ISSN 2079-5459 (print)
ISSN 2413-4295 (online)

УДК 510.635:004.891(045) doi:10.20998/2413-4295.2021.02.08

РОЛЬ ТЕСТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ ДЛЯ КОМАНДНОЇ РОЗРОБКИ

А. І. ВАВІЛЕНКОВА

кафедра комп'ютеризованих систем управління ФККПІ, Національний авіаційний університет, Київ, УКРАЇНА
e-mail: vavilenkovaa@gmail.com

АНОТАЦІЯ *Досліджено особливості процесів тестування програмного продукту під час роботи у Scrum-команді. Виокремлено види тестування, характерні для методики гнучкого тестування, що дають змогу зробити процес тестування постійно інтегрованим у командну роботу, як на етапі формування вимог та проєктування, так і на етапі кодування та генерації тестових наборів. Це стає можливим при внесенні рекомендацій тестера у процес формування користувацьких історій, плануванні виходу версії програмного продукту з точки зору тестування та дефектів, плануванні спринту на основі користувацьких історій та дефектів, виконанні спринту з неперервним тестуванням, регресійному тестуванні після завершення спринту та формуванні звітів про результати тестування. Виділено етапи процесу гнучкого тестування у Scrum-команді. Запропоновано автоматизувати процес гнучкого тестування для навчального командного проєкту у програмному середовищі Visual Studio на базі Team Foundation Server. Продемонстровано приклад автоматизації процесу гнучкого тестування шляхом його розбиття на чотири квадранти для дотримання принципів роботи за гнучкою методологією розробки програмного забезпечення. У першому квадранті відбувається дослідження якості внутрішнього коду програмного продукту, тобто проведення модульного тестування. Для реалізації модульного тестування в Visual Studio створюється нове рішення з метою генерування класу для тестування, у тілі якого прописуються умови для тестування. Другий квадрант реалізує принципи системного тестування, тому на цьому етапі продемонстровано особливості створення такого тестового артефакту, як тестові випадки, що передбачає виконання певних умов для перевірки функціональності програмного продукту, який розробляється; встановлення зв'язку між створеними тестовими випадками та користувацькими історіями, представленими у вигляді задач командного проєкту Product Backlog. Наведено приклад ручного тестування за допомогою спеціального інструменту Microsoft Test Manager, який дозволяє створювати плани, додавати та оновлювати тестові випадки, виконувати ручні тести. У третьому квадранті здійснено дослідне тестування у Microsoft Test Manager та створено ще один тестовий артефакт – звіти про помилки. У четвертому квадранті здійснюється автоматичне тестування нефункціональних вимог до програмного забезпечення.*
Ключові слова: *гнучке тестування; гнучка методологія; Scrum-команда; командний проєкт; Product Backlog; тестовий випадок; програмне забезпечення*

THE ROLE OF SOFTWARE TESTING FOR TEAM DEVELOPMENT

A. VAVILENKOVA

Department of Computerized Control Systems, National Aviation University, Kyiv, UKRAINE

ABSTRACT *The peculiarities of software product testing processes while working in the Scrum-team were studied. The types of testing characteristic of the flexible testing technique are singled out, that make it possible to make the testing process constantly integrated into teamwork, both at the stage of requirements formation and design, and at the stage of coding and generation of test sets. This is possible by making the tester's recommendations in the process of creating user stories, planning the release of the software version in terms of testing and defects, planning a sprint based on user stories and defects, performing a sprint with continuous testing, regression testing after sprint ends and reporting of results of testing. The article highlights the stages of the flexible testing process in the Scrum team. It is proposed to automate the process of flexible testing for a training team project in the Visual Studio software environment based on Team Foundation Server. Author demonstrates an example of automation of the process of flexible testing by dividing it into four quadrants to comply with the principles of working on a flexible methodology of software development. In the first quadrant there is a study of the quality of the internal code of the software product, the modular testing. To implement modular testing in Visual Studio, a new solution is created in order to generate a class for testing, where the conditions for testing are prescribed in the body of the class. The second quadrant implements the principles of system testing. At this stage the features of creating such a test artifact as test cases are demonstrated. That provides for the fulfillment of certain conditions to verify the functionality of the developing software product; linking between created test cases to user stories presented as team project tasks Product Backlog. The study demonstrates an example of manual testing using a special tool Microsoft Test Manager, which allows you to create plans, add and update test cases, perform manual tests. In the third quadrant, it was performed a trial test in Microsoft Test Manager and it was created another test artifact - error reports. The fourth quadrant automatically tests non-functional software requirements.*
Keywords*: flexible testing; flexible methodology; Scrum-team; team project; Product Backlog; test case; software*

