STEP IT ACADEMY

КЕРУВАННЯ ПРОГРАМНИМИ ПРОЄКТАМИ

УРОК 2

ДОКЛАДНІШЕ ПРО КЕРУВАННЯ ПРОЄКТОМ

ЗМІСТ

1. ПРОЄКТ

Складові управління проєктом

Управління проєктом можна уявити, як сукупність дій, які виконує керівник проєкту, щоб зробити проєкт успішним.

Коли говорять про успішність проєкту, зазвичай мають на увазі, що успішний проєкт той, що досяг цілей, уклався в затверджені терміни та бюджет.

У дослідженні PMI (американського Інституту управління проєктами) 2018 року наводиться наступна статистика успішності проєктів (рис. 1):

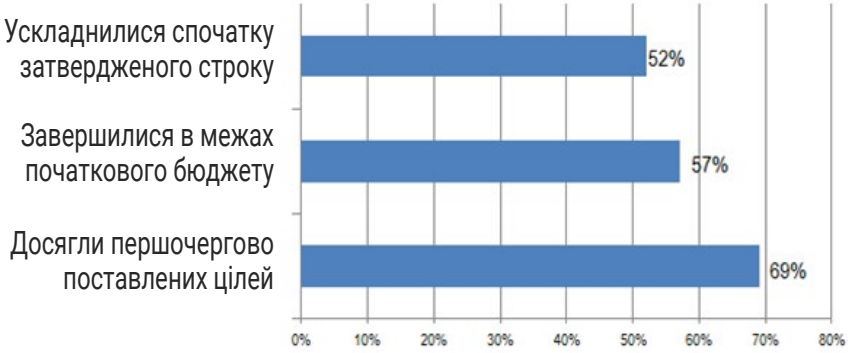


Рисунок 1

Для того щоб реалізувати успішний проєкт, керівнику проєкту потрібно знати та вміти досить багато, але в першу чергу йому потрібно розуміти, як працює «трикутник проєкту».

Параметри проєкту

Для того щоб описати параметри проєкту, в управлінні проєктами використовується термін **«трикутник проєкту»**, відомий також як «трикутник обмежень проєкту» або «залізний трикутник проєкту».

Будь-який проєкт має бути обмежений такими параметрами (рис. 2):

Рисунок 2

1. **Зміст проєкту (функціональність).**
   Зміст ІТ-проєкту зазвичай формується так:
   Мета проєкту > Результати проєкту > Вимоги до результатів проєкту.

Оскільки з формулюванням цілей в більшості проєктів виникають складнощі (нижче буде наведено діаграму з описанням чинників невдач проєктів), тому для форму-

лювання мети проєкту рекомендують використовувати критерії **SMART**.

Мета має бути:

- **S (Specific)** – конкретной. конкретною. Іноді розшифровують ***S*** як ***significant*** (***суттєвою***).

Конкретність має на увазі використання такого формулювання, яке дозволяє зрозуміти, яким має бути результат (результати) при досягненні мети. При цьому у формулюванні мети слід уникати «процесного» опису. Наприклад, мета: «Безперервно покращувати процес тестування» звучить, як процес, а мета: «Поліпшити процес тестування» звучить, як опис результату.

- **M (Measurable)** – вимірною, мають вказуватися кількісні параметри для вимірювання ступеня досягнення мети.

В опис мети додаємо показник, за допомогою якого можна виміряти ступінь її досягнення, і отримуємо: «Знизити показник "Частка повторно відкритих багів" на X відсотків».

- **A (Achievable (Ac), Ambitious (Am)** – з одного боку, досяжною, і з іншого – амбітною.

Необхідно перевірити, чи вистачає ресурсів для досягнення мети та чи є мета настільки надихаючою, щоб її досягнення стало викликом.

- **R (Relevant)** – співвідносною з іншими цілями.

Зазвичай для перевірки на релевантність необхідно у стратегічному документі компанії знайти стратегічну мету, на досягнення якої впливає ця мета.

- **T (Time bound, Trackable)** – визначеною та відстежуваною в часі.

Додаємо до формулювання мети обмеження за часом і отримуємо: «Знизити показник "Частка повторно відкритих багів" на X відсотків за перші півроку з моменту впровадження автоматизованих тестів».

Вважається, що вперше термін **SMART** узвичаїв відомий гуру менеджменту **Пітер Друкер** у 1954 року у своїй книзі «**The Practice of Management**». Не відомо, чи підбирав Друкер абревіатуру так, щоб вона відповідала англійському слову ***smart*** (у перекладі – ***розумний***) чи це вийшло випадково, але нині, у науковій літературі з менеджменту – це єдиний загальновизнаний підхід до визначення цілей (крім підходів з НЛП).

Оскільки після формулювання мети керівник проєкту і замовник зазвичай формулюють результати проєкту, а потім встановлюють термін і бюджет проєкту, тому частина критеріїв SMART може додаватися лише після розрахунку планових термінів та бюджету проєкту.

Наприклад, мета IT-проєкту може бути сформульована так: «Розробити програмний продукт, за допомогою якого компанія зможе збільшити середню прибутковість угод із клієнтами на 5% протягом першого року з моменту введення у промислову експлуатацію».

У такому формулюванні мети дотримуються два з п’яти критеріїв: **S (Specific)** та **M (Measurable)**.

Для перевірки того, чи є мета амбітною і при цьому досяжною, керівнику проєкту треба створити модель розкладу проєкту і перевірити, якою кількістю ресурсів і за який термін можна реалізувати проєкт. Після прогнозу термінів та бюджету проєкту ціль буде відповідати критеріям **A (Achievable (Ac)**, **Ambitious (Am)** і **T (Time bound)**.

Для перевірки критерію **R (Relevant)** керівнику проєкту необхідно переконатися, що ця мета є однією зі стратегічних цілей компанії і внесена до стратегічного документа компанії, або дозволяє досягти одній зі стратегічних цілей.

Результатами даного IT-проєкту стануть:

- працюючий програмний продукт;
- документація на продукт (для розробників);
- матеріали навчальних програм для користувачів;
- навчені роботі в програмному продукті та за новими процесами співробітники компанії;
- дані про середню прибутковість угод із клієнтами за період, що дорівнює одному року з моменту введення програмного продукту в промислову експлуатацію.

Вимоги до програмного продукту будуть описані у вигляді списку необхідної функціональності від програмного продукту і оформлені в документі Software Requirements Specification.

Для проєкту розробки програмного продукту зміст буде обмежено вимогами, описаними в Software Requirements Specification, які зрештою трансформуються у WBS-проєкт і список завдань проєкту. Як тільки всі вимоги

будуть реалізовані, проєкт можна завершувати і результати здавати замовнику проєкту.

2. **Бюджет проєкту (вартість проєкту)** – грошова сума, яку замовник проєкту готовий витратити на досягнення цілей проєкту та отримання очікуваних результатів.

Плановий бюджет ІТ-проєкту зазвичай формується з кількох статей витрат. Статті витрат ІТ-проєкту можуть бути наступними:

- витрати на оплату праці команди проєкту;
- податки із заробітної плати;
- закупівля «заліза» (комп'ютерів, серверів, планшетів тощо);
- придбання ліцензійного програмного забезпечення (СУБД, операційні системи тощо);
- резерв на ризики проєкту;
- плановий прибуток проєкту.

Найбільшою статтею витрат для команди виконавця ІТ-проєкту, як правило, є стаття «Витрати на оплату праці».

3. **Термін реалізації проєкту** – тривалість проєкту в часі, яка розраховується як кількість днів, починаючи з моменту офіційного старту проєкту і закінчуючи датою офіційного фінішу проєкту.
4. **Якість** – ступінь виконання вимог до проєкту. Зміна будь-якої сторони трикутника призводить до зміни рівня якості.

Як працює «трикутник проєкту»?

- Щоб скоротити терміни проєкту, можна залучити більше ресурсів (це збільшить вартість проєкту) або відмовитися від частини вимог (зменшення змісту проєкту).
- У разі відхилення від планових термінів проєкту можна було б запропонувати виконавцям працювати понаднормово за вищими ставками, проте це призведе до збільшення бюджету проєкту.
- Щоб зробити продукт проєкту більш привабливим для споживачів можна додати до нього раніше не заплановані функції, проте це призведе до необхідності змінити крайній термін проєкту або залучити додаткових співробітників, щоб залишитися в межах термінів (але це призведе до збільшення бюджету).

Треба розуміти, що універсального визначення терміна «якість» для трикутника проєкту не існує. Для кожного проєкту «якість» визначається замовником проєкту. Для одного замовника «якість означає «вкластися в бюджет», а для іншого важливіше у плановий термін вивести новий продукт на ринок. Керівнику проєкту важливо на початку проєкту дізнатися у замовника проєкту, як він трактує термін «якість» і що для нього важливіше: виконати проєкт у строк, вкластися у бюджет або реалізувати всі заплановані продукти (результати) і їх функціональність.

Замовнику проєкту слід обрати одну із стратегій, зображених на рисунку 3.

Рисунок 3

На місці замовника більшість з нас хоче зробити проєкт ***Швидко-Дешево-Якісно***, проте частка таких проєктів вкрай мала, тому цю стратегію називають утопічною.

Для визначення обмежень щодо проєкту, пропонуємо наступну послідовність дій:

1. Визначити із замовником мету проєкту. Перевірити формулювання мети на відповідність критеріям SMART.
2. Обговорити із замовником і зафіксувати заплановані результати проєкту.
3. Видокремити (виконати збір) та проаналізувати вимоги до результатів проєкту.

4. Виходячи з вимог до результатів проєкту, здійснити проєктування WBS-проєкту.
5. Спрогнозувати обсяг робіт за всіма завданнями з WBS та за проєктом в цілому.
6. Отримавши прогнози щодо обсягів робіт завдань проєкту, спрогнозувати термін реалізації проєкту (наприклад, використовуючи метод критичного шляху).
7. Розрахувати бюджет на проєкт.
8. Оформити домовленості стосовно змісту, термінів, бюджету проєкту в договорі або Статуті проєкту.

Фактори, які впливають на вартість проєкту

Вартість ІТ-проєкту багато в чому залежить від обсягу робіт, який необхідно зробити, щоб реалізувати запланований зміст проєкту, і ставок людино-годин учасників проєкту.

Наприклад, якщо обсяг робіт за проєктом оцінюється командою проєкту в 1000 людино-годин, з яких 350 годин – це робота програмістів, 250 годин – робота тестувальників, 200 годин – завдання бізнес-аналітиків, 100 годин – робота керівника проєкту та 100 годин – робота технічного письменника, то, знаючи ставки співробітників, можна розрахувати витрати на їхню заробітну плату в проєкті. Стаття «Витрати на заробітну плату команди проєкту» є найбільшою статтею витрат у бюджетах більшості ІТ-проєктів.

Принципи оцінки вартості проєкту

Для оцінки вартості проєкту керівнику ІТ-проєкту необхідно оцінити обсяги робіт з проєкту.

У розробці програмного забезпечення є багато підходів для оцінки обсягів робіт за проєктом, умовно їх можна поділити на дві групи:

1. **Традиційні підходи до оцінки.**
2. **Підходи до оцінки, які використовуються в Agile.**

До **традиційних** підходів належать:

- модель COCOMO II;
- Functional Point Analysis (FPA);
- метод оцінки за трьома точками (PERT).

Модель COCOMO II

Оцінка трудомісткості проєкту ґрунтується на аналізі результатів дослідження 63 проєктів розробки програмного забезпечення, виконаний **Баррі Боемом** (***Barry Boehm***) та опублікований у 1981 році.

Через 10 років з'явилася нова версія COCOMO II, що враховує з того часу нові життєві цикли проєктів розробки ПЗ та використовує більший набір даних.

