ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«МЕТІНВЕСТ ПОЛІТЕХНІКА»

«ПОГЛИБЛЕНИЙ КУРС БІЗНЕС-АНАЛІЗУ ІТ ПРОЄКТІВ»

КУРС ЛЕКЦІЙ

з дисципліни за освітньо-професійною програмою
другого (магістерського) рівня
«Комп'ютерні науки та цифровий інтелект»
(спеціальність 122 Комп'ютерні науки)

Рекомендовано Науково-методичною радою ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА»
(протокол № 6 від «4» липня 2023 р.)
Обов'язково до розміщення в репозитарії

Запоріжжя 2023

Поглиблений курс бізнес-аналізу ІТ проєктів: курс лекцій з дисципліни за освітньо-професійною програмою другого (магістерського) рівня «Комп'ютерні науки та цифровий інтелект» (спеціальність 122 Комп'ютерні науки) / Уклад. Шевченко Н.Ю. Запоріжжя, ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА», 2023. 160 с.

Конспекті лекцій містить матеріал для поглибленого вивчення бізнес-аналізу ІТ проєктів: основні поняття бізнес-аналізу, роль бізнес-аналізу в управлінні ІТ проєктами, особливості планування та виконання ІТ проєктів, опис життєвого циклу ІТ проєкту; підходи до проведення збору ідей, виконання аналізу зацікавлених сторін, визначення бізнес-потреб, цілей та формулювання бачення продукту, його результатів; правила документування бізнес та функціональних вимог, проведення валідації вимог, пріоритезація вимог; підходи до побудови дизайну програмного забезпечення як методу опису/візуалізації вимог та інш.

Самостійне електронне текстове мережеве видання

Затверджено на засіданні кафедри цифрових технологій та проєктно-аналітичних рішень
Протокол № 1 від «30» травня 2023 р.

Узгоджено:
Секретар Редакційної ради

_____ Малій Х. В.
«1» червня 2023 р.

ЗМІСТ

ВСТУП

Цифровізація суспільства ставить виклики не тільки до якості програмного забезпечення, але й до наявності кадрів та професійного рівня спеціалістів з аналізу та управління проєктами в сфері інформаційних технологій. Від вибору чи побудови якісної моделі управління ІТ проєктом залежить успіх його реалізації: від своєчасного виконання замовлення з заданою якістю кінцевого продукту в межах виділеного бюджету до надмаксимального задоволення вимог замовника. А цінність кінцевого продукту для замовника безпосередньо залежить від якості проведення бізнес-аналізу на всіх стадіях життєвого циклу проєкту. Рівень розвитку ІТ-технологій, конкуренція та обізнаність у цій сфері замовників вимагає сучасних підходів до розробки вимог до програмного забезпечення, які базуються на використанні класичних підходів, їх комбінуванні та вдосконаленні.

Дисципліна «Поглиблений курс бізнес-аналізу ІТ проєктів» спрямована на формування базових компетентностей фахівців з інформаційних технологій щодо ефективного застосування методів бізнес-аналізу при управлінні ІТ проєктами.

Протягом курсу у здобувачів освіти будуть сформовані практичні навички щодо основних підходів та засад бізнес-аналізу ІТ проєктів в умовах цифрової трансформації економіки України, навички використання практичних інструментів управління ІТ проєктами в залежності від ролі в ІТ команді, навички ведення документів (документ концепції та меж, документ користувацьких вимог, специфікації вимог до програмного забезпечення), виявлення вимог до програмного забезпечення, управління ризиками.

Матеріал, викладений в конспекті лекцій, дозволить зрозуміти основні поняття бізнес-аналізу, роль бізнес-аналізу в управлінні ІТ проєктами, знати особливості планування та виконання ІТ проєктів, визначати життєвий цикл ІТ проєкту; надасть уявлення як проводити збори ідей, виконувати аналіз зацікавлених сторін, визначати бізнес потреби, цілі та бачення продукту, його результати; як документувати бізнес та функціональні вимоги, проводити валідацію вимог, створювати план дій для розробки, виконувати пріоритезацію вимог; допоможе визначати (будувати) варіанти дизайну програмного забезпечення і знаходити можливості для покращення.

1 БІЗНЕС-АНАЛІЗ ІТ ПРОЄКТІВ. КОНСПЕКТ

ТЕМА 1. БІЗНЕС-АНАЛІЗ В ІТ: АКТУАЛЬНІСТЬ, ТЕНДЕНЦІЇ, БАЗОВІ ПОНЯТТЯ БІЗНЕС-АНАЛІТИКИ

Тенденції розвитку ІТ сфери в Україні. Теоретико-методичні підходи до проведення бізнес-аналізу ІТ-проєктів. Бізнес-аналіз як важлива складова управління проєктів у сфері ІТ. Методичні засади проведення бізнес-аналізу з акцентом на аналіз вимог щодо проєкту. Використання методологій моделювання бізнесу та бізнес-процесів (Business Model Canvas, IDEF, UML) у процесі бізнес-аналізу на прикладі ІТ-проєкту. Аналітика як наука поглибленого аналізу. Економічна аналітика. Сутність та зміст бізнес-аналітики. Сутність та види бізнес-процесів. Об'єкт та предмет бізнес-аналітики.

Передумова актуальності та затребуваності бізнес-аналізу в ІТ – розвиток галузі[1]

На сучасному етапі розвитку цивілізації відбувається (відбувся) поступовий перехід від індустріального до постіндустріального або інформаційного суспільства. Одне з його визначень таке: **інформаційне суспільство** – історична фаза можливого розвитку цивілізації, у якій **основними продуктами виробництва є інформація й знання**.

Головні риси такого суспільства:

– збільшення ролі інформації та знань;

– зростання числа людей, зайнятих інформаційними комунікаціями та виробництвом інформаційних продуктів і послуг;

– створення глобального інформаційного простору, що забезпечує ефективну інформаційну взаємодію людей, їхній доступ до світових інформаційних ресурсів і задоволення їхніх потреб в інформаційних продуктах і послугах.

Інформатизація й комп'ютеризація вимагають від людей нових навичок, нових знань і нового мислення, покликаних полегшити адаптацію людини до умов і реалій комп'ютеризованого суспільства й забезпечити йому гідне місце в цьому суспільстві.

[1] Звіт за 2021 р. Костянтин Васюк, виконавчий директор Асоціації IT Ukraine. https://drive.google.com/file/d/1LujaT9pHEGhgpRRojfnlZgQikkyiIlbE/view

В Україні на законодавчому рівні затверджена **Національна програма інформатизації**[2]. У відповідному законі визначені основні поняття, функції суб'єктів інформатизації, механізми розвитку тощо.

Загальні тенденції

Українська ІТ-галузь за останні 25 років зробила шалений ривок уперед. Стартувавши практично з нуля, вона перетворилася на високоінтелектуальну індустрію, де працює майже 300 тисяч фахівців і яка щороку зростає на 25-30%.

В 2020 р. вона вийшла на перше місце за обсягом експорту послуг (понад 5 млрд дол. на рік) і принесла Україні понад 4% ВВП. Україна є одним із найбільших експортерів ІТ-послуг в Європі.

ІТ-галузь вже стала модною: згідно із опитуваннями, проведеними в 2021 р. 54% українських старшокласників хочуть працювати в ІТ. В 2021 р. на спеціальності «комп'ютерні науки» та «інженерія ПЗ» в українських університетах подали заявки понад 80 тисяч майбутніх студентів. А в 2022 р. – понад 100 тис. заяв абітурієнтів.

Водночас компанії продовжують інвестувати в освіту та підготовку кадрів, з'являються нові сучасні освітні платформи.

ІТ-сфера в Україні є однією з найбільших бюджетоутворювальних галузей економіки і здійснює значний вплив на різні напрями економіки і суспільства. ІТ-галузь здійснює свою діяльність з використанням проєктного підходу.

Проєкт – це тимчасова робота, спрямована на створення унікального продукту, послуги чи результату.

А ІТ-проєкт розглядається як набір взаємно пов'язаних ресурсів, що забезпечує випуск одного чи декількох ІТ-продуктів (інформаційної системи, програмного забезпечення тощо) для клієнта чи кінцевого користувача.

Але багато ІТ-проєктів не дають бажаних результатів або приносять незначний позитивний ефект через неправильне розуміння вимог, які замовники або інші зацікавлені сторони неспроможні чітко сформулювати на початковій стадії проєкту.

[2] https://zakon.rada.gov.ua/laws/show/74/98-%D0%B2%D1%80#Text

PMI (Project Management Institute) 2017 року проводив глобальне дослідження, у якому взяли участь понад 3 234 фахівці з управління проєктами. Одне із запитань дослідження — «Які основні причини неуспіху проєктів, розпочатих у вашій організації за останні 12 місяців? (Вкажіть до 3-х причин)». Перші п'ять причин, які набрали найбільшу кількість голосів, безпосередньо пов'язані з бізнес-аналізом: 1) зміни у пріоритетах організації (41 %); 2) помилки на етапі збору вимог (39 %); 3) зміна цілей проєкту (36 %); 4) неадекватне бачення цілі проєкту (30%); 5) слабка комунікація у проєкті (30 %). Тому саме розвиток бізнес-аналітичних компетенцій здатний збільшити кількість успішних проєктів.

Зазвичай, до уваги беруться два фактори якості роботи бізнес-аналітика: продуктивність та ефективність. Загалом, **ефективність** визначається, як «робити те саме, із меншими витратами», а **продуктивність** — як «робити більше, з тими самими витратами».

Дуже багато IT-компаній беруть за мету досягнення ефективності. Саме тому вони завжди прагнуть скоротити певні ресурси. Основною метою бізнес-аналітиків, з операційної позиції, є саме продуктивність, що дозволяє впорядковувати операції та поліпшувати результати, а отже, забезпечувати успіх проєкту.

Деякі дослідники вважають, що поняття «бізнес-аналітика» об'єднує дві різні сутності: «бізнес-аналіз» (BA) і «бізнес-інтелідженс» (BI). Але існує інша думка: «бізнес-аналіз» є окремою від «бізнес-аналітики» сферою діяльності. BI займається «сирою інформацію», а BA, у свою чергу, користується вихідними даними, підготовленими для подальшого аналізу для ухвалення ефективних управлінських рішень. Незважаючи на висвітлені відмінності між бізнес-аналізом і бізнес-аналітикою, вони можуть використовуватися консолідовано і допомагати одне одному в пошуку рішень бізнес-проблем.

Бізнес-аналітика – це «наука аналізу» (поглибленого аналізу). Вона передбачає застосування під час аналізу явищ та процесів великих баз даних (big date), широкого кола аналітичних показників, статистичних і математичних методів, комп'ютерних технологій і програмного забезпечення для прийняття управлінських рішень.

Аналітика як наука поглибленого аналізу використовується в усіх сферах життя, вона не тільки констатує, яка ситуація склалася, а й

обов’язково доводить взаємозв’язок між явищами та процесами, чому склалася така ситуація, дає підстави для обробки великих масивів даних, дає можливість виконувати моделювання та прогнозування досліджуваних явищ і на цій основі розробляти управлінські рішення.

Економічна аналітика – це наука аналізу економічних процесів, які відбуваються в усіх видах діяльності й життя суспільства, на різних рівнях управління: національному, регіональному, на рівні підприємств усіх форм власності (державна, приватна, колективна).

Отже, **бізнес-аналітика** – це наука аналізу, яка вивчає кількісну та якісну сторону бізнес-процесів у різних видах діяльності, застосовуючи репродуктивну, розрахункову й аналітичну функції показників, статистичні та математичні методи аналізу, моделювання й прогнозування, інформаційні технології та програмне забезпечення для обґрунтування ефективних управлінських рішень.

Об’єктом бізнес-аналітики є явища та процеси, пов’язані з підприємницькою діяльністю та її бізнес-процесами.

Предмет бізнес-аналітики – кількісна оцінка та аналіз бізнес-процесів для розробки ефективних управлінських рішень на основі застосування статистичних і математичних методів, інформаційних та телекомунікаційних технологій.

Бізнес-процес – це сукупність логічних, послідовних і взаємопов’язаних заходів або завдань, спрямованих на створення продукту (послуги) для споживачів. **Бізнес-процес** визначають також як будь-яку діяльність, що має вхідний продукт, додає цінність до нього та забезпечує вихідний продукт для внутрішнього та зовнішнього споживача.

Бізнес-аналіз (англ. Business analysis, BA):

діяльність, яка уможливлює проведення змін в організації, які приносять користь зацікавленим сторонам, шляхом виявлення потреб та обґрунтування рішень, що описують можливі шляхи реалізації змін (International Institute of Business Analysis);

набір методів, які допомагають зрозуміти структуру, особливості компанії клієнта (проєкта), визначити потреби і запропонувати варіанти рішення задачі;

це набір технік, інструментів і завдань, необхідних для визначення потреб бізнесу, знаходження і розробки шляхів задоволення цих потреб.

Відомий консультант із питань управління Т. Брайс виступав за збільшення часу, проведеного саме на ранніх етапах IT-проєкту, для кращої чіткості визначення вимог і створення кращих специфікацій для розробників. За цим сценарієм, він пропонує виділяти до 60 % на ранні етапи, які включають системний аналіз і проєктування, і лише 15 % на програмування проти загальноприйнятого рівня у 85, та 25 % на тестування і підтримку. Значна увага до аналізу також пояснюється тим, який приріст продуктивності він здатний забезпечити на проєкті [7].

Бізнес-аналітик – посередник між зацікавленими особами для збору, аналізу, комунікації та перевірки вимог щодо зміни бізнес-процесів, регламентів і інформаційних систем (International Institute of Business Analysis).

Бізнес-аналітик розуміє проблеми та можливості бізнесу в контексті вимог і рекомендує рішення, що дозволяють організації досягти своїх цілей.

Для отримання продуктивного результату, який призведе до гарантованого успіху в проєкті, бізнес-аналітики використовують процес, який складається з поетапного аналізу бізнесу. Узагальнено етапи процесу бізнес-аналізу такі: 1) збір потрібної інформації; 2) визначення зацікавлених сторін; 3) окреслення цілей бізнесу; 4) оцінка варіантів; 5) визначення обсягу роботи; 6) розроблення плану з отримання рішення; 7) визначення вимог до проєкту; 8) підтримка реалізації проєкту; 9) оцінка отриманої цінності від продукту.

Чи відрізняються підходи до бізнес-аналізу в залежності від сфери?[3]

Бізнес-аналіз (Business Analysis) – це процес розуміння бізнес-потреб, визначення проблем та можливостей, аналізування варіантів розв'язання та встановлення вимог до розробки програмного забезпечення або інших бізнес-рішень.

Бізнес-аналітик (Business Analyst) – це професійний фахівець, який займається бізнес-аналізом та забезпечує зв'язок між замовниками та розробниками програмного забезпечення або іншими стейкхолдерами в процесі вирішення бізнес-проблем. Бізнес-аналітик допомагає розробникам зрозуміти вимоги до продукту, а також визначає, як цей

[3] Розробка вимог до програмного забезпечення / Карл Вігерс, Джой Бітті

продукт може вирішити потреби клієнтів та сприяти досягненню бізнес-цілей.

Якщо розкласти дії бізнес-аналітика за етапами реалізації проєкту, то матимемо наступний комплекс обов'язків:

Ініціалізації та планування. Збір ідей, визначення бізнес потреб, цілей та бачення продукту, оцінювання витрат робіт, аналіз здійсненності й оцінювання ризиків.

Виявлення, збір та аналіз вимог. Документація бізнес та функціональних вимог. Валідація вимог. Створення плану дій для розробки. Визначення скоупу (scope) та пріоритезація. Комунікація вимог в команді.

Дизайн. Прототипи, варіанти дизайну, можливості для покращення.

Розробка продукту. Демо за результатами спринтів, збір фідбеків від стейкхолдерів.

Тестування. Аналіз результатів.

Деплоймент та підтримка ПЗ. Перевірка готовності, підготовка документації, пост-релізна комунікація.

Вивчення бізнес-процесів та виявлення проблем та можливостей в компанії.

Встановлення вимог до програмного забезпечення або інших рішень згідно з потребами бізнесу та користувачів.

Створення бізнес-моделей та аналіз їх ефективності.

Забезпечення зв'язку між стейкхолдерами та розробниками, включаючи збір та передачу вимог та комунікацію стосовно результатів проєкту.

Аналіз ринку та конкурентів, відстеження тенденцій та рекомендації щодо стратегії компанії.

У **BABOK® Guide** типи вимог до IT-продукту визначені таким чином:

1) бізнес-вимоги (твердження абстрактного рівня, що описують цілі та завдання бізнесу на рівні підприємства);

2) вимоги зацікавлених сторін, які визначають потреби певної групи стейкхолдерів, і те, що вони вимагають від конкретного рішення;

3) вимоги до рішення описують, які характеристики повинно мати рішення для того, щоб відповідати потребам зацікавлених сторін і бізнесу;

4) вимоги щодо переходу описують можливості, які має мати рішення, та умови, яким повинно відповідати рішення, щоб полегшити перехід від поточного стану до майбутнього, але які не потрібні після завершення цих змін. Вони відрізняються від інших типів вимог тим, що мають тимчасовий характер.

Концептуально аналіз вимог включає чотири види діяльності:

— визначення вимог: процес спілкування із замовниками і користувачами для визначення їхніх очікувань від IT-продукту;

— аналіз вимог: процес визначення того, чи є отримані вимоги нечіткими, неповними, неоднозначними чи суперечливими, а також вирішення цих недоліків;

— моделювання вимог: вимоги можуть бути задокументовані в різних формах, таких як звичайні документи, випадки використання (use cases), історії користувачів (user stories) або технічні характеристики;

— огляд і ретроспектива: члени команди розмірковують про те, що сталося за час реалізації, та визначають заходи щодо вдосконалення на майбутнє.

Техніки, з якими працює бізнес-аналітик:

- Аналіз стейкхолдерів, RACI matrix.
- Brainstorm.
- Business Model Canvas.
- Аналіз процесів.
- Аналіз прийняття рішень.
- Моделювання.
- Сценарії та варіанти користування.
- Аналіз потоків даних.
- Функціональна декомпозиція.
- Спостерігання.

Технічні навички бізнес-аналітика в IT:

- **Software Development Methods**
- Agile, Waterfall, DevOps
- **Web/Mobile/Desktop Applications**
- **Research Skills**
- **Statistics and Probability**
- Перестановки і комбінації. Розподіл ймовірностей. Теорема Байєса. Регресійний аналіз. Формування вибірки. Перевірка гіпотези. Тестування Т-розподілу. Дисперсійний аналіз (ANOVA).
- **Documentation and Presentation**
- BRD (Business Requirement Document). SRS (Software Requirement Specification). Use Cas. User Story. User Guide /Manual. Reports.

- **Advanced Process Modelling**
- SIPOC (suppliers, inputs, process that is being improved, outputs, and the customers that receive the outputs). BPMN. UML. Value Stream Mapping. IPO (input-process-output model). Gantt. Program Evaluation and Review Technique (PERT) diagrams.

Ресурс https://www.ba.in.ua наводить таке співвідношення вимог до суміжних знань бізнес-аналітика (рис. 1.1).

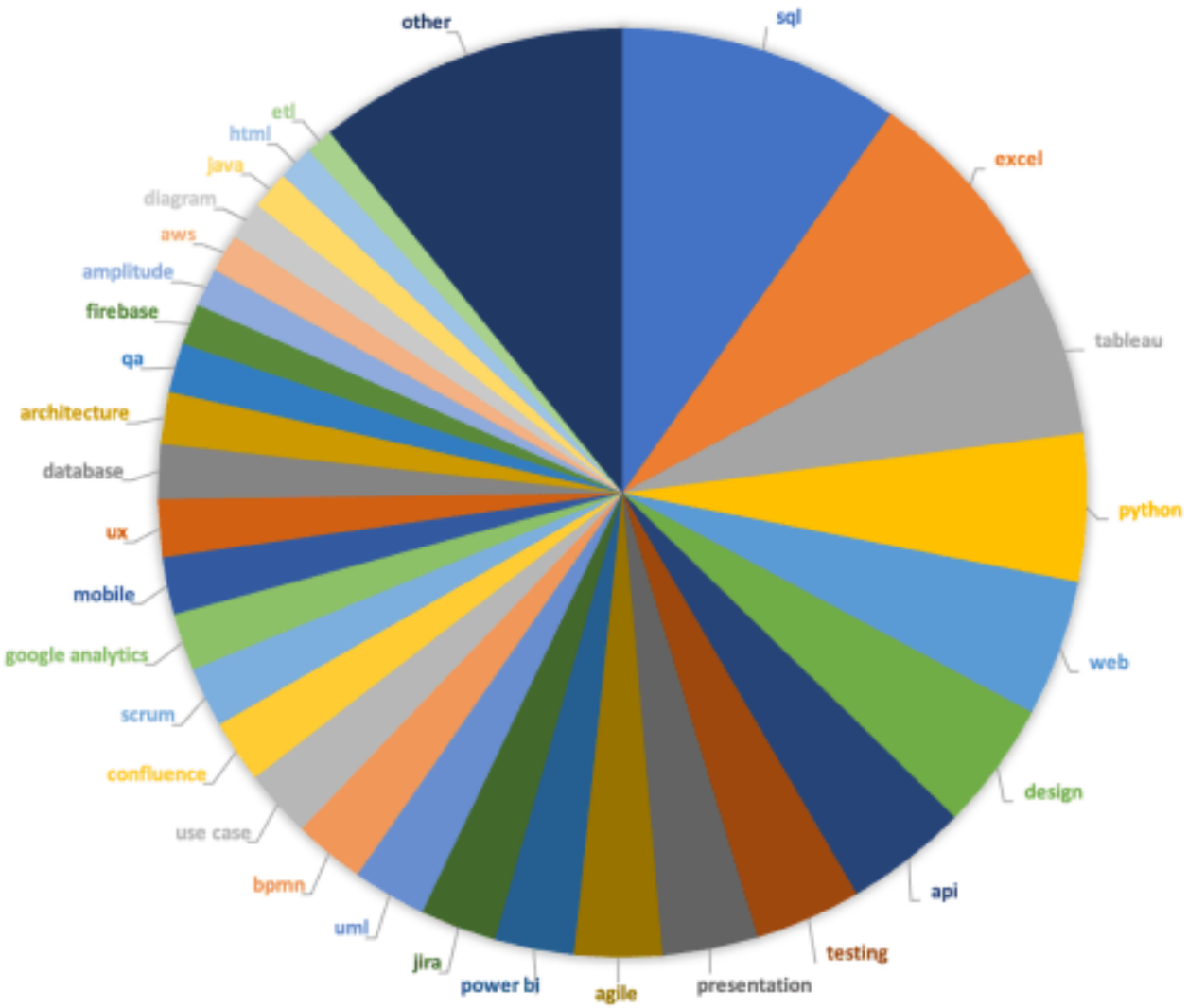


Рис. 1.1. Суміжні знання бізнес-аналітика

Використання методів візуального моделювання [7]

Приклад. IT-проєкт створення «розумної речі» Eagle EYE, продуктом якого є інтелектуально-інтерактивна система спостереження, головний елемент якої — смарт-камера, наділена можливостями комп'ютерного

бачення. Така система позиціонується як цілісний IoT (Internet of Things) — продукт у формі мережі смарт-камер для житла.

Умовно зацікавлені сторони проєкту — керівник IT-проєкту (CEO), проєктний менеджер (PM) і бізнес-аналітик (BA) (рис. 1.2).

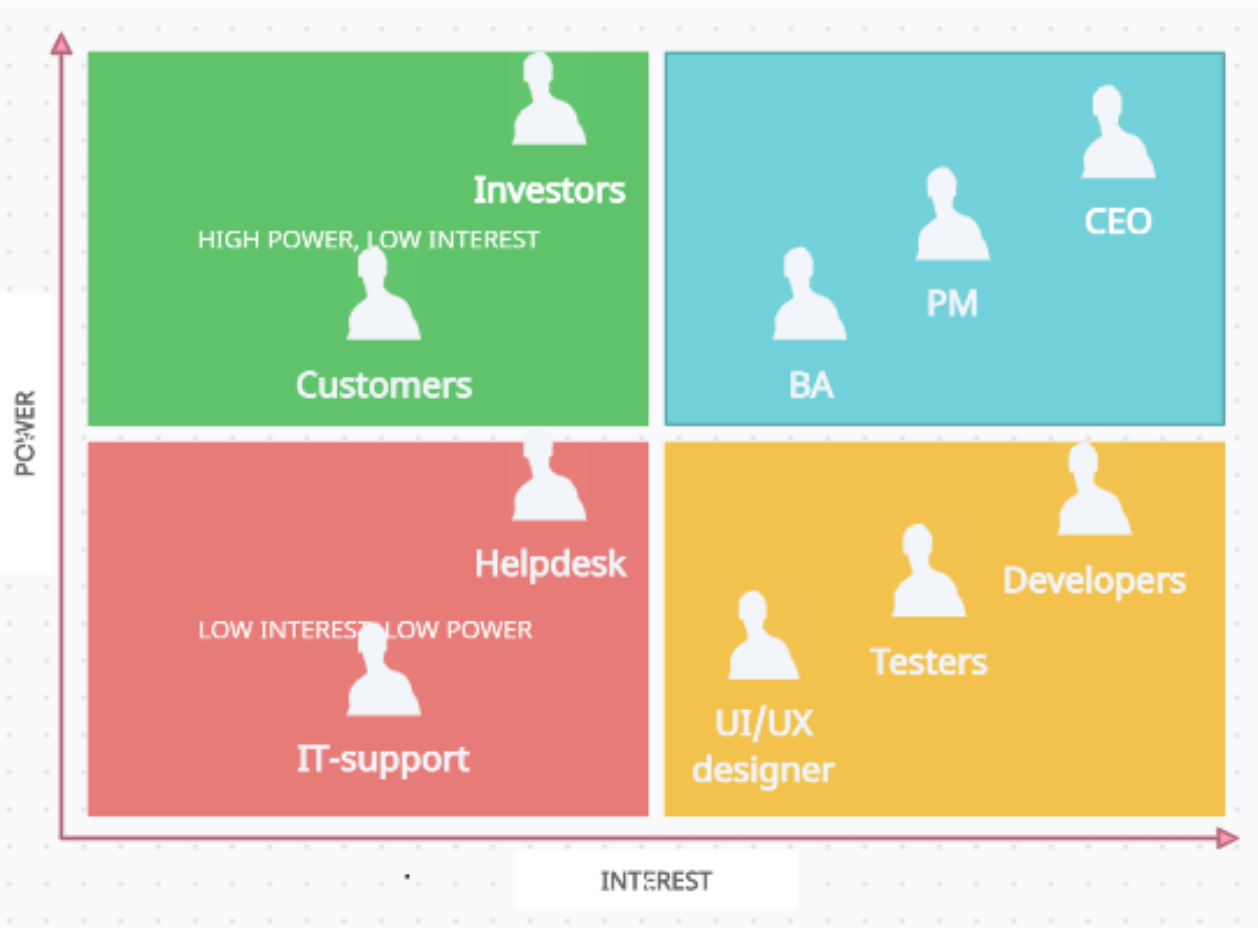


Рис. 1.2. Матриця зацікавлених сторін проєкту «Eagle EYE»

Хоча кожен з етапів процесу бізнес-аналізу вносить свій унікальний елемент у його реалізацію, особливо важливе значення він набуває на етапі визначення вимог до продуктів IT-проєктів.

Моделювання вимог: правильна візуальна модель упорядковує дані та полегшує зацікавленим сторонам виявлення і розуміння проєктних стратегій, відносин і відповідальності.

Роль бізнес-аналітика вимагає не тільки обґрунтованого аналізу даних, але і їхнього ефективного моделювання.

Модель — це представлення набору компонентів процесу, системи або предметної області.

Для початку варто розробити бізнес-модель проєкту, наприклад, на основі Business Model Canvas — інструменту стратегічного управління, що дозволяє візуалізувати та оцінити конкретну бізнес-ідею чи концепцію.

Наступний етап моделювання вимог стосується **бізнес-процесів** — сукупності видів діяльності, яка бере об'єкт як вхідні дані та додає йому цінності для того, щоб задовольнити вимоги. Метою створення моделі бізнес-процесу є краще розуміння, аналіз, удосконалення або заміна бізнес-процесу.

Часто модель бізнес-процесів використовується разом із реінжинірингом бізнес-процесів. Реінжиніринг бізнес-процесів — це систематизований, дисциплінований підхід до вдосконалення, який вивчає, переосмислює і переробляє процеси виконання місій задля досягнення значних поліпшень в ефективності у сферах, важливих для клієнтів і зацікавлених сторін.

Існує велика кількість усталених методик, що підтримують абстрактне моделювання бізнес-процесів.

Ці методи включають визначення інтеграції для моделювання функцій (англ. IDEF — Integration Definition for Function Modeling), UML моделювання (англ. UML — Unified Modeling Language), блок-схеми тощо.

Незалежно від вибору, існує одна **ціль**, яка є спільною для всіх цих методів: надати інформативне зображення об'єктів, які виконують певні функції для отримання результату.

Для проєкту «Eagle EYE» первинна діаграма IDEF0 відображена на рис. 1.3.

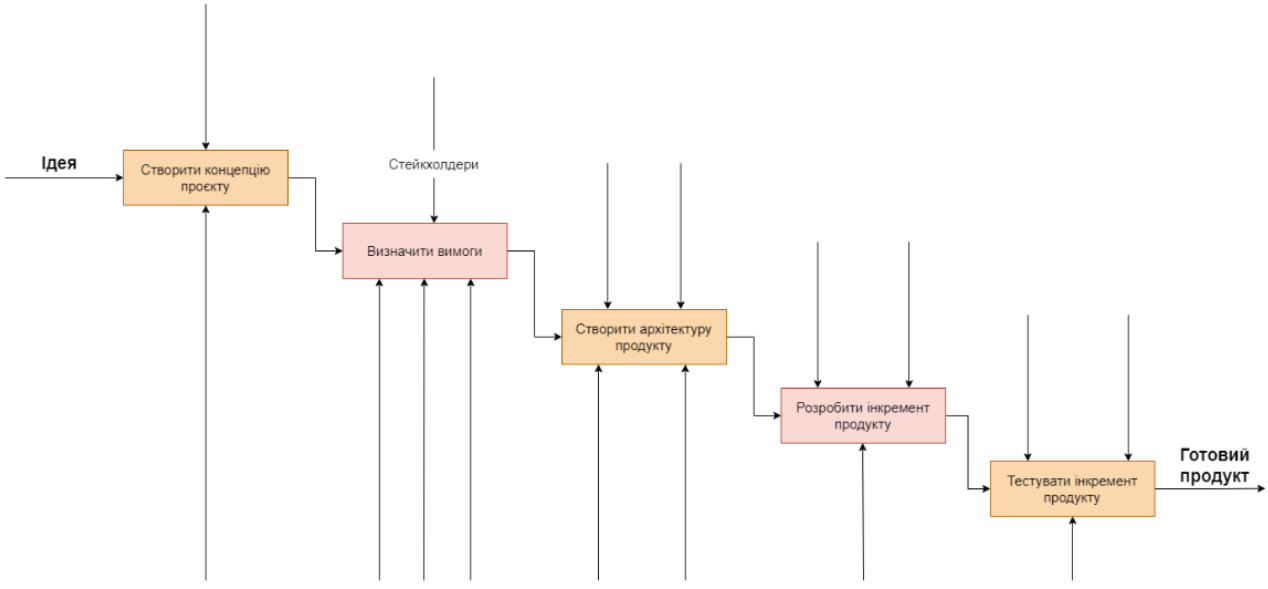


Рис. 1.3. Первинна діаграма IDEF0 (спрощений вигляд)

Тут, початковими вхідними даними виступає ідея проєкту. На її основі створюється концепція проєкту.

Регулюючим елементом є бізнес-аналітик, а механізмом здійснення — Business Model Canvas.

На виході продукується реальна концепція, базуючись на якій, можна зрозуміти загальний проєктний план. Спираючись на нього, можна отримати можливість точніше визначити стейкхолдерів, які, у свою чергу, нададуть необхідні вимоги.

На етапі створення архітектури IT-продукту бізнес-аналітик розробляє UML-діаграми, які полегшать початкову фазу розроблення. Для цієї мети використовуються CASE-засоби (Computer-Aided Software Engineering) — спеціальний набір технік і методів програмної інженерії для проєктування програмного продукту.

Діаграма прецедентів побудована за допомогою CASE-засобу Draw.io (альтернатива: Visual Paradigm) (рис. 1.4).

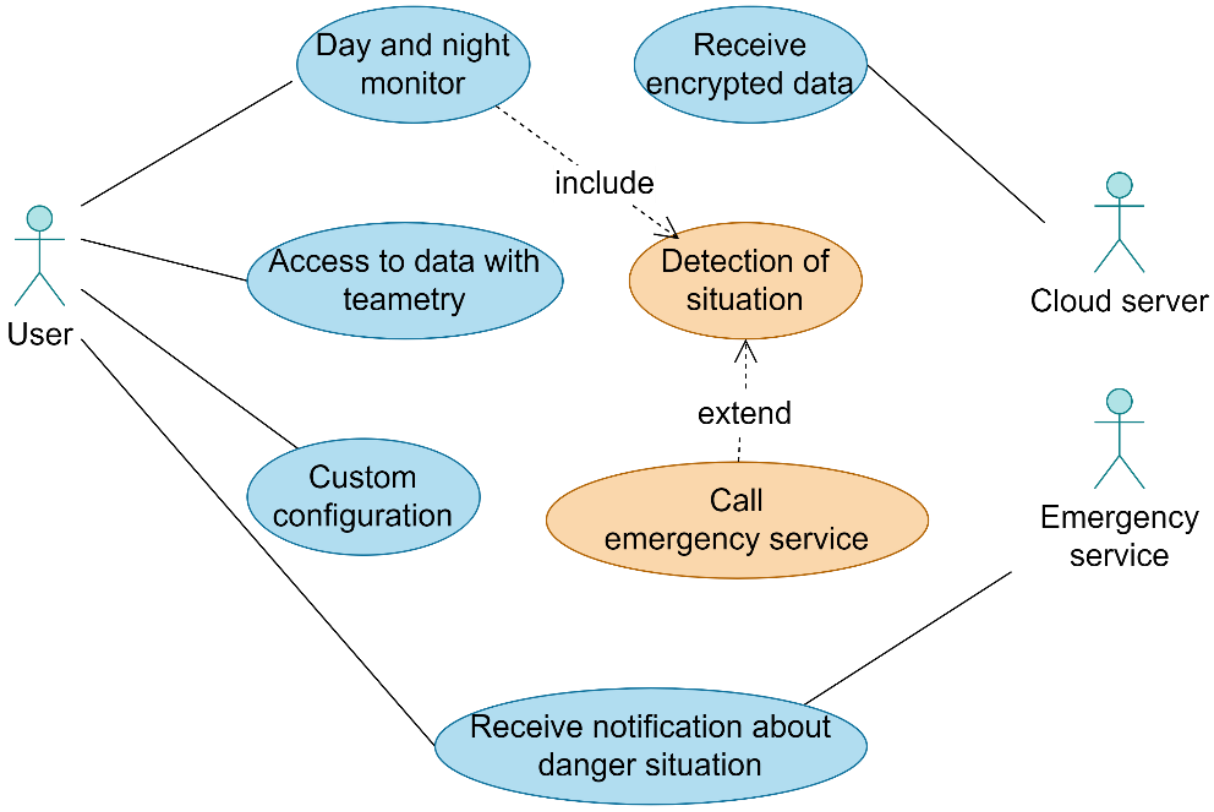


Рис. 1.4. Діаграма прецедентів

Елементи діаграми: користувач (user), служба із надзвичайних ситуацій (emergency service), хмарне середовище (cloud server) для зберігання даних.

Висновок: кожний IT-продукт виникає через реалізацію його проєкту. Успіх IT-проєкту залежить від багатьох чинників, серед яких: пріоритети організації, якісний збір вимог, чіткі цілі та бачення проєкту і комунікація у проєкті.

Бізнес-аналітик відповідає за визначення реальних потреб зацікавлених сторін і старається полегшити комунікацію між організаційними підрозділами, виконуючи роль «зв’язкового», для узгодження потреб бізнес-сторони з командою розробників. Також бізнес-аналітик координує обов’язки з проєктним менеджером щодо планування діяльності з аналізу бізнесу.

ТЕМА 2. ЖИТТЄВИЙ ЦИКЛ ІТ ПРОЄКТУ. МЕТОДИ БІЗНЕС-АНАЛІЗУ ПРИ ІНІЦІАЛІЗАЦІЇ ТА ПЛАНУВАННІ ІТ ПРОЄКТУ

Сутність проєктної діяльності: поняття, основні характеристики, класифікація проєктів. Управління проєктами як специфічна галузь менеджменту. Цілі, процеси та функції в управлінні проєктами. Характеристика моделі управління проєктами. Оточення та учасники проєкту. Життєвий цикл розробки програмного забезпечення.

Етапи процесу бізнес-аналізу. Застосування системного підходу в процесі бізнес-аналізу. Зовнішнє середовище системи. Зв'язки всередині системи та зв'язки системи в зовнішньому середовищі. Формалізовані та неформалізовані методи системного аналізу. Система планування проєкту. Розробка концепції проєкту. Обґрунтування доцільності проєкту.

Загальні поняття

З точки зору системного підходу проєкт може розглядатися як процес переходу з початкового стану в кінцевий – результат за участю ряду обмежень і механізмів.

Проєкт — це сукупність цілеспрямованих, послідовно орієнтованих у часі, одноразових, комплексних і нерегулярно повторюваних дій (заходів або робіт), орієнтованих на досягнення кінцевого результату в умовах обмеженості ресурсів і заданості термінів їх початку і завершення.

Проєкт – це задача з певними вихідними даними і необхідними результатами (цілями), що обумовлюють спосіб її вирішення.

Проєкт – тимчасове виробництво, спрямоване на створення унікального продукту, послуги або результату (PMBOK[4], PMI).

Проєкт включає в себе задум (проблему), засоби його реалізації (вирішення проблеми) і одержувані в процесі реалізації результати.

Проєкт як послідовність дій існує рівно стільки часу, скільки його потрібно для отримання кінцевого результату.

ІТ проєкт – цілеспрямоване, обґрунтоване та сплановане створення або модернізація програмно-технічних засобів, програмних комплексів, технічної та організаційної документації для них, прийняття управлінських

[4] A Guide to the Project Management Body of Knowledge (PMBOK® Guide)" by Project Management Institute

рішень в межах матеріальних, фінансових, трудових та інших ресурсів, що виділені для реалізації проєкту.

Відмінними рисами проєкту є:

- спрямованість на досягнення конкретних цілей;
- координоване виконання взаємозалежних дій;
- обмеженість у ресурсах та фінансах;
- обмеженість у часі з визначеними початком та завершенням;
- **унікальність.**

Спрямованість на досягнення окреслених цілей. Чітка постановка кінцевої мети проєкту сприяє успішній його реалізації за умови правильного формулювання проміжних взаємозалежних цілей. Реалізація проєкту означає послідовне досягнення цілей з найбільш низького рівня до вищого, тобто до досягнення кінцевої мети.

Координоване виконання дій виконавців. Одні дії необхідно виконувати паралельно, інші – послідовно, і будь-яке порушення порядку їх виконання може поставити під загрозу виконання проєкту взагалі.

Обмеженість в часі і ресурсах. Проєкти виконують протягом певного часу, по можливості більш чітко окреслюючи початок і завершення. Запорукою успішної реалізації проєкту є оптимальний розподіл зусиль і ресурсів у часі, яке забезпечується приведенням в порядок послідовності виконання робіт і заходів в межах проєктної діяльності.

Ресурси проєкту мають бути заздалегідь визначені та обмежені обсягом інвестицій у цей проєкт.

Унікальність. Всі проєкти мають відмінні риси і ознаки. Не існує ідентичних проєктів, навіть якщо вони передбачають виконання аналогічних дій.

ПРОЄКТ – це процес, обмежений часом та бюджетом.

ПРОДУКТ – це результат, який готовий виконувати певні операції, його життєвий цикл немає жорстких тимчасових рамок, а для його створення та функціонування найчастіше необхідно кілька проєктів.

Таблиця 2.1 – Класифікація IT проєктів

Варіант № 1	**Варіант № 2 (все залежить від ознаки: девайси, технології, платформи тощо)**
NEW PROJECT DEVELOPMENT (нова унікальна розробка)	PRODUCTS (націлені на певні сегменти, напрями, девайси. Наприклад, веб-сайт, мобільний додаток, класичний десктопний застосунок тощо під певну операційну систему)
CONTINUES OF THE EXITED PROJECT (вдосконалення/переробка/розширення існуючої розробки)	SERVICES (націлені на автоматизацію певної послуги: доставка, бронювання, пересування, доступ до даних (фото, відео) тощо)
MAINTENANCE AND SUPPORT (обслуговування та підтримка раніше розробленого продукту)	SYSTEMS (складні комплекси, які керують підсистемами, девайсами. Наприклад, операційні системи)

Управління проєктом — це процес управління командою і ресурсами проєкту за допомогою специфічних методів, завдяки яким проєкт завершується успішно і досягає своєї мети.

PMBOK: *управління проєктами* — це застосування знань, навичок, інструментів і методів до операцій проєкту для задоволення потреб, які висуваються до проєкту.

Управління проєктом як процесом передбачає досягнення цілі згідно з визначеними вимогами з врахуванням обмежень за термінами, вартістю і показникам якості.

PM складається з десяти функцій (областей знань):

- Управління інтеграцією проєкту (Project Integration Management).
- Управління змістом проєкту (Project Scope Management).
- Управління термінами проєкту (Project Time Management).
- Управління вартістю проєкту (Project Cost Management).
- Управління якістю проєкту (Project Quality Management).

• Управління людськими ресурсами проєкту (Project Human Resource Management).

• Управління комунікаціями проєкту (Project Communications Management).

• Управління ризиками проєкту (Project Risk Management).

• Управління поставками проєкту (Project Procurement Management).

• Управління зацікавленими сторонами проєкту (Project Stakeholder Management).

Основні функції PMBOK визначено **за цілями**, за досягнення яких відповідає проєктний менеджер, а додаткові — **за об'єктами**, на які спрямовується діяльність керівника. Проте в назві всіх цих функцій наявне спільне слово управління, що, в свою чергу, передбачає виконання в їх межах таких функцій управління, як організація, планування, контроль, мотивація. Інакше кажучи, проєктний менеджер повинен здійснювати основні функції управління щодо специфічних цілей та об'єктів очолюваних ним проєктів.

Функції проєктного менеджменту ще називають областями знань по управлінню проєктами[5].

Дуже часто проєкт виникає як відповідь на наявні проблеми підприємства. Тоді для його реалізації треба здійснити такі кроки (рис. 2.1):

Перший етап — розробка альтернативних рішень (1–5-й кроки).

Другий етап — прийняття рішення (6-й крок).

Третій етап — впровадження (7–10-й кроки).

Поєднати основні функції управління проєктами з інструментарієм, який для цього застосовується, можна за допомогою моделі управління проєктами (рис. 2.2). Ефективність проєкту залежить від рішень на кожній стадії його здійснення, причому неправильне вихідне розуміння цілей спричиняє по ланцюжку помилки у постановці задач та у визначенні обсягу робіт за проєктом, що, в свою чергу, призводить до втрат часу і коштів.

Встановлення цілей проєкту передбачає дотримання таких правил:

• результат проєкту повинен бути чітко окреслений (обсяг робіт);

• проєкт має здійснюватися у визначеному зовнішньому середовищі (учасники);

[5] Відмінність завдань від функцій: у завдання є конкретне рішення, функція повинна виконуватися постійно.

• повинні бути встановлені терміни проєкту (строки);

• бюджет проєкту не повинен перевищувати заданої величини (затрати);

• продукт має задовольняти визначеним стандартам (якість).

