**Лекція 1.** **Класифікація та сутність процесів розроблення програмного забезпечення з погляду вирішення проблем якості**

**https://github.com/TetyanaLumpova/QC-testing**

***Базові поняття, види програмного забезпечення***

Технологія програмування – дисципліна, що вивчає технологічні процеси програмування та порядок їх проходження.

Системна інженерія ***(System engineering)*** – розділ науки, що вивчає питання розробки комп'ютерних систем (архітектура, дизайн, інтеграція ПЗ та ін.).

Програмна інженерія ***(Software engineering)*** – дисципліна, спрямована на розробку і супровід ПЗ систем, що функціонують надійно і ефективно, можуть удосконалюватися і еволюціонувати і відповідають вимогам, визначених замовником.

Програмування ***(Programming)*** – процес підготовки завдань для їх вирішення за допомогою комп'ютера; ітераційний процес складання програм.

Програма – дані, призначені для управління конкретними компонентами системи обробки інформації з метою реалізації певного алгоритму, послідовність машинних команд, призначених для досягнення конкретного результату.

Програмне забезпечення ***(***ПЗ ***- Software) –*** комп'ютерні програми, процедури, а також документація і дані, що стосуються функціонування комп'ютерної системи.

Вперше термін *software* ввів відомий статистик Джон Тьюк (John Tukey) в 1958 році для позначення різниці апаратного забезпечення ЕОМ (*hardware*) від засобів обробки даних. Бьярне Страуструп (Bjarne Stroustrup) відзначив, що добре ПЗ неможливо побачити, але можна відчути, коли воно працює з помилками.

За видами виконуваних функцій програмне забезпечення підрозділяється на системне*,* прикладне та інструментальне*.* Такий поділ є умовним, оскільки широке впровадження комп'ютеризації призвело до того, що майже кожна програма має ознаки декількох видів ПЗ.

Інтегроване середовище розробки програмного забезпечення ***(Integrated development environment, IDE)*** - це система програмних засобів, що використовується програмістами для розробки програмного забезпечення. Як правило, середовище розробки включає текстовий редактор, компілятор і/або інтерпретатор, засоби автоматизації збирання, налагодження та різноманітні інструменти для конструювання графічного інтерфейсу користувача. Значного поширення об'єктно-орієнтованого програмування (ООП) призвело до того, що сучасні інструменти розробки включають браузер класів та інспектор об'єктів. На сьогодні в середовища розробки підключають систему керування версіями, засоби тестування та ін.

***Види програмного забезпечення***

Системне ПЗ ***(System software)*** призначене для управління роботою комп'ютера, розподілу його ресурсів, підтримки діалогу з користувачами, а також для часткової автоматизації розроблення нових програм. Як правило, системні програми забезпечують взаємодію інших програм з апаратними складовими, організацію інтерфейсу користувача. Виділяють три типи системного ПЗ:

− операційна система *(*ОС*)* – програмне забезпечення, що забезпечує інфраструктуру, на якій можуть працювати прикладні програми. Найпоширеніші ОС: Microsoft Windows, Mac OS та Linux, а для мобільних пристроїв Android, iOS, Windows Phone.

− системи програмування – призначені для полегшення та часткової автоматизації процесу розроблення та відлагодження програм;

− сервісні програми *(*утиліти*) –* розширюють можливості ОС. До утиліт відносять архіватори, антивіруси, драйвери та ін.

Прикладне ПЗ ***(application, application software)*** – комп'ютерна програма, що вирішує конкретні задачі фахової діяльності користувача.

Інструментальне ПЗ призначене для розроблення всіх видів інформаційно-програмного забезпечення. При цьому під інформаційним

забезпеченням розуміють сукупність попередньо підготовлених даних, що необхідні для роботи програмного забезпечення. До інструментального ПЗ відносять текстові редактори, системи управління базами даних, транслятори мов програмування.

***Програмне забезпечення як виріб***

Сучасні програми вирішують найрізноманітніші завдання за змістом і галузевим значенням. Часто програми створюються в єдиному екземплярі для вирішення приватних дослідницьких завдань, для обробки експериментального матеріалу, прискорення обчислень, моделювання процесів тощо. Такі програми не мають масового застосування та доступні для використання тільки тим, хто їх розробив. Вони стають об'єктами науково-технічної творчості і рідко стають промисловими виробами.

