https://orcid.org/0000-0002-9630-4951

УДК 510.635:004.891(045)

А.І. ВАВІЛЕНКОВА[*]

КОМАНДНА РОЗРОБКА ПРОГРАМНИХ ПРОДУКТІВ В ОСВІТІ

[*]Національний авіаційний університет, м. Київ, Україна

Анотація. *У статті окреслено проблему відсутності верифікації отриманих студентами під час навчання теоретичних основ у реальних умовах, зокрема, проблему практичної реалізації командних проєктів у сфері розробки програмного забезпечення та управління проєктами у закладах освіти, хоча в сучасних IT-компаніях така практика береться за основу роботи. Проаналізовано популярні програмні продукти, що використовуються сьогодні для автоматизованого планування та контролю командної роботи підприємствами, це Basecamp, Jira, Asana, Trello. Зазначено, що основною задачею для освітніх закладів є відтворення умов роботи команди на підприємствах та занурення студентів у максимально доступний для сприйняття процес програмного створення командного проєкту з метою своєчасної розробки якісного програмного продукту. З цією метою для реалізації навчального командного проєкту обрано методологію Microsoft Solution Framework. Метою даної статті є формалізація процесу планування та реалізації спринту у вигляді алгоритму дій, що повинні здійснюватися та програмно фіксуватися у створеному навчальному командному проєкті. Узагальнено схему взаємодії робочих елементів Scrum-команди у програмному середовищі Visual Studio. Уніфіковано алгоритм роботи членів командного проєкту для реалізації всіх можливостей, запропонованих Microsoft на основі роботи Team Foundation Server. Описано алгоритм етапу реалізації спринту згідно із гнучкою методологією Scrum. Він містить детальний опис робіт, що виконуються на кожному спринті, тобто етапі роботи команди, і передбачає безпосереднє виконання задач кожним членом команди. Для висвітлення результатів роботи проводять щоденні Daily Meeting, у кінці виконання спринту отримується інкремент програмного продукту.*

Ключові слова: *командний проєкт, гнучка методологія, управління проєктами, Scrum-команда, Visual Studio, спринт, інкремент.*

Abstract. *The article outlines the problem of the lack of verification obtained by students during the study of theoretical foundations in real conditions, in particular the problem of practical implementation of team projects in the field of software development and project management in educational institutions, although in modern IT companies such practice is taken as a basis of their work. The following popular software products used for automated planning and teamwork control in enterprises are analyzed: Basecamp, Jira, Asana, Trello. It is noted that the main task of educational institutions is to recreate for the team the working conditions in companies and immerse students in the most accessible process of creating a team project in order to develop a quality software product. For this purpose the Microsoft Solution Framework methodology was chosen in order to implement the educational team project. The aim of this article is to formalize the process of planning and implementation of a sprint in the form of an algorithm of actions that must be carried out and programmatically recorded in the created educational team project. A scheme of Scrum-team working elements interaction in the Visual Studio software environment is generalized. An algorithm of work of team project members is unified with the aim of realization of all opportunities offered by Microsoft on the basis of Team Foundation Server work. The article describes an algorithm of the sprint implementation stage according to the flexible Scrum methodology. It contains a detailed description of the work performed on each sprint, i.e. every stage of the teamwork, and provides direct performance of tasks by each team member. To cover the results of the work, a Daily Meeting is held, and at the end of the sprint an increment of the software product is obtained.*

Keywords: *team project, flexible methodology, project management, Scrum-team, Visual Studio, sprint, increment.*

DOI: 10.34121/1028-9763-2021-2-74-80

1. Вступ

Дуже важливим аспектом для сфери освіти в умовах стрімкого розвитку інформаційних технологій є «іти в ногу» з вимогами, які диктує для ІТ-сфери ринок праці, та готувати сьогоднішніх студентів до сучасних реалій роботи на підприємствах. Суттєвим недоліком у сфері сучасної освіти є відсутність верифікації отриманих студентами під час навчання теоретичних основ у реальних умовах.

Сьогодні одним із трендових понять у сфері розробки програмного забезпечення та управління проєктами є «командна робота», прикладів якої бракує у закладах освіти. Не зважаючи на це, у сфері інформаційних технологій продовжують виходити на ринок нові програмні продукти, які базуються на сучасних концепціях побудови гнучких технологій розробки програмного забезпечення та впевнено впроваджуються у життя сучасних підприємств та ІТ-компаній.

