УДК 004.91 DOI: 10.30837/0135-1710.2024.182.005

М.В. ЄВЛАНОВ, І.О. ЮР'ЄВ, Б.Є. КРУК

РОЗРОБКА МЕТОДУ ВИРІШЕННЯ ЗАДАЧІ ФОРМУВАННЯ І КЛАСИФІКАЦІЇ ЗАПИТІВ НА ЗМІНИ ІТ-ПРОДУКТУ

Розглянуто основні існуючі інформаційні технології, моделі і методи формування запитів на зміни ІТ-продуктів під час виконання ІТ-проєктів. Встановлено, що єдиної сталої системи класифікації RFC, яка охоплювала б увесь життєвий цикл ІТ-проєкту, не існує. Розроблено теоретико-множинну модель запиту на зміну, який виникає під час виконання ІТ-проєкту розробки чи модифікації ІТ-продукту. На основі цієї моделі розроблено загальний метод вирішення задачі формування і класифікації запитів на зміну ІТ-продукту.

1. Вступ

Управління змінами – це комплексний, циклічний і структурований підхід до зміни окремих осіб, груп, і організації з поточного стану в майбутній стан із передбачуваними бізнес-вигодами [1], [2]. Цей підхід зараз описано у вигляді фреймворку – конвенцій, принципів і методів управління змінами в галузі управління програмами, портфелями проєктів та окремими проєктами. Цей фреймворк охоплює тему стратегічної гнучкості як відображення здатності організації відчувати зовнішні чи внутрішні загрози і можливості та реагувати на них. Стратегічна гнучкість визначається як здатність бізнесу завчасно оцінювати і реалізувати зміни бізнес-середовища та демонструвати стійкість до непередбачених змін [2].

Фреймворк управління змінами розглядає перехід з поточного стану в майбутній стан із передбачуваними бізнес-вигодами як процес «Управління змінами» із такими підпроцесами [2]:

а) формулювання змін – підпроцес, який здійснюється шляхом виявлення та уточнення потреби у змінах, оцінки готовності до змін та окреслення масштабу змін;

б) планування змін – підпроцес, який здійснюється шляхом визначення підходу до змін і планування залучення зацікавлених сторін як переходів та інтеграції;

в) впровадження змін – підпроцес, який здійснюється шляхом підготовки організації до змін, мобілізації зацікавлених сторін та досягнення результатів проєкту;

г) управління переходом до змін – підпроцес, який здійснюється шляхом переведення результатів у бізнес-операції, вимірювання рівня впровадження та результатів і переваг змін, а також коригування плану для усунення розбіжностей;

д) підтримка змін на постійній основі – підпроцес, який здійснюється шляхом спілкування, консультації та представництва зацікавлених сторін, проведення смислотворчої діяльності та вимірювання реалізації вигод.

Застосування розглянутого фреймворку управління змінами під час управління таким різновидом проєктів, як ІТ-проєкти, вимагає приділення особливої уваги підпроцесу «Формулювання змін» і, зокрема, роботі «Визначте/уточніть потребу в змінах». Ця увага обумовлена тим, що саме ця робота є першою роботою, в результаті виконання якої ІТ-проєкт та система управління цим проєктом отримує інформацію про необхідність зміни та перший опис цієї зміни.

Але, на жаль, існуючий фреймворк управління змінами, розглянутий у [2], та стандарти і зведення знань з управління проєктами та, зокрема, ІТ-проєктами [1], [3] не дають конкретних рекомендацій з виконання підпроцесу «Формулювання змін» і роботи «Визначте/уточніть потребу в змінах». Так у [2] вказано два таких варіанти виконання

цього підпроцесу та роботи:

а) якщо проєкт є частиною програми, визначення та роз’яснення потреби в змінах повинно бути добре розробленою і сформульованою частиною статуту проєкту;

б) якщо проєкт не є частиною програми, необхідно під час управління портфелем окремих проєктів підприємства виконати відповідні дії, щоб переконатися, що уточнення потреб в змінах виконано до завершення статуту проєкту.

Таким чином, незалежно від особливостей ІТ-проєкту, потреби в змінах та перші описи можливих змін згідно з існуючим фреймворком управління змінами повинні бути сформульовані до завершення робіт зі складання та ухвалення статуту ІТ-проєкту. При цьому роботу «Визначте/уточніть потребу в змінах» пропонується розглядати як різновид робіт з формування вимог, які висуваються до ІТ-продукту та ІТ-проєкту перед початком планування та виконання цього проєкту [2]. Таке позиціонування підпроцесу «Формулювання змін» і роботи «Визначте/уточніть потребу в змінах» значно обмежує можливість внесення змін до ІТ-проєкту під час його планування та виконання і, відповідно, зменшує гнучкість проєкту та можливості його адаптації до обставин, що постійно змінюються.

Слід зазначити, що існуючий фреймворк управління змінами передбачає можливість виникнення змін під час виконання проєкту. Зокрема, згідно з цим фреймворком, керівнику проєкту може знадобитися виконати завдання, які виходять за встановлені межі цього проєкту. В ідеалі ці додаткові завдання мають бути частиною узгодженого обсягу проєкту, щоб забезпечити успішне виконання змін. Інакше керівник проєкту повинен попередити спонсора проєкту та вимагати зміни обсягу проєкту [2]. Таким чином, виходить замкнене коло: керівник проєкту повинен прийняти рішення про виконання роботи «Визначте/уточніть потребу в змінах», базуючись на оцінках обсягу ІТ-проєкту до та після внесення зміни, яка пропонується. Але ці оцінки неможливо розрахувати, не знаючи зміст зміни, яка пропонується, тобто не виконавши роботу «Визначте/уточніть потребу в змінах».

Тому виникає необхідність у проведенні науково-прикладних досліджень, які дозволили б формалізувати виконання підпроцесу «Формулювання змін» і роботи «Визначте/уточніть потребу в змінах» під час виконання ІТ-проєкту незалежно від конкретних особливостей цих змін та ІТ-проєктів.

2. Аналіз літературних даних і постановка проблеми дослідження

Особливості виконання процесів, підпроцесів та окремих робіт існуючого фреймворку управління змінами визначаються в загальному випадку прийнятим у [2] життєвим циклом (ЖЦ) управління змінами. Його схема наведена на рис. 1 [2].

За наведеною на рис. 1 схемою, у [2] запропоновано розглядати кожен з розглянутих підпроцесів управління змінами як множину окремих робіт, кожна з яких виконується у встановленому фреймворком порядку (але не обов'язково суворо послідовно одна за одною). Весь процес управління змінами та його підпроцеси у [2] запропоновано описувати з використанням ітераційної моделі ЖЦ з врахуванням можливості постійного виникнення адаптивних змін у відповідь на зміни обставин.

Таке представлення ЖЦ управління змінами дозволяє мінімізувати зв’язки між окремими підпроцесами та роботами існуючого фреймворку управління змінами і розглядати їх як окремі самостійні об’єкти наукових досліджень. На технологічному рівні це означає можливість створення та застосування для автоматизації відповідних підпроцесів та робіт окремих інформаційних технологій (ІТ), які взаємодіють між собою за сервіс-орієнтованою парадигмою.

Для того, щоб можна було оцінити зміну обсягу IT-проєкту після внесення зміни, яка пропонується, треба визначити, якої саме частини проєкту стосується ця зміна. Тому виникає необхідність дослідити і проаналізувати існуючі класифікації запитів на зміни (request for change, RFC), які можуть локалізувати місце впливу RFC на обсяг IT-проєкту. Такі класифікації є спробою формалізувати різновиди RFC, базуючись, зазвичай, на таких результатах:

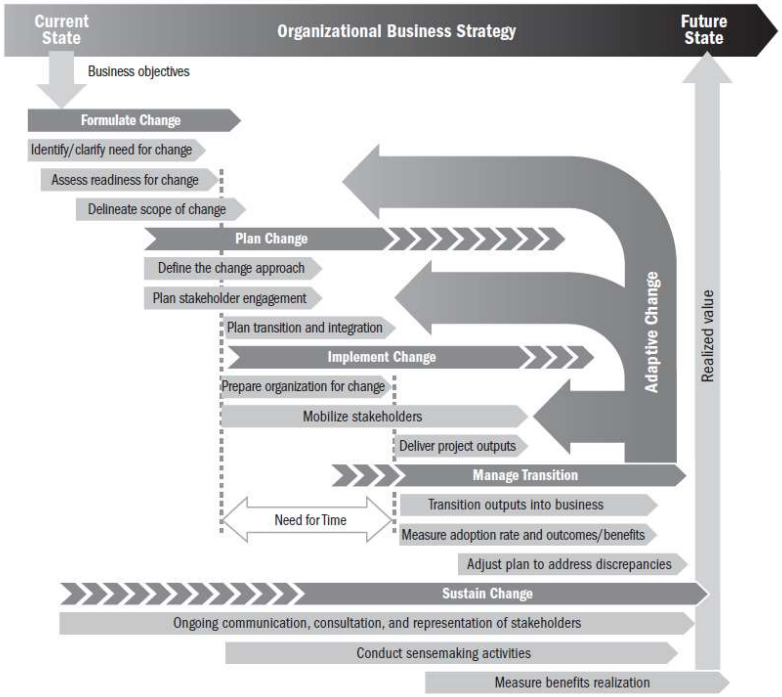

Рис. 1. Схема життєвого циклу управління змінами [2]

а) досвід управління змінами в окремій українській IT-компанії;

б) досвід управління змінами в закордонних IT-проєктах, які відносяться до одного й того ж різновиду.

