Скиба О.В., Троцик С.М. Державний науково-дослідний інститут випробувань і сертифікації озброєння та військової техніки

ЗАСТОСУВАННЯ ТЕХНІКИ НЕГАТИВНОГО ТЕСТУВАННЯ СПЕЦІАЛЬНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРИ ВИПРОБУВАННІ ЗРАЗКІВ ОЗБРОЄННЯ ТА ВІЙСЬКОВОЇ ТЕХНІКИ

У статті розкриваються погляди на мету і спрямованість негативних перевірок спеціального програмного забезпечення, надається орієнтовний перелік питань, які рекомендовано перевірити із застосуванням техніки негативного тестування з урахуванням положень нормативно-правових актів України та практичного досвіду в області випробування програмного забезпечення. Запропоновані підходи з проведення негативного тестування, пропонується врахувати при перевірці спеціального програмного забезпечення, яке встановлюється на автоматизовані робочі місця пунктів управління Збройних Сил України, а також на озброєння та військову техніку.

Ключові слова: програмне забезпечення, програмний продукт, спеціальне програмне забезпечення, тестування програмного забезпечення, метод тестування програмного забезпечення, динамічне тестування, пошукове тестування, тестування методом "чорної скриньки", негативне тестування, техніка стресового тестування, помилка програмного забезпечення, дефект програмного забезпечення, передбачення помилки.

Постановка проблеми. За статистичними даними програмні засоби, готові до використання, містять від 5 до 50 помилок на тисячу рядків програмного коду. Їх мають навіть ті програми, які вважаються всебічно перевіреними, позбавленими дефектів і позиціонуються розробниками як цілком завершений якісний програмний продукт. Це засвідчує, що програмне забезпечення (ПЗ) не зазнає належного випробування. Одночасно й підтверджується загальне переконання фахівців у галузі тестування ПЗ, що всі помилки у роботі програми виявити неможливо: *"У міф про повне тестування вірять і деякі нещасні тестувальники. Їх одвічно пригнічує відчуття провини, оскільки... все одно у програмах залишаються помилки"* [1, С. 40].

Основним способом перевірки якості ПЗ є його практичне тестування. Для цього застосовуються як позитивні перевірки, що передбачають виконання ПЗ відповідно до експлуатаційної документації (ЕД) і технічного завдання (ТЗ), так і негативні, що спрямовані на перевірку працездатності ПЗ в нестандартних умовах і на виявлення помилок.

У науково-технічній літературі, що стосується сфери випробування програмних продуктів, призначення тестування ПЗ збігається саме з метою негативних перевірок:

"Тестування забезпечує виявлення (констатацію фактів) розбіжностей з вимогами" [2, С. 12];

"Головною задачею тестувальника є пошук та документування помилок" [1, С. 93];

"Завдання тестування – визначення умов, за яких проявляються дефекти системи ..." [3, С. 4];

"Під цим (тестуванням – прим. авт.) розуміють процес, під час якого виконується програмне забезпечення з метою виявлення місць некоректного функціонування коду" [4];

"Тестирование – это процесс исполнения программы с целью обнаружения ошибок" [5, п. 1.2].

При чому, як засвідчує практичний досвід тестувальників, велику частку помилок у роботі ПЗ виявлено саме під час негативних перевірок. Відтак, такому виду тестування повинно підлягати все ПЗ, розробники якого зацікавлені у досягненні високої якості цих програмних продуктів: від ПЗ загального та комерційного призначення до спеціального ПЗ (СПЗ), що використовується, зокрема, у фінансовій, аерокосмічній та військовій сфері.

Техніка негативного тестування, як і решта способів випробування ПЗ, потребує вмілого та досконалого планування самих перевірок. *"Стратегія негативного тестування*

ніколи не повинна нагадувати стрільбу навмання. Навпаки, плани тестування повинні бути ретельно продуманими, несуперечливими, завершеними та ефективно забезпечувати виявлення помилок”[6, С.216].

Ця стаття присвячена питанням планування пошуку помилок у СПЗ під час його тестування.

Актуальність дослідження.

Практика ведення збройної боротьби останніх десятиліть засвідчує, що обсяг даних, які потрібно обробити для врахування всіх можливих чинників, котрі можуть вплинути на хід протистояння, постійно зростає. І в той же час вимога оперативності, що висувається до управління військами, потребує суттєвого зменшення термінів на обробку інформації, від якої залежить прийняття рішень.

На стратегічному та оперативному рівнях це питання сплановано вирішити шляхом створення єдиної автоматизованої системи Збройних Сил України (ЗСУ), яка передбачає об’єднання автоматизованих робочих місць (АРМ) пунктів управління у єдину мережу.