Обсяг робіт з проєкту під час використання моделі COCOMO II залежить від такого показника, як кількість оціночних тисяч рядків коду (***KLOC – kilo lines of code***) та від поправочних коефіцієнтів, виведених Б. Боемом на підставі аналізу 63 реалізованих проєктів.

Є три рівні оцінки, які використовуються в COCOMO II:

1. **Базовий рівень оцінки** – не враховує вплив на обсяг проєкту ряду факторів, які складно врахувати на ранніх стадіях розробки, наприклад, досвід команди в розробці ПЗ.
2. **Середній рівень оцінки** – розмір проєкту залежить від розміру програмного продукту, оцінки характеристик продукту, характеристик проєкту, характеристик учасників проєкту та характеристик апаратного забезпечення.
3. **Детальний рівень оцінки** – враховує вплив окремих фаз проєкту на його трудомісткість і, відповідно, вартість.

Functional Point Analysis (FPA)

Для використання FPA керівник проєкту повинен мати функціональні вимоги до програмного продукту.

Усі функціональні вимоги поділяються на п'ять категорій: виходи, запити, входи, внутрішні файли та зовнішні інтерфейси. Кожній функції надається оцінне число функціональних балів залежно від її складності. Оцінка трудомісткості функції розраховується за такою формулою:

$$FP = UAF \times VAF$$

в якій:
FP – функціональна точка;
UAF – некоригована функціональна точка;

VAF – коефіцієнт поправки.

Оцінка всього проєкту виходить із додавання оцінок усіх функціональних точок.

Метод трьох точок (PERT – Program/Project Evaluation and Review Technique)

Для оцінки трудомісткості одного завдання проєкту використовують три оцінки (див. рис. 4):

O – оптимістична оцінка трудомісткості завдання (якщо все піде добре);

P – песимістична оцінка трудомісткості завдання (якщо все піде погано і трапляться всі можливі проблеми);

M – найбільш вірогідна оцінка.

Формула, яка використовується для розрахунку PERT-оцінки:

$$E = (O + 4*M + P) / 6$$

Оптимістичну оцінку трудомісткості можна отримати у виконавця завдання, якого керівник проєкту вважає оптимістом: виконавець найчастіше не вкладається у встановлений термін виконання завдань.

Песимістичну оцінку робить потенційний виконавець завдання, який найчастіше прогнозує оцінку вище, ніж його колеги.

І найбільш вірогідну оцінку трудовитрат можна отримати зі статистики за фактичними трудовитратами для аналогічних завдань з раніше виконаних проєктів.

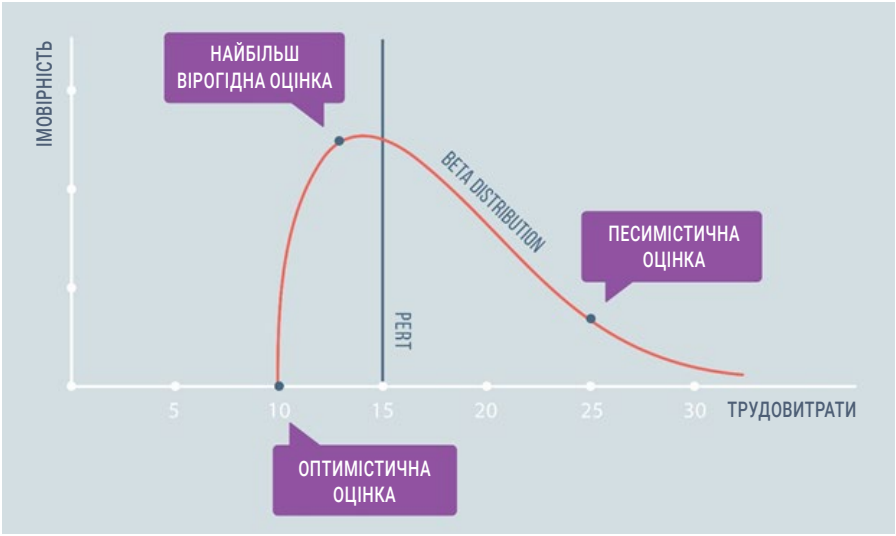


Рисунок 4

Розрахувавши оцінки PERT для всіх завдань проєкту, керівник проєкту може отримати прогнозну оцінку трудомісткості всього проєкту.

Для цього потрібно вивести суму з отриманих за формулою PERT оцінок трудомісткостей за всіма завданнями проєкту і додати резерв на непередбачені обставини. Цей резерв можна спрогнозувати за допомогою розрахунку **середнє квадратичне відхилення** по проєкту: підрахувати середнє квадратичне відхилення (СКО) з кожного завдання проєкту, отримане значення піднести до квадрата, після чого добути квадратний корінь із суми квадратів СКО за всіма завданнями.

У разі, якщо керівник проєкту хоче отримати прогноз трудомісткості проєкту з точністю до 95%, потрібно використовувати такі формули (де ***i*** – номер завдання) (див. рис. 5).

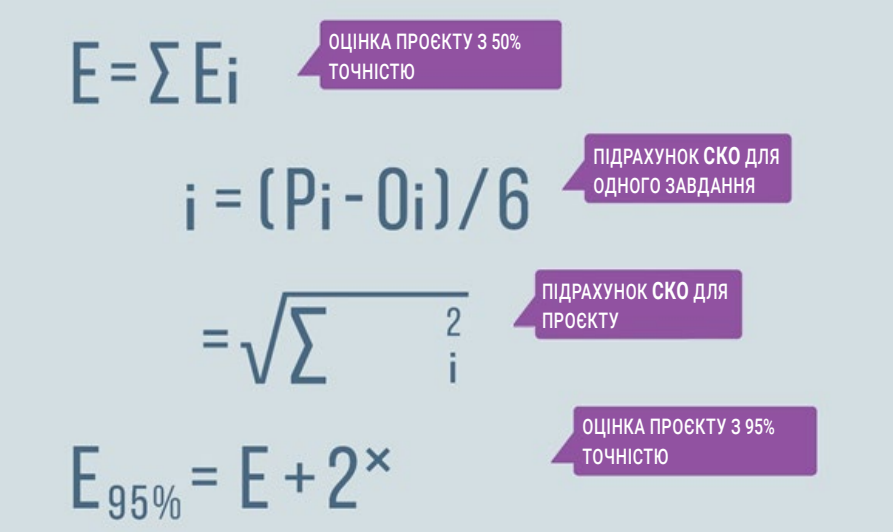


Рисунок 5

Підходи до оцінки обсягів робіт, які використовуються в Agile

Покер планування

Покер планування являє собою модифікацію **методу Дельфі**. У методі Дельфі оцінка ґрунтується на думці групи експертів, при цьому, дуже важливо позбавитися так званого ефекту «якорування». **Д. Канеман** так описує ефект «якорування»: «*Ефект якорування виявляється тоді, коли перш ніж оцінити невідоме значення, учасники досліджень стикаються з довільним числом. Цей дослід загалом дає один із найнадійніших і найстабільніших результатів в експериментальній психології: оцінки не віддаляються від запропонованого піддослідним числа, іншими словами – якоря*».

Для того щоб отримати групову незалежну оцінку, ніхто з експертів не повинен називати свою оцінку, поки всі не зроблять вибір, і тільки після цього оголошуються результати.

Як відбувається покер планування?

Учасники оцінки (потенційні виконавці завдань) і модератор наради збираються в одному приміщенні. Кожен учасник, який виконує оцінку трудовитрат завдань проєкту, має колоду карт, наприклад, таку (рис. 6):

Рисунок 6

Значення на картах даної колоди відповідають ряду Фібоначчі, в якому для визначення наступного значення ряду використовується правило: сума двох попередніх чисел має давати наступне число. Є декілька карт, які використовуються скоріше як «розважальні»: наприклад,

карта зі знаком «?», експерт витягує її у разі, якщо до кінця не розуміє вимоги до результатів завдання, або карта із зображенням «чашки», яка використовується, якщо експерт втомився від оцінки і бажає піти на перерву.

Є кілька карт з цифрами, які не відповідають ряду Фібоначчі: 20, 40 та 100. Якщо учасник оцінки витягує такі карти, то цим він демонструє свою думку про те, що завдання занадто масштабне. У цьому випадку модератору мітингу рекомендовано декомпозувати завдання на дрібніші підзавдання та оцінювати їх.

Процедура використання карток з оцінками наступна.

Модератор зачитує побажання (або формулювання завдання), оцінювачі ставлять питання щодо вимог до побажання, і замовник проєкту відповідає на їх питання.

Після того, як усі відповіді отримані, модератор просить кожного учасника зробити оцінку трудовитрат. Продумавши оцінку, експерт повинен витягнути картку з цифрою, що відповідає цій оцінці, та покласти її зворотною стороною, щоб ніхто не бачив його оцінку. Тобто, якщо експерт оцінює реалізацію завдання у 8 годин, він витягне картку з цифрою 8 і покладе її зворотною стороною.

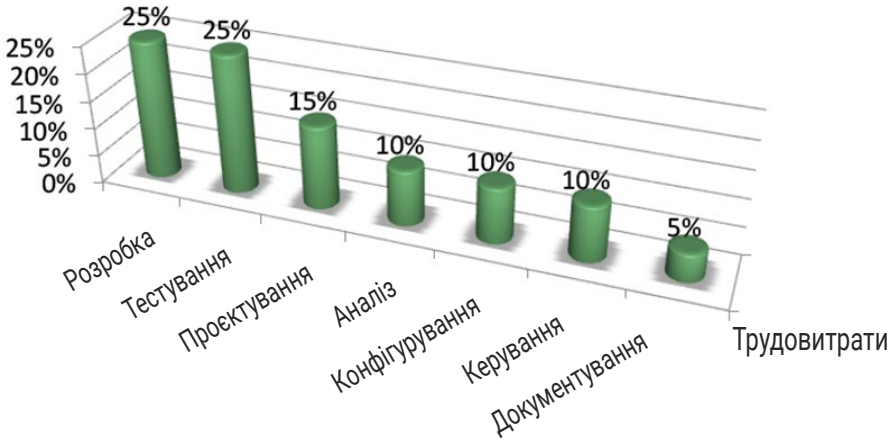
Після того, як експерти витягли карти, вони їх перевертають і модератор оголошує результати. В результаті група отримує діапазон оцінок, який може бути дуже широким. Для наближення оцінок експертів, модератор просить пояснити свою думку того експерта, в якого вийшла найнижча оцінка. Експерт пояснює думку і часто виявляється, що він неправильно зрозумів вимоги. Після цього модератор просить висловитися експерта,

який зробив найвищу оцінку, і часто виявляється, що він вибрав не найоптимальніший підхід до реалізації. Вислухавши думки двох експертів, як правило, вимоги стають більш зрозумілими, і в ході колективного обговорення група обирає єдиний підхід до реалізації. Модератор пропонує учасникам оцінки проголосувати вдруге. Після другого туру голосування діапазон оцінок експертів зменшується. Можна провести й третій тур голосування, однак, другого туру зазвичай достатньо, щоб усереднити оцінки експертів і отримати одну оцінку трудомісткості реалізації даної задачі.

Зрештою, всі завдання проєкту (або ітерації проєкту) одержують свої оцінки. Підсумувавши ці оцінки, керівник проєкту отримує оцінку трудомісткості всього проєкту (або ітерації).

Як правило, команда оцінює лише ті завдання, які пов'язані з розробкою (програмуванням). В індустрії розробки програмного забезпечення є діаграма (див. рис. 7), яка дозволяє зробити загальну оцінку робіт з проєкту, пов'язаного з розробкою ПЗ.