Типовим прикладом класифікації, яка базується на результатах групи а), є класифікація, запропонована у [4]. За цією класифікацією, уся множина RFC поділяється на такі класи:

– зміни системної логіки/поведінки;

– зміни скоупу (під цим терміном тут слід розуміти обсяг, межі або масштаб IT-проєкту чи IT-продукту);

– зміни в назвах/формулюваннях;

– зміни в інтерфейсі користувача;

– технічні зміни (інтеграції, мови програмування тощо).

Як можливий окремий клас RFC у [4] пропонується розглядати запит на видалення елементів зі скоупу.

Але запропонована у [4] класифікація характеризується тим, що вона не використовується навіть самим автором під час визначення особливостей виконання робіт процесу «Управління змінами». Так у [4] представлена візуальна модель кастомного процесу управління змінами, адаптованого під команди, з якими працював автор. Але головною проблемою прийняття рішення щодо подальшого управління зміною IT-проєкту за цією моделлю визнано проблему відділення потреб зацікавлених сторін проєкту у змінах від потреб цих же сторін у ліквідації знайдених програмних помилок. Виходячи з цього, пропонуються описи дій з управління RFC, які незначно відрізняються від загальних рекомендацій загального фреймворку управління змінами за ступенем деталізації. Тому можна зробити висновок, що запропонований варіант класифікації є декларативним та суб'єктивним, тому що він не впливає на виконання робіт процесу управління змінами та підпроцесу «Формулювання змін».

Результатом застосування групи б) результатів є класифікації RFC, які виникли та виникають внаслідок узагальнення досвіду закордонних IT-компаній. На відміну від результатів групи а), ці класифікації є результатом наукового узагальнення одного з таких варіантів досвіду:

а) досвід управління змінами на окремій стадії IT-проєктів;

б) досвід управління змінами в IT-проєктах, які належать до одного й того ж різновиду;

в) узагальнений досвід управління змінами різних IT-проєктів.

Як приклад класифікації RFC, яка використовується для управління змінами на окремій стадії IT-проєкту, пропонується розглянути запропоновану у [5] класифікацію технічних (або інженерних) змін. Така зміна у [6] визначається як зміна деталей, креслень або програмного забезпечення, які вже були випущені в процесі розробки продукту. Ця зміна може бути будь-якого розміру або типу; вона може включати будь-яку кількість людей і тривати протягом будь-якого часу [5], [6]. Запит на технічну зміну визначається як RFC, створений для інформування про вихід проєкту чи продукту із цільових значень [7]. За іншим визначенням, запит на технічну зміну – це запит на технічну зміну стану моделі продукту, яка вже була випущена для подальшої розробки або виробництва [5].

Для вирішення задачі класифікації авторами у [5] було розроблено атрибутивну модель запиту на технічну зміну за результатами дослідження літературних джерел та великої кількості текстів запитів на технічні зміни. Ця модель наведена на рис. 2 [5].

	Attribute	Characteristic value					
Object of change	Concerned object	Document	Component	Process	Module	Equipment	Product
	Concerned object specification	Physical Specification	Functionality	Software	Procedures	Information	
	Extend of change*	Document	Component	Process	Module	Equipment	Product
	Volume of change	Single position		Group		All	
Properties	Phase of life-cycle	Development	Prototyping / Series start-up	Series production	Use phase		
	Architecture	Modular		Partly integrated		Integrated	
	Complexity	Low		Medium		High	
Influencing factors	Root of change	Customer requirement	Law / Certification	Safety	Cost	Technology	Productivity
	Reason of change	Current situation faulty	Optimization of the current situation	Target requirements changed	Incorrect target requirements		
	Urgency	Change to next release		Important, short-term change		Immediate change	
	Concerned Stakeholder	R&D	R&D and other departments	Involvement of the management	Involvement of external stakeholders		
Effect	Effect on cost	Decreasing	Slightly decreasing	Slightly increasing	Increasing		
	Effect on time schedules	Low		Medium		High	
	Effect on the product	None	Necessary correction	Fulfillment of individual customer requirements	Improvement of functionality	Quality improvement	

Descriptive attributes

Evaluating attributes (expert knowledge required)

*The CVs of concerned object and extend of change are the same. The diffrence is, that concerned object identifies the objects from where the change originates, while extend of the change describes objects that are impacted by the change of the object. *Example: If the dimension of a component (=object) is changed, most likely the drawing document will be changed too (=extend of change)*

Рис. 2. Атрибутивна модель запиту на технічну зміну [5]

Для використання моделі, наведеної на рис. 2, у [5] було розроблено рішення задачі автоматичної класифікації запитів на технічні зміни. Загальне графічне представлення цієї задачі наведено на рис. 3 [5].

Для реалізації цієї задачі у [5] запропоновано формалізувати опис запиту на технічну зміну як кортеж таких текстових атрибутів:

а) «Причина зміни»;

б) «Компонент/агрегат»;

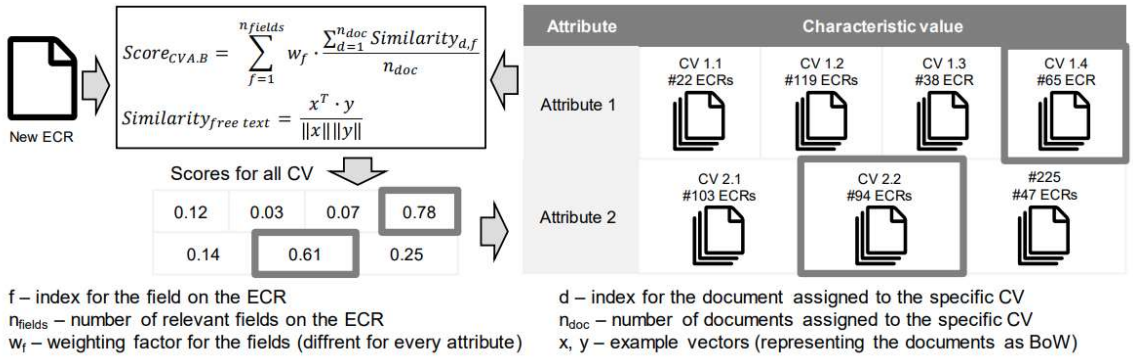

Рис. 3. Графічне представлення задачі автоматичної класифікації запитів на технічні зміни [5]

в) «Опис поточної ситуації»;

г) «Вигода/ціль зміни»;

д) «Запропонована реалізація».

Апробацію ІТ, яка реалізувала цю задачу класифікації, проводили на 40 запитах на технічні зміни. За результатами апробації було встановлено, що ця технологія досягла загальної точності вирішення задачі класифікації запитів на технічні зміни 79 %, а максимальна точність дорівнювала 90 % [5].

Як інший приклад класифікації RFC, яка використовується для управління змінами на окремій стадії ІТ-проєкту, розглянемо запропоновану у [8] онтологічну модель. Ця модель описує класи RFC, які виникають під час змін у системних вимогах, що висунуті до створюваного ІТ-продукту. Модель представлено у вигляді діаграми класів, яка відображає ієрархію класів RFC системних вимог (див. рис. 4). Специфікації значень важливих елементів цієї онтологічної моделі наведено у табл. 1.

Як приклад класифікації RFC, яка використовується для управління змінами в ІТ-проєктах, які належать до одного й того ж різновиду, розглянемо класифікацію, яку у [9] запропонували використовувати під час управління змінами в ІТ-проєктах розробки мобільних застосунків. Схема ЖЦ релізу розробки мобільного застосунку із впровадженою технологією управління змінами наведено на рис. 5 [9]. На цьому рисунку пунктирною лінією виділено основу технології управління змінами та її інтерфейси, які передають встановлені зміни до інструментальних засобів, що використовуються на різних стадіях відповідного ІТ-проєкту. Схему підходу, який знаходиться в основі цієї технології, наведено на рис. 6 [9].

Основними обмеженнями системи управління змінами на основі запропонованої на рис. 5 схеми ЖЦ є [9]:

– низька точність класифікації та кластеризації через низьку якість відгуків користувачів;

– труднощі в роботі з різними словниками, які використовуються для опису досвіду використання користувачами застосунків;

– існування розривів в словниках ІТ-проєкту між термінами з відгуків користувачів і позначеннями артефактів програмного забезпечення.

