https://doi.org/10.31891/2219-9365-2024-77-18
УДК 004.9

ВОЙЧУР Юрій
Хмельницький національний університет
https://orcid.org/0000-0003-3085-7315
e-mail: voichury@khmnhu.edu.ua
МЕДЗАТИЙ Дмитро
Хмельницький національний університет
https://orcid.org/0000-0002-1879-2945
e-mail: medza@ukr.net

МЕТОД АНАЛІЗУ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ НА ПРЕДМЕТ ПОШУКУ ЗНАЧЕНЬ АТРИБУТІВ ЯКОСТІ

Ринок програмного забезпечення (ПЗ) зростає дуже стрімко. ПЗ стає все більш складним, і на нього покладається все більша відповідальність. Зі зростанням надійності апаратного забезпечення та зростаючою складністю програмного забезпечення якість програмного забезпечення викликає дедалі більше занепокоєння як у розробників, так і у користувачів, щонайменше виходячи з прагнення досягти поставлені бізнес-цілі.

Наразі існує суперечність між зростаючою складністю програмного забезпечення та відповідальністю, яка на нього покладається, розширенням вимог до якості програмного забезпечення, з одного боку, і недосконалістю моделей, методів та засобів прогнозування якості ПЗ на ранніх етапах життєвого циклу. Відтак, прогнозування рівня якості програмного забезпечення на ранніх етапах життєвого циклу на основі атрибутів якості є актуальною задачею.

Розроблена структура специфікації вимог до ПЗ, придатної для автоматичного опрацювання системою для прогнозування рівня якості програмного забезпечення на основі вимог, накладає певні обмеження на формулювання вимог, що містять атрибути якості, та призначена для подальшого препроцесінгу специфікаційї вимог до ПЗ.

Метод аналізу вимог до програмного забезпечення на предмет пошуку значень атрибутів якості забезпечує вибір значень атрибутів якості ПЗ з природомовної специфікації вимог до ПЗ, які далі можуть бути використані для визначення значень характеристик якості ПЗ та для комплексного оцінювання якості ПЗ. Розроблений метод є важливим для повної автоматизації опрацювання вимог та повного усунення суб'єктивного впливу і участі людини у процесах опрацювання інформації та здобуття знань. Метод аналізу вимог до ПЗ на предмет пошуку значень атрибутів якості є теоретичним підгрунтям для розроблення модулю автоматичного аналізу вимог до ПЗ пропонованої авторами системи для прогнозування рівня якості програмного забезпечення на основі вимог.

Ключові слова: програмне забезпечення, якість програмного забезпечення, атрибути якості програмного забезпечення, вимоги до програмного забезпечення, аналіз вимог до програмного забезпечення.

VOICHUR Yurii, MEDZATYI Dmytro
Khmelnytskyi National University

METHOD FOR ANALYZING THE SOFTWARE REQUIREMENTS FOR FINDING THE VALUES OF QUALITY ATTRIBUTES

The software market is growing very rapidly. Software is becoming more and more complex, and more and more responsibility is being placed on it. With the increasing reliability of hardware and the growing complexity of software, software quality is a growing concern for both developers and users, at least in terms of achieving business goals.

Currently, there is a contradiction between the growing complexity of software and the responsibility it carries, the expansion of software quality requirements, on the one hand, and the imperfection of models, methods and tools for predicting software quality at early stages of the life cycle, on the other. Therefore, predicting the level of software quality at the early stages of the life cycle based on quality attributes is an urgent task.

The developed structure of the software requirements specification, suitable for automatic processing by a system for predicting the quality level of software based on requirements, imposes certain restrictions on the formulation of requirements containing quality attributes and is intended for further preprocessing of software requirements specifications.

