АНАЛІЗ ЖИТТЄВОГО ЦИКЛУ РОЗРОБКИ ПРОЄКТУ WEB СИСТЕМИ

***Ткаченко В.П.**, проф. каф. МСТ, ХНУРЕ*
***Силантьєв В.Є.**, аспірант, кафедра МСТ, ХНУРЕ*

В доповіді наведено аналіз розробки WEB системи з визначеними вимогами з точки зору життєвого циклу та задіяних загальних ресурсів. Вибір інструментів, засобів, мов програмування, технічного завдання та побудови архітектури явно не проаналізовано у даному матеріалі. Розробка WEB системи з визначеними вимогами є технічним проектом, відповідає характеристикам проекту, а саме: має визначений початок і кінець, має ціль та вимоги, містить ресурси та витрати, включає ключові ролі, досягає визначеного результату. Результатом розробки є продукт або сервіс, який відповідає вимогам та потребам кінцевого користувача.

В залежності від особливостей та терміну виконання, існують декілька методологій розробки проекту, однак для будь-якого проекту можна виділити чотири основні фази його життєвого циклу: ініціація, планування, виконання та закриття [1]. На старті проекту, тобто його стадії ініціації (initiation) організовується стартова зустріч сторін (kick-off) з участю замовника та виконавців (відповідальна особа, керівник проекту, та команда, залучена у розробку та підтримку майбутнього проекту). На цій зустрічі обговорюються правила та основна ідея проекту. Замовник може презентувати своє бачення тих чи інших особливостей системи, продемонструвати базовий набір вимог та рекомендацій. Ці дані зрештою стануть основою для дорожньої карти проекту (roadmap). Цей документ визначає основні вимоги верхнього рівня та основні дати очікування їх реалізації. Також проводиться аналіз ризиків та невизначеності. Ризик – це міра невизначеності та можливих негативних наслідків прийнятих рішень або дій [2]. Ризики можна охарактеризувати за впливом (severity) та за вірогідністю проявлення (probability). В залежності від видів ризиків, можливо вжити профілактичних заходів, або зменшити їх вплив.

Враховуючи особливість таких вхідних даних та харакер вимог до проекту, що розробляється, доречно визначити методологію розробки проекту. Проаналізувавши основні принципові методології, можна зазначити, що будь-яка функціонально багата WEB система залучає велику кількість користувачів, містить багато розгалужених функцій, потребує ітеративного підходу для вдосконалення та підтримки. Виходячи з цього, пропонується використовувати Agile методологію для розробки проекту WEB системи. Scrum – це одна з найпопулярніших методологій Agile, він дозволяє команді розробників більш ефективно працювати над проектом, зменшуючи час і ризики, пов'язані з процесом розробки. Цей підхід заснований на впровадженні пріорітету функціональності, набору пропозицій для користувача, постійному оновлені функціональної бази системи, що розробляється [3].

Визначивши основні ролі, базові вимоги та методологію проекту починається наступна фаза проекту: планування (planning). Планування є часто

недооціненим, так як часто всі очікування перекладаються на наступну фазу виконання, насправді більшість зусиль, що зроблено лідером проекту під час планування, відіграє потім під час виконання, коли ризик зміни, модифікації або відтермінування проекту вже є значно складнішим та ресурснішим. Як правило, чим більш детально розроблена фаза планування, тим менші витрати потрібні на фазі розробки, оскільки вже були враховані багато факторів та ризиків, що дозволяють уникнути непередбачуваних ситуацій та знизити витрати на виправлення помилок. Це не виключає оновлень з боку замовника вже під час виконання проекту, або інших непередбачених ризиків, а навпаки підкреслює важливість планування та оцінки ризиків якомога раніше.

На етапі планування затверджується дорожня карта проекту, план, який відображає майбутні кроки розробки проекту, часові рамки та очікувані результати. Кожен крок – затверджена вимога або функціональне рішення верхнього рівня, яке було затверджене попередньо. Весь термін життєвого циклу проекту розбивається на рівномірні відрізки, для розробки WEB системи оптимально взяти один відрізок довжиною один календарний місяць, так як за цей час найбільш оптимально накопичувати відгуки користувачів, нові вимоги, здійснити їх аналіз, розробити необхідні оновлення, та запровадити їх результат. Один рівний часовий відрізок, що є компонентом циклу проекту по методології Scrum називається спрінт (sprint). До кожного спрінту, згідно з Scrum, додається один або більше з пунктів вимог roadmap.

