УДК 681.3.06

МОДЕЛЮВАННЯ ВИМОГ ДО ЯКОСТІ ПРОГРАМНИХ СИСТЕМ ОБРОБЛЕННЯ ДАНИХ

Г.І. Коваль, Г.Б. Мороз

Інститут програмних систем НАН України
03187, Київ, проспект Академіка Глушкова, 40,
тел.: (044) 526 4579;
Київський національний університет внутрішніх справ України
02121, Київ, Колекторна, 4,
тел.: (044) 561 1813

Розроблено трирівневу модель якості програмних систем (ПС) на базі характеристики якості – завершеність (зрілість), яка забезпечує взаємозв'язок метрик внутрішньої, зовнішньої і експлуатаційної якості, а саме: кількість (щільність) дефектів, ймовірність безвідмовної роботи і задоволеність надійною роботою ПС. Подано модель вимог до завершеності окремих компонентів системи, яка дозволяє апріорі визначити оптимальний рівень завершеності кожного модуля шляхом максимізації функції корисності (як міри експлуатаційної якості ПС), уникаючи отримання математичного формулювання надійності функцій ПС.

A threelevel *quality model* of software system (SS) is developed on the base of one quality characteristic – *maturity*, which makes interconnection of the internal, external and operating (in use) quality metrics, namely: number (or density) *of defects*, *probability of faultless work* and *satisfaction by the reliable work* of SS. *A model of requirements* for the maturity of the system components is presented, which allows to define a priori an optimum level of maturity of every module by maximization of utility function (as a measures of the quality in use), avoiding receipt of mathematical formulation of reliability of the software system's functions.

Введення

Забезпечення якості програмних систем (ПС) є невід'ємною складовою процесу розроблення та необхідною умовою конкурентоспроможності організацій-розробників.

Існує чимало моделей, призначених для оцінювання показників якості ПС (зокрема, надійності) на завершальних стадіях життєвого циклу (ЖЦ) ПС. Але не так багато моделей, які б допомагали замовникам та розробникам встановлювати обґрунтовані *кількісні вимоги* до цих показників на початку розробки.

Очевидно, що залежно від класу системи (критична / не критична), частоти застосування окремих компонентів користувачами різних категорій у ділових процесах, наслідків відмов окремих компонентів ПС для користувачів, а також інших чинників (вартості розроблення, ціни втрат тощо), вимоги до якості тих чи інших компонентів ПС будуть різними. Тому моделювання вимог до якості кожного компонента ПС є актуальною задачею.

У роботі проаналізовано переваги та недоліки існуючих підходів до визначення вимог до якості ПС та запропоновано модель встановлення кількісних цільових рівнів якості для однієї з найважливіших характеристик якості ПС – *завершеності* (*зрілості*). Разом з моделями прогнозування дефектів та систематичного кількісного контролю досяжності встановлених цільових рівнів якості ця модель забезпечує прийняття обґрунтованих ефективних рішень з керування якістю ПС в ході виконання програмних проектів [1]. Робота виконана в рамках досліджень за проектом ІПС НАН України 1/02-02 "Розроблення концепції, методів та методичного апарата вдосконалення та стандартизації процесів життєвого циклу систем програмного забезпечення".

1. Характеристика програмних систем оброблення даних

Програмну систему визначатимемо як групу інтегрованих програмних засобів, які підтримують певний діловий процес споживача (або його частину) і спільно використовують бази даних (БД). Приклад ПС - група програмних застосувань (ПрЗ), що складають автоматизоване робоче місце фахівця в предметній області (ПрО).

Якість, за визначенням стандарту ДСТУ 2844-94, це "сукупність властивостей програмної системи, що забезпечують її здатність задовольняти встановлені або передбачувані потреби відповідно до призначення" [2].

Які саме властивості продукту будуть асоціюватися з поняттям якості ПС, тобто, якою буде модель її якості, залежить від особливостей ПрО, призначення ПС та категорій її користувачів, а також від кінцевої мети моделювання якості з позицій того, хто його виконує. Більшість відомих робіт з моделювання якості, насамперед, робіт В.В. Ліпаєва, стосуються проблем забезпечення якості великих виробничих систем, працюючих у реальному масштабі часу [3].

У даній роботі розглядаються *ПС оброблення даних*, що не є критичними з позицій безпеки функціювання та призначені здебільшого для вирішення задач:

– автоматизації керівної адміністративної діяльності (системи організаційного управління);

– вирішення добре структурованих задач (розрахунково-оптимізаційні системи);
– пошуку і видачі (регулярно або по запитах) необхідної інформації у вигляді стандартизованих повідомлень, регламентованих звітів та довідок (інформаційно-пошукові системи);
– підтримки прийняття рішень у слабко структурованих ПрО (системи підтримки прийняття рішень).