З цієї діаграми бачимо, що в загальному обсязі робіт проєкту з розробки ПЗ кодування займає приблизно 25% трудовитрат. Тому, щоб правильно розрахувати бюджет проєкту, потрібно прогнозування по трудовитратах на розробку необхідного функціоналу ПЗ помножити на 4, що в результаті дозволить врахувати решту видів робіт.



© Лекції з управління програмними проектами,
Сергій Архіпенков

Рисунок 7

Після того як ми зробили оцінку обсягів робіт, залишається помножити оціночні години розробників, тестувальників, бізнес-аналітиків і керівника проєкту на їх ставки людино-години, і отримаємо основну статтю витрат у бюджеті проєкту – заробітну плату учасників проєкту. Далі керівнику проєкту потрібно розрахувати планове значення за іншими статтями витрат:

- податки із заробітної плати – як правило, частка податків від заробітної плати – це відома цифра помножена на отримане значення за статтею «Заробітна плата»;
- закупівля «заліза» (комп'ютерів, серверів, планшетів тощо) – цю статтю бюджету необхідно погодити із

замовником після збору та аналізу вимог до «заліза», вибору варіантів «заліза»;

- придбання ліцензійного програмного забезпечення (СУБД, операційні системи, програма для керування проєктами і т. д.) – для деяких проєктів витрат за цією статтею може не бути, оскільки все необхідне ПЗ вже є у замовника;
- резерв на ризики проєкту – зазвичай, цей резерв можна розрахувати після аналізу ризиків проєкту або розрахувати, використовуючи відсоток від витрат на заробітну плату команди проєкту;
- накладні витрати на утримання офісу;
- плановий прибуток проєкту – визначається керівництвом компанії, яка виконує проєкт.

Приклад розрахунків оцінки вартості

Команда проєкту розробки програмного продукту від замовника проєкту отримала технічне завдання на розробку програмного продукту з описом процесів, описом сценаріїв використання та вимог до програмного продукту. Завдання команди проєкту полягає в тому, щоб розрахувати бюджет проєкту.

Перше, що треба зробити, – розробити список робіт за проєктом. Як правило, у керівника проєкту не вистачає компетенцій, щоб зробити детальний список робіт, тому він запрошує компетентних у предметній галузі експертів і просить їх зробити перехід від функціональних вимог до детального списку завдань.

Можна віддати вимоги на вивчення системному архітектору, щоб він «накидав» чорновий варіант архітектури продукту, після чого команда визначить список завдань. Потім команда робить оцінку трудомісткості методом покерного планування і отримує оцінку трудомісткості завдань з розробки необхідних функцій.

Припустимо, отримана оцінка робіт з розробки в результаті покер планування показала, що для кодування команді потрібно буде 1000 людино-годин.

На основі отриманих даних, розрахуємо планове значення статті витрат «Зарплати команди проєкту»:

Вид робіт	Трудовитрати, людино-годин	Ставка людино-годин на даний вид робіт, $	Вартість, $
Кодування функціоналу, описаного у ТЗ	1000	15	15 000
Тестування	1000	12	12 000
Проєктування ПЗ	600	20	12 000
Аналіз вимог	400	14	5 600
Конфігурування	400	15	6 000
Керування проєктом	400	20	8 000
Документування	200	10	2 000
Разом	4000	–	60 600

Рассчитаем оставшиеся статьи затрат по проекту:

Стаття витрат	Формула розрахунку	Вартість, $	Коментар
Заробітна плата учасників проєкту		60 600	
Податки із заробітної плати	45% від статті «Заробітна плата»	27 270	
Закупівля «заліза»	За вартістю рахунку на придбання товару	10 000	За інвойсом від постачальника сервера
Придбання ліцензійного програмного забезпечення		0	У цьому проєкті закупівля не потрібна
Резерв на ризики проєкту	10% від заробітної плати учасників	6 060	
Плановий прибуток проєкту	25% від заробітної плати учасників	15 150	
Накладні витрати на утримання офісу	5% від заробітної плати учасників	3 030	
Разом: бюджет проєкту		**116 050**	

З одержаних вище розрахунків, керівник проєкту може сформувати ставку 1 людино-години роботи учасника команди проєкту для замовника проєкту:

Ставка 1 людино-години = 116 050 / 4000 = $29

Отже, бюджет проєкту розрахований, залишилося погодити його із замовником проєкту.

Планування термінів проєкту. Метод критичного шляху

Для планування термінів ІТ-проєкту можна використовувати метод критичного шляху, який передбачає використання мережевого графіка проєкту та деяких розрахунків.

Команда проєкту робить прогноз по трудомісткості завдань, а керівник проєкту отримує інформацію про те, скільки часу може приділяти завданням проєкту кожен учасник. Після цього керівник проєкту робить розрахунок термінів реалізації кожного завдання, використовуючи формулу:

Термін завдання = Трудомісткість завдання * 100 % / % доступності виконавця

в якій

% доступності виконавця – частка від робочого дня, яку виконавець може приділити роботі над завданням.

Потім команда проєкту визначає залежності в реалізації завдань і проєкту є мережевий графік. При заповненні даних у полі «Попередник», команда вказує назву завдання, яке має завершитися до старту цього завдання:

Назва завдання	Тривалість, днів	Попередник
A	2	–
B	15	A
C	10	A

Назва завдання	Тривалість, днів	Попередник
D	13	A
E	18	A
F	15	C, D
G	10	F, B
H	5	E, G

Для того щоб визначити критичний шлях, команда створює мережевий графік робіт (рис. 8):

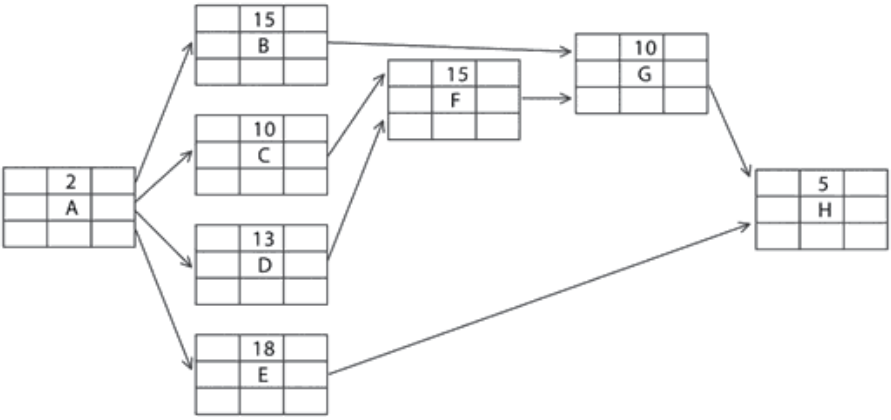


Рисунок 8

Параметри завдань на діаграмі розміщуються у прямокутнику. Легенда для опису значень клітинок прямокутника представлена нижче:

РАННІЙ СТАРТ	ТРИВАЛІСТЬ	РАННІЙ ФІНІШ
КОД ЗАВДАННЯ		
ПІЗНІЙ СТАРТ	РЕЗЕРВ	ПІЗНІЙ ФІНІШ

Послідовність виконання завдань проєкту позначається стрілками. Наприклад, завдання C може починатися тільки тоді, коли буде завершено завдання A, а завдання H стартує, коли повністю будуть завершені завдання E та G.

Щоб спростити розрахунки параметрів графіка, домовимося використовувати не календарні дати, а номери днів. Також для простоти розрахунків приймемо, що завдання A може починатися саме сьогодні, а сьогоднішній день має номер 0 (нуль).

Для завдання A розрахуємо параметри ранніх дат старту та фінішу:

Ранній старт завдання A – день із номером 0.

Ранній фініш завдання A розраховується як «ранній старт» плюс «тривалість», тобто 0 + 2 = 2-й день.

Зробимо ще одне припущення: наступне завдання можна починати у той же день, коли закінчується попереднє завдання.

Виконаємо розрахунки для завдання B:

Ранній старт: день із номером 2.

Ранній фініш: 2 + 15 = 17-й день.

У такий спосіб виконуються розрахунки дат раннього старту і раннього фінішу для всіх завдань, зазначених у мережевому графіку. У тому випадку, коли в одного завдання кілька завдань-попередників (наприклад, завдання H), при розрахунку дня раннього старту вибирається найбільше значення з наявних.

Для визначення найдовшого ланцюжка завдань у проєкті, необхідно розрахувати дати пізнього старту та пізнього фінішу за завданням.

Для їх розрахунку використовуються такі правила:

1. Пізня дата фінішу поставленого завдання дорівнює пізній даті старту задачі-послідовника.
2. Якщо завдань-послідовників декілька, як у завдання А, для визначення пізньої дати фінішу вибирається найменше значення з наявних.
3. Пізня дата старту завдання визначається як різниця між її пізнім фінішом та тривалістю.

Залишається визначити критичні завдання в графіку. Для цього слід розрахувати резерви по завданнях. Значення резерву по завданню розраховується як різниця між значенням пізнього старту завдання та значенням раннього старту поставленого завдання. Ті завдання, в яких резерв часу дорівнює нулю, називаються критичними і визначають критичний шлях проєкту. Завдання, в яких значення резерву відрізняється від нуля, не є критичними. Це означає, що для цих завдань термін реалізації може бути збільшено під час їх резерву або старт завдання можна перенести на дату пізнього старту (рис. 9).

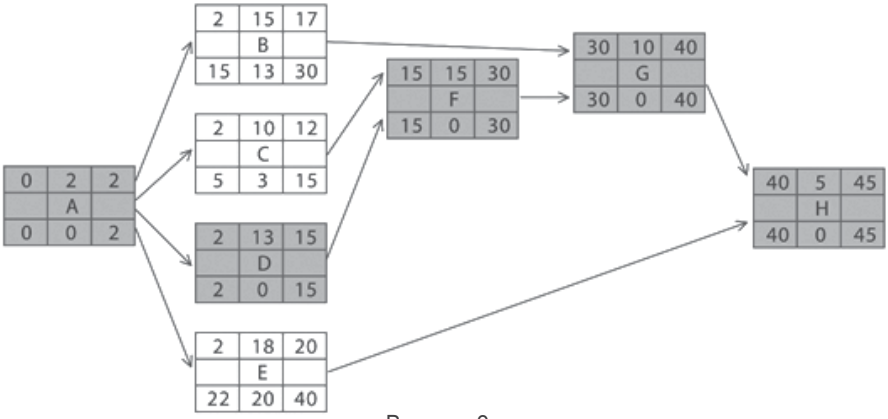


Рисунок 9

Для чого керівнику проєкту корисно знати список і параметри завдань, що лежать на критичному шляху?

- Він отримує уявлення про оптимістичне значення тривалості проєкту.
- Він розуміє, для яких завдань потрібен посилений контроль, оскільки завдання критичного шляху визначають термін всього проєкту.
- Керівник проєкту може призначити виконавців для завдань проєкту, побачити, чи є перевантаження у деяких виконавців, вирівняти навантаження і дізнатися справжнє значення критичного шляху і термінів проєкту.

Методу критичного шляху скоро виповниться 60 років, але для проєктів, в основі яких лежить водоспадна модель життєвого циклу, він досі актуальний для планування термінів проєктів.

На сьогоднішній день у світі існує безліч комп'ютерних програм, які автоматизують розрахунки параметрів мережевого графіку.

Учасники та персонал проєкту

Для того щоб підвищити ймовірність успішності ІТ-проєкту, у складі учасників проєкту мають виконуватися обов'язкові ролі.