Тому в наведеній на рис. 6 технології для подолання цих обмежень запропоновано поділити усі можливі відгуки користувачів на такі чотири категорії [9]:

а) «Надання інформації» (речення, які інформують або оновлюють інформацію для користувачів або розробників щодо аспекту, пов’язаного з програмою);

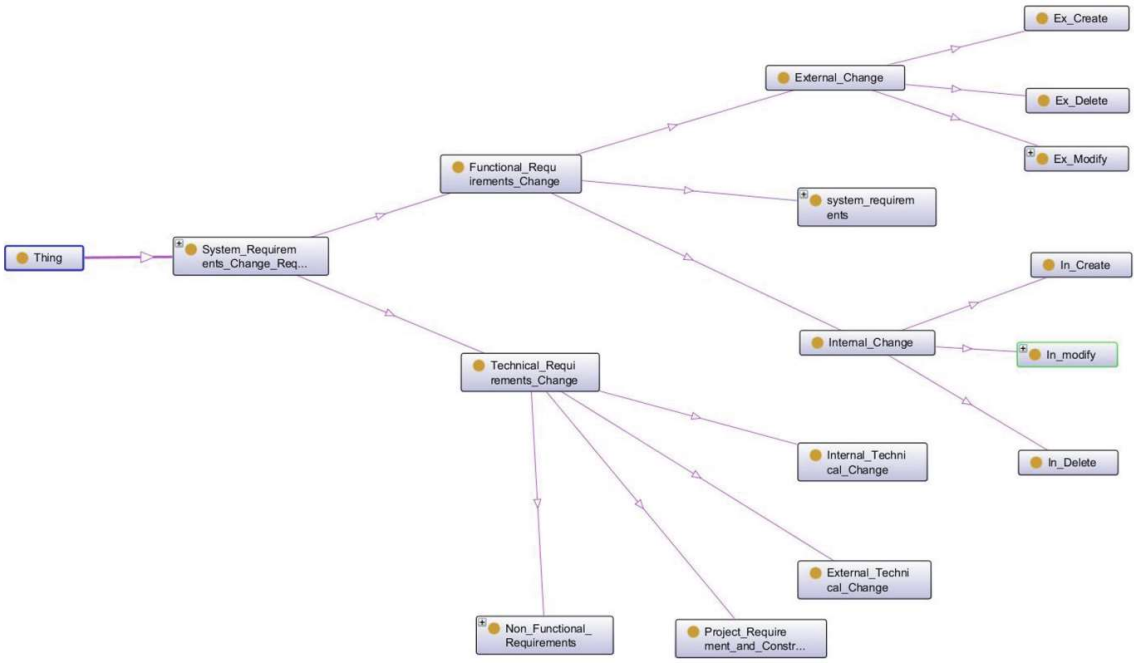

Рис. 4. Онтологічна модель класифікації запитів на зміни системних вимог [8]

Таблиця 1

Специфікації класів онтологічної моделі

Клас	Опис
System requirements change requests (RFC системних вимог)	Запити майбутніх користувачів на зміни системних вимог зі сховища вимог
Functional change (Функціональна зміна)	Запити майбутніх користувачів на зміни, які впливають на функціональні вимоги користувачів
Technical change (Технічна зміна)	Запити майбутніх користувачів на зміни, які впливають на нефункціональні вимоги або проєктні вимоги та обмеження
System requirements (Системні вимоги)	RFC комунікаторів – внутрішніх (системні аналітики, розробники, проєктувальники тощо) або зовнішніх (оператори, користувачі, керівники тощо)
Delete (Видалити)	Містить вимоги, які необхідно видалити
Modify (Модифікувати)	Містить вимоги, які необхідно модифікувати
Create (Створити)	Містить вимоги, які необхідно створити

б) «Пошук інформації» (інформація про тих, хто намагається отримати інформацію або допомогу);

в) «Запит на функцію» (ідеї, пропозиції чи потреби щодо покращення функціональності та продуктивності програми);

г) «Виявлення проблеми» (речення, що описують проблеми з програмами або їхню несподівану поведінку).

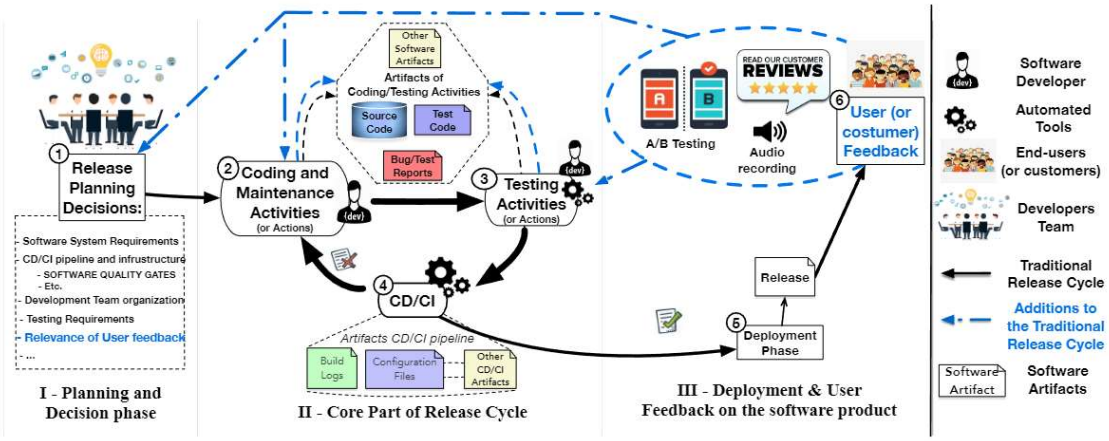

Рис. 5. Схема життєвого циклу розробки релізу мобільного застосунку із вбудованою системою управління змінами [9]

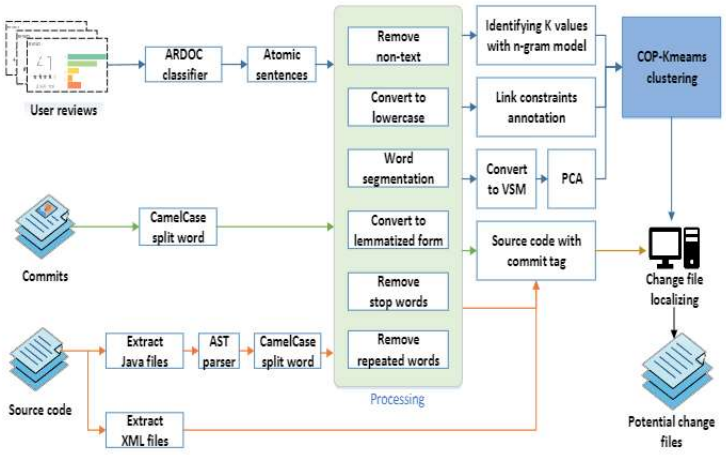

Рис. 6. Схему підходу, який знаходиться в основі запропонованої у [9] технології управління змінами

Для перетворення інформації з відгуків користувачів у RFC у [9] запропоновано вирішити задачу класифікації відгуків користувачів, виділяючи для подальшої обробки тільки ті відгуки, які слід віднести до категорій в) або г), наведених вище. Для цього застосовано ARDOC-класифікатор, який перетворює відгуки користувачів на окремі речення, а потім класифікує ці речення в одну з чотирьох згаданих вище категорій.

Недоліком такої класифікації є надто примітивний поділ усіх RFC на два великих класи (категорії в) та г)). Тому підмножина відгуків користувачів, які за допомогою ARDOC-класифікатора визнали такими, що належать до категорій в) або г), розглянутих вище, далі обробляється шляхом вирішення задачі кластеризації. Результатом вирішення цієї задачі є розподіл усіх відгуків користувачів вихідної підмножини на окремі кластери, які визначають актуальні для поточної ітерації ЖЦ релізу мобільного застосунку класи RFC. Для вирішення задачі кластеризації у [9] запропоновано використовувати алгоритми сімейства k-means із додатковими обмеженнями.

Як приклад класифікації RFC, яка враховує узагальнений досвід управління змінами різних IT-проєктів, пропонується розглянути класифікацію RFC, що формуються під час перегляду програмного коду. Для цього у [10] було проаналізовано 10 різних IT-проєктів,

які виконувалися за період 2012-2017 рр. за різноманітними замовленнями та які розрізнялися значеннями характеристик. Крім того, автори [10] спиралися на результати визначення таксономії RFC, яку було розроблено у [11]. Результатом дослідження [10] є скоригована модель змін, які виникають під час перегляду коду. Ця модель (з виключенням розгорнутих описів класів RFC) наведена у табл. 2.

Таблиця 2

Скоригована модель змін, які виникають під час перегляду коду

Artifact	Activity	Category	Topic	Detailed Change
1	2	3	4	5
Production & Test Code	Maintain-ability & Perfective Maintenance	Documentation (D)	- Textual Documentation	(D.1) - Naming. (D.2) - Comments. (D.3) - License Header. (D.4) - Typos. (D.5) - Other.
			- Language Supported Documentation	(D.6) - Immutability. (D.7) - Visibility.
		Style (S)	- Style	(S.1) – Brackets & Braces. (S.2) – Indentation. (S.3) – Blank Lines. (S.4) – Long Lines. (S.5) – Whitespace Usage. (S.6) – Grouping. (S.7) – Commented out code.
		Structure (STR)	- Re-implementation	(STR.1) – Semantic Duplication. (STR.2) – Semantic Dead Code. (STR.3) – Change Function. (STR.4) – Standard Coding Conventions. (STR.5) – New Functionality. (STR.6) – Strings (Wording). (STR.7) – Logging. (STR.8) – Testing.
	Maintain-ability & Perfective Maintenance	Structure (STR)	- Organization	(STR.9) – Imports. (STR.10) – Move Functionality. (STR.11) – Long Sub Routine. (STR.12) – Dead Code. (STR.13) – Duplication / Redundant Code. (STR.14) – Complex Code/ Simplification. (STR.15) – Statement Issue. (STR.16) – Consistency. (STR.17) – Architectural Changes.
	Functionality / Corrective Maintenance	Interface (I)	- Interface	(I.1) – Function Call. (I.2) – Parameter.