Вступ

Результатом командної роботи під час розробки програмного забезпечення є вчасно виготовлений, якісний програмний продукт. Часто у цьому процесі недооцінюють роль тестування, як важливого процесу перевірки відповідності поставлених до продукту вимог та реально

реалізованої функціональності, що спостерігається у штучно створених ситуаціях на обмеженому наборі тестів [1].

Згідно з практикою гнучкого тестування, що притаманна гнучким методологіям розробки програмного забезпечення, розпочинається тестування на початку проєкту з постійної інтеграції між розробкою та перевіркою проміжних результатів командної роботи [2]. Таким чином, гнучке тестування неперервне, на відміну від тестування за каскадною моделлю, де воно виконується лише після етапу кодування.

Тестування проходить через увесь життєвий цикл розробки програмного забезпечення, а тестери поряд з іншими членами Scrum-команди, вносять свій внесок на етапах формування вимог, проєктування, кодування та генерації тестових наборів [3].

Існує досить багато видів тестування, кожен з яких доцільно використовувати на тому чи іншому етапі розробки програмного забезпечення залежно від поставлених на даному спринті задач (рис. 1).

Рис. 1 – Види тестування при розробці програмного забезпечення

Своєчасне застосування конкретного виду тестування забезпечує реалізацію основної функції тестування – техніки контроля якості, здійснюючи аналіз та планування, розробку тестових сценаріїв, оцінку критеріїв закінчення тестування, написання звітів, рецензування документації та проведення статистичного аналізу [4,5]. Тому проблема автоматизації процесів тестування та вибору програмного забезпечення для цього є, безумовно, актуальною.

Мета роботи

Фактично баг або дефект існує при одночасному виконанні трьох умов: коли відомий очікуваний результат; коли відомий фактичний результат; коли фактичний результат відрізняється від очікуваного [6]. При цьому джерелами дефектів можуть бути помилки в специфікації, дизайні або реалізації програмного продукту, помилки використання програмного забезпечення, зумисне нанесення шкоди, умови зовнішнього середовища та потенційні наслідки попередніх помилок.

Для своєчасної фіксації та усунення дефектів при розробці програмного забезпечення членам команди потрібно мати постійний доступ до інформації та здійснювати постійну комунікацію щодо результатів роботи тестера. Це стає можливим завдяки автоматизації процесів тестування та використанню спеціальних програмних засобів, які дозволяють автоматично компілювати та перевіряти додатки, формувати звіти про помилки та здійснювати аналіз результатів тестування.

Тому метою даної статті є дослідження особливостей процесів тестування програмного продукту під час роботи у Scrum-команді.

Виклад основного матеріалу

Згідно з методикою гнучкого тестування, яку використовує командний проєкт, що функціонує на основі методології Scrum [7], тестування повинне включати в себе [8]:

- внесення рекомендацій тестера у процес формування користувацьких історій, що базується на очікуваній поведінці системи, тобто створення тестових випадків;
- планування виходу версії програмного продукту з точки зору тестування та дефектів;
- планування спринту на основі користувацьких історій та дефектів;
- виконання спринту з неперервним тестуванням;
- регресійне тестування після завершення спринту;
- формування звітів про результати тестування.

Усі ці етапи вдало поєднуються завдяки створенню та реалізації тестових артефактів: специфікації вимог, тест-плану, тестових випадків та звітів про помилки [9]. Автоматизувати процес гнучкого тестування у навчальному командному проєкті можна у програмному середовищі Visual Studio на базі Team Foundation Sever або інших його версій.