Найчастіше в ІТ-проєктах є дві сторони проєкту: компанія-замовник та компанія-виконавець, але іноді, ІТ-проєкт робиться силами внутрішнього ІТ-підрозділу компанії. Розглянемо, як розподіляються ролі в ІТ-проєкті в першому випадку.

Учасники проєкту з боку замовника

У світі є багато підходів до керування проєктами і в кожному з них описується своє бачення того, які ролі мають виконуватися в проєкті.

Наприклад, в PMBOK описано наступну модель ролей проєкту:

Назва ролі	Відповідальність	Повноваження
Керівник проєкту	За всі параметри трикутника проєкту: досягнення цілей у запланований термін та бюджет	В проєктній організаційній структурі: підбір виконавців у команду, формування команди, планування та контроль виконання завдань
Замовник проєкту	Затвердження документа за вимогами до IT-проєкту (наприклад, SRS). Організація та виконання робіт з приймання отриманих результатів проєкту	Зміна вимог до результатів проєкту, визначення пріоритетів у реалізації вимог
Спонсор проєкту	Затвердження цілей проєкту, планових термінів та бюджету. Забезпечення того, що узгоджені виконавці для проєкту виділяють запланований час на завдання проєкту	Вирішення питань, на які не вистачає влади у керівника та замовника проєкту
Команда проєкту	Виконання завдань проєкту у погоджений термін та заплановані трудовитрати	Максимально швидко сигналізувати керівнику проєкту про виникнення проблем під час вирішення поставлених завдань

Для прикладу розглянемо проєкт із впровадження системи **ERP** (з англ. *Enterprise Resource Planning – планування ресурсів підприємства*) в деякій компанії, керівництво якої ухвалило рішення для реалізації проєкту залучити компанію, що спеціалізується на впровадженні ERP-систем.

Очевидно, що роль замовника проєкту потрібно виконувати комусь із співробітників компанії, в якій запроваджується ERP.

Чи може роль замовника проєкту виконувати декілька осіб, наприклад, усі керівники підрозділів, які використовуватимуть ERP? Ви чули приказку: «У семи няньок дитина без носа»? Ця приказка про проєкт, в якому роль замовника виконують декілька осіб, серед яких немає головного. Замовником проєкту має бути одна людина, щоправда – це не заважає йому зібрати робочу групу (або команду), з якою він радитиметься з приводу прийняття важливих рішень щодо проєкту, але відповідальність за прийняті рішень він повинен нести один.

Щоб ухвалити рішення про призначення конкретного співробітника на роль замовника у проєкті впровадження ERP, варто відповісти на такі питання:

1. Співробітники яких підрозділів компанії будуть застосовувати ERP?
2. Хто з керівників цих підрозділів найбільше зацікавлений у тому, щоб ERP-система допомогла йому покращити показники, за які він відповідає?
3. Чи може кандидат на роль замовника проєкту витрачати на нього мінімум 4-6 годин на тиждень?

Для розрахунку необхідного часу замовник проєкту, для виконання своїх функцій, може використовувати дані компанії Standish Group про те, що на кожні $ 1000 від такої статті витрат як «Вартість часу людей» в IT-проєкті доводиться ухвалювати 1,5 рішення.

Припустимо, проєкт впровадження ERP-системи оцінений у $ 250 000, з яких вартість ліцензій програмного забезпечення коробкової ERP-системи складає $100 000, а $ 150 000 складає вартість доробок та впровадження ERP (а це і є вартість часу людей).

Якщо вірити розрахункам Standish Group, то у цьому проєкті потрібно буде прийняти близько 225 важливих рішень, причому Standish Group вказує на те, що замовник має взяти участь як мінімум у 20% із цих рішень.

У результаті виходить, що замовник ухвалить рішення у 45 випадках. Припустимо, що в середньому на прийняття одного рішення замовник витрачає 4 години, виходить 45 важливих рішень по 4 години на кожне – це 180 годин.

Якщо тривалість проєкту становить приблизно 1 рік, то 180 годин, поділені на 48 тижнів, дають результат – 3,75години на тиждень замовник повинен витрачати на прийняття рішень в проєкті.

При цьому очевидно, що замовник витрачатиме час не лише на прийняття важливих рішень, оскільки йому ще потрібно:

1. Забезпечити наявність **бачення проєкту**, в якому описати, як в продукті будуть задоволені очікування різних зацікавлених сторін.

2. Погодити обмеження проєкту за термінами, бюджетом та змістом (ці обмеження мають бути прописані у договорі на проєкт або у Статуті проєкту).
3. Вибрати метод вимірювання прогресу проєкту, який, з одного боку, не повинен вимагати багато часу на вимірювання, а з іншого боку – надає правильні дані про хід проєкту.
4. Узгоджувати документи щодо вимог.
5. Організовувати роботи з прийняття розробленого функціоналу.
6. Організовувати роботи з навчання користувачів.
7. Аналізувати звіти керівника проєкту про хід проєкту.
8. Брати участь у створенні та коригуванні плану проєкту – власник продукту повинен розуміти прогнози щодо завершення проєкту (чи вистачить бюджету, чи вкладемося в строк) і приймати рішення з пріоритетів реалізації вимог до продукту проєкту.

Ось і виходить, що 4 години на тиждень – це мінімум часу, який потрібен на адекватне виконання ролі замовника проєкту ERP. Зауважимо, що в деяких IT-проєктах замовник може витрачати весь свій робочий час на виконання ролі замовника проєкту.

В деяких підходах до управління проєктами (наприклад, в Scrum), замовника проєкту називають «власником продукту проєкту», і ця назва відображає суть його діяльності: створити продукт, який буде цінним для його споживачів. Власник продукту повинен бути зацікавлений

в отриманні видатного продукту у затверджений термін і бюджет.

Чи потрібний спонсор проєкту впровадження ERP?

На наш погляд – це одна з ключових ролей в успіху проєкту. За статистикою PMI, слабка підтримка спонсора проєкту потрапила на дев'яте місце списку факторів провалу проєктів (див. рис. 10).

Рисунок 10

Спонсор (куратор) проєкту – керівник вищої ланки організації, в якій користуватимуться результатами проєкту. Спонсор (куратор) проєкту відповідає за досягнення проєктом кінцевих цілей та реалізацію вигід для організації.

У чому відмінність ролі спонсора проєкту від замовника?

1. **Спонсор проєкту** повинен мати більше влади, ніж замовник проєкту, інакше користі від нього не буде.
2. **Спонсор** повинен відповідати за створення цінності в проєкті та передачі цієї цінності у бізнес-підрозділи для експлуатації.

Спонсором проєкту впровадження ERP, на наш погляд, має стати CEO компанії або хтось із його заступників. Впровадження ERP – це дорогий проєкт, результатом якого стане зміна багатьох бізнес-процесів компанії, і якщо цьому проєкту не надати потрібного статусу, то він може виявитися провальним. Тому, якщо CEO компанії візьме на себе роль спонсора проєкту, проєкт тільки виграє від прийняття такого рішення.

Чи потрібно компанії, яка замовляє проєкт, призначати **керівника проєкту** з-поміж своїх співробітників? Відповідь залежить від того, яку частку робіт від загального обсягу робіт у проєкті виконуватимуть співробітники компанії-замовника. Якщо це менше 20-25%, то на наш погляд, керівник проєкту від замовника не потрібен, за більшої частки – його потрібно призначити, оскільки комусь треба буде планувати, організовувати та контролювати виконання завдань, які будуть доручені співробітникам компанії-замовника.

Персонал проєкту з боку фірми розробника

З боку компанії-розробника в IT-проєкті зазвичай бере участь команда, яка здатна вирішити завдання проєкту, що стоять перед нею. У цієї команди обов'язково має бути керівник проєкту. Склад інших учасників залежить від

робіт, які потрібно виконувати в проєкті. Але, зазвичай, до складу команди входять:

- розробники;
- тестувальники;
- бізнес-аналітики;
- в деяких випадках, до команди проєкту потрібні ще дизайнери інтерфейсів, системний архітектор, технічний письменник.

Чи повинен керівник ІТ-проєкту мати досвід роботи програмістом, аналітиком або тестувальником, щоб очолити команду?

На це питання є наступна відповідь: чим масштабніший проєкт, тим менше потрібно керівнику проєкту бути предметником і більше потрібно навичок в управлінні проєктами (рис. 11).

Рисунок 11

Розглянемо докладніше роль керівника проєкту.

Роль керівника проєкту та його відповідальність за проєкт по-різному трактується в різних підходах до керування проєктами. Так, підхід **MSF (Microsoft Solutions Framework)** припускає, що за успіх проєкту відповідальна вся команда, яка складається з наступних рольових кластерів:

- **керування програмою (Program manager)** – відповідає за розробку архітектури рішення, планування та контроль термінів проєкту;
- **розробка (Developer)** – відповідає за розробку продукту;
- **тестування (QAE)** – відповідає за планування, розробку тестів та звітність за тестами;
- **керування випуском (Release manager)** – відповідає за створіння інфраструктури, випуск готового продукту;
- **задоволення замовника (User experience)** – відповідає за навчання, ергономіку, графічний дизайн, технічну підтримку;
- **керування продуктом (Product manager)** – відповідає за розміщення пріоритетів, маркетинг продукту, дотримання інтересів замовника.

Виходить, що в MSF відповідальність за успіх проєкту колективна, але за реалізацію проєкту у встановлений термін і бюджет відповідає роль «Керування програмою».

У підході PMBOK за успіх проєкту відповідає керівник проєкту, а успіх вимірюється тим, чи змогла команда проєкту виконати його у визначений термін, вкластися

в запланований бюджет і досягти поставлених перед проєктом цілей.

Які компетенціями повинен мати керівник ІТ-проєкту?

На наш погляд, у першу чергу, він повинен володіти м'якими навичками, такими як:

- вміння підібрати у проєкт правильних людей і створювати з них команду;
- навички ведення переговорів;
- вміння мотивувати учасників команди;
- вміння вибрати найбільш підходящий підхід (або гібрид підходів) для керування проєктом і впровадити його у проєкт;
- навички впровадження засобів автоматизації проєктної роботи.

Вміння обирати правильних людей і створювати з них команду – це найскладніша навичка, яку потрібно придбати керівнику проєкту. Нижче ми розглянемо підходи, які можуть допомогти керівнику проєкту під час підбору людей до команди.

Принципи відбору співробітників у команду проєкту

Чим команда відрізняється від групи людей, які спільно працюють над проєктом?

На наш погляд, команда повинна мати наступні характеристики:

1. Спільна довіра членів команди, яка виявляється в тому, що кожен учасник команди впевнений, що всі інші не хочуть йому нашкодити або маніпулювати ним.

2. Конструктивні суперечки стосовно справи. Якщо всередині команди люди не сперечаються про справу – їм байдуже, а це призводить до ухвалення не найоптимальніших рішень.
3. Усвідомлення загальної відповідальності результат. Кожен учасник прагне не просто виконати свої завдання, а й надати допомогу колегам.
4. Аналіз продуктивності команди. Команда визначає метрики, з допомогою яких вона аналізує свою результативність.
5. Постійне покращення командної роботи. Учасники команди періодично зустрічаються та визначають, що ще можна і потрібно покращити, ставлять завдання та впроваджують зміни в процесах спільної роботи.

При підборі співробітників у команду важливо враховувати не лише їх професійні навички, а й особисті якості.

У багатьох компаніях для оцінки особистісних якостей працівників використовується **підхід MBTI**, який має декілька практичних аспектів:

1. Знання психотипу співробітника дозволяє менеджеру підбирати правильні канали комунікацій, спрогнозувати стиль поведінки співробітника в роботі та конфліктах.
2. Знання MBTI допомагає менеджеру розуміти особливості свого колективу і розробляти плани навчання учасників команди м'яким навичкам.
3. MBTI допомагає зрозуміти, при вирішенні яких завдань співробітник проявить себе якнайкраще, а які вирішувати йому буде важко.

