Ефективність і гнучкість системи нерозривно пов’язані з розумінням потреб клієнтів або користувачів системи. Відповідно, ключовим компонентом будь-якого процесу розробки, який відіграє центральну роль у цьому процесі, є саме специфікація вимог. Процес розробки специфікації вимог називається інженерією вимог.

Багато проблем у сфері програмного забезпечення виникають через недоліки процесів визначення, документування, узгодження вимог до продукту, а також внесення змін до вимог. До проблем зазвичай призводять неформальний збір інформації, неправильні припущення, погано визначені вимоги та неконтрольований процес внесення змін. Різноманітні дослідження показують, що помилки, які виникли під час роботи з вимогами, становлять від 40 до 50 відсотків усіх дефектів, виявлених у програмному продукті. Неадекватні дані, отримані від користувачів, і недоліки у специфікації та управлінні вимогами клієнтів є основними факторами неуспішності проектів. Незважаючи на результати досліджень, багато організацій все ще практикують неефективні методи роботи з вимогами.

Саме на етапі визначення вимог найбільше перетинаються інтереси всіх зацікавлених сторін у проекті. Ці зацікавлені сторони, стейкхолдери, включають клієнтів, користувачів, бізнес-аналітиків, розробників та багатьох інших. При ефективному управлінні процесом визначення вимог, це перехрестя інтересів може призвести до задоволених клієнтів і задоволених розробників. В іншому випадку перехрестя інтересів може стати джерелом непорозумінь і суперечок, які підривають якість і бізнес-цінність продукту. Оскільки вимоги є основою як для розробки програмного забезпечення, так і для діяльності з управління проектами, усі стейкхолдери повинні взяти на себе зобов’язання застосовувати хороші практики вимог, які, як відомо, дають продукти найвищої якості.

**Визначення вимог**

(1) Деякі умови або можливість, необхідні користувачеві для вирішення проблеми або досягнення мети.

(2) Умови або здатність, яким повинна відповідати або володіти система або системний компонент, щоб задовольнити контракт, стандарт, специфікацію чи інші офіційно встановлені документи.

(3) Задокументоване представлення умов або можливостей, як у (1) або (2).

(4) Вимоги - це специфікація того, що має бути реалізовано. Це опис того, як повинна поводитися система, або властивостей чи атрибутів системи. Вони можуть бути деяким обмеженням процесу розробки системи.

Коли група людей починає обговорювати вимоги, вони часто стикаються з термінологічною проблемою. Різні спостерігачі можуть класифікувати одне судження як вимогу користувача, вимогу до програмного забезпечення, бізнес-вимогу, функціональну вимогу, системну вимогу, вимогу до продукту, вимогу до проекту, історію користувача, функцію або обмеження. Тому не припускайте, що всі зацікавлені сторони вашого проекту мають спільне уявлення про вимоги. Сформулюйте визначення наперед, щоб ви всі говорили про ті ж самі речі.

**Типи вимог**

Загалом вимоги поділяються на дві великі категорії: “орієнтовані на ринок” та “специфічні для клієнта”. Ці дві широкі категорії вимог мають різні характеристики і часто по-різному розглядаються в процесі розробки. Розробка інформаційних систем полягає не лише в проектуванні структур баз даних і алгоритмів, але й у розумінні потреб кінцевих користувачів та інших стейкхолдерів у межах підприємства та забезпеченні відповідності системи вимогам користувачів і бізнес-стратегії.

Оскільки існує дуже багато різних типів інформації про вимоги, були встановлені певні терміни для цих типів, щоб замінити перевантажений термін «вимога». Вимоги до програмного забезпечення включають три різні рівні: бізнес-вимоги, вимоги користувача та функціональні вимоги. Крім того, кожна система має ряд нефункціональних вимог.

Хоча зібрані пакети вимог називаються «документами», це не обов’язково повинні бути традиційні паперові або електронні документи. Їх краще представляти як контейнери для зберігання знань про вимоги. Такий контейнер дійсно може бути традиційним документом, або це може бути електронна таблиця, набір діаграм, база даних, інструмент керування вимогами або деяка комбінація усього наведеного.

| Термін | Визначення |
| --- | --- |
| Бізнес-вимога | Бізнес-ціль високого рівня організації, яка створює продукт, або замовника, який його закуповує. |
| Бізнес-правило | Політика, директива, стандарт або нормативний акт, який визначає або обмежує певний аспект бізнесу. Не вимога програмного забезпечення сама по собі, а джерело декількох видів вимог. |
| Обмеження | Обмеження, яке накладається на альтернативи, доступні розробнику при проектуванні та конструюванні продукту. |
| Вимога до зовнішнього інтерфейсу | Опис зв’язку між системою та користувачем, іншою програмною системою або апаратним пристроєм. |
| Характеристика (feature) | Одна або декілька логічно пов’язаних системних можливостей, які необхідні користувачу та описуються набором функціональних вимог. |
| Функціональна вимога | Опис поведінки, яку система демонструватиме за певних умов. |
| Нефункціональна вимога | Опис властивості або характеристики, яку має демонструвати система, або обмеження, якого вона повинна дотримуватися. |
| Атрибут якості | Різновид нефункціональної вимоги, що описує сервіс або характеристику продуктивності системи. |
| Системна вимога | Вимога верхнього рівня для продукту, який містить декілька підсистем, які можуть бути всі програмними або програмними та апаратними. |
| Вимога користувача | Цілі чи завдання, які певні класи користувачів повинні бути в змозі виконати за допомогою системи, або бажаний атрибут продукту. |

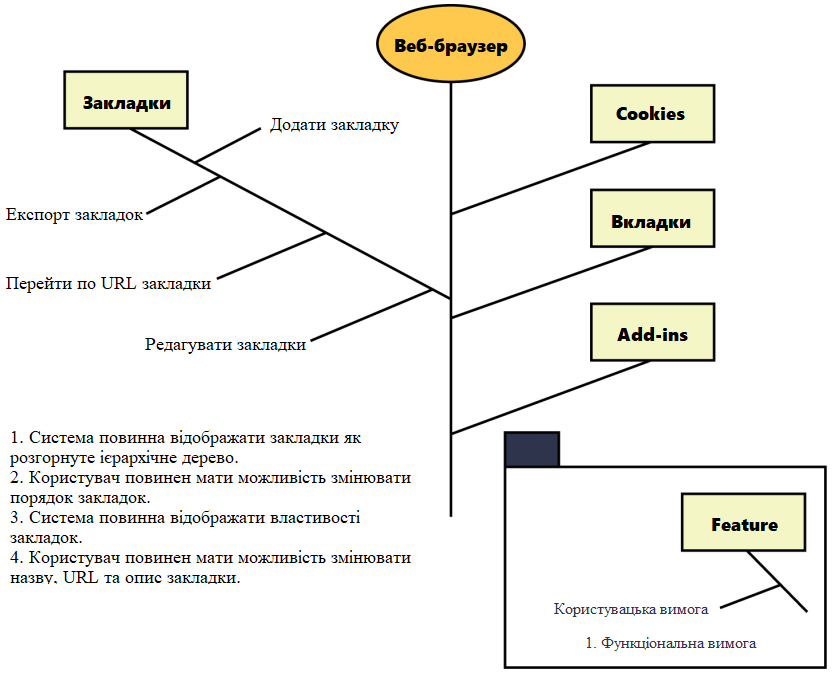
**Бізнес-вимоги** описують, чому організація впроваджує систему — бізнес-вигоди, яких організація сподівається отримати. Основна увага приділяється бізнес-цілям організації або клієнта, які замовили систему. Бізнес-вимоги зазвичай надходять від спонсора проекту, замовника, менеджера кінцевих користувачів, відділу маркетингу або візіонера продукту. Зазвичай бізнес-вимоги фіксуються в документі концепції та обсягу проекту. Інші документи, які іноді використовуються для цієї мети, включають статут проекту, бізнес-кейси та документ про ринкові (або маркетингові) вимоги.

