**Лекція №4. Моделі надійності в програмній інженерії**

Основою інженерних методів у програмуванні є підвищення якості. Для досягнення цього були  сформульовані  методи визначення вимог до якості, підходи до вибору й удосконалення моделей метричного аналізу показників якості, методи кількісного виміру ризиків  на процесах ЖЦ.

Головна складова якості –**надійність**, якій приділяється велика увага у сфері надійності технічних засобів і тих критичних систем (реальний час, радарні системи, системи безпеки й ін.), для яких надійність є головною цільовою функцією оцінки їхньої реалізації. Як наслідок у проблематиці надійності розроблено понад сотні математичних моделей надійності, що є функціями  помилок, які залишилися в ПС,  інтенсивності відмов або частоти виникнення дефектів у ПС. На їхній основі здійснюється оцінка надійності ПС.

Надійність – одна з ключових проблем сучасних програмних систем, і її роль буде постійно зростати, оскільки постійно підвищуються вимоги до якості комп'ютерних систем. Новий напрямок – інженерія програмної надійності (Software reliability engineering) – орієнтується на кількісне вивчення операційної поведінки компонентів системи відносно  користувача, який очікує надійну роботу системи. Він складається з:

- вимірювання надійності, тобто проведення її кількісної оцінки за допомогою передбачень, збирання даних про поведінку системи в процесі експлуатації й сучасних моделей надійності;

- стратегії й метрики конструювання й вибору готових компонентів, процес розробки компонентної системи, а також середовище функціонування, що впливає на надійність роботи системи;

- застосування сучасних методів інспектування, верифікації, валідації й тестування під час розроблення систем та їх експлуатації.

Верифікація застосовується для встановлення відповідності готового ПС до визначених специфікацій, а валідація – для встановлення відповідності системи до вимог користувача, висунутих замовником.

Оцінка надійності ПС – це трудомісткий процес, що вимагає створення сталої роботи системи щодо помилок  ПС, тобто ймовірності того, що система відновиться довільно в певній точці після виникнення і внесення змін через  відмови і інші ситуації.

**Суть і основні елементи теорії надійності**

Обчислювальні пристрої, ПК і комп‘ютерні системи є важливою складовою сучасного виробництва. Нині ця галузь розвивається особливо інтенсивно. Однак розвиток засобів комп‘ютерної техніки неможливий без забезпечення їх надійності. Її , в свою чергу, характеризують багато показників і параметрів. Однією з особливостей комп‘ютерних систем і мереж порівняно з іншими, наприклад, механічними , є те, що їх надійність визначає надійність елементів та надійність програмного забезпечення.

***Надійність*** – властивість об‘єкта зберігати в часі у встановлених межах значення всіх параметрів, які характеризують здатність виконувати потрібні функції в заданих режимах та умовах застосування, технічного обслуговування, зберігання і транспортування.

***Відмова*** – подія, яка полягає в утраті функційним модулем (системою) здатності виконувати потрібну функцію.

***Ймовірність безвідмовної роботи P(t)*** - функція розподілу ймовірностей безвідмовної роботи об‘єкта, яка характеризує ймовірність того, що об‘єкт за час t не втратить працездатності. Ймовірність безвідмовної роботи є одним із основних показників надійності технічних виробів. Значення P(t1, t2) за відносно короткі проміжки часу (година, доба) для сучасних електронних пристроїв близьке до 1. Наприклад, ймовірність безвідмовної роботи типової інтегральної схеми(ІС) за кілька годин роботи становить 0,999999. Проводити розрахунки з такими числами незручно тому доцільно перейти до відповідних ймовірностей відмови – 10-7...10-6.

***Технічний ресурс виробу*** - прогнозована тривалість його експлуатації від початку до моменту, коли інтенсивність відмов збільшується до рівня, який робить подальшу експлуатацію неможливою або недоцільною з економічних міркувань.

## Методи забезпечення надійності

В теорії надійності всі об‘єкти поділяють на два великих класи: відновлювані і невідновлювані. Підхід для забезпечення надійності обох класів зовсім різний. Забезпечення надійності відновлюваних об‘єктів полягає в організації ефективного ремонту обладнання з мінімальними затратами часу.

**Відновлювані об‘єкти** – об‘єкти, які в період експлуатації в разі виникнення відмов можуть бути «відремонтовані» шляхом заміни несправних компонентів. Для відновлюваних об‘єктів основним показником надійності вважають коефіцієнт готовності, що визначає частину корисного часу tk протягом якого об‘єкт нормально працює щодо загального часу експлуатації. Тобто: https://elib.lntu.edu.ua/sites/default/files/elib_upload/%D0%95%D0%9D%D0%9C%D0%9A_%D0%9D%D0%B0%D0%B4%D1%96%D0%B9%D0%BD%D1%96%D1%81%D1%82%D1%8C/page6.files/image002.png (1)

де - tb час, що втрачено на відновлення працездатності. Він містить дві складові: t1 – час, що втрачено на пошук несправності; t2 – час власне відновлення.

