UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Esame di Ingegneria, Gestione ed Evoluzione del Software

PROGETTO CODESMILE

Pre-Modifications System Testing

TEAM MEMBERS

Dario Mazza - 0522501553

Nicolò Delogu - 0522501556

REPOSITORY

`https://github.com/xDaryamo/smell_ai`

VERSIONE

0.1

Anno Accademico 2024-2025

1 Introduzione

Lo scopo del presente documento è quello di presentare l'insieme dei casi di test designati a realizzare il testing di sistema del tool CodeSmile prima di apportare le modifiche proposte nelle *Change Request*. La tipologia scelta è quella del testing *black-box*, che ci permette di concentrarci sul comportamento esterno del sistema.

2 Identificazione delle categorie e dei parametri

Il criterio adottato è quello del *category partitioning*, allo scopo di individuare le categorie di input e i parametri a cui associare ciascuna di esse. Di seguito sono elencati entrambi i gruppi che abbiamo definito per il testing di sistema di CodeSmile:

- Parametro **`File`**: fa riferimento agli aspetti che riguardano i files forniti in input al tool;
  `Categorie`
  - **`Numero di File Input (NF)`**: fa riferimento al numero di file dati in input al tool, che può essere un singolo file o più file.
  - **`Estensione del File (EF)`**: fa riferimento all'estensione del file, che può essere '.py' o un formato diverso.
- Parametro **`Smell`**: fa riferimento agli aspetti di debito tecnico, che riguardano l'input fornito al tool;
  `Categorie`
  - **`Numero di Code Smells (NCS)`**: fa riferimento al numero di code smell identificati nel file.
  - **`Tipo di Code Smell (TCS)`**: fa riferimento alla tipologia di code smell rilevata, ad esempio generico o specifico per API.
- Parametro **`Struttura`**: fa riferimento agli aspetti strutturali dei progetti forniti in input al tool;
  `Categorie`
  - **`Numero dei Progetti (NP)`**: fa riferimento alla scelta di analizzare un singolo progetto o più progetti contemporaneamente.
  - **`Struttura del Progetto (SP)`**: fa riferimento alla complessità della struttura della directory (singola, annidata).
- Parametro **`Configurazione del Tool`**: fa riferimento agli aspetti di configurazione del tool;
  `Categorie`
  - **`Modalità di Esecuzione (ME)`**: fa riferimento all'uso del tool tramite interfaccia grafica o riga di comando.
  - **`Esecuzione Parallela (EP)`**: fa riferimento al flag che può abilitare più walkers per effettuare l'analisi.

- **`Numero di Walkers (NW)`**: fa riferimento alla quantità di walkers che compiono l'analisi.
- **`Errore (ERR)`**: fa riferimento agli eventuali errori (in formato .txt) restituiti durante l'esecuzione del tool.
- **`Resume (RES)`**: fa riferimento alla possibilità, da parte del tool, di riprendere un'analisi multi progetto, interrotta precedentemente, dall'ultimo progetto contenuto nel log di esecuzione.

3 Scelte e Combinazioni

3.1 Analisi

CodeSmile è capace di analizzare i codebase per rilevare la presenza di code smell. Di conseguenza, per realizzare questa funzionalità, è necessario considerare entrambi i parametri identificati nella fase precedente.

Categoria	**Scelte**
Numero di File (NF)	1. 0 [error] 2. 1 3. >1
Estensione del File (EF)	1. file Python [processato correttamente] 2. altro [ignorato]

Tabella 1: Parametro File

Categoria	**Scelte**
Numero dei Progetti (NP)	1. 1 2. >1
Struttura del Progetto (SP)	1. singola 2. annidata

Tabella 2: Parametro Struttura

Categoria	**Scelte**
Numero di Code Smells (NCS)	1. 0 [se processato correttamente] 2. 1 [se processato correttamente, proprietà SMELL] 3. >1 [se processato correttamente, proprietà SMELL]
Tipo di Code Smell (TCS)	1. Generico [se SMELL] 2. Specifico per API [se SMELL] 3. Altro [se SMELL]

