2. Приймальне тестування (Acceptance/qualification testing):

Перевіряє поведінку системи на предмет задоволення вимог замовника. Це можливо в тому випадку, якщо замовник бере на себе відповідальність, пов'язану з проведенням таких робіт, як сторона «приймаюча» програмну систему, або специфіковані типові завдання, успішна перевірка (тестування) яких дозволяє говорити про задоволення вимог замовника.

Такі тести можуть проводитися як із залученням розробників системи, так і без них.

- Установче тестування (Installation testing):

З назви випливає, що дані тести проводяться з метою перевірки процедури інсталяції системи в цільовому оточенні.

- Альфа- і бета-тестування (Alpha and Beta testing):

Перед тим, як випускається програмне забезпечення, як мінімум, воно повинно проходити стадії альфа (внутрішнє пробне використання) і бета (пробне використання із залученням відібраних зовнішніх користувачів) версій. Звіти про помилки, що надходять від користувачів цих версій продукту, обробляються відповідно до певними процедурами, що включають підтверджуючі тести (будь-якого рівня), що проводяться фахівцями групи розробки. Даний вид тестування не може бути заздалегідь спланований.

Альфа-тестування - використання незавершеною (альфа) версії ПЗ, в якій реалізована не вся функціональність, запланована для даної версії продукту. При альфа-тестуванні проводиться імітація реальної роботи з системою (виконується штатними розробниками) або реальна робота з системою потенційних користувачів (замовника) з метою виявлення помилок у роботі реалізованих модулів і функцій для їх подальшого усунення перед бета-тестуванням. Найчастіше альфа-тестування проводиться на ранній стадії розробки продукту, але в деяких випадках може застосовуватися для закінченого продукту в якості внутрішнього приймального тестування. Іноді альфа-тестування виконується під відладчиком або з використанням оточення, яке допомагає швидко виявляти знайдені помилки. Виявлені помилки можуть бути передані тестувальникам для додаткового дослідження.

Бета-тестування - інтенсивне використання майже готової версії ПЗ з метою виявлення максимального числа помилок в його роботі для їх подальшого усунення перед остаточним виходом (релізом) продукту на ринок, до масового споживача. У деяких випадках виконується поширення версії з обмеженнями (по функціональності або часу роботи) для деякої групи осіб, з тим, щоб переконатися, що продукт містить достатню мала кількість помилок. Іноді бета-тестування виконується для того, щоб отримати зворотній зв'язок про продукт від його майбутніх користувачів (їх називають бета-тестерами). Бета-версія не є фінальною версією продукту, тому розробник не гарантує повної відсутності помилок, які можуть порушити роботу комп'ютера і привести до втрати даних.

При розробці замовленого ПЗ фазу альфа- і бета-тестування замінюють проектні випробування. Під час цих випробувань замовник засвідчується, що система працює відповідно до його потребами (як зафіксованими в технічному завданні на систему, так і не зафіксованими). Замовник може проводити такі випробування самостійно, виконуючи заздалегідь підготовлені тести системи, або проводити їх спільно з представниками виконавця. У цьому випадку тестові приклади також готуються розробниками, наприклад на основі тестових прикладів, що використовувалися на етапі системного тестування.

Завершуються проектні випробування або підписанням акта приймання, або видачею замовником додаткових вимог до системи, які повинні бути виправлені до приймання системи. Після усунення всіх недоліків системи проектні випробування повторюються (можливо, за скороченою програмою). Після успішного підписання акта система надходить в експлуатацію замовнику.

- Функціональні тести/тести відповідності (Conformance testing/Functional testing/Correctness testing):

Ці тести можуть називатися по різному, проте, їх суть проста - перевірка відповідності системи, висунутим до неї вимогам, описаним на рівні специфікації поведінкових характеристик.

Досягнення та оцінка надійності (Reliability achievement and evaluation):

Допомагаючи ідентифікувати причини збоїв, тестування увазі і підвищення надійності програмних систем. Випадково генеруються сценарії тестування можуть застосовуватися для статичної оцінки надійності. Обидві цілі - підвищення і оцінка надійності - можуть досягатися при використанні моделей підвищення надійності.