Так, для контролю версій програмного забезпечення, тобто системи, що записує зміни у файл або набір файлів протягом певного часу та дозволяє пізніше повернутися до певної версії, розроблена технологія Git [1–3], що базується на регулярному створенні посилань на поточний стан проєкту, який зберігається у певний момент часу, й забезпечує можливість повернення до необхідного комміту. Однак перед створенням певної поточної версії програмне забезпечення проходить необхідні етапи розробки згідно з обраною моделлю життєвого циклу та методологією розробки програмного забезпечення.

Однією з найбільш популярних сьогодні концепцій створення командних проєктів для розробки програмних продуктів є Agile [4–5], розгалуженням якої є гнучка методологія Scrum, в основі якої лежать чотири типи подій: планування спринту (Sprint Planning), щоденні зібрання (Daily Scrum), демонстрація версії програмного продукту, створеної на поточному спринті (Sprint Review), та обговорення отриманих на спринті результатів із метою уникнення помилок на наступних етапах, тобто ретроспективи спринту (Sprint Retrospective) [6–7]. Ще одним розгалуженням концепції Agile є гнучка методологія Kanban, згідно з якою проєкт ділиться на етапи, які візуалізуються у вигляді Kanban-дошки, а задачі у вигляді карток переміщаються з етапу на етап, при цьому задача знімається з дошки не тоді, коли час її виконання сплив, а тоді, коли задача завершена [8–9].

Застосування сучасних методологій розробки програмного забезпечення з коректним теоретичним обґрунтуванням дасть змогу створити базис для впровадження практичної реалізації командних проєктів в освітніх закладах.

2. Постановка задачі

Для автоматизованого планування та контролю командної роботи підприємствами використовується досить багато програмних продуктів, наприклад:

– Basecamp – дає можливість відслідковувати задачі та час їх вирішення, завантажувати файли, має просту систему управління проєктом, дає можливість планувати строки у вигляді графіків [10], але досить нестійка та незрозуміла для роботи з метою навчання побудови командних проєктів;

– Jira – працює на основі гнучких методологій Kanban та Scrum, дає змогу планувати Backlog, відслідковувати та виконувати задачі, створювати релізи та звіти, проте процес командної роботи важко сприймається через велику кількість налаштувань та величезну кількість функцій [11];

– Asana – дає змогу створювати, призначати та управляти задачами, дозволяє помістити кожну задачу у власний контекст, призначений для застосування у невеликих проєктах, містить зручні інструменти для спілкування членів команди, організації робочого процесу [12];

– Trello – дозволяє організувати робочий процес для зустрічей та командних проєктів, в основі своєї роботи використовує дошку з колонками та картками (за методологією Kanban), завдяки яким можна відслідковувати продуктивність, структурувати роботу [13].

Як видно з опису популярних програмних систем планування командної роботи, кожна з них містить певні недоліки, а для побудови навчальних командних проєктів важливі простота дій, візуалізація процесів, доступність для розуміння та використання сучасних гнучких методологій розробки програмного забезпечення. Однією з важливих задач, що стоять у наш час перед освітніми закладами, – відтворити умови роботи команди на підприємствах та занурити студентів у максимально доступний для сприйняття процес програмного створення командного проєкту з метою своєчасної розробки якісного програмного продукту.

Однією з методологій розробки програмного забезпечення, що реалізує управління життєвим циклом програмного забезпечення, описує принципи управління людьми та робочими процесами при розробці програмних рішень, є методологія MSF (Microsoft Solution Framework). MSF являється частиною програмного інструментарію VSTS (Visual Studio Team System), яка дозволяє створювати командні проєкти на основі шаблонів Scrum та Agile, реалізує візуальну складову роботи за допомогою інструментів Visual Studio на базі Team Foundation Server та його оновлених версій. На базі MSF можна створювати програмні системи різного призначення, дотримуючись різних підходів до управління процесами розробки програмного забезпечення [14].

Для реалізації навчальних командних проєктів у програмному середовищі Visual Studio пропонується використовувати шаблон Microsoft Visual Studio Scrum 2.2, який базується на поняттях та принципах роботи гнучкої методології Scrum. Проте ретельне дослідження процесу створення командних проєктів на базі цього шаблону показало, що термінологія та етапи роботи команди цілком зрозумілі для створення командних проєктів студентами, проте відсутня послідовність дій, яку повинна виконувати команда на кожному із спринтів. Також досить не зрозумілою є реалізація процесу планування спринту.