Особливістю сучасного електронного обладнання є його модульна побудова, тому відновлення таких об‘єктів найчастіше може полягати у простій заміні несправного модуля (блока) справним. Вона не потребує великих затрат часу. Пошук несправності-навпаки, і впливає саме на коефіцієнт готовності. До основних методів забезпечення надійності відновлюваної електронної апаратури належать:

- створення максимально сприятливих умов для прискорення ремонту;

- застосування автоматизованого пошуку несправностей на основі апаратного або програмного самоконтролю; тестового діагностування та використання спеціальних автоматизованих систем контролю і діагностики.

Для невідновлюваних об‘єктів підхід до забезпечення надійності зовсім інший, оскільки вони не підлягають ремонту.

**Невідновлювані об‘єкти** - об‘єкти, для яких ремонт із конструктивних причин або умов експлуатації є неможливим. Наприклад, інтегральні схеми, кінескопи, мікропроцесорні системи керування бортової апаратури тощо. Показники надійності таких пристроїв залежать від складності невідновлюваних об‘єктів, при ймовірності безвідмовної роботи менше ніж 0,5 система стає непрацездатною. Однак цьому можна запобігти, якщо застосувати методи введення надлишковості. Надлишковість поділяють на часову, інформаційну, структурну(апаратну).

Комп‘ютерні системи, які використовують для управління реальними об‘єктами, не можна беззастережно зарахувати до відновлюваних. ні до невідновлюваних об‘єктів. До методів забезпечення надійності комп‘ютерних систем належать як методи, що застосовують для відновлюваних систем (прискорення пошуку несправностей), так і властиві невідновлюваним системам методи, наприклад, (структурна надлишковість). Принципова особливість таких систем полягає в багаторазових обчисленнях. Вони утворюються шляхом n-кратного повторення обчислювального процесу в таких розрізах:

·         часовому (повторний рахунок);

·         просторовому (на інших апаратних засобах);

·         інформаційному (з використанням інших програм і даних).

## Резервування апаратури

Одним із методів підвищення надійності обчислювальних систем є резервування її складових.

**Резервування** – метод підвищення надійності шляхом включення резервних елементів при розроблені системи або в процесі її експлуатації. Розрізняють поелементне і загальне резервування. За включенням резерву виокремлюють постійне резервування і резервування заміщенням. Постійне резервування полягає у включенні резервної апаратури протягом усього часу функціонування основної. Резервування заміщенням полягає у включенні резервного елемента структури замість елемента, який вийшов з ладу. За такого резервування резерв може бути:

·         ненавантаженим(апаратура не увімкнута у процесі функціонування);

·         полегшеним (увімкнута частина резервної апаратури);

·         навантаженим (увімкнута вся резервна апаратура).

До переваг цього способу резервування належать:

·         один резервний елемент використовують для кількох основних

·         резервні елементи знаходяться у пасивному стані(наприклад, знеструмлені), що зберігає їх технічний ресурс.

Мажоритарний метод резервування вважається класичним. Він був запропонований Дж.фон Нейманом. Реалізують його як постійне резервування з логічним перемиканням на резервні елементи. Елементом цього методу є відновлюючий орган (ВО), функція якого полягає в утворенні результуючого сигналу за більшистю значень вхідних сигналів.

**Надійність програмного забезпечення**

Забезпечення надійності програмних засобів ПК і обчислювальних систем істотно відрізняється від забезпечення надійності апаратури. Щоб зрозуміти це потрібно ознайомитися із загальною методологією діагностування ПЗ.

**Системний підхід** – сумісний аналіз і розроблення всіх елементів системи.

**Модуль** – елемент програми, стандартизований за формою запису і зовнішніми зв’язками.

**Надійність програмного забезпечення** – властивість програми виконувати задані функції в заданих умовах роботи і на заданій ЕОМ.

**Безвідмовність програмного забезпечення**. Оцінюють імовірністю роботи ПЗ без відмов протягом заданого часу в певних умовах зовнішнього середовища.

**Відновлювальність ПЗ**. Її характеризують витрати часу і праці на усунення відмов через помилки в програмі та їхні наслідки. Високу надійність програми забезпечує швидке реагування на її спотворення, а також спотворення даних або обчислювального процесу, відновлення працездатності програм за час, менший ніж поріг між збоєм і відмовою.

**Усталеність функціювання**. Це одна з характеристик ПЗ, яка полягає в здатності обмежувати наслідки власних помилок і несприятливих впливів зовнішнього середовища або протистояти їм. Механізм виникнення відмов - переважно це порушення специфікації. Трапляється це насамперед у складних програмних системах, де окремі помилки програміста важко виявити. Неточна або неповна специфікація виникає, коли при її складанні є невідомі фактори, що впливають на роботу програми.