Характерними рисами ПС оброблення даних є:
– наявність складних моделей даних у ПрО і взаємодія з СКБД для ведення великих об'ємів інформації;
– функціонально складний багатовіконний інтерфейс користувача;
– наявність засобів захисту інформації;
– складне розподілене середовище експлуатації;
– застосування компонентного підходу до розроблення [4];
– застосування широкого спектру інструментів підтримки розроблення - від простих засобів візуального програмування (Delphi, Visual Basic, Visual C тощо) до інтегрованих CASE-засобів (наприклад, Oracle Designer, Oracle Developer та Oracle Express);
– використання можливостей загальносистемного програмного забезпечення (зокрема клієнтських і серверних компонентів СКБД);
– інтеграція нових та повторно-використовуваних компонентів програмного забезпечення [5].

Досвід створення ПС оброблення даних показує, що визначальною характеристикою їх якості є завершеність (безвідмовність) - властивість ПС уникнути відмови через приховані дефекти. Тому в даній роботі саме завершеність визначена ключовою характеристикою якості, яка покладається в основу моделювання якості. Метою моделювання якості є покращення керованості проектами за критерієм якості, тобто за основу взятий погляд на якість менеджера проекту.

2. Підхід до моделювання якості ПС оброблення даних

Якість ПС вимірюється за допомогою *метрик якості*. За визначенням стандарту ISO/IEC 9126-2 *метрика якості* ПС являє собою "модель вимірювання атрибута, що пов'язується з однією характеристикою якості ПС. Це комбінація конкретного *методу вимірювання (*способу отримання значень) атрибута сутності й *шкали вимірювання*" [6].

За видом (станом) об'єкта вимірювання (працююча версія програми або сукупність документів та програмного коду) метрики поділяються на зовнішні, внутрішні й експлуатаційні метрики ПС.

Внутрішні метрики призначені для оцінювання *внутрішньої якості* – множини властивостей продукту, яка визначає його здатність задовольняти встановленим або реальним потребам при використанні в певних умовах та забезпечує можливість користувачам, менеджерам та розробникам оцінювати якість проміжних і кінцевих продуктів ПС безпосередньо за їх властивостями, без виконання на комп'ютері.

Зовнішні метрики використовують виміри працюючого на комп'ютері програмного продукту, отримані в результаті вимірювання його поведінки в ході роботи. Вони призначені для оцінювання *зовнішньої якості* - міри, до якої продукт відповідає встановленим (заявленим) і передбачуваним вимогам під час його використання у певних умовах.

Метрики експлуатаційної якості вимірюють, до якої міри програмний продукт, встановлений і використовуваний у певному середовищі експлуатації задовольняє потреби користувачів стосовно ефективного, продуктивного та безпечного вирішення задач, а також справляє загальне позитивне враження. Вони допомагають оцінити не властивості самої ПС, а видимі результати її експлуатації – *експлуатаційну якість*. В.В. Ліпаєв у роботі [7] використовує інший термін для означення цього рівня якості, а саме "качество программного средства *в использовании*". Характеристиками якості ПС у використанні є продуктивність, результативність, безпека функціонування та задоволеність від використання програмного продукту [6].

Для того щоб ефективно керувати якістю за визначеною характеристикою або множиною характеристик, необхідно розробити таку *модель якості*, яка б встановлювала взаємозв'язок мір і відповідних метрик внутрішньої, зовнішньої та експлуатаційної якості ПС, тобто поєднувала погляди на якість як розробників системи, так і її безпосередніх користувачів.

На початку розроблення ПС специфікуються значення (область значень) зовнішніх метрик, які слугуватимуть критерієм досягнення рівня якості при випробовуванні ПС, а потім визначаються найбільш придатні (з позиції прогнозу значень зовнішніх метрик) внутрішні метрики і планується поетапне досягнення зовнішніх вимог до якості. Очевидно, що зовнішні вимоги до якості ПС треба встановлювати з позицій забезпечення максимально можливої експлуатаційної якості програмного продукту. В даній роботі пропонується експлуатаційну якість ПС пов'язувати з загальною *задоволеністю* користувачів безвідмовним функціонуванням ПС, якій відповідає характеристика якості ПС у використанні – *задоволеність* ("satisfaction").