Тому метою даної статті є формалізація процесу планування та реалізації спринту у вигляді алгоритму дій, що повинні здійснюватися та програмно фіксуватися у створеному навчальному командному проєкті.

3. Процес реалізації роботи Scrum-команди

Робота у Scrum-команді, а відповідно і робота у командному проєкті, що створюється на основі цього шалону, передбачає дотримання семи основних принципів [15]:

– цілісності – Scrum-команда вважається самодостатньою, зовнішньою системою, до якої входять власник продукту Product Owner, керівник проєкту Scrum-master та команда розробників;

– сумісності елементів у системі, що означає постійну взаємодію трьох складових Scrum-команди та прийняття спільних рішень;

– організованості зв’язків – члени Scrum-команди мають строгу ієрархію та відповідають за виділені під них задачі;

– емерджентності – що означає гнучкість до адаптації на ринку та швидкість реакції на проблеми;

– цілеспрямованості – мета Scrum-команди – створити робочий та конкурентоспроможний продукт у заплановані строки;

– актуалізації або синергетичного ефекту – дає можливість досягти глобальних цілей завдяки можливості кожного з елементів системи оперативно вносити зміни до програмного продукту;

– комплексності – реалізується у необхідності взаємодії всіх трьох складових Scrum-команди.

Саме завдяки реалізації цих принципів та їх інтерпретації у програмному середовищі Visual Studio можна сформувати узагальнену схему взаємодії робочих елементів Scrum-команди (рис. 1).

