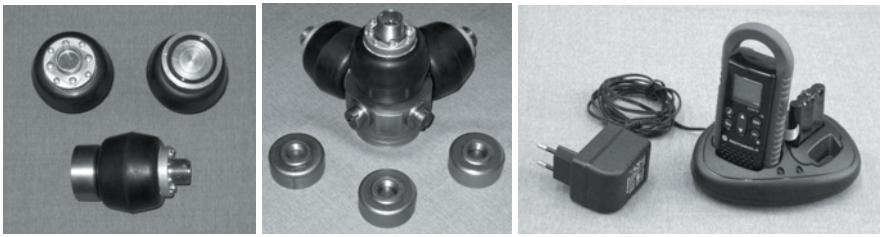
несколько этапов модернизации экспериментального программного обеспечения для его адаптации к выявленным наиболее существенным особенностям генерации, распространения и регистрации полезных данных и мощных сопутствующих им сигналов. Идет изучение особенностей вскрытых сочетаний известных акустических явлений, разработка и пробы алгоритмов оценки соответствующих им параметров с целью дальнейшего количественного контроля этих параметров и дистанционной оценки состояния металла.



а) б) в)

Рис.4. Вибродатчики ВДМ-3 с магнитными держателями (а и б), радиопередатчик синхросигнала (в)

1. *Владимирский А.А., Владимирский И.А.* Разработка структуры экспериментальной системы активно-пассивного низкочастотного диагностирования состояния трубопроводов. Зб. наук. пр. ІПМЕ НАН України. Вип. 64, Київ, 2012р.-с.55-57.

Поступила 18.03.2013р.

УДК 004.4:006.015.5 (045)

І.Е. Райчев, м. Київ

ПРИНЦИПИ ТЕСТУВАННЯ КРИТИЧНИХ ПРОГРАМНИХ СИСТЕМ ПРИ ЇХ СЕРТИФІКАЦІЇ

Abstract. In the article the questions of creating procedures and scripts of testing of critical program systems are considered. The obtained procedures of testing allow to generate optimal test data sets that enables to increase efficiency of tests, to reduce input of certification and to raises reliability of her results.

Вступ

При сертифікаційних випробуваннях критичних програмних систем (ПС), що проводяться з метою забезпечення необхідного рівня їх якості, здійснюється всеосяжне тестування на області визначення вхідних даних домена проблемної галузі [1–3]. Тестування використовують для визначення

показників якості ПС [4–6]. Проблема, що виникає при випробуваннях ПС, полягає в тім, що відомі стратегії і процедури тестування орієнтовані на повне покриття елементів класів еквівалентності по вхідним даним і є трудомісткими й витратними. При цьому використовувані методи часто дублюють один одного, а тестові набори даних (ТНД) перетинаються на множинах областей визначення реалізованих функцій ПС [7–10].

Для вирішення цієї проблеми необхідно розробити підходи до побудови методології тестування, яка була б оптимальною для кожного з класів критичних ПС. Методи тестування мають покривати відомі класи еквівалентності досліджуваного домена проблемної галузі, але покриття не повинне бути надлишковим. Крім того, необхідно досягти високої здатності набору сценаріїв тестування (стратегії) до виявлення ще не знайдених помилок у ПС, тобто високої детектабільності методології тестування [11,12].

Як приклад розглянемо клас програмних систем контролю польотів літальних апаратів (ЛА), що є критичними, бо пов'язані з безпекою життєдіяльності. Вхідною областю даних для ПЗ автоматизованих систем контролю польотів (ПЗ АСКП) є параметрична польотна інформація (ПІ), що містить структуровані дані, які відображають динаміку поведінки ЛА. За допомогою аналізу ПІ вирішуються задачі контролю режимів польоту, перевіряється додержання правил льотної експлуатації, оцінюється працездатність ЛА і визначаються причини авіаційних інцидентів.

Вимоги до функціональності ПЗ АСКП викладені у галузевому стандарті [13], а питання специфікації вимог до безпеки критичних систем розглянуті в роботі [14]. Перевірка відповідності властивостей ПЗ вимогам проводиться при сертифікації. Питання сертифікації ПЗ АСКП розглянуті в роботах [15,16]. Однією із найбільш вагомих проблем під час випробувань є організація тестування, що слугує для визначення фактичних значень показників якості ПЗ. Результати недбалого тестування ПЗ систем контролю можуть бути досить серйозні, оскільки помилкові результати оцінки стану об'єкта контролю призводять до катастрофічних наслідків.