Зовсім іншим класом програм є індустріальні програмні засоби*,* які можна кваліфікувати як продукцію виробничо-технічного призначення. Вони являють собою програми на носіях даних з технічною (експлуатаційною та технологічною) документацією, розроблені відповідно до діючих стандартів і пройшли державні, міжвідомчі або відомчі випробування.

Програмні засоби, прийняті у виробництво, виготовляються за затвердженою в установленому порядку технологією. Вони повинні відповідати затвердженим технічним умовам і діючій нормативно-технічній документації, забезпечуватися гарантіями постачальника, у тому числі щодо забезпечення якості самого програмного засобу та реалізованого процесу за визначеним в технічному завданні алгоритмі.

Під програмним виробом (ПВ) розуміється універсальне програмне забезпечення, яке призначається для широкого кола користувачів, можливо, навіть не відомих заздалегідь, і повинно рекламуватися, підтримуватися в працездатному стані, розширюватися протягом тривалого періоду часу.

Програмний виріб – це власне програми плюс документація, гарантія якості, рекламні матеріали, навчання, поширення та супровід. Окрема програма або сукупність програм і програмний виріб – не тотожні поняття. Програмний виріб є продукт ретельного планування і цілеспрямованої розробки, що супроводжується чіткою документацією, який пройшов всі необхідні випробування; який описано у відповідних технічних документах, розмножено у необхідній кількості примірників, обслуговується і контролюється постачальником за заздалегідь продуманим планом і може розглядатися як товар.

З питань розробки систем програмного забезпечення (але не програмних виробів) можна визначити:

− розробник створює програмне забезпечення для себе або для організаційно пов'язаний з користувачами програмного забезпеченні, що розробляється;

− користувач формулює свої вимоги безпосередньо розробнику, якщо останній сам не є одночасно користувачем;

− користувач бере активну участь у розробці або в обслуговуванні програмного забезпечення;

− програмне забезпечення повинно працювати тільки на певній конфігурації комплексу технічних і програмних засобів в обмеженому діапазоні змін його складу і структури даних;

− розробник сам вводить в дію програмне забезпечення у користувача;

− проблеми, що виникли при використанні програмного забезпечення, вирішуються користувачем спільно з розробником або з персоналом, що здійснює його технічне обслуговування (супровід);

− програми не мають масового застосування та доступні для використання тільки тим, хто їх розробив;

− використання програми припиняється після отримання результату.

При розробці програмного виробу (за винятком особливого випадку розробки програмного забезпечення за контрактом для єдиного користувача) зазвичай складаються ситуації:

− розробник не знайомий з користувачем;

− вимоги користувача формуються або розробником, або передаються йому посередницькою організацією (наприклад, поставляє програмне забезпечення);

− користувачі не беруть участь у розгляді та узгодженні проектних рішень, якщо не вважати рідкісних випадків, коли їх інтереси представлені посередниками;

− програмне забезпечення повинно зберігати працездатність в широкому діапазоні конфігурацій обчислювальних комплексів і при самих різних системних програмних засобах;

− користувачі вводять програмне забезпечення в дію або самі, або з сторонньою допомогою, але ця допомога виходить не від розробника;

− проблеми, що виникли при використанні програмного забезпечення, вирішуються шляхом письмової фіксації / листування, а іноді через посередника;

− програмний виріб призначений для широкого кола користувачів, можливо, невідомих;

− програмне забезпечення використовується багаторазово і тривалий час.

***Технологія розробки програмного забезпечення***

Технології – сукупність знань про способи і засоби проведення виробничих процесів. В одному крайньому випадку одна людина здійснює поетапну розробку програми зі свого комп‘ютера в невимушеній обстановці. Природно, він створює порівняно невеликі програми, що не вимагають особливої оцінки. В іншому звичайному випадку розробляється дуже складне програмне забезпечення, що призначене для функціонування в реальному масштабі часу і вимагає трудовитрат обсягами в тисячі людино-годин.

