DARTS – (Detection and Refactoring of Test Smells)

Maintenance

Versione	1.0
Data	28/10/2021
Destinatario	Prof. A. De Lucia et Al.
Presentato da	DT, GR, DDD

Composizione Team e Acronimi

Nome e Cognome	Matricola	Acronimo
Dario Tecchia	0522500736	DT
Gilberto Recupito	0522500842	GR
Dario Di Dario	0522500848	DDD

Revision History

Data	Versione	Descrizione	Autore
14/05/2021	0.1	Prima stesura	Team
14/09/2021	0.2	Impact Analysis	Team
02/10/2021	0.3	Analisi Post-Implementazione	Team
30/10/2021	0.4	Aggiunti Sviluppi Futuri	Team
25/10/2021	0.5	Aggiunta Activity Diagram	Team
30/10/2021	0.6	Aggiunti Sviluppi Futuri	Team
28/10/2021	1.0	Formattazione	Team

Sommario

1 Scopo del documento 9

2 Panoramica del sistema attuale 9

2.1 Requisiti Funzionali 9

2.2 Attori 10

2.3 Design 10

2.4 Implementazione 10

2.4.1 Interface Layer 11

2.4.2 Application Layer 11

3 Analisi delle modifiche richieste 11

3.1 Conversione a Progetto Gradle 12

3.1.1 Descrizione 12

3.1.2 Problematiche affrontate 12

3.1.3 Impact Analysis 13

3.1.4 Studio di fattibilità 13

3.1.5 Analisi Post-Implementazione 14

3.2 Separazione Infrastruttura Plugin dalla logica Core 14

3.2.1 Descrizione 14

3.2.2 Problematiche affrontate 14

3.2.3 Impact Analysis 15

3.2.4 Studio di fattibilità 15

3.2.5 Analisi Post-Implementazione 15

3.3 Statistiche e Misurazioni 15

3.3.1 Descrizione 15

3.3.2 Problematiche affrontate 16

3.3.3 Impact Analysis 17

3.3.4 Studio di fattibilità 19

3.3.5 Analisi Post-Implementazione 20

3.4 Invio statistiche a un server remoto [Bonus] 20

3.4.1 Descrizione 20

3.4.2 Problematiche affrontate 21

3.4.3 Impact Analysis 22

3.4.4 Studio di fattibilità 22

3.4.5 Analisi Post-Implementazione 23

4 Architettura DARTS e DARTSStat 24

4.1 Architettura DARTS 24

4.2 Architettura DARTSStat 24

5 Testing 27

5.1 DARTS 27

5.1.1 Obiettivi del testing 27
5.1.2 Approccio 27
5.1.3 Componenti di testing 28
5.1.4 Test Case Plan 28
5.1.4.1 D_ST_SE 28
5.1.4.1.1 Densità Smell Eager Test (Density ET) 28
5.1.4.1.2 Densità Smell General fixture (Density GF) 30
5.1.4.1.3 Densità Smell Lack Of Cohesion (Density LOC) 31
5.1.4.1.4 GetAction Method 32
5.1.4.2 D_ST_ST 33
5.1.4.2.1 AddSession Method 33
5.1.4.3 D_UT_STB 34
5.1.4.3.1 Blast Method 34
5.1.4.4 D_UT_STS 35
5.1.4.4.1 DeleteJsonFile Method 35
5.1.4.4.2 FileExist Method 35
5.1.4.4.3 Serialize Method 36
5.1.4.5 D_UT_STU 37
5.1.4.5.1 MD5 Method 37
5.1.5 Integration Test Case 38
5.1.6 Test Case Specification 39
5.1.6.1 Eager Test Density 39
5.1.6.1.1 TC_0_0_DensityET 39
5.1.6.1.2 TC_0_1_DensityET 39
5.1.6.1.3 TC_0_2_DensityET 41
5.1.6.1.4 TC_0_3_DensityET 42
5.1.6.1.5 TC_0_4_DensityET 43
5.1.6.1.6 TC_0_5_DensityET 44
5.1.6.2 General Fixture Density 45
5.1.6.2.1 TC_1_0_DensityGF 45
5.1.6.2.2 TC_1_1_DensityGF 46
5.1.6.2.3 TC_1_2_DensityGF 47
5.1.6.2.4 TC_1_3_DensityGF 48