**Коректність програми** – це відповідність її специфікації. Прихованість помилок полягає у виявленні помилки в результаті великої кількості комбінацій вихідних даних. Інтенсивність відмов – інтегральний критерій, що характеризує густину розподілу напрацювання до першої відмови, розраховану за умови, що до моменту часу, який розглядають система пропрацювала безвідмовно. Оцінювання і прогнозування надійності програм. Ці процедури здійснюються на основі математичних моделей Джелінські-Моранди, Шумана, Шика-Волвертона.

Експериментальне оцінювання кількості помилок у програмі: визначають стосовно кількості виявлених природних і штучних помилок, якщо в програму була введена певна кількість однотипних з природними штучних помилок.

Надійність програмного комплексу: визначають її як функцію надійності складових частин програмного комплексу.

**Помилки програмного забезпечення**. Під поняттям „помилка” розуміють неправильність, похибку або ненавмисне спотворення об’єкта чи процесу. При цьому приймають, що правильний стан об’єкта або процесу відомий. Залежно від складності первинні помилки ПЗ поділяють на такі типи:

* технологічні помилки підготовки машинних носіїв і документації, а також виведення і введення програм у комп’ютер та його засоби відображення
* алгоритмічні помилки, пов’язані з неповним формуванням необхідних умов розв’язання і некоректною постановкою завдання
* системні помилки, зумовлені відхиленням функціювання ПЗ у реальній системі від очікуваних під час проектування
* програмні помилки, пов’язані з некоректною постановкою завдання.

**Моделі надійності**

 В останні роки вартість розроблення програмного забезпечення (ПЗ) та витрати на відмови програмних систем стали одними із основних затрат на розроблення складних технічних систем. Відмова ПЗ може призвести до неочікуваного стану системи або потоку непередбачених дій. Внаслідок відмови ПЗ можливі пошкодження чи руйнування майна, поранення або загибель людей, втрата грошових коштів.

Загальновизнаним в індустрії програмного забезпечення є той факт, що висококваліфіковані програмісти допускають в середньому 6 помилок в програмному забезпеченні при написанні 1000 рядків коду. При такому показнику звичайне комерційне ПЗ, яке складається з 350000 рядків коду, містить близько 2000 помилок, зокрема помилки, пов’язані з витоком пам’яті, пов’язані з текстом програми, помилки використання сторонніх бібліотек, стандартні бібліотечні помилки та інші. Тому питання оцінювання та забезпечення надійності ПЗ відіграють все більшу роль у розробленні складних технічних систем, а побудова все адекватніших моделей надійності ПЗ є важливою науково-технічною проблемою.

**Загальна характеристика та класифікація моделей надійності програмного забезпечення**

Дослідження в галузі надійності ПЗ беруть початок з 1970-х років. Незважаючи на зусилля провідних світових розробників ПЗ, завдання зниження кількості помилок у програмних системах не отримало ефективного практичного вирішення. Об’єктивно це зумовлено надзвичайно високою структурною складністю програмних систем, динамічністю версій і технологій. Одним із шляхів підвищення рівня надійності ПЗ є використання на етапах тестування і випробувань ПЗ моделей, що дають змогу отримати гарантовані оцінки показників безпеки ПЗ і ефективності технології його розроблення. Більшість таких моделей запозичено з теорії надійності технічних систем, тому в літературі їх часто називають моделями надійності ПЗ.

Надійність складних ПС істотно відрізняється від надійності апаратур. Носії даних (файли, сервер і т.п.) мають високу надійність, записи на них можуть зберігатися тривалий час без руйнування, оскільки фізичному руйнуванню вони практично не підлягають.

З погляду прикладної науки *надійність* – це здатність ПС зберігати свої властивості (безвідмовність, стійкість та ін.), перетворювати вихідні дані на результати протягом певного проміжку часу за певних умов експлуатації. Зниження надійності ПС відбувається через помилки у вимогах, проектуванні й виконанні. Відмови й помилки залежать від способу виробництва продукту й з'являються в програмах при їхньому виконанні на деякому проміжку часу.

Таким чином, оцінка надійності ПС залежить від числа помилок, що залишилися не усунутими у програмах. У процесі експлуатації ПС помилки виявляють й усувають. Якщо при виправленні помилок не вносяться нові або, принаймні, нових помилок вноситься менше, ніж усувається, то в процесі експлуатації надійність ПС безперервно зростає. Чим інтенсивніше здійснюється експлуатація, тим інтенсивніше виявляються помилки й швидше зростає надійність системи й відповідно її якість.

Надійність є функцією  помилок, що залишилися в ПС після введення його до експлуатації. ПС без помилок є абсолютно надійним. Але для більшості програм абсолютна надійність практично недосяжна. Помилки, що залишилися незнайденими, виявляють себе час від часу за певних умов (наприклад, при певній сукупності вихідних даних) супроводу й експлуатації системи.

