**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ**
**Кафедра економічної кібернетики**

Завідувач кафедри
економічної кібернетики
_____Іванченко Н.О.
ДОПУСТИТИ ДО
ЗАХИСТУ
«____» _____2020 р.

# КВАЛІФІКАЦІЙНА РОБОТА

(Пояснювальна записка)
здобувача освітньо ступеня «Магістр»

**Тема:** Agile як модель управління проектами в сфері IT

**Виконав:** Коваль Юрій Леонідович

**Керівник:** к.е.н., доц. Іванченко Надія Олександрівна

**Консультанти з розділів:**

Розділ 1: к.е.н., доц. Іванченко Н.О.

Розділ 2: к.е.н., доц. Іванченко Н.О.

Розділ 3: к.е.н., доц. Іванченко Н.О.

**Нормоконтролер із ЄСКД (ЄСПД):**

к.е.н. Густера Олег Михайлович

Національний авіаційний університет
Факультет економіки та бізнес-адміністрування
Кафедра економічної кібернетики
Освітній ступінь «Магістр»
Спеціальність 051 «Економіка», ОПП «Цифрова економіка»

ЗАТВЕРДЖУЮ
Завідувач кафедри
економічної кібернетики
_____Іванченко Н.О.
«____» _____2020 р.

# ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студента: Коваля Юрія Леонідовича
Тема роботи: «Agile як модель управління проектами в сфері IT» затверджена наказом ректора № 1967/ст. від 08 жовтня 2020р.

1. Термін здачі студентом закінченої роботи на кафедру “10” грудня 2020 р.
2. Вихідні дані до роботи: використано літературні джерела та матеріали мережі Інтернет.
3. Зміст дослідження (перелік питань до розробки):
- здійснено аналіз процесів управління проектами;
- досліджено характеристику управління проектами в IT сфері;
- обґрунтовано особливості гнучких методів управління;
- проведено опис методології «скрам»для управління проектами в IT сфері;
- запропоновано та реалізовано моделювання та аналіз впровадження проекту за допомогою «гнучкою» методології «скрам».
4. Перелік обов’язкових демонстраційних матеріалів : 16 слайдів.

КАЛЕНДАРНИЙ ПЛАН

| № п/п | Назва етапів та питань, які повинні бути розроблені відповідно до завдання | Термін виконання | Позначки керівника про виконання завдань |
|---|---|---|---|
| 1 | Отримання завдання на кваліфікаційну роботу | 12.10.2020 | |
| 2 | Огляд літератури за темою | 19.10.2020 | |
| 3 | Характеристика управління проектами в ІТ сфері | 2.11.2020 | |
| 4 | Аналіз процесів управління проектами | 9.11.2020 | |
| 5 | Моделі управління проектами | 16.11.2020 | |
| 6 | Особливості гнучких методів управління | 23.11.2020 | |
| 7 | Вибір методологій для моделювання процесу управління | 30.11.2020 | |
| 8 | Опис методології «скрам»для управління проектами в ІТ сфері | 7.12.2020 | |
| 9 | Написання пояснювальної записки. Аналіз отриманих результатів з керівником | 10.12.2020 | |
| 10 | Створення слайдів та написання доповіді | 14.12.2020 | |
| 11 | Попередній захист дипломної роботи | 15.12.2020 | |
| 12 | Корегування роботи за результатами попереднього захисту | 16.12.2020 | |
| 13 | Остаточне оформлення пояснювальної записки та слайдів | 17.12.2020 | |
| 14 | Підписання відгуку та рецензії | 17.12.2020 | |

1. Дата видачі завдання “12” жовтня 2020 р.

Керівник _____ к.е.н., доцент Н.О. Іванченко
(підпис)

Завдання прийняла для виконання _____ Ю.Л. Коваль
…….. (підпис)

## Реферат

У кваліфікаційній роботі магістра був змодельований план «скрам»-проекту, що задовольняє якість і вимоги замовника з урахуванням максимально можливих відхилень по тривалості вирішення завдань. В результаті відхилення по тривалості виконання проекту склали 18,6%, а за вартістю 17,5% при практично мінімальних відхиленнях від якості.

Проведений аналіз показує, що в умовах швидко мінливих і нечітко визначених вимогах (такі умови як раз-таки властиво банківській сфері), а так само при низькому рівні комунікації між відділами компанії, використання методології «скрам» в управлінні IT-проектами є найбільш ефективним як для вендора, так і для замовника. Говорячи про вендорів, варто відзначити, що злагоджена робота, яка визначається самими фахівцями, які виконують її, набагато більше стимулює виконавців на якісне вирішення завдань, ніж постійні доручення менеджера проекту і перебування у вічному стресі. Для замовника вигода очевидна: проект впроваджується швидше, якісніше і з меншими витратами в порівнянні з традиційним методом.

Основний зміст роботи викладено на 81 сторінці. Робота містить 3 таблиці, 19 рисунків, 42 літературні джерела.

*Ключові слова*: проект, управління, гнучкі методології, IT прект, Agile, моделювання.

## ABSTRACT

The high-quality robots of the master of the design department have a "scrum" plan - a project that is satisfied with the quality of the deputy's deputy for the highest possible views of the triviality of the plant. As a result, 18.6% of respondents received a trivial view of the project, and 17.5% for a rate with practically minimal results in terms of quality.

Carrying out analysis shows that in the minds of smartly small and unclear ways in the minds of the most powerful banking spheres, as well as, with a low level of comunication in the management of projects, in the company's IT projects for the

vendor, as well as for the deputy. Talking about vendors, it means that the robot is grateful, because it's supposed to be started by the people themselves, because they'll be able to see them, and more incentivized to make inquiries on how the manager's decision is made to the project. For the deputy, the vigoda is obvious: the project will be carried out more quickly, better and with less vitrates, in accordance with the traditional method.

The main content of the work is set out on 81 pages. The work contains 3 tables, 19 figures, 42 references.

*Key words:* project, management, nasty methodology, IT project, Agile, modeling.

**ЗМІСТ**

# ВСТУП

*Актуальність теми.* У сучасному світі редакції приділяють увагу не тільки способам і жанрами подачі інформації, а й організації праці, методам і способам управління людськими та іншими ресурсами, оптимізації діяльності.

Новою тенденцією в регулюванні роботи IT компаній сьогодні є проектний менеджмент, який включає в себе ряд методологій. їх можна розділити на три групи: гнучкі, жорсткі і гібридні. В той час як західні колеги застосовують гібридні методології, українські IT компанії тільки починають робити перші кроки в бік застосування гнучкого управління, зокрема, Agile. Його використання допомагає компаніям в виявленні прихованого потенціалу, вивченні і застосуванні нових підходів до управління. Актуальність переходу на гнучкі методології для IT компаній обумовлюється наступним: безперервний розвиток науково-технічного прогресу, постійне посилення конкуренції на сучасному ринку інформаційних технологій України та іншими інтеграційними явищами.

Управління проектами сьогодні відрізняється від управління проектами в 1950-х, коли ця наука стала вперше застосовуватися будівельними та інженерними копання в якості інструменту для ефективної організації нової діяльності. Зараз управління проектами використовується практично у всіх областях, так як це дозволяє підтримувати економічну активність організацій.

З розвитком інформаційних технологій багатьом компаніям стало очевидно, що, використовуючи різні автоматизовані інформаційні системи, можна підвищити рентабельність бізнесу. Особливо це стало актуально для банків, так як вони намагаються підвищити свою конкурентоспроможність на ринку за допомогою використання спеціалізованих програмних систем, орієнтованих на банківську діяльність. Таким чином, з підвищенням ступеня залученості IT-продуктів в життя суспільства, тема про ефективне управління IT-проектами стала дуже популярною.

У даній кваліфікаційній роботі управління IT-проектом буде розглядатися в банківській сфері, яка вважається досить вразливою через сильну кореляцію з

зовнішнім середовищем і економікою країни. Зважаючи на ці своєрідні властивості, а також характеристики IT-проекту, управління впровадженням або створенням ПЗ в банках стикається з безліччю труднощів, такими як порушення термінів, неналежну якість, перевищення по запланованій вартості тощо. Неналежна якість продукту, що є результатом управління будь-яким IT-проектом в банку, може сильно позначитися на подальшій економічній активності цього банку.

*Об'єктом дослідження* кваліфікаційної роботи магістра є процес моделювання управління проектами в IT сфері.

*Предметом дослідження* є технології Agile, методи та моделі управління проектами в IT сфері.

*Метою даної кваліфікаційної роботи* є аналіз і подальший розвиток гнучких методів управління проектами, інформаційних технологій та моделей проектування за допомогою «гнучких» методологій.

*Завданнями кваліфікаційної роботи магістра є:*

- аналіз процесів управління проектами;
- дослідити характеристику управління проектами в IT сфері;
- обґрунтувати особливості гнучких методів управління;
- провести опис методології «скрам»для управління проектами в IT сфері;
- запропонувати та реалізувати моделювання та аналіз впровадження проекту за допомогою «гнучкою» методології «скрам».

*Методи дослідження* – теоретичним та методологічним підґрунтям для написання кваліфікаційної роботи стали праці вітчизняних і зарубіжних вчених, що присвячені питанням управління проектами в IT сфері.

*Практичне значення одержаних результатів* полягає в тому, що змодельований план «скрам»-проекту задовольняє якість і вимоги замовника з урахуванням максимально можливих відхилень по тривалості вирішення завдань. В результаті відхилення по тривалості виконання проекту склали 18,6%, а за вартістю 17,5% при практично мінімальних відхиленнях від якості.

*Апробація результатів дослідження*. За результатами кваліфікаційної роботи магістра опублікована стаття у Науковому фаховому виданні «Modern Economiscs» 2020. – Вип.20. на тему: “Безпека підприємства в умовах цифрової трансформації економіки”.

# РОЗДІЛ 1
# АНАЛІЗ МЕТОДІВ ТА ТЕХНОЛОГІЙ УПРАВЛІННЯ ПРОЕКТАМИ

## 1.1. Характеристика управління проектами в IT сфері

Управління проектами в області інформаційних технологій за останній час завоювало визнання як найкращий метод планування та управління реалізацією інвестиційних проектів. За американськими оцінками застосування методології управління проектами забезпечує високу надійність досягнення цілей проекту і на 10-15% скорочує витрати на його реалізацію.

У світі накопичено величезний досвід застосування управління проектами в області інформаційних технологій. Зокрема, ця методологія застосовується у всіх великих IT компаніях світу. Програмні засоби для управління проектами встановлені на мільйонах комп'ютерах в усьому світі - тільки пакет Microsoft Project встановлений більш ніж на два мільйони комп'ютерів.

Асоціація управління проектами Project Management Institute (Інститут Управління Проектами) об'єднує близько 40 тисяч членів і має відділення майже на всіх континентах.

Розглянемо основні поняття і методи управління проектами.

Проект - це тимчасове підприємство, призначене для створення унікальних продуктів або послуг. В даному контексті "тимчасове" означає, що у кожного проекту є початок і неодмінно настає завершення, коли досягаються поставлені цілі, або приходить розуміння, що ці цілі не можуть бути досягнуті [2].

В даному контексті "унікальних" означає, що створювані продукти або послуги відрізняються від інших аналогічних продуктів і послуг. Приклади проектів: розробка нового обладнання, розробка або впровадження програмних засобів і т.д.

Згідно PMBook управління проектами - це застосування знань, досвіду, методів і засобів до робіт проекту для задоволення вимог, що пред'являються до проекту, і очікувань учасників проекту. Щоб задовольнити ці вимоги й очікування необхідно знайти оптимальне поєднання між всіма характеристиками проекту [1].

Управління проектами підпорядковується чіткої логіці, яка зв'язує між собою різні області знань і процеси управління проектами. Кожен проект приводить до створення унікального продукту, послуги або результату.

Перш за все, у проекту обов'язково є одна або кілька цілей. Під цілями розуміється не тільки кінцеві результати проекту, а й обрані шляхи досягнення цих результатів (наприклад, застосовувані в проекті технології, система управління проектом).

Досягнення цілей проекту може бути реалізовано різними способами. Для порівняння цих способів необхідні критерії успішності досягнення поставлених цілей. Зазвичай в число основних критеріїв оцінки різних варіантів виконання проекту входять терміни і вартість досягнення результатів.

При цьому заплановані цілі і якість зазвичай служать основними обмеженнями при розгляді та оцінці різних варіантів. Звісно, можливе використання і інших критеріїв, і обмежень - зокрема, ресурсних.

Для управління проектами необхідні важелі. Впливати на досягнення результатів проекту, цілі, якість, терміни і вартість виконання робіт можна, вибираючи сучасні технології, склад, характеристики і призначення ресурсів на виконання тих чи інших робіт. Таким чином, застосування технологій і ресурсів проекту можна віднести до основних важелів управління проектами.

Крім цих основних існують і допоміжні засоби, призначені для управління основними. До таких допоміжним важелів управління можна віднести, наприклад, контракти, які дозволяють залучити потрібні ресурси в потрібний термін. Крім того, для управління ресурсами необхідно забезпечити ефективну організацію робіт. Це стосується структури управління проектом, організації інформаційної взаємодії учасників проекту, управління персоналом.

Інформація, яка використовується в управлінні проектами, зазвичай не буває стовідсотково достовірною. Облік невизначеності вихідної інформації необхідний і при плануванні проекту і для грамотного укладання контрактів.

Аналізу та обліку невизначеностей присвячений аналіз ризиків.

Управління проектом здійснюється за допомогою належного застосування та інтеграції логічно згрупованих процесів управління проектом, об'єднаних в 5 груп [1]:

- Процеси ініціації - ухвалення рішення про початок виконання проекту;
- Процеси планування - визначення цілей і критеріїв успіху проекту і розробка робочих схем їх досягнення;
- Процеси виконання - координація людей і інших ресурсів для виконання плану;
- Процеси моніторингу і контролю - визначення відповідності плану і виконання проекту поставленим цілям і критеріям успіху і прийняття рішень про необхідність застосування коригувальних дій і визначення необхідних коригувальних впливів, їхнє узгодження, затвердження і застосування;
- Процеси закриття - формалізація виконання проекту і підведення його до впорядкованого фіналу.

Практично методологія управління проектами допомагає:

- обґрунтувати доцільність інвестицій,
- розробити оптимальну схему фінансування робіт,
- скласти план робіт, що включає терміни виконання робіт,
- споживання ресурсів, необхідні витрати,
- оптимальну організацію

Проекти часто використовуються з наступних стратегічних міркувань [2]:

вимога ринку (наприклад, автомобілебудівна компанія авторизує проект по виготовленню більш економічних автомобілів у відповідь на брак бензину);

стратегічна можливість / бізнес-потреба (наприклад, тренінгова компанія авторизує проект зі створення нового курсу навчання з метою збільшення доходів);

соціальна потреба (наприклад, неурядова організація в країні, що розвивається авторизує проект з надання систем питного водопостачання, туалетів і санітарної освіти співтовариствам, страждаючим від високого рівня інфекційних захворювань);

захист навколишнього середовища (наприклад, державна компанія авторизує проект по створенню нового сервісного центру для електромобілів, які сприяють скороченню забруднення навколишнього середовища);

вимога замовника (наприклад, компанія-виробник електроенергії для громадського користування авторизує проект по будівництву нової підстанції для електропостачання нового промислового району);

технологічний прогрес (наприклад, виробник комп'ютерної техніки авторизує проект по розробці більш швидкодіючого, економічного і компактного ноутбука з використанням досягнень в технології виготовлення комп'ютерної пам'яті і електронних компонентів);

юридична вимога (наприклад, виробник хімічних речовин авторизує проект по розробці керівних вказівок щодо поводження з новим токсичним матеріалом).

Організації здійснюють керівництво для визначення стратегічного напрямку і параметрів продуктивності. Даний стратегічний напрямок надає мету, очікування, завдання і дії, необхідні для керівництва діяльністю організації, і приводиться у відповідність з бізнес-цілями. Роботи з управління проектом повинні бути приведені у відповідність з напрямком організації на верхньому рівні, і в разі зміни мети проект повинен бути переглянутий. В умовах виконання проекту зміни в цілі проекту впливають на ефективність і успіх проекту.

Основними принципами нової концепції управління є [4]:

1) поглиблення рівня обгрунтованості прийнятих інвестиційних рішень, використовуючи механізм різноманітних і багатофакторних (технологічних, економічних, соціальних, екологічних та інших) оцінок;

2) високий ступінь координації та контролю робіт в процесі виконання проекту;

3) систематичний аналіз і врахування зовнішніх змін (кон'юнктури ринку за всіма видами ресурсів, непередбачених обставин і негативних чинників) при реалізації проектів.

Управління проектами базується на системному підході, що дає можливість декомпозиції і структуризації проекту будь-якої складності при прийнятті рішень в складних умовах (ситуаціях). Відмінна особливість методології управління проектами полягає в зосередженні прав і відповідальності за досягнення цілей проекту на одній людині або невеликій групі. Ці права і обов'язки здійснює керівник проекту. При цьому забезпечується [6]:

1. Визначення всіх видів робіт, необхідних для досягнення цілей проекту, їх структуризація і визначення взаємозв'язків.

2. Складання і контроль кошторису витрат по реалізації проекту.

3. Розробка та контроль графіків робіт, необхідних для досягнення бажаного результату.

4. Розподіл ресурсів, виділених для реалізації проекту, в умовах невизначеностей і ризиків.

5. Управління якістю виконання всіх робіт по проекту, включаючи досягнення планованого якості продукції, пропонованої проектом, і виконання інших вимог замовника.

6. Управління ризиком (ризиками) на всіх етапах здійснення проекту.

7. Забезпечення зв'язку з клієнтами, замовниками, споживачам продукції, з різними групами і особами, залученими до проекту (соціальні групи, місцеве населення, влада, засоби масової інформації) для вирішення всіх питань, пов'язаних з досягненням успіху проекту.

При розгляді та вивченні діяльності з управління реалізацією проектів можна виділити ряд підходів, що визначають структуризацію цієї діяльності. Найбільш поширені: функціональний, динамічний і предметний підходи.

a) Функціональний підхід відображає загальний підхід до проблеми управління і передбачає розгляд основних управлінських функцій або видів управлінської діяльності при здійсненні проектів у відповідності з класифікацією, прийнятою в менеджменті. У той же час, специфіка управління проектами проявляється в певній деталізації окремих функцій менеджменту і деякому зміщенні акцентів в їх утриманні стосовно реалізації та специфіки конкретних проектів. З урахуванням цього в рамках функціонального підходу розглядаються управлінські дії, визначаються як функції управління проектами, включаючи: аналіз, планування, організацію, контроль і регулювання.

b) Предметний підхід визначає структуризацію управлінського процесу по об'єктах проекту, на які направлено управління. У рамках предметного підходу розглядаються два типи об'єктів: виробничі об'єкти (перший тип) і елементи, пов'язані з діяльністю щодо забезпечення реалізації проекту (другий тип). Об'єктами другого типу можуть бути: фінанси, кадри, маркетинг, контроль, ризики, матеріальні ресурси, якість, інформація і інші елементи, що забезпечують отримання бажаного результату при реалізації проекту. Основні об'єкти першого і другого типів розглядаються в міру необхідності при вивченні дисципліни.

c) Динамічний підхід (відомий в літературі як спеціальний менеджмент - project management), передбачає розгляд у часі процесів, пов'язаних з реалізацією основних стадій і етапів здійснення проекту: аналіз проблеми, розробка концепції проекту, проектування, будівництво, монтаж, пусконалагоджувальні роботи, експлуатація і завершення проекту.