5.1.6.2.5 TC_1_4_DensityGF 49
5.1.6.2.6 TC_1_5_DensityGF 50
5.1.6.3 Lack Of Cohesion Density 51
5.1.6.3.1 TC_2_0_DensityLOC 51
5.1.6.3.2 TC_2_1_DensityLOC 52
5.1.6.3.3 TC_2_2_DensityLOC 53
5.1.6.3.4 TC_2_3_DensityLOC 54
5.1.6.3.5 TC_2_4_DensityLOC 55
5.1.6.3.6 TC_2_5_DensityLOC 56
5.1.6.4 GetActionMethod 57
5.1.6.4.1 TC_3_0_DensityLOC 57
5.1.6.4.2 TC_3_1_DensityLOC 58
5.1.6.5 D_ST_ST: AddSession Method 59
5.1.6.6 D_UT_STB: Blast Method 61
5.1.6.7 D_UT_STS: DeleteJsonFile Method 63
5.1.6.8 D_UT_STS: FileExist Method 65
5.1.6.9 D_UT_STS: Serialize Method 67
5.1.6.10 D_UT_STU 70
5.1.6.11 MD5 Method 70
5.1.7 Test Execution Report 72
5.1.7.1.1 TER_0_0_DensityET 72
5.1.7.1.2 TER_0_1_DensityET 72
5.1.7.1.3 TER_0_2_DensityET 73
5.1.7.1.4 TER_0_3_DensityET 73
5.1.7.1.5 TER_0_4_DensityET 74
5.1.7.1.6 TER_0_5_DensityET 74
5.1.7.1.7 TER_1_0_DensityGF 75
5.1.7.1.8 TER_1_1_DensityGF 75
5.1.7.1.9 TER_1_2_DensityGF 76
5.1.7.1.10 TER_1_3_DensityGF 76
5.1.7.1.11 TER_1_4_DensityGF 77
5.1.7.1.12 TER_1_5_DensityGF 77
5.1.7.1.13 TER_2_0_DensityLOC 78
5.1.7.1.14 TER_2_1_DensityLOC 78
5.1.7.1.15 TER_2_2_DensityLOC 79
5.1.7.1.16 TER_2_3_DensityLOC 79
5.1.7.1.17 TER_2_4_DensityLOC 80
5.1.7.1.18 TER_2_5_DensityLOC 80
5.1.7.1.19 TER_3_0_GetActionMethod 81
5.1.7.1.20 TER_3_1_GetActionMethod 81

5.1.7.2 D_ST_ST: AddSession Method 82
5.1.7.3 D_UT_STB: Blast Method 82
5.1.7.4 D_UT_STS: DeleteJsonFile Method 83
5.1.7.5 D_UT_STS: FileExist Method 84
5.1.7.6 D_UT_STS: Serialize Method 85
5.1.7.7 D_UT_STU: MD5 Method 87
5.2 DARTSStat 88
5.2.1 Obiettivi del testing 88
5.2.2 Approccio 88
5.2.3 Componenti di testing 89
5.2.4 Test Case Plan 89
5.2.4.1 DS_SS_AC/SE: Sottosistema Action/Session 90
5.2.4.1.1 Find All 90
5.2.4.1.2 Find One 91
5.2.4.1.3 Delete All 92
5.2.4.2 DS_SS_ST: Sottosistema Stat 92
5.2.4.2.1 Create 92
5.2.4.2.2 Find All 93
5.2.4.2.3 Find One 94
5.2.4.2.4 Delete All 95
5.2.4.3 DS_SS_GE: Sottosistema General 96
5.2.4.3.1 General 96
5.2.5 Test Case Specification 97
5.2.5.1 DS_SS_AC/SE: Sottosistema Action/Session 97
5.2.5.1.1 Find All 97
5.2.5.1.2 Find One 98
5.2.5.1.3 Delete All 100
5.2.5.2 DS_SS_ST: Sottosistema Stat 101
5.2.5.2.1 Create 101
5.2.5.2.2 Find All 102
5.2.5.2.3 Find One 103
5.2.5.2.4 Delete All 104
5.2.5.3 DS_SS_GE: Sottosistema General 105
5.2.5.3.1 General 105
5.2.6 Test Execution Report 106
6 Test di Regressione 106
6.1 Eager Test 107
6.2 General Fixture 107

6.3 Lack Of Cohesion of methods 107

7 Sviluppi Futuri..... 107

7.1 Nuovo Metodo per il calcolo delle classi di test e di produzione in DARTS..... 107

7.2 Nuovo Metodo per l'assegnazione di un identificativo dell'utente e del progetto al Server 108

7.3 Diff check degli artefatti rifattorizzati 108

7.4 Separazione della logica applicativa da quella di plugin con sotto progetti Gradle 108

1 Scopo del documento

In questo documento saranno descritti gli obiettivi del processo di estensione del plugin **DARTS: Detection And Refactoring of Test Smell**.