Для оцінки надійності ПС використовуються такі статистичні показники, як ймовірність і час безвідмовної роботи, можливість відмови й частота (інтенсивність) відмов. Оскільки як причини відмов розглядаються тільки помилки в програмі, які не можуть самоусуватися, то ПС варто відносити до класу  систем, що не оновлюють.

При кожному вияві нової помилки, як правило, здійснюється її локалізація й виправлення. При цьому накопичена до цього статистика про відмови втрачає своє значення, оскільки після внесення змін програма, власне кажучи, є новою програмою на відміну від тієї, яка до цього випробовувалася.

У зв'язку з виправленням помилок у ПС надійність, тобто її окремі атрибути, будуть увесь час змінюватися, як правило, у бік покращання. Отже, їхня оцінка буде мати тимчасовий і наближений характер. Тому виникає необхідність у використанні нових властивостей, адекватних реальному процесу вимірювання надійності, таких, як залежність інтенсивності виявлених помилок від числа прогонів програми й залежність відмов від часу функціонування ПС тощо.

Для багатьох систем (реального часу, радарні системи, системи безпеки, медичне устаткування з вбудованими програмами та ін.) надійність – головна цільова функція їх реалізації. Для таких  типів систем  висувають високі вимоги до надійності, такі, як відсутність помилок, вірогідність, безпека та ін.

Від 70-х років минулого століття було розроблено значну кількість моделей надійності ПЗ. Як основні критерії класифікації таких моделей вибрано два доволі прості предмети дослідження:

1) дослідження кількості відмов за певний період часу (вимір часу “в режимі настінного годинника” або вимір часу відносно виконання процесів пристроями комп’ютера);

2) дослідження часових проміжків між помилками.

Вважали, що моделі, класифіковані цим способом, взаємно не перетинаються і можуть містити часткові випадки в кожному дослідженні. Однією з перших складних моделей надійності ПЗ вважають модель Муси і Окумото. Для побудови цієї моделі використано набір атрибутів, зокрема:

• часовий проміжок;

• загальну кількість відмов, які можливо виявити за нескінченний або скінченний проміжок часу;

• розподіл кількості відмов, які відбулися за час *t* (розподіл Пуассона або біноміальний розподіл);

• клас відмови, або функціональна форма активності відмов за певний час (застосовується тільки для випробувань із скінченними часовими проміжками);

• тип відмови, або вигляд функції інтенсивності відмов протягом нескінченного проміжку часу (застосовується тільки для випробувань із нескінченними часовими проміжками).

З розвитком технологій розроблення і практичного застосування ПЗ схема класифікації моделей надійності ПЗ значно розширилася. Не тільки розроблено нові критерії класифікації, але й ускладнено саму структуру класифікації внаслідок об’єднання і перетину декількох критеріїв у різних моделях.

До найбільш вживаних методик і факторів надійності ПЗ належать: стадія життєвого циклу розроблення ПЗ (класифікація моделей залежить від етапу, на якому розраховується надійність ПЗ), можливість раннього прогнозування помилок, орієнтованість на інформацію або архітектуру (класифікацію роблять на підставі перевірки правильності вхідних / вихідних даних, або перевірки функціонального наповнення ПЗ), зростання надійності ПЗ в процесі виявлення та виправлення помилок тощо. Класифікація за такими факторами є найповнішою та дає можливість представити не тільки самі моделі, але і взаємозв’язок між ними (рис. 1).

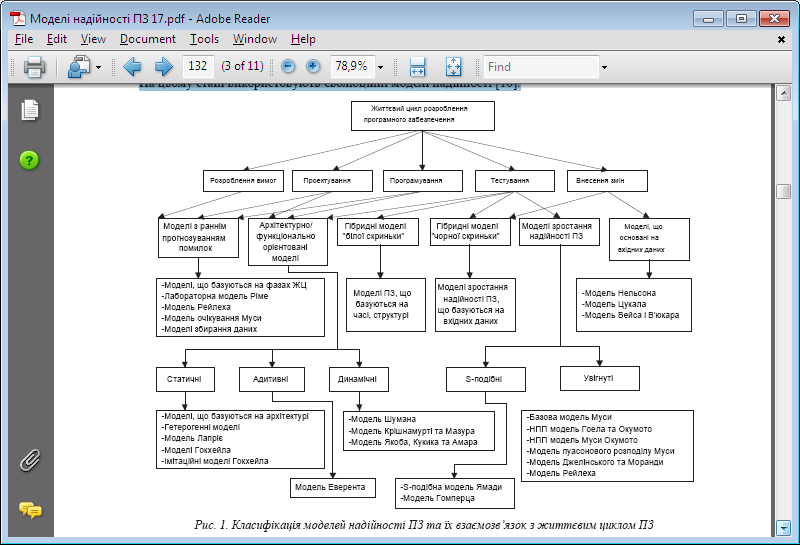
Кількість моделей надійності ПЗ сьогодні перевищує сотню і продовжує зростати.

Залежно від визначення поняття надійності ПЗ і глибини дослідження обирають різні критерії і характеристики моделей надійності. Тому наведена на рис. 1 схема класифікації моделей надійності ПЗ, як і розглянуті критерії, не є єдиною.