3. Трирівнева модель якості ПС у контексті характеристики "завершеність"

У контексті *завершеності* ПС оброблення даних головним показником її внутрішньої якості є дефекти, зовнішньої – відмови, а експлуатаційної – загальне враження користувачів системи.

Пропонується трирівнева *модель якості*, яка забезпечує взаємозв'язок таких мір внутрішньої, зовнішньої та експлуатаційної якості (рис. 1):

D_0 – кількість (щільність) дефектів у кожному компоненті ПС (внутрішня міра);

$R(t)$ – безвідмовність, ймовірність безвідмовної роботи кожного компонента ПС, тобто ймовірність того, що протягом заданого часу t експлуатації компонента у визначених умовах не виникне послідовність вхідних даних у сценарії його використання, яка призведе до відмови (зовнішня міра);

Q_{nc} – міра експлуатаційної якості ПС, що пов'язується з загальною *задоволеністю* користувачів безвідмовним функціонуванням ПС.

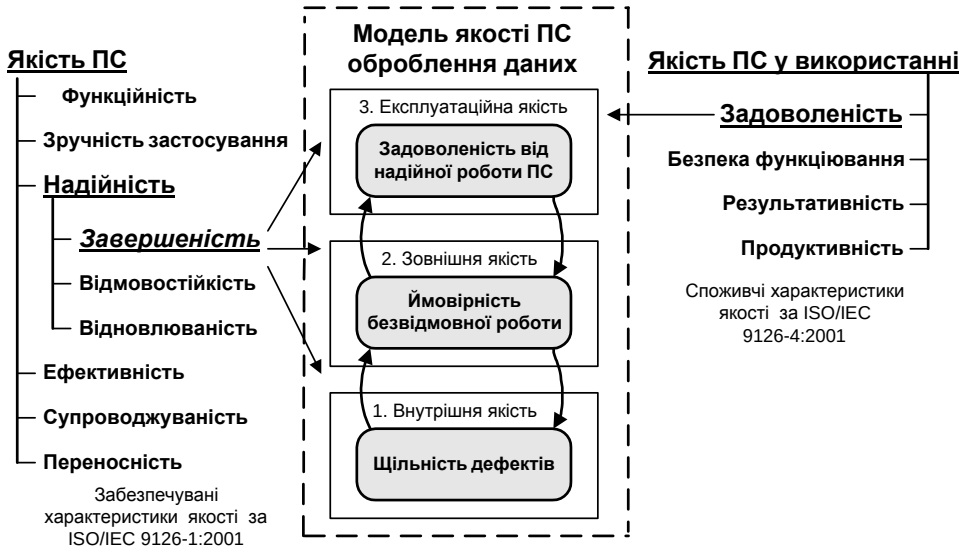

Рис. 1. Трирівнева модель якості ПС оброблення даних

У лівій частині рис. 1 – дворівнева структура характеристик і підхарактеристик якості в еталонній моделі якості, у правій – перелік характеристик якості ПС у використанні [6]. У центрі рис. 1 – власне трирівнева модель якості за вибраною підхарактеристикою – завершеність.

4. Огляд підходів до встановлення кількісних вимог до якості ПС

Стислий огляд підходів до встановлення кількісних вимог до якості ПС у контексті вибраної характеристики "завершеність" подано у таблиці. В ній описано критерії розподілу вимог по компонентах ПС, переваги та недоліки підходів (моделей), а також подано посилання на використані джерела інформації. Оскільки в спеціалізованій літературі традиційно досліджується характеристика якості *надійність ПС*, а не її підхарактеристика – *завершеність*, в таблиці збережена термінологія, що склалася.

Підходи до розподілу кількісних вимог до надійності ПС — Таблиця

Назва (автор) підходу	**Сутність підходу до розподілу надійності**	**Переваги та недоліки**
Не критеріальний	Призначення однакового цільового значення надійності всім компонентам (модулям) (без розподілу)	Встановлює необґрунтовані вимоги до надійності. Якщо вимоги надто високі, вони можуть бути не досяжними або надто ресурсоємними
Технічні підходи		
Ахмед Абд-Аллах [8]	Побудова блок-схеми надійності системи з апаратно-програмними компонентами	Успадкоємлені з теорії надійності технічних систем. Прийнятні лише для вбудованого програмного забезпечення та за умови наявності розрахункових значень характеристик надійності для всіх складових системи, включаючи апаратні
А. Нейфелдер [9]	Рівно пропорційне розподілення з урахуванням послідовної / паралельної роботи модулів	
Підходи на основі специфікації використання програмних застосувань [10]		
Р.Чеінг [11]	Розподілення з урахуванням ймовірностей міжмодульних переходів у сценаріях роботи ПрЗ	Історично перша робота (1980 рік), яка дала поштовх до розвинення підходу
Дж.Муса [12]	Розподілення за функціональним або операційним профілем. Використовує ієрархічну будову (дерево) для розподілення характеристик	Переваги – можливість урахування особливостей використання великомасштабних систем. Побудова сценаріїв тестування з максимальним врахуванням характеристики

