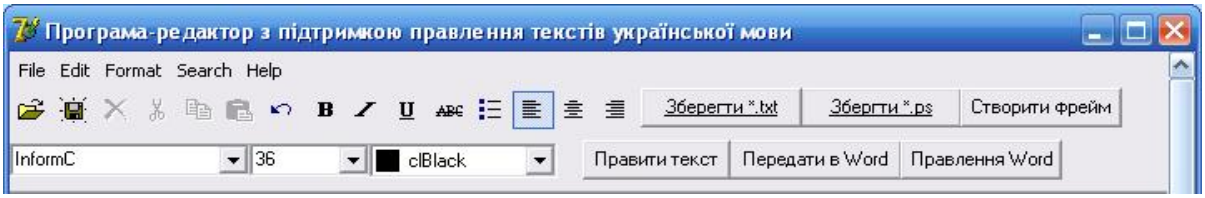


Рис. 1. Панель управління програми-редактора з підтримкою правлення текстів

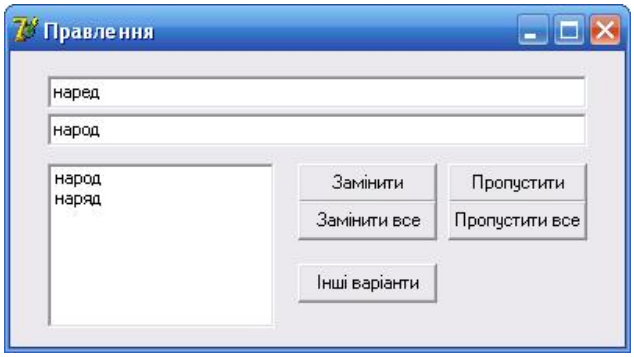


Рис. 2. Діалогове вікно правлення україномовних слів

Користувач має можливість замінити слово на запропонований варіант чи обрати один з альтернативних варіантів, поданих у відповідному полі. Також користувач може залишити слово без змін. На бажання можна замінити одразу всі слова на варіанти, вибрані програмою, чи відмі-

нити одразу всі зміни. У програмі реалізована можливість форматування україномовного тексту. Файл набирають вручну або завантажують готовий текстовий файл натисканням на кнопку “Відкрити”. На панелі інструментів є засоби для форматування тексту. У відповідних спадних меню користувач обирає гарнітуру, кегель і колір шрифту. Натисканням кнопки, яка відповідає за вирівнювання тексту, користувач вирівнює текст відповідно до центру чи країв сторінки. В результаті роботи редактора тексту створюється PostScript файл, який можна перетворити у PDF-файл. Для створення PDF із PostScript-файлу потрібно скористатися програмою Adobe Distiller, яка входить у пакет Adobe Acrobat. У [3–4] запропоновано алгоритм виправлення орфографічних помилок у словах на основі словника української мови. Словник представлено у вигляді окремого текстового файлу і містить понад 50000 слів української мови. Наповнення словника проведено згідно з чинними нормами. Для визначення ефективності цієї програми-редактора було вирішено порівняти результати її роботи на прикладі вибраних слів. Результати порівняння наведено у табл. 2.

Таблиця 2

Порівняння виправлених помилок у різному програмному забезпеченні [3]

Програмне забезпечення	Вибірка слів із помилками (допущено 16 помилок)	Слова після правлення	% виправлених помилок
Програма-редактор із підтримкою правлення текстів	магіср, лінґвстика, шерхтіння, епітит, шнтропія, ентозіузм, віжун, амирикансько-украінський, кравесь	маґістр, лінґвістика, шерехтіння, епітет, ентропія, ентузіазм, відун, американсько-український, кравець	95
ORFO	магіср, лінґвстика, шерхтіння, епітит, шнтропія, ентозіузм, віжун, амирикансько-украінський, кравесь	маґіср, лінґвстика, шерехтіння, епітет, ентропія, ентозіузм, відун, амирикансько-український, кравець	90
Microsoft Office Word	магіср, лінґвстика, шерхтіння, епітит, шнтропія, ентозіузм, віжун, амирикансько-украінський, кравесь	маґіср, лінґвстика, шерехтіння, епітет, ентропія, ентозіузм, віжун, амирикансько-український, кравець	75
QuarkXPress	Не підтримує виправлення орфографічних помилок		
Adobe InDesign	Не підтримує виправлення орфографічних помилок		
LaTeX	Не підтримує виправлення орфографічних помилок		

Програма-редактор із підтримкою правлення текстів працює зі словником української мови, в якому містяться маловживані слова. Для оцінки ефективності алгоритму та програмного забезпечення було вибрано слова української мови, які містять літеру “ґ”, оскільки закордонні розроблення такі слова не розрізняють. Було порівняно програму-редактор із можливостями правлення текстів української мови з такими програмами: Microsoft Office Word, ORFO, Adobe InDesign, QuarkXPress, LaTeX. Під час правлення Microsoft Office Word не розрізняє помилок у словах з літерою “ґ”, а система ORFO виправляє помилки у словах, проте літеру “ґ” заміняє на “г”. В інформаційних технологіях Adobe InDesign, QuarkXPress та LaTeX підтримку правлення орфографічних помилок у словах української мови не реалізовано. Представлена програма править слова, в яких допущено одну або дві помилки. Порядок букв, у яких допущені помилки, не впливає на алгоритм роботи програми. Програма вказує варіанти контекстної заміни і помилки виправляються, якщо слова є в словнику. У Microsoft Office Word та ORFO слова з помилками у двох літерах не виправляються.

Порівнявши розроблену інформаційну систему з найближчими аналогами, можна дійти висновку, що вона є як і актуальною, так і важливою для командирів тактичних ланок. Використовуючи таку систему, командири з великою ймовірністю пришвидшують процес управління, що підвищує прийняття правильного та раціонального рішення. Використання такої системи буде зручним для користувачів, адже вона буде мати тільки необхідний функціонал та зручний графічний інтерфейс. Різні програми системи будуть відповідати тільки за конкретну сферу діяльності та відповідно до неї матимуть свої переваги: кросплатформність, робота з даними в реальному масштабі часу, захищена робота в локальній мережі, інтеграція інформації в матеріали вже відомих та популярних програм [6–8].

Формулювання мети та завдання

Система повинна давати змогу користувачам мати такі можливості: об’єднання словника та інформаційного матеріалу, додавання термінів та абревіатур у словник, класифікація об’єктів для радіотехнічної розвідки, візуалізація повітряних об’єктів, класифікація повітряних об’єктів, користування інформаційними матеріалами, організування інформаційних матеріалів.

Мета роботи – розроблення інтелектуальної системи комплексного аналізу військової інформації на основі машинного навчання та опрацювання природньої мови для допомоги командирам тактичних ланок. Для цього потрібно виконати такі завдання:

1. Проаналізувати засоби вирішення проблем та побудувати концептуальну модель.
2. Розробити модуль інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур.
3. Розробити модуль класифікації об’єктів для радіотехнічної розвідки.
4. Розробити модуль візуалізації та класифікації повітряних об’єктів в реальному часі.
5. Розробити модуль структуризації військової інформації.
6. Об’єднати розроблені модулі для формування комплексної інформаційної системи.
7. Провести тестування та впровадити проєкт.

Для узагальнення деталей користувачів системи та їхньої взаємодії із системою [9] взято діаграму варіантів використання (рис. 3). Комплексна інформаційна система опрацювання військових даних для допомоги командирам тактичних ланок містить певну кількість програм. Ці програми є не обов’язково одного виду та не обов’язково самостійні. Вони можуть використовувати зовнішні сервіси та самі бути вебсервісом. Тож робота деяких програм буде більш складною ніж проста програма для персонального комп’ютера. Для відображення обчислювальних вузлів під час роботи програм, компонентів та об’єктів, що виконуються на цих вузлах, використано діаграму розгортання (рис. 4). Отже, можна побачити, як саме розгортається інформаційна система, як взаємодіють компоненти між собою та із зовнішніми об’єктами.

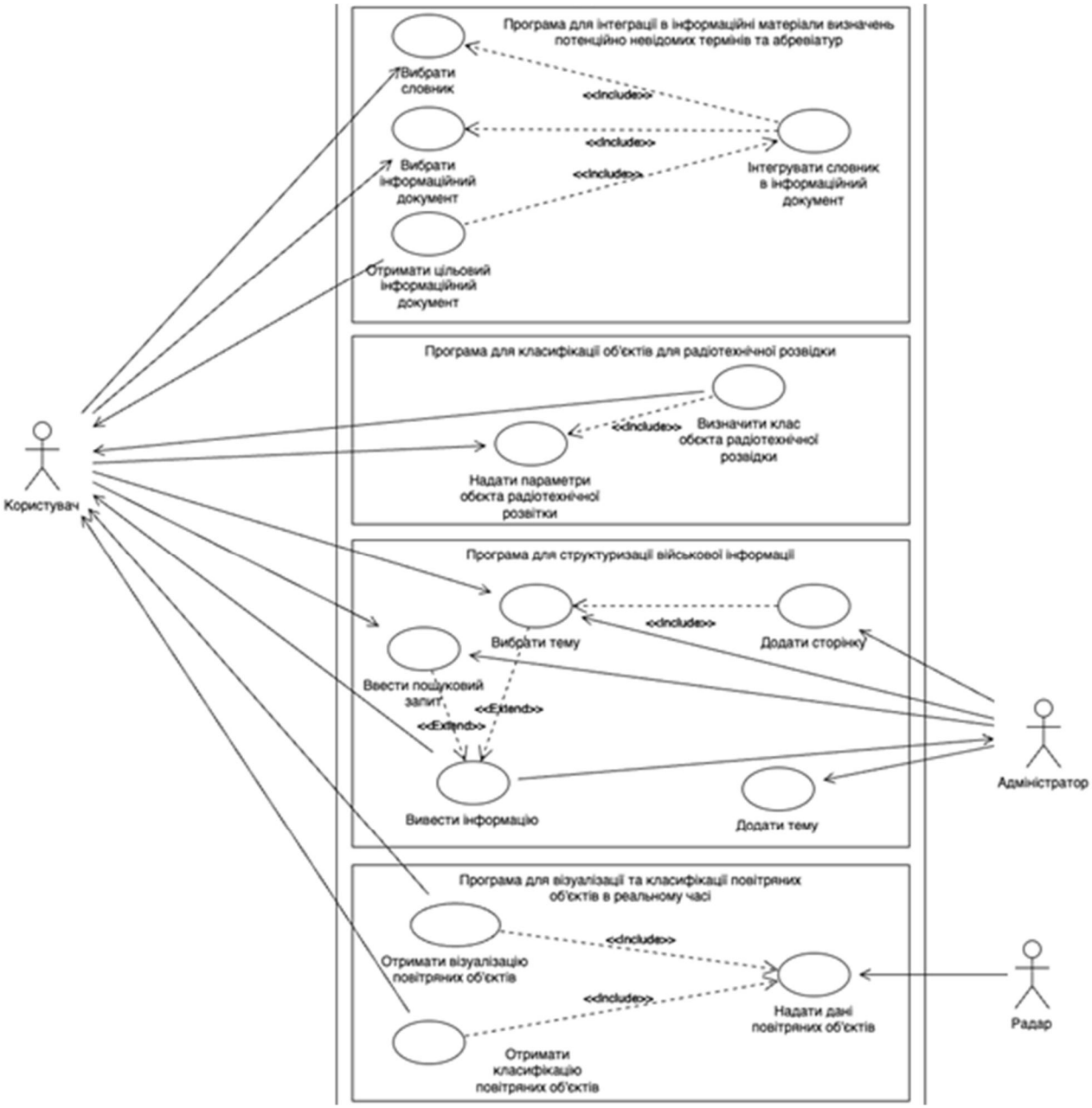


Рис. 3. Діаграма варіантів використання системи

Діаграми, зображені на рис. 1 та рис. 2, описують систему загалом. Для детальнішого опису процесів необхідно описувати кожну програму, що є в системі, яка розробляється. Програма для структуризації військової інформації передбачає роботу 2-х користувачів. Відповідно для відображення всіх процесів, що відбуваються в програмі, потрібно дві діаграми – діаграма активності для відвідувача (рис. 3) та діаграма активності для адміністратора (рис. 4).