**Бізнес-правила** включають корпоративну політику, державні постанови, галузеві стандарти та обчислювальні алгоритми. Бізнес-правила самі по собі не є вимогами до програмного забезпечення, оскільки вони існують за межами будь-якої конкретної програмної програми. Однак вони часто диктують, що система повинна містити функціональні можливості для дотримання відповідних правил. Іноді, як і у випадку з корпоративною політикою безпеки, бізнес-правила є джерелом конкретних атрибутів якості, які потім реалізуються у функціональності. Таким чином, ви можете простежити походження певних функціональних вимог від певного бізнес-правила.

**Вимоги користувача** описують цілі або завдання, які користувачі повинні мати можливість виконати за допомогою системи. Користувацькі вимоги також включають опис атрибутів продукту або характеристик, які потрібні користувачу. Способи представлення вимог користувачів включають варіанти використання (use-cases), історії користувачів і таблиці “подія-відповідь”. В ідеалі цю інформацію нададуть представники реальних користувачів. Користувацькі вимоги описують, що користувач зможе робити в системі. *Наприклад для веб-сайту авіакомпанії має бути варіант використання «Реєстрація на рейс». Написана як історія користувача, та сама вимога може звучати так: «Як пасажир, я хочу зареєструватися на рейс, щоб я міг сісти на борт свого літака».* Важливо пам’ятати, що більшість проектів мають декілька класів користувачів, а також інших зацікавлених сторін, чиї потреби також мають бути виявлені. Для цього використовується ширший термін «вимоги стейкхолдерів».

**Характеристика (feature)** складається з однієї або декількох логічно пов’язаних системних можливостей, які є цінними для користувача та описуються набором функціональних вимог. Список характеристик продукту, які бажає отримати клієнт, не є еквівалентним опису потреб користувача, пов’язаних із виконанням певних задач. *Закладки веб-браузера, перевірка орфографії, можливість визначати спеціальну програму тренувань для тренажера та автоматичне оновлення сигнатур вірусів у продукті для захисту від зловмисного програмного забезпечення є прикладами характеристик.* Вони можуть охоплювати декілька користувацьких вимог, кожна з яких передбачає виконання певних функціональних вимог, щоб дозволити користувачеві виконувати завдання, описане кожною користувацькою вимогою.

Дерево характеристик показує, як характеристику можна ієрархічно розкласти на набір менших характеристик, які стосуються конкретних вимог користувача та призводять до визначення наборів функціональних вимог.



**Функціональні вимоги** визначають поведінку, яку демонструватиме продукт за певних умов. Вони описують те, що розробники повинні реалізувати, щоб дозволити користувачам виконувати свої задачі (вимоги користувача), таким чином задовольняючи бізнес-вимоги. Таке узгодження між трьома рівнями вимог є важливим для успіху проекту. Функціональні вимоги часто записуються у формі традиційних тверджень «повинен мати»: *«Пасажир повинен мати можливість роздрукувати посадкові талони для всіх сегментів польоту, на які він зареєструвався» або «Якщо в профілі Пасажира не вказано місце для сидіння, система бронювання повинна призначити місце».* Бізнес-аналітик (БА) документує функціональні вимоги в специфікації вимог до програмного забезпечення (SRS), яка настільки повно описує очікувану поведінку програмної системи, наскільки це необхідно. SRS використовується в розробці, тестуванні, забезпеченні якості, управлінні проектом та інших пов’язаних функціях проекту. Також цей контейнер вимог називається документом бізнес-вимог, функціональною специфікацією, документом вимог тощо. SRS може бути представлений у вигляді звіту, створеного на основі інформації, що зберігається в інструменті керування вимогами (requirements management tool).

Крім функціональних вимог, SRS містить ряд **нефункціональних вимог**. Нефункціональні вимоги можуть визначати не те, що система робить, а радше те, наскільки добре вона виконує ці речі. Вони можуть описувати важливі характеристики або властивості системи. До них належать доступність системи, зручність використання, безпека, продуктивність та багато інших характеристик, не лише атрибути якості. Наприклад, обмеження дизайну та впровадження також є нефункціональними вимогами, як і вимоги зовнішнього інтерфейсу.

Інші нефункціональні вимоги стосуються середовища, в якому працює система, наприклад платформи, портативності, сумісності та обмежень. На багато продуктів також впливають вимоги відповідності (наказам, стандартам тощо), нормативні та сертифікаційні вимоги. Це можуть також бути вимоги до локалізації продуктів, які мають враховувати культуру, мови, закони, валюти, термінологію, правопис та інші характеристики користувачів. Незважаючи на те, що такі вимоги визначені в нефункціональних термінах, бізнес-аналітик, як правило, отримує з них досить велику кількість інформації про необхідну для реалізації функціональність, щоб переконатися, що система має всі бажані поведінки та властивості.

**Системні вимоги** описують вимоги до продукту, який складається з декількох компонентів або підсистем (ISO/IEC/IEEE 2011). «Система» в цьому сенсі — це не просто будь-яка інформаційна система. Система може бути повністю програмною, або вона може включати як програмну, так і апаратну підсистеми. Люди та процеси також є частиною системи, тому виконання певних системних функцій може бути покладене на людей. *Хорошим прикладом такої системи є робоче місце касира в супермаркеті. Є сканер штрих-кодів, оснащений вагами, а також ручний сканер штрих-кодів. Касир використовує клавіатуру, дисплей, касовий ящик. Крім того є пристрій для зчитування карток і введення PIN-коду, а також, можливо, автомат для видачі здачі. І ще є декілька принтерів для квитанцій, чеків, купонів тощо. Усі ці апаратні пристрої взаємодіють під управлінням програмного забезпечення.* Таким чином, вимоги до системи або продукту в цілому змушують бізнес-аналітика визначати конкретні функціональні можливості, які повинні бути призначені тій чи іншій із цих складових підсистем, а також вимагати узгодження взаємодії між ними.