Si studierà come le modifiche identificate impattano sugli artefatti del sistema esistente, per ognuna di esse è stato condotto uno studio di fattibilità apposito.

Infine, tale documento include la pianificazione del testing delle componenti che derivano dalle CR, con la descrizione di come viene effettuato Regression Testing.

2 Panoramica del sistema attuale

DARTS (Detection And Refactoring of Test Smells è un plugin IntelliJ che permette la rilevazione, attraverso le tecniche presenti nello stato dell'arte, di tre tipi particolari di test smell: General Fixture, Eager Test e Lack of Cohesion of Methods e permette di utilizzare le operazioni di refactoring provviste dalle API integrate di IntelliJ.

Durante l'evoluzione di questo tool, e dato il continuo evolversi dell'IDE Intellij, è nata la necessità di portare i plugin a una versione moderna, che permettesse di rispettare i nuovi standard per i plugin di Intellij, utilizzando quindi Gradle. Inoltre, si vogliono raccogliere informazioni statistiche sull'utilizzo del tool, i quali permettono di fornire dati quantitativi utili per l'evoluzione del tool.

2.1 Requisiti Funzionali

Le funzionalità del sistema, oggetto di questa sezione, sono le seguenti:

- **FR 1**: Il plugin deve permettere all'utente di eseguire l'analisi sugli Smells automaticamente al momento del commit.
- **FR 2**: Il plugin deve permettere all'utente di eseguire l'analisi sugli Smells in qualsiasi momento tramite una Action dedicata.
- **FR 3**: Il plugin deve permettere all'utente di scegliere il tipo di analisi da eseguire: testuale o strutturale.
- **FR 4**: Il plugin deve permettere di eseguire le operazioni di refactoring automatico sugli Smells trovati.
- **FR 5**: Il plugin deve permettere all'utente di annullare le modifiche apportate tramite refactoring.

2.2 Attori

Il sistema, sino al momento dello sviluppo, prevede un unico attore, lo sviluppatore, il quale mediante l'interfaccia messa a disposizione può rilevare i vari test smell ed effettuare le operazioni di refactoring.

2.3 Design

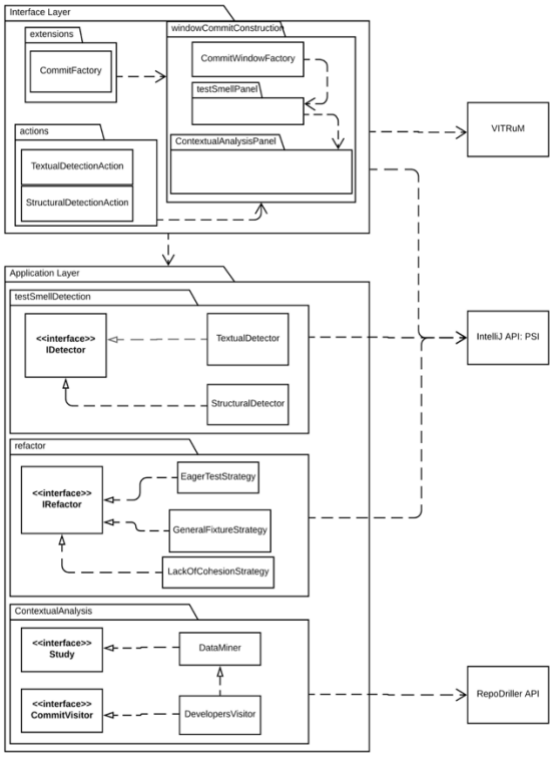
Il sistema è stato progettato in maniera tale da essere modulare; infatti, esso si compone di due layer differenti:

- **Interface Layer**: questo livello contiene tutta la logica necessaria per mostrare le informazioni all'utente e dare la possibilità di interagire col sistema;
- **Application Layer**: questo livello contiene tutta la logica del plugin, ovvero la possibilità di identificare gli smell ed effettuare il refactoring.

Il sistema software è stato sviluppato utilizzando il linguaggio di programmazione Object Oriented Java, coniugato alle librerie JetBrains per l'implementazione della logica di refactoring.