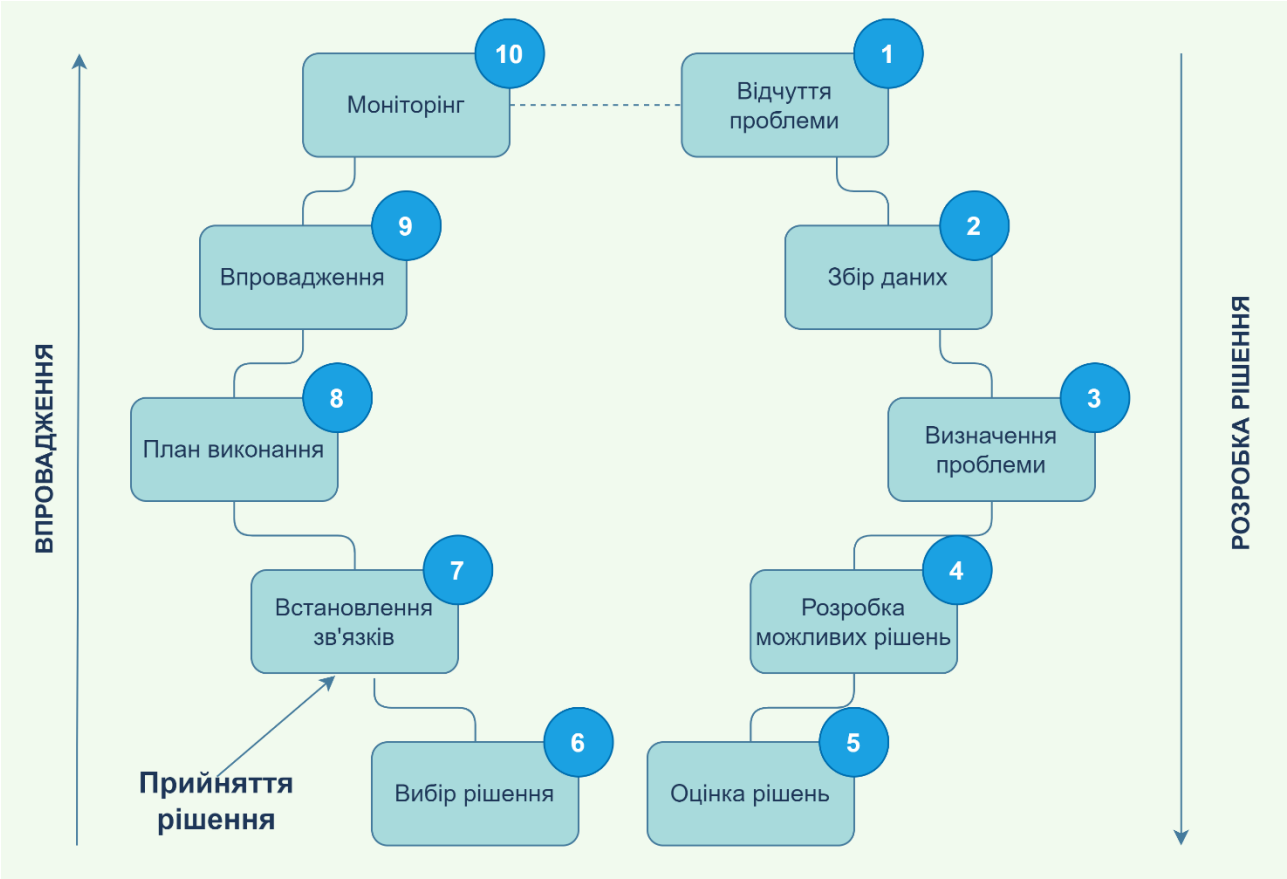


Рис. 2.1. Кроки реалізації проєкту як циклу вирішення проблеми

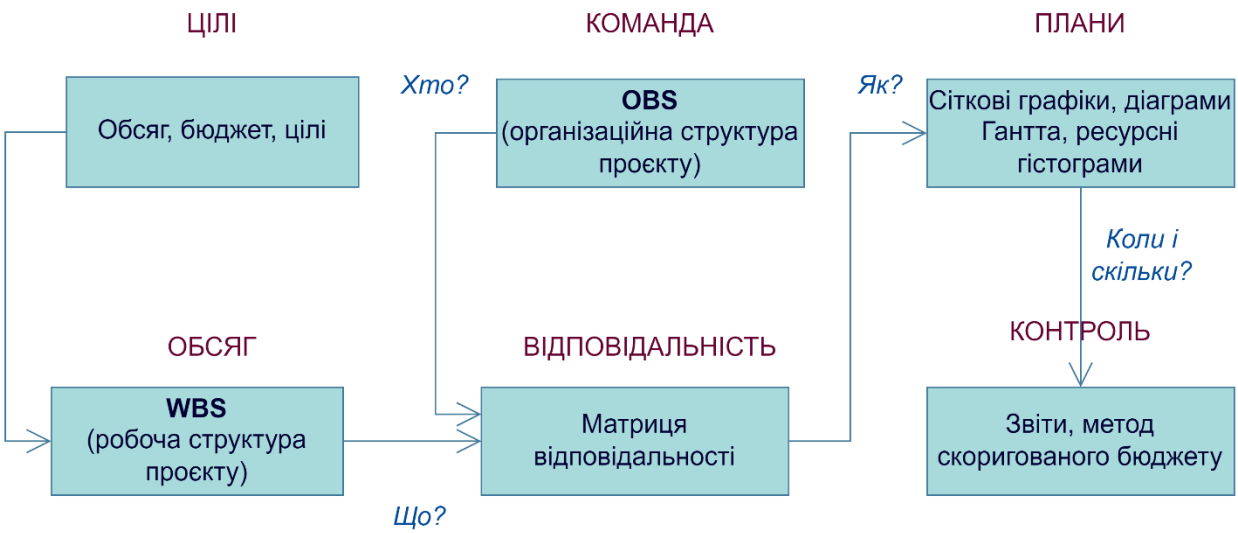


Рис. 2.2. Модель управління проєктом

Цей список можна продовжити. Проте, щоб основні вимоги не були взаємовиключаючими, всі вони повинні бути погоджені на ранніх стадіях проєкту.

Цілі проєкту й основні його характеристики фіксуються у так званому формулярі проєкту як результат першої фази проєктного менеджменту — узгодження проєкту.

Після встановлення цілей і з'ясування основних вимог до проєкту та його результатів в управлінні проєктом починається фаза планування. Для декомпозиції проєкту на доступні для огляду (на етапі планування) і керовані (на етапі реалізації) частини використовується робоча структура проєкту — **WBS (Work Breakdown Structure).** Вона відповідає на запитання «Що треба зробити за проєктом?». Залежно від масштабу проєкту кількість рівнів декомпозиції може бути різною, аж до виокремлення робіт, готових для включення в сіткову модель.

Формування WBS логічно тягне за собою запитання «А хто виконуватиме ці роботи?», відповіддю на яке є створення організаційної структури проєкту **OBS (Organization Breakdown Structure)**. Вона визначає відносини між учасниками проєкту, їх відповідальність і повноваження в процесі реалізації проєкту.

Оскільки найпоширенішою формою проєктних структур є матрична організаційна структура, якій притаманна подвійна підпорядкованість членів проєктної команди проєктному менеджеру і керівникові функціонального підрозділу, то інструментом, який дозволяє вирішувати можливі проблеми і суперечки, виступає матриця відповідальності, що пов'язує структуру робіт і відповідальність персоналу, дає відповідь на запитання «Хто що робить і за що відповідає?».

Після цього значно легше здійснюється наступний блок планування — планування термінів виконання проєктних робіт — складанням укрупнених сіткових графіків, обчисленням їх параметрів методом критичного шляху з подальшою розробкою діаграм Гантта як інструментів календарного планування. По кожній роботі графіка визначаються ресурси, які поєднуються в ресурсні гістограми. Оцінені в грошах витрати подаються в часі у вигляді бананоподібної кривої. На цьому планування завершується, оскільки визначено, як досягатимуться всі проєктні цілі.

На етапі реалізації проєкту домінує функція контролю. Система контролю встановлює основу для спостереження, оцінки й приведення початкового плану у відповідність зі змінами, що відбулися. Контроль тісно пов'язаний із системою звітності й оцінки. Звітність встановлює

інформаційну систему проєкту, що дозволяє контролювати процес його виконання і чисельно оцінювати його результативність. Основним методом оцінки проєкту є метод скоригованого бюджету (Earned Value), який дає змогу визначити рівень виконання проєктних робіт щодо встановлених термінів, обсягів і витрат.

Таким чином, у моделі управління проєктом зведені разом цілі, функції та інструменти проєктного менеджменту, які узагальнені у табл. 2.1.

Таблиця 2.1 – Характеристика моделі управління проєктом

ЦІЛІ Інструмент — контракт	Визначаються вимоги до проєкту з огляду на обсяги, витрати, час і якість, а також наголошується, які з них домінують
ЩО (обсяг) Інструмент — WBS	Визначаються обсяги робіт розробкою робочої структури проєкту (WBS)
ХТО (команда) Інструмент — OBS	Призначається керівник і формується команда за допомогою створення організаційної структури (OBS) і порівняння вимог проєкту зі здібностями виконавців
ХТО ЩО РОБИТЬ (відповідальність) Інструмент — матриця відповідальності	Створюється матриця відповідальності, в якій роботи закріплюються за виконавцями із визначенням міри відповідальності
ЯК (плани) Інструменти — сіткові графіки, діаграми Гантта, ресурсні гістограми	Узгоджуються плани виконання проєкту щодо встановлених цілей і взаємовідношень робочих елементів
КОЛИ і СКІЛЬКИ (контроль) Інструмент — інформаційні та аналітичні звіти, метод скоригованого бюджету	Визначаються документи, які містять інформацію для контролю щодо термінів, обсягів, бюджету шляхом визначення відхилень від плану

Оточення та учасники проєкту

Оточення проєкту (Project Environment) — сукупність зовнішніх та внутрішніх сил, які сприяють чи заважають досягненню цілей проєкту.

До факторів ближнього оточення проєкту перш за все відноситься керівництво підприємства, яке визначає цілі та основні вимоги до проєкту. Великий вплив на проєкт можуть надавати основні структурні підрозділи підприємства, інфраструктура підприємства, ставлення громадських організацій і колективу в цілому. Через взаємодію проєкту зі службами та відділами відбувається вплив близького оточення на проєкт. Враховуючи його «близькість», такий вплив є найрегулярнішим і найсуттєвішим.

До факторів далекого оточення належать ті сфери, галузі життєдіяльності суспільства, в яких діє організація. Насамперед це ринки, на яких працює організація. Крім того, великий вплив на проєкт мають такі фактори, як політика держави, стан її економіки та науки, особливості законодавства, культурних та природних чинників.

Враховуючи, що проєкт реалізується зазвичай у конкретному середовищі, слід мати на увазі й зовнішні фактори, в яких він реалізується. Такими факторами є політичні, економічні, соціальні, правові, науково-технічні, природні та екологічні.

До внутрішніх належать чинники, пов'язані з організацією проєкту. Організація проєкту є розподілом прав, відповідальності та обов'язків між учасниками проєкту.

Учасники проєкту – це люди або організації, які залучені до виконання проєкту, а також ті, хто залежить або зацікавлений у результатах проєкту, його успішному виконанні.

Синонімом словосполучення «учасники проєкту» є «зацікавлені сторони» **(stakeholders)** — цей термін був схвалений ISO (Міжнародна Організація зі Стандартизації) і прийнятий в базисі компетенцій IPMA (Міжнародна Асоціація Управління Проєктами).

Stakeholders[6] **–** особа, група осіб або організація, які можуть впливати, перебувати під впливом або вважати себе під дією впливу рішення, операції або кінцевого результату проєкту, програми або портфеля.

Аналіз стейкхолдерів – метод систематичного збору та аналізу кількісної та якісної інформації, щоб визначити, чиї інтереси необхідно враховувати впродовж проєкту.

[6] https://pmiukraine.org/pmbok7/

Всі зацікавлені сторони можуть здійснювати вплив на проєкт прямо і опосередковано. Такі джерела впливу, як інтереси зацікавлених сторін, організаційна зрілість в управлінні проєктом, встановлений порядок управління проєктом, стандарти, проблеми, тенденції і повноваження можуть впливати на зародження і розвиток проєкту.

Учасники проєкту реалізують різні інтереси у процесі здійснення проєкту, формують власні вимоги відповідно до цілей та мотивації і впливають на проєкт, виходячи зі своїх інтересів, компетенцій та ступеню залучення до проєкту.

Склад учасників проєкту, їх ролі, розподіл функцій і відповідальності залежать від типу, виду, масштабу й складності проєкту, а також від фаз його життєвого циклу.

Учасники можуть буту активними, тобто такими, які самостійно реалізують діяльність по проєкту чи діяльність, результати якої впливають на проєкт (взаємодіють з проєктом), та пасивними, тобто тими, хто відчуває вплив зі сторони проєкту.

Крім того, учасники можуть бути безпосередніми (активними чи пасивними), тобто учасниками самої діяльності по проєкту, або непрямими (активними чи пасивними), тобто учасниками діяльності, яка реалізується об’єктами навколишнього середовища і впливає на проєкт чи відчуває вплив проєкту.

Розрізняють таких учасників проєкту:

1. Менеджер (керівник) проєкту (Project Manager) – особа, відповідальна за управління проєктом.

2. Спонсор (куратор) проєкту (Project Sponsor) – особа усередині або поза організацією, що забезпечує фінансові ресурси проєкту.

3. Замовник (Project Customer) – особа (організація), яка приймає результати роботи і платить за її виконання.

4. Користувач продукту проєкту (User) – особа усередині або поза організацією, яке використовуватиме результати проєкту.

5. Виконуюча організація (Performing organization) – організація, співробітники якої безпосередньо залучені у виконання проєктних робіт.

6. Члени проєктної команди (Project team members) – група, що виконує роботу за проєктом.

7. Впливові особи (Influencers) – особи або групи осіб, які прямо не будуть розпоряджатися або використовувати результати проєкту, але

через своє положення в організації можуть вплинути, позитивно або негативно, на просування проєкту.

8. Проєктний офіс (Project Management Office) – підрозділ, що прямо або побічно відповідає за результат проєкту.

9. Команда управління проєктом – частка проєктної команди, що бере участь в управлінні.

Наведений перелік учасників може змінюватися і доповнюватися залежно від умов конкретного проєкту.

Виділяють ще учасника – «ініціатор» – учасник проєкту, який є носієм основної ідеї проєкту та ініціативи його реалізації. В якості ініціатора може виступати будь-який з учасників проєкту.

Життєвий цикл проєкту[7]

Життєвим циклом проєкту або проєктним циклом є відрізок часу між початком проєкту і його завершенням. Початком проєкту можна вважати момент зародження ідеї або момент початку її реалізації.

Життєвий цикл ПЗ – стадії, що проходить програмний продукт від появи ідеї до її реалізації в коді, імплементації у бізнес і подальшої підтримки.

Стандартні етапи ЖЦ:

1. Ініціалізація.
2. Аналіз вимог.
3. Проєктування.
4. Програмування.
5. Тестування і налагодження.
6. Експлуатація, супровід і підтримка.

Всі моделі ЖЦ можна розділити на дві великі групи: послідовні та ітераційні моделі.

Waterfall (каскадна модель): етапи залежать один від одного і наступний починається, коли завершений попередній, утворюючи таким чином поступальний (каскадний) рух уперед.

Паралелізм етапів у каскадній моделі, хоч і обмежений, але можливий для абсолютно незалежних між собою робіт. При цьому інтеграція паралельних частин все одно відбувається на якомусь

[7] https://evergreens.com.ua/ua/articles/software-development-metodologies.html
https://training.qatestlab.com/blog/technical-articles/popular-software-development-life-cycles/

наступному етапі, а не в межах одного. Команди різних етапів між собою не комунікують, відповідаючи тільки за свій етап.

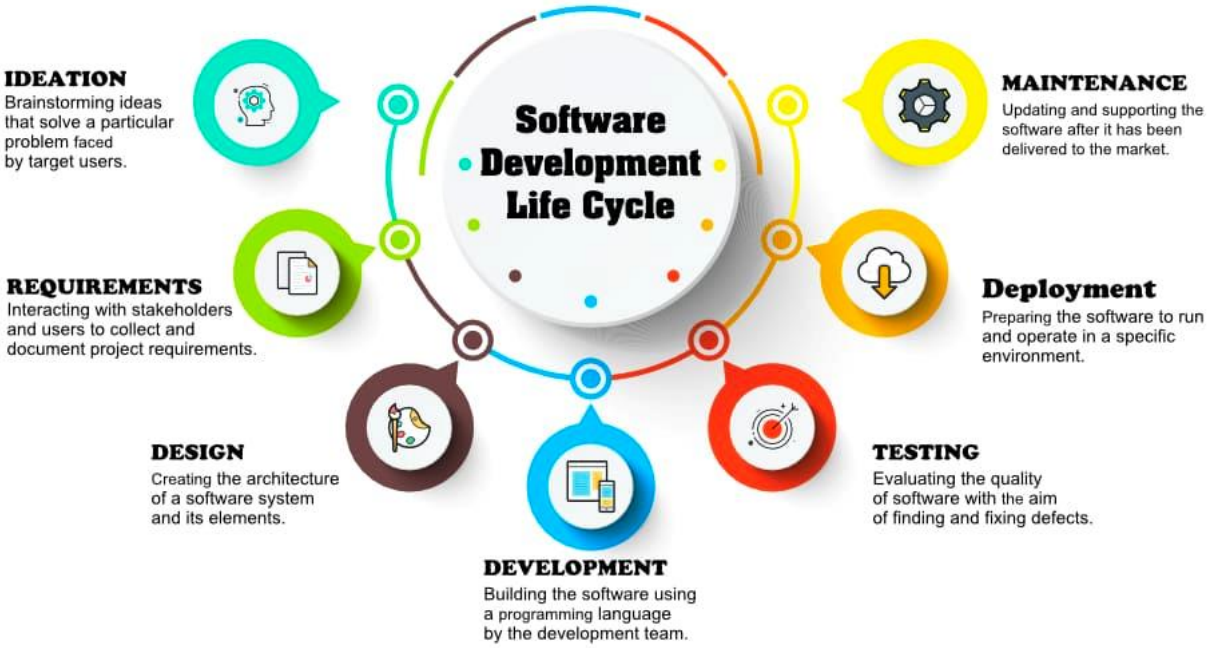


Рис. 2.3. Етапи ЖЦ продукту

Застосовується на довготривалих і великих проєктах.

Переваги:

– всі стадії проєкту виконуються в чіткій послідовності;

– чіткість етапів дозволяє планувати терміни завершення всіх робіт і відповідні ресурси (грошові і людські);

– вимоги залишаються незмінними протягом усього циклу.

Мінуси:

– складності при формулюванні чітких вимог і неможливість їхньої зміни;

– тестування починається тільки з середини розвитку проєкту;

– до завершення процесу розробки користувачі не можуть переконатися, чи якісний продукт, який розробляється.

Внесення замовником значних змін під час процесу розробки або виникнення неврахованих ризиків призводять до перебудови та перепланування всього проєкту.

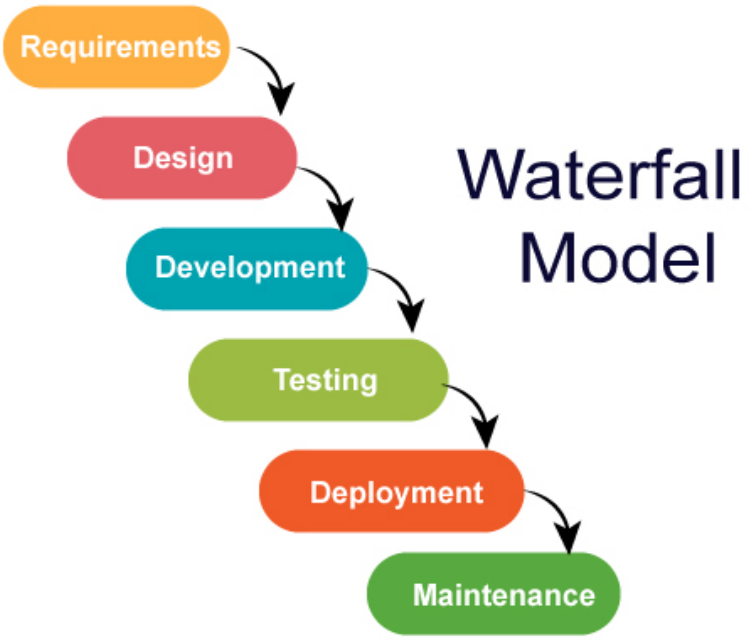


Рис. 2.4. Каскадна модель ЖЦ

Incremental model передбачає розбиття проєкту на частини (етапи, ітерації) і проходження етапів життєвого циклу на кожному з них. Кожен етап є закінченим сам по собі, сукупність етапів формує кінцевий результат. З кожним етапом розробка наближається до кінцевого бажаного результату або уточнюються вимоги до результату по ходу розробки, і відповідно в будь-який момент поточна ітерація може виявитися останньою або черговою на шляху до завершення. Даний підхід дозволяє боротися з невизначеністю, знімаючи її етап за етапом, і перевіряти правильність технічного, маркетингового або будь-якого іншого рішення на ранніх стадіях.

Плюси:

– замовник може дати свій відгук щодо кожної версії продукту;

– є можливість переглянути ризики, які пов'язані з витратами і дотриманням графіка;

– звикання замовника до нової технології відбувається поступово.

Мінуси:

– функціональна система повинна бути повністю визначена на початку життєвого циклу для виділення ітерацій;

– при постійних змінах структура системи може бути порушена;

– терміни здачі системи можуть бути збільшені через обмеженість ресурсів (виконавці, фінанси).

Рис. 2.5. Інкрементна модель ЖЦ

Ітеративна модель не передбачає повного обсягу вимог для початку робіт над продуктом. Розробка програми може починатися з вимог до частини функціоналу, які можуть згодом доповнюватися і змінюватися. Процес повторюється, забезпечуючи створення нової версії продукту для кожного циклу.

У дещо спрощеному вигляді, ітеративна модель складається з чотирьох основних стадій, які повторюються у кожній з ітерацій (plan-do-check-act):

– визначення та аналіз вимог;

– дизайн та проектування – згідно вимогами. Причому дизайн може як розроблятися окремо для даної функціональності, так і доповнювати вже існуючий;

– розробка і тестування – кодування, інтеграція і тестування нового компонента;

– фаза рев'ю – оцінка, перегляд поточних вимог та пропозиції щодо доповнення цих вимог.

За результатами кожної ітерації приймається рішення – чи будуть використані її результати для доповнення існуючої функціональності в якості вхідної точки для початку наступної ітерації (т.зв. інкрементальне прототипування). Зрештою, досягається точка, в якій всі вимоги були втілені в продукт – відбувається реліз.

Ітеративну модель використовують:

– для великих проєктів;

– коли відомі, принаймні, ключові вимоги;
– коли вимоги до проєкту можуть мінятися в процесі розробки.

Spiral model (спіральна модель): усі етапи життєвого циклу йдуть витками, на кожному з яких відбуваються проєктування, кодування, дизайн, тестування і т.д. Такий процес відображає суть назви: піднімаючись, проходиться один виток (цикл) спіралі для досягнення кінцевого результату. Причому не обов'язково, що один і той же набір процесів буде повторяться від витка до витка. Але результати кожного з витків ведуть до головної мети.

Плюси:
– приділяється особлива увага управлінню ризиками;
– додаткові функції можуть бути додані на пізніх етапах;
– є можливість гнучкого проєктування.

Мінуси:
– оцінка ризиків на кожному етапі є досить витратною;
– постійні відгуки і реакція замовника може провокувати все нові і нові ітерації, які можуть призводити до тимчасового затягування розробки продукту;
– більш застосовується для великих проєктів.

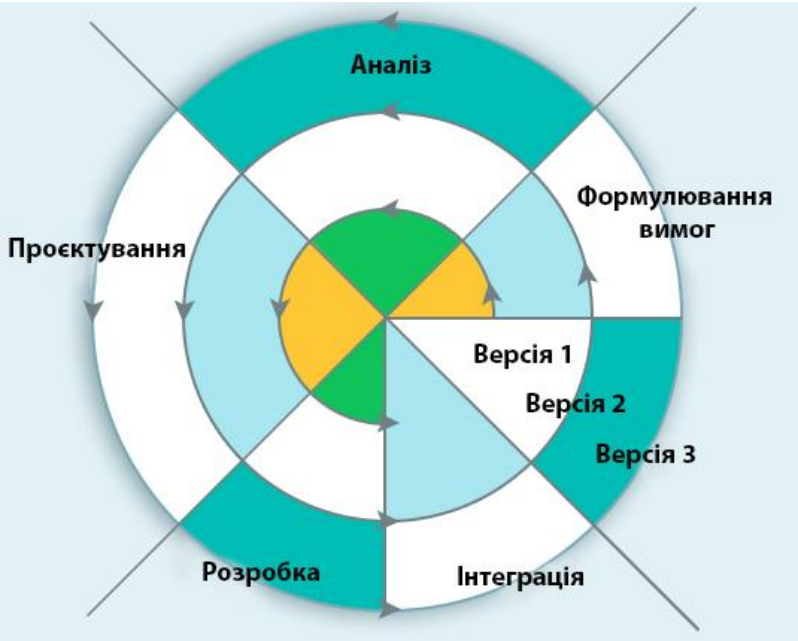


Рис. 2.6. Спіральна модель ЖЦ

Agile – набір принципів гнучкої розробки (12 принципів[8]) та ідей.

[8] https://agilemanifesto.org/iso/uk/manifesto.html

Основні ідеї Agile:

1. **Люди та співпраця** важливіші за процеси та інструменти.
2. **Працюючий продукт** важливіший за вичерпну документацію.
3. **Співпраця із замовником** важливіша за обговорення умов контракту.
4. **Готовність до змін** важливіша за дотримання плану.

Один з принципів – взаємодія – має на увазі, що замовник взаємодіє з командою, команда з замовником – усі між собою. Це дозволяє обмінюватися досвідом між учасниками команди і клієнтом і кожному з них впливати на прийняття рішень. За рахунок такого підходу знижуються ризики втрати часу і грошей і підвищується здатність команди вирішувати складні нестандартні завдання з високим ступенем невизначеності.

Однак взаємодії всіх і з усіма можуть вилитися у хаос, що впливає на всі сфери розробки. Тому використовуючи Agile потрібно розуміти обмеження: команди повинні бути невеликі, учасники повинні бути компетентні та мотивовані, ітерації короткі з максимально зрозумілими цілями, встановлені чіткі обмеження за часом і кінцевий результат повинен бути очевидним.

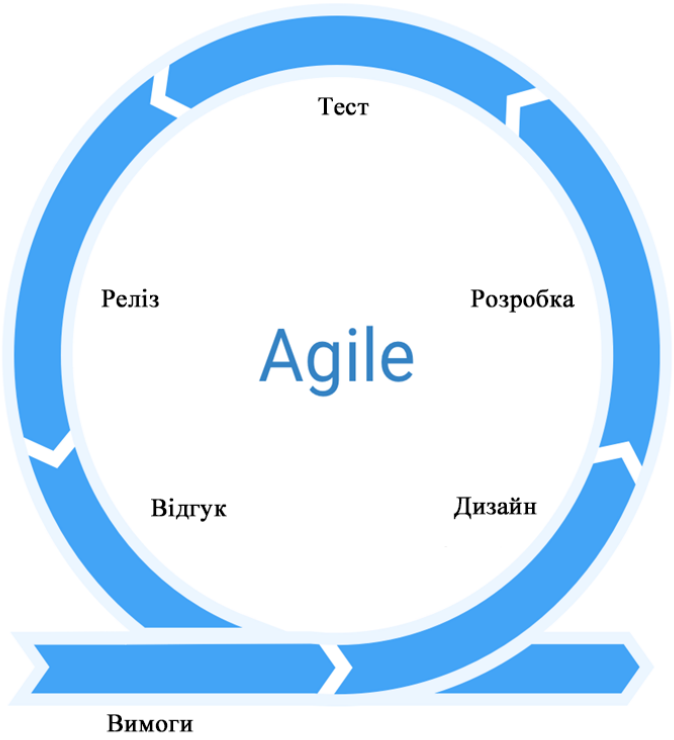


Рис. 2.7. Agile-підхід до ЖЦ

Плюси:

– швидке прийняття рішень завдяки постійним комунікаціям;

– мінімізація ризиків;

– полегшена робота з документацією.

Мінуси:

– велика кількість мітингів і обговорень, що може збільшити час розробки продукту;

– складно планувати процеси, так як вимоги постійно змінюються;

– рідко використовується для реалізації великих проєктів.

Scrum – це гнучка модель розробки ПЗ, в якій робиться акцент на якісному контролі процесу розробки. Ролі в методології (Scrum Master, Product Owner, Team) дозволяють чітко розподілити обов'язки в процесі розробки. За успіх Scrum в проєкті відповідає Scrum Master і є сполучною ланкою між менеджментом і командою. За розробку продукту відповідає Product Owner, який також ставить завдання і приймає остаточні рішення для команди.

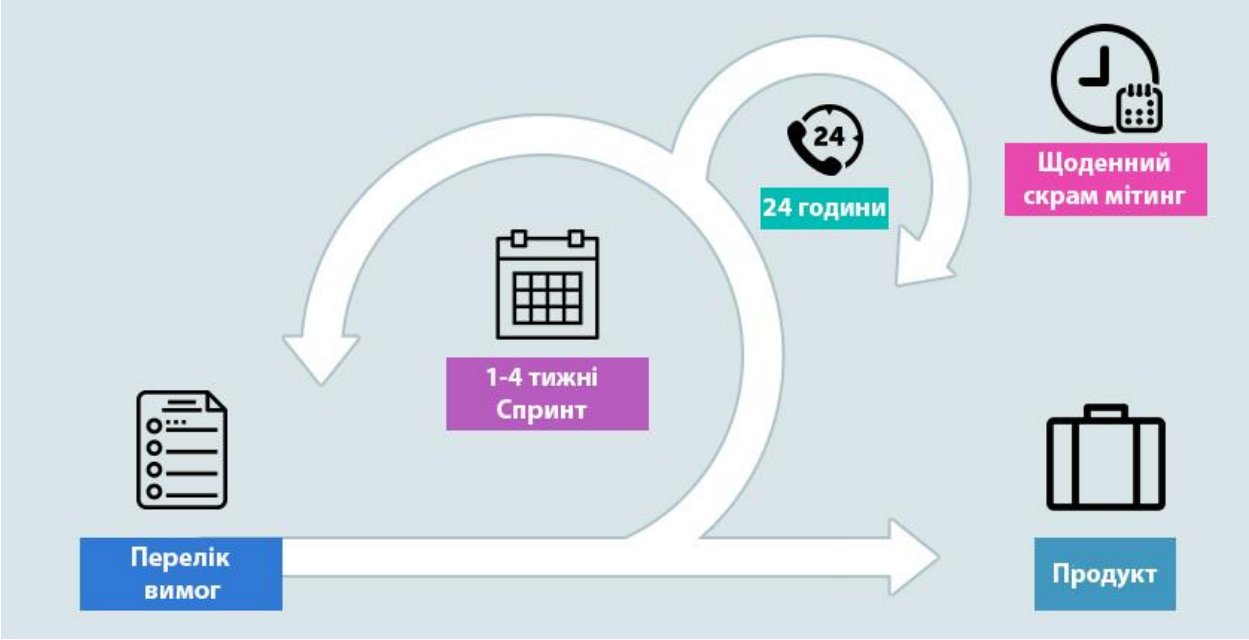


Рис. 2.8. Розробка ПЗ за моделлю SCRUM

Команда – це єдине ціле, в ній результати оцінюються не по кожному окремому учаснику, а по тому, що виходить в результаті у всіх.

Спринти в даній методології тривають від 1 до 4 тижнів. Після кожного спринту команда надає варіант закінченого продукту.

Плюси:

– швидкий зворотній зв'язок від фахівців в різних сферах (дизайнерів, архітекторів, тестувальників та ін.);

– завдяки залученості тестувальника в роботу відбувається швидке додавання нового функціоналу і швидкий запуск продукту з мінімальними функціями;

– самостійна і самоорганізована команда.

Мінуси:

– деякі люди, які знають продукт, стають незамінними, так як документація не надається в процесі розробки;

– неможливо спланувати точну дату завершення, так як все уточнюється за результатами попереднього спринту;

– замовники не завжди можуть зрозуміти суть даної методології і необхідно витратити час на роз’яснення.

Lean – ідея ощадливого ставлення до ресурсів (у тому числі часу) і вирішення завдання найпростішим способом. Наприклад: не робиться весь продукт, щоб зрозуміти, що він нікому не потрібен – робиться лендінг і форма підписки і дається на нього реклама, щоб перевірити, що цей продукт викликає інтерес клієнтів. Далі приймається усвідомлене рішення про необхідність розробки.

Коли доходить до розробки продукту, або робиться якесь поліпшення, виробниче або інженерне, спочатку робиться його **MVP (minimum viable product)**. **MVP** – така версія продукту, що виконує свою головну функцію і при цьому її не відхиляють клієнти і визнають її корисність. Звичайно, її можна покращувати і покращувати, але загалом продукт на стадії MVP повинен бути корисним, зрозумілим клієнтові і таким, щоб можна було прийняти рішення про його подальше поліпшення або визнати експеримент невдалим і тестувати нову гіпотезу, витративши при цьому якомога менше ресурсів.

Загалом Lean підхід орієнтується на тестування потреб і цінностей і потрапляння в очікування ринку мінімальними засобами. Lean-підхід не про «технічні» проблеми і їхні рішення інженерними засобами, а про підприємницький підхід до вирішення завдань. Lean передбачає, що команда шукає найпростіше рішення для досягнення результату: технічно, організаційно і т.п., спрощуючи все, що не є дійсно важливим.

У спрощенні сила і головна «пастка» Lean: прагнення все спростити іноді призводить до ситуацій, в яких продукт спрощують настільки, що

губляться дійсно важливі функції і продукт по суті виявляється непотрібним, оскільки не несе цінності користувачеві.

За **Kanban** методологією проєкт ділиться на етапи, що візуалізуються у вигляді канбан-дошки. Етапи, це наприклад: планування, розробка, тестування, реліз і ет.п. Завдання у вигляді «карток» переміщуються з етапу на етап. Нові завдання можна додавати у будь-який час. Завдання закривається не по закінченню конкретного часу, а по зміні статусу на «завершено».

Таблиця 2.2 – Відмінності між Scrum і Kanban

Scrum	Kanban
Команда приймає участь у конкретній ітерації	Необов'язкова участь
Використання **швидкості** як засобу покращення процесу	Використання **часових рамок** у якості засобів для покращення процесів
Попередня оцінка	Попередня оцінка
Виконання спрінта належить одній команді	Канбан-дошка може бути розділена між декількома командами
Включає в себе використання як мінімум 3 ролей (власник продукту, скрам-майстер, скрам команда)	Відсутність ролей
Скрам-дошка змінюється між спринтами	Канбан-дошка незмінна
Для кожного спринта пріоритет встановлюється в backlog спринта	Призначення пріоритетів не обов'язкове

Команда проєкту

Команда проєкту – група осіб, яка підтримує керівника проєкту у виконанні проєкту для досягнення цілей проєкту. Команда проєкту створюється на період реалізації проєкту та поєднує людей таким чином, що робота набувала синергічного ефекту.

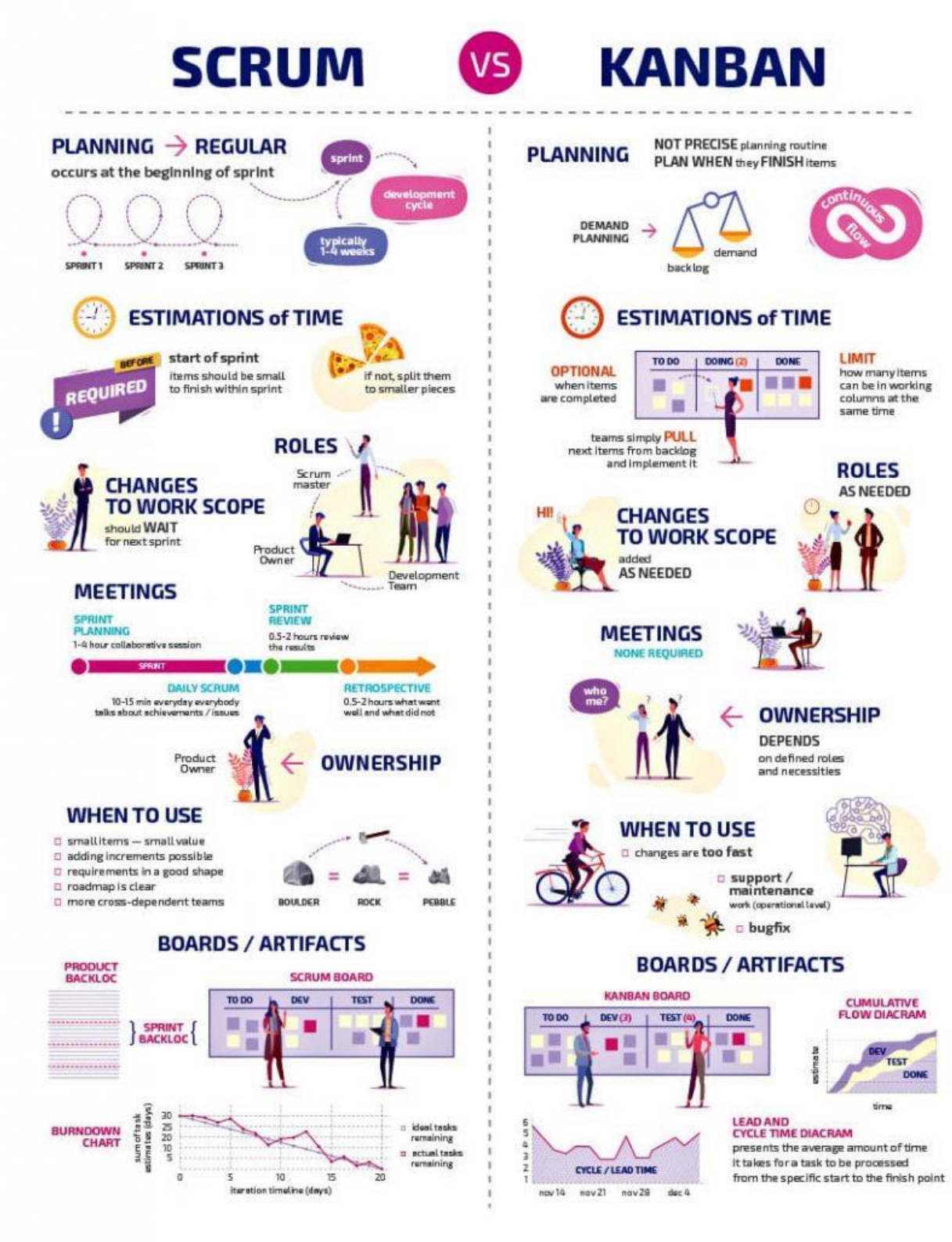


Рис. 2.9. Відмінності між Scrum та Kanban

Життєвий цикл команди послідовно проходить декілька етапів, які змінюють один одний:

– формування команди (постановка цілей, розподілення ролей в команді);

– притирання (осмислення цілей, визначення спільного вектору руху);

– нормалізація (досягнення цілей внаслідок компромісів та налагодження комунікації);

– функціонування (збільшення продуктивності за рахунок оптимізації процесів розробки і самоорганізації членів команди);

– розформування команди (отримання результатів, завершення проєкту).

Провідні компанії спочатку визначаються з методологією впровадження проєкту, а вже потім створюють ідеальну команду.

В управлінні проєктами значну роль відіграють застосовані підходи до того чи іншого процесу: жорсткі (орієнтовані на задачу), гнучкі (орієнтовані на команду) та незалежні команди з високим ступенем незалежності.

Методи, за якими працюють команди IT-проєктів різнопланові: Scrum, Kanban, XP, Lean та інші.

Етап ініціалізації проєкту

Ініціація проєкту (Project Initiating) – це стадія процесу управління проєктом, результатом якої є санкціонування початку проєкту.

Ініціація проєкту включає наступні завдання і процедури:

1. Розробка концепції проєкту:

1.1 Аналіз проблеми і потреби в проєкті.

1.2 Збір вихідних даних.

1.3 Визначення цілей і завдань проєкту.

1.4 Розробка концепції за окремими функціями управління проєкту:

1.4.1 Предметна область (розгляд існуючих варіантів).

1.4.2. Час проєкту (вибір методів і визначення процедур управління проєктом, по тимчасових параметрах, вибір програмного забезпечення для календарного планування, визначення тимчасових обмежень, розробка укрупненого календарного плану здійснення проєкту, визначення вимог до системи управління проєктом по часових параметрів).

1.4.3 Вартість (вироблення стратегії управління вартістю і фінансами проєкту (визначення цілей і завдань, критеріїв успіху і невдач, обмежень і припущень, проведення економічного аналізу та обґрунтування проєкту, проведення маркетингу, оцінка вартості та джерел фінансування, розробка прогнозу виконання, загальна економічна оцінка проєкту, розробка укрупненого графіка

фінансування, визначення вимог до системи управління вартістю та фінансуванням в проєкті).

1.4.4 Якість проєкту (вироблення стратегії управління якістю в проєкті, визначення загальних вимог і принципів забезпечення якості, вимоги до системи управління якістю).

1.4.5 Ризики (визначення цілей управління ризиками в проєкті, аналіз факторів ризику і невизначеності, визначення можливих джерел ризиків, вибір стратегії управління ризиками в проєкті, аналіз альтернатив, визначення вимог до системи управління ризиками).

1.4.6 Персонал (вироблення стратегії управління персоналом, визначення потреби в трудових ресурсах проєкту, визначення структури і функцій команди проєкту, формування життєвого циклу команди, аналіз можливостей забезпечення проєкту потрібними фахівцями, визначення вимог до управління персоналом).

1.4.7 Комунікації (визначення учасників проєкту, визначення базової документації проєкту, визначення вимог до комунікацій, обґрунтування і вибір комунікаційних технологій для управління проєктом).

1.4.8 Контракти (проведення маркетингу ринку продуктів і послуг, розробка стратегії управління контрактами, складання специфікації продуктів і послуг, визначення можливих джерел придбання ресурсів).

1.4.9 Зміни (вироблення стратегії управління змінами, аналіз можливих змін, визначення принципів інтеграції процесів управління змінами).

2. Розгляд та затвердження концепції.

3. Ініціювання проєкту (фактичне):

3.1 Ухвалення рішення про початок проєкту.

3.2 Визначення та призначення керуючого проєктом.

3.3 Ухвалення рішення про забезпечення ресурсами виконання проєкту.

Етап ініціювання проєкту характеризується великим ступенем **невизначеності** вихідних і результуючих даних, можливістю їх зміни і обмеженим часом для прийняття рішення.

Методи генерації ідей

Метод мозкового штурму (мозковий штурм, мозкова атака, англ. brainstorming) – метод вирішення завдань, в якому учасники обговорення генерують максимальну кількість ідей вирішення завдання, у тому числі найфантастичніші. Потім з отриманих варіантів вибирають найкращі рішення, які можна використовувати практично.

Правильний мозковий штурм включає 3 етапи:

Попередній етап – постановка проблеми. На цьому етапі чітко формулюють завдання, відбирають учасників штурму, визначають ведучого та розподіляють інші ролі учасників залежно від завдання та обраного способу проведення штурму.

Основний етап – генерація ідей. На цьому етапі генерують варіанти розв'язання задачі. Для максимальної ефективності в процесі генерації важливо дотримуватись кількох правил:

- Головне – кількість ідей. Не робіть жодних обмежень.
- Повна заборона на критику та будь-яку оцінку ідей, включаючи позитивну, оскільки оцінка відволікає від основного завдання та збиває ритм роботи та творчий настрій.
- Незвичайні і навіть абсурдні ідеї бажані.
- Комбінуйте та покращуйте будь-які ідеї.

Експертний етап – угруповання, відбір та оцінка ідей. На цьому етапі хаотичні ідеї класифікують, аналізують та оцінюють. Цей етап дозволяє виділити найцінніші ідеї та дати остаточний результат мозкового штурму. Якість експертного етапу безпосередньо залежить від суворості та одноманітності критеріїв відбору ідей учасників. Часто цей етап пропускають і учасники просто вибирають варіант, що сподобався.

Фрірайтинг (вільний лист, англ. free writing) – техніка і методика листа, що допомагає знайти неординарні рішення та ідеї, подібна до методу мозкового штурму. Це механічне записування всіх думок, що виникають у голові, протягом певного часу (зазвичай 10-20 хвилин). Текст пишуть без редагування, змін, не турбуючись про граматику чи стиль.

Шість основних правил:

1. Не докладайте надзусиль замість того, щоб підходити до письмового столу зі стиснутими зубами, вимагаючи від себе негайних віртуозних рішень, розслабтеся і спробуйте постаратися на 90%.
2. Пишіть швидко і безперервно, коли ви пишете безперервно, ви тим самим відтискаєте схиблену на редагуванні сторону свого розуму на

другий план, щоб сторона, що генерує ідеї, могла безперешкодно видавати слова.

3. Працюйте в жорстких часових рамках – це добре мотивує. Письмо протягом коротких, обмежених проміжків часу (зазвичай від 10 до 20 хвилин) змушує розум зосередитися.

4. Пишіть так, як ви вважаєте, фрірайтінг — це не письменство в чистому вигляді, це спосіб відстеження свого розумового процесу. Пишіть собі.

5. Розвивайте думку. Розвиваючи думку, ви припускаєте, що певне твердження вірне і робите ряд логічних кроків, відштовхуючись від нього. Іноді це призводить до дивовижних висновків.

6. Переорієнтуйте свою увагу, ставте собі прості питання у письмовій формі, це допомагає переорієнтувати розум на недосліджені елементи ситуації.

Морфологічний аналіз заснований на побудові таблиці, в якій перераховуються всі основні елементи, складові об'єкту і вказується велике число відомих варіантів реалізації цих елементів. Комбінуючи варіанти реалізації елементів об'єкта, можна отримати найнесподіваніші нові рішення.

Особливості даної методики:

1) слід максимально точно формулювати цілі морфологічного дослідження;

2) всі параметри повинні бути приблизно рівнозначними з погляду поставленої мети;

3) ніяких оцінок варіантів не слід проводити до повного оформлення морфологічної множини;

4) після визначення повного обсягу морфологічної множини варто перевірити, чи не можна морфологічну таблицю розділити на 2-3 блоки, які можна було б аналізувати не відразу, а послідовно, один за одним.

Суть методу морфологічного аналізу полягає в тому, що поставлена проблема розділяється учасниками процесу на більш дрібні етапи, які в подальшому будуть проаналізовані і оцінені окремо один від одного. У процесі аналізу складаються всі можливі варіанти поєднань ймовірних властивостей і принципів дій, а потім для кожного складається відповідний проєкт.

Приклади:

Маркетплейс репетиторських послуг					
	Комерціалізація	Аудиторія	Складність	Візуалізація	Супровід
1	ставка	учні дошкільного віку	примітивний інтерфейс	статична	консультант (методист)
2	відсоток	учні дошкільного +шкільного віку	зрозуміла ієрархія	динамічна	розділ автоматичних підказок
3	ставка+відсоток	молодь	доступна ієрархія	доповнена реальність	консультант онлайн (data mining)
4	ставка+відсоток+премії за рейтинг	старше покоління	багато розгалужень	віртуальні кімнати	консультант онлайн (artificial intelligence)

Рис. 2.10. Приклад морфологічного ящика

Метод «Шість капелюхів» (Капелюхи де Боно) було запропоновано британським письменником, психологом та спеціалістом з творчого мислення Едвардом де Боно у 1985 році. Сенс цієї методики полягає в тому, щоб дослідити будь-яку ідею з 6 незалежних одна від одної точок зору.

Висловлюється 6 точок зору на ідею:

Білий капелюх: точка зору інформаційної забезпеченості – відомі факти, інформація, якої не вистачає та з яких джерел їх можна отримати (аналітичне мислення).

Червоний капелюх: емоційні оцінки, відчуття та інтуїція (емоційне мислення).

Чорний капелюх: критика, наявність недоліків, ризики, впровадження (критичне мислення).

Жовтий капелюх: пошук переваг ідеї, конкурентоспроможність та оптимістичний прогноз (оптимістичне мислення).

Зелений капелюх: пошук альтернатив, модифікація вже наявних рішень (творче мислення).

Синій капелюх: підбиття підсумків, прийняття рішень до ідеї (рефлексія, підсумок).

