Laurea Magistrale in Informatica - Università di Salerno

Corso di Ingegneria del Software Tecniche Avanzate - Prof. Andrea De Lucia

Progetto ISTA - 2025-2026

MARK 2.0 Plus

https://github.com/vmedica/MARK-2.0-Plus

Cerchia Giovanni (NF22500202)
Medica Vincenzo (NF22500203)

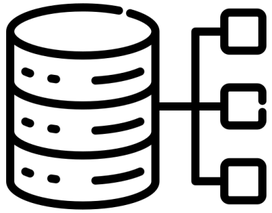
Roadmap

1. Introduzione
2. Change Requests
3. Attività pre-modifiche
4. Impact Analysis
5. Testing
6. Esito

Contesto

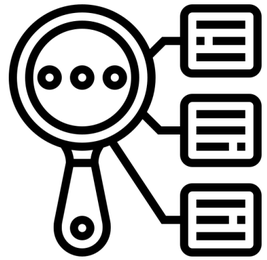
Problema

- Dataset ML affidabili da GitHub
- Selezione manuale costosa
- Bias



Soluzione

- MARK: classificazione automatica
- Analisi statica
- Supporto ai ricercatori



MARK 2.0: funzionamento

Analisi statica di repository Python

Scan repository Python

Scansione ricorsiva dei file Python del progetto

Librerie & keyword

Rilevazione librerie ML e keyword appartenenti ad una knowledge-base

Regole euristiche e classificazione

Applicazione di regole euristiche e assegnamento classe: Producer o Consumer

2 funzionalità principali:

- **Cloning** → Da GitHub data una lista di nomi di repo
- **Analisi** → e classificazione in Producer/Consumer data una directory

Change Requests (Enhancement)

Additive / Perfective — ISO/IEC/IEEE 14764:2022

CR1

Metriche qualità (Radon)

- Aggiunge CC e MI aggregati a livello progetto
- Supporta interpretazione risultati e selezione repo più manutenibili

CR2

GUI (Tkinter)

- Elimina parametri hard-coded nel main
- Configura input/step e consulta risultati da interfaccia

CR3

Dashboard

- Grafici + statistiche aggregate per run
- Riduce lettura/interpretazione manuale dei file di output

Pre-Modification System Testing (Baseline)

Category Partition → Regression baseline

UC Analisi

Categorie:

- Esistenza input directory
- Contenuto directory
- Cardinalità Producer
- Cardinalità Consumer

Combinazioni iniziali: **54**

Test case (finali): **12**

UC Cloning

Categorie:

- Esistenza file CSV
- Contenuto file CSV

Combinazioni iniziali: **12**

Test case (finali): **4**

Esecuzione pre-modifica: tutti i test PASS — nessuna failure rilevata

Impact Analysis (CR1)

Approccio call-graph based

1

Artefatti & SLO

Artefatti disponibili: codice sorgente
SLO iniziali: metodi

2

Relazione di impatto

Impatto diretto: fan-in + fan-out
(propagazione bidirezionale)

3

SIS (CR1)

`MLAnalyzer`: `analyze_single_file`, `analyze_project`, `analyze_projects_set`

4

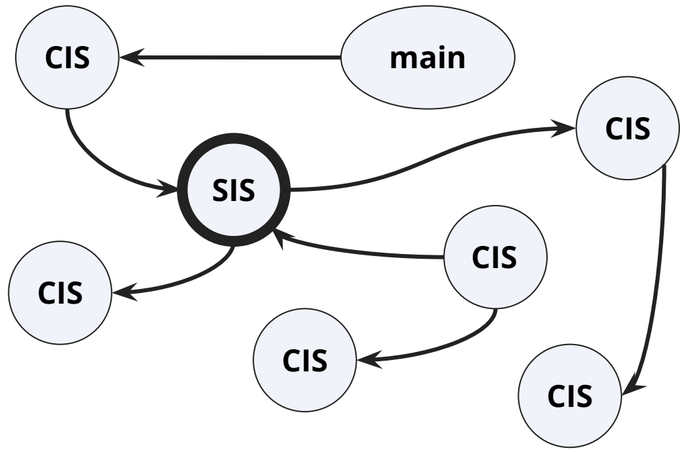
Call Graph

Espansione iterativa: caller + callee dei nodi aggiunti fino a main

5

Costruzione del CIS → AIS

Valutazione individuale dei nodi → CIS; confronto post-impl. con AIS



|CIS| = 13 componenti

|FP| = 3

Precision ≈ 0,77

Recall = 1,0

Analisi conservativa: priorità a non perdere impatti reali

Impact Analysis (CR2–CR3)

