УДК 004.891.3: 004.3
Т.О.Говорущенко
Хмельницький національний університет

ВИЗНАЧЕННЯ НЕОБХІДНОСТІ ПОВТОРНОГО ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В ПРОЦЕСІ ЕКСПЕРТИЗИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В статті описано визначення необхідності повторного тестування програмного забезпечення як додаткове незалежне випробування з метою перевірки відповідності ПЗ встановленим вимогам.

Т.А.Говорущенко. Определение необходимости повторного тестирования программного обеспечения в процессе экспертизы программного обеспечения. *В статье описано определение необходимости повторного тестирования программного обеспечения как дополнительное независимое испытание с целью проверки соответствия программного обеспечения установленным требованиям.*

T.O.Govorushchenko. Determination of software retesting necessity in software examination process. *In article determination of software retesting necessity as additional independent examination with the purpose of inspection of software conformity the established requirements is described.*

Вступ. Процес розробки програмного продукту подібний до інших інженерних задач. Його найбільш вдала адаптація для програмістів називається Уніфікований процес розробки програмного забезпечення (USDP- Unified Software Development Process). Інститутом IEEE було розроблено стандарти документації такого процесу розробки програмного забезпечення (ПЗ). Згідно цих стандартів, документація проекту складається з наступних складових, розроблюваних в порядку перерахування [1]:

- *План експертизи програмного забезпечення* (SVVP – Software Verification and Validation Plan): документ описує, яким чином і в якій послідовності повинні перевірятись стадії проекту і сам продукт на відповідність вимогам. Іншими словами, цей документ обумовлює те, як замовник і розробник переконаються в тому, що вони зробили те, що хотіли зробити, але не описує, що і як вони робитимуть.
- План контролю якості ПЗ (SQAP – Software Quality Assurance Plan) - показує, яким чином проект повинен досягти відповідності встановленому рівню якості;
- План управління конфігураціями ПЗ (SCMP – Software Configuration Management Plan) - описує, де і як розробник зберігатиме версії документації, програмного коду і як визначатиме, яким чином код пов'язаний з документацією;
- План управління програмним проектом (SPMP – Software Project Management Plan) - описує, як розробник керуватиме проектом створення програмного продукту, щоб довести розробку до вдалого фіналу;
- Специфікація вимог до програмного забезпечення (SRS – Software Requirements Specification) - це найважливіший документ. Складаться з двох частин: перша включає те, що "попросив" замовник, друга - те, що "зроблять програмісти";
- Проектна документація ПЗ (SDD – Software Design Document) - це архітектура і деталі проектування додатку;
- Документація по тестуванню ПЗ (STD – Software Test Documentation) - опис того, як повинно проводитись тестування, хто при цьому присутній, результати тестування і т.д.

Експертна оцінка програм (peer review) призначена для ефективного та раннього виявлення дефектів в ПЗ і може бути реалізована за допомогою перегляду вихідних текстів, наскрізного структурного контролю або інших методів колективного вивчення.

Основу методології експертизи ПЗ складає оцінка виконання вимог до ПЗ на різних етапах життєвого циклу. При цьому необхідно оцінювати виконання функційних вимог і вимог до цілісності безпеки [2].

Місце процесу повторного тестування в процесі експертизи ПЗ. Розглянемо методи оцінки виконання вимог до ПЗ. Така оцінка може проводитись не лише експертами в процесі ліцензування, але й розробником (в ході тестування і верифікації) та замовником ПЗ (при прийнятті ПЗ в експлуатацію). Метою проведення оцінки ПЗ є перевірка його відповідності встановленим вимогам. Не менш важливою метою роботи експерта є здійснення реального впливу на підвищення рівня якості і надійності ПЗ. Для цього всі зауваження та рекомендації експертів повинні передаватись розробникам для оперативного усунення виявлених недоліків. В результаті сумісної діяльності розробників і експертів можуть

вноситись корективи в проект і знижуватись кількість невиявлених дефектів ПЗ. Застосовувані методи можуть включати проведення додаткових незалежних випробувань [2-5].