Цю методику генерації ідей можна застосовувати як в групі, так і самостійно, «приміряючи кожен капелюх на себе».

Ментальні карти – це спосіб викладення інформації в формі дерева, серцевиною якого є ідея, можливість або проблема. Від центру розгалужуються нові гілки, які містять образи, нові ідеї, обмеження, можливості та ін.

Методика побудована на взаємодії лівої та правої півкулі мозку. Наявність графічних знаків та символів активує ліву півкулю мозку, а образні картини та кольори – праву півкулю. Ментальна карта є результатом процесу осмислення та структурування інформації про ідею у візуальній формі. Автором методики є відомий психолог Тоні Бьюзен, який вивчав способи мислення найвідоміших вчених світу. Ця методика може використовуватися як для індивідуальної, так і групової роботи з ідеями.

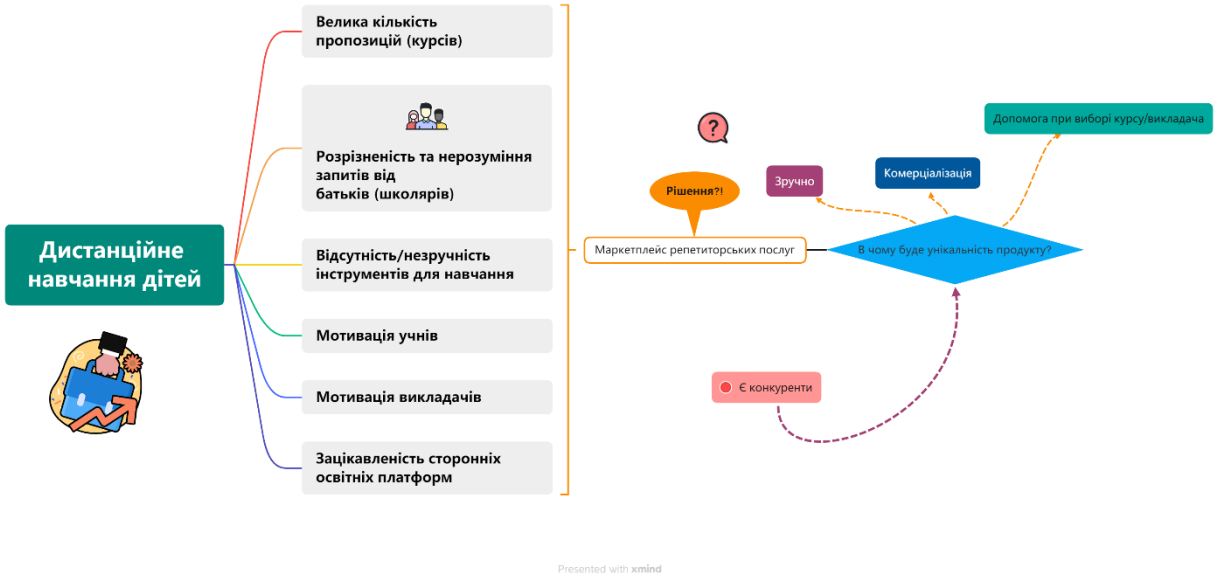


Рис. 2.10. Приклад ментальної карти

Цілі і завдання проєкту

Цілі проєкту (Project Objectives) – бажаний результат діяльності, що досягається в результаті успішного здійснення проєкту в заданих умовах його виконання.

При визначенні цілей проєкту можна обмежитися тільки завданням абстрактного бажаного результату. Необхідно знайти відповіді на наступні питання:

– як в точності повинен виглядати результат проєкту? (якісні і кількісні характеристики результату проєкту);

– які умови повинні враховуватися при реалізації проєкту? (вимоги і обмеження).

Цілі проєкту описують весь спектр основних питань, пов'язаних з проєктом, наприклад, технічні, фінансові та організаційні аспекти, питання, пов'язані з якістю, безпекою, людськими ресурсами, поставками,

інформаційними системами і технологіями. Вони складаються з трьох основних показників: **результати** (продукція і послуги необхідної якості), **час** (тривалість і конкретні дати) і **витрати** (людино-години і витрати).

Для кожного проєкту може бути побудовано безліч взаємопов'язаних цілей, які повинні бути чітко визначені.

Процеси визначення цілей і завдань:

1) формулювання;
2) структурування;
3) узгодження;
4) фіксація.

Ціль визначає бажаний кінцевий результат, якого потрібно досягти. Відповідно, завдання – це діяльність, яка виконується для досягнення цього результату.

Правильна постановка цілей – це радше мистецтво, аніж наука. Незрозумілий текст, відсутність часових рамок та критеріїв, яким слід відповідати, – найпоширеніші помилки у формулюванні цілей проєкту. Для запобігання цим та іншим помилкам при правильному плануванні проєкту застосовують SMART-аналіз (рис. 2.12) чи CLEAR-аналіз.

Система встановлення цілей SMART дає можливість узагальнити всю наявну інформацію на початковому етапі проєктної діяльності, встановити прийнятні терміни, визначити достатність ресурсів та надати всім учасникам процесу чіткі, точні та конкретні завдання. Назва аналізу базується на англомовних термінах, що формують цю абревіатуру і мають конкретне смислове навантаження:

Specific, конкретність. Характеризується чітко визначеними цілями, що збільшує ймовірність досягнення мети.

Measurable, вимірюваність. Обумовлюється наявними критеріями або метриками для вимірювання процесу досягнення цілей, а саме такими показниками, як якість, кількість та ціна.

Achievable, досяжність. Розумні цілі повинні бути досяжними, оскільки вони впливають на реальність завдання для мотивації підрядника. Якщо мета недосяжна – ймовірність її досягнення дорівнюватиме нулю.

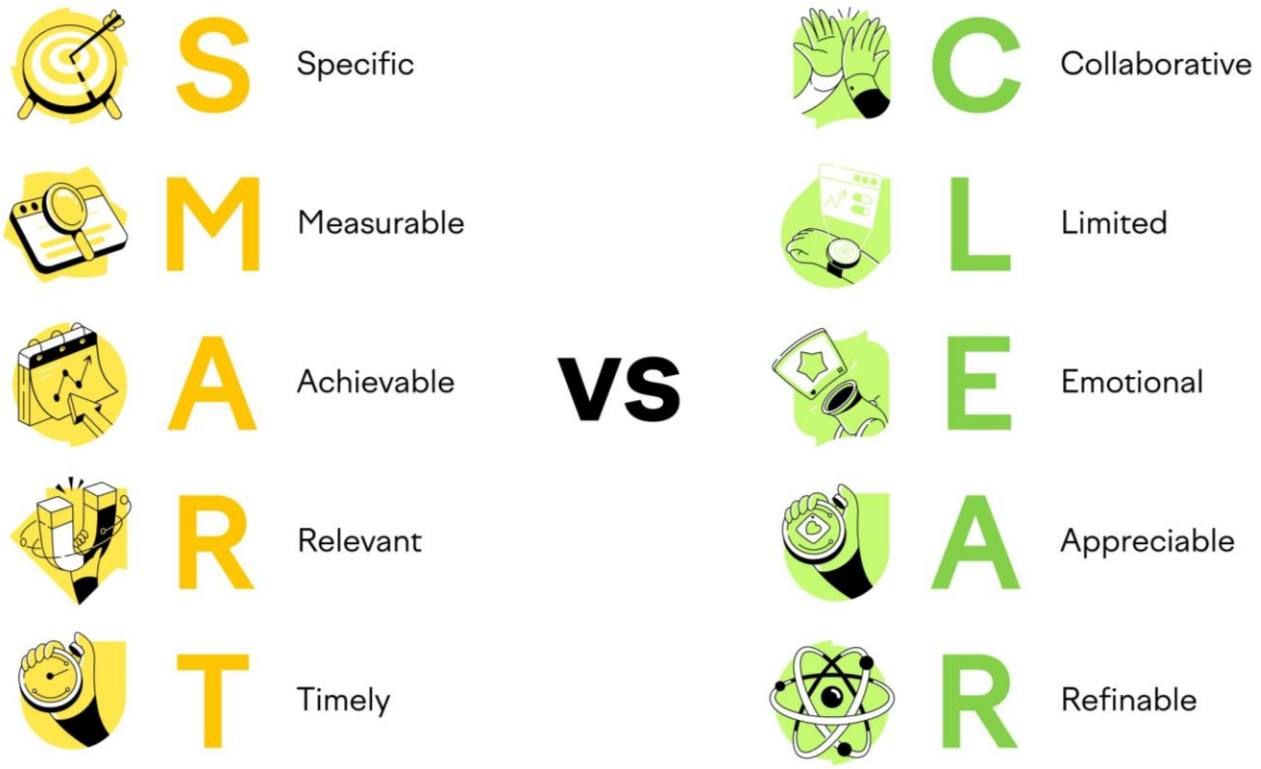


Рис. 2.12. Сучасна постановка проєктних цілей

Relevant, актуальність. Щоб визначити актуальність мети, важливо зрозуміти, як вирішення конкретного завдання сприятиме досягненню глобальних стратегічних цілей компанії. При встановленні поточної мети виникає питання: які переваги матиме компанія? Якщо суспільство загалом не виграє від цієї мети – ця ціль вважається непотрібною і марною тратою ресурсів суспільства.

Time-bound, обмеженість у часі. Ціль для SMART повинна бути обмежена в часі, що встановлює кінцевий термін, перевищення якого вказує на те, що мета не досягнута; у протилежному випадку марно й говорити про досяжність – це може засвідчувати, що певна мета теоретично досяжна, проте неможливо вказати часові рамки, у які її вдасться досягнути. Встановлення часових рамок та меж реалізації мети допомагає керувати процесом управління. У такому разі часові рамки мають установлюватися з урахуванням своєчасного досягнення цілі.

Система CLEAR:

Collaborative, спільний – забезпечення взаємодії учасників команди.

Limited, обмежений – обмеження в часі та обсязі.

Emotional, емоційний – спонукання, мотивація учасників в команди до дії.

Appreciable, важливий, помітний – виділення (поділ) цілей на першочергові та другорядні.

Refinable, той, що удосконалюється – цілі повинні бути гнучкими.

Приклад опису продукту на етапі ініціалізації

ПРОЄКТ «ОНЛАЙН-ОСВІТА (ПОРТАЛ РЕПЕТИТОРСЬКИХ ПОСЛУГ)»

1. ОГЛЯД

Продукт — це маркетплейс приватних послуг додаткової освіти дітей.

Основні цінності продукту:

1) Для педагогів:

- комерціалізація своїх професійних компетенцій;
- інтелектуальний підбір актуальних напрямів освітніх послуг;
- сучасне цифрове середовище для організації та проведення занять;
- розмір комісії, що стягується з батьківської плати, в залежності від якості виконуваних послуг (додаткова мотивація педагога).

2) Для учнів та його батьків:

- вузькоспеціалізований сервіс пошуку послуг за найчастішими запитами: підготовка до школи, до іспитів, до олімпіад, поглиблене вивчення шкільних предметів, що цікавлять дитину, тощо;
- інтелектуальний підбір педагога, що найбільше підходить конкретному учню та його запитам, потребам та індивідуально-психологічним і фізичним особливостям;
- додатковий контроль та персональний супровід того, хто навчається особистим цифровим помічником (асистентом) на основі технологій штучного інтелекту;
- система регулярного зворотного зв'язку батькам про динаміку процесу навчання, проблеми та ризики, зібраної на основі спілкування цифрового помічника з дитиною.

3) Для юридичних осіб та індивідуальних підприємців, які надають платні освітні послуги:

- база потенційних клієнтів у складі учнів та його батьків;
- база потенційних кадрів із числа педагогів, які працюють із платформою;
- унікальні сервіси та інструменти, що дозволяють зробити і освітній процес, і взаємини з клієнтами максимально комфортними та зручними.

2. ОСНОВНИЙ ПРОФІЛЬ СПОЖИВАЧА, НА ЯКОГО ОРІЄНТОВАНИЙ ПРОДУКТ

Потенційними клієнтами продукту є:

– педагоги, які будуть пропонувати власні послуги;

– батьки дітей дошкільного та шкільного віку, зацікавлені в отриманні платних послуг з додаткової освіти своїх дітей;

– юридичні особи та індивідуальні підприємці, які надають платні освітні послуги та зацікавлені у додатковому залученні нових клієнтів з числа батьків, у підборі нових педагогів для реалізації своїх послуг на ринку.

3. АКТУАЛЬНІСТЬ

Тенденція до ускладнення та перенасичення шкільних програм, а також пандемія, військовий стан та використання дистанційного навчання стають причиною зростаючої потреби батьків до залучення за додаткову оплату сторонньої педагогічної допомоги (педагогів додаткової освіти, репетиторів) до процесу навчання своїх дітей. Подібні послуги також актуальні для учнів з обмеженими можливостями здоров'я.

Ринок приватних занять із педагогом (індивідуальних чи групових) активно розвивається, існують ресурси та сервіси у мережі «Інтернет» для пошуку педагогів. Проте характерними проблемами для цієї галузі є відносно невисокі гарантії якості послуг, що надаються при підборі педагога, відсутність підприємницьких компетенцій у педагогів за родом їх діяльності і, як наслідок, незнання ними потенційних можливостей та затребуваних напрямків для комерціалізації своїх послуг. Негативна реакція батьківського співтовариства на дистанційне навчання пов’язана з навантаженням на сім'ю, від якої вимагається тотальний контроль за процесом навчання дитини.

4. СТРАТЕГІЧНА ЦІЛЬ/БАЧЕННЯ (перше наближення)

Завоювати за рік після релізу не менш ніж 50% цільової аудиторії від рівня провідного маркетплейсу за рахунок впровадження:

1) інтелектуального підбору найбільш відповідного педагога – нейромережа аналізує запит учня (його батька) та зіставляє з базою знань щодо компетенцій педагогів і відповідей інших учнів (наприклад, переваги в каналах отримання інформації (аудіо, відео, текст), рівень активності дитини, її завантаженість, а також завантаженість педагога);

2) технології скорингу – система накопичує дані про зворотний зв'язок якості роботи педагога (наприклад, кількість відмов від його послуг, кількість зірваних уроків, оцінки клієнтів тощо). Скоринговий бал впливає на розмір комісії, яка вираховується з батьківської плати за послугу – чим він вищий (і отже, вища якість роботи педагога), тим нижча комісія та вища

частина оплати, яку отримає педагог на руки. Мотивує педагогів до якісного виконання своїх зобов'язань;

3) зручного цифрового середовища для організації та проведення занять (вебінарний майданчик, можливість завантаження власних навчальних матеріалів, автоматичне складання розкладу для педагога та учня);

4) особистого цифрового помічника для учня з урахуванням технології штучного інтелекту. Помічник супроводжує дитину протягом усього навчання та дозволяє автоматизувати функцію батьківського контролю в частині перевірки виконання завдань, підключення до уроку вчасно, організації навчального часу. Крім того, помічник виконує функцію збору зворотного зв'язку від учня для батька та педагога, яка відбувається у форматі бесіди з учнем про наявність можливих складнощів та проблем у навчанні.

5. СТАТУС БІЗНЕСУ

Цей бізнес є стартапом, який зараз проходить етап дослідження ринку.

6. КОНКУРЕНЦІЯ

Наявність аналогів: https://buki.com.ua/

7. БІЗНЕС-ПОТРЕБИ

Інтелектуальні (розробка алгоритмів елементів штучного інтелекту для прийняття рішень).

Трудові (для розробки програмного забезпечення).

Документ про концепцію та межі продукту

Структура документу:

1. Бізнес-вимоги:

1.1 Вихідні дані (Обгрунтування нового продукту чи змін в існуючому. Передумови прийняття рішення про створення продукту).

1.2 Можливості бізнеса (Опис бізнес-задачі чи бізнес-процесу, для яких створюється продукт. Опис середовища використання системи. Опис ринкових можливостей (для комерційного продукту). Порівняльна характеристика з аналогами, переваги продукту. Потреби типових клієнтів. Варіанти використання продукту. Критичні відомі вимоги до якості та інтерфейсу (загальні, при наявності)).

1.3 Бізнес-цілі (Переваги бізнесу від продукту в кількісному та вимірюваному вигляді. Цілі можуть бути фінансовими та нефінансовими.

Бізнес-мета – це спосіб вимірювання досягнення орієнтирів, які дозволяють вирішити існуючу бізнес-проблему).

1.4 Критерії успіху (Фактори, що впливають на успіх проєкту (внутрішні та зовнішні). Критерії успіху повинні оцінювати те, що важливо для бізнеса, а не те, що легко оцінити).

1.5 Положення про концепцію проєкту (***Для*** (цільова аудиторія). ***Який*** (потреби чи можливості). ***Ця/цей*** (назва продукту). ***Є*** (категорія продукту). ***Який*** (основні функції, переваги, основна причина для придбання). ***На відміну від*** (основна перевага від конкурента, існуючої системи чи бізнес-процесу). ***Наш продукт*** (головна відмінність нового продукту)).

1.6 Бізнес-ризики (Ринкова конкуренція. Часові обмеження. Прийнятність для користувачів. Проблеми, пов'язані з реалізацією. Можливі негативні фактори, що впливають на бізнес. Кількісна оцінка ризиків та ймовірності їх настання. Можливі дії щодо пом'якшення ситуації).

1.7 Припущення та залежності (Гіпотези (припущення) щодо кількісних параметрів цілей (чому так?). Залежності від зовнішніх факторів (якщо … то). Припущення та залежності можуть трансформуватися у бізнес-ризики).

2. Межі та обмеження проєкту:

2.1 Основні функції (Опис основних функцій продукту чи можливостей користувачів. Що відрізняє продукт від попередньої версії. Ідентифікація функцій (ID)).

2.2 Обсяг першої версії (Узагальнені основні заплановані функції (користувацькі історії, варіанти використання тощо)).

2.3 Обсяг наступних версій (Перелік відкладених функцій та бажані терміни наступних релізів).

2.4 Обмеження та виключення (Перелік всіх можливостей чи характеристик, які можуть очікувати зацікавлені сторони, але включення яких не заплановано (це нагадує про межі проєкту)).

3. Бізнес-контекст

3.1 Профілі зацікавлених осіб (Групи клієнтів. Цільові ринкові сегменти. Різноманітні класи користувачів, що входять до цих сегментів. Структурна схема чи список осіб).

3.2 Пріоритети проєкту (5 вимірів: функції (обсяг), якість, графік, витрати, кадри. 3 категорії: обмеження, провідний фактор, ступінь вільності).

3.3 Особливості розгортання (Інформація та дії, необхідні для ефективного розгортання рішення в робоче середовище. Опис доступу (коли, в який час). Зміни (при необхідності) інфраструктури, доступу до сховища даних).

Для представлення меж проєкту можуть використовуватися:

Контекстна діаграма (найзагальніший опис системи), рис. 2.13.

Карта екосистеми (демонстрація всіх систем, які пов'язані з продуктом із вказанням природи взаємодії між ними), рис. 2.14.

Дерево функцій (перелік функцій, що об'єднані в логічні групи з ієрархічним розбиттям кожної функції на більш дрібні), рис. 2.15.

Перелік подій (зовнішні події, які можуть ініціювати певну поведінку в системі). Перелік подій (even list) визначає межі системи шляхом перелічення можливих бізнес-подій, що ініційовані користувачем/часом/обладнанням, відображає реакцію системи на події у вигляді функціональних вимог відображаються у SRS.

Класи користувачів (рис. 2.16) – це не обов'язково люди. Це можуть бути програмні агенти, пошукові системи, сканери. Опосередкований користувач може працювати з даними та сервісами продукту, не звертаючись до нього безпосередньо (через інші додатки чи звіти).

Профілі зацікавлених осіб (stakeholders) містять:

1. Основна цінність чи перевага для зацікавленої особи:

– підвищення продуктивності;

– менше виправлень;

– зниження собівартості;

– прискорення бізнес-процесів;

– автоматизація ручних задач;

– можливість виконувати нові задачі;

– відповідність стандартам та правилам;

– зручність та легкість використання у порівнянні з існуючою системою.

2. Ставлення до продукту.

3. Найважливіші функції та характеристики.

4. Всі відомі обмеження, яких необхідно дотримуватись.

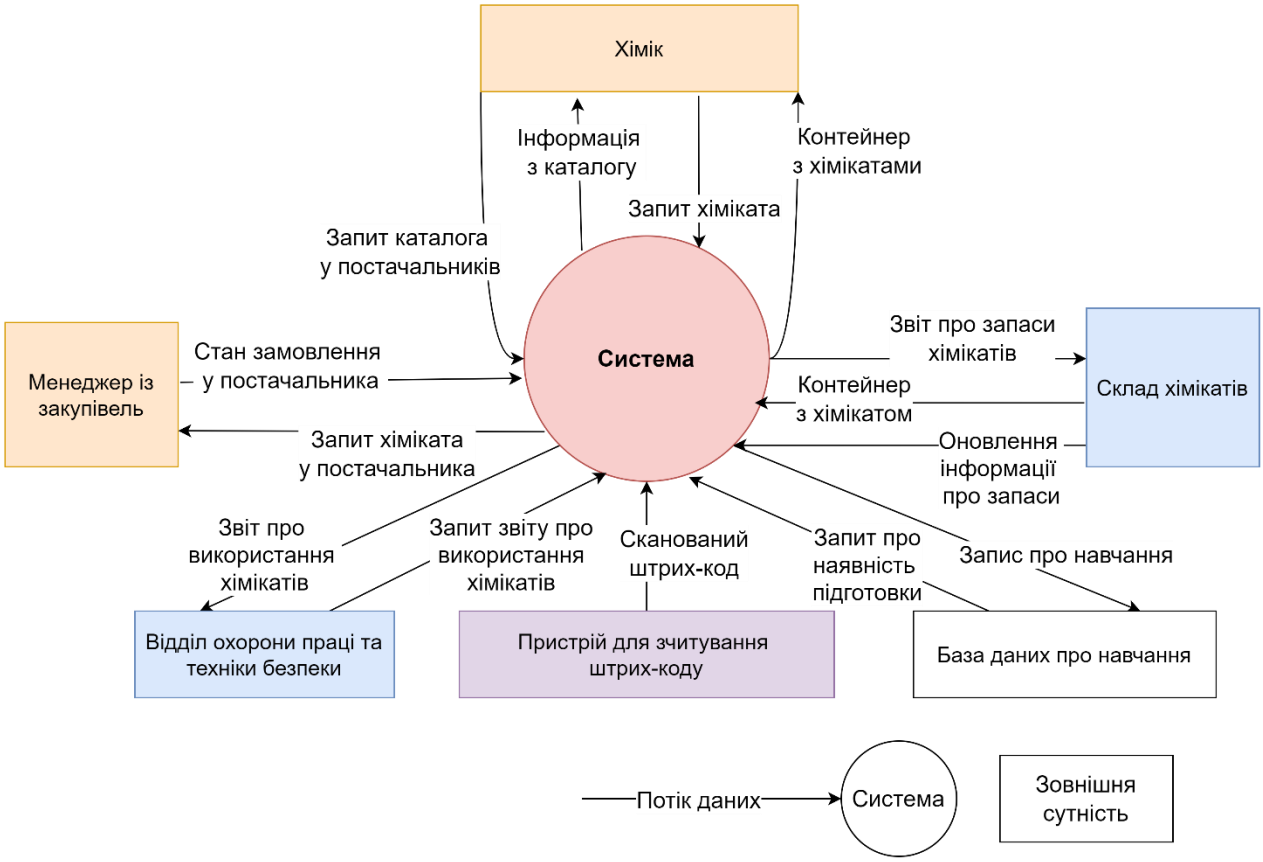


Рис. 2.13. Приклад контекстної діаграми (context diagram)

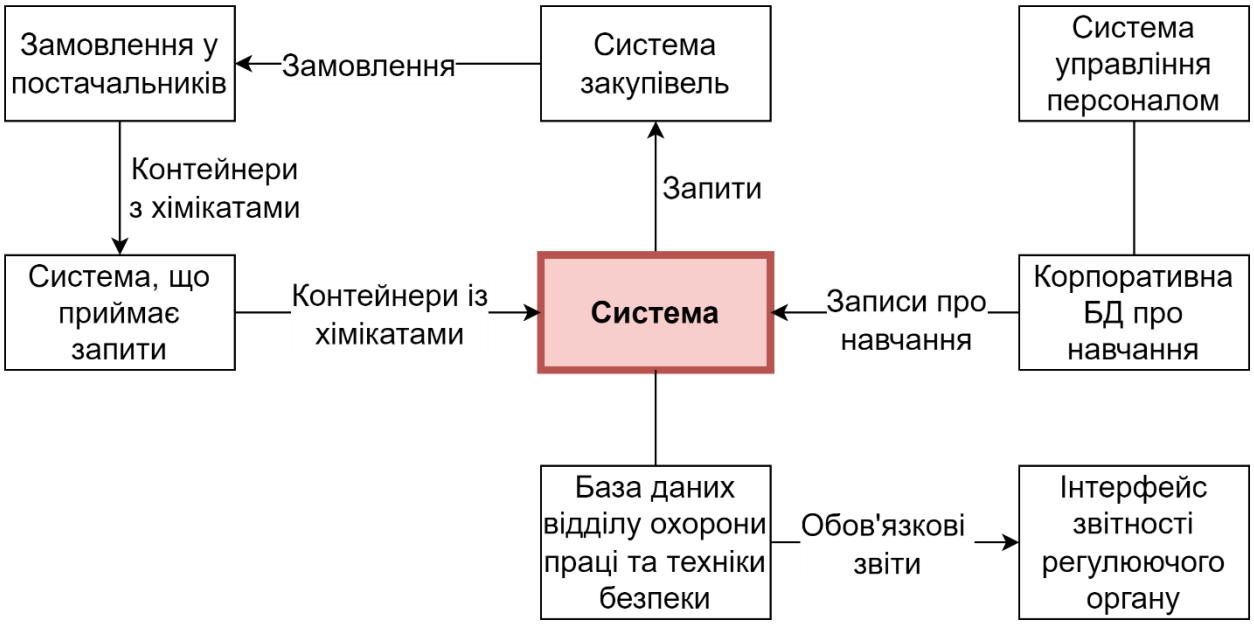


Рис. 2.14. Приклад карти екосистеми (ecosystem map)

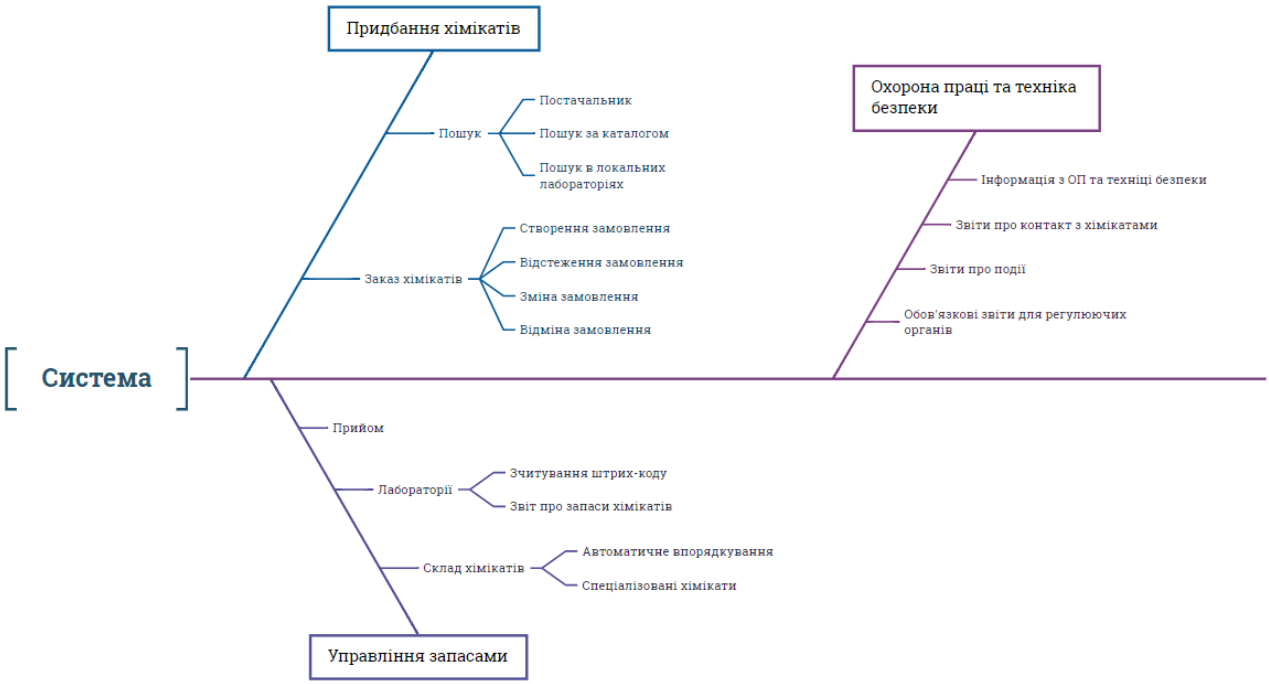


Рис. 2.15. Приклад дерева функцій (feature tree)

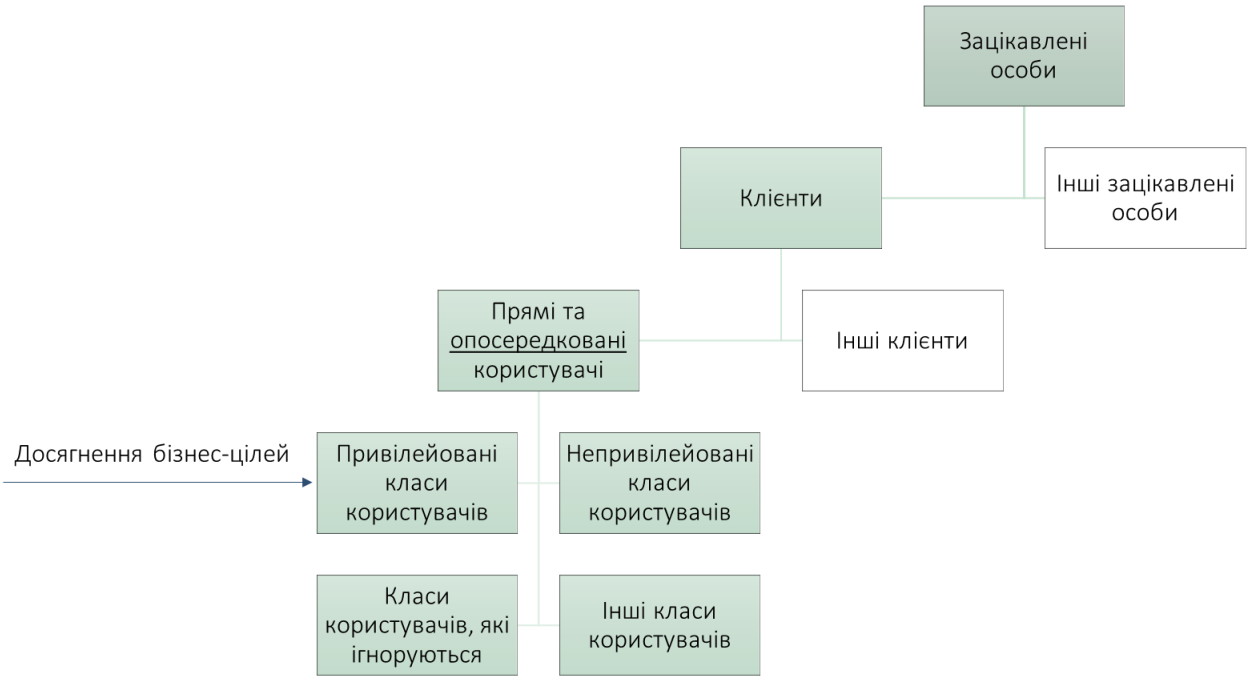


Рис. 2.16. Класи користувачів

Приклад профілю (система замовлення співробітниками їжі з кафетерію) за К. Вігерсом та Д. Бітті.

Таблиця 2.2. Профілі зацікавлених сторін

Зацікавлена особа	Основна цінність	Ставлення	Основні інтереси	Обмеження
Керівництво компанії	Зростання продуктивності праці співробітників, скорочення витрат	Сильна підтримка	Економія витрат від втрати продуктів повинна перевищити витрати на ПЗ	Не визначені
Співробітники кафетерію	Більш ефективне використання робочого часу співробітників протягом дня, більше задоволення клієнтів	Нормальне сприйняття	Збереження робочих місць	Необхідність навчання співробітників
Постійні клієнти кафетерію	Кращій вибір страв, економія часу, зручність	Зацікавлені, але можуть використовувати систему не так часто як очікується (соціальний фактор спілкування)	Простота використання, надійність доставки, можливість вибору страв	Необхідність доступу до корпоративної мережі чи наявність мобільного пристрою
Відділ розрахунку заробітної плати	Немає вигоди. Необхідно розробити схему утримання вартості замовлень з заробітної плати	Не дуже щасливі приймати участь при розробці ПЗ, однак розуміють цінність для компанії та співробітників	Мінімум змін в існуючому ПЗ для розрахунку заробітної плати	Ще не виділено ніяких ресурсів на зміну ПЗ
Менеджери ресторанів	Зростання продажів, вихід на нові кластери ринку для залучення нових клієнтів	Підтримують, але з обережністю	Мінімум нових технологій, стурбованість ресурсами та витратами на доставку страв	Можуть не мати достатньо персоналу та можливостей для обробки потрібних обсягів замовлень. Не у всіх є меню в інтернеті

Також для опису користувачів можуть використовуватися архетипи (типові представники кожного класу користувачів) або персонажі.

Персонаж (Persona) – це архетип користувача, який представляє певну модель поведінки; це опис групи користувачів з їхніми очікуваннями, переживаннями, досвідом, потребами, втілений в одному фіктивному профілі.

Складові опису:

1. Опис користувача (демографічні та психографічні дані). Як правило, до нього входить інформація про вік, стать, сферу діяльності, цілі, мотивуючі та фруструючі фактори.

2. Опис середовища (контекст використання: де та коли відбувається взаємодія).

3. Опис завдань (які завдання виконує користувач, як часто і т.п.).

Приклад документу про концепцію та межі продукту

ПРОЄКТ «ОНЛАЙН-ОСВІТА (ПОРТАЛ РЕПЕТИТОРСЬКИХ ПОСЛУГ)»

1. Бізнес-вимоги:

1.1 Вихідні дані

Тенденція до ускладнення та перенасичення шкільних програм, а також пандемія, військовий стан та використання дистанційного навчання стають причиною зростаючої потреби батьків до залучення за додаткову оплату сторонньої педагогічної допомоги (педагогів додаткової освіти, репетиторів) до процесу навчання своїх дітей. Подібні послуги також актуальні для учнів з обмеженими можливостями здоров'я.

Ринок приватних занять із педагогом (індивідуальних чи групових) активно розвивається, існують ресурси та сервіси у мережі «Інтернет» для пошуку педагогів. Проте характерними проблемами для цієї галузі є відносно невисокі гарантії якості послуг, що надаються при підборі педагога, відсутність підприємницьких компетенцій у педагогів за родом їх діяльності і, як наслідок, незнання ними потенційних можливостей та затребуваних напрямків для комерціалізації своїх послуг. Негативна реакція батьківського співтовариства на дистанційне навчання пов’язана з навантаженням на сім'ю, від якої вимагається тотальний контроль за процесом навчання дитини. *При цьому треба враховувати, що батьки іноді також не можуть однозначно визначити за яким напрямом розвивати навички та знання власної дитини. Як батьки, так і учні «блукають» в мережі великої кількості курсів, освітніх програм, шкіл,*

ресурсів та інш. Поруч з цим, завжди треба ставити акцент на «сумісність» викладача та слухача (дитини), щоб забезпечити комфортні умови співпраці.

1.2 Можливості бізнеса

Вирішенням проблеми надання якісної та доступної освіти може стати маркетплейс приватних освітніх послуг для додаткової освіти **дітей дошкільного та шкільного віку**.

На українському ринку приватних освітніх послуг вже функціонує схожа платформа https://buki.com.ua/, яка має наступний функціонал: вибір напрямів послуг за каталогом, вибір міста для пошуку репетитора за довідником або формату онлайн, заповнення заявки слухачем, реєстрація викладача та розміщення ним інформації про себе, помічник (адміністратор), збір відгуків, онлайн-школа BUKI School (збір заявок).

Кількісні показники конкурента:

– 90 тис. репетиторів;

– 220 тис. учнів знайшли репетиторів;

– 115 тис. відгуків про роботу репетиторів.

Аналіз конкурента дозволив виявити фактори, які потенційно забезпечать конкурентоспроможність **нашого продукту**, це:

1) інтелектуальний підбір найбільш відповідного педагога – нейромережа аналізує запит учня (його батька) та зіставляє з базою знань щодо компетенцій педагогів і відповідей інших учнів (наприклад, переваги в каналах отримання інформації (аудіо, відео, текст), рівень активності дитини, її завантаженість, а також завантаженість педагога);

2) технологія скорингу – система накопичує дані про зворотний зв'язок якості роботи педагога (наприклад, кількість відмов від його послуг, кількість зірваних уроків, оцінки клієнтів тощо). Скоринговий бал впливає на розмір комісії, яка вираховується з батьківської плати за послугу – чим він вищий (і отже, вища якість роботи педагога), тим нижча комісія та вища частина оплати, яку отримає педагог на руки. Мотивує педагогів до якісного виконання своїх зобов'язань;

3) зручне цифрове середовище для організації та проведення занять (вебінарний майданчик, можливість завантаження власних навчальних матеріалів, автоматичне складання розкладу для педагога та учня);

4) особистий цифровий помічник для учня з урахуванням технології штучного інтелекту. Помічник супроводжує дитину протягом усього навчання та дозволяє автоматизувати функцію батьківського контролю в

частині перевірки виконання завдань, підключення до уроку вчасно, організації навчального часу. Крім того, помічник виконує функцію збору зворотного зв'язку від учня для батька та педагога, яка відбувається у форматі бесіди з учнем про наявність можливих складнощів та проблем у навчанні.

1.3 Бізнес-цілі

Завоювати за рік після релізу не менш ніж 50% цільової аудиторії від рівня провідного маркетплейсу (https://buki.com.ua/).

Забезпечити окупність створення маркетплейсу на рівні 30% протягом перших 6 місяців після релізу.

Забезпечити окупність створення маркетплейсу на рівні 100% протягом року після релізу.

1.4 Критерії успіху

Досягти в першій версії:

– кількість курсів – не менш ніж 100;

– кількість викладачів – не менш ніж 30.

Забезпечити задоволення користувачами отриманими на маркетплейсі послугами на рівні 3 бали (за шкалою від 1 до 6) протягом перших 6 місяців.

Забезпечити зростання рівня задоволеності на 0,5 балу кожні 3 місяці після перших 6 місяців.

1.5 Положення про концепцію проєкту (бачення продукту)

Для дітей та їх батьків, ***які*** бажають отримати приватні освітні послуги, а також ***для*** викладачів, ***які*** бажають комерціалізувати власний професійний досвід, ***даний*** продукт ***є*** веб-платформою, ***яка*** забезпечую єдину точку доступу до каталогу курсів, бази даних викладачів, освітніх матеріалів та технічних інструментів навчання. Продукт буде опитувати дітей та/чи батьків й формувати індивідуальний підхід (підбір викладача, рекомендації за курсами, методи навчання тощо) для слухача. Буде відстежувати результати навчання та формувати карту компетенцій дитини, робити підказки та супроводжувати протягом навчання. Продукт буде відстежувати відгуки щодо викладачів, формувати рейтинги викладачів, визначати/пропонувати умови комерціалізації послуг викладачів. Крім того продукт дає доступ до вебінарного майданчику, надає можливість завантаження власних навчальних матеріалів, автоматично складає розклад для педагога та учня. Цей продукт

дозволить охопити не менш ніж 50% цільової аудиторії конкурента через рік після релізу, бо орієнтований на вузьку аудиторію, врахує психологічні особливості дитини при організації навчання, надасть технічні інструменти навчання. ***На відміну*** від конкурента ***наш продукт*** буде генерувати розклад занять, формувати рекомендації педагогам щодо методів навчання, підтримувати мотивацію до навчання через роботу цифрового помічника.

1.6 Бізнес-ризики

Конкурент може впровадити цифрового помічника на кшталт запропонованого.

В школах відмінили дистанційне навчання, що призведе до зниження попиту на додаткові освітні послуги.

Замало кваліфікованих викладачів, охочих залучитися до платформи.

1.7 Припущення та залежності

Якщо буде недостатньо користувачів, необхідно буде проводити додаткові рекламні заходи.

Якщо викладачів не буде влаштовувати запропонована система комерціалізації, необхідно буде вирішувати питання в індивідуальному порядку.

2. Межі та обмеження проєкту:

2.1 Основні функції

FE-1. Вибір та оплата курсів з каталогу.

FE-2. Створення, перегляд, редагування курсів в каталозі.

FE-3. Створення, перегляд, редагування переліку викладачів.

FE-4. Призначення, редагування системи оплати праці викладачів.

FE-5. Заповнення анкети та формування профілю дитини.

FE-6. Розрахунок рейтингу викладачів.

FE-7. Створення цифрового помічника.

FE-8. Генерація розкладу занять.

FE-9. Перегляд досягнень дитини.

FE-10. Формування рекомендацій викладачу.

FE-11. Забезпечення доступу до технічних інструментів (вебінарний майданчик, інтерактивна дошка, завантаження/видалення файлів).

FE-12. Забезпечення доступу до платформи (інтернет, мобільний пристрій).

2.2 Обсяг першої та наступних версій

Функція	Версія 1.0	Версія 1.1
FE-1. Вибір та оплата курсів з каталогу.	Реалізована повністю.	–
FE-2. Створення, перегляд, редагування курсів в каталозі.	Реалізована повністю.	–
FE-3. Створення, перегляд, редагування переліку викладачів.	Реалізована повністю.	–
FE-4. Призначення, редагування системи оплати праці викладачів.	Призначення системи оплати.	Редагування системи оплати.
FE-5. Заповнення анкети та формування профілю дитини.	Реалізована повністю.	–
FE-6. Розрахунок рейтингу викладачів.	Не реалізована.	Реалізована повністю.
FE-7. Створення цифрового помічника.	Реалізована, якщо є час.	Реалізована повністю.
FE-8. Генерація розкладу занять.	Реалізована повністю.	–
FE-9. Перегляд досягнень дитини.	Не реалізована.	Реалізована повністю.
FE-10. Формування рекомендацій викладачу.	Реалізована, якщо є час.	Реалізована повністю.
FE-11. Забезпечення доступу до технічних інструментів (вебінарний майданчик, інтерактивна дошка, завантаження/видалення файлів).	Не реалізована.	Реалізована повністю.
FE-12. Забезпечення доступу до платформи (інтернет, мобільний пристрій).	Доступ через інтернет.	Додаток для мобільних пристроїв.

2.3 Обмеження та виключення

Відсутні.

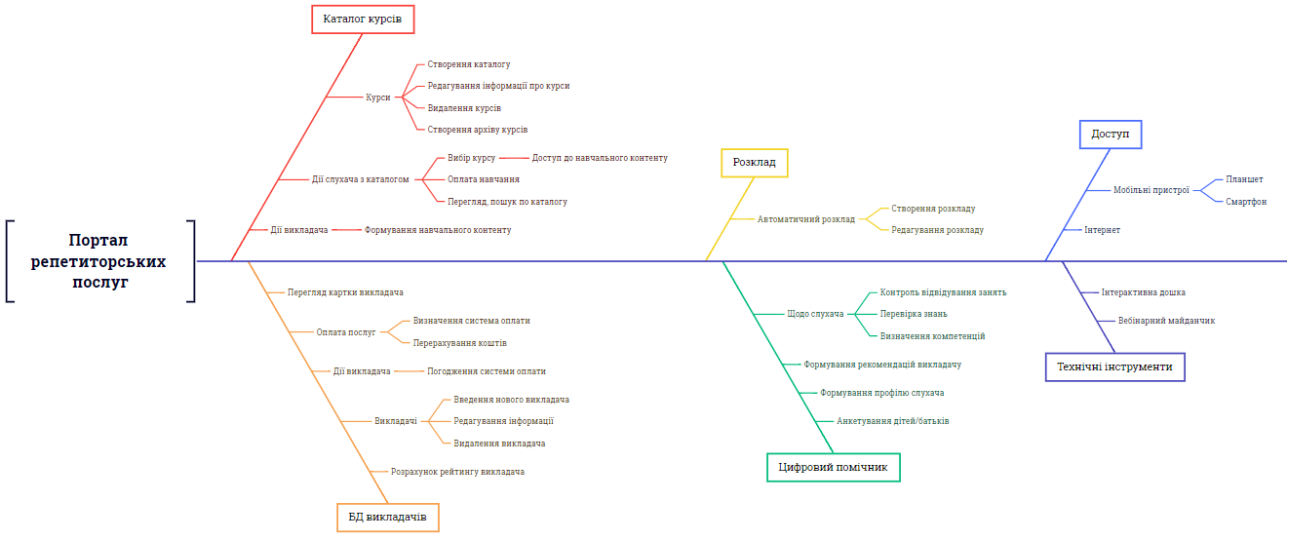


Рис. 2.13. Дерево функцій продукту «Портал репетиторських послуг»

3. Бізнес-контекст

3.1 Профілі зацікавлених осіб

Зацікавлена особа	Основна цінність	Ставлення	Основні інтереси	Обмеження
Власник стартапу	Забезпечити комерціалізацію проєкту	Максимальна зацікавленість	Рівень доходів повинен забезпечити покриття витрат на розробку, а далі – отримання прибутку	Не виявлені
Батьки/молодь	Економія часу з пошуком репетиторів; контроль за навчальним процесом	Достатньо висока зацікавленість. Можливі сумніви з приводу наявності альтернативи (конкурента)	Підбір репетитора за інтересами (простота, підказки)	Необхідність реєстрації; заповнення анкети
Діти (слухачі)	Отримання послуги з навчання; супровід	Середня зацікавленість. Можлива відсутність мотивації до навчання	Простота використання технічних інструментів; розклад занять та система нагадувань	Необхідність реєстрації; наявність стабільного інтернету

Зацікавлена особа	Основна цінність	Ставлення	Основні інтереси	Обмеження
Викладачі	Отримання фінансової винагороди за надані послуги; професійне визнання	Достатньо висока зацікавленість. Стурбованість щодо офіційного оформлення умов співпраці	Простота реєстрації, проведення занять, використання технічних інструментів	Необхідність реєстрації; наявність стабільного інтернету; навчання роботі з технічними інструментами

Persona 1. Тамара Петрівна (представник категорії Батьки), рис. 2.14.

