УДК 004.9: 004.05
DOI https://doi.org/10.32838/2663-5941/2022.2/12

Говорущенко Т.О.
Хмельницький національний університет

Лопатто І.Ю.
Хмельницький національний університет

Капустян М.В.
Хмельницький національний університет

МУЛЬТИАГЕНТНА СИСТЕМА ВРАХУВАННЯ ІНФОРМАЦІЇ ПРЕДМЕТНОЇ ГАЛУЗІ НА ВСІХ ЕТАПАХ РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Невід'ємною частиною процесу розробки програмного забезпечення (ПЗ) є забезпечення його якості. Для підвищення якості програмного забезпечення необхідним є максимально можливе врахування інформації предметної галузі на всіх етапах його розроблення.

Інформацію предметної галузі слід враховувати на всіх етапах розроблення програмного забезпечення, щоб підвищити його якість, функціональну безпеку, надійність, гарантоздатність та живучість. Перевірка врахованості інформації предметної галузі на всіх етапах розроблення ПЗ є наразі актуальною задачею. Для розв'язання такої задачі слід розробити мультиагентну систему врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення, яка складається з двох агентів.

Перший інтелектуальний агент на основі онтологій виконує семантичний парсинг реальних документів на предмет пошуку елементів інформації предметної галузі, які повинні міститись в тому чи іншому документі. Всі знайдені агентом інформаційні елементи заносяться у відповідну реальну онтологію та буде проводитись підрахунок кількості наявних інформаційних елементів. Другий інтелектуальний агент на основі онтологій виконує, власне, порівняння онтологій для кожного документу, формування множини відсутніх елементів інформації предметної галузі в кожному документі та оцінювання величини втрат інформації предметної галузі для кожного документу.

У статті запропоновано структуру мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення, яка: виконує семантичний парсинг реальних документів на предмет пошуку необхідних елементів інформації предметної галузі; формує відповідні реальні онтології для кожного документу; підраховує кількості наявних інформаційних елементів в кожному документі; виконує порівняння ідеальних та реальних онтологій; формує множини відсутніх у відповідних документах інформаційних елементів; оцінює величини втрат інформації предметної галузі для кожного документу.

Ключові слова: *якість програмного забезпечення (ПЗ), інформація предметної галузі, мультиагентна система, інтелектуальний агент на базі онтологій.*

Постановка проблеми. Програмне забезпечення (ПЗ) проникає у всі аспекти нашого життя в епоху інформації: приватні особи та компанії покладаються на операційні системи та програми для виконання щоденних завдань. Саме тому витрати на корпоративне програмне забезпечення мають найвищі темпи зростання в технічній галузі. Ринок програмного забезпечення для підприємств є найбільш швидкозростаючим сектором в ІТ-галузі. Згідно із інформацією компанії Statista, наразі витрати на корпоративне програмне забезпечення становлять 491 мільярдів доларів США, розмір ринку видавництва програмного забезпечення в США становить 285 мільярдів доларів США [1].

Як двигун цифрової трансформації, програмне забезпечення буде відігравати дедалі більшу роль у наступні роки після економічного спаду, спричиненого пандемією COVID-19, коли витрати на корпоративне програмне забезпечення будуть наближатись до 500 мільярдів доларів США.

При цьому велика кількість програмних проектів не відповідають початковим цілям та бізнес-намірам і не виконуються в рамках бюджету у встановлені часові рамки [2]. Організації втрачають в середньому 97 мільйонів доларів на

кожен вкладений 1 мільярд доларів через низьку ефективність програмних проектів [3].

Отже, невід'ємною частиною процесу розробки програмного забезпечення є забезпечення його якості. Якість ПЗ – це ступінь, з яким програмний продукт відповідає встановленим вимогам при його застосуванні у визначених умовах [4]. Враховуючи вищевикладене, *актуальною задачею* наразі є підвищення якості програмного забезпечення.