Підвищити достовірність тестування (ймовірність проведення вірного і вичерпного тестування ПЗ, під час якого не припускались помилки) і відповідно якість ПЗ можна не тільки шляхом тестування дефектів на етапах розроблення та налагодження, а й шляхом повторного тестування з метою виявлення прихованих помилок у програмах після основного тестування. Це підтверджується тим, що достовірність тестування і якість ПЗ залежать від кількості виявлених помилок, у т.ч. і прихованих.

Для проведення оцінки замовником ПЗ з метою перевірки його відповідності встановленим вимогам та підвищення рівня якості і надійності ПЗ можна використовувати повторне тестування - тестування з метою виявлення прихованих помилок, яке здійснюється після розроблення та налагодження ПЗ і є окремим технологічним процесом [6, 7].

Повторне тестування здійснюється на етапі вхідного контролю, який здійснює замовник, тобто допомагає замовнику оцінити якість тестування програмного забезечення, яке приймається, і вказує на наявність в ньому прихованих помилок [8, 9]. Повторне тестування може застосовуватись на вимогу замовника або якщо на його необхідність вказують результати основного тестування (по перевищенню порогових значень кількостей помилок кожного рівня).

Використовуючи паралельну каскадну модель життєвого циклу ПЗ, можна показати, яке місце відводиться повторному тестуванню в життєвому циклі ПЗ (рис. 1).