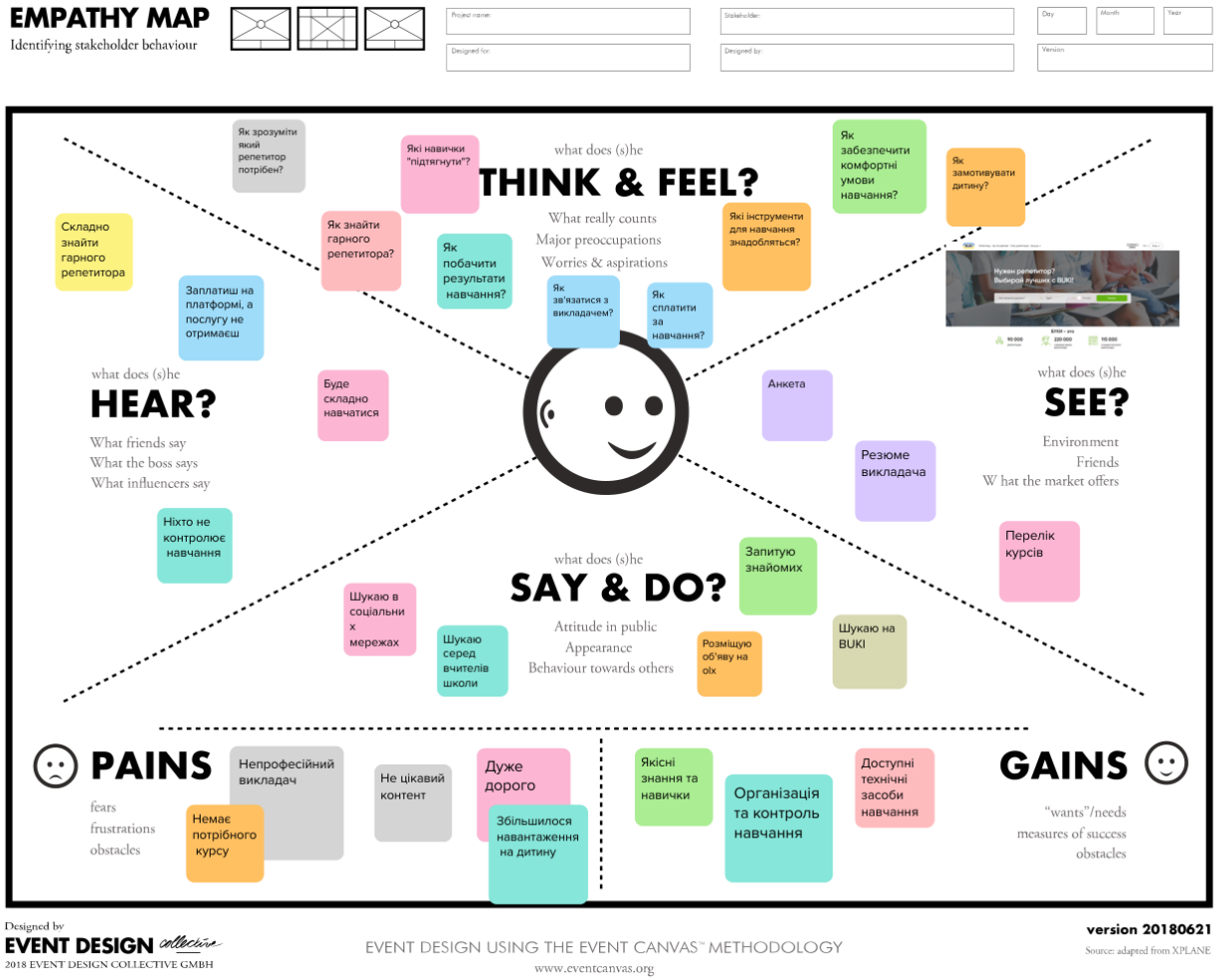


Рис. 2.14. Карта емпатії для представник категорії Батьки

Persona 2. Любов Іванівна (представник категорії Викладачі), рис. 2.15.

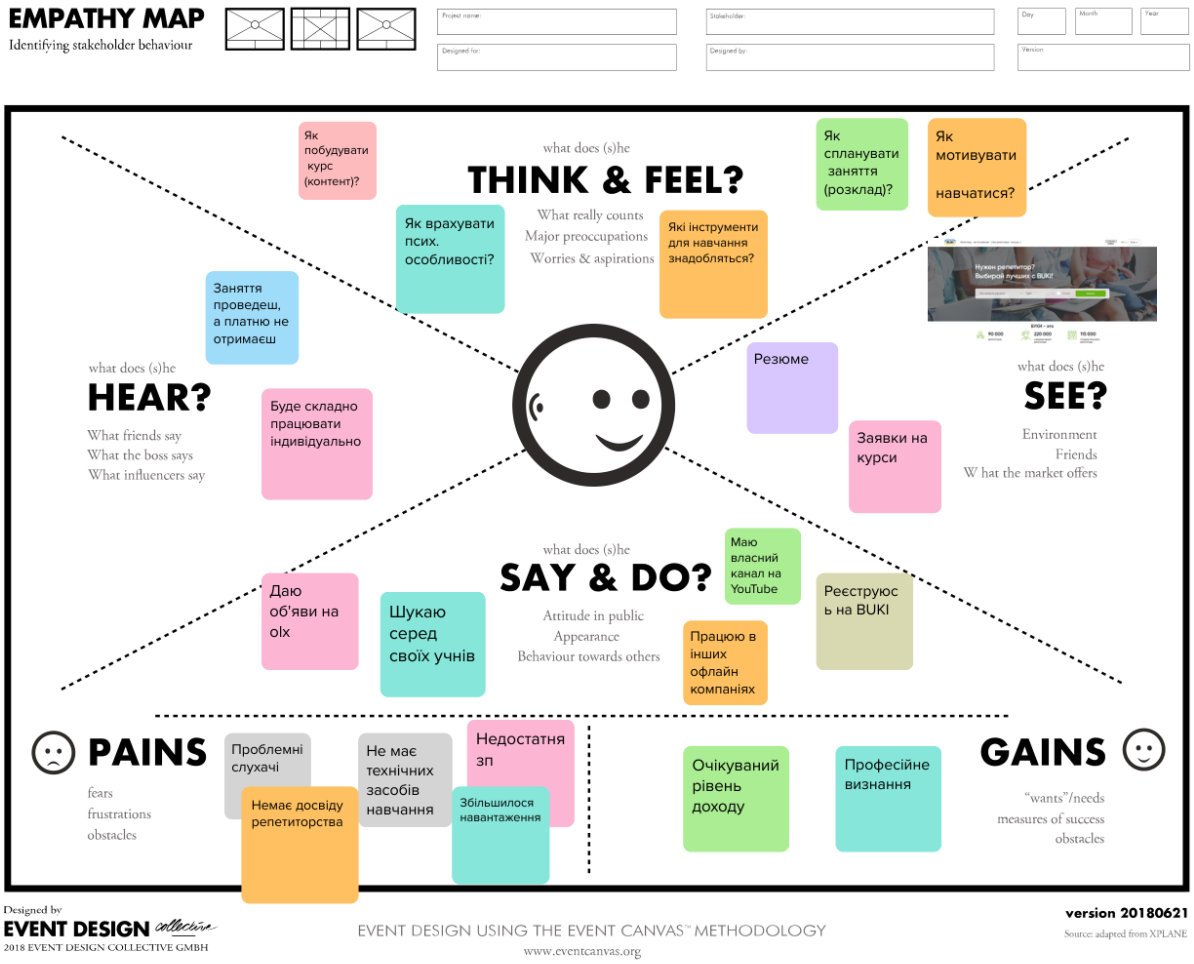


Рис. 2.15. Карта емпатії для представник категорії Викладачі

3.2 Пріоритети проєкту

Виміри	Обмеження	Рушійна сила	Ступінь вільності
Функції	Всі функції версії 1.0 повинні бути реалізовані.		
Якість	95% користувацьких тестів для перевірки повинні бути виконані; всі тести на захищеність повинні бути виконані		
Терміни			За планом реліз версії 1.0 через 6 місяців після початку розробки.

Виміри	Обмеження	Рушійна сила	Ступінь вільності
			Реліз версії 1.1 – через 3 місяці після релізу версії 1.0
Витрати			В межах 15% від перевищення бюджету проєкту без узгодження з донором
Персонал	Розробник (0,5 ставки) з побудови та навчання штучної нейромережі	Запланований склад команди: менеджер проєкту (0,5 ставки), 2 розробника, тестувальник (0,5 ставки)	

3.3 Особливості розгортання

Обрати веб-сервер.

В межах версії 1.1 розробити додатки для смартфонів та планшетів під ОС iOS та Android.

Розробити навчальні відеоролики для роботи з технічним інструментарієм тривалістю не більш ніж 2 хвилин.

ТЕМА 3. БІЗНЕС-АНАЛІЗ НА ЕТАПІ ВИЯВЛЕННЯ, ЗБОРУ ТА АНАЛІЗУ ВИМОГ

Методи виявлення вимог. Виявлення вимог через моделювання бізнес-процесів. Пошук втрачених вимог. Перевірка вимог. Виявлення та документування бізнес-правил. Атрибути якості програмного забезпечення.

Вимоги – це специфікація того, що повинно бути реалізовано. В них описана поведінка системи, її властивості чи атрибути, які можуть використовуватися як обмеження при розробці системи.

Таблиця 3.1. Класифікація вимог

Поняття	Визначення
Бізнес-вимога **(business requirement)**	високорівнева бізнес-мета підприємства чи замовника
Бізнес-правило **(business rules)**	політика, стандарт, правило, що обмежує деякі елементи бізнес-процесу
Обмеження **(constraints)**	обмеженість варіантів розробки чи проєктування програмного забезпечення
Зовнішня вимога до інтерфейсу	опис взаємодії між ПЗ та користувачем, іншими програмами чи пристроями
Характеристика **(feature)**	логічно пов'язані можливості системи, які мають цінність для користувача та описані певними функціональними вимогами
Функціональна вимога **(functional requirement)**	опис поведінки системи, що вимагається в певних умовах
Нефункціональна вимога **(nonfunctional requirement)**	опис властивості чи особливості, які повинна мати система, або обмеження, якому повинна відповідати (виконувати) система
Атрибут якості **(quality attributes)**	вид нефункціональної вимоги, що описує характеристику сервісу або продуктивності продукту
Системна вимога **(system requirement)**	вимога верхнього рівня до продукту, що містить багато підсистем (ПЗ) чи ПЗ та обладнання

Зв’язок типів інформації для формування вимог наведений на рис. 3.1

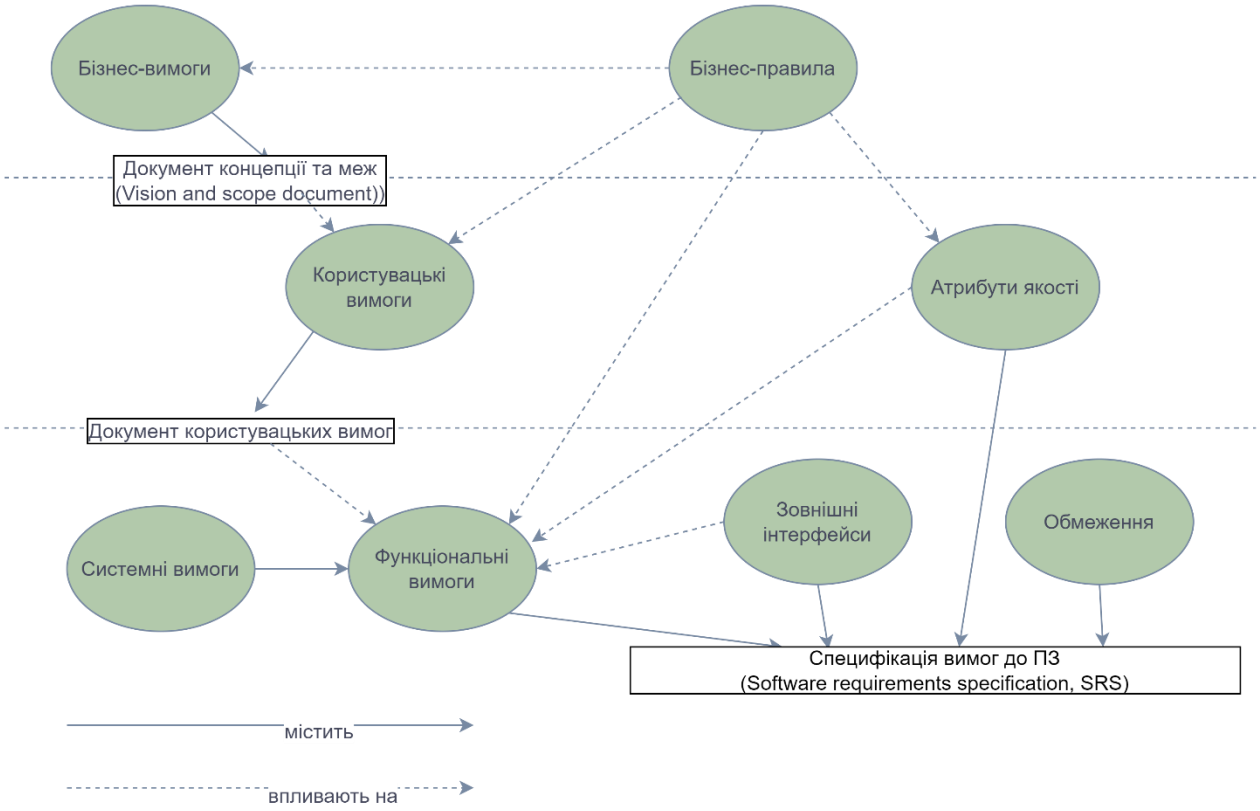


Рис. 3.1. Зв’язок типів інформації для формування вимог

Виявлення вимог може бути здійснено шляхом безпосереднього спілкування із зацікавленими сторонами або через проведення деяких досліджень, експериментів.

Базові методи:

1) Аналіз зацікавлених сторін (будується реєстр стейкхолдерів та матриця зацікавленості/впливу).

2) Мозковий штурм.

3) Інтерв’ю.

Це найпоширеніша техніка, яка використовується для виявлення вимог. Методи співбесіди слід використовувати для побудови міцних відносин між бізнес-аналітиками та зацікавленими сторонами. У цій техніці інтерв’юер направляє запитання зацікавленим сторонам для отримання інформації. Може використовуватися техніка «5 чому».

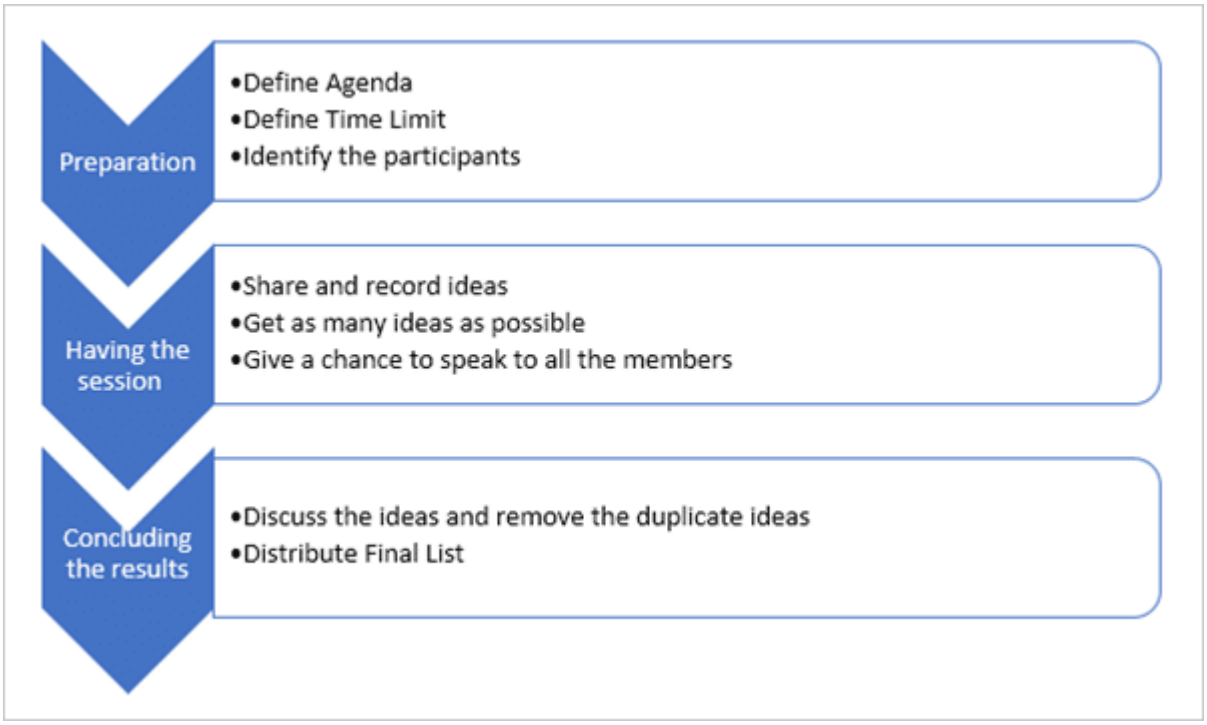


Рис. 3.1. Етапи мозкового штурму

4) Аналіз / огляд документів.

Цей прийом використовується для збору інформації шляхом перегляду / вивчення наявних матеріалів, що описують бізнес-середовище. Цей аналіз корисний для підтвердження впровадження поточних рішень, а також для розуміння бізнес-потреб.

Аналіз документів включає перегляд бізнес-планів, технічних документів, звітів про проблеми, існуючих документів про вимоги тощо.

Техніка важлива для виявлення прогалин у системі, для порівняння процесу **As is** із процесом **To be**.

5) Фокусна група.

До фокус-групи входять експерти з предметної галузі. Завдання цієї групи – обговорити тему та надати інформацію. Якщо продукт знаходиться на стадії розробки, то результатом буде оновлення існуючих вимог, або отримання нових вимог. Якщо ПЗ готове до поставки, обговорення буде йти про випуск продукту.

6) Аналіз інтерфейсу.

Аналіз інтерфейсу використовується для огляду системи, людей та процесів. Цей аналіз використовується для визначення способу обміну інформацією між компонентами.

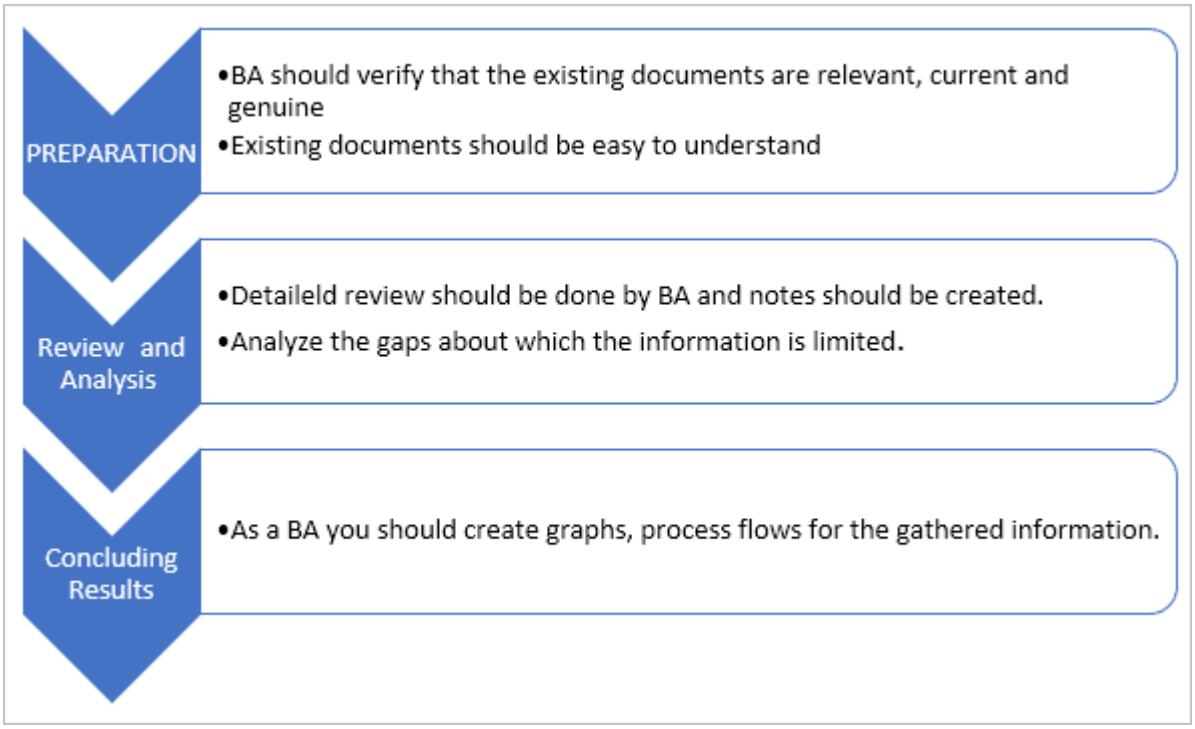


Рис. 3.2. Етапи аналізу / огляду документів

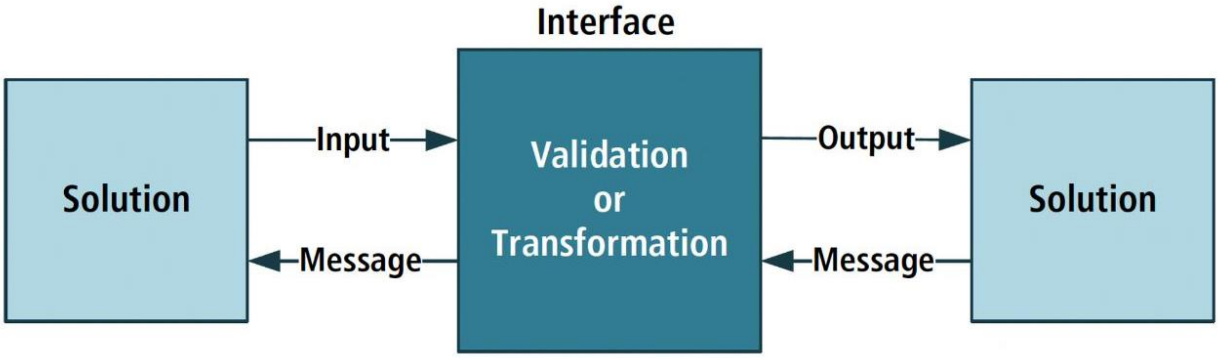


Рис. 3.3. Аналіз інтерфейсу

7) Спостереження.

Основна мета сеансу спостереження – зрозуміти діяльність, завдання, використовувані інструменти та події, що виконуються іншими.

План спостереження гарантує, що всі зацікавлені сторони знають про мету сесії спостереження, вони погоджуються на очікувані результати та що сесія відповідає їхнім очікуванням.

Спостереження може бути як активним, так і пасивним.

Активне спостереження полягає в тому, щоб задавати питання і намагатися спробувати роботу, яку виконують інші люди.

Пасивне спостереження – просто спостереження без втручання в роботу.

8) Прототипування.

Прототипування використовується для виявлення відсутніх або невизначених вимог. Функції демонструються клієнту шляхом створення прототипів, щоб клієнт міг скласти уявлення про те, як буде виглядати продукт.

9) Опитування / анкета.

Для опитування / анкетування зацікавленим сторонам надається набір питань для кількісної оцінки їх думок. Після збору відповідей аналізуються дані для виявлення сфери інтересів зацікавлених сторін. Питання повинні базуватися на високопріоритетних ризиках. Питання повинні бути прямими і однозначними.

Визначення вимог через аналіз бізнес-процесів

Аналіз (моделювання) бізнес-процесів – це візуальне відображення суб'єктивного бачення реально існуючих в організації сукупностей робіт за допомогою графічних, табличних і текстових засобів.

BPMN (англ. Business Process Model and Notation, модель та нотація бізнес-процесів) – система умовних позначень (нотація) для моделювання бізнес-процесів. Розроблена Business Process Management Initiative (BPMI) та підтримується Object Management Group після їх злиття в 2005 році. Остання версія BPMN – 2.0, що була прийнята у січні 2011 року.

Модель та нотація бізнес-процесів є стандартом для моделювання бізнес-процесів, що надає графічну нотацію для **визначення бізнес-процесу у вигляді "Діаграми бізнес-процесу"** (Business Process Diagram, BPD). Така діаграма ґрунтується на представлені бізнес-процесу у вигляді блок-схеми, що семантично схожа на діаграму діяльності.

BPMN підтримує тільки набір концепцій, що необхідні для моделювання виключно бізнес-процесів. Моделювання інших аспектів бізнесу (підприємства) таких як дані, організаційна структура чи інформаційні потоки не є предметом моделювання в BPMN. Тим не менше, в нотації BPMN передбачено можливість моделювання потоків даних та потоків повідомлень, а також асоціації даних та дій.

Моделювання з використанням BPMN виконується у вигляді діаграм, що складаються з різних елементів.

Розрізняють чотири категорії елементів:

об'єкти потоку керування: дії, події та логічні оператори;

поєднуючі елементи: потік керування, потік повідомлень та асоціації;

ролі: пули та доріжки;
артефакти: дані, групи та текстові анотації.

BPMN. Елементи Пул і Доріжка

Весь бізнес-процес складається з пулів: сукупності операцій + осіб, які ці операції виконують.

Пул використовується для позначення меж бізнес-процесу.
Доріжка використовується для відображення відповідальних виконавців і їх ролей в процесі.

BPMN елемент "Дія"

Під "дією" розуміється одиниця роботи, що виконується в ході виконання бізнес-процесу. Дії можуть бути як елементарними (завдання / task), так і складними (підпроцес / sub-process).

Є кілька типів елементарних дій, які відрізняються умовами виконання:

– **багаторазове** виконання дії в межах одного процесу. Наприклад, одну дію можна виконувати паралельно для кожного товару в замовленні клієнта.

– **циклічна дія** виконується багаторазово, поки задана умова вірна.

BPMN передбачає наступні графічні відображення для основних типів дій:

	**Абстрактна задача** використовується для позначення простої дії або операції, яка не має подальшої декомпозиції в межах поточного бізнес-процесу.
	**Підпроцес** використовується для відображення процесу, включеного до складу даного процесу. Підпроцесс описаний більш детально на своїй діаграмі.
	**Процес-посилання** використовується для позначення посилання на один з найбільш часто повторюваних процесів.

BPMN елементи "Розвилка" або "Шлюз"

Під шлюзами розуміються елементи, що визначають розгалуження і злиття потоків робіт.

BPMN описує багато типів розвилок. В якості основних виділяють 2 типи:

	**Шлюз «або», що виключає,** використовується для створення альтернативних потоків процесу або потоків управління, що сходяться.
	**Паралельний шлюз** використовується для створення паралельних шляхів без оцінки умови або для потоків, що сходяться, і синхронізації паралельних гілок виконання процесу.

Приклад використання шлюзу «або» для створення альтернативних потоків процесу «Дзвінок клієнту з метою оцінити якість обслуговування»:

1. Якщо клієнт задоволений, фіксація позитивної оцінки, закриття бізнес-процесу.

2. Якщо клієнт незадоволений, з'ясування причини.

Подальша схема може сильно гілкуватися: якщо клієнт незадоволений доставкою, то потрібно зв'язатися з начальником цієї служби; а якщо якістю продукції, то наступним етапом буде передача претензії до відділу виробництва, або ескалація (підняття ієрархічного рівня) з метою донести відомості про таку претензії до більш високого керівництва.

Фактично, шлюзи є одними з найбільш відповідальних і складних етапів бізнес-процесів. Від того, наскільки грамотно будуть прописані всі умови і слідства за принципом "Якщо ... то", багато в чому залежить ефективність роботи всієї системи.

BPMN елемент "Подія"

"Подія" є одним з головних елементів BPMN і служить для опису того, що має статися (на відміну від завдання, коли щось має бути зроблено). Подією може бути, наприклад, підписання договору, або розмова з клієнтом.

Графічні елементи подій в BPMN класифікують двома способами:

– залежно від положення події на схемі процесу;

– за типом події.

Перший спосіб:

	Початкова подія (ініціює бізнес-процес)
	Проміжна подія
	Кінцева подія (що закінчує бізнес-процес)

У випадку з доставкою товару початковим подією буде, очевидно, заявка клієнта. Або ж – дзвінок менеджера клієнта з пропозицією здійснити покупку.

Кінцевою подією в такому ланцюжку стане факт доставки, підтверджений підписом клієнта.

Другий спосіб:

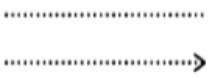
	**Проста подія** представляє не типізовану подію.
	**Подія-повідомлення** показує відправку або отримання повідомлення.
	
	**Подія-таймер** використовується для моделювання регулярних подій. Також таймер може використовуватися для моделювання моментів часу, часових проміжків і перевищення ліміту часу.

Дуже часто початкові і кінцеві події є подіями-повідомленнями.

BPMN елементи "Потоки"

Потік – це послідовність дій, яка позначається стрілкою. Елемент "потік" показує яку дію після якої необхідно здійснити.

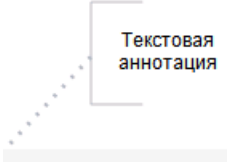
	**Потік управління**. На стандартний потік керування не впливають умови і він не проходить через шлюзи, тобто є неконтрольованим.
	**Умовний потік управління**. Використовується для того, щоб показати, що подальше виконання процесу буде відбуватися за певним потоком тільки в тому випадку, якщо буде виконана задана умова.

	Ромбик біля основи стрілки додається, якщо умовний потік управління є вихідним від процесу. Ромбик не додають, якщо умовний потік управління є вихідним від шлюзу.
	Потік управління за замовчуванням. Використовується тоді, коли необхідно показати, що подальше виконання процесу буде відбуватися за певним потоком тільки якщо не виконується жодна з заданих умов.
	**Потік повідомлень**. Використовується для відображення взаємодії між процесами – відображає передачу повідомлень або об'єктів з одного процесу в інший процес або зовнішнє середовище.
	**Асоціація**. Застосовується для візуалізації зв'язку між елементами потоку і об'єктами, які не є елементами потоку (артефактами).

BPMN елементи "Артефакт"

Під артефактами в BPMN розуміють об'єкти, які не впливають на виконання бізнес-процесу безпосередньо. Це можуть бути документи, дані, інформація.

Основні види артефактів:

	**Група об'єктів**. Використовується для групування графічних елементів, що належать одній і тій же категорії, і дозволяє підвищити простоту сприйняття діаграми.
	**Текстова анотація**. Застосовується для уточнень на діаграмі – коментарів і пояснень, які збільшать читабельність діаграми.
	**Об'єкт даних**. Використовується для відображення інформації про дані, які обробляються в ході процесу.

Приклад 1. Створення схеми BPMN для процесу укладення контракту (рис. 3.4).

Приклад 2. Забезпечення замовлення покупця (рис. 3.5).

Точкою входу служить отримання замовлення від покупця. Точкою виходу – "резервування товару".

Після отримання замовлення стрілка веде до умови:

1. Якщо весь товар є, то менеджер виконує підпроцес «резервування товарів». Підпроцес використовується для того, щоб мати можливість за потреби деталізувати дії менеджера. А потім – до точки виходу "Резервування товарів проведено".

2. Якщо товарів немає, то менеджер виконує запит до відділу закупівлі. Інформація про замовлення переходить у відділ закупівлі до іншого виконавця – менеджера із закупівель, який створює замовлення постачальнику. На схемі також видно, що замовлення постачальнику створено на основі запиту на постачання та замовлення постачальникам.

Рівні моделювання

Залежно від цілей побудови BPMN-діаграм, розрізняють 3 рівні моделювання:

1. **Описове моделювання –** коли потрібно показати успішний шлях виконання бізнес-процесу, наприклад, щоб узгодити його з бізнес-користувачем. Тут використовуються найпростіші елементи нотації, а сама діаграма навмисно максимально спрощується.

2. **Аналітичне моделювання –** використовується, коли потрібно повністю показати всі варіанти виконання бізнес-процесу, включаючи логічні розгалуження та альтернативи. Така діаграма зазвичай створюється для досвідчених користувачів та бізнес-аналітиків за допомогою розширеного алфавіту нотації, включаючи не лише її базові найпростіші елементи, а й складніші.

3. **Виконуване моделювання –** призначений для запуску на виконання в BPMS-движку, щоб створити веб-додаток. Тут можна використовувати все різноманіття алфавіту цієї нотації, включаючи додавання спеціальних параметрів і скриптів, створюваних розробниками.

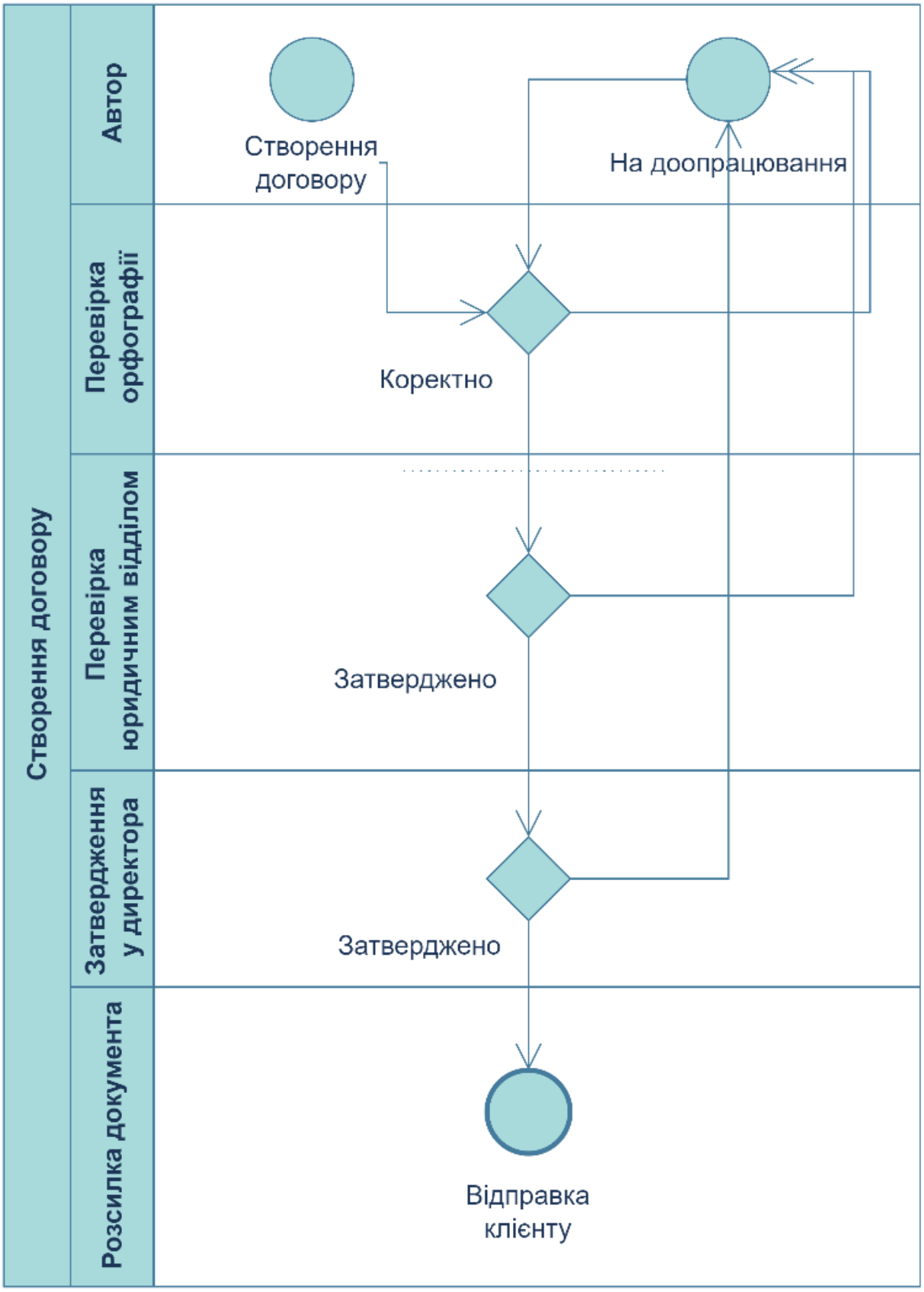


Рис. 3.4.1. Приклад BPMN схеми узгодження договору. Варіант 1

Рис. 3.4.2. Приклад BPMN схеми узгодження договору. Варіант 2

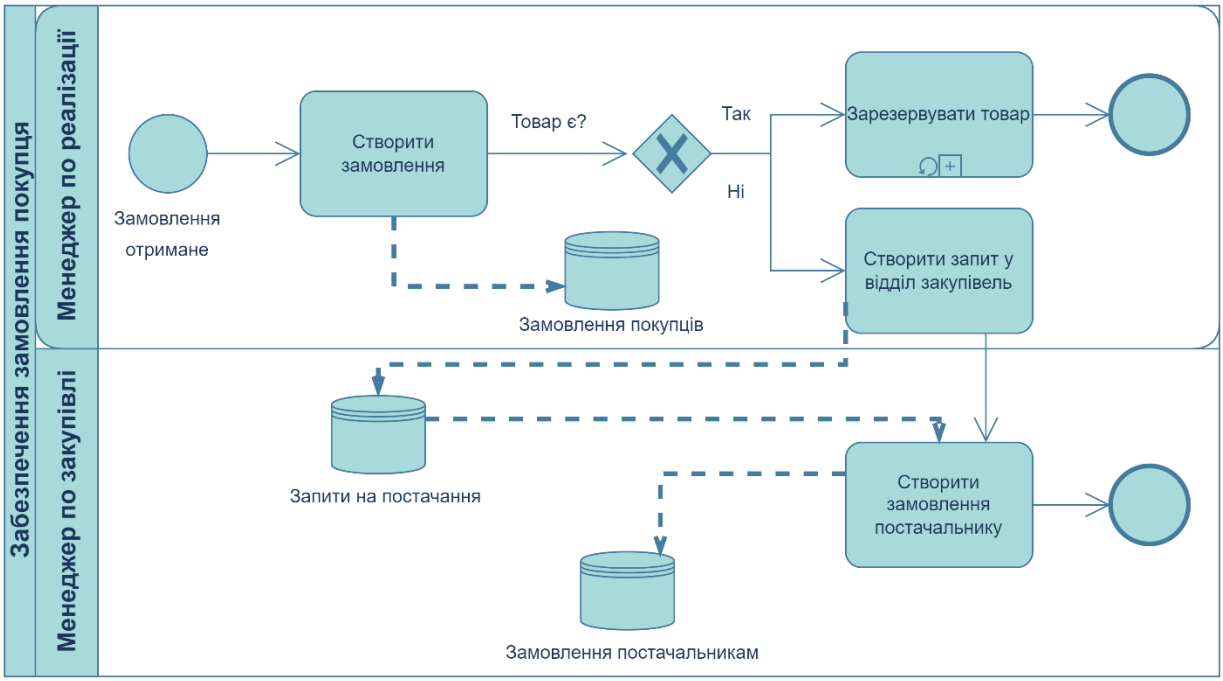


Рис. 3.5. Забезпечення замовлення покупця

Приклад 3. Створення схеми BPMN для процесу формування заяви на відпустку (рис. 3.6).

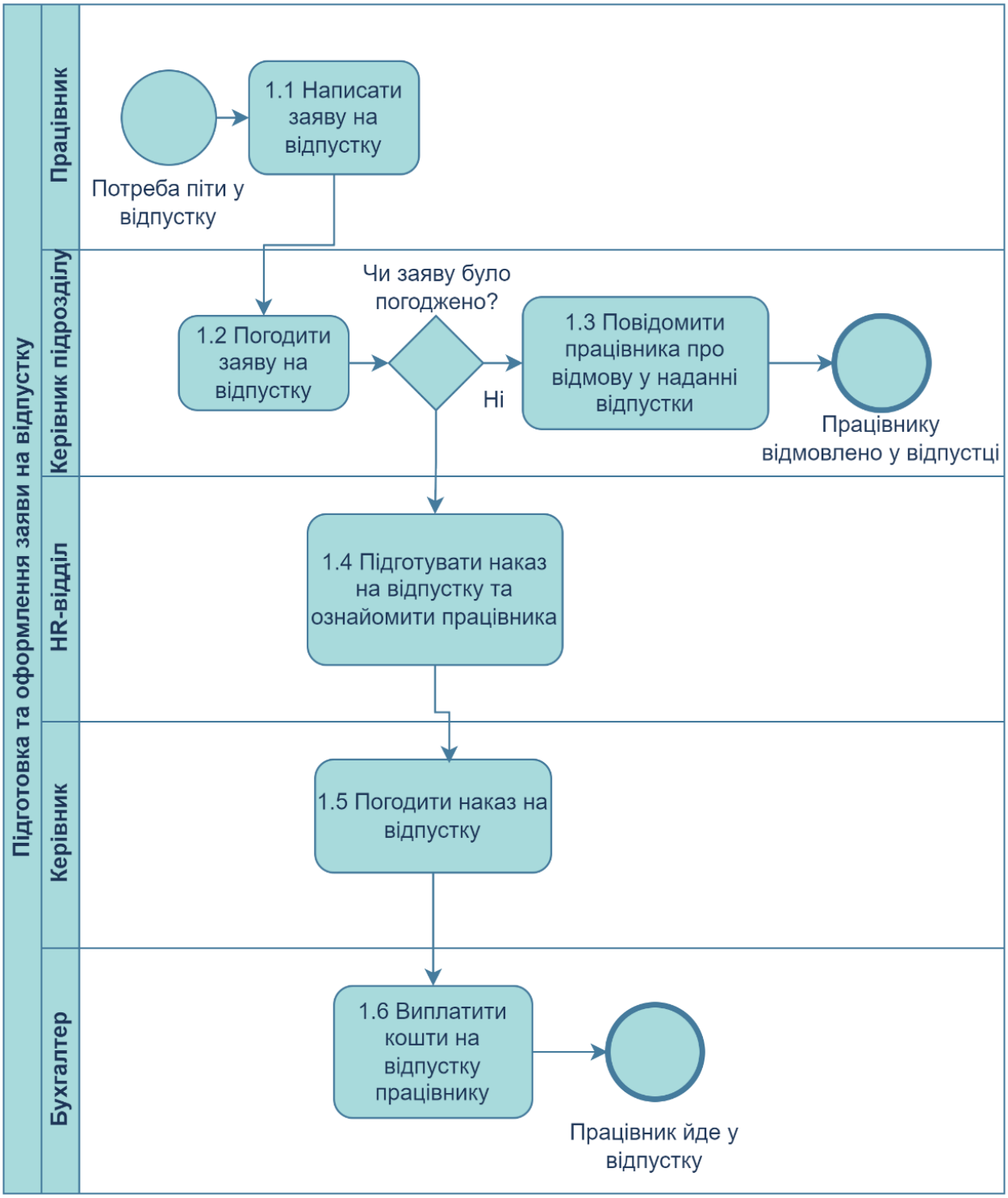


Рис. 3.6. Приклад BPMN схеми оформлення заяви на відпустку

Рекомендації щодо використання BPMN

Зважаючи на три рівні моделювання BPMN і надмірний алфавіт цієї нотації, можна зробити висновок, що при проєктуванні діаграм «для людей» (без запуску на виконання в BPMS-системах) слід навмисно обмежити кількість елементів, що використовуються:

- Використовувати лише користувацькі та ручні завдання — без сценаріїв, сервісів та бізнес-правил, надсилання та отримання повідомлень.
- Використовувати лише згорнуті підпроцеси, розкриваючи їх деталі на окремій діаграмі.
- Використовувати тільки XOR та AND, без подійних шлюзів та OR, так як різниця між виключаючим та не виключаючим АБО зрозуміла не всім користувачам.
- Використовувати події з типом просте, таймер, повідомлення та зупинка.

Для спрощення сприйняття діаграми варто дотримуватись правил найменування:

- Зовнішніх контрагентів показувати як закриті, вони ж — згорнуті пули (пули, в яких немає дій).
- Називати закриті пули ролями чи бізнес-одиницями, а відкриті – процесами.
- Називати доріжки також як роль, посаду або структурний підрозділ.
- Називати дії (завдання) у стилі Дієслово-Іменник, наприклад, «Перевірити рахунок», «Підтвердити заявку», «Оформити договір».
- Називати події як факт, що відбувся в минулому часі, наприклад, «Надійшла заявка», «Пройшло 3 дні».
- Підписувати стрілки, що виходять з XOR, наприклад, «Так» і «Ні», а також відзначати потік за замовчуванням.

Також рекомендується:

- Показувати успішне та неуспішне завершення процесу різними фінішними подіями.
- Не виводити потік керування за межі підпроцесу.
- Взаємодію між різними пулами показувати через потік повідомлень (пунктирною стрілкою), який не може приєднуватися до шлюзів, на відміну від потоку керування.

Нарешті, при розробці будь-якої діаграми потрібно пам'ятати про головне правило аналітика: незалежно від нотації, ваша схема має бути МАКСИМАЛЬНО простою та зрозумілою читачеві БЕЗ знання тонкощів процесного моделювання!

Загалом алгоритм розробки BPMN-діаграми можна подати як набір наступних 7 кроків:

1. Визначити межі процесу, тобто стартову та кінцеву події, учасників та корисний результат.
2. Описати "щасливий" шлях (happy path), що веде до створення корисного результату (продукту).
3. Додати умови та альтернативні потоки.
4. Додати неуспішні завершення.
5. Додати артефакти (об'єкти та сховища даних).
6. Розкрити на нових зв'язаних діаграмах згорнуті підпроцеси.
7. Додати проміжні потоки подій до зовнішніх пулів.

Щоб розпочати роботу над бізнес-процесом, попередньо потрібно отримати текстовий опис. Найчастіше для цього проводять інтерв'ю співробітників компанії. При цьому задають уточнюючі питання і на їх основі створюється звичайний текст, який звичайною мовою описує, з чого починається процес, хто і які дії виконує, що має вийти в результаті.

