**Лабораторна робота №8-1. Механізми та методи тестування. Розроблення форми фіксації дефектів та інструкції до неї**

**Мета роботи**. Навчитися застосовувати на практиці знання щодо організації проведення тестування програмного продукту.

**Завдання на лабораторну роботу**

1. Ознайомитись з теоретичними відомостями, необхідними для виконання роботи.

2. Для власного курсового проекту потрібно розробити тестовий план (майстер тест-план ) і тест-кейси для конкретної функціональності програмного застосунку або сайту, а також форму, за якою будуть фіксуватися виявлені в процесі тестування дефекти.

3. Оформити звіт, де надати у вигляді таблиць тестовий план, тест-кейси, форму для фіксації дефектів, виявлених в процесі тестування, додавши відповідні пояснення щодо обраного рішення.

**Контрольні запитання**

1. Що таке тестування?
2. Що таке відмовостійкість?
3. Які види тестування ви знаєте?
4. В чому полягає суть тестування?
5. Що найчастіше тестується в програмних засобах?
6. В яких моделях немає тестування?
7. Що таке модульне тестування?
8. Що таке інтеграційне тестування?
9. В чому полягає зміст тестування за допомогою «білого» ящику?

Результати надсилати на електронну адресу викладача [**t.i.lumpova@gmail.com**](mailto:t.i.lumpova@gmail.com)у вигляді текстових файлів з іменем у форматі **QA<Номер групи><Номер лекції / лабораторної>[-<Номер завдання>][літера позначення типу роботи L – лекція, R - лабораторна]<Прізвище англійською>**. Наприклад, **QA4112R**buts.doc. Відповіді повинні бути не довгими і змістовними. Не копіюйте фрагментів з різних інформаційних джерел, подумайте і викладіть свою точку зору. При наявності відповідей-"близнюків" відповідь буде зараховуватися першому за часом надсилання.

**Строк відсилки ЛР ІПЗ-41 - 28.03.2025**

**ІПЗ-42 - 31.03.2025**

**ТЕОРЕТИЧНІ ВІДОМОСТІ**

Логіка створення ефективних звітів про дефекти

При створенні звіту про дефект рекомендується дотримуватися

наступного алгоритму:

0. Виявити дефект.

1. Зрозуміти суть проблеми.

2. Відтворити дефект.

3. Перевірити наявність опису знайденого вами дефекту у системі управління дефектами.

4. Сформулювати суть проблеми у вигляді «що зробили, що отримали, що очікували отримати».

5. Заповнити поля звіту, починаючи з детального опису.

6. Після заповнення всіх полів уважно перечитати звіт, виправивши неточності та додавши подробиці.

7. Ще раз перечитати звіт, т.к. у пункті 6 ви точно щось пропустили.

Тепер про кожен крок докладніше.

Зрозуміти суть проблеми . Все починається саме з розуміння того, що відбувається з додатком. Тільки за наявності такого розуміння ви зможете написати по-справжньому якісний звіт про дефект, правильно визначити важливість дефекту та дати корисні рекомендації щодо його усунення. В ідеалі звіт про дефект описує суть проблеми, а

не її зовнішній прояв.

Порівняйте два звіти про ту саму ситуацію (додаток «Конвертер файлів» не розрізняє файли та символічні посилання на файли, що призводить до серії аномалій у роботі з файловою системою).

Поганий звіт, при написанні якого суть проблеми не зрозуміла:

|  |  |  |
| --- | --- | --- |
| Короткий опис | Докладний опис | Кроки з відтворення |
| Обробляються файли поза SOURCE\_DIR.  . | Іноді з незрозумілих причин програма обробляє випадкові файли поза каталогом  SOURCE\_DIR  Act: обробляються окремі  файли поза SOURCE\_DIR.  Exp: обробляються лише  файли, що знаходяться у  SOURCE\_DIR.  На жаль, не вдалося  виявити послідовність  кроків, що призводять до  появи цього дефекту. | На жаль, не вдалося  виявити послідовність  кроків, що призводять до появи цього дефекту. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Відтворюваність | Терміновість | Можливість  обійти | Важливість | Симптом | Коментар |
| Іноді | Висока | Ні | Висока | Некоректна  операція |  |
|  |  |  |  |  |  |

Хороший звіт, при написанні якого суть проблеми зрозуміла:

|  |  |  |
| --- | --- | --- |
| Короткий опис | Докладний опис | Кроки з відтворення |
| Програма не розрізняє файли та символічні посилання на файли. | Якщо в каталог SOURCE\_DIR помістити символічне посилання на  файл, виникає така  помилкова поведінка:  а) Якщо символічне  посилання вказує на  файл усередині  SOURCE\_DIR, файл  обробляється двічі, а до  DESTINATION\_DIR  переміщається як сам  файл, так і символічне  посилання на нього.  б) Якщо символічне  посилання вказує на  файл поза SOURCE\_DIR,  програма обробляє цей  файл, переміщає  символічне посилання і  схоже, проблема у цьому. Також цей дефект призводить до спроби  опрацювати каталоги як  файли. У алгоритмі  обробки SOURCE\_DIR  явно є логічна помилка: за  жодних умов додаток має  обробляти файли, що є поза SOURCE\_DIR,  тобто. щось не так з генерацією чи перевіркою  повних імен файлів. | 1. У довільному місці  створити таку структуру каталогів: /SRC/ /DST/ /X/  2. Помістити в каталоги SRC та X кілька довільних  файлів допустимого  формату та розміру.  3. Створити в каталозі  SRC два символічні  посилання: а) на будь-  який із файлів усередині каталогу SRC; б) на будь- який із файлів усередині  каталогу X.  4. Запустити програму.  Дефект: у каталог DST  переміщені як файли, і  символічні посилання;  вміст каталогу X  сам файл у  DESTINATION\_DIR, а  потім продовжує  обробляти файли в  каталозі, в якому спочатку був оброблений файл.  Act: додаток вважає  символічні посилання  файли самими файлами  (див. подробиці вище).  Exp: якщо у каталозі  SOURCE\_DIR програма виявляє символічне посилання, вона припиняє роботу з виведенням  повідомлення «Symbolic link [ім'я символічного  посилання] in  SOURCE\_DIR folder detected. Remove it manually or restart ap-plication with -  -force\_file\_operations key to remove automatically.» оброблено та переміщено до каталогу DST. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Відтворюваність | Важливість | Терміновість | Можливість  обійти | Симптом | Коментар |
| Завжди | Висока | Звичайна | Ні | Некоректна  операція | Швидкий погляд код показав, що замість is\_file()  використовуєтьс  я file\_exists().  Схоже, проблема у цьому. Також цей дефект призводить до  спроби опрацювати каталоги як файли. У алгоритмі обробки SOURCE\_DIR  явно є логічна  помилка: за  жодних умов  додаток має  обробляти  файли, що є поза  SOURCE\_DIR,  тобто. щось не  так з генерацією  чи перевіркою  повних імен  файлів |
|  |  |  |  |  |  |

Відтворити дефект . Ця дія не тільки допоможе надалі правильно заповнити поле «Відтворюваність», а й дозволить уникнути неприємної ситуації, в якій за дефект програми буде прийнято якийсь короткочасний збій, який (швидше за все) стався десь у вашому комп'ютері або в іншій частині ІТ-інфраструктури, яка не має відношення

до додатка, що тестується.

Перевірити наявність опису знайденого вами дефекту .

Обов'язково варто перевірити, чи немає в системі управління дефектами опису саме того дефекту, який ви щойно виявили. Це проста дія, що не стосується безпосередньо написання звіту про дефект, але значно скорочує кількість звітів, відхилених з резолюцією «дублікат».

Сформулювати суть проблеми . Формулювання проблеми у вигляді «що зробили (кроки з відтворення), що отримали (фактичний результат у докладному описі), що очікували отримати (очікуваний результат у докладному описі)» дозволяє не тільки підготувати дані для заповнення полів звіту, але ще краще зрозуміти суть проблеми.

У загальному випадку формула «що зробили, що отримали, що очікували отримати» хороша з наступних причин:

* Прозорість і зрозумілість: дотримуючись цієї формули, ви готуєте саме дані для звіту про дефект, не скочуючи в розлогі абстрактні міркування.
* Легкість верифікації дефекту: розробник, використовуючи ці дані, може швидко відтворити дефект, а тестувальник після виправлення дефекту переконатися, що справді виправлений.
* Очевидність для розробників: ще до спроби відтворення дефекту видно, чи насправді описане є дефектом, чи тестувальник десь помилився, записавши в дефекти коректну поведінку програми.
* Позбавлення зайвої безглуздої комунікації: докладні відповіді на «що зробили, що отримали, що очікували отримати» дозволяють вирішувати проблему та усувати дефект без необхідності запиту, пошуку та обговорення додаткових відомостей.
* Простота: на фінальних стадіях тестування із залученням кінцевих користувачів можна відчутно підвищити ефективність зворотного зв'язку, що надходить, якщо пояснити користувачам суть цієї формули і попросити їх дотримуватися її при написанні повідомлень про виявлені проблеми. Інформація, отримана на даному етапі, стає фундаментом для всіх подальших дій щодо написання звіту.

Заповнити поля звіту . Поля звіту про дефект ми вже розглянули раніше, тепер лише підкреслимо, що починати найкраще з докладного опису, оскільки в процесі заповнення цього поля може виявитися безліч додаткових деталей, а також з'являться думки щодо формулювання стисненого та інформативного короткого опису.

Якщо ви знаєте, що для заповнення якогось поля у вас не вистачає даних, проведіть додаткове дослідження. Якщо й воно не допомогло, опишіть у відповідному полі (якщо воно текстове), чому ви утруднюєтеся з його заповненням, або (якщо поле є список) виберіть значення, яке, на вашу думку, найкраще характеризує проблему (у деяких випадках інструментальний засіб дозволяє вибрати значення на кшталт "невідомо", тоді виберіть його).

Якщо у вас немає ідей щодо усунення дефекту, або він настільки тривіальний, що не потребує подібних пояснень, не пишіть у коментарях «текст заради тексту»: коментарі виду «рекомендую усунути цей дефект» не просто безглузді, але ще й дратують.

Перечитати звіт (і ще раз перечитати звіт) . Після того, як все написано, заповнено та підготовлено, ще раз уважно перечитайте написане. Дуже часто ви зможете виявити, що в процесі доопрацювання тексту десь вийшли логічні нестиковки чи дублювання, десь вам захочеться покращити формулювання, десь щось змінити.

Ідеал недосяжний, і не варто витрачати вічність на один звіт про дефект, але й відправляти невичитаний документ - теж ознака поганого тону.

Після оформлення звіту про дефект рекомендується додатково дослідити ту область програми, в якій ви щойно виявили дефект.

Практика показує, що дефекти часто проявляються групами (результат, який бути відповідно до вимог).