Щоб проілюструвати деякі з цих різноманітних вимог, розглянемо проект розробки наступної версії програми текстового редактора. Бізнес-вимогою може бути «Збільшення продажів за межами США на 25 відсотків протягом 6 місяців». Відділ маркетингу визначає, що конкурентні продукти мають лише засоби перевірки орфографії англійською мовою, тому вони вирішили, що нова версія включатиме функцію перевірки орфографії на декількох мовах. Відповідні вимоги користувача можуть включати такі завдання, як «Вибір мови для перевірки орфографії», «Знайти орфографічні помилки» та «Додати слово до словника». Перевірка орфографії має багато індивідуальних функціональних вимог, які стосуються таких операцій, як виділення слів з орфографічними помилками, автовиправлення, відображення запропонованих замін слів і глобальна заміна слів з орфографічними помилками виправленими словами. Вимоги до зручності використання визначають, як програмне забезпечення має бути локалізовано для його використання з певними мовами та наборами символів.

**Вимоги до продукту та вимоги до проекту**

Проекти мають очікування та результати, які не є частиною програмного забезпечення, яке впроваджує команда, але є необхідними для успішного завершення проекту в цілому. Це вимоги до проекту, але не до продукту. SRS містить вимоги до продукту, але цей документ не повинен включати деталі дизайну чи реалізації (окрім відомих обмежень), плани проекту, плани тестування чи подібну інформацію. Варто відокремлювати такі елементи, щоб в процесі розробки вимог можна було зосередитися на розумінні того, що команда збирається створити. Вимоги до проекту включають:

1. Фізичні ресурси, необхідні команді розробників, наприклад робочі станції, спеціальні апаратні пристрої, тестові лабораторії, інструменти та обладнання для тестування, кімнати для команд і обладнання для відеоконференцій.
2. Все, що потрібно для навчання персоналу.
3. Документація користувача, включаючи навчальні матеріали, підручники, довідкові посібники та примітки до релізу.
4. Допоміжна документація, наприклад ресурси довідкової служби та інформація про технічне обслуговування та сервісне обслуговування апаратних пристроїв.
5. Необхідні зміни в інфраструктурі операційного середовища.
6. Вимоги та процедури для випуску продукту, його установки в операційному середовищі та налаштування, тестування установки.
7. Вимоги та процедури для переходу від старої системи до нової, такі як вимоги до міграції та перетворення даних, налаштування безпеки, перемикання виробництва та навчання для усунення прогалин у навичках персоналу; їх іноді називають вимогами переходу (IIBA 2009).
8. Сертифікація продукції та вимоги до відповідності.
9. Оновлені (затверджені) політики, процеси, організаційні структури та подібні документи.
10. Пошук, придбання та ліцензування програмного та апаратного забезпечення сторонніх виробників.
11. Вимоги до бета-тестування, виробництва, упаковки, маркетингу та розповсюдження.
12. Угоди на рівні обслуговування клієнтів.
13. Вимоги щодо отримання правового захисту (патентів, торгових марок або авторських прав) інтелектуальної власності, пов’язаної з програмним забезпеченням.

**Стейкхолдери**

Зацікавлена ​​сторона — це особа, група чи організація, яка бере активну участь у проекті, на яку впливає його процес чи результат або яка сама впливає на процес чи результат проекту. Стейкхолдери можуть бути внутрішніми або зовнішніми стосовно команди проекту та організації-розробника. Звичайно, не всі перераховані в таблиці стейкхолдери стосуватимуться кожного проекту чи ситуації.

| **За межами організації-розробника** | | | |
| --- | --- | --- | --- |
| Безпосередній користувач | Експерт | Державна установа | Бета-тестер |
| Замовник | Консультант | Аудитор | Юридичний персонал |
| Підрядчик | Постачальник | Інвестор | Громадськість |
| **В межах організації-розробника** | | | |
| Менеджер розробки | Менеджмент проекту | Юзабіліті-експерт | Юридичний персонал |
| Маркетинг | Експерт | Супроводжувач | Власник компанії |
| Інсталятор | Тренінг-персонал | Проектувальник | Виконавчий спонсор |
| **Команда проекту** | | | |
| Менеджер проекту | Менеджер продукту | Складач документації | Інженер апаратного забезпечення |
| Дизайнер | Адмін БД | Бізнес-аналітик | Контролер якості |
| Розробник | Тестувальник | Проектувальник | Аналітик |

Аналіз зацікавлених сторін є важливою частиною розробки вимог. Шукаючи потенційних стейкхолдерів для конкретного проекту, варто розглянути всі більш-менш підходящі варіанти, щоб не залишити поза увагою якусь важливу спільноту. Потім ви можете звузити цей список кандидатів, залишивши основну групу зацікавлених сторін, чий внесок вам дійсно потрібен. Таким чином можна переконатися, що ви розумієте всі вимоги та обмеження проекту, щоб ваша команда могла видати продукт, який задовольнить всі сторони.

Клієнти — це підмножина зацікавлених сторін. Клієнт — це особа або організація, яка отримує пряму чи непряму вигоду від продукту. Клієнти програмного забезпечення можуть запитувати, оплачувати, вибирати, вказувати, використовувати або отримувати результати, створені програмним продуктом. Клієнти включають безпосереднього користувача, непрямого користувача, виконавчого спонсора, персонал із закупівель та замовника. Деякі зацікавлені сторони не є клієнтами, наприклад юридичний персонал, аудитори з відповідності, постачальники, підрядники та венчурні капіталісти.

Користувацькі вимоги мають надходити від людей, які фактично будуть використовувати продукт, безпосередньо чи опосередковано. Ці користувачі (їх часто називають кінцевими користувачами) є підмножиною клієнтів. Безпосередні користувачі будуть працювати/взаємодіяти з продуктом. Непрямі користувачі можуть отримувати результати від системи, не торкаючись її самостійно, наприклад, менеджер складу, який отримує електронним листом автоматичний звіт про щоденну складську діяльність. Користувачі можуть описати завдання, які їм потрібно виконати з продуктом, результати, які треба отримати в процесі роботи, та якісні характеристики, які користувачі очікують від продукту.

Стосовно програмних проектів треба прийняти безліч рішень, часто досить критичних, щоб мати можливість рухатися вперед. Можливо, вам знадобиться вирішити певний конфлікт, прийняти (або відхилити) запропоновану зміну або схвалити набір вимог для певного релізу. Групі прийняття рішень необхідно визначити свого керівника та вибрати правило прийняття рішень, яке описує, як вони прийдуть до своїх рішень. Існує багато правил прийняття рішень, які можна застосувати, включаючи наступні:

* Лідер, який приймає рішення, робить вибір разом з іншими або без них.
* Група голосує і приймається рішення з більшістю голосів.
* Група голосує, але для схвалення рішення результат має бути одноголосним.
* Група обговорює та веде переговори для досягнення консенсусу. Кожен з групи зобов’язується підтримувати прийняте рішення.
* Лідер, який приймає рішення, делегує повноваження щодо прийняття рішення одній особі.
* Група приймає рішення, але якась особа має право вето на це рішення.

