УДК 681.51
© М. Сенів[1], Д. Федасюк[1], Я. Чабанюк[1], В. Яковина[1], 2010

АНАЛІЗ ВИКОРИСТАННЯ МОДЕЛІ НАДІЙНОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ З ДИНАМІЧНИМ ПОКАЗНИКОМ СКЛАДНОСТІ ПРОЕКТУ ПРОТЯГОМ ЖИТТЄВОГО ЦИКЛУ

В роботі проведено дослідження основних припущень та обмежень моделі надійності програмного забезпечення з динамічним показником складності проекту, які доповнюють математичний формалізм моделі. Проведено обґрунтування та подані рекомендації стосовно застосування моделі в умовах реального циклу виробництва програмного забезпечення.

In this this paper the basic assumptions and limitations of software reliability model with dynamic index of project complexity, which supplement the mathematical formalism of the model, have been investigated. The recommendations for models application in a real software production cycle have been given.

1. ВСТУП

За останні 30 років було запропоновано багато моделей для вирішення проблеми оцінювання та прогнозування надійності ПЗ. Ці підходи базуються головним чином на історії спостереження помилок ПЗ і можуть бути класифіковані згідно з процесом дослідження помилок наступним чином [1]:

Моделі на основі часу між помилками. В цьому класі моделей предметом дослідження є час між виявленнями помилок. Найбільш поширеним припущенням є те, що час між (*i*-1)-ою та *i*-ою помилками підлягає певному виду розподілу, параметри якого залежать від кількості помилок, що залишаються в програмі протягом цього інтервалу. Оцінки параметрів отримуються на основі спостережених часових інтервалів між помилками, а оцінки надійності ПЗ (середній час до наступної помилки тощо) потім отримують на основі моделі з визначеними параметрами. Інший підхід трактує час між помилками як реалізацію стохастичного процесу і використовує відповідні моделі часових рядів для опису процесу виявлення помилок, що лежить в основі цього процесу.

Моделі на основі кількості помилок. Предметом дослідження цього класу моделей є кількість помилок у визначеному часовому інтервалі,

[1] Національний університет “Львівська політехніка”

а не час між помилками. Вважають, що кількість помилок відповідає відомому стохастичному процесу з дискретною або неперервною інтенсивністю виявлення помилок, що залежить від часу. Параметри функції інтенсивності виявлення помилок можна оцінити на основі спостережених значень кількості помилок чи з часів між помилками. Оцінка надійності ПЗ, як і в попередньому випадку, може бути отримана з відповідних рівнянь.

Моделі на основі висівання помилок. Основним підходом цього класу моделей є "висівання" відомої кількості помилок в програму, яка, як вважається, має невідому кількість власних помилок. Після цього програма тестується і підраховується виявлена кількість висіяних та власних помилок. На основі цього отримують оцінку кількості помилок в програмі до висівання, яка використовується для отримання оцінок надійності ПЗ та інших відповідних характеристик.

Моделі на основі області вхідних даних. Основним підходом є генерування набору тестових прикладів з вхідного розподілу, який, як вважається, репрезентує цільове використання програми. Через складність отримання такого розподілу, область вхідних даних розділяють на набір класів еквівалентності, кожен з яких зазвичай пов'язують з одним зі шляхів виконання програми. Оцінку надійності ПЗ отримують на основі помилок, виявлених шляхом фізичного чи символьного виконання тестових прикладів взятих з області вхідних даних.

Авторами було запропоновано модель надійності ПЗ [2], яка відноситься до класу моделей на основі кількості помилок і узагальнює пуассонів розподіл на випадок динамічного показника величини проекту на відміну від існуючих моделей [1, 3–5].

Метою цієї роботи є аналіз припущень та обмежень математичної моделі надійності ПЗ з динамічним показником величини проекту [2] та обґрунтування застосування цієї моделі на різних етапах життєвого циклу ПЗ з урахуванням основних припущень та спрощень.

2. МАТЕМАТИЧНИЙ ОПИС МОДЕЛІ ПЗ.

В запропонованій моделі [2] вважається, що час виявлення помилок розподілений за законом Пуассона. Крім того індекс величини проекту є параметром моделі та визначається на основі експериментальних даних і набуває значення з дійсного діапазону і завжди більший від нуля.

Пропонується наступний вигляд функції інтенсивності виявлення несправностей:

$$\lambda(t) = \alpha\beta^{s+1}t^{s}\exp(-\beta t).$$

де α – коефіцієнт, що визначає загальну кількість помилок в ПЗ, β – коефіцієнт, що характеризує загальну тривалість процесу виявлення помилок, s – індекс величини проекту, що узагальнює S-подібну модель [4].