Схема, наведена на рис. 1, поділяє всі моделі надійності ПЗ на аналітичні та емпіричні.

Емпіричні моделі надійності, своєю чергою, поділяються на моделі складності і моделі, що визначають час, необхідний на “доведення” програми. Аналітичні моделі надійності ПЗ поділяють на динамічні (дискретні і неперервні) і статичні (за областю помилок, за областю даних).

Життєвий цикл ПЗ загалом зазвичай поділяють на такі етапи: специфікація вимог, проектування, кодування, тестування, експлуатація і супровід. Етап проектування може включати попереднє проектування і деталізоване проектування. Етап тестування може містити модульне, інтеграційне та регресійне тестування, тестування в умовах експлуатації. Етап супроводу може містити один або два цикли, кожен з яких має всі етапи стадії розробки. Згідно з ДСТУ -1999 (ІSO/IEC 12207:1995) основними процесами життєвого циклу ПЗ є процеси: замовлення, постачання, розроблення, експлуатації та супроводу. Своєю чергою, процес розроблення ПЗ містить такі дії: реалізацію процесу, аналіз системних вимог, проектування архітектури системи, аналіз вимог до ПЗ, проектування архітектури ПЗ, розроблення детального проекту ПЗ, кодування і тестування ПЗ, інтеграцію ПЗ, кваліфікаційне тестування ПЗ, системну інтеграцію, кваліфікаційне тестування системи, встановлення ПЗ та забезпечення приймання ПЗ.



*Рис. 1. Класифікація моделей надійності ПЗ та їх взаємозв’язок з життєвим циклом ПЗ*

На ранніх стадіях життєвого циклу ПЗ потрібна модель прогнозування надійності, оскільки даних про відмови немає. Моделі такого типу призначені для передбачення кількості помилок у програмі перед тестуванням, і в деяких літературних джерелах належать до детерміністичних (статичних) моделей надійності ПЗ. На етапі тестування показники надійності ПЗ покращуються завдяки відлагодженню програми. Модель зростання надійності на цьому етапі потрібна для оцінювання поточного рівня надійності, часу і ресурсів, потрібних для досягнення заданого рівня надійності ПЗ. Впродовж цього етапу оцінка надійності ґрунтується на аналізі цих відмов. Моделі такого типу належать до імовірнісних (динамічних) моделей надійності ПЗ.

Після введення програми в експлуатацію при визначенні її надійності необхідно враховувати додавання нових модулів, усунення старих модулів, усунення виявлених помилок, поєднання нового коду з попередньо написаним кодом, зміну середовища користувача, зміну апаратного забезпечення тощо. На цьому етапі використовують еволюційні моделі надійності.

За моделями, в основу яких покладено підрахунок відмов (динамічні моделі), припускають, що концептуально в програмі наявна скінченна кількість помилок. Враховуючи, що кількість помилок є цілим числом, за динамічними моделями обчислюють кількість початкових помилок на етапі відлагодження програми і кількість помилок, що залишились під час чи в кінці етапу відлагодження. Моделі підрахунку відмов використовують інтенсивність відмов як основну характеристику появи відмови. Залежно від типу моделі припускають, що інтенсивність відмов кожної помилки є або сталою функцією часу відлагодження, або випадковою змінною із заданим законом розподілу. Як тільки інтенсивність відмов, пов’язана з помилками певного типу, визначено, інтенсивність відмов програми загалом обчислюють як добуток кількості помилок, що залишились в програмі, на інтенсивність відмов, породжених помилкою кожного типу.

Під час етапу відлагодження кількість помилок, що залишились, змінюється. Одним із способів моделювання такого процесу відмов є представлення кількості помилок, що залишились, як стохастичного процесу.

Оцінка надійності ПЗ є все важливішою під час розроблення та тестування нових програмних продуктів. Перед тим, як нове ПЗ випустити в користування, його ретельно перевіряють на наявність помилок, які могли з’явитися в процесі розроблення ПЗ. Незважаючи на те, що виявлені помилки негайно видаляються, нові помилки можуть бути введені під час процесу відлагодження. Програмне забезпечення, яке містить помилки і випускається на ринок, несе високі ризики, пов’язані з відмовами. Відлагодження і тестування, з іншого боку, зменшує кількість помилок, але збільшує витрати на розроблення. Отже, існує необхідність визначити оптимальний час для зупинки тестування ПЗ. Під час тестування системи надійність є важливим критерієм при ухваленні рішення, коли саме потрібно випустити ПЗ. Декілька інших критеріїв, такі як кількість помилок, що залишилися, частота відмов, вимоги до надійності або загальна вартість системи можуть бути використані для визначення оптимального часу тестування.