На тактичному рівні, намагаючись вирішити цю дилему, розробники сучасних та модернізованих зразків озброєння та військової техніки (ОВТ) все частіше оснащують їх спеціальним обладнанням з СПЗ, що дозволяє їх інтегрувати в єдину інформаційну (інформаційно-розрахункову) систему.

Як і ОВТ, програмна компонента має власні показники якості, зокрема, функціональна придатність, надійність, захищеність, мобільність та інші, які повинні відповідати своїм специфікованим вимогам.

Випробування цих аспектів повинно проводитися із застосуванням як позитивних, так і негативних перевірок. Адже, як засвідчує досвід тестування, за результатами позитивних перевірок якості програми може оцінюватися достатньо високо, проте негативне випробування ПЗ, зазвичай, надає зовсім протилежну оцінку.

Відтак, негативні перевірки особливо чутливі та важливі для СПЗ, що застосовуються для управління ОВТ, оскільки їх помилки можуть мати вкрай негативні наслідки.

Зв’язок авторського доробку з практичними завданнями

Відповідно до положень нормативно-правових документів, що визначають порядок технічного оснащення Збройних Сил України (ЗСУ) [7, 8], зразки ОВТ підлягають випробуванню, під час яких досліджуються тактичні, маневрені, механічні, кліматичні та інші характеристики, які стосуються технічних аспектів.

Досвід випробувань, набутий Державним науково-дослідним інститутом випробувань і сертифікації ОВТ (ДНДІ ВС ОВТ), засвідчив, що програмний компонент обов’язково повинен бути охоплений випробуванням та оцінений, як і будь-яка інша складова ОВТ. Це дозволить більш об’єктивно визначити можливість допуску зразка ОВТ до підконтрольної експлуатації.

Враховуючи відсутність єдиного підходу у ЗСУ до перевірки СПЗ ОВТ, ДНДІ ВС ОВТ здійснює розроблення типової методики оцінки якості СПЗ, один із аспектів якої має зв’язок з матеріалами цієї статті.

Виклад основного матеріалу. Намагаючись зробити ОВТ більш ефективним, а його СПЗ більш функціональним та цікавим для можливих замовників/покупців, розробники програм намагаються створити їх багатоцільовими.

Натомість, підвищення можливостей СПЗ обумовлює більшу складність їхнього розроблення та необхідність ретельного взаємоузгодження різних складових, а це, у свою чергу, формує передумови до виникнення помилок у функціонуванні СПЗ.

У даній статті до помилок (дефектів) СПЗ віднесено такі ситуації:

- факт прояву непрацездатності СПЗ або окремих його функцій;
- коротка втрата працездатності СПЗ, яка автоматично відновлюється власними засобами програми;

- запис елемента ПЗ, використання якого призводить (може призвести) до небажаного (недопустимого, неправильного) результату.

При цьому до помилок не відносяться недоліки зовнішнього вигляду елементів інтерфейсу, некомфортність використання програми, відсутність довідки. Також не вважається за помилку ситуація, коли СПЗ власними вбудованими можливостями виявляє дефект і проінформує користувача, залишаючись у цілком працездатному стані.

Причини помилок.

Класифікація помилок у різних літературних джерелах є не ідентичною. Авторами статті, з урахуванням вищезазначених обмежень, пропонується розподілити їх за декількома категоріями.

За фазами взаємодії СПЗ з програмно-апаратним комплексом, при яких проявляються помилки:

- при інсталяції СПЗ;
- при першому запуску;
- при введенні (отриманні, імпортуванні) вхідних даних;
- при виконанні СПЗ (передусім, проведення розрахунків);
- при інформаційному обміні з іншим ПЗ (СПЗ);
- при одночасному виконанні на одному технічному засобі з іншим ПЗ (СПЗ),

взаємодія з яким не передбачена;

- при виведенні вихідних даних;
- при завершенні роботи.

Окремі негативні прояви ПЗ можуть виявлятися навіть після її деінсталяції. І це також є показником СПЗ.