Назва (автор) підходу	Сутність підходу до розподілу надійності	Переваги та недоліки
	використання системи у відповідності до категорій користувачів, режимів системи, функцій та операцій	середовища експлуатації ПС, економія ресурсів тестування. Недоліки – чутливість моделей до зміни операційного профілю
Дж. Мунсон [13]	Розподілення на основі виконавчого профілю ПрЗ. Використовує моделювання функцій ПрЗ стохастичним процесом запуску та відпрацювання модулів, які їх реалізують	Аналіз переходів між модулями дозволяє побудувати шаблони виконання кожної функції ПрЗ Недоліки – потребує вбудовування додаткового коду в модулі для оцінювання частоти звернення
Дж. Рейгопал [14]	Розподілення на основі марківських моделей передачі керування між модулями ПрЗ	Враховує погляд розробників на можливі сценарії роботи системи. Недоліки – складність побудови моделей для великих систем
Дж. Вайтеккер [15]	Розподілення на основі марківських моделей переходу ПрЗ із одного стану в інший	Враховує погляд користувачів на сценарії використання системи. Недоліки – ті самі
П. Раннісан [16]	Розподілення на основі ієрархічних моделей станів. Моделі мають ієрархічну та поведінкову складові. Поведінкові складові є ланцюгами Маркова (з дискретним часом), а ієрархічні – структурами-деревами	Враховує погляд користувачів на сценарії використання системи. Але, оскільки модель є композицією марківських моделей, загальний простір станів системи не визначається. Недоліки – ті самі
Г. Хечт [17]	Розподілення за чинниками операційної критичності та режимами відмов. Ґрунтується на методі SFMEA (Software Failure Mode and Effect Analysis)	Призначений для систем, розроблюваних за допомогою UML, використовує всі переваги інструментів програмної підтримки UML для SFMEA
Розподілення з урахуванням властивостей модулів		
Н. Олссон [18]	Розподілення на основі оцінювання складності модулів та їх схильності до помилок	Недоліки – не враховує погляд користувачів на сценарії використання системи
Розподілення з урахуванням ресурсів проекту		
М. Ліу [19]	Розподілення з урахуванням витрат часу на тестування (мінімізація часу тестування - критерій оптимізації). Задача нелінійної оптимізації	Придатна для комплексного розподілення встановленого цільового значення інтенсивності відмов для ПС з багатьма ПрЗ. Недолік – не враховує погляд користувачів на важливість компонентів ПС
М. Хелендер [20]	Розподілення надійності на основі операційного профілю ПС з урахуванням витрат на досягнення цілей надійності. Задача нелінійної оптимізації	Поєднує два підходи до планування надійності та вартості розробки: мінімізація вартості за умови обмеження надійності та максимізація надійності за умови обмеження бюджету. Недолік – чутливість моделі до зміни операційного профілю

На основі аналізу переваг та недоліків розглянутих підходів сформульовано такі вимоги до моделі розподілу надійності за компонентами програмних систем оброблення даних:

– ґрунтування на погляді *користувачів* ПС на важливість виконання окремих функцій у загальному діловому процесі організації;

– врахування *думки* розробників ПС стосовно важливості окремих програмних застосувань для виконання функцій ПС;

– дотримання *диференційованого підходу* до компонентів (модулів) ПС з урахуванням інтенсивності звернення до них у операційних сценаріях;

– забезпечення максимального рівня *загальної задоволеності* замовника постаченим програмним продуктом через досягнення встановлених цільових значень надійності компонентів ПС, адекватних потребам користувачів.

У розділі 5 подано модель, яка задовольняє висунутим вимогам і грунтується на побудованій трирівневій моделі якості у контексті завершеності компонентів ПС оброблення даних.

Спочатку на верхньому рівні моделі встановлюються цільові рівні зовнішньої міри завершеності – безвідмовності кожного компонента ПС з урахуванням потреб забезпечення найвищої загальної експлуатаційної якості системи.

Далі визначається комплексна зовнішня метрика завершеності компонентів ПС, яка (як складову) включає внутрішню метрику завершеності – щільність дефектів у компоненті ПС.