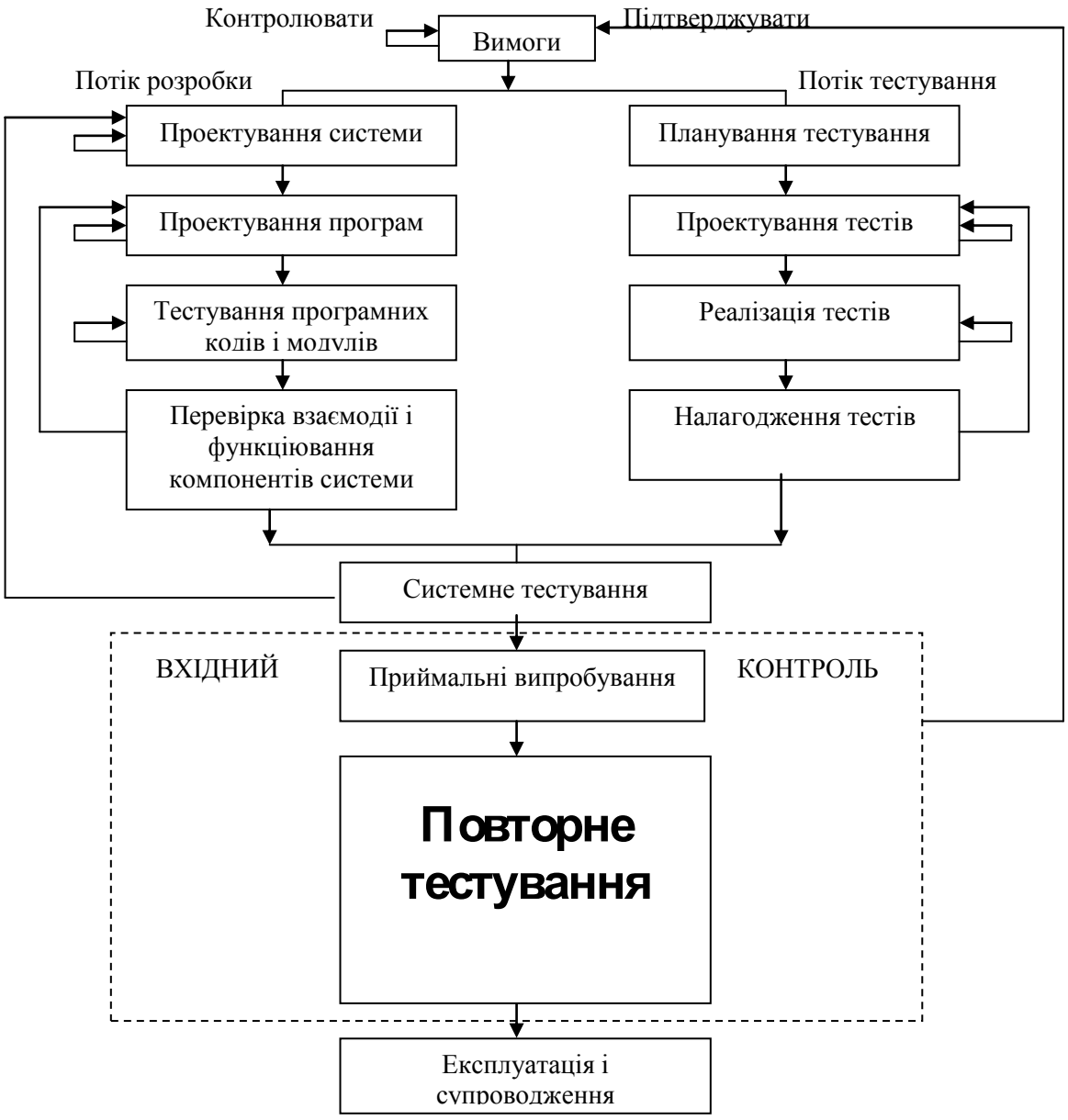

Рис.1. Місце повторного тестування в паралельній каскадній моделі життєвого циклу програмного забезпечення

Нейромережна категорійна модель процесу повторного тестування ПЗ [6-9]. Всі приховані помилки розподілимо за їх небезпечністю та ступенем впливу на ПЗ на незначні (НПП), помірні (ППП), серйозні (СПП) та катастрофічні (КПП) приховані помилки [6]. Незначним прихованим помилкам присвоїмо найнижчий рівень категорійності – перший. Помірним прихованим помилкам присвоїмо, відповідно, рівень 2; серйозним – рівень 3. Найвищим рівнем вважатимемо катастрофічний – рівень 4. Таким чином, рівнів прихованих помилок буде чотири.

В [6] вперше була запропонована концептуальна модель підвищення достовірності тестування ПЗ з виявленням прихованих помилок різних типів шляхом повторного тестування ПЗ з розподілом прихованих помилок на різні категорії і припущенням, що певна кількість помилок попередньої за серйозністю категорії призводить до появи окремих типів помилок наступної категорії, що забезпечило вибір та обгрунтування моделі процесу на базі ШНМ.

На основі запропонованої концепції повторного тестування і розподілу прихованих помилок за категорійністю з врахуванням порогів допустимої кількості помилок і важливості помилок [6, 8, 9] розроблено математичну модель процесу повторного тестування, в основі якої лежить штучна нейронна мережа (ШНМ), структура якої представлена на рис.2.

