УДК 004.052

В. С. ЯКОВИНА

Національний університет "Львівська політехніка", Україна

МЕТОД АНАЛІЗУ НАДІЙНОСТІ ПРОГРАМНИХ ЗАСОБІВ З УРАХУВАННЯМ ЇХ СКЛАДНОСТІ

Розглядається метод аналізу надійності програмних засобів з урахуванням їх складності на різних етапах життєвого циклу. Метод включає використання моделей надійності як на основі неоднорідного пуассонового процесу, так і марковського процесу вищого порядку з дискретним та неперервним часом. Використання запропонованого методу дає можливість розробникам програмного забезпечення обрати відповідну модель надійності, виходячи з особливостей процесу розробки та складності програмних засобів, і оцінити значення показників надійності досліджуваного програмного засобу. Запропонований метод аналізу надійності програмних засобів включає настанови та рекомендації щодо отримання вхідних параметрів відповідних моделей надійності.

Ключові слова: *програмний засіб, надійність, складність, компонентний підхід, модель надійності, неоднорідний пуассонів процес, ланцюг Маркова вищого порядку.*

Вступ

Актуальною прикладною проблемою сучасності було і залишається питання забезпечення надійності техніки. Розвиток сучасних інформаційних технологій та використання програмно-керованих комплексів та систем в усіх сферах людського життя приводить до значного зростання ризиків відмов такої техніки. При цьому, внаслідок специфічної природи програмних засобів (ПЗ) та особливостей їх поведінки в сенсі надійності питання аналізу та забезпечення їх надійності все ще залишає широке поле для досліджень.

На даний час для вирішення задачі аналізу та оцінювання показників надійності програмних засобів використовують цілий ряд моделей надійності, які можна розділити на два великих класи – детерміністичні та імовірнісні. Перший клас розглядає відмови програмних засобів як детермінований процес, до цього класу відносять як статичні моделі на основі метрик коду ПЗ [1, 2], так і методи на основі динаміки програмних систем [3]. Натомість моделі імовірнісного класу представляють появу відмов та усунення помилок програмних засобів як випадкові події [1, 4]. Розвиток сучасних інформаційно-комп'ютерних технологій і технологій створення програмних продуктів зокрема, привів до значного підвищення складності програмних систем, більшість з яких створюється на основі компонентного підходу, що приводить до виникнення протиріччя між складністю сучасних програмних засобів та моделями і методами аналізу їх надійності. Для вирішення цього протиріччя за останні двадцять років набули значного поширення моделі і методи аналізу надійності програмних засобів на основі компонентного підходу [5, 6].

Крім того, внаслідок наявності великої кількості моделей надійності програмних засобів та значних відмінностей в їх характеристиках, метриках, області використання тощо, виникає важлива прикладна проблема вибору моделі надійності, яка найбільш адекватно описує властивості і поведінку конкретного програмного засобу. Одним зі шляхів вирішення цієї проблеми є побудова методики вибору моделей надійності програмних засобів на основі аналізу припущень, характерних для процесу розробки, тестування і експлуатації програмних засобів, а також особливостей самого програмного засобу як продукту проектування [7, 8].

В даній роботі запропоновано метод аналізу надійності програмних засобів, який дає можливість вибрати модель надійності в залежності від складності програмного засобу та оцінити його показники надійності, а також описує практичні аспекти використання цих моделей на різних етапах життєвого циклу програмного забезпечення.

Постановка задачі дослідження

У попередніх роботах автором зі співавторами було запропоновано ряд моделей надійності програмних засобів, які можуть бути використані для ПЗ різної складності та у випадку різних припущень про структуру і поведінку програмних засобів.

Модель надійності ПЗ на основі неоднорідного пуассонового процесу з показником складності [9] передбачає наступний вигляд параметра потоку від-

мов ПЗ:

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

де α – коефіцієнт, який характеризує загальну кількість відмов, які були виявлені в програмі від початку спостереження;

β – коефіцієнт, що характеризує швидкість зміни функції параметру потоку відмов (β>0);

s – показник складності ПЗ [10].

В такому випадку прогнозована загальна кількість помилок в ПЗ може бути обчислена як [9]:

$$\mu(\infty) = \alpha s\Gamma(s), \quad (2)$$

де Γ(s) – гама-функція.