На сам кінець вимоги до значень зовнішньої міри завершеності компонентів трансформуються у цільові рівні щільності дефектів у ПС.

5. Модель вимог до завершеності компонентів ПС

5.1. Встановлення кількісних вимог до безвідмовності компонентів ПС. Для встановлення цільових рівнів безвідмовності компонентів ПС оброблення даних пропонується чотирирівнева ієрархічна структура, кожний рівень якої відповідає *рівню бачення* проблеми якості відповідною категорією учасників проекту ПС, а саме:

– замовника, який зацікавлений у *загальній якості ПС при її використанні* ($Q_{пс}$) та підвищенні ефективності ділового процесу завдяки надійній роботі впровадженої ПС;

– користувачів, які пов'язують загальну якість системи з надійним виконанням множини функцій ПС (F_1, …, F_k);

– менеджерів (та аналітиків), які пов'язують надійне виконання кожної функції F_i з безвідмовною роботою множини розроблюваних програмних застосувань (Z_1, …, Z_l), призначених для автоматизованої підтримки функцій;

– проектувальників (та програмістів), які пов'язують надійність кожного ПрЗ Z_i з безвідмовністю множини розроблюваних, а також повторно використовуваних *незалежних модулів* (M_1, …, M_m), до яких є звернення у ПрЗ. Припущення незалежності відповідає сучасним концепціям об'єктно-орієнтованого та компонентного підходів до розроблення ПС.

Приклад ієрархічної структури ПС показано на рис. 2.

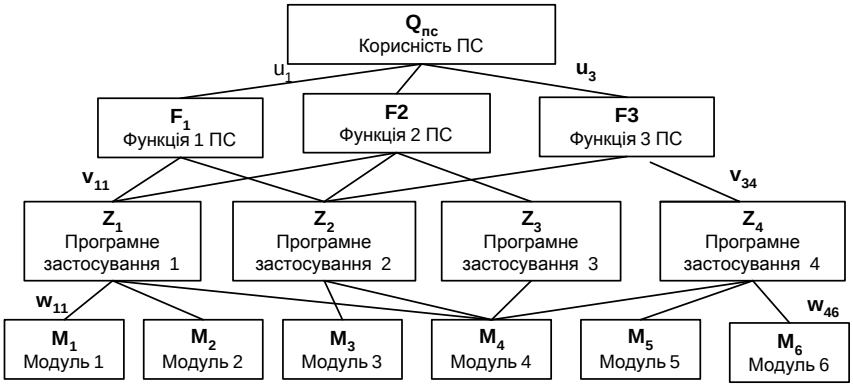

Рис. 2. Чотирирівнева ієрархічна структура ПС

Ієрархічна декомпозиція є природним засобом спрощення проблеми в системах оброблення даних, не пов'язаних з функціонуванням в реальному масштабі часу. Вона властива сучасним CASE-технологіям, застосовним для побудови ПС досліджуваного класу. Наприклад, Oracle Designer'2000 надає засоби для автоматизованого опису ділового процесу в організації-замовнику та побудови діаграм ієрархії функцій у цьому процесі, а також для визначення на нижньому рівні ієрархії модулів-кандидатів трьох типів – модулів-форм, модулів-звітів та модулів-меню.

Основна мета побудови і аналізу ієрархії ПС – отримати *параметри* моделі вимог до безвідмовності на кожному її рівні з урахуванням *важливості компонентів* кожного з рівнів 2 – 4 для загальної якості ПС.

Для визначення ваги окремих компонентів у ієрархії доцільно використати *метод аналізу ієрархій* (MAI) [21].

Міру експлуатаційної якості ПС пропонується визначати як функцію корисності вигляду

$$Q_{пс} = \sum_{i=1}^{k} a_i \cdot R_i \ ,$$

де a_i – міра важливості функції ПС для ділового процесу користувача, R_i – надійність виконання функції у заданому періоді t експлуатації системи.

Надійність виконання функцій ПС можна оцінити лише під час системного тестування та експлуатації ПС, що з позицій керування якістю надто пізно. Тому виконується декомпозиція проблеми і вирішується задача апріорного знаходження оптимального рівня безвідмовності кожного компонента (модуля та ПрЗ) системи, який забезпечує максимізацію функції корисності з урахуванням технічних та ресурсних обмежень проекту ПС.