Аналіз останніх досліджень і публікацій. Відповідно до звіту Consortium for Information & Software Quality (CISQ) [5], світова індустрія інформаційних технологій наблизилася до досягнення обсягу в 5 трлн. дол. США в 2020 році, з яких 32% від загальної суми припадає на США. Це приблизно 1,6 трлн. дол. США в 2020 році. 25% від останньої суми грошей було витрачено на проекти розвитку, що становить 400 млрд. дол. США, і 65% з них були витрачені на невдалі і проблемні проекти. Отже, певною мірою ціна низької якості самого програмного забезпечення та процесів його розробки (Cost of Poor Software Quality, CPSQ) через невдалі проекти в США у 2020 році склала 260 млрд дол. США (у порівнянні з 177,5 млрд дол. США у 2018 році).

Недостатня якість ПЗ є причиною багатьох катастроф, аварій, фінансових втрат і витоків інформації [6, 7], а дефекти, наприклад, у вимогах до ПЗ [8, 9], програмному коді [10-12] або інтерфейсах користувача [13-15] впливають на якість ПЗ в цілому.

Відповідно до опублікованого звіту [16], який підготувала компанія УНДО (Undo) – розробник платформи для відтворення збоїв ПЗ спільно з Кембриджською бізнес-школою (Cambridge Business School) в 2020 році (рис. 3), компанії-розробники ПЗ 26% (620 млн годин на рік) від усього часу, що виділяється на розробку ПЗ, витрачають на знаходження і виправлення дефектів ПЗ, що обходиться приблизно в 61 млрд дол. США. Це еквівалентно обсягу акцій компаній на суму 1,2 трлн дол. США.

Причини недостатньої якості ПЗ можуть бути досить різноманітними. До них можна віднести, наприклад, дефекти розробки вимог, проектування, програмування, тестування тощо. Складність, а, відповідно, і вартість перебування і виправлення дефектів ПЗ залежить від етапу розробки ПЗ. Чим пізніше виявлений дефект, тим складніше і дорожче його виправити [17].

Однією з причин низької якості ПЗ може бути недостатність уваги до інформації предметної галузі на різних етапах життєвого циклу. Розробники ПЗ можуть прискіпливо аналізувати та ретельно враховувати деяку інформацію предметної галузі, а от іншу інформацію взагалі залишати поза своєю увагою, причому навіть інформація з високою ймовірністю може залишатись поза увагою розробників.

Постановка завдання. Інформацію предметної галузі слід враховувати на всіх етапах розроблення програмного забезпечення, щоб підвищити його якість, функціональну безпеку, надійність, гарантоздатність та живучість. Відтак, перевірка врахованості інформації предметної галузі на всіх етапах розроблення ПЗ є наразі актуальною задачею. Для розв’язання такої задачі слід розробити інтелектуальну мультиагентну систему для підвищення якості програмного забезпечення шляхом врахування інформації предметної галузі на всіх етапах його розроблення, яка якраз і буде виконувати перевірку врахованості інформації предметної галузі на всіх етапах розроблення ПЗ. Метою представленого дослідження є розроблення структури мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення.

Виклад основного матеріалу дослідження. Інтелектуальна мультиагентна система на основі онтології для оцінки специфікацій вимог до програмного забезпечення була розроблена авторами у [18]. Вона передбачає порівняння ідеальної онтології (містить всю необхідну інформацію якості у специфікації вимог) та реальної онтології (містить всю наявну інформацію якості у реальній специфікації вимог), що уможливлює ідентифікацію втрат інформації у специфікації вимог до ПЗ.

Запропонований у [18] підхід використаємо тепер для розроблення мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення.