Деталізація розглянутих процесів стосовно до конкретного проекту (Етапів і стадій робіт здійснення проекту) може змінюватися в широких межах.

Методологія управління проектами відбивається в стандартах управління проектами. В даний час існують наступні види стандартів [19]:

- міжнародні - стандарти, які отримали міжнародне значення в процесі свого розвитку або призначені для міжнародного використання;

- національні - створені для застосування всередині однієї країни або отримали загальнонаціональний статус в процесі свого розвитку;

- громадські - підготовлені і прийняті спільнотою фахівців;

- приватні - комплекси знань, пропаговані для вільного використання приватними особами, компаніями або установами;

- корпоративні - розроблені для застосування в межах однієї компанії або всередині групи споріднених компаній.

Міжнародні стандарти є повні системи, включають, крім опису вимог до управління проектами, навчання, тестування, аудит, консалтинг та інші елементи. Всеохоплюючих міжнародних стандартів управління проектами поки не існує, але найбільш відомі такі нормативні документи.

1. Project Management Body of Knowledge (PMBOK) Американського інституту управління проектами (Project Management Institute - PMI). Цей стандарт оновлюється приблизно один раз в чотири роки. Одна з найбільш поширених редакцій датується 2000 р а найактуальніша, четверта, версія стандарту - The Guide to the PMBOK, 4th Edition - вийшла в кінці 2008 р [1]

Стандарт був спочатку прийнятий Американським національним інститутом стандартів (ANSI) в якості національного стандарту в США, а в даний час знайшов світове визнання.

В основі стандарту лежить процесний підхід до управління проектами. Загальна безліч можливих процесів представимо у вигляді тривимірного простору. По осях координат відкладені т вимірювання, які згадуються в рамкових стандартах. Можуть бути запропоновані та інші, наприклад рівні управління, календарні періоди. Кожна точка цього простору є елементарний процес управління. наприклад, "Планування ризиків на стадії впровадження системи".

Вибрані елементарні процеси утворюють процедури управління проектами, які можуть бути побудовані за "осьового" принципом.

Стандарт містить узагальнені принципи та підходи, які використовуються в області проектного менеджменту, формалізовані і структуровані таким чином, щоб їх можна було використовувати в більшості проектів більшості випадків. Детально описуються дев'ять областей знань, пов'язаних з управлінням проектами [1]:

- управління інтеграцією проекту (Project Integration Management);
- управління змістом проекту (Project Scope Management);
- управління термінами проекту (Project time Management);
- управління вартістю проекту (Project Cost Management);
- управління якістю проекту (Project Quality Management);
- управління людськими ресурсами проекту (Project Human Resource Management);
- управління взаємодією в проекті (Project Communications Management);
- управління ризиками проекту (Project Risk Management);
- управління контрактами проекту (Project Procurement Management).

Джером при реалізації проекту на тому чи іншому етапі. Застосований орієнтований підхід в управлінні проектами, використовуваний в стандарті, передбачає чіткий, формальний опис вхідних документів і даних, необхідних менеджеру для реалізації процесу, методів і засобів, які він може використовувати при його реалізації, і переліку вихідних документів процесу.

2. IPMA Competence Baseline (ICB) є міжнародним нормативним документом, що визначає систему міжнародних вимог до компетентності менеджерів проектів [3]. Цей стандарт розроблений міжнародної асоціацією IPMJI (International Project Managers Association). На його основі проводиться розробка національних систем вимог до компетентності фахівців в країнах, які є членами IPMA. Національні системи вимог повинні відповідати ICB IPMA і офіційно затверджуватис відповідними уповноваженими органами IPMA. для 32 країн - членів IPMA він є основою для розробки національних зводів знань; в даний час затверджені національні зводи знань, відповідні ICB, мають 16 країн.

ICB, на відміну від PMBOK, дотримується компетентнісного, діяльнісного підходу, тобто визначає області кваліфікації та компетентності в управлінні проектами, а також принципи оцінки кандидата на отримання сертифіката. ICB містить 42 елемента (28 основних і 14 додаткових), що визначають області вимог до знань, майстерності та професійного досвіду в менеджменті проектів.

ICB видано англійською, німецькою та французькою мовами. основою для нього послужило кілька національних розробок: Body of 'Knowledge of APM (Великобританія); Beurteilungsstruktur, VZPM (Швейцарія); PM-Kanon, PMZERT / GPM (Німеччина); Criteres d'analyse, AFITEP (Франція).

Кожна що входить в IPMA національна асоціація відповідальна за розробку та затвердження власних Національних вимог по компетентності (National Competence Baseline - NCB) з посиланням на ICB і відповідно до нього, а також з урахуванням національних особливостей і культури.

Національні вимоги оцінюються спеціальним Комітетом IPMA на відповідність ICB і основним критеріям сертифікації відповідно до стандарту EN45013.

3. Стандарт ISO 10006. Звернення до питань ефективності проектного управління об'єктивно виявило гостру потребу в розробці системи управління якістю проекту. При цьому особливого значення поряд з вимогами до якості кінцевого продукту стало надаватися якості процесів проекту, відсутність належної уваги до яких призводило до НЕ менш значущих негативних наслідків безпосередньо для створюваного продукту [4].

Стандарт ISO 10006 є основоположним документом із серії стандартів розглянутого профілю, підготовленим технічним комітетом ISO / TC 176 "Управління якістю і забезпечення якості" Всесвітньої федерації національних органів стандартизації (члени ISO).

Основний упор зроблений на принцип ефективності проектування оптимального процесу і контролю цього процесу, а не на контролі кінцевого результату.

У цій серії стандартів процеси згруповані у дві категорії. До першої категорії віднесені процеси, пов'язані із забезпеченням продукту проекту (Проектування, виробництво, перевірка). Опису останніх присвячений стандарт ISO 9004-1. Друга категорія охоплює безпосередньо процеси управління проектом і представлена стандартом ISO 10006.

Даний стандарт охоплює десять груп процесів управління проектом. Перша група являє процес розробки стратегії, який фокусує проект на задоволення потреб замовника і визначає напрямок ходу робіт. Друга група охоплює управління взаємозв'язками процесів. Решта вісім груп - це процеси, пов'язані з проектним завданням, термінами, витратами, ресурсами, кадрами, інформаційними потоками, ризиком і матеріально-технічним постачанням (закупівлями).

Міжнародний стандарт ISO 10006 орієнтований на проекти самого широкого спектра - малі і великі, короткострокові і довгострокові, для різних навколишніх умов. Він безотносителен до типу проектованого продукту (включаючи технічні засоби, програмне забезпечення, напівфабрикати, послуги або їх поєднання). Це означає, що закладені в ньому рамкові вимоги вимагають подальшої адаптації даного керівництва до конкретних умов розробки і реалізації окремого проекту [4].

Стандарт запозичує ключові визначення з ICO 8402, включаючи такі терміни, як проект, продукт проекту, план проекту, учасник проекту, процес, оцінка ходу робіт. Для всіх процесів управління проектом (планування, організація, моніторинг і контроль) застосовуються процеси і завдання менеджменту якості.

На основі міжнародних стандартів розробляються і національні стандарти управління проектами. Відзначимо, що в Україні національний стандарт відсутній.

4. Стандарти зрілості управління проектами, теж набувають функції міжнародних. У 2004 р PMI був випущений стандарт оцінки рівня зрілості організації з управління проектами OPM3 (Organization Project Management

Maturity Model), що містить методологію визначення стану управління проектами в організації [4].

Термін «організаційна зрілість з управління проектами» описує здатність організації відбирати проекти та управляти ними таким чином, щоб це максимально ефективно підтримувало досягнення стратегічних цілей компанії.

ОРМЗ - це стандарт, який представляє собою всебічний підхід, який допомагає організаціям оцінювати і розвивати свої можливості по ефективної реалізації проектів. Він є свого роду ключем до організаційної зрілості управління проектами і містить три взаємопов'язаних елемента:

елемент «знання» (knowledge) являє собою сотні кращих практик з управління проектами, що характеризують ті чи інші рівні організаційної зрілості управління проектами;

елемент «оцінка» (assessment) є інструментом, що допомагає організаціям оцінити поточну зрілість управління проектами та визначити галузі поліпшення;

якщо організація приймає рішення розвивати практики управління проектами і переходити на нові більш високі рівні зрілості, то в справу вступає елемент «покращення» (improvement), який допомагає компаніям побудувати схему розвитку управління проектами таким чином, щоб забезпечити максимально ефективне досягнення своїх стратегічних цілей.

Основне призначення ОРМЗ - бути стандартом для корпоративного управління проектами та організаційної зрілості з управління проектами. Основна відмінна риса ОРМЗ - це наявність унікальної бази даних, яка містить сотні кращих практик, опис тисяч ключових факторів успіху, результатів та іншої інформації, що характеризує розвиток зрілості управління проектами в організації [4].

ОРМЗ спроектований таким чином, щоб бути легким в розумінні і використанні, масштабується, гнучким і налаштованим на споживача.

Грунтуючись на базі ОРМЗ як стандарту управління проектами, організація може успішно перейти до такого стану, коли проекти будуть

досягати поставлених цілей в рамках бюджету, термінів і, що більш важливо, переслідуючи корпоративні стратегічні цілі.

Специфіка IT-проектів знаходить відображення також у специфічній методології управління проектами:

- CMMi;
- Microsoft Solution Framework;
- Rational Unified Process.

### 1.2. Аналіз процесів управління проектами

Розбиваючи життєвий цикл проекту на фази з проміжними результатами (рис. 1.1.), ми, тим самим, робимо його більш керованим, знижуючи ступінь невизначеності від фази до фази. Більш того, деяка фаза може вилитися в окремий підпроект. Наприклад, на фазі визначення концепції проекту можуть знадобитися глибокі маркетингові дослідження ринку, що можна виділити в окремий проект маркетингового аналізу зі своїми власними фазами [8].

Для того, щоб провести проект по фазах до результату, необхідно виконати деякі серії дій. Причому в кожному проекті виконуються схожі процеси, які не залежать від предметної області. Такими загальними для всіх проектів процесами зі схожим змістом є ініціація, планування, виконання, управління і завершення проекту. Їх взаємозв'язок показана на рис. 1.1. - стрілками вказані напрямки потоків інформації.

Як видно, за процесом ініціації проекту слідують процеси планування і виконання проекту. Процеси управління можуть повертатися до процесів планування, якщо не досягнуть кінцевого результату який задовольняє цілі і обмеженням проекту. процеси завершення закривають проект.

Інтегративний характер управління проектом вимагає, щоб група процесів моніторингу та контролю взаємодіяла з іншими групами процесів, як показано на рис. 1.1. Процеси моніторингу і контролю здійснюються в той же самий час, що і процеси, що входять в інші групи процесів. Таким чином, процес

моніторингу та контролю зображений як «фонова» група процесів для інших чотирьох груп.

Ступінь взаємодії розглянутих груп процесів протягом життєвого циклу проекту різний. У реальності вони накладаються один на одного, як показано на рис1.1.

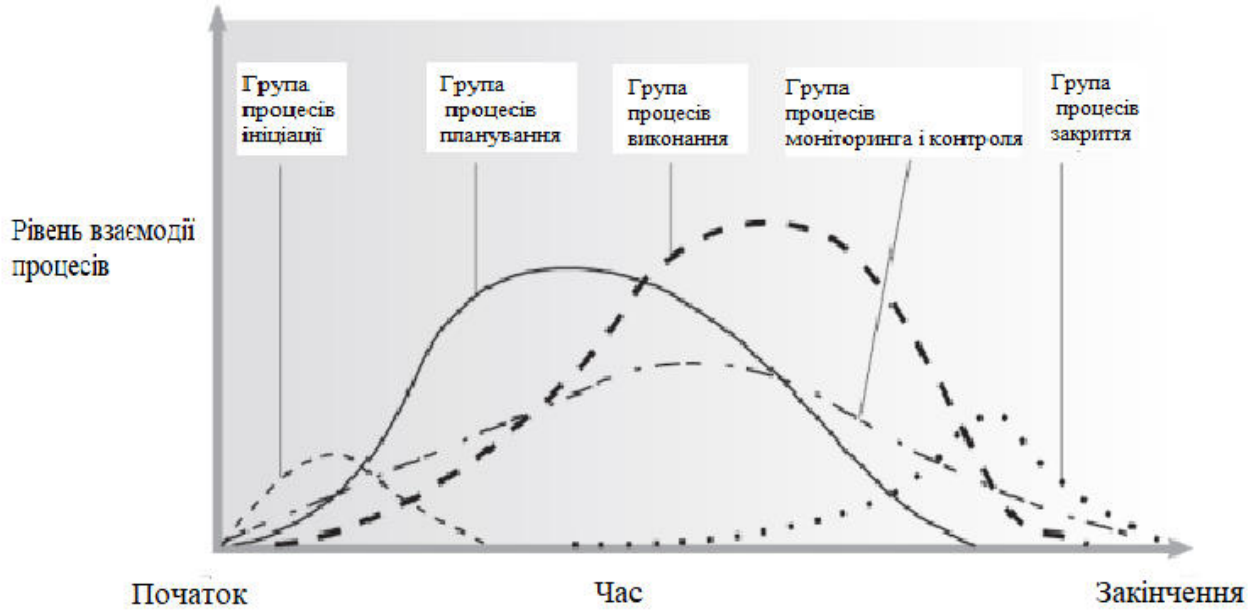


Рис.1.1. Взаємодія груп процесів [4].

Зміст процесів управління проектами Ініціація (1) - визначення ділової потреби в проекті і його авторизація, а саме:

- вибір проекту і визначення ділових потреб;
- збір інформації;
- визначення цілей проекту;
- визначення обмежень і припущень;
- розробка опису продукту;
- визначення обов'язків менеджера проекту;
- визначення вимог до людських ресурсів (кадри, кваліфікація);
- оціночне визначення ресурсів;
- остаточне доопрацювання статуту проекту і призначення менеджера проекту.

Процеси планування (2) спрямовані на розробку планів по складовим проекту (розклад, вартість, бюджет, якість, персонал, ризики, взаємодія, контракти та ін.) і їх інтеграцію в цілісний, узгоджений документ - План проекту. Як показано на рис.1.1, планування - це процес, що постійно повторюється протягом усього життєвого циклу проекту. [4]

У плануванні виділяють основні процеси, присутні завжди і виконуються в строго визначеному порядку, і допоміжні процеси, залежать від характеру проекту. На рис.1.2. показаний склад і зв'язку процесів планування. Коротко пояснимо кожен процес.

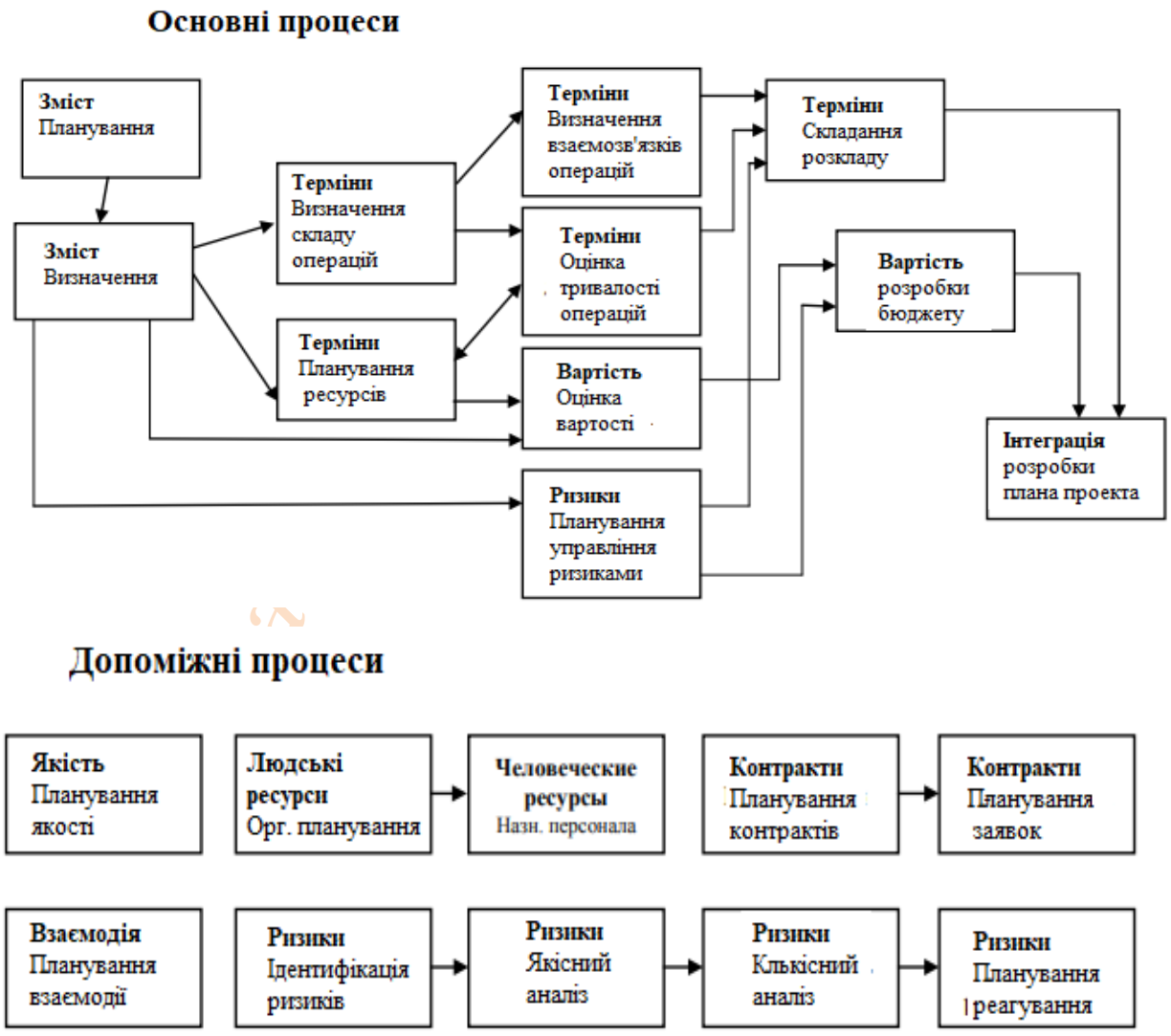


Рис.1.2. Основні і допоміжні процеси [4]

Планування змісту - на основі статуту проекту і інших вхідних документів, складається документ, що описує:

а) уточнений опис продукту і результатів поставки;

б) класифікацію можливих змін і спосіб їх виявлення;

в) порядок оцінки та включення змін в проект.

Визначення змісту - розбиття, декомпозиція цілей проекту на менші і більш керовані частини (підцілі). Глибина декомпозиції повинна забезпечувати можливість призначення закінчених груп робіт і виконавців на ці частини. Результатом визначення змісту є ІСР (Ієрархічна Структура робіт), англ. WBS (Work Breakdown Structure).

Визначення складу операцій - підготовка переліку всіх операцій, виконуваних за проектом, і уточнення ІСР. Операції, що не включені в уточнену ІСР, вважаються включеними в проект і не підлягають виконання.

Планування ресурсів - визначення потреби (складу, кількості) в людських і матеріальних ресурсах, необхідних для виконання операцій проекту.

Визначення взаємозв'язків операцій - виявлення взаємозв'язків і взаємозалежностей операцій, побудова мережевих діаграм робіт проекту. Частина операцій пов'язана між собою жорсткою логікою, інші операції можуть виконуватися в довільній послідовності, тобто пов'язані м'якою логікою.