Для заданої функції інтенсивності функція кумулятивної кількості несправностей має вигляд:

$$\mu(t) = \int_0^t \lambda(\tau) d\tau = \alpha\left[-\beta^s t^s e^{-\beta t} + s\Gamma_{\beta t}(s)\right],$$

де $\Gamma_z(p) = \int_0^z t^{p-1} e^{-t} dt$, $(\mathrm{Re}\, p > 0)$, – неповна гама-функція. Зауважимо, що при $s = 1$ функція інтенсивності виявлення несправностей та кумулятивна функція співпадають з виглядом відповідних функцій S-подібної моделі.

Загальна кількість помилок в ПЗ визначається кумулятивною функцією при $t \to \infty$, таким чином:

$$\mu(\infty) = \alpha s\Gamma(s),$$

де $\Gamma(s)$ – гама-функція.

Отже, аналітичний вигляд побудованої моделі дозволяє узагальнити вираз для загальної кількості помилок в системі, яка залежить від величини та складності проекту і визначається параметрами моделі. Крім того, частковий випадок при $s = 1$ (S-подібна модель) з урахуванням того, що $\Gamma(1) = 1$, дає значення $\mu(\infty) = \alpha$, що відповідає S-подібній моделі.

Проведені попередні дослідження дозволяють встановити наступні інтервали для індексу величини проекту:

при $s \in (0; 0{,}7)$ проект можна вважати невеликим,

при $s \in [0{,}7; 1{,}5)$ проект можна вважати середньої величини,

при $s \in [1{,}5; 2{,}2)$ проект можна вважати великим,

при $s \in [2{,}2; e]$ – проект є надвеликий.

Важливим прикладним аспектом моделей надійності ПЗ може стати встановлення кількісного критерію достатності процесу тестування програмного продукту, який би дозволив керівникам програмних проектів більш обгрунтовано приймати рішення про виділення ресурсів на тестування та про завершення цього етапу розробки ПЗ. Такі дослідження на основі побудованої моделі проводились авторами в роботі [6].

В цій роботі показано, що на пізніх етапах тестування ПЗ, коли корельовано помилки виявлені та усунені, а час виявлення тих помилок, що залишились відповідає пуассоновому розподілу, якісна характеристика розподілу (параметр s) вже практично не змінюється, а змінюються в основному кількісні характеристики (параметри α та β), що дозволило формалізувати критерій достатності процесу тестування ПЗ наступним чином [6]:

$$\frac{ds(t)}{dt} \to 0 .$$

Зауважимо, що інші відомі моделі надійності ПЗ на основі розподілу Пуассона не дозволяють отримати такий критерій внаслідок відсутності необхідних параметрів моделей, які б описували якісну зміну форми розподілу.

Запропонований критерій достатності процесу тестування можна використати у поєднанні із визначенням залишкової кількості помилок в програмному продукті (як і в інших моделях) і прийняти обґрунтоване рішення про розподіл ресурсів проекту зі створення програмного продукту.

Основні припущення та обмеження моделі.

Точне і несуперечливе визначення основних припущень і обмежень необхідне для розроблення будь-якої математичної моделі. Основні припущення та обмеження класів моделей надійності ПЗ наведено в [1]. При описі моделі надійності з динамічним показником величини проекту [2] слід враховувати наступне: модель належить до класу моделей на основі кількості помилок; для таких моделей в більшості випадків застосовують розподіл Пуассона з різним виглядом функції інтенсивності виявлення помилок в різних моделях [1] оскільки розподіл Пуассона широко використовується при моделюванні процесів де об'єктом вивчення є кількість появ випадкової величини на певних інтервалах часу. В такому випадку основними обмеженнями і припущеннями моделі будуть [1, 2]:

Кількість помилок, виявлених протягом кожного інтервалу часу, що не перетинаються, є незалежною;

Кількості помилок, виявлених на відповідних часових інтервалах є незалежними для будь-якої скінченої кількості часових проміжків. Кількість помилок на заданий момент часу відповідає пуассоновому процесу. Також вважається, що кумулятивна кількість помилок є обмеженою, не спадаючою функцією часу.

Це припущення означає, що помилки ПЗ є непов'язані між собою, при цьому одній помилці, виявленій під час тестування, відповідає один дефект в коді програми. Крім того, загальна кількість помилок в

програмній системі є скінченною величиною. Останнє припущення видається достатньо коректним, в той час як на рахунок першого існують різні думки [7]. Дослідження [2, 6, 7] дозволяють стверджувати, що на пізніх етапах тестування, коли з системи видалена достатня кількість дефектів це припущення є цілком справедливим, в той час як до висновків моделей цього класу, отриманих на початкових етапах тестування слід ставитись з обережністю і підтверджувати їх статистичною перевіркою гіпотези про належність експериментальних даних до розподілу Пуассона із заданим ступенем достовірності.