Адміністратор зазначеної програми є користувачем із широкими повноваженнями. Він має такі самі можливості, що і звичайний користувач (рис. 5), і також додаткові можливості, що є тільки в такого виду користувача (рис. 6). Концепція такої програми передбачає наявність 4-х об’єктів: користувач, адміністратор, програма та база даних. Для відображення взаємодій цих об’єктів використано діаграму послідовності (рис. 7). Також ця діаграма показує взаємодії об’єктів, упорядковані за часом.

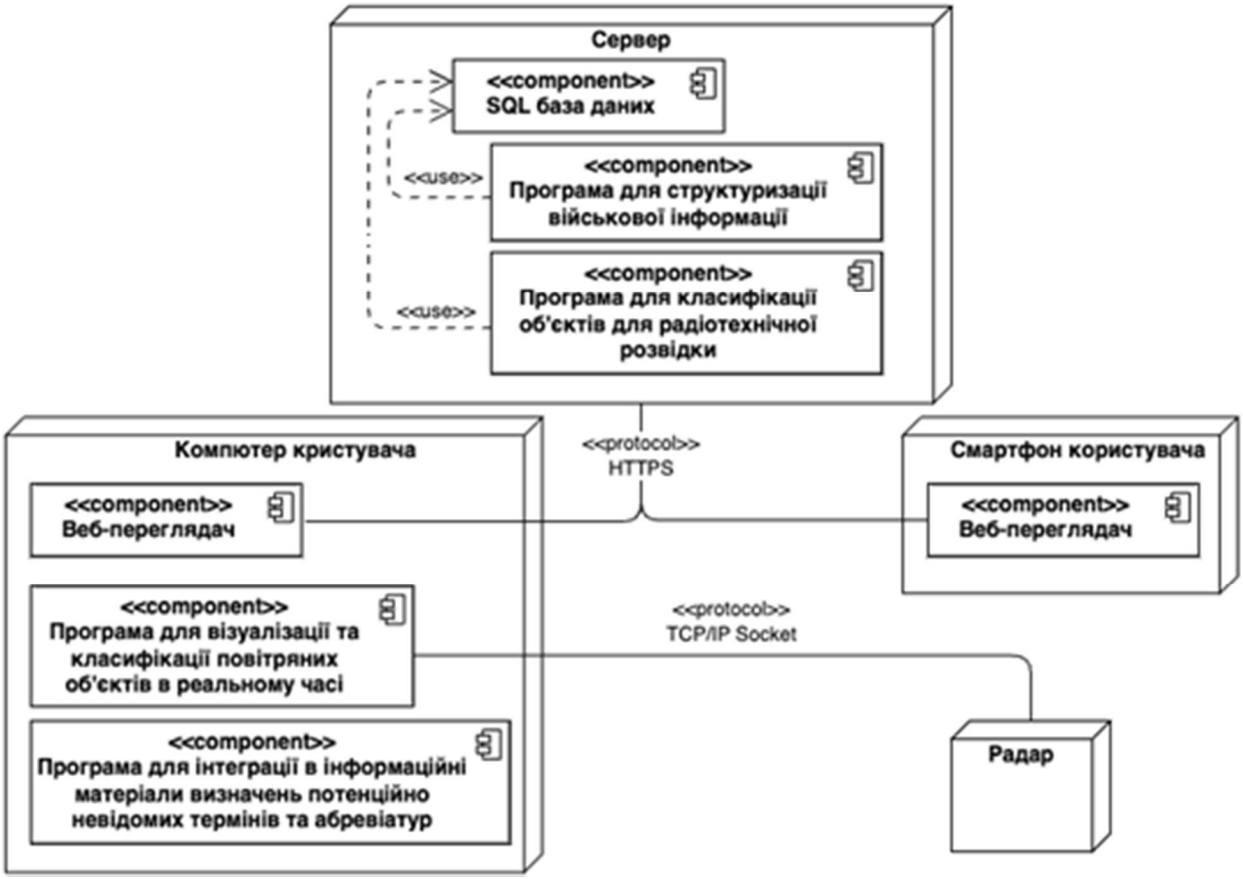


Рис. 4. Діаграма розгортання системи

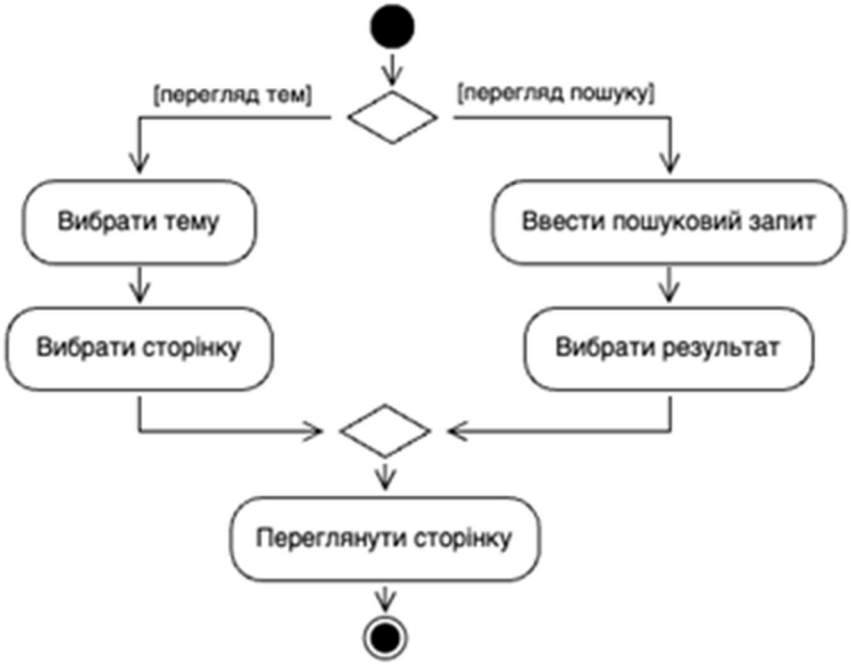


Рис. 5. Діаграма активності для відвідувача програми для структуризації військової інформації

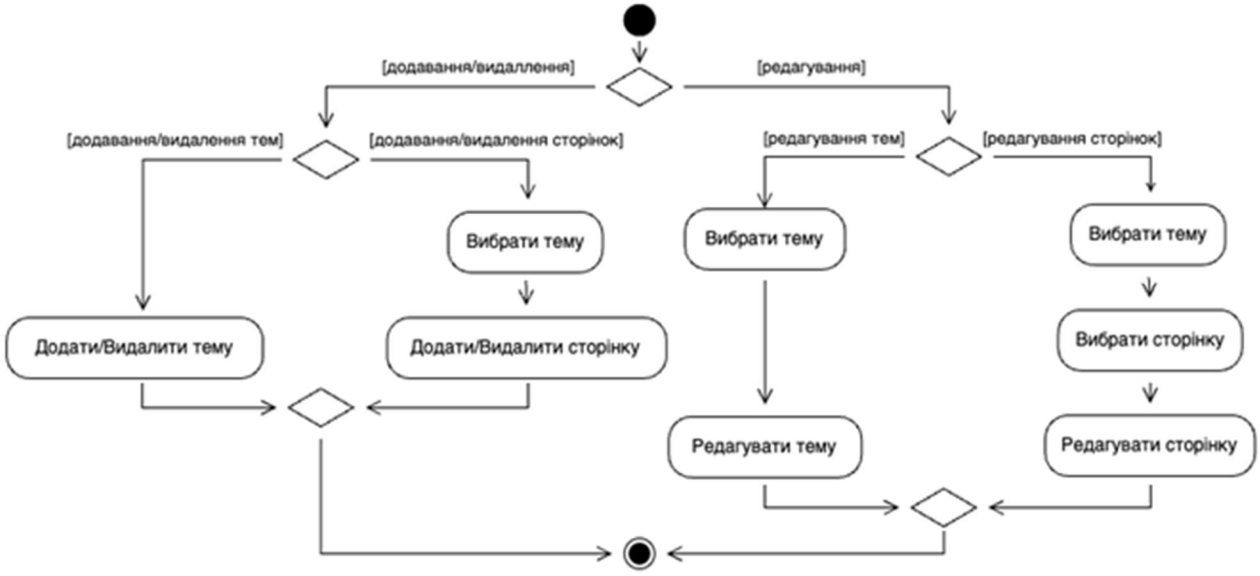


Рис. 6. Діаграма активності для структуризації військової інформації

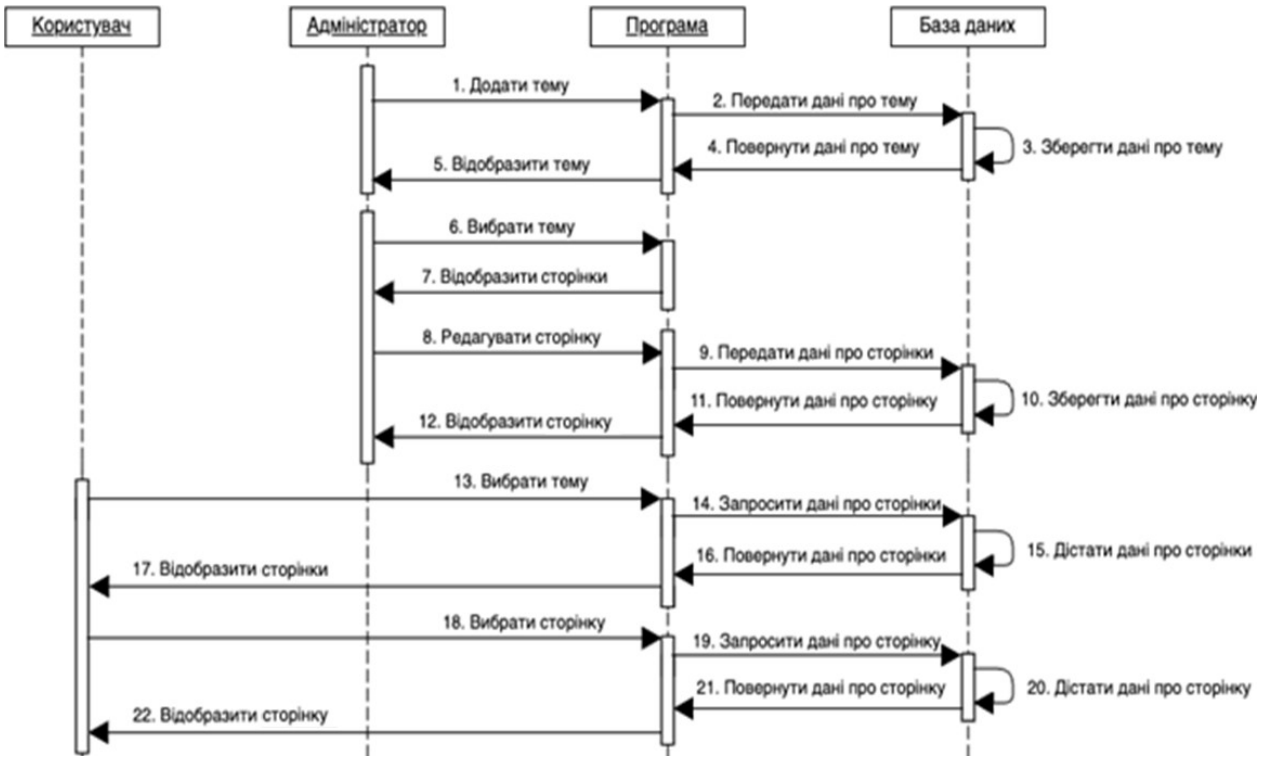


Рис. 7. Діаграма послідовності програми для структуризації військової інформації

Виклад основного матеріалу

Програма для візуалізації та класифікації повітряних об’єктів у реальному часі передбачає роботу одного користувача, проте вона взаємодіє із зовнішнім джерелом інформації – радаром (рис. 8, 9, *а*, *б*). Програма для інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур передбачає роботу одного користувача. Вона є самостійною, не взаємодіє із зовнішніми об’єктами. Програма, що розробляється, використовує файли інших програм, зокрема PDF для інформаційних файлів, в які будуть додаватися невідомі терміни та абревіатури зі

словника, та файл із словником, що має розширення CSV. Для використання словника такого типу потрібно його перетворити в інший формат (рис. 10).

