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 **`Contenuto del File`**: fa riferimento agli aspetti che riguardano l'input fornito 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.
  - **`Codifica del File (ENC)`**: fa riferimento all'encoding del file, che può essere UTF-8, ASCII, Unicode, o altro.
  - **`Dimensione del File (DF)`**: fa riferimento alla grandezza del file, che può essere piccola, media, grande o vuota.
- Parametro **`Complessità`**: fa riferimento agli aspetti 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.
  - **`Struttura del Progetto (SP)`**: fa riferimento alla complessità della struttura della directory (singola, annidate)
- Parametro **`Configurazione del Tool`**: fa riferimento ad aspetti di configurazione del tool;
  `Categorie`
  - **`Modalità di Esecuzione (ME)`**: fa riferimento all'uso del tool tramite interfaccia grafica o riga di comando.
  - **`Numero dei Progetti (NP)`**: fa riferimento alla scelta di analizzare un singolo progetto o più progetti contemporaneamente.
  - **`Numero di Walkers (NW)`**: fa riferimento alla quantità di walkers che compiono l'analisi.

- **`Errore (ERR)`**: fa riferimento agli eventuali errori restituiti durante l'esecuzione del tool
- **`Log (LOG)`**: fa riferimento alla presenza e alla completezza del log generato dal tool.

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]
Dimensione del File (DF)	1. Vuoto (0 KB) 2. Piccolo (>0 e <10 KB) 3. Medio (>10 e <100 KB) 4. Grande (>100 KB)
Encoding del File (ENC)	1. UTF-8 2. ASCII 3. Unicode

Tabella 1: Parametro Contenuto del File

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]
Struttura del Progetto (SP)	1. singola 2. annidata

Tabella 2: Parametro Complessità

Categoria	Scelte
Modalità di Esecuzione (ME)	1. GUI 2. CLI
Numero dei Progetti (NP)	1. 1 2. >1
Numero di Walkers (NW)	1. <5 2. 5 Default 3. >5
Errore (ERR)	1. File non leggibile 2. Percorso mancante 3. Interruzione durante l'esecuzione
LOG	1. generato correttamente 2. mancante o incompleto

Tabella 3: Parametro Configurazione del Tool

3.2 Report

Categoria	Scelte
Dimensione del File (DF)	1. Vuoto (0 KB) 2. Piccolo (>0 e <10 KB) 3. Medio (>10 e <100 KB) 4. Grande (>100 KB)
Encoding del File (ENC)	1. UTF-8 2. ASCII 3. Unicode

Tabella 4: Parametro Contenuto del File

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 Complessità

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 Criterion. 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), Dimensione del File (DF), 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 Complessità, risulterebbe superfluo considerare una la categoria NCS, se si considera EF2 del parametro Contenuto del File.