Методика MBTI дозволяє визначити вподобання кожної людини за чотирма аспектами, а потім сприяти найкращому застосуванню цих вподобань у житті та роботі (рис. 12):

Рисунок 12

- **Джерело енергії: екстраверти та інтроверти.**

Екстраверт отримує енергію в комунікаціях з іншими, а інтроверт черпає її на самоті.

- **Функція збору інформації: сенсорики та інтуїти.**

Сенсорики, збираючи інформацію, довіряють своєму досвіду. У інтуїтів збір інформації заснований на припущенні про те, що будь-яка інформація може мати певний ступінь достовірності, навіть якщо подібного в досвіді не було. Інтуїт – стратег, а сенсорик – тактик.

- **Функція прийняття рішень: думаючі та чуйні.**

Логіки при ухваленні рішень не думають про те, що відчують ті, кого стосується це рішення. Чуйні – навпаки, приймають рішення з урахуванням того, як воно вплине на інших.

- **Стиль життя: раціонали та ірраціонали.**

Раціонали люблять приймати рішення, наявність відкритих питань, з яких немає відповідей викликає у них стрес. Ірраціонали віддають перевагу збору інформації для прийняття рішень і відтягують момент прийняття рішення до останнього.

Кожен з нас має схильність до того чи іншого полюса за шкалою «екстраверсія-інтроверсія», «сенсорика-інтуїція», «логіка-відчуття», «раціоналізм-ірраціоналізм». Кожна людина може проявляти себе як екстраверт в одних ситуаціях і як інтроверт – в інших, при цьому схильність до однієї з цих якостей буде вищою.

Усього в MBTI існують 16 типів (рис. 13):

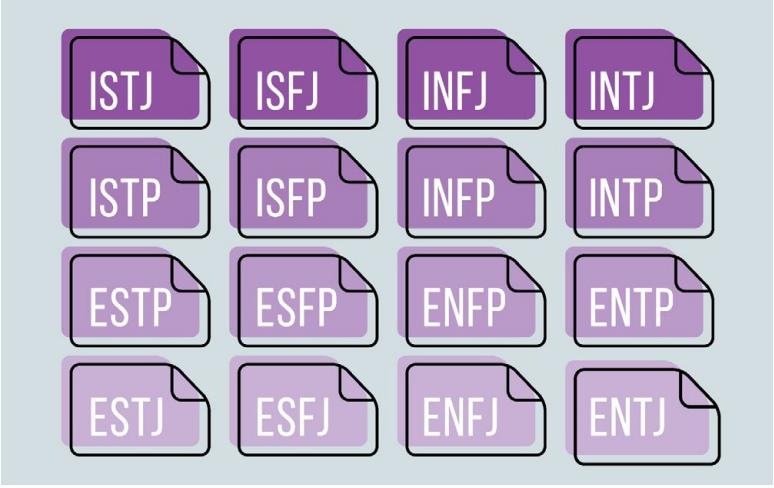


Рисунок 13

Цікаво, що близько 70% компаній Fortune 500 успішно застосовують MBTI®, це говорить про популярність підходу MBTI (https://mbtitraininginstitute.myersbriggs.org/home.htm?bhcp=1).

Ще одним важливим моментом при відборі співробітників у команду проєкту є вивчення того, що мотивує кожного потенційного учасника на роботу в даному проєкті.

Ідеально, якщо ключовим мотиватором для кандидата є не гроші, а внутрішні мотиви, такі як: бажання створити продукт світового рівня, бажання здобути новий досвід успішної роботи в команді тощо.

Керівнику проєкту необхідно володіти навичками визначення домінуючих мотиваторів потенційних учасників проєкту.

Управління командою проєкту

Що таке «управління командою»? На наш погляд, управління полягає в тому, щоб вимірювати показники результативності команди та коригувати поведінку команди так, щоб ці показники досягали свого максимуму.

При використанні фреймворку Scrum команда може вимірювати два ключові показники: **фокус-фактор** (**Focus Factor**) та **швидкість** (**Velocity**).

Приклад розрахунку Focus Factor

Команда взяла відповідальність реалізувати за один спринт 20 завдань трудомісткістю у 400 людино-годин. По завершенні спринту, команда змогла продемонструвати результат у 14 завдань з 20-ти запланованих. За плановими оцінками трудомісткості, ці 14 завдань оцінювалися

*у 200 людино-годин. **Focus factor** даного спринту склав 200 / 400 = 0,5, при цьому **velocity** команди склала 200 людино-годин.*

Якщо вимірювати цей показник після кожного спринту і відстежувати його тренд, можна судити про те, на якій стадії життєвого циклу перебуває команда проєкту. Теорію життєвого циклу команди розробив **Б. Такман**. Він виділив такі стадії групової динаміки в команді (рис. 14):

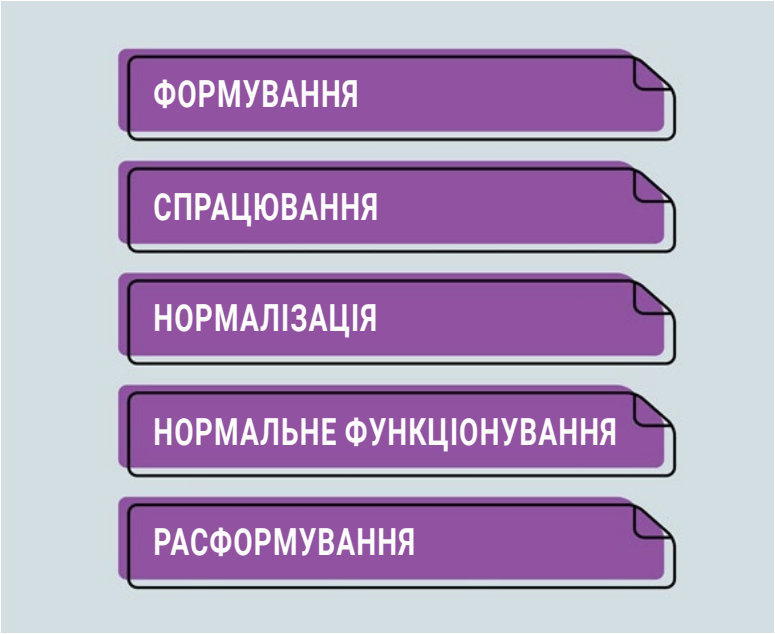


Рисунок 14

Продуктивність команди зазвичай змінюється так, як показано на рисунку 15 на на сторінці 44.

Якщо фокус-фактор команди коливається у перших спринтах від 0,3 до 0,5 і то зростає, то падає, це говорить про те, що команда перебуває на етапі спрацьовування

і ніяк не може подолати його. Якщо фокус-фактор протягом 2-3 спринтів стійко зростає, то команда, швидше за все, перейшла на етап нормалізації. І якщо фокус-фактор продовжує стійке зростання протягом 3-6 спринтів, при цьому досягає рівня 1 і знаходиться на цьому рівні протягом декілька спринтів, то можна констатувати, що команда досягла етапу нормального функціонування.

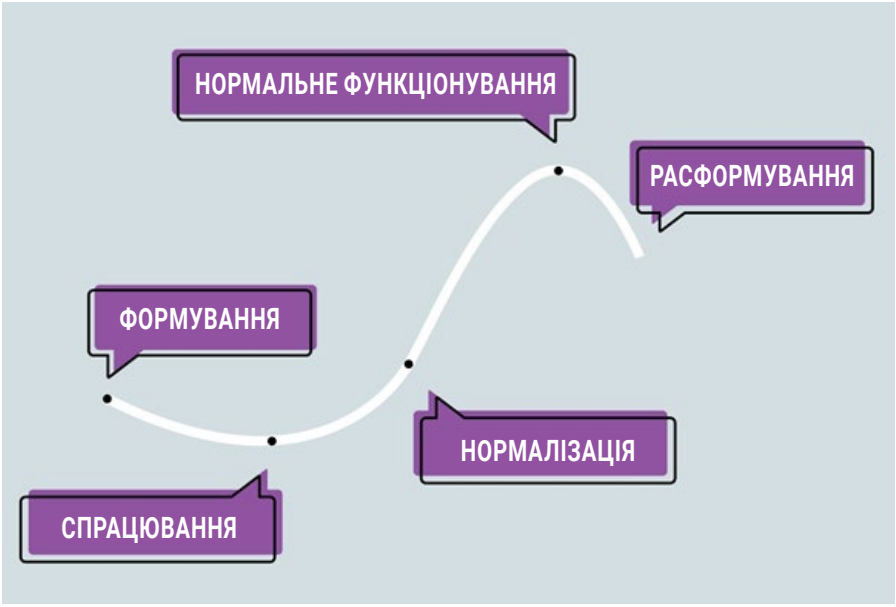


Рисунок 15

Ролі в рамках проєкту

В індустрії розробки та впровадження програмних продуктів, за десятиліття її розвитку, з'явилося чимало професій. Ці професії вплинули і на появу нових ролей в IT-проєкті:

1. **Project manager** – відповідає за реалізацію проєкту в строк, за бюджет і досягнення мети, що стоять перед проєктом.
2. **Бізнес-аналітик** – займається вилученням і аналізом вимог до програмного продукту, пошуком і аналізом проблем, виробленням рішення для виявлених проблем, документуванням вимог.
3. **Архітектор продукту** – відповідає за проєктування продукту та опис його архітектури.
4. **Розробники** – кодують вимоги так, щоб вони відповідали опису і створена функціональність продукту проходила тест-кейси.
5. **Тестувальники** – розробляють сценарії тестування вимог, проводять тестування та фіксують виявлені в ході тестування відхилення від вимог (у тому числі й помилки).
6. **Team Leader** – в деяких проєктах кількість учасників перевищує 9 осіб і керівник проєкту приймає рішення поділити їх на декілька команд. Для планування та контролю результатів команд у них вибираються Team Leader, які відповідають іноді і за розвиток потрібних для проєкту компетенцій у учасників своєї команди.

2. РИЗИКИ В ПРОЄКТІ

Що таке «ризики проєкту»?

У будь-якому ІТ-проєкті, незважаючи на те, скільки часу керівник проєкту витратив на планування, щось піде не так. Щоб не боротися з наслідками, а передбачати і не допускати проблем в проєкті, потрібно навчитися керувати ризиками проєкту.

Існує декілька визначень терміну «ризик»:

> ***Ризик** – це невизначена подія або умова, настання якої негативно чи позитивно позначається на цілях проєкту.*
>
> © PMBOK

> ***Ризик** – це вплив невизначеності для досягнення цілей.*
>
> © ДСТУ Р 56275-2014

За своєю суттю, ризик для проєкту – це те, що створить проблему у майбутньому, тому нам імпонує таке визначення: ризик – це потенційна проблема проєкту, яка може виникнути в майбутньому.

Тобто, на поточний момент ця проблема ще не виникла, але якщо вона виникне, то вплине на реалізацію проєкту у визначений термін і запланований бюджет, а можливо, й призведе навіть до того, що мета проєкту виявиться неможливою реалізувати повною мірою.

Суть управління ризиками ІТ-проєкту полягає в тому, щоб керувати «проактивно», не допускаючи виникнення проблем, а не займатися нескінченним «гасінням пожеж».

На нашу думку, більшість керівників проєкту надто покладаються на свою інтуїцію та досвід при плануванні проєкту і не хочуть подивитися на проєкт «без рожевих окулярів».

Якщо повернутися до дослідження **Pulse of the profession 2017**, то невизначені ризики проєкту потрапили на 8 сходинку списку причин провалу проєктів.