Постановка задачі. Введемо позначення:

u_i – коефіцієнт відносної ваги функції F_i у забезпеченні $Q_{пс}$, i = 1,… ,k;

v_{ij} – коефіцієнт відносної ваги j-го ПрЗ у забезпеченні виконання i-ої функції, i = 1,…, k; j = 1,…, l;

w_{js} – коефіцієнт відносної ваги s-го модуля в забезпеченні виконання j-го програмного застосування $s = 1,\ldots, m; j = 1,\ldots, l$;

r_s – безвідмовність модуля M_s у період експлуатації t;

E_j – множина номерів усіх модулів, які необхідні для виконання j-го ПрЗ;

α_s – нижня межа безвідмовності модуля M_s;

β_s – верхня межа безвідмовності модуля M_s;

G – загальна ціна ПС;

C – собівартість створення ПС організацією-розробником;

c_s – накладні витрати, пов'язані з розробленням модуля M_s ;

d_s – витрати, необхідні для досягнення одиничного рівня безвідмовності модуля M_s;

δ – частка прибутку в ціні ПС.

Необхідно знайти такий розподіл безвідмовності модулів, при якому функція корисності Q_{nc} буде досягати максимуму:

$$Q_{nc}(r_1,...,r_m) = \sum_{j=1}^{l}(\sum_{i=1}^{k} u_i v_{ij} \cdot \prod_{n\in E_j} r_n) \to \max \qquad (1)$$

при обмеженнях

$$0 < \alpha_s \le r_s \le \beta_s \le 1; \qquad s = 1, \ldots, m, \qquad (2)$$

$$c_s + d_s \cdot r_s \le (1-\delta)\cdot G \cdot \sum_{j=1}^{l}\sum_{i=1}^{k} u_i v_{ij} w_{js}\ , \qquad (3)$$

$$\sum_{s=1}^{m}(c_s + d_s \cdot r_s) \le C\,. \qquad (4)$$

Задача (1) – (4) є задачею нелінійної оптимізації з лінійними обмеженнями, яка практично вирішується за допомогою математичного пакета MATLAB 6.

Параметри u_i, v_{ij}, w_{js} ($i = 1,\ldots, k$; $j = 1,\ldots, l$; $s = 1,\ldots, m$) являють собою *оцінки внеску* кожної функції, програмного застосування та модуля в ієрархічній структурі ПС у забезпечення максимальної корисності системи. Вони знаходяться за МАІ шляхом парного порівняння та послідовного визначення локальних пріоритетів компонентів ПС в межах кожного рівня ієрархії щодо компонентів попереднього (вищого) рівня.

Обмеження (2) задають допустимі нижні α_s, верхні β_s межі безвідмовності модулів, виходячи з міркувань важливості (або ризику відмов) кожного модуля.

Обмеження (3) встановлюють відповідність загальних витрат на розроблення модуля, з одного боку, і частки ціни системи G, яка припадає на цей модуль з огляду на його внесок в якість системи, та з урахуванням частки прибутку δ розробника, з другого боку. Припускається лінійна залежність між вартістю модуля і його безвідмовністю.

Обмеження (4) встановлює відповідність сумарних витрат на розроблення всіх модулів і собівартості створення ПС. Витрати на розроблення модуля складаються з накладних (непрямих) та прямих витрат безпосередньо на його створення.

Таким чином за допомогою описаної моделі встановлюються вимоги до безвідмовності кожного модуля (r_i), а потім і кожного програмного застосування (q_i) з огляду на незалежність модулів у структурі ПрЗ, тобто визначаються цільові рівні безвідмовності ПрЗ.

Наступна задача – вибір метрики для апріорного (до тестування) оцінювання безвідмовності й встановлення взаємозв'язку цільових рівнів безвідмовності з цільовими рівнями дефектів у кожному ПрЗ.

5.2. Визначення кількісних вимог до рівнів дефектів у компонентах ПС. Процес відмов кожного компонента ПС оброблення даних моделюється неоднорідним пуассонівським процесом [22]. Для пуассонівських моделей надійності умовна функція надійності $R(t\,|\,T)$ визначається за формулою

$$R(t\,|\,T) = \exp(-(m(T+t) - m(T)))\,, \qquad (5)$$

де $R(t\,|\,T)$ – умовна ймовірність того, що протягом заданого часу t експлуатації ПрЗ у визначених умовах середовища його функціонування не виникне послідовність вхідних даних, яка призведе до його відмови, якщо ПрЗ тестувалося протягом часу T;

m(t) – функція зростання надійності, яка являє собою середню кількість дефектів у ПрЗ, виявлених під час його функціонування протягом часу t.