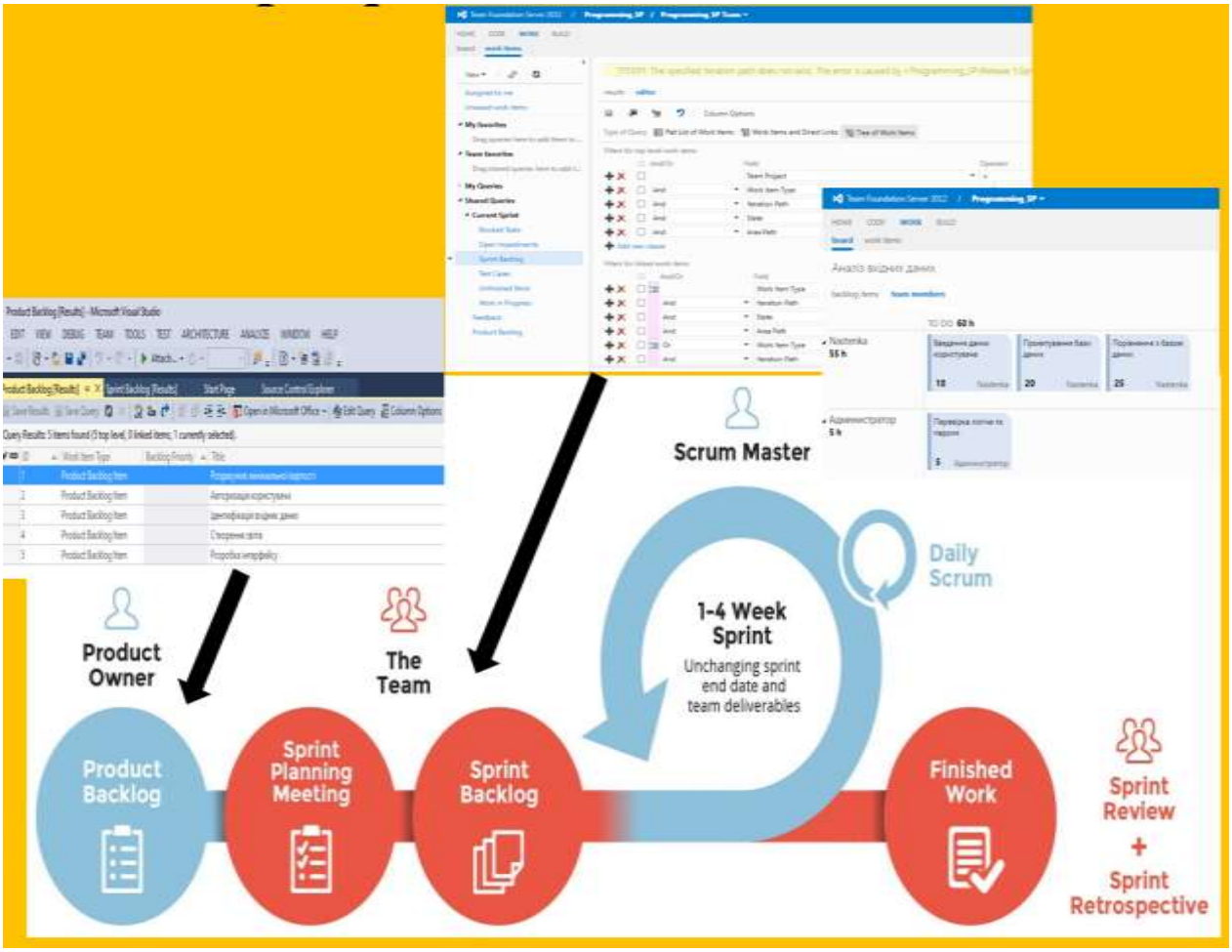

Рисунок 1 – Узагальнена схема роботи Scrum-команди

1. Формування власником продукту списку задач (Product Backlog) на основі вимог клієнта. Для створення Product Backlog у Visual Studio потрібно на стартовій сторінці вибрати «TEAM»-«New Work Item»-«Product Backlog Item», де вказати назву, опис та прив'язку поставленої власником продукту задачі до конкретного спринту. Також задачі Product Backlog можна створити у Microsoft Excel або Microsoft Project із подальшою їх синхронізацією з Visual Studio.

2. Планування спринту – це подія в Scrum, яка означає початок спринту, планується об'єм роботи на спринт, тобто Sprint Backlog – способи виконання поставлених задач. У плануванні спринту бере участь уся команда. Цей етап передбачає призначення пріоритету для кожної задачі «WORK»-«Work items»-«Product Backlog», а для задач із найвищим пріоритетом, тобто таких, що потрапляють до поточного спринту, визначити додаткові завдання Tasks та розподілити їх між членами командного проєкту у полі «Assigned To».

3. Реалізація спринту – передбачає безпосереднє виконання задач кожним членом команди та проведення щоденних Daily Meeting для висвітлення результатів роботи. Результатом виконання задачі із Sprint Backlog певним членом команди буде зміна статусу задачі, тобто значення поля «State».

4. Огляд спринту – проводиться в кінці спринту для перевірки інкременту, тобто суми всіх виконаних вимог із Sprint Backlog впродовж поточного спринту та усіх попередніх. Результатом огляду є виправлений журнал Product Backlog, з якого визначатимуться елементи для Sprint Backlog наступного спринту. Після зміни статусу виконаних задач членами команди на дошці «Board»-«Backlog Item» задачі будуть розміщені у відповідних колонках, а результати виконання спринту можна переглянути на діаграмі згоряння.

5. Ретроспектива спринту – дає можливість перевірити, наскільки успішно завершено спринт, і скласти новий план покращень роботи у команді. На цьому етапі відбувається формування Sprint Backlog для наступного спринту. Для цього новим не виконаним задачам призначаються додаткові завдання, які розподіляються між членами команди.

Як видно з наведених прикладів, кожен етап, який відображено на узагальненій схемі роботи Scrum-команди, перегукується з послідовністю дій щодо створення та реалізації командного проєкту в Visual Studio, що дає можливість наглядно простежити та безпосередньо брати участь у процесі створення й виконання задач для створення власного програмного продукту.

4. Алгоритм етапу реалізації спринту

Особливої уваги у процесі створення програмного продукту за методологією Scrum набуває етап реалізації спринту, адже передбачає виконання великої кількості завдань та потребує деталізації [16].

На рис. 2 зображено схему алгоритму процесу реалізації певного поточного спринту k із N запланованих.

1. Вхідними даними для спринту є список умов Definition of Ready, за яким буде визначатися, чи достатньо проаналізовані задачі та чи наявні всі необхідні артефакти для прийняття задачі в роботу; сформульована мета спринту щодо виконання тестів, щоб досягти поставленої цілі; а також розроблений попередньо Sprint Backlog.

2. Спринт k розпочинається оцінюванням задач спринту, кожній із них проставляється пріоритет.

3. Після цього відбувається етап декомпозиції, на якому задачі розділяють на завдання або підзадачі та признаються членам команди, в результаті чого відбувається деталізація Sprint Backlog.

