# Управління якістю програмного виробу

## Якість програмного виробу

Усі розглянуті підходи до створення програмної продукції мають на основній меті створення високоякісних програмних виробів. Тут під якістю програмного виробу розуміють його відповідність чітко і явно встановленим вимогам до функціональних характеристик, відповідність стандартам, що визначають документування і проведення розробки, а також іншим характеристикам, які передбачається отримати від розробленого виробу. В приведеному визначенні істотними є наступні три моменти:

* 1. Вимоги до програмного виробу – це основа, на якій оцінюється і вимірюється якість. Порушення або втрата яких-небудь вимог приводить до втрати якості виробу.
  2. Стандарти визначають сукупність вимог і критеріїв розробки, що вказують яким чином повинен розроблятися виріб. Порушення цих вимог також приводить до зниження якості.
  3. Є також множина інших (іноді і неявних) вимог, на які часто не звертають уваги, і в результаті, якщо вони були упущені, якість може бути знижена.

Якість в загальному випадку характеризується широким набором властивостей програмного виробу, який залежить від прикладної галузі і від категорії користувачів.

Чинники, що впливають на якість виробу, класифікуються по трьох аспектах програмного продукту:

* Робочі характеристики;
* Пристосованість до внесення змін;
* Пристосованість до зміни навколишнього оточення.

До першої групи можна віднести наступні характеристики:

* Правильність (коректність), які характеризують міру функціональної відповідності програмного виробу вимогам користувача.
* Надійність визначає вірогідність того, що програмний виріб працюватиме без збоїв протягом певного інтервалу часу або при виконанні певного обсягу робіт, тобто міру впевненості в тому, що програма виконуватиме призначені функції з необхідною точністю.
* Ефективність характеризує оперативність виконання функціональних задач, а також обсяги комп’ютерних ресурсів і об’єм коду програми, необхідний для виконання останньою своїх функцій.
* Зручність і простота використання програмного виробу відображає простоту навчання роботі з виробом, простоту підготовки вихідних даних і інтерпретації вихідних повідомлень програми. Часто зручність і простота визначають, наскільки „дружнім” є інтерфейс користувача з комп’ютером. Ця властивість програмного виробу характеризує:
* Фізичні і інтелектуальні зусилля, необхідні для навчання роботі з даним програмним продуктом;
* Час, необхідний для того, щоб використовування системи стало ефективним;
* Підвищення продуктивності праці при експлуатації програмного виробу користувачем середньої кваліфікації;
* Суб’єктивну оцінку відношення користувача до системи.
* Цілісність визначає міру, з якою може бути проконтрольований несанкціонований доступ до даних і програм.

Перераховані характеристики можуть бути доповнені і деталізовані. Наприклад, надійність програмного виробу оцінюється такими додатковими характеристиками, як можливість відновлень системи після збоїв, готовність до використання, середній час між відмовами і т.п.

Друга група характеристик пов’язана з супроводом програмного виробу. В ній можна виділити наступні основні показники:

* Зручність і простота супроводу в першу чергу оцінюються тими зусиллями, які необхідні для виявлення і виправлення помилок в програмному виробі. Зручність і простота супроводу передбачати також модифікованість, яка означає простоту внесення змін у виріб при зміні вимог користувача. Цю властивість називають також гнучкістю програмного виробу.
* Зручність і простота тестування відображає зусилля, необхідні для тестування програми, з метою гарантії того, що програма виконує потрібні функції.

Третя група характеристик – можливості використання програмного виробу в нових навколишніх умовах, включає:

* Переносимість (мобільність) характеризує зусилля, необхідні для перенесення програми з одного технічного виробу на інший або перенесення її в інше операційне середовище.
* Придатність до повторного використання показує, наскільки програмний виріб або його частина можуть бути використані в інших застосуваннях.
* Сумісність визначає можливість взаємодії з іншими програмними продуктами.

Одним з основних чинників, що впливають на якість програмного виробу і на більшість перерахованих характеристик, є складність програмного виробу. Складність – головна причина появи помилок і дефектів в програмному виробі і його ненадійності. В першу чергу складність програмного виробу обумовлюється складністю вирішуваної проблеми і залежить від складності окремих компонент програмного виробу і зв’язків між ними. Для підвищення якості програмного виробу прагнуть зниження складності програмного виробу.