1. Побудова специфікацій вимог безпеки ПЗ критичних систем

Основною задачею, що вирішується при тестуванні, є перевірка повної відповідності ПЗ вимогам. Це особливо важливо для систем критичного призначення, невідповідність яких вимогам призводить до небезпечних ситуацій. Вимоги до безпеки ПЗ критичних систем будемо визначати шляхом аналізу потенційних небезпек і ризиків, що можуть виникнути при роботі системи. На основі цього аналізу треба конкретизувати вимоги до ПЗ таким чином, щоб або усунути ці небезпеки і ризики, або зменшити їхній вплив.

Цю процедуру можна виконати, базуючись на концепції “безпечного” життєвого циклу ПЗ, яку запропоновано у стандарті IEC 61508 [17]. Життєвий цикл критичних систем для ПЗ складається з визначення можливостей і предметної галузі системи, аналізу небезпек та ризиків, розподілу вимог по підсистемам і специфікації вимог безпеки.

Аналіз небезпек і ризиків складається з аналізу системи і її оточення. В результаті виявляються потенційні небезпеки, виконується класифікація небезпек і вибираються ті з них, які є потенційно найбільш небезпечними і ймовірними. Для небезпек, отриманих у результаті вибору, аналізуються можливі причини їхнього виникнення, наприклад, будується дерево відмов [18]. Після цього оцінюються ризики, пов'язані з виявленими небезпеками.

На наступному етапі аналізу виявляємо небезпеки і ризики, що можуть бути викликані неправильним функціонуванням ПЗ. Аналізуючи механізми реалізації небезпек, формулюємо вимоги до ПЗ для попередження виникнення небезпек і зменшення імовірності ризиків. Для ПЗ контролю польотів сукупність небезпек і ризиків можна описати у виді неправильного визначення стану ЛА в контрольованих ситуаціях. З аналізу системи визначаємо, що потенційно серйозною небезпекою, що може виникнути через неякісне ПЗ, є помилкове визначення стану ЛА, як об'єкта контролю. При цьому можливі два варіанти помилкових рішень: $\alpha = S_i\left(\overline{x_i} \in X / x_i \in \overline{X}\right)$ і $\beta = S_i\left(\overline{x_i} \in \overline{X} / x_i \in X\right)$. Тут $\overline{x_i}$, x_i – дійсне й обчислене значення вектора контрольованих параметрів у момент часу t_i, X, $\overline{X}$ – припустима і неприпустима області значень, S_i – алгоритм i-ї події контролю.

Оскільки ці рішення можуть реалізуватися з визначеною імовірністю, то, з огляду на технічні й економічні фактори, накладаються обмеження: $P(\alpha) \le \alpha_d$ і $P(\beta) \le \beta_d$. Тут α_d і β_d – досить малі граничні значення імовірностей подій. Такі обмеження можна трактувати як вимоги до ПЗ системи контролю. Відзначимо, що $P(\alpha)$ і $P(\beta)$ є метриками атрибута "вірогідність контролю", який можна віднести до підхарактеристики "точність" характеристики "функціональність" моделі якості ПС [5].

Розглянемо використання цього підходу при проектуванні процедур тестування ПЗ автоматизованих систем контролю польотів.

2. Раціональне застосування класів еквівалентності при тестуванні критичних програмних систем

Як показано в п.1, головною задачею, яку вирішують більшість критичних ПС, є діагностування стану об'єкта контролю (у нашому випадку – стану ЛА). Тому в процесі тестування програмних комплексів контролю важливу роль відіграє поняття алгоритмів контролю.

Основними програмними комплексами ПЗ АСКП є три комплекси. У комплексі програм відтворення польотної інформації реалізовані алгоритми попередньої обробки параметричної інформації і перетворення кодових значень аналогових параметрів (АП) у фізичні величини. У комплексі допускового контролю на множині АП, котрі описують траєкторію руху ЛА як динамічної системи, реалізовані алгоритми обчислення складних логічних функцій (алгоритми контролю), що описують контрольовані ситуації.

Комплекс програм контролю якості виконання польоту реалізує алгоритми відтворення значень параметрів у контрольованих точках польоту з метою обчислення показників якості виконання елементів польоту.