Рис. 8. Діаграма: а – станів; б – послідовності програми для візуалізації та класифікації повітряних об'єктів у реальному часі

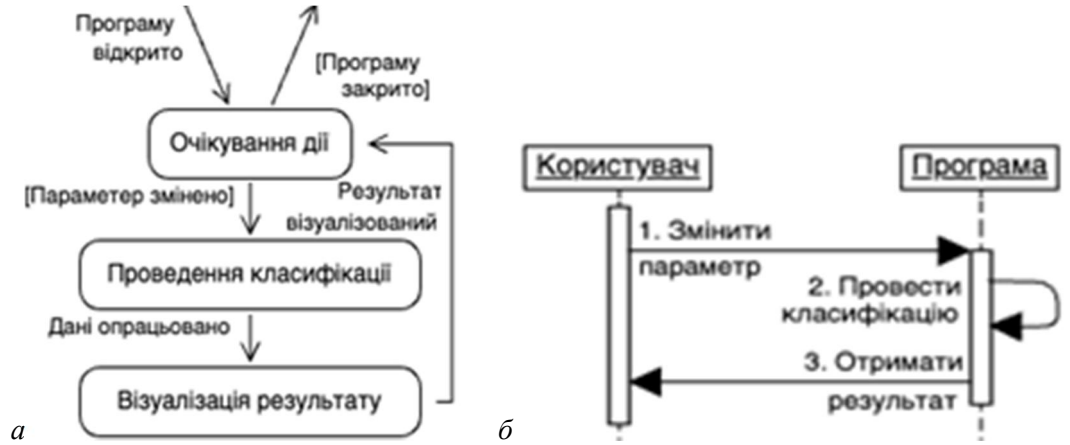


Рис. 9. Діаграма станів програми для класифікації об'єктів для радіотехнічної розвідки

Рис. 10. Діаграма станів програми для інтеграції в інформаційні матеріали визначень потенційно невідомих термінів та абревіатур

Концепція згаданої програми передбачає наявність двох об’єктів – користувача та програми. Діаграма послідовності (рис. 11, *а*) відображає процеси взаємодії цих об’єктів. На діаграмі не подано інших програм, які утворюють інформаційний файл чи формують словник, оскільки для роботи з програмою для інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур потрібно мати вже готові файли для об’єднання. Вони використовуються як вхідні дані. Змоделювавши процеси, які відбуваються в інформаційній системі, з’являється уявлення, як саме система має функціонувати. Для її реалізації необхідно змоделювати саму об’єктну модель програмного забезпечення [10].

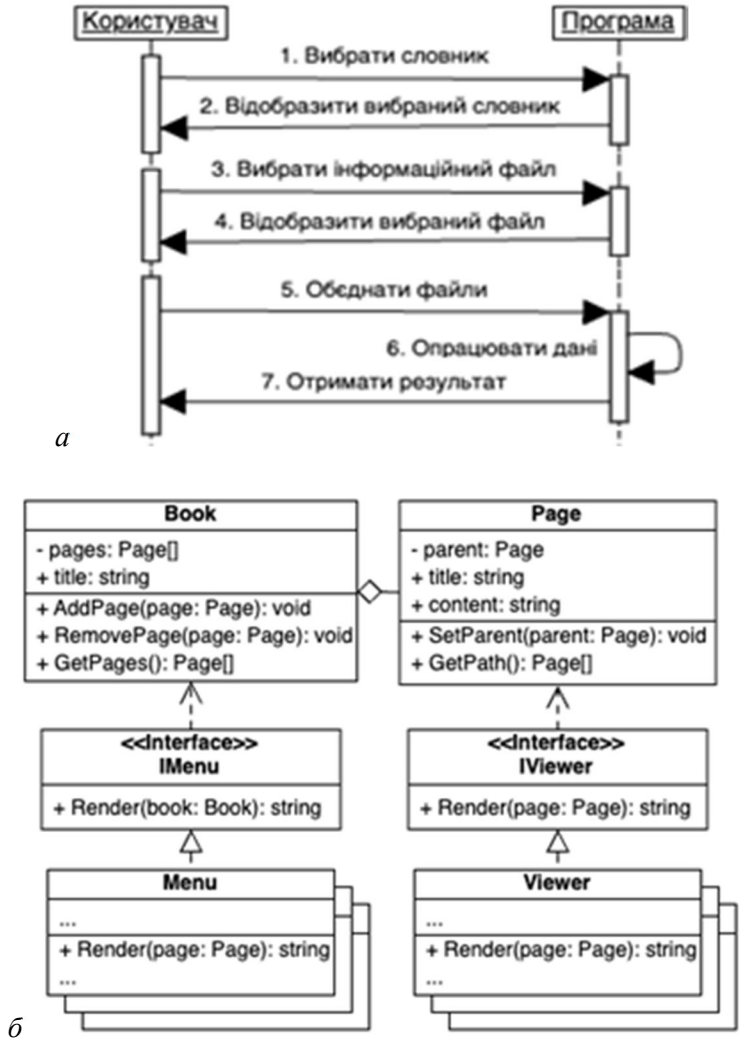


Рис. 11. Діаграма: а – послідовності програми для інтеграції в інформаційні матеріали для визначення невідомих термінів та абревіатур; б – класів програми для структуризації військової інформації

Програма для структуризації військової інформації передбачає наявність зручної навігації по сторінках, хорошої візуалізації інформації та ієрархічне з’єднання сторінок. Для вирішення цих основних завдань розроблено відповідну діаграму класів (Рис. 12, *б*). Інтерфейси IMenu та IViewer дають можливість експериментувати з різними реалізаціями меню та переглядача; без суттєвого втручання в архітектуру можна швидко підібрати кращу реалізацію. А наявність в класі Page поля children дає можливість використовувати ієрархічне з’єднання сторінок [11].

Програма для візуалізації та класифікації повітряних об’єктів у реальному масштабі часу передбачає наявність класифікатора, який буде класифікувати об’єкти. Проте класифікаційні алго-

ритми та дані можуть швидко змінюватися. Для вирішення цього основного завдання була розроблена відповідна діаграма класів (рис. 12). Використання шаблонних класів надає архітектурі можливість змінювати клас, який відповідає за дані об'єкту, що піддається класифікації і потенційно може змінюватися. Тож створюється новий клас, який реалізує інтерфейс IClassifier відповідного типу для опрацювання нових моделей даних чи нових алгоритмів [12–17].

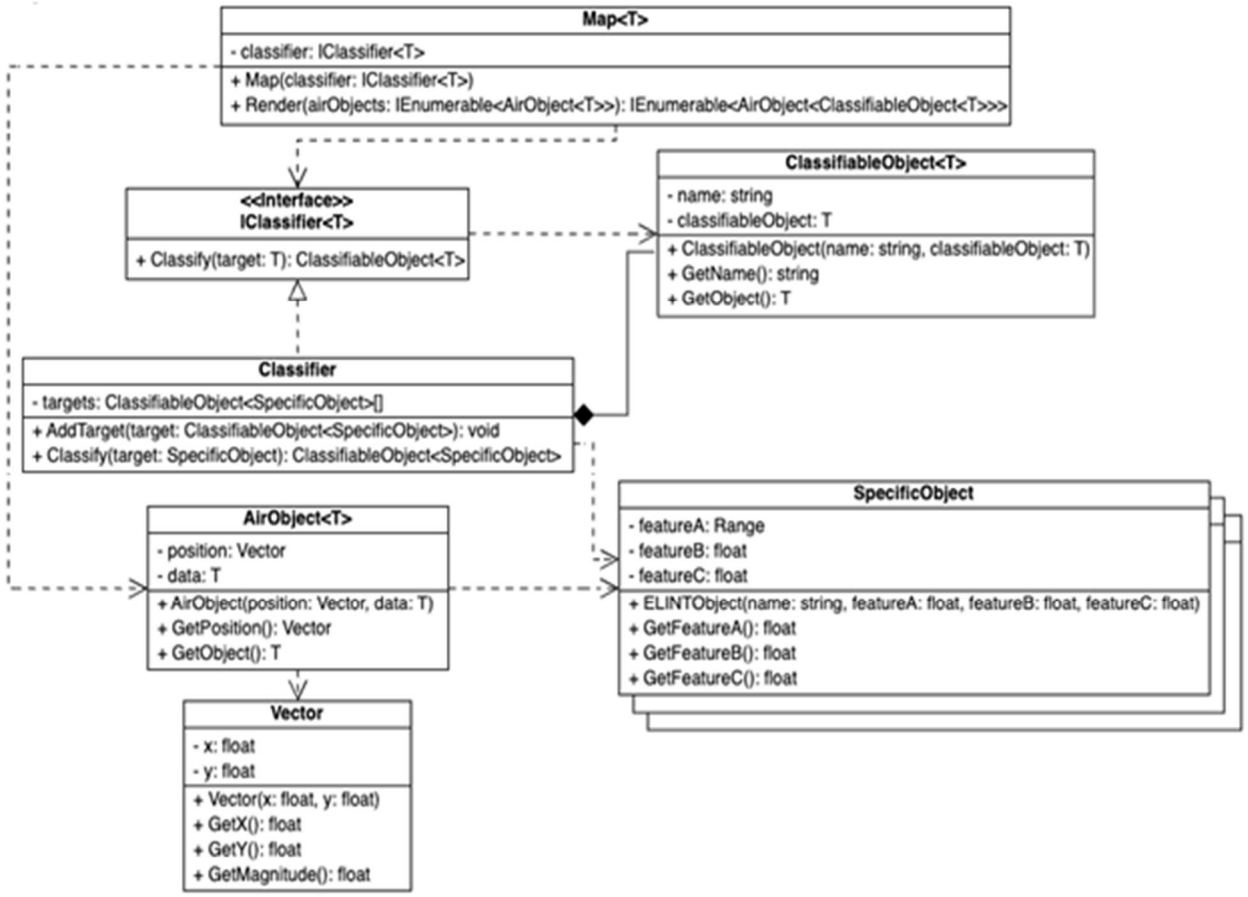


Рис. 12. Діаграма класів візуалізації та класифікації повітряних об'єктів в реальному масштабі часу

Програма для класифікації об'єктів для радіотехнічної розвідки передбачає як і попереднього разу, наявність класифікатора, який буде класифікувати об'єкти. Загальна схема дуже схожа до випадку з програмою для візуалізації та класифікації повітряних об'єктів у реальному масштабі часу. Для вирішення цього основного завдання розроблено відповідну діаграму класів (рис. 13). Незважаючи на те, що основне завдання є таким самим, що і попереднього разу, проте класифікатор усе ж відрізняється. Він має повертати ймовірності всіх класів, а значить – і спеціальні об'єкти, як і зображено на діаграмі.