## Забезпечення якості програмного забезпечення

Проблема забезпечення якості програмних виробів останніми роками набула особливої значущості у зв’язку з тим, що програмне забезпечення набуло статусу комерційного продукту. Показники якості тепер стали включати в контракти на розробку програмних виробів різного призначення. Забезпечення якості програмного забезпечення – це планована систематична сукупність дій, потрібних для створення якісної продукції, яка охоплює різноманітні задачі, вирішення яких включає наступні види діяльності:

* Використання спеціальних методів і засобів, що дозволяють аналітику досягти високоякісних специфікацій, а проектувальнику створити високоякісний проект.
* Проведення формальних технічних оглядів, які служать основним інструментом для оцінки якості виконаних робіт.
* Тестування програмного виробу, яке об’єднує стратегію проведення тестових випробувань з методами проектування тестових наборів даних, що дозволяють ефективно знаходити помилки в програмному виробі. При цьому слід мати на увазі, що тестування не дозволяє однаково ефективно знаходити всі типи помилок.
* Примусове впровадження стандартів і формальних процедур, що визначають розробку програмного виробу, і контроль за їх використання.
* Контроль всіх змін, що вносяться в програмний продукт, що є об’єктом конфігураційного управління. Кожна зміна, що вноситься, є потенційним джерелом помилок і побічних ефектів; контроль за змінами необхідний як під час розробки, так і в процесі супроводу.
* Вимірювання характеристик якості, необхідні для постійного контролю рівня якості, а також для оцінки впливу методологічних і процедурних змін на покращення якості.
* Реєстрація і документування всіх процедур, пов’язаних із забезпеченням якості виробів, призначена для подальшого використання і тривалого зберігання.

Формальні технічні огляди – найважливіший вид діяльності по забезпеченню якості програмного виробу. Основні цілі цих оглядів:

* виявлення помилок у функціях, логіці або способах реалізації програмного виробу;
* підтвердження того, що програмний виріб відповідає поставленим вимогам;
* забезпечення того, що програмний виріб був представлений відповідно до встановлених стандартів;
* встановлення одноманітності розробки окремих частин виробу;
* спрощення процесу управління розробкою програмного виробу.

Формальні огляди служать основою для знайомства з програмним виробом широким колом фахівців, засобом для вивчення різних підходів до аналізу, проектування і реалізації виробу. Формальні огляди повинні проводитися під час кожного кроку розробки виробу, і в його проведенні, як правило, беруть участь декількох чоловік. Для кожного огляду заздалегідь складається список питань, що підлягають обговоренню.

Для перевірки програмного виробу можуть використовуватися в загальному випадку три види оглядів:

* технічний огляд;
* наскрізний перегляд;
* уважне вивчення представлених матеріалів.

Всі огляди формальні, оскільки мають чіткі цілі і строго певні процедури; всі призначені для ідентифікації дефектів програмного забезпечення і відхилень від встановлених специфікацій, планів і стандартів.

## Вимірювання якості програмного забезпечення

Для визначення якості програмного виробу, а також рівня якості розробок програмного забезпечення в цілому по організації використовується статистичний підхід, який передбачає:

* Збір і класифікацію інформації про дефекти програмних виробів.
* Виявлення причин кожного дефекту шляхом докладного аналізу дефектів і їх трасування.
* Виявлення найістотніших причин дефектів в програмній продукції на основі використання принципу Парето (80% усіх дефектів обумовлено 20% можливих причин).
* Усунення найістотніших причин, що викликають дефекти в програмному виробі.

Ретельна систематизація статистичної інформації дозволяє виявити причини появи помилок. Це може бути:

* неповнота або помилковість специфікацій;
* неправильна інтерпретація інформації замовника;
* умисне відхилення від специфікацій;
* порушення стандартів програмування;
* помилки в представленні даних;
* несумісні інтерфейси між модулями;
* помилки в логіці проектних рішень;
* неповне або помилкове тестування;
* неточна або неповна документація;
* неоднозначність і суперечність людино-машинного інтерфейсу.