У більшості випадків ризик можна усунути або знизити його вплив на проєкт, додавши незначні зусилля або вклавши невеликі кошти.

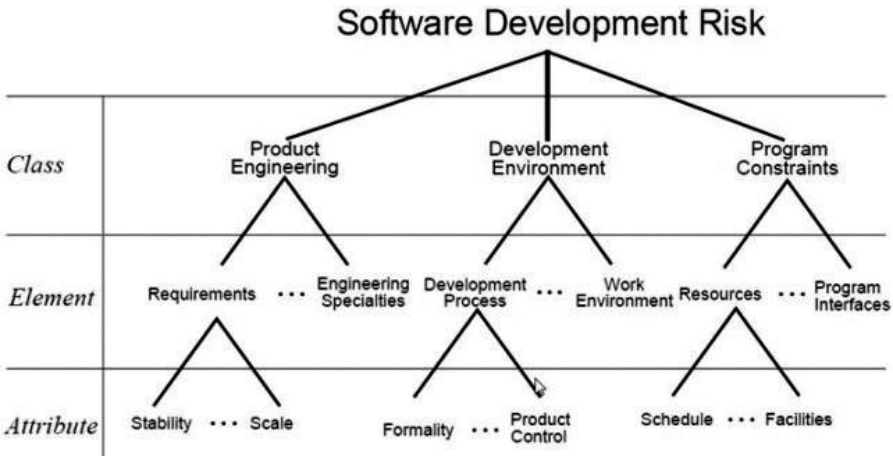
Якби вигоди від управління ризиками в проєктах були очевидними та доведеними, то це б набуло стандартної практики. Але є такі стереотипи, які відволікають керівників проєкту від управління ризиками*:*

1. Якщо займатися ідентифікацією ризиків в IT-проєкті, то ніколи буде працювати.
2. Довгий список ризиків відіб'є будь-яке бажання реалізовувати проєкт.
3. Погані думки притягують погані події. Краще не думати про погане.
4. Хто не ризикує, той не п'є шампанського.

На наш погляд, проєкт тільки виграє навіть від того, якщо ідентифікувати хоча б кілька десятків ризиків, і вжити за ними антиризикових заходів – це різко підвищить стійкість проєкту до зовнішніх потрясінь.

Типи ризиків

У 1993 році **SEI** (**Software Engineering Institute**) опублікував класифікатор ризиків ІТ-проєктів – **Taxonomy – Based Risk Identification**. Незважаючи на досить об'ємний зміст, цей документ достатньо простий у використанні. Він описує **три класи ризиків для софтверних проєктів**, кожен клас декомпозується на елементи та атрибути (рис. 16).



© http://resources.sei.cmu.edu/library/asset-view.cfm?assetid=11847

Рисунок 16

Для ідентифікації ризиків ІТ-проєкту можна вивчити всі елементи ризиків, описані в документі, і залишити лише ті з них, які на ваш погляд, актуальні для вашого ІТ-проєкту. Після чого можна заглибитися і вивчити атрибути і конкретні ризики, пов'язані з цими атрибутами. В результаті, у вас залишаться сухі тези – список актуальних для вашого проєкту ризиків, які слід внести в документ «Реєстр ризиків», правильно сформулювавши ризики.

Процес управління ризиками

Процес управління ризиками складається з наступних дій:

- ідентифікація ризиків (пошук ризиків та їх фіксація);
- якісний аналіз ризиків (оцінка важливості ризиків);
- планування антиризикових заходів (додавання до списку завдань проєкту нових завдань, пов'язаних зі зниженням, передачею або активним прийняттям ризиків);
- моніторинг та контроль ризиків.

Ідентифікація ризиків

Проводиться для того, щоб виявити якомога більше ризиків ІТ-проєкту і записати їх у документ «Реєстр ризиків».

Ідентифікація ризиків – це ітеративний процес, оскільки з розвитком проєкту у межах його життєвого циклу можуть виявлятися нові ризики. Частота ітерацій і склад учасників ітерації у кожному разі можуть бути різними.

Для керівника проєкту важливо навчитися робити правильні формулювання ризику.

В MSF пропонують наступний формат для формулювання ризиків (див. рис. 17).

Якісний аналіз ризиків

Список ризиків може бути дуже довгим, при цьому різні ризики можуть мати різну важливість для проєкту. Як оцінити важливість ризиків?

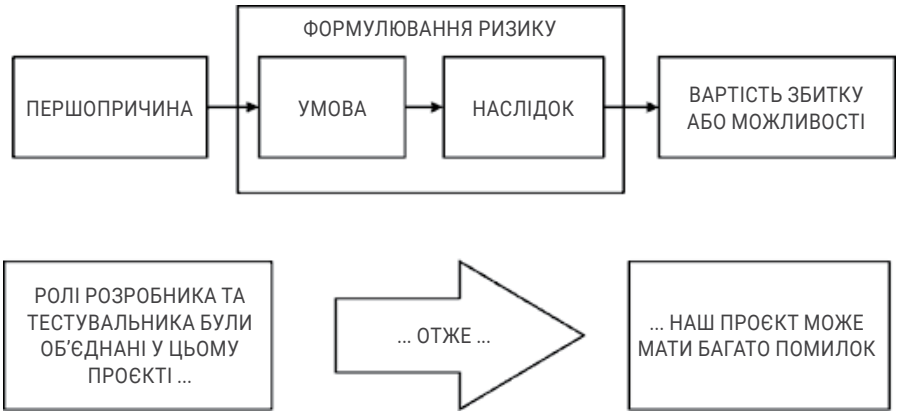


Рисунок 17

Для оцінки важливості ризиків використовується дві характеристики: ймовірність виникнення ризику і його вплив на проєкт.

Знаючи ці два параметри, можна прорахувати важливість ризику.

Як оцінити ймовірність виникнення ризику?

Можна використовувати статистику з випадків, які призводять до матеріалізації ризику або експертної думки.

Наприклад, ви знайшли декілька професіоналів, які готові допомогти вам допомогти з оцінкою ймовірності. Запропонуйте їм вибрати інтервал ймовірностей виникнення ризику:

Інтервал ймовірностей	Формулювання	Оцінка в числах
Від 0% до 33%	Низька	1
Від 34% до 67%	Середня	2
Від 68% до 100%	Висока	3

Для того щоб оцінити вплив ризику на проєкт, можна використовувати наступну матрицю впливу:

Формулювання	Оцінка в числах
Низьке	1
Середнє	2
Високе	3

А для розрахунку впливу на проєкт використовується формула:

Важливість ризику = Ймовірність × Вплив

Планування реагування на ризики

Для ризиків з найвищим значенням важливості, керівник проєкту повинен продумати такі заходи, які дозволять йому прибрати ризики з проєкту або різко знизити їх ймовірність або вплив на проєкт.

Для вироблення антиризикових заходів використовуються чотири можливі стратегії роботи з ризиками. Слід продумати конкретні варіанти реагування за кожною стратегією і вибрати ті з них, які коштують дешевше, ніж ймовірний збиток від виникнення ризику.

У PMBOK описують 4 стратегії роботи з ризиками:

1. **Ухилення.** Припускає вибір таких заходів, які дозволять повністю виключити виникнення проблеми, і цим позбавити проєкт від наслідків ризику.

2. **Передача.** Передбачає пошук можливості передати ризик разом з відповідальністю за реагування на ризик на третій бік (наприклад, до страхової компанії). Передача ризику означає, що у разі матеріалізації ризику третя сторона виплатить керівнику проєкту або замовнику матеріальну компенсацію, проте це не дозволяє бути впевненим, що ризик не відбудеться.
3. **Зниження.** Має на увазі продумування та планування заходів, які дозволять знизити ймовірність та/або наслідки від виникнення ризику до прийнятних меж.
4. **Прийняття.** Є два варіанти для стратегії прийняття – активне та пасивне прийняття. Активне прийняття передбачає створення резерву часу та грошей на усунення наслідків матеріалізації ризику. Пасивне прийняття передбачає наявність запасного плану на випадок матеріалізації ризику.

Контроль ризиків

Періодично керівнику проєкту рекомендується оцінювати статус ризиків: стався ризик чи ні, переглядати важливість ризиків та ідентифікувати нові ризики проєкту.

Для цього необхідно залучити учасників команди до процесу керування ризиками та переконати їх в тому, що робота з ризиками проєкту сильно підвищує ймовірність вкластися у строк та бюджет.

3. УПРАВЛІННЯ ЯКІСТЮ В ПРОЄКТІ

Що таке «управління якістю»?

В IT-індустрії використовуються три поняття, які іноді ототожнюються, але насправді є принципово відмінними: **контроль якості (QC)**, **забезпечення якості (QA)** і **тестування продукту**.

Тестування проводиться після того, як продукт або його частина була створена, а контроль якості – це набір активностей, який гарантує якість продукту на усіх етапах його створення. Контроль якості (QC) зосереджений на пошуках дефектів у програмному продукті і забезпеченні дій щодо їх виправлення. А тестування є невід'ємною частиною контролю якості.

Забезпечення якості програмного продукту (QA) – це набір дій, який відповідає за розробку та впровадження процесів для покращення всього життєвого циклу розробки продукту.

Співвідношення між трьома поняттями добре ілюструє рисунок 18 на сторінці 54.

До функціональних обов'язків QA-команди входять такі функції:

- підготовка планів забезпечення якості проєкту;
- перевірка відповідності процесів проєкту до планів якості;
- проведення регулярних перевірок продуктів проєкту;

QA, QC and Testing in software development process

Рисунок 18

- повідомлення вищестоящому керівництву про те, що є відхилення від стандартів;
- контроль наявності процедур керування змінами проєктів;
- контроль наявності процедур керування конфігураціями продуктів;
- проведення здобуття уроків у процесі контролю якості та втіленням рекомендацій, що ґрунтуються на засвоєних уроках.

У деяких організаціях функцію QA виконує Проєктний офіс компанії. Це відповідає критеріям незалежності, однак, організації, що дотримується цієї моделі, необхідно переконатися, що ця команда складається з навчених та/або спеціалізованих аналітиків із забезпечення якості.

Метрики

Контроль якості програмного продукту не можливий без наявності метрик, які дозволяють оцінити досягнення того чи іншого рівня якості програмного продукту.

> ***Метрика*** *– це кількісний масштаб та метод, який може використовуватись для вимірювання.*
>
> © ISO 14598

Один із способів групування метрик є групування за типами сутностей, що беруть участь у забезпеченні якості та тестуванні програмного забезпечення:

1. Метрики з тестових випадків (Test Cases).
2. Метрики з багів/дефектів.

Рассмотрим каждую из групп метрик.

Метрики з тест-кейсів

Назва метрики	Опис
Пройдені/провальні тест-кейси	Співвідношення кількості успішно пройдених тест-кейсів до провальних. По завершенню проєкту кількість провальних тестів має дорівнювати нулю.
Не запущені тест-кейси	Кількість тест-кейсів, які необхідно виконати для даного етапу проєкту. Маючи цю інформацію, ми можемо проаналізувати та виявити причини, з яких тести не були проведені.

Метрики з багів

Назва метрики	Опис
Відкриті/закриті баги	Співвідношення відкритих багів до закритих.

Назва метрики	Опис
Знову відкриті/ закриті баги	Співвідношення кількості багів, переведених зі стану знову «закритий» у стан «відкритий», до загальної кількості закритих.
Відхилені/відкриті баги	Співвідношення кількості відхилених багів до відкритих.

Практичне використання метрик

Для кожної метрики варто продумати планове значення та виміряти фактичне.