The method for analyzing the software requirements for finding the values of quality attributes provides the selection of values of software quality attributes from the natural language specification of software requirements, which can then be used to determine the values of software quality characteristics and for a comprehensive assessment of software quality. The developed method is important for full automation of requirements processing and complete elimination of subjective influence and human participation in the processes of information processing and knowledge acquisition. The method for analyzing the software requirements for finding the values of quality attributes is the theoretical basis for developing a module for automatic analysis of software requirements of the system proposed by the authors for predicting the level of software quality based on requirements.

Keywords: software, software quality, software quality attributes, software requirements, software requirements analysis.

Постановка проблеми у загальному вигляді та її зв'язок із важливими науковими чи практичними завданнями

Ринок програмного забезпечення (ПЗ) зростає дуже стрімко. ПЗ стає все більш складним, і на нього покладається все більша відповідальність. Зі зростанням надійності апаратного забезпечення та зростаючою складністю програмного забезпечення якість програмного забезпечення викликає дедалі більше занепокоєння як у розробників, так і у користувачів, щонайменше виходячи з прагнення досягти поставлені бізнес-цілі.

В даний час розробка програмного забезпечення перетворилася в одну з найдорожчих індустрій, і будь-які вузькі місця в технологічному процесі його створення можуть призвести до небажаних наслідків (збільшення термінів розробки програмного забезпечення, подорожчання ПЗ, зниження продуктивності ПЗ, неможливість виконання необхідних функціональних задач розробленим ПЗ, зниження конкуретнтоздатності та репутації компанії-розробника, неефективне прийняття рішень інформаційною технологією, складовою частиною якої є розроблене ПЗ, тощо) [1, 2].

Сучасна індустрія ПЗ характеризується високою конкуренцією. Із зростанням глобалізації та вільних ринків користувачі програмного забезпечення стають все більш впливовими, маючи можливість купувати або відмовлятися від програмного забезпечення через його незадовільну якість.

Якість ПЗ – це ступінь, в якому ПЗ відповідає потребам користувача при його застосуванні у визначених умовах [3]. Очевидно, що найбільш зацікавленим у якості програмного забезпечення є кінцевий користувач, який використовує програмний продукт. Чимало програмних проектів не відповідають бізнес-вимогам і потребам користувача і не виконуються у встановлені часові рамки в рамках виділеного бюджету, тобто не є якісними та успішними [4]. Частота появи та відсотки впливу на якість ПЗ факторів, пов'язаних із ранніми етапами життєво-го циклу, є найвищою (порядку третини) [5]. Тому, розвиток сучасних технологій розроблення ПЗ вимагає динамічного розвитку засобів оцінки, а особливо прогнозування якості ПЗ на ранніх етапах життєвого циклу (з точки зору економічної та часової доцільності). Своєчасне прогнозування якості ПЗ може бути використано для прийняття будь-яких превентивних заходів для зменшення кількості збоїв ПЗ під час його роботи. Бажано вже за наявності специфікації вимог до ПЗ розуміти, чи за цією специфікацією може бути розроблене якісне ПЗ. Ряд проведених досліджень [6, 7] показали, що версії ПЗ, написані різними розробниками за однаковими вимогами, містили ряд спільних помилок, пов'язаних із помилками або неточностями вимог, і навпаки версії ПЗ, написані одним і тим же колективом розробників за різними вимогами, суттєво різнились своєю якістю та успішністю. Загальновизнано, що якісна інженерія вимог приводить до підвищення якості про-грамного забезпечення і значно знижує ризик невдачі або перевитрати бюджету проектів з розробки програмного забезпечення [8].