Усі знайдені в процесі розробки і подальшій експлуатації програмного забезпечення помилки розподіляються по групах, в кожній з яких виділяються серйозні, середні і незначні помилки. На основі цієї інформації виділяються групи найчастіших і серйозних помилок, а потім розглядаються способи усунення основних причин їх появи і методи вдосконалення процесу розробки.

Зібрана статистична інформація може бути використана згодом як база для оцінки міри підвищення або зниження якості виробів, що підлягають розробці в подальші роки, і для яких також збиратимуться відомості про дефекти.

Для кількісної оцінки якості програмної продукції було розроблено ряд підходів, що знайшли потім віддзеркалення в стандартах. Так, згідно Американському стандарту для визначення індексу якості структури проекту на основі сукупності придатних для вимірювання характеристик програмного проекту необхідно знати:

* Загальна кількість модулів в архітектурному проекті;
* Кількість модулів, чия правильне функціонування залежить від джерела вхідних даних або які породжують дані, що використовуються в інших модулях;
* Кількість елементів бази даних, включаючи об’єкти і їх атрибути;
* Загальну кількість унікальних елементів бази даних;
* Кількість різних типів записів в базі даних;
* Кількість модулів з єдиним входом і виходом.

На основі перерахованих даних проводиться просте визначення таких проміжних величин, як незалежність модулів між собою і від попередньої обробки, розмір бази даних і її розділеність на частини і показник вхід/вихід модуля. Сума названих проміжних величин з відповідними ваговими коефіцієнтами, що відображають відносну вагу кожного з показників, дає значення індексу для даного проекту. Це значення може бути зіставлено з індексами минулих проектів для того, щоб зробити висновок про рівень якості проекту.

Інший стандарт – індекс завершеності програмного виробу – характеризує стабільність програмної продукції при розробці нових версій виробу і пов’язаний зі змінами, що вносяться в процесі вдосконалення виробу.

Індекс завершеності визначається за формулою:



де Ml – кількість модулів в поточній версії, Fc – кількість змінених модулів в поточній версії, Fa – кількість доданих модулів в поточній версії, Fd – кількість модулів попередньої версії, виключених з поточної версії.

Вважається, що продукт починає стабілізуватися, коли SMI наближається до 1. Цей показник використовується також і для планування діяльності по супроводу. Середній час розробки нової версії програмного виробу може бути пов’язаний з індексом завершеності за допомогою простої емпіричної формули.

## Управління конфігурацією програмного виробу

Управління конфігурацією програмного забезпечення, або конфігураційне управління, розглядається як діяльність, яка виконується на протязі всіх етапів ЖЦПЗ і здійснює ідентифікацію, опис (документування) і контроль всіх елементів програмного виробу і всіх модифікацій, що появлятися в процесі його експлуатації.

Управління конфігурацією – одночасно і технічна, і управлінська діяльність, направлена на підтримку високої якості виробу і ретельний контроль всіх вироблюваних змін. Конфігураційне управління грає особливу роль в організації супроводу програмної продукції.

Зміни відбуваються як під час розробки програмної продукції, так і під час її експлуатації. Якщо ці зміни не аналізувати всебічно до їх реалізації, виникне плутанина, для мінімізації якій і було призначено конфігураційне управління.

Конфігурація програмного виробу утворюється з сукупності взаємозв’язаних об’єктів, які називають елементами конфігурації. До елементів конфігурації відносяться:

* Комп’ютерні програми (у формі вихідного або виконуваного коду);
* Документи, які описують програми (з орієнтацією або на користувача, або на обслуговуючий персонал);
* Структури даних, що містяться або в програмах, або зовні них.

Можливий набір основних елементів, що включаються в сферу конфігураційного управління і визначаючих конфігурацію програмного виробу, охоплює:

* Вимоги користувача.
* Специфікацію вимог до програмного виробу.
* Технічне завдання на розробку програмного виробу.
* План проектування програмного виробу.
* Попереднє керівництво користувача.
* Специфікацію проекту (опис архітектурного проекту, проекту даних, проектів окремих модулів).
* Документовані лістингу вихідних кодів.
* Матеріали тестування: (план і процедура тестування, тестові набори і результати тестування).
* Операційне керівництво і керівництво по введенню в дію.
* Робочі програми (виконувані модулі і модулі зв’язків).
* Опис бази даних: (схема бази і структури файлів, первинне наповнення).
* Документи по супроводу програмного виробу.
* Стандарти і процедури, що визначають порядок виконання робіт.