Особливістю гнучкого тестування є поділ всього процесу тестування на чотири квадранти, що допомагає Scrum-команді спланувати процес тестування [10].

Квадрант Q1 – передбачає дослідження якості внутрішнього коду програмного продукту, базується на створенні та проведенні модульного тестування. Модульні тести зберігаються в системі контролю версій та виконуються при кожній побудові додатку. Модульні тести є основою регресійного тестування, яке виконується при додаванні нових можливостей або модифікацій програмного забезпечення [11,12].

ISSN 2079-5459 (print)
ISSN 2413-4295 (online)

Для реалізації модульного тестування в Visual Studio створюється нове рішення (модульний тест) на основі шаблону «Native Unit Tests Project» з метою генерування класу для тестування (рис. 2), у тілі якого прописуються умови для тестування.

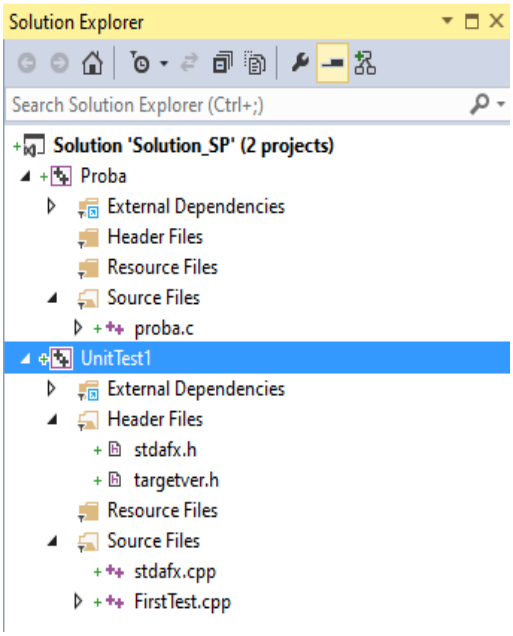

Рис. 2 – Створення модульного тесту в Visual Studio

Квадрант Q2 – реалізує принципи системного тестування, містить тестові випадки – один із артефактів тестування, орієнтований на вимоги користувача. Це передбачає створення тестових випадків, тобто певних умов для перевірки функціональності програмного продукту.

Робочий елемент «Test Case» у Visual Studio призначений для опису умов тестування вимог користувача. Наприклад, у командному проєкті створювався тестовий випадок для тестування коректності розрахунку мінімальної вартості, для якого на вкладці «Steps» вводилася послідовність дій, які необхідно зробити при тестуванні, та очікуваний результат у відповідь на вказану дію (рис. 3).

Кожен тестовий випадок підв'язується під певну задачу, що запланована та вирішується на конкретному спринті.

Для безпосереднього проведення тестування у програмному середовищі Visual Studio на основі Team Foundation Server використовується спеціальний інструмент тестування Microsoft Test Manager, який дозволяє створювати плани, додавати та оновлювати тестові випадки, виконувати ручні та автоматичні тести [13].