Найкраще вести графіки за метриками та відстежувати тренди. Наприклад, якщо частка знову відкритих багів до закритих зростає, необхідно провести аналіз причинно-наслідкових зв'язків і зрозуміти, що є корінними причинами знову відкритих багів і як ці причини усунути.

Або якщо частка відкритих до закритих багів зростає, це сигнал для того, щоб розібратися в причинах негативних трендів, які можуть бути наступними:

- неадекватна реалізація програмістами вимог (не розуміння вимог);
- неуважність розробників та погане тестування самим розробником перед передачею функціоналу на тестування.

План контролю якості

План контролю якості – це документ, який використовується, зокрема, у методології RUP. Даний документ містить план оцінки і контролю якості проєкту, і може

посилатися на низку інших артефактів, що використовуються в RUP.

Нижче представлені розділи, які містять шаблон Плану контролю якості, що використовується в RUP:

- визначення, акроніми та абревіатури;
- керування;
- документація;
- стандарти та рекомендації;
- метрики;
- план оцінки та контролю;
- оцінка та тестування;
- усунення несправностей та коригувальні дії;
- інструменти, методики та методології;
- управління конфігурацією;
- контроль за постачальниками та субпідрядниками;
- записи про якість;
- навчання;
- управління ризиками.

4. ДОКУМЕНТАЦІЯ ТА ДОКУМЕНТООБІГ

Цілі та завдання документації в рамках проєкту

Документування проєкту виконується для досягнення наступних цілей:

1. Зниження ймовірності того, що досягнуті домовленості між замовником проєкту та керівником проєкту будуть забуті або спотворені через деякий час після старту проєкту.
2. Підвищення ймовірності того, що всі зацікавлені сторони проєкту однаково розумітимуть зміст проєкту, плани реалізації завдань проєкту тощо.

Типи документації

Умовно всю документацію в IT-проєкті можна розділити на такі типи:

1. Документи про управління проєктом:
   - Project Charter (Статут проєкту);
   - WBS (ICP – ієрархічна структура робіт проекту);
   - розклад проєкту;
   - реєстр ризиків;
   - звіт про хід проєкту.

2. Документи про продукту проєкту:
   - Vision and Scope;
   - Software Requirements Specification (SRS);
   - Product Backlog;
   - користувацька документація.

Статут проєкту

Статут проєкту – документ, з якого починається проєкт і за яким відбувається оцінка успішності проєкту. Керівнику проєкту потрібен Статут проєкту для фіксації в ньому всіх важливих аспекти проєкту, узгоджені із замовником. Статут проєкту може стати додатком до договору і дозволяє всім учасникам проєкту отримати однакове уявлення про проєкт.

В своїх проєктах ми використовуємо шаблон Статуту проєкту, структура якого містить наступні розділи (рис. 19):

Рисунок 19

Ми вважаємо, що чорновий варіант Статуту проєкту в більшості випадків розроблятиме керівник проєкту, а погоджуватимуть його спонсор та керівник проєкту.

Статут проєкту не є юридично значущим документом, але може чудово доповнювати договір або слугувати внутрішнім документом, яким будуть користуватися команда проєкту і замовник проєкту у разі відхилення від спочатку узгоджених параметрів проєкту.

WBS (ICP – ієрархічна структура робіт проекту)

Для того щоб перейти від результатів проєкту до списку робіт, обмежити рамки проєкту та переконатися в тому, що важливі роботи проєкту не будуть упущені, в деяких методологіях керування проєктами (наприклад, PMBOK, MSF) пропонують використовувати документ **Work Breakdown Structure** (**WBS**, Ієрархічну структуру робіт). WBS дозволяє розділити проєкт на частини (пакети робіт) (рис. 20):

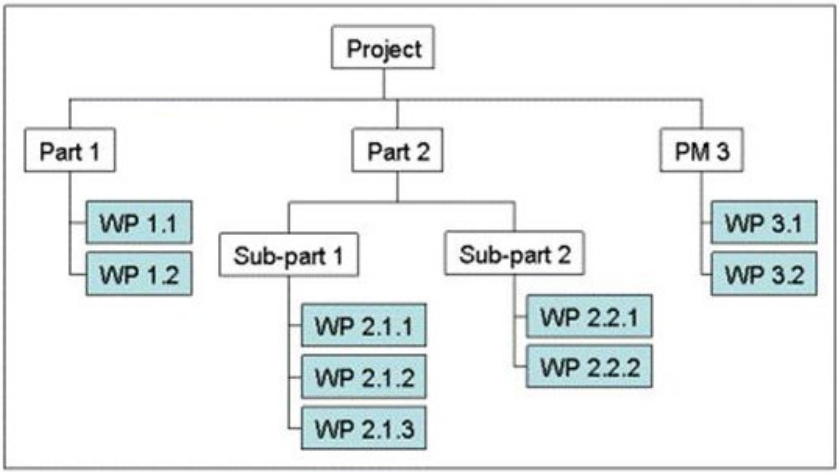


Рисунок 20

У MSF пишуть про те, що WBS пов'язує функціональні специфікації та рішення, які необхідно створити із завданнями, які мають бути для цього виконані.

Існує 3 підходи до декомпозиції проєкту:

1. **За результатами проєкту.**
2. **За функціями (спеціалізації праці).**
3. **За життєвим циклом.**

MSF дає наступну рекомендацію для вибору способу декомпозиції: «*Перший рівень структури WBS може відповідати фазам життєвого циклу проєкту*».

Ось один із варіантів розділення проєкту для впровадження CRM з використанням підходу за результатами проєкту:

ПРОЄКТ ВПРОВАДЖЕННЯ CRM			
Вимоги до програмного продукту	Програмний продукт CRM, який відповідає вимогам	Нормативні документи по процесах CRM	Користувачі, навчені роботі в CRM

Описані у WBS результати проєкту є необхідними для досягнення кінцевої мети проєкту – CRM у компанії має приносити ефект для бізнесу.

Розглянемо другий варіант декомпозиції проєкту – за етапами життєвого циклу:

ПРОЄКТ ВПРОВАДЖЕННЯ CRM			
Розробка вимог до програмного продукту	Вибір програмного продукту та постачальника	Доопрацювання програмного продукту під вимоги	Впровадження програмного продукту

Цей проєкт впровадження CRM можна декомпозувати за спеціалізацією праці:

ПРОЄКТ ВПРОВАДЖЕННЯ CRM				
Збір та аналіз вимог	Розробка архітектури та технічного проєкту	Доопрацювання програмного продукту	Впровадження програмного продукту	Керування проєктом

Очевидно, що три підходи, які використовуються для декомпозиції проєкту на верхньому рівні, на нижніх рівнях WBS мають привести приблизно до того ж списку завдань. При цьому, ніякі важливі завдання проєкту не мають бути втрачені.

При створенні WBS рекомендується дотримуватися наступних правил:

1. Після декомпозиції батьківського завдання потрібно переконатися, що досягнення результатів щодо дочірніх завдань призведе до досягнення результату за батьківським завданням.
2. Дочірні завдання, що декомпозують батьківське завдання, мають бути однаковими за термінами або обсягом робіт.
3. При переході від одного рівня WBS до іншого можна змінювати підхід до декомпозиції (наприклад, на другому рівні використовувався підхід «від результатів», але на третьому можна застосувати підхід «за функціями»).

4. Використовувати різні підходи до декомпозиції робіт слід таким чином, щоб максимальна частка зв'язків між завданнями потрапила на нижні рівні WBS.

Тепер відповімо на запитання щодо глибини декомпозиції WBS. По-перше, різні пакети робіт WBS можуть мати різну глибину деталізації.

По-друге, декомпозицію можна зупинити у разі, коли завдання нижнього рівня WBS відповідають наступним умовам:

- вимоги до завдання зрозумілі потенційному виконавцю та співробітникам, які беруть участь у декомпозиції робіт;
- на одне завдання можна призначити одного виконавця;
- обсяг робіт із завдання не перевищує деякого порогового значення.

Наприклад, якщо трудомісткість завдання не перевищує 40 людино-годин, декомпозицію варто зупинити.

У MSF стосовно розміру WBS-завдань є наступна рекомендація для IT-проєктів: *«Оцінка часу виконання кожного завдання не має бути менше одного або більше 40 днів»*.

Розклад проєкту

Розклад проєкту потрібен замовнику та керівнику проєкту для того, щоб якомога раніше виявити відхилення від затвердженого базового розкладу та приймати важливі рішення щодо коригуючих дій.

Грамотний розклад проєкту дозволяє:

1. Зрозуміти, яка є потреба в виконавцях на будь-який період часу, наприклад, наступного місяця, тижня або дня.
2. Вносити до нього розклад зміни в обсязі доступного часу виконавців, виділених на завдання проєкту, зміни в датах старту завдань, зміни в списку завдань.
3. Отримати прогноз того, чи встигає команда реалізувати проєкт у плановий термін.

Керівник проєкту для створення грамотного розкладу проєкту використовує такі підходи:

1. Розробка WBS-проєкту для отримання повного списку робіт з проєкту.
2. Оцінка обсягів робіт та призначення виконавців на завдання проєкту для визначення термінів виконання завдань.
3. Метод критичного шляху для визначення загальної тривалості проєкту.

Розробка розкладу проєкту, як правило, виконується в декілька ітерацій, а для його швидкого редагування потрібно створити модель проєкту.

Модель проєкту – список робіт з проєкту з певними зв'язками у послідовності виконання завдань, з оцінкою трудовитрат за завданнями, призначеними ресурсами та оцінками тривалості за кожним завданням.

Модель проєкту дозволяє керівнику проєкту визначити потребу у виконавцях на будь-який варіант планування проекту, без особливих зусиль вносити зміни і отримувати прогнозовану дату завершення проєкту, відтворювати

сценарії на кшталт: «Що буде з терміном проєкту, якщо в команду проєкту взяти ще одного програміста?», прогнозувати терміни завершення проєкту.

Для створення та роботи з моделлю проєкту існує спеціальне програмне забезпечення, таке як MS Project, Oracle Primavera, Easy Projects, Clarizen і т. д.

Зазвичай використовуються наступні види розкладу проєкту:

1. Мережева діаграма проєкту.
2. Діаграма Ганта.
3. План по віхах.
4. Календарний план проєкту.

Нижче наведена одна з форм Календарного плану проєкту.

Календарний план проєкту

_____

назва проєкту

Планова дата початку проєкту:
Планова дата завершення проєкту:

№ п/п	Найменування завдання	Дата старту завдання	Дата фінішу завдання	Виконавець	Критерії приймання результату

Реєстр ризиків

Реєстр ризиків не має єдину загальновизнану структуру. В Інтернеті можна знайти різні шаблони для цього документа. Ми наведемо один з них.

Приклад Реєстру ризиків проєкту:

Опис ризику	Завдання щодо реагування на ризик	Виконавець	Термін	Статус ризику
Недостатньо опрацьовані вимоги призведуть до великого обсягу доробок та зриву термінів	Відібрати лише критичні для запуску системи історії та сконцентруватися на їх реалізації	Якубович	21.04.22	Актуальний

Звіт про хід проєкту

Керівник проєкту повинен інформувати Замовника про перебіг проєкту. Як зробити звіт про хід проєкту у такій формі, щоб замовник витрачав лише 10-15 хвилин на отримання інформації про проєкт?

Є проста форма звіту, яка складається з 3-х розділів:

1. Отримані результати проєкту:

Опис результату	Статус результату	Фактичні витрати

2. Завдання в реалізації:

Завдання	%	Прогноз витрат

3. Проблеми (ризики):

Опис проблеми (ризику)	План реагування

В деяких ІТ-проєктах замовнику пропонують використовувати форму звіту на одному аркуші, яка містить до 7-10 зон уваги.