Виявлені помилки одразу виправляються;

В моделях, що базуються на цьому припущенні, припускається, що програмна система проходить через процес очистки, тобто в процесі тестування виявляють нові помилки. Таким чином це припущення принаймні є допустимим для багатьох ситуацій в тестуванні промислових програмних продуктів. Інколи, процес тестування продовжується без видалення виявленої помилки. В цьому випадку можна припустити, що подальший процес виявлення помилки відбувається в такий спосіб, наче ця помилка була видалена. Якщо, не зважаючи на це, помилка залишається в тій частині коду програми, яка надалі проходить тестування, це припущення буде справджуватись тільки тоді, коли ця помилка буде вилучена до такого тестування або ж при створенні нових тестових наборів, які оминають цю помилку.

При виправленні виявленої помилки не вноситься нових помилок;

Зміст цього припущення полягає в твердженні, що процес виявлення помилок, який моделюється, відповідає монотонному шаблону. Це означає, що по мірі того як помилки видаляються з системи, в ній залишається менше помилок, ніж було до того. В загальному випадку це може не відповідати дійсності, оскільки під час виправлення виявленої помилки можуть бути скоректовані інші ділянки програми з уведенням нових помилок в систему. Тому дане припущення в загальному вважається обмежуючим припущенням стосовно моделей надійності ПЗ. Єдиний спосіб задовольнити цю умову полягає ретельному контролі процесу виправлення, так щоб у процесі виправлення не було введено жодної нової помилки. Якщо ж, однак, введені в процесі виправлення помилки складають невелику частку в їх загальній кількості, то відхилення від цього припущення матиме мінімальний практичний вплив на результати використання моделі.

Інтенсивність виявлення помилок спадає з часом;

Таке припущення означає, що ПЗ покращується в статистичному сенсі в процесі тестування. Це припущення є прийнятним в більшості випадків і може бути обгрунтоване наступним чином. В процесі тестування з програми вилучаються помилки. Вони видаляються перед про-

довженням процесу тестування, або ж вони не видаляються, а тестування охоплює інші частини програми. В першому випадку подальша інтенсивність виявлення помилок зменшується в явному вигляді. В іншому випадку інтенсивність виявлення помилок (стосовно цілого програмного продукту) зменшується неявним чином, адже в подальшому тестується все менша частина коду програми.

Інтенсивність виявлення помилок пропорційна кількості дефектів, що залишились в програмі (імовірності виявлення будь-якої помилки на будь-якій ітерації є однаковими);

Це припущення означає, що кожна помилка, яка залишилась в програмі, має однакову імовірність бути виявленою в заданому інтервалі тестування між помилками. Це припущення є доволі правдоподібним, якщо тестові набори вибираються так, щоб забезпечити однакову імовірність виконання усіх частин програмного коду. Однак, якщо один зі шляхів виконання програми тестується більш ретельно ніж інші, в ньому буде виявлено більше помилок ніж в останніх. Помилки, які містяться в не реалізовуваних, або жодного разу не тестованих частинах коду програми очевидно будуть мати низьку або ж нульову імовірність виявлення.

Інтенсивність виявлення помилок є функцією кількості дефектів, що залишились в програмі;

Це припущення означає, що усі помилки, які залишились в коді програми з однаковою імовірністю виявляться при експлуатації системи, і використовується для оцінювання надійності ПЗ на основі кількості залишкових помилок. Якщо використання програми однорідне, то це припущення є однозначно коректним. Якщо ж деякі частини ПЗ будуть виконуватись з більшою імовірністю (частотою) ніж інші, це припущення не справджуватиметься. Разом з тим, надійність системи може бути перерахована з урахуванням інформації про відмінності у використанні. Іншими словами, в цьому випадку оцінювання надійності на основі шляхів використання є більш придатним, ніж на основі кількості залишкових помилок. Якщо ж, однак, такої інформації немає, єдиним прийнятним припущенням є припущення про однорідне використання. В цьому випадку отримана оцінка надійності повинна інтерпретуватись з обережністю.