Оцінка тривалості операцій - встановлення кількості одиниць часу на операції проекту, обчислення резервів часу і критичного шляху з мінімальної гнучкістю за часом.

Оцінка вартості - кількісна оцінка можливих витрат на залучені ресурси, складання кошторису і плану управління вартістю. Планування управління ризиками - встановлення підходу і заходів (коли, як і що робити) при загрозі або настання небажаних і незапланованих подій і відхилень, з метою їх запобігання або ефективного реагування.

Складання розкладу - аналіз даних про послідовності і тривалості операцій і необхідних ресурсів з метою створення розкладу виконання проекту.

Розробка бюджету - визначення кошторисної вартості по окремих пакетам робіт і проекту в цілому.

Розробка плану проекту - інтеграція даних попередніх процесів і складання узгодженого плану проекту - в одному примірнику або зібрання документів.

Допоміжні процеси планування встановлюють стандарти якості, розподіл ролей і відповідальності, інформаційні потреби учасників і способи взаємодії, виявляють ризики і наслідки їх впливу на цілі проекту і т.д.

Виконання плану проекту - є безпосереднє виконання складових його операцій. Допоміжні процеси забезпечують гарантії якості, комплектацію / розподіл робіт і інформації, проведення нарад про хід робіт і ідентифікацію змін, розвиток навичок і знань команди, збір пропозицій постачальників, управління відносинами з постачальниками цілому.

Допоміжні процеси забезпечують посвідчення правильності виконання робіт, фіксацію і прийняття змін, контроль і зміну розкладу і бюджету, відповідність стандартам якості і усунення причин його зниження, відстеження та виявлення ризиків, оцінку заходів щодо зниження ризиків.

Процеси завершення (5) впорядковують закриття проекту і складаються з процесів закриття контрактів і адміністративного завершення, а саме:

- перевірка та тестування кінцевого продукту;
- остаточні розрахунки з усіма учасниками проекту, фінансове закриття;
- остаточне оновлення документів проекту;
- завершення звіту про виконання проекту;
- збір, інтеграція накопичених знань і формування архіву проекту;
- офіційне приймання проекту замовником, передача і запуск в експлуатацію;
- звільнення задіяних ресурсів.

Грамотне та ефективне виконання перерахованих процесів вимагає від менеджера проекту знань в наступних областях:

1. Управління інтеграцією проекту.
2. Управління змістом проекту.
3. Управління термінами проекту.
4. Управління вартістю проекту.
5. Управління якістю проекту.
6. Управління людськими ресурсами проекту.
7. Управління комунікаціями проекту.
8. Управління ризиками проекту.
9. Управління закупівлями проекту.
10. Управління зацікавленими особами.

1.3. Моделі управління проектами

## 1.3. Моделі управління проектами

Каскадна модель (англ. Waterfall model, іноді перекладають як модельи«Водоспад») - модель процесу розробки програмного забезпечення, в якійипроцес розробки виглядає як потік, що послідовно проходить фазиианалізу вимог, проектування, реалізації, тестування, інтеграції таипідтримки. Як джерело назви часто вказують статтю,иопубліковану У. У. Ройсом (W. W. Royce) в 1970 році; при тому, що сам Ройсивикористовував ітеративну модель розробки [5].

У 1970 році в своїй статті Ройс описав у вигляді концепції те, що зараз прийнято називати «каскадна модель», і обговорював недоліки цієї моделі. Там же він показав як ця модель може бути доопрацьована до итеративной моделі.

В оригінальній каскадної моделі Ройса, такі фази йшли в такому порядку [8]:

- Визначення вимог
- Проектування
- Конструювання (також «реалізація» або «кодування»)
- Втілення
- Тестування та налагодження (також «верифікація»)

- Інсталяція
- Підтримка

Перехід від однієї фази до іншої відбувається тільки після повного і успішного завершення попередньої дотримуючись каскадної моделі, розробник переходить від однієї стадії до іншої строго послідовно. Спочатку повністю завершується етап «визначення вимог », в результаті чого виходить список вимог до ПО. Після того як вимоги повністю визначені, відбувається перехід до проектування, в під час якого створюються документи, що детально описують для програмістів спосіб і план реалізації зазначених вимог. Після того як проектування повністю виконано, програмістами виконується реалізація отриманого проекту. На наступній стадії процесу відбувається інтеграція окремих компонентів, що розробляються різними командами програмістів. Після того як реалізація і інтеграція завершені, проводиться тестування і налагодження продукту; на цій стадії усуваються всі недоліки, що з'явилися на попередніх стадіях розробки. Після цього програмний продукт впроваджується і забезпечується його підтримка - внесення нової функціональності і усунення помилок.

Тим самим, каскадна модель має на увазі, що перехід від однієї фази розробки до іншої відбувається тільки після повного і успішного завершення попередньої фази, і що переходів назад або вперед або перекриття фаз – НЕ відбувається.

Проте, існують модифіковані каскадні моделі (включаючи модель самого Ройса), що мають невеликі або навіть значні варіації описаного процес [8].

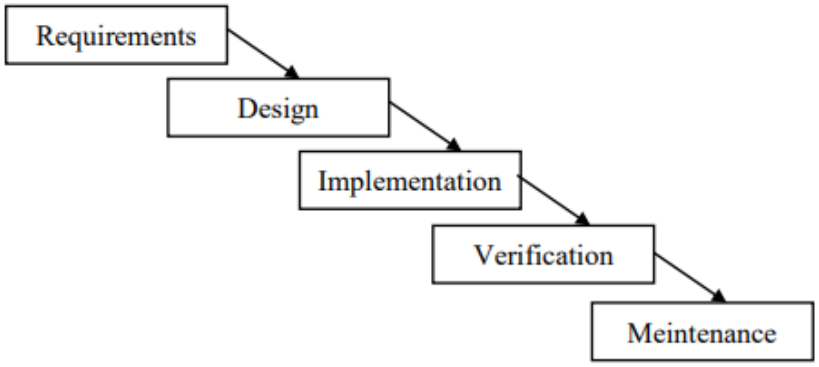


Рис. 1.3. Каскадна модель управління проектами [8].

Методику «Каскадна модель» досить часто критикують за недостатню гнучкість і оголошення самоціллю формальне управління проектом на шкоду термінами, вартості та якості. Проте, при управлінні великими проектами формалізація часто була дуже великою цінністю, так як могла кардинально знизити багато ризиків проекту і зробити його більш прозорим [5].

Тому навіть в PMBOK 3-ої версії формально була закріплена лише методика «Каскадної моделі» і не були запропоновані альтернативні варіанти, відомі як итеративное ведення проектів [1].

Починаючи з PMBOK 4-й версії вдалося досягти компромісу між методологами, прихильними формальному і поступальному управління проектом, з методологами, що роблять ставку на гнучкі ітеративні методи.

Таким чином, починаючи з 2009 року, формально Інститутом управління проектами (PMI) пропонується як стандарт гібридний варіант методології управління проектами, що поєднує в собі як плюси від методики «Водограю», так і досягнення ітеративних методологів.

Модель управління проектами PRINCE2 PRojects IN Controlled Environments 2 (PRINCE2) являє собою структурований метод управління проектами, схвалений урядом Великобританії в якості стандарту управління проектами в соціальній сфері [5].

Методологія PRINCE2 включає в себе підходи до менеджменту, контролю та організації проектів.

Спочатку метод був розроблений в 1989 році Central Computer and Telecommunications Agency (CCTA) у Великобританії як стандарт для керівництва проектами в сфері інформаційних технологій. В даний час широко використовується і є «de facto» стандартом для керівництва проектами у Великобританії.

PRINCE2 є структурований підхід до управління проектами, т. е. є метод для управління проектами в рамках чітко визначеної структури. PRINCE2 описує процедури для координації діяльності команди проекту при розробці і контролю над проектом, а також процедури, які використовуються при зміні

проекту або якщо є істотні відхилення від початкового плану. У методі кожен процес визначається зі своїми основними входами і виходами, і з конкретними цілями і заходами, які будуть здійснюватися, що дає автоматичний контроль будь-яких відхилень від плану. За рахунок поділу процесів на керовані етапи, метод дає можливість ефективного управління ресурсами.

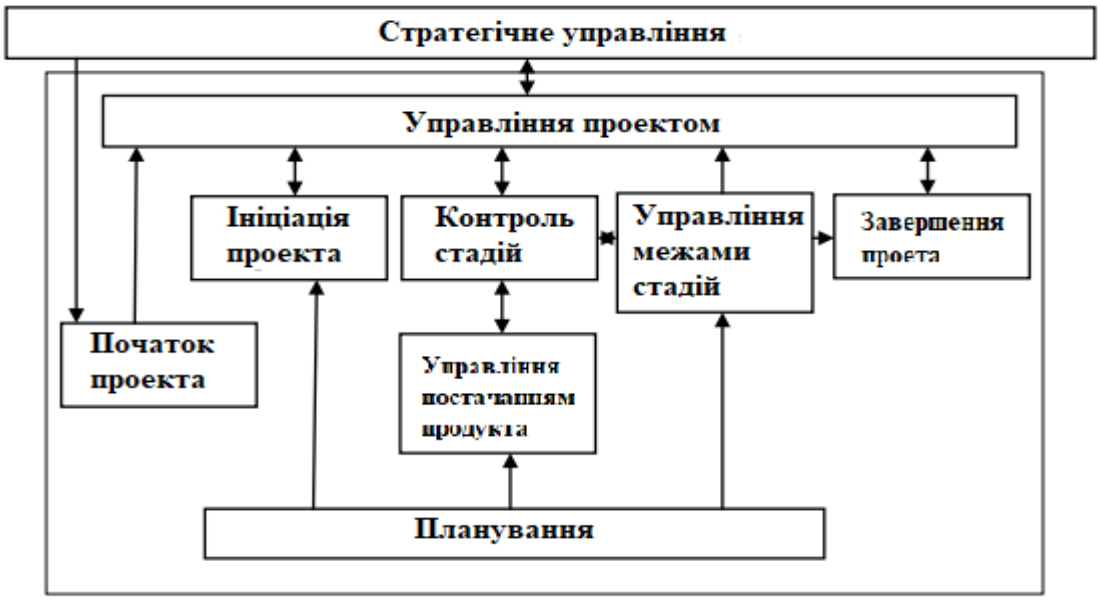


Рис. 1.4. Діаграма процесів методу PRINCE2 [5]

До недоліків можна віднести відсутність будь-якого регламентування з боку методології підходів до управління контрактами поставок, учасниками проекту та іншими процесами, які були винесені творцями за рамки. Вважається, що кожен менеджер проекту вибирає власні методи і підходи до цієї роботи.

Діаграма показує процеси методу PRINCE2. стрілки показують направлення інформаційних потоків.

Початок проекту. Як від початкової ідеї проекту (відображеної в мандаті на проект) перейти до безпосередньо реалізації цих ідей. Створюється організація - мінімум призначаються керівник проекту і голова комітету проекту.

Формулюється короткий опис проекту (project brief) і підхід до його реалізації. Детально планується стадія запуску проекту. Ініціація проекту.

Проводиться планування проекту, включаючи план якості. Створюється економічне обґрунтування проекту (Business Case) і відкривається журнал ризиків, проводиться оцінка ризиків проекту. Плануються віхи, точки контролю проекту.

Управління проектом. Тут зосереджені терміни прийняття рішень комітетом проекту (в тому числі щодо дострокового завершення проекту) та ситуаційне управління по значним проблемам і відхиленям.

Контроль стадій. Безпосередня робота керівника щоденного управління проектом - видача і приймання завдань, фіксація складнощів і ризиків, прийняття рішення про ескалацію, звітність перед комітетом.

Управління виробництвом продукту. Заходи, які виконавці та робочі групи повинні зробити для визначення обсягів роботи, звіти про прогреси і передачу виконаної роботи. Контроль кордонів стадій. Тут відбувається аналіз виконання плану стадії, проміжне планування наступної стадії, запасних планів, огляд ризиків і бізнес-плану. Служить для переходу між стадіями.

Завершення проекту. Як закрити проект, як управляти наступними діями, як розбирати огляди переваг проекту.

Планування. Як планувати, незалежно від того, коли здійснюється планування.

Менеджер проекту в традиційному розумінні.

Комітет проекту (project board), перед яким регулярно звітує менеджер складається з 3х чоловік - замовника, головного користувача і головного фахівця. Рада проекту відповідальна за прийняття стратегічних рішень.

Менеджер проекту зобов'язаний відстежувати можливі проблеми і пропонувати раді альтернативні рішення. Рада вирішує - який шлях краще.

Служба project assurance (аналог проектного офісу), мета якої надавати незалежну думку про проект з точки зору тих же трьох груп людей - замовників, користувачів і фахівців (в предметній області).

Служба готує три звіти -

- business report (звіт про фінансовий стан проекту і вигідності проекту в цілому),
- user report (наскільки добре виконуються вимоги користувачів),
- technical report (наскільки хороший проект в технологічному плані – туди чи він рухається).

Є служба адміністративної підтримки (адміністратори проектів і т. п.), відповідальна за проведення зустрічей, доведення потрібної інформації до всіх її адресатів, збереження проектної інформації і т. п. У разі маленьких проектів це робить менеджер проекту.

Ще одним класом моделей є гнучкі методології управління проектами.

Гнучка методологія розробки (англ. Agile software development, agileметоди) - серія підходів до розробки програмного забезпечення, орієнтованих на використання ітеративної розробки, динамічне формування вимог і забезпечення їх реалізації в результаті постійної взаємодії всередині самоорганізованих робочих груп, що складаються з фахівців різного профілю. Існує кілька методик, що відносяться до стратегічного управління класу гнучких методологій розробки, зокрема екстремальне програмування, DSDM, Scrum, FDD [9].

Застосовується як ефективна практика організації праці невеликих груп (які роблять однорідну творчу роботу) в поєднанні з управлінням ними комбінованим (ліберальним і демократичним) методом.

Більшість гнучких методологій націлені на мінімізацію ризиків шляхом відомої розробки серії коротких циклів, званих ітераціями, які зазвичай тривають два-три тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі і включає всі завдання, необхідні для видачі міні-приросту по функціональності: планування, аналіз вимог, проектування, програмування, тестування і документування. Хоча окрема ітерація, як правило, недостатня для випуску нової версії продукту, мається на увазі, що гнучкий програмний проект готовий до випуску наприкінці кожної ітерації. Після закінчення кожної ітерації команда виконує переоцінку пріоритетів розробки.

Agile-методи роблять наголос на безпосереднє спілкування віч-на-віч. Більшість agile-команд розташовані в одному офісі, іноді званому англ. bullpen. Як мінімум, вона включає і «замовників» (англ. Product owner - замовник або його повноважний представник, який визначає вимоги до продукту; цю роль може виконувати менеджер проекту, бізнес-аналітик або клієнт). Офіс може також включати тестувальників, дизайнерів інтерфейсу, технічних письменників і менеджерів.

Основний метрикою agile-методів є робочий продукт. Віддаючи перевагу безпосередньому спілкуванню, agile-методи зменшують обсяг письмової документації в порівнянні з іншими методами. Це призвело до критики цих методів як недисциплінованих.

У лютому 2001 в штаті Юта США був випущений «Маніфест гнучкої методології розробки програмного забезпечення ». Він був альтернативою керованим документацією, «великоваговим» практикам розробки програмного забезпечення, таким як «метод водоспаду», який був золотим стандартом розробки в той час. Даний маніфест був схвалений і підписаний представниками методологій: екстремального програмування, Crystal Clear, DSDM, Feature driven development, Scrum, Adaptive software development, Pragmatic Programming. Гнучка методологія розробки використовувалася багатьма компаніями і до прийняття маніфесту, однак входження Agile-розробки в маси відбулося саме після цієї події [9].

Agile - сімейство процесів розробки, а не єдиний підхід в розробці програмного забезпечення, і визначається Agile Manifesto. Agile НЕ включає практик, а визначає цінності і принципи, якими керуються успішні команди.

Agile Manifesto розроблений і прийнятий 11 - 13 лютого 2001 року на лижному курорті The Lodge at Snowbird в горах Юти. Agile Manifesto містить 4 основні ідеї та 12 принципів рис. 1.5. Примітно, що Agile Manifesto не містить практичних порад [9].


Рис.1.5. Основні 4 ідеї Agile Manifesto

Основні ідеї:

- люди і взаємодія важливіше процесів та інструментів;
- працює продукт важливіше вичерпної документації;
- співпраця з замовником важливіше узгодження умов контракту;
- готовність до змін важливіше проходження попереднім планом.

Принципи, які роз'яснює Agile Manifesto [9].:

- задоволення потреб замовника важливіше суперництва і внутрішньої ієрархії. Це досягається безперервною роботою і миттєвим виправленням помилок.
- швидка реакція на зміни вимог до продукту.
- безперервний темп з повторюваною швидкістю роботи.
- підтримка залучених співробітників. Мотивована команда виконує роботу краще, ніж незадоволені умовами праці фахівці.
- тісна комунікація з замовником і всіма членами команди розробників протягом усього проекту.
- простота, як основа роботи.

- маркетинговий план повинен оновлюватися якомога частіше - раз на два тижні або півтора місяця.
- відмова від умовностей і суб'єктивних думок на користь детального вивчення замовника, впровадження змін та аналізу результатів.
- кілька короткострокових тестувань актуальніше одного довгострокового експерименту.

RAD (від англ. Rapid application development - швидка розробка додатків) - концепція створення засобів розробки програмних продуктів, що приділяє особливу увагу швидкості і зручності програмування, створення технологічного процесу, що дозволяє програмісту максимально швидко створювати комп'ютерні програми. Практичне визначення: RAD – це життєвий цикл процесу проектування, створений для досягнення більш високої швидкості розробки і якості ПО, ніж це можливо при традиційному підході до проектування. З кінця XX століття RAD набула широкого поширення і схвалення. Концепцію RAD також часто пов'язують з концепцією візуального програмування [5].

Концепція RAD стала відповіддю на незграбні методи розробки програм 1970-х і початку 1980-х років, такі як «модель водоспаду» (англ. Waterfall model).

Ці методи передбачали настільки повільний процес створення програми, що часто навіть вимоги до програми встигали змінитися до закінчення розробки. Засновником RAD вважається співробітник IBM Джеймс Мартін, який в 1980-х роках сформулював основні принципи RAD, грунтуючись на ідеях Баррі Бойм і Скотта Шульца. А в 1991 році Мартін опублікував відому книгу, в якій детально виклав концепцію RAD і можливості її застосування. В даний час RAD стає загальноприйнятою схемою для створення засобів розробки програмних продуктів.

RAD передбачає, що розробка ПО здійснюється невеликою командою розробників за термін близько трьох-чотирьох місяців шляхом використання інкрементного прототипирования із застосуванням інструментальних засобів

візуального моделювання та розробки. Технологія RAD передбачає активне залучення замовника вже на ранніх стадіях - обстеження організації, вироблення вимог до системи. Останнє з зазначених властивостей передбачає повне виконання вимог замовника як функціональних, так і не функціональних, з урахуванням їх можливих змін в період розробки системи, а також отримання якісної документації, забезпечує зручність експлуатації і супроводу системи. Це означає, що додаткові витрати на супровід відразу після поставки будуть значно менше. Таким чином, повний час від початку розробки до отримання прийнятного продукту при використанні цього методу значно скорочується.

Технологію RAD доцільно застосовувати, коли чітко визначені деякі пріоритетні напрямки розробки проекту [5].