Після того, як цей текст узгоджений із замовником, на його основі можна розпочинати роботу з моделювання бізнес-процесу.

Приклад 4. Текстовий опис процесу:

«При потребі в товарі, якого немає в наявності, ***Менеджер з реалізації*** *створює документ “Заявка на закупівлю” і направляє його на погодження* ***Менеджеру із закупівель****.* ***Менеджер із закупівель*** *перевіряє необхідність закупівлі даного товару і якщо він дозволяє закупити товар згідно з документом “Заявка на закупівлю”, то* ***Менеджер з реалізації*** *інформується про дозвіл закупити товар, і* ***Менеджер із закупівель*** *створює документ “Замовлення постачальнику”. Інакше заявка анулюється з коментарем, що містить причину відмови у закупівлі товару.* ***Менеджер з реалізації*** *інформується про відмову у закупівлі товару».*

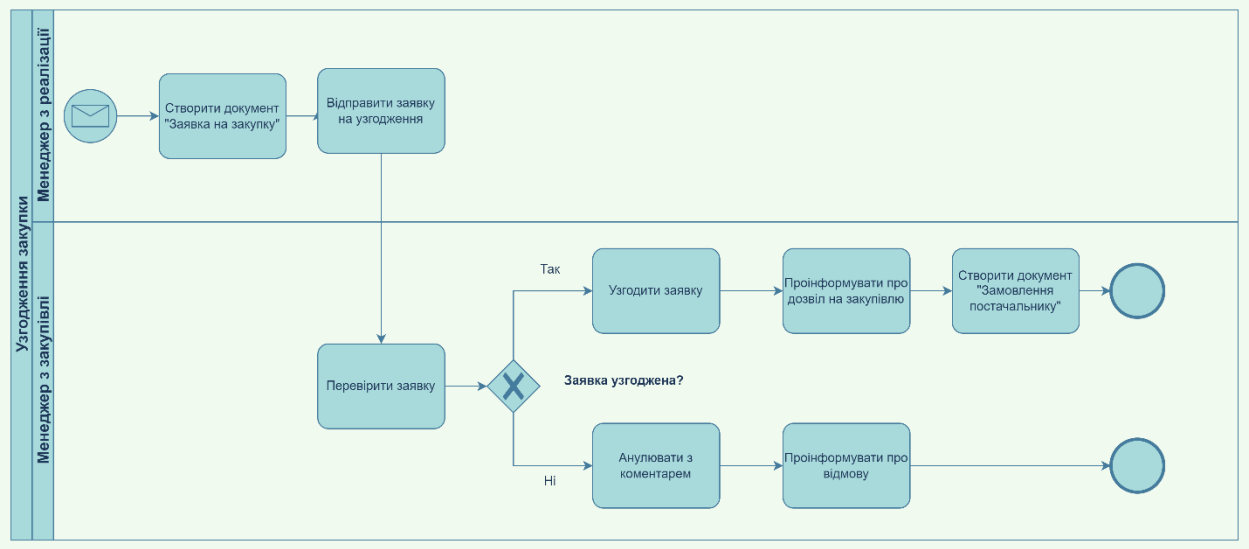


Рис. 3.7. Модель процесу узгодження закупок. Варіант 1

Опис бізнес-процесу:

1. Створити документ "Заявка на закупівлю".

Після отримання заявки від покупця ***Менеджер з реалізації*** формує в системі документ «Заявка на закупівлю», що включає весь перелік товарів, яких немає у ***Менеджера із закупівель***.

2. Надіслати заявку на погодження.

Менеджер з реалізації надсилає сформований документ на узгодження ***Менеджеру із закупівель***.

3. Перевірити заявку.

Менеджер із закупівель перевіряє «Заявку на закупівлю» і визначає, чи ці товари необхідно закупити у постачальника.

Якщо ***Менеджер із закупівель*** приходить до висновку, що закупівля є доцільною:

4. ***Менеджер із закупівель*** підтверджує заявку.

5. Інформація про дозвіл на закупівлю товару надсилається ***Менеджеру з реалізації***.

6. ***Менеджер із закупівель*** створює документ Замовлення постачальнику. **На цьому процес завершено.**

Якщо ***Менеджер із закупівель*** заявку не схвалює:

7. ***Менеджер із закупівель*** анулює заявку із коментарем, який пояснює причини відмови.

8. ***Менеджер з реалізації*** отримує повідомлення про відмову із коментарем ***Менеджера із закупівель***. **На цьому процес завершено.**

Приклад 5. Текстовий опис:

«При потребі в товарі, якого немає в наявності, ***Менеджер з реалізації*** *створює документ «Заявка на закупівлю» і направляє в сторонній додаток. Сторонній додаток перевіряє суму заявки і, якщо сторонній додаток схвалює заявку, вона відправляється до* ***Менеджера із закупівель****. Якщо заявка не була схвалена стороннім додатком, скрипт автоматично анулює її з коментарем та інформує про відмову* ***Менеджера з реалізації****.* ***Менеджер із закупівель*** *приймає заявку та перевіряє необхідність закупівлі даного товару і, якщо* ***Менеджера із закупівель*** *дозволяє закупити товар згідно з документом «Заявка на закупівлю», то він інформує* ***Менеджера з реалізації*** *про дозвіл закупити товар і одночасно створює документ «Замовлення постачальнику». Інакше заявка анулюється із коментарем, що містить причину відмови у закупівлі товару.* ***Менеджер з реалізації*** *інформується про відмову у закупівлі товару».*

Також у процесі затвердження процесу було вирішено, що завдання «Створити документ «Замовлення постачальнику» та «Інформувати про дозвіл на закупівлю товару» мають виконуватися одночасно.

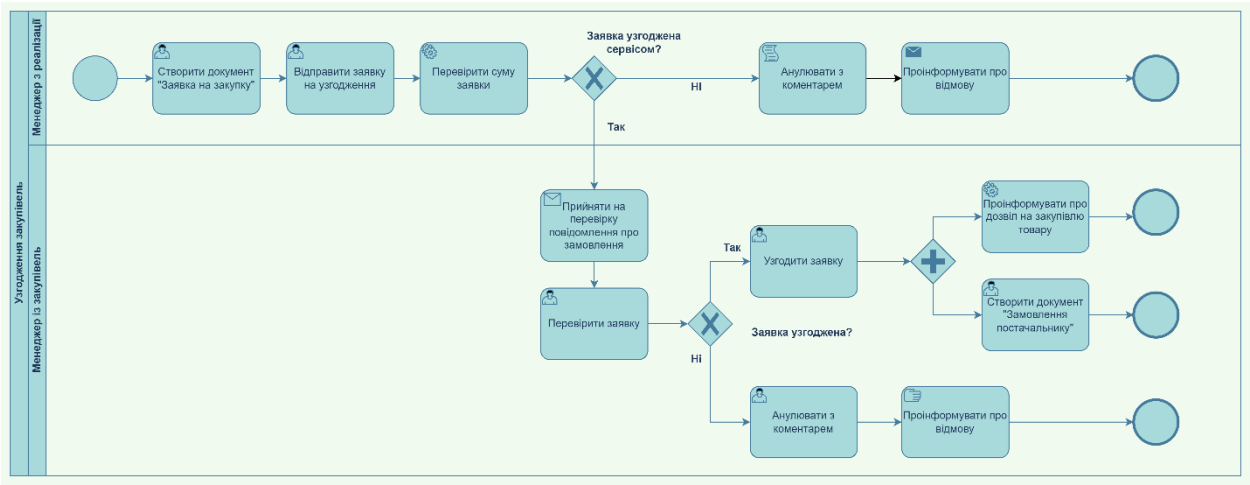


Рис. 3.8. Модель процесу узгодження закупок. Варіант 2

Опис бізнес-процесу:

1. Створити документ "Заявка на закупівлю".

Менеджер з реалізації створює в системі документ «Заявка на закупівлю», що включає весь перелік товарів, які, на думку ***Менеджера з реалізації***, необхідно закупити, ґрунтуючись на поточних заявках від покупців і угодах, що відбулися.

2. Надіслати заявку на погодження.

Менеджер з реалізації спрямовує сформований документ на погодження у сторонній сервіс.

3. Перевірити суму заявки:

– Якщо заявка не відповідає заданим умовам:

3.1.1 Сервіс автоматично анулює її з коментарем, який пояснює відмову.

3.1.2. ***Менеджер з реалізації*** інформується про відмову. **На цьому процес завершено.**

– Якщо заявку схвалено сервісом:

3.2.1. ***Менеджер із закупівель*** приймає повідомлення про замовлення на перевірку.

3.2.2. ***Менеджер із закупівель*** перевіряє заявку.

– Якщо ***Менеджера із закупівель*** приймає рішення схвалити заявку:

3.2.2.1.1 ***Менеджер із закупівель*** схвалює заявку в інформаційній системі.

3.2.2.1.2. Одночасно формується документ «Замовлення постачальнику» та надсилається повідомлення ***Менеджеру з реалізації*** про дозвіл закупити товар. **Процес завершено.**

– Якщо ***Менеджер із закупівель*** не схвалює заявку:

3.2.2.2.1 Заявка анулюється із коментарем, який пояснює причини відмови.

3.2.2.2.2 ***Менеджер з реалізації*** інформується про відмову та її причини. **Процес завершено.**

Приклад 6. Текстовий опис:

«При потребі в товарі, якого немає, ***Менеджер з реалізації*** *створює документ «Заявка на закупівлю» і направляє в сторонній додаток. У сторонній IT-системі перевіряється сума заявки і, якщо сторонній додаток схвалює заявку, вона надсилається* ***Менеджеру із закупівель****.* ***Менеджер із закупівель*** *перевіряє необхідність закупівлі даного товару і, якщо він дозволяє закупити товар згідно з документом «Заявка на закупівлю», то* ***Менеджер з реалізації*** *інформується про дозвіл закупити товар, і* ***Менеджер із закупівель*** *створює документ*

«Замовлення постачальнику». Інакше заявка анулюється із коментарем, що містить причину відмови у закупівлі товару. ***Менеджер з реалізації*** *інформується про відмову у закупівлі товару».*

У цьому прикладі ми використовуємо для перевірки суми замовлення сторонню IT-систему, а щоб правильно відобразити графічно, використовується елемент «Сервісне завдання».

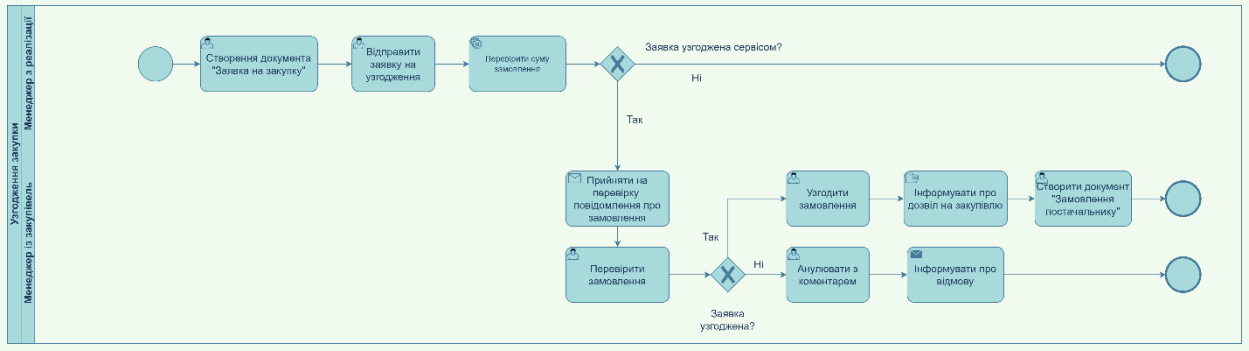


Рис. 3.9. Модель процесу узгодження закупок. Варіант 3

Опис бізнес-процесу:

1. Створити документ "Заявка на закупівлю".

Менеджер з реалізації формує в інформаційній системі перелік товарів, які потрібно закупити.

2. Надіслати заявку на погодження.

Менеджер з реалізації надсилає Заявку на погодження, вона надсилається до стороннього додатка.

3. Перевірити суму заявки.

Сторонній сервіс (на графічній нотації це видно завдяки використанню елемента «Сервісне завдання») перевіряє суму заявки.

– Якщо заявку не схвалено сервісом, **процес завершено**.

– Якщо заявку схвалено сервісом:

4. ***Менеджер із закупівель*** отримує повідомлення про замовлення на перевірку.

5. ***Менеджер із закупівель*** перевіряє заявку.

– Якщо заявку схвалено:

6. Схвалити заявку.

Менеджер із закупівель у системі підтверджує схвалення заявки.

7. Інформувати про дозвіл на закупівлю товару.

Система автоматично надсилає ***Менеджеру з реалізації*** інформацію про те, що Заявка на закупівлю схвалена.

8. Створити документ "Замовлення постачальнику".

Менеджер із закупівель формує "Замовлення постачальнику". **Процес завершено.**

– Якщо заявка не схвалена:

9. Анулювати заявку із коментарем.

Менеджер із закупівель анулює заявку, у коментарі пояснює причини відмови.

10. Поінформувати про відмову.

Менеджеру з реалізації надсилається повідомлення про відмову із коментарем ***Менеджера із закупівель***. **Процес завершено.**

Приклад 7. Текстовий опис:

*«Якщо **Менеджер з реалізації** отримує повідомлення про потребу у товарі, він створює документ заявку на закупівлю та спрямовує його на узгодження **Менеджеру із закупівель**. **Менеджер із закупівель** перевіряє необхідність закупівлі даного товару і, якщо він дозволяє закупити товар згідно з заявкою на закупівлю, то **Менеджер з реалізації** інформується про дозвіл закупити товар і **Менеджер із закупівель** створює замовлення постачальнику. **Менеджер із закупівель** к інформує клієнта про статус замовлення. Інакше заявка анулюється із коментарем, що містить причину відмови у закупівлі. **Менеджер з реалізації** інформується про відмову у закупівлі».*

Опис бізнес-процесу:

Процес ініціюється після того, як ***Менеджер з реалізації*** отримує замовлення від клієнта.

1. Створити документ "Заявка на закупівлю".

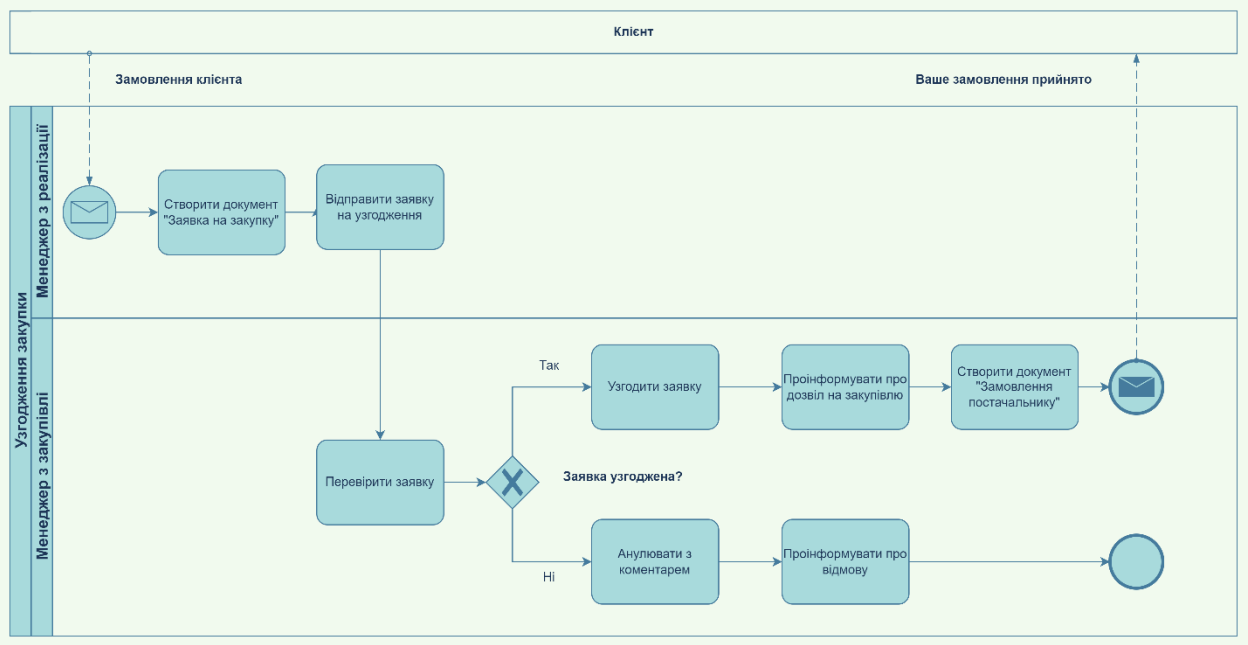
Менеджер з реалізації на основі замовлення клієнта формує в інформаційній системі документ "Заявка на закупівлю".

2. Надіслати заявку на погодження.

Після того, як документ повністю сформований, він вирушає на узгодження ***Менеджеру із закупівель***.

3. Перевірити заявку.

Менеджер із закупівель перевіряє заявку на закупівлю та приймає рішення, чи справді вказаний список товарів потрібно закупити у постачальника.



[9]

Рис. 3.10. Модель процесу узгодження закупок. Варіант 4

– Якщо заявку схвалено:

4. Схвалити заявку.

Після ухвалення позитивного рішення ***Менеджер із закупівель*** у системі підтверджує схвалення заявки.

5. Інформувати про дозвіл на закупівлю товару.

Менеджеру з реалізації надсилається повідомлення, в якому він інформується про схвалення заявки.

6. Створити документ "Замовлення постачальнику".

Менеджер із закупівель формує документ "Замовлення постачальнику".

7. Система надсилає клієнту листа з інформацією «Ваше замовлення прийнято» та завершує процес.

– Якщо заявка не схвалена:

8. Анулювати із коментарем у разі відмови.

[9] Схема для виявлення помилок у проєктуванні

Менеджер із закупівель анулює заявку, але пише коментар, де вказує причини відмови.

9. Поінформувати про відмову.

Менеджер з реалізації отримує повідомлення про відмову, яка також включає коментар Закупника. **Процес завершено.**

Формулювання вимог

Користувацька вимога **(user requirement)** – задача, яку певні класи користувачів повинні мати можливість вирішувати в системі, або певний атрибут продукту.

Функціональна вимога **(functional requirement) –** опис поведінки системи, що вимагається в певних умовах.

Нефункціональна вимога **(nonfunctional requirement) –** опис властивості чи особливості, які повинна мати система, або обмеження, якому повинна відповідати (виконувати) система.

Атрибут якості **(quality attributes)** – вид нефункціональної вимоги, що описує характеристику сервісу або продуктивності продукту.

Системна вимога **(system requirement) –** вимога верхнього рівня до продукту, що містить багато підсистем (ПЗ) чи ПЗ та обладнання.

Правила формулювання вимог:

Повнота – кожна вимога повинна містити всю інформацію, необхідну читачеві, щоб її зрозуміти.

Коректність – кожна вимога повинна точно описувати можливість, яка задовольнятиме якусь потребу, яку необхідно реалізувати.

Здійсненність – можливість реалізувати вимогу при відомих обмеженнях системи та робочого середовища, а також в межах тимчасових, бюджетних та ресурсних обмежень проєкту.

Необхідність – кожна вимога повинна відображати можливість, яка надасть очікувану бізнес-користь, виділить продукт на ринку або необхідна для дотримання зовнішніх стандартів, політик чи правил.

Призначення пріоритетів – визначайте пріоритети бізнес-вимог на підставі важливості для отримання необхідної користі.

Недвозначність – не більше одного способу інтерпретації вимоги.

Перевірка – чи зможе тестувальник розробити тести або застосувати інші прийоми, щоб встановити, чи дійсно у продукті реалізовано кожну вимогу.

Шлях від користувацьких вимог до функціональних наведений на рис. 3.11.

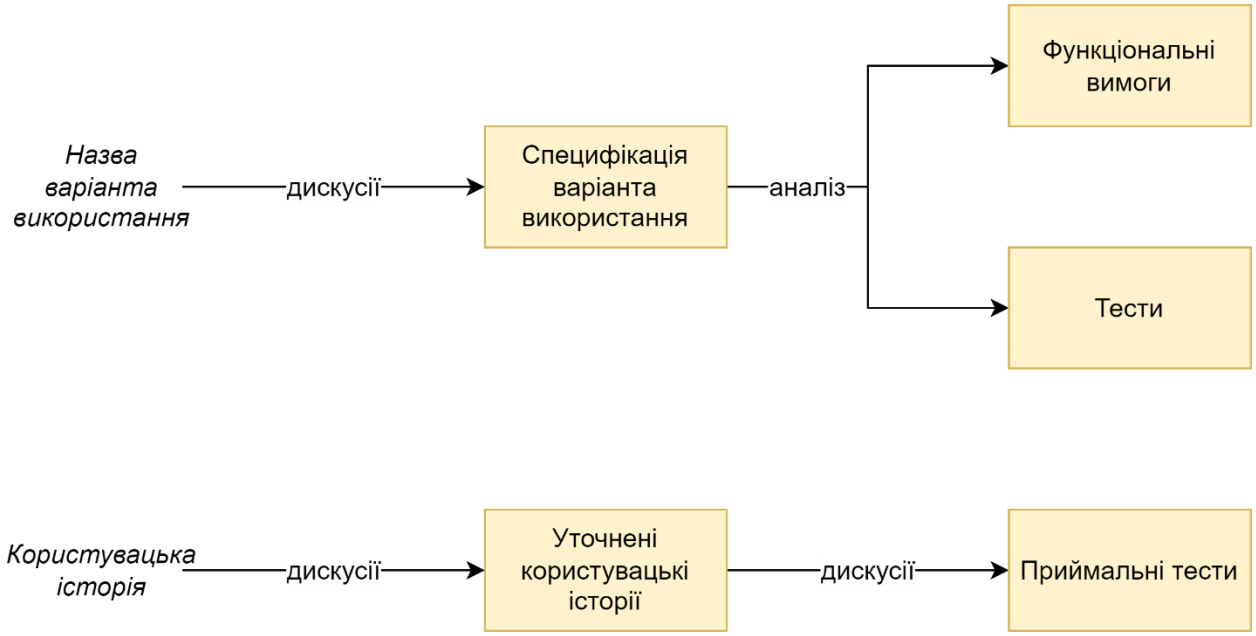


Рис. 3.11. Шлях від користувацьких вимог до функціональних

Варіант використання (use case) описує послідовність взаємодії системи та зовнішньої діючої особи, в результаті якої діюча особа отримує корисний результат.

Приклад діаграми варіантів використання наведений на рис. 3.12.

Сценарій – опис одного випадку використання системи; це окремий приклад варіанту використання.

Деякі елементи структури шаблона:

1. Вихідні умови:

- те, що користувач може спостерігати (залишок на рахунку);
- фізичні результати (банкомат видав гроші та роздрукував чек);
- зміна внутрішнього стану системи (зміна суми рахунку на суму знятих коштів.

2. Альтернативні сценарії – можуть призводити до успішного виконання завдання, задовольняють початковим умовам, але є менш пріоритетними чи менш популярними варіантами задачі чи способу її виконання.

3. Виключення – очікувана помилкова умова, яка може виникнути під час виконання варіанту використання, спосіб її обробки (деякі мови користувач може виправити, деякі призводять до безуспішного завершення варіанту використання.

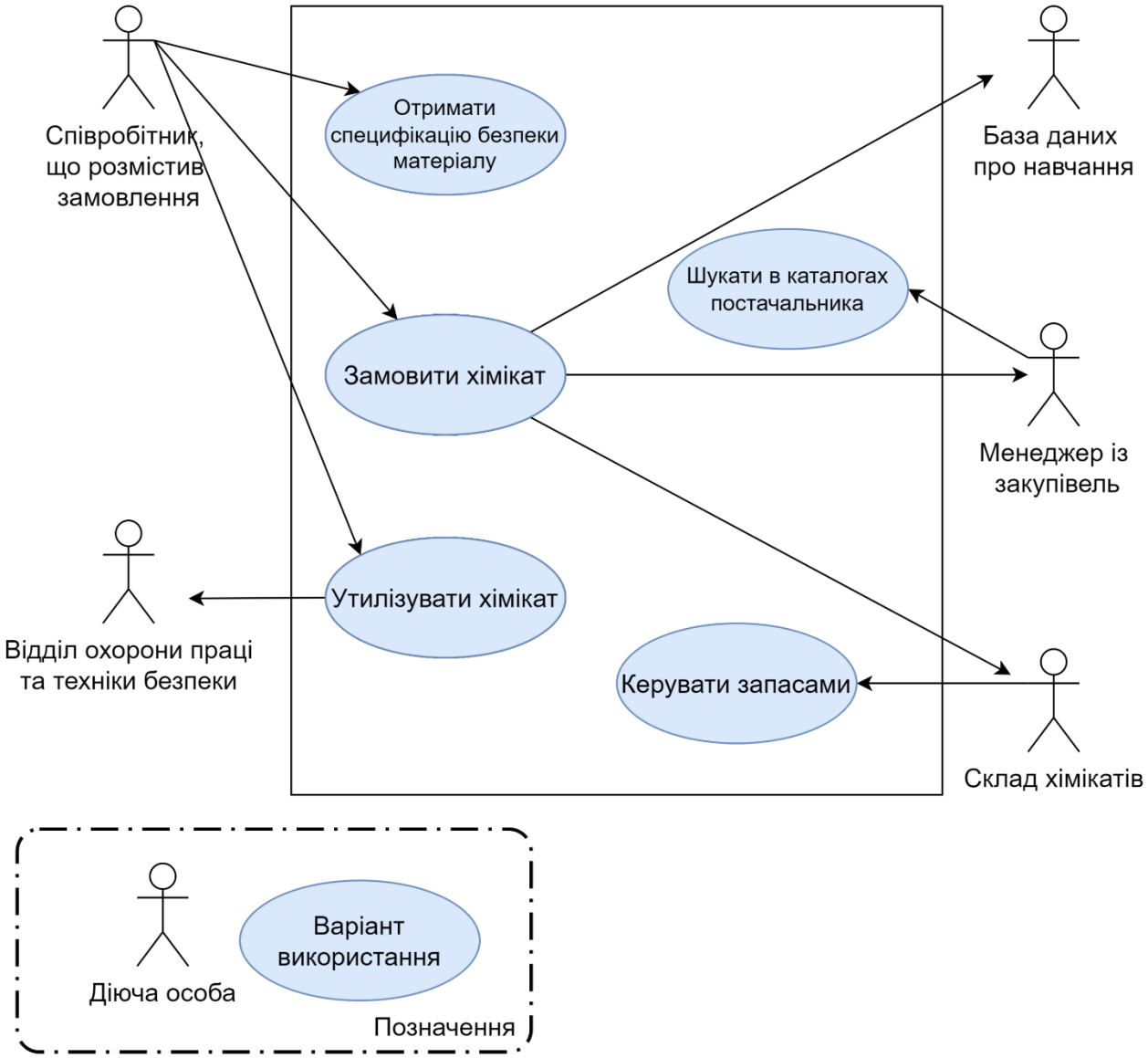


Рис. 3.12. Приклад діаграми варіантів використання

Як визначити варіанти використання?

1. Визначити дійових осіб, потім бізнес-процеси, що підтримуються системою, далі варіанти використання для дій, в яких приймають участь діючі особи та система.

2. Виразити бізнес-процеси в термінах визначених сценаріїв, узагальнити сценарії у варіанти використання та визначити діючих осіб для кожного варіанта.

3. Використовуючи опис бізнес-процесу, задати питання «Які задачі повинна виконати система, щоб реалізувати цей процес або перетворити вхідні дані у вихідні?». Ці задачі можуть бути варіантами використання.

4. Визначити зовнішні події, на які система повинна реагувати, а потім співвіднести ці події з діючими особами та певними варіантами використання.

5. Застосувати CRUD (Create, Read, Update, Delete – створення, читання, оновлення, видалення) для визначення сутностей даних, яким необхідні варіанти використання для створення, читання, оновлення, видалення чи інших операцій с даними.

6. Проаналізувати контекстну діаграму, запитуючи «Яких цілей потрібно досягти кожній з зовнішніх сутностей при використанні системи?»

На що звернути увагу при побудові use case?

1. Забагато варіантів використання → перевірити дублювання вимог, співвіднести з бізнес-вимогами.

2. Надто складні варіанти використання → правильно обрати нормальній напрям виконання use case.

3. Включення дизайну до опису варіантів використання → виключити вимоги до дизайну.

4. Включення визначення даних (наборів даних) → створити окремо словник даних.

5. Варіанти використання, які незрозумілі користувачам → перевірити зв'язок з бізнес-процесами та задачами користувачів.

Користувацька історія (user story) – стислий простий опис функції з позиції користувача:

Як **<тип користувача>** я хочу **<мета>** для того, щоб **<причина>**

Acceptance criteria – критерій прийняття, деталі, необхідні для виконання конкретної user story, опис того, що має бути виконано.

Наприклад: «Як клієнт я хочу завантажити виписку з мого рахунку, щоб я міг переглянути всі свої транзакції, здійснені за певний період».

Acceptance criteria:

– можливість вибрати період, протягом якого я хочу завантажити виписку;

– можливість вибрати обліковий запис, для якого я хочу завантажити виписку;

– можливість переглянути завантажений файл;

– можливість завантажити свою виписку у форматах doc, excel та pdf.

Виявлення та документування бізнес-правил

Бізнес-правило – це вказівка, що визначає чи обмежує певний аспект бізнесу. Вона необхідна для встановлення бізнес-структури чи для управління та впливу на бізнес-діяльність.

Рис. 3.13. Класифікація бізнес-правил

Факти (facts) – вірні твердження про бізнес на певний момент часу; описують зв'язки та відношення між важливими бізнес-термінами.

Приклади:

на кожний хімічний контейнер нанесений унікальний штрих-код;

сплачується доставка кожного замовлення;

якщо книга має розмір більш ніж 16 дюймів, то вона розміщується в розділі широкоформатних книг бібліотеки.

Обмеження (circumscription) – визначають які операції не може виконати система та її користувачі (ключові слова при зборі вимог: повинен, не повинен, не може, тільки).

Приклади:

1. Політики організації. Постійний відвідувач бібліотеки може відкласти для себе до 10 книг.

2. Державні нормативи. Екіпажи комерційних рейсів повинні кожні 24 години відпочивати не менш ніж 8 годин.

3. Галузеві стандарти. Веб-додатки не повинні містити ніяких HTML-тегів чи атрибутів, що відповідають стандарту HTML-5.

Активатор (action enable) – правило, що при певних умовах ініціює виконання певних дій.

Синтаксис: Якщо *<якась умова вірна чи настала певна подія>,* то *<щось відбудеться>.*

Висновки (inference) – можливе знання чи похідний факт, що створює новий факт на основі інших фактів.

Синтаксис: Якщо-то.

Приклади:

Якщо платіж не надійшов протягом 30 календарних днів з моменту відправлення рахунку, то рахунок вважається простроченим.

Якщо постачальник не може відправити замовлений товар протягом п'яти днів з моменту отримання замовлення, то замовлення вважається невиконаним.

Атомарні бізнес-правила (atomic) – правила, які не можуть бути розбиті на більш дрібні правила.

Трансформування бізнес-правил в обмеження

Обмеження накладають межі на доступний розробнику вибір дизайну та реалізації. Обмеження можуть накладатися зовнішніми зацікавленими особами, іншими системами, що взаємодіють з розроблюваним ПЗ. Інші обмеження виходять із існуючих нормативних актів, стандартів, угод, технічних рішень тощо, тобто з бізнес-правил.

Приклади джерел обмежень:

– певні технології, засоби, мови програмування та бази даних, які необхідно використовувати чи уникати;

– обмеження, що накладаються операційним середовищем чи платформою продукту (наприклад, версійність браузерів, встановлені ОС);

– обов’язкові угоди чи стандарти розробки (наприклад, певна нотація або її версійність);

– зворотна сумісність з раніше випущеними продуктами та можливості сумісництва з майбутніми версіями, наприклад, яка версія програми використовувалася для створення конкретного файлу;

– обмеження чи вимоги, що накладаються нормативними документами та іншими бізнес-правилами;

– обмеження, пов’язані з обладнанням, наприклад, вимоги до термінів, обмеження пам’яті, процесора, розміру, ваги, матеріалів, витрат тощо;

– фізичні обмеження, що накладаються операційним середовищем, характеристиками чи обмеженнями користувачів;

– угоди, пов’язані з використанням інтерфейсу існуючого продукту, яких необхідно дотримуватися при покращенні існуючого продукту;

– інтерфейси з іншими існуючими системами, наприклад, формати даних та протоколи зв’язку;

– обмеження, що пов’язані із розміром екрана, наприклад, при роботі на смартфоні чи планшеті;

– стандартний формат обміну даними для електронного бізнеса.

Документування правил можна здійснювати за допомогою реєстру (табл. 3.3).

Таблиця 3.3. Реєстр бізнес-правил

ID	Визначення правила	Тип правила	Статичне чи динамічне	Джерело
ORDER-5	Якщо клієнт замовив книгу автора, який написав декілька книг, то перед оформленням замовлення необхідно запропонувати придбати інші книги того ж автора	Активатор операцій	Статичне	Маркетингова політика
ACCESS-8	У всіх зображеннях веб-сайту повинен бути альтернативний текст для використання пристроями електронного читання, які використовуються особами з вадами зору	Обмеження	Статичне	Стандарти ADA для дизайну з врахуванням обмежених можливостей
DISCOUNT-13	Розмір знижки розраховується на основі розміру замовлення відповідно до табл….	Розрахунок	Динамічне	Корпоративна цінова політика

Джерела бізнес-правил (рис. 3.14):

– загальновідомі знання компанії;

– успадковані системи, у вимогах та коді яких реалізовані бізнес-правила;

– моделювання бізнес-процесів (обмеження, ініціація подій, правила розрахунків, пов'язані факти);

– аналіз існуючої документації, в тому числі специфікацій вимог з попередніх проєктів, нормативні документи, галузеві стандарти, документи корпоративних політик, контракти, бізнес-плани;

– аналіз даних, наприклад, різних станів об'єктів та умов, при яких користувач чи система можуть змінити стани об'єктів.

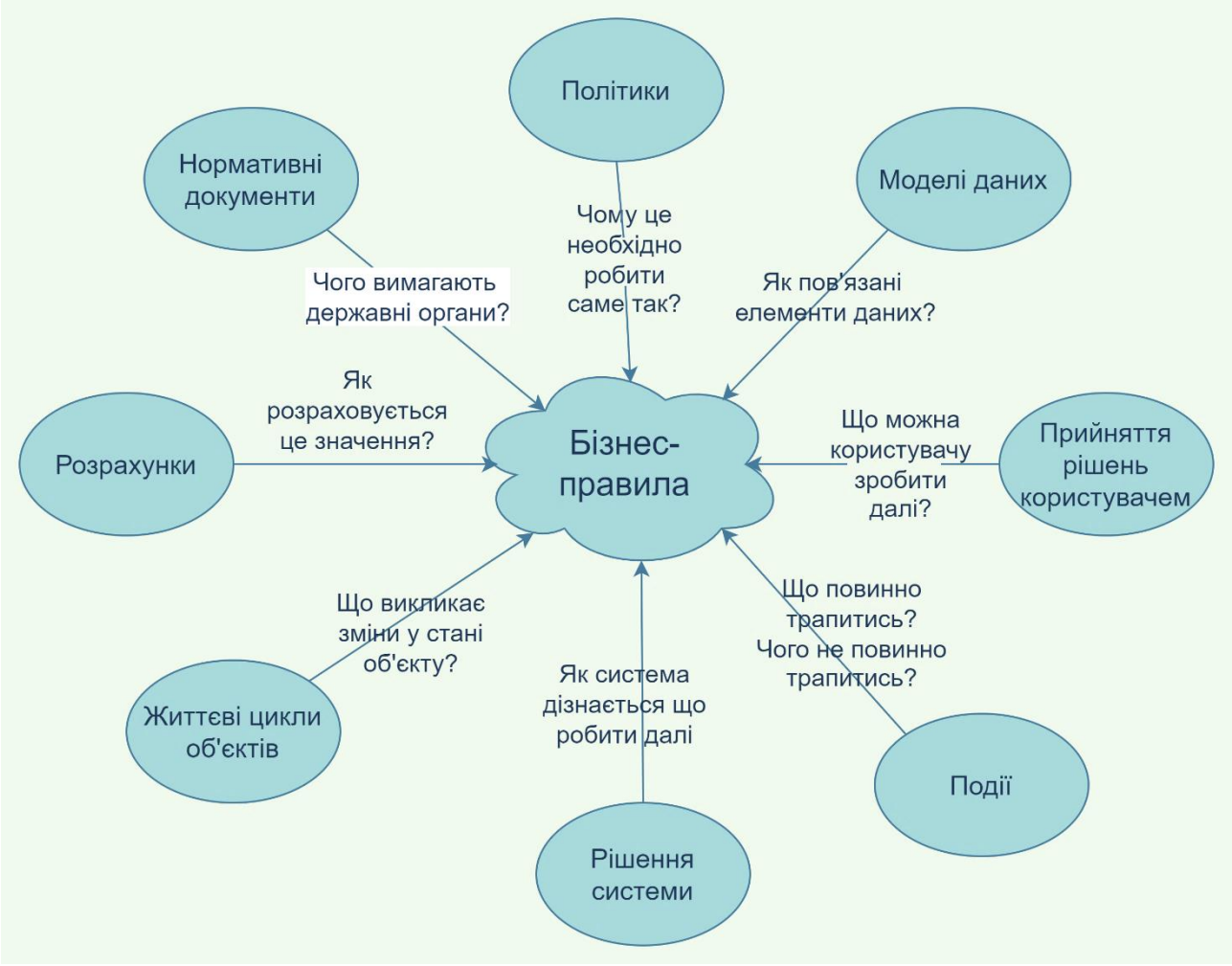


Рис. 3.14. Процес виявлення бізнес-правил

ТЕМА 4. БІЗНЕС-АНАЛІЗ НА ЕТАПІ ПРОЄКТУВАННЯ/ДИЗАЙНУ ТА РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Атрибути якості програмного забезпечення. Побудова прототипів. Rough estimation. MVP, пріоритезація.

Атрибути якості програмного забезпечення

Нефункціональні вимоги складаються з атрибутів якості та вимог до зовнішніх інтерфейсів.

Вимоги до зовнішніх інтерфейсів містять:

1. Користувацькі інтерфейси:

- ✓ стильові рекомендації для родини продуктів;
- ✓ стандарти шрифтів, значків, назв кнопок, зображень, кольорових схем, фірмовий стиль;
- ✓ розмір та конфігурація екрана, обмеження щодо роздільної здатності;
- ✓ сполучення клавіш;
- ✓ стандарти перевірки даних (при вводі);
- ✓ спеціальні можливості для користувачів з вадами зору.

2. Інтерфейси ПЗ:

- ✓ зв'язки ПЗ з іншими застосунками, БД, операційними системами, бібліотеками тощо;
- ✓ формати, зміст повідомлень, контрольні значення даних, що використовуються при обміні інформації.

3. Інтерфейси обладнання:

- ✓ типи пристроїв, що підтримуються;
- ✓ взаємодія даних, протоколи взаємодії;
- ✓ вхідні та вихідні дані, формат, набір даних, діапазони, часові характеристики.

4. Комунікаційні інтерфейси:

- ✓ функції: електронна пошта, веб-браузер, мережеві протоколи, електронні форми;
- ✓ формати повідомлень
- ✓ особливості безпеки, шифрування даних, швидкість передачі даних, механізми синхронізації;
- ✓ обмеження функцій (наприклад, можливість вкладення файлів певних типів при надсиланні повідомлень).

Атрибути якості включають фактори якості, вимоги до якості, вимоги до рівня сервісу. Атрибути якості поділяються на зовнішні (табл. 4.1) та внутрішні (табл. 4.2).

Таблиця 4.1. Зовнішні атрибути якості ПЗ

Зовнішня якість	**Стислий опис. Питання**
Доступність	Наскільки сервіси системи доступні, де та коли вони потрібні. 1. Для яких частин ПЗ критично важливо залишатися доступними? 2. Які бізнес-наслідки недоступності системи для користувачів? 3. Якщо необхідне регулярне обслуговування, то коли його краще проводити? 4. Які повинні надходити сповіщення користувачам, якщо система стала недоступною? 5. Які є залежності доступності між різними функціями системи?
Зручність встановлення	Наскільки просто правильно встановити, видалити та повторно встановити ПЗ. 1. Які установочні операції будуть вимагати перезавантаження ПЗ, комп'ютера? 2. Що повинно ПЗ виконати після вдалої установки? 3. Які операції повинні бути виконані для перевірки вірної установки? 4. Які права та доступи необхідні установнику? 5. Як повинна поводитися система при незавершені установки?
Цілісність	Наскільки добре система захищена від неточності та втрати даних. 1. Як часто необхідно виконувати архівування даних? В яких файлах, БД? 2. Які дані необхідно архівувати, коли архівувати, наскільки довго та з якими вимогами до видалення?
Сумісність	Наскільки просто система може взаємодіяти та обмінюватись даними з іншими системами та компонентами. 1. З якими іншими системами повинна взаємодіяти система? 2. Які стандартні формати даних необхідні для обміну?

	3. Які апаратні компоненти повинні взаємодіяти з системою? 4. Які повідомлення чи коди система повинна отримувати та обробляти від інших систем та пристроїв? 5. Які стандартні протоколи зв'язку необхідні для забезпечення сумісності? 6. Яким обов'язковим вимогам сумісності повинна задовольняти система?
Продуктивність	Як швидко та передбачувано система реагує на введення інформації користувачами та інші події.
Надійність	Як довго система працює до першого збою. 1. Як ви будете визначати надійність системи? 2. Які наслідки збою при виконанні певних операцій? 3. Що ви вважаєте критичною відмовою? 4. Якщо система стає нефункціонуючою, скільки часу ви зможете без неї обійтись? 5. При яких умовах відмова системи може дуже вплинути на бізнес-операції?
Стійкість	Як добре система реагує на неочікувані умови роботи
Захист	Наскільки добре система захищає від пошкоджень. 1. При яких умовах використання цього продукту може зашкодити користувачу? Як система може відслідкувати такі умови? Як повинна реагувати? 2. Які види відмов можуть приносити школу людям чи майну? 3. Які дії оператора можуть принести шкоду людям та майну? 4. Чи є якість режими роботи, які можуть принести ризик для людей та майна?
Безпечність	Як добре система захищає від неправомірного доступу в систему та даним. 1. Які конфеденційні дані необхідно захистити від несанкціонованого доступу? 2. Хто має право переглядати конфіденційну інформацію? Хто не має права? 3. При яких бізнес-умовах уповноважені користувачі мають доступ до функціональності? 4. Які перевірки повинні виконуватися до середовища, в якому працює користувач? 5. Чи повинен використовуватися якийсь особливий метод перевірки дійсності користувача?

Зручність використання	Як легко людям навчитися використовувати систему. 1. Середній час, який необхідний певному типу користувачів, для вірного виконання задачі. 2. Скільки транзакцій може вірно виконати користувач за заданий проміжок часу. 3. Яку частку задач може користувач виконати вірно без додаткової допомоги. 4. Скільки помилок робить користувач при виконанні задач. 5. Скільки спроб необхідно користувачу для виконання задачі. 6. Затримка чи час очікування при виконанні задачі. 7. Кількість операцій, необхідних для отримання потрібної інформації чи виконання задачі.

Таблиця 4.2. Внутрішні атрибути якості ПЗ

Внутрішня якість	**Стислий опис. Питання**
Ефективність	Наскільки ефективно система використовує ресурси комп'ютера. 1. Яка кількість одночасно працюючих користувачів очікується зараз та в майбутньому? 2. Наскільки можуть погіршитися час відклику та інші показники продуктивності, перш ніж це почне серйозно впливати на роботу користувачів та бізнес-операції? 3. Скільки операцій система повинна виконувати одночасно в нормальних та екстремальних умовах?
Можливість модифікації	Наскільки легко обслуговувати, модифікувати, покращувати та реструктурувати систему.
Переносимість	Наскільки легко заставити систему працювати в іншій операційній системі. 1. На яких інших платформах повинно працювати ПЗ зараз та в майбутньому? 2. Які частини ПЗ повинні розроблятися з урахуванням більш високої переносимості у порівнянні з іншими частинами? 3. Які файли, програмні компоненти та інші елементи повинні бути доступні для перенесення? 4. Які атрибути якості можуть потерпати від переносимості?

Можливість повторного використання	В якому ступені компоненти можуть використовуватися в інших системах. 1. Які існуючі вимоги, моделі, компоненти дизайну, дані, тести можна повторно використати в даному додатку? 2. Яка функціональність, що є в зв'язаних додатках, задовольняє деяким вимогам даного ПЗ? 3. Які частини цього ПЗ добре підходять для повторного використання? 4. Які особливі дії необхідно виконати, щоб частини даного ПЗ можна було використати повторно?
Масштабованість	Як добре система впорається зі збільшенням кількості користувачів, транзакцій, серверів та інших розширень. 1. Як та чому у майбутньому може зрости необхідність в потужностях зберігання даних? 2. Які мінімально допустимі критерії продуктивності повинні бути дотримані незалежно від кількості числа користувачів? 3. Які плани майбутнього росту серверів, центрів обробки даних, кількості встановлених екземплярів системи? 4. Яку максимальну кількість користувачів повинна обслуговувати система через декілька місяців, кварталів, років?
Здатність до перевірки та тестування	Як швидко розробники та тестувальники можуть підтвердити, що система реалізована вірно. 1. Як можна підтвердити, що певні розрахунки дають очікувані результати? 2. Чи є частини системи, що недерміновані, тому складно перевірити чи вірно вони працюють? 3. Чи можливо створити набір тестових даних, які з високою ймовірністю дозволять виявити які-небуть помилки у вимогах чи їх реалізації? 4. Які стандартні звіти можна використовувати для перевірки того, що система повертає вірні результати?

Формулюючи вимоги до атрибутів якості, бізнес-аналітик повинен вміти знаходити компроміси між різними атрибутами. Для цього можна використати наступну матрицю (рис. 4.1).