Кінець таблиці 2

1	2	3	4	5
Production & Test Code	Functionality / Corrective Maintenance	Logic (L)	- Logic	(L.1) – Compare. (L.2) – Computation. (L.3) – Wrong Location. (L.4) – Algorithm / Performance.
		Resource (R)	- Resource	(R.1) – Variable Initialization. (R.2) – Memory Management. (R.3) – Data & Resource Manipulation. (R.4) – Security. (R.5) – Concurrency.
		Check (C)	- Check	(C.1) – Check Function. (C.2) – Check Variable. (C.3) – Check User Input.
		Larger Defects (LD)	- Larger Defects	(LD.1) – Completeness. (LD.2) – GUI. (LD.3) – Check outside code / Domino Effects.
Other Changes	–	–	–	(O.1) – Commit Message. (O.2) – Continuous Integration / Continuous Delivery configurations. (O.3) – Automated Static Analysis Tools configurations. (O.4) – Language and Framework specific. (O.5) – External Software Documentation. (O.6) – Runtime Configurations. (O.7) – Other.

Результати порівняльного аналізу розглянутих варіантів класифікацій RFC дозволяють зробити такі висновки:

а) єдиної сталої системи класифікації RFC, яка охоплювала б увесь ЖЦ ІТ-проєкту, не існує;

б) кожен різновид ІТ-проєкту може мати власну множину типів RFC, яка, в свою чергу, може розглядатися як підмножина загальних моделей класифікації таких RFC;

в) в будь-якому випадку загальний поділ RFC на окремі підмножини виділяє функціональні і технічні RFC;

г) кількість класів RFC може сильно залежати від таких характеристик ІТ-проєкту:

– розмір ІТ-проєкту;

– час, який відведено на виконання ІТ-проєкту;

– необхідна швидкість реагування на потреби у змінах;

– рівень кваліфікації учасників ІТ-проєкту, які мають право визначати потреби у змінах та безпосередньо формувати RFC;

– кількість очікуваних або вже наявних RFC, які вимагають класифікації та ранжування.

Проведений аналіз існуючих класифікацій RFC показав, зокрема, що спроби вирішення задачі класифікації цих запитів приводить до створення складних моделей

класифікації, які є надлишковими з точки зору окремих IT-проєктів. Цю надлишковість слід вважати проявою складного характеру управління змінами. Тому виникає проблема надання гнучкості існуючим процесам, підпроцесам і роботам за умови, що ця гнучкість допоможе подолати складності управління змінами. Ця проблема вважається одним з найважливіших викликів для такого напряму робіт, як автоматизація управління змінами [12]. Для подолання цього виклику у [12] запропоновано розглядати автоматизовані системи управління змінами у проєктах як різновид систем підтримки гнучкого прийняття рішень, що є одним з напрямів досліджень в галузі штучного інтелекту. Окремі сучасні дослідження [12], [13] вважають, що методи штучного інтелекту здатні не тільки підтримувати окремі роботи, а й керувати комплексом процесів актуального фреймворку управління змінами. Тому виникає необхідність проведення дослідження можливості застосування артефактів штучного інтелекту у елементах актуального фреймворку управління змінами.

Для проведення такого дослідження у [12] було запропоновано розглядати процеси, підпроцеси і роботи актуального фреймворку управління змінами як систему, яка повинна вдовольнити визначені у [14] загальні бізнес-потреби:

а) бізнес-потреба 1 – автоматизація (як основний спосіб подолання збільшення складності та зростання потужностей, необхідних для втілення управління змінами в систему управління IT-проєктом);

б) бізнес-потреба 2 – підтримка прийняття рішень;

в) бізнес-потреба 3 – оптимізація;

г) бізнес-потреба 4 – контроль та нагляд.

Для вдоволення цих бізнес-потреб у [12] було досліджено можливості застосування таких артефактів штучного інтелекту:

– алгоритми машинного навчання;

– методи штучного інтелекту, спрямовані на комунікацію, сприйняття та дію (communicating, perceiving and acting AI-methods);

– алгоритми вирішення проблем;

– алгоритми здобування знань, міркування та планування;

– підходи до обробки невизначених знань та міркувань, зокрема, ймовірнісне представлення знань, байєсівські мережі, нечітка логіка, мультиагентні системи.

Крім цих артефактів, в [12] також були розглянуті такі прогнози щодо можливих напрямів розвитку артефактів штучного інтелекту в сфері управління змінами:

– дослідження застосування штучного інтелекту в управлінні змінами, зосереджені в основному на етапах попереднього затвердження RFC та після впровадження цих запитів;

– дослідження мета-евристичних алгоритмів та алгоритм машинного навчання на основі популяції як універсальних алгоритмів штучного інтелекту для оптимізації та автоматизації завдань, що виникають під час управління змінами;

– дослідження мультиагентних систем, потрібні для повної автоматизації процесів управління змінами, оскільки потоки даних у фреймворку управління змінами надто децентралізовані, а процеси управління змінами надто складні;

– дослідження особливостей виконання більшості тестів алгоритмів штучного інтелекту на даних реального світу (тобто окремих проєктів) внаслідок відсутності стандартного набору даних для тестування нових алгоритмів в галузі управління змінами;

– розширення впровадження результатів досліджень методів і моделей штучного інтелекту в галузі управління змінами;

– процеси управління знаннями керуються експертними знаннями та дуже індивідуальні, що вимагає значної обробки даних;

– дані, які виникають у фреймворку управління змінами, здебільшого розпливчасті, а результати процесів часто невизначені, що вимагає використання нечіткої логіки та ймовірнісних методів;

– дослідження управління змінами на основі штучного інтелекту рідко порівнюють свій підхід із уже існуючими рішеннями, демонструючи загальну відсутність строгості.

Результати дослідження можливості застосування артефактів штучного інтелекту для вдоволення бізнес-потреб управління змінами представлено у табл. 3 [12].

Таблиця 3

Результати дослідження можливості застосування артефактів штучного інтелекту для вдоволення бізнес-потреб управління змінами

Бізнес-потреба	Галузь дослідження штучного інтелекту	Можливість застосування знань з інших досліджень штучного інтелекту
Бізнес-потреба 1	Моделювання і розробка мультиагентних систем	Так
Бізнес-потреба 2	Підтримка прийняття рішень шляхом використання об'єктивних даних, машинне навчання без вчителя для автоматичної кластеризації змін у схожі пакети	Так
Бізнес-потреба 3	Мета-евристичний підхід для визначення оптимального терміну набрання чинності	Так
Бізнес-потреба 4	Постійне прогнозування результату виконання елементів фреймворку управління змінами	Ні
	Валідація в польових умовах на прикладі конкретного ІТ-проєкту та порівняння різних підходів	Так

Базуючись на наведених у табл. 3 результатах, можна зробити такі висновки:

а) вибір методів збору та класифікації потреб користувачів ІТ-продукту як інструментальних засобів робіт створення та ранжування RFC як елементів актуального фреймворку управління змінами бажано розглядати як задачу підтримки прийняття рішень на основі об'єктивних даних;

б) рекомендацію найвигіднішого методу збору та класифікації потреб користувачів ІТ-продукту із запропонованих альтернатив бажано розглядати як результат застосування одного із засобів реалізації мета-евристичного підходу;

в) перевірку рекомендованого методу бажано проводити в польових умовах на прикладі конкретного ІТ-проєкту;

г) бажано розглядати рішення задачі вибору методів збору та класифікації потреб користувачів ІТ-продукту як інструментальних засобів робіт створення та ранжування RFC як елементів актуального фреймворку управління змінами у вигляді елементів мультиагентної системи.

Результати проведеного аналізу дозволяють стверджувати, що роботу «Визначте/уточніть потребу в змінах» слід визнати практично недослідженою роботою, виконання якої визначає усі особливості виконання подальших робіт актуального фреймворку управління змінами. Тому проведення досліджень, які дадуть змогу виробити рекомендації з

виконання цієї роботи у конкретних ІТ-проєктах та узагальнити ці рекомендації для окремих різновидів ІТ-проєктів, слід вважати теоретично та практично актуальним.

3. Мета і задачі дослідження

Метою даного дослідження є розробка методу вирішення задачі формування і класифікації RFC ІТ-продукту. Досягнення цієї мети дозволить автоматизувати вирішення задачі формування і класифікації RFC ІТ-продукту, що значно спростить управління змінами у різних ІТ-проєктах однієї й тієї ж ІТ-компанії.

Для досягнення цієї мети треба вирішити такі задачі:

– розробка моделі RFC ІТ-продукту;

– розробка методу вирішення задачі формування і класифікації RFC ІТ-продукту.