За джерелом походження та причиною виникнення помилок:

- помилки програмного коду СПЗ (тобто прорахунки, які допущені програмістом);
- помилки вхідних даних:
 - а) введення користувачем;
 - б) файлові дані, що містять вхідну інформацію;
 - в) вхідні дані від іншого ПЗ (СПЗ);
- помилки використання ресурсів:
 - а) недостатність оперативної та/або відео- пам'яті;
 - б) недостатність дискового простору;
 - в) недоступність ресурсу;
- помилки виведення вихідних даних:
 - а) помилка виводу на екран;
 - б) помилка збереження,
 - в) помилки друку;
 - г) невідповідність прав доступу;
- критичні умови функціонування СПЗ:
 - а) одночасна робота певної кількості користувачів у мережі;
 - б) одночасна передача даних великого обсягу;
 - в) обмін даними в умовах нестійкого зв'язку;
 - г) робота з АРМ великої кількості периферійних пристроїв;
- помилки взаємодійності з іншим ПЗ (СПЗ) та іншими АРМ з таким самим СПЗ:
 - а) синхронний обмін даними між АРМ;
 - б) взаємодійність СПЗ різних версій;
 - в) сумісність СПЗ різних інтерфейсів;
- помилки співіснності з іншим ПЗ (СПЗ).

Схематично, в узагальненому вигляді, фази та джерела (причини) виникнення помилок зображені на рисунку 1.

Рис.1. Схематичне зображення джерел (причин) розповсюджених помилок у роботі СПЗ під час його взаємодії з технічним засобом та програмним середовищем

Зазначений перелік не вважається вичерпним, що обумовлено призначенням і можливостями конкретного СПЗ.

У вищевказаних категоріях дефектів кожен елемент переліку має свій набір характерних дій, що з великою вірогідністю призводять до виникнення помилкової ситуації у функціонуванні СПЗ.

Саме на виявлення таких ситуацій спрямовує негативні перевірки розробник плану тестування та безпосередньо сам тестувальник. У [9, п. 4.14] такий підхід йменується терміном “передбачення помилки”.

На відміну від позитивних перевірок, які розробляються на основі знань того, що і як СПЗ повинно виконувати, для планування негативних потрібно застосовувати дещо інший підхід. “Основна ідея його полягає у тому, щоб перерахувати у певному списку можливі помилки або ситуації, у яких вони можуть з’явитися, а потім на основі цього списку написати тести” [5, п. 3.2.4]. При чому, як зазначено у [1, С. 196] “Не варто гаяти час на пошуки пояснень того, чому певні входні значення або місце програми здається вам підозрілим. Просто протестуйте його”.

При цьому потрібно максимально охопити не лише функції ПЗ, а й всі його фази взаємодії з технічним засобом та програмним середовищем, де передбачено використовувати – з процесу інсталяції до деінсталяції.

Під час інсталяції можливо виявити найперші прояви дефектів, які в подальшому можуть себе і не проявляти. Такі помилки можуть бути обумовлені апаратними особливостями технічного засобу, на який встановлюється СПЗ, його програмним середовищем (операційна система та інші програми, зокрема антивірусні), політикою безпеки тощо. Не виключено, що за таких умов, для успішного завершення інсталяції, необхідно буде вимушено (що не передбачено ЕД) провести певні налаштування, які потребуватимуть додаткового часу і певних навичок, що не входить до переліку питань угоди (договору) та положень ТЗ.

Проведення негативних перевірок при інсталяції передбачають:

- інсталяцію СПЗ на АРМ (технічний засіб), що має обмежені характеристики (оперативна пам’ять, процесор, відеокарта, кількість вільного дискового простору);
- інсталяція на декілька АРМ, що мають різні операційні системи;
- встановлення СПЗ в директорії, відмінний від того, що пропонується інсталятором за умовчанням;
- інсталяція від імені користувача з найменшими правами, наданими відповідно до політики безпеки;

- встановлення на АРМ, що не має жодного прописаного драйвера принтера (для СПЗ, що передбачає роздрукування інформації);
- інсталяція нової версії СПЗ поверх попередньої.

Першому запуску СПЗ властиві дефекти, які можуть бути спричинені незавершеністю інсталяції програми, відсутністю початкових налаштувань, невідповідністю технічних можливостей АРМ або несумісністю (конфліктністю) з іншим ПЗ (СПЗ), відсутністю необхідних прав користувача. При цьому окремі помилки, як і при інсталяції, можуть проявитися лише один раз, і це також потребуватиме непередбачених умовою (договором) зусиль.

Не дивлячись на те, що проблеми інтерфейсу цією статтею не відносяться до помилок, доречно вказати на такий аспект, як мова інтерфейсу за умовчанням: деякі СПЗ автоматично застосовують англomовний, а для його заміни на український варіант потребується пошук спеціальних файлів *.ini (або інших) та ручне редагування. При чому такі питання можуть бути не відображені ні в ЕД, ні в будь-якому іншому документі, що надається з СПЗ.

Перевірка поведінки СПЗ при першому входженні проводиться після завершення інсталяції і передбачає наступне:

- запуск на АРМ (технічному засобі), що має обмежені характеристики (оперативна пам'ять, процесор, відеокарта, роздільна здатність екрану);
- запуск на АРМ, що мають різні операційні системи;
- запуск від імені користувача з найменшими правами, наданими відповідно до політики безпеки;
- запуск на АРМ, де встановлено антивірусне програмне забезпечення.