Існує також важлива концепція встановлення базової лінії на погодження з визначеними вимогами, такі спеціальні маркери на певні моменти в процесі визначення вимог. Базова лінія вимог — це набір вимог, який було переглянуто та узгоджено на певний момент часу, та який служить основою для подальшої розробки вимог, для продовження процесу визначення вимог. Після того, як особи, які приймають рішення, визначать базову лінію, БA повинен контролювати всі зміни, які вносяться до вимог. Це дозволяє команді змінювати обсяг проекту, коли це необхідно, контрольованим способом, що включає аналіз впливу кожної запропонованої зміни на усталений графік робіт та інші фактори успіху. Фіксація початкових дій із розробки вимог за допомогою чіткої угоди допомагає налагодити спільне партнерство між клієнтом і розробником на шляху до успішного завершення проекту.

**Визначення бізнес-вимог**

«Бізнес-вимоги» - це набір інформації, яка в сукупності описує проблему, заради вирішення якої створюються один або декілька проектів, тобто для вирішення конкретних задач, задоволення певних потреб та отримання бажаних кінцевих бізнес-результатів. Ділові нагоди, бізнес-цілі, показники успіху та бачення (концепції) проекту складають бізнес-вимоги.

Проблеми в процесі визначення бізнес-вимог повинні бути вирішені до того, як можна буде повністю визначити функціональні та нефункціональні вимоги. Затвердження обсягу та обмежень проекту дуже допомагає в обговоренні запропонованих функцій і цільових релізів. В подальшому на бізнес-вимоги посилаються в процесі прийняття рішень щодо запропонованих змін і вдосконалень вимог. Під час кожного сеансу виявлення вимог слід мати під рукою основні визначені бізнес-цілі, бачення та обсяг, щоб команда могла швидко визначити, чи знаходиться запропонована вимога в межах чи поза межами проекту.

Бізнес-вимоги можуть надходити від фінансуючих спонсорів, керівників компаній, менеджерів з маркетингу чи візіонерів продукту. Бізнес-вимога має представляти справжню цінність для спонсорів проекту та клієнтів продукту. Наприклад, просте об’єднання двох систем в одну не є розумною бізнес-метою. Клієнтам байдуже, чи використовують вони програму, яка включає 1, 5 чи навіть 10 систем. Їх цікавлять такі питання, як збільшення доходу та зменшення витрат. Об’єднання двох систем може бути частиною рішення, але рідко це справжня бізнес-ціль. Проекти нормативно-правової відповідності також мають чіткі бізнес-цілі. Часто ці цілі формулюються як уникнення ризику, наприклад, щоб уникнути судового процесу або вилучення з бізнесу.

Двома основними елементами бізнес-вимог є концепція та обсяг. Концепція продукту стисло описує кінцевий продукт, який дозволить досягти бізнес-цілей. Цей продукт може служити повним рішенням для бізнес-вимог або лише частиною рішення. Концепція описує, про що продукт і чим він може стати в кінцевому підсумку. Це забезпечує контекст для прийняття рішень протягом усього життєвого циклу продукту та об’єднує всіх зацікавлених сторін у спільному напрямку. Обсяг проекту визначає, яку частину кінцевого бачення продукту розглядатиме поточний проект або ітерація розробки. Обсяг проекту визначає межу між тим, що входить до нього, і тим, що в проект включати не планується.

Концепція стосується продукту в цілому, тому має змінюватися відносно повільно, оскільки стратегічне позиціонування продукту або бізнес-цілі компанії змінюються з часом. Обсяг стосується конкретного проекту або ітерації, яка реалізує наступне збільшення функціональності продукту. Тобто для кожного релізу продукту розробляється окремий документ обсягу проекту, і всі ці документи знаходяться в межах однієї розробленої концепції продукту.

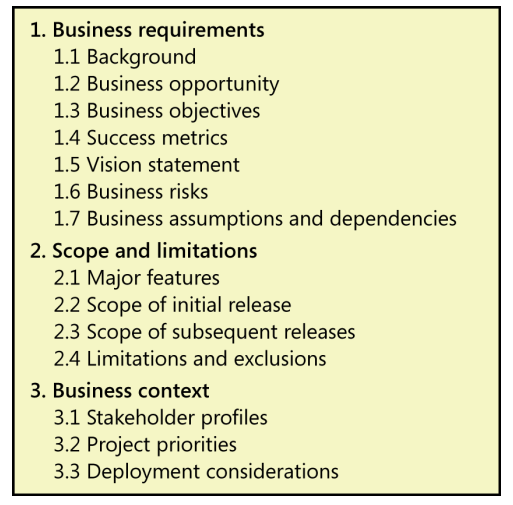
Обсяг є більш динамічним, ніж концепція, оскільки стейкхолдери коригують вміст кожного релізу в межах його графіка, бюджету, ресурсів і обмежень якості. Обсяг поточного випуску має бути чітким, але обсяг випусків, що плануються колись там у майбутньому, буде більш “розмитим”, чим далі, тім “розмитіше”. Мета команди полягає в тому, щоб керувати обсягом конкретного проекту розробки або вдосконалення як визначеної підмножини стратегічного бачення продукту.

**Документ концепції та обсягу**

Документ про концепцію та обсяг збирає бізнес-вимоги в єдиний документ/контейнер, який закладає основу для подальшої розробки. Деякі організації створюють статут проекту або діловий документ, який служить подібній меті. Організації, які розробляють комерційне програмне забезпечення, часто створюють документ про ринкові (або маркетингові) вимоги (MRD). Цей документ може детальніше розповісти про цільові сегменти ринку та проблеми, які стосуються комерційного успіху.

Власником документа про концепцію та обсяг проекту є виконавчий спонсор проекту, орган фінансування або хтось із подібною роллю. Бізнес-аналітик може працювати з цією особою, щоб сформулювати бізнес-вимоги та створити цей документ. Внесок у бізнес-вимоги має надходити від людей, які чітко розуміють, чому вони беруться за проект. Це можуть бути покупці або вище керівництво організації-розробника, візіонер або менеджер продукту, експерти з відповідної сфери або члени відділу маркетингу.

Документ про концепцію та обсяг визначає лише обсяг на високому рівні, деталі обсягу визначає команда на кожній усталеній базовій лінії релізу. Великі нові проекти повинні мати як повний документ про концепцію та обсяг, так і спеціфікацію (SRS). Кожна ітерація, випуск або проект удосконалення для продукту, що розвивається, може включати власне визначення обсягу у загальну документацію щодо вимог цього проекту, а не створювати окремий документ про концепцію та обсяг.