As-is

MARK 2.0
Sistema esistente

To be

GUI (CR2)
+ Dashboard (CR3)

MARK 2.0
Sistema esistente

L’impatto sui componenti esistenti è **NULLO**

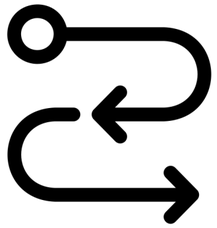
GUI e Dashboard sono implementati come moduli separati (client dei moduli del sistema esistente)
|SIS| = 0, |CIS| = 0

Planning: Post-Modification Testing

3 obiettivi di verifica

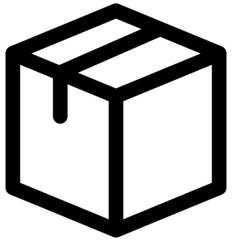
Unit e integration

Verifica whitebox dei metodi modificati/aggiunti



System testing

Verifica blackbox delle funzionalità aggiunte



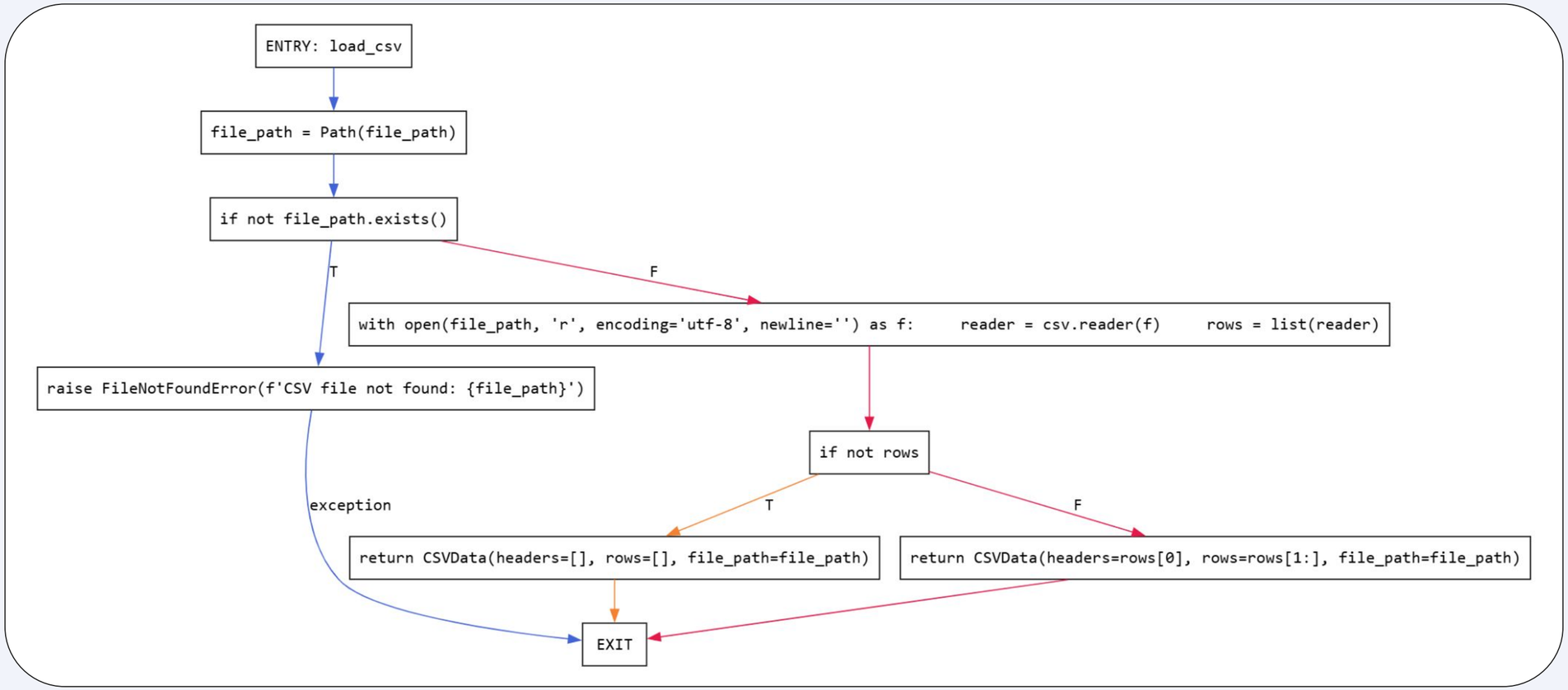
Regression

Assicurare assenza di regressioni rispetto alla baseline pre-mod

Unit & Integration Testing

- **Basic Unit Testing**: whitebox sui metodi target. Mock dove necessario (es. filesystem)
- **Integration Testing**: whitebox sugli stessi metodi ma senza mocking, verificando la corretta interazione con il sistema

La selezione degli input è stata guidata dalla costruzione del **CFG** dei metodi target (script Python basato su **AST**), scegliendo percorsi che coprono il maggior numero di branch (obiettivo della **branch coverage >= 80%**)



Risultati Unit ed Integration Testing

Risultati conseguiti nelle attività di Unit e Integration Testing, con il raggiungimento dell'obiettivo di una branch coverage ≥ 80% sulle classi oggetto di test.

Unit

WhiteBox - Risultati

Package/Classe	Branch coverage	Note
modules/analyzer/MLAnalyzer	98%	CR1
gui/services/pipeline_service.py	100%	CR2
gui/services/output_reader.py	93%	CR2/CR3
Totale	**97%**	

Integration

WhiteBox - Risultati

Package/Classe	Branch coverage	Note
modules/analyzer/MLAnalyzer	100%	CR1
gui/services/pipeline_service.py	100%	CR2
gui/services/output_reader.py	93%	CR2/CR3
gui/controller.py	85%	CR2/CR3
Totale	**95%**	

Post-Modification System Testing