4. Матеріали і методи дослідження

Об'єктом дослідження є робота «Визначте/уточніть потребу в змінах» підпроцесу «Формулювання змін» процесу «Управління змінами» [1]. Предметом дослідження є методи збору та класифікації потреб користувачів ІТ-продукту.

Головною особливістю дослідження слід вважати відсутність формального представлення моделей класифікації RFC. Тому як основну гіпотезу даного дослідження запропоновано розглядати модель RFC та метод вирішення задачі формування і класифікації запитів на зміну ІТ-продукту як результати узагальнення існуючих ІТ, які використовувалися для перетворення описів потреб учасників ІТ-проєкту у змінах на RFC та подальшої класифікації отриманих RFC.

Результати порівняльного аналізу розглянутих вище варіантів класифікацій RFC дозволяють встановити два основні різновиди таких ІТ.

Перший різновид має в своїй основі апріорно визначену керівництвом ІТ-проєкту модель, яка декларативно описує множину припустимих класів RFC. Приклад такої ІТ наведено у [5]. Ця апріорно визначена модель використовується в ІТ для вирішення задачі класифікації, яка встановлює належність кожної окремої потреби у зміні до конкретного класу RFC та дає змогу перетворити опис цієї потреби у публікацію відповідного RFC встановленого класу.

В загальному випадку цей перший різновид ІТ можна представити як послідовність таких кроків.

Крок 1. Учасником ІТ-проєкту формується та публікується потреба у зміні ІТ-продукту (у вигляді тексту).

Крок 2. Формується дерево рішень на основі моделі класифікації RFC, яку було апріорно обрано керівництвом ІТ-проєкту.

Крок 3. Для публікації потреби у зміні вирішується задача класифікації з використанням дерева рішень, сформованого на Кроці 2.

Крок 4. На основі класифікованої публікації потреби у зміні формується публікація RFC, клас якого визначено як результат виконання Кроку 3. Завершення роботи технології.

Діаграму потоків даних ІТ на основі апріорно визначеної моделі, яка декларативно описує множину припустимих класів RFC, наведено на рис. 7. Для побудови цієї та наступних діаграм потоків даних використано нотацію, яку було запропоновано у онлайн-CASE-засобі Draw.io v. 2.0.4.

Тут і надалі термін «публікація» пропонується застосовувати за аналогією із запропонованим у [15] терміном «публікація вимог до інформаційної системи». Така аналогія викликана загальним представленням RFC як різновидів вимог, які висуваються до

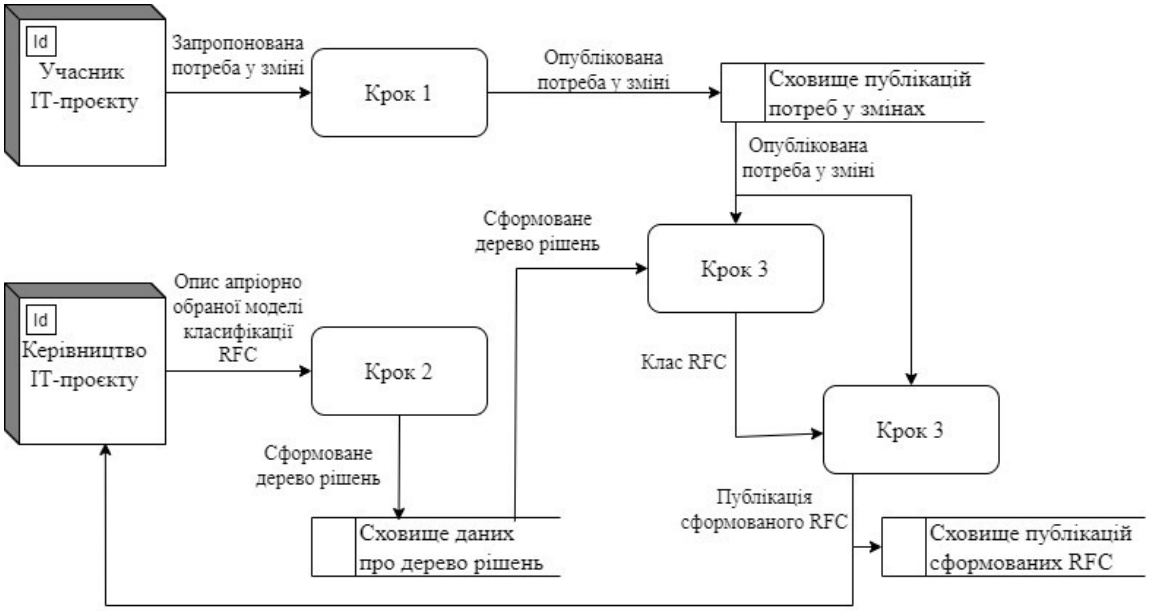

Рис. 7. Діаграма потоків даних інформаційної технології на основі апріорно визначеної моделі, яка декларативно описує множину припустимих класів запитів на зміни

ІТ-продуктів протягом ЖЦ існування цих продуктів.

Під терміном «публікація вимоги до інформаційної системи» слід розуміти [15]:

а) опис умови або можливості, які необхідні Споживачу ІТ-послуг для вирішення проблеми або досягнення мети, виконаний одним з припустимих в рамках методології розробки інформаційної системи способів;

б) опис, виконаний одним з припустимих в рамках методології розробки інформаційної системи способів, умови або можливості, якими повинна володіти інформаційна система або компонент інформаційної системи (ІТ-послуга, ІТ-сервіс) з точки зору Постачальника або Споживача ІТ-послуг та які відповідають договору, стандарту, специфікації або іншому офіційному документу,.

Виходячи з цього визначення, під терміном «публікація потреби у зміні» слід розуміти опис умови або можливості, які необхідні Споживачу ІТ-послуг для вирішення проблеми або досягнення мети, які не були визначені у поточній версії ІТ-продукту, виконаний у вигляді неструктурованого або слабко структурованого тексту. Під терміном «публікація сформованого RFC» слід розуміти:

а) опис, виконаний одним з припустимих в рамках методології розробки інформаційної системи способів, умови або можливості, які необхідні Споживачу ІТ-послуг для вирішення проблеми або досягнення мети та які не були визначені у поточній версії ІТ-продукту,;

б) опис, виконаний одним з припустимих в рамках методології розробки ІТ-продукту способів, умови або можливості, якими повинна володіти наступна версія ІТ-продукту з точки зору Постачальника або Споживача ІТ-послуг та які відповідають договору, стандарту, специфікації або іншому офіційному документу.

Визначення термінів «Споживач ІТ-послуг» та «Постачальник ІТ-послуг» наведено у [15].

Другий різновид ІТ автоматизованого виконання роботи «Визначити/уточнити потреби у змінах» побудовано з врахуванням можливості відсутності під час виконання цієї роботи апріорно визначеної керівництвом ІТ-проєкту існуючої моделі класифікації RFC. Приклад такої технології розглянуто у [9]. В загальному випадку цей другий різновид ІТ можна представити як послідовність таких кроків.

Крок 1. Формування та публікація учасником ІТ-проєкту потреби у зміні ІТ-продукту (у вигляді тексту).

Крок 2. Попереднє розподілення опублікованих потреб у змінах ІТ-продукту на класи функціональних та технічних потреб в змінах.

Крок 3. Для кожного з класів потреб в змінах, визначених на Кроці 2, вирішення задачі виділення множин ключових слів, які характеризують кожну конкретну публікацію потреби в зміні ІТ-продукту.

Крок 4. Для кожного з класів потреб в змінах, визначених на Кроці 2, вирішення задачі кластеризації на основі множин ключових слів, сформованих на Кроці 3.

Крок 5. Формування та публікація RFC з врахуванням результатів виконання Кроку 2 і Кроку 4. Завершення роботи технології.

Діаграму потоків даних ІТ з відсутньою апріорно визначеною моделлю, яка декларативно описує множину припустимих класів RFC, наведено на рис. 8.

