UNIVERSITÀ DEGLI STUDI DI SALERNO

DIPARTIMENTO DI INFORMATICA

Ingegneria del Software: Tecniche Avanzate

Progetto ISTA – Evoluzione di CodeSmile

Change Requests

Professore:
Ch.mo Prof.
Andrea De Lucia

Tutor di riferimento:
Giammaria Giordano
Gilberto Recupito

Studenti:
Antonio D'Auria
Mat. NF22500063
Emanuele D'Auria
Mat. 0522501918
Luca Casillo
Mat. NF22500037

ANNO ACCADEMICO 2025/2026

Indice

Indice
1 Motivazione delle Modifiche 1
2 Change Requests 2
2.1 CR1 — Generazione del Call Graph (Enhancement) 2
2.2 CR2 — Estensione della CLI con Opzioni Avanzate (Modifica Perfettiva) 2
2.3 CR3 — Portabilità tra Ambiente Locale e Docker (Modifica Adattiva) 3
2.4 CR4 — Visualizzazione del Call Graph nella WebApp (Enhancement) 3

1 Motivazione delle Modifiche

Nel corso dell'analisi preliminare di CodeSmile sono emerse alcune aree in cui il sistema può essere ulteriormente esteso o perfezionato al fine di migliorare la comprensione dei risultati dell'analisi statica, la configurabilità degli strumenti disponibili e la portabilità del progetto nei diversi ambienti di esecuzione.

Sulla base dello studio del codice e dell'osservazione dei flussi operativi previsti dal tool, sono stati individuati quattro interventi che rafforzano le capacità già offerte da CodeSmile:

- **Analisi delle dipendenze interne**: attualmente il tool rileva correttamente gli ML-specific code smells, ma non fornisce una rappresentazione delle relazioni tra funzioni e file. L'introduzione di un Call Graph consentirebbe una comprensione più profonda della struttura del progetto e del contesto in cui gli smell compaiono.
- **Integrazione dei nuovi risultati nella WebApp**: al fine di rendere più immediata la consultazione delle informazioni strutturali prodotte dall'analisi, si propone di estendere la WebApp con una sezione dedicata alla visualizzazione interattiva del Call Graph generato.
- **Miglioramento della CLI**: la Command Line Interface offre le funzionalità necessarie per eseguire l'analisi statica, ma può essere arricchita con opzioni aggiuntive che ne aumentino la configurabilità. In particolare, si propone di introdurre parametri per attivare o disattivare la generazione del Call Graph, specificare il percorso di output per gli artefatti prodotti, personalizzare il formato del report e facilitare l'esclusione di directory non rilevanti dall'analisi.
- **Uniformazione dell'esecuzione locale e in Docker**: alcuni servizi richiedono modifiche manuali al codice per adattarsi alle due modalità di esecuzione. Una gestione centralizzata e automatica aumenterebbe la portabilità e ridurrebbe le possibilità di errore.

Le Change Request proposte intendono quindi estendere il sistema senza alterarne il comportamento attuale, introducendo strumenti e miglioramenti volti a facilitare l'analisi, la navigazione e l'utilizzo del tool nei diversi contesti di esecuzione.

2 Change Requests

In questa sezione vengono descritte nel dettaglio le modifiche proposte per l'estensione e il miglioramento del sistema CodeSmile. Le Change Request sono state formulate sulla base delle osservazioni emerse durante la fase di comprensione, con l'obiettivo di ampliare le capacità analitiche del tool, migliorarne l'usabilità e incrementarne la portabilità nei diversi ambienti di esecuzione. Le proposte sono organizzate in quattro interventi distinti: due di tipo *enhancement*, uno *perfettivo* e uno *adattivo*.

Le Change Request sono inoltre presentate nell'ordine in cui si prevede di realizzarne l'implementazione. Tale sequenza riflette le dipendenze tra le modifiche proposte e garantisce una progressione coerente delle attività di evoluzione del sistema.

2.1 CR1 — Generazione del Call Graph (Enhancement)

Descrizione. Si propone l'introduzione di un modulo dedicato alla generazione di un *Call Graph* all'interno del processo di analisi statica. Il nuovo componente dovrà analizzare la struttura dei file Python forniti in input, identificando le relazioni di chiamata tra funzioni, metodi e moduli tramite l'analisi dell'AST. Il risultato sarà una rappresentazione strutturata delle dipendenze interne al progetto analizzato.

Motivazione. Sebbene CodeSmile identifichi con precisione le occorrenze dei machine learning-specific code smells, non offre attualmente una vista che mostri il contesto strutturale in cui tali smell compaiono. Il Call Graph permette di comprendere l'organizzazione del codice e di analizzare la propagazione potenziale degli effetti degli smell all'interno delle parti del progetto.