Dopo aver svolto lo Unit e l'Integration Testing si è condotto il System Testing in modalità black-box, progettando i Casi di Test tramite Category Partition per ogni CR.

UC-CR1

Use Case - Metriche

Descrizione	L'utente esegue l'analisi su una directory locale contenente repository già disponibili; il tool calcola le metriche Maintainability Index (MI) e Cyclomatic Complexity (CC) per i progetti contenuti, salvando i risultati in `io/output/`.
Attori	Utente (ricercatore)
Entry condition	`main_args.py` disponibile; dipendenze installate (inclusa Radon); directory `--repository-path` accessibile (per scenari "success").
Exit condition	**Successo:** metriche (MI, CC) calcolate e risultati salvati in `io/output/`. **Errore:** messaggio d'errore e assenza di output significativo.
Flusso di eventi principale	1. Invocazione con `--repository-path` e `--metrics`. 2. Validazione del path. 3. Scansione dei repository nella directory. 4. Calcolo MI e CC sui progetti analizzati. 5. Salvataggio output metriche in `io/output/`.

Post-Modification System Testing

- Parametro: Path della directory contenente i repository (repository_path)
- Oggetti dell’ambiente: Filesystem, contenuto dei progetti della directory analizzata

UC-CR1

Categorie - Metriche

Categoria	ID	Descrizione	Proprietà / Vincoli
Esistenza input directory	ED1	Directory non esiste	[Error]
	ED2	Directory esiste	[property DirOk]
Contenuto directory	CD0	Directory vuota (0 progetti)	[if DirOk] [property Empty]
	CD1	Directory con 1+ progetti	[if DirOk] [property MultiProject]
Composizione Progetti	CP0	Tutti i progetti senza file Python	[if DirOk and if MultiProject]
	CP1	Tutti i progetti con file Python ma vuoti	[if DirOk and if MultiProject]
	CP2	Tutti i progetti con file Python validi e con codice	[if DirOk and if MultiProject]
	CP3	Mix di progetti validi e non validi	[if DirOk and if MultiProject]

