UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Informatica

Corso di Laurea Magistrale in Informatica

REPORT FINALE DEL PROGETTO

Ingegneria del Software: Report Finale del Progetto

Luigi Auriemma

Matricola: NF22500161

Ivan Chiello

Matricola: NF22500167

Repository: `https://github.com/LuigiAuriemma/smell_ai.git`

Anno Accademico 2025-2026


sesa
lab
SOFTWARE ENGINEERING
SALERNO

Indice

CAPITOLO 1

Introduzione

In questo documento viene illustrato il percorso evolutivo del sistema CodeSmile, descrivendo le attività di manutenzione ed estensione svolte a partire dall'architettura esistente. Il documento sintetizza le fasi di analisi preventiva, l'implementazione delle richieste di modifica (Change Requests) e la successiva validazione tramite test, evidenziando come la qualità del software sia stata preservata e migliorata.

- Evoluzione Strutturale (CR01): La CLI è stata potenziata con un modulo per la generazione automatica dei Call Graph, consentendo l'esportazione delle dipendenze del codice in diversi formati.
- Innovazione nell'Interfaccia (CR02): La Web Application è stata arricchita con una dashboard interattiva per la visualizzazione dei grafi, migliorando drasticamente la capacità dell'utente di diagnosticare visivamente le relazioni tra i componenti e gli smell rilevati.

CAPITOLO 2

Sviluppo del Progetto

CodeSmile si configura come uno strumento avanzato per l'analisi del codice, specializzato nell'identificazione di Code Smells tipici dei progetti di Machine Learning. Il sistema integra tecniche di analisi statica tradizionale (basata su AST) con funzionalità sperimentali basate sull'Intelligenza Artificiale.

2.1 Componenti Core (CLI/GUI e Web App)

La logica applicativa risiede in un nucleo centrale modulare, organizzato in package distinti (cli, components, code_extractor, detection_rules). Questa architettura gestisce il flusso di analisi, l'estrazione delle metriche e l'applicazione delle regole di detection. Tale suddivisione ha permesso di introdurre le nuove funzionalità di tracciamento (Call Graph) come moduli aggiuntivi, minimizzando l'impatto sul codice legacy.
La controparte web adotta un'architettura Gateway-Services sviluppata con FastAPI per il backend e Next.js per il frontend. Questa separazione netta ha facilitato l'implementazione della CR02, permettendo di agire indipendentemente sul servizio di backend e sull'interfaccia utente.

2.2 Change Requests

Questa sezione riassume le Change Requests (CR) utili all'evoluzione di CodeSmile.Le modifiche hanno l'obiettivo di migliorare e estendere l'analisi dei progetti e la visualizzazione dei risultati ottenuti.

ID	**Titolo**	**Categoria**	**Esito**
CR01	Creazione di un Call graph per l'analisi delle dipendenze dei file contententi smell nella CLI	Perfective	Rilasciata
CR02	Visualizzazione del Call Graph con Code Smells nella Web App	Perfective	Rilasciata

CR01 - Creazione di un Call graph per l'analisi delle dipendenze dei file contententi smell nella CLI
Obiettivo: estensione della CLI attraverso una funzionalità che permette la generazione automatica di un Call Graph nel processo di analisi dei file contenenti smell. Il sistema dovrà mappare le dipendenze tra i file analizzati inseriti dall'utilizzatore dello strumento, producendo un output chiaro e strutturato che mostra la propagazione dei code smells.

CR02 - Visualizzazione del Call Graph con Code Smells nella Web App
Obiettivo: integrazione nella dashboard web di un visualizzatore di grafi. L'utente dovrà poter visualizzare i nodi e gli archi, con evidenziazione grafica per i componenti contenenti smell. Cliccando su un nodo, dovranno essere mostrati i dettagli dell'analisi specifica.

2.3 Strategia di Testing

La verifica del sistema ha seguito un approccio incrementale definito nel Master Test Plan, coprendo sia le nuove funzionalità che la non-regressione dell'esistente.

2.3.1 Gestione degli Incidenti (Pre-Modification)

Durante la fase preliminare di testing, è emerso un comportamento errato da parte di test presenti nella suite di test. I test sono stati riscritti per adattarsi alla configurazione dello strumento prima delle modifiche.

2.3.2 Testing delle Nuove Funzionalità (Post-Modification)

CLI (CR01): Sono stati eseguiti test funzionali per accertare la corretta generazione fisica dei file .dot e .json e l'assenza di impatti negativi sulle performance dell'analisi standard.
Web App (CR02): È stata introdotta una suite di test End-to-End (E2E) specifici per verificare l'interattività del grafo, inclusa la corretta colorazione dei nodi "smelly" e la risposta ai click dell'utente.

2.3.3 Esito della Regressione

La riesecuzione completa della suite di test storica ha confermato che le modifiche non hanno introdotto difetti nelle logiche core. Tutti i test di regressione si sono conclusi con esito positivo, validando la robustezza dell'approccio modulare adottato.