В якості аргументу інтенсивності виявлення помилок використовується час;

Більшість моделей використовують час в якості аргументу для визначення змін інтенсивності виявлення помилок. Таке використання передбачає, що трудомісткість тестування пропорційна календарному чи процесорному часу. Також час в загальному достатньо легко піддається кількісному вимірюванню і більшість протоколів тестування ве-

дуться із зазначенням часових рамок. Іншою перевагою використання часу в якості аргументу функції є те, що час згладжує різниці в трудомісткості тестування. Якщо ж, однак, процес тестування не пропорційний часу, моделі все одно залишаються валідними для будь-якого іншого адекватного аргументу. Наприклад, можна використовувати кількість протестованих рядків коду, кількість протестованих функцій, кількість виконаних тестових наборів, чи нормувати затрати на процес тестування, наприклад, в людино-годинах.

Процес тестування є репрезентативною вибіркою експлуатаційного використання програми;

Це припущення є необхідним у випадку, коли оцінка надійності, побудована на основі процесу тестування, проектується на етап експлуатації ПЗ. Найбільш важливим це припущення є в моделях на основі області вхідних даних. Моделі на основі часу між помилками та кількості помилок також повинні базувати на цьому припущенні, якщо метою є визначення експлуатаційної надійності. Тестові набори зазвичай вибираються так, щоб забезпечити усі функціональні вимоги до системи. Однак конкретний користувач програмної системи може використовувати функції програми в іншій пропорції, ніж це використовувалось на етапі тестування. В цьому випадку тестування не відображатиме експлуатаційне використання програми. Якщо інформація про шаблонні схеми використання програмної системи наявна, процес тестування повинен бути модифікований для забезпечення репрезентативності профілю використання.

Слід зазначити, що аргументи, наведені в обґрунтуванні припущень моделі, не є універсальними для будь-якого процесу розробки ПЗ, оскільки процес розробки ПЗ дуже сильно відрізняється як в теорії, так і на практиці, і, крім того, залежить від середовища розробки, технологій програмування тощо. Положення і твердження, справедливі для одного середовища, можуть не справджуватись для іншого і т.д. Тому навіть коректні і обґрунтовані припущення, які справджуються, наприклад, під час тестування однієї функції чи підсистеми, можуть не бути такими при подальшому тестуванні цієї функції. Остаточне рішення про прийнятність припущень, що лежать в основі певної моделі, та застосовність моделі в даній конкретній ситуації повинно прийматись користувачем моделі. З іншого боку, для максимально коректного використання результатів і висновків моделі надійності слід якнайретельніше намагатися дотримуватись описаних припущень.

Використання моделі на різних етапах життєвого циклу ПЗ.

Розглянемо можливість застосування запропонованої в [2] моделі надійності ПЗ на різних етапах життєвого циклу ПЗ. Деякі зроблені вище загальні зауваження, зокрема стосовно важливості та правильної

інтерпретації припущень, що лежать в основі моделі, залишаються в силі і при розгляді цього питання. Зокрема, слід зазначити, що точне визначення припущень необхідне для моделювання навіть якщо процес розробки суттєво відрізняється від моделі. Адже будь-яка модель - це спроба описати складний реальний процес з метою його розуміння і управління, нехтуючи несуттєвими для поставленої задачі властивостями системи. Для практичних потреб модель надійності ПЗ, таким чином, повинна бути достатньо простою і не може описувати в деталях кожну особливість процесу поведінки помилок в системі. Реалізація накладених обмежень в математичній моделі допомагає у виборі математичної моделі, найбільш адекватної заданому етапу життєвого циклу ПЗ, або у виборі етапів життєвого циклу, на який дана модель є найбільш адекватною. Далі розглянемо описані вище класи моделей надійності ПЗ стосовно їх використання на етапах проектування, модульного тестування, інтеграційного тестування та на етапі експлуатації і визначимо етапи життєвого циклу, на яких побудована в [2] модель є найбільш адекватною з метою розроблення методу використання цієї моделі протягом життєвого циклу ПЗ.

Етап проектування.

На етапі проектування дефекти можуть бути виявлені візуально чи за допомогою інших формальних чи неформальних процедур. Існуючі моделі надійності ПЗ непридатні для використання на цьому етапі оскільки для виявленні дефектів потрібні тестові набори (які ще не існують) у випадку моделей на основі висівання помилок чи на основі області вихідних даних, а у випадку моделей на основі часу між помилками чи кількості помилок відсутня історія виявлення помилок [1]. Таким чином усі існуючі моделі надійності ПЗ можуть бути застосовані не раніше етапу тестування програмного продукту.

Етап модульного тестування.

Основною особливістю модульного тестування є те, що тестові набори, що генеруються для вхідних даних модуля не є репрезентативними по відношенню до експлуатаційного використання системи в цілому. Крім того, час між виявленими помилками модуля може бути не випадковим, оскільки використана стратегія тестування може не включати випадкового тестування. В реальних проектах тестові набори зазвичай мають детермінований характер.