Якщо при першому входженні СПЗ потребує вводу даних ідентифікації (ім'я та пароль), доцільно спочатку спробувати увійти під недійсним акаунтом.

Функціонування програми – це етап випробування ПЗ, в ході якого проявляється найбільше помилок і вони є більш критичними. Відповідно до [10, п. 5.1], помилки виконання є найбільш непередбачуваними.

Одним із поширених джерел провокування помилок в ході виконання ПЗ є вхідні дані. При чому, не завжди такі дефекти обумовлені введенням хибних даних користувачем (випадково або навмисно).

Частина проблем схована у програмному коді через помилкові порівняння, прописані програмістами. Наприклад, дуже розповсюджений у програмуванні оператор "if" у середовищі тестувальників пов'язаний з безліччю проблем, що виникають через неправильне порівняння: застосування ">" замість "≥" ("<" замість "≤") [6, С. 215]. Для порівняння, у нашому буденному житті дуже поширене слово "до", якщо вживається при визначенні кінцевого терміну (наприклад, до 25 травня), може сприйматися двояко: одні люди вважають, що це тотожно фразі "включно по" або "не пізніше" (тобто можливо виконати і 25 травня), інші – вважають, що потрібно завершити до настання строку (тобто термін закінчується о 00.00 25 травня). Ідентичні помилки притаманні й програмістам.

Тому дуже корисними є перевірки вхідних даних, що перебувають на межі допустимих діапазонів або піддіапазонів, а саме:

- якщо поля даних передбачають введення чисел від 10 до 50, доцільно перевірити такі вхідні дані: 9, 10, 50, 51;
- якщо поля даних передбачають введення чисел різної розрядності (одиниці, десятки, сотні, тисячі), то при вибірці тестових значень вхідних даних обов'язково повинні бути представлені варіанти різної кількості цифр (від 0 до 9, від 10 до 99, від 100 до 999 і т.д.);
- якщо поле вводу передбачає введення тексту довжиною від 3 до 10 символів, доцільно спробувати ввести текст, що складається з 2, 3, 10 та 11 символів.
- якщо вхідні дані передбачають введення великих і малих літерних символів англійського алфавіту, потрібно перевірити граничні "A", "Z", "a", "z", а також ті символи, що в таблиці символів ASCII є суміжні з ними: "@" (знаходиться перед "A"), "[" (знаходиться після "Z"), "`" (знаходиться перед "a"), "{" (знаходиться після "z");

– якщо передбачаються введення (вставка) великих і малих літер українського алфавіту, аналогічним чином перевіряються “А”, “Я”, “а”, “я” (як граничні літери діапазонів), літери “І”, “Є”, “І”, “І”, “є”, “і”, а також суміжні з ними символи (див. Таблицю 1).

Таблиця 1

Перелік літер українського алфавіту та суміжних
інших символів відповідно до кодування таблиці ASCII

	Символ перед літерою кирилиці	Літери кирилиці	Символ після літери кирилиці
Символ	□	Г	
Код	164	165	166
Символ	©	Є	«
Код	169	170	171
Символ	®	І	°
Код	174	175	176
Символ	±	І, і, г	μ
Код	177	178...180	181
Символ	№	Є	»
Код	185	186	187
Символ	s	А...Я, а...я	-
Код	190	191...255	

Окремим символом, схильним до провокування помилок у роботі СПЗ є 0 (нуль). Тому, якщо діапазон допустимих значень включає цю цифру, її рекомендовано перевірити в якості вхідної інформації.

Якщо СПЗ передбачає графічне відображення даних (наприклад, на карті обстановки), рекомендується перевірити можливість та безпомилковість нанесення даних обстановки на межі видимої екранної форми карти та на невидимій частині.

Крім цього, обов’язковими є перевірки наявності вбудованого захисту ПЗ (СПЗ) на введення інших хибних даних. Для цього:

- перевірити реагування ПЗ (СПЗ) на залишення поля вводу пустим (не вводячи в нього будь-яких даних);
- якщо у поле вводу передбачено введення лише цілих чисел, спробувати, ввести число зі знаком “+”, а також дробове число з нульовим значенням дробової частини (наприклад, “9,0”);
- спробувати не ввести дані через клавіатуру, а вставити недопустимий текст з буферу обміну (у т.ч. здійснити спробу вставки тексту більшої довжини, ніж передбачено в ЕД). Такі дії можуть бути корисними, оскільки деякі поля вводу можуть передбачати перевірку символів під час їх вводу, але не пристосованими до перевірки тексту, який вставляється.