1. Необхідно виконання проекту в стислі терміни. швидке виконання проекту дозволяє створити систему, що відповідає вимогам сьогоднішнього дня. Якщо система проектується довго, то дуже висока ймовірність, що за цей час істотно зміняться фундаментальні положення, що регламентують діяльність організації, тобто, система морально застаріє ще до завершення її проектування.

2. Нечітко визначені вимоги до ПЗ. У більшості випадків замовник дуже приблизно уявляє собі роботу майбутнього програмного продукту і не може чітко сформулювати всі вимоги до ПЗ. Вимоги можуть бути взагалі не визначені до початку проекту або можуть змінюватися по ходу його виконання.

3. Проект виконується в умовах обмеженості бюджету. Розробка ведеться невеликими RAD-групами в короткі терміни, що забезпечує мінімум трудовитрат і дозволяє вписатися в бюджетні обмеження.

4. Інтерфейс користувача (GUI) є головний фактор. Немає сенсу змушувати користувача малювати картинки. RAD-технологія дає можливість продемонструвати інтерфейс в прототипі, причому досить скоро після початку проекту.

5. Можливо розбиття проекту на функціональні компоненти. Якщо передбачувана система велика, необхідно, щоб її можна було розбити на дрібні

частини, кожна з яких має чітку функціональність. Вони можуть випускатися послідовно або паралельно (в останньому випадку залучається кілька RAD-груп).

6. Низька обчислювальна складність ПЗ. RAD-технологія не є універсальною, тобто її застосування доцільно не завжди. Наприклад, в проектах, де вимоги до програмного продукту чітко визначені і не повинні змінюватися, залучення замовника в процес розробки не потрібно і більш ефективною може бути ієрархічна розробка (каскадний метод). Те ж стосується проектів, ПЗ, складність яких визначається необхідністю реалізації складних алгоритмів, а роль і обсяг призначеного для користувача інтерфейсу невеликий.

Принципи RAD технології спрямовані на забезпечення трьох основних її переваг - високої швидкості розробки, низької вартості і високої якості. Досягти високої якості програмного продукту досить непросто і одна з головних причин виникаючих труднощів полягає в тому, що розробник і замовник бачать предмет розробки (ПЗ) по-різному.

- Інструментарій повинен бути націлений на мінімізацію часу розробки.
- Створення прототипу для уточнення вимог замовника.
- Циклічність розробки: кожна нова версія продукту грунтується на оцінці результату роботи попередньої версії замовником.
- Мінімізація часу розробки версії, за рахунок перенесення вже готових модулів і додавання функціональності в нову версію.
- Команда розробників повинна тісно співпрацювати, кожен учасник повинен бути готовий виконувати кілька обов'язків.

Управління проектом має мінімізувати тривалість циклу розробки.

Принципи RAD застосовуються не тільки при реалізації, а й ширюються на всі етапи життєвого циклу, зокрема на етап обстеження організації, побудови вимог, аналіз і дизайн.

Технологія швидкої розробки додатків (RAD) дозволяє забезпечити:

- швидкість просування програмного продукту на ринок;

- інтерфейс, що влаштовує користувача;
- легку адаптованість проекту до мінливих вимог;
- простоту розвитку функціональності системи.

## ВИСНОВКИ ДО РОЗДІЛУ 1

У першому розділі кваліфікаційної роботи магістра зазначено, що управління проектами в області інформаційних технологій за останній час завоювало визнання як найкращий метод планування та управління реалізацією інвестиційних проектів. За американськими оцінками застосування методології управління проектами забезпечує високу надійність досягнення цілей проекту і на 10-15% скорочує витрати на його реалізацію.

Зазначено, що проект - це тимчасове підприємство, призначене для створення унікальних продуктів або послуг. В даному контексті "тимчасове" означає, що у кожного проекту є початок і неодмінно настає завершення, коли досягаються поставлені цілі, або приходить розуміння, що ці цілі не можуть бути досягнуті

Визначено, що управління проектами базується на системному підході, що дає можливість декомпозиції і структуризації проекту будь-якої складності при прийнятті рішень в складних умовах (ситуаціях). Відмінна особливість методології управління проектами полягає в зосередженні прав і відповідальності за досягнення цілей проекту на одній людині або невеликій групі.

Слід зазначити, що міжнародні стандарти є повні системи, які включають, крім опису вимог до управління проектами, навчання, тестування, аудит, консалтинг та інші елементи. Всеохоплюючих міжнародних стандартів управління проектами поки не існує, але найбільш відомі такі нормативні документи.

Для того, щоб провести проект по фазах до результату, необхідно виконати деякі серії дій. Причому в кожному проекті виконуються схожі процеси, які не залежать від предметної області. Такими загальними для всіх

проектів процесами зі схожим змістом є ініціація, планування, виконання, управління і завершення проекту.

Наголошено, що сьогодні більшість гнучких методологій націлені на мінімізацію ризиків шляхом відомої розробки серії коротких циклів, званих ітераціями, які зазвичай тривають два-три тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі і включає всі завдання, необхідні для видачі міні-приросту по функціональності: планування, аналіз вимог, проектування, програмування, тестування і документування.

# РОЗДІЛ 2
# ВИБІР ГНУЧКИХ МЕТОДІВ УПРАВЛІННЯ ПРОЕКТАМИ В СФЕРІ ІТ

## 2.1. Особливості гнучких методів управління

Незважаючи на успішний досвід японських корпорацій до недавнього часу гнучкі методи управління проектами активно використовувалися лише при розробці програмного забезпечення. Однак, поступово керівники багатьох компаній по всьому світу стали приходити до розуміння застосовності ключових цінностей гнучких методів управління і в інших сферах діяльності і почали впроваджувати гнучкі практики управління в своїх організаціях [9].

Сьогодні під «agile» в широкому сенсі розуміється не тільки об'єднання ряду гнучких методів та інструментів управління, а й культурних зміни в організації, які відповідають єдиної філософії, базується на чотирьох ключових цінностях Agile Маніфесту. До таких культурних змін відноситься відмова від командно-адміністративних методів управління, розвиток культури довіри та взаємодопомоги, заохочення креативності та нестандартного мислення і т.д. У вузькому ж сенсі «agile» являє собою ряд практик і методів, характеризуються певними відмінностями [10].

З точки зору організації процесів головна відмінність гнучких методів полягає в особливості життєвого циклу проекту. Життєвий цикл проекту - це «набір фаз, через які проходить проект з моменту його ініціації до моменту закриття ».

Серед основних видів життєвого циклу проекту фахівці виділяють предикативний життєвий цикл, ітеративний життєвий цикл, інкрементальний життєвий цикл і їх різні комбінації [5].

Основна особливість інтелектуального життєвого циклу полягає в тому, що зміст, строки і вартість проекту розраховуються і фіксуються на найпершій стадії проекту і далі не змінюються.

Типовим прикладом інтелектуального життєвого циклу є каскадна модель або модель «водоспаду» (waterfall). Дана модель складається з ряду послідовних фаз, кожна з яких представляє собою певну групу операцій і процесів: ініціювання, планування, проектування, будівництва, випробування, передача. Кожна наступна фаза починається тільки після завершення попередньої.

Основні недоліки моделі пов'язані з низьким рівнем якості початкового варіанта продукту, відсутністю прозорості та неможливістю внесення значних змін до затвердженого проекту.

Проблеми з якістю пов'язані з тим, що фаза тестування продукту є однією з останніх і на неї часто не залишається часу і бюджету. Крім цього, в разі виявлення помилок на ранніх стадіях проект доводиться повертати на доопрацювання на один або кілька кроків назад, що сильно впливає на терміни реалізації. Якщо ж помилки не вдалося виявити, весь проект може виявитися провальним. Частково дана проблема пов'язана з тим, що кожен співробітник, що бере участь в створенні продукту, відповідальний тільки за свою частину роботи і може навіть не перетинатися з іншими виконавцями проекту. Також оскільки учасники проекту не контактують один з одним, буває деколи складно зрозуміти, яка частина проекту вже реалізована. Що стосується внесення змін до проекту, то вони вкрай небажані, оскільки для їх інтеграції доведеться заново запускати процес планування і стверджувати новий зміст проекту. [4]

Ітеративний життєвий цикл проекту повинен був вирішити деякі проблеми каскадних моделей. У даній моделі передбачається розробка продукту шляхом здійснення ряду повторюваних циклів - ітерацій. При цьому в ході кожної ітерації реалізуються операції з усіх груп процесів управління проектом, тобто одночасно здійснюється планування, розробка, тестування і аналіз. Для цього зазвичай створюються кроссфункціональні команди, члени яких перебувають в безпосередньому контакті один з одним і несуть колективну відповідальність за результати проекту. У такі команди можуть входити розробники, аналітики, тестувальники, системні адміністратори,

маркетинг, управління продуктом, управління каналами продажів і багато інших фахівців [8].

Горизонт планування обмежений однією ітерацією (спочатку є лише певне бачення кінцевого продукту), в зв'язку з цим, дана модель стійка до внесення змін в ході проекту. Крім того, оскільки кожна ітерація має фіксований термін (однаковий для всіх ітерацій), стає простіше регулювати дати випуску продукту на ринок (за рахунок регулювання обсягу виконуваних робіт).

Чудовим прикладом життєвого циклу проекту, який є одночасно і предиктивним, і ітеративним є спіральна модель, розроблена в 1988 році Баррі Боем. Спіральна модель була покликана вирішити проблеми каскадної моделі в частині більш простого внесення змін і підвищення якості продукту. Дана модель, як і раніше відрізнялася об'ємним попереднім плануванням, однак, на відміну від «Водоспаду», поділена на кілька ітерацій. Часто, протягом одної або декількох перших ітерацій розроблявся прототип продукту, щоб клієнт міг внести коригування в спочатку позначені вимоги [8].

У той же час залишалася проблема того, що клієнт отримував готовий продукт лише через 6-12 місяців після завершення всього проекту. Для вирішення цієї проблеми був розроблений інкрементний життєвий цикл. його суть полягає в послідовному збільшенні функціональності продукту, тобто кожна фаза проекту завершувалася створенням так званого готового результату, який несе певну цінність для клієнта. Процес влаштований таким чином, що в ході однієї або декількох перших ітерацій створюється мінімально життєздатний продукт (Minimum Viable Product), який потім удосконалюється і доповнюється в ході подальших ітерацій. Таким чином, вже через кілька ітерацій замовник мав на руках готовий продукт, що приносить прибуток.

Адаптивний життєвий цикл або гнучкі методи управління є прикладом ітеративного і інкрементного життєвого циклу проекту. Кожна ітерація може становити від двох до чотирьох тижнів, що дозволяє вносити постійні зміни в проект і коригувати розробку продукту. Команда знаходиться в постійному

контакті з замовником і всіма зацікавленими особами, і за результатами презентації продукту отримує від них зворотний зв'язок, який потім впроваджує в проект в ході подальших ітерацій. Зазвичай, в ході кожної ітерації здійснюється одночасно кілька процесів, хоча в ході попередніх циклів команда може більше зосереджуватися на плануванні. Терміни кожного проекту фіксуються в ході переговорів з клієнтом, тому команда не може змінювати загальне число ітерацій. Але вона може змінювати обсяг робіт (прибираючи другорядні або додаючи нові завдання), що виконуються в процесі кожній ітерації в залежності від уже досягнутого прогресу і отриманої

«Гнучкі» методології базуються на емпіричному управлінні, тобто на такому управлінні, де рішення приймаються, виходячи з проміжних результатів проекту. Крім того, дані результати є прозорими, що означає, що всі залучені в проект люди обізнані статусом робіт за проектом, кількістю змін і можливими проблемами. (Layton, 2012). Не можна не відзначити, що в разі IT-проектів емпіричне управління дозволяє швидко вносити коригування в програмний продукт, і це є великою перевагою в порівнянні з Водоспадної моделлю життєвого циклу [27].

Останні кілька років в США проводилося багато досліджень, спрямованих на вивчення переваг «гнучких» методологій в порівнянні з традиційною Водоспадної моделлю. Наприклад, згідно з сьомим щорічним державним дослідженням «гнучкої» розробки Інтернет-ресурсу Version One, 84% респондентів (розробників) визнали, що «гнучкі» підходи здаються їм більш ефективними, ніж традиційні підходи.

Крім того, 70% респондентів цього ж дослідження визнали, що IT-проекти, що впроваджуються за допомогою «гнучких» методологій, реалізовуються набагато швидше IT-проектів, ніж впроваджуваних за допомогою Водоспадної моделі життєвого циклу рис.2.1. [39].


Рис.2.1. Проекти впроваджуються швидше, якщо використовується "гнучка" методологія [39]

Як завершальний етап аналізу природи «гнучких» методологій і їх переваг нижче наведена таблиця 2.1, в якій ще раз вказані основні відмінності «гнучких» методологій і Водоспадної моделі, які не були відображені явно в «Маніфесті» [42]:

Таблиця 2.1

Ключові відмінності «гнучких» методологій і Водоспадної моделі

| | **«Гнучкі» методології** | **Водоспадна модель** |
|---|---|---|
| **Вимоги замовника** | Ітеративний збір | Деталізовані вимоги чітко визначаються до початку фази безпосереднього впровадження |
| **Вартість доопрацювань** | Низька | Висока |
| **Напрямок розробки** | Може бути змінено в будь-який момент | Фіксована |
| **Тестування** | Після кожної ітерації | Під час відповідної фази, наступній за впровадженням / розробкою |
| **Залученість замовника** | Висока | Низька |

## 2.2. Вибір методологій для моделювання процесу управління

В даний час існує безліч гнучких методологій, але найбільш популярними є «екстремальне програмування» (XP), «скрам», «ощадлива розробка» (lean), «розробка, керована функціональністю» (fdd) і «канбан». У таблиці 2.2 наведено порівняльний аналіз даних методологій (Abrahamsson, 2002) [10]:

Таблиця 2.2

Порівняльний аналіз "гнучких" методологій

| Назва | Ключові моменти | Унікальні особливості | Недоліки |
|---|---|---|---|
| «Екстремальне програмування» (XP) | Маленькі команди, щоденні наради, неформальний тип комунікації всередині команди, мінімум документації | Постійне коригування проекту для підвищення його ефективності і для адаптації до змін | Більше підходить для індивідуальних практик, ніж для глобального управління, так як в останньому випадку є ризик формування недисциплінованих команд |
| «Скрам» | Незалежна, невелика, самоорганізована команда розробників, довжина ітерації - 2-4 тижні, команда сама вирішує скільки часу їй потрібно для виконання завдання, неформальний тип комунікації всередині команди, щоденні наради, присутній тільки базова документація | Високий рівень комунікації і взаємодії всередині команди, чітко прописана формальна організація цієї методології, наявність «надсмоторщіка» за командою, повна орієнтація на вимоги замовника | Є ризик збільшення часу проекту за рахунок витрат «скрам»-заходів |
| «Ощадлива розробка» (lean) | Використання візуалізуючих інструментів, розробка через тестування, короткі ітерації | Головними є ті функції ПЗ, які цінні для замовника, має місце постійне мотивування команди | Рішення приймаються довго, недолік дисципліни, походить тільки для маленьких проектів |
| «Розробка, керована | П'яти кроковий процес, об'єктно- | Простота методу, об'єктне | Дана методологія фокусує свою увагу |

| Назва | Ключові моменти | Унікальні особливості | Недоліки |
|---|---|---|---|
| функціональністю» (fdd) | орієнтована розробка, дуже короткі ітерації (можуть досягати за тривалістю кілька годин) | моделювання | виключно на проектуванні та впровадженні, дуже мало уваги приділено саме розробці |
| «Канбан» | Самоорганізована команда, загублений сенс поняття «ітерація» - весь упор на завдання | Немає обмежень за часом виконання, зате є обмеження на число «роботи в даний момент», ефективно, коли невідомо, що може в подальшому знадобитися замовнику від ПО | Недолік дисципліни, затяжний характер |

FDD фокусується на п'яти кроковому підході, який базується на ідентифікації, розробці та впровадженні характеристик. У FDD також покладається, що частина роботи по проекту вже зроблено. В результаті, багато фаз проекту залишаються не до кінця реалізованими. «Канбан» може бути ефективний всередині конкретної організації, навіть всередині якогось відділу, в разі складних контрактних відносин «замовник-виконавець» його дуже важко реалізувати. «Ощадлива розробка» і «екстремальне програмування» теж втрачають свою ефективність, коли в проект залучено більш ніж одна організація: відсутність явного спостерігача за процесом може створити плутанину в обов'язках.

«Скрам» - найбільш популярна «гнучка» методологія, широко використовувана зарубіжними компаніями, такими як Yahoo !, PayPal, Nike, Google, SAP, GE для управління проектами (Deemer, 2007). Згідно шостому і сьомому щорічному державному дослідженням «гнучкої» розробки Інтернет-ресурсу Version One, методологія «скрам» використовується як метод «гнучкого» управління IT-проектом в кілька разів частіше, ніж інші методології (рис. 2.2.-2.3) [35].

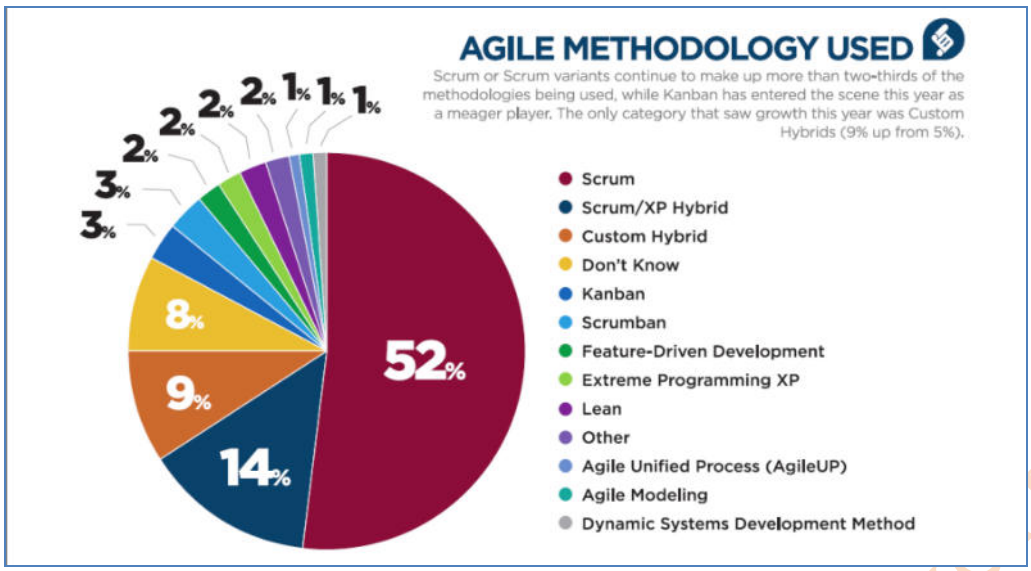


Рис.2.2. Популярність методології "скрам" в 2016 році.