Ефективний спосіб занесення та моніторингу запланованих задач можливий за допомогою графічного представлення об'єму задач, наприклад за допомогою діаграми Ганта. Лідер проекту переносить всі задачі до списку. Кожна комірка на діаграмі характеризує визначену вимогу, яка обмежена строками їх виконання. Кольорами можливо визначити вид діяльності у проекті за технологією або за відповідальною особою. З такою діаграмою дуже легко наглядно репрезентувати активність проекту та відслідковувати поточні зміни, вона рекомендується до впровадження на етапі планування та використання на етапі розробки. Приклад діаграми, зображений на рисунку 1 (для прикладу, на діаграмі перераховані вимоги уявного проекту по розробці WEB системи).

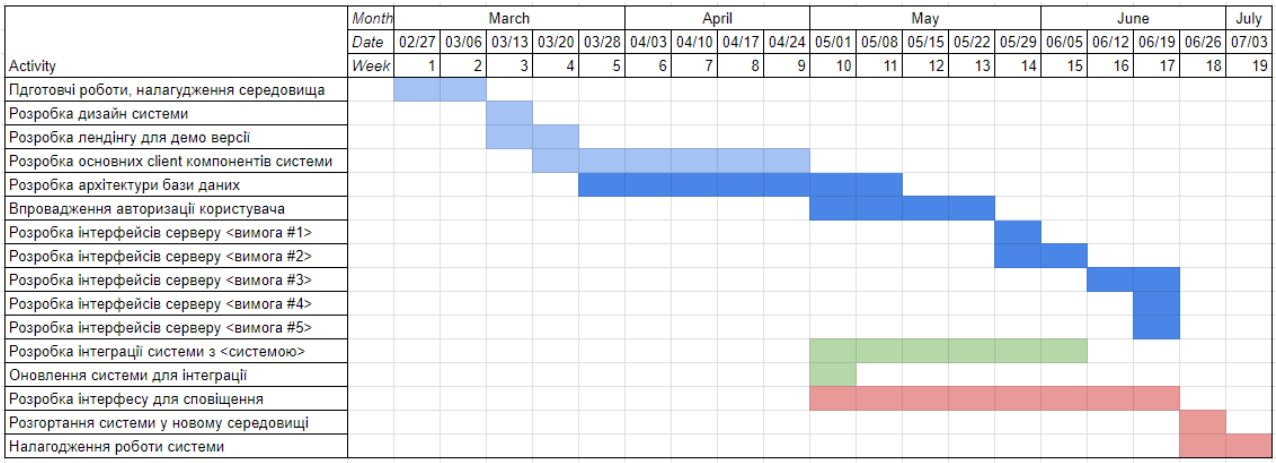

Рисунок 1 – Діаграма Ганта – Життєвий цикл проекту WEB системи

Планування визначає перелік завдань та відповідний рівень кваліфікації для їх виконання. Враховуються навички та знання (hard skills), а також досвід та вміння комунікувати та співпрацювати у команді (soft skills). Лідер команди має найбільш відповідальну роль в організаційній та комунікативній діяльності. Поведінка лідера команди визначається принципом: відреагувати (react), визначити (read), відповісти (respond), вирішити (resolve/repeat) [4]. При настанні надзвичайної події, лідер проекту повинен перш за все професійно відреагувати, відповісти замовнику щодо ознайомлення з проблемою, повідомити, що вживаються всі заходи щодо ліквідації та надати якомога конструктивну інформацію найбільш оперативно. Комунікація – одна з найбільш важливих компонентів зрілого процесу ведення проектів. Далі, необхідно зібрати команду та обговорити причини появлення проблеми. Попередження та профілактика ризиків, є більш важливою, ніж реагування на ситуацію, що вже трапилася на наступному етапі – етапі виконання проекту.

Фаза проекту виконання (execution) починається з першим спринтом , що визначається на плануванні та починається згідно з зазначеними датами. При нормальному перебігу подій, всі задачі що заплановані на спринт повинні бути виконані. Процес розробки складається з виконання визначених завдань, процесом координації та комунікації всередині команди, сповіщення замовника щодо перебігу подій, підтримкою службових процесів у проекті. Кожен день проводиться зустріч всієї команди (можливо наявно або дистанційно). Ця денна зустріч (daily) слугує мінімальним зрізом оновлень по проекту: на ній кожен доповідає результат роботи за добу, труднощі які його спіткали, та що планується зробити за поточний день. Комунікація з замовником відбувається на постійній домовленій основі, як правило щотижнево (weekly) методом email сповіщення. У звіті вказується перелік виконаних та поточних дій над проектом, дані можуть бути підкріплені прикладами та візуальними матеріалами, посиланнями до джерел моніторингу та управління проектом. Також невід'ємною складовою фази execution є нагляд і контроль проекту (monitoring and control). Він виконується командою розробки та операції (DevOps), що займаються забезпеченням безперервної інтеграції та доставки ПЗ (Continuous Integration/Continuous Development), управління інфраструктурою, автоматизацією розгортання системи. Кожен спрінт закінчується збіркою системи, що працює, т.з. мінімально життєздатний продукт, minimal viable product (MVP), та є оновленою версією попередньої системи. В кінці спрінта також проводиться зустріч (retrospective), на якій обговорюються сильні та слабкі сторони минулого спринту, ризики та проблеми заносяться лідером проекту у документацію.