Deliverable.

- Modulo dedicato alla costruzione del Call Graph tramite AST.
- Artefatto di output (es. file JSON o equivalente) contenente il grafo.
- Integrazione del modulo nel flusso di analisi statica esistente.

2.2 CR2 — Estensione della CLI con Opzioni Avanzate (Modifica Perfettiva)

Descrizione. Si propone l'estensione dell'interfaccia a linea di comando (`CLI`) con opzioni aggiuntive che migliorino la configurabilità del processo di analisi. In particolare, si intendono introdurre parametri che permettano di attivare o disattivare la generazione del Call Graph, specificarne il percorso di output, scegliere il formato del report degli smell e facilitare l'esclusione di percorsi o directory non rilevanti.

Motivazione. La CLI attuale fornisce le funzionalità essenziali per l'esecuzione dell'analisi statica, ma può essere ulteriormente migliorata per supportare scenari d'uso più flessibili. L'introduzione di opzioni avanzate permette agli utenti di personalizzare il flusso di analisi senza intervenire direttamente sul codice, aumentando l'usabilità complessiva del tool.

Deliverable.

- Nuove opzioni CLI, tra cui:
  - `--enable-callgraph` per attivare la generazione del Call Graph;
  - `--callgraph-output <path>` per specificare il percorso del file generato;
  - `--exclude-paths <paths>` per escludere directory dall'analisi;

  - `--format {csv,json}` per scegliere il formato del report degli smell.
- Aggiornamento della documentazione CLI.

2.3 CR3 — Portabilità tra Ambiente Locale e Docker (Modifica Adattiva)

Descrizione. Attualmente l'esecuzione dei servizi backend di CodeSmile richiede modifiche manuali al codice per funzionare correttamente in ambiente locale o tramite Docker, a causa di differenze nei percorsi e nelle strutture di esecuzione. La presente Change Request propone l'introduzione di una gestione uniforme dei percorsi e della configurazione, in modo da garantire il corretto funzionamento del sistema in entrambi gli ambienti senza dover effettuare interventi sul codice.

Motivazione. La necessità di adattare manualmente alcune parti del codice per cambiare ambiente di esecuzione rappresenta una potenziale fonte di errore e riduce la portabilità del software. Automatizzare tale processo migliora l'affidabilità del sistema e semplifica il flusso di lavoro degli sviluppatori.

Deliverable.

- Codice uniforme utilizzabile sia in esecuzione locale sia in Docker.
- Documentazione aggiornata per l'avvio dei servizi.

2.4 CR4 — Visualizzazione del Call Graph nella WebApp (Enhancement)

Descrizione. Si propone l'estensione della WebApp tramite l'aggiunta di una sezione dedicata alla visualizzazione del Call Graph generato dall'analisi statica. La visualizzazione sarà realizzata mediante una libreria frontend per grafi, che consente di rappresentare in modo chiaro ed intuitivo le relazioni di chiamata tra funzioni o metodi del progetto. La pagina permetterà di esplorare la struttura del codice attraverso funzionalità interattive e strumenti grafici dedicati.

Motivazione. Integrare il Call Graph nella WebApp consente di affiancare alla rappresentazione testuale dei ML-specific code smells una vista strutturale che mostra il contesto in cui tali smell si manifestano. Una visualizzazione grafica interattiva rende immediata la comprensione dei rapporti tra funzioni, delle possibili propagazioni e dell'impatto locale degli smell rilevati, migliorando la capacità del tool di supportare attività di manutenzione ed evoluzione.

Deliverable.

- Visualizzazione strutturale del Call Graph.
- Colorazione intelligente dei nodi:
  - Rosso: nodi contenenti almeno uno smell;
  - Arancione: nodi che chiamano funzioni smelly;
  - Verde: nodi privi di smell.
- Interazioni base: controlli di zoom, pan, minimappa.
- Selezione interattiva dei nodi, al click su un nodo:
  - Il nodo selezionato viene evidenziato;
  - Gli archi a esso collegati cambiano stile;
  - Viene mostrata una sidebar contenente informazioni utili quali: identificativo, file di appartenenza e lista degli smell associati.

- Filtri dinamici tramite semplici checkbox:
  - Mostra nodi smelly;
  - Mostra nodi che dipendono da nodi smelly;
  - Mostra nodi clean.
- Esportazione dei risultati:
  - Download del JSON del Call Graph;
  - Esportazione del grafo come immagine.