Як і будь-який інший шаблон, шаблон документа концепції та обсягу має бути адаптований відповідно до конкретних потреб певного проекту. Деякі елементи документа концепції та обсягу можуть бути повторно використані від проекту до проекту, наприклад бізнес-цілі (objectives), бізнес-ризики та профілі стейкхолдерів.

Не треба заповнювати кожен пункт шаблону в тому порядку, в який представлено на рисунку вище. Замість цього різні розділи заповнюються в міру накопичення інформації в ході проекту. Порожні розділи підкреслюють прогалини в поточних знаннях про проект. Якщо певний розділ шаблону, скажімо, бізнес-ризики, не стосується поточного проекту, найкраще рішення — розмістити в цьому розділі чітке повідомлення: «Бізнес-ризики не виявлено».

Якщо певні розділи шаблону використовуються рідко, видаліть їх. Бажано створити невеликий набір шаблонів для використання в проектах різних типів, наприклад шаблони SRS, придатні для використання у великих нових проектах розробки; при розробці невеликих сайтів, проектів вдосконалення тощо. Навіть якщо ви зберігаєте свої вимоги в якомусь сховищі, відмінному від традиційного документа, шаблон може допомогти вам розглянути всю інформацію про вимоги, яку потрібно зібрати для свого проекту.

**Методи представлення обсягу проекту**

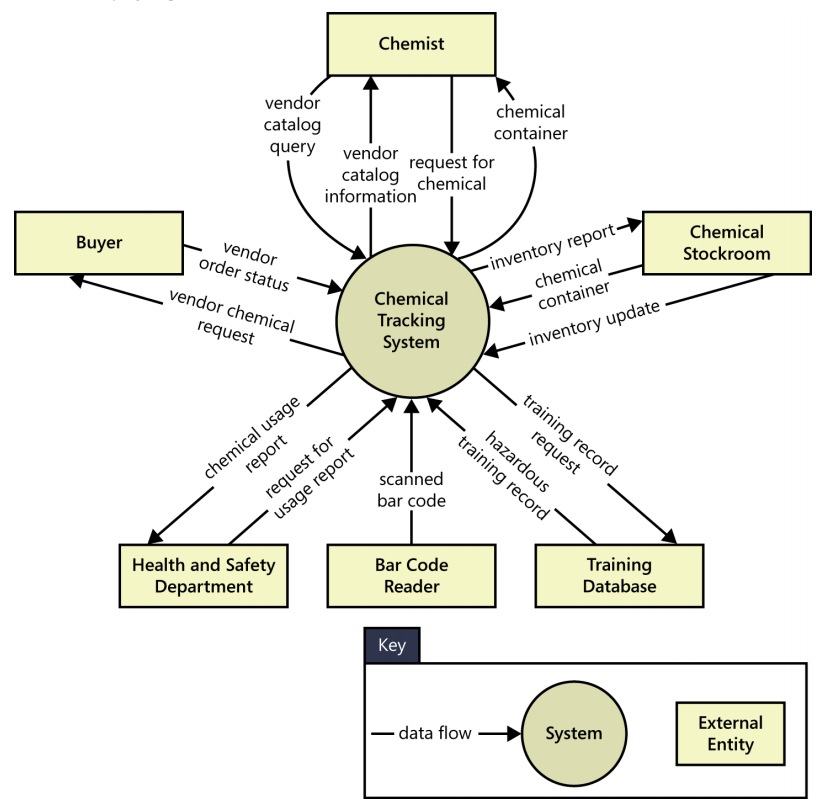
Призначення таких інструментів, як контекстна діаграма, карта екосистеми, дерево характеристик (feature tree) і список подій, полягає в тому, щоб сприяти якісній комунікації між зацікавленими сторонами проекту. Ця зрозумілість та точність важливіші, ніж догматичне дотримання правил «правильної» діаграми.

**Контекстна діаграма**

Опис обсягу встановлює границі та зв’язки між системою, яку ви розробляєте, та всім іншим у всесвіті. Контекстна діаграма наочно ілюструє ці границі. Діаграма ідентифікує зовнішні сутності (також звані термінаторами) поза системою, які певним чином з нею взаємодіють, а також потоки даних, управління та матеріалів між термінаторами та системою. Контекстна діаграма є верхнім рівнем діаграми потоку даних, розробленої відповідно до принципів структурованого аналізу, але ця модель корисна для всіх проектів.

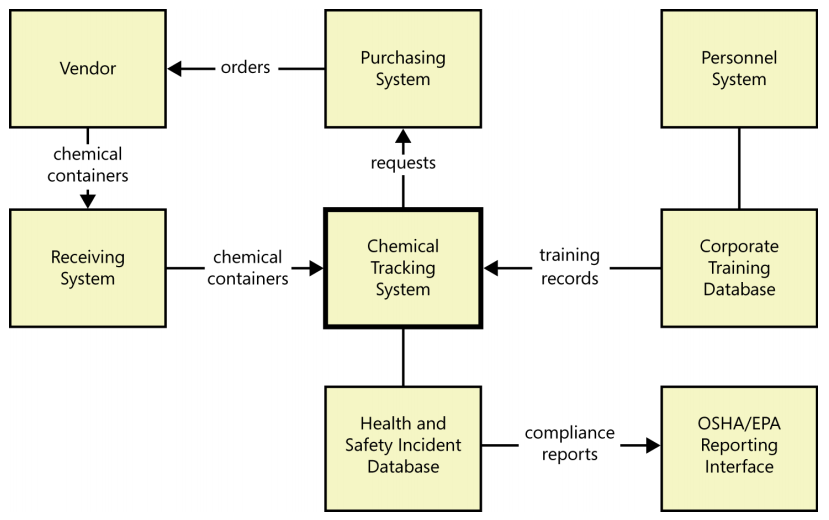
Рисунок на наступній сторінці ілюструє часткову контекстну діаграму для системи хімічного відстеження. Вся система зображена як одне коло, бо контекстна діаграма навмисно не забезпечує видимості внутрішніх об’єктів, процесів або даних системи. «Система» всередині кола може охоплювати будь-яку комбінацію програмного забезпечення, апаратного забезпечення та працівників/користувачів. Зовнішні сутності в прямокутниках можуть представляти класи користувачів (Хімік/Chemist, Покупець/Buyer), організації (Відділ охорони здоров’я та безпеки / Health and Safety Department), інші системи (Навчальна база даних / Training Database) або апаратні пристрої (Зчитувач штрих-коду / Bar Code Reader). Стрілки на діаграмі представляють потік даних (наприклад, запит на хімікат / request for chemical) або потік фізичних елементів (наприклад, контейнер для хімікатів / chemical container) між системою та її зовнішніми об’єктами.