З метою аналізу надійності програмних засобів, розроблених на основі компонентного підходу, було побудовано декілька архітектурних моделей надійності ПЗ, що базуються на математичному апараті марковських процесів вищого порядку [11, 12].

Зокрема у випадку моделі з дискретним часом [11] ймовірність безвідмовної роботи цілої системи визначається як:

$$R(t) = \prod_{i=1}^{N} R_i(t), \quad (3)$$

де $R_i(t)$ – ймовірність безвідмовної роботи кожного модуля (i=1…N), яка обчислюється з використанням формалізму марковських процесів вищого порядку як

$$R_l = \exp\left(-\int_0^{\sum_{ij..k} V_{ij..kl}t_{ij..kl}} \lambda_l(t)dt\right). \quad (4)$$

Тут $V_{ij\ldots kl}$ – очікувана кількість відвідувань модуля l в залежності від виконання попередніх K модулів;

$t_{ij\ldots kl}$ – час виконання модуля l у залежності від виконання попередніх K модулів.

Для знаходження $V_{ij\ldots kl}$ потрібно розв'язати таку систему лінійних алгебраїчних рівнянь:

$$V_{j..kl} = b_{ij..k} + \sum_{i=1}^{N-1} V_{ij..k}p_{ij..kl}, \quad (5)$$

де $p_{ij\ldots kl}$ – ймовірність переходу в модуль l в залежності від виконання попередніх K модулів;

$b_{ij\ldots kl}$ – початковий ймовірнісний вектор.

Якщо потік управління між модулями системи моделюється як процес Маркова з неперервним часом (стан C_i процесу відповідає виконанню i-го програмного модуля), то інтенсивність відмов програмної системи, яка складається з N модулів, може бути записана як [11]:

$$\lambda(t) = \sum_{i=1}^{N} p_i(t)\lambda_i(t), \quad (6)$$

де $\lambda_i(t)$ – інтенсивність відмов i-ого модуля;

$p_i(t)$ – імовірність виконання i-ого модуля в момент часу t.

У випадку недосконалої інтеграції модулів в систему (коли можлива відмова при передачі управління між модулями програмного засобу) інтенсивність відмов системи запишеться як [12]:

$$\lambda(t) = \sum_{i=1}^{N} p_i(t)\lambda_i + \sum_{i\neq j} a_{ij}(t)q_{ij}, \quad (7)$$

де $a_{ij}(t)$ – інтенсивність переходу зі стану i в стан j;

q_{ij} – імовірність виникнення відмови при передачі управління між модулями i та j.

Для вирішення поставленої задачі вибору моделі надійності для заданого програмного засобу та оцінювання показників його надійності використовується наведений нижче метод аналізу надійності програмних засобів.

Метод аналізу надійності програмних засобів з урахуванням їх складності

Схема методу аналізу надійності ПЗ на різних етапах життєвого циклу з урахуванням його складності наведена на рис. 1. Наведемо опис методу, зображеного на цих рисунках, у відповідності з основними етапами життєвого циклу програмного забезпечення.

1. Етап аналізу вимог до ПЗ

Першим кроком узагальненого методу аналізу надійності ПЗ є отримання інформації про модель поведінки програмної системи, що включає оцінки імовірностей виконання різних сценарії використання ПЗ, на ранніх етапах життєвого циклу.

Одним із перших етапів життєвого циклу розробки ПЗ є збір та аналіз вимог. Так як стандартною практикою аналізу вимог до ПЗ є використання мови UML, а саме діаграми випадків використання, доцільним є використати цей інженерний засіб для аналізу надійності ПЗ, зокрема для наближеного отримання значень матриці ймовірностей переходів між модулями [13].

