Laurea Magistrale in informatica
Università di Salerno

Test Incident Report
Post-Unit and Integration Testing

Team members
Saverio Napolitano - 0522501400
Gerardo Festa - 0522501452
Alessandra Parziale - 0522501422

https://github.com/GerardoFesta/infozilla

Sommario

1. Introduzione

Nel presente report saranno dettagliati i difetti e le problematiche riscontrate nell'ambito dell'esecuzione dei casi di test.
L'obiettivo principale di questo documento è quello di fornire un'analisi più dettagliata dei fault emersi durante il processo di testing preliminare di sistema.
Attraverso una valutazione dettagliata dei problemi riscontrati, si mira a migliorare il processo complessivo di sviluppo.

2. Incident Report - Unit Testing

2.1. Incident Report UTI1

- **Informazioni di Base**:
    - Progetto: "InfoZilla".
    - Ambiente: Ambiente di test.
    - Nome Incident: UTI1 (Unit Testing Incident 1).
    - Test Case: *TestGetOutputText2*, *TestGetOutputText3.*
    - Priorità: Alta.
    - Gravità: Critica.

- **Descrizione dell'Incidente**:
    - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
      Le patch vengono visualizzate all'interno del file cleaned, ma secondo la documentazione non dovrebbero comparire.
    - Passi per Riprodurre**:**
        1. Eseguire il programma dando in input una patch.
        2. Verificare il file cleaned creato
    - Comportamento Attuale: L'applicazione riporta la patch nel file cleaned.

- **Risoluzione e Pianificazione**:
    - Stato: aperto.
    - Cause Identificate: La posizione iniziale e finale della patch non viene riconosciuta correttamente.
    - Componente Coinvolto: *FilterPatches*, *PatchParser*.
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.

2.2. Incident Report UTI2

- **Informazioni di Base**:
    - Progetto: "InfoZilla".
    - Ambiente: Ambiente di test.
    - Nome Incident: UTI2 (Unit Testing Incident 2).
    - Test Case: testParseForPatches.
    - Priorità: Alta.
    - Gravità: Critica.

- **Descrizione dell'Incidente**:
    - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
      Il testo presente negli hunk non viene individuato correttamente.
    - Passi per Riprodurre**:**
      1. Creare un file di testo con delle patch e nella sezione hunk inserire a fine riga “\n\n”.
      2. Eseguire il tool a linea di comando.
      3. Verificare nel file .xml la corretta visualizzazione degli elementi presenti negli hunk.
    - Comportamento Attuale: L'applicazione riporta un’individuazione errata degli hunk, eliminando tutti gli elementi che fanno parte di quello specifico hunk e che sono situati dopo “\n\n”.

- **Risoluzione e Pianificazione**:
    - Stato: Aperto.
    - Cause Identificate: La presenza di “\n\n” a fine di una riga nella regione di hunk.
    - Componente Coinvolto: *PatchParser*.
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.

2.3. Incident Report UTI3

- **Informazioni di Base**:
    - Progetto: "InfoZilla".
    - Ambiente: Ambiente di test.
    - Nome Incident: UTI3 (Unit Testing Incident 3).
    - Test Case: *testDiffPatch1*.
    - Priorità: Alta.

        - Gravità: Critica.

- **Descrizione dell'Incidente**:
    - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
      Le patch in formato unified diff non vengono riconosciute.
    - Passi per Riprodurre**:**
      1. Creare un file di testo con patch che rispettano il formato unified diff.
      2. Eseguire il tool a linea di comando.
      3. Verificare il file xml di output.
    - Comportamento Attuale: L'applicazione non riconosce il formato diff unified.

- **Risoluzione e Pianificazione**:
    - Stato: Aperto.
    - Cause Identificate: Il problema è causato dal modo in cui vengono individuate le patches.
    - Componente Coinvolto: *PatchParser*
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.

2.4. Incident Report UTI4

- **Informazioni di Base**:
    - Progetto: "InfoZilla".
    - Ambiente: Ambiente di test.
    - Nome Incident: UTI4 (Unit Testing Incident 4).
    - Test Case: *testDiffPatchHunks*.
    - Priorità: Alta.
    - Gravità: Critica.