За таких умов для оцінювання надійності найбільш придатними видаються моделі на основі висівання помилок [1], оскільки можна з високим ступенем достовірності вважати, що вбудовані і уведені помилки мають однакову імовірність проявитись при тестуванні. Суттєвою перешкодою для використання таких моделей, однак, може бути випадок, коли на цьому етапі програміст є водночас і тестером, що

особливо характерно для невеликих програмних проектів. Використання моделей на основі області вхідних даних утруднюється тим фактом, що тестовий профіль може не відповідати експлуатаційному профілю використання програми. Завдяки цьому, такі моделі можуть бути використані, однак це найбільш імовірно не матиме практичного сенсу [1].

Моделі, залежні від часу, особливо моделі на основі часу між помилками, скоріш за все не можна використовувати на цьому етапі, оскільки припущення про незалежність часу між помилками значно порушується [1].

Однак у випадку оцінювання і прогнозування надійності модуля, а не системи в цілому і за умови ретельного відбору випадкових і однорідних тестових наборів, модель на основі кількості помилок [2] може бути використана і на цьому етапі життєвого циклу, включаючи використання критерію достатності процесу тестування (пам'ятаючи, що це стосується виключно окремого модуля програми).

Етап інтеграційного тестування.

На етапі інтеграційного тестування усі компоненти і модулі інтегруються в цілісну систему, а тестові набори слугують для перевірки коректності роботи інтегрованої системи. Тестові набори для цієї задачі можуть генеруватись випадково з множини вхідних даних, або ж створюватись детерміновано слідуючи певній стратегії тестування, при цьому останнє виглядає більш ефективним і більш широко використовується в реальній практиці розробки ПЗ. Виявлені помилки виправляються, однак є висока імовірність, що під час виправлення виявлених дефектів можуть вноситись нові.

За таких умов тестування теоретично можна застосовувати моделі на основі висівання помилок [1] оскільки на цьому етапі можна дозволити собі вносити помилки в систему. Так само можуть застосовуватись моделі на основі області вхідних даних із застосування чіткого розподілу тестових наборів [1]. Складність застосування таких моделей полягає в дуже великій кількості логічних шляхів виконання усієї програмної системи.

Якщо використовується детерміністичне тестування (наприклад аналіз граничних значень, тестування шляхів виконання тощо), використання моделей на основі часу між помилками може бути неприйнятним, оскільки порушується умова незалежності часу між помилками [1]. Моделі на основі кількості помилок можуть застосовуватись, якщо в множині тестових наборів ці набори є взаємно незалежними, навіть якщо тести в межах множини вибираються детерміновано. Це пов'язано з тим, що основним припущенням такого класу моделей є те, що

інтенсивність виявлення помилок спадає в результаті виконання множини тестових випадків, а не після кожної помилки.

Якщо здійснюється випадкове тестування згідно з передбачуваним розподілом вхідних даних, тоді можна використовувати більшість існуючих моделей надійності ПЗ. Так, моделі на основі області вхідних даних повинні використовувати розподіл тестових даних, які статистично еквівалентні розподілу експлуатаційних даних. Так само можна використовувати моделі на основі висівання помилок, оскільки (як вже зазначалось) на цьому етапі життєвого циклу можна вносити контрольні помилки, а припущення про рівну імовірність виявлення помилок не повинно значно порушуватись завдяки випадковій природі процесу генерування тестів. Моделі на основі часу між помилками та кількості помилок найбільш придатні у випадку випадкового тестування.

Таким чином можна вважати запропоновану у [2] модель надійності ПЗ завдяки її перевагам над основними відомими моделями цього класу [2, 6] найбільш придатною для використання на етапі інтеграційного тестування, при цьому слід звернути особливу увагу на використання взаємно незалежних наборів тестових даних та однорідного охоплення процесом тестування принаймні усіх модулів та компонентів програмної системи. Перевагою цієї моделі над моделями на основі висівання помилок є відсутність необхідності внесення помилок в код програми (тим більше ці помилки повинні мати рівну імовірність виявлення з неконтрольованими помилками); перевагою над моделями на основі часу між помилками є її толерантність (за дотримання описаних вище умов) до детермінованого процесу тестування, який найчастіше використовується при розробці промислових програмних продуктів; а перевагою порівняно з моделями на основі області вхідних даних є менш строга вимога до передбачення експлуатаційного виконання програми, яка може бути замінена вимогою до тестування усіх модулів програми замість усіх логічних шляхів виконання програми.

Фаза приймального тестування.