Оцінювання якості ПЗ за стандартом ISO 25010:2011 [3] здійснюється наступним чином: на основі 138 атрибутів якості, зазначених в ISO 25023:2016 [9], оцінюються підхарактеристики та характеристики якості, які, в свою чергу, надають комплексну оцінку якості ПЗ. Зараз оцінювання атрибутів для оцінювання якості ПЗ відбувається, в основному, лише для готового програмного коду, проте всі необхідні атрибути закладено у специфікації вимог до ПЗ. Саме у вимогах до ПЗ описано поведінку майбутнього ПЗ, функціональні можливості та обмеження ПЗ, його властивості та атрибути. Значення атрибутів якості враховують і мету програмного проекту, і його тип. Якщо атрибут(и) вписано у вимоги специфікації, визначено значення такого атрибуту, відтак розробники будуть зобов'язані забезпечити наявність такого атрибуту в своєму ПЗ і зазначене у вимогах значення атрибуту, інакше ПЗ не пройде верифікації та валідації. Отже, вже на основі значень атрибутів зі специфікації вимог можна прогнозувати рівень якості програмного забезпечення для програмних проектів будь-якого типу.

Проведений у [10] аналіз відомих моделей, методів та засобів для прогнозування рівня якості програмного забезпечення на ріннх етапах життєвого циклу показав, що, хоча проаналізовані моделі, методи та засоби і мають великий потенціал для використання в різних контекстах, проте вони не задають залежності характеристик якості від атрибутів, не забезпечують обчислення кількісних значень характеристик якості на основі атрибутів, не задають залежності якості від характеристик якості, не забезпечують обчислення кількісних значень якості на основі характеристик та не забезпечують прогнозування рівня якості ПЗ на основі отриманого кількісного значення.

Отже, наразі існує суперечність між зростаючою складністю програмного забезпечення та відповідальністю, яка на нього покладається, розширенням вимог до якості програмного забезпечення, з одного боку, і недосконалістю моделей, методів та засобів прогнозування якості ПЗ на ранніх етапах життєвого циклу. Відтак, прогнозування рівня якості програмного забезпечення на ранніх етапах життєвого циклу на основі атрибутів якості є актуальною задачею.

У [11] розроблена система для прогнозування рівня якості програмного забезпечення на основі вимог, але одним з її модулів є модуль автоматичного аналізу вимог до ПЗ на предмет пошуку знаяень атрибутів якості, який працюватиме на основі методу аналізу вимог до ПЗ на предмет пошуку значень атрибутів якості, який є важливим для повної автоматизації опрацювання вимог та повного усунення суб'єктивного впливу і участі людини у процесах опрацювання інформації та здобуття знань, і розробленню якого і буде присвячено дане дослідження.

Метод аналізу вимог до програмного забезпечення на предмет пошуку значень атрибутів якості

Враховуючи атрибути якості, зазначені в ISO 25023:2016 [9], та використовуючи метод ідеалізації (накладення певних обмежень), розробимо відповідну структуру специфікації вимог до ПЗ для виконання препроцесінгу специфікації.