Кожен етап розроблення ПЗ виконується на основі певного документу (план, специфікація вимог до ПЗ, дизайн ПЗ, сирцевий код, продукт, тощо), в якому повинна відображатись певна інформація предметної галузі, яка може і повинна бути врахована у відповідному документі (як правило, її стандарти визначають). На основі стандартних форм таких документів будуються ідеальні онтології для кожного з документів, які відображають необхідні елементи інформації предметної галузі, які згідно із стандартами та іншою нормативною документацією повинні бути присутні у тому чи іншому документі.

На основі реальних документів, доступних під час розроблення ПЗ, для кожного окремого ПЗ будуть будуватись реальні онтології. Для цього необхідним є перший інтелектуальний агент на основі онтологій, який виконуватиме семантичний парсинг реальних документів на предмет пошуку елементів інформації предметної галузі, які повинні міститись в тому чи іншому документі. Всі знайдені агентом інформаційні елементи будуть заноситись у відповідну реальну онтологію та буде проводитись підрахунок кількості наявних інформаційних елементів.

Розроблені ідеальні та реальні онтології порівнюються, на основі чого формуються множини відсутніх у відповідних документах інформаційних елементів, які, власне, і демонструють втрати інформації предметної галузі (та їх величину) у певному документі.

Для оцінювання величини втрат інформації предметної галузі можна адаптувати запропоновані авторами раніше [18] формули числової оцінки рівня достатності наявної інформації.

Другий інтелектуальний агент на основі онтологій буде виконувати, власне, порівняння онтологій для кожного документу, формування множини відсутніх елементів інформації предметної галузі в кожному документі та оцінювання величини втрат інформації предметної галузі для кожного документу.

Візуалізоване представлення множин відсутніх інформаційних елементів в кожному документі допоможе розробнику зрозуміти прогалини інформації в документі і, таким чином, забезпечить підвищення якості наповнення того чи іншого документу.

На вхід мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення можуть бути подані як всі документи всіх етапів розроблення ПЗ одночасно, так і будь-який один (готовий на даний момент часу) документ, тобто система застосовна і ефективна на будь-якому етапі розроблення ПЗ (в тому числі і на найбільш ранніх етапах, коли доступний лише план або специфікація).

Тоді структура мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення представлена на рис. 1.

В контексті вищевикладеної концепції та структури інтелектуальної мультиагентної системи для підвищення якості програмного забезпечення шляхом врахування інформації предметної галузі на всіх етапах його розроблення, перспективною задачею наразі є наповнення ідеальних онтологій для всіх документів на всіх етапах розроблення програмного забезпечення. Для цього потрібно проаналізувати відповідні стандарти, нормативні документи, рекомендації, звідки слід вибрати ті елементи інформації предметної галузі, які повинні бути присутні у тому чи іншому документі. Наразі такі онтології реалізовані лише для специфікації вимог до ПЗ [18]. Наповнення і розробка інших ідеальних онтологій будуть метою іншої, перспективної, частини нашого дослідження.

Для прикладу припустимо, що план повинен містити 120 різних елементів інформації предметної галузі згідно із нормативною документацією, дизайн ПЗ повинен містити 150 різних інформаційних елементів, сирцевий код повинен містити 230 різних елементів інформації предметної галузі, продукт повинен містити 250 різних інформаційних елементів. Оскільки для специфікації вимог до ПЗ вже розроблено ідеальну онтологію, то вже відомо, що специфікація вимог до ПЗ повинна містити 138 різних елементів інформації (атрибутів). Наприклад, в плані наявні 80 елементів інформації з необхідних 120 елементів, в специфікації наявні 98 елементів з необхідних 138 елементів, в дизайні наявні 140 елементів з необхідних 150 елементів, в коді наявні 210 елементів з необхідних 230 елементів, в продукті наявні 130 елементів з необхідних 250 елементів, то мають місце наступні втрати інформації предметної галузі у кожному документі (з врахуванням запропонованих авторами у [18] формули числової оцінки рівня достатності наявної інформації): 33% у плані, 29% у специфікації вимог, 7% в дизайні, 9% в коді, 48% в продукті.