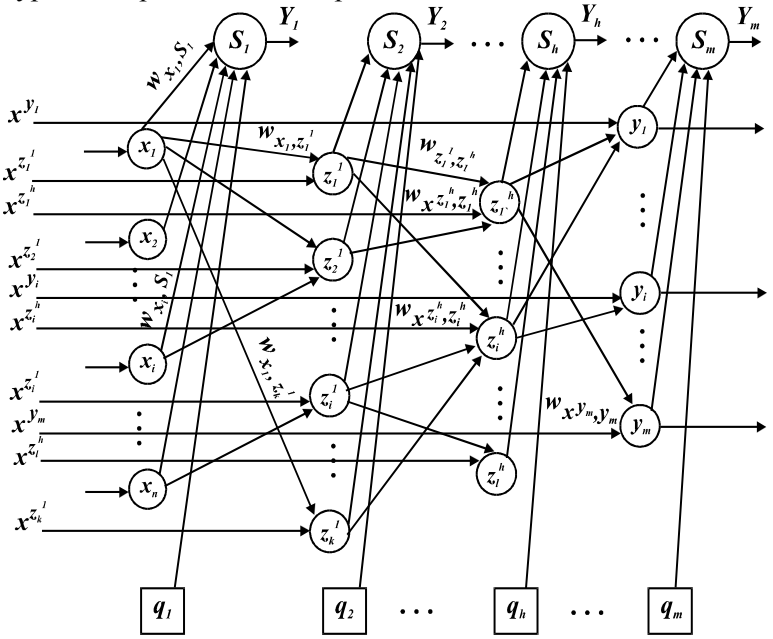

Рис.2. Нейромережна категорійна модель процесу повторного тестування

Нейромережний метод процесу повторного тестування ПЗ [10]. Початковими даними для реалізації повторного тестування є інформація про типи помилок (множина $P=\{p_k \mid k=1..n\}$), виявлених під час основного тестування, та методи (множина $M=\{m_k \mid k=1..n\}$) і операції, що були застосовані для їх виявлення (множина $O=\{o_k \mid k=1..n\}$). Ця інформація береться із звітів про результати основного тестування. Оскільки основне тестування здійснює тестувальник, то на результати тестування можливий вплив суб'єктивного та людського факторів, що може як позитивно, так і негативно впливати на ефективність повторного тестування. Саме для зменшення зазначеного суб'єктивного фактора враховуються не тільки типи виявлених помилок, а й методи та операції тестування.

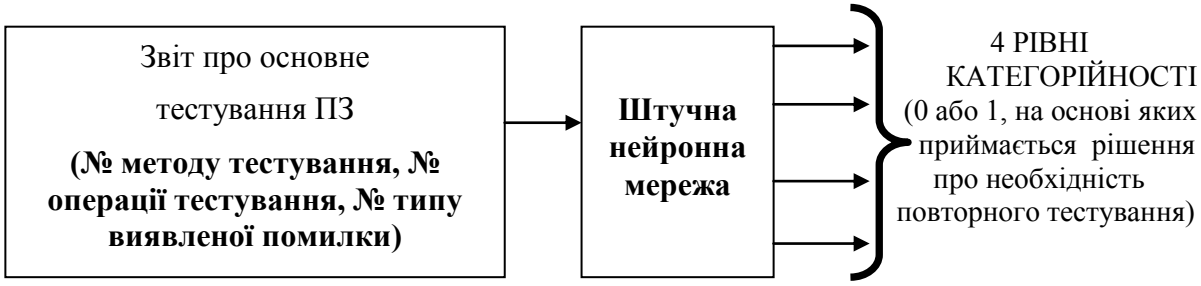

Рис.3. Принцип застосування ШНМ для процесу повторного тестування

ШНМ опрацьовує набір вхідних векторів згідно методу вирішення задачі повторного тестування та видає матрицю вихідних векторів, на основі аналізу якої робиться висновок про необхідність та тип повторного тестування.

Програмна реалізація та дослідження ШНМ в пакеті Matlab [8, 11]. В пакеті Matlab було виконано програмну реалізацію моделі ШНМ. Структурну схему імітаційної моделі ШНМ в пакеті Matlab представлено на рис.4.