UC-CR1

Test Frame - Metriche

ID	Combinazioni (categorie/scelte)	Oracolo (risultato atteso)
TF1	ED1	Visualizza messaggio di errore “Input folder not found”
TF2	ED2, CD0	Nessuna metrica calcolata
TF3	ED2, CD1, CP0	Per tutti i progetti i risultati dell’MI e della CC sono pari a 0
TF4	ED2, CD1, CP1	Per tutti i progetti i risultati dell’MI e della CC sono pari a 0
TF5	ED2, CD1, CP2	Per tutti i progetti i risultati dell’MI e della CC sono dei valori esatti
TF6	ED2, CD1, CP3	Per tutti i progetti i risultati dell’MI e della CC sono dei valori esatti

Post-Modification System Testing

UC-CR1

Test Case - Metriche

TC	Input (repository_path)	Descrizione ambiente	Oracolo
TC1	*"repo_does_not_exist"*	Directory inesistente	Messaggio “Input folder not found”
TC2	*"empty_repo"*	Directory vuota	Nessun metrica calcolata
TC3	*"test_repos/TC3"*	Più progetti senza file Python	Per tutti i progetti i risultati dell’MI e della CC sono pari a 0
TC4	*"test_repos/TC4"*	Più progetti con file Python vuoti	Per tutti i progetti i risultati dell’MI e della CC sono valori pari a 0
TC5	*"test_repos/TC5"*	Più progetti con file Python contenente codice valido	Valori esatti calcolati manualmente: **project1:** CC_avg = 1.67, MI_avg = 77.5 **project2:** CC_avg = 1.33, MI_avg = 88.75
TC6	*"test_repos/TC6"*	Più progetti senza file Python, con file Python vuoti e contenenti file Python con codice valido	Valori esatti per ogni progetto calcolati manualmente: **project_empty_python_1:** CC_avg = 0, MI_avg = 0; **project_empty_python_2:** CC_avg = 0, MI_avg = 0; **project_no_python_1:** CC_avg = 0, MI_avg = 0; **project_no_python_1:** CC_avg = 0, MI_avg = 0; **project1:** CC_avg = 1.67, MI_avg = 77.51; **project2:** CC_avg = 1.33, MI_avg = 88.75 .

Test eseguiti

Test

Tutti i test eseguiti

Fase	Tipo	Riferimento	Numero
Test eseguiti post-modification	**Unit Test**	CR1	8
		CR2	8
		CR3	2
	Integration Test	CR1	8
		CR2	13
		CR3	7
	System Test (CR1, CR2, CR3)	CR1	6
		CR2	12
		CR3	4
Regression Testing	**System Test Pre Modification**	UC-1 Analysis	12
		UC-2 Cloning	4
Totale			**84**

Risultati

Regression Testing

Test eseguiti	Superati	Falliti
16	16	0

Risultati

Risultati complessivi

Test eseguiti	Superati	Fallimenti	Incident / anomalie rilevate
84	84	0	nessuna

Conclusioni

L’estensione MARK 2.0 Plus ha prodotto i seguenti risultati:

GUI di settaggio (CR2)

Una GUI in Tkinter permette di configurare ed eseguire l’analisi senza usare la CLI né modificare manualmente il codi

Elevata copertura di test

Le componenti modificate raggiungono una branch coverage complessiva del 95% nei test white-box post-modification.

Stabilità del baseline garantita

Tutti i test di sistema pre-modification sono stati rieseguiti con successo, confermando l’assenza di regressioni.

Integrazione delle metriche di qualità (CR1)

Il core è stato esteso con il calcolo di CC e MI, con aggregazione a livello di progetto.