У роботах [19, 20] запропонована і розглянута комплексна стратегія тестування модулів ПЗ АСКП, як критичної системи. Вона складається з еквівалентної розбивки (метод 1), методу аналізу граничних значень (метод 2), комбінаторного покриття умов (метод 3), динамічного стохастичного тестування (метод 4) і методу припущення про помилку (метод 5). Покажемо, що така стратегія дає можливість провести більш повне й ефективне тестування модулів ПЗ контролю польотів і забезпечити виявлення максимальної кількості невідповідностей специфікаціям у порівнянні з використанням кожного з методів окремо. Особливо важливим при тестуванні множини алгоритмів контролю критичних ПС є раціональне використання еквівалентної розбивки і комбінаторного покриття умов. Ефективне застосування цих методів дозволяє усунути дефекти логіки у модулях ПС.

Розглянемо більш детально особливості застосування класів еквівалентності при тестуванні ПЗ АСКП. Алгоритми контролю (події) складаються з умов. Кожен предикат займає одну позицію в алгоритмі контролю, а кожна умова є логічною функцією від, як правило, більш ніж одного предиката. Для довільної події контролю S_k можна визначити класи еквівалентності, причому для кожного предиката визначається свій клас еквівалентності. Наприклад, для події $S073 = (7120 \le H \le 10300) \wedge V \ge 588$ визначаються три класи еквівалентності по параметрі висота барометрична (виміряється в метрах): $X^{(1)}$ $(7120 \le H \le 10300)$, $X^{(2)}$ $(H < 7120)$, $X^{(3)}$ $(H > 10300)$, і два класи для швидкості приладової (виміряється в км/год): $X^{(4)}$ $(V \ge 588)$ і $X^{(5)}$ $(V < 588)$. Тут $X^{(1)}$ і $X^{(4)}$ – правильні, а інші – неправильні класи еквівалентності, як показано в таблиці для визначення класів еквівалентності при тестуванні ПЗ АСКП.

Вхідні умови	Правильні класи еквівалентності	Неправильні класи еквівалентності
Обмеження на висоту польоту	Висота барометрична більше чи дорівнює 7120м і менше чи дорівнює 10300м $(7120 \le H \le 10300)$ $X^{(1)}$	$(H < 7120)$ $X^{(2)}$ $(H > 10300)$ $X^{(3)}$
Обмеження на швидкість польоту	Швидкість приладова більше чи дорівнює 588 км/год $(V \ge 588)$ $X^{(4)}$	$(V < 588)$ $X^{(5)}$

Оскільки розглянуті комплекси записуються на мовах програмування високого рівня, то умови контролю кодуються за допомогою операторів if, if ... then, if ... then ... else, do ... while, і while. Якщо умови в операторах записані вірно, то для правильних вхідних даних (елементів із правильних вхідних

класів еквівалентності) одержимо правильні результати (очікувані дані з правильних вихідних класів еквівалентності). Звідси випливає, що метод комбінаторного покриття умов стратегії "білої скриньки" (метод 3) сполучається з методом еквівалентної розбивки стратегії "чорної скриньки" (метод 1). Ці методи взаємно доповнюють один одного. Дійсно, фрагмент програми для події S073 швидше за все буде виглядати так: if((H<=10300) && (H>=7120) && (V>=588)) then S073 =1 else S073 =0;

Якщо вхідні дані не попадають в області правильних класів еквівалентності, то подія контролю не має місця. Покриваючи умови в операторі за допомогою ТНД (метод 3), ми використовуємо набори елементів $x_i^{(j)}$, що належать класам $X^{(1)}$, $X^{(2)}$, $X^{(3)}$, $X^{(4)}$, $X^{(5)}$ (метод 1). Тут j – номер класу (j=1,2...,5), а i – номер елемента в ньому (i=1,2...,n). Тому, для перевірки правильності функціонування програм контролю досить для кожного алгоритму контролю скористатися декількома (двома-трьома) ТНД, побудованими відповідно до методів 1 і 3. Слід, також, скористатися методом аналізу граничних значень.