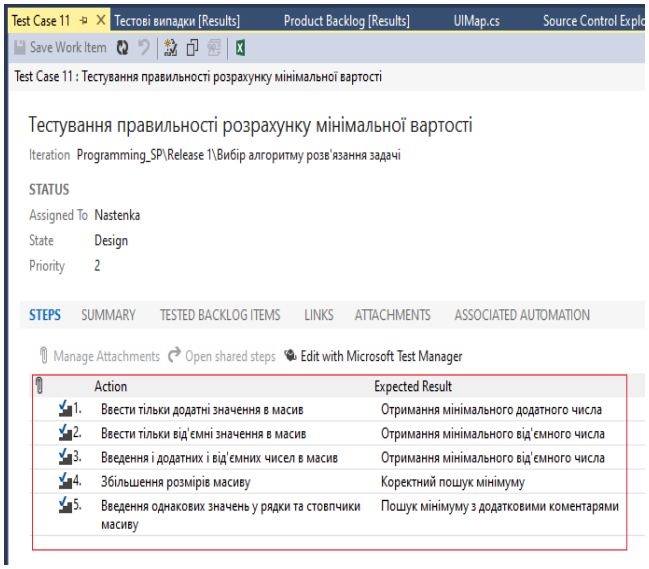

Рис.3 – Приклад створення тестового випадку в Visual Studio

На етапі здійснення системного тестування проводилося ручне тестування. Для цього на сторінці «Test» в Microsoft Test Manager обрався та запускався в роботу тестовий випадок. Після цього тестер виконує запропоновані кроки тестового випадку, цим самим виконуючи заплановані у тестовому плані дії, та співвідносить фактичний результат з очікуваним (рис. 4).

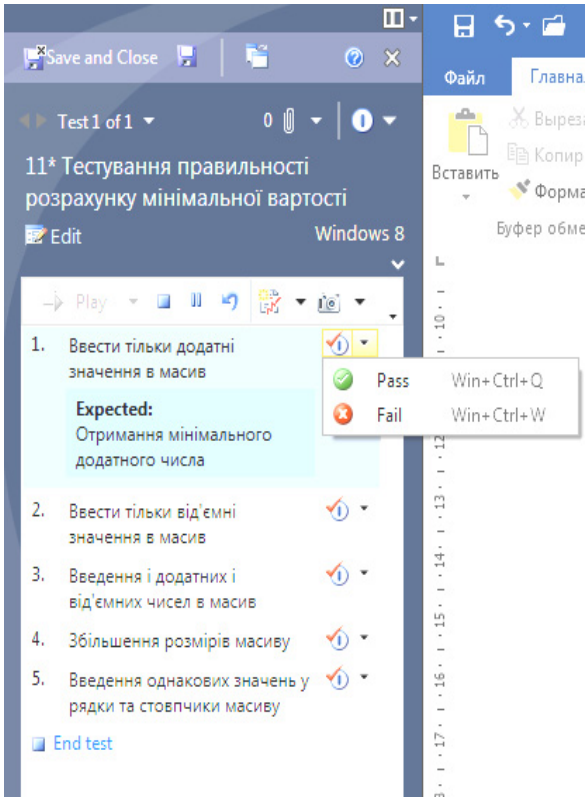

Рис. 4 – Приклад проведення ручного тестування у Microsoft Test Manager

Саме тому хід думок спеціаліста, що здійснює тестування, повинен відрізнятися від мислення розробника, а тестер повинен бути незалежним, володіти необхідними знаннями у предметній області та програмуванні, за рахунок чого може більше ефективно виявити дефекти в системі, ніж програміст. У тестуванні виділяють чотири рівні незалежності тестування:

- тести для програми розробляються та проводяться автором програми;
- тести розробляються та реалізуються людьми, не причетними до написання коду та розробки алгоритму;
- тести розробляються представниками інших організацій або спеціалізованими тестерами для проведення тестування тестерами команди-розробника;
- тести розробляються та виконуються спеціалістами з інших організацій.

Тестер виконує дії кожного кроку та встановлює індикатор відповідно до того, чи досягнуто очікуваний результат (рис. 5).

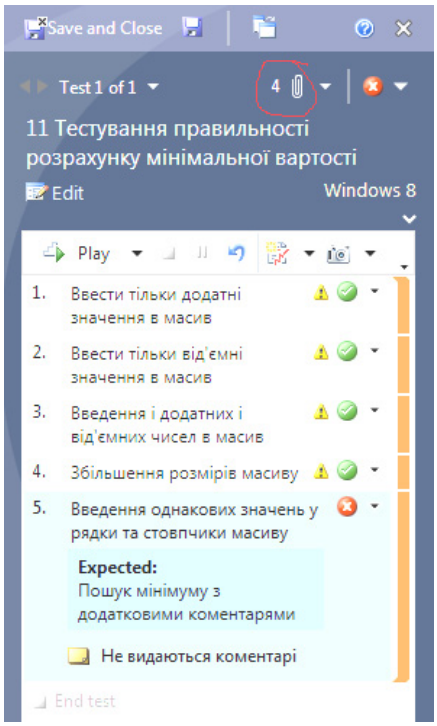

Рис. 5 – Результат проведення ручного тестування у Microsoft Test Manager

Квадрант Q3 – забезпечує зворотний зв'язок, а тестові випадки використовуються як основа для автоматичного тестування. У цьому квадранті проводиться багато циклів ітераційних перевірок шляхом здійснення дослідного тестування, тестування юзабіліті, приймального тестування.