Ці дві взаємно протилежні ситуації характеризуються різним ступенем формалізації та проведенні процесу розробки програмних засобів. Ступінь формалізації та проведення процесу розробки програмного забезпечення безпосередньо залежить від цілей його створення, його величини, чисельності групи розробників та інших факторів. Від того, наскільки правильно і вдало з погляду технології розробки програмного забезпечення побудовано додаток, залежить якість і життєздатність кінцевого продукту.

Під технологією розробки програмного забезпечення (ТРПЗ) розуміється сукупність узагальнених і систематизованих знань, або наука про оптимальні способи (прийоми) проведення процесу розробки програмного забезпечення, що забезпечує в заданих умовах отримання програмної продукції із заданими властивостями. Технологія розробки програмного забезпечення являє собою інженерний підхід до розробки програмних засобів, що охоплює методологію програмування, проблеми забезпечення надійності програм, оцінки робочих характеристик і якості проектів.

Технологія розробки програмного забезпечення розглядає питання управління проектуванням систем програмного забезпечення, а також засоби і стандарти розробки програм. Технологія розробки програмного забезпечення визначає деяку професійну культуру роботи фахівців (не тільки програмістів), що забезпечує заданий рівень продуктивності праці і якості одержуваної в результаті програмної продукції.

Технологія розробки програмного забезпечення охоплює процес розробки програмного забезпечення від появи потреби в ньому до його виготовлення, передачі користувачеві, модифікації в процесі експлуатації і припинення його використання внаслідок морального старіння.

В ідеалі технологія розробки програмного забезпечення повинна задовольняти основним нижче перерахованим вимогам, виконання яких є запорукою забезпечення якості кінцевого продукту*.*

1. Необхідна стандартизація мов проектування програм, оформлення та випробування програмних модулів, а також гарантії їх якості. Це дозволить значно скоротити дублюючі розробки, впровадити компонентне програмування і вести накопичення на підприємствах і в країні високоякісного програмного продукту для його багаторазового використання як типових комплектуючих виробів.

2. Вести постійний контроль і забезпечення якості програм.

3. Програми не повинні містити неперевірених шляхів і ситуацій функціонування, які призводять до несподіваних результатів.

4. Користувачеві або покупцю програм необхідно дати чітке уявлення про можливості даної програми і технологічні умови експлуатації, при яких гарантуються певні функції і якості.

5. Технологія розробки програмного забезпечення повинна забезпечувати відторгнення програмного виробу від його розробника, тобто людський фактор у програмуванні має бути зведений до мінімуму.

6. Технологія розробки програмного забезпечення та засоби її підтримки (автоматизації) повинні забезпечувати цілеспрямовану роботу, насамперед колективу програмістів, а не окремих особистостей. Вона повинна спонукати колектив працювати тільки правильно і злагоджено; повинна автоматично блокувати будь-які не санкціоновані технологією дії.

7. Необхідно вести акуратне документування всіх етапів розробки. Документація повинна також заноситися і зберігатися на магнітних носіях. Доступ до цієї інформації має бути відкритим, простим і автоматизованим.

8. Робота користувача повинна забезпечуватися розвиненою інформаційно-довідковою системою.

9. Засоби автоматизації технології повинні охоплювати всі етапи роботи колективу програмістів.

10.Технологія розробки програмного забезпечення повинна бути простою в освоєнні, із засобами підказки, що автоматично включаються.

11.Технологія розробки програмного забезпечення повинна мати засоби автоматичної фіксації в хронологічному порядку всіх дій, які виконуються в процесі колективного виготовлення програмного виробу - повинні вестися і зберігатися в системі журнали (протоколи, щоденники) розробки. Ці засоби повинні дозволяти відновлювати будь-який стан процесу розробки на будь-якому інтервалі виготовлення програмного продукту.

***Проблеми розробки складних програмних систем***

Більшість сучасних програмних систем об'єктивно дуже складні. Ця складність обумовлюється багатьма причинами, головною з яких є логічна складність розв*'*язуваних ними завдань*.* У наш час, коли створено потужні комп'ютерні мережі, з'явилася можливість перекласти на них рішення складних ресурсномістких завдань, про комп'ютеризацію яких раніше ніхто і не думав. Зараз до процесу комп'ютеризації залучаються зовсім нові предметні області, а для вже освоєних областей ускладнюються вже сформовані постановки завдань.