Якщо q_i – ймовірність використання системи користувачем u_i, P_{ij} – ймовірність, що користувач u_i використовує функціональну можливість f_j, то ймовірність виконання випадку використання x обчислюється наступним чином: $P_{ij}(x) = \sum_{i=1}^{m} q_i P_{ix}$, де m – кількість типів користувачів (акторів). На основі ймовірностей (або відносних частот) виконання усіх випадків використання можна оцінити ймовірності виконання модулів ПЗ та переходів між ними. Необхідно пам'ятати, що на цьому етапі життєвого циклу можна тільки наближено говорити про модулі ПЗ, а згадані імовірності отримують апріорно на основі експертних оцінок. Підхід, описаний у [14],

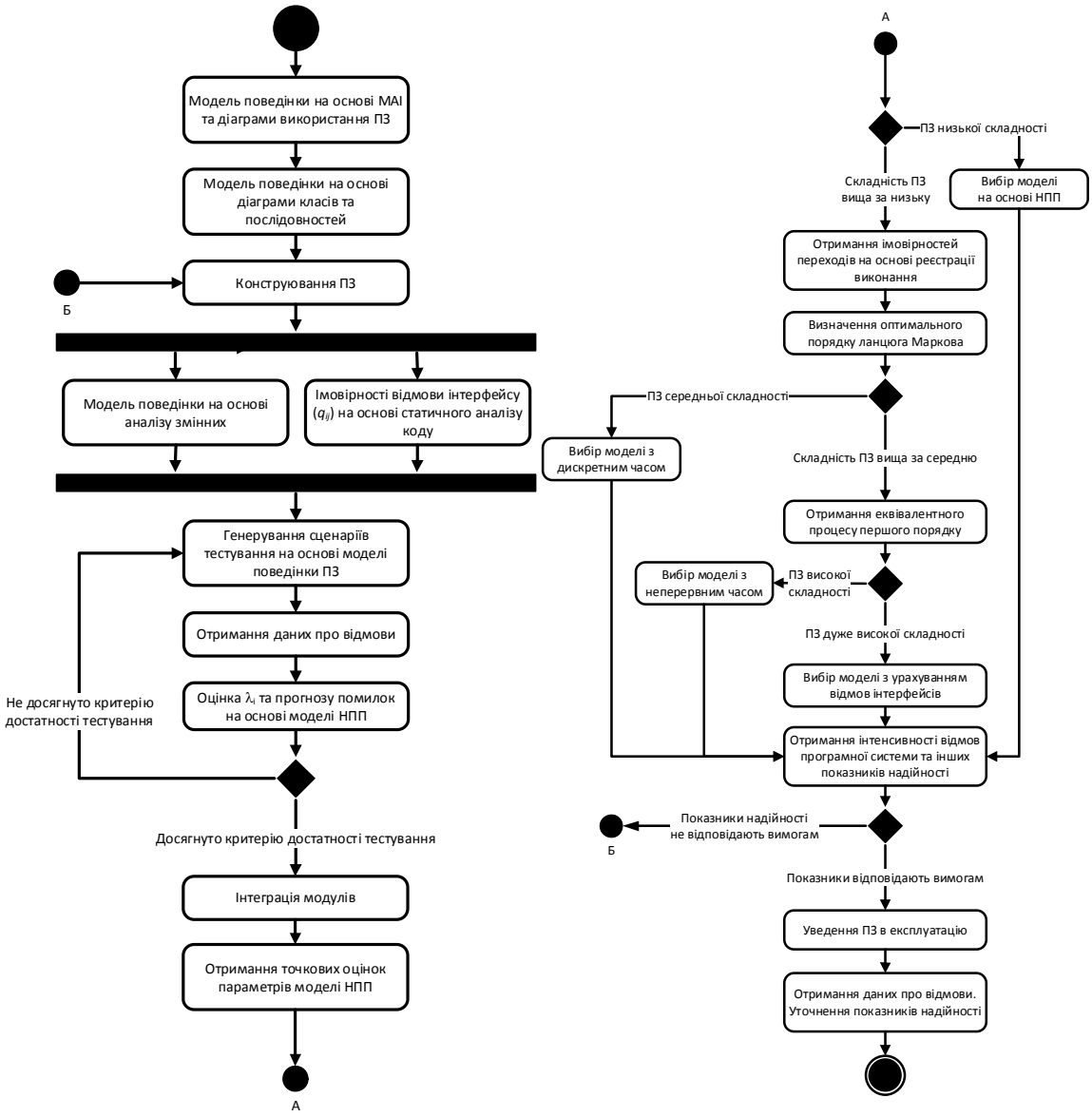

Рис. 1. Схема методу аналізу надійності ПЗ

потребує знання оцінок ймовірностей використання системи кожним типом акторів, а також ймовірностей виконання кожного випадку використання типами користувачів. Проте, на цьому етапі життєвого циклу розробник не має необхідних даних, а для замовника, який є основним експертом з випадків використання системи, задача оцінювання таких ймовірностей виявляється занадто складною. Для усунення даного обмеження в [13] запропоновано використати метод аналізу ієрархій (MAI), який передбачає декомпозицію задачі та використання експертом попарних порівнянь.