Приклад звіту розміщено на наступній сторінці (див. рис. 21 на стор. 68).

Розділ «Цілі проєкту». Тут потрібно сформулювати одну або декілька цілей проєкту.

Розділ «Завдання проєкту». Сюди треба внести 20-25 завдань з розкладу проєкту. Якщо завдань в проєкті більше, потрібно вносити завдання другого або третього рівня WBS.

На перетині стовпця «Цілі» та рядків із завданнями проєкту потрібно вказати, які завдання призведуть до досягнення даної мети проєкту. Коли всі кружечки в стовпці з назвою «Цілі» будуть зафарбовані кольором – мета вважається досягнутою.

	Керівник проєкту: М. Якубович	Проєкт: Впровадження ERP	Дата: 20.08.2021
	Мета проєкту: Зменшити трудовитрат на рутинні операції, підвищити точність обліку		

Цілі					Основні завдання	Завершити проєкт до: 20.12.2021										Відповідальний/Роль					
				1	Провести навчання користувачів у Відні											В	Вик.		К		
			В	2	Провести навчання користувачів у Берліні					В						В			Вик.	К	
			В	3	Провести навчання користувачів у Варшаві					В						В		Вик.			К
			В	4	Провести навчання користувачів у Києві						В					В		Вик.		К	
			В	5	Провести навчання користувачів у Празі					В						В	Вик.		К		
				6	Реалізувати відсутні звіти												В			Вик.	К
	В			7	Отримати пыдтвердження правильносты даних у звітах					В							В	Вик.		К	
В				8	Покращити інтерфейс програми					В							В			К	Вик.
В				9	Провести хронометраж для основних процесів					В							В		К		Вик.
В				10	Проаналізувати висвободження часу за рахунок						В						В		К		Вик.
В				11	Розрахувати ROI впровадження						В					К		В	Вик.		
В				12	Запустіть блок документообігу					В	В					Вик.	В				К
		В		13	Розробити процедури та інструкції для нового ІС					В								В	Вик.	К	
В				14	Протестувати рольову модель					В							В	Вик.	К		
В				15	Доопрацювати рольову модель					В	В						Вик.		В		К
В				16	Створити опис архітектури модуля «Виробництво»					В	В						В		К		Вик.
В				17	Створити опис архітектури модуля «Фінанси»					В	В						Вик.		В		К
				18	Провести закриття періоду в новому ІС					В	В					К		Вик.	Вик.	Вик.	В
В				19	Зібрати зауваження за підсумками закриття періоду					В	В						Вик.	В			К
В				20	Доопрацювати процедуру закриття періоду					В	В					В	К		Вик.		
		В		21	Доопрацювати процедури та інструкції для нового ІС											В	К		Вик.		
		В		22	Затвердити процедури та інструкції для нового ІС					В						В		Вик.			К
			В	23	Провести навчання співробітників усіх офісів						В						В		К		К

Впровадити модулі ERP	Отримайте точний звіт	Затвердити нові процедури	Навчити користувачів	Основні завдання / Цілі / Терміни / Витрати / Висновки та прогнози	Липень	Серпень	Вересень	Жовтень	Листопад	Грудень	Січень	Лютий	Березень	Якубович	Ніколаєв	Карпін	Шалімов	Сусько	Патрікєєв
					Витрати на аналітика									17000			25000		
					Преміальний фонд									0		20000			
					Витрати на доопрацювання									33000					50000
					Для контролю розробки процедур впровадити щатижневі наради														
					У Варшаві та Празі не вистачає бюджету на навчання, створено запит на Бюджетний комітет														
					Для прискорення роботи покращення інтерфейсу запрошена аутсорсингова команда														

Рисунок 21

Розділ «Терміни». Цей розділ дозволяє візуалізувати на діаграмі періоди, в яких заплановані роботи із завданням. Якщо завдання планується виконувати у декількох періодах, то навпроти цих періодів малюються кружечки.

Розділ «Відповідальний/Роль». Внизу вказуються прізвища ключових учасників проєкту, а вище прізвищ заповнюється матриця відповідальності. В даній формі використовуються такі значення:

- Відповідальний (**В**) — з нього запитають отримання результату за завданням;
- Виконавець (**Вик.**) — безпосередньо той, хто виконує завдання;
- Контролер (**К**) — приймає результати завдання та формулює список помилок.

Розділ «Витрати». Вносяться планові статті бюджету проєкту і плановий розмір витрат за ними. Чим більший плановий бюджет, тим довше відрізок.

Розділ «Висновки та прогнози». Керівнику проєкту потрібно переконати замовника, що він знає, як скерувати проєкт.

Як заповнюється звіт у частині виконання завдань?

Якщо обсяг фактично виконаної роботи був не менший запланованого, то в звіті, в рядку для завдання, кружечок замальовується зеленим кольором, якщо фактичний обсяг виявився менший запланованого, але є шанс надолужити розклад – використовується жовтий колір, а якщо в зазначені терміни по завданню нічого не виконувалося – використовується червоний колір.

Та сама система «світлофорів» використовується і для зафарбовування відрізків для статей бюджету. Якщо бюджет йде за планом – зелений колір, у звітному періоді були перевитрати – використовується жовтий, плановий

розмір за статтею бюджету перевищено – замальовуємо червоним.

Така форма звіту є дуже зручною і подобається замовникам проєктів.

Product Backlog

Даний документ використовується у підході Scrum для збору та управління послідовністю всіх вимог до програмного продукту. Відповідальним за те, які вимоги потраплять у Product Backlog та за наявність пріоритетів вимог зазвичай є Product Owner (*докладніше про роль PO читайте в уроці про Scrum*).

Структура Product Backlog не є суворою і не описана в Scrum Guide. Нижче наведено один із варіантів структури даного документа:

Вимога	Важливість	Сценарій тестування	Оцінка

Вимога – опис Epic або User story (*див. урок про Scrum*).

Важливість – це значення заповнює Product owner, він визначає шкалу, за якою вимірюються важливість вимог. В літературі зустрічається рекомендація щодо довжини списку вимог, яка є адекватною для ефективної роботи зі списком – близько 100 штук. Виходячи з цього, PO може вибрати шкалу від 100 (максимальна важливість) до 1 (мінімальна важливість).

Сценарій тестування – у цьому полі описується сценарій, при відтворенні якого вимога працюватиме так, як очікує Product owner.

Оцінка – в цей стовпець вносяться значення оцінки складності реалізації вимог. Оцінки робить, як правило, один із учасників

команди проєкту і вони мають попередній характер. Це означає, що вони можуть уточнюватися під час планування спринту.

Vision and Scope (Документ про образ та межі проєкту)

Цей документ дозволяє зрозуміти відповіді на питання:

- Які проблеми або можливості вирішує продукт?
- Які ключові бізнес-вимоги до продукту?
- Хто користувачі продукту?
- Які бізнес-вимоги потраплять до якого релізу продукту?

Структуру цього документа можна запозичити з книги **К. Вігерса «Розробка вимог до програмного забезпечення»**:

1. Бізнес-вимоги
1.1. Початкові дані, можливості бізнесу, потреби клієнтів
1.2. Бізнес-цілі та критерії успіху
1.3. Фактори бізнес-ризиків
2. Образ рішення
2.1. Положення про образ проєкту
2.2. Основні функції (характеристики)
2.3. Припущення та залежності
3. Масштаби та обмеження
3.1. Обсяг першого та наступних випусків
3.2. Обмеження та винятки

4. Бізнес-контекст
4.1. Профілі зацікавлених у проєкті осіб (учасників)
4.2. Пріоритети проєкту

Зауважимо, що деякі розділи цього документа перетинаються із документом Статут проєкту, тому керівнику проєкту варто адаптувати один із документів, щоб уникнути дублювання інформації.

Software Requirements Specification SRS

Цей документ має містити всі вимоги до програмного продукту. Класифікація вимог до програмного продукту буде розглядатися у наступному уроці.

Не існує єдиної версії структури документа SRS, але є 2 шаблони, які використовуються досить часто.

Структуру цього документа можна запозичити з книги **К. Вігерса «Розробка вимог до програмного забезпечення»**:

1. Вступ
1.1. Призначення
1.2. Обсяг проєкту та функції продукту
1.3. Посилання
2. Загальний опис
2.1. Загальний погляд на продукт
2.2. Класи та характеристики користувачів
2.3. Операційне середовище
2.4. Обмеження дизайну та реалізації
2.5. Документація для користувачів
2.6. Припущення та залежності

3. Функції системи
3.1. Функція 1
3.2. Функція 2
3.3. Функція 3
3.4. …..
3.5. Функція N
4. Вимоги до зовнішнього інтерфейсу
4.1. Інтерфейси користувачів
4.2. Інтерфейси обладнання
4.3. Програмні інтерфейси
4.4. Інтерфейси передачі даних
5. Інші нефункціональні вимоги
5.1. Вимоги до продуктивності
5.2. Вимоги до охорони праці
5.3. Вимоги до безпеки
5.4. Атрибути якості ПЗ
Додаток А: Словник та модель даних
Додаток В: Моделі аналізу

Другий шаблон SRS був запропонований у колись популярному для IT-проєктів підході – RUP і має наступну структуру:

1. Introduction
1.1. Purpose
1.2. Scope
1.3. Definitions, Acronyms, and Abbreviations
1.4. References
1.5. Overview

2. Overall Description
3. Specific Requirements
3.1. Functionality
3.1.1. <Functional Requirement One>
3.2. Usability
3.2.1. <Usability Requirement One>
3.3. Reliability
3.3.1. <Reliability Requirement One>
3.4. Performance
3.4.1. <Performance Requirement One>
3.5. Supportability
3.5.1. <Supportability Requirement One>
3.6. Design Constraints
3.6.1. <Design Constraint One>
3.7. On-line User Documentation and Help System Requirements
3.8. Purchased Components
3.9. Interfaces
3.9.1. User Interfaces
3.9.2. Hardware Interfaces
3.9.3. Software Interfaces
3.9.4. Communications Interfaces
3.10. Licensing Requirements
3.11. Legal, Copyright, and Other Notices
3.12. Applicable Standards
4. Supporting Information

А ось структура SRS, рекомендована стандартом IEEE 830:

- Вступ
  - Цілі
  - Угоди про терміни
  - Передбачувана аудиторія та послідовність сприйняття
  - Масштаб проєкту
  - Посилання на джерела
- Загальний опис
  - Бачення продукту
  - Функціональність продукту
  - Класи та характеристики користувачів
  - Середовище функціонування продукту (операційне середовище)
  - Межі, обмеження, правила та стандарти
  - Документація для користувачів
  - Припущення та залежності
- Функціональність системи
  - Функціональний блок X (таких блоків може бути декілька)
    - Опис та пріоритет
    - Причинно-наслідкові зв'язки, алгоритми (хід процесів, workflows)
    - Функціональні вимоги
- Вимоги до зовнішніх інтерфейсів
  - Інтерфейси користувача (UX)

- Програмні інтерфейси
- Інтерфейси обладнання
- Інтерфейси зв’язку та комунікації

- Нефункціональні вимоги
  - Вимоги до продуктивності
  - Вимоги до безпеки (даних)
  - Критерії якості програмного забезпечення
  - Вимоги до безпеки системи

Користувацька документація

У більшості випадків, для користувачів програмного продукту необхідно мати наступні види документації для роботи з програмою:

- вбудована довідка про роботу з програмою;
- матеріали з описом кейсів з використання продукту (найчастіше їх створюють у форматі відеоуроків, але зустрічаються ще й описи у вигляді документів).

Урок 2
ДОКЛАДНІШЕ ПРО КЕРУВАННЯ ПРОЄКТОМ