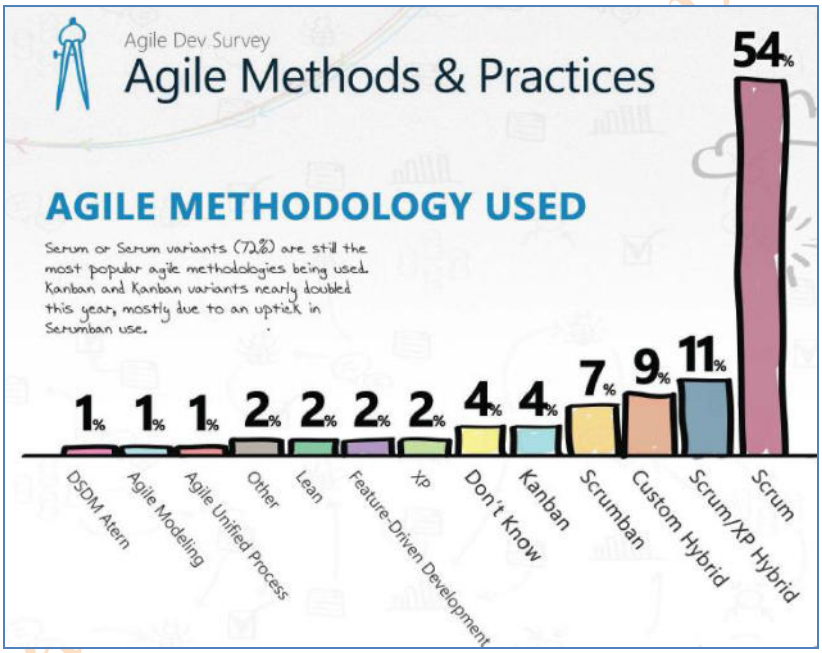


Рис.2.3. Популярність методології "скрам" в 2017 році

Методологія «скрам» націлена на взаємодію з замовником, і, не дивлячись на те, що команда розробників сама вирішує, які завдання вона буде виконувати протягом однієї ітерації, в даній методології присутній спостерігач (Скрам-майстер), який контролює дотримання «скрам» - процесу. Крім усього іншого, «скрам» добре документований: існує спеціальне «керівництво по скрам», перекладене на багато мов світу. Певна ступінь формалізованності в термінах «гнучкої» методології розробки є найкращою стратегією управління IT-проектом, коли в проекті беруть участь люди з різних організацій [35].

### 2.3. Опис методології «скрам»для управління проектами в IT сфері

Методологія «скрам» була вперше представлена в 1990-х роках Кеном Швабер і Джефом Сазерлендом у вигляді чітко задокументованого і формалізованого «керівництва по« скрам ».

«Скрам» - це гнучкий і легкий процес управління та контролю програмним забезпеченням і процесом розробки продукту в швидко мінливих умовах [35]. Дана методологія встановлює певні правила управління IT-проектом (розробкою або впровадженням інформаційних технологій), які ґрунтуються на можливості постійного коректування вимог і на внесення тактичних змін.

Перед початком проекту учасники обговорюють глобальні цілі і базову стратегію, спрямовану на досягнення цих цілей. Особа з боку замовника проекту представляє інтереси своєї компанії шляхом опису бізнес-функцій впроваджуваного або настроюваного програмного забезпечення. З боку виконавця визначається склад команди розробників і інші ресурси, після чого сторонами обговорюються приблизні рамки проекту, а так само дата початку першої ітерації. Ітерація в «скрам» отримала назву спринт, і вона триває два-чотири тижні. Всі спринти мають відносно фіксовану тривалість: це означає, що якщо запланована дата закінчення спринту, то при настанні цієї дати (з мінімальною похибкою) всі роботи даного спринту повинні закінчитися, незалежно від того, чи встигла команда проекту виконати завдання чи ні. Мінімальна похибка означає, що на практиці якісь затримки все ж можливі, але, тим не менш, вони не повинні перевищувати двох-трьох днів для спринту, тривалість якого дорівнює двом тижням.

Модель методології «скрам» складається з трьох головних компонентів: ролей, артефактів і заходів. У даній методології присутні три ролі: Власник продукту, Скрам-майстер і Скрам-команда.

Власник продукту найчастіше є замовником, який підтримує в актуальному стані список вимог до проекту. Чим краще і чіткіше власник продукту опише вимоги, тим менше буде питань у розробників, тим рідше буде

змінюватися функціонал програмного забезпечення з плином часу. Саме Власник продукту визначає пріоритет вимог, тобто сам замовник вирішує, які функції програмного забезпечення є найбільш терміновими і важливими для компанії на якийсь конкретний момент;

Скрам-майстер відповідальний за розуміння сутності всього «скрам» - процесу іншими учасниками. Скрам-майстер чимось схожий на традиційного менеджера проекту, але у нього є одна відмінність: він не віддає явних наказів і не встановлює терміни розробникам, він, навпаки, допомагає їм при виникненні будь-яких труднощів і стежить, щоб їх робота була злагодженою. Саме від Скрам-майстра залежить атмосфера в команді розробників, а, як наслідок, і їх ініціативність, задоволеність і загальний результат. Скрам-майстер вирішує будь-які проблеми, які можуть якось перешкодити команді, будь то зламане обладнання або зовнішній тиск з боку замовника. Скрам-майстер також стежить, щоб всі події «скрам» (про які буде написано нижче) відбувалися регулярно і з максимальною ефективністю;

Скрам-команда складається з професіоналів, які виконують роботу з розробки потенційно готового до випуску нової версії продукту в кінці кожного спринту [41]. Зазвичай кількість розробників і програмістів не перевищує восьми, і всі вони є зацікавленими в успішній реалізації IT-проекту. Перед початком кожної ітерації Скрам-команда ставить досяжну і значиму для замовника мету, а під час ітерації спрямовує всі свої зусилля на досягнення цієї мети в терміни. Досягнення мети характеризується наявністю запланованого коду, який згодом був налагоджений, а можливі дефекти ітерації усунені. Скрам-команда сама встановлює собі терміни (обговоривши їх зі Скрам-майстром і Власником продукту), сама оцінює свої можливості і розподіляє час.

Серед артефактів методології «скрам» виділяють журнал продукту, журнал спринту і діаграми згоряння.

Журнал продукту створюється Власником продукту (замовником) після аналізу потреб бізнесу у вигляді списку вимог до програмного забезпечення. Іншими словами, замовником описуються головні бажані функціонали

розробляється або впроваджуваної програми. Після цього кожній вимозі приписується ступінь важливості для замовника, тобто пріоритет, причому найбільш пріоритетні завдання повинні знаходитися вище за списком в журналі продукту, ніж найменш пріоритетні. Протягом всього проекту Власник продукту має право додавати до журналу продукту нові вимоги або модифікувати старі. Саме така можливість дозволяє даному підходу бути «гнучким до змін». У реальному житті у замовника може постійно змінюватися список вимог до програмного забезпечення, відповідно, Власник продукту може вести їх безпосередній облік і служити «з'єднуючою» ланкою між компанією-замовником, інтереси якої Власник продукту являє, і Скрам-командою (розробниками). Обрані на певний спринт вимоги з журналу продукту заносяться в журнал спринту.

Схематично це можна представити таким чином (рис2.4):

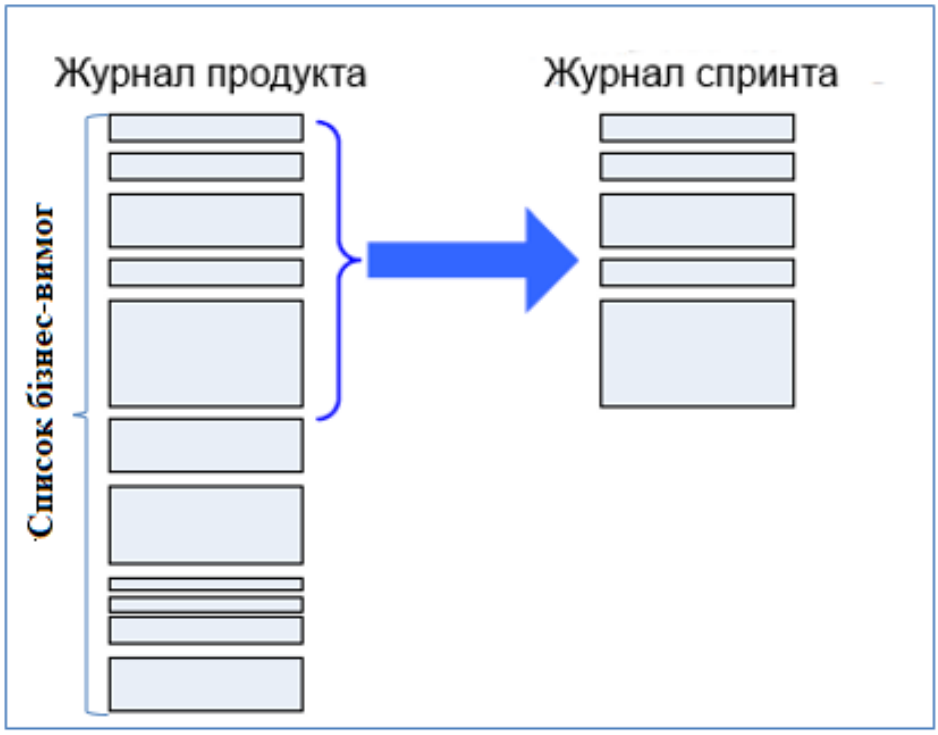


Рис.2.4. Формування журналу спринту

На відміну від журналу продукту, який може бути змінений або поповнений Власником продукту в будь-який момент часу проекту, журнал спринту на час конкретної ітерації є фіксованим і затвердженим командою.

Завдання з журналу спринту можуть бути виключені тільки в форс-мажорних ситуаціях, коли їх виконання більше не буде націлене на успіх всього IT-проекту.

Як було сказано раніше, методологія «скрам» зовсім не виключає документацію повністю, її просто набагато менше, ніж в традиційному підході Водоспадної розробки. На практиці все вирішується Власником продукту і Скрам-командою: якщо обидві сторони приходять до угоди оформити процес реалізації якогось завдання з журналу продукту, то розробниками створюється відповідний невеликий документ. Даний документ не буде подобою технічного завдання, на нього не будуть посилатися під час обговорення контрактних умов, він необхідний для більш конкретного розуміння функціонування будь-яких частин програми.

Для того щоб вести облік обсягу виконаної роботи, а також обсягу майбутньої роботи, використовуються діаграми згоряння. Перед кожною ітерацією команда планує кількість зусиль, які необхідно для виконання завдань з журналу спринту. Після завершення кожного завдання Скрам-майстер перераховує кількість залишеної роботи на цей спринт. Ці значення відзначаються на графіку, де по осі абсцис знаходиться тривалість спринту, а по осі ординат кількість залишеної роботи. Діаграма згоряння корисна в якості допоміжного інструменту, що дозволяє оцінити, чи в правильному напрямку йде робота команди в рамках конкретного спринту.

Крім спринту в методології «скрам» існують ще чотири заходи: планування спринту, щоденна нарада, огляд підсумків спринту і ретроспективна нарада. Перед початком спринту відбувається його планування. Спочатку Власник продукту демонструє журнал продукту, в якому знаходяться вимоги до програмного забезпечення, відсортовані за пріоритетом на конкретний момент. При цьому Власник продукту обговорює зі Скрам-командою можливі нюанси завдань, тим самим, мінімізуючи можливість появи суб'єктивізму в розробці. Після цього команда вибирає найбільш важливі завдання з журналу продуктів, які поміщаються в журнал спринту. Ключовим

моментом тут є визначення необхідного часу для виконання цих завдань самими розробниками. Так як в даному випадку вони все вирішують самі, без зовнішнього примусу від менеджера проекту, то ймовірність до невиправданого завищення часу мінімальна. Після того, як завдання на наступний спринт призначені, Скрам-команда починає між собою обговорювати кожну з них більш детально. Для успішного виконання завдання, вона може деталізувати зрозумілим розробникам чином на міні-завдання. Крім цього, Скрам-команда обговорює сам процес розробки коду, взаємозалежність між різними компонентами і так далі. Іншими словами, розробники вирішують, як вони будуть реалізовувати вимогу замовника.

Методологія «скрам» націлена на організацію злагодженої роботи всередині команди. Планування спринту триває близько чотирьох годин і головним результатом цього заходу є домовленість між Власником продукту і Скрам-командою з приводу кількості завдань, над якими будуть вестися роботи протягом планованого спринту, а так само впевненість в тому, що розуміння прийшло до всіх учасників зустрічі. Незважаючи на «гнучкість» методології «скрам», команда повинна бути впевнена в незмінності та стабільності завдань в рамках конкретного спринту. Якщо Власник продукту хоче поміняти вимоги, то для цього йому варто дочекатися планування наступного спринту. Такий підхід одночасно «гнучкий» і одночасно захищає проект від можливого безладу і розбіжностей. Після закінчення планування починається спринт. Кожен робочий день Скрам-майстер організовує коротку нараду, що не перевищує за тривалістю п'ятнадцяти хвилин. Метою щоденного наради є надання можливості учасникам команди поділитися власним прогресом і виниклими труднощами. Кожен учасник команди відповідає на три питання:

1. Що було зроблено з моменту минулого такої наради?
2. Що планується зробити до наступного такого наради?
3. Які існують складності, які гальмують процес виконання завдання?

При виникненні якихось проблем, Скрам-майстер робить все можливе, щоб усунути їх після закінчення щоденного наради. Таким чином,

п'ятнадцятихвилинні зустрічі допомагають тримати в курсі учасників Скрам-команди про те, що роблять їхні колеги і на якій стадії знаходяться завдання з журналу продукту. Власник продукту теж може бути присутнім на цій зустрічі, проте ніяких дискусій між ним і розробниками бути не повинно: зустріч організовується тільки для короткої звітності [35].

Якщо Скрам-команді не вдалося досягти мети спринту, то на останній щоденній нараді розробники визнають цей факт і намагаються розібратися в причинах події. Цілком можливо, що на етапі планування спринту неправильно були оцінені можливості команди і необхідний час для виконання завдань. Дуже часто це відбувається на найперших спринтах, що логічно: команда ще не до кінця могла спрацюватися з Власником продукту і з новим IT-проектом в цілому. Однак в майбутньому такі невдачі повинні бути виправлені за допомогою потрібного впливу Скрам-майстра.

Після завершення спринту проводиться огляд його підсумків, де демонструється основна функціональність, що з'явилася в проекті за цей спринт. На цій зустрічі може бути присутнім будь-який бажаючий (наприклад, хтось із керівництва компанії-замовника хоче побачити новий функціонал). Важливо відзначити, що огляд підсумків спринту не передбачає організацію і планування презентації, на нараді має бути безпосередня демонстрація того, що було зроблено в минулому спринті. Власник продукту визначає повноту і якість виконання завдання, тим самим налагоджуючи зворотний зв'язок з розробниками (рис.2.5.)


Рис.2.5. Організація зворотного зв'язку в рамках огляду підсумків спринту

Згодом Власником продукту коригується журнал продукту і робляться деякі висновки щодо подальших термінів проекту. Результатами зустрічі є скоригований журнал продукту і плани на подальший спринт.

Ретроспективна нарада також проводиться по завершенні спринту, однак на цій зустрічі присутні тільки учасники Скрам-команди, Скрам-майстер і Власник продукту (для додаткового зворотного зв'язку).

Тривалість наради варіюється від п'ятнадцяти хвилин до двох годин, в залежності від того, довгий чи був спринт, чи багато виявлено проблем і так далі. Під час цієї зустрічі розробники обговорюють між собою проблеми і позитивні моменти минулого спринту. Якщо ж команда приходить до висновку, що в проекті присутня проблема, яка заважає всім учасникам, то виноситься рішення про зміни, які можна здійснити в наступному спринті для того щоб поліпшити робочий процес. Приклад результатів ретроспективного наради наведено на рис2.6.

Рис.2.6. Візуалізація позитивних і негативних моментів під час ретроспективного наради

У загальному і цілому, «скрам»-процес може бути схематично представлений наступним чином (рис.2.7).


Основні ролі всередині Agile Scrum-команди :

Рис.2.7. Повноцінний "скрам" процес

Підводячи підсумок, можна відзначити, що завдяки постійному аналізу виконаної роботи і можливостям здійснювати коригування напрямку проекту між ітераціями (спринт) методологія «скрам» дозволяє більш продуктивно досягти результатів і, отже, більш якісно розробити програмне забезпечення. Методологія передбачає, що команда вибирає ті завдання, які є найбільш пріоритетними для бізнесу і відповідають технічним можливостям. Крім того, відсутність зовнішнього тиску і самостійність у виборі обсягу роботи сприяють тому, що розробники відчувають себе повністю залученими в процес розробки фахівцями, від яких залежить якість продукту, а не простими виконавцями, які лише виконують доручення свого менеджера. Такий підхід стимулює прояв ентузіазму і активності у розробників. Крім усього іншого, самостійний вибір обсягу робіт передбачає, що працівник знає свою продуктивність, а значить ймовірність відхилення від графіка значно нижче, ніж при тій же традиційної моделі управління IT-проектом. Від тривалості залежить вартість проекту, відповідно, якщо замовнику доведеться сплатити додатковий час роботи розробника (якщо розробник не вклався в свій власне поставлений термін), то ця вартість буде набагато нижче вартості запиту на зміну, який міг би виникнути в аналогічній ситуації в Водоспадної моделі.

Методологія «скрам» націлюється на постійну зміну пріоритетів вимог бізнесу, що збільшує прибутковість проекту навіть на самих ранніх етапах. Крім того, за рахунок постійних зустрічей Скрам-команди і Власника продукту, на яких учасники отримують зворотний зв'язок один від одного, команді вдається розробляти саме таке програмне забезпечення, яке максимально відповідає очікуванням замовника. Так як в проекті важливу роль відіграє задоволення замовника, то IT-проект можна вважати якісним не тільки коли він налагоджено працює після впровадження, але і коли відповідає вимогам замовника. Значною перевагою є можливість спостереження за проміжним продуктом, розробленим або впровадженим протягом певного спринту. Це дозволяє виявляти і виправляти помилки впровадженого фрагмента на ранніх етапах. В цьому і полягає ступінь якості проекту. Власник продукту разом зі

своїми колегами може сам побачити роботу програми, нехай і з мінімум можливостей.

## ВИСНОВКИ ДО РОЗДІЛУ 2

У другому розділі кваліфікаційної роботи зазначено, що незважаючи на успішний досвід японських корпорацій до недавнього часу гнучкі методи управління проектами активно використовувалися лише при розробці програмного забезпечення. Однак, поступово керівники багатьох компаній по всьому світу стали приходити до розуміння застосовності ключових цінностей гнучких методів управління і в інших сферах діяльності і почали впроваджувати гнучкі практики управління в своїх організаціях

Розкрито, що сьогодні під «agile» в широкому сенсі розуміється не тільки об'єднання ряду гнучких методів та інструментів управління, а й культурних зміни в організації, які відповідають єдиної філософії, базується на чотирьох ключових цінностях Agile Маніфесту. До таких культурних змін відноситься відмова від командно-адміністративних методів управління, розвиток культури довіри та взаємодопомоги, заохочення креативності та нестандартного мислення і т.д. У вузькому ж сенсі «agile» являє собою ряд практик і методів, характеризуються певними відмінностями.

Досліджено, що «гнучкі» методології базуються на емпіричному управлінні, тобто на такому управлінні, де рішення приймаються, виходячи з проміжних результатів проекту. Крім того, дані результати є прозорими, що означає, що всі залучені в проект люди обізнані статусом робіт за проектом, кількістю змін і можливими проблемами.

Отже, в даний час існує безліч гнучких методологій, але найбільш популярними є «екстремальне програмування» (XP), «скрам», «ощадлива розробка» (lean), «розробка, керована функціональністю» (fdd) і «канбан».

«Скрам» - найбільш популярна «гнучка» методологія, широко використовувана зарубіжними компаніями, такими як Yahoo !, PayPal, Nike, Google, SAP, GE для управління проектами.