	Доступність	Ефективність	Зручність встановлення	Цілісність	Сумісність	Можливість модифікації	Продуктивність	Переносимість	Надійність	Повторне використання	Стійкість	Захист	Масштабованість	Безпека	Зручність використання	Здатність до перевірки та тестування
Доступність									+		+					
Ефективність	+				-	-	+	-			-		+		-	
Зручність встановлення	+								+					+		
Цілісність			-		-		-			-		+		+	-	-
Сумісність	+		-	-			-	+	+		+	-		-		
Можливість модифікації	+		-				-		+	+			+			+
Продуктивність		+			-	-		-			-		-		-	
Переносимість		-			+	-	-			+				-	-	+
Надійність	+	-		+		+	-				+	+		+	+	+
Повторне використання		-		-	+	+	-	+						-		+
Стійкість	+	-	+	+	+		-		+			+	+	+	+	
Захист		-		+	+		-				+			+	-	-
Масштабованість	+	+		+			+	+	+		+					
Безпека	+			+	+		-	-	+		+	+			-	-
Зручність використання		-	+				-	-	+		+	+				-
Здатність до перевірки та тестування	+		+	+		+			+	+	+	+		+	+	

Рис. 4.1. Матриця компромісів атрибутів якості ПЗ («-» - збільшення величини атрибуту в рядку негативно впливає на атрибут в стовпчику)

Побудова прототипів

Прототип ПЗ – часткове, можливе та попереднє втілення нового продукту.

Виділяють наступні класи атрибутів:

– за призначенням:

Модель – демонстрація користувацького інтерфейсу.

Експериментальний зразок (для перевірки технічних рішень);

– за можливістю використання у майбутньому:

Одноразовий прототип.

Еволюційний прототип;

– за формою:

Паперовий прототип;

Робочий (електронний) прототип.

Модель (mock-up) – горизонтальний прототип (horizontal prototype).

Особливості:

– не містить реальної функціональності;

– демонструє зовнішній вигляд, поведінку користувача, доступ до інформації (навігація);

– реалізація деяких варіантів використання.

Моделі дозволяють: виявити упущення, невірні та зайві функції, допомагають виявити альтернативні напрямки, варіантів використання, пропущені кроки взаємодії, додаткові виключення, пропущенні вихідні умови та бізнес-правила.

Експериментальна модель (proof of concept) – вертикальний прототип (vertical prototype) – спайк (spike) в Agile; це майже реальна система, але побудована без дотримання вимог до якості коду. Використовується при сумнівах в технічній реалізації певних вимог, при необхідності оптимізації алгоритмів. Будуються в середовищі, аналогічному середовищу розробки.

Одноразовий прототип (throwaway prototype) – прототип, що не випускається (nonreleasable prototype).

Особливості:

– максимально дешево та швидко;

– переваги надаються модифікації, а не стійкості, надійності;

– можна використовувати повторно, але неможна використовувати в реальному проєкті;

– будується, коли є двозначність чи незрозумілість вимог, коли є складності з візуальною подачею інформації;

– допомагає виявити пробіли в документації, чи виявлені/описані вимоги достатні для реалізації бізнес-процесів.

Каркаси (wireframe) – форма одноразового прототипу, що використовується для дизайну нестандартних користувацьких інтерфейсів.

Використовуються для візуалізації:

– концептуальних вимог;

– дизайну навігації;

– дрібного дизайну сторінок.

Еволюційний прототип (evolutionary prototype) – фундамент для створення кінцевого продукту. Використовується в Agile. Код будується з пріоритетом на якість. Вимагає більше часу, ніж одноразовий прототип з

однаковим функціоналом. Перевага – користувач швидко отримує діючий зразок.

Паперовий прототип (paper prototype) - низькоякісний прототип. Дозволяє зрозуміти однозначне тлумачення вимог розробниками та замовником. Демонструє можливий інтерфейс продукту. Використовується для вивчення функціональності та потоків.

Електронний прототип (digital prototype) заснований на використанні мов програмування високого рівня абстракції, таких як Java, Perl, Python, Haskell і т.п.

Варіації використання прототипів наведені у таблиці 4.1.

Таблиця 4.1. Варіації використання прототипів ПЗ

	Одноразовий	**Еволюційний**
Модель	Пояснення та уточнення користувацьких та функціональних вимог. Виявлення пропущеної функціональності. Дослідження можливих варіантів користувацького інтерфейсу	Реалізація базових користувацьких вимог. Реалізація додаткових користувацьких вимог за пріоритетами. Реалізація та доробка сайтів. Адаптація системи до вимог бізнесу
Експериментальний зразок	Демонстрація технічної здійсненності. Оцінка продуктивності. Отримання інформації для уточнення оцінок	Реалізація та нарощування важливої багаторівневої функціональності та рівнів комунікації. Реалізація та оптимізація основних алгоритмів. Тестування та налаштування продуктивності

Можна навести наступний алгоритм використання прототипів при проєктуванні програмного забезпечення (рис. 4.1):

Рис. 4.1. Алгоритм використання прототипів

Організувати роботу з прототипом можна наступним чином (рис. 4.2, 4.3):

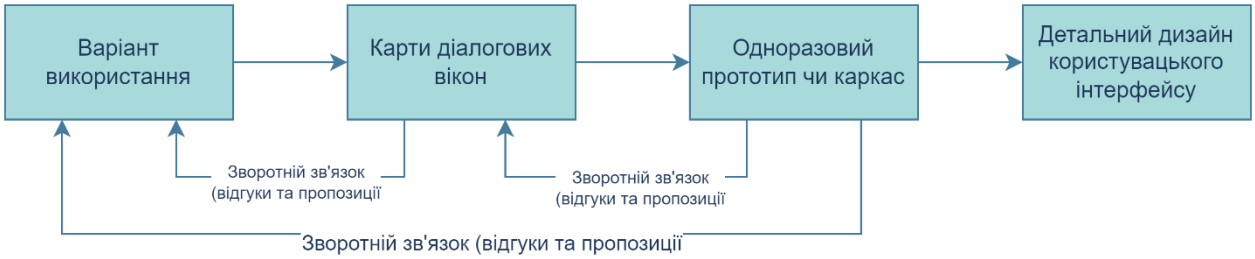


Рис. 4.2. Процес роботи з прототипом

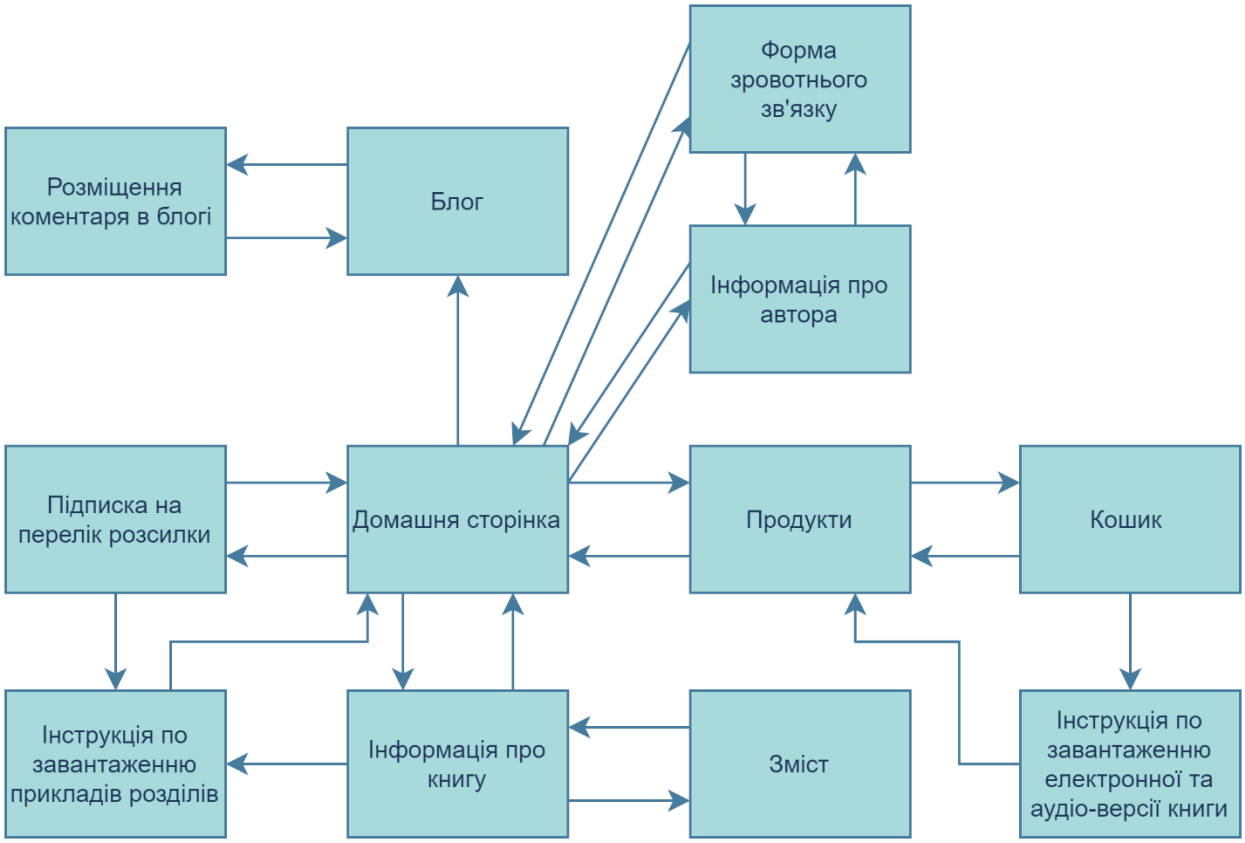


Рис. 4.3. Приклад карти діалогових вікон

Продумане використання та правильна реалізація прототипів є цінним інструментом виявлення та уточнення вимог при перетворені потреб у рішення.

Правила створення прототипів:
– включення задачі зі створення прототипу в план проєкту;
– формулювання мети кожного прототипу;
– плануйте створення декількох прототипів;
– одноразові прототипи створюйте максимально швидко та дешево;

– не вбудовуйте в одноразові прототипи перевірку вхідних даних, методи безпечного програмування, обробку помилок в коді;

– не створюйте прототипи того, що вже зрозуміло;

– використовуйте лише правдоподібні дані для дашбордів, форм, повідомлень тощо;

– не очікуйте, що прототип повністю замінить специфікацію вимог до ПЗ.

Питання, які доречно задати замовнику на етапі узгодження прототипу:

Реалізує чи ні прототип всі необхідні функції, які Ви очікували?

Чи не пропустили деяку функціональність у прототипі?

Побачили Ви чи ні якісь помилки? Що не враховує прототип?

Чи не має прототип непотрібних функцій?

Наскільки зрозумілою та логічною є навігація?

За Вашою думкою, чи є логічні ланцюжки, які краще спростити?

При роботі з прототипом чи було таке, щоб Ви не розуміли що саме робити?

Приклад прототипу наведений на рис. 4.4.

Рис. 4.4.1. Приклад прототипу (Power Point)

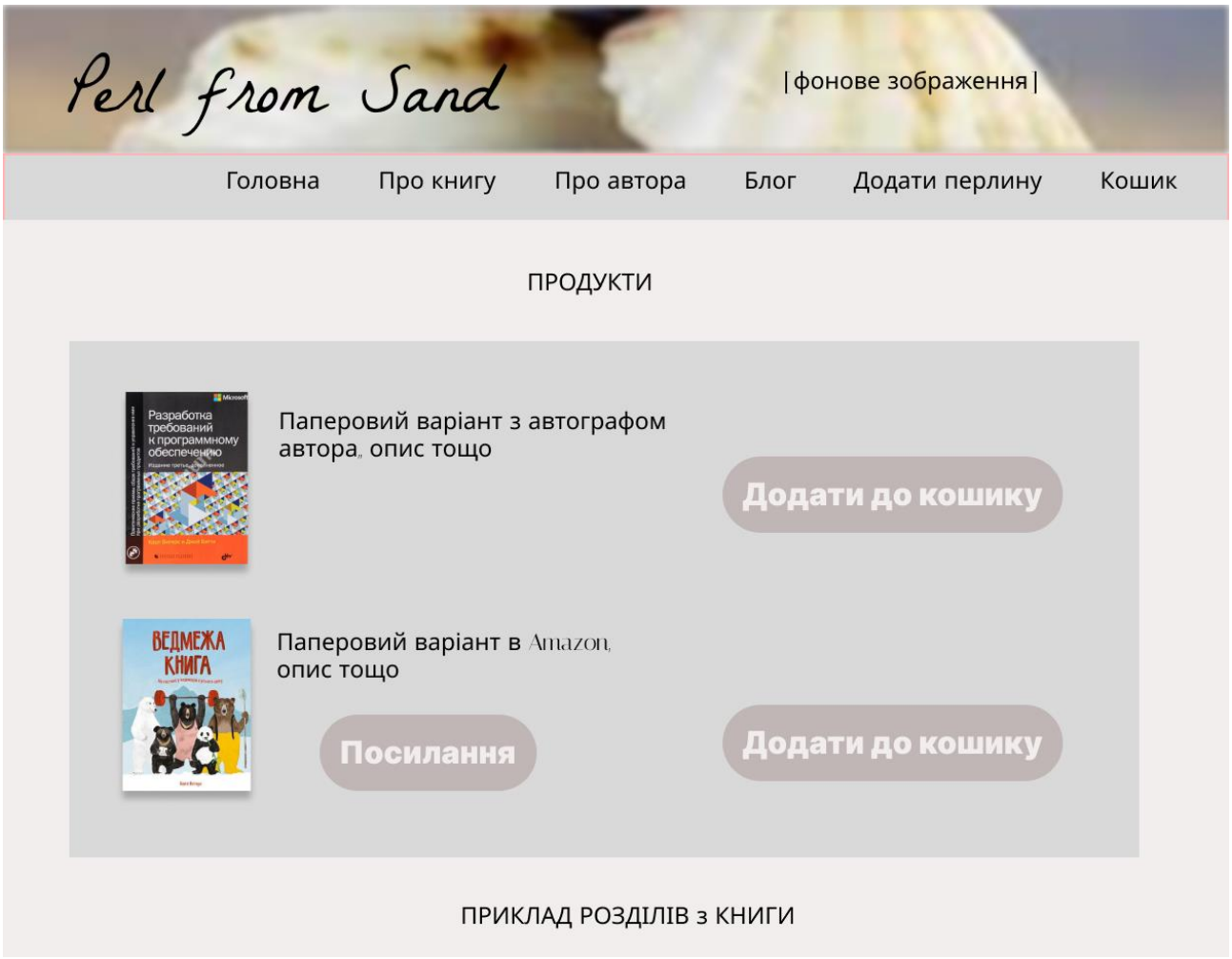


Рис. 4.4.2. Приклад прототипу (Figma)

Rough estimation. MVP

Rough estimation (груба оцінка) в управлінні проєктами – це процес надання швидкої та приблизної оцінки зусиль, часу, ресурсів та вартості, необхідних для виконання проєкту досягнення його основних цілей. Це зроблено на початковій стадії проєкту, коли інформація про нього обмежена, а детальна оцінка ще не проведена.

Основна мета грубої оцінки – з'ясувати, чи варто продовжувати більш докладний процес оцінки, який може бути більш часо- та ресурсо-витратним. Це також допомагає учасникам проєкту отримати загальне уявлення про масштаби та складність проєкту.

Необхідно враховувати, що груба оцінка може бути неточною, оскільки базується на обмеженій інформації та швидкому аналізі. При розвитку проєкту, коли отримуються більше деталей, оцінка може змінюватись. Тому, після грубої оцінки, виконується детальна оцінка, щоб отримати більш точні результати.

Існує кілька методів для здійснення грубої оцінки проєкту, такі як експертні оцінки, метод "три кільця", порівняльний аналіз з аналогічними проєктами та інші. Важливо пам'ятати, що груба оцінка є лише початковим етапом управління проєктом, і вона повинна бути уточнена під час подальшого планування та виконання проєкту.

MVP (Minimum Viable Product) – це стратегічний підхід до розробки продукту, програмного забезпечення або сервісу, який дозволяє створити мінімальний функціонал, необхідний для того, щоб продукт був життєздатним та міг задовольнити основні потреби користувачів. MVP дозволяє швидко вивчити реакцію ринку на продукт та перевірити гіпотези щодо його успішності, зменшуючи витрати та ризики.

Основні характеристики MVP:

1. Мінімальний функціонал. MVP включає лише необхідний мінімальний набір функцій, який дозволяє продукту працювати та бути придатним для використання. Ці функції часто визначаються на основі грубої оцінки та пріоритезації.

2. Швидкий випуск. Головною метою MVP є швидкий випуск продукту на ринок. Це дозволяє зв'язатися з користувачами та швидко отримати зворотній зв’язок, щоб зрозуміти, як вони реагують на продукт та які можливості можуть бути вдосконалені або додані.

3. Експериментування. MVP сприяє експериментуванню та знаходженню оптимальних рішень. Команда проєкту може перевіряти різні гіпотези та ідеї, які допоможуть покращити продукт та забезпечити його більшу цінність для користувачів.

4. Зосередження на ядрі. MVP допомагає зосередитись на ядрі продукту, тобто на його основних функціях та особливостях, які є найважливішими для користувачів. Це дозволяє уникнути зайвої складності та перенасиченості продукту.

5. Поступове вдосконалення. Залежно від отриманого фідбеку та аналізу реакції ринку, MVP постійно вдосконалюється та розширюється. Нові функції та можливості додаються після збору додаткових даних про користувачів та їх потреби.

6. MVP дозволяє зекономити час і ресурси, орієнтуючись на найбільш важливі аспекти продукту та швидке залучення користувачів. Це часто використовується стартапами, але також може бути корисним методом розробки для великих компаній, які прагнуть прискорити введення продукту на ринок та встановити зв'язок зі своїми клієнтами.

Перехід від грубої оцінки (Rough estimation) до MVP (Minimum Viable Product) – це важливий етап в розробці проєкту, особливо в сфері програмного забезпечення та стартапів.

Базові кроки, які можуть допомогти перейти від грубої оцінки до MVP:

1. Вибір ключових функцій. На основі грубої оцінки ідентифікуються ключові функції або основні властивості продукту, які є необхідними для досягнення основних цілей проєкту та забезпечення його мінімальної придатності для використання.

2. Пріоритезація функцій. Визначається пріоритетність цих ключових функцій для створення MVP. Важливо обрати ті функції, які дозволять продукту працювати, але при цьому зменшать обсяг розробки та терміни впровадження.

3. Дизайн та розробка MVP. Розробляється MVP з використанням обраних ключових функцій. MVP повинен бути досить функціональним, щоб здійснити мінімальний придатність до використання та дослідження реакції ринку.

4. Тестування та збір фідбеку. Тестування MVP серед малої групи користувачів або цільової аудиторії. Збір зворотного зв’язку від користувачів про продукт, їх потреби та проблеми, які вони зустрічають.

5. Аналіз результатів. Проаналізувати зібраний фідбек та вивчити реакцію ринку на MVP. Це допоможе з'ясувати, чи вдалося вам відповісти на потреби користувачів та чи необхідні зміни для подальшого вдосконалення продукту.

6. Вдосконалення продукту. Враховуючи отриманий фідбек, здійснюються необхідні вдосконалення продукту, додаються нові функції та виправляються помилки. Поступово розширюється функціонал продукту на основі вимог користувачів та реакції ринку.

7. Поступовий реліз. Запускаються поступові версії продукту, додаючи новий функціонал та враховуючи реакцію користувачів. Цей процес може повторюватись декілька разів, доки продукт не стане повноцінним та задовольнить потреби ринку.

Перехід від грубої оцінки до MVP допомагає прискорити розробку продукту та зменшити ризики, пов'язані з впровадженням повного функціоналу без вивчення реакції ринку. MVP дозволяє команді проєкту зосередитись на важливих аспектах та забезпечити мінімальний життєздатність продукту, що робить його придатним для використання та взаємодії з потенційними користувачами.

Приклад складання MVP наведений за посиланням:
https://miro.com/app/board/uXjVMS-0LiA=/?share_link_id=666205745192

Пріоритезація

При управлінні пріоритетами вимог доцільно дотримуватися правила трикутника PM:

Рис. 4.5. Критерії управління пріоритетами вимог

При пріоритезації вимог необхідно керуватися:

– потребами клієнтів;

– відносною важливістю вимог для клієнтів;

– послідовність, в якій повинні передаватися функції;

– вимогами, що є основою для інших вимог, інших зав'язків між вимогами;

– потребами в реалізації певних груп вимог разом;

– витратами на задоволення кожної вимоги.

Бізнес-аналітик при визначені пріоритетів вимог може (та повинен) звернутися до зацікавлених сторін з такими можливими питаннями:

Чи є інший спосіб задовільнити цю вимогу клієнтів?

Що станеться, якщо цю вимогу прибрати чи відкласти?

Що відбудеться з бізнес-вимогами проєкту, якщо ця вимога не буде реалізована в найближчі місяці?

Чому користувачі будуть незадоволені, якщо реалізація цієї вимоги буде відкладена до наступного випуску?

Чи варто з-за цієї функції відкладати випуск всіх інших функцій з тим самим пріоритетом?

Якщо після пріоритезації Ви отримали приблизно однакові пріоритети вимог, то Ви не виконали ніякої пріоритезації.

Методи пріоритезації вимог

Методика «100 доларів». Кожному учаснику команди надається перелік вимог. Він має уявні 100 доларів (або 100 балів) і самостійно вирішує, скільки він готовий «заплатити» (дати балів) за ту чи іншу вимогу. Логіка полягає в тому, що складна задача і «коштує» відповідно. В результаті вимоги з більшою кількістю балів мають вищий пріоритет, тому починати роботу потрібно з них.

Визначення пріоритетів на основі цінності, вартості та ризику.

В процедурі приймають участь: менеджер проєкта, бізнес-аналітик, представники клієнта – прихильники продукту, власник продукту, маркетологи, представники розробників.

Алгоритм методу:

1. Навести всі функції/варіанти використання/користувацькі історії. Дотримуватися одного рівня абстрагування.

2. Запросити клієнтів оцінити кожну функцію за шкалою: 1-9 (1 – некорисна; 9 – дуже цінна).

3. Оцінити відносні втрати, що може зазнати замовник , якщо функція не буде реалізована, за шкалою: 1-9 (1 – не помітять її зникнення; 9 – виключення дуже зашкодить бізнесу).

Питання щодо втрат:

Чи програє Ваш продукт у порівнянні з іншим ПЗ, де є така функція?

Чи будуть якісь юридичні чи контрактні наслідки?

Чи не порушите Ви якусь норму права чи галузевий стандарт?

Чи не втратять користувачі функціонал, необхідний їм для вирішення очікуваних задач?

Чи буде складно реалізувати цю функцію пізніше в межах розвитку системи?

Чи виникнуть проблеми, якщо відділ маркетингу вже пообіцяв цю функцію деяким клієнтам?

4. Визначити % цінності, втрат, ризику від суми балів щодо оцінюваного в таблиці функціоналу (а не всього набору функцій).

5. Запросити розробників оцінити відносну вартість реалізації кожної функції за шкалою: 1-9 (1 – легко та швидко; 9 – трудомістко та коштовно).

Вартість визначається на основі складності, невизначеності, можливості повторного використання коду, обсягу тестування тощо.

6. Аналогічно розробники оцінюють рівень технічного ризику за шкалою 1-9 (1 – легко запрограмуєте; 9 – стурбовані чи взагалі можливо реалізувати, відсутність відповідних кадрів, використання незнайомих технологій тощо).

Розрахувати пріоритет:

$$\text{Пріоритет} = \frac{\text{Цінність, \%}}{\text{К}_1 \cdot \text{Вартість\%} + \text{К}_2 \cdot \text{Ризик\%}}.$$

7. Відсортувати перелік за значенням Пріоритету: за зменшенням значення. Найпріоритетніші функції – добре поєднання цінності, вартості та ризику.

8. При необхідності залучити декілька представників клієнта до оцінювання, встановити важливість їх думки (пріоритети) та розрахувати загальний Пріоритет.

Шаблон таблиці розрахунків має вигляд:

Таблиця 4.3. Визначення пріоритетів на основі цінності, вартості та ризику

Відносна вага	**2**	**1**			**1**		**0,5**		
Функція	**Відносна цінність**	**Відносні втрати**	**Загальна цінність**	**Цінність, %**	**Відносна вартість**	**Вартість, %**	**Відносний ризик**	**Ризик,%**	**Пріоритет**

Приклад застосування методу наведений в таблиці 4.4.

Таблиця 4.4. Приклад застосування методу визначення пріоритетів

Відносна вага	2	1			1		0,5		
Функція	Відносна цінність	Відносні втрати	Загальна цінність	Цінність, %	Відносна вартість	Вартість, %	Відносний ризик	Ризик,%	Пріоритет
Друк переліку матеріалів	2	4	8	5,2%	1	2,7%	1	3,0%	1,22
Запит про статус замовлення	5	3	13	8,4%	2	5,4%	1	3,0%	1,21
Створення звіту з інвентаризації на складі	9	7	25	16,1%	5	13,5%	3	9,1%	0,89
Перегляд історії певного контейнера	5	5	15	9,7%	3	8,1%	2	6,1%	0,87
Пошук хімікату за каталогом	9	8	26	16,8%	3	8,1%	8	24,2%	0,83
Підтримка переліку небезпечних хімікатів	3	9	15	9,7%	3	8,1%	4	12,1%	0,68

Зміна поточних замовлень на хімікати	4	3	11	7,1%	3	8,1%	2	6,1%	0,64
Створення звітів з інвентаризації окремих лабораторій	6	2	14	9,0%	4	10,8%	3	9,1%	0,59
Перевірка в БД з навчання інформації щодо навчання	3	4	10	6,5%	4	10,8%	2	6,1%	0,47
Імпорт хімічних структур з графічних редакторів	7	4	18	11,6%	9	24,3%	7	21,2%	0,33
Всього	**53**	**49**	**155**	**100%**	**37**	**100%**	**33**	**100%**	

Матриця Ейзенхаурера

Матриця Ейзенхауера допомагає зрозуміти і визначити пріоритетність кожної вимоги залежно від двох основних факторів: важливості та складності.

Матриця включає дві осі:

Ось важливості (Importance Axis) – шкала, яка вказує, наскільки важливою є вимога для користувача, замовника або бізнесу. Важливість може визначатися на основі таких критеріїв, як стратегічне значення, вплив на користувачів, конкурентні переваги тощо. Зазвичай, важливість оцінюється від 1 до 5 або від 1 до 10, де більше значення відповідає більшій важливості.

Ось складності (Complexity Axis) – відображає технічну складність або затрати, пов'язані з реалізацією вимоги. Враховують таки чинники, як ресурси (час, гроші, трудові ресурси), складність реалізації, необхідність в додатковому дослідженні або розробці нових технологій тощо. Складність також оцінюється за шкалою від 1 до 5 або від 1 до 10, де більші значення відповідають більшій складності.

Кожна вимога або задача представляється у вигляді точки на цій двовимірній матриці, де ось важливості відповідає одній координаті, а ось складності – іншій координаті. Чим ближче точка до верхнього лівого кута матриці (висока важливість і низька складність), тим вищий пріоритет має ця вимога (рис. 4.6).

Ця матриця дозволяє команді проєкту зрозуміти, які вимоги або задачі мають найвищий пріоритет і повинні бути виконані в першу чергу. Також вона допомагає збалансувати пріоритети, враховуючи важливість і складність кожної вимоги, щоб досягти максимального результату при обмежених ресурсах.

Оцінка важливості і складності може виконуватись експертно командою проекту або за допомогою аналітичних методів, таких як опитування зацікавлених сторін або проведення оцінки вимог у груповій дискусії.

Рис. 4.6. Матриця Ейзенхаурера

Багатоетапна пріоритезація

Багатоетапна пріоритезація вимог – це процес розподілу пріоритетів між вимогами або задачами у проєкті шляхом декомпозиції та розгляду їх на декількох рівнях.

Дерево декомпозиції пріоритетів наведено на рис. 4.7.

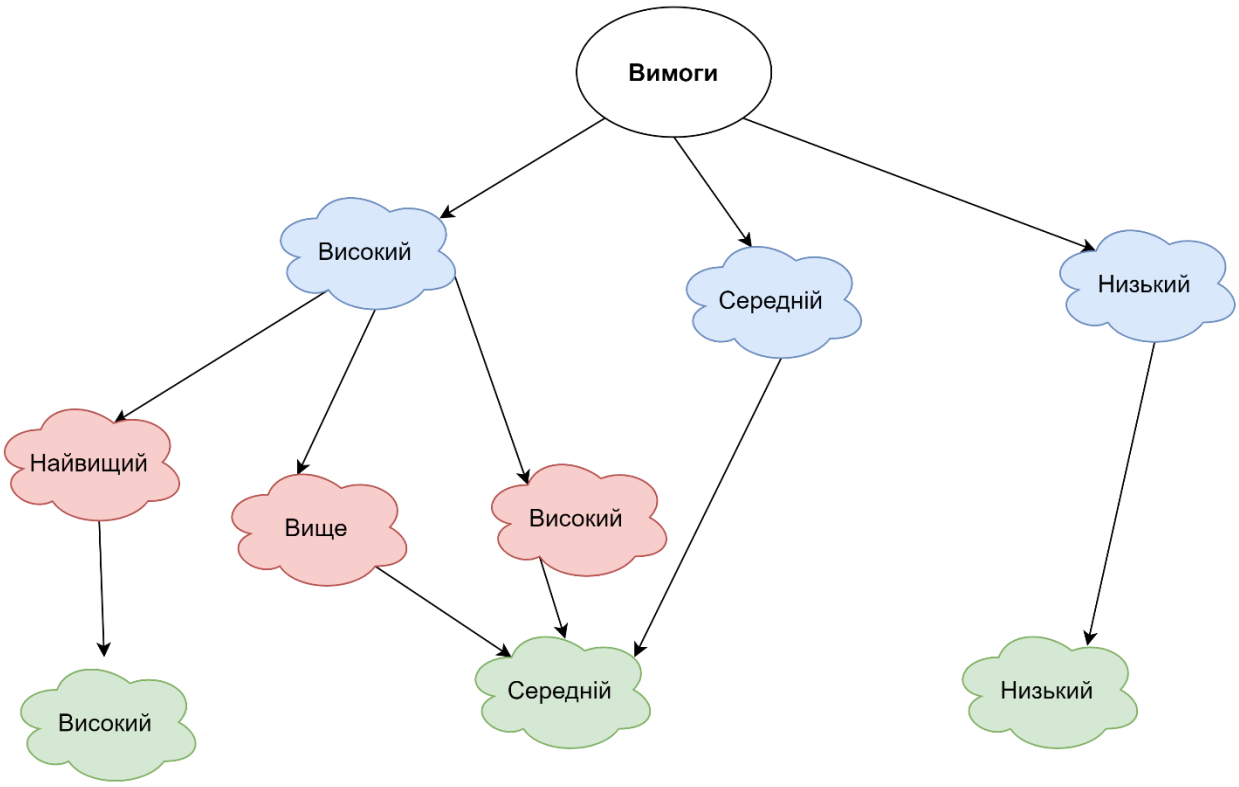


Рис. 4.7. Приклад дерева декомпозиції пріоритетів

Багатоетапна пріоритезація може включати наступні кроки:

1. Ідентифікація вимог – збір всіх вимог і задач, які необхідно розглянути в проєкті (вимоги користувачів, бізнесу, технічні вимоги та інші елементи).

2. Групування вимог за схожими характеристиками або за пов'язаністю з певними аспектами проекту.

3. Первинна пріоритезація – узагальнена оцінка важливості та складності кожної групи вимог.

4. Детальна пріоритезація – більш детальний розгляд вимог на рівні окремих етапів або фаз проєкту. На цьому етапі можуть братись до уваги конкретні критерії, вплив на користувачів, вартість реалізації, технічна складність тощо.

5. Порівняння і пріоритизація. Кожну вимогу можна порівняти з іншими на одному етапі або з усіма вимогами на загальному рівні. Застосування матриць пріоритетів, SWOT-аналізу або інших методів допоможе визначити рейтинг пріоритетів для кожної вимоги.

Техніка ABCDE

Група A – вимоги, без виконання яких проєкт не буде завершений.

Група B – вимоги, які бажано виконати, оскільки невиконання цих пунктів також матиме негативні наслідки, але не такі серйозні, як для вимог групи A.

Група C – вимоги, які було б непогано виконати, оскільки вони нададуть кінцевому продукту кращий вигляд. При цьому їх невиконання не має критично серйозних наслідків.

Група D – вимоги, які можуть бути необов'язковими, нетерміновими і не представляють великого інтересу для проєкту.

Група E – вимоги, які слід видалити з поточної ітерації або проєкту взагалі. Їх впровадження не впливає на кінцевий продукт, але потребує часу і ресурсів.

ТЕМА 5. ДОКУМЕНТУВАННЯ ПРОТЯГОМ ЖИТТЄВОГО ЦИКЛУ ІТ ПРОЄКТУ

Проєктна документація. Беклог продукту. WBS, milestone, User Story. Change log.

Протягом життєвого циклу ІТ-проєкту формуються різноманітні документи, які допомагають забезпечити ефективне планування, виконання і контроль за проєктом. Основні документи, що можуть бути сформовані протягом різних стадій життєвого циклу ІТ-проєкту, включають:

– на стадії ініціації (Initiation):

1. Документ проєктних ініціатив (Project Charter) – документ, що затверджує і посилює дослідження проєкту, місію, цілі, сторони, ресурси, ризики та структуру управління.

2. Документ концепції та меж проєкту (Vision and Scope Document) служить для чіткого визначення мети проєкту, його масштабу, цілей та очікувань від проекту.

– на стадії планування (Planning):

1. План управління проектом (Project Management Plan) – документ, що встановлює якість, механізм, процеси і критерії прийняття рішень для виконання проєкту.

2. Опис вимог (Requirements Document) – опис потреб та вимог замовника щодо функціональності та характеристик проєкту.

3. Розклад проекту (Project Schedule) – графік робіт і завдань для досягнення цілей проекту.

– на етапі виконання (Execution):

1. Документація програмного забезпечення (Software Documentation) – технічна документація, яка описує архітектуру, функціональність, конфігурації, процеси розробки та реалізації програмного забезпечення.

2. Документація для тестування (Testing Documentation) – документи, що описують тестовий план, тестові випробування, звіти про тестування та результати.

– на етапі впровадження (Delivery):

1. Документація з експлуатації (User Manuals) – документація для кінцевих користувачів, яка надає інструкції щодо використання програмного забезпечення або системи.

2. Документація для супроводу та підтримки (Support and Maintenance Documentation) – документація, що включає інструкції з супроводу, підтримки, усунення неполадок та оновлення.

– *на етапі завершення (Closure):*

1. Звіт за проєктом (Project Closure Report) – документ, що містить інформацію про досягнення цілей, управління ризиками та ресурсами, а також рекомендації для майбутніх проєктів.

2. Підсумки (Lessons Learned) – документ, що містить вивчені «уроки», досягнення та пропозиції щодо покращення.

Важливо зазначити, що тип і кількість документів можуть змінюватись залежно від розміру та складності проєкту, його типу та специфічних вимог замовника. Однак, завжди важливо вести належну документацію протягом усього життєвого циклу проєкту, щоб забезпечити успішне виконання, контроль і передачу проєкту після завершення.

Product backlog

Product Backlog (список продукту) є основним інструментом управління вимогами у методологіях Agile, зокрема Scrum. Це централізований і постійно оновлюваний список всіх вимог та функціональності, які потрібно реалізувати у проєкті.

Основні характеристики Product Backlog:

– містить усі вимоги, функціональність та задачі, які плануються реалізувати у продукті. Вимоги можуть бути представлені у вигляді коротких описів або user stories;

– вимоги упорядковані за пріоритетами, починаючи з найбільш важливих. Пріоритет визначається замовником або власником продукту і враховує важливість та стратегічні потреби бізнесу;

– постійно оновлюється та змінюється протягом життєвого циклу проєкту. Нові вимоги можуть бути додані, а існуючі вимоги можуть змінюватись або вилучатись, щоб відповідати потребам і змінам у проєкті;

– кожній вимозі може бути призначено оцінку, яка визначає складність і затрати на реалізацію. Це допомагає команді проекту упорядкувати роботу та планувати ітерації (спрінти) у Scrum;

– великі та складні вимоги можуть бути розбиті на менші елементи, що дозволяє команді реалізувати їх крок за кроком упродовж проєкту;

– Product Backlog керує власник продукту (Product Owner), який відповідає за призначення пріоритетів і забезпечення того, щоб Product Backlog відображав бажаність замовника і відповідав стратегічним цілям продукту.

Product Backlog є динамічним інструментом, що допомагає забезпечити зростання та постійну орієнтацію на потреби замовника. Він

є вихідним пунктом для планування ітерацій, а також встановлює робочий порядок розробки та пріоритетність функціональності, що забезпечує більшу гнучкість та адаптивність у розробці продукту.

WBS (work breakdown structure) проєкту

WBS розробляється на етапі планування проєкту. Планування починається від моменту запуску проєкту і триває аж до завершальних стадій. Це багаторазові процедури, реалізовані на кожній фазі розв'язання проєктної задачі. Метою є детальна розробка змісту, плану дій з управління проєктом і складання календарного розкладу робіт.

Планування проєктів передбачає розробку WBS (декомпозиції проєкту) та Network diagram (пошук залежностей між активностями проєкту); оцінювання тривалості та вартості проєкту, планування ресурсів.

Результатом цього етапу стають project schedule, project budget & project scope.

Декомпозиція – це метод, за допомогою якого розбивають project scope і project deliverables на дрібніші елементи, якими легше керувати.

Work package – це елементи робіт, розташовані на найнижчому рівні WBS, для якого можлива оцінка cost and duration, а також управління ними.

Алгоритм побудови **WBS***:*

1. Проаналізувати всі задачі, що входять у скоуп, чи всі вони там враховані, чи правильне їх розуміння.

2. Організувати рівні розбиття. Поміркувати, що буде на головному рівні, а що на дочірніх.

3. Розбити верхні рівні на нижні.

4. Перевірити, чи достатня декомпозиція. Для цього ставиться питання *що*? Якщо далі при розбитті йде питання *як*? – достатній рівень декомпозиції.

Нижчі рівні WBS повинні згортатися на вищі рівні, щоб нічого не було пропущено і щоб не виконувалася зайва робота. Іноді це називають «правилом 100%».

Варто пам'ятати:

Cost. Елементи WBS не містять costs.

Importance. Елементи WBS не вказують на важливість (пріоритетність).

Levels of decomposition. WBS не має обмежень за кількістю рівнів декомпозиції.

Relationships. WBS не містить взаємозв'язків елементів.

Resources. Елементи WBS не показують призначені ресурси. Це просто елементи.

Time. WBS не містить тривалості та послідовності елементів.

***WBS** checklist:*

Чи визначає WBS 100% роботи, яка має бути виконана на проєкті? Тобто це мають бути не лише продукти після імплементації, а й інші активності, наприклад, важлива документація (PM Plans).

Чи кожний елемент є результатом (deliverables)? Чи не потрапили туди активності (питання *Як?*)?

Чи зможуть стейкхолдери зрозуміти сферу проєкту із WBS? Тобто кожна людина, яка залучена до проєкту, зрозуміє кожен пункт WBS?

Чи охоплює WBS всі зовнішні та внутрішні результати, включаючи результати управління проєктами?

Чи кожен рівень становить 100% роботи, необхідної для отримання батьківського рівня?

Чи достатньо декомпозиції, щоб прописати активності до кожного work package?

Чи використовується ієрархічна структура?

Чи WBS створили ті, хто буде виконувати роботу? Якщо відповідь ні, хоча б ознайомте з результатом команду.

Чи регулярно оновлюється WBS після затвердження проєкту?

Чи можете ви чітко визначити критерії прийняття кожного work package? Definition of Done, або Acceptance Criteria.

Чи дає змогу WBS точно оцінити витрати?

Чи не є WBS занадто громіздкою або навпаки?

Network diagram

WBS не містить взаємозалежностей чи зв’язків, але ж їх треба врахувати, тому що певна частина робіт має вищий пріоритет. А ще без результатів певних робіт неможливо почати виконання наступних.

У Network diagram показано управління проєктами, що потрібно зробити і в якому порядку.

У термінології управління проєктами **Network diagram** – це «логічне подання завдань, що визначають послідовність роботи над проєктом».

Діаграма може містити дати початку та кінця, назви завдань, ім’я людини, відповідальної за виконання кожного завдання.

Діаграма повинна з першого погляду показати, що має відбутися і в якому порядку.

Network diagram корисні, оскільки:

✓ Показують послідовність завдань і залежності між ними.

✓ Допомагають виявити основні етапи проєкту, які потім використовують для моніторингу та контролю.

✓ Демонструють, як проєкт може бути реалізований.

✓ Дають змогу планувати проєкт відповідно до наявного часу та ресурсів.

Milestone

Milestone (віхи) – це основні маркери, які позначають основні досягнення, ключові події чи завершення ключових результатів у рамках графіка проєкту.

Основні характеристики:

– віхи мають визначені часові рамки, протягом яких очікується досягнення певної події або стадії, допомагають встановити терміни для критичних подій у проєкті;

– кожна віха пов'язана з досягненням певних ключових результатів або виконанням критичних завдань у проєкті, допомагають оцінити прогрес і успішність проєкту на різних етапах;

– віхи служать контрольними пунктами для визначення, чи виконується проєкт згідно з планом, дозволяють ідентифікувати проблеми чи затримки у проєкті та приймати вчасні коректуючі заходи;

– віхи можуть відзначати завершення окремих фаз проєкту, що допомагає управлінню проєктом і забезпечує раціональне використання ресурсів на кожній стадії;

– віхи є важливим інструментом комунікації між всіма учасниками проєкту, такими як команда, замовник, менеджмент, а також інші зацікавлені сторони.

Віхи часто представляються у вигляді точок на графіку прогресу проєкту, де їх досягнення вказує на переміщення до наступного етапу чи успішне виконання ключових результатів.

Change log

Change log – журнал або запис усіх помітних змін, внесених у проєкті, програмному забезпеченні, документах або будь-якому іншому продукті протягом його життєвого циклу.

Основні характеристики:

– реєструються всі зміни, що впливають на продукт або проєкт. Це можуть бути зміни вимог, дизайну, функціональності, програмного коду, інтерфейсу, документації тощо;

– кожний запис містить деталі про зміну, такі як дата, автор зміни, опис внесених змін, причини і результати зміни;

– зміни можуть бути призначені для виправлення дефектів (багів), вдосконалення продукту, доповнення функціональності, забезпечення безпеки, оновлення або для інших цілей;

– Change log зазвичай пов'язаний із системами контролю версій, такими як Git, SVN або іншими, що дозволяють зберігати історію змін у продукті;

– історія змін може бути використана для аудиту, аналізу прогресу проєкту, забезпечення дотримання зміни вимог та плану, а також для вирішення спорів чи суперечок стосовно змін у продукті.

Change log є важливим інструментом управління змінами і допомагає команді проєкту та іншим зацікавленим сторонам зберігати зрозумілу і структуровану історію змін, що сприяє ефективному керуванню проєктом і забезпечує стабільний розвиток продукту.

Приклад Change log наведений на рис. 5.1.

CHANGE LOG								
ID	DATE	REQUESTED BY	CHANGE REQUEST DESCRIPTION	REASONS FOR THE CHANGE	STATUS	PROGRESS OF PENDING AND APPROVED CHANGES / REASON FOR REJECTION	ACTIONS RELATED TO THE CHANGE	NOTES
1	Feb-21	P swift	Change custom menu 238 to new status list.	New status list introduced.	APPROVED	Tested and approved for go live.	no further action.	Added to go live plan task 148.
2	Jan-21	D James	Save to favourites function.	Ability to save pages under customer's account page.	REJECTED	Rejected as just a nice to have consider for future phase.	no further action.	Added to draft phase 2 requirements.
3	Dec-20	D James	Additional action box widget.	To show latest stock prices.	PENDING	more information needed on pricing of api calls.	D. James getting estimate of monthly cost based on 1 call per hour.	
4	Dec-20	P swift	Change color <h1> tags to dark green.	To match customer scheme.	PENDING	Approved pending confirmation of exact hexdecimal code.	P Swift to get css and # from customer marketing team.	Task ID 138 on project schedule.
5	Jan-21	T Dent	Add recall page for product W132 SCANNER	Recall required urgent	APPROVED	Page copy with marketing for appoval.	T Dent to change marketing and submit to web team.	Task ID 138 on project schedule.
6	Jan-21	T Dent	Change 'client' to 'customer' throughout site.	To match new style guide.	APPROVED	Completed, tested and approved for go live	no further action.	Added to go live plan task 149.

Рис. 5.1. Приклад Change log

Software Requirements Specification

Software Requirements Specification (SRS, Специфікація вимог до програмного забезпечення) – це документ, що визначає детальні вимоги до програмного забезпечення, ключає функціональні та нефункціональні вимоги до системи, а також набір критеріїв для тестування та оцінки продукту.

Шаблон SRS наведений у Додатку А.

Для опису функціональних та нефункціональних вимог до системи можуть використовуватися Use case, user story та wireframes.

Use case

Варіант використання (use case) описує послідовність взаємодії системи та зовнішньої діючої особи, в результаті якої діюча особа отримує корисний результат.

Формула побудови варіанту використання наведена на рис. 5.2.