Відповідно до [13] опитування замовника (експерта) здійснюється у вигляді анкетування за допомогою попарного порівняння згідно з шкалою відносної важливості методу аналізу ієрархій. На основі анкетування заповнюється матриця попарних порівнянь та здійснюється розрахунок ймовірностей виконання різних випадків використання. Для спрощення опитування замовника можливе використання модифікації MAI, яка базується на порівнянні за стандартами. Використання подібної системи пріоритетів є поширеною практикою в індустрії розробки ПЗ при побудові специфікацій програмних продуктів під час роботи з замовником.

2. Етап проектування архітектури ПЗ

Другим кроком методу аналізу надійності (див. рис. 1) є отримання прогнозованого графу станів системи, який відображає виконання модулів та передачу управління між ними, а також оцінок матриці імовірності переходів між станами на етапі проектування ПЗ. Оскільки етап проектування належить до ранніх етапів життєвого циклу, на якому ще немає коду програми та, відповідно, інформації про відмови, в [13] описано підхід з використанням стандартного засобу програмної інженерії – універса-

льної мови моделювання UML.

При розробці архітектури об'єктно-орієнтованих систем використовується UML діаграма класів. За допомогою діаграми класів здійснюється опис модулів системи, а також зв'язків між ними. З діаграми класів можна сформувати структуру матриці ймовірностей переходів між модулями, проте у цьому випадку неможливо отримати ймовірності переходів та час виконання модулів, оскільки така діаграма не відображає поведінку системи.

Для проектування модулів системи та взаємодії між ними використовується UML діаграма послідовності дій. В цьому методі діаграма послідовності дій використовується для оцінки кількості переходів між модулями, а також часу виконання модуля [13]. Провівши аналіз UML діаграм послідовності під час проектування ПЗ необхідно порахувати сумарну кількість переходів з модуля в модуль і сформувати матрицю переходів між модулями. Для визначення часу виконання кожного модуля ПЗ необхідно просумувати часи виконання модулів. Для того щоб отримати матрицю ймовірностей переходів усієї системи, необхідно подібним чином урахувати всі сценарії використання системи, оскільки діаграма послідовності зазвичай будується для кожного сценарію використання системи, а імовірність виконання кожного сценарію отримують на першому кроці даного методу.

3. Етап конструювання ПЗ

На цьому етапі, під час якого створюється працездатне програмне забезпечення з використанням методів верифікації, кодування та тестування модулів, розроблений метод аналізу надійності дає можливість отримати прогнозований перелік лімітуючих (в сенсі надійності) модулів системи; оцінити імовірності відмови інтерфейсів ПЗ на основі статичного аналізу коду та його метрик [12] та побудувати модель поведінки ПЗ на основі аналізу його змінних програмного коду [15].

На початку етапу конструювання програмного продукту можна скористатись виразом (6) та отриманими на попередніх кроках методу оцінками імовірностей виконання модулів ПЗ для моделювання впливу інтенсивності відмов модулів на інтенсивність відмов програмної системи. При цьому можна отримати верхні оцінки інтенсивності відмов модулів та визначити модулі, внесок яких в загальну надійність системи буде найвагомішим, з метою підвищеної уваги до таких модулів під час написання їх програмного коду та визначення стратегії їх тестування. На цьому кроці методу аналізу надійності недоцільно використовувати математичний апарат ланцюгів Маркова вищих порядків внаслідок постійних змін в програмному продукті та відсутності уточнених значень параметрів моделі надійності.