Програма для інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур 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 розроблено відповідну діаграму класів (рис. 14). Для хорошого дизайну архітектуру спроєктовано на базі інтерфейсів. Тож можна дописувати різну реалізацію інтерфейсів IResource та IDictionary, що надає можливість у майбутньому розширювати програму для багатьох інших типів файлів.

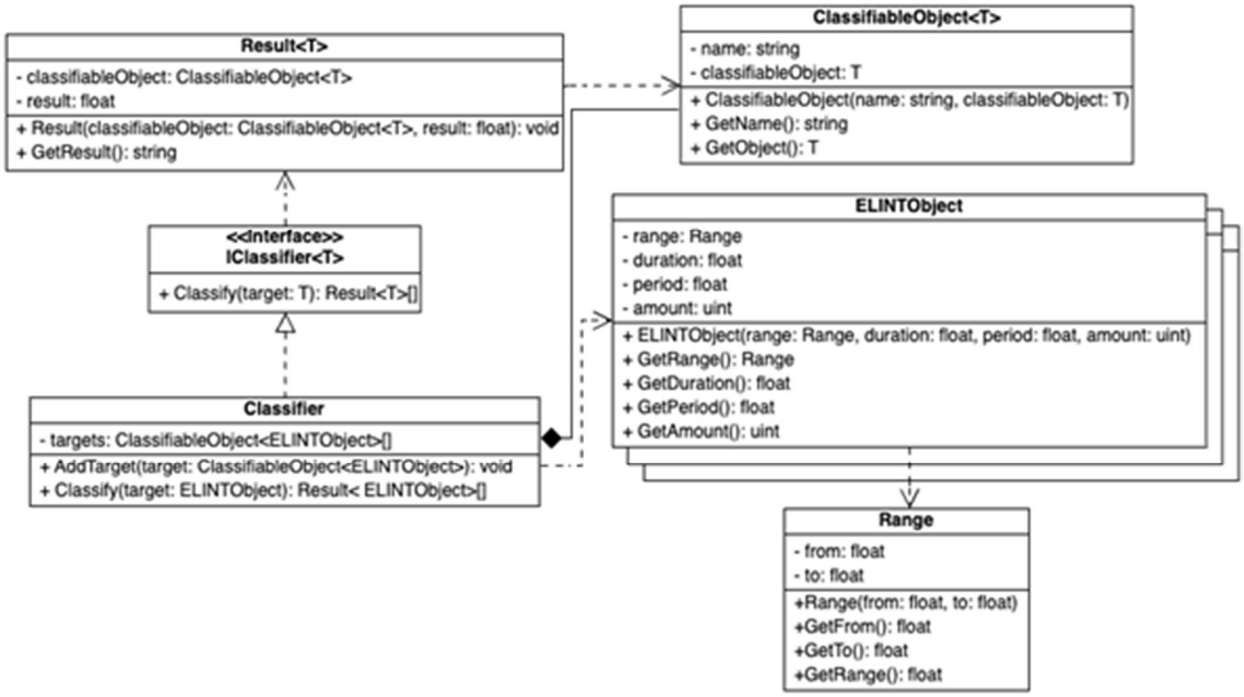


Рис. 13. Діаграма класів програми для класифікації об'єктів для радіотехнічної розвідки

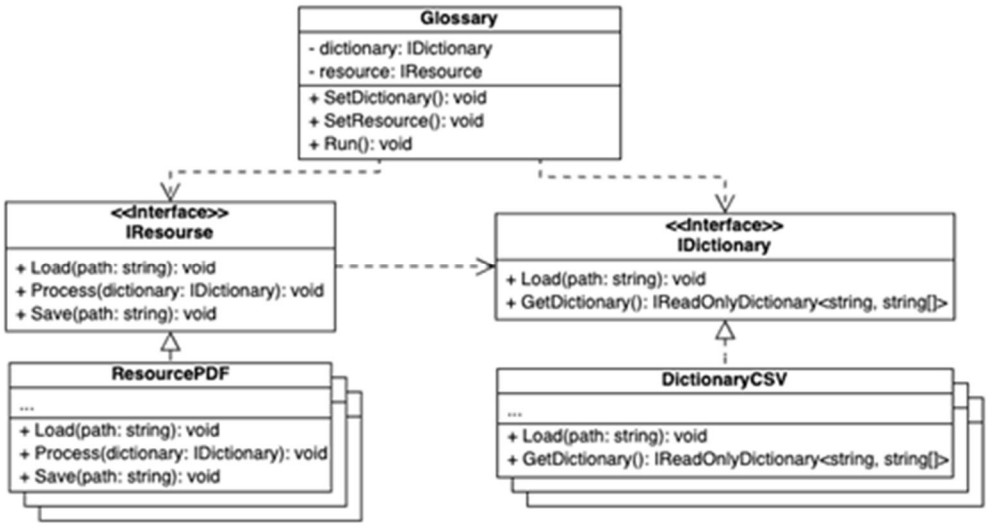


Рис. 14. Діаграма класів програми інтеграції в інформаційні матеріали з визначеннями термінів

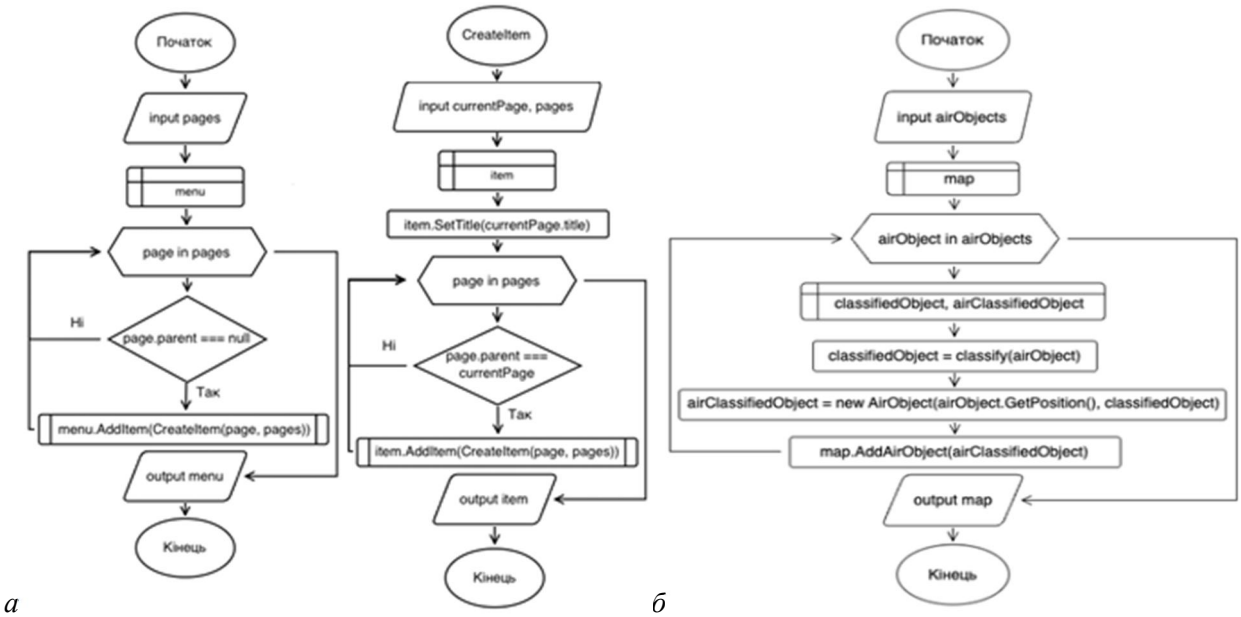
Комплексна інформаційна система опрацювання військових даних для допомоги командирам тактичних ланок складається з низки програм різних за призначенням, але об'єднаних однією метою. Відповідно кожна з них має різні вимоги та різні основні функціональні особливості.

1. Опрацювання природної мови (NLP) поєднує комп'ютерну лінгвістику – моделювання людської мови на основі правил – зі статистичними моделями, моделями машинного та глибинного навчання [18–21]. Переважно дослідження з опрацювання природної мови значною мірою покладаються на машинне навчання [22–26]. Парадигма машинного навчання передбачає використання статистичних методів для автоматичного вивчення правил за допомогою аналізу великих корпусів текстів (форма множини корпусу, це набір документів, можливо, з людськими або комп'ютерними анотаціями) та типових прикладів із реального світу. Однак основним недоліком статистичних методів є те, що вони вимагають детального розроблення ознак. Отже, галузь значною мірою від-

мовилася від статистичних методів і перейшла до нейронних мереж для машинного навчання [27–33]. Популярні методи передбачають використання вбудовування слів для фіксації семантичних властивостей слів і збільшення наскрізного вивчення завдання вищого рівня (наприклад, відповіді на запитання) замість виконання окремих проміжних завдань (наприклад, тегування частин мови та розбір залежностей) [33–39]. Опрацювання природної мови є необхідним функціоналом для програми, що сприяє інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур.

2. Метод класифікації. Однією з особливостей є класифікація даних. Класифікація – це процес розпізнавання, розуміння та групування ідей і об’єктів у попередньо встановлені категорії або “підгрупи” [9]. Використовуючи попередньо класифіковані навчальні набори даних, програми машинного навчання використовують різноманітні алгоритми для класифікації майбутніх наборів даних за категоріями. Існує кілька основних методів класифікації: логістична регресія, наївний класифікатор Баєса, k-найближчих сусідів, дерево рішень та машина опорних векторів [12–18].

Кожний модуль ІС має свої власні унікальні алгоритми. Ці алгоритми потрібно розбирати окремо. Проте є також не основні алгоритми, вони організовують роботу програми таким способом, щоб вона могла правильно функціонувати. Ці алгоритми не є особливими, вони сформовані за основними принципами програмування і просто необхідні ІС для функціонування. Програма для структуризації військової інформації передбачає опрацювання алгоритму формування ієрархічного меню (рис. 15, *а*). Алгоритм опрацьовує об’єкти сторінок, які пов’язані між собою однонаправленим зв’язком. Функція повинна опрацьовувати ці зв’язки і формувати об’єкти меню, що має ієрархічну будову. Алгоритм реалізовано за допомогою рекурсії, яка дала можливість динамічно формувати підпункти для кожного пункту меню.



а *б*

Рис. 15. Алгоритм: а – структуризації військової інформації та б – візуалізації та класифікації повітряних об’єктів у реальному масштабі часу

Модуль візуалізації та класифікації повітряних об’єктів у реальному часі передбачає опрацювання алгоритму формування моделі карти (рис. 15, *б*). Алгоритм отримує на вхід дані з радара. Вони містять інформацію про повітряні об’єкти. Алгоритм опрацьовує ці дані за допомогою класифікатора, що використовує алгоритм k найближчих сусідів та формує структуру даних, яка буде містити модель карти з усіма класифікованими повітряними об’єктами в реальному часі.