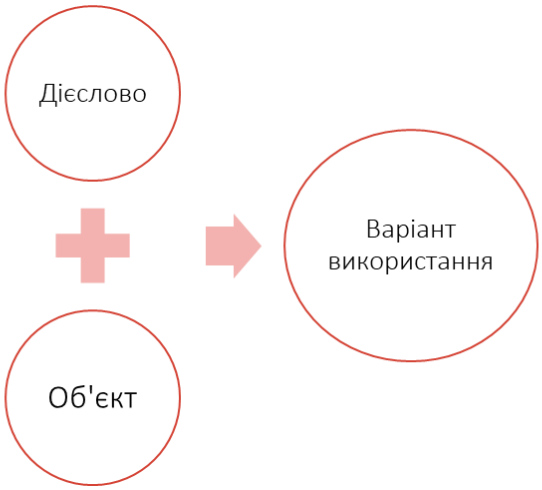


Рис. 5.2. Формула побудови варіанту використання

Шлях від користувацьких вимог до функціональних можна зобразити у вигляді схеми:

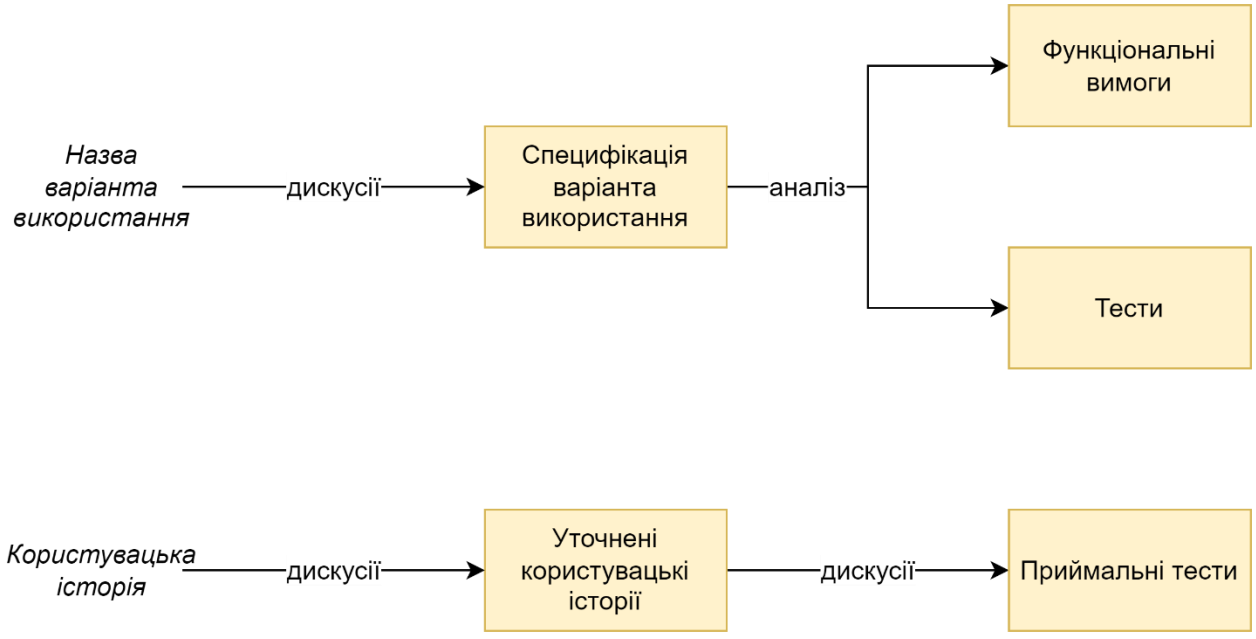


Рис. 5.3. Шлях від користувацьких вимог до функціональних

Приклади варіантів використання наведені в табл. 5.1.

Таблиця 5.1 – Приклади варіантів використання

Програмне забезпечення	Приклад варіантів використання
Система відстеження хімікатів	1. Замовити хімікат 2. Змінити замовлення хімікату 3. Перевірити стан хімікату 4. Згенерувати квартальні звіти про використання хімікатів
Термінал реєстрації в аеропорту	1. Зареєструватися на рейс 2. Роздрукувати квитки 3. Змінити місця 4. Сплатити більш зручне місце
Книжковий інтернет-магазин	1. Оновити профіль клієнта 2. Знайти товар 3. Купити товар 4. Відстежити відвантажену партію
Бухгалтерська система	1. Сформувати рахунок-фактуру 2. Звірити банківську виписку 3. Роздрукувати податкові декларації 4. Знайти потрібну транзакцію

Один варіант використання включає опис декількох сценаріїв.

Сценарій – опис одного випадку використання системи – окремий приклад варіанту використання.

Деякі елементи структури шаблона для опису варіанту використання:

1. Вихідні умови:

– те, що користувач може спостерігати (залишок на рахунку);

– фізичні результати (банкомат видав гроші та роздрукував чек);

– зміна внутрішнього стану системи (зміна суми рахунку на суму знятих коштів.

2. Альтернативні сценарії – можуть призводити до успішного виконання завдання, задовольняють початковим умовам, але є менш пріоритетними чи менш популярними варіантами задачі чи способу її виконання.

3. Виключення – очікувана помилкова умова, яка може виникнути під час виконання варіанту використання, спосіб її обробки (деякі мови користувач може виправити, деякі призводять до безуспішного завершення варіанту використання.

К. Вігерс пропонує наступний шаблон для опису варіанту використання (на прикладі Системи замовлення хімікатів (табл. 5.2) та на прикладі Системи замовлення їжі (табл. 5.3):

Таблиця 5.2 – Приклад опису варіанту використання для Системи замовлення хімікатів

Ідентифікатор та назва:	**UC-4 Запитати хімікат**	
Автор:	Лорі	
Основна діюча особа:	Співробітник, що розмістив замовлення на хімікат	
Опис:	Співробітник, що розмістив замовлення на хімікат, вказує в запиті необхідний хімікат, вводячи його назву або ідентифікатор або імпортуючи його структуру із відповідного графічного редактора. Система виконує запит, пропонуючи контейнер з хімікатом із складу чи дозволяючи створювати запит на замовлення у постачальника.	
Тригер:	Співробітник вказує, що хоче замовити хімікат.	
Попередні умови:	PRE-1. Особа користувача аутентифікована. PRE-2. Користувач має право запитувати хімікат. PRE-3. БД запасів хімікатів в поточний момент доступна.	
Вихідні умови:	POST-1. Запит зберігається в системі. POST-2. Запит надсилається на склад хімікатів.	
Нормальний напрям розвитку варіанта використання:	**4.0 Запитати хімікат зі складу** 1. Співробітник вказує необхідний хімікат. 2. Система наводить перелік контейнерів з необхідним хімікатом, що є на складі. 3. Співробітник може продивитися історію будь-якого контейнера. 4. Співробітник обирає певний контейнер чи просить відправити запит постачальнику (см. п. 4.1). 5. Співробітник вводить іншу інформацію, щоб завершити запит. 6. Система зберігає запит та спрямовує його до складу хімікатів.	
Альтернативний напрям розвитку варіанта використання:	**4.1 Запитати хімікат у постачальника** 1. Співробітник шукає хімікат за каталогами постачальника (см. п.4.1.E1). 2. Система відображає перелік постачальників, де також вказані розміри, клас, ціна контейнерів. 3. Співробітник обирає постачальника, розмір, клас, кількість контейнерів. 4. Співробітник вводить іншу інформацію, необхідну для запиту. 5. Система зберігає запит та спрямовує його до постачальника.	
Виключення:	**4.1.E1 Хімікату немає в продажі** 1. Система відображає повідомлення «У постачальників немає такого хімікату». 2. Система пропонує співробітнику запитати інший хімікат чи вийти із програми. 3а. Співробітник просить запитати інший хімікат. 4а. Система знов починає нормальний напрям розвитку варіанта використання. 3б. Співробітник вирішує вийти із програми. 4б. Система завершує варіант використання.	
Пріоритет:	Високий	
Частота використання:	Приблизно 5 разів на тиждень кожним хіміком, 200 разів на тиждень кожним співробітником складу.	
Бізнес-правила:	BR-28, BR-31	
Спеціальні вимоги:	Система повинна імпортувати хімічні структури в стандартній закодованій формі з будь-яких графічних редакторів, що підтримують малювання хімічних структур.	
Припущення	Імпортовані хімічні структури повинні бути вірними.	

Таблиця 5.3 – Приклад опису варіанту використання для Системи замовлення їжі

Ідентифікатор та назва:	**UC-5 Реєстрація на оплату через утримання з зарплатні**	
Автор:	Ненсі	
Основна діюча особа:	Клієнт	
Опис:	Клієнти кафетерію, що використовують Систему та замовляють страви з доставкою, повинні бути зареєстровані для оплати через утримання з зарплатні. Для безготівкових розрахунків через Систему кафетерій буде виставляти рахунки на оплату в системі розрахунку зарплатні, яка буде утримувати вартість замовлення з тієї суми, яку клієнт повинен отримати наступного разу прямим зарахуванням на депозит в день зарплатні.	
Тригер:	Клієнт запитав реєстрацію для сплати через утримання з зарплатні чи клієнт погодився на реєстрацію, коли його про це запитала Система.	
Попередні умови:	PRE-1. Клієнт зайшов у Систему.	
Вихідні умови:	POST-1. Клієнт зареєстрований для оплати через утримання з зарплатні.	
Нормальний напрям розвитку варіанта використання:	**5.0 Реєстрація для оплати через утримання з зарплатні** 1. Клієнт запитує в Системі розрахунку зарплатні інформацію, чи може він зареєструватися для утримання з зарплатні. 2. Система розрахунку зарплатні підтверджує, що клієнт має на це право утримання з зарплатні. 3. Система просить клієнта підтвердити бажання зареєструватися для оплати через утримання з зарплатні. 4. Якщо отримано позитивну відповідь, Система надсилає запит до Системи розрахунку зарплатні на включення оплати через утримання з зарплатні для клієнта. 5. Система розрахунку зарплатні підтверджує, що оплата через утримання з зарплатні увімкнена. 6. Система повідомляє клієнту, що оплата через утримання з зарплатні увімкнена.	
Альтернативний напрям розвитку варіанта використання:	**Немає**	
Виключення:	**5.0.E1 Клієнт не має права на оплату через утримання з зарплатні**	

	5.0.E2 Клієнт вже зареєстрований для оплати через утримання з зарплатні
Пріоритет:	Високий
Частота використання:	Необхідно очікувати високу частоту виконання цього варіанта використання в перші два тижні після випуску системи
Бізнес-правила:	BR-86, BR-88 керують правом клієнта на оплату через утримання з зарплатні.
Спеціальні вимоги:	-
Припущення	-
Інша інформація	-

Як визначити варіанти використання?

1. Визначити дійових осіб, потім бізнес-процеси, що підтримуються системою, далі варіанти використання для дій, в яких приймають участь діючі особи та система.
2. Виразити бізнес-процеси в термінах визначених сценаріїв, узагальнити сценарії у варіанти використання та визначити діючих осіб для кожного варіанта.
3. Використовуючи опис бізнес-процесу, задати питання «Які задачі повинна виконати система, щоб реалізувати цей процес або перетворити вхідні дані у вихідні?». Ці задачі можуть бути варіантами використання.
4. Визначити зовнішні події, на які система повинна реагувати, а потім співвіднести ці події з діючими особами та певними варіантами використання.
5. Застосувати CRUD (Create, Read, Update, Delete – створення, читання, оновлення, видалення) для визначення сутностей даних, яким необхідні варіанти використання для створення, читання, оновлення, видалення чи інших операцій с даними.
6. Проаналізувати контекстну діаграму, запитуючи «Яких цілей потрібно досягти кожній з зовнішніх сутностей при використанні системи?».

При побудові use case необхідно звернути увагу на наступне:

✓ Забагато варіантів використання → перевірити дублювання вимог, співвіднести з бізнес-вимогами.

✓ Надто складні варіанти використання → правильно обрати нормальній напрям виконання use case.

✓ Включення дизайну до опису варіантів використання → виключити вимоги до дизайну.

✓ Включення визначення даних (наборів даних) → створити окремо словник даних.

✓ Варіанти використання, які незрозумілі користувачам → перевірити зв'язок з бізнес-процесами та задачами користувачів.

User Story

Користувацька історія (user story) **–** стислий простий опис функції з позиції користувача за формулою (рис. 5.4):

Як **<тип користувача>** я хочу **<мета>** для того, щоб **<причина>**

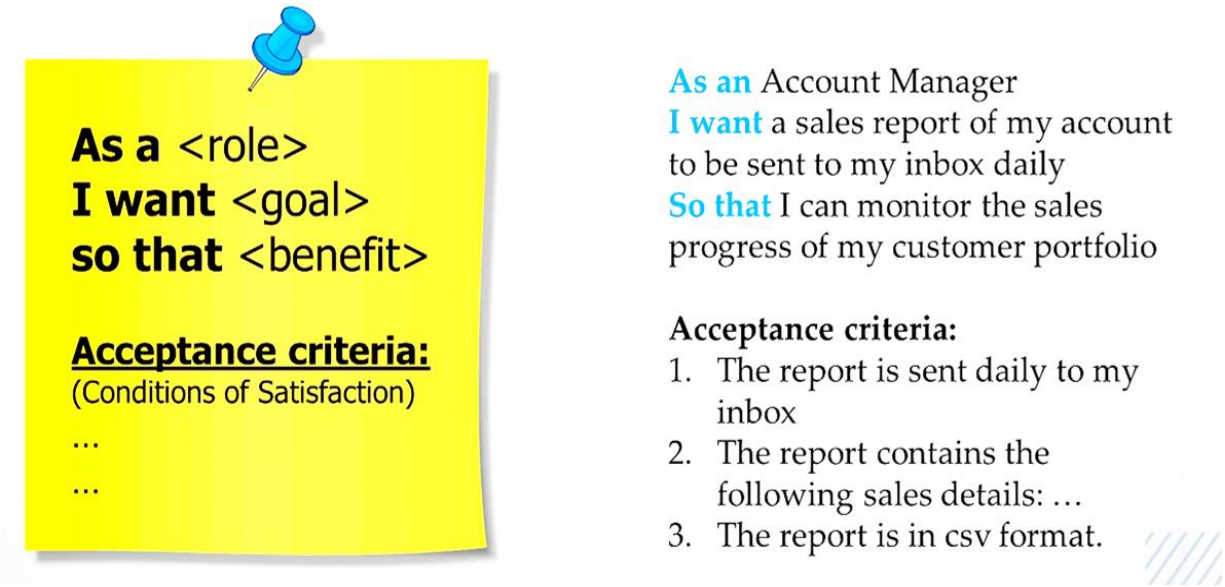


Рис. 5.4. Алгоритм формулювання користувацької історії

Acceptance criteria – критерій прийняття, деталі, необхідні для виконання конкретної user story, опис того, що має бути виконано.

Наприклад: «Як клієнт я хочу завантажити виписку з мого рахунку, щоб я міг переглянути всі свої транзакції, здійснені за певний період».

Acceptance criteria:

– Можливість вибрати період, протягом якого я хочу завантажити виписку.

– Можливість вибрати обліковий запис, для якого я хочу завантажити виписку.

– Можливість переглянути завантажений файл.

– Можливість завантажити свою виписку у форматах doc, excel та pdf.

Приклади співвіднесення *Use case та user story* наведені у таблиці:

Таблиця 5.4 – Приклади співвіднесення Use case та user story

Програмне забезпечення	**Приклад варіанта використання**	**Відповідна користувацька історія**
Система відстеження хімікатів	Замовити хімікат	Як **хімік** я хочу **замовити хімікат**, щоб **виконувати експерименти**
Термінал реєстрації в аеропорту	Зареєструватися на рейс	Як пасажир я хочу зареєструватися на рейс, щоб полетіти до місця призначення
Книжковий інтернет-магазин	Оновити профіль клієнта	Як клієнт я хочу оновити свій профіль, щоб всі наступні покупки сплачувати новою кредитною карткою
Бухгалтерська система	Створити рахунок-фактуру	Як власник маленької компанії я хочу створити рахунок-фактуру, щоб отримати оплату від клієнта

Отже, за допомогою *Use case та user story* описуються вимоги до програмного забезпечення. До цих формулювань висувається низка правил, які можна представити у вигляді ознак «ідеальних» вимог:

Повнота – кожна вимога повинна містити всю інформацію, необхідну читачеві, щоб її зрозуміти.

Коректність – кожна вимога повинна точно описувати можливість, яка задовольнятиме якусь потребу, яку необхідно реалізувати.

Здійсненність – можливість реалізувати вимогу при відомих обмеженнях системи та робочого середовища, а також в межах тимчасових, бюджетних та ресурсних обмежень проєкту.

Необхідність – кожна вимога повинна відображати можливість, яка надасть очікувану бізнес-користь, виділить продукт на ринку або необхідна для дотримання зовнішніх стандартів, політик чи правил.

Призначення пріоритетів – визначайте пріоритети бізнес-вимог на підставі важливості для отримання необхідної користі.

Недвозначність – не більше одного способу інтерпретації вимоги.

Перевірка – чи зможе тестувальник розробити тести або застосувати інші прийоми, щоб встановити, чи дійсно у продукті реалізовано кожну вимогу.

ТЕМА 6. НАВИЧКИ БІЗНЕС-АНАЛІЗУ НА ЕТАПІ ТЕСТУВАННЯ ТА ВПРОВАДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Negative use cases. Метрики, KPI, OKR, тощо. Ведення проєкту, управління очікуваннями замовника. Реліз проєкту та подальша передача продукту клієнту.

На етапі тестування програмного забезпечення бізнес-аналітик може виконувати декілька важливих завдань, що сприяють успішному завершенню проєкта та досягненню бізнес-цілей.

Деякі навички та завдання бізнес-аналітика на етапі тестування ПЗ:

1. Валідація вимог. Бізнес-аналітик може бути важливим членом команди, який перевіряє, чи вимоги до програмного продукту були правильно інтерпретовані та реалізовані. Вони співпрацюють з тестувальниками для переконання, що всі аспекти програми відповідають вимогам.
2. Розробка тестових сценаріїв. Бізнес-аналітик може співпрацювати з командою тестувальників для розробки та документування тестових сценаріїв.
3. Відстеження дефектів. Бізнес-аналітик може допомагати у відстеженні виявлених дефектів, співпрацювати з розробниками для їх виправлення та перевіряти, чи вони дійсно вирішені згідно з вимогами.
4. Аналіз результатів тестування. Бізнес-аналітик може аналізувати результати тестування, щоб переконатись, що програмний продукт відповідає бізнес-вимогам та очікуванням.
5. Уточнення вимог. Якщо під час тестування виявляються непорозуміння, бізнес-аналітик може спілкуватись з замовниками та командою розробки для уточнення та вдосконалення вимог.
6. Комунікація зі зацікавленими сторонами. Бізнес-аналітик може координувати комунікацію між різними зацікавленими сторонами, такими як замовники, тестувальники, розробники та управлінці, для забезпечення зрозумілості та згоди щодо результатів тестування.
7. Оцінка ризиків. Бізнес-аналітик може оцінювати ризики, пов'язані з дефектами та пропущеними вимогами, і розробляти плани для їх управління та розв'язання.
8. Підготовка документації. Бізнес-аналітик може допомагати у підготовці документації, пов'язаної з результатами тестування, аналізом дефектів, змінами в вимогах тощо.

Важливо пам'ятати, що співпраця між бізнес-аналітиком, розробниками та тестувальниками є ключовою для успішної реалізації проекту та досягнення бізнес-цілей.

Верифікація – перевірка відповідності продукту вимогам щодо виконання певної задачі.

Затвердження – перевірка відповідності продукту вимогам замовника (на різних етапах видобування та аналізу вимог):

✓ в специфікації достатньо повно описані очікувані можливості та характеристики системи, що задовольняють потреби різних стейкхолдерів;

✓ вимоги точно відображають бізнес-вимоги, системні вимоги, бізнес-правила та інші джерела;

✓ вимоги повні, такі, що реалізуються та перевіряються;

✓ всі вимоги необхідні, та вимог достатньо для досягнення бізнес-мети;

✓ всі вимоги узгоджені одна з одною;

✓ вимоги забезпечують якісну основу для дизайну та створення програмного забезпечення.

Рецензування вимог (peer review) та експертиза вимог (inspection) – два методи перевірки та оцінки вимог до програмного забезпечення на ранніх етапах проєкту з метою виявлення та виправлення можливих проблем, невідповідностей та помилок. Обидва методи націлені на забезпечення якості вимог та запобігання їх подальшого розповсюдження до більш пізніх стадій розробки, де виправлення помилок може бути більш витратним завданням.

Рецензування вимог – метод, при якому група експертів або членів команди перевіряє вимоги для забезпечення їх якості та відповідності бізнес-цілям. Основна ідея полягає у тому, що інші члени команди розглядають вимоги з своєї перспективи та можуть виявити можливі непорозуміння, пропуски чи невідповідності. Рецензування може включати письмові або обговорювальні процедури, де експерти діляться своїми враженнями та висновками щодо якості вимог.

Експертиза вимог – структурований метод перевірки вимог, який базується на документованому процесі та фіксованих правилах (рис. 6.1). Під час експертизи вимог, група експертів докладно аналізує документ з вимогами з точки зору визначених критеріїв, які можуть включати повноту,

якість, зрозумілість, вимірюваність та інші аспекти. Експерти використовують спеціальні методи та шаблони для виявлення потенційних помилок та невідповідностей.

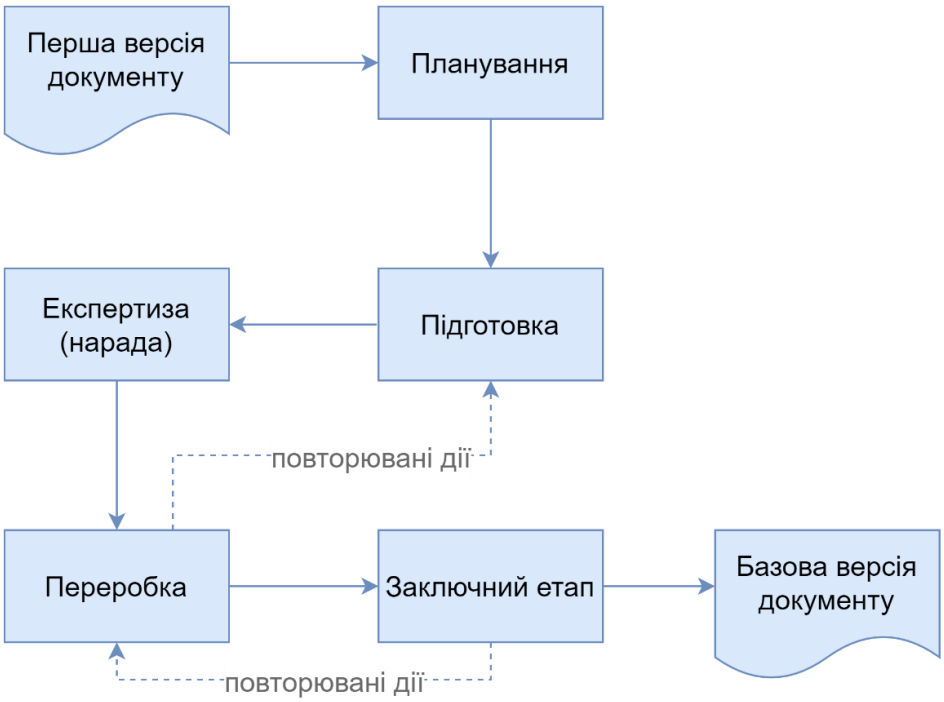


Рис. 6.1. Алгоритм проведення експертизи вимог

Кількість виявлених дефектів залежить від темпу інспектування (рис. 6.2):

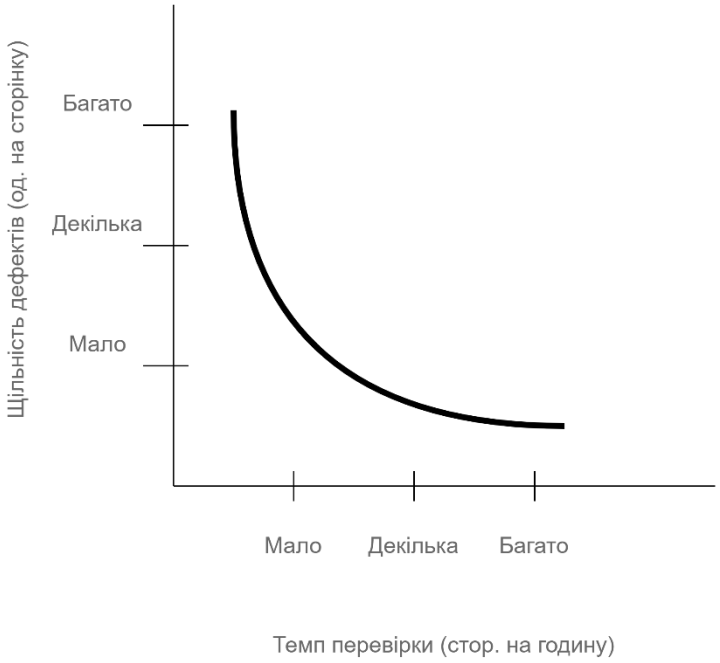


Рис. 6.2. Залежність кількості виявлених дефектів від темпу інспектування (експертизи)

Неофіційне рецензування:

– перевірка «за столом» (peer deskcheck) – перевіряє один член команди;

– колективна перевірка (passaround) – паралельно перевіряють декілька членів команди;

– наскрізний розбір (walkthrought) – автор описує продукт та просить його прокоментувати.

При проведені експертизи бізнес-аналітик може запропонувати експертам чек-лист з критеріями перевірки вимог:

– *Повнота:*

Чи включені до вимог всі потреби клієнта?

Чи визначені всі алгоритми, які відповідають функціональним вимогам?

Чи визначені всі інтерфейси взаємодії?

Чи задокументована очікувана поведінка системи для всіх очікуваних помилкових умов?

Чи забезпечують вимоги адекватну основу для дизайну та тестування?

Чи вказаний пріоритет кожної вимоги?

Чи знаходиться кожна вимога в межах проєкту, випуску, ітерації?

– *Коректність:*

Чи конфліктують вимоги між собою або дублюють одна одну?

Чи можна перевіряти кожну вимогу за допомогою тестування, демонстрації, перегляду чи аналізу?

Чи всі повідомлення про помилки виразні та зрозумілі?

Чи є вимоги дійсно вимогами, а не рішеннями чи обмеженнями?

Чи можна реалізувати всі вимоги, не перетинаючи межі встановлених обмежень?

– *Атрибути якості:*

Чи всі цілі продуктивності, безпеки та захисту сформульовані на достатньому рівні?

Чи всі інші атрибути якості задокументовані, мають кількісну оцінку?

Чи всі критичні за часом функції визначені та заданий графік їх реалізації?

– *Можливість відстеження:*

Чи на одному рівні деталізації написані всі вимоги?

Чи кожна вимога унікальна, ідентифікована, описана коректно?

Чи вказано для кожної вимоги посилання на відповідне джерело (бізнес-правило, системну вимогу)?

– *Інші проблеми:*

Чи не втрачені при описі альтернативні сценарії, виключення та інша інформація?

Чи всі бізнес-правила визначені?

Чи побудовані всі візуальні моделі?

Чи для всіх звітів побудовані специфікації, наскільки вони повно описані?

Вхідні та вихідні критерії експертизи вимог:

– *вхідні:*

✓ документ повинен відповідати певному шаблону;

✓ повинні бути посилання на ідентифікатори вимог;

✓ всі питання, які не вдалося вирішити позначаються як TBD (to be determined);

✓ координатор експертизи виділяє не більш ніж 3 суттєвих дефекти після 10-хвилинної перевірки;

– *вихідні:*

✓ всі питання, що виникли, повинні бути розв'язані;

✓ всі зміни, що внесені у документ, повинні бути внесені коректно;

✓ для відкритих питань повинні бути визначені шляхи вирішення, дата та виконавець.

Тестування та вимоги пов'язані синергетичним зв'язком, оскільки вони взаємодоповнюють погляди на систему. Розробка та тестування мають одне спільне джерело. Тести охоплюють нормальні сценарії, альтернативні сценарії та виключення.

Взаємодію бізнес-аналітика та тестувальника при перевірці вимог можна представити схемою на рис. 6.3.

Якщо варіанти використання написані повно, акуратно та зрозуміло, то процес складання тестів стає дуже простим.

Якщо варіанти використання описані неповно, двозначно, то спроба написати тести дозволить скорегувати варіанти використання.

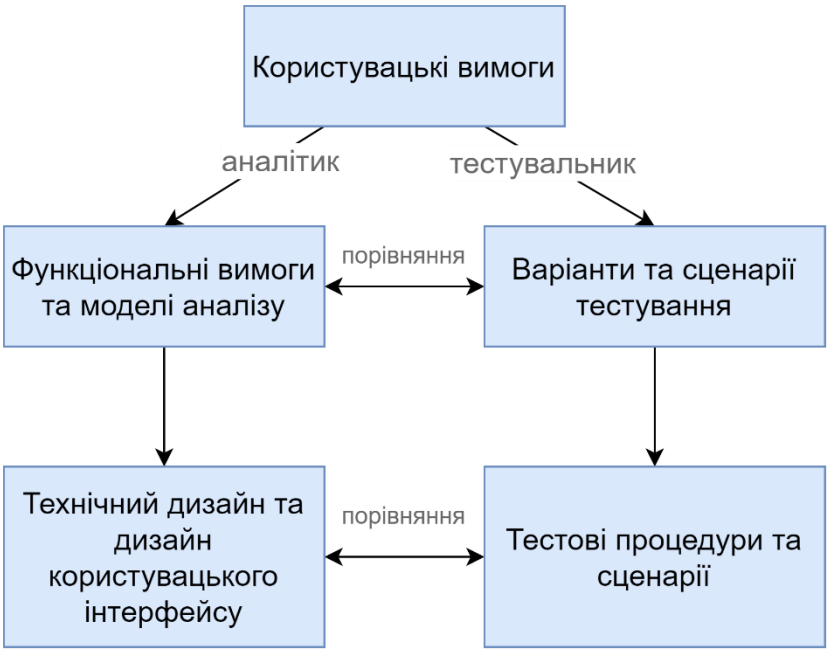


Рис. 6.3. Взаємодія бізнес-аналітика та тестувальника при перевірці вимог

Negative use cases

Negative use cases (негативні сценарії використання) – це аспекти аналізу вимог до програмного забезпечення, які описують ситуації, дії або події, які можуть призвести до небажаних або негативних результатів. Ці сценарії допомагають ідентифікувати потенційні проблеми, уразливості та помилки в програмі, які можуть виникнути, якщо користувачі або зовнішні фактори діють некоректно або зловживають системою.

Негативні сценарії використання можуть включати:

– некоректні дані. Використання неправильних або некоректних даних, яке може призвести до некоректної роботи програми або виводу невірних результатів;

– введення зловмисних даних;

– помилкові дії користувачів. Сценарії, де користувачі можуть ненавмисно або помилково виконувати дії, які призводять до небажаних результатів;

– нестабільні умови. Випадкові або неконтрольовані умови, такі як втрата зв'язку, низька витримка, недоступність ресурсів, які можуть вплинути на роботу програми;

– взаємодія з іншими системами. Проблеми, які виникають під час взаємодії з іншими системами або сервісами, такі як відмови у взаємодії, невідповідність інтерфейсів тощо;

– обробка великих обсягів даних. Проблеми, пов'язані з обробкою великої кількості даних, які можуть призвести до перевантаження системи або зниження продуктивності;

– безпека та захист. Негативні сценарії, пов'язані з порушенням безпеки, несанкціонованим доступом, витоком конфіденційної інформації тощо.

Тестування вимог є невід’ємною частиною управління вимогами – процесу, що охоплює етапи управління версіями, управління змінами, відстеження стану вимог, відстеження зв’язку вимог.

- ✓ **Управління версіями:**
  - Визначення схеми ідентифікації версій.
  - Відстеження версій окремих вимог.
  - Відстеження версій наборів вимог.
- ✓ **Управління змінами:**
  - Пропозиції змін.
  - Аналіз впливу змін.
  - Прийняття рішень.
  - Оновлення окремих вимог.
  - Оновлення наборів вимог.
  - Оновлення планів.
  - Оцінка змінності вимог.
- ✓ **Відстеження стану вимог:**
  - Визначення можливих станів вимог.
  - Фіксування стану кожної вимоги.
  - Відстеження стану розподілу всіх вимог.
- ✓ **Відстеження зв'язку вимог:**
  - Визначення зав'язків з іншими вимогами.
  - Визначення зав'язків з іншими елементами системи.

Для забезпечення управління вимогами бізнес-аналітик визначає перелік атрибутів, стан яких необхідно відстежувати.

До таких атрибутів можна віднести: дату створення вимог, номер поточної версії вимог, автора, пріоритет, стан вимоги, джерело вимоги, обґрунтування вимоги, номер випуску чи ітерації, відповідального за зміни вимоги, метод перевірки чи критерії приймання.

Рекомендовані *стани вимог*:

Proposed – вимога запрошена уповноваженою особою.

In Progress – бізнес-аналітик активно працює над вимогами.

Drafted – написана початкова версія вимоги.

Approved – вимога проаналізована, вплив на проєкт розрахований, вимога розміщена в базовій версії. Ключові стейкхолдери погодилися з цією вимогою, а розробники зобов'язалися реалізувати.

Implemented – код розроблений, протестований. Визначена залежність коду з відповідним дизайном. ПЗ, що реалізує відповідну вимогу, готове для тестування, рецензування.

Verified – вимога задовольняє критеріям приймання, т.т. підтверджена коректна функціональна реалізація вимоги.

Deferred – затверджена вимога запланована до реалізації в більш пізнішій версії.

Deleted – затверджена вимога видалена з базової версії. Описані причини видалення, вказаний автор рішення.

Rejected – вимога запропонована, але не запланована для реалізації в жодному з наступних випусків. Вказані причини відхилення вимоги та автор рішення.

Відстеження станів вимог на практиці здійснюється за допомогою графіку станів (рис. 6.4).

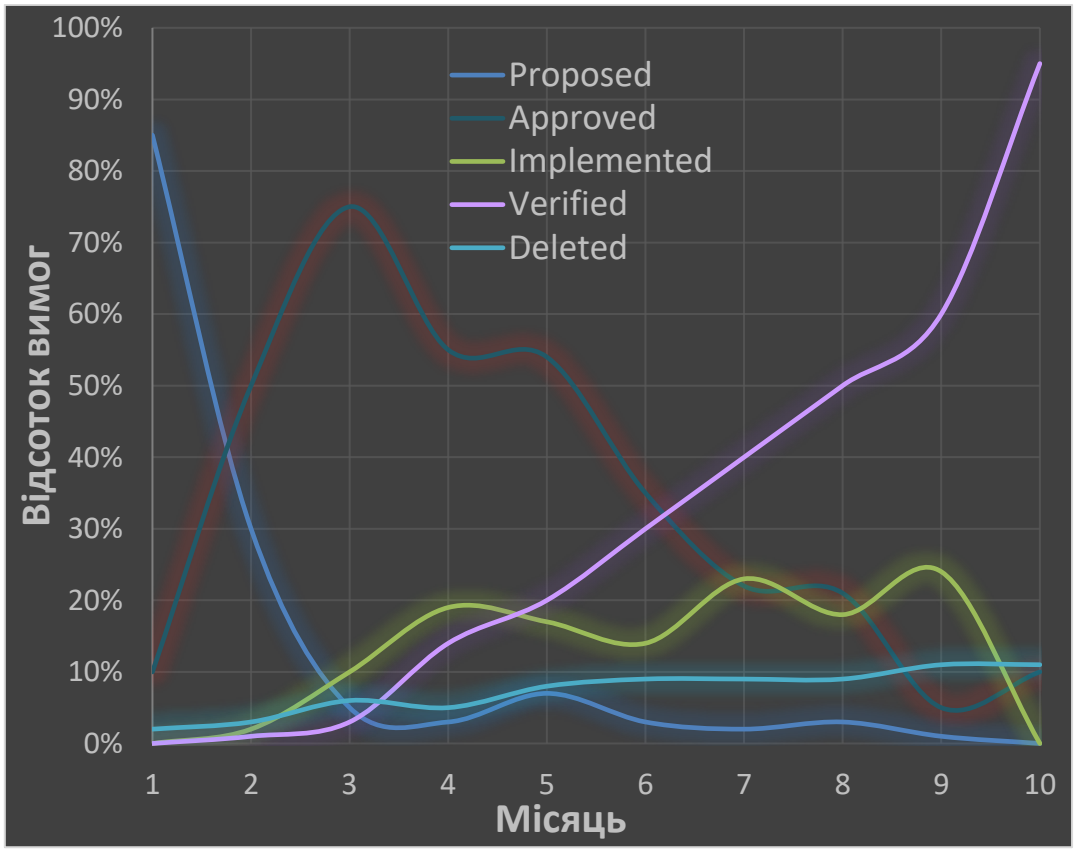


Рис. 6.4. Динаміка станів вимог

Гнучке управління вимогами — це ітеративний і поступовий підхід до управління вимогами, який характеризується своєю гнучкістю та чутливістю до змін.

В проєктах гнучкої розробки використовуються такі стани вимог, а відстеження їх стану візуалізується за допомогою графіку згорання задач (рис. 6.5):

In backlog – в резерві, історія поки що не призначена на ітерацію.

Defined – подробиці історії визначені, зрозумілі, написані приймальні тести.

In progress – історія в процесі реалізації.

Completed – історія повністю завершена.

Accepted – пройдені приймальні тести.

Blocked – розробник не може продовжити розробку, доки не будуть вирішені інші питання.

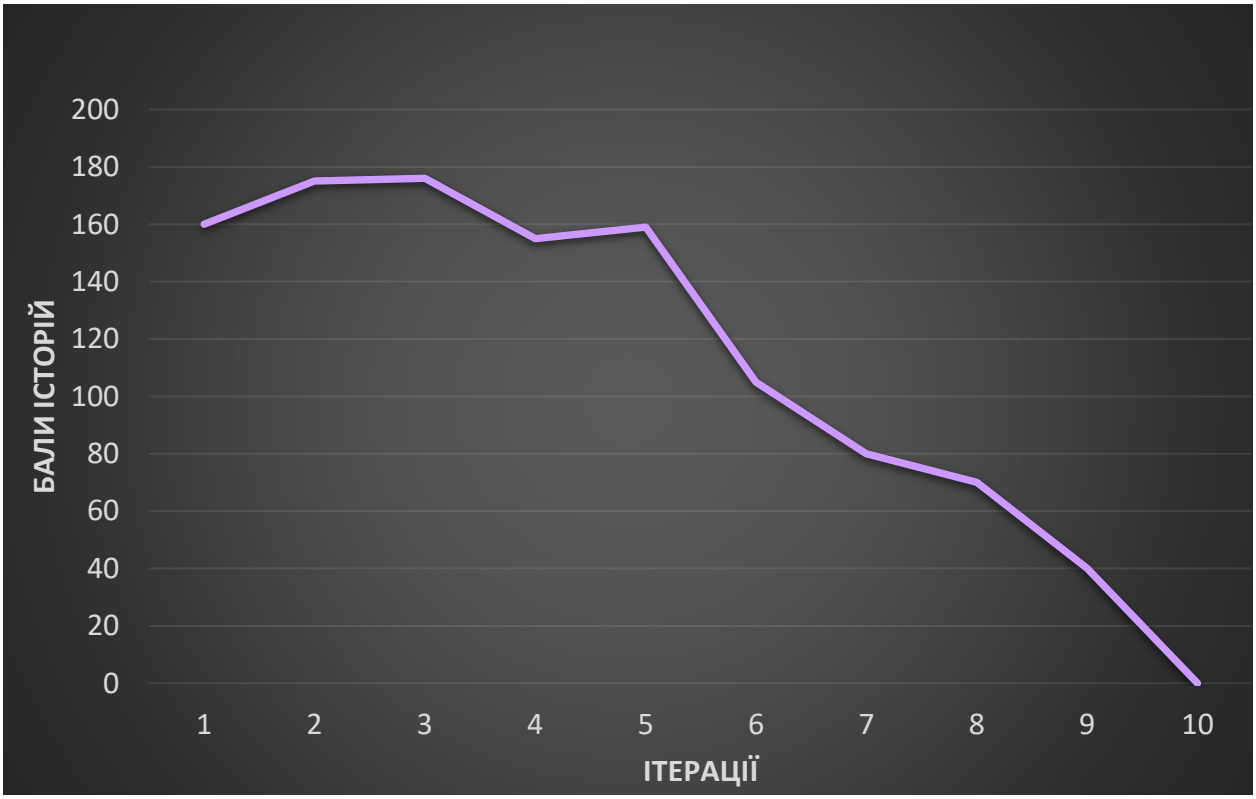


Рис. 6.5. Графік згорання задач

Проблеми, що виникають з вимогами в процесі розробки, наведені в табл. 6.1.

Таблиця 6.1 – Проблеми, що виникають з вимогами в процесі розробки

Тип проблеми	Опис
Питання за вимогою	Є незрозумілості, щось не вирішено
Відсутність вимоги	В процесі розробки виявлена відсутня вимога
Невірна вимога	Вимога невірно описана. Вимогу необхідно виправити чи видалити
Питання з реалізації	В процесі розробки у розробника виникають питання як повинно працювати чи щодо дизайну
Дублювання вимог	Виявлено дві чи більше еквівалентних вимоги. Необхідно залишити одну, а інші видалити
Зайва вимога	Вимога більше не потрібна

Для запобігання наведених вище проблем доцільно дотримуватися виконання (використання) наступних заходів:

– конфігурування інструментів управління вимогами;
– пропозиція зміни вимог, оцінка змін;
– оновлення сховища вимог;
– інформування про зміни вимог;
– контроль та звіт про стан вимог;
– збір інформації про зв’язок вимог.

Ведення проєкту, управління очікуваннями замовника

Ведення проєкту та управління очікуваннями замовника є критичними аспектами успішної реалізації проєктів. Ефективна взаємодія між командою проєкту та замовником допомагає досягнути бажаних результатів, забезпечити високу якість та уникнути ризиків.

Практичні поради щодо ведення проєкту та управління очікуваннями замовника:

1. Чітке визначення обсягу та вимог проекту. Важливо зрозуміти та документувати всі вимоги та очікування замовника щодо проєкту. Це допоможе уникнути непорозумінь під час реалізації та забезпечити однозначний контекст для роботи команди.

2. Реалістичні терміни та ресурси. Забезпечте реалістичні терміни виконання та необхідні ресурси для проекту. Переконайтеся, що команда має достатньо часу та засобів для виконання завдань.

3. Чітка комунікація. Підтримуйте постійний потік комунікації з замовником. Регулярні звіти, зустрічі та оновлення допоможуть зберегти замовника в курсі подій та виключити непорозуміння.

4. Документування змін. Будь-які зміни в обсязі, вимогах або плані проєкту повинні бути детально документовані та погоджені зі замовником. Це допоможе уникнути непорозумінь та можливих конфліктів.

5. Демонстрація результатів. Регулярно демонструйте замовнику проміжні результати та виконані завдання. Це дозволить замовнику бачити прогрес та вносити зміни, якщо необхідно.

6. Управління ризиками. Розгляньте можливі ризики та складні ситуації заздалегідь. Плануйте можливі сценарії та шляхи вирішення проблем.

7. Відкритий підхід до замовника. Створіть атмосферу відкритості та довіри зі замовником. Дозвольте йому ділитися своїми думками, змінами в планах та очікуваннями.

8. Збереження позитивної співпраці. Постійно підтримуйте конструктивну атмосферу співпраці з замовником. Розглядайте його побажання та вимоги як можливість для покращення проєкту.

9. Запити на зворотний зв'язок. Активно запитуйте замовника про його думки, враження та зауваження. Це допоможе відразу виявляти можливі невідповідності та вирішувати їх.

Можливі *ролі бізнес-аналітика* в контексті ведення проєкту, управління очікуваннями замовника:

1. Збір та аналіз вимог замовника. Бізнес-аналітик глибоко досліджує бізнес-потреби та вимоги замовника, допомагає визначити реальні бізнес-цілі та вимоги до програмного продукту, співпрацює з замовником для забезпечення зрозуміння його потреб та очікувань.

2. Документування вимог. Бізнес-аналітик перетворює вимоги замовника на документ, який буде основою для розробки, докладно описує функціональні та нефункціональні вимоги, сценарії використання, умови та обмеження.

3. Узгодження та погодження. Бізнес-аналітик допомагає узгодити вимоги замовника з можливостями команди розробки, слідкує за тим, щоб вимоги були реалістичними та досяжними.

4. Управління змінами. Під час реалізації проєкту можуть виникати зміни в вимогах. Бізнес-аналітик відстежує зміни, оцінює їх вплив та допомагає узгодити їх з замовником та командою.

5. Управління очікуваннями. Бізнес-аналітик грає ключову роль у керуванні очікуваннями замовника, допомагає встановити реалістичні та досяжні цілі, пояснює можливості та обмеження, допомагає уникнути надмірних очікувань, які можуть вплинути на успіх проєкту.

6. Комунікація. Бізнес-аналітик забезпечує постійний потік комунікації між замовником та командою розробки, допомагає замовнику розуміти прогрес робіт, виявляти можливі ризики та вносити зміни.

7. Вирішення конфліктів. Якщо виникають непорозуміння або конфлікти між замовником та командою, бізнес-аналітик може виступити посередником, допомагаючи знайти компроміс та знаходячи рішення, яке задовольнить обидві сторони.

8. Забезпечення якості. Бізнес-аналітик допомагає забезпечити якість реалізації проєкту, переконуючись, що вимоги замовника коректно реалізовані та перевірені.

9. Фасилітація робочих груп. Бізнес-аналітик може організовувати та сприяти роботі робочих груп замовника та команди розробки для досягнення спільних рішень та вирішення питань.

Реліз проєкту та подальша передача продукту клієнту

Реліз проєкту формується на завершальному етапі розробки програмного продукту, коли реалізована система передається замовнику для використання і підтримки.

Підготовка до релізу:

Впевнитися, що всі функції та вимоги були виконані та перевірені.

Проводиться внутрішнє тестування, включаючи різні сценарії використання.

Вирішуються всі виявлені проблеми та дефекти.

Тестування та QA:

Проводиться остаточне тестування, включаючи тестування сумісності, безпеки та продуктивності.

Перевіряється чи програмний продукт задовольняє усім вимогам та стандартам якості.

Документування:

Готується вся необхідна документація, включаючи посібники користувача, технічну документацію та інструкції.

Підготовка інфраструктури:

Перевіряється чи інфраструктура для розгортання, використання та підтримки продукту готова.

Реліз та розгортання:

Виконується реліз програмного продукту через розгортання його на середовищі клієнта або інфраструктурі замовника.

Передача знань:

Забезпечується навчання користувачів щодо використання продукту, проводяться тренінги та семінари.

Підтримка та обслуговування:

Забезпечується технічна підтримка та обслуговування продукту після передачі замовнику.

Відстежуються запити замовника, вирішуються проблеми та надаються консультації.

Зворотний зв'язок:

Збирається зворотний зв'язок від клієнта щодо продукту, функціональності та можливих поліпшень.

Моніторинг та оновлення:

Відстежується робота продукту у реальних умовах та вчасно надаються оновлення та виправлення.