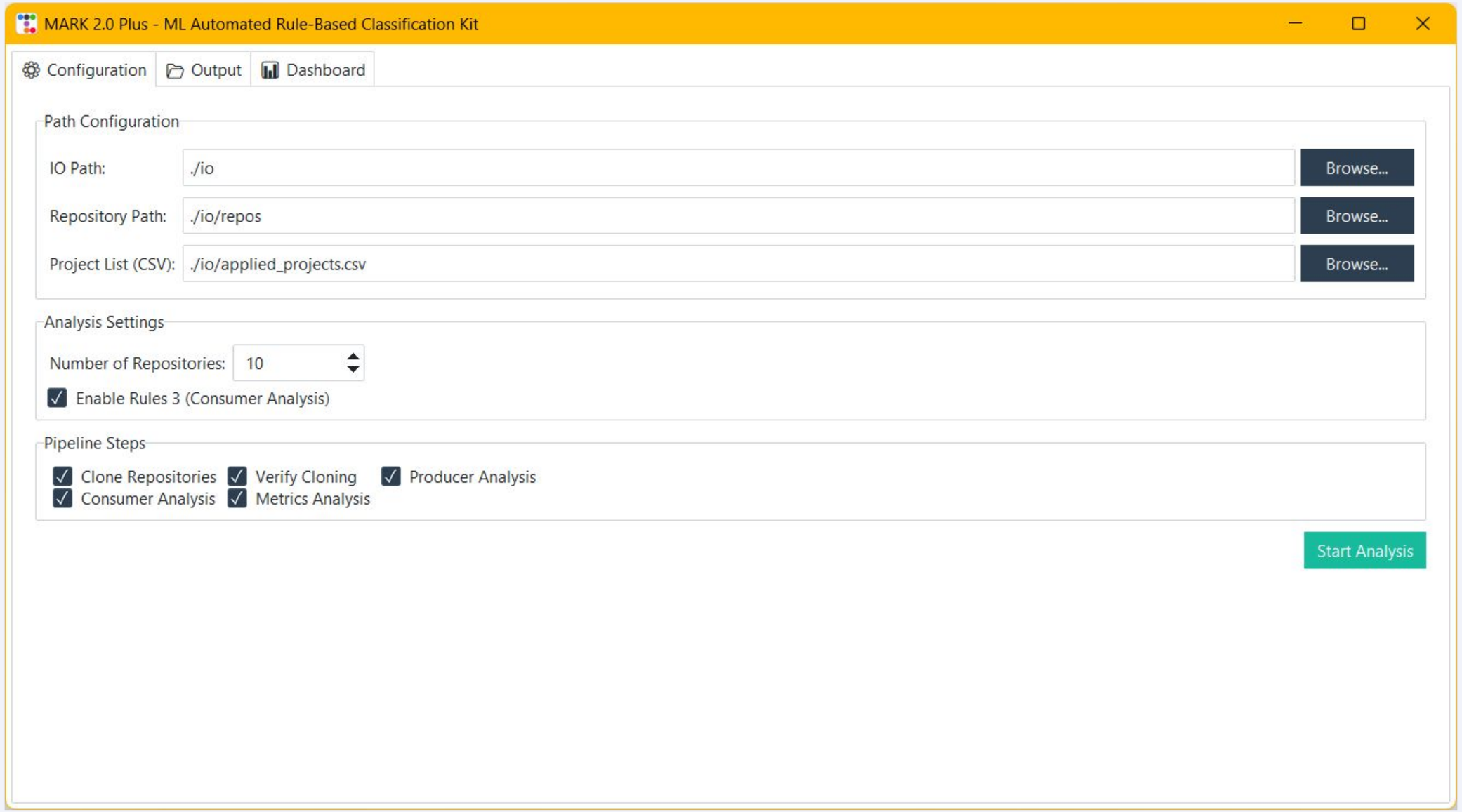
Dashboard grafica (CR3)

I risultati sono mostrati con grafici dinamici che facilitano l’interpretazione.

Baseline verificata

Sono stati definiti due casi d’uso (Analysis e Cloning) da cui è stata derivata una suite di 16 system test per il regression testing.

Demo


MARK 2.0 Plus - ML Automated Rule-Based Classification Kit
Configuration
Output
Dashboard
Path Configuration
IO Path:
./io
Browse...
Repository Path:
./io/repos
Browse...
Project List (CSV):
./io/applied_projects.csv
Browse...
Analysis Settings
Number of Repositories:
10
Enable Rules 3 (Consumer Analysis)
Pipeline Steps
Clone Repositories
Verify Cloning
Producer Analysis
Consumer Analysis
Metrics Analysis
Start Analysis

Laurea Magistrale in Informatica - Università di Salerno

Corso di Ingegneria del Software Tecniche Avanzate - Prof. Andrea De Lucia

Progetto ISTA - 2025-2026

Grazie dell'attenzione

MARK 2.0 Plus

https://github.com/vmedica/MARK-2.0-Plus

Cerchia Giovanni (NF22500202)
Medica Vincenzo (NF22500203)