Обгрунтовано, що методологія «скрам» націлена на взаємодію з замовником, і, не дивлячись на те, що команда розробників сама вирішує, які завдання вона буде виконувати протягом однієї ітерації, в даній методології присутній спостерігач (Скрам-майстер), який контролює дотримання «скрам» - процесу. Крім усього іншого, «скрам» добре документований: існує спеціальне «керівництво по скрам», перекладене на багато мов світу. Певна ступінь формалізованності в термінах «гнучкої» методології розробки є найкращою стратегією управління IT-проектом, коли в проекті беруть участь люди з різних організацій

Отже, «Скрам» - це гнучкий і легкий процес управління та контролю програмним забезпеченням і процесом розробки продукту в швидко мінливих умовах. Дана методологія встановлює певні правила управління IT-проектом (розробкою або впровадженням інформаційних технологій), які ґрунтуються на можливості постійного коректування вимог і на внесення тактичних змін.

# РОЗДІЛ 3
# ПРОЕКТУВАННЯ ТА ВПРОВАДЖЕННЯ СПЕЦІАЛІЗОВАНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В БАНК

## 3.1. Характеристика впроваджуваного програмного забезпечення в Банк

Банком «XXX» впроваджувалося спеціалізоване програмне забезпечення для автоматизованої оцінки кредитоспроможності клієнта і для прийняття рішення про видачу кредиту. В якості замовника розробки і впровадження ПЗ виступав безпосередньо Банк, в якості виконавця - постачальник програмного забезпечення для фінансового ринку.

Впроваджувана програма являє собою промислову платформу, призначену для перевірки ризикових правил в процесі розгляду заявки на отримання кредиту. Впроваджувана програма дозволяє:

- Автоматизувати бізнес-процес кредитування в Банку;
- Автоматизувати сучасні методи оцінки ризику клієнта і заявки відповідно до кредитної політики Банку;
- Мінімізувати час розгляду заявки на видачу кредиту;
- Зменшити операційні ризики Банку;
- Сформувати новий продуктову пропозицію клієнту, відповідно його ризик-сегменту.

Впровадження такої програми в Банк зможе скоротити кількість людського впливу, що, в результаті, може мінімізувати помилки, пов'язані з похибкою працівників.

Для повноцінної роботи впроваджуване програмне забезпечення інтегрується з фронт-офісною системою, що забезпечує коректну підтримку бізнес-процесу кредитування. В рамках розгляду однієї заявки буде вироблено кілька звернень до впроваджуваної системи за допомогою призначеного для користувача інтерфейсу. Обмін даними між системами здійснюється за

допомогою обміну XML-повідомленнями, в яких при кожному виклику дописується нова інформація.

Після одного з останніх звернень до системи XML-повідомлення містить в собі ризик-сегмент заявки, після чого, відповідно до кредитної політики банку приймається рішення про видачу кредиту.

План проекту по впровадженню автоматизованої системи щодо прийняття рішення про видачу кредиту складався на базі традиційної моделі життєвого циклу проекту тривалістю 8 років, тобто, у вигляді «водоспаду» - послідовних фаз, де кожна фаза не може бути розпочато до завершення попередньої. Офіційна дата початку проекту - 23 травня 2012 року.

Базовий план проекту і розподіл ресурсів були побудовані в програмі управління проектами Microsoft Office Project (рис.3.1):

| | Название задачи | Баз. длительност | Базовое начало | Базовое окончание | Базовые трудозатраты | Базовые затраты |
|---|---|---|---|---|---|---|
| 1 | **Разработка программного обеспечения для принятия решения о выдаче кредита** | **102,5 дней** | **Ср 23.05.12** | **Пт 12.10.12** | **1 920 ч** | **937 600,00 р.** |
| 2 | Подготовка "опросника" по внедряемой ИС | 1 день | Ср 23.05.12 | Ср 23.05.12 | 8 ч | 4 000,00 р. |
| 3 | Сбор бизнес требований и документирование стратегии принятия решений | 35 дней | Чт 24.05.12 | Ср 11.07.12 | 560 ч | 280 000,00 р. |
| 4 | Проектирование модели данных на базе кредитной политики | 3 дней | Чт 12.07.12 | Пн 16.07.12 | 48 ч | 21 600,00 р. |
| 5 | Проектирование общей логики архитектуры решения | 2 дней | Вт 17.07.12 | Ср 18.07.12 | 32 ч | 14 400,00 р. |
| 6 | Определени необходимости интеграции с дополнительными сервисами | 6 дней | Чт 19.07.12 | Чт 26.07.12 | 96 ч | 48 000,00 р. |
| 7 | Анализ формата данных, предоставляемых дополнительными сервисами | 2 дней | Пт 27.07.12 | Пн 30.07.12 | 16 ч | 6 400,00 р. |
| 8 | Разработка схемы XSD модели данных | 2 дней | Вт 31.07.12 | Ср 01.08.12 | 16 ч | 8 000,00 р. |
| 9 | Разработка тест-стенда для проверки корректности реализованной стратегии | 37 дней | Вт 31.07.12 | Ср 19.09.12 | 592 ч | 296 000,00 р. |
| 10 | Программирование стратегий в внедренной ИС | 33 дней | Чт 02.08.12 | Пн 17.09.12 | 264 ч | 132 000,00 р. |
| 11 | Модульное тестирование разработанной стратегии | 6 дней | Вт 18.09.12 | Вт 25.09.12 | 48 ч | 19 200,00 р. |
| 12 | Интеграционное тестирование внедренной ИС с фронт-офисным решением | 10 дней | Ср 26.09.12 | Вт 09.10.12 | 160 ч | 72 000,00 р. |
| 13 | Тренинг по стратегии | 2,5 дней | Ср 10.10.12 | Пт 12.10.12 | 40 ч | 18 000,00 р. |
| 14 | Тренинг по администрированию системы | 2,5 дней | Вт 18.09.12 | Чт 20.09.12 | 40 ч | 18 000,00 р. |

.

Рис.3.1. Базовий план проекту і розподіл ресурсів

Відповідно до базового плану діаграма Ганта (візуалізована ланцюжок завдань) представлена на рис.3.2.

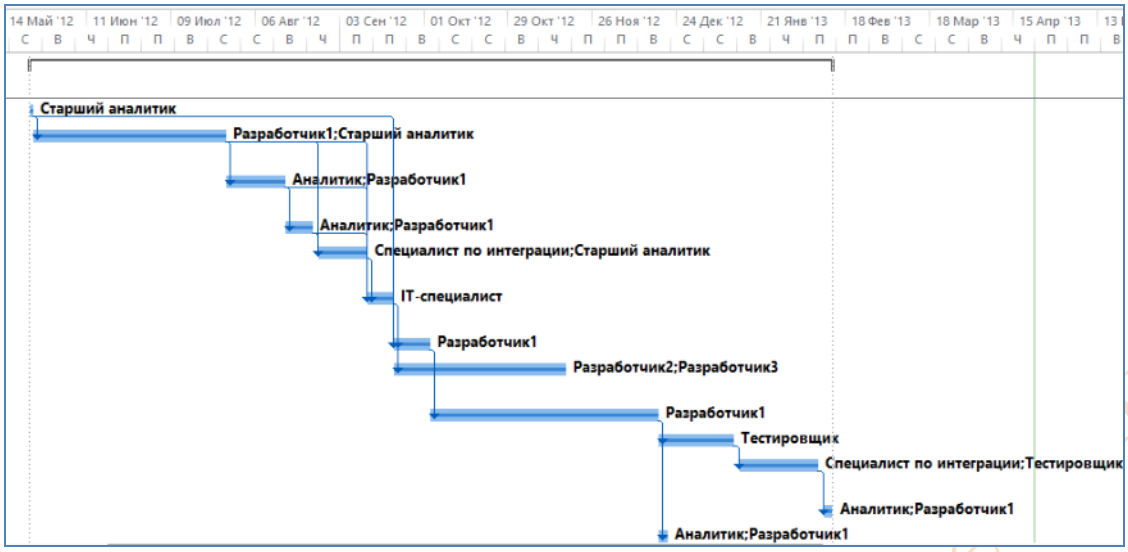


Рис.3.2. Відповідно до базового плану діаграма Ганта

Як видно з рис.3.2, проект був спланований менеджером як класичний «водоспад», причому етапи сформульовані досить широко. У представленому вище плані проекту відсутні операції злиття і практично відсутні дрібні операції.

Після того, як менеджером проекту і командою проекту було розпочато роботи з впровадження програмного забезпечення, виявилося, що терміни виконання завдань постійно зсуваються, причому слідом за завданням, виконуваної зі збільшенням базового терміну, зсувається початок виконання всіх наступних завдань. Під час реалізації проекту фіксувалася фактична тривалість кожного завдання, після чого був побудований реальний календарний план.

Порівняння запланованих і фактичних термінів виконання проекту в процесі виконання робіт продемонстровано на рис.3.3. Серед 14 завдань 9 з них мали суттєві відхилення по тривалості: завдання № 3, завдання № 4, завдання № 5, завдання № 6, завдання № 7, завдання № 8, завдання № 10, завдання № 11, завдання № 12. На рисунку вони виділені кольором:

| | Ре за | Название задачи | Начало | Окончание | Базовое начало | Базовое окончание | Базовая длительность | Длительнос | Базовые затраты | Затраты |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | **Разработка программного обеспечения для принятия решения о выдаче кредита** | **Ср 23.05.12** | **Ср 13.02.13** | **Ср 23.05.12** | **Пт 12.10.12** | **102,5 дней** | **191 дней** | **937 600,00 р.** | **1 465 600,00 р.** |
| 2 | | Подготовка "опросника" по внедряемой ИС | Ср 23.05.12 | Ср 23.05.12 | Ср 23.05.12 | Ср 23.05.12 | 1 день | 1 день | 4 000,00 р. | 4 000,00 р. |
| 3 | | Сбор бизнес требований и документирование стратегии принятия решений | Чт 24.05.12 | Чт 26.07.12 | Чт 24.05.12 | Ср 11.07.12 | 35 дней | 46 дней | 280 000,00 р. | 368 000,00 р. |
| 4 | | Проектирование модели данных на базе кредитной политики | Пт 27.07.12 | Ср 15.08.12 | Чт 12.07.12 | Пн 16.07.12 | 3 дней | 14 дней | 21 600,00 р. | 100 800,00 р. |
| 5 | | Проектирование общей логики архитектуры решения | Чт 16.08.12 | Пт 24.08.12 | Вт 17.07.12 | Ср 18.07.12 | 2 дней | 7 дней | 14 400,00 р. | 50 400,00 р. |
| 6 | | Определени необходимости интеграции с дополнительными сервисами | Пн 27.08.12 | Вт 11.09.12 | Чт 19.07.12 | Чт 26.07.12 | 6 дней | 12 дней | 48 000,00 р. | 96 000,00 р. |
| 7 | | Анализ формата данных, предоставляемых дополнительными сервисами | Ср 12.09.12 | Чт 20.09.12 | Пт 27.07.12 | Пн 30.07.12 | 2 дней | 7 дней | 6 400,00 р. | 22 400,00 р. |
| 8 | | Разработка схемы XSD модели данных | Пт 21.09.12 | Вт 02.10.12 | Вт 31.07.12 | Ср 01.08.12 | 2 дней | 8 дней | 8 000,00 р. | 32 000,00 р. |
| 9 | | Разработка тест-стенда для проверки корректности реализованной стратегии | Пт 21.09.12 | Пт 16.11.12 | Вт 31.07.12 | Ср 19.09.12 | 37 дней | 41 дней | 296 000,00 р. | 328 000,00 р. |
| 10 | | Программирование стратегий в внедренной ИС | Ср 03.10.12 | Пн 17.12.12 | Чт 02.08.12 | Пн 17.09.12 | 33 дней | 54 дней | 132 000,00 р. | 216 000,00 р. |
| 11 | | Модульное тестирование разработанной стратегии | Вт 18.12.12 | Пт 11.01.13 | Вт 18.09.12 | Вт 25.09.12 | 6 дней | 19 дней | 19 200,00 р. | 60 800,00 р. |
| 12 | | Интеграционное тестирование внедренной ИС с фронт-офисным решением | Пн 14.01.13 | Пт 08.02.13 | Ср 26.09.12 | Вт 09.10.12 | 10 дней | 20 дней | 72 000,00 р. | 144 000,00 р. |
| 13 | | Тренинг по разработке стратегий | Пн 11.02.13 | Ср 13.02.13 | Ср 10.10.12 | Пт 12.10.12 | 2,5 дней | 3 дней | 18 000,00 р. | 21 600,00 р. |
| 14 | | Тренинг по администрированию системы | Вт 18.12.12 | Чт 20.12.12 | Вт 18.09.12 | Чт 20.09.12 | 2,5 дней | 3 дней | 18 000,00 р. | 21 600,00 р. |

Рис.3.3. орівняння запланованих і фактичних термінів виконання проекту

Для наочності відхилень була побудована діаграма Ганта з відстеженням, де сірим виділені завдання базового плану, а синім - завдання реального плану (рис.3.4). Навіть з цього малюнку видно, що фактична тривалість проекту набагато перевищує заплановану. Цей же факт дає підстави припускати, що і вартість проекту значно збільшилася.

За допомогою спостереження за ходом виконання проекту були виявлені і систематизовані такі причини затримок завдань проекту:

Завдання № 3 «Збір бізнес вимог та документування стратегії прийняття рішень».

Базова тривалість передбачалася рівною 35 дням, фактична склала 46 днів.

Дане завдання є одним з основних етапів проекту, на якому оформлялося технічне завдання на бізнес-специфікацію правил настройки ризикової стратегії. З боку виконавця на даному етапі брали участь старший аналітик і основний розробник.

Довготривале формування вимог, прагнення до мінімізації ризиків шляхом повної аналітики можливих розбіжностей в майбутньому. Як наслідок - затримка моменту підписання технічного завдання з боку Банку (замовника);

Чи вносилися коригування і додавання в ризикову політику, а як наслідок, в вимоги до програмного забезпечення;

Довга невизначеність операційного відділу в наданні своїх вимог аналітикам по роботі впроваджуваного програмного забезпечення;

За затримки з банку не стягувалася додаткова плата, так як на той момент не було підписано технічне завдання (воно стало результатом даного етапу).

Завдання №4 «Проектування моделі даних на базі кредитної політики»:

Базова тривалість передбачалася рівною 3 дня, фактична склала 14 днів.

Під час попереднього етапу відділ маркетингу не брав участі у формуванні вимог до програмного забезпечення, але на поточному етапі його представники вирішили внести власні побажання по роботі впроваджуваної програми з клієнтом. Ці зміни у вимогах, за словами виконавця, в деяких місцях суперечили ТЗ, тому за доопрацювання цих вимог Банк заплатив 79 200 рублів.

Завдання №6 «Проектування загальної логіки архітектури рішення»:

Базова тривалість передбачалася рівною 6 днів, фактична склала 12 днів.

Затримки на цьому етапі пов'язані з комунікацією між виконавцем і компаніями, які надавали додаткові сервіси (CreditRegistry, ЦФТ).

Завдання №8 «Розробка схеми XSD моделі даних»:

Базова тривалість передбачалася рівною 2 дня, фактична склала 8 днів.

На даному етапі розробнику довелося змінити початкову модель даних, так як операційний відділ і відділ маркетингу Банку запросили новий функціонал, необхідний їм для роботи з клієнтом (можливість відсилати заявку клієнта на попередній етап). Повідомлення про порушення коштував банку 24 000 рублів.

Завдання №9 «Розробка тест-стенду для перевірки коректності реалізованої стратегії»:

Базова тривалість передбачалася рівною 37 дням, фактична склала 41 днів.

Тест-стенд отримує на вхід XML-повідомлення, потім звертається до сервера, де знаходиться програмне забезпечення щодо прийняття рішення про видачу кредиту, обробляє запит, і після цього на екран повертається нове,

доповнене XML-повідомлення з результатами перевірки. Тест-стенд в своєму першому релізі не мав функціоналу збереження XML-відповідей, що здалося вкрай незручним Банку. Проблема затримки на даному етапі викликана різним тлумаченням ТЗ розробниками і працівниками банку. Виконавець вважав, що завдання виконане коректно, принаймні предмет спору не описаний в ТЗ, в той час як Банк порахував що такий очевидний пункт не варто навіть докладно описувати. Додаткові роботи виконавець оцінив в 32 000 рублів.

Завдання № 10 «Програмування стратегій у впровадженої ІС»:

Базова тривалість передбачалася рівною 33 дням, фактична склала 54 днів.

Затримки на даному етапі пов'язані з явними змінами бізнес вимог програмного забезпечення, викликаними коригуваннями відділу маркетингу, операційного відділу та відділу ризиків. Багато з нововведень спричинили за собою зміни в моделі даних (андеррайтери з операційного відділу захотіли бачити в інтерфейсі розподіл доходу по місяцях, відбулося уточнення функціоналу деяких кнопок на екранних формах, наприклад, з'явилася можливість здійснення візуальної оцінки клієнта). Запити на зміни на поточному етапі обійшлися Банку в 84 000 рублів.

Завдання № 11 «Модульне тестування розробленої стратегії»:

Базова тривалість передбачалася рівною 6 днів, фактична вийшла 19 днів.

З результатами тестування були ознайомлені працівники відділу ризиків Банку. Затримки під час виконання даного завдання викликані розбіжністю між розробниками програми і співробітниками відділу ризиків з приводу коректності налаштованої стратегії. Замовником були встановлені нові вимоги, що враховують динаміку заявки клієнта в часі (в ТЗ даний функціонал описаний не був). Трудовитрати на додавання такого функціоналу були оцінені менеджером проекту в 32 години, що в грошовому вираженні дорівнює 16 000 рублів згідно зі ставкою розробника.

Завдання №12: «Інтеграційний тестування впровадженої ІС з фронт-офісним рішенням»:

Базова тривалість передбачалася рівною 10 дням, фактична склала 20 днів.

Так як на даному етапі в якості людського ресурсу використовується фахівець з інтеграції.

Проаналізувавши цілком проект впровадження програмного забезпечення в банк, можна відзначити наступне:

1) Головним критерієм успішності проекту була якість. Під якістю буде матися на увазі можливість виявлення і виправлення помилок на ранніх етапах проекту, а також повну відповідність результатів проекту вимогам замовника. Так як впроваджуване програмне забезпечення в майбутньому має автоматично оцінювати клієнтів банку, при впровадженні проекту було необхідно мінімізувати операційні ризики, які могли б виникнути в подальшому. Як результат, тривалість і бюджет проекту автоматично збільшувалися через постійні доробоки програмного забезпечення (протягом 8 років). Це означає, що для замовника якість була ключовим моментом, заради якого він готовий був пожертвувати тривалістю і бюджетом;

2) В середині і наприкінці проекту команда розробників перебувала в постійно стресовому стані, так як відсутність комунікації з іншими працівниками спричинило за собою суб'єктивне розуміння технічного завдання. В результаті менеджером проекту було вирішено призначити їм додатковий обсяг роботу за рахунок Банку, але, незважаючи на це, продуктивність працівників була нижчою передбачуваної через людський фактор;

3) Відхилення по часу становить 88,5 днів (в базовому плані передбачалося 102,5 днів, фактично вийшло 191 день);

4) У плані передбачалася вартість проекту дорівнює 937 600 рублів, фактична склала 1 465 600 рублів, з яких 323 200 рублів - запити на зміни, відповідно, вартість проекту для Банку дорівнює 1 260 800 рублів

Зважаючи на специфіку Водоспадної моделі, на підставі якої проектувався життєвий цикл цього IT-проекту, тестування і налагодження програмного забезпечення відбувається набагато пізніше розробки, що

автоматично виключає можливість виявлення помилок на ранніх етапах і їх подальше виправлення. Тому критерій якості в даному випадку залежить безпосередньо від задоволення замовника від отриманого, в кінцевому рахунку, продукту.