Для дослідного тестування в інструменті Microsoft Test Manager передбачено можливість одночасного тестування (рис. 6) та створення ще одного важливого тестового артефакту - детального звіту про помилку (рис. 7).

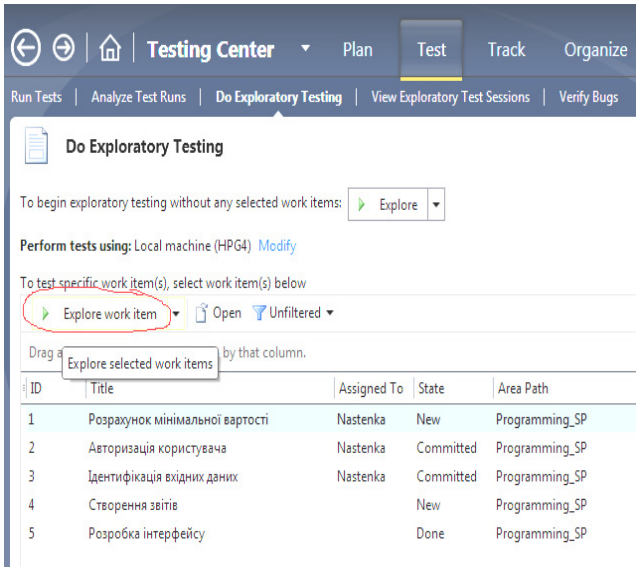

Рис. 6 – Приклад проведення дослідного тестування у Microsoft Test Manager

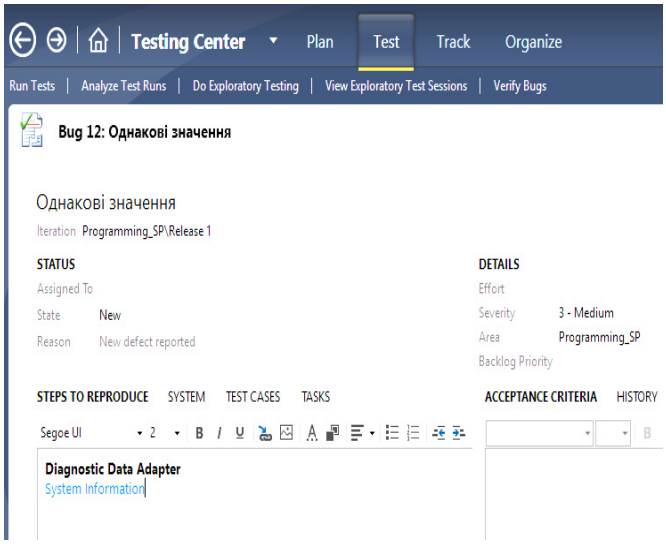

Рис. 7 – Приклад створення звіту про помилку

Квадрант Q4 – концентрує увагу на нефункціональних вимогах, таких як продуктивність, безпека, стабільність, для чого використовують спеціальні інструменти, а також тестування автоматизації.

У Visual Studio можна створювати декілька видів автоматичних тестів (закодовані тести користувача, модульні тести та тести для веб-форм), тип яких визначається конкретною метою задачі. Тому створене рішення для автоматичного тестування обов'язково підв'язується під задачу Product Backlog (рис. 8).