Модель поведінки ПЗ – це модель програмної системи, що відображає її процес виконання та деякий набір характеристик. Коректна модель поведінки ПЗ з урахуванням необхідних характеристик програм дає можливість точніше здійснювати автоматизоване формування сценаріїв тестування, а також оцінювати надійність ПЗ моделями, побудованими на основі архітектурного підходу. У [15] описано нову математичну модель функціонування програмного продукту у вигляді орієнтованого графу G={C, P}, де C – множина модулів ПЗ, P – множина переходів між відповідними модулями. Кожен вузол графа C_i є набором множин змінних та відповідних класів еквівалентності, які використовуються та змінюються у модулі, а також змінних та відповідних класів еквівалентності, що можуть викликати відмови у модулі C_i, а також характеризується списком інцидентних дуг, що в свою чергу містять ймовірності передавання контролю p_{ij} до іншого вузла C_j (i, j = 1…N).

4. Фаза модульного тестування

Під час цієї фази життєвого циклу програмного забезпечення здійснюється тестування розроблених модулів (переважно розробниками) без інтеграції їх в систему та виправлення виявлених помилок. Слід зауважити, що розроблений узагальнений метод аналізу надійності програмного забезпечення передбачає проведення усіх видів тестування [16] на основі моделі поведінки, що базується на змінних коду [15].

Після генерування тестових сценаріїв відповідно до стратегії тестування на основі моделі поведінки, отримуються дані про відмови програмного продукту. З отриманих даних про відмови під час процесу відлагодження ПЗ (який характеризується зміною з часом кількості помилок в продукті, а відповідно залежністю інтенсивності відмов від часу) з використанням моделі на основі неоднорідного пуассонового процесу (1) отримують інтенсивність відмов модуля та прогнозовану кількість помилок згідно (2). Тривалість процесу модульного тестування визначають на основі критерію достатності процесу тестування [17] або ж удосконаленої процедури оцінки кількості помилок в програмному продукті [18]. Як уже зазначалось якість процесу тестування забезпечується використанням методу автоматизованого генерування тестових сценаріїв із забезпеченням метрики покриття змінних коду [16]. Після прийняття рішення про припинення тестування і завершення поточної ітерації розробки програмного модуля переходять до наступного кроку методу аналізу надійності.

5. Фаза інтеграційного тестування

На цьому кроці відбувається інтеграція модулів в систему, проводиться тестування системи в цілому

з метою виявлення (і усунення) помилок взаємодії модулів, що передбачає виконання складних тестових сценаріїв, які стосуються декількох модулів системи. Для проведення такого тестування особливо ефективним є використання засобів автоматизованого генерування тестових сценаріїв [16]. Водночас, результати інтеграційного тестування дають інформацію про відмови програмної системи в цілому, яка може бути використана для оцінювання її надійності або для засобів підтримки прийняття рішень при виробництві ПЗ.

Після отримання даних про відмови програмної системи та виправлення виявлених помилок інтеграції, метод (див. рис. 1) передбачає використання узагальненої моделі надійності на основі пуассонового процесу з показником складності [9], як першого наближення при аналізі надійності програмного засобу. На цьому етапі для уточненої оцінки кількості помилок в програмному продукті метод аналізу надійності передбачає використання удосконаленої процедури оцінки кількості помилок [18]. Після отримання значень параметрів пуассонового процесу (1) здійснюють оцінювання складності програмного продукту згідно [10]. При цьому, якщо програмна система відноситься до класу нескладних ($s<0,66$) рекомендується використовувати для аналізу надійності системи модель на основі неоднорідного пуассонового процесу і перейти до оцінювання показників надійності програмної системи. В іншому випадку слід використати одну з моделей на основі ланцюга Маркова вищого порядку, описаних вище в цьому розділі.

Для отримання значень імовірностей передачі потоку управління між модулями системи, можна використати моніторинг виконання програми [19] чи формалізм прихованих марковських моделей. Важливо пам'ятати, що при отриманні імовірностей переходів системи між станами шляхом моніторингу виконання програми (яке здійснюється не в умовах реальної експлуатації, а під час тестування розробниками), сценарії виконання програми повинні бути максимально наближені до сценаріїв її подальшої експлуатації – наприклад до отриманих на першому кроці цього методу на основі методу аналізу ієрархій за результатами експертизи замовника.