Tabella 3: Parametro Smell

Categoria	Scelte
Modalità di Esecuzione (ME)	1. GUI 2. CLI
Esecuzione Parallela (EP)	1. True [proprietà Parallel] 2. False
Numero di Walkers (NW)	1. <5 [se Parallel] 2. 5 Default [se Parallel] 3. >5 [se Parallel]
Errore (ERR)	1. File non leggibile 2. Percorso mancante 3. Interruzione durante l'esecuzione
Resume (RES)	1. True 2. False

Tabella 4: Parametro Configurazione del Tool

3.2 Report

Categoria	Scelte
Numero di Code Smells (NCS)	1. 0 2. 1 [proprietà SMELL] 3. >1 [proprietà SMELL]
Tipo di Code Smell (TCS)	1. Generico [se SMELL] 2. Specifico per API [se SMELL] 3. Altro [se SMELL]

Tabella 5: Parametro Smell

Categoria	Scelte
Errore (ERR)	1. File non leggibile 2. Interruzione durante l'esecuzione

Tabella 6: Parametro Configurazione del Tool

3.3 Applicazione del criterio Weak Equivalence Class

Nel processo di definizione delle categorie per la funzionalità di reporting e di analisi, è stato applicato il criterio del Weak Equivalence Class. Questo criterio prevede che, per ciascuna categoria identificata, vengano selezionati rappresentanti di almeno una scelta da ogni classe di equivalenza. Tale approccio garantisce che ogni

classe di equivalenza sia testata almeno una volta, riducendo il numero di test necessari senza compromettere la copertura delle possibili combinazioni di input.

In particolare, per ciascuna categoria di input (ad esempio, Numero di File (NF), Estensione del File (EF), ecc.), sono stati individuati i valori rappresentativi delle diverse classi di equivalenza. I test case sono stati quindi costruiti combinando questi valori in modo da coprire tutti i comportamenti significativi attesi dal sistema, minimizzando al contempo il numero di configurazioni testate.

4 Test Frame

Di seguito riportiamo i Test Frame (con i relativi oracoli) generati adoperando le categorie individuate precedentemente e scegliendo uno dei valori possibili per ciascuna di esse. Se l'alias di una categoria è affiancato al numero 0, significa che non è stato individuato un valore valido. Tale scelta è risultata dal seguente razionale: nel caso del parametro Smell, risulterebbe superfluo considerare una la categoria NCS, se si considera EF2 del parametro File.