- **Descrizione dell'Incidente**:
    - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
      In presenza di una patch in formato Unified *Diff Format with Index Line* e successivamente un'altra patch in formato *Unified Diff Format*, il sistema riconosce gli *hunk* della seconda patch come parte della prima.
    - Passi per Riprodurre**:**

1. Creare un file con delle patches dove la prima è un formato riconosciuto e le altre sono in un formato differente da Unified Diff Format with Index Line.
2. Eseguire il tool a linea di comando.
3. Verificare il numero di hunk della patch riconosciuta.

- Comportamento Attuale: L'applicazione aggiungerà il numero di *hunk* delle altre *patch* in formato non riconosciuto nella patch riconosciuta se quest'ultima si trova subito prima delle altre.

- **Risoluzione e Pianificazione**:
  - Stato: Aperto.
  - Cause Identificate: Non riconoscendo la seconda *Patch* di formato *Unified Diff Format*, quest'ultima viene integrata alla prima *Patch* di formato *Unified Diff Format with Index Line*.
  - Componente Coinvolto: *PatchParser*
  - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.

2.5. Incident Report UTI5

- **Informazioni di Base**:
  - Progetto: "InfoZilla".
  - Ambiente: Ambiente di test.
  - Nome Incident: UTI5 (Unit Testing Incident 5).
  - Test Case: *testRunFilterCharEnumsANDCharEnums*, *testRunFilter2CharEnumsAND2CharEnums*, *testRunFilterNumEnumsANDNumEnums*.
  - Priorità: Alta.
  - Gravità: Critica.

- **Descrizione dell'Incidente**:
  - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
    Le enumerazioni della stessa tipologia non vengono riconosciute correttamente.
  - Passi per Riprodurre**:**
    1. Creare un file con due enumerazioni della stessa tipologia, di cui la seconda con due elementi.
    2. Eseguire il tool a linea di comando.

3. Verificare l’output prodotto all'interno del file .xml.
    - Comportamento Attuale: L'applicazione non riconosce la seconda enumerazione.
- **Risoluzione e Pianificazione**:
    - Stato: Aperto.
    - Cause Identificate: *SymbolCount* viene posto a zero dopo l’individuazione della prima enumerazione.
    - Componente Coinvolto: *FilterEnumeration.*
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.
    - Soluzione: porre *SymbolCount* a uno.

2.6. Incident Report UTI6

- **Informazioni di Base**:
    - Progetto: "InfoZilla".
    - Ambiente: Ambiente di test.
    - Nome Incident: UTI6 (Unit Testing Incident 6).
    - Test Case: *testRunFilterItemizationsANDCharEnums*, *testRunFilterNumEnumsANDItemizations*, *testRunFilterNumEnumsANDCharEnums*, *testRunFilterCharEnumsANDNumEnumsANDItemization*.
    - Priorità: Alta.
    - Gravità: Critica.

- **Descrizione dell'Incidente**:
    - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
      Avendo due o più enumerazioni, della stessa o di diversa tipologia, verranno conteggiate insieme.
    - Passi per Riprodurre**:**
      1. Creare un file con delle enumerazioni della stessa tipologia.
      2. Eseguire il tool a linea di comando.
      3. Verificare l’output prodotto all'interno del file *.xml*.
    - Comportamento Attuale: L'applicazione riconosce le diverse enumerazioni ma la prima enumerazione conterrà tutte le enumerazioni, la seconda conterrà le successive e così via nel caso di altre enumerazioni.

- **Risoluzione e Pianificazione**:
    - Stato: Aperto
    - Cause Identificate: La fine delle enumerazioni non viene riconosciuta correttamente.
    - Componente Coinvolto: *FilterEnumeration.*
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.

2.7. Incident Report UTI7

- **Informazioni di Base**:
    - Progetto: "InfoZilla".
    - Ambiente: Ambiente di test.
    - Nome Incident: UTI7 (Unit Testing Incident 7).
    - Test Case: *testGetOutputText.*
    - Priorità: Alta.
    - Gravità: Critica.

- **Descrizione dell'Incidente**:
    - Descrizione**:** Durante il testing di unità volto a individuare gli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema:
      L'eliminazione dell'ultimo elemento dell'enumerazione non viene eseguita.
    - Passi per Riprodurre**:**
      1. Creare un file con delle enumerazioni (senza "\n" finale).
      2. Eseguire il tool a linea di comando.
      3. Verificare l'output prodotto all'interno del file *.xml*.
    - Comportamento Attuale: L'applicazione non elimina l'ultimo elemento dell'enumerazione.