Аналіз ряду пуассонівських моделей показує доцільність використання функції зростання надійності *m(t)* експоненційної моделі Дж. Муси [23]. Ця модель, на відміну від інших моделей цього класу, не потребує даних про відмови і, саме тому, придатна для *раннього* моделювання безвідмовності ПрЗ.

З огляду на застосування *m(t)* моделі Дж. Муси у (5) отримуємо метрику безвідмовності:

$$R(t\,|\,T) = \exp(-(m(T+t) - m(T))) = \exp[-N_0 \exp(-\frac{\lambda_0}{N_0}T)(1-\exp(\frac{\lambda_0}{N_0}\cdot t))]\,, \qquad (6)$$

де N_0 – кількість прихованих дефектів у ПрЗ на початку системного тестування;

λ_0 - інтенсивність відмов ПрЗ на початку системного тестування, що визначається за формулою

$$\lambda_0 = N_0 \cdot \frac{\rho \cdot K}{I \cdot \varphi},$$

де ρ – інтенсивність виконання коду (швидкість процесора);

$K = 4 \cdot 10^{-7}$ – коефіцієнт виявлення дефектів (постійний для моделі Дж. Муси);

I – кількість інструкцій початкового коду;

φ – коефіцієнт розширення коду (число інструкцій виконуваного коду, яке припадає на одну інструкцію початкового);

t – встановлений час безвідмовного функціювання ПрЗ у експлуатації.

Якщо у формулі (6) покласти $T = 0$, тобто розглянути ситуацію, коли ПрЗ взагалі не перебуватиме на стадії системного тестування, тоді

$$R(t) = R(t|0) = \exp[-D_0 \cdot I \cdot (1 - \exp(\frac{\lambda_0}{D_0 \cdot I} \cdot t))], \quad (7)$$

де $D_0 = N_0 / I$ – щільність прихованих дефектів на початку тестування.

Формули (6) та (7) є зовнішніми метриками, які дозволяють моделювати безвідмовність ПрЗ з урахуванням (або без) часу його тестування.

Маючи встановлені цільові значення безвідмовності q_i (отримані з формули 4) для і-го компонента ПС, можна обчислити відповідні цільові рівні кількості N_{0i} (або щільності D_{0i}) дефектів шляхом розв'язку рівнянь (6) або (7) стосовно N_{0i} (або D_{0i}).

Розмір (об'єм) ПрЗ I_i на початку розробки доцільно визначати за одним з методів у методології FPA (Function points analysis) в умовних одиницях функціональності та конвертувати отримане значення в одиниці KSLOC (тисяча рядків некоментованого початкового коду) [24].

Встановлені цільові рівні кількості N_{0i} (або щільності D_{0i}) дефектів для кожного *i*-го компонента ПС – це *максимальні* значення, які не будуть перешкодою для досягнення відповідної безвідмовності q_i і забезпечення максимальної експлуатаційної якості у контексті завершеності ПС.

Керування якістю ПС полягає у систематичному прогнозуванні дефектів у компонентах ПС з урахуванням множини чинників проекту [25], кількісному контролі досяжності встановлених вимог до показників якості та своєчасному усуненні дефектів шляхом верифікації або тестування.

Висновки

Правильний вибір характеристик якості ПС та встановлення обгрунтованих кількісних вимог до показників якості є основою ефективного керування якістю.

У даній роботі запропоновано підхід до моделювання вимог до якості на прикладі однієї з характеристик, а саме завершеності ПС. Поряд з нею доцільно було б розглянути таку важливу характеристику якості ПС оброблення даних, як безпека інформації ("security"), що є підхарактеристикою функційності ПС, але на теперішній час проблема полягає у визначенні внутрішніх та зовнішніх метрик для цієї (а також для більшості інших) характеристик якості та встановленні формалізованого взаємозв'язку між ними.

Оцінки внеску різних компонентів у якість ПС можна було б виконувати необов'язково за допомогою MAI (наприклад, простим ранжуванням або парним порівнянням елементів ієрархії), що спростило б практичну реалізацію моделі, позбавило необхідності визначати власні значення та вектори матриць, але .MAI був залучений до моделювання з огляду на перспективи його застосування для побудови більш складних моделей якості з поєднанням в одній моделі різних характеристик, упорядковуванням їх за пріоритетами та, за потреби, визначенням інтегрального показника якості ПС.

