Laurea Magistrale in informatica
Università di Salerno

Pre-Modifications System Testing

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

https://github.com/GerardoFesta/infozilla

Sommario

1. Introduzione

Per cercare di comprendere meglio il funzionamento del sistema, basandoci sulla breve specifica riportata all'interno del ReadMe del progetto e su quanto abbiamo visto dall'analisi statica del codice, strutturiamo un primo test di sistema.
La tecnica utilizzata è il category partitioning; andiamo quindi a dividere il dominio dei dati di ingresso in **classi** di casi di test.
I **parametri** per il file di input sono le informazioni che il tool riesce ad estrarre, quindi:

- Java Source Code Regions
- Patches
- Enumerations
- Java Stacktraces

I restanti parametri di input, passati a linea di comando, sono:

- with-source-code Process and extract source code regions (default=true)
- with-lists Process and extract enumerations (default=true)
- with-patches Process and extract patches (default=true)
- with-stacktraces Process and extract stacktraces (default=true)
- file-path

Le possibili scelte per i parametri del file di input sono in realtà piuttosto limitate. Essendo che il tool fa semplice estrazione di informazioni, non abbiamo situazioni nelle quali ci si aspetta che questo riporti un errore. Semplicemente, qualora uno dei "parametri" non dovesse rispettare la specifica formattazione che permette al tool di capire che si tratta di quel particolare costrutto, questo non verrà individuato dal tool e non sarà quindi presente nell'output estratto.
Per il momento quindi, ciascuno dei parametri del file di input avrà come singola categoria Quantità e come scelte 0, 1 e Many.
I parametri di input a riga di comando with-source-code, with-lists, with-patches, with-stacktraces hanno spazio di input booleano. Per questo motivo abbiamo selezionato come categoria Value e come scelte True, False e Malformed.
Infine, file-path avrà bisogno della categoria Validity e avrà due scelte: Valid, se il file path esiste e Invalid se il file path non esiste.

2. Parametri, categorie e scelte

With-source-code	
Categoria	**Scelte**
Value (WSCV)	1. True [if fileOk] 2. False [if fileOk] 3. Malformed [error]

with-lists (enumerations)	
Categoria	**Scelte**
Value (WLV)	1. True [if fileOk] 2. False [if fileOk] 3. Malformed [error]

with-patches	
Categoria	**Scelte**
Value (WPV)	1. True [if fileOk] 2. False [if fileOk] 3. Malformed [error]

with-stacktraces	
Categoria	**Scelte**
Value (WSTV)	1. True [if fileOk] 2. False [if fileOk] 3. Malformed [error]

file-path	
Categoria	**Scelte**
Valid (FPV)	1. Valid [property fileOk] 2. Invalid [error]

Java Source Code Regions	
Categoria	**Scelte**
Quantità (JSCQ)	1. 0 [if fileOk] 2. 1 [if fileOk] 3. Many [if fileOk]

Patches	
Categoria	**Scelte**
Quantità (PQ)	1. 0 [if fileOk] 2. 1 [if fileOk] 3. Many [if fileOk]

Enumerations	
Categoria	**Scelte**
Quantità (EQ)	1. 0 [if fileOk] 2. 1 [if fileOk] 3. Many [if fileOk]

Java Stacktraces	
Categoria	**Scelte**
Quantità (JSTQ)	1. 0 [if fileOk] 2. 1 [if fileOk] 3. Many [if fileOk]

Talkback Traces	
Categoria	**Scelte**
Quantità (TTQ)	1. 0 [if fileOk] 2. 1 [if fileOk] 3. Many [if fileOk]

3. WECT Test Cases

Il criterio di copertura scelto è il Weak Equivalence Class Testing, con l'aggiunta di alcuni casi che potrebbero rivelarsi particolari (l'assenza di ogni costrutto contemporaneamente con i flag=true, ad esempio). Formuliamo dunque i seguenti casi di test:

Test Case ID	**Combinazione**	**Oracolo**
TC1	FPV2	FAILURE
TC2	JSCQ1, PQ1, EQ1, JSTQ1, WSCV1, WLV1, WPV1, WSTV3, FPV1	FAILURE
TC3	JSCQ1, PQ1, EQ1, JSTQ1, WSCV1, WLV1, WPV3, WSTV1, FPV1	FAILURE
TC4	JSCQ1, PQ1, EQ1, JSTQ1, WSCV1, WLV3, WPV1, WSTV1, FPV1	FAILURE
TC5	JSCQ1, PQ1, EQ1, JSTQ1, WSCV3, WLV1, WPV1, WSTV1, FPV1	FAILURE
TC6	JSCQ1, PQ1, EQ1, JSTQ1, WSCV1, WLV1, WPV1, WSTV1, FPV1	File xml di output vuoto
TC7	JSCQ3, PQ3, EQ3, JSTQ3, WSCV1, WLV1, WPV1, WSTV2, FPV1	File xml con molte (2) patch, molte (3) source code area, molte (3) enumerations, 0 stackTrace
TC8	JSCQ3, PQ3, EQ3, JSTQ3, WSCV1, WLV1, WPV2, WSTV1, FPV1	File xml con 0 patch, molte source code area, molte enumerations, molte (2) stacktrace
TC9	JSCQ3, PQ3, EQ3, JSTQ3, WSCV1, WLV2, WPV1, WSTV1, FPV1	File xml con molte patch, molte source code area, 0 enumerations, molte (2) stacktrace
TC10	JSCQ3, PQ3, EQ3, JSTQ3, WSCV2, WLV1, WPV1, WSTV1, FPV1	File xml con molte patch, 0 source code area, molte enumerations, molte (2) stacktrace
TC11	JSCQ3, PQ3, EQ3, JSTQ3, WSCV1, WLV1, WPV1, WSTV1, FPV1	File xml con molte patch, mote source code area, molte enumerations, molte (2) stacktrace
TC12	JSCQ2, PQ2, EQ2, JSTQ2, WSCV1, WLV1, WPV1, WSTV1, FPV1	File xml con una patch, una source code area, una enumeration, una stacktrace