Після отримання необхідних значень параметрів марковських моделей (матриці імовірностей переходів між станами, початкового вектору, інтенсивності відмов модулів тощо) наступним кроком методу аналізу надійності (див. рис. 1) на основі одного з інформаційних критеріїв визначають оптимальний порядок марковського процесу. Після цього, якщо програмна система відноситься до систем середнього ступеня складності (значення показника складності моделі пуассонового процесу $0,66\leq s<1,6$, оптимальний порядок процесу не дуже високий (слід зауважити, що питання взаємозв'язку оптимального порядку ланцюга Маркова в моделі надійності зі складністю програмного продукту потребує подальших досліджень), а імовірності відмови інтерфейсів системи, які оцінено на третьому кроці цього методу, є нехтувально малими) рекомендується використовувати для аналізу надійності системи модель на основі ланцюга Маркова вищого порядку з дискретним часом (див. вирази (3)–(5)) і перейти до оцінювання показників надійності програмної системи. Використання моделі з дискретним часом є доцільним внаслідок невеликих обчислювальних витрат на роботу з цією моделлю та незначного зменшення ступеня адекватності при роботі з програмними продуктами середнього ступеня складності. В іншому випадку слід використати одну з моделей на основі ланцюга Маркова вищого порядку з неперервним часом.

При використанні моделей з неперервним часом наступним кроком методу (див. рис. 1) є отримання еквівалентного процесу першого порядку (див. [11, 12]) для побудови та розв'язку системи рівнянь Колмогорова–Чепмена. Розв'язки цієї системи рівнянь дають значення імовірностей перебування системи в станах (часові залежності імовірностей виконання модулів системи), що разом з отриманими під час модульного тестування значеннями інтенсивності відмов модулів, на основі виразу (6), дає можливість отримати функцію інтенсивності відмов програмної системи для програмного продукту високої складності (значення показника складності моделі пуассонового процесу $1,6\leq s<2,28$, а імовірностями відмови інтерфейсів системи можна знехтувати). В такому випадку переходять до оцінювання показників надійності програмної системи (див. рис. 1).

Якщо ж програмна система є дуже високою (значення показника складності моделі пуассонового процесу $s>2,28$, а імовірностями відмови інтерфейсів системи не можна знехтувати), то використовують модель надійності ПЗ (7) та переходять до оцінювання показників надійності програмної системи.

Таким чином узагальнений метод аналізу надійності програмного забезпечення з урахуванням його складності впродовж різних етапів життєвого циклу, зображений на рис. 1, дає можливість отримати функцію інтенсивності відмов програмної системи на основі виразів (1), (3), (6), (7) залежно від її складності. Після отримання часової залежності інтенсивності відмов системи, інші показники надійності можуть бути розраховані згідно відомих із загальної теорії надійності виразів [20]. На основі отриманих значень показників надійності прийма-

ється рішення про введення цієї версії програмного засобу в експлуатацію, чи продовження тестування та відлагодження (а в окремих випадках і повернення до наступної ітерації розробки модулів системи).

6. Етап експлуатації ПЗ

Після уведення програмної системи в експлуатацію продовжується збір статистики використання системи та даних про її відмови з використанням сучасних засобів та інструментів інженерії програмного забезпечення (див. напр. [21]). На основі отриманих даних, згідно обраної на етапі розробки ПЗ моделі надійності, здійснюється уточнення показників надійності та експлуатаційних показників програмної чи програмно-апаратної системи, в складі якої експлуатується досліджуване ПЗ. Інформація про показники надійності та сценарії використання ПЗ, разом з новими вимогами (як функціональними, так і іншими) лягає в основу процесу розробки нових версій даного програмного продукту.

Висновки

На основі побудованих і описаних в попередніх працях моделей розроблено узагальнений метод аналізу надійності програмних засобів з урахуванням їх складності впродовж різних етапів життєвого циклу. Розроблений метод включає модель надійності ПЗ на основі неоднорідного пуассонового процесу з показником складності, моделі надійності ПЗ на основі марковського процесу вищого порядку з дискретним та неперервним часом, а також модель на основі ланцюга Маркова вищого порядку з неперервним часом, яка враховує відмови інтерфейсів програмних модулів. Показано використання цих моделей та допоміжних процедур на різних етапах життєвого циклу ПЗ в залежності від складності програмного засобу, надійність якого аналізується. Узагальнений метод аналізу надійності програмних засобів з урахуванням їх складності впродовж різних етапів життєвого циклу дає можливість отримати функцію інтенсивності відмов та інші показники надійності програмного засобу. Отримані значення показників надійності можуть служити основою для прийняття рішення про введення програмного засобу в експлуатацію.