1. *Коваль Г.І.* Основні задачі підтримки прийняття рішень в інженерії надійності програмних систем // Проблеми програмування. – 2005. – № 3. – С. 35–41.
2. *ДСТУ 2844-94* Програмні засоби ЕОМ. Забезпечення якості. Терміни та визначення. – К.: Держстандарт України. – 1994. – 18 с.
3. *Липаев В.В.* Методы обеспечения качества крупномасштабных программных систем. – М.: СИНТЕГ. – 2003. – 510 с.
4. *Грищенко В.Н., Лаврищева Е.М.* Методы и средства компонентного программирования // Кибернетика и системный анализ. – 2003. – № 1. – С. 39–55.
5. *Лаврищева Е.М.* Парадигма интеграции в программной инженерии // Проблемы программирования.– 2000.– № 1-2. – С. 351 – 360.
6. *Основы* инженерии качества программных систем / Ф.И. Андон, Г.И. Коваль, Т.М. Коротун, В.Ю. Суслов // Под ред. И.В. Сергиенко. – Киев: Академпериодика. – 2002. – 504 с.
7. *Липаев В.В.* Обеспечение качества программных средств. Методы и стандарты. – М.: СИНТЕГ. – 2001.– 380 с.
8. *Abd-Allah A.* Extending reliability block diagrams to software architectures / Center for Software Engineering. Computer Science Department. University of Southern California. Los Angeles. Technical Report: USC-CSE-97-501.- http:// sunset.usc.edu/publications/ TECHRPTS/1997/usccse97-501/usccse97-501, ps.
9. *Lakey P.B., Neufelder A.M.* System and software reliability assurance notebook // Rome Laboratory Report, Griffiss Aif Force Base, Rome NY. – 1997. – 186 с.
10. *Мороз Г.Б., Коваль Г.И., Коротун Т.М.* Концепция профилей в инженерии надежности программных систем // Математичні машини і системи. – 2004. – № 1. – С. 166 – 184.

11. *Cheung R.* A User-oriented Software Reliability Model // IEEE Trans. Soft. Eng., – 1980. – SE-6, N. 2. – P. 11– 125.
12. *Musa J.D.* Operational Profiles in Software Reliability Engineering // IEEE Software. –1993. –V.10, N.2. — P. 14 – 32.
13. *Munson J., Elbaum S.* Software reliability as a function of user execution patterns // Proc. of the 32nd Hawaii International Conference on System Sciences. – 1999. – P. 1 – 12.
14. *Rajgopal J., Mazumdar M., Majety S.V.* Optimum Combined Test Plans for Systems and Components // IIE Transactions. – 1999. – V.31. – P. 481 – 490.
15. *Whittaker J., Thomason M.* Markov chain model for statistical Software testing // IEEE Trans. Soft. Eng. – 1994.–SE-20, N.10. – P.812–824.
16. *Runesson P., Wohlin C.* Usage Modelling: The Basis for Statistical Quality Control // Proceedings 10th Annual Software Reliability Symposium, Denver, Colorado. – 1992. –- P. 77 – 84.
17. *Hecht H.* An Alternative Software Reliability Assessment // Proceedings 14h International Symposium on Software Reliability Engineering (ISSRE 2003), Denver, Colorado. – 2003. – P. 293 – 295.
18. *Ohlsson N., Helander M., Wohlin C.* Quality Improvement by Identification of Fault-Prone Modules using Software Design Metrics // Proceedings Sixth International Conference on Software Quality. – 1996. – P. 1 – 13.
19. *Lyu M.R., Rangarajan S., vanMoorsel A.P.A.* Optimization Of Reliability Allocation And Testing Schedule For Software Systems // Proceedings Eighth International Symposium on Software Reliability Engineering (ISSRE '97). – 1997. – P. 336 – 438.
20. *Helander M.E., Zhao M., Ohlsson N.* Planning Models for Software Reliability and Cost //IEEE Trans. Softw. Eng. – 1998. – V. 24. – N. 6. – P. 420 – 434.
21. *Саати Т.* Принятие решений. Метод анализа иерархий. – М.: Радио и связь, 1993. – 315 с.
22. *Мороз Г.Б.* Пуассоновские модели роста надежности программного обеспечения и их приложение. Аналитический обзор // УСиМ. – 1996. - № 1-2. – С. 69 – 85.
23. *Мороз Г.Б., Лаврищева Е.М.* Модели роста надежности ПО. – Киев, 1992. – 25 с. – (Препр. / АН Украины. Институт кибернетики им. В. М. Глушкова; 92–38).
24. *Коваль Г.И.* Методы определения размера ПО // Проблемы программирования. – 1999 – № 1. – С. 63–71.
25. *Коваль Г.І.* Байєсівські мережі як засіб оцінювання та прогнозування якості програмного забезпечення // Проблеми програмування. – 2005. – № 2. – С. 15–23.