Тоді, якщо в нашому прикладі S073 =1 для деякого набору вхідної послідовності, той і інші набори вхідних даних повинні приводити до рішення S073 =1, якщо набір складається з елементів тих же класів. Однак, це далеко не очевидне твердження. Не завжди виявляються помилки тільки одного типу для елементів класу еквівалентності. Реально різні елементи введеного класу можуть виявити два (рідше, три і більш) типів помилок.

Дійсно, наприклад, при кодуванні події контролю S073 для умови $V \geq 588$ (класи $X^{(4)}$ і $X^{(5)}$) замість >= може бути записаний > (1-й вид помилок), чи <= (2-й вид помилок), і навіть V>=598 чи V>=578 (3-й вид помилок), замість правильної умови V>=588.

1-й вид помилок виявиться, швидше за все, тільки при тестуванні по методу граничних значень, 2-й – при використанні методу еквівалентних розбивок, але 3-й вид може бути виявлений тільки з застосуванням декількох тестових умов: вибірок із правильних і неправильних класів еквівалентності.

Тому в даній роботі пропонується використовувати не більш 3-5 значень для кожного класу з двосторонніми обмеженнями: одне значення посередині і по одному-двох ближче до границь (але не втручаючись у метод граничних значень). Для однобічного обмеження досить використовувати два значення. Наприклад, для швидкості граничними значеннями будуть 585, 588, 591 (ціна одиниці коду параметра дорівнює 3 км/год). Тоді при тестуванні по методу еквівалентної розбивки для класу $X^{(4)}$ варто вибрати 594 і 600, а для $X^{(5)}$ – 582 і 576 (тобто не слід вибирати значення, що віддалені більш ніж на 10-20 одиниць коду АП від границі класу).

З вищевикладеного випливає, що для правильних і неправильних відкритих класів еквівалентності не слід вибирати занадто віддалені від границі значення (це не забезпечить виявлення додаткових помилок в

умовах). Замість цього варто зайнятися пошуком верхнього або нижнього обмеження для відкритої підобласті вхідних даних класу еквівалентності.

3. Використання моделей детектабільності для оцінки стратегії тестування програмних систем контролю

Розглянемо можливість використання ***моделей детектабільності*** для оцінки стратегій тестування. Оскільки для тестування ПС застосовують різні стратегії, то необхідно дослідити питання формування критеріїв вибору найкращих методів тестування с точки зору їхньої здатності виявляти помилки. До надійності ПЗ критичних систем висуваються вимоги на порядок вище, ніж вимоги до ПЗ загального призначення. Однак, надійність ПЗ високоцілісних систем [11] через їхню складність далеко не завжди має задовільні показники і немає повної впевненості, що система не містить дефектів, бо традиційне практичне тестування не в змозі забезпечити необхідний рівень якості. Акцентуємо увагу на наступних проблемах:

1) відсутність достовірної інформації про те наскільки ефективно застосовані методи тестування виявляють помилки;
2) наскільки одна стратегія тестування краще (ефективніше) іншої.

На практиці для оцінки ефективності застосованих методів частіше використовують евристичні підходи. У роботах [11,12], навпроти, для оцінки методів тестування використовується модель, що дозволяє одержати числову міру ефективності кожного методу і стратегії в цілому.

Для вибору найкращих методів з існуючого спектра використаємо поняття детектабільності [11], тобто здатності цих методів до виявлення помилок чи невідповідностей специфікаціям. Цей інструмент застосуємо для визначення того факту наскільки один метод краще іншого. Як міру застосуємо P -міру, запропоновану в [11], тобто імовірність виявлення хоча б одного дефекту в програмі.

Більшість відомих стратегій тестування мають загальну властивість: вхідна область даних розбивається на підобласті, усередині якої вибирається одне або кілька тестових даних. Один з методів вибору елемента підобласті $x_j^i \in X^{(i)}$ – випадковий (наприклад, відповідно до рівномірного розподілу).

При аналізі багатокритеріальних ситуацій контролю легко бачити, що повна вхідна область даних програми D компонується із сукупності вхідних областей параметрів D_h, що входять у формули алгоритмів контролю. Шукана міра P для будь-якої події контролю в цьому випадку може бути визначена зі співвідношення $P=P_1 \cdot P_2 \cdot \ldots \cdot P_h \cdot \ldots \cdot P_H$, де H – кількість предикатів в алгоритмі контролю $(h=\overline{1,H})$, а P_h – міра для параметра h.