Можна було б подумати, що постачальники хімічних речовин на цій діаграмі мають бути показані як зовнішня сутність. Зрештою, компанія направлятиме замовлення постачальникам для виконання, постачальники надсилатимуть хімічні контейнери та рахунки-фактури до компанії, а відділ закупівель оплачуватиме рахунки постачальникам. Однак ці процеси відбуваються поза сферою дії “Системи відстеження хімічних речовин”, як частина операцій відділів закупівлі та прийому. Їх відсутність у контекстній діаграмі дає зрозуміти, що ця система не бере безпосередньої участі в розміщенні замовлень у постачальників, отриманні продуктів або оплаті рахунків.



**Карта екосистеми**

Карта екосистеми показує всі системи, пов’язані з поточною системою, які взаємодіють одна з одною, і характер цих взаємодій. Карта екосистеми демонструє обсяг, показуючи всі системи, які взаємопов’язані та які, можливо, потребують модифікації для пристосування до нової системи. Карти екосистем відрізняються від контекстних діаграм тим, що вони показують інші системи, які мають зв’язок із системою, над якою ви працюєте, включно з тими, які не мають прямих інтерфейсів. Ви можете ідентифікувати системи, на які здійснюється вплив, визначивши, які з них споживають дані з вашої системи. Коли ви визначите всі можливі дані, на які впливає ваш проект, ви визначите обсяг пов’язаних систем.



На рисунку вище представлена часткова карта екосистеми для системи відстеження хімічних речовин. Усі системи відображаються в полях (наприклад, система закупівель / Purchasing System, система отримання / Receiving System). У цьому прикладі основна система, над якою ми працюємо, виділена жирним шрифтом (Система відстеження хімічних речовин / Chemical Tracking System), але якщо всі системи мають однаковий статус у вашому проекті, ви можете використовувати той самий стиль поля для них. Лінії зі стрілками показують наявність потоків даних між системами (наприклад, зв’язок системи закупівель із системою відстеження хімічних речовин). Мітки на стрілках показують, які основні фрагменти даних переходять від однієї системи до іншої (наприклад, записи про навчання (training records) передаються з корпоративної бази даних навчання (Corporate Training Database) до системи відстеження хімічних речовин). Деякі з цих самих потоків також можуть відображатися на контекстній діаграмі.

**Дерево характеристик**

Дерево характеристик (feature) — це візуальне зображення характеристик продукту, організованих у логічні групи, з ієрархічним поділом кожної характеристики на додаткові рівні деталізації. Дерево забезпечує стислий огляд усіх характеристик, запланованих для проекту, що робить його ідеальною моделлю для демонстрації стейкхолдерам, які хочуть швидко ознайомитись із обсягом проекту. Дерево може відображати до трьох рівнів характеристик, які зазвичай називають рівнем 1 (L1), рівнем 2 (L2) і рівнем 3 (L3). Характеристики L2 є підхарактеристиками для L1, а характеристики L3 є підхарактеристиками для L2.

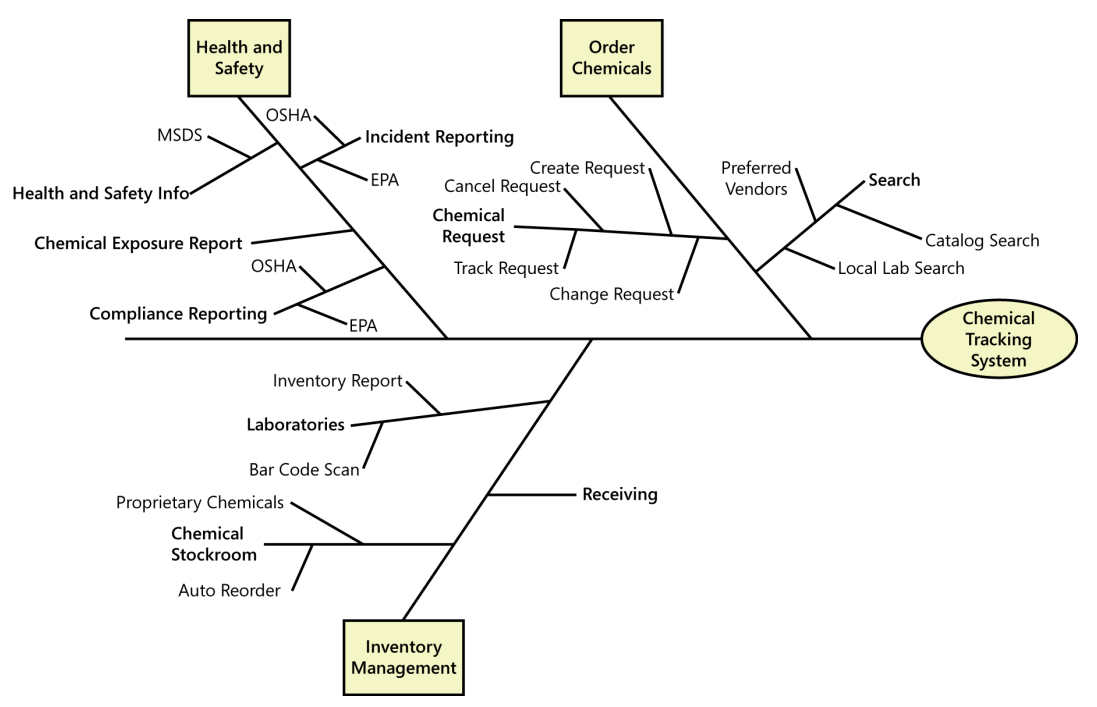


Рисунок наведений вище показує часткове дерево характеристик для системи відстеження хімічних речовин. Головна гілка дерева посередині представляє продукт, який реалізується. Кожна характеристика має власну лінію або «гілку», що відходить від центральної головної гілки. Прямокутники представляють такі характеристики рівня L1, як замовлення хімікатів (Order Chemicals) і керування запасами (Inventory Management). Рядки, що відходять від гілки L1, є характеристиками L2: пошук (Search) і запит хімікатів (Chemical Request) є підхарактеристиками замовлення хімікатів (Order Chemicals). Відгалуження від гілки L2 є характеристиками L3: пошук в місцевій лабораторії (Local Lab Search) є підхарактеристикою пошуку (Search).

**Список подій (Event list)**

Список подій визначає зовнішні події, які можуть спровокувати певну поведінку системи. Список подій зображує межі сфери дії системи, називаючи можливі бізнес-події, ініційовані користувачами, ініційовані певною датою (або часом) події або сигнальні події, отримані від зовнішніх компонентів, наприклад апаратних пристроїв. Список подій лише їх перераховує, а функціональні вимоги, які описують, як система реагує на ці події, деталізовані в SRS за допомогою таблиць подія-відповідь.

Нижче наведений частковий список подій для системи відстеження хімічних речовин. Кожен елемент у списку вказує, що викликає подію (Хімік щось робить або щось треба зробити за планом у певну дату/час), а також визначає активність для цієї події. Список подій є корисним інструментом визначення обсягу, оскільки ви можете призначити певні події для реалізації в конкретних релізах продукту чи ітераціях розробки.