Враховуючи вище доведений факт, що для підвищення якості програмного забезпечення необхідним є максимально можливе врахування інформації предметної галузі на всіх етапах його розроблення, слід прагнути до мінімізації інформаційних втрат – найкращим результатом буде досягнення рівня інформаційних втрат 0-10%. Для наведеного прикладу інформаційні втрати у сирцевому коді та у дизайні можна вважати незначними і в цілому припустимими, а от над планом, специфікацією та продуктом розробникам слід працювати з метою мінімізації інформаційних втрат.

Висновки. Перевірка врахованості інформації предметної галузі на всіх етапах розроблення ПЗ є наразі актуальною задачею. Для розв'язання такої задачі слід розробити мультиагентну систему врахування інформації предметної галузі

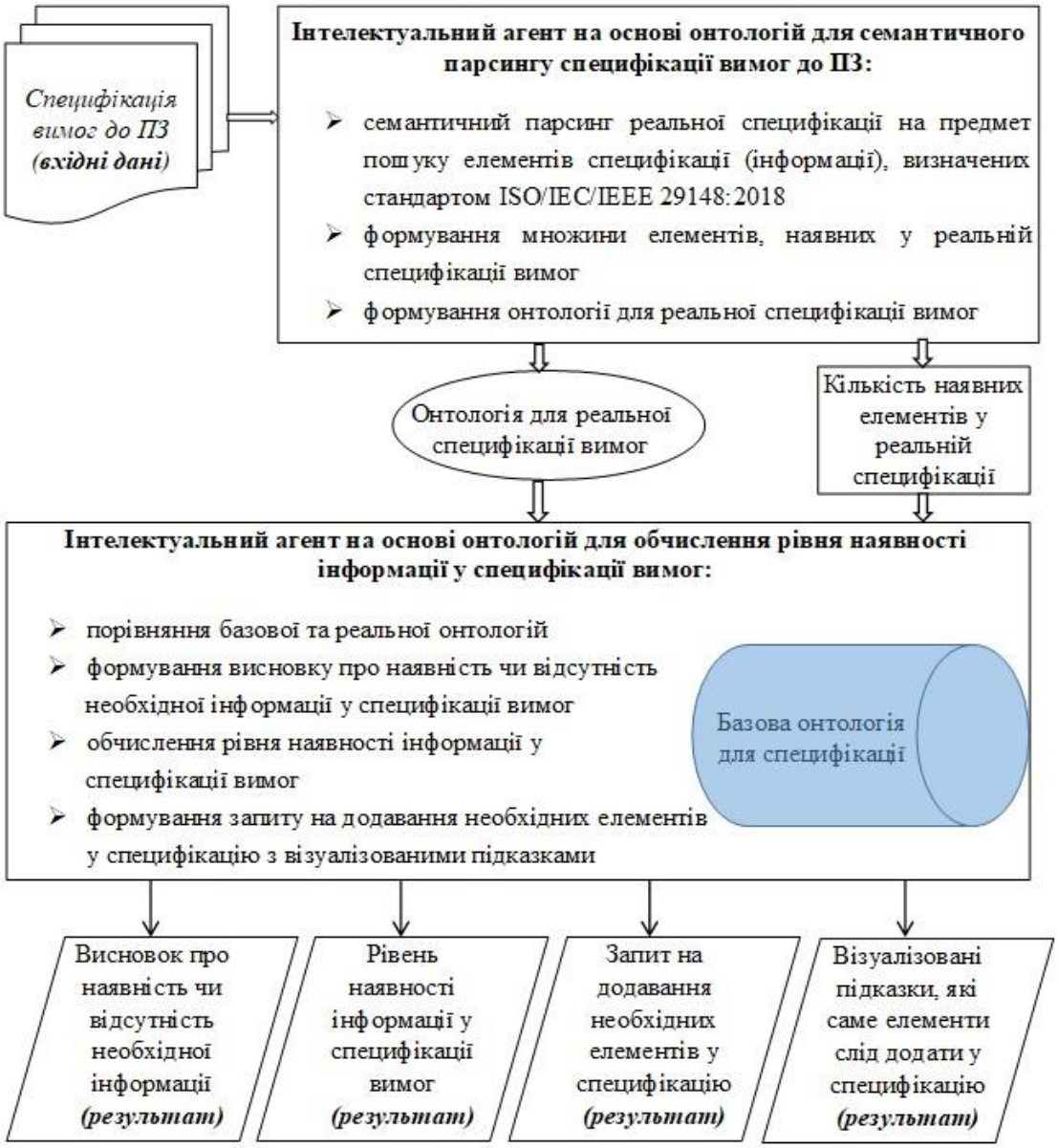

Рис. 1. Структура мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення

на всіх етапах розроблення програмного забезпечення, яка складається з двох інтелектуальних агентів. Перший інтелектуальний агент на основі онтологій виконує семантичний парсинг реальних документів на предмет пошуку елементів інформації предметної галузі, які повинні міститись в тому чи іншому документі. Всі знайдені агентом інформаційні елементи заносяться у відповідну реальну онтологію та буде проводитись підрахунок кількості наявних інформаційних елементів. Другий інтелектуальний агент на основі онтологій виконує, власне, порівняння онтологій для кожного документу, формування множини відсутніх елементів інформації предметної галузі в кожному документі та оцінювання величини втрат інформації предметної галузі для кожного документу.

У статті запропоновано структуру мультиагентної системи врахування інформації предметної галузі на всіх етапах розроблення програмного забезпечення, яка: виконує семантичний парсинг реальних документів на предмет пошуку необхідних елементів інформації предметної галузі; формує відповідні реальні онтології для кожного документу; підраховує кількості наявних інформаційних елементів в кожному документі; виконує порівняння ідеальних та реальних онтологій; формує множини відсутніх у відповідних документах інформаційних елементів; оцінює величини втрат інформації предметної галузі для кожного документу.

Список літератури:

1. Software. Web-site. URL: https://www.statista.com/markets/418/topic/484/software/ (Last accessed: February 20, 2022).
2. Farokhad M., Otegi-Olaso J., Pinilla L., Gandarias N., De Lacalle L. Assessing the Success of R&D Projects and Innovation Projects through Project Management Life Cycle. *2019 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*: Proceedings (Metz, France, September 2019). Metz (France), 2019. Pp. 1104-1110.
3. PMI's Pulse of the Profession 9-th Global Project Management Survey. Web-site. URL: https://www.pmi.org/-/media/pmi/documents/public/pdf/learning/thought-leadership/pulse/pulse-of-the-profession-2017.pdf (Last accessed: February 20, 2022).
4. ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models. [Introduced 01.03.2011]. Geneva (Switzerland), 2011. 34 p. (International standard).
5. The Cost of Poor Software Quality in the US: A 2020 Report. Web-site. URL: https://www.it-cisq.org/pdf/CPSQ-2020-report.pdf (Last accessed: February 20, 2022).
6. Wong W. E., Debroy V., Surampudi A., Kim H., Siok M. F., Recent Catastrophic Accidents: Investigating How Software was Responsible. *Fourth International Conference on Secure Software Integration and Reliability Improvement*: Proceedings (Singapore, June 9-11, 2010), Singapore, 2010. Pp. 14-22.
7. Wong W., Debroy V., Restrepo A. The Role of Software in Recent Catastrophic Accidents: IEEE Reliability Society 2009 Annual Technology Report. 2009. 8 p.
8. Hu W., Carver J. C., Anu V. K, Walia G. S., Bradshaw G. Detection of Requirement Errors and Faults via a Human Error Taxonomy: A Feasibility Study. *The 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*: Proceedings (Ciudad Real, Spain, September 8-9, 2016), Ciudad Real, 2016. Pp. 1–10.
9. Dinkar S. Requirement Engineering Errors: Errors and Ambiguities of Visualization. *International Journal of Computer Applications*. 2014. Vol. 92. Pp. 19-23.
10. Huckle T., Neckel T. Bits and Bugs: A Scientific and Historical Review of Software Failures in Computational Science. Society for Industrial & Applied Mathematics, 2019. 251 p.
11. Hobbs C. Embedded Software Development for Safety-Critical Systems: Second Edition. CRC Press, Taylor & Francis Group, 2016. 384 p.
12. Wang S., Lo D. Version history, similar report, and structure: putting them together for improved bug localization. *The 22nd International Conference on Program Comprehension (ICPC 2014)*: Proceedings (USA, 2014). USA, 2014. Pp. 53–63.
13. Zhang Yi, Masci P., Jones P., Thimbleby H. User Interface Software Errors in Medical Devices: Study of U.S. Recall Data. *Biomedical Instrumentation & Technology,* 2019. Vol. 53 (3). Pp. 182-194.
14. Natella R., Winter S., Cotroneo D., Suri N. Analyzing the Effects of Bugs on Software Interfaces. *IEEE Transactions on Software Engineering*. 2020. Vol. 46. No. 3. Pp. 280-301.
15. Faily S., Lyle J., Fléchais I., Simpson A. Usability and Security by Design: A Case Study in Research and Development. *NDSS Workshop on Usable Security*: Proceedings (San Diego, CA, USA, February 8-11, 2015). San Diego, 2015. 10 p.
16. What is the actual cost of software failures? Web-site. URL: https://undo.io/the-cost-of-software-failures (Last accessed: February 20, 2022).
17. Pomorova O., Hovorushchenko T. The Way to Detection of Software Emergent Properties. *The 8-th IEEE International Conference on IDAACS*: Proceedings (Warsaw (Poland), September 24-26, 2015). Warsaw, 2015. Vol. 2. Pp. 779-784.
18. Hovorushchenko T., Pavlova O. Evaluating the Software Requirements Specifications Using Ontology-Based Intelligent Agent. *IEEE International Scientific and Technical Conference "Computer Science and Information Technologies":* Proceedings (Lviv (Ukraine), September, 2018). Lviv, 2018. Pp.215-218.

Lopatto I.Yu., Hovorushchenko T.O., Kapustian M.V. MULTIAGENT SYSTEM OF ACCOUNTING SUBJECT AREA INFORMATION AT ALL STAGES OF SOFTWARE DEVELOPMENT

Ensuring software quality is an integral part of the software development process. For improving the quality of software, it is necessary to take into account the information of the subject area at all stages of its development.

Subject area information should be considered at all stages of software development to improve its quality, functionality, reliability, dependability, and survivability. Checking the consideration of information in the subject area at all stages of software development is currently an urgent task. To solve this problem, we need

to develop a multi-agent system for accounting for information in the subject area at all stages of software development, which consists of two agents.

The first ontology-based intelligent agent performs semantic parsing of real documents in order to search for information elements of the subject area that should be contained in a document. All information elements found by the agent are entered into the appropriate real ontology and the number of available information elements will be counted. The second intelligent agent on the basis of ontologies, in fact, compares ontologies for each document, forming a set of missing elements of information of the subject area in each document and estimates the amount of information loss of subject area for each document.

The article proposes the structure of a multi-agent system of accounting for subject area information at all stages of software development, which: performs semantic parsing of real documents in order to find the necessary elements of subject area information; forms appropriate real ontologies for each document; counts the amount of available information elements in each document; compares ideal and real ontologies; forms sets of information elements missing in the relevant documents; estimates the amount of information loss of the subject area for each document.

Key words: *software quality, subject area information, multi-agent system, ontology-based intelligent agent.*