- Регресійне тестування (Regression testing):

Визначення успішності регресійних тестів (IEEE 610-90 Standard Glossary of Software Engineering Terminology) говорить: «повторне вибіркове тестування системи або компонент для перевірки зроблених модифікацій не повинно призводити до непередбачуваних ефектів». На практиці це означає, що якщо система успішно проходила тести до внесення модифікацій, вона повинна їх проходити і після внесення таких. Основна проблема регресійного тестування полягає в пошуку компромісу між наявними ресурсами і необхідністю проведення таких тестів в міру внесення кожної зміни. Певною мірою, завдання полягає в тому, щоб визначити критерії «масштабів» змін, з досягненням яких необхідно проводити регресійні тести.

- Тестування продуктивності (Performance testing):

Спеціалізовані тести перевірки задоволення специфічних вимог, що пред'являються до параметрів продуктивності. Існує особливий підвид таких тестів, коли робиться спроба досягнення кількісних меж, обумовлених характеристиками самої системи та її операційного оточення.

- Навантажувальне тестування (Stress testing):

Необхідно розуміти відмінності між розглянутим вище тестуванням продуктивності з метою досягнення її реальних (досяжних) можливостей продуктивності та виконанням програмної системи з підвищенням навантаження, аж до досягнення запланованих характеристик і далі, з відстеженням поведінки на всьому протязі підвищення завантаження системи.

- Порівняльне тестування (Back-to-back testing):

Одиничний набір тестів, що дозволяють порівняти дві версії системи.

- Відновлювальні тести (Recovery testing):

Мета - перевірка можливостей рестарту системи у випадку непередбаченої катастрофи (disaster), що впливає на функціонування операційного середовища, в якій виконується система.

- Конфігураційне тестування (Configuration testing):

У випадках, якщо програмне забезпечення створюється для використання різними користувачами (в термінах «ролей»), даний вид тестування спрямований на перевірку поведінки і працездатності системи в різних конфігураціях.

- Тестування зручності та простоти використання (Usability testing):

Мета - перевірити, наскільки легко кінцевий користувач системи може її освоїти, включаючи не тільки функціональну складову - саму систему, але і її документацію; наскільки ефективно користувач може виконувати завдання, автоматизація яких здійснюватися з використанням даної системи; нарешті, наскільки добре система застрахована (з погляду потенційних збоїв) від помилок користувача.

- Розробка, керована тестуванням (Test-driven development):

По суті, це не стільки техніка тестування, скільки стиль організації процесу розробки, життєвого циклу, коли тести є невід'ємною частиною вимог (і відповідних специфікацій) замість того, щоб розглядатися незалежної діяльністю з перевірки задоволення вимог програмною системою.

3. Автоматизоване тестування використовує програмні засоби для виконання тестів і перевірки коректності результатів виконання, що спрощує тестування і скорочує його тривалість. Головна перевага автоматизованого тестування полягає в можливості повторного прогону тестів без участі людини.

Традиційний і найбільш популярний серед розробників спосіб полягає в організації автоматизації тестування на рівні коду. Даний підхід буде детально розглянуто при описі модульного тестування. Автоматизоване тестування на рівні коду часто критикують за неможливість тестування користувальницького інтерфейсу програми. Однак прихильники TDD показали, що при правильному використанні патернів сімейства MVC (Model-View-Controller) можливо організувати програмну імітацію дій користувача без використання GUI (Graphical User Interface). Цей підхід дозволяє організувати тестування обробників дій користувача, залишаючи не покритий тестами лише частину, що відноситься до безпосереднього відображенню даних.

Другий спосіб автоматизації тестування полягає в імітації дій користувача з використанням спеціальних інструментальних засобів (GUI-тестування). Даний вид тестування відноситься до тестування методом «чорного ящика».

Існують чотири покоління інструментів і технік, призначених для організації GUI-тестування:

- Утиліти запису і відтворення (capture/playback tools) записують дії під час ручного тестування. Надалі вони дозволяють відтворити раніше записані дії без участі людини, значно збільшуючи продуктивність і усуваючи повторення одноманітних дій. Основним недоліком інструментальних засобів даного покоління є те, що будь-яка зміна розташування візуальних елементів програми призводить до необхідності повторної записи ручних тестів.

- Сценарій (scripting) - форма автоматизації тестування з використанням спеціалізованих скриптових мов. Мова повинен підтримувати емуляцію дій користувача та отримання результатів дій. Розробкою тестів займаються програмісти, які працюють окремо від тестувальників, безпосередньо запускають те6сти. Зміни в тестованому ПО вимагають внесення виправлень і у відповідних скриптах.