Під конфігураційний контроль поміщають програмні засоби, що також додатково використовуються: конкретні версії редакторів, компіляторів, CASE-засобу і т.п. Оскільки ці засоби використовуються для створення різної документації і вихідного кодів, тому вони повинні бути доступними, коли з’являється необхідність внесення змін. На практиці конфігураційні елементи каталогізуються в базі даних проекту.

У міру просування розробки програмного виробу росте число елементів конфігурації, оскільки кожний елемент у свою чергу породжує інші елементи.

При створенні виробу зміни відбуваються в процесі роботи постійно, у міру накопичення розробниками досвіду і інформації.

Управління конфігурацією програмного виробу, таким чином, дисципліна використання адміністративного і технічного керівництва і нагляду з метою:

* Ідентифікації і документування фізичних і функціональних характеристик елементів конфігурації;
* Контролю за зміною цих характеристик;
* Реєстрації, обробки і документування змін і встановлення статусу реалізації;
* Перевірки узгодженості проведених дій зі встановленими вимогами.

Особливу трудність для супроводу представляють програмні вироби складної структури, що мають численних користувачів. Для оптимальної організації робіт по внесенню змін застосовується конфігураційне управління, що базується на:

* Ревізійному контролі;
* Ідентифікаційному контролі;
* Управлінні розповсюдженням.

Ревізійний контроль пов’язаний з чіткою класифікацією всіх компонентів програмного виробу і з контролем за їх змінами. Класифікація використовує для програмного виробу такі поняття, як виріб, варіант, версія, редакція.

Вироби використовуються для отримання можливостей, що різняться між собою, варіанти допускають різні умови роботи, версії забезпечують різні властивості, а редакції забезпечують різну міру коректності. Таким чином, версії, відрізняючись деякими властивостями, зберігають всі основні функції виробу і сумісні з попередніми версіями цього виробу. Нова редакція частіше за все пов'язана з усуненням знайдених помилок або покращенням експлуатаційних характеристик. По кожній зміні, що вноситься, необхідно приймати рішення про те, до якого результату приведе дана зміна, для чого заздалегідь проводиться оцінка того, які зміни і в яких компонентах повинні бути проведені. Прийняття рішення базується на заздалегідь складених матрицях впливів і таблицях прийняття рішень, що дозволяють достатньо надійно визначати міру впливу можливих змін, що вносяться у виріб. Зміни, пов'язані із створенням нового виробу або варіанту, вимагають нового проектування. Нові вироби і варіанти не замінюють старі; нові версії і редакції використовуються замість колишніх.

Друга функція ревізійного контролю – контроль за змінами. Кожна зміна породжує послідовність змін, які повинні бути внесені в інші компоненти і документацію. З цією метою складаються матриці впливу, де вказується, які компоненти виробу вимагають перегляду при створенні нової редакції, версії і т.д. Так для створення нового виробу потрібна розробка нових документів і нового керівництва, а для нової версії – лише перегляд деяких документів.

В складних виробах, побудованих за модульним принципом, звичайно створюється база даних програмного виробу, в якій кожний компонент виробу повинен мати унікальний ідентифікатор. Ідентифікаційний контроль служить для присвоєння всім компонентам ідентифікаційних кодів. Оскільки система компонентів утворює ієрархічну багаторівневу систему, ідентифікація будується на основі послідовного коду. Окрім ідентифікаційного коду, кожний компонент забезпечується набором кодів, що характеризують номер варіанту, версії, редакції.

Третя складова конфігураційного управління – управління розповсюдженням, яке гарантує виборчий розподіл матеріалів – повідомлень про внесені зміни користувачів. При цьому ведеться облік користувачів, списки розсилки матеріалів. Наявність стрункої і чіткої системи ідентифікації спрощує роботу і дозволяє автоматизувати процеси інформаційного обслуговування користувачів.