Протягом фази приймального тестування створюються набори вхідних даних на основі експлуатаційного використання продукту для перевірки придатності продукту до уведення в експлуатацію та оцінювання його надійності. У цій фазі висівання помилок є неприйнятним з практичної точки зору, а виявлені помилки не завжди одразу виправляються. Таким чином, моделі на основі висівання помилок та часу між помилками є непридатними для оцінювання надійності у цій фазі життєвого циклу [1]. Інші міркування подібні до наведених у частині, присвяченій інтеграційному тестуванню, тому моделі на основі кількості помилок та області вхідних даних в загальному є прийнятним у цій фазі.

Таким чином запропонована модель з динамічним показником величини проекту може використовуватись і на цьому етапі життєвого циклу ПЗ, крім того, на відміну від існуючих моделей, у цій фазі суттєву підтримку в процесі прийняття рішення про уведення програмного продукту в експлуатацію надасть використання критерію достатності процесу тестування [6] разом з оцінкою кількості залишкових дефектів в програмі та імовірності їх появи чи середнього часу напрацювання на відмову (див. нижче).

Експлуатаційний етап

Коли якість ПЗ визнається достатньою програмний продукт випускається в експлуатацію. Під час експлуатації вхідні дані користувачів можуть не бути випадковими оскільки, наприклад, користувач може використовувати тільки деякі одні й ті ж функції продукту під час використання програми. Вхідні дані також можуть бути корельованими (наприклад в системах реального часу) втрачаючи таким чином свою випадковість і однорідність розподілу. Крім того, помилки зазвичай не виправляються одразу після виявлення. За таких умов видається, що моделі на основі кількості помилок є єдиними придатними для відстеження інтенсивності виявлення помилок чи для визначення оптимального часу для встановлення нової версії продукту [1].

Таким чином, розглянуті особливості використання моделей надійності ПЗ на різних етапах життєвого циклу дозволяють зробити висновок, що побудована авторами модель [2] володіє рядом переваг порівняно з існуючими моделями і може бути застосована практично на усіх етапах (крім аналізу вимог та проектування) життєвого циклу ПЗ навіть у випадках коли порушуються деякі припущення моделі.

Опис і характеристика основних показників надійності ПЗ.

Надійність ПЗ визначається його безвідмовністю і відновлюваністю. Безвідмовність ПЗ – це властивість зберігати працездатність при використанні його для обробки інформації в інформаційних системах. Безвідмовністю програмного забезпечення оцінюється ймовірність його роботи без відмов при визначених умовах зовнішнього середовища протягом заданого періоду спостереження [9].

У наведеному визначенні під відмовою ПЗ розуміється неприпустиме відхилення характеристик функціонування цього забезпечення від висунутих вимог. Визначені умови зовнішнього середовища це сукупність вхідних даних і стан самої інформаційної системи. Заданий період спостереження відповідає часу, необхідному для виконання на ЕОМ задачі, що розв’язується.

Безвідмовність ПЗ може характеризуватися середнім часом виникнення відмов при функціонуванні програми. При цьому припускається, що апаратні засоби ЕОМ знаходяться в справному стані. З точки зору надійності, принципова відмінність ПЗ від апаратних засобів полягає в тому, що програми не зношуються і їхній вихід з ладу через поломку неможливий. Отже, характеристики функціонування ПЗ залежать тільки від його якості, зумовленої процесом розробки. Це означає, що безвідмовність ПЗ визначається його коректністю і залежить від наявності в ньому помилок, внесених на етапі його створення. Крім того, прояв помилок ПЗ пов'язаний ще і з тим, що в деякі моменти часу на обробку можуть надходити множини даних, які раніше не зустрічалися і які програма не в змозі коректно обробити. Тому вхідні дані у деякій мірі впливають на функціонування ПЗ [8].

Виходячи з не зовсім чітких, з математичної точки зору, визначень надійності та безвідмовності ПЗ виникає потреба у визначенні математичних основ та критеріїв надійності, зокрема кількісних.

Критерієм називається ознака (міра), за якою оцінюється надійність. Показником надійності називається чисельне значення критерію [9].

Відмова елемента являється випадковою подією, а час ξ до її появи – випадковою величиною. Основною характеристикою надійності елементу будь-якої системи, в тому числі невідновлюваної, є функція розподілу тривалості його безвідмовної роботи $F(t) = P\{\xi < t\}$, визначена при $t \geq 0$. На її основі, в загальному, можуть бути отримані наступні показники надійності [9]:

$P(t)$ – ймовірність безвідмовної роботи за час t;

$Q(t) = 1 - P(t)$ – ймовірність відмови за час t;

m_t – середній час безвідмовної роботи (середній час напрацювання до відмови);

$f(t)$ – щільність розподілу часу безвідмовної роботи;

$\lambda(t)$ – інтенсивність відмов в момент часу t.