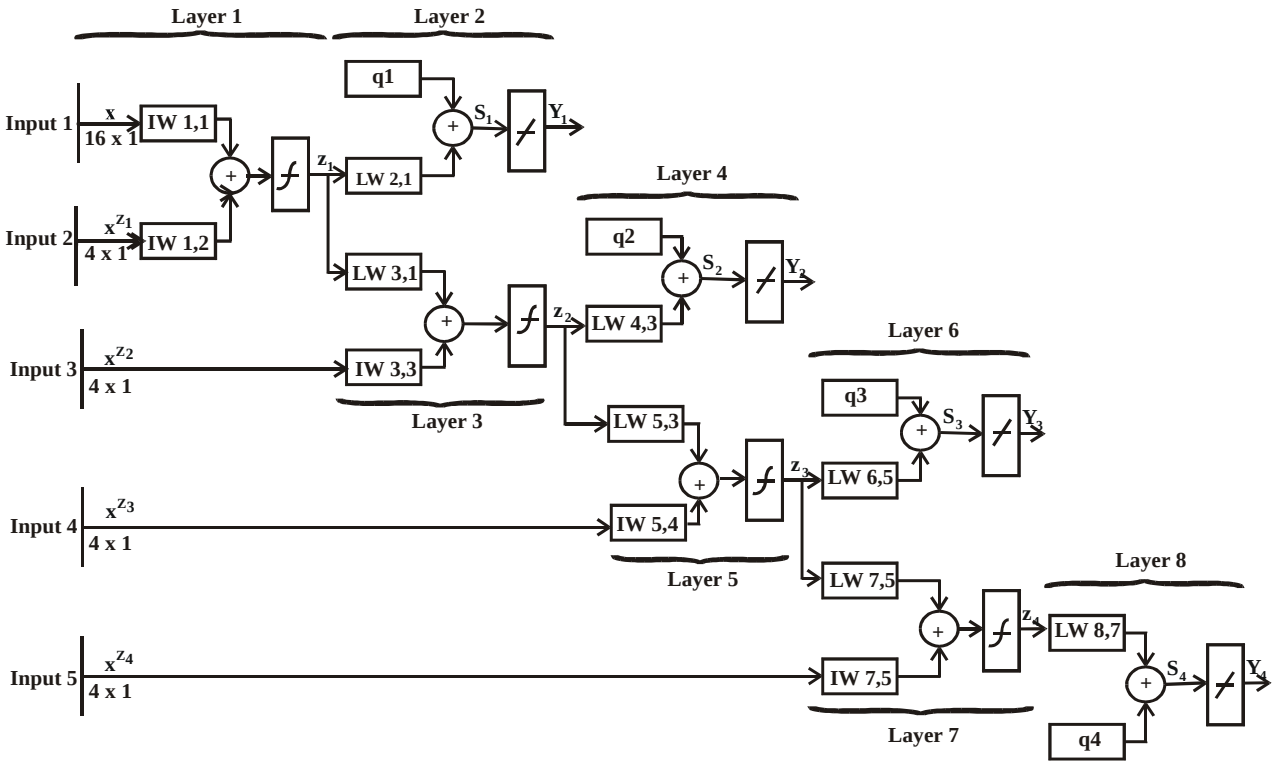

Рис.4. Структурна схема імітаційної моделі ШНМ в пакеті Matlab

На кожен з входів q_1 - q_4 потрібно подати “одиницю”, тому що тестування здійснюється одним з методів тестування, які утворюються внаслідок об’єднання двох методів тестування під одним номером, що відображено в матриці присвоєння номерів методам тестування. На входи Input2 (x^{Z_1}), Input3 (x^{Z_2}), Input4 (x^{Z_3}), Input5 (x^{Z_4}) подаються номери операцій основного тестування ПЗ. На вхід Input1 (x) подаються номери результатів операцій основного тестування ПЗ, тобто номери типів виявлених під час основного тестування помилок. Кожен з виходів Y_i відповідає за один з чотирьох рівнів категорійності (Y_1 - перший рівень категорійності, Y_2 - другий рівень категорійності, Y_3 - третій рівень категорійності, Y_4 - четвертий рівень категорійності) і приймає значення „1”, якщо штучною нейронною мережею спрогнозовано наявність в програмі помилок i-го рівня категорійності, в протилежному випадку значення виходу Y_i становить „0”.

Для вибору алгоритму навчання ШНМ та критерію оцінки якості навчання ШНМ досліджувалась при навчанні вибіркою з 2250 векторів різними алгоритмами з використанням різних критеріїв оцінки якості навчання [8, 11]. Для тестування ШНМ було побудовано тестову вибірку з 200 векторів, яка також підлягала масштабуванню.

Порогові значення кількостей помилок кожного рівня категорійності.

Оскільки уточнення підходу розподілу помилок за пріоритетами і категоріями [12] щодо опису прихованих помилок з введенням концепції категорійності помилок проведено вперше, то порогові значення кількостей помилок кожного рівня категорійності, по перевищенню яких приймається висновок про необхідність повторного тестування, в відомих літературних джерелах не описані.

Для встановлення цих порогових значень проводились дослідження кількості помилок програмного забезпечення, яке складалось з різної кількості операторів, з врахуванням і без врахування впливу помилок одного типу (рівня категорійності) на виникнення помилок наступного типу (рівня категорійності), відображені в таблиці 1.