4. З метою синхронізації роботи членів команди згідно з методологією Scrum проводиться щоденний Daily Meeting, під час якого визначається, хто і що виконав за попередній день, що планується виконати в поточний день та які проблеми заважають розв’язати поставлену задачу. Щоденні зміни стану задач відображаються на діаграмі згоряння (burn down chart).

5. Створення Commits або Code Review – це верифікація коду іншими членами команди.

6. Попередня демонстрація (Deploy) – передбачає демонстрацію розробленого коду з метою подальшого тестування та попереднього узгодження виконаної роботи.

7. Демонстрація інкремента продукту (Review) – відбувається на зустрічі, де розробники демонструють реалізацію мети спринту, тобто демонструють новий працездатний інкремент продукту. Таким чином, робота команди на поточному спринті закінчена, а стейкхолдери вирішують, чи буде реалізовано наступний спринт.

8. Надходження відгуків (feedback) – відбувається отримання зворотного зв’язку на якість роботи команди.

9. Ретроспектива спринту k – передбачає зібрання команди розробників для обговорення проблем, що виникали у команди протягом k-го спринту з метою покращення роботи команди на наступних етапах.

10. Кінець спринту k – передавання нового інкременту на моніторинг та фіксація поточної версії програмного продукту.

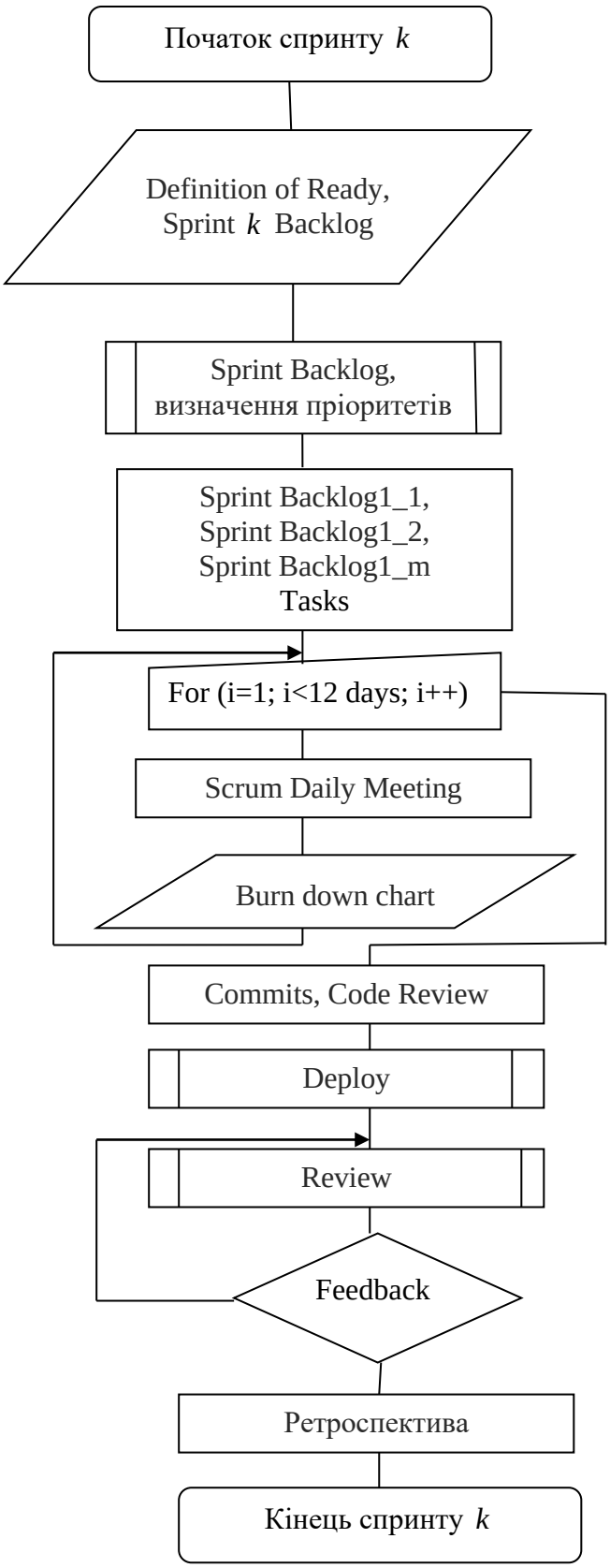

Рисунок 2 – Схема алгоритму процесу реалізації k -го спринту

Після завершення поточного k-го спринту із списку задач Product Backlog виділяються задачі для формування Spring $k+1$ Backlog.