Література

1. Математичні моделі та методи аналізу надійності радіоелектронних, електротехнічних та програмних систем [Текст] : монографія / Ю. Я. Бобало, Б. Ю. Волочій, О. Ю. Лозинський, Б. А. Мандзій, Л. Д. Озірковський, Д. В. Федасюк, С. В. Щербовських, В. С. Яковина. – Львів : Видавництво Львівської політехніки, 2013. – 300 с.

2. Маевский, Д. А. Оценка количества дефектов программного обеспечения на основе метрик сложности [Текст] / Д. А. Маевский, С. А. Яремчук // Електротехнічні та комп'ютерні системи. – 2012. – № 7. – С. 113–120.

3. Maevsky, D. A. A New Approach to Software Reliability [Text] / D. A. Maevsky // Lecture Notes in Computer Science : Software Engineering for Resilient Systems.– 2013. – № 8166. – P. 156–168.

4. Goel, A. L. Software reliability models: assumptions, limitations, and applicability [Text] / A. L. Goel // IEEE Transactions on software engineering. – 1985. – Vol. SE-11, No 12. – P. 1411–1423.

5. Goševa-Popstojanova, K. Architecture-based approach to reliability assessment of software systems [Text] / K. Goševa-Popstojanova, K. S. Trivedi // Performance Evaluation. – 2001. – Vol. 45. – P. 179–204.

6. Palviainen, M. The reliability estimation, prediction and measuring of component-based software [Text] / M. Palviainen, A. Evesti, E. Ovaska // The Journal of Systems and Software. – 2011. – Vol. 84. – P. 1054–1070.

7. The Method of Software Reliability Growth Models Choice Using Assumptions Matrix [Text] / V. S. Kharchenko, O. M. Tarasyuk, V. V. Sklyar, V. Yu. Dubnitsky // Computer Software and Applications: Proceedings of the26th Annual Int. Conf., Oxford, England, 26-29 August 2002. – 2002. – P. 541-546.

8. Основи надійності цифрових систем [Текст] : підручник / В. С. Харченко, В. Я. Жихарев, В. М. Ілюшко, В. А. Краснобаєв, П. М. Куліков, І. В. Лисенко, М. В. Нечипорук, Г. М. Тимонькін. – Харків : Нац. аерокосм. ун-т "Харк. авіац. інт-т", 2004. – 573 с.

9. Чабанюк, Я. М. Побудова і дослідження моделі надійності програмного забезпечення з індексом величини проекту [Текст] / Я. М. Чабанюк, В. С. Яковина, Д. В. Федасюк, М. М. Сенів, У. Т. Хімка // Інженерія програмного забезпечення. – 2010. – № 1. – С. 24–29.

10. Яковина, В. С. Моделювання параметру потоку відмов програмного забезпечення та визначення діапазонів показника його складності [Текст] / В. С. Яковина // Вісник Національного університету "Львівська політехніка". Комп'ютерні системи та мережі. – 2014. – № 806. – С. 296–302.

11. Yakovyna, V. Discrete and Continuous Time High-Order Markov Models for Software Reliability Assessment [Електронний ресурс] / V. Yakovyna, O. Nytrebych // ICTERI 2015 : Proceedings of the 11-th Int. Conf., Lviv, Ukraine, May 14-16 2015. – P. 419–431. – Режим доступу: http://ceur-ws.org/Vol-1356/. – 12.05.2015.

12. Яковина, В. С. Компонентні моделі надійності програмного забезпечення вищого порядку [Текст] / В. С. Яковина / Електротехнічні та комп'ютерні системи. – 2015. (подано до друку)

13. Яковина, В. С. Використання засобів UML для прогнозування надійності програмного забезпе-

чення на етапі його проектування [Текст] / В. С. Яковина, Ю. І. Парфенюк // Вісник Національного університету "Львівська політехніка" Комп'ютерні системи та мережі. – 2013. – № 773. – С. 151–156.