Також джерелом вхідних даних можуть бути:

- файлові дані, розміщені на АРМ, на якому встановлено СПЗ;
- файлові дані, розміщені на інших технічних засобах;
- інформація, отримана від інших АРМ, технічних засобів;
- інформація, отримана від іншого ПЗ (СПЗ), що встановлено на АРМ, де функціонує СПЗ, яке тестується.

При отриманні програмою такого типу вхідних даних, імовірні наступні помилки:

- довжина текстової змінної, в яку зчитуються дані з файлу, є коротшою, ніж запис рядка файлу, що призводить до обрізання вхідної інформації;
- зчитування записів з файлу у список (ComboBox, ListBox) здійснюється не з першого рядка, а з другого або завершується передостаннім (характер цієї помилки теж

пов'язаний з граничними значеннями, а також з тим, що нумерація починається з нуля, а не з одиниці);

- ідентичні дефекти при обміні даними між такими елементами інтерфейсу, як ComboBox, ListBox;

- некоректне переведення зчитаних з файлу текстових величин у числові;

- при копіюванні рядків даних між ComboBox, ListBox не всі дані переносяться (нумерація починається з нуля);

- зміст файлу вхідних даних був змінений у ході функціонування СПЗ (наприклад, з іншого АРМ або через інше діалогове вікно цього ж СПЗ), але це не враховується у тому вікні (елементі інтерфейсу), до якого було завантажено вхідні дані до їх зміни;

- неадекватне реагування СПЗ, якщо при зверненні до файлу з вхідними даними він не мав жодного запису (характерне для першого звернення);

- неадекватне реагування СПЗ, якщо на момент його звернення до файлу з вхідними даними він був зайнятий іншим ПЗ (СПЗ);

- неадекватне реагування СПЗ, якщо на момент його звернення до файлу з вхідними даними виявлено його відсутність (таке можливо як через хибне видалення користувачем так і в результаті програмно-апаратних збоїв);

- неадекватне реагування СПЗ при відсутності зв'язку з іншим технічним засобом, який містить файл з вхідними даними;

- формат (розширення) вхідного файлу, створеного іншим ПЗ, не дозволяє СПЗ його зчитати, оскільки ПЗ, що його створило, було оновлено (наприклад, файли MS Office версій до 2007 року, що мали розширення *.doc, *.xls, а після оновлення – *.docx, *.xlsx відповідно);

- деякі спеціальні символи (апостроф, лапки, довге тире, дюйм, градус тощо), що містяться у прийнятих файлах від іншого АРМ, відображаються некоректно (найпоширеніший, наприклад – це їх заміна на "□").

Тому, для негативного тестування СПЗ плануються перевірки, які передбачають з'ясування цілісності даних (при їх завантаженні, копіюванні, переформатуванні) та моделювання поданих вище критичних ситуацій щодо змісту й доступності файлів вхідних даних.

Також поширеними причинами дефектів є помилкові розрахунки, округлення, позбавлення дробової частини (повністю або частково), переведення величин з однієї одиниці до іншої.

Помилкові розрахунки можуть бути обумовлені:

- складністю формул (в результаті можуть бути невірно розставлені дужки, пропущені змінні тощо);

- хибним посиланням на елемент масиву;

- використання величин у невідповідних одиницях;

- неврахування особливостей рельєфу (наприклад, при розрахунку відстані на карті);

- округленням даних або видаленням дробової частини.

При плануванні негативних перевірок щодо виявлення помилок при обчисленні, округленні та перетворенні величин одним із дієвих варіантів є завчасне проведення контрольних розрахунків з використанням відібраного переліку вхідних даних та порівняння з результатами виконання СПЗ.

Звісно, якщо помилки допущені у формулі або у помилкових округленнях, то тестуючи СПЗ як "чорний ящик", достеменно не можливо довести, що проблема саме у формулі.

Помилками, які спроможний виявити тестувальник, можуть бути:

- використання декількох однакових екранних форм для різних завдань. Наприклад, для розрахунку бойового потенціалу своїх військ та військ противника застосовується однаковий підхід та однакові формули. Тому, програміст може створити лише одне діалогове вікно для обох випадків. Не дивлячись на те, що для виклику кожного з них передбачено окремий елемент інтерфейсу, останні введені дані будуть єдиними і для своїх військ, і для противника (якщо не передбачено присвоєння їх значень певним змінним);

– використання даних, які були збережені раніше в інших величинах. Наприклад, коли величиною відстані було встановлено “кілометр”, отримані значення відстаней були записані в певний файл. У подальшому користувач застосував іншу величину подання відстані – в метрах. Але при імпортуванні даних, збережених у кілометрах, СПЗ може сприйняти їх як метрові величини і прийняти до обчислення;

– різновиди форматів дат, що обумовлено відмінністю у послідовності викладення її елементів (ДД.ММ.РРРР, ММ.ДД.РРРР);

– відмінність підходів у визначенні дня тижня (для більшості країн першим днем є понеділок, для деяких держав (США, Ізраїль, Великобританія) – неділя).