Оскільки кожен параметр має свою область визначення, будемо вважати, що міри P_h взаємно незалежні. Введемо наступні позначення. Через D_{hi} позначимо підобласть тестування з області D_h ($i=\overline{1,k}$, k – кількість підобластей для параметра h). Через дискретність кодів АП (ціна одиниці

коду), легко бачити, що кожна підобласть має скінченну кількість входів. Розмір області позначимо d, розміри підобластей (кількість точок входу чи тестових даних) позначимо d_i, коректні входи – c_i, дефектні – b_i, n_i – число обраних входів (елементів ТНД) для i-ї підобласти, а n – загальна кількість входів для області D_h. Очевидно, що $c_i = d_i - b_i$, а інтенсивність відмов для підобласті $\theta_i = b_i / d_i$. Тоді, якщо p_i частота влучання елемента в D_{hi} при випадковому його виборі з D_h, то інтенсивність відмов програми θ для тестових елементів по параметрі h виміряється по формулі :

$$\theta = \sum_{i=1}^{k} p_i \cdot \theta_i \qquad (1)$$

При цьому тестові елементи вибираються незалежно і відповідно до рівномірного розподілу. Будемо оцінювати ефективність стратегії тестування відповідно до сукупного значення детектабільності методів, що її складають. Дослідимо проблеми, що виникають при оцінці ефективності стратегій на прикладі запропонованої стратегії тестування модулів контролю, для якої вважаємо, що перші чотири методи базуються на роздільному тестуванні, тобто на тестуванні підобластей, у випадку, коли підобласті складають повну вхідну область для розглянутого параметра і не перетинаються між собою. Метод припущення про помилку є додатковим. Він використовує емпіричні критерії і не погіршить загальну оцінку стратегії в цілому.

Перші два методи (еквівалентна розбивка і метод граничних значень) будемо оцінювати з залученням моделі детерміністичного роздільного тестування. Для третього і четвертого методів (комбінаторне покриття умов і динамічне стохастичне тестування) оберемо систематичне тестування підобластей з m тестовими випадками для кожної підобласті (m фіксується) [11]. Вище показано, що метод комбінаторного покриття умов сполучається із методом еквівалентної розбивки (вони базуються на підобластях), а динамічне стохастичне тестування автори рекомендують застосовувати для підобластей граничних значень. При стохастичному тестуванні ТНД генеруються у виді множин випадкових величин із заданими математичним сподіванням і розподілом. При цьому для виявлення дефектів логіки і перевірки точності обчислень можна застосовувати невелику кількість коротких вибірок.

Для детерміністичної моделі маємо формулу:

$$P_{dp} = 1 - \prod_{i=1}^{k} (1 - \theta_i)^{n/k} \qquad (2)$$

для n обраних випадків і k підобластей (n і k фіксовані). У цій моделі передбачається, що для кожної підобласті випадково (наприклад, відповідно до рівномірного розподілу) вибирається однакова кількість елементів (n/k).

Для систематичного тестування, якщо кількість елементів n_i

вибирається випадково для кожної підобласті, міру детектабільності оцінюємо як:

$$P_s = 1 - \prod_{i=1}^{k} (1 - \theta_i)^{n_i} \tag{3}$$

Однак у цьому випадку важко оцінювати ефективність моделі, навіть якщо $n = \sum_{i=1}^{k} n_i$. Тому приймемо $n_i = n/k = m$, що в середньому не змінить величину оцінки. Обрані моделі порівняємо зі статистичним роздільним тестуванням і випадковим тестуванням. Міра детектабільності статистичного тестування обчислюється відповідно до співвідношення:

$$P_{sp} = 1 - (1 - \sum_{i=1}^{k} \frac{1}{k} \cdot \theta_i)^n, \tag{4}$$

тому що $p_i = \frac{1}{k}$. Статистичне роздільне тестування підобластей допускає випадковий вибір підобласті і далі випадковий вибір тестового елемента в ній. При використанні ж випадкового тестування не враховується розподіл на підобласті при виборі тестового даного, а дані вибираються з повної області D_h для кожного параметра, що приводить до наступної міри:

$$P_r = 1 - (1 - \sum_{i=1}^{k} p_i \cdot \theta_i)^n \tag{5}$$

Виконаємо сукупну оцінку для перших двох методів на прикладі предиката $7120 \le H \le 10300$ з події S073 (див. табл. 1). Для того, щоб покрити всю область визначення, до трьох перших класів (еквівалентна розбивка) додамо ще два класи $X^{(6)}$ і $X^{(7)}$, які будуть служити для тестування граничних значень, що показано на наступному рисунку:

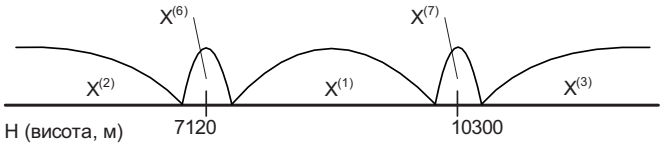


Рис.1. Еквівалентна розбивка і метод граничних значень для предиката $7120 \le H \le 10300$

Класи $X^{(1)}$, $X^{(2)}$, $X^{(3)}$ звузимо на 4, 1 і точку відповідно. З огляду на дискретність фізичних значень параметрів, це будуть величини: 7120, 7180, 10240, 10300 для $X^{(1)}$, і 7060 і 10360 для $X^{(2)}$ і $X^{(3)}$, причому 7060, 7120, 7180 ввійдуть у клас $X^{(6)}$, а 10240, 10300 і 10360 - у клас $X^{(7)}$. Для цих п'яти класів побудуємо вибірку з 15 значень (3 значення для кожного класу: k=5, n=15, m=3). Нехай для послідовності підобластей $X^{(2)}, X^{(6)}, X^{(1)}, X^{(7)}, X^{(3)}$ інтенсивності відмов складають: $\theta_1 = 0.05, \theta_2 = 0.33, \theta_3 = 0.02, \theta_4 = 0.33, \theta_5 = 0$

відповідно (реальний тестовий випадок, що мав місце при тестуванні ПЗ АСКП у процесі сертифікаційних випробувань).

Оцінимо міру для статистичного тестування, що визначиться згідно (4):

$$P_{sp}=1-(1-\frac{1}{5}\cdot(0.05+0.33+0.02+0.33))^{15}=1-(1-0.146)^{15}=1-0.094=0.906\ .$$

Для моделі (2): $P_{dp}=1-(0.95^3\cdot 0.67^3\cdot 0.98^3\cdot 0.67^3)=1-0.073=0.927$. Звідси робимо висновок, що модель, що перебирає послідовно всі підобласті, має кращу детектабільність. Оцінка міри систематичного тестування P_s з (3) для третього і четвертого методів при зроблених вище припущеннях збігається з розрахованим значенням P_{dp}. Розрахуємо тепер міру P_r для випадкового тестування, вважаючи, що частоти влучень при рівномірному розподілі для наших підобластей $p_1=0.174, p_2=0.035, p_3=0.581, p_4=0.035, p_5=0.174$. Шукану міру згідно (5) обчислимо таким чином:

$P_r=1-(1-(0.174\cdot 0.05+0.035\cdot 0.33+0.581\cdot 0.02+0.035\cdot 0.33))^{15}=1-(1-0.043)^{15}=1-0.514=0.486$

Для стратегії тестування модулів допускаємо, що залишилося $1-0.927=0.073$ частки помилок. Тоді вірогідність їхнього виявлення складе $0.073\cdot 0.927=0.067$. Таким чином, міра розробленої стратегії $P_{\text{mod}}=0.927+0.067=0.994$, а міра стратегії методів випадкового тестування $P_{ran}=0.906+0.046=0.952$ (оскільки $1-0.906=0.094$, а $0.094\cdot 0.486=0.046$). Тому отримана стратегія є кращою згідно оцінки міри детектабільності. Помітимо, що $P_{dp}\geq P_s\geq P_{sp}\geq P_r$, а $P_s(m)=P_{dp}$, якщо $m=n/k$.

Якщо ж оцінювати обрані методи окремо, то для оцінки стратегії в цілому необхідно використовувати детерміністичне тестування для еквівалентної розбивки, потім систематичне тестування підобластей для методу граничних значень, далі знову (2) для комбінаторного покриття умов і (3) для стохастичного тестування. Для порівняльного аналізу можна використати (4), тому що випадкове тестування вимагає повну вхідну область. Отримані в цьому випадку при тестуванні ПЗ АСКП оцінки мають низькі показники.