5. Висновки

Формалізоване представлення процесу реалізації k-го спринту дає змогу чітко окреслити послідовність дій для отримання результату виконання одного спринту згідно із гнучкою методологією Scrum – інкремент, попередню версію готового програмного продукту. Кожен з етапів продемонстрованого алгоритму відповідає набору дій для візуалізації робіт спринту у програмному середовищі Visual Studio.

Поетапна структуризація та проведення паралелі між концептами гнучкої методології Scrum та програмними елементами для їх реалізації дають змогу реалізувати реальний командний проєкт, моделюючи цим самим роботу певного підприємства. При створенні власних навчальних проєктів розробки програмного забезпечення у Visual Studio на основі Team Foundation Server студенти опиняються в реальних умовах взаємодії членів Scrum-команди і виступають у ролі адміністраторів, розробників та тестувальників. Реалізація таких проєктів дає неоціненний досвід роботи в команді навіть при дистанційній роботі і висвітлює новий погляд на проведення практичних занять у закладах освіти.

СПИСОК ДЖЕРЕЛ

1. Santacroce F., Olsson A., Voss R., Narebski J. Git: Mastering Version Control. Birmingham: Packt Publishing, 2016. 839 p.
2. Tsitoara M. Beginning Git and GitHub: A Comprehensive Guide to Version Control, Project Management, and Teamwork for the New Developer. New York: Apress, 2020. 289 p. URL: https://doi.org/10.1007/978-1-4842-5313-7.
3. Вавіленкова А., Литвиненко О., Жолдаков О. Управління проєктами інформатизації: навч. посіб. Київ: НАУ, 2015. 220 с.
4. Alaidaros H., Omar M, Romli R. An improved model of Agile Kanban method: verification process through experts' review. *International Journal of Agile Systems and Management*. 2020. Vol. 13, N 4. P. 390–416. URL: https://dx.doi.org/10.1504/IJASM.2020.112337.
5. Putri D., Rozilawati. R., Zulkefli M. Team Formation for Agile Software Development: A Review. *International Journal on Advanced Science, Engineering and Information Technology*. 2020. Vol. 10, N 2. P. 555–561. URL: https://doi.org/10.18517/ijaseit.10.2.10191.
6. Nidagundi P., Novickis L. Introducing Lean Canvas Model Adaptation in the Scrum Software Testing, *Procedia Computer Science*. 2017. Vol. 104. P. 97–103. URL: https://doi.org/10.1016/j.procs.2017.01.078.
7. Morampudi N., Gaurav R. Evaluation Strengths and Weaknesses of Agile Scrum Framework Using Knowledge Management. *International Journal of Computer Applications*. 2013. Vol. 65, N 23. P. 1–6.
8. Андерсен Д.Дж. Канбан. Успішні еволюційні зміни для вашого технологічного бізнесу. Харків: Видавництво Фабула, 2021. 288 с.
9. Андерсен Д.Дж. Канбан. Альтернативный путь в Agile. М.: Видавництво Манн, Иванов и Фербер, 2019. 336 с.
10. Basecamp. URL: https://basecamp.com/ (accessed: 26.02.2021).
11. Jira. URL: https://www.atlassian.com/ru/software/jira (accessed: 26.02.2021).
12. Asana. URL: https://www.saasworthy.com/product/asana#features (accessed: 26.02.2021).
13. Trello. URL: https://trello.com/ru?&aceid=&adposition=&adgroup=113770539286&campaign=11821318272&creative=486381327088&device=c&keyword=%2Btrello&matchtype=b&network=g&placment=&ds_kids=p59407420887&ds_e=GOOGLE&ds_eid=700000001550057&ds_e1=GOOGLE&gclid=CjwKCAjw3pWDBhB3EiwAV1c5rMNeFtsjXaozbaTdphNMtgQJ-JdoXfS3FOzf6LsOzJrJYPxOIIdVCRoCayoQAvD_BwE&gclsrc=aw.ds (accessed: 26.02.2021).
14. Microsoft Solutions Framework. Basic Principles. URL: https://newline.tech/microsoft-solutions-framework-basic-principles/ (дата звернення: 25.01.2021).
15. Брауде Э. Технология разработки программного обеспечения. СПб.: Питер, 2004. 655 с.

Стаття надійшла до редакції 02.04.2021