Отже, специфікація вимог до ПЗ, придатна для автоматичного опрацювання системою для прогнозування рівня якості програмного забезпечення на основі вимог, повинна містити такі 138 атрибутів якості (атрибути можуть бути наявні у специфікації вимог тільки, якщо після назви атрибуту в специфікації наявне значення атрибуту): 1) кількість функцій (number of functions); 2) повнота функціональної реалізації (functional implementation completeness); 3) функціональна адекватність (functional adequacy); 4) покриття функціональної реалізації (functional implementation coverage); 5) час роботи (operation time); 6) кількість неточних обчислень, з якими стикаються користувачі (number of inaccurate computations encountered by users); 7) кількість елементів даних (number of data items); 8) точність обчислень (computational accuracy); 9) точність (precision); 10) кількість завдань (number of tasks); 11) час відгуку (response time); 12) кількість оцінок (number of evaluations); 13) час відгуку (turnaround time); 14) час виконання завдання (task time); 15) середня пропускна здатність (mean amount of throughput); 16) кількість відмов (number of failures); 17) кількість помилок, пов'язаних з вводом/виводом (number of IO related errors); 18) час очікування користувачем використання пристрою вводу/виводу (user waiting time of IO device utilization); 19) кількість помилок, пов'язаних з пам'яттю (number of memory related errors); 20) кількість помилок, пов'язаних з передачею даних (number of transmission related error); 21) пропускна здатність каналу передачі даних (transmission capacity); 22) завантаженість вводу/виводу (кількість буферів) (IO utilization (number of buffers)); 23) кількість рядків безпосередньо коду (number of line of code directly); 24) ліміти завантаження вводу/виводу (IO loading limits); 25) максимальне завантаження пам'яті (maximum memory utilization); 26) максимальне завантаження передачі (maximum transmission utilization); 27) середня частота виникнення помилок передачі (mean occurrence of transmission error); 28) кількість одночасних користувачів (number of concurrent users); 29) пропускна здатність зв'язку (communication bandwidth); 30) розмір бази даних (size of database); 31) кількість інструкцій (number of tutorials); 32) кількість елементів даних вводу/виводу (number of IO data items); 33) повнота опису (completeness of description); 34) зрозумілість функцій (function understandability); 35) зрозумілість вводу/виводу (understandable input and output); 36) легкість вивчення функцій (ease of function learning); 37) частота надання допомоги (help frequency); 38) ефективність користувацької документації та/або системи допомоги (effectiveness of the user documentation and/or help system); 39) доступність допомоги (help accessibility); 40) повнота користувацької документації та/або довідкової системи (completeness of user documentation and/or help facility); 41) кількість виправлених помилок (error correction); 42) кількість екранів або форм (number of screens or forms); 43) кількість помилок, зроблених користувачем (number of user errors or changes); 44) кількість спроб налаштування (number of attempts to customize); 45) кількість операцій (number of operations); 46) кількість елементів, які можна перевірити на достовірність даних (number of items which could check for valid data); 47) кількість реалізованих повідомлень (number of messages implemented); 48) кількість елементів інтерфейсу (number of interface elements); 49) фізична доступність (physical accessibility); 50) кількість легко зрозумілих повідомлень (number of easily understood messages); 51) кількість невдало вирішених ситуацій (number of unsuccessfully recovered situation); 52) період часу роботи під час спостереження (operation time period during observation); 53) кількість випадків помилкових дій користувача (number of occurrences of user's human error operation); 54) кількість помилок введення, які користувач успішно виправив (number of input errors which the user successfully corrects); 55) кількість спроб виправити помилки введення (number of attempts to correct input errors); 56) кількість помилкових умов, які користувач успішно виправив (number of error conditions which the user successfully corrects); 57) загальна кількість протестованих помилкових умов (total number of error conditions tested); 58) кількість функцій, реалізованих з толерантністю до помилок користувача (number of functions implemented with user error tolerance); 59) загальна кількість функцій, що вимагають толерантності (total number of functions requiring the tolerance capability); 60) загальна кількість неправильних шаблонів роботи (total number of incorrect operation patterns); 61) кількість графічних елементів інтерфейсу (number of interface graphical elements); 62) ступінь збільшення зручності використання для користувача (degree of increase the pleasure of user); 63) ступінь збільшення задоволення користувача (degree of increase the satisfaction of user); 64) ступінь ергономічної привабливості (degree of ergonomic attractiveness); 65) ступінь використання метафор реального світу (degree of real world metaphors use); 66) ступінь можливості використання програмного забезпечення користувачами з обмеженими можливостями (extent to which software can be used by users with specified disabilities); 67) ефективність роботи користувачів з обмеженими можливостями (effectiveness of work of users with specified disabilities); 68) відсутність ризику для користувачів з обмеженими можливостями (freedom from risk for users with specified disabilities); 69) задоволеність користувачів з обмеженими можливостями (satisfaction of users with specified disabilities); 70) наявність властивостей, що підтримують доступність (presence of properties that support accessibility); 71) кількість збоїв (number of faults); 72) розмір продукту (product size); 73) кількість тестових кейсів (number of test cases); 74) кількість усунутих відмов (number of resolved failures); 75) кількість виправлених збоїв (number of corrected faults); 76) щільність відмов по відношенню до тестових кейсів (failure density against test cases); 77) кількість усунутих несправностей (failure resolution); 78) кількість усунутих збоїв (fault removal); 79) середній час між відмовами (mean time between failures); 80) зрілість тестів (test maturity); 81) оцінена