Таблиця 1

Кількість помилок програмного забезпечення з врахуванням і без врахування впливу помилок одного типу на виникнення помилок наступного типу

	Кількість виявлених помилок без врахування взаємовпливу помилок					Реальна кількість помилок				
Кількість операторів в програмі	100	500	1000	5000	10000	100	500	1000	5000	10000
Загальна кількість помилок	10	18	25	42	68	16	24	36	42	85
Незначні	8	14	19	31	51	8	14	19	31	51
Помірні	2	4	5	11	16	5	9	14	11	33
Серйозні	0	0	1	0	1	2	1	2	0	1
Катастрофічні	0	0	0	0	0	1	0	1	0	0

Порогові значення вводяться на основі евристичних оцінок (за таблицею 1):

1. якщо кількість помилок 4-го рівня категорійності (катастрофічних) перевищує 1, то повторне тестування необхідне за причини можливості відмови програмної системи;

2. якщо кількість помилок 3-го рівня категорійності (серйозних) перевищує 2, то повторне тестування необхідне за причини виникнення помилок вищого рівня категорійності;

3. якщо кількість помилок 2-го рівня категорійності (помірні) дорівнює або перевищує 50% від загальної кількості виявлених під час основного тестування помилок, то повторне тестування необхідне за причини виникнення помилок вищих рівнів категорійності;

4. якщо кількість помилок 1-го рівня категорійності (незначні) дорівнює або перевищує 75% від загальної кількості виявлених під час основного тестування помилок, то повторне тестування необхідне за причини виникнення помилок вищих рівнів категорійності.

При аналізі таблиці 1 видно, що при перевищенні саме таких значень виникають приховані помилки більш високих рівнів категорійності, тому ці значення використовуються в якості порогових при формуванні висновку про необхідність повторного тестування прикладного програмного забезпечення.

Програмні засоби для реалізації процесу повторного тестування ПЗ. На основі нейромережного методу процесу повторного тестування розроблено структуру та виконано програмну реалізацію системи визначення необхідності повторного тестування, яка на основі звіту про основне тестування ПЗ дає висновок про необхідність повторного тестування ПЗ на основі прогнозу наявності прихованих помилок в аналізованому ПЗ, тобто дає можливість замовнику оцінити якість одержуваного ПЗ (рис.5).

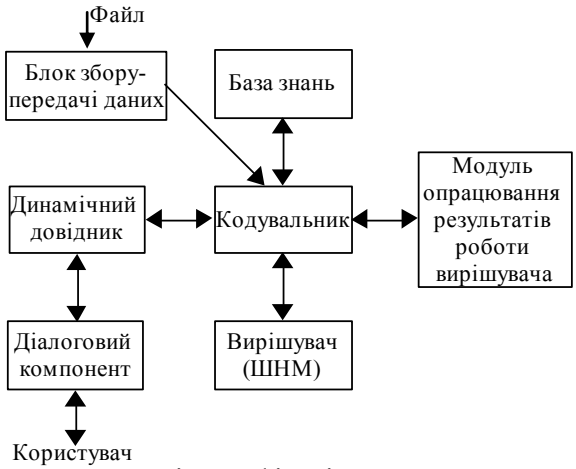

Рис.5. Структурна схема системи ідентифікації прихованих помилок ПЗ

Система ідентифікації прихованих помилок ПЗ складається з наступних компонентів: 1) блок збору – передачі даних – підключає наданий користувачем файл з результатами основного тестування, представленими у вигляді журналу “Метод тестування – Операція тестування – Тип виявленої помилки”; 2) кодувальник – виконує перетворення вхідних даних з лінгвістичної форми представлення в

кількісну форму, формування вхідних векторів для модуля вирішувача та передачу результуючих векторів вирішувача модулю опрацювання результатів роботи вирішувача; 3) база знань – містить допоміжні таблиці та правила для формування висновку про необхідність повторного тестування; 4) вирішувач – штучна нейронна мережа, на входи якої подається інформація про методи і операції основного тестування та типи виявлених під час основного тестування помилок, а на виході одержується рівень категорійності прихованих помилок; 5) модуль опрацювання результатів роботи вирішувача – на основі правил та таблиць бази знань генерує висновок про необхідність повторного тестування; 6) динамічний довідник – надає користувачу довідку про формат вхідного файлу, про відомі системі методи, операції та типи помилок основного тестування ПЗ; 7) діалоговий компонент – візуалізує повідомлення динамічного довідника та видає їх користувачу в зрозумілій для сприйняття формі.