Правильна передача продукту клієнту – це важливий етап, який вимагає уважного планування, підготовки та співпраці замовника та команди розробки. Успішний реліз і підтримка після передачі допоможуть забезпечити задоволення клієнта та досягти поставлених цілей проєкту.

Release notes (або Примітки до випуску) – це документ або повідомлення, яке надає користувачам та зацікавленим сторонам інформацію про зміни, вдосконалення та нововведення, які були внесені в програмний продукт у певному випуску (релізі).

Release notes містять інформацію про наступне:

Зміни функціональності. Опис нових функцій, можливостей або вдосконалень, які були додані до програми.

Виправлення помилок. Повідомлення про виправлення деяких проблем, багів або дефектів, які були виявлені в попередніх версіях.

Оптимізації та покращення. Інформація про оптимізацію продукту, що призвела до покращення продуктивності, використання ресурсів тощо.

Зміни інтерфейсу. Опис змін в інтерфейсі користувача, які можуть вплинути на взаємодію з продуктом.

Сумісність. Інформація про сумісність з іншими програмами, операційними системами або платформами.

Відомі проблеми. Повідомлення про відомі проблеми, які можуть залишатися невирішеними або які не встигли виправити.

Інструкції щодо оновлення. Поради та інструкції для користувачів щодо процесу оновлення до нової версії.

Ліцензійна інформація. Інформація про ліцензію та умови використання продукту.

Інші важливі відомості. Додаткова інформація, яка може бути корисною для користувачів, така як посилання на ресурси для підтримки, контактні дані та інше.

Release notes допомагають користувачам та адміністраторам зрозуміти, які зміни відбулися у новій версії продукту, і як це може вплинути на їхнє використання. Цей документ є важливою частиною комунікації між розробниками та користувачами програмного продукту.

Базовий набір розділів:

Заголовок – назва продукту, назва версії, дата випуску і т.д.

Загальна частина – резюме змін у версії.

Мета – короткий опис мети випуску версії зі списком змін, виправлень помилок, доданої функціональності.

Опис помилок – список дефектів, опис виправлень цих помилок.

Перелік потрібних оновлень – для оновлення апаратної та програмної частини, документації.

Юридична інформація – ліцензії, гарантії, відмова від відповідальності, тощо.

Контакти – контактна інформація служби підтримки продукту.

Приклад Release note (англійською мовою):

June, 3

We are glad to announce the release of a new product for finding a companion for cultural holidays "Meeting Place"

What is our mission?

Our mission is to connect people who share a passion for cultural holidays and provide them with a platform to find suitable companions for their travel experiences. We believe that cultural exploration is enhanced when

shared with like-minded individuals, and our goal is to make it easier for people to connect and embark on memorable cultural adventures together.

What can be useful for you?

Smart Search for Events: Our platform incorporates advanced search algorithms that consider individual preferences such as location, interests, date range, and specific cultural activities. This enables users to find relevant events that align with their preferences and desired experiences.

Visit History and Reviews: Users can keep track of their past cultural visits within the platform, providing a convenient way to revisit memories and reflect on their experiences. Additionally, they can share reviews and ratings of the events they attended, helping others make informed decisions and fostering a community of cultural enthusiasts.

Calendar and Reminders: "Meeting Place" provides a personal calendar where users can manage their upcoming cultural events, set reminders, and receive notifications. This ensures they never miss out on an event they're interested in and helps them stay organized with their cultural holiday plans.

Opportunity to Buy a Ticket: For selected events, our platform integrates with ticketing systems, allowing users to directly purchase event tickets. This simplifies the booking process and ensures a seamless experience from event discovery to ticket acquisition.

Selection of a Companion for the Event: One of the core features of "Meeting Place" is the ability for users to find companions for their cultural holidays. Our platform offers various ways to connect with potential companions, including profiles, messaging, and compatibility matching based on shared interests and preferences. This helps users find like-minded individuals to join them on their cultural adventures, fostering new friendships and enriching the overall experience.

You can get temporary access to full functionality for 7 days and enjoy all the benefits of the application. Try!

Ask us: https://meetingplace.ua

ТЕМА 7. БІЗНЕС-АНАЛІЗ В УПРАВЛІННІ РИЗИКАМИ ПРОЄКТУ

Управління ризиками в проєктах. Поняття ризику та невизначеності. Класифікація проєктних ризиків. Ідентифікація ризиків. Аналіз проєктних ризиків. Планування заходів з реагування на ризики. Моніторинг і контроль ризиків.

Управління ризиками є важливою складовою успішного проєкту, оскільки допомагає зменшити невизначеність, попередити негативні наслідки та забезпечити досягнення цілей проєкту відповідно до плану.

Управління ризиками в проєктах – це процес ідентифікації, аналізу, планування, виконання та моніторингу заходів, спрямованих на виявлення, зниження або управління ризиками, які можуть вплинути на успішність проєкту.

Управління ризиками як процес включає наступні етапи: оцінювання, попередження, контроль (рис. 7.1).

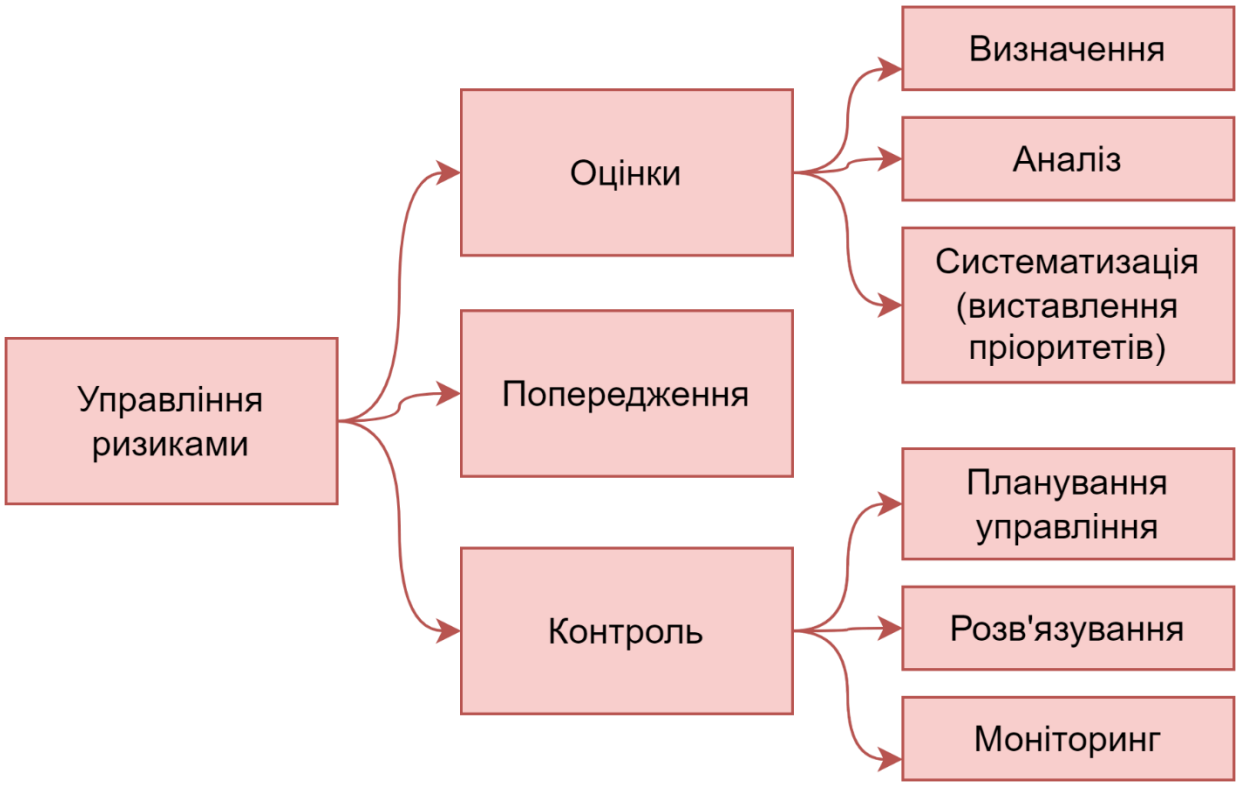


Рис. 7.1. Управління ризиками

Ризик – це потенційна подія або ситуація, яка може призвести до небажаного результату, втрати або негативного впливу на проєкт.

Невизначеність – це відсутність чіткості або точності в структурі, діяльності або результаті проєкту.

Ідентифікація ризиків – визначення можливих ризиків для проєкту шляхом аналізу різних аспектів (технічних, фінансових, графічних тощо); виявлення потенційних небезпек та вимірювання їх впливу та ймовірності.

Аналіз проєктних ризиків – оцінка ризиків за їхнім впливом та ймовірністю, а також вибір та пріоритезація ризиків, які мають найбільший вплив на проєкт.

Ризики проєкту можуть поділятися в залежності від джерела ризику, за етапами проєкту, власником ризику та іншими ознаками (рис. 7.2).

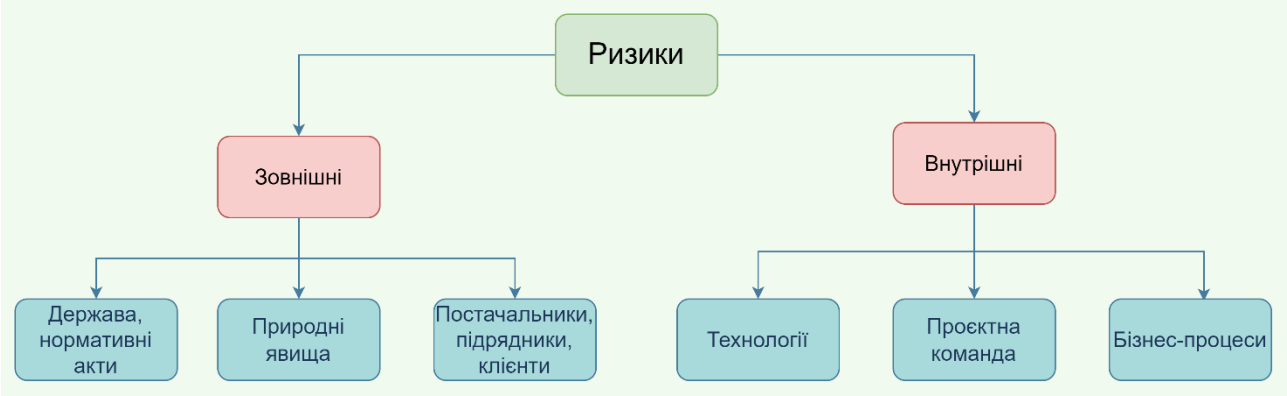


Рис. 7.2. Ризики проєкту

Одна з класифікацій проєктних ризиків:

– Стратегічні ризики. Пов'язані з напрямком розвитку компанії, ринковою кон'юнктурою тощо.

– Операційні ризики. Виникають внаслідок діяльності компанії, процесів та ресурсів.

– Фінансові ризики. Пов'язані зі змінами валютних курсів, фінансовими ринками тощо.

– Технічні ризики. Виникають через технічні обмеження, недоліки або незнання.

– Людські ризики. Пов'язані зі здоров'ям, відсутністю ключових спеціалістів, змінами в команді тощо.

Ризики необхідно виявляти, оцінювати та управляти ними, пам’ятаючи, що «ризик» - це ще не «проблема».

Планування заходів з реагування на ризики може включати такі етапи:

1. Ідентифікація ризиків. Визначаються можливі ризики, які можуть вплинути на проєкт або організацію. Розглядаються як внутрішні, так і зовнішні фактори, які можуть створити проблеми.

2. Аналіз ризиків. Оцінюються потенційні наслідки кожного ризику і його ймовірність виникнення. Визначають, як ці ризики можуть вплинути на цілі та завдання.

3. Визначення стратегій реагування. Обирається стратегія, яка допоможе зменшити вплив ризиків: уникнення, зменшення, передачу або прийняття ризику.

4. Розробка плану дій. Для кожного ідентифікованого ризику створюється докладний план дій, які необхідно вжити, якщо ризик виникає. В плані вказуються відповідальні особи, кроки для запобігання чи зменшення ризику, а також можливі наслідки та ресурси, необхідні для втручання.

5. Впровадження плану. Здійснюються заходи, передбачені планом реагування на ризики, коли ситуація вказує на необхідність дій. Проводиться моніторинг ризиків та результатів впровадження, необхідно вчасно коригувати стратегії, якщо це необхідно.

6. Комунікація та співпраця. Необхідно забезпечити ефективну комунікацію між всіма сторонами, які відповідають за реагування на ризики. Важливо співпрацювати з командою проєкту або колегами, щоб вчасно реагувати на зміни ситуації.

7. Моніторинг та оновлення. Необхідно постійно відстежувати стан ризиків і результати впровадження планів реагування. Вчасно необхідно виправляти недоліки, вносити корективи в стратегії, якщо ситуація змінюється або нові ризики з'являються.

8. Вивчення і вдосконалення. По завершенні проєкту або після врегулювання ризиків проводиться аналіз того, як ефективно були впроваджені плани реагування. Визначаються навчальні моменти та можливості для поліпшення підходів до реагування на ризики у майбутньому.

Ще раз про стратегії управління ризиком:

1. Прийняття – прийняти те, що ризик стане проблемою, і планувати її вирішення.

2. Уникнення – вжити дій для того, щоб ризик не став проблемою.

3. Зменшення (мінімізація) – запланувати дії, що знизять ймовірність та вплив ризиків на проєкт.

4. Передача – передати ризики на клієнта або іншого виконавця.

Для відстеження ризиків формується реєстр ризиків (табл. 7.1, 7.2).

Таблиця 7.1 – Реєстр ризиків

ID	***Date raised***	***Risk Description***	***Likelihood of the risk***	***Impact if the risk occurs***	***Owner***	***Mitigation Action***	***Status History***	***Decision***
ІД	***Дата виявлення***	***Опис ризику***	***Ймовірність настання***	***Вплив***	***Власник***	***Стратегія управління***	***Статус***	***Рішення***
1	*19.04.2023*	*Не вистачить бюджету на реалізацію функції чату*	*Medium*	*Low*	*PM*	*Відхилення*	*Відкрита*	*Перенести частину функцій на наступний реліз*

Таблиця 7.2 – Реєстр ризиків (приклад)

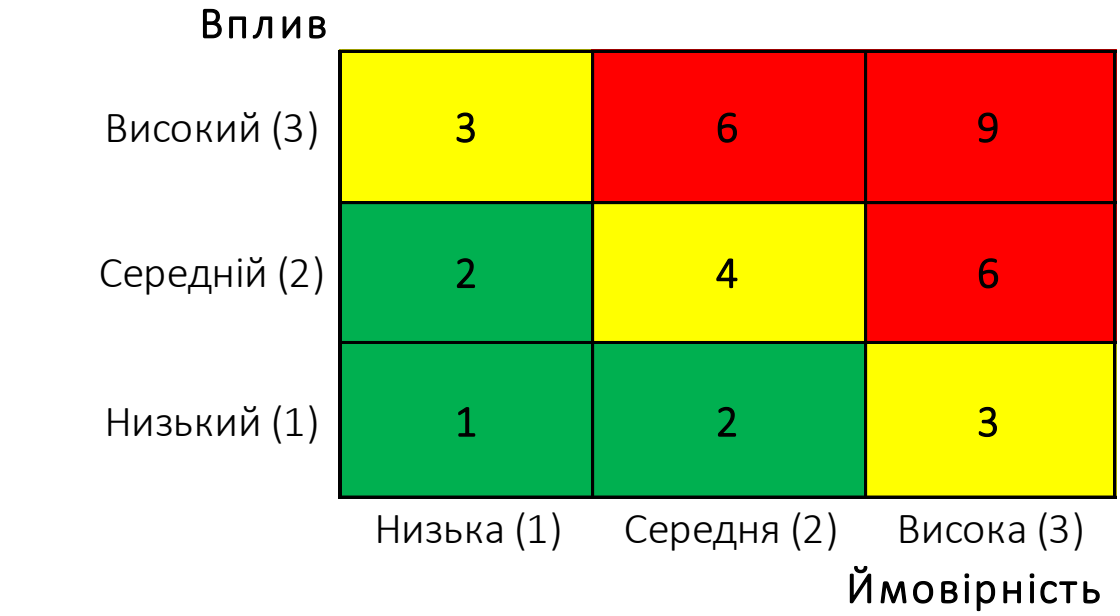
ID	**Date raised**	**Risk Description**	**Likelihood of the risk occurring**	**Impact if the risk occurs**	**Owner**	**Mitigation Action**	**Status History**	**Decision**
ІД	**Дата виявлення**	**Опис ризику**	**Ймовірність настання**	**Вплив**	**Власник**	**Стратегія управління**	**Статус**	**Рішення**
1	19.04.2023	Не вистачить бюджету на реалізацію функції чату	Medium	Low	PM	Відхилення	Відкрита	1. Перенести частину функцій на наступний реліз. 2. Знайти додаткові джерела фінансування.
2	19.04.2023	Інвестор скоротить строки випуску MVP	Low	Medium	PM	Мінімізація	Відкрита	1. Перенести частину функцій на наступний реліз, узгодивши це з інвестором. 2. Розглянути питання залучення додаткових спеціалістів. 3. Організувати стратегічну сесію з інвестором щодо узгодження строків.
3	19.04.2023	Не зможемо налаштувати API з сайтом-партнером,	Medium	High	PM, Тімлід	Передача	Відкрита	1. Знайти альтернативний сайт, з яким можлива

ID	Date raised	Risk Description	Likelihood of the risk occurring	Impact if the risk occurs	Owner	Mitigation Action	Status History	Decision
		бо не знайдемо відповідної документації						інтеграція по подіям. 2. Знайти альтернативні шляхи взаємодії, наприклад, взаємне посилання на сайти один одного, спільне проведення маркетингових кампаній або інші форми партнерської співпраці.
4	19.04.2023	Розробник команди не впорається з алгоритмом рекомендацій подій, який використовує AI	High	Medium	PM, Інвестор	Мінімізація	Відкрита	1. Організувати стратегічну сесію з інвестором з метою або залучення додаткового спеціаліста, або передачі цієї функції іншому розробнику. 2. Запропонувати альтернативний алгоритм.
5	19.04.2023	Інвестор не затвердить вчасно дизайн	Low	Low	PM	Прийняття	Відкрита	1. Підготувати декілька альтернативних прототипів на етапі планування та збирання вимог.

Оцінювання ризиків може здійснюватися за допомогою карти ризиків (рис. 7.3).

Карл Вігерс пропонує наступний шаблон для відстеження ризиків (табл. 7.3).

Приклад заповнення шаблону наведений в табл. 7.4.



Прийнятний ризик. Не потребує втручання

Помірний ризик. Необхідні заходи з контролю

Неприйнятний ризик. Необхідні негайні заходи щодо зниження ризику

Рис. 7.3. Карта ризиків

Таблиця 7.3 – Шаблон для відстеження ризиків

Ідентифікатор	***Номер***
Автор	*Особа, що виявила/звернула увагу на ризик*
Дата виявлення	*Дата виявлення ризику*
Дата закриття	*Дата закриття ризику*
Опис	*Опис ризику за схемою «причина-наслідок»*
Масштаб впливу	*Проектні команди, бізнес- та функціональні області, на які може вплинути ризик*
Ймовірність	*Ймовірність переходу ризику в проблему*
Вплив	*Оцінка в балах втрат, якщо ризик стане проблемою*

	(1 бал – немає проблеми; 10 – повний жах)
Схильність	*Ймовірність, помножена на вплив*
План управління	*Методи управління, запобігання, зменшення наслідків ризику, тощо*
План забезпечення неперервності роботи	*Порядок дій для ситуації, коли план управління неефективний*
Відповідальна особа	*Особа, що відповідає за вирішення/закриття ризику*
Термін виконання	*Дата, до якої необхідно виконати дії щодо пом'якшення ризику*

Таблиця 7.4 – Приклад заповнення шаблону (К. Вігерс)

Ідентифікатор	***1***	
Автор	*Хтось*	
Дата виявлення	*22.02.23*	
Дата закриття	*відкритий*	
Опис	*Недостатня залученість користувачів до складання вимог може призвести до необхідності масштабної переробки користувацького інтерфейсу після бета-тестування*	
Масштаб впливу	*Може впливати на всю систему, враховуючи всі користувацькі доповнення, які були розроблені для інтегрованих компонентів серійної системи*	
Ймовірність: 0,6	**Вплив**: 7	
План управління	1. *Зібрати вимоги щодо зручності використання на ранніх стадіях виявлення вимог.* 2. *Провести зустрічі з прихильниками продукту для розробки вимог.* 3. *Розробити одноразовий прототип базової функціональності на основі інформації прихильників продукту.* 4. *Виконати оцінювання прототипу користувачами декількох типів.*	

План забезпечення неперервності роботи	*Залучити UA-дизайнера для перевірки відповідності користувацького інтерфейсу рекомендаціям.*
Відповідальна особа	*Ще один Хтось*
Термін виконання	*13.12.23*

Бізнес-аналітик акцентує свою увагу на управлінні ризиками, які стосуються вимог до програмного забезпечення.

Методи/інструменти зниження ризиків, пов'язаних з вимогами, у відповідності до етапів роботи з вимогами:

- ✓ Виявлення:
  - концепція та межі проєкту;
  - час, витрачений на розробку вимог;
  - залученість клієнта;
  - повнота та коректність специфікації вимог;
  - вимоги до інноваційних продуктів (додатковий аналіз ринку);
  - визначення нефункціональних вимог;
  - несформульовані вимоги (виявлення прихованих очікувань);
  - використання існуючого продукту як основи вимог;
  - довіра між бізнесом та розробниками.
- ✓ Аналіз вимог:
  - визначення пріоритетів вимог;
  - управління технічно складними функціями;
  - обережне ставлення до незнайомих технологій, методів, мов, інструментів чи обладнання.
- ✓ Специфікація вимог:
  - однакове розуміння вимог;
  - усунення поспіху при наявності незавершених проблем;
  - однозначна термінологія;
  - виключення дизайну з вимог.
- ✓ Затвердження вимог:
  - затвердження вимог;
  - якість перевірки (експертизи) вимог.
- ✓ Управління вимогами:
  - контроль зміни вимог;
  - відстеження нереалізованих вимог;
  - контроль розширення меж проєкту.

ПЕРЕЛІК ЛІТЕРАТУРИ

1. Alan Cooper. The Inmates Are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity. 2004. 288 p.
2. Cobb G. (2016) Making Sense of Agile Project Management: Balancing Control and Agility, N.Y. Wiley, 2016. 265 p.
3. Cohn M. (2015) Succeeding with Agile: Software Development Using Scrum – Boston: Addison-Wesley Professional, 2015. 504 p.
4. Karl Wiegers, Joy Beatty. Software Requirements, Third Edition, 2014. 673 p. URL: https://www.booksfree.org/wp-content/uploads/2022/03/Software_Requirements_3rd_Edition_compressed.pdf.
5. Блага Н.В. Управління проєктами: навч. посібник. Львів: Львівський державний університет внутрішніх справ. 2021. 152 с.
6. Кузьміних В.О., Тараненко Р.А. Основи управління ІТ проектами: навч. посіб. для студ. спеціальності 122 «Комп'ютерні науки». Київ: КПІ ім. Ігоря Сікорського. 2019. 75 с.
7. Лисак В., Ноздріна Л. Методи і моделі бізнес-аналізу в ІТ-галузі // Вісник Університету банківської справи. 2020. № 3 (39). С. 94–103. URL: https://www.researchgate.net/publication/350660394_METODI_I_MODELI_BIZNES-ANALIZU_V_IT-GALUZI
8. Старченко Г.В. Управління проектами: теорія та практика : навч. посіб. Чернігів: видавець Брагинець О. В. 2018. 306 с.
9. Приймак В.М. Управління проектами. Навчальний посібник. К.: Київський національний університет імені Тараса Шевченка. 2017. 464 с.
10. Стівен Р. Кові. 7 звичок надзвичайно ефективних людей. Потужні інструменти розвитку особистості. Видавництво: КСД. 2021. 384 с.
11. Дж. Ханк Рейнвотер. Як пасти котів. Видавництво: Фабула. 2020. 320 с.
12. https://www.ba.in.ua
13. http://www.iiba.org
14. https://www.edx.org/learn/business-analysis
15. https://www.bridging-the-gap.com

ДОДАТОК А

ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА»

Специфікація вимог до програмного забезпечення
SRS – Software Requirements Specification

ЗМІСТ

1. ВСТУП

1.1. Призначення

Назва продукту, мета, версія.

Для кого призначений цей документ (Наприклад: Документ призначений для розробників, менеджерів проєкту, маркетологам, користувачам, тестувальникам).

1.2. Регламенті документи

Перелік стандартів, нормативних документів, топографічні угоди (стильові обмеження, нотація), вимоги до нумерації/ідентифікації вимог.

1.3. Межі проєкту

Стислий опис продукту, зв'язок з бізнес-цілями.

Може бути наведений стислий перелік головних функцій продукту в межах версії, для якої формується SRS.

При наявності документа про концепцію та межі проєкту – вказується лише посилання на документ.

1.4. Посилання

Перелік всіх документів та ресурсів, на які є посилання в SRS (наприклад, контракти, специфікації до системних вимог, специфікації до ПЗ зв'язаних продуктів, специфікації інтерфейсів тощо).

Формат подання інформації: назва, автор, номер версії, дата, джерело, місце зберігання/посилання.

2. ЗАГАЛЬНИЙ ОПИС

2.1. Загальний погляд на продукт

Стислий опис контексту продукту, чи є він частиною існуючої системи (тоді ще додається опис взаємодії з існуючою системою), чи це нова його версія, чи новий продукт – заміна існуючого.

Доречно додати контекстну діаграму чи карту екосистеми для демонстрації взаємозв'язку продукту з іншими системами.

2.2. Класи та характеристики користувачів

Опис класів користувачів, визначення привілейованих класів.

Якщо опис повторює документ про концепцію та межі проєкту, то можна лише навести відповідне посилання.

Приклад.

Клас користувачів	Опис
Клієнт (привілейований клас)	Це співробітник, який бажає замовити харчування з доставкою з кафетерію кампанії. Всього потенційних клієнтів – 600, з

	яких 400 очікувано будуть використовувати систему в середньому 5 разів на тиждень. Іноді клієнти будуть замовляти харчування на декілька осіб. Очікується, що 60% замовлень будуть надходити через корпоративну мережу, а 40% – з домашніх комп'ютерів чи додатків для планшетів та смартфонів.
Співробітник кафетерію	В кафетерії на теперішній час працюють приблизно 20 співробітників, які будуть отримувати замовлення через систему, готувати страви, пакувати їх для доставки, друкувати інструкції для доставки та запитувати доставку. Більшість співробітників необхідно буде вчити користуватися комп'ютером та використовувати систему.

2.3. Операційне середовище

Опис апаратної платформи, операційних систем та їх версій, географічне місцерозташування користувачів, серверів та баз даних у відповідності до організацій, де вони розташовані. Опис всіх інших компонентів ПЗ, додатків, з якими система повинна бути сумісною.

При необхідності описуються вимоги до інфраструктури.

Приклад:

ОЕ-1. Система повинна надавати доступ користувачам через корпоративну мережу, VPN-канал та зі смартфонів та планшетів під управлінням Android, iOS та Windows.

2.4. Обмеження дизайну та реалізації

Вказується при необхідності визначена мова програмування, бібліотеки тощо. Опис всіх факторів, що обмежують розробників, з обгрунтуванням цих обмежень.

Приклад:

СО-1. Система повинна використовувати поточну версію СУБД Oracle, що є корпоративним стандартом.

2.5. Припущення та залежності

Припущення (assumption) – твердження, яке вважається вірним при відсутності знань та доказів про протилежне. Припущення стосуються виключно функціональності.

Опис залежності системи від всіх зовнішніх факторів чи компонентів поза її впливом (наприклад, попередньо необхідно встановити певне програмне забезпечення).

Приклад:

AS-1. Кафетерій відкритий для сніданків, обідів та вечерь кожного робочого дня компанії, коли співробітники мають бути на робочих місцях.

DE-1. Робота системи залежить від змін в системі розрахунку заробітних плат, що дозволяють приймати запити на оплату за харчування, яке замовляється через систему.

3. ФУНКЦІЇ СИСТЕМИ

3.1. Функція системи X

Назва/опис функції, пріоритет (високий/середній/низький).

Перелік функціональних вимог, які пов'язані з цією функцією (наприклад, варіант використання).

Опис реагування системи на очікувані помилки, невірне введення інформації, хибні дії.

Приклад.

Автентифікація користувача в системі

Опис	Автентифікація користувача в системі
Розширюється	–
Узагальнює	–
Рівень	Користувача
Користувач	Співробітник підприємства
Передумови	Користувач ініціює операцію входу в Систему
Основний потік	1. Користувач через функцію/ кнопку «Вхід» на веб-порталі ініціює автентифікацію користувача. 2. Система відображає екранну форму заповнення атрибутів автентифікації відповідно до атрибутів (табл. 3.1.1). 3. Користувач заповнює атрибути екранної форми автентифікації користувача відповідно списку атрибутів та опису їх заповнення, наданого в (табл. 3.1.1). 4. Користувач ініціює вхід до Системи через функцію/ кнопку «Увійти». 5. Система перевіряє повноту, обов'язковість та коректність заповнення атрибутів екранної форми автентифікації користувача. 6. Система відображає дані профілю користувача, за яким користувач виконав автентифікацію.

Альтернативні потоки	5а. Система виявляє помилки в атрибутах екранної форми автентифікації користувача: 5а_1. Система відображає повідомлення про помилку згідно табл. 5.1.1. Повідомлення 1. 5а_2. Система припиняє виконання операції автентифікації. 5б. Якщо за даними адреси електронної пошти користувача не знайдено в списку користувачів: 5б_1. Система відображає повідомлення згідно табл. 5.1.1. Повідомлення 1. 5б_2. Система відкриває сторінку з посиланням «Подати заявку на первинну реєстрацію».
Результат	Автентифікація користувача в системі
Постумови	–
Додаткові вимоги	**Ошибка! Источник ссылки не найден.**

Таблиця 3.1.1. Атрибути форми аутентифікації користувачів

№ атрибуту	Атрибут	Обов'язковість заповнення	Символи, що можуть бути внесені	Мінімальна та максимальна кількість символів	Значення за замовченням	Коментар
1.	Бажаєте увійти як	+	[Набір 1.3]	{0-50}	у відповідності до довідника типів користувачі	вибір за допомогою випадаючого списку: - співробітник компанії; - співробітник кафетерію; - співробітник служби доставки.
2.	Тип документу, що посвідчує особу		[Набір 1.3]	{0-50}		вибір за допомогою випадаючого списку
3.	Серія та номер документу, що посвідчує особу		[Набір 1.3]	{3-15}		ручне введення текстового поля, за наявності

4. ВИМОГИ ДО ДАНИХ

4.1. Логічна модель даних

Модель даних – візуалізація подання об'єктів та наборів даних, які буде обробляти система, а також зав'язків між ними (наприклад, схеми даних, діаграми «сутність-зв'язок», діаграми класів UML).

Приклад:

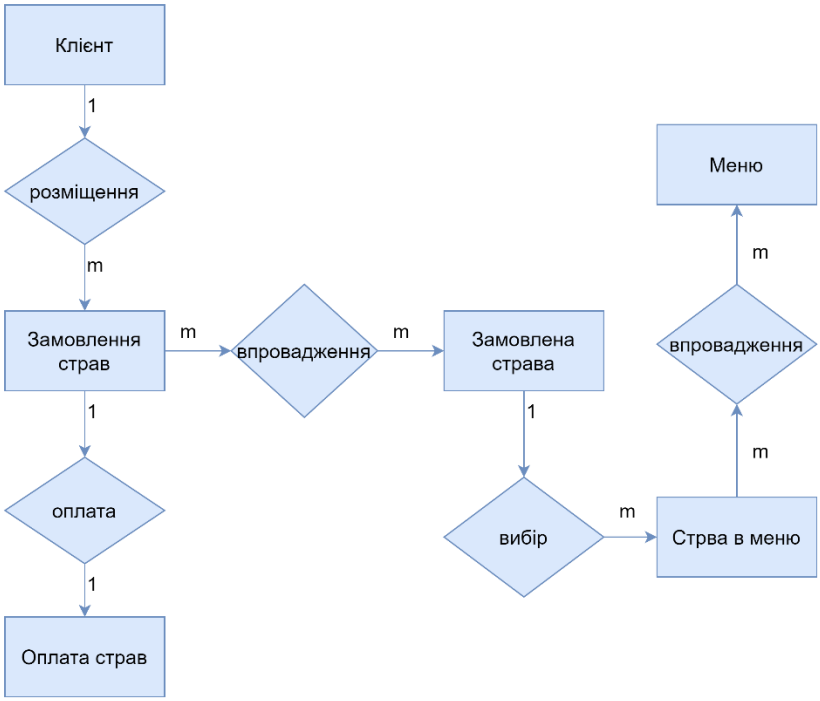


Рис. 4.4.1. Фрагмент моделі даних

Рис. 4.4.2. Діаграма класів

4.2. Словник даних

Словник визначає склад структур даних, а також їх значення, тип даних, довжину, формат та дозволені значення елементів даних, з яких

складаються структури даних. Доречно словник даних зберігати як окремий артефакт, а в SRS наводити відповідне посилання на нього.

Приклад:

Позначення	Перелік символів	Приклад використання
[Набір 1.1]	0,1,2,3,4,5,6,7,8,9	Індекс
[Набір 1.2]	А, а, Б, б, В, в, Г, г, Ґ ґ, Д д, Е е, Є є, Ж ж, З з, И и, І і, Ї ї, Й й, К к, Л л, М м, Н н, О о, П п, Р р, С с, Т т, У у, Ф ф, Х х, Ц ц, Ч ч, Ш ш, Щ щ, Ь ь, Ю ю, Я я, Ё ё , Ъ ъ,Ы ы ,Ь ь ,Э э, ‘,-,’ , ‘,’,	ПІБ
[Набір 1.3]	0,1,2,3,4,5,6,7,8,9 А, а, Б, б, В, в, Г, г, Ґ ґ, Д д, Е е, Є є, Ж ж, З з, И и, І і, Ї ї, Й й, К к, Л л, М м, Н н, О о, П п, Р р, С с, Т т, У у, Ф ф, Х х, Ц ц, Ч ч, Ш ш, Щ щ, Ь ь, Ю ю, Я я, Ё, ё , Ъ, ъ,Ы, ы ,Ь, ь ,Э, э A, a, B, b,C, c, D, d, E, e, F, f,G, g, H, h ,I, i ,J, j ,K, k, L, l, M, m, N, n, O, o, P, p, Q, q, R, r, S, s, T, t, U, u, V, v, W, w, X, x, Y, y, Z, z,’, !, », №, ;, %, :, ?, *, (,), _, +, /, *, -	Додаткові дані
[Набір 1.4]	Aa, Bb,Cc, Dd, Ee, Ff,Gg, Hh ,Ii ,Jj ,Kk, Ll, Mm, Nn, Oo, Pp, Qq, Rr, Ss, Tt, Uu, Vv, Ww, Xx, Yy, Zz, 0,1,2,3,4,5,6,7,8,9, “-“, “_”, “@”, “.”	Email
[Набір 1.5]	Aa, Bb,Cc, Dd, Ee, Ff,Gg, Hh ,Ii ,Jj ,Kk, Ll, Mm, Nn, Oo, Pp, Qq, Rr, Ss, Tt, Uu, Vv, Ww, Xx, Yy, Zz, 0,1,2,3,4,5,6,7,8,9 “-“, “_”,	Пароль
[Набір 1.6]	0,1,2,3,4,5,6,7,8,9,”- “,”(“,”) “	Телефон
[Набір 1.7]	0,1,2,3,4,5,6,7,8,9,”,“,”.“	Сума

4.3. Звіти

При умові генерації системою звітів наводиться перелік звітів та їх характеристики.

Якщо є певні регламентовані форми, то наводяться посилання чи макети.

Приклад:

Ідентифікатор звіту	COS-RPT-1
Заголовок звіту	Історія замовлень страв
Мета звіту	Клієнт бажає побачити перелік всіх страв, які він раніше замовляв в кафетерії чи локальних ресторанах за певний період часу до пів року до поточної дати, для того, щоб ще раз замовити страви, що сподобалися
Пріоритет	Середній
Користувачі звіту	Постійні клієнти кафетерію

Джерела даних	База даних про раніш розміщені замовлення
Частота та використання	Звіт генерується за запитом клієнта. Дані в звіті статичні. Звіт відображається у вікні веб-браузера користувача на комп'ютері, планшеті чи смартфоні. Його можна надрукувати, якщо пристрій підтримує друк
Час доступу	Готовий звіт повинен відображатися протягом 3 сек. після його запиту
Візуальний макет	Альбомна орієнтація
Верхній та нижній колонтитули	Верхній колонтитул повинен містити заголовок звіту, ім'я клієнта. Нижній – діапазон дат. При друці в нижньому колонтитулі повинен міститися номер сторінки
Тіло звіту	Назви стовбців: Номер замовлення Дата замовлення Місце замовлення (кафетерій/ресторан) Склад замовлення (перелік всіх страв у замовленні з вказанням кількості та вартості) Загальна вартість Податок Вартість доставки Всього (сума вартості страв з урахуванням податку та вартості доставки) Критерій відбору (діапазон дат, що вибрав клієнт, включаючи початкову та кінцеву дати) Критерій сортування: зворотній хронологічний порядок
Ознака кінця звіту	Немає
Інтерактивність	Клієнт може дивитися подробиці інгредієнтів та відомості про харчову цінність для кожної страви
Обмеження безпеки доступу	Клієнт може дивитися історію тільки власних замовлень

4.4. Отримання, цілісність, зберігання та утилізація даних

При необхідності наводиться опис як дані надходять до системи та як вони обслуговуються. Перелічуються вимоги до цілісності даних системи. Вказуються вимоги до резервного копіювання, перевірки коректності, створення контрольних точок, дзеркального відображення.

Перелічуються політики щодо зберігання чи утилізації даних, в тому числі тимчасових даних, метаданих, кінцевих даних, даних кешу, локальних копій, архівів, проміжних архівів.

5. ВИМОГИ ДО ЗОВНІШНІХ ІНТЕРФЕЙСІВ

5.1. Користувацькі інтерфейси

Опис логічних характеристик кожного користувацького інтерфейсу (вимоги щодо зручності використання, наприклад:

посилання на стандарти графічного інтерфейсу користувачів чи стильові рекомендації для родини продуктів;

стандарти шрифтів, позначок, назв кнопок, зображень, кольорових схем, послідовність полів вкладок, часто використовувані елементи управління, графіки фірмового стилю, повідомлення про зареєстровані товарні знаки та про конфіденційність;

розмір, конфігурація екрану чи обмеження роздільної здатності;

сполучення клавіш;

стандартні перевірки даних;

спеціальні можливості для користувачів з вадами зору тощо).

Приклад.

Таблиця 5.1.1. Повідомлення

Номер повідомлення	Дія/операція	Тип повідомлення	Зміст повідомлення
1.	Перевірка коректності заповнення форми автентифікації користувача	Помилка	Текст повідомлення щодо некоректності заповнення форми автентифікації користувача
2.	Перевірка наявності користувача серед	Помилка	Користувача за даними адреси електронної пошти не знайдено в списку користувачів

Номер повідомлення	Дія/операція	Тип повідомлення	Зміст повідомлення
	зареєстрованих співробітників компанії для роботи в Системі		
3.	Інформування користувача щодо умов подання заявки на сплату страв із заробітної плати	Інформація	Після того як заявку буде відправлено, вона стане недоступною для редагування. Для того щоб надіслати заявку натисніть кнопку «Відправити заявку».

5.2. Інтерфейси ПЗ

Опис зав'язків системи з іншими додатками, базами даних, операційними системами, веб-сайтами тощо. Вказуються призначення, формати та зміст повідомлень, даних, якими обмінюється система. Опис відповідності між вхідними та вихідними даними, перетворення даних.

Визначаються нефункціональні вимоги, що впливають на інтерфейс (наприклад, частота відклику, обмеження безпеки).

5.3. Інтерфейси обладнання

Опис характеристик кожного інтерфейсу між компонентами ПЗ та обладнанням системи (наприклад, типи пристроїв, елементи управління ПЗ та обладнанням, протоколи взаємодії). Вказуються всі вхідні та вихідні дані, їх формат, дозволені значення та їх діапазони, часові обмеження.

5.4. Комунікаційні інтерфейси

Опис функцій взаємодії продукту з електронною поштою, веб-браузером, опис електронних форм та мережевих протоколів. Визначаються формати повідомлень, особливості безпеки чи шифрування даних, швидкість передачі даних, механізмів узгодження та синхронізації. Вказуються всі обмеження комунікаційних інтерфейсів, наприклад, допустимість певних типів вкладень у повідомленнях електронної пошти.

6. АТРИБУТИ ЯКОСТІ

6.1. Зручність використання

Перелік вимог до легкості вивчення, простоти використання, попередження помилок, відновлення, ефективності взаємодії, спеціальних вимог.

6.2. Продуктивність

Перелік вимог до продуктивності. Якщо у різних функціональних задач різні вимоги до продуктивності, то необхідно це вказати в розділі з опису функціональних вимог.

6.3. Безпека

Перелік вимог до безпеки та конфіденційності даних, джерелом яких зазвичай є бізнес-правила. Якщо існує окремий довідних бізнес-правил, то тут наводиться відповідне посилання.

6.4. Техніка безпеки

Опис можливих втрат, пошкоджень, витрат, які можуть бути спричинені використанням продукту. Визначаються заходи безпеки, превентивні заходи, потенційно небезпечні дії, які можна попередити. Наводиться перелік сертифікатів та політик безпеки, яким повинен відповідати продукт.

6.5. [Інші]

Для кожного окремого атрибута якості створюється власний розділ в SRS.

7. ВИМОГИ ДО ІНТЕРНАЦІОНАЛІЗАЦІЇ ТА ЛОКАЛІЗАЦІЇ

Опис вимог щодо використання продукту в інших країнах, відповідно до інших регіональних стандартів тощо. Вимоги можуть бути спрямовані на корегування валют, форматування дат, чисел, адрес, телефонних номерів, мов, символів, що використовуються, часових поясів, одиниць ваги, базуватися на культурних та політичних традиціях, розмірі паперу тощо. Такі вимоги можуть бути оформлені окремим документом та використовуватися в інших проєктах.

8. ІНШІ ВИМОГИ

Опис інших вимог, які ще не були описані в SRS. Наприклад, фінансові, юридичні, законодавчі вимоги до інсталяції системи.

ДОДАТОК А. СЛОВНИК ТЕРМІНІВ

Наводяться спеціальні терміни для вірного розуміння SRS. Наводяться скорочення та визначення всіх термінів. Якщо існую

загальний корпоративний словник термінів, то наводиться відповідне посилання.

ДОДАТОК В. МОДЕЛІ АНАЛІЗУ

Наводяться діаграми потоків даних, дерева функцій, діаграми переходів станів тощо.

Приклад:

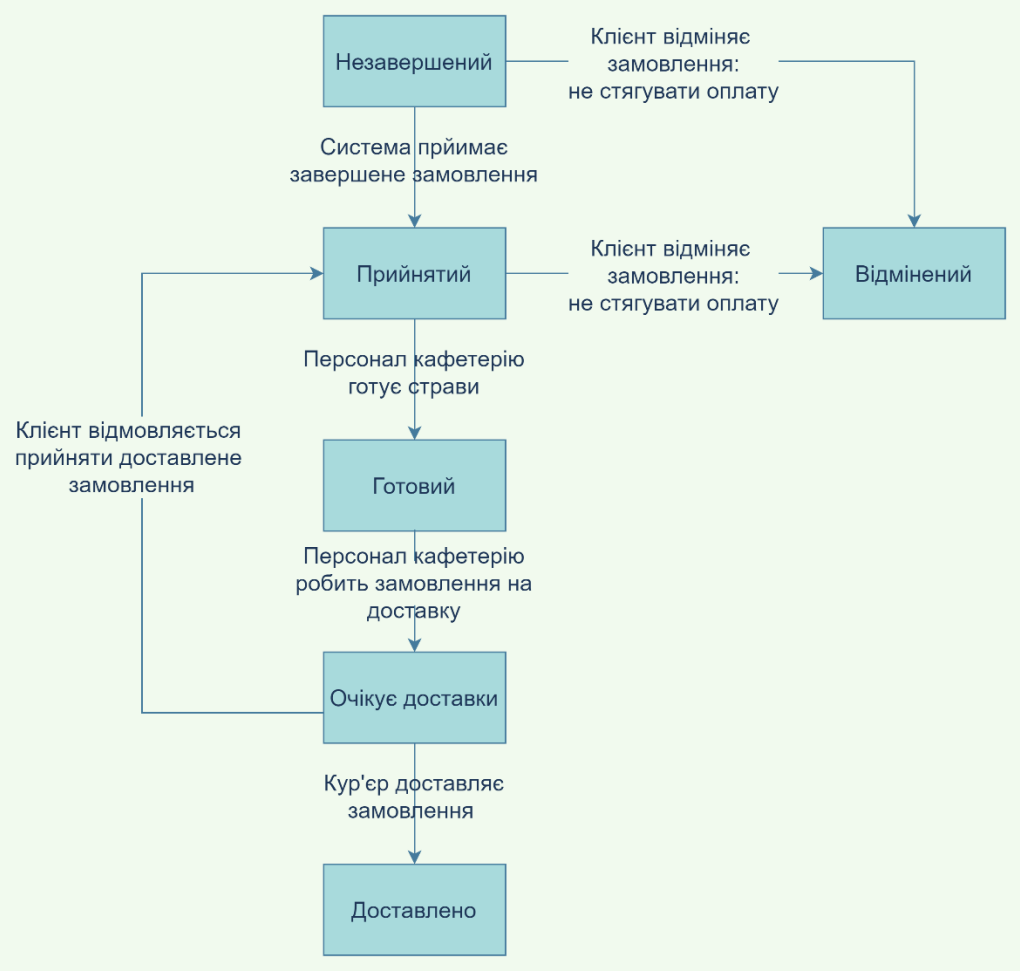


Рис. В.1. Діаграма станів для станів замовлень страв