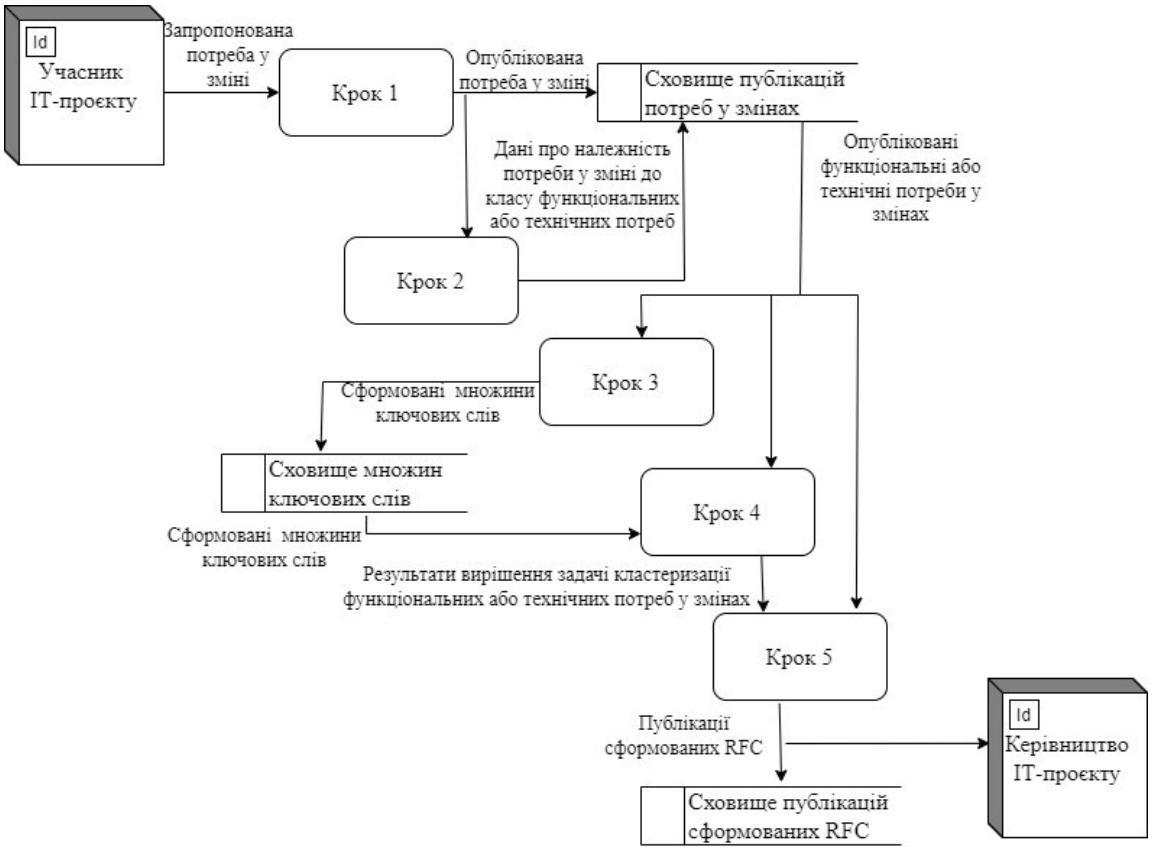

Рис. 8. Діаграма потоків даних інформаційної технології з відсутньою апріорно визначеною моделлю, яка декларативно описує множину припустимих класів запитів на зміни

Подібність розглянутих технологій дозволяє висунути припущення, що ці різновиди є частковими випадками реалізації загального методу вирішення задачі формування і класифікації RFC.

5. Результати дослідження

5.1. Розробка моделі запиту на зміну ІТ-продукту

Першим припущенням, на якому базується модель RFC ІТ-продукту, є припущення про формування і публікацію будь-якої потреби у зміні ІТ-продукту у вигляді

неструктурованого чи слабко структурованого тексту від якогось стейкхолдера ІТ-проєкту. Цим стейкхолдером може бути:

– майбутній оператор чи користувач ІТ-продукту;

– керівництво організації, яка виступає в ролі Споживача ІТ-послуг;

– один чи кілька виконавців ІТ-проєкту, очікуваним результатом якого є ІТ-продукт;

– керівництво ІТ-компанії, яка виступає в ролі Постачальника ІТ-послуг;

– представник фізичної чи юридичної особи, яка виступає в ролі спонсора ІТ-проєкту;

– будь-яка фізична особа чи представник юридичної особи, які приймають на себе одну з ролей в ІТ-проєкті, очікуваним результатом якого є ІТ-продукт.

Це припущення дозволяє формально описати будь-яку потребу у зміні, що виникає під час ЖЦ ІТ-проєкту, очікуваним результатом якого є ІТ-продукт, як кортеж

$$N_C = \langle Id_S, Text_C \rangle, \tag{1}$$

де N_C – потреба у зміні (need for change); Id_S – ідентифікатор ролі та/або особи (стейкхолдера), яка виступає як джерело потреби у зміні; $Text_C$ – текстовий опис потреби у зміні.

Розглянуті вище моделі класифікації RFC є частковими випадками деревоподібної структури, корінь якої визначає належність RFC (потреби у зміні), на основі якої формується RFC, до одного з двох базових класів – функціональних або технічних RFC (потреб у зміні). Тому друге припущення, на якому базується модель RFC ІТ-продукту, полягає у можливості формального представлення будь-якої моделі класифікації RFC, яку керівництво ІТ-проєкту може апріорно визначити для застосування під час вирішення задачі формування і класифікації RFC, як деревоподібного графу

$$G_{Cl_{RFC}} = (Attr, V), \tag{2}$$

де $G_{Cl_{RFC}}$ – деревоподібний граф, який формально описує моделі класифікації RFC; $Attr$ – множина вершин деревоподібного графа $G_{Cl_{RFC}}$, яка одночасно є множиною атрибутів, що описують ознаки класифікації RFC (потреби у зміні), яка має вигляд

$$Attr = (attr_t, attr_1, attr_2, ..., attr_i, ..., attr_z), \tag{3}$$

де $attr_t$ – атрибут, значення якого визначає належність RFC (потреби у зміні) до одного з двох базових класів – функціональних або технічних RFC (потреб у зміні), та який є коренем деревоподібного графу $G_{Cl_{RFC}}$; $attr_i$ – і-й атрибут, що є вузлом розгалуження або листом деревоподібного графу $G_{Cl_{RFC}}$, значення якого визначає належність RFC (потреби у зміні) до окремої підмножини класів (вузол розгалуження) або конкретного класу (лист) RFC (потреби у зміні); z – кількість вершин деревоподібного графу, яка визначається як кількість атрибутів у апріорно обраній моделі класифікації RFC, обраної керівництвом ІТ-проєкту; V – множина ребер, які зв'язують окремі пари вершин $(attr_i, attr_j)$ деревоподібного графу $G_{Cl_{RFC}}$ та ідентифікуються окремими значеннями атрибуту $attr_i$, який є початком відповідного ребра.

В результаті вирішення задачі класифікації встановлюється клас RFC (потреби у зміні). Цей клас може бути представлений як послідовність значень атрибутів класифікації, які описують шлях деревоподібним графом (2) від кореневої вершини до одного з листів графа. Але слід врахувати такі умови, які накладаються на визначення цього шляху:

а) апріорно визначена модель класифікації у вигляді деревоподібного графа (2) в

конкретному IT-проєкті може бути відсутня;

б) апріорно визначена модель класифікації у вигляді деревоподібного графа (2) може доповнюватися новими вершинами та ребрами, які визначаються за результатами вирішення задачі кластеризації публікацій потреб у змінах IT-продукту;

в) апріорно визначена модель класифікації у вигляді деревоподібного графа (2) може створюватися в результаті багатократного вирішення задачі кластеризації публікацій потреб у змінах IT-продукту.

Врахування цих умов дозволяє описати будь-який і-й клас RFC (потреби у зміні) як кортеж з двох підмножин, що має вигляд

$$C_{iRFC} = \left(C_{iPt}, C_{igr}\right), \tag{4}$$

де C_{iRFC} – *i*-й клас RFC (потреби у зміні); C_{iPt} – підмножина значень атрибутів апріорно визначеної керівництвом IT-проєкту моделі класифікації RFC, що має вигляд

$$C_{iPt} = \begin{cases} \langle v_t, v_a, ..., v_b \rangle \text{ у випадку застосування моделі класифікації;} \\ \langle v_t \rangle \text{ в іншому випадку.} \end{cases}, \tag{5}$$

де $\langle v_t, v_a, ..., v_b \rangle$ – кортеж ребер, які описують шлях від кореневої вершини до b-го листа графу (2); $\langle v_t \rangle$ – кортеж, який вказує на належність RFC (потреби у зміні) до класу функціональних або технічних RFC (потреб у змінах); C_{igr} – підмножина значень додаткових атрибутів, встановлених у результаті вирішення задачі кластеризації, що має вигляд

$$C_{igr} = \begin{cases} \langle v_{b1}, ..., v_{bk} \rangle \text{ у випадку вирішення задачі кластеризації;} \\ \langle \varnothing \rangle \text{ в іншому випадку.} \end{cases}, \tag{6}$$

де $\langle v_{b1}, ..., v_{bn} \rangle$ – кортеж значень атрибутів, які використовуються для опису окремих кластерів, що виникають в результаті вирішення задачі кластеризації публікацій потреб у змінах.

Для забезпечення можливості вирішення задачі кластеризації з метою уточнення існуючої моделі класифікації потреб у змінах введемо множину Tez_{ITP} ключових слів, які описують IT-продукт, його елементи, їхні характеристики та властивості. При цьому кожна публікація потреби у змінах буде характеризуватися множиною ключових слів, які однозначно визначають цю публікацію, що має вигляд

$$\begin{bmatrix} kw_1 \\ ... \\ kw_j \\ ... \\ kw_p \end{bmatrix} \in Tez_{ITP}, \tag{7}$$

де kw_j – *j*-те ключове слово, яке характеризує публікацію потреби у зміні, $j=1,...,p$.