Модуль класифікації об’єктів для радіотехнічної розвідки передбачає опрацювання унікального алгоритму класифікації (рис. 14, *а*). Алгоритм опрацьовує процес, сформований на підходах методів k найближчих сусідів та наївного класифікатора Баєса. Він вираховує значення помилок, щоб визначити, наскільки віддалена характеристика від цільової характеристики. Використовуючи логарифми та функцію ступеня, нормалізує дистанції та робить допустимий діапазон. Також для алгоритму кожна характеристика є однаковою з точки зору ваги, тож якщо 3 з 4 характеристики будуть мати значення 1, а остання – значення 0, то тоді результат буде 75 %. Після того, як усі значення ймовірностей для класів отримано, алгоритм формує об’єкт, щоб повернути кінцеву структуру даних з усією інформацією. Програма для інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур передбачає опрацювання алгоритму формування структури даних словника, яка з’єднує всі слова з усіма визначеннями (рис. 16, *б*). Алгоритм отримує на вхід дані словника в текстовій формі з метою перетворення цих даних 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). В словнику може бути декілька визначень для декількох слів. Цей алгоритм визначає групи та об’єднує сутності для укомплектування декількох визначень одного слова чи словосполучення. В результаті ці об’єкти об’єднуються в єдину структуру даних, що повертається. Основна частина розроблення ПЗ передбачає два паралельні етапи – розроблення частини клієнта [21–23] та серверної частини програми [24–25]. Перед розробленням проєкту розгорнутий локальний сервер XAMPP, створена база даних (рис. 17).

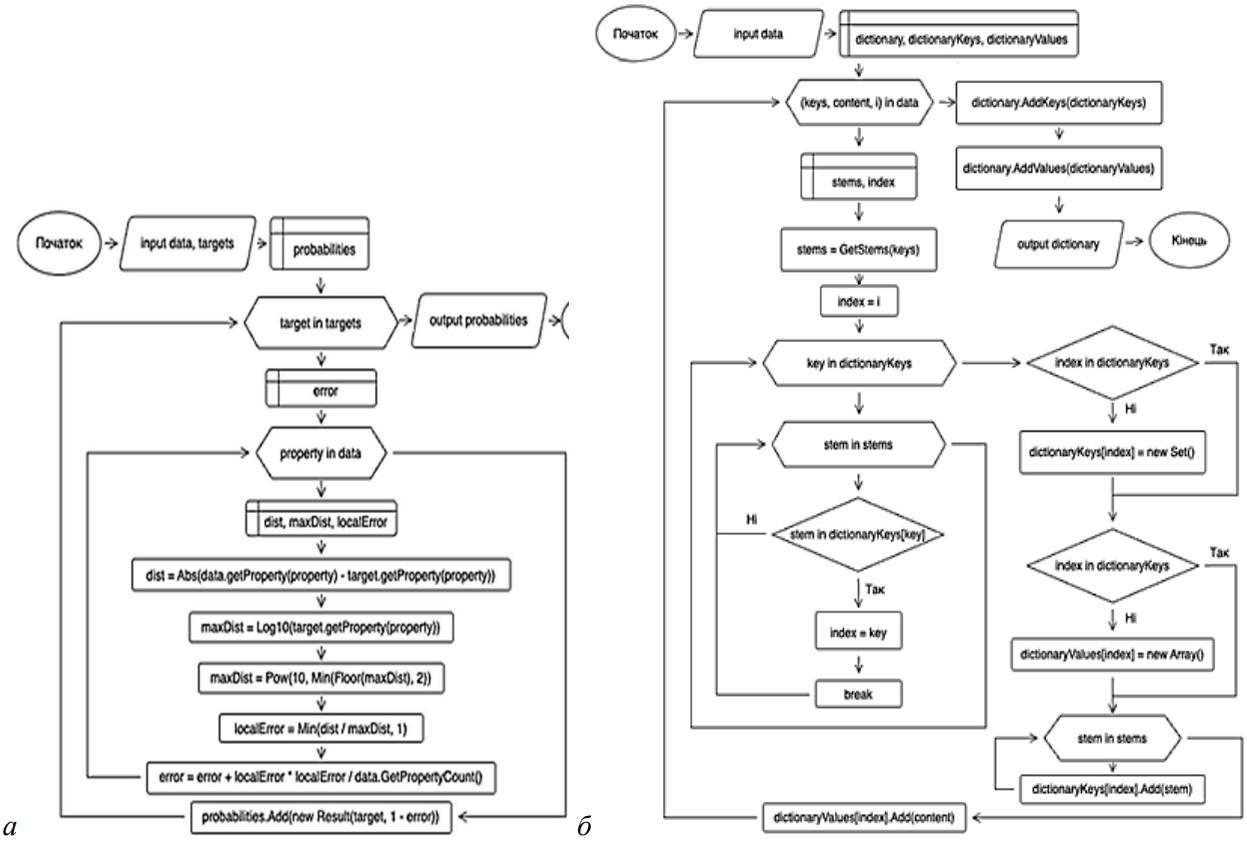


Рис. 16. Алгоритм: а – класифікації об’єктів для радіотехнічної розвідки;
б – інтеграції в інформаційні матеріали з визначеннями термінів

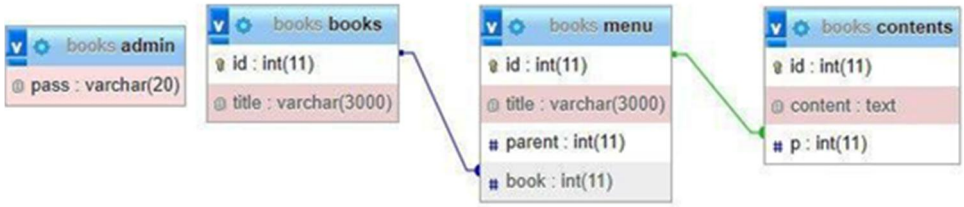


Рис. 17. Структура бази-даних

Розроблено верстку графічного інтерфейсу (рис. 18, *а*).

```
<body>
  <div class="sidebar">
    <ul>
      <li>
        <div>
          <a href="#"><b>Сторінка книги</b></a>
        </div>
      </li>
    </ul>
  </div>

  <main>
    Конент сторінки
  </main>

  <script type="text/javascript">
    function setCookie(cname, cvalue, exdays) {
      var d = new Date();
      d.setTime(d.getTime() + (exdays*24*60*60*1000));
      var expires = "expires="+ d.toUTCString();
      document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
    }

    function getCookie(cname) {
      var name = cname + "=";
      var decodedCookie = decodeURIComponent(document.cookie);
      var ca = decodedCookie.split(';');
      for(var i = 0; i <ca.length; i++) {
```

а

```
<body>
  <div class="sidebar">
    <div class="controls">
      <a href="add" style="background: #13ce66;">Додати</a>
      <a href="delete" style="background: #f00;">Видалити</a>
      <a href="rename" style="background: #aaa;">Змінити</a>
    </div>
    <ul>...</ul>
  </div>

  <main>
    <form method="POST" action="#">
      <textarea name="editor" id="editor" rows="10" cols="80"></textarea>
      <div class="submit">
        <button type="submit">Зберегти</button>
      </div>
    </form>
  </main>
```

б

```
$admin = false;

session_start();

$conn = new mysqli("localhost", "**********", "**********", "books");
$conn->set_charset("utf8");
if ($conn->connect_error) {
  die("Connection failed: " . $conn->connect_error);
}

if (isset($_GET["exit"])) {
  session_destroy();
  header("location: index.php");
} else if (isset($_GET["admin"]) && !isset($_SESSION["admin"])) {
  if (strlen($_GET["admin"]) > 0) {
```

в

Рис. 18. Код верстки сторінки: а – користувача; б – адміністратора; в – під'єднання до БД

Для створення зручного користувацького інтерфейсу для адміністраторів, на основі сторінки користувача побудовано додаткові компоненти (рис. 19, *б*). За допомогою PHP виконано з'єднання програми з базою даних (рис. 18, *с*) та опрацьовано генерування відповідної частини графічного інтерфейсу (рис. 19, *а*). На цьому етапі XAMPP може запускати програми, написані на PHP, та повертати відповідь, використовуючи HTTP запити. Також для зручного налаштування роботи програми створено файл конфігурації (рис. 19, *б*). Після організації навколишнього середовища створено графічний інтерфейс програми (рис. 19, *с*). Він виконаний за вебтехнологіями, оскільки ElectronJS дає можливість запускати програми на ПК.

```
<body>
  <div class="sidebar">
    <?php
      if (isset($_SESSION["admin"]) && $_SESSION["admin"] == 1) {
        ?>
    <div class="controls">
      <a href="add" style="background: #13ce66;">Додати</a>
      <a href="delete" style="background: #f00;">Видалити</a>
      <a href="rename" style="background: #aaa;">Змінити</a>
    </div>
        <?php
      }
    ?>
    <ul <?php if(isset($_SESSION["admin"]) && $_SESSION["admin"] == 1) {echo
      "id=\"bookList\"";} ?>>
  <?php
    for ($i=0; $i<count($mainData[0]); $i++) {
      ?>
      <li data-info="<?php echo ($i+1); ?>" <?php if ($page ==
          $mainData[0][$i]["id"]) {echo "class=\"active\"";} ?>>
        <div>
          <?php if (isset($mainData[$mainData[0][$i]["id"]])) { ?>
            <button class="sh"></button>
            <div class="sh-l"></div>
          <?php } ?>
```

а

settings.json

```
{
  "radar": {
    "size": 1000,
    "cells": 24
  },
  "knn": 5,
  "data": {
    "path": "/includes/data/data.txt",
    "full": false
  },
  "updateDelay": 500
}
```

б

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Radar</title>
  <link rel="stylesheet" href="includes/css/common.css">
</head>
<body class="loading">
  <div id="container">
    <div></div>
  </div>

  <div class="legend">
    <div class="class">*****</div>
    <div class="class red">*******</div>
  </div>

  <script src="renderer.js"></script>
</body>
</html>
```

в

Рис. 19. Фрагмент коду програми (а); файл конфігурації (б); код графічного інтерфейсу (в)

Наступним кроком є розроблення бізнес-логіки програми. Використовуючи фреймворк, організовано загальну роботу програми (рис. 20, *а*). Далі розроблено код, який формує карту для відображення класифікованих повітряних об'єктів. Для організації роботи в реальному часі потрібно з'єднати радар з програмою [26]. Оскільки багато радарів не мають можливості працювати через постійний відкритий канал використано підхід посередника. Програма повинна тільки зчитувати дані з визначеного файлу з деяким інтервалом часу, а завдання радара чи програми-адаптера радара – редагувати цей файл (рис. 20, *б*). Файл повинен містити всі дані, які надає радар. Тож програма використала один інтерфейс роботи з різними видами радарів.