щільність прихованих помилок (estimated latent fault density); 82) щільність помилок (fault density); 83) загальний час, протягом якого програмне забезпечення перебуває у працездатному стані (total time during which the software is in an up state); 84) кількість спостережуваних несправностей (number of observed breakdowns); 85) загальний час простою (total down time); 86) кількість несправностей (number of breakdowns); 87) кількість несанкціонованих операцій (number of illegal operations); 88) час ремонту (time to repair); 89) час простою (down time); 90) кількість перезапусків (number of restarts); 91) кількість відновлень (number of restoration); 92) відновлюваність (restartability); 93) кількість форматів даних, що розглядаються інструментом (number of data formats regarded by tool); 94) кількість форматів даних для обміну (number of data formats to be exchanged); 95) кількість інтерфейсних протоколів (number of interface protocols); 96) обмінність даних (data exchangeability); 97) кількість випадків пошкодження даних (number of instances of data corruption); 98) кількість типів доступу (number of access types); 99) кількість контрольованих вимог (number of controllability requirements); 100) контрольованість доступу (access controllability); 101) кількість правильно зашифрованих/розшифрованих елементів даних (number of data items correctly encrypted/decrypted); 102) кількість елементів даних, що потребують шифрування/розшифрування (number of data items to be required encryption/decryption); 103) кількість подій, що обробляються з використанням цифрового підпису (number of events processed using digital signature); 104) кількість подій, що потребують властивості неспростування (number of events requiring non-repudiation property); 105) кількість доступів до системи та даних, зареєстрованих у системному журналі (number of accesses to system and data recorded in the system log); 106) кількість фактичних доступів (number of accesses actually occurred); 107) кількість наданих методів автентифікації (number of provided authentication methods); 108) кількість зроблених модифікацій (number of modifications made); 109) кількість змінних (number of variables); 110) кількість модулів (number of modules); 111) функціональна спільність (functional commonality); 112) нефункціональна спільність (nonfunctional commonality); 113) розмаїття варіативності (variability richness); 114) застосовність (applicability); 115) пристосованість (tailorability); 116) заміннність компонентів (component replaceability); 117) час помилки (error time); 118) кількість елементів, що підлягають реєстрації (number of items required to be logged); 119) кількість необхідних діагностичних функцій (number of diagnostic functions required); 120) можливість ведення журналу аудиту (audit trail capability); 121) кількість переглянутих версій (number of revised versions); 122) можливість контролю за змінами (change control capability); 123) кількість несправностей за певний період часу до модифікації (number of troubles within certain period before modification); 124) кількість несправностей за той самий період після модифікації (number of troubles in same period after modification); 125) кількість необхідних вбудованих тестових функцій (number of built in test functions required); 126) кількість тестових залежностей від інших систем (number of test dependencies on other systems); 127) кількість контрольних точок (number of checkpoints); 128) зручність перенесення (porting user friendliness); 129) кількість структур даних (number of data structures); 130) адаптивність структур даних (adaptability of data structures); 131) адаптивність апаратного середовища (hardware environmental adaptability); 132) адаптивність програмного середовища (software environmental adaptability); 133) кількість операційних функцій, завдання яких не були виконані або неадекватні (number of operational functions of which tasks were not completed or adequated); 134) загальна кількість функцій, які були протестовані в різних середовищах (total number of functions which were tested in different environment); 135) кількість операцій з налаштування (number of setup operations); 136) кількість кроків інсталяції (number of installation steps); 137) простота інсталяції (ease of installation); 138) кількість об'єктів (number of entities).