- **Risoluzione e Pianificazione**:
    - Stato: Aperto.
    - Cause Identificate: Il problema è causato dalla mancanza dello "\n" finale.
    - Componente Coinvolto: *FilterEnumeration*, FiliterTextRemover, *file xml*.
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.

2.8. Incident Report UTI8

- **Informazioni di Base**:
  - Progetto: "InfoZilla".
  - Ambiente: Ambiente di test.
  - Nome Incident: UTI8 (Unit Testing Incident 8).
  - Test Case: *testFindStackTracesWithCause.*
  - Priorità: Bassa.
  - Gravità: Bassa.

- **Descrizione dell'incidente**
  - Descrizione: Durante il testing di unità volto a individuare le cause degli errori rilevati durante il testing preliminare di sistema, abbiamo riscontrato il seguente problema: l'eccezione CausedBy viene riportata due volte, una volta riconosciuta correttamente come clausola di Caused e una volta come eccezione standard.
  - Passi per riprodurre:
    1. Creare un file contenente una stacktrace con almeno una eccezione e una Caused By, ad es:
       "Root Exception:\n" +
       "java.lang.TestException: RootException\n" +
       " at com.example.TestClass.method(TestClass.java:42)\n" +
       "Caused by:\n" +
       "java.lang.RuntimeException: CauseException\n" +
       " at com.example.TestClass.method2(TestClass.java:50)";
    2. Eseguire il tool a linea di comando.
    3. Verificare l'output prodotto all'interno del file *.xml*.

  - Comportamento Attuale: la clausola caused by viene riportata correttamente come eccezione, ma viene inserita due volta (una volta come "Caused By" e una volta no).

- **Risoluzione e Pianificazione**:
  - Stato: Chiuso.

- Cause Identificate: Il problema avviene perché c'è leggera sovrapposizione tra le regular expression che cercano espressioni e cause.
- Componente Coinvolto: *FilterStackTraceJAVA*.
- Risoluzione: Il modo più semplice di risolvere il problema è ciclare sulla lista di StackTrace che viene restituita ed eliminare una stacktrace se è uguale ad un'altra stacktrace eccetto per il flag *isCause*. A quel punto, si tiene la StackTrace che ha il flag *isCause* = *true*.
  La modifica non richiede impact analysis, in quanto non impatta in alcun modo né sulla correttezza degli altri test, né sul funzionamento degli altri filtri. Quest'ultimo non viene alterato perché la classe marca il testo per l'eliminazione prima della cancellazione della StackTrace dalla lista. Il funzionamento risulta ora corretto.

3. Incident Report - Integration Testing

3.1. Incident Report ITI1

- **Informazioni di Base**:
  - Progetto: "InfoZilla".
  - Ambiente: Ambiente di test.
  - Nome Incident: ITI1 (Integration Testing Incident 1).
  - Test Case: *TcTracePatch*, *TcSourcePatch*, *TcPatchEnum*, *TcPatchTraceSource*, *TcPatchSourceEnum*, *TcComplete.*
  - Priorità: Alta.
  - Gravità: Critica.

- **Descrizione dell'Incidente**:
  - Descrizione**:** Durante il testing di integrazione abbiamo riscontrato il seguente problema:
    La *Patch* viene riconosciuta ma non eliminata dal file prodotto come output finale.
  - Passi per Riprodurre**:**
    1. Abilitare l'opzione *Patch*.
    2. Creare un file con la presenza di almeno una *Patch*.
    2. Eseguire il tool a linea di comando.
    3. Verificare l'output prodotto all'interno del file *output.txt*.
  - Comportamento Attuale: L'applicazione riconosce le patch ma non le elimina e le visualizza all'interno dell'*output.txt*.

- **Risoluzione e Pianificazione**:
    - Stato: Aperto
    - Cause Identificate: La posizione iniziale e finale della patch non viene riconosciuta correttamente
    - Pianificazione: la modifica potrebbe coinvolgere più classi quindi riteniamo di dover eseguire Impact analysis.