14. Cortellessa, V. Early reliability assessment of UML based software models [Text] / V. Cortellessa, H. Singh, B. Cukic // Software and Performance : Proceedings of the 3rd Int. Workshop, 2002. – P. 302–309.

15. Variable state-based software usage-model based on its variables [Text] / D. Fedasyuk, V. Yakovyna, P. Serdyuk, O. Nytrebych // Econtechmod. – 2014. – Vol. 3, No 2. – P. 15–20.

16. Метод побудови сценаріїв тестування програмного забезпечення на основі аналізу його змінних [Текст] / Д. В. Федасюк, В. С. Яковина, П. В. Сердюк, О. О. Нитребич // Інформаційні технології та комп'ютерна інженерія. – 2014. – № 2 (30). – С. 50–58.

17. Критерій достатності процесу тестування програмного забезпечення [Текст] / В. С. Яковина, М. М. Сенів, Я. М. Чабанюк, Д. В. Федасюк, У. Т. Хімка // Вісник Національного університету "Львівська політехніка" Комп'ютерні науки та інформаційні технології. – 2010. – № 672. – С. 346–358.

18. Яковина, В. С. Удосконалена процедура визначення кількості дефектів програмного продукту на ранніх етапах тестування [Текст] / В. С. Яковина, М. М. Сенів, І. Я. Гаранджа // Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту : праці Міжнар. наук. конф., Євпаторія, 27–31 травня 2012. – Євпаторія, 2012. – С. 238.

19. Yakovyna, V. Determination of transition probabilities between software components, written in java, based on monitoring of its execution [Text] / V. Yakovyna, I. Parfeniuk // The Experience of Designing and Application of CAD Systems in Microelectronics : Proceedings of the XIIth Int. Conf., Polyana, 19-23 February 2013. – L., 2013. – P. 382-383.

20. Половко, А. М. Основы теории надежности [Текст] / А. М. Половко, С. В. Гуров. – СПб. : БХВ-Петербург, 2006. – 704 с.

21. Yakovyna, V. The use of modern software development tools to analyze its reliability [Text] / V. Yakovyna, I. Parfeniuk // Perspective Technologies and Methods in MEMS Design : Proceedings of Xth Int. Conf., Lviv, 22-24 June 2014. – L., 2014. – P. 112–114.

Надійшла до редакції 12.05.2015, розглянута на редколегії 18.06.2015

МЕТОД АНАЛИЗА НАДЕЖНОСТИ ПРОГРАММНЫХ СРЕДСТВ С УЧЕТОМ ИХ СЛОЖНОСТИ

В. С. Яковина

Рассматривается метод анализа надежности программных средств с учетом их сложности на разных этапах жизненного цикла. Метод включает использование моделей надежности, базирующихся как на неоднородном пуассоновском процессе, так и на марковском процессе высшего порядка с дискретным и непрерывным временем. Использование предложенного метода дает возможность разработчикам программного обеспечения выбрать соответствующую модель надежности, исходя из особенностей процесса разработки и сложности программных средств, и оценить значения показателей надежности исследуемого программного средства. Предложенный метод анализа надежности программных средств включает руководства и рекомендации касательно получения исходных параметров соответствующих моделей надежности.

Ключевые слова: программные средства, надежность, сложность, компонентный подход, модель надежности, неоднородный пуассоновский процесс, цепь Маркова высшего порядка.

METHOD OF SOFTWARE RELIABILITY ANALYSIS CONSIDERING ITS COMPLEXITY

V. S. Yakovyna

Method of software reliability analysis considering its complexity at various lifecycle stages is offered. The method includes using the nonhomogeneous Poisson software reliability model along with discrete and continuous time higher-order Markov models. By using this method, software developers are able to choose an appropriate reliability model considering software complexity and development process peculiarities, as well as to assess software reliability measures. The proposed method of software reliability analysis includes guidelines and recommendations allowing to get input parameters of corresponding reliability models.

Key words: software, reliability, complexity, component approach, reliability model, nonhomogeneous Poisson process, higher-order Markov chain.

Яковина Віталій Степанович – канд. фіз.-мат. наук, доцент, доцент кафедри програмного забезпечення, Національний університет "Львівська політехніка", Львів, Україна, e-mail: vitaliy.s.yakovyna@lpnu.ua.