**Формальне визначення моделей надійності ПС.**

З теоретичної точки зору під *надійністю*ПС розуміють здатність системи зберігати свої властивості (безвідмовність, відновлюваність) на певному рівні протягом фіксованого проміжку часу за певних умов експлуатації. Інакше кажучи, є імовірність того, що функціонуюча ПС у заданому середовищі не дасть відмов у інтервалі часу *(t,. t+k)* за умови, що остання відмова ПС й усунення відповідної помилки відбулися до моменту часу *t.* Чим більше часу працює система без відмов, тим вона надійніше.

Дослідження надійності проводиться за допомогою методів теорії ймовірностей, математичної статистики, відновлення систем й ін. Формально надійність можна визначити як таку ймовірність:

*P (i) = P*, якщо немає відмов у*i* прогонах програми на тестах,

*P(t) = P,* якщо немає відмов у інтервалі часу  *(0, t )* виконання програми.

Якщо при відмові працездатність системи підлягає відновленню в процесі експлуатації, то така система називається відновлюваною. Головними показниками такої системи є час безвідмовної роботи ПС і час її відновлення після відмови.

**Шляхи забезпечення надійності ПЗ.**

Основні фактори, що визначають надійність функціювання ПЗ, об’єднують у 3 групи:

·         фактори, які безпосередньо визначають надійність програм

·         методи проектування коректних програм

·         методи контролю і забезпечення надійності програм.

Визначальне значення для забезпечення надійності програм має третя група факторів. Для її підвищення і захисту інформації програмно-алгоритмічними методами використовують часову, інформаційну та програмну надлишковість.

**Часова надлишковість**. Це використання певної частини апаратних засобів і продуктивності ЕОМ, зокрема ПК, для контролю виконання програм і відновлення обчислювального процесу. Передбачається певний резерв продуктивності, необхідний для контролю та підвищення надійності функціювання ПЗ.

**Інформаційна надлишковість**. Вона полягає у дублюванні накопичених початкових і проміжних даних. Її використовують для забезпечення достовірності даних, які потребують значного часу для відновлення чи найбільше впливають на нормальне функціювання ПЗ. Вони характеризують певні інтегральні відомості про зовнішній керований процес, і в разі їх руйнування можуть надовго припинити керування цим процесом та відповідне оброблення інформації.

**Програмна надлишковість**. Вона контролює і забезпечує достовірність найважливіших рішень з оброблення інформації та керування. Її сутність полягає в застосуванні кількох варіантів програм, що відрізняються методами розв’язання певного завдання або програмною реалізацією одного й того самого методу.

## Методологія діагностування програмного забезпечення(ПЗ)

Методологія діагностування ПЗ пояснює суть методів і способів тестування програм, методики побудови тестів, процедур, які доводять програми до робочого стану. Вона визначає також послідовність видів тестування. Діагностування як процес виявлення помилок у ПЗ є одним з основних методів підвищення його надійності. Здійснюють діагностування на всіх етапах життєвого циклу ПЗ, а саме на етапі планування, проектування і кодування. Суть тестування полягає у перевірці роботи програм за результатами їх виконання зі спеціально підібраними наборами вихідних даних (тестами). Діагностування ПЗ здійснюють також і методом верифікації. У процесі діагностування ПЗ крім тестування використовують процедури доведення, контролю, випробування, атестації, відлагодження. З погляду на обсяг тестування може бути повним або вибірковим. За повного тестування програму тестують повністю, а за вибіркового – в окремих точках блоку вхідних даних. Під час вибіркового тестування надійність програми повністю не забезпечують. Частина помилок може бути прихованою і не виявитися. Тому найчастіше використовують структурне вибіркове тестування. Воно полягає у розділенні блока вхідних даних на класи, кожен з яких дає змогу підтвердити певні властивості або працездатність окремих елементів структури. Вибір шляхів у структурі програми повинен здійснюватись так, щоб усі оператори програми були задіяні хоча б один раз.

Для перевірки результатів роботи програми використовують числове і символьне тестування. **Числове** полягає у перевірці правильності числових результатів роботи програм при певних тестових наборах. **Символьне** передбачає виконання процедур, що ґрунтуються на символьних входах. Вхідні змінні позначають так, що вони дають змогу виразити вихідні змінні також у символьному вигляді. При цьому різним шляхам у програмі відповідають різні символьні входи і виходи. Контроль складових ПЗ необхідно проводити у зазначеній послідовності й повному обсязі. Його визначають залежно від вимог надійності.

## Тестування модулів та комплексне тестування програмної системи

Процес тестування ПЗ починається з тестування модулів за наявності спроектованих тестів. Тому слід приділити цим процедурам особливу увагу. Суть процесу тестування модулів. При тестуванні ПЗ увагу приділяють насамперед тестуванню автономних модулів. Суть цього процесу полягає в контролі окремого програмного модуля. Його метою є знаходження невідповідностей між логікою модулів та їх спряганням, з одного боку, і з зовнішніми специфікаціями, з іншої. Зовнішні специфікації охоплюють опис функцій, вхідних і вихідних даних та зовнішніх ефектів. Компіляцію модуля теж розглядають як тестування, оскільки компілятор виявляє більшість синтаксичних помилок, а також певні логічні і семантичні.