*Додатковими факторами, що збільшують складність розробки програмних систем*, є:

− складність формального визначення вимог до програмних систем;

− відсутність задовільних засобів опису поведінки дискретних систем із великою кількістю станів при недетермінованій послідовності вхідних впливів;

− колективна розробка;

− необхідність збільшення ступеня повторюваності кодів.

*Складність визначення вимог до програмних систем.*

Складність визначення вимог до програмних систем обумовлюється двома факторами. По-перше, при визначенні вимог необхідно врахувати велику кількість різних факторів. По-друге, розробники програмних систем не є фахівцями в предметних областях, що автоматизуються, а фахівці в предметній області, як правило, не можуть сформулювати проблему в потрібному ракурсі. Більш детально проблематику визначення вимог з погляду забезпечення якості будуть розглянуті в подальших лекціях.

*Колективна розробка****.***

Завдяки великим обсягам проектів розробка програмного забезпечення здійснюється колективом фахівців. Працюючи в колективі, окремі фахівці повинні взаємодіяти один з одним, забезпечуючи цілісність проекту, причому, чим більший колектив розробників, тим складніше організувати процес роботи.

*Необхідність збільшення ступеня повторюваності кодів*.

На складність програмного продукту, що розробляється, впливає і те, що для збільшення продуктивності праці компанії прагнуть до створення бібліотек компонентів, які можна було б використовувати в подальших розробках. Однак у цьому випадку компоненти приходиться робити більш універсальними, що зрештою збільшує складність розробки.

Разом узяті, ці фактори суттєво збільшують складність процесу розробки. Однак очевидно, що всі вони безпосередньо пов'язані зі складністю об'єкта розробки - програмної системи.

Як показують дослідження **[**1, 2**]**, якість програмних продуктів залишається не надто високою. Основними негативними факторами впливу на якість кінцевого продукту є слабка формалізація вимог якості, відсутність методів та процедур їх комунікації на стадіях життєвого циклу, а також термінологічні розбіжності між замовниками та розробниками ПЗ. Для нівелювання негативного впливу, наведених вище факторів на якість кінцевого програмного продукту, необхідно вирішити низку проблем, пов'язаних з інтерпретацією вимог до ПЗ та інтеграцією процесу управління вимогами на стадіях життєвого циклу.

Рівні управління вимогами.

Для отримання якісного програмного продукту, що відповідатиме запитам замовників, дуже важливе значення має етап збирання вимог. До цієї роботи повинні бути залучені всі зацікавлені в проекті особи, а саме:

- замовники, які фінансують проект;

- користувачі, які безпосередньо працюють з додатком (підклас замовників);

- аналітики вимог, які пишуть вимоги і передають їх розробникам ;

- розробники, які створюють, розгортають і підтримують продукт;

- тестувальники, які визначають відповідність поведінки ПЗ бажаному;

- технічні письменники, які відповідають за створення керівництва користувачів, тренувальних матеріалів та довідкової системи;

- менеджер по проекту, який планує процес і керує командою розробників аж до успішного випуску продукту ;

- співробітники правового відділу, які стежать, щоб продукт не суперечив чинним законам і постановам;

- співробітники відділу продажів і маркетингу, виїзної служби підтримки та всі ті, кому доведеться працювати з продуктом і його користувачами.

Наведемо основні визначення якими користуються в процесі аналізу вимог при розробці програмних систем (ПС).

Вимоги до ПЗ( software requirement):

1. це умови або можливості, необхідні користувачам для вирішення проблем або досягнення цілей;
2. це умови або можливості, якими повинна володіти ПС або системні компоненти, щоб виконати контракт або задовільнити стандартам, специфікаціям або іншим формальним документам;
3. задокументоване представлення умов або можливостей для пунктів 1-2.

Вимоги до ПЗ – це специфікація того що повинно бути реалізовано. Це описи поведінки системи, властивості системи або її атрибути. Вони можуть бути обмеженні процесом розробки системи.