ISSN 2079-5459 (print)
ISSN 2413-4295 (online)

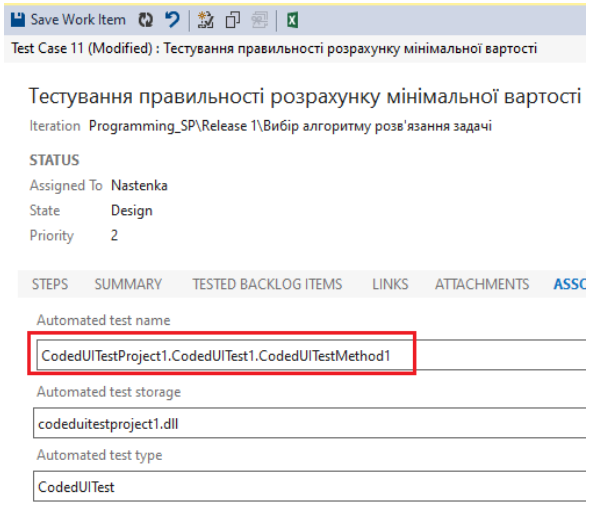

Рис. 8 – Приклад застосування автоматичного тесту

Після закінчення генерації проекту автоматичного тестування він додається у рішення всього командного проєкту і стає доступним для всіх членів Scrum-команди.

Наведені приклади реалізації механізмів тестування у програмному середовищі Visual Studio на базі Team Foundation Sever з використанням інструменту Microsoft Test Manager дають змогу простежити весь процес тестування створеного програмного продукту під час командної розробки згідно з гнучкою методологією Scrum.

Висновки

Тестування програмного забезпечення є складним, інтегрованим в усі етапи розробки процесом. Саме тому згідно принципів гнучкої методології Scrum потрібно дотримуватися основних вимог до тестування, зокрема, орієнтуватися на створення та роботу з тестовими артефактами [14]. Так, перший із них – це специфікація вимог – закінчений опис поведінки програми, яку потрібно розробити. Тест-план – документ, що описує весь обсяг робіт з тестування, починаючи з опису об'єктів тестування, стратегій, розкладу, критеріїв початку та закінчення, до необхідного в процесі тестування обладнання, спеціальних знань, а також оцінки ризиків. Тестові випадки, на створення яких та роботу з якими спрямовано матеріал даної статті, є набором дій або умов, необхідних для перевірки функціональності програмного продукту. Ще один важливий тестовий артефакт – це звіт про помилку, тобто технічний документ, який містить повний опис бага, включаючи інформацію про саму помилку та умови її виникнення. Таким чином, здійснення тестування на всіх етапах розробки програмного забезпечення при командній розробці – це запорука своєчасного створення якісного програмного продукту.

Список літератури

1. Брауде Э. *Технология разработки программного обеспечения*. СпБ.: Питер, 2004. 655 с.
2. Nidagundi P., Novickis L. Introducing Lean Canvas Model Adaptation in the Scrum Software Testing. *Procedia Computer Science*. 2017. Vol. 104. P. 97-103. doi: 10.1016/j.procs.2017.01.078.
3. Satherland D. Scrum. *The Art of Doing Twice the Work in Half the Time*. Random House, 2015. 256 p.
4. Tahera K., Wynn D. C., Earl C. et al. Testing in the incremental design and development of complex products. *Res Eng Design*. 2019. Vol. 30. P. 291–316. doi: 10.1007/s00163-018-0295-6.
5. Вавіленкова А. І., Литвиненко О. Є., Жолдаков О. О. *Управління проектами інформатизації*: навч. посіб. К.: НАУ, 2015. 220 с.
6. Kaner C., Falk J., Nguyen H.Q. *Testing Computer Software Second Edition*. Dreamtech Press, 2000. 478 p.
7. Вавіленкова А. І. Аналіз гнучких методологій розробки програмного забезпечення для реалізації у командних проєктах. *Вісник Національного технічного університету «ХПІ». Серія: Нові рішення в сучасних технологіях*. – Харків: НТУ «ХПІ». 2021. № 1 (7). С. 39-46. doi:10.20998/2413-4295.2021.01.06.
8. Black R. *Pragmatic Software Testing Becoming an Effective and Efficient Test Professional*. John Wiley & Sons, 2011. 366p.
9. Gokceoglu M., Sozer H. Automated defect prioritization based on defects resolved at various project periods. *Journal of Systems and Software*. 110993. 2021. doi: 10.1016/jss.2021.110993.
10. Shvaber K. Scrum. *Flexible product management and business*. Alpina Publisher Ukraine, 2019. 236 p.
11. Mitchell J., Black R. *Advanced Software Testing* - Vol. 3, 2nd Edition: Guide to the ISTQB Advanced Certification as an Advanced Technical. Test Analyst Rocky Nook, Inc., 2015. 480 p.
12. Giotis T. C. How to deliver successful IT projects using MSF team model and MSF process model. Paper presented at PMI® Global Congress 2007— EMEA, Budapest, Hungary. Newtown Square, PA: Project Management Institute. URL:https://www.pmi.org/learning/library/deliver-project-microsoft-solutions-framework-7413 (дата звернення: 28.04.2021).
13. Резник С., Бьйорк А., де ла Маза М. *Scrum с Team Foundation Server 2010. Профессиональный подход*, 2012. 416 с.
14. Breno G. Tavares, E, Carlos da Silva, Adler D. de Souza Risk Management in Scrum Projects: A Bibliometric Study. *Journal of Communications Software*. 2017. Vol. 13. No. 1. 1. doi: 10.24138/jcomss.v13i1.241.