Надійність відновлюваних систем, якими являються програмні системи оцінюють наступними показниками [9]:

T_0 – середній час роботи між відмовами (середнє напрацювання на відмову);

T_B – середній час відновлення;

$\omega(t)$ – параметр потоку відмов;

$K_Г(t)$ – функція готовності – ймовірність того, що система справна в момент t;

$K_П(t)$ – функція простою – ймовірність того, що в момент t система несправна і відновлюється;

$K_Г$ – коефіцієнт готовності – ймовірність того, що система буде справною при тривалій експлуатації (стаціонарний режим);

$K_П$ – коефіцієнт простою – ймовірність того, що система буде несправною при тривалій експлуатації.

1. Обчислення імовірності безвідмовної роботи за час t.

Позначимо через T час безвідмовної роботи продукту (інтервал часу від початку роботи продукту до першої відмови). T – випадкова величина (або напрацювання на відмову), t – можливі значення випадкової величини T.

$P(t) = P\{T \geq t\}$ – імовірність того, що час безвідмовної роботи продукту буде більшим або рівним деякого значення t [9, 10]. Оскільки прогнозування часу безвідмовної роботи починається з деякого моменту часу τ, який відповідає моменту закінчення тестування (нагадаємо, що модель надійності передбачає, що усі помилки, виявлені до моменту часу τ, були виправлені без уведення нових помилок, а закон розподілу появи помилок не змінюється і відповідає пуассоновому процесу), то імовірність безвідмовної роботи описується виразом

$$P(t) = \exp\left(-\int_{\tau}^{\infty} \hat{\lambda}(t)dt\right), \ (t > \tau). \tag{1}$$

В свою чергу імовірність відмови $q(t)$ – це імовірність того, що час безвідмовної роботи менший, ніж деяке значення t, тобто це імовірність того, що протягом заданого часу виникне хоча б одна відмова:

$$q(t) = P\{T < t\} = 1 - P(t). \tag{2}$$

2. Обчислення середнього часу безвідмовної роботи.

Середній час безвідмовної роботи (середнє напрацювання на відмову) – це математичне сподівання випадкової величини T [9].

$$m_t = E[T] = \int_{\tau}^{\infty} t \cdot f(t)dt, \tag{3}$$

де E – знак математичного сподівання, $f(t)$ – частота відмов (густина імовірності випадкової величини T) [10],

$$f(t) = -dP(t)/dt = \hat{\lambda}(t) \cdot P(t).$$

3. Обчислення дисперсії часу безвідмовної роботи.

До числових характеристик опису безвідмовної роботи системи відноситься і дисперсія часу безвідмовної роботи:

$$D_t = E[T - m_t]^2 = E(T^2) - m_t^2,$$

або

$$D_t = \int_{\tau}^{\infty} t^2 \cdot f(t) dt - m_t^2. \quad (4)$$

4. Обчислення функції готовності.

Функція готовності $K_Г(t)$ – це імовірність перебування системи в стані працездатності, тобто імовірність, що за час t не з'явиться жодної помилки:

$$K_Г(t) = P\{\xi = 0\}. \quad (5)$$

Модель [2], як і більшість моделей цього класу базується на припущенні, що кількість помилок m_i є незалежними пуассоновими величинами з математичним сподіванням $\mu(t_i) - \mu(t_{i-1})$, то для оцінки імовірності появи m_{k+2} помилок за час $t_{k+2} - t_{k+1}$ використаємо розподіл Пуассона у вигляді

$$P\{\xi = m_{k+2}\} = \frac{[\hat{\mu}(t_{k+2}) - \hat{\mu}(t_{k+1})]^{m_{k+2}}}{(m_{k+2})!} \cdot \exp[\hat{\mu}(t_{k+1}) - \hat{\mu}(t_{k+2})]. \quad (6)$$

Таким чином з рівнянь (5) та (6) отримуємо:

$$K_Г(t) = P\{\xi = 0\} = \frac{[\hat{\mu}(\infty) - \hat{\mu}(t)]^0}{0!} \cdot e^{-\hat{\mu}(\infty)+\hat{\mu}(t)} = e^{-\hat{\mu}(\infty)+\hat{\mu}(t)}. \quad (7)$$

Оскільки модель надійності ПЗ [2] відноситься до класу моделей на основі кількості помилок і розглядає кількість помилок на часових інтервалах як негомогенний пуассонів процес, процеси виправлення помилок та відновлення після збоїв у розгляді цієї моделі нехтуються, а тривалість перебування в стані відновлення вважається нехтувально малою. Тому такі кількісні показники надійності як функція простою, коефіцієнт готовності, коефіцієнт простою, середній час відновлення тощо неможливо обчислити виходячи з параметрів моделі, і в цій роботі вони не розглядаються. Для коректного визначення таких показників надійності можна, наприклад, розглядати поведінку ПЗ як марківський процес [11, 12]. Такий опис поведінки надійності ПЗ є предметом подальших досліджень.