```
const createWindow = () => {
  mainWindow = new BrowserWindow({
    width: 1000,
    height: 600,
    minHeight: 400,
    minWidth: 400,
    icon: `${__dirname}/includes/img/icon.ico`,
    webPreferences: {
      preload: `${__dirname}/preload.js`,
      nodeIntegration: true,
      contextIsolation: false,
      enableRemoteModule: true,
    },
  });

  mainWindow.loadURL(`file://${__dirname}/index.html`);
  mainWindow.setMenuBarVisibility(false);

  mainWindow.on('closed', () => {
    mainWindow = null;
  });
};

app.on('ready', createWindow);

app.on('window-all-closed', () => {
  if (process.platform !== 'darwin') {
    app.quit();
  }
});
```

а

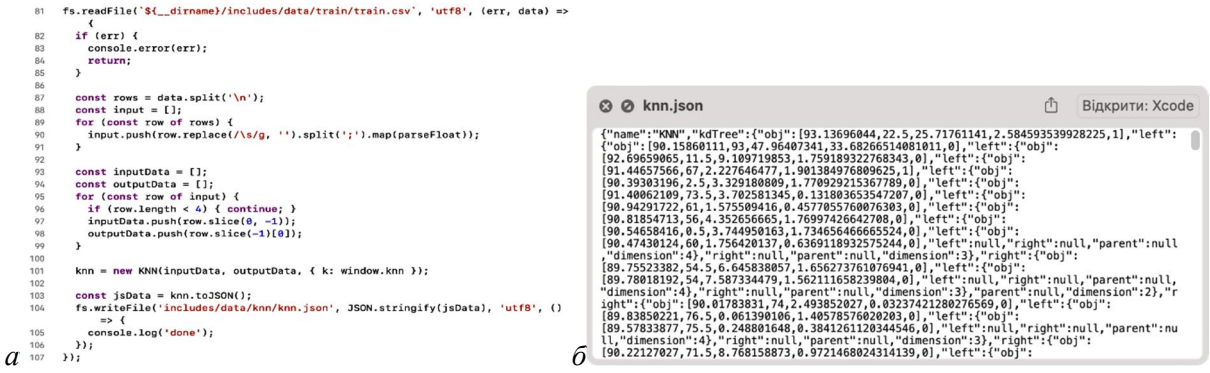
data.txt

```
94.34227942, 66, 5.401525452, 0.3246340465268527, 483, 122
95.16805951, 168, 17.54035745, 6.037746408235948, -152, -44
96.27306041, 128, 1.697072879, 2.716848951194665, 121, 395
91.69138542, 104, 12.66592337, 1.109905599599542, -400, 275
```

б

Рис. 20. Фрагмент коду програми (а); файл для зв'язку радару та програми (б)

Для класифікації об'єктів потрібно мати робочий класифікатор, а для його роботи – спочатку його сформувати. Відповідно написано код для створення класифікатора (рис. 21, *а*), щоб він сформував структуру даних (рис. 21, *б*), яку в подальшому буде використовувати програма для класифікації повітряних об'єктів. Останнім кроком є формування виконавчого файлу за допомогою операцій, що надає CLI фреймворку ElectronJS для запуску програми на Windows. Перед розробленням проєкту організовано робочий простір під час розроблення програми для класифікації об'єктів для радіотехнічної розвідки. Був розгорнутий локальний сервер XAMPP для використання ІС у браузері через локальну мережу. Далі розроблено верстку графічного інтерфейсу (рис. 22, *а*). Для роботи програми написано код мовою JavaScript (рис. 22, *б*), для зручного та швидкого розширення програми використано атрибути, які надають інформацію про клас (рис. 22, *в*). Потрібно тільки додати нові об'єкти з атрибутами для отримання ймовірності належності вхідного об'єкта до нових класів.



```
fs.readFile(`${__dirname}/includes/data/train/train.csv`, 'utf8', (err, data) =>
  {
  if (err) {
    console.error(err);
    return;
  }

  const rows = data.split('\n');
  const input = [];
  for (const row of rows) {
    input.push(row.replace(/\s/g, '').split(';').map(parseFloat));
  }

  const inputData = [];
  const outputData = [];
  for (const row of input) {
    if (row.length < 4) { continue; }
    inputData.push(row.slice(0, -1));
    outputData.push(row.slice(-1)[0]);
  }

  knn = new KNN(inputData, outputData, { k: window.knn });

  const jsData = knn.toJSON();
  fs.writeFile('includes/data/knn/knn.json', JSON.stringify(jsData), 'utf8', ()
      => {
    console.log('done');
  });
});
```

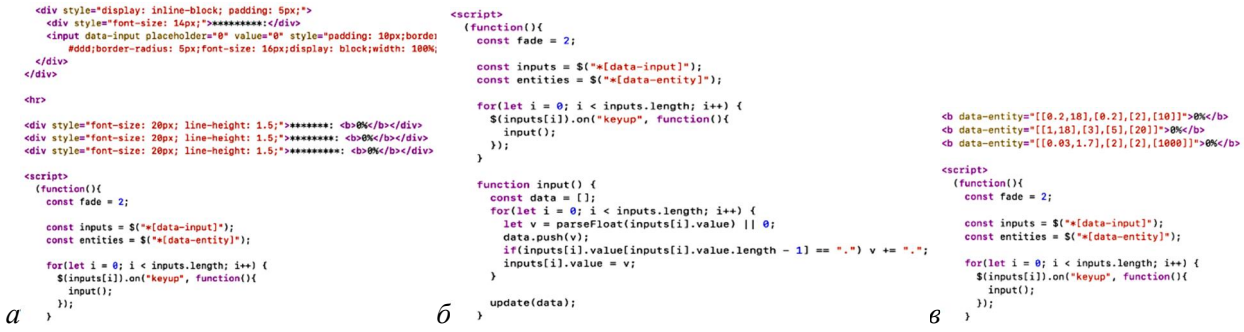
а

knn.json — Відкрити: Xcode

```
{"name":"KNN","kdTree":{"obj":[93.13696044,22.5,25.71761141,2.584593539928225,1],"left":
{"obj":[90.15860111,93,47.96407341,33.68266514081011,0],"left":{"obj":
[92.69659065,11.5,9.109719853,1.759189322768343,0],"left":{"obj":
[91.44657566,67,2.227646477,1.901384976809625,1],"left":{"obj":
[90.39303196,2.5,3.329180809,1.770929215367789,0],"left":{"obj":
[91.40062109,73.5,3.702581345,0.1318036535472078,0],"left":{"obj":
[90.94291722,61,1.575509416,0.4577055760076303,0],"left":{"obj":
[90.81854713,56,4.352656665,1.76997426642708,0],"left":{"obj":
[90.54658416,0.5,3.744950163,1.734656466665524,0],"left":{"obj":
[90.47430124,60,1.756420137,0.6369118932575244,0],"left":null,"right":null,"parent":null
,"dimension":4},"right":null,"parent":null,"dimension":3},"right":{"obj":
[89.75523382,54.5,6.645838057,1.656273761076941,0],"left":{"obj":
[89.78018192,54,7.587334479,1.562111658239804,0],"left":null,"right":null,"parent":null,
"dimension":4},"right":null,"parent":null,"dimension":3},"parent":null,"dimension":2},"r
ight":{"obj":[90.01783831,74,2.493852027,0.03237421280276569,0],"left":{"obj":
[89.83850221,76.5,0.061390106,1.40578576020203,0],"left":{"obj":
[89.57833877,75.5,0.248801648,0.3841261120344546,0],"left":null,"right":null,"parent":nu
ll,"dimension":4},"right":null,"parent":null,"dimension":3},"right":{"obj":
[90.22127027,71.5,8.768158873,0.9721468024314139,0],"left":{"obj":
```

б

Рис. 21. Фрагмент коду для створення моделі класифікатора (а); файл моделі класифікатора (б)



```
<div style="display: inline-block; padding: 5px;">
  <div style="font-size: 14px;">*********:</div>
  <input data-input placeholder="0" value="0" style="padding: 10px;borde
      #ddd;border-radius: 5px;font-size: 16px;display: block;width: 100%;
</div>
</div>

<hr>

<div style="font-size: 20px; line-height: 1.5;">*******: <b>0%</b></div>
<div style="font-size: 20px; line-height: 1.5;">*******: <b>0%</b></div>
<div style="font-size: 20px; line-height: 1.5;">*********: <b>0%</b></div>

<script>
  (function(){
    const fade = 2;

    const inputs = $("*[data-input]");
    const entities = $("*[data-entity]");

    for(let i = 0; i < inputs.length; i++) {
      $(inputs[i]).on("keyup", function(){
        input();
      });
    }
```

а

```
<script>
  (function(){
    const fade = 2;

    const inputs = $("*[data-input]");
    const entities = $("*[data-entity]");

    for(let i = 0; i < inputs.length; i++) {
      $(inputs[i]).on("keyup", function(){
        input();
      });
    }

    function input() {
      const data = [];
      for(let i = 0; i < inputs.length; i++) {
        let v = parseFloat(inputs[i].value) || 0;
        data.push(v);
        if(inputs[i].value[inputs[i].value.length - 1] == ".") v += ".";
        inputs[i].value = v;
      }

      update(data);
    }
```

б

```
<b data-entity="[[0.2,18],[0.2],[2],[10]]">0%</b>
<b data-entity="[[1,18],[3],[5],[20]]">0%</b>
<b data-entity="[[0.03,1.7],[2],[2],[1000]]">0%</b>

<script>
  (function(){
    const fade = 2;

    const inputs = $("*[data-input]");
    const entities = $("*[data-entity]");