Висновки

З вищевикладеного випливає, що сполучаючи методи створення тестів у складений метод (наприклад, еквівалентна розбивка + граничні значення, чи комбінаторне покриття умов + динамічне стохастичне тестування), ми одержуємо збільшення міри детектабільності за рахунок збільшення кількості підобластей і тестових елементів у них. Показано, що включення в стратегію тестування критичних ПС великої кількості методів невиправдане, й веде до зайвих витрат, істотно не збільшуючи сукупну детектабільність стратегії.

На основі аналізу області застосування ПЗ АСКП обґрунтований вибір тих методів тестування, які при відсутності зайвого дублювання даних у ТНД

забезпечують виявлення найбільшої кількості дефектів у модулях ПЗ. При оцінці стратегії отримані високі показники імовірності виявлення помилок.

1. ДСТУ 2462–94. Сертифікація. Основні поняття. Терміни та визначення. – Чинний від 01.01.95. –К.: Держстандарт України, 1994. – 27 с.
2. ДСТУ 2853–94. Програмні засоби ЕОМ. Підготовлення і проведення випробувань. – Чинний від 01.01.96. –К.: Держстандарт України, 1994. – 17 с.
3. ДСТУ 2851–94. Програмні засоби ЕОМ. Документування результатів випробувань. – Чинний від 01.01.96. –К.: Держстандарт України, 1994. – 12 с.
4. ISO/IEC 12119. IT – Software packages – Quality requirements and testing, 1994. – 29 p.
5. ISO/IEC 9126-1. Software engineering – Product quality – Part 1: Quality model, 2001. – 26 p.
6. Основы инженерии качества программных систем / *Ф.И.Андон, Г.И.Коваль., Т.М.Коротун, Е.М.Лаврищева, В.Ю.Суслов.* –К.: Академпериодика, 2007. –672 с.
7. *Брауде Э.* Технология разработки программного обеспечения. – СПб.: Питер, 2004. – 655 с.
8. *Канер Сэм, Фолк Дж., Нгуен Енг Кен*. Тестирование программного обеспечения : Пер. с англ. –К.: Диасофт, 2001. – 544 с.
9. *Майерс Г*. Искусство тестирования программ :Пер. с англ. –М.: Мир, 1982. – 176с.
10. *Дастин Э., Рэшка Д., Пол Дж*. Автоматизированное тестирование программного обеспечения. Внедрение, управление и эксплуатация : Пер. с англ. –М.: Изд. “Лори”, 2003. – 567с.
11. *Мороз Г.Б.* Надежность и трастабильность программных средств высокоцелостных систем // УСиМ. – 1999. – №2. – С.59–68.
12. *Мороз Г.Б.* Некоторые случаи оптимизации трастабильности программных средств // Проблемы программирования. – 2000. – №1-2. – С.361–366.
13. ДСТУ 3275–95. Системи автоматизованого оброблення польотної інформації наземні.Загальні вимоги.–Чинний від 01.07.96. –К.:Держстандарт України, 1996. –7с.
14. *Замковий В.В., Харченко О.Г., Галай І.О.* Використання моделей якості для специфікації вимог при проектування програмних систем // Зб. наукових праць Камя’нець-Подільського національного університету. – К-Пд.: К-Пд.НУ. –2008. – №11. – С. 131–137.
15. *Райчев І.Е.* Проблеми сертифікації програмного забезпечення автоматизованих систем контролю // Вісник НАУ. –2004. –№1. –С. 23–28.
16. *Райчев І.Е., Харченко О.Г*. Концепція побудови сертифікаційної моделі якості програмних систем // Проблемы программирования. –2006. –№2-3. – С. 275–281.
17. IEC 61508 : Functional Safety of electrical / electronic / progammable electronic safety-related systems.
18. *Соммервилл И.* Инженерия программного обеспечения : Пер. с англ. – М.: Вильямс, 2002. –624с.
19. *Райчев И.Э., Харченко А.Г., Яцков Н.А*. Исследование методов тестирования программных модулей обработки полетной информации // Вісник КМУЦА. – 2000. – №1-2. – С.127–133.
20. *Райчев И.Э., Харченко А.Г., Яцков Н.А.* Методы создания тестовых наборов данных при сертификационных испытаниях комплексов программ контроля полетов // Вісник НАУ. – 2001. – №1. – С. 126–132.

Поступила 4.03.2013р.