3. ВИСНОВКИ

В роботі проведено аналіз основних припущень та обмежень моделі надійності ПЗ з динамічним показником величини проекту. Показані шляхи забезпечення коректності використання моделі надійності у випадку, коли деякі з припущень не справджуються в процесі розробки програмного продукту.

Детально описано використання моделі на різних етапах життєвого циклу програмного забезпечення. Розглянуті особливості використання моделей надійності ПЗ на різних етапах життєвого циклу дозволяють зробити висновок, що модель з динамічним показником складності проекту володіє рядом переваг порівняно з існуючими моделями і може бути застосована практично на усіх етапах (крім аналізу вимог та проектування) життєвого циклу ПЗ навіть у випадках коли порушуються деякі припущення моделі.

Показано, що цю модель завдяки її перевагам над основними відомими моделями такого класу найбільш придатною для використання на етапі інтеграційного тестування, при цьому слід звернути особливу увагу на використання взаємно незалежних наборів тестових даних та однорідного охоплення процесом тестування принаймні усіх модулів та компонентів програмної системи.

Перевагою цієї моделі над моделями на основі висівання помилок є відсутність необхідності внесення помилок в код програми; перевагою над моделями на основі часу між помилками є її толерантність до детермінованого процесу тестування, який найчастіше використовується при розробці промислових програмних продуктів; а перевагою порівняно з моделями на основі області вхідних даних є менш строга вимога до передбачення експлуатаційного виконання програми, яка може бути замінена вимогою до тестування усіх модулів програми замість усіх логічних шляхів виконання програми.

Розроблено процедуру обчислення кількісних показників надійності з урахуванням основних припущень моделі.

1. Goel A.L. Software reliability models: assumptions, limitations, and applicability // IEEE Transactions on software engineering. 1985, Vol. SE-11, No 12, pp. 1411-1423. 2. Чабанюк Я.М., Яковина В.С., Федасюк Д.В., Сенів М.М., Хімка У.Т. Побудова і дослідження моделі надійності програмного забезпечення з індексом величини проекту // Інженерія програмного забезпечення, 2010. (в друці). 3. Shooman M.L. Probabilistic models for software reliability prediction // in Statistical Computer Performance Evaluation. – W. Freiberger, Ed. – New York: Academic. – 1972. – P. 485–502. 4. Yamada S., Ohba M., Osaki S. S-shaped reliability growth modeling for software error detection // IEEE Transactions on Reliability. – Vol. R-32. – No.5. – 1983. – P. 475–478. 5. Тимошенко Ю.О., Дідковська М.В. Узагальнена модель негомогенного пуасонівського процесу для оцінювання надійності програмного забезпечення // Проблеми програмування.

– № 2–3. – 2004. – С.480–489. 6. Яковина В.С., Сенів М.М., Чабанюк Я.М., Федасюк Д.В., Хімка У.Т. Критерій достатності процесу тестування програмного забезпечення // Вісник Національного університету "Львівська політехніка" Комп'ютерні науки та інформаційні технології, 2010. (в друці). 7. Cai K.-Y., Hu D.-B., Bai Ch.-G., Hu H., Jing T. Does software reliability growth behavior follow a non-homogeneous Poisson process // Information and Software Technology. – Vol. 50. – 2008. – P. 1232–1247. 8. Ермаков А.А. Основы надежности информационных систем: учебное пособие. – Иркутск: ИрГУПС, 2006. – 151 с. 9. Половко А.М., Гуров С.В. Основы теории надежности. – СПб.: БХВ-Петербург, 2006. – 704 с. 10. Липатов И.Н. Надежность функционирования автоматизированных систем. Конспект лекций. – Пермь: Изд-во Пермского государственного технического университета, 1996. – 66 с. 11. Durand J.B., Gaudoin O. Software reliability modelling and prediction with hidden Markov chains // Statistical Modelling. – Vol. 5 (1). – 2005. – P. 75–93. 12. Волочій Б.Ю. Технологія моделювання алгоритмів поведінки інформаційних систем. – Львів: Вид-во НУ "Львівська політехніка", 2004. – 220 с.