Таким чином, щоб дійсно досягти більш-менш бажаний рівень якості, тривалість проекту довелося збільшити на 86,3%, а вартість на 34,5%.

Введемо наступні позначення:

ω-якість проекту,

t-тривалість проекту,

c-вартість проекту.

Банк і менеджер проекту впровадження (представник компанії-Виконавця) прагнуть мінімізувати можливі, небажані відхилення від необхідної якості (тобто, цільової результат має на увазі відповідність продукту очікуванням замовника), отже, для того, щоб мінімізувати відхилення за якістю (яке було вибрано найбільш пріоритетним з всіх критеріїв проекту), тривалість проекту і його бюджет довелося збільшити:

$\Delta\omega \rightarrow \min$ при $\Delta t = 86{,}3\%$ і $\Delta c = 34{,}5\%$

### 3.2. Моделювання та аналіз впровадження проекту за допомогою «гнучкою» методології «скрам»

Проаналізувавши недоліки Водоспадної моделі при її використанні для управління IT-проектами в банку, а також переваги «гнучких» методологій розробки та вибравши методологію «скрам» як найбільш підходящу методологію для управління IT-проектом з огляду на її формалізації і організованості, була змодельована ситуація впровадження програмного забезпечення для прийняття рішення про видачу кредиту з використанням методології «скрам».

Процес моделювання можна описати таким чином:

Вводяться передумови, що врівноважують проектовану модель з реальною ситуацією:

Продуктивності команд при використанні методології «скрам» і Водоспадної моделі рівні;

Кількість фахівців тієї чи іншої області в двох проектах однаково, і воно приведено на рис.3.4.

| Название ресурса | Тип | Единицы измерения материалов | Краткое название | Группа | Макс. единиц | Стандартная ставка |
|---|---|---|---|---|---|---|
| Старший аналитик | Трудовой | | С | | 100% | $16,12/час |
| Аналитик | Трудовой | | А | | 100% | $13,00/час |
| IT-специалист | Трудовой | | I | | 100% | $13,00/час |
| Тестировщик | Трудовой | | Т | | 100% | $13,00/час |
| Разработчик 1 | Трудовой | | Р | | 100% | $16,12/час |
| Разработчик 2 | Трудовой | | Р | | 100% | $16,12/час |
| Разработчик 3 | Трудовой | | Р | | 100% | $16,12/час |
| Специалист по интегра | Трудовой | | С | | 100% | $16,12/час |

Рис.3.4. Ресурси

1. Час виконання завдань при використанні методології «скрам» включає час проведення командою нарад, що є теж оплачуваних замовником;

2. Збір нових вимог не займає додаткового часу, а відбувається протягом «спринту»;

3. Впровадження програми відбувається безпосередньо в офісі банку, тобто для якісного застосування методології «скрам» краще уникати віддаленої роботи.

Описується хронологія ведення проекту і формалізація дій:

Перед офіційним початком проекту організовується зустріч представника з боку замовника (Власника продукту), яким є співробітник відділу ризиків, Скрам-майстра і Скрам-команди для обговорення спільного бачення проекту. Обговорюється програма, яка буде впроваджуватися, Скрам-командою відбувається усвідомлення базових моментів, які бажає спостерігати замовник. Замовник, в свою чергу, намагається зрозуміти специфіку програмного забезпечення і принципи роботи Скрам-команди. Протягом цього часу (7-ми днів) відбувається обмін загальними домовленостями і обіцянками (наприклад, учасники проекту приходять до висновку про документування будь-яких довідкових значень).

«Скрам» -проект буде проходити в такий спосіб:

- Власник Продукту готує до кожного спринту журнал продукту, відсортований за пріоритетом;
- Кожен спринт починається з його планування, тобто команда визначає кількість часу, який необхідний кожному учаснику для виконання завдань з журналу продукту;
- Вибрані завдання з журналу продукту переносяться в журнал спринту і, якщо треба, деталізуються розробниками;
- Під час планування спринту учасники команди обговорюють загальну ідею виконання завдань, їх корелюються з іншими фахівцями та інше;
- Протягом спринту команда виконує завдання, кожен день звітуючи один одному про виконану роботу під час щоденного наради;
- Виконання термінів нарад, а також підтримання атмосфери в команді забезпечує Скрам-майстер;
- В кінці спринту команда демонструє налаштований функціонал Скрам-майстру, Власнику продукту і розповсюджує за бажанням будь кому з боку замовника (наприклад, топ-менеджменту Банку);
- Якщо Скрам-команда не встигає зробити завдання за один спринт, то, в разі дрібних недоліків, спринт може затриматися на пару днів, а в разі великих недоробок це ж завдання виноситься знову в журнал продукту;
- Протягом спринту у Власника продукту можуть з'явитися нові вимоги до програмного забезпечення, які він повинен записувати в журнал продукту, після чого він чекає зустрічі з планування нового спринту для того, щоб внести зміни в роботу.

Оцінка базової тривалості завдань, описаних за допомогою методології «скрам»:

Згідно з методологією «скрам» команда розробників сама оцінює час, який їм може знадобитися для виконання завдань. Для моделювання проекту було проведене експертне опитування розробників даного IT-проекту для виявлення передбачуваної тривалості і максимально можливих відхилень. Їм

була пояснена ідея впровадження програмного забезпечення по ітераціям за допомогою методології «скрам» і був продемонстрований приклад журналу продукту з базовими вимогами від замовника, відсортований за спаданням важливості для бізнесу на момент складання:

1. Здатність впроваджуваної системи перевіряти клієнта по мінімальним вимогам кредитної політики;

2. Налаштування функціоналу, призначеного для тестування налаштованої стратегії у програмному забезпеченні;

3. Здатність впроваджуваної системи перевіряти і оцінювати зовнішню кредитну історію клієнта;

4. Здатність впроваджуваної системи здійснювати пошук клієнта по «чорним» списками (список неплатників і клієнтів, які є боржниками);

5. Здатність впроваджуваної системи розраховувати скоринговий бал за формулою, визначеною Власником продукту;

6. Здатність впроваджуваної системи аналізувати результати телефонного верифікації клієнта;

7. Здатність впроваджуваної системи визначати ризик-сегмент заявки;

8. Здатність впроваджуваної системи аналізувати результати перевірки клієнта по службі безпеки;

9. Здатність впроваджуваної системи визначати необхідність перегляду заявки при зміні клієнтом схвалених умов;

10. Здатність впроваджуваної системи визначати роль для остаточного прийняття рішення;

Іншими словами Власник продукту представив ряд функцій, які має виконувати програмне забезпечення. Як правило, це функції перевірки даних клієнта, взяті з анкетної інформації і з зовнішніх джерел. Для Власника продукту пріорітетніше всього спочатку налаштувати базові перевірки, а після цього здійснити настройку агрегованих (наприклад, визначення ризик-сегмента заявки на основі отриманих даних і проаналізованих результатів).

При проведенні опитування розробники були попереджені, що кожна ітерація триває 14-30 днів, на що вони згодом робили упор при моделюванні. Таким чином, на найпершу ітерацію (тобто, в журнал спринту) розробники призначили перше завдання з журналу продукту і частину другого (рис.3.5). План проекту будувався за допомогою доповнення до програмного забезпечення Microsoft Office Project 2010 яке називається Scrum Tool, яке просто допомагає формалізувати процес.

| Название задачи | Баз. длительность | Базовое начало | Базовое окончание | Базовые трудозатраты | Базовые затраты | Предшественники | Названия ресурсов |
|---|---|---|---|---|---|---|---|
| **Sprint 1** | **15 дней** | **Ср 23.05.12** | **Вт 12.06.12** | **256 часов** | **$3 976,96** | | |
| **Функциональность проверки соответствия заявки клиента минимальным требованиям кредитной политики Банка** | **11 дней** | **Ср 23.05.12** | **Ср 06.06.12** | **128 часов** | **$1 913,60** | | |
| Проанализировать кредитную политику | 2 дней | Ср 23.05.12 | Чт 24.05.12 | 16 часов | $257,92 | | Старший аналитик |
| Проанализировать пользовательский интерфейс ввода данных | 1 день | Ср 23.05.12 | Ср 23.05.12 | 8 часов | $104,00 | | Аналитик |
| Разработать необходимый фрагмент модели данных | 2 дней | Пт 25.05.12 | Пн 28.05.12 | 16 часов | $257,92 | 3;4 | Разработчик 1 |
| Настроить соответствующий интеграционный слой | 4 дней | Пт 25.05.12 | Ср 30.05.12 | 32 часов | $515,84 | 3;4 | Специалист по интеграции |
| Настроить стратегию в программе | 2 дней | Вт 29.05.12 | Ср 30.05.12 | 16 часов | $257,92 | 5 | Разработчик 1 |
| Тестирование (настроенной стратегии и интеграционное) | 5 дней | Чт 31.05.12 | Ср 06.06.12 | 40 часов | $520,00 | 6;7 | Тестировщик |
| **Начало разработки тест-стенда для тестирования стратегии** | **15 дней** | **Ср 23.05.12** | **Вт 12.06.12** | **128 часов** | **$2 063,36** | | |
| Проектирование дизайна тест-стенда | 6 дней | Ср 23.05.12 | Ср 30.05.12 | 48 часов | $773,76 | | Разработчик 2 |
| Настройка соединения с сервером для получения ответа от автоматизированной программы принятия решения | 1 день | Ср 23.05.12 | Ср 23.05.12 | 8 часов | $128,96 | | Разработчик 3 |
| Написание кода для проверки сратегии и отображения результатов | 9 дней | Чт 31.05.12 | Вт 12.06.12 | 72 часов | $1 160,64 | 10 | Разработчик 2 |

Рис.3.5. Моделювання першого спринту

Як видно з вище представленого рисунку, протягом першого спринту були задіяні три розробника, два аналітика і один фахівець з інтеграції. Разом вони деталізували вимоги з журналу спринту і оцінили час, який їм потрібен для виконання своєї частини завдання.

З урахуванням важливих змін у вимогах, які були описані при розгляді плану проекту, побудованого за допомогою Водоспадної моделі, був змодельований фінальний журнал продукту (після завершення проекту) (рис.3.6).

| | | | Deliverable | S | Priori Rank | Sp | Task Status | Item Type |
|---|---|---|---|---|---|---|---|---|
| 9 | ✓ | | Начало разработки тест-стенда для тестирования стратегии | | 99 | 1 | Completed | Product Backlog |
| 2 | ✓ | | Функциональность проверить соответствия заявки клиента минимальным требованиям кредитной политики Банка | | 100 | 1 | Completed | Product Backlog |
| 25 | ✓ | | Добавить в расчет минимальных требований новые условия (по запросу отдела Маркетинга) | | 90 | 2 | Completed | Product Backlog |
| 29 | ✓ | | Функциональность проверки внешней кредитной истории клиента | | 98 | 2 | Completed | Product Backlog |
| 14 | ✓ | | Функциональность проверки клиентов по черным спискам | | 99 | 2 | Completed | Product Backlog |
| 20 | ✓ | | Окончание разработки тест-стенда для тестирования стратегии | | 100 | 2 | Completed | Product Backlog |
| 42 | ✓ | | Посчитать скоринговый бал клиента | | 97 | 3 | Completed | Product Backlog |
| 41 | ✓ | | Настроить возможность сохранения результатов сработавшей стратегии тест-стендом | | 98 | 3 | Completed | Product Backlog |
| 37 | ✓ | | Исправление обнаруженных ошибок при тестировании | | 99 | 3 | Completed | Product Backlog |
| 36 | ✓ | | Полное тестирование трех настроенных модулей (Минимальные требования, ЧС, КИ) | | 100 | 3 | Completed | Product Backlog |
| 59 | ✓ | | Функциональность анализа результатов проведенной андеррайтером телефонной верификации клиента | | 97 | 4 | Completed | Product Backlog |
| 55 | ✓ | | Настроить транслитерацию результатов автоматизированной системы принятия решений для отображения во фронт-офисном решении | | 98 | 4 | Completed | Product Backlog |
| 49 | ✓ | | Внести изменения отображение результатов на окно андеррайтера | | 100 | 4 | Completed | Product Backlog |
| 86 | ✓ | | Функциональность определения риск-сегмента заявки | | 97 | 5 | Completed | Product Backlog |
| 79 | ✓ | | Добавить функционал того, чтобы клиент с плохой визуальной оценкой попадал на телефонную верификацию | | 98 | 5 | Completed | Product Backlog |
| 74 | ✓ | | Добавить функциональность определения стратегией актуальных проверок (Минимальные требования, СБ, КИ, ТВ, ЧС, скоринг) | | 99 | 5 | Completed | Product Backlog |
| 67 | ✓ | | Функциональность проверки клиентов по службе безопасности | | 100 | 5 | Completed | Product Backlog |
| 105 | ✓ | | Добавить новый функционал, который был не учитан на предыдущих этапах (авторы новых требований - специалисты отдела Рисков) | | 98 | 6 | Completed | Product Backlog |
| 99 | ✓ | | Функциональность определения необходимости полного пересмотра заявки при изменении клиентом условий | | 99 | 6 | Completed | Product Backlog |
| 93 | ✓ | | Функциональность определения роли для окончательного принятия решения | | 100 | 6 | Completed | Product Backlog |
| 116 | ✓ | | Исравление правила определения роли для окончательного принятия решения | | 99 | 7 | Completed | Product Backlog |
| 112 | ✓ | | Добавить влияние наличия дополнительных кредитных услуг при проверки необходимости пересмотра заявки | | 100 | 7 | Completed | Product Backlog |

Рис.3.6. Стан журналу продукту в кінці проекту

Як видно з рисунків, крім назви вимоги-завдання в журнал пишуться так само ступінь пріоритету, розподіл в спринті (вимоги з журналу продукту відсортовані за хронологією проекту, тобто в міру виникнення нових ітерацій-спринтів), і статус вимоги-завдання.

Важливо відзначити, що журнал продукту являє собою вимоги замовника, написані зрозумілою йому мовою, («функціональність перевірки зовнішньої кредитної історії клієнта», «додано вплив наявності додаткових кредитних послуг при перевірці необхідності перегляду заявки»), на відміну від етапів Водоспадної моделі, складених менеджером проекту, який більше орієнтувався на розуміння етапів своєю командою ( «розробка схеми XSD моделі даних», «програмування стратегії в IC»).

Детальний опис бізнес-вимог знижує ризик відхилення від бажаного замовником якості, коли мова йде про відповідність його вимогам (т. Е. Δω).

Таким чином, змодельований повноцінний проект наведено на рис.3.7.

| | Название задачи | Баз. длитель | Базовое начало | Базовое окончание | Базовые трудозатрать | Базовые затраты | Предш |
|---|---|---|---|---|---|---|---|
| 1 | Sprint 1 | 15 дней | Ср 23.05.12 | Вт 12.06.12 | 256 часов | $3 976,96 | |
| 2 | Функциональность проверить соответствия заявки клиента минимальным требованиям кредитной политики Банка | 11 дней | Ср 23.05.12 | Ср 06.06.12 | 128 часов | $1 913,60 | |
| 9 | Начало разработки тест-стенда для тестирования стратегии | 15 дней | Ср 23.05.12 | Вт 12.06.12 | 128 часов | $2 063,36 | |
| 13 | Sprint 2 | 19 дней | Ср 13.06.12 | Пн 09.07.12 | 608 часов | $9 276,80 | 1 |
| 14 | Функциональность проверки клиентов по черным спискам | 17 дней | Ср 13.06.12 | Чт 05.07.12 | 184 часов | $2 691,52 | |
| 20 | Окончание разработки тест-стенда для тестирования стратегии | 19 дней | Ср 13.06.12 | Пн 09.07.12 | 288 часов | $4 642,56 | |
| 25 | Добавить в расчет минимальных требований новые условия (по запросу отдела Маркетинга) | 15 дней | Вт 19.06.12 | Пн 09.07.12 | 24 часов | $361,92 | |
| 29 | Функциональность проверки внешней кредитной истории клиента | 16 дней | Пт 15.06.12 | Пт 06.07.12 | 112 часов | $1 580,80 | |
| 35 | Sprint 3 | 20 дней | Вт 10.07.12 | Пн 06.08.12 | 224 часов | $3 311,36 | 13 |
| 36 | Полное тестирование трех настроенных модулей (Минимальные требования, ЧС, КИ) | 5 дней | Вт 10.07.12 | Пн 16.07.12 | 40 часов | $520,00 | |
| 37 | Исправление обнаруженных ошибок при тестировании | 6 дней | Вт 17.07.12 | Вт 24.07.12 | 48 часов | $773,76 | 36 |
| 41 | Настроить возможность сохранения результатов сработавшей стратегии тест-стендом | 2 дней | Вт 10.07.12 | Ср 11.07.12 | 32 часов | $515,84 | |
| 42 | Посчитать скоринговый бал клиента | 20 дней | Вт 10.07.12 | Пн 06.08.12 | 104 часов | $1 501,76 | |
| 48 | Sprint 4 | 17 дней | Вт 07.08.12 | Ср 29.08.12 | 280 часов | $4 139,20 | 35 |
| 49 | Внести изменения отображение результатов на окно андеррайтера | 10 дней | Вт 07.08.12 | Пн 20.08.12 | 104 часов | $1 501,76 | |
| 55 | Настроить транслитерацию результатов автоматизированной системы принятия решений для отображения во фронт-офисном решении | 9 дней | Пн 13.08.12 | Чт 23.08.12 | 80 часов | $1 214,72 | |
| 59 | Функциональность анализа результатов проведенной андеррайтером телефонной верификации клиента | 17 дней | Вт 07.08.12 | Ср 29.08.12 | 96 часов | $1 422,72 | |

Рис.3.7. Представлення проекту впровадження ПЗ в банк за допомогою «скрам» -методологіі

Як видно з прикладу, всі ітерації дуже схожі між собою, що було описано у другому розділі роботи: кожна ітерація - це, свого роду, міні-проект настройки певного функціоналу.

(Оцінка максимально можливих відхилень по тривалості завдань, описаних за допомогою методології «скрам»:

На підставі експертної думки розробників були змодельовані максимально можливі відхилення, які доводять, що, в будь-якому випадку, домогтися ідеального проекту неможливо, але можна знизити ймовірність затримок шляхом проведення постійного аналізу та випуску інкрементів продукту (що і передбачає методологія «скрам»). Крім усього іншого, при використанні «скрам» -методології можливо організовувати роботу паралельно, що не дозволяє зробити класичний водоспадний підхід.

Конкретний IT-проект, життєвий цикл якого був представлений за допомогою Водоспадної моделі, не має деталізації завдань, які самі по собі описані досить широко. На підставі цих чинників розробники, спираючись на свій експертний досвід, запропонували можливі максимальні відхилення, більша частина яких припадає на етапи, присвячені тестуванню.