Вимоги до ПЗ складаються з 3-х рівнів:

бізнес-вимоги;

вимоги користувачів;

функціональні вимоги.

При цьому кожна система характеризується нефункціональними вимогами. Модель на рис. 1 ілюструє спосіб представлення вимог, схематично показує організацію вимог. Овал – позначає тип інформації для вимог, прямокутник спосіб зберігання інформації (документи, діаграми, бази даних) [1].

*Функціональні вимоги* *Нефункціональні вимоги*

Документ про образ та межи проекту

Документ про варіанти використання

Специфікація вимог до ПЗ

Рисунок 1 - Класифікація та зв’язок вимог

*Бізнес-вимоги* (business requirements) містять високорівневі цілі організації або замовників ПЗ. Як правило, їх висловлюють, ті хто фінансує проект, покупці системи, менеджери користувачів. В цьому документі роз‘яснено для чого необхідна така система, тобто описано цілі які організація має намір досягнути. Часто бізнес вимоги записують в формі документа про образ і межі проекту, який називають уставом проекту (project charter) або документом ринкових вимог (market requirement document). Визначення меж проекту є першим етапом управління загальними проблемами «розповзання» меж.

*Вимоги користувачів* (user requirements) описують цілі і задачі, які дозволить користувачу вирішувати дана система. До способів представлення таких вимог відносяться варіанти використання, сценарії і таблиці «подія-відгук». В цьому документі вказано, що клієнти можуть робити з допомогою системи.

*Функціональні вимоги* (functional requirements) визначають функціональність ПЗ, яку розробники повинні забезпечити, щоб користувачі змогли виконати свої завдання в межах бізнес-вимог.

Інколи їх називають вимоги поведінки (behavioral requirements), вони містять положення з традиційним «повинна». Наприклад, «Система повинна по електронній пошті відправляти користувачу підтвердження замовлення».

Функціональні вимоги описують, що розробнику необхідно реалізувати.

Терміном *системні вимоги* (system requirements) позначають високорівневі вимоги до ПС, які містять багато підсистем.

*Бізнес-правила* (business rules) включають корпоративну політику, постанови управління, промислові стандарти і чисельні алгоритми. Бізнес-правила не є вимогами до ПЗ, тому що вони знаходяться зовні меж будь-якої системи ПЗ. Однак вони часто накладають обмеження, визначаючи хто може виконувати конкретні варіанти використання, або якими функціями повинна володіти система, яка підлягає відповідним правилам. Інколи бізнес-правила є джерелом атрибутів якості, які реалізуються в функціональності.

Функціональні вимоги документуються в специфікації вимог до ПЗ (software requirement specification, SRS), де описується очікувана поведінка системи.

Це може бути документ, база даних, таблиця з вимогами, сховище даних в комерційному інструменті управління вимогами, або навіть набір карток для невеликого проекту. Специфікація вимог до ПЗ використовується при розробці, тестуванні, гарантії якості продукту, управлінні проектом і зв’язаних з проектом функціях. Окрім функціональних в специфікації містяться нефунціональні вимоги, де описані цілі і атрибути якості.

*Атрибути якості* (quality attributes) є додатковим описом функцій продукту, опис характеристик важливих для користувачів або розробників. До таких характеристик відносяться легкість і простота використання, простота переміщення, цілісність, ефективність і стійкість до збоїв. Інші не функціональні вимоги описують зовнішні взаємодії системи і середовища, а також обмеження дизайну і реалізації.

*Обмеження* (constraints) стосуються вибору можливості розробки зовнішнього вигляду і структури продукту.

*Характеристика* (feature) – це набір логічно зв’язаних функціональних вимог, які забезпечують можливості для користувача і відповідають бізнес-цілям. В області комерційного ПЗ характеристика є групою вимог, яку вирізняють всі зацікавленні особи, які важливі при прийняті рішення про покупку. Характеристики продукту, які перераховує клієнт, не еквівалентні тим які необхідні для вирішення задач користувача.

Хоча в моделі на рис.1 потік вимог показано в напрямку зверху вниз, проте слід очікувати і циклів, і ітерацій між бізнес-вимогами, вимогами користувачів та функціональними вимогами.