Процес тестування модуля здійснюється у кілька етапів:

* підготовка на основі зовнішніх специфікацій модуля тесту для кожної ситуації, для кожної з меж областей допустимих значень всіх недопустимих умов – усіх вхідних даних та областей зміни даних;
* перевірка тесту програми.
* ·з’ясування того, чи охоплюють тести відповідно до тексту програми велику кількість можливих шляхів;
* перевірка чутливості програми, згідно з її текстом, стосовно окремих особливих значень вхідних даних.

Тестування автономних модулів здійснюють на основі кількох аксіом, що є фундаментальними принципами тестування:

* Високоякісним є той тест, який має високу ймовірність виявлення помилок, а не той, що демонструє правильну роботу програми;
* Однією з найважливіших проблем при тестуванні , виходячи з першої аксіоми, є з’ясування термінів закінчення тестування;
* Неможливо тестувати особисту програму;
* Необхідною частиною будь-якого тесту повинен бути опис очікуваних вихідних даних або результатів;
* Слід уникати невідтворюваних тестів;
* Необхідно розробляти тести як для правильних, так і для неправильних вхідних даних;
* Слід детально аналізувати результати проходження кожного тесту;
* Тестування проводять найкваліфікованіші програмісти;
* При розробленні ПЗ тестованість є ключовим завданням;
* Проект системи повинен передбачати , що кожний модуль, який підключається до системи , має бути в єдиному варіанті;
* Не можна змінювати програму з метою полегшення її тестування;
* Тестування слід починати з визначення цілей;
* Усі розгалуження потрібно виконувати в кожному з можливих напрямів хоча б один раз.

Тестування модулів ПЗ здійснюють на основі відповідних методів – статичного і динамічного.

**Статичне** тестування полягає в тому, що одним з критеріїв обрання шляхів у структурі програми при тестуванні є задіяння всіма операторами хоча б один раз обраних шляхів. При цьому тестують команди за послідовністю їх розташування в тексті програми. Статичний аналіз програми спеціалісти-тестувальники виконують неавтоматизовано.

**Динамічне** тестування (тестування гілок). Вимагає обрання таких шляхів, які перекривають усі гілки програми або всі розгалуження в усіх напрямах. Цей метод гарантує однократне тестування всіх гілок і операторів. Головна вимога до програм або програмних систем – виконувати задані функції. Вони найповніше реалізуються при застосуванні функційного, структурного, символьного і регресивного тестування.

**Інтеграція модулів, тестування зовнішніх функцій і комплексів програм**

Тестування модулів є дуже важливим етапом діагностування комплексу ПЗ. Наступними завданнями, які розв’язують під час розроблення і налагодження ПЗ, є інтеграція модулів у систему , її комплексне тестування і налагодження. Суть і методи інтеграції програмних модулів. Після проектування здійснюють інтеграцію модулів – злиття модулів у програму або програмну систему. Вибір послідовності злиття модулів у систему визначає форму, в якій записуються тести, послідовність програмування модулів, вибір засобів тестування і відповідно ефективність тестування.

**Висхідне** тестування. Суть його полягає в тому, що програму збирають і тестують знизу до гори. Тестують першими і автономно модулі найнижчого рівня. Потім – модулі , що безпосередньо викликають уже перевірені. Для тестування модулів вищого рівня повинні бути протестовані всі модулі нижчого рівня, які він викликає. За висхідного тестування для тестування кожного модуля потрібен драйвер – програма, що імітує надходження інформації з попередніх модулів.

**Низхідне** тестування. Полягає в тому, що програму збирають і тестують зверху вниз. Автономно тестують тільки головний модуль, що відповідає вершині графа. Потім до нього приєднують один за одним модулі, безпосередньо зв’язані з ним, і тестують одержану комбінацію. Процес повторюють, поки не будуть зібрані й перевірені всі модулі . низхідне тестування дає змогу чітко дотримуватися певної послідовності тестування.

Метод «**великого стрибка**» . Він є одним з найпоширеніших методів тестування. Використовують його лише для невеликих програм. Особливість цього методу полягає в автономному тестуванні спочатку кожного модуля. Потім усіх їх одночасно інтегрують у систему. Від низхідного тестування його відрізняє те, що всі модулі інтегрують у систему одночасно в останній момент. Це зумовлює ситуацію, коли у спряганнях між модулями тривалий час залишаються помилки, які можуть бути непоміченими. Для невеликих добре спроектованих програм, метод „великого стрибка” ефективний, для великих програм – неприйнятий.

**Метод Сандвича** . Це компромісний варіант між висхідним і низхідним тестуванням. Головна мета його полягає у використанні переваг цих методів й усуненні їх недоліків. Суть методу полягає в тому, що висхідне і нисхідне тестування починаються одночасно. Програміст збирає програму як знизу, так і зверху. Використання цього методу має сенс при інтеграції і тестуванні великих програмних систем ( операційних систем ) та пакетів прикладних програм.