4.1 Analisi

Test Case ID	Test Frame	Oracolo
TC_1	NF0, EF0, NP0, SP0, NCS0, TCS0, ME2, EP0, NW0, ERR2, RES0	Il tool restituisce un errore poichè il percorso di input/output è mancante.
TC_2	NF3, EF1, NP1, SP2, NCS3, TCS2, ME2, EP1, NW3, ERR3, RES0	Il tool rileva correttamente code smells API-specifici nei files Python di un progetto a struttura annidata.
TC_3	NF1, EF0, NP0, SP0, NCS0, TCS0, ME1, EP2, NW0, ERR1, RES0	Il tool restituisce un errore poiché il file è non leggibile. Nessun code smell è rilevato.
TC_4	NF3, EF1, NP2, SP2, NCS3, TCS1, ME1, EP1, NW2, ERR0, RES0	Il tool analizza correttamente più files Python e rileva code smells generici in più progetti a struttura annidata.
TC_5	NF1, EF0, NP1, SP1, NCS1, TCS0, ME1, EP0, NW0, ERR0, RES0	Il tool non rileva alcun code smell in quanto il progetto a singola direcorry è vuoto.
TC_6	NF3, EF1, NP1, SP2, NCS1, TCS0, ME2, EP0, NW0, ERR0, RES0	Il tool analizza correttamente più files Python in un progetto a struttura annidata, senza rilevare code smells.
TC_7	NF2, EF1, NP1, SP2, NCS3, TCS2, ME2, EP0, NW0, ERR0, RES0	Il tool rileva correttamente più code smells API-specifici in un file Python in un progetto con struttura a singola directory
TC_8	NF3, EF1, NP2, SP1, NCS1, TCS0, ME2, EP1, NW2, ERR0, RES0	Il tool analizza correttamente più file Python in molteplici progetti a singola directory, senza rilevare nessun code smell.
TC_9	NF1, EF1, NP1, SP1, NCS2, TCS1, ME1, EP2, NW0, ERR0, RES0	Il tool rileva correttamente un code smell generico nel file Python in un progetto a singola directory.
TC_10	NF1, EF1, NP1, SP2, NCS3, TCS2, ME1, EP1, NW1, ERR0, RES0	Il tool analizza correttamente un file Python in un progetto a struttura annidata, rilevando più code smells API-specifici.
TC_11	NF1, EF1, NP2, SP1, NCS2, TCS2, ME1, EP1, NW3, ERR2, RES0	Il tool, durante l'analisi di un file Python con un code smell specifico per API in progetti multipli e struttura singola, è stato interrotto.
TC_12	NF3, EF1, NP1, SP2, NCS3, TCS3, ME2, EP1, NW1, ERR0, RES0	Il tool analizza correttamente più file Python e rileva due code smells, uno generico e uno specifico per API, con un progetto annidato.
TC_13	NF3, EF1, NP2, SP2, NCS0, TCS0, ME2, EP1, NW3, ERR1, RES0	Il tool restituisce un errore poiché i files Python dei vari progetti annidati risultano non leggibili.
TC_14	NF1, EF1, NP1, SP2, NCS2, TCS2, ME1, EP2, NW0, ERR0, RES0	Il tool analizza correttamente un file Python con uno smell API-specifico in un progetto con struttura annidata.
TC_15	NF1, EF2, NP1, SP1, NCS0, TCS0, ME1, EP2, NW0, ERR0, RES0	Il file con estensione diversa da '.py', all'interno di un progetto con un'unica directory, viene ignorato. Nessun code smell viene rilevato.
TC_16	NF3, EF1, NP2, SP2, NCS3, TCS3, ME2, EP1, NW2, ERR2, RES0	L'esecuzione si interrompe durante l'analisi di più files Python in piu progetti con struttura annidata. Il tool segnala l'interruzione durante l'esecuzione
TC_17	NF3, EF1, NP2, SP2, NCS0, TCS0, ME2, EP0, NW0, ERR1, RES0	Più files Python non sono leggibili in piu progetti con struttura annidata. Nessun code smell viene rilevato dai file non processabili.
TC_18	NF3, EF1, NP2, SP2, NCS0, TCS0, ME1, EP1, NW2, ERR0, RES0	Il tool analizza correttamente più file Python in più progetti annidati senza rilevare alcun code smell.
TC_19	NF3, EF1, NP1, SP1, NCS3, TCS3, ME2, EP2, NW0, ERR0, RES0	Il tool rileva correttamente sia code smells generici che specifici in un progetto con struttura semplice.
TC_20	NF2, EF1, NP1, SP2, NCS2, TCS2, ME1, EP2, NW0, ERR0, RES0	Il tool rileva correttamente un code smell API-specifico in un file Python di un progetto con struttura semplice.
TC_21	NF3, EF2, NP2, SP2, NCS0, TCS0, ME2, EP1, NW3, ERR0, RES0	I files con estensione non supportata all'interno di più progetti con strutture annidate vengono ignorati. Nessun code smell viene rilevato.

Tabella 7: Test Frame associati ai Test Case

4.2 Report

Test Case ID	Test Frame	Oracolo
TC_1	NCS0, TCS0, ERR0	I reports risultano vuoti poiché non ci sono code smells da aggregare.
TC_2	NCS1, TCS1, ERR0	I reports mostrano un singolo code smell di tipo "Generico" correttamente aggregato per "name_smell" o "project_name".
TC_3	NCS2, TCS2, ERR0	I reports mostrano due code smells di tipo "Specifico per API" correttamente aggregati per "name_smell" o "project_name".
TC_4	NCS1, TCS1, ERR2	I reports non vengono generati a causa di un'interruzione durante l'esecuzione.
TC_5	NCS3, TCS1, ERR0	I reports mostrano correttamente i dati per più progetti, con code smells aggregati in modo corretto per "project_name" o "name_smell".

Tabella 8: Test Frame associati ai Test Case