Розглянуті особливості вирішення задачі формування і класифікації RFC дозволяють представити модель *i*-го RFC IT-продукту M_{RFC_i} як кортеж, що має вигляд:

$$M_{RFC_i} = \left\langle Id_{S_i}, G_{Cl_{RFC}}, \left\langle C_{ij_{Pt}}, C_{ij_{gr}} \right\rangle, Tez_{ITP}, \begin{bmatrix} kw_{i1} \\ \dots \\ kw_{ij} \\ \dots \\ kw_{ip} \end{bmatrix}, Text_{RFC_i} \right\rangle, \quad (8)$$

де Id_{S_i} – ідентифікатор джерела *i*-го RFC (потреби у зміні); $C_{ij_{Pt}}$ – підмножина значень атрибутів апріорно визначеної керівництвом ІТ-проєкту моделі класифікації, які визначають клас C_j для *i*-го RFC (потреби у зміні), $j=1,\dots,m$; $C_{ij_{gr}}$ – підмножина значень додаткових атрибутів, встановлених у результаті вирішення задачі кластеризації, які визначають клас C_j для *i*-го RFC (потреби у зміні), $j=1,\dots,m$; m – кількість класів моделі класифікації M_{RFC_i}; $Text_{RFC_i}$ – публікація і-го RFC (потреби у зміні).

Модель (8) дозволяє розробити загальний метод вирішення задачі формування і класифікації RFC як основний інструмент формування і публікації потреб у змінах ІТ-продукту.

5.2. Розробка методу вирішення задачі формування і класифікації запитів на зміну ІТ-продукту

Розроблена модель (8) дозволяє деталізувати опис змісту етапів загального методу вирішення задачі формування і класифікації RFC. З використанням моделі (8) цей метод може бути представлений як послідовність таких етапів і кроків.

Етап 1. Визначення похідної моделі класифікації RFC, яку слід застосовувати для формування та класифікації потреб у змінах ІТ-продукту в конкретному ІТ-проєкті.

Крок 1.1. Вибір керівництвом ІТ-проєкту похідної моделі класифікації RFC $G_{Cl_{RFC}}$ на основі досвіду виконання попередніх ІТ-проєктів та аналізу результатів науково-прикладних досліджень.

Крок 1.2. Якщо в результаті виконання Кроку 1.1 обрано конкретну модель класифікації RFC $G_{Cl_{RFC}}$, то визначення підмножини атрибутів $Attr_{i_{Pt}}$ та їхніх значень, які утворюють постійну основу дерева рішень з класифікації RFC (потреби у змінах) ІТ-продукту в межах конкретного ІТ-проєкту.

Крок 1.3. Якщо в результаті виконання Кроку 1.1 жодну з існуючих моделей класифікації RFC $G_{Cl_{RFC}}$ не було обрано, то визначення підмножини атрибутів $Attr_{i_{Pt}}$, яка складається з атрибуту $attr_t$, та значень цього атрибуту, які визначають належність об'єкту класифікації до функціональних чи технічних RFC (потреб у змінах).

Крок 1.4. Визначення порогового значення ε_{RFC}, перевищення якого визначає необхідність уточнення моделі класифікації RFC (потреб у змінах) ІТ-продукту у конкретному ІТ-проєкті.

Етап 2. Формування та попередньої класифікації потреб у змінах ІТ-продукту.

Крок 2.1. Формування зацікавленим стейкхолдером ІТ-проєкту потреби в зміні ІТ-продукту N_C за моделлю (1).

Крок 2.2. У випадку, якщо підмножина $Attr_{i_{Pt}}$ атрибутів похідної моделі класифікації RFC складається з одного атрибуту $attr_t$, визначення стейкхолдером ІТ-проєкту типу RFC, якому відповідає сформована потреба у зміні ІТ-продукту.

Крок 2.3. У випадку, якщо підмножина $Attr_{i_{Pt}}$ атрибутів похідної моделі класифікації

RFC складається з двох чи більшої кількості атрибутів, визначення стейкхолдером ІТ-проєкту переліку значень цих атрибутів $\langle v_t, v_a, ..., v_b \rangle$, які визначають клас сформованої потреби у зміні ІТ-продукту.

Крок 2.4. Публікація сформованої потреби у зміні ІТ-продукту стейкхолдером ІТ-проєкту.

Етап 3. Уточнення та адаптація похідної моделі класифікації RFC до особливостей ІТ-продукту та ІТ-проєкту.

Крок 3.1. Визначити кількість публікацій Q_{N_j} потреб у змінах ІТ-продукту в кожному з класів C_j, $j=1,...,m$, виявлених за результатами виконання Кроків 2.2 та 2.3.

Крок 3.2. Якщо хоча б для одного з класів C_j виконується умова

$$Q_{N_j} > \varepsilon_{RFC}, \tag{9}$$

то перейти до Кроку 3.3. В іншому випадку перейти до Кроку 3.9.

Крок 3.3. З кожної публікації потреби в зміні, яку було віднесено до класу C_j, виділити множину ключових слів $\begin{bmatrix} kw_1 \\ ... \\ kw_j \\ ... \\ kw_p \end{bmatrix} \in Tez_{ITP}$, які однозначно характеризують дану публікацію.

Крок 3.4. Сформувати з множин ключових слів $\begin{bmatrix} kw_1 \\ ... \\ kw_j \\ ... \\ kw_p \end{bmatrix}$ публікацій потреб у змінах, які віднесено до класу , вибірку похідних даних для вирішення задачі кластеризації.

Крок 3.5. Вирішити задачу кластеризації для вибірки, сформованої на Кроці 3.4, з додатковим обмеженням на кількість елементів в кожному з кластерів, яка не повинна перевищувати значення ε_{RFC}.

Крок 3.6. Запропонувати множину кластерів, сформованих в результаті виконання Кроку 3.5, керівництву ІТ-проєкту як додаткові підкласи $C_{ij_{gr}}$, сформовані на основі публікацій потреб у змінах, віднесених до класу .

Крок 3.7. Якщо керівництво ІТ-проєкту приймає рішення про доцільність багаторазового використання запропонованих додаткових підкласів, сформувати підмножину атрибутів $Attr_{j_{gr}}$, елементи якої однозначно ідентифікують кожен з цих підкласів.

Крок 3.8. Адаптувати похідну модель класифікації RFC $G_{Cl_{RFC}}$ до нових особливостей ІТ-продукту та ІТ-проєкту, виконавши операцію

$$Attr_{i_{Pt}} = Attr_{i_{Pt}} \cup Attr_{i_{gr}}, \tag{10}$$

після чого повернутися на Крок 3.1.

Крок 3.9. Сформувати підмножини публікацій потреб у змінах ІТ-продукту, віднесених до виділених класів C_j, $j=1,...,m$, RFC.

Етап 4. Публікації та розсилання RFC стейкхолдерам ІТ-проєкту.

Крок 4.1. Перетворення публікацій потреб в змінах ІТ-продукту на публікації RFC за моделлю (8).

Крок 4.2. Визначення переліку стейкхолдерів ІТ-проєкту, яким слід надіслати відповідні публікації RFC, отримані в результаті виконання Кроку 4.1.

Крок 4.3. Розсилання публікацій класифікованих RFC множині стейкхолдерів ІТ-проєкту для наступних робіт з пріоритезації та оцінювання цих RFC. Завершення застосування методу.

6. Обговорення результатів дослідження

Розроблена модель (8) *i*-го RFC ІТ-продукту дозволяє описати множину функціональних і технічних RFC, що виникають під час розробки ІТ-продукту. На відміну від існуючих моделей, які зосереджено на описі результатів класифікації вже сформульованих RFC, ця модель описує RFC як результат виконання послідовності робіт з похідного формулювання потреби у зміні та поступового перетворення цієї потреби на RFC, який віднесено до відповідного класу. Ця відмінність дозволяє забезпечити покращення робіт з управління змінами, зокрема, з трасування джерел потреб, сформульованих потреб у змінах, публікацій RFC та зв'язків між ними.

Розроблений метод вирішення задачі формування і класифікації RFC базується на моделі (8) і дозволяє уніфікувати вирішення цієї задачі незалежно від особливостей обраної моделі класифікації, ІТ-проєкту та ІТ-продукту, який розробляється. При цьому, на відміну від існуючих, метод дозволяє не тільки застосовувати апріорно визначену модель класифікації RFC, а й адаптувати існуючу модель до нових особливостей ІТ-проєкту та ІТ-продукту та розробляти власну модель класифікації RFC, що якнайкраще підходить до особливостей діяльності ІТ-компанії, її ІТ-проєктів та ІТ-продуктів.

Використання запропонованих у дослідженні моделі (8) та методу вирішення задачі формування і класифікації RFC дає змогу автоматизувати виконання роботи «Визначте/уточніть потребу в змінах» підпроцесу «Формулювання змін». При цьому застосування розробленого методу дозволяє, на відміну від запропонованих у [5, 9] ІТ, зробити рішення з автоматизації достатньо близьким до універсального, яке не буде вимагати значних змін внаслідок можливої зміни сценарію вирішення задачі формування і класифікації RFC.

Головним недоліком отриманих результатів дослідження слід вважати їхню орієнтацію виключно на RFC, які виникають тільки під час виконання ІТ-проєкту з розробки чи модифікації ІТ-продукту. Це обмежує можливість застосування отриманих результатів і вимагає проведення додаткових досліджень формальних описів RFC, які можуть виникати на інших стадіях ЖЦ ІТ-проєкту та ІТ-продукту.