Розроблена структура специфікації вимог до ПЗ, придатної для автоматичного опрацювання системою для прогнозування рівня якості програмного забезпечення на основі вимог, накладає певні обмеження на формулювання вимог, що містять атрибути якості, та призначена для подальшого препроцесінгу специфікаційї вимог до ПЗ.

Метод аналізу вимог до програмного забезпечення на предмет пошуку значень атрибутів якості складається з наступних етапів:

1) препроцесінг специфікації вимог до програмного забезпечення – представлення вимог у вигляді, придатному для автоматичного опрацювання розробленою системою для прогнозування рівня якості програмного забезпечення на основі вимог, згідно із пропонованою вище структурою даних, причому атрибути можуть бути наявні у специфікації вимог тільки, якщо після назви атрибуту в специфікації наявне значення атрибуту;

2) автоматичний аналіз специфікації вимог до ПЗ – пошук кожного атрибуту якості у реальних, підготовлених для опрацювання, вимогах до ПЗ;

3) вибір значень кожного атрибуту якості у реальних, підготовлених для опрацювання, вимогах до ПЗ .

Вхідними даними розробленого методу є специфікація вимог до програмного забезпечення, результуючими даними – значення атрибутів якості, наявних у вимогах до ПЗ.

Етап препроцесінгу у вигляді накладання певних обмежень на формування специфікації вимог до ПЗ шляхом структурування застосовується лише до тих вимог, які містять атрибути якості. Решта вимог до ПЗ можуть бути викладені довільним чином.

Метод аналізу вимог до програмного забезпечення на предмет пошуку значень атрибутів якості забезпечує вибір значень атрибутів якості ПЗ з природомовної специфікації вимог до ПЗ, які далі можуть бути використані для визначення значень характеристик якості ПЗ та для комплексного оцінювання якості ПЗ. Розроблений метод є важливим для повної автоматизації опрацювання вимог та повного усунення суб'єктивного впливу і участі людини у процесах опрацювання інформації та здобуття знань. Метод аналізу вимог до ПЗ на предмет пошуку значень атрибутів якості є теоретичним підгрунтям для розроблення модулю автоматичного аналізу вимог до ПЗ пропонованої авторами системи для прогнозування рівня якості програмного забезпечення на основі вимог.

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямі

1) Наразі існує суперечність між зростаючою складністю програмного забезпечення та відповідальністю, яка на нього покладається, розширенням вимог до якості програмного забезпечення, з одного боку, і недосконалістю моделей, методів та засобів прогнозування якості ПЗ на ранніх етапах життєвого циклу. Відтак, прогнозування рівня якості програмного забезпечення на ранніх етапах життєвого циклу на основі атрибутів якості є актуальною задачею.

2) Розроблена структура специфікації вимог до ПЗ, придатної для автоматичного опрацювання системою для прогнозування рівня якості програмного забезпечення на основі вимог, накладає певні обмеження на формулювання вимог, що містять атрибути якості, та призначена для подальшого препроцесінгу специфікаційї вимог до ПЗ.

3) Метод аналізу вимог до програмного забезпечення на предмет пошуку значень атрибутів якості забезпечує вибір значень атрибутів якості ПЗ з природомовної специфікації вимог до ПЗ, які далі можуть бути використані для визначення значень характеристик якості ПЗ та для комплексного оцінювання якості ПЗ. Розроблений метод є важливим для повної автоматизації опрацювання вимог та повного усунення суб'єктивного впливу і участі людини у процесах опрацювання інформації та здобуття знань. Метод аналізу вимог до ПЗ на предмет пошуку значень атрибутів якості є теоретичним підгрунтям для розроблення модулю автоматичного аналізу вимог до ПЗ пропонованої авторами системи для прогнозування рівня якості програмного забезпечення на основі вимог.

Література

1. Tamura Y., Yamada S. Deep Learning Based on Fine Tuning with Application to the Reliability Assessment of Similar Open Source Software. International Journal of Mathematical, Engineering and Management Sciences. 2023. Vol. 8, no. 4. P. 632–639.