    for(let i = 0; i < inputs.length; i++) {
      $(inputs[i]).on("keyup", function(){
        input();
      });
    }
```

в

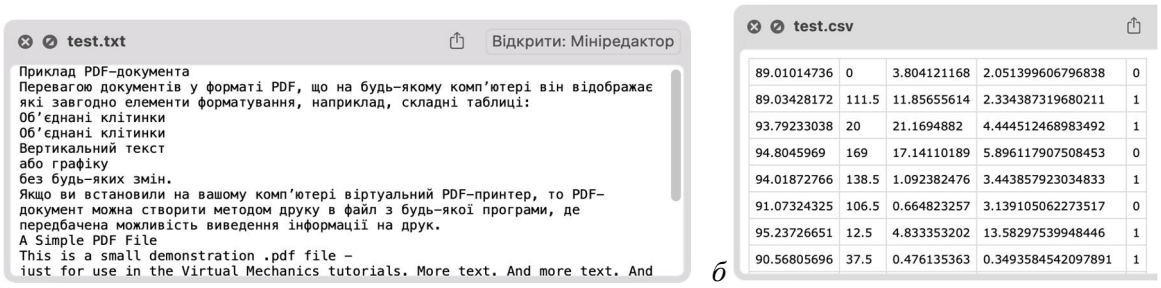
Рис. 22. Верстка інтерфейсу та фрагмент коду (а); програми (б); об'єктів класифікації (в)

З використанням інструментів платформи Figma створено дизайн прототипу ІС (рис. 23, *а*). Підібрано зручний і красивий графічний інтерфейс ІС [27]. Після створення дизайну розроблено верстку графічного інтерфейсу з допомогою бібліотеки Tkinter (рис. 23, *б*). Наступним кроком була розроблення основного коду програми (рис. 23, *в*).

б в

Рис. 23. Фрагменти: а – дизайну графічного інтерфейсу; б – коду графічного інтерфейсу, в – коду програми

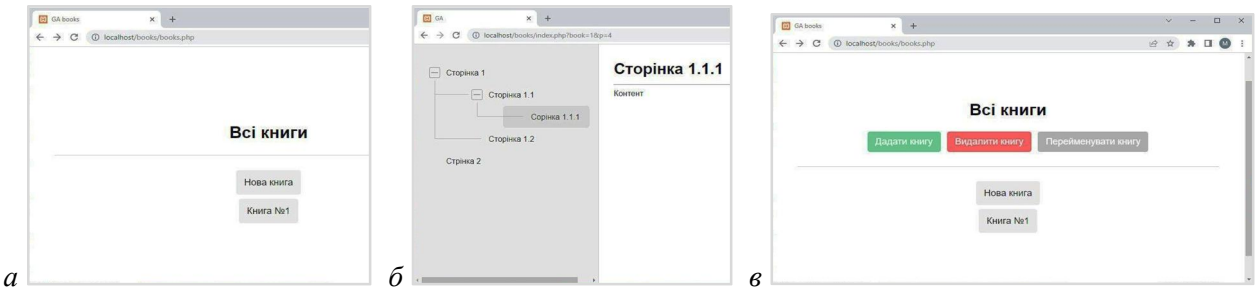
З використанням гнучкої методології Behavior Driven Development (BDD) розроблено програмне забезпечення, яке документується та розробляється відповідно до поведінки, яку очікує користувач під час взаємодії з нею [28]. BDD пропонує можливість розширити пул вхідних даних і відгуків, щоб залучити зацікавлені сторони бізнесу та кінцевих користувачів, які можуть мати недостатні знання про розробку ПЗ. Завдяки цьому розширеному циклу зворотного зв'язку команди розробників можуть легше використовувати BDD у середовищах постійної інтеграції та безперервної доставки. Щодо математичної частини тестування алгоритмів, то використовувались тестові дані (Рис. 24) для перевірки роботи всіх алгоритмів, що передбачають роботу з даними.



а *б*

Рис. 24. Один із файлів тестових даних (а); тестові дані для візуалізації та класифікації (б)

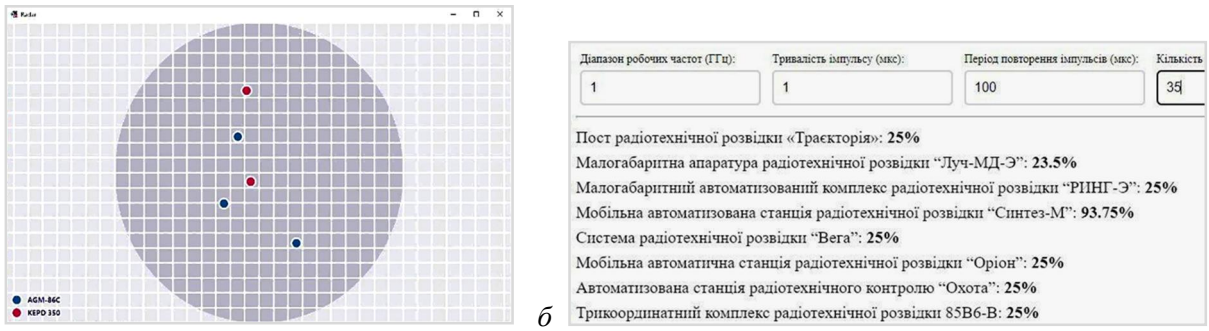
Програма для структуризації військової інформації має декілька сторінок, які надають увесь необхідний функціонал. Відкривши вебзастосунок через браузер, користувач потрапляє на сторінку вибору книги (рис. 25, *а*). Вибравши книгу користувач переходить до перегляду книги з візуалізацією її структури в лівій частині сторінки (рис. 25, *б*). Використовуючи меню ліворуч користувач може переходити зі сторінки на сторінку. Щоб керувати інформацією, що міститься в програмі, користувачу потрібно авторизуватися, щоб використовувати програму як адміністратор. Після цього на всіх сторінках, що має в доступі простий користувач, будуть додаткові елементи керування (рис. 25, *в*), за допомогою яких можна додавати, видаляти та редагувати інформацію.



а *б* *в*

Рис. 25. Вибір книги (а); перегляд книги (б); елементи керування адміністратора (в)

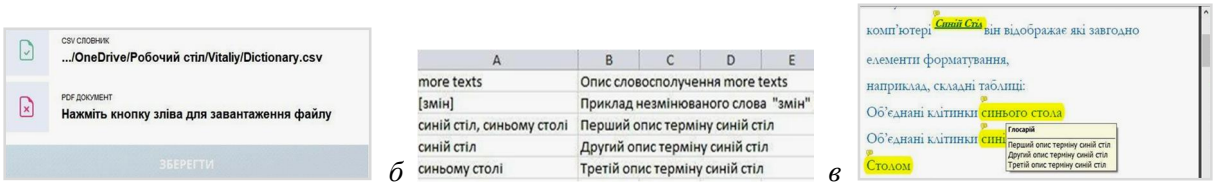
Модуль візуалізації та класифікації повітряних об’єктів в реальному часі має графічний інтерфейс, який складається з одного вікна (рис. 26, *а*). Модуль класифікації об’єктів для радіотехнічної розвідки також має одне вікно, через яке відбувається взаємодія з програмним продуктом. Відкривши програму в браузері, користувач потрапляє на сторінку програми (рис. 26, *а*), де він може вводити параметри для класифікатора. Класифікатор реагує на зміну кожного параметра, а це означає, що після кожної зміни характеристики, відповідно до актуальних даних, на основі внесених в форму характеристик буде проводитися процедура класифікації та виводитися ймовірність приналежності об’єкта до наявних класів.



а *б*

Рис. 26. Графічний інтерфейс модулів

Модуль інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур, як і дві попередні програми, має одне вікно для взаємодії користувача з ІС (рис. 27, *а*). На відміну від інших програм, дана програма передбачає спочатку формування словника (рис. 27, *б*) в будь-якій програмі, яка дає змогу експортувати дані з файлу типу CSV. Цей файл, як і інформаційний файл типу PDF, буде завантажуватися в програму для їхнього об’єднання. В результаті завантажується копія інформаційного файлу з додатковими даними, які відображаються при перегляді як нотатки (рис. 27, *в*). Користувач може переглядати документ, а коли з’являється потреба в додатковій інформації, то він може отримати її, наводячи курсор на відповідне слово.



а *б* *в*

Рис. 27. Інтерфейс модуля (а); словник – дані про слова та словосполучення (б); згенерований файл (в)

Висновки

Проведено дослідження процесів комплексного аналізу військової інформації на основі машинного навчання та опрацювання природньої мови для допомоги командирам тактичних ланок. Система надає користувачам наступні можливості: об'єднання словника та інформаційного матеріалу, додавання термінів та абревіатур в словник, класифікації об'єктів для радіотехнічної розвідки, візуалізація повітряних об'єктів, класифікації повітряних об'єктів, користування інформаційними матеріалами, організування інформаційних матеріалів. Розглянуто етапи додрукарської підготовки і досліджено процедури формування верстки. Результатом верстки є електронний макет видання, який для наступних етапів повинен бути переведений у PostScript-файл. Досліджено особливості мови PostScript та її можливості для виведення форматованого тексту, який містить широкий набір інструментів для роботи з текстом і графікою. Робота з текстом – найважливіша частина формування верстки видань. Розроблено підпрограми, які дають можливість створювати власні функції для додаткової роботи: розбиття тексту на рядки, вирівнювання тексту, переходу на новий рядок чи нову сторінку. Перевагою PostScript є простота кодів, їх зручно перетворювати у формат PDF для оперативного друку чи передачі через інтернет. Розроблена інтелектуальна система складається з чотирьох модулів, а саме з модуля інтеграції в інформаційні матеріали для визначення потенційно невідомих термінів та абревіатур, модуля класифікації об'єктів для радіотехнічної розвідки, модуля візуалізації та класифікації повітряних об'єктів в реальному масштабі часу та модуля структуризації військової інформації. Також у системі розроблено модуль правлення орфографічних та граматичних помилок у тексті на основі алгоритму перебору та словника, що містить 30000 слів української мови. В статті описана загальна структура розробленої інформаційної системи, її структури та алгоритмів функціонування кожного її модуля. Наведені також функціональні вимоги до інформаційної системи та окремо до кожного модуля. Здійснений опис експериментальної апробації розробленого програмного забезпечення.

Подяка

Ця стаття підготована завдяки грантовій підтримці Національного Фонду Досліджень України, реєстраційний номер проєкту 187/0012 від 1/08/2024 (2023.04/0012) "Розроблення інформаційної системи автоматичного виявлення джерел дезінформації та неавтентичної поведінки користувачів чатів" за конкурсом "Наука для зміцнення обороноздатності України".

СПИСОК ЛІТЕРАТУРИ

1. Khatsaiuk, O., et. al. (2021). Preparing future officers for performing assigned tasks through special physical training. ***Revista Romaneasca pentru Educatie Multidimensionala, 13***(2), 457–475. DOI: 10.18662/rrem/13.2/431.

2. Barbar, A. E. (2023). Challenges for Ethical Humanitarian Health Responses in Contemporary Conflict Settings. ***Dædalus, 152***(2), 53–62. DOI:10.1162/daed_a_01992.

3. Назаркевич, М. А. (2011). *Методи підвищення ефективності поліграфічного захисту засобами Ateb-функцій:* монографія. Львів: Видавництво Національного університету "Львівська політехніка".

4. Назаркевич, М. А., et. al. (2010). Інструментальні засоби для розроблення компонентів лінгвістичного забезпечення. *Матеріали міжнародної конференції "Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту" ISDMCI*, Євпаторія, 376.

5. Назаркевич, М. (2009). Розроблення програмного пакета для шифрування електронних документів засобами Ateb-функцій. *Вісник Державного університету "Львівська політехніка", 638*, 55–61.

6. Pashchetnyk, O., et. al. (2021). The Ontological Decision Support System Composition and Structure Determination for Commanders of Land Forces Formations and Units in Ukrainian Armed Force. ***CEUR Workshop Proceedings, 2870,*** 1077–1086.

7. Pattern-Oriented Software. http://www.dre.vanderbilt.edu/~arvindk/public_html/ECE255/ECE255.pdf.
8. Bass, L., et. al. (2023). Software architecture in practice. Addison-Wesley Professional.
9. Evergreen. UML. (2023). https://evergreens.com.ua/ua/articles/uml-diagrams.html.
10. Jacobsen, I., et.al. (1992). Object Oriented Software Engineering.
11. Hierarchical database model. (2023). https://www.heavy.ai/technical-glossary/hierarchical-database.
12. Classical machine learning. (2023). https://quantumalgorithms.org/chap-machinelearning.html.
13. Logistic regression. (2023). https://www.ibm.com/topics/logistic-regression.
14. Naive Bayes classifier. (2023). https://www.ibm.com/topics/naive-bayes.
15. The k-nearest neighbors algorithm. (2023). https://www.ibm.com/topics/knn.
16. Geeksforgeeks. Decision Tree. (2023). https://www.geeksforgeeks.org/decision-tree/.
17. Support Vector Machine. (2023). https://scikit-learn.org/stable/modules/svm.html.
18. Gyansetu. NLP in Machine Learning. https://gyansetu.in/blogs/what-is-natural-language-processing/.
19. Hastie, T., et. al. (2009). The elements of statistical learning: data mining, inference, and prediction, 2, 1–758. New York: Springer. DOI: 10.1007/978-0-387-21606-5
20. Stuart, Russell м (2002). Artificial Intelligence. https://www.sti-innsbruck.at/sites/default/files/Knowledge-Representation-Search-and-Rules/Russel-&-Norvig-Inference-and-Logic-Sections-7.pdf.
21. Visual Studio Code. (2023). https://code.visualstudio.com/docs/editor/whyvscode.
22. Simplilearn. Node.js. (2023). https://www.simplilearn.com/node-js-vs-java-article/.
23. Coursera. What Does a Front-End Developer Do? https://www.coursera.org/articles/front-end-developer/.
24. Schaefer, K. E., et. al. (2021). Human-autonomy teaming for the tactical edge: the importance of humans in artificial intelligence research and development. *Systems Engineering and Artificial Intelligence*, 115–148. Cham: Springer International Publishing. DOI: 10.1007/978-3-030-77283-3_7.
25. Wang, W., et. al. (2020). Investigation on works and military applications of artificial intelligence. *IEEE Access, 8,* 131614–131625. DOI: 10.1109/ACCESS.2020.3009840.
26. SearchUnifiedCommunications. https://www.techtarget.com/searchunifiedcommunications/definition/real-time-application-RTA/.
27. Smashing magazine. (2023). https://www.smashingmagazine.com/2010/10/what-is-user-experience-design-overview-tools-and-resources/
28. Solis, C., et. al. (2011). A study of the characteristics of behaviour driven development. *EUROMICRO conference on software engineering and advanced applications*, 383–387. DOI: 10.1109/SEAA.2011.76.
29. Danylyk, V., et. al. (2020). Detecting items with the biggest weight based on neural network and machine learning methods. *Conference on Data Stream Mining and Processing*, 383–396. DOI:10.1007/978-3-030-61656-4_26