*Зовнішні події для “Системи відстеження хімічних речовин”*

* Хімік виконує запит хімікатів.
* Треба відсканувати штрих-код контейнера з хімікатами.
* Постачальник випускає новий каталог хімікатів.
* Хімік робить запит на формування звіту про вплив хімічних речовин.
* Нова запатентована хімічна речовина введена у систему.
* Доданий новий постачальник до списку обраних постачальників.
* Постачальник зазначив, що хімікату тимчасово немає в наявності.

Зверніть увагу, як список подій доповнює контекстну діаграму та карту екосистеми. Контекстна діаграма та карта екосистеми спільно описують залучених зовнішніх акторів і системи, тоді як список подій визначає, що ці суб’єкти та системи можуть зробити, щоб спровокувати ту чи іншу поведінку в указаній системі. Ви можете перевірити відповідність списку подій контекстній діаграмі та карті екосистеми, а також його повноту, як описано нижче.

* Чи кожна зовнішня сутність на діаграмі контексту є джерелом якихось подій: «Чи викликають будь-які дії хіміків поведінку в Системі відстеження хімікатів?»
* Чи провокують будь-які системи на карті екосистеми події у вашій системі.
* Чи є у вас для кожної події відповідні зовнішні об’єкти на діаграмі контексту або відповідні системи на карті екосистеми: «Якщо хімічний контейнер можна отримати від постачальника, чи з’являється Постачальник на діаграмі контексту та/або на карті екосистеми?»

Якщо ви виявили невідповідності, подумайте, чи був в моделі пропущений якийсь елемент. У прикладі Постачальник не вказаний на контекстній діаграмі, оскільки “Система відстеження хімічних речовин” не взаємодіє безпосередньо з постачальниками. Однак Постачальник включений в карту екосистеми.

Обсяг не повинен бути статичною структурою, визначеною раз і назавжди. Бізнес-вимоги та розуміння того, як клієнти використовуватимуть продукт, надають цінні інструменти при внесенні змін до обсягу проекту. Зміна обсягу не є поганою річчю, якщо вона допомагає вам спрямовувати проект на задоволення клієнтських потреб, які постійно розвиваються. Інформація в документі про концепцію та обсяг дає змогу оцінити, чи підходять запропоновані вимоги для включення в проект. Ви можете змінити обсяг для майбутньої ітерації або для цілого проекту, якщо це зроблено свідомо, компетентними людьми, для правильних бізнес-цілей, з розумінням і прийняттям компромісів.

Також можливо, що запропонована нова вимога виходить за рамки проекту, але це така гарна ідея, що обсяг слід розширити, щоб відповідати їй, з відповідними змінами в бюджеті, графіку робіт та/або персоналі. Тобто існує цикл зворотного зв’язку між вимогами користувача та вимогами бізнесу. Це вимагатиме від вас оновлення документу концепції та обсягу, отже варто вести облік причин відхилення вимог, бо може знадобитися повернути їх у документ.

**Правильні вимоги**

“Правильно” організовані та проведені процеси визначення вимог спираються на співпрацю при розробці продукту, що передбачає партнерство зацікавлених сторін протягом усього проекту. Виявлення вимог дозволяє команді розробників краще зрозуміти свою спільноту користувачів та ринок, що є критичним фактором успіху.

Явний розподіл системних вимог між програмними, апаратними підсистемами і людьми підкреслює системний підхід до розробки продукту. Ефективний процес контролю за змінами вимог зведе до мінімуму негативний вплив змін. Задокументовані та чіткі вимоги значно полегшують тестування системи. Усе це збільшує шанси на випуск високоякісних продуктів, які задовольнять усіх стейкхолдерів.

Вартість “кращих” вимог включає розробку нових процедур і шаблонів документів, навчання команди та придбання інструментів. Потенційна винагорода включає:

* Менше дефектів у вимогах і в готовому продукті.
* Менше переробки на всіх етапах життєвого циклу проекту.
* Швидша розробка та реліз продукту.
* Менше функцій, які виявилися непотрібними і невикористовуваними.
* Менше витрат на вдосконалення.
* Менше непорозумінь.
* Менше ймовірність внесення змін до обсягу проекту після його затвердження.
* Менше хаосу на всіх етапах життєвого циклу проекту.
* Вища задоволеність клієнтів і членів команди.
* Реліз продуктів, які роблять те, що вони повинні робити.

**Правильні вимоги: уникнення 10 типових помилок**

**Помилка №1: розширення обсягу проекту**

Розширення обсягу зазвичай означає додавання характеристик і функцій до попередньо схвалених вимог продукту без відповідного збільшення ресурсів, графіку чи бюджету. Потенційні причини різняться залежно від організації, проекту та окремих осіб, але розширення, як правило, відбувається, коли вимоги проекту не визначені та не контролюються належним чином.

Неконтрольоване розширення також може статися, коли групи розробників приступають до роботи до того, як визначать, що справді потрібно бізнесу. Бізнес-вимоги проекту зазвичай розглядаються командами розробників як занадто високорівневі, розпливчасті та загатом непридатні до конкретно розробки. Їм здається, що краще зосередитися на детальних вимогах до продукту. Однак бізнес-вимоги — це реальні вимоги, і вони мають бути достатньо деталізованими, щоб уникнути пізнішого розширення обсягу. Задовольняючи вимоги до продукту, а не бізнес-вимоги, команди не можуть розробити те, що має цінність, і тому надмірно компенсують це додаванням зайвих характеристик.

Уникнення розширення обсягу спрощує планування, допомагає зберегти бюджети незмінними та виконувати проекти за графіком. Найголовніше те, що це допомагає отримати бажаний прибуток від інвестицій.

**Помилка №2: запитувати клієнтів, чого вони хочуть**

Клієнти зазвичай говорять про характеристики, а не про те, що їм справді потрібно. Правда в тому, що люди часто не знають, чого хочуть. І коли клієнти не знають, чого вони хочуть, а розробники не розуміють проблеми, це може призвести до неправильно продуманих рішень і продуктів.

Запитайте клієнтів, чому їм потрібне певне рішення/функція/характеристика. Це може призвести до кращого розуміння їхніх очікувань і плідного обговорення конкретних вимог.

Визначте правильних стейкхолдерів. Цільова аудиторія або кінцевий користувач може не бути особою, яка відповідає за проект або інвестувала в його результат. Для опитування варто підібрати групу, де правильно поєднані користувачі, клієнти, відповідальні особи, які фінансують проект, представників уряду, які встановлюють правила, проектні команди, команди підтримки тощо.

**Помилка №3: нездатність адаптуватися до змін**

Як показують останні зміни у світовій економіці, зовнішній вплив може швидко змінити вимоги до проекту. У будь-якій економіці завжди виникають нові вимоги, які треба враховувати, оскільки змінюються бізнес-пріоритети, приймаються нові державні постанови та визначаються нові зацікавлені сторони. Команди проекту повинні врахувати ці зміни.