2. Bhandari K., Kumar K., Sangal A. L. Data quality issues in software fault prediction: a systematic literature review. Artificial Intelligence Review. 2022.

3. ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evalua-tion (SQuaRE). System and software quality models. 2011.

4. Assessing the Success of R&D Projects and Innovation Projects through Project Management Life Cycle / M. R. Farokhad et al. // Proceedings of 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Metz, France, 18–21 September 2019. 2019.

5. H. Shane and W. Stéphane, Standish Group 2015 Chaos Report – Q&A with Jennifer Lynch, 2015. URL: http://www.infoq.com/articles/standish-chaos-2015.

6. Recent Catastrophic Accidents: Investigating How Software was Responsible / W. E. Wong et al. // Proceedings of 2010 Fourth International Conference on Secure Software Integration and Reliability Improvement, Singapore, Singapore, 9–11 June 2010. 2010.

7. Hovorushchenko T., Pavlova O., Bodnar M. Development of an intelligent agent for analysis of nonfunctional characteristics in specifications of software requirements. Eastern-European Journal of Enterprise Technologies. 2019. Vol. 1, no. 2. P. 6–17.

8. A systematic literature review of requirements engineering education / M. Daun et al. Requirements Engineering. 2022.

9. ISO 25023:2016. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). Measurement of system and software product quality. 2016.

10. Method for forecasting the level of software quality based on quality attributes / T. Hovorushchenko et al. Journal of Intelligent & Fuzzy Systems. 2022. P. 1–15.

11. Hovorushchenko T., Voichur Y., Medzatyi D. Information technology for prediction of software quality level. Radioelectronic and Computer Systems. 2023. No. 3. P. 238–254.

References

1. Tamura Y. Deep Learning Based on Fine Tuning with Application to the Reliability Assessment of Similar Open Source Software / Yoshinobu Tamura, Shigeru Yamada // International Journal of Mathematical, Engineering and Management Sciences. – 2023. – Vol. 8, no. 4. – P. 632–639.
2. Bhandari K. Data quality issues in software fault prediction: a systematic literature review / Kirti Bhandari, Kuldeep Kumar, Amrit Lal Sangal // Artificial Intelligence Review. – 2022.
3. ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evalua-tion (SQuaRE). System and software quality models. 2011.
4. Assessing the Success of R&D Projects and Innovation Projects through Project Management Life Cycle / Mahboobeh Ramezani Farokhad [et al.] // 2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Metz, France, 18–21 September 2019.
5. H. Shane and W. Stéphane, Standish Group 2015 Chaos Report – Q&A with Jennifer Lynch, 2015. URL: http://www.infoq.com/articles/standish-chaos-2015.
6. Recent Catastrophic Accidents: Investigating How Software was Responsible / W. Eric Wong [et al.] // 2010 Fourth International Conference on Secure Software Integration and Reliability Improvement, Singapore, Singapore, 9–11 June 2010.
7. Hovorushchenko T. Development of an intelligent agent for analysis of nonfunctional characteristics in specifications of software requirements / Tetiana Hovorushchenko, Olga Pavlova, Mykyta Bodnar // Eastern-European Journal of Enterprise Technologies. – 2019. – Vol. 1, no. 2. – P. 6–17.
8. A systematic literature review of requirements engineering education [Electronic resource] / Marian Daun [et al.] // Requirements Engineering. – 2022.
9. ISO 25023:2016. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). Measurement of system and software product quality. 2016.
10. Method for forecasting the level of software quality based on quality attributes / Tetiana Hovorushchenko [et al.] // Journal of Intelligent & Fuzzy Systems. – 2022. – P. 1–15.
11. Hovorushchenko T. Information technology for prediction of software quality level / Tetiana Hovorushchenko, Yurii Voichur, Dmytro Medzatyi // Radioelectronic and Computer Systems. – 2023. – No. 3. – P. 238–254.