Рисунок 2 - Піраміда вимог за Леффінгуелом [3]

На етапі встановлення вимог здійснюється виявлення вимог і їх визначення, переважно, у вигляді формулювань природною мовою. Формальне моделювання вимог з використанням мови UML проводиться пізніше на етапі специфікації вимог. Проте, під час встановлення вимог постійно ведеться діяльність по узагальненому візуальному уявленню зібраних вимог, яку називаємо бізнес-моделюванням вимог.

Для встановлення рамок системи потрібно, щонайменше, високорівнева візуальна модель, що дозволяє позначити ключові прецеденти і ввести найбільш істотні бізнес-класи. На рис. 3 показані залежності між цими трьома моделями етапу встановлення вимог і моделі інших етапів ЖЦ розробки.

Провідна роль діаграм прецедентів в ЖЦ розробки знайшла своє відображення на рис. 3, з якого ясно, що джерелом тестових прецедентів, призначеної для користувача документації і проектних планів виступають моделі прецедентів. Більш того, діаграми прецедентів і моделі класів використовуються паралельно і по черзі грають роль "лідера гонки" в рамках послідовних ітерацій розробки. Проектування і реалізація також тісно переплетені і можуть ініціювати зворотний зв'язок з моделями специфікації вимог.

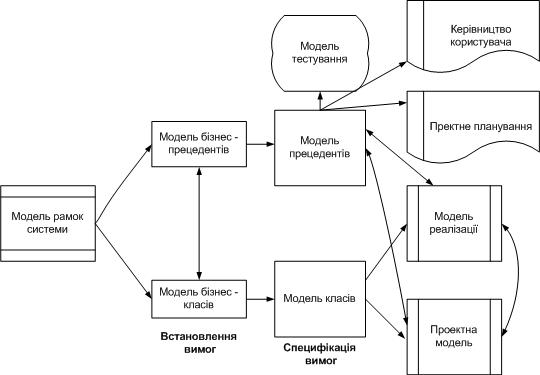


Рисунок 3 - Використання бізнес моделей на різних етапах вироблення вимог

Характеристики якісних вимог

- Повнота. Вимога повинна повністю описувати функціональність.

- Коректність. Вимога повинна точно описувати функціональність.

- Реалізовуваність. Вимога повинна бути реалізовуваною.

- Необхідність. Вимога повинна відображати те, що потрібно користувачу.

- Пріоритетність. Вимоги повинні бути розставленими за пріоритетами.

- Недвозначність. Вимога повинна однозначно інтерпретуватись.

- Тестованість. Вимога повинна легко перевірятись.

*Для самостійного вивчення* *(2 години)*: Вивчення лекційного матеріалу та додаткових джерел. Розгляд запитань і виконання завдань для самостійної роботи, запропонованих на лекції.

*Література*.

1. The Standish group report. CHAOS Manifesto 2013, Think Big, Act Small. [Electronic resource].– Mode of access http://www.versionone.com/assets/img/files/ChaosManifesto2013.pdf.

2. The Standish group report. The CHAOS Manifesto 2012: The Year of the Executive Sponsor. [Electronic resource]. – Mode of access [http://www.versionone.com/assets/img/files / CHAOSManifesto2012.pdf](http://www.versionone.com/assets/img/files%20/%20CHAOSManifesto2012.pdf).

3. Леффингуэлл Дин, Уидриг Дон. Принципы работы с требованиями к программному обеспечению. Унифицированный подход. М.: Вильямс, 2002. — 448 с.

4. Вигерс Карл Разработка требований к программному обеспечению.

Пер, с англ. - М.:Издательско-торговый дом «Русская Редакция», 2004. - 576с.

*Запитання для самоперевірки.*

1. В чому полягає важливість процесу збирання вимог ?
2. Надайте визначення вимоги та конкретизуйте його стосовно ПЗ.
3. Визначте класифікацію вимог.
4. Яким чином визначаються зацікавлені в програмному продукті особи на стадії збирання вимог?
5. Визначте джерела вимог.
6. Які Ви знаєте методи виявлення вимог
7. Які перешкоди виникають при виявленні вимог?
8. Для чого встановлюють пріоритети вимог?