Як зазначено вище, на роботу СПЗ впливають такі технічні характеристики АРМ, як об’єм оперативної і відеопам’яті, частота процесору, кількість вільного дискового простору. І певні дефекти у роботі СПЗ, обумовлені цими чинниками, можуть проявитися при інсталяції. Проте, достатньо випадків, коли ні при інсталяції, ні при першому запуску та навіть при штатному режимі застосування СПЗ з цим проблем не виникало. Вони схильні проявлятися, якщо СПЗ або АРМ навантажити обробкою достатньої кількості даних (здійснити стресове тестування) [9, п. 4.43]. Це здійснюється шляхом:

– виконання на СПЗ декількох завдань одночасно (якщо у СПЗ впроваджена така можливість);

– вибору декількох графічних та відеофайлів для перегляду (якщо у СПЗ впроваджена така можливість);

– обміну великою кількістю даних з іншим ПЗ (СПЗ) або АРМ;

– обміну файлами великого об’єму з іншим ПЗ (СПЗ) або АРМ;

– інтенсивної роботи у мережі максимально можливої кількості користувачів;

– запуску на АРМ іншого ПЗ та відкриття в ньому великої кількості файлів або обробки великого обсягу інформації;

– підключення до АРМ максимальної кількості периферійних пристроїв та їх одночасного використання.

Для об’єктивності результату перевірки потрібно поексплуатувати СПЗ у таких умовах якомога довше. При цьому можливо проводити інші тести СПЗ.

Проблеми, пов’язані з виведенням даних, мають різний характер та причини.

При виведенні на друк можливі дефекти у роботі СПЗ через:

– фізичну недоступність принтеру (вимкнений, відсутній зв’язок з мережевим принтером);

– відсутність у СПЗ можливості вибрати принтер для поточного друку;

– занятість принтера друком іншого завдання;

– відсутність прав доступу користувача для здійснення друку;

– відправку на друк файлу великого обсягу або великої кількості файлів;

– відсутність паперу у принтері.

При виведенні інформації на екран можливі такі помилки:

– невідповідність масштабу відображуваної обстановки;

– недотримання співрозмірності при використанні моніторів з нестандартним співвідношенням висоти й ширини (квадратні або з надмірною шириною).

При збереженні (експортуванні) даних у файл можливі проблеми, пов’язані з:

– відсутністю файлу (особливо чутливо для першого запуску СПЗ) або його пошкодженням;

– заміною усього змісту файлу замість додавання в кінець;

– записом нової інформації поверх існуючого останнього запису у файлі;

– недоступністю файлу через його використання іншим ПЗ або АРМ;

– відсутністю прав доступу користувача для зміни файлу вихідних даних;

– невідповідністю формату файлу (вище наведений приклад файлів MS Office).

Негативні перевірки взаємодії з іншим ПЗ спрямовують на пошук дефектів за ускладнених та нестандартних ситуацій. При цьому перевіряється:

– можливість АРМ-адресанта здійснити донадсилання даних після відновлення зв'язку (з практичної точки зору це важливо, якщо враховувати реальні умови, в яких може функціонувати АРМ);

– можливість АРМ-адресата здійснити доотримання даних після відновлення зв'язку;

– синхронний обмін даними між двома АРМ з СПЗ;

– обмін даними максимального об'єму даних (заданих у кілобайтах, мегабайтах тощо або довжиною текстового рядка);

– обмін між СПЗ, що встановлені на АРМ різних апаратних платформ: стаціонарна електронно-обчислювальна машина (або ноутбук), планшет або смартфон, у т.ч., що функціонують під керівництвом різних операційних систем;

– можливість обміну СПЗ різних версій (наприклад, версія для керівника підрозділу та версія для чергового офіцера).

– можливість обміну СПЗ різних інтерфейсів (англомовного та україномовного, якщо такі варіанти передбачені).