Отже, очікуйте та включайте в план ймовірність того, що деякі вимоги змінюватимуться протягом вашого процесу розробки. Змінюйте пріоритетність вимог на основі мінливих обставин, таких як бізнес-потреби, важливість клієнта, передбачувані докладені зусилля та витрати. Інформуйте своїх стейкхолдерів про зміни — вони повинні брати участь у процесі визначення пріоритетів.

**Помилка №4: відсутність ефективної комунікації**

У команд розробників, клієнтів, кінцевих користувачів і відповідальних осіб проекту різна перспектива, як і їхні потреби в комунікації щодо проекту.

Треба знати свою аудиторію та спілкуватися так, щоб допомогти їй зрозуміти певну інформацію. Для цього використовуйте діаграми, історії користувачів та ескізи. Створюйте глосарії, шаблони документів і форми зворотного зв’язку, які є зрозумілими, лаконічними та простими у використанні. Використовуйте прототипи, щоб допомогти зацікавленим сторонам візуалізувати майбутній продукт. Це може або доповнювати текст, або повністю замінити його залежно від необхідного рівня деталізації. Отримайте зворотній зв’язок від усіх ваших представників зацікавлених сторін і пам’ятайте, що одна або дві людини, як правило, висловлюються найбільше. Не робіть помилки, не звертаючи уваги на відгуки інших, і взагалі завжди реагуйте на відгуки.

**Помилка №5: відсутність регулярної комунікації**

Неспроможність спілкуватися із стейкхолдерами на ранній стадії проекту часто призводить до однієї головної проблеми: переробки. Найбільша причина цього полягає в тому, що ми часто думаємо, що знаємо, чого хочуть наші клієнти, настільки добре, що нам не потрібно з ними консультуватися. А різні стейкхолдери зазвичай мають різні пріоритети.

Визначте ключових зацікавлених сторін, включаючи покупців. Виберіть представника з кожної групи, щоб регулярно спілкуватися з ним. Для семінарів (workshop) з великою кількістю стейкхолдерів варто залучити кваліфікованого фасилітатора (координатора).

Встановіть регулярні зустрічі-перевірки (checkpoints) з вашими зацікавленими сторонами, а також заплануйте час для інформування стейкхолдерів у разі виникнення неочікуваних змін в проекті.

**Помилка №6: громіздкі документи та забагато інформації**

Двома поширеними помилками є додавання занадто великої кількості деталей до вимог на занадто ранньому етапі процесу та вимагання більшої відстежуваності (документування) для вимог, ніж це необхідно для ефективного управління життєвим циклом. Виконання більшої кількості роботи, ніж необхідно, і додавання непотрібних деталей до документації коштує як часу, так і грошей.

Краще зосередитися на якості, а не на кількості. Варто додати достатньо деталей до своїх вимог і визначити достатню відстежуваність, щоб виконати роботу — не всю роботу, а поточну частину роботи або ітерацію, яку потрібно завершити. Використовуйте візуальні методи для моделювання вимог до бізнесу та продукту. Діаграми бізнес-процесів і варіантів використання, макети та ескізи можуть допомогти зменшити кількість тексту. Надайте глосарій галузевих термінів, абревіатур і домен-специфічних термінів для полегшення комунікації.

Управління вимогами – це неперервний процес. На пізніших етапах життєвого циклу, коли це буде доцільніше, будуть інші можливості додати більше деталей до вимог і задокументувати їх.

**Помилка №7: приховані артефакти проекту**

Розробники та тестувальники часто навіть не ознайомлені з концепцією проекту або не можуть знайти документацію щодо архітектури чи бізнес-вимог, якщо вони взагалі були створені. Доступність всіх артефактів проекту має вирішальне значення для успіху будь-якого програмного проекту.

Зберігайте всі артефакти проекту в центральному сховищі, доступному для членів команди проекту. Необхідність шукати відповідні документи в багатьох джерелах забирає багато часу та зусиль. Переконайтеся, що документи класифіковані та зберігаються таким чином, щоб в них було легко виконати пошук. Крім того, коли відбуваються зміни в документах, команда має бути проінформована.

**Помилка №8: неоднозначні вимоги**

Неоднозначні вимоги є результатом нечіткої або відсутньої важливої інформації. Це призводить до плутанини та переробки. Конструкторам дуже важко розробляти відповідні моделі, розробникам писати бездефектний код, а тестувальникам розробляти правильні тестові кейси без чітких вимог. Неоднозначність також має тенденцію провокувати перенесення ризиків на наступну фазу проекту. Тоді вимоги потрібно переробити, а це створює проблеми для графіка робіт та вартості проекту. Ця типова помилка особливо шкідлива для проектів з фіксованою вартістю.

Що викликає неоднозначність? Погане формулювання, неточна/неперевірена інформація та припущення, що всі думають однаково і мають на увазі ті ж самі речі. Тобто завжди припускайте, що ви формуєте вимоги для когось, хто не дуже добре розуміється на цій темі. Не думайте, що аудиторія буде знати кожен термін і розуміти кожне поняття. Створіть глосарій і переконайтеся, що всі абревіатури та технічні терміни включені. Доповніть текст візуальними елементами.

**Помилка №9: нездатність правильно оцінювати процеси вимог**

Відсутні та неоднозначні вимоги можуть легко призвести до зриву графіку робіт, перевищення витрат і зниження продуктивності та якості, оскільки кінцеві результати проекту не забезпечують цінності для зацікавлених сторін. Щоб уникнути цього, регулярно оцінюйте статус свого проекту.

Організації повинні мати можливість переглядати, оцінювати та вдосконалювати свій процес формування вимог. Оцінка результатів проектів і процесів дозволяє постійно вдосконалювати процеси протягом усього життєвого циклу програмного забезпечення, що зменшує ймовірність провалу проектів і знижує витрати бізнесу.

Визначайте та збирайте показники, які гарантують успіх проекту. Наприклад, оцінюйте вплив змін на вимоги, покриття тестових випадків, пріоритет, вартість і зусилля бізнесу та вимоги до продукту. Набуваючи більшого досвіду оцінювання, ви знайдете правильну комбінацію показників, яка дозволить вам постійно вдосконалювати процес створення вимог.

**Помилка №10: ізолювання вимог**

Розгляд вимог як ізольованих сутностей, нездатність створити зв’язки між вимогами та іншими артефактами і нездатність розпізнати залежності між вимогами призводить до збільшення ризиків проекту та переробки.

Визначте зв’язки між вимогами, а потім контролюйте їх як одну сутність. Створіть потрібний рівень трасування між вимогами та відповідними документами, що зробить більш продуктивною звітність, необхідну для ефективного управління життєвим циклом.

Коли ви вносите зміни до вимог і змінюєте їх пріоритети, враховуйте вплив цих змін на пов’язані вимоги та відповідні зміни в документації. Використовуйте інструменти, які дозволяють легко візуалізувати визначені вами зв’язки.