Змодельовані відхилення фактичного плану від базового, побудованого за методологією «скрам» представлені на рис.3.7.

| | Режим задачи | Название задачи | Начало | Окончание | Базовое начало | Базовое окончание | Отклон. начала | Отклон. окончания |
|---|---|---|---|---|---|---|---|---|
| 1 | | Sprint 1 | Ср 23.05.12 | Вт 12.06.12 | Ср 23.05.12 | Вт 12.06.12 | 0 дней | 0 дней |
| 2 | | Функциональность проверить соответствия заявки клиента минимальным требованиям кредитной политики Банка | Ср 23.05.12 | Пн 11.06.12 | Ср 23.05.12 | Ср 06.06.12 | -0,13 дней | 3 дней |
| 9 | | Начало разработки тест-стенда для тестирования стратегии | Ср 23.05.12 | Вт 12.06.12 | Ср 23.05.12 | Вт 12.06.12 | -0,13 дней | 0 дней |
| 13 | | Sprint 2 | Ср 13.06.12 | Пт 13.07.12 | Ср 13.06.12 | Пн 09.07.12 | -0,13 дней | 4 дней |
| 14 | | Функциональность проверки клиентов по черным спискам | Ср 13.06.12 | Чт 05.07.12 | Ср 13.06.12 | Чт 05.07.12 | -0,13 дней | 0 дней |
| 20 | | Окончание разработки тест-стенда для тестирования стратегии | Ср 13.06.12 | Пт 13.07.12 | Ср 13.06.12 | Пн 09.07.12 | -0,13 дней | 4 дней |
| 25 | | Добавить в расчет минимальных требований новые условия (по запросу отдела Маркетинга) | Вт 19.06.12 | Пн 09.07.12 | Вт 19.06.12 | Пн 09.07.12 | -0,13 дней | 0 дней |
| 29 | | Функциональность проверки внешней кредитной истории клиента | Пт 15.06.12 | Пт 06.07.12 | Пт 15.06.12 | Пт 06.07.12 | -0,13 дней | 0 дней |
| 35 | | Sprint 3 | Пн 16.07.12 | Пт 17.08.12 | Вт 10.07.12 | Пн 06.08.12 | 3,88 дней | 9 дней |
| 36 | | Полное тестирование трех настроенных модулей (Минимальные требования, ЧС, КИ) | Пн 16.07.12 | Ср 25.07.12 | Вт 10.07.12 | Пн 16.07.12 | 3,88 дней | 7 дней |
| 37 | | Исправление обнаруженных ошибок при тестировании | Чт 26.07.12 | Чт 02.08.12 | Вт 17.07.12 | Вт 24.07.12 | 6,88 дней | 7 дней |
| 41 | | Настроить возможность сохранения результатов сработавшей стратегии тест-стендом | Пн 16.07.12 | Вт 17.07.12 | Вт 10.07.12 | Ср 11.07.12 | 3,88 дней | 4 дней |
| 42 | | Посчитать скоринговый бал клиента | Пн 16.07.12 | Пт 17.08.12 | Вт 10.07.12 | Пн 06.08.12 | 3,88 дней | 9 дней |
| 48 | | Sprint 4 | Пн 20.08.12 | Чт 13.09.12 | Вт 07.08.12 | Ср 29.08.12 | 8,88 дней | 11 дней |
| 49 | | Внести изменения отображение результатов на окно андеррайтера | Пн 20.08.12 | Пт 31.08.12 | Вт 07.08.12 | Пн 20.08.12 | 8,88 дней | 9 дней |
| 55 | | Настроить транслитерацию результатов автоматизированной системы принятия решений для отображения во фронт-офисном решении | Пт 24.08.12 | Ср 05.09.12 | Пн 13.08.12 | Чт 23.08.12 | 8,88 дней | 9 дней |
| 59 | | Функциональность анализа результатов проведенной андеррайтером телефонной верификации клиента | Пн 20.08.12 | Чт 13.09.12 | Вт 07.08.12 | Ср 29.08.12 | 8,88 дней | 11 дней |

Рис. 3.7. Змодельовані відхилення фактичного плану від базового за методологією "скрам"

Порівняння базового і фактичного планів:

Проаналізувавши тривалості базового і фактичних планів, можна помітити, що відхилення по тривалості $\Delta t = 18{,}6\%$ ($t_{баз} = 123$ дня, $t_{факт} = 146$ днів).

Так як проект, змодельований за допомогою методології «скрам", не передбачає наявність запитів на зміну, час виконання яких оцінюється в Водоспадної моделі менеджером проекту, важливо відзначити, що відхилення по вартості проекту з пропонованої методологією невеликі, так як відбувається тільки доплата за дні затримки . Базова вартість проекту оцінюється в 915 487 рублів, а фактична - в 1 076 171 рубль, таким чином, відхилення по вартості $\Delta c = 17{,}5\%$

Як було неодноразово зазначено раніше, ітеративний метод управління проектами хороший тим, що уточнення і демонстрація чергової версії ПЗ відбувається досить часто, після закінчення чергової ітерації. При такому підході, очевидно, що виникають на ранніх етапах помилки можуть бути

відразу ж виправлені (на відміну від Водоспадної моделі, де помилки можна виявити тільки на етапі тестування). Крім того, одержуваний в кінці проекту продукт більше відповідає вимогам замовника, ніж «чорний ящик», як у випадку водоспаду. Відповідно, оперуючи даними судженнями, можна відзначити, що відхилення за якістю продукту, отриманого за допомогою методології «скрам» набагато менше відхилення за якістю продукту, отриманого за допомогою Водоспадної моделі:

Δω_ "скрам" <Δω_водопад

У таблиці 3 представлено співвідношення загальних критеріїв двох моделей: Водоспадної і «скрам»:

Таблиця 3.1

Порівняння показників

| **Модель** | **Якість** | **Тривалість (дні)** | | **Відхилення (%)** | **Вартість(руб)** | | **Відхилення (%)** |
|---|---|---|---|---|---|---|---|
| | | **Базова** | **Фактична** | | **Кошторисна** | **Фактична** | |
| Водоспадна | $\Delta\omega_{"скрам"} < \Delta\omega_{водопад}$ | 102,5 | 191 | 86,3% | 937 600 | 1 260 800 | 34,5% |
| «Скрам» | $\Delta\omega_{"скрам"} < \Delta\omega_{водопад}$ | 123 | 146 | 18,6% | 915 487 | 1 076 171 | 17,5% |

Незважаючи на те, що базова тривалість проекту, змодельованого за допомогою методології «скрам» більше базової тривалості проекту, реалізованого за допомогою Водоспадної моделі, відхилення і по тривалості і по вартості проекту, змодельованого за допомогою методології «скрам» набагато менше, ніж у Водоспадної моделі при очевидному виграші для впровадженого ПЗ.

# ВИСНОВКИ ДО РОЗДІЛУ 3

В роботі впроваджувалося спеціалізоване програмне забезпечення для автоматизованої оцінки кредитоспроможності клієнта і для прийняття рішення про видачу кредиту. В якості замовника розробки і впровадження ПЗ виступав безпосередньо Банк, в якості виконавця - постачальник програмного забезпечення для фінансового ринку.

Впроваджувана програма являє собою промислову платформу, призначену для перевірки ризикових правил в процесі розгляду заявки на отримання кредиту. Впроваджувана програма дозволяє:

- Автоматизувати бізнес-процес кредитування в Банку;
- Автоматизувати сучасні методи оцінки ризику клієнта і заявки відповідно до кредитної політики Банку;
- Мінімізувати час розгляду заявки на видачу кредиту;
- Зменшити операційні ризики Банку;
- Сформувати новий продуктову пропозицію клієнту, відповідно його ризик-сегменту.

Зазначено, що впровадження такої програми в Банк зможе скоротити кількість людського впливу, що, в результаті, може мінімізувати помилки, пов'язані з похибкою працівників.

Визначено, що план проекту по впровадженню автоматизованої системи щодо прийняття рішення про видачу кредиту складався на базі традиційної моделі життєвого циклу проекту тривалістю 8 років, тобто, у вигляді «водоспаду» - послідовних фаз, де кожна фаза не може бути розпочато до завершення попередньої. Офіційна дата початку проекту - 23 травня 2012 року.

Проаналізувавши цілком проект впровадження програмного забезпечення в банк, можна відзначити наступне:

1) Головним критерієм успішності проекту була якість. Під якістю буде матися на увазі можливість виявлення і виправлення помилок на ранніх етапах проекту, а також повну відповідність результатів проекту вимогам замовника. Так як впроваджуване програмне забезпечення в майбутньому має автоматично

оцінювати клієнтів банку, при впровадженні проекту було необхідно мінімізувати операційні ризики, які могли б виникнути в подальшому. Як результат, тривалість і бюджет проекту автоматично збільшувалися через постійні доробоки програмного забезпечення (протягом 8 років). Це означає, що для замовника якість була ключовим моментом, заради якого він готовий був пожертвувати тривалістю і бюджетом;

2) В середині і наприкінці проекту команда розробників перебувала в постійно стресовому стані, так як відсутність комунікації з іншими працівниками спричинило за собою суб'єктивне розуміння технічного завдання. В результаті менеджером проекту було вирішено призначити їм додатковий обсяг роботу за рахунок Банку, але, незважаючи на це, продуктивність працівників була нижчою передбачуваної через людський фактор;

3) Відхилення по часу становить 88,5 днів (в базовому плані передбачалося 102,5 днів, фактично вийшло 191 день);

4) У плані передбачалася вартість проекту дорівнює 937 600 рублів, фактична склала 1 465 600 рублів, з яких 323 200 рублів - запити на зміни, відповідно, вартість проекту для Банку дорівнює 1 260 800 рублів

Проаналізувавши недоліки Водоспадної моделі при її використанні для управління IT-проектами в банку, а також переваги «гнучких» методологій розробки та вибравши методологію «скрам» як найбільш підходящу методологію для управління IT-проектом з огляду на її формалізації і організованості, була змодельована ситуація впровадження програмного забезпечення для прийняття рішення про видачу кредиту з використанням методології «скрам».

# ВИСНОВКИ

Таким чином, після проведення дослідження можна виділити наступні результати:

1. Були проаналізовані визначення термінів «проект» і «IT-проект» для виявлення відмінних характеристик останнього. Було з'ясовано, що управління IT-проектом набагато відрізняється від управління проектом, не пов'язаним з інформаційними технологіями.

2. Були проаналізовані критерії-показники, що свідчать про ефективність проекту (бюджет, тривалість, область охоплення, якість); було виявлено, що зміна одного з цих показників має істотний вплив на всі інші;

3. Було розглянуто поняття «життєвий цикл IT-проекту» і були проаналізовані переваги і недоліки основних моделей: Водоспадної (традиційної), ітеративної і спіральної. В результаті було з'ясовано, що сильні сторони ітеративної і спіральної моделей стали основою створення «гнучких» методологій;

4. Була проаналізовано специфіку предметної області дослідження (банківської сфери) на предмет застосування тієї чи іншої моделі життєвого циклу при управлінні IT-проектом. Як наслідок, була висунута гіпотеза про найбільш ефективне застосування «гнучкою» методології управління IT-проектом, яка є найкращим синтезом ітеративної і спіральної моделей;

5. В результаті розгляду характеристик і переваг «гнучких» методологій управління IT-проектами, а також після порівняльного аналізу видів «гнучких» методологій, була обрана методологія «скрам», що є найбільш чітко прописаною і більш придатною для відносин типу «замовник-виконавець» ;

6. У роботі був розроблений базовий план проекту впровадження спеціалізованого програмного забезпечення для автоматизованої оцінки кредитоспроможності клієнта і для прийняття рішення про видачу кредиту в Банк «XXX», заснований на Водоспадній моделі життєвого циклу. Після порівняння термінів і вартості базового проекту з фактично реалізованим було

виявлено, що для мінімізації можливих відхилень по необхідній замовником якості ПЗ тривалість проекту була збільшена на 86,3% а вартість на 34,5%.

7. Для моделювання проекту за запропонованою методологією «скрам» було проведено опитування у команди розробників, з метою визначення ними тривалості виконання завдань і максимально можливих відхилень. За цією методологією «скрам» був змодельований проект впровадження спеціалізованого програмного забезпечення в Банк.

Також був змодельований план «скрам»-проекту, що задовольняє якість і вимоги замовника з урахуванням максимально можливих відхилень по тривалості вирішення завдань. В результаті відхилення по тривалості виконання проекту склали 18,6%, а за вартістю 17,5% при практично мінімальних відхиленнях від якості.

Проведений аналіз показує, що в умовах швидко мінливих і нечітко визначених вимогах (такі умови як раз-таки властиво банківській сфері), а так само при низькому рівні комунікації між відділами компанії, використання методології «скрам» в управлінні IT-проектами є найбільш ефективним як для вендора, так і для замовника. Говорячи про вендорів, варто відзначити, що злагоджена робота, яка визначається самими фахівцями, які виконують її, набагато більше стимулює виконавців на якісне вирішення завдань, ніж постійні доручення менеджера проекту і перебування у вічному стресі. Для замовника вигода очевидна: проект впроваджується швидше, якісніше і з меншими витратами в порівнянні з традиційним методом.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. A Guide to the Project Management Body of Knowledge (PMBOK® Guide). — Third Edition: PMI Standart. — 388 p.

2. Ноздріна Л. В., Ящук В. І., Полотай О. І. Управління проектами: Підручник / За заг. ред. Л. В. Ноздріної. — К.: Центр учбової літератури, 2010. — 432 с.

3. Товб А.С., Ципес Г.Л. Стандарты и нормы в управлении проектами // Журнал «Управление проектами и программами» 2010 — № 01 (21) — С.76—79 — URL: www.grebennikov.ru

4. Маркина Т.А. Управление проектами в информационных технологиях. Учебное пособие. – СПб: Университет ИТМО, 2016. – 88 с.

5. Богданов В. Управление проектами. Корпоративная система – шаг за шагом. – М.: Манн, Иванов и Фербер, 2012. – 248 с.

6. Грей, Клиффорд Ф. Управление проектами: учебник: пер. с англ. третьего, полн. перераб. изд./ Клиффорд Ф. Грей, Эрик У. Ларсон; [науч. Ред. перевода В. М. Дудников]. – М.: Издательство «Дело и Сервис», 2007. – 608 с. – Доп. тит. л. англ.

7. Дитхелм Г. Управление проектами. В 2 т. Т. I. пер. с нем. – СПБ.: Издательский дом «Бизнес-пресса», 2004. – 400 с.

8. Дацко М., Семенів Г. Аналіз моделей життєвого циклу проектів галузі інформаційних технологій // Формування ринкової економіки в Україні. — № 18-2008. — С. 63-69.

9. Bill Holtsnider, Tom Wheeler, George Stragand and Joseph Gee. Agile Development & Business Goals, by Elsevier Inc, 2010. – 256 p.

10. Pekka Abrahamsson, Outi Salo, Jussi Ronkainen and Juhani Warsta. Agile Software Development Methods - Review and Analysis, by Technical Research Centre of Finland, 2002. – 107 p.

11. Брукс Фредерик. Мифический человеко-месяц, или как создаются программные системы. – М, Символ-плюс, 2010. – 304 с.

12. Вольфсон Б. Гибкие методологии разработки, версия 1.2 (электронная книга), 2012.

13. Грекул В. И., Коровкина Н. Л., Куприянов Ю. В. Методические основы управления ИТ-проектами. - Интернет-университет информационных технологий, 2011. – 392 с.

14. Гробовцов Г. Я. УПРАВЛЕНИЕ ПРОЕКТОМ: Учебно-методический комплекс. – М.: Изд. центр ЕАОИ, 2009. – 288 с.

15. Дитхелм Г. Управление проектами. В 2 т. Т. II. пер. с нем. – СПБ.: Издательский дом «Бизнес-пресса», 2004. – 288 с.

16. Мазур И. И., Шапиро В. Д., Ольдерогге Н. Г. Управление проектами: Учебное пособие / Под общ. ред. И.И. Мазура. — 2-е изд. — М.: Омега-Л, 2004. — с. 664

17. Портни, Стэнли И. Управление проектами для «чайников».: Пер. с англ. – М.: Издательский дом «Вилямс», 2005. – 352 с.: ил. – Парал. тит. англ.

18. Разу М. А., Бронникова Т. М., Разу Б. М., Титов С. А., Якутин Ю. В. Управление проектом. Основы проектного управления: учебник / кол. авт,: под ред. проф. М. А. Разу. – М.: КНОРУС, 2006 – 768 с.

19. Товб А. С., Ципес Г. Л. Управление проектами: стандарты, методы, опыт. – М.: ЗАО «Олимп-Бизнес», 2003. – 240 с.: ил.

20. Грекул В. Проектирование информационных систем (лекции), 2010

21. Черных Е. А. Agile Project Management и его история, М., «Менеджмент качества», 02(02)2008

22. Pekka Abrahamsson, Outi Salo, Jussi Ronkainen and Juhani Warsta. Agile Software Development Methods - Review and Analysis, by Technical Research Centre of Finland, 2002. – 107 p.

23. Gary Chin. Agile Project Management: How to Succeed in the Face of Changing Project Requirements, by AMACOM, 2004. – 229 p.

24. Charles G. Cobb. Making Sense of Agile Project Management: Balancing Control and Agility, by Wiley, 2011. – 265 p.

25. Michael Cohn. Succeeding with Agile: Software Development Using Scrum, by Addison-Wesley Professional, 2009. – 504 p.

26. Clifford F. Gray, Erik W. Larson. Project Management: The Managerial Process, 4th Edition, by McGraw-Hill/Irwin, 2008. - 608 p.

27. Mark C. Layton. Agile Project Management for Dummies, by John Wiley & Sons, 2012. – 360 p.

28. James P. Lewis. Fundamentals of Project Management (3rd Edition) by AMACOM Books, 2006. - 176 p.

29. Stanley E. Portny. Project Management For Dummies; 2nd Edition, by Willey Publishing, Inc., 2007. – 384 p.

30. Paul Roberts. Guide to Project Management, by Profile Books/The Economist, 2007. – 319 p.

31. Peter Schuh. Integrating Agile Development in the Real World, by Charles River Media, 2004. – 364 p.

32. Jason Westland, Project Management Life Cycle, by Kogan Page Ltd., 2006. – 255 p.

33. H. Frank Cervone, (2011), "Understanding agile project management methods using Scrum", OCLC Systems & Services, Vol. 27 Iss: 1 pp. 18-22

34. Dr. Alistair Cockburn, "Using Both Incremental and Iterative Development", 2008, The Journal of Defense Software Engineering pp.27-30

35. Pete Deemer and Gabrielle Benefield. THE SCRUM PRIMER An Introduction to Agile Project Management with Scrum Version 1.04, goodagile from www.goodagile.com, 2007

36. Gabrielle Benefield. Rolling out Agile in a Large Enterprise, HICSS '08 Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences, 2008.

37. Pankaj Jalote, Aveejeet Palit, Priya Kurien, V.T. Peethamber, "Timeboxing: a process model for iterative software development", The Journal of System and Software (2004), pp. 117-127

38. Yu Beng Leau, Wooi Khong Loo, Wai Yip Tham and Soo Fun Tan. “Software Development Life Cycle AGILE vs Traditional Approaches”, International Conference on Information and Network Technology (ICINT 2012)

39. Louise Ledbrook, “Waterfall Project Management: An Overview”, 2012, http://projectcommunityonline.com/waterfall-project-management-an-overview.html

40. Xavier Amatriain Rubio, Gemma Hornos Cirera. Agile Methods in Research (presentation), by Telefonica, 2008. – 42 s.

41. Ken Schwaber, Jeff Sutherland, “The Scrum Guide”, 2011. – 16 p.

42. Melonfire, “Understanding the pros and cons of the Waterfall Model of software development”, 2006, http://www.techrepublic.com/article/understanding-the-pros-and-cons-of-the-waterfall-model-of-software-development/6118423