Завершений спрінт супроводжується звітами по тестуванню системи по завершенню кожної ітерації. Тестування відбувається залученою до команди або окремою командою оцінки якості системи (Quality assurance). Принцип, за якими діють відповідальні за тестування особи можна охарактеризувати як вхід (input), що заснований на документації, процесі (process) – експертної оцінки та аналізу якості системи: автоматизованій або мануальній, та виході (output) – результат та звіт з тестування. Тестувальники отримують регулярно оновлювану документацію

для (на даний момент) фінальної версії системи. Поміж оновленням архітектури та характеристиками системи, важливим є також функціональна документація, яка орієнтована на задачі та функції (task-oriented documentation). Вона відображає суть системи найбільш ефективно та доцільно. Розробляючи документацію, лідеру проекту слід дотримуватись правил оформлення документації.

Таблиця 1 – Показники якості функціональної документації проекту WEB системи

Характеристика	Опис
Користність	Вирішення конкретної проблеми, релевантність
Пошуковість	Зручна навігація, впровадження SEO (для web документації з відкритим доступом;
Візуальність	Присутня інфографіка, переважає графіка над текстом, застосовується відео, діаграми
Зручність	Витриманість стилю, наявність посилань, організація згідно з типовими сценаріями використання;
Читабельність	Регулярне оновлення, актуальність, точність, перевіреність (validated tested);
Надійність	Легкість, структурованість, значимість
Граматика	Можливість легкого перекладу на іншу мову; корректний тон, простота;
Доступніть	Доступність online, джерело, на яке можна дати посилання.

По завершенні останнього спрінта проект переходить у заключну стадію закриття (closure). На цьому етапі готується фінальна версія системи та документації, ретроспективно зазначаються ризики та невизначеності, дороблюються фінальні правки та оновлення. Графічно весь життєвий цикл проаналізоаного методу розробки можна представити на рисунку 2.

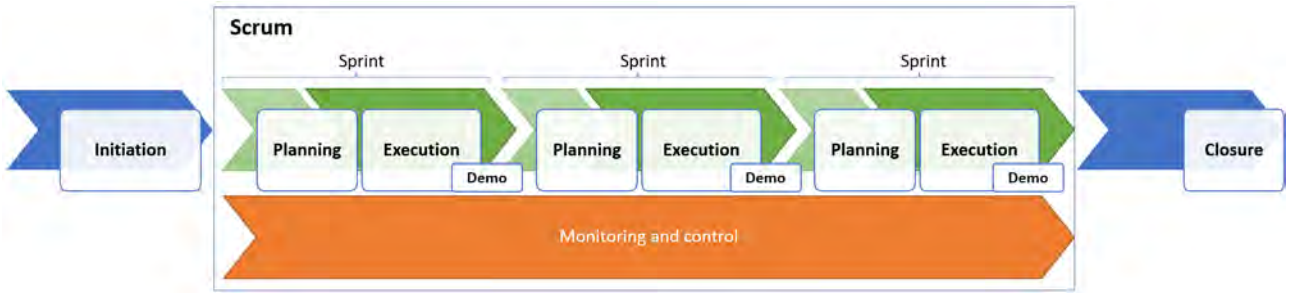

Рисунок 2 – Scrum життєвий цикл проекту розробки WEB системи

Таким чином, аналіз процесу розробки на прикладі WEB системи з зазначеними вимогами дозволяє визначити важливість фази ініціації та планування, при яких виявляється більшість потенційних ризиків та завдань. Підготовлена команда краще буде спроможною довести проект до завершення в умовах ітеративних змін, методологія заснована на функціональності системи та індивідуальному підході найбільш життєздатна, ніж система, яка не може підлягати модифікаціям. Комунікація та актуалізація змін, у вигляді документів або звітів – допомагає мати єдине бачення між замовником та виконавцем. Обрана методологія дозволяє ефективно та оперативно відображати зміни у проекті та реагувати на нові вимоги та зауваження в умовах стрімкого розвитку та використання затребуваних WEB систем.

Список літератури

1. Project Management Institute. (2017). Project Management Body of Knowledge (PMBOK) ПМІ.
2. Хілсон, Д. (2015). Ідентифікація та оцінка ризиків в проектах. Манн, Іванов та Фербер.
3. Schwaber, K. (2004). Agile Project Management with Scrum. Microsoft Press.
4. Baca, C.M. (2016). The Project Manager's Book of Checklists: How to Complete a Project Successfully, Smoothly and On Time. CRC Press.