Крім того, фахівці рекомендують у ході тестування застосовувати інші прийоми, які в деяких ПЗ призводили до виникнення дефектів:

– в ході виконання СПЗ будь-якого із завдань здійснити хаотичне натискання клавіш клавіатури (виключаючи Alt+F4 і Ctrl+Esc) та оцінити поведінку СПЗ. Провести подібні дії в режимі очікування СПЗ;

– при перебуванні СПЗ в режимі очікування здійснити безсистемний перехід управляючими елементами інтерфейсу з чергуванням хаотичного натискання кнопок клавіатури та оцінити поведінку СПЗ;

– додатково перевірити наявність захисту СПЗ від виконання користувачем окремих функцій у неправильній послідовності (всупереч порядку, зазначеному в ЕД);

– якщо СПЗ складається з декількох модулів, обов'язково перевірити взаємозв'язок між ними. Це пояснюється тим, що різні модулі могли проектувати різні фахівці, а їх програмування теж могло виконуватися різними програмістами. І це є передумова для виникнення помилок у ПЗ через несумісність форматів даних, неузгодженість найменувань змінних та порядку спільного використання елементів інтерфейсу тощо;

– якщо тестується оновлене ПЗ, обов'язково перевірити ті модулі та можливості, що зазнали змін, а також інші модулі ПЗ, які взаємодіють з тим, що був оновлений. Помилки, які виникають при цьому, є наслідком саме проведених змін, які не були ретельно узгоджені з іншими компонентами.

Прикладами такого можуть бути:

а) довжину текстової змінної збільшено, але елемент інтерфейсу (наприклад, текстове поле, список), що використовує цей текст, залишено без змін, що призведе до урізання правої частини тексту;

б) точність розрахунків збільшено. Для цього змінено формат змінних з *integer* на *real* та виключено операцію округлення. Але, на останньому етапі, для виведення результату у діалогове вікно залишено команду *IntToInt* замість *FloatToInt*;

в) кількість аргументів, що розраховується під час виконання процедури збільшено, проте виклик процедури залишено без врахування цих змін;

г) ім'я або розташування файлу, де зберігаються параметри функціонування СПЗ змінено або призначено режим "для читання". Але в окремих процедурах, через які реалізована можливість внесення змін у зазначений файл, залишилося посилання на попереднє ім'я чи шлях файлу або не передбачено зняття атрибуту "для читання".

Ще одним правилом роботи тестувальника є апробація всіх елементів інтерфейсу вручну та перегляд всіх вікон "...усі шляхи виконання програми будуть охоплені відповідними тестовими наборами" [11, п. 8.3]. Це пояснюється тим, що при цих діях можлива дуже короткочасна (миттєва) поведінка ПЗ (СПЗ), яку може помітити людське око, а програма, яка автоматизує процес тестування, взагалі пропустить.

Під час деінсталяції з'ясовується вимога до прав користувача, який має право запуску цього процесу (відповідно до політики безпеки), перевіряється можливість безперешкодного

видалення ПЗ з АРМ і відсутність від іншого ПЗ помилкових сповіщень, що стосуються видалення СПЗ, яке тестується.

Після деінсталяції з'ясовується відсутність хибних посилань на деінстальовану програму. Це перевіряється перевіркою файлу реєстру операційної системи і меню “Автозавантаження” та шляхом запуску інших програм, які взаємодіяли з СПЗ, що тестується.

Головні висновки та перспективи використання результатів дослідження. Виходячи з матеріалу, викладеного у статті, стає очевидним, що для визначення якості СПЗ, яке поставляється зі зразками ОВТ або на елементах АСУ, потрібно проводити перевірки не тільки заявлених можливостей програми, а й цілеспрямовано здійснювати пошук прихованих дефектів у ній. Запропонована техніка негативного тестування дозволить виявити існуючі помилки у СПЗ, що у свою чергу підвищить ефективність випробування зразків ОВТ (АСУ).

Для цього рекомендується під час розроблення плануючих документів тестування і безпосередньо в ході його проведення залучати фахівців, що мають достатній досвід випробування ПЗ (СПЗ) та мають “інстинкт” тестувальника.