Запропонована система ідентифікації прихованих помилок програмного забезпечення дозволяє користувачу, на основі звіту про результати основного тестування, одержати висновок про необхідність повторного тестування, а також про наявність у програмному забезпеченні прихованих помилок та їх рівень категорійності. Результат роботи системи визначення необхідності повторного тестування ПЗ виводиться у лінгвістичній формі: "Повторне тестування не потрібне, оскільки жодне порогове значення не досягнуте" або "Повторне тестування необхідне по досягненню певного порогового значення".

Систему ідентифікації прихованих помилок ПЗ було реалізовано в Borland C++ Builder 6.0.

Висновки. Експертиза програмного забезпечення полягає в тому, що замовник повинен переконатись, що розробник виконав поставлені вимоги і розробив якісний безпомилковий продукт із заданою функційністю. Важливою метою роботи експерта в процесі проведення оцінки (експертизи) ПЗ є перевірка відповідності ПЗ встановленим вимогам та здійснення впливу на підвищення якості, достовірності і надійності розробленого ПЗ. Для цього експерти можуть використовувати проведення додаткових незалежних випробувань.Одним з таких випробувань для проведення оцінки замовником ПЗ з метою виявлення прихованих помилок може бути визначення необхідності повторного тестування програмного забезпечення на основі прогнозування наявності прихованих помилок ПЗ та встановлення їх небезпечності та ступеня впливу на ПЗ. Повторне тестування допомагає замовнику оцінити якість програмного забезпечення, яке приймається, та якість тестування цього ПЗ.

Література

1. http://creograf.ru/?messPress_ShowR_161=1
2. Скляр В.В. Оценка качества и экспертиза программного обеспечения. Лекционный материал. - Харьков: НАУ "ХАИ", 2008. - 204 с.
3. Ястребенецкий М.А., Васильченко В.Н., Виноградская С.В. и др. Безопасность атомных станций: Информационные и управляющие системы. - К.: Техніка, 2004. - 472 с.
4. Сидельников Ю.В. Экспертология - новая научная дисциплина // Автоматика и телемеханика. - 2000. - №2. - С.107-126
5. Харченко В.С., Скляр В.В., Гордеев А.А. Верификация программного обеспечения. - Харьков: НАУ "ХАИ", 2006. - 132 с.
6. Локазюк В.М., Пантелєєва (Говорущенко) Т.О. Категорійна модель процесу повторного тестування дефектів програмного забезпечення // Вісник Технологічного університету Поділля – Хмельницький: ТУП, 2004. – ч.1, т.1, с. 53 – 58.
7. Lokazyuk V.M., Govoruschenko T.O. Category Model of Process of Repeated Software Testing // Proceedings of the Third IEEE Workshop on Intelligent Data Acquisition and Advanced Computing Systems : Technology and Applications. – Sofia, Bulgaria, 2005. – p.241-245
8. Говорущенко Т.О. Підвищення достовірності тестування програмного забезпечення // Вісник Національного університету “Львівська політехніка” “Комп’ютерні науки та інформаційні технології” – Львів: Видавництво Національного університету “Львівська політехніка”, 2007 – с.186-196
9. V.Lokasyuk, O.Pomorova, T.Govorushchenko. Neural Nets Method for Estimation of the Software Retesting Necessity // Proceedings of the 2008 international workshop on Software Engineering in east and south Europe – Germany, Leipzig, 2008. – pp. 9-14.
10. Говорущенко Т.О. Система повторного тестування програмного забезпечення // Радіоелектронні і комп’ютерні системи – Харків: НАУ “ХАІ”, 2005. - №4, с.120-126
11. Говорущенко Т.О. Дослідження моделі вирішувача системи повторного тестування прикладного програмного забезпечення // Вісник Хмельницького національного університету – Хмельницький: ХНУ, 2007 - №3, т.1, с.236-244
12. Калбертсон Р., Браун К., Кобб Г. Быстрое тестирование: Пер. с англ. – М.: Издательский дом “Вильямс”, 2002. – 384 с.