Виходячи з цього недоліку, головним напрямом подальших досліджень в галузі управління формулюванням змін слід визнати дослідження можливості уніфікації формальних описів RFC на різних стадіях ЖЦ ІТ-проєкту та ІТ-продукту. Підтвердження цієї можливості встановлює, у свою чергу, можливість розробки уніфікованих методів та інструментальних засобів управління змінами в межах відповідних ЖЦ. Спростування цієї можливості дозволить визначити якісні та кількісні ознаки ІТ-проєктів та ІТ-продуктів, наявність яких буде означати необхідність зміни методів та інструментів управління змінами.

Ще одним напрямом подальших досліджень у цій галузі є дослідження особливостей

методів збирання вимог, які слід застосовувати для збирання потреб у змінах IT-продукту. При цьому слід виходити з припущення, що потреби у змінах та RFC, створені на їх основі, розглядаються керівництвом IT-проєкту як вимоги до IT-продукту, що виникають після завершення стадії планування IT-проєкту і до моменту початку ініціації нового IT-проєкту, який буде стосуватися цього ж IT-продукту.

7. Висновки

У ході даного дослідження було вирішено задачу розробки методу вирішення задачі формування і класифікації запитів на зміну IT-продукту. Під час вирішення цієї задачі було здійснено:

– розробку моделі (8) *i*-го RFC IT-продукту, яка на формальному рівні дозволяє описати будь-які функціональні та технічні RFC, що виникають під час виконання IT-проєкту з розробки чи модифікації IT-продукту;

– розробку методу вирішення задачі формування і класифікації RFC, який дозволяє уніфікувати вирішення цієї задачі.

Отримані результати дозволяють в подальшому вирішити задачу розробки уніфікованих засобів автоматизованого вирішення задачі формування і класифікації RFC у вигляді елементів мультиагентної чи сервіс-орієнтованої інформаційної системи управління ЖЦ IT-продукту.

Перелік посилань:

1. Настанова до Зводу знань з управління проєктами. Настанова PMBOK. Сьоме видання. Стандарт з управління проєктами. Project Management Institute, Inc., 14 Campus Boulevard Newtown Square, Pennsylvania 19073-3299 USA, 2021. 370 c.

2. Managing change in organizations: a practice guide. Project Management Institute, Inc., 14 Campus Boulevard Newtown Square, Pennsylvania 19073-3299 USA. 2013. 122 p.

3. ISO/IEC/IEEE 12207:2017. Systems and software engineering – Software life cycle processes. Чинний від 2017-11-01. 150 p.

4. Віжан К. Процес управління змінами в IT-компанії, або Як побороти страх роботи з чендж реквестами. *DOU*. URL: https://dou.ua/forums/topic/42501/ (дата звернення 27.03.2024).

5. Riesener M, Dölle C, Mendl-Heinisch M, Schuh G, Keuper A. Derivation of Description Features for Engineering Change Request by Aid of Latent Dirichlet Allocation. *Proceedings of the Design Society*: *DESIGN Conference.* 2020. Vol. 1. P. 697-706. doi:10.1017/dsd.2020.98.

6. Jarratt T.A.W. et al. Engineering change. An overview and perspective on the literature. *Research in Engineering Design*. 2011. Vol. 22, No. 2. P. 103-124. https://doi.org/10.1007/s00163-010-0097-y

7. Feldhusen J., Grote K.-H. Pahl/Beitz Konstruktionslehre: Methoden und Anwendung erfolgreicher Produktentwicklung. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. https://doi.org/10.1007/978-3-642-29569-0

8. Sakhrawi Z., Sellami A., Bouassida N. Requirements Change Requests Classification: An Ontology-Based Approach. In: Abraham A., Siarry P., Ma K., Kaklauskas A. (eds) *Intelligent Systems Design and Applications. ISDA 2019. Advances in Intelligent Systems and Computing*, Vol. 1181. Springer, Cham. 2021. P. 487-496. https://doi.org/10.1007/978-3-030-49342-4_47

9. Zhou Y., Su Y., Chen T., Huang Z., Gall H.C., Panichella S. User Review-Based Change File Localization for Mobile Applications. *IEEE Transactions on Software Engineering*. 2019. Vol. 47 (12). P. 2755-2770.

10. Panichella S., Zaugg N. An Empirical Investigation of Relevant Changes and Automation Needs in Modern Code Review. *Empiric Software Engineering*. 2020. Vol. 25. P. 4833–4872. https://doi.org/10.1007/s10664-020-09870-3.

11. Beller M, Bacchelli A, Zaidman A, Jurgens E. Modern code reviews in open-source projects: which problems do they fix? *11Th working conference on mining software repositories*, *MSR 2014, proceedings*, 2014, Hyderabad, India. P. 202–211. https://doi.org/10.1145/2597073.2597082

12. Burggräf P., Wagner J., Saßmannshausen T. et al. AI-artifacts in engineering change management – a systematic literature review. *Research in Engineering Design*. 2024. Vol. 35. P. 215–237. https://doi.org/10.1007/s00163-023-00430-6.

13. Burggräf P, Wagner J, Saßmannshausen T.M. Sustainable Interaction of Human and Artificial Intelligence in Cyber Production Management Systems. In: Behrens B-A, Brosius A, Hintze W, Ihlenfeldt S, Wulfsberg JP (eds). *Production at the leading edge of technology.* WGP 2020. Lecture Notes in Production Engineering. Berlin, Heidelberg:

Springer, Berlin Heidelberg, 2021. P. 508–517.

14. Radisic-Aberger O., Weisser T., Saßmannshausen T., Wagner J., Burggräf P. Concept of a multi-agent system for optimized and automated engineering change implementation. *Proceedings of the Design Society*. 2022. Vol. 2. P. 1689–1698. https://doi.org/10.1017/pds.2022.171/

15. Левыкин, В.М., Евланов, М.В., Керносов, М.А. Паттерны проектирования требований: моделирование и применение: монография. Харьков: ООО «Компанія «Сміт»», 2014. 320 с.

Надійшла до редколегії 24.04.2024

Євланов Максим Вікторович, доктор технічних наук, професор, професор кафедри ІУС ХНУРЕ, м. Харків, Україна, e-mail: maksym.ievlanov@nure.ua, ORCID: https://orcid.org/0000-0002-6703-5166 (науковий керівник здобувача вищої освіти Крука Б.Є.).

Юр'єв Іван Олексійович, кандидат технічних наук, доцент, доцент кафедри ІУС ХНУРЕ, м. Харків, Україна, e-mail: ivan.iuriev@nure.ua, ORCID: https://orcid.org/0000-0002-5178-519X.

Крук Богдан Євгенійович, здобувач вищої освіти, група УПГІТм-22-1, факультет комп'ютерних наук, ХНУРЕ, м. Харків, Україна, e-mail: bohdan.kruk@nure.ua.

УДК 004.8:004.9 DOI: 10.30837/0135-1710.2024.182.025

С.Ф. ЧАЛИЙ, В.О. ЛЕЩИНСЬКИЙ

ПОБУДОВА ПРОЦЕСНО-ОРІЄНТОВАНИХ ПОЯСНЕНЬ В ІНТЕЛЕКТУАЛЬНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ НА ОСНОВІ МОЖЛИВІСНИХ КАУЗАЛЬНИХ ЗАЛЕЖНОСТЕЙ

Розглянуто процес побудови пояснень в інтелектуальних інформаційних системах. Запропоновано процесно-орієнтовану модель пояснення, яка інтегрує темпоральні та каузальні залежності процесу формування рішення. Модель забезпечує визначення послідовності станів процесу формування рішення на основі можливісних причинно-наслідкових зв'язків між діями такого процесу. Розроблено метод побудови процесно-орієнтованого пояснення на основі можливісних каузальних залежностей, який включає формування каузальних правил, оцінку можливості та необхідності цих правил, а також представлення пояснень з урахуванням атрибутів станів. Метод орієнтований на підтримку внутрішніх користувачів інтелектуальної системи.

1. Вступ

Інтелектуальні інформаційні системи (ІІС) сьогодні знаходять широке застосування у таких галузях, як охорона здоров'я, фінансові послуги, промисловість та управління бізнес-процесами. Ці системи здатні обробляти великі масиви даних, виявляти приховані закономірності та приймати рішення на основі алгоритмів машинного навчання [1]. Однак такі системи зазвичай залишаються «непрозорими» для користувачів внаслідок використання складних алгоритмів формування рішень, що може викликати недовіру до результатів роботи ІІС [2]. Відсутність розуміння того, як система дійшла до певного рішення, не лише знижує рівень довіри до її результатів, але й створює труднощі для виявлення та виправлення потенційних помилок [3]. Для вирішення цієї проблеми необхідно забезпечити користувачам доступ до зрозумілих пояснень щодо процесу прийняття рішень [4]. Такі пояснення повинні бути адаптованими до потреб різних категорій користувачів – як зовнішніх (кінцевих споживачів рішень), так і внутрішніх (розробників та аналітиків). Зовнішні користувачі зазвичай не мають глибоких технічних знань щодо внутрішньої структури ІІС і потребують обґрунтування рішень на концептуальному рівні. Внутрішні користувачі відповідають за налаштування та підтримку ІІС. Для них важливо мати детальне пояснення ключових етапів процесу формування рішень з метою оптимізації роботи системи та виявлення можливих «вузьких