2.4 Implementazione

L'implementazione del software è stata effettuata in riferimento al modello descritto in precedenza, in particolare associando ad ogni layer un insieme di classi che sviluppano l'insieme delle funzionalità descritte in fase di analisi dei requisiti. In seguito, è osservabile l'architettura del software proposto:



Come si può notare dal diagramma, le componenti destinate alla logica di presentazione, e quindi all' *Interface Layer,* e le componenti destinati alla logica core, e quindi all' *Application Layer,* sono due componenti distinte e separate.

I due layer quindi si compongono delle seguenti componenti:

2.4.1 Interface Layer

1. **extensions**: Contiene la classe *CommitFactory* che interfaccia il plugin all'handler di IntelliJ per effettuare il commit.
2. **actions:** Contiene le Classi e i metodi che saranno richiamati all'avvio del plugin, nel momento in cui vengono premuti i pulsanti all'interno del menu contestuale di IntelliJ.
3. **windowCommitConstruction:** contiene le classi e i metodi relativi all'interfaccia necessaria per la visualizzazione dei risultati della detection.

2.4.2 Application Layer

- **Test Smell Detection:** Contiene la logica relativa alle regole per identificare i *test smell*. Ovviamente le *rule* fanno riferimento ai tipi di smell che il plug-in va a considerare.
- **Refactor:** Questo modulo fa riferimento alle operazioni di refactoring implementate dal tool. Ovviamente, le operazioni di refactoring possono essere "chiamate" solo se la fase di Detection è terminata. Quindi si pone l'obiettivo di effettuare una manipolazione (in termini di struttura) del codice scritto in modo da eliminare lo smell rilevato in precedenza.
- **Contextual Analysis:** Questo modulo contiene la logica relativa all'estrazione dei dati e che fa riferimento ad una componente esterna: *RepoDriller* in modo tale da clonare la repository e raccogliere dati su di essa.

Il plugin inoltre fa utilizzo di diversi tool esterni, i quali sono:

- **ViTRuM:** Plugin per la visualizzazione e l'analisi di metriche relative ai test.
- **Intellij API, PSI:** Tool per la modellazione semantica e sintattica di codice.
- **Repodriller API:** Java Framework per il Mining Software Repositories (MSR)

A valle di quanto appena menzionato nella sezione relativa al Design, si può apprezzare come le componenti destinati alla logica di presentazione e le componenti destinati alla logica core siano separate.

3 Analisi delle modifiche richieste

Ci sono tre diverse tipologie di modifiche richieste:

- Conversione a Progetto Gradle
- Separazione Infrastruttura Plugin dalla logica Core
- Statistiche e Misurazioni
    - Invio statistiche a un server remoto [Bonus]

Ovviamente deve essere preservato il comportamento attuale del plugin.

Di seguito vengono studiate nel dettaglio le singole Change Request.

3.1 Conversione a Progetto Gradle

3.1.1 Descrizione

Quasi tutti i tool seguono la struttura di progetto IntelliJ Platform Plugin, una struttura e sistema di build proprio di IntelliJ adibito ai soli plugin. Questa struttura è adesso sconsigliata, e JetBrains consiglia di migrare ad una struttura Gradle, noto sistema di build per Java simile a Maven e già usato per le app native Android. Bisogna anzitutto convertire il progetto secondo Gradle.

3.1.2 Problematiche affrontate

1. **Issue 1:** Alcune librerie utilizzate non sono integrabili tramite Gradle.
    1.1. **Proposal 1.1:** Aggiungere queste librerie localmente.
    1.2. **Criterion 1.1:** Facilitare il recupero delle librerie necessarie al funzionamento del plugin.
    Criterion 1.2: Agevolare il mantenimento del plugin.
    1.3. **Argument 1.1:** Gradle prevede l'aggiunta di libreria locali, supponiamo quindi che sia la scelta più veloce e semplice.
    1.4. **Solution 1:** Le librerie esterne non integrabili tramite Gradle verranno aggiunte localmente all'interno del progetto e messe a disposizione all'interno della repository GitHub.
2. **Issue 2:** Non esiste una chiara documentazione per quanto riguarda la conversione di un progetto non Gradle ad uno Gradle.
    2.1. **Proposal 2.1**: Utilizzare esempi reperibili sul web
    Proposal 2.2: Utilizzare tool di scaffolding messi a disposizione da IntelliJ
    2.2. **Criterion 2.1:** Rendere la migrazione il più semplice possibile
    2.3. **Argument 2.1**