References (transliterated)

1. Braude E. *Technologia razrabotki programmnogo obespecheniya*. SpB. Piter, 2004. 655 p.
2. Nidagundi P., Novickis L. Introducing Lean Canvas Model Adaptation in the Scrum Software Testing. *Procedia Computer Science*, 2017, Vol. 104, pp. 97-103, doi: 10.1016/j.procs.2017.01.078.
3. Satherland D. Scrum. *The Art of Doing Twice the Work in Half the Time*. Random House, 2015. 256 p.
4. Tahera K., Wynn D. C., Earl C. et al. Testing in the incremental design and development of complex products. *Res Eng Design,* 2019, Vol. 30, pp. 291–316, doi: 10.1007/s00163-018-0295-6.

5. Vavilenkova A., Litvinenko O., Zholdakov O. *Informatization Project Management.* K. NAU, 2015. 220 p.
6. Kaner C., Falk J., Nguyen H.Q. *Testing Computer Software Second Edition*. Dreamtech Press, 2000. 478 p.
7. Vavilenkova A. Analysis of flexible methodologies of software development for implementation in team projects. *Bulletin of the National Technical University "KhPI". Series: New solutions in modern technology*. – Kharkiv: NTU "KhPI", 2021, no. 1 (7), pp. 39–46, doi:10.20998/2413-4295.2021.01.06.
8. Black R. *Pragmatic Software Testing Becoming an Effective and Efficient Test Professional*. John Wiley & Sons, 2011. 366 p.
9. Gokceoglu M., Sozer H. Automated defect prioritization based on defects resolved at various project periods. *Journal of Systems and Software,* 110993, 2021, doi: 10.1016/jss.2021.110993.
10. Shvaber K. Scrum. *Flexible product management and business*. Alpina Publisher Ukraine, 2019. 236 p.
11. Mitchell J., Black R. *Advanced Software Testing* - Vol. 3, 2nd Edition: Guide to the ISTQB Advanced Certification as an Advanced Technical. Test Analyst Rocky Nook, Inc., 2015. 480 p.
12. Giotis T. C. How to deliver successful IT projects using MSF team model and MSF process model. Paper presented at PMI® Global Congress 2007— EMEA, Budapest, Hungary. Newtown Square, PA: Project Management Institute. Available at: https://www.pmi.org/learning/library/deliver-project-microsoft-solutions-framework-7413 (accessed: 28.04.2021).
13. Reznik S., Byork A., Maza M. *Scrum with Team Foundation Server 2010. Professional Approach*, 2012. 416 p.
14. Breno G. Tavares, E, Carlos da Silva, Adler D. de Souza Risk Management in Scrum Projects: A Bibliometric Study, *Journal of Communications Software*, 2017, Vol. 13, 1, 1, doi: 10.24138/jcomss.v13i1.241.