СПИСОК ЛІТЕРАТУРИ

1. Канер Сэм. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений: пер. с англ. / Сэм Канер, Джек Фолк, Енг Кек Нгуен. — К.: Издательство “ДиаСофт”, 2001. — 544 с.
2. Котляров В.П. Основы тестирования программного обеспечения: учебное пособие. Изд. 2-е. — М.: Интуит, 2016. — 348 с.
3. Слукин С. Введение в тестирование программного обеспечения: авторский курс. — 2009. — 134 с.
4. Тестирование программ – процесс обнаружения ошибок в программном продукте. Публікація на сайті Fb.ru // Компьютеры. Программное обеспечение. — Режим доступу: <https://fb.ru/article/255524/testirovanie-programm---protsess-obnaruzeniya-oshibok-v-programmnom-produkte>. — дата доступу: 12.06.2020.
5. Степанченко И.В. Методы тестирования программного обеспечения: учеб. пособие. — Режим доступу: http://window.edu.ru/catalog/pdf2txt/765/45765/22383?p_page=1. — дата доступу: 04.06.2020.
6. Калбертсон. Быстрое тестирование / Браун, Кобб. — М.: Издательский дом “Вильямс”, 2002. — 374 с.
7. Порядок постачання озброєння, військової і спеціальної техніки та боєприпасів під час особливого періоду, введення надзвичайного стану, проведення заходів із забезпечення національної безпеки і оборони, відсічі і стримування збройної агресії та у період проведення антитерористичної операції: Постанова Кабінету Міністрів України від 25.02.2015 № 345. — Режим доступу: <https://zakon.rada.gov.ua/laws/show/345-2015-%D0%BF>. — дата доступу: 05.06.2020. (Постанови Кабінету Міністрів України).
8. Інструкція про порядок прийняття на озброєння (постачання) Збройних Сил України озброєння та військової техніки іноземного виробництва: наказ Міністерства оборони України від 11.11. 2014 № 803. (Накази Міністерства оборони України).
9. Інженерія систем і програмних засобів. Тестування програмних засобів. Частина 1. Поняття та визначення (ISO/IEC/IEEE 29119-1/2013, IDT): ДСТУ ISO/IEC/IEEE 29119-1:2017. — [Чинний від 2019-01-01]. — Київ: Держспоживстандарт України, 2018. — 53 с. (Державний стандарт України).
10. Федорова Г.Н. Разработка, внедрение и адаптация программного обеспечения отраслевой направленности: учеб. пособие / Г.Н. Федорова. — М.: КУРС : ИНФРА-М, 2019. — 336 с.

11. Венчковский Л.Б. Разработка сложных программных изделий: учеб. пособие для вузов. – Режим доступа: <https://studfile.net/preview/4003581/>. – дата доступа 09.06.2020.

Скиба Олег Васильович

науковий співробітник Державного науково-дослідного інституту випробувань і сертифікації озброєння та військової техніки, Чернігів, Україна
<https://orcid.org/0000-0002-6252-6775>
+38050-411-61-80

Троцик Сергій Миколайович

науковий співробітник Державного науково-дослідного інституту випробувань і сертифікації озброєння та військової техніки, Чернігів, Україна
<https://orcid.org/0000-0003-1088-4877>
+38093-329-29-09

Oleh Skyba

Researcher of State Scientific Research Institute of Armament and Military Equipment Testing and Certification, Chernihiv, Ukraine
<https://orcid.org/0000-0002-6252-6775>
+38050-411-61-80

Trotsky Sergii

Researcher of State Scientific Research Institute of Armament and Military Equipment Testing and Certification, Chernihiv, Ukraine
<https://orcid.org/0000-0003-1088-4877>
+38093-329-29-09

**APPLICATION OF NEGATIVE SPECIAL SOFTWARE TESTING TECHNIQUE
WITHIN TESTING SAMPLES OF WEAPONS AND MILITARY EQUIPMENT
O. Skyba, S. Trotsyk**

The article reveals views on the purpose and direction of negative testing of special software. The research paper also provides an indicative list of questions recommended to be planned for testing using the technique of negative testing. The provisions of legal acts of Ukraine and practical experience in the field of software testing are also taken into account in the article.

Negative testing is very relevant for testing special software products used in weapons. This is because the risks of error are very critical and sensitive.

The main sources and causes of errors in the software are: user, input data, resources, procedures for output and storage of data, exchange with other programs or computers, operating conditions of the program, compatibility and coherence with other software.

Negative testing is proposed to be used at the following stages of software use: installation, first run, authorization, data entry, software module startup, calculations, data conversion, data exchange with other systems, data storage, display information, data printing, saving settings software operation.

The article states that the procedure for conducting negative testing should be carefully thought out. To do this, at the planning stage of software testing, specialists with experience in successful testing such software products are involved.

It is assumed that when planning a negative test, a group of experts uses information about the program, their own testing experience and intuition of the tester. At the same time, the main efforts are aimed at creating such conditions under which the software may work incorrectly. Negative testing is planned to cause just that non-standard behavior of the program.

The suggested approaches to perform negative testing are proposed to be taken into account when testing the special software that is installed at the workplaces of the Command Posts of the Armed Forces of Ukraine, as well as on weapons and military equipment.

Keywords: software, software product, special software product, software testing, software testing method, negative software testing, , dynamic software testing, exploratory software testing, “black-box” testing, software error, stress testing method, software failure, software error guessing.