30. Danylyk, V., et. al. (2024). Information Technology for the Operational Processing of Military Content for Commanders of Tactical Army Units, *International Journal of Computer Network and Information Security(IJCNIS), 16*(3), 115–143. DOI:10.5815/ijcnis.2024.03.09.
31. Danylyk, V., Vysotska, V. (2024). Information Technology for Detecting Fakes and Propaganda Based on Machine Learning and Sentiment Analysis. Qeios. https://doi.org/10.32388/IZFOXN
32. Pasichnyk, V., et. al. (2024). Expert assessment of educational content in IT specialists training process. Ceur Workshop Proceedings. https://ceur-ws.org/Vol-3723/paper8.pdf.
33. Hryhorovych, V. (2022). Analysis of Scientific Texts by Semantic Inverse-Additive Metrics for Ontology Concepts, *COLINS,* 801–816. https://ceur-ws.org/Vol-3171/paper60.pdf.
34. Oleksiv, N., et. al. (2022, May). Recommendation System for Monitoring the Energy Value of Consumer Food Products Based on Machine Learning, *COLINS*, 1321–1350. https://ceur-ws.org/Vol-3171/paper97.pdf.
35. Pashchetnyk, O., et. al. (2021). The Ontological Decision Support System Composition and Structure Determination for Commanders of Land Forces Formations and Units in Ukrainian Armed Force, *COLINS,* 1077–1086. https://ceur-ws.org/Vol-2870/paper81.pdf.
36. Albota, S. (2023). Creating a Model of War and Pandemic Apprehension: Textual Semantic Analysis, *COLINS, 2,* 228–243. https://ceur-ws.org/Vol-3396/paper19.pdf.

37. Albota, S. (2023). Linguistic and Semantic Representation of the WAR concept by Phraseological Units, *SCIA,* 355–364. https://ceur-ws.org/Vol-3608/paper27.pdf.

38. Albota, S. (2022). War implications in the Reddit news feed: semantic analysis. *International Conference on Computer Sciences and Information Technologies*, 99–102. DOI: 10.1109/CSIT56902.2022.10000515.

39. Lytvyn, V., et. al. (2021). Developing Methods for Building Intelligent Systems of Information Resources Processing Using an Ontological Approach. *Advances in Intelligent Systems and Computing, 1293*. Springer, Cham. https://doi.org/10.1007/978-3-030-63270-0_23.

REFERENCES

1. Khatsaiuk, O., et. al. (2021). Preparing future officers for performing assigned tasks through special physical training. *Revista Romaneasca pentru Educatie Multidimensionala, 13*(2), 457–475. DOI: 10.18662/rrem/13.2/431.

2. Barbar, A. E. (2023). Challenges for Ethical Humanitarian Health Responses in Contemporary Conflict Settings. *Dædalus, 152*(2), 53–62. DOI:10.1162/daed_a_01992.

3. Nazarkevych, M. A. (2011). *Methods of increasing the effectiveness of polygraphic protection by Ateb-functions*: monograph. Lviv: Publishing House of the National University "Lviv Polytechnic".

4. Nazarkevych, M. A., et. al. (2010). Instrumental tools for the development of linguistic support components. *Materials of the international conference "Intelligent decision-making systems and problems of computational intelligence" ISDMCI,* Yevpatoria, 376.

5. Nazarkevych, M. (2009). Development of a software package for encryption of electronic documents using Ateb functions, *Bulletin of the Lviv Polytechnic State University, 638,* 55–61.

6. Pashchetnyk, O., et. al. (2021). The Ontological Decision Support System Composition and Structure Determination for Commanders of Land Forces Formations and Units in Ukrainian Armed Force. *CEUR Workshop Proceedings, 2870,* 1077–1086.

7. Pattern-Oriented Software. http://www.dre.vanderbilt.edu/~arvindk/public_html/ECE255/ECE255.pdf.

8. Bass, L., et. al. (2023). Software architecture in practice. Addison-Wesley Professional.

9. Evergreen. UML. (2023). https://evergreens.com.ua/ua/articles/uml-diagrams.html.

10. Jacobsen, I., et.al. (1992). Object Oriented Software Engineering.

11. Hierarchical database model. (2023). https://www.heavy.ai/technical-glossary/hierarchical-database.

12. Classical machine learning. (2023). https://quantumalgorithms.org/chap-machinelearning.html.

13. Logistic regression. (2023). https://www.ibm.com/topics/logistic-regression.

14. Naive Bayes classifier. (2023). https://www.ibm.com/topics/naive-bayes.

15. The k-nearest neighbors algorithm. (2023). https://www.ibm.com/topics/knn.

16. Geeksforgeeks. Decision Tree. (2023). https://www.geeksforgeeks.org/decision-tree/.

17. Support Vector Machine. (2023). https://scikit-learn.org/stable/modules/svm.html.

18. Gyansetu. NLP in Machine Learning. https://gyansetu.in/blogs/what-is-natural-language-processing/.

19. Hastie, T., et. al. (2009). *The elements of statistical learning: data mining, inference, and prediction, 2,* 1–758. New York: Springer. DOI: 10.1007/978-0-387-21606-5

20. Stuart, Russell м (2002). Artificial Intelligence. https://www.sti-innsbruck.at/sites/default/files/Knowledge-Representation-Search-and-Rules/Russel-&-Norvig-Inference-and-Logic-Sections-7.pdf.

21. Visual Studio Code. (2023). https://code.visualstudio.com/docs/editor/whyvscode.

22. Simplilearn. Node.js. (2023). https://www.simplilearn.com/node-js-vs-java-article/.

23. Coursera. What Does a Front-End Developer Do? https://www.coursera.org/articles/front-end-developer/.

24. Schaefer, K. E., et. al. (2021). Human-autonomy teaming for the tactical edge: The importance of humans in artificial intelligence research and development. *Systems Engineering and Artificial Intelligence,* 115–148. Cham: Springer International Publishing. DOI: 10.1007/978-3-030-77283-3_7.

25. Wang, W., et. al. (2020). Investigation on works and military applications of artificial intelligence. *IEEE Access, 8,* 131614–131625. DOI: 10.1109/ACCESS.2020.3009840.

26. SearchUnifiedCommunications. https://www.techtarget.com/searchunifiedcommunications/definition/real-time-application-RTA/.

27. Smashing magazine. (2023). https://www.smashingmagazine.com/2010/10/what-is-user-experience-design-overview-tools-and-resources/

28. Solis, C., et. al. (2011). A study of the characteristics of behaviour driven development. *EUROMICRO conference on software engineering and advanced applications*, 383–387. DOI: 10.1109/SEAA.2011.76.

29. Danylyk, V., et. al. (2020). Detecting items with the biggest weight based on neural network and machine learning methods. *Conference on Data Stream Mining and Processing*, 383–396. DOI:10.1007/978-3-030-61656-4_26

30. Danylyk, V., et. al. (2024). Information Technology for the Operational Processing of Military Content for Commanders of Tactical Army Units, *International Journal of Computer Network and Information Security(IJCNIS), 16*(3), 115–143. DOI:10.5815/ijcnis.2024.03.09.

31. Danylyk, V., Vysotska, V. (2024). Information Technology for Detecting Fakes and Propaganda Based on Machine Learning and Sentiment Analysis. Qeios. https://doi.org/10.32388/IZFOXN

32. Pasichnyk, V., et. al. (2024). Expert assessment of educational content in IT specialists training process. Ceur Workshop Proceedings. https://ceur-ws.org/Vol-3723/paper8.pdf.

33. Hryhorovych, V. (2022). Analysis of Scientific Texts by Semantic Inverse-Additive Metrics for Ontology Concepts, *COLINS*, 801–816. https://ceur-ws.org/Vol-3171/paper60.pdf.

34. Oleksiv, N., et. al. (2022, May). Recommendation System for Monitoring the Energy Value of Consumer Food Products Based on Machine Learning, *COLINS,* 1321–1350. https://ceur-ws.org/Vol-3171/paper97.pdf.

35. Pashchetnyk, O., et. al. (2021). The Ontological Decision Support System Composition and Structure Determination for Commanders of Land Forces Formations and Units in Ukrainian Armed Force, *COLINS*, 1077–1086. https://ceur-ws.org/Vol-2870/paper81.pdf.

36. Albota, S. (2023). Creating a Model of War and Pandemic Apprehension: Textual Semantic Analysis. *COLINS, 2*, 228–243. https://ceur-ws.org/Vol-3396/paper19.pdf.

37. Albota, S. (2023). Linguistic and Semantic Representation of the WAR concept by Phraseological Units, *SCIA*, 355–364. https://ceur-ws.org/Vol-3608/paper27.pdf.

38. Albota, S. (2022). War implications in the Reddit news feed: semantic analysis. *International Conference on Computer Sciences and Information Technologies*, 99–102. DOI: 10.1109/CSIT56902.2022.10000515.

39. Lytvyn, V., et. al. (2021). Developing Methods for Building Intelligent Systems of Information Resources Processing Using an Ontological Approach, *Advances in Intelligent Systems and Computing, 1293*. Springer, Cham. https://doi.org/10.1007/978-3-030-63270-0_23.

INTELLIGENT SYSTEM FOR COMPLEX MILITARY INFORMATION ANALYSIS BASED ON MACHINE LEARNING AND NLP TO ASSIST TACTICAL LINKS COMMANDERS

Vitaliy Danylyk[1], Vasyl Lytvyn[2], Victoria Vysotska[3]

[1, 2, 3] Lviv Polytechnic National University,
Information Systems and Networks Department, Lviv, Ukraine
[1] E-mail: Vitalii.M.Danylyk@lpnu.ua, ORCID: 0000-0001-5928-7235
[2] E-mail: Vasyl.V.Lytvyn@lpnu.ua, ORCID: 0000-0002-9676-0180
[3] E-mail: Victoria.A.Vysotska@lpnu.ua, ORCID: 0000-0001-6417-3689

Summary. The article describes the results of research into the processes of complex analysis of military information based on machine learning and natural language processing to help commanders of tactical units. The system should allow users to have the following capabilities: combining the

dictionary and information material, adding terms and abbreviations to the dictionary, classifying objects for radio technical intelligence, visualizing aerial objects, classifying aerial objects, using information materials, organizing information materials. The developed intelligent system consists of four modules, namely, a module for integrating definitions of potentially unknown terms and abbreviations into information materials, a module for classifying objects for radio technical intelligence, a module for visualization and classification of aerial objects in real time, and a module for structuring military information. Also, the system has developed a module for correcting spelling and grammatical errors in the text based on a sorting algorithm and a dictionary of about 30,000 words of the Ukrainian language. The article describes the general structure of the developed system and, accordingly, the structures and algorithms of the functioning of each developed module of the system. The functional requirements of the system and separately for each module are also given. A description of the experimental testing of the developed software was carried out.

Keywords: military data, tactical unit commander, information structuring, natural language processing, object classification, intelligent system, machine learning, error correction, naive Bayes classifier, support vector machine method, k nearest neighbours, logistic regression.