Важливу роль при діагностуванні ПЗ відіграють тестування спрягань і зовнішніх функцій. Тестування спрягань - під час цієї процедури перевіряють взаємозв’язок між модулями щодо використовуваної інформації і часу її надходження. Тести для контролю спрягань розробляють на основі аналізу архітектури програмної системи і структури програми. Цей аналіз грунтується на схемі інформаційних зв’язків у графі програмної системи і матриці взаємозв’язків модулів. Характер вихідних даних для кожного модуля визначають з таблиць вихідних змінних.

Тестування зовнішніх функцій. Його здійснюють з метою виявлення розходжень між програмою і її зовнішніми специфікаціями. Тому необхідною умовою такого тестування є наявність чітких і точних специфікацій. Тести зовнішніх функцій розробляють так, щоб їх можна було використати багаторазово.

Збирання і комплексне тестування програмної системи. Після завершення автономного тестування модулів, тестування спрягань і тестування зовнішніх функцій приступають безпосередньо до інтеграції програмної системи. Цей процес починають з того, що складають план поступового збирання системи. Кожний рівень або версію системи називають спіном. Збирання починають з нульового базового спіну і закінчують N-м спіном. За нульовий спін приймають операційну систему. Збирання призначене виділити збірні вузли і скласти календарний план їх розроблення . Ця робота повинна відповідати таким критеріям:

* у плані збирання слід враховувати обмеження, пов’язані з наявними ресурсами і графіком розроблення діагностичного забезпечення;
* послідовність збирання необхідно базувати на каркасній (0-й спін) версії системи, що вже готова і працює;
* у плані має враховуватися залежність одних збірних вузлів від інших;
* послідовність збирання має забезпечувати включення найважливіших функцій в перші спіни;
* у плані збирання слід мінімізувати застосування програм, що імітують відсутні у всіх спінах, крім останнього , певні фрагменти системи.

Після збирання системи здійснюють її комплексне тестування.

**Комплексне тестування** – пошук невідповідності системи своїй вихідній (початковій) меті. Комплексні тести не зводять до перевірки окремих функцій системи. Їх розробляють у формі сценаріїв – послідовних дій користувача. Загальні тенденції щодо розроблення комплексних тестів. Отже, тестування проводити:

* стресових ситуацій реального середовища;
* великих обсягів даних протягом тривалого часу;
* конфігурації апаратури ПЗ;
* сумісності з розробленими раніше версіями і програмними засобами;
* захисту від несанкціонованого доступу;
* вимог до пам’яті ПК;
* продуктивності та ефективності системи;
* налагодження системи;
* надійності та готовності системи;
* засобів відновлення після втрати даних або відмов;
* зручності обслуговування системи;
* документації для користувача;
* психологічних факторів, зокрема невідповідності зовнішнім специфікаціям, неправильного формулювання відповідей і повідомлень;
* зручності експлуатації системи.

Комплексне тестування проводять перед приймальними випробуваннями на завершальній стадії розроблення.

***Контрольні запитання для самоперевірки***.

1. Визначте поняття якість ПС і рівні моделі якості ПС.

2. Визначте характеристики якості ПС і їхнє призначення.

3. Які методи визначають показники якості?

4. Визначте  типи моделей надійності і їхній базис.

**Література**

1. Бабенко Л.П., Лавріщева К.М*.* Основи програмної інженерії.– Навч. посібник.–К.: Знання, 2001.– 269с.
2. И. Соммервиль. Инженерия программного обеспечения,  
    6 изд. – И.д. "Вильямс", 2002.
3. Лавріщева К.М. Програмна інженерія. Електронний підручник: http://csc.knu.ua/uk/library/books/lavrishcheva-6.pdf
4. Лавріщева К.М. Програмна інженерія. – Підручник.–К.:Академперіодика, 2008.–415с.
5. Лаврищева Е.М. Сборочное программирование. Основы индустрии программных продуктов: монография / Е.М. Лаврищева, В.Н. Грищенко. — 2-е изд., дополн. и перераб. — К.: Наук. думка, 2009. — 372 с.
6. ДСТУ 2844–1994. Программные средства ЭВМ. Обеспечение качества. Термины и определения.
7. ДСТУ 2850–1994. Программные средства ЭВМ. Обеспечение качества. Показатели и методы оценки качества программного обеспечения.
8. ДСТУ 3230–1995. Управление качеством и обеспечение качества. Термины и определения.
9. Лаврищева Е.М.Методы программирования. Теория, инженерия, практика.–Киев: Наук. думка, 2006.–451с.

***Для самостійного вивчення***: Поглибити матеріал лекції за наданою літературою. Вивчення лекційного матеріалу та додаткових джерел. Розгляд запитань і виконання завдань для самостійної роботи, запропонованих на лекції.