Відомості про авторів (About authors)

Вавіленкова Анастасія Ігорівна – доктор технічних наук, доцент, Національний авіаційний університет, професор кафедри комп'ютеризованих систем управління ФККПІ; м. Київ, Україна; ORCID: 0000-0002-9630-4951; e-mail: vavilenkovaa@gmail.com.

Anastasiia Vavilenkova – DSc, Docent, Professor, Department of Computerized Control Systems, Kyiv, Ukraine; ORCID: 0000-0002-9630-4951; e-mail: vavilenkovaa@gmail.com.

Будь ласка, посилайтесь на цю статтю наступним чином:

Вавіленкова А. І. Роль тестування програмного продукту для командної розробки. *Вісник Національного технічного університету « ХПІ». Серія: Нові рішення в сучасних технологіях*. – Харків: НТУ « ХПІ». 2021. № 2 (8). С. 56-61. doi:10.20998/2413-4295.2021.02.08.

Please cite this article as:

Vavilenkova A. The role of software testing for team development. *Bulletin of the National Technical University "KhPI". Series: New solutions in modern technology*. – Kharkiv: NTU "KhPI", 2021, no. 2 (8), pp. 56-61, doi:10.20998/2413-4295.2021.02.08.

Пожалуйста, ссылайтесь на эту статью следующим образом:

Вавиленкова А. И. Роль тестирования программного продукта при командной разработке. *Вестник Национального технического университета «ХПИ». Серия: Новые решения в современных технологиях*. – Харьков: НТУ «ХПИ». 2021. № 2 (8). С. 56-61. doi:10.20998/2413-4295.2021.02.08.

АННОТАЦИЯ *Статья посвящена исследованию особенностей процессов тестирования программного продукта во время работы в Scrum-команде. Выделены виды тестирования, которые характерны для методик гибкого тест. Как на этапе формирования требований к проекту и проектирования, так и на этапе кодирования и генерации тестовых наборов. Это становится возможным при внесении рекомендаций тестера в процесс формирования пользовательских историй, планирование выхода версий программного продукта с точки зрения тестирования и дефектов, планирование спринта на базе пользовательских историй и дефектов, выполнение спринта с непрерывным тестированием, регрессивное тестирование после завершения спринта и формирование отчетов по результатам тестирования. Выделены этапы процесса гибкого тестирования в Scrum-команде. Предложено автоматизировать процесс гибкого тестирования для учебного командного проекта в программной среде Visual Studio на базе Team Foundation Server. Продемонстрирован пример автоматизации процесса гибкого тестирования путем его разделения на четыре квадранта с целью соответствия принципам работы согласно гибкой методологии разработки программного обеспечения. В первом квадранте осуществляется исследование качества внутреннего кода программного продукта, то есть модульное тестирование. Для этого в Visual Studio создается новое решение с целью генерации класса для тестирования, в теле которого прописываются условия тестирования. Второй квадрант реализует принципы системного программирования, поэтому на этом этапе продемонстрированы особенности создания такого тестового артефакта, как тестовые случаи, что предусматривает выполнение определенных условий для проверки функциональности разрабатываемого программного продукта; установление связи между созданными тестовыми случаями и пользовательскими историями, представленными в виде задач командного проекта Product Backlog. Приведен пример ручного тестирования с помощью специального инструмента Microsoft Test Manage, который позволяет создавать планы, добавлять и обновлять тестовые случаи, выполнять ручные тесты. В третьем квадранте проделано исследовательское тестирование в Microsoft Test Manage и создан еще один тестовый артефакт – отчеты об ошибках. В четвертом квадранте осуществляется автоматическое тестирования нефункциональных требований к программному обеспечению.*

Ключевые слова: *гибкое тестирование; гибкая методология; Scrum-команда; командный проект; Product Backlog; тестовый случай; программное обеспечение*

Надійшла (received) 01.05.2021