4.1 Analisi

Test Case ID	Test Frame	Oracolo
TC_1	NF1, EF0, DF1, ENC1, NCS1, TCS0, SP1, ME2, NP1, NW1, ERR1, LOG2	Il tool restituisce un errore poiché non ci sono file da analizzare. Nessun log viene generato
TC_2	EF1, DF3, ENC2, NCS2, TCS1, SP1, ME2, NP1, NW2, ERR2, LOG2	Il tool restituisce un errore poichè il percorso di input/output è mancante. Il log è mancante o incompleto
TC_3	NF2, EF1, DF4, ENC3, NCS3, TCS2, SP2, ME1, NP2, NW3, ERR3, LOG1	Il tool rileva correttamente code smells API-specifici nei file di dimensione grande e struttura annidata. Il log è generato correttamente e segnala l'interruzione durante l'esecuzione
TC_4	NF2, EF1, DF3, ENC1, NCS1, TCS0, SP1, ME2, NP1, NW1, ERR1, LOG2	Il tool restituisce un errore poiché il file è non leggibile. Nessun code smell è rilevato e il log è mancante o incompleto
TC_5	NF3, EF1, DF3, ENC2, NCS2, TCS1, SP1, ME2, NP2, NW2, ERR0, LOG1	Il tool analizza correttamente più file Python di dimensione media e rileva code smells generici. Il log è generato correttamente
TC_6	NF1, EF0, DF1, ENC0, NCS1, TCS0, SP2, ME1, NP1, NW1, ERR0, LOG1	Il tool non rileva alcun code smell in quanto il progetto è vuoto. Il log è generato correttamente poichè l'esecuzione è comunque portata a termine
TC_7	NF3, EF1, DF4, ENC1, NCS1, TCS0, SP2, ME2, NP2, NW3, ERR0, LOG1	Il tool analizza correttamente più di un file Python di dimensione grande e struttura annidata, senza rilevare code smells. Il log è generato correttamente
TC_8	NF2, EF1, DF2, ENC3, NCS3, TCS2, SP1, ME1, NP1, NW2, ERR0, LOG1	Il tool rileva correttamente code smells API-specifici in un file di dimensione media in un progetto con struttura a singola directory. Il log è generato correttamente
TC_9	NF3, EF1, DF4, EN2, NCS1, TCS0, SP1, ME2, NP2, NW1, ERR0, LOG1	Il tool analizza correttamente più file Python in molteplici progetti a singola directory, senza rilevare nessun code smell. Il log è generato correttamente
TC_10	NF2, EF1, DF2, ENC1, NCS2, TCS1, SP1, ME1, NP1, NW2, ERR0, LOG1	Il tool rileva correttamente un code smell generico nel file Python di dimensione media in un solo progetto a singola directory. Il log è generato correttamente
TC_11	NF2, EF1, DF4, ENC2, NCS3, TCS1, SP2, ME1, NP1, NW2, ERR0, LOG1	Il tool analizza correttamente un file Python di grande dimensione in una struttura annidata, rilevando più code smells API-specifici. Il log è generato correttamente.
TC_12	NF2, EF1, DF3, ENC1, NCS1, TCS2, SP1, ME2, NP2, NW3, ERR1, LOG2	Il tool durante l'analisi del file di dimensione media con un code smell specifico per API in un progetto multiplo e struttura singola è stata interrotta. Il log è generato correttamente e segna l'interruzione dell'esecuzione.
TC_13	NF3, EF1, DF2, ENC3, NCS3, TCS3, SP2, ME1, NP1, NW3, ERR0, LOG1	Il tool analizza correttamente più file Python di dimensione piccola e rileva due code smells, uno generico e uno specifico per API, con progetto annidato. Il log è generato correttamente.
TC_14	NF3, EF1, DF3, ENC2, NCS0, TCS0, SP1, ME2, NP2, NW1, ERR1, LOG2	Il tool restituisce un errore poiché i files python di dimensione media dei vari progetti risultano non leggibili. Il log è mancante o incompleto.
TC_15	NF2, EF1, DF2, ENC1, NCS2, TCS2, SP2, ME1, NP1, NW3, ERR0, LOG1	Il tool analizza correttamente un file Python di piccola dimensione con uno smell API-specifico in una struttura annidata con più walkers. Il log è generato correttamente.
TC_16	NF2, EF2, DF2, ENC1, NCS0, TCS0, SP1, ME2, NP1, NW2, ERR0, LOG1	Il file con estensione diversa da '.py' viene ignorato. Nessun code smell viene rilevato. Il log è generato correttamente.
TC_17	NF3, EF1, DF4, ENC2, NCS3, TCS2, SP2, ME1, NP2, NW3, ERR3, LOG2	L'esecuzione si interrompe durante l'analisi di più file grandi in piu progetti con struttura annidata. I code smells rilevati fino al momento dell'interruzione vengono riportati. Il log è incompleto.
TC_18	NF3, EF1, DF3, ENC2, NCS1, TCS0, SP2, ME2, NP2, NW2, ERR1, LOG2	Più file non sono leggibili in piu progetti con struttura annidata. Nessun code smell viene rilevato dai file non processabili. Il log è incompleto.

Tabella 7: Test Frame associati ai Test Case

4.2 Report

Test Case ID	Test Frame	Oracolo
TC_1	DF1, ENC1, NCS0, TCS0, ERR0	Il report risulta vuoto poiché non ci sono code smells da aggregare.
TC_2	DF2, ENC2, NCS1, TCS1, ERR0	Il report mostra un singolo code smell di tipo "Generico" correttamente aggregato per "name_smell" o "project_name".
TC_3	DF3, ENC3, NCS2, TCS2, ERR0	Il report mostra due code smells di tipo "Specifico per API" correttamente aggregati "per name_smell" o "project_name".
TC_4	DF3, ENC1, NCS1, TCS1, ERR2	Il report non viene generato correttamente a causa di un'interruzione durante l'esecuzione.
TC_5	DF4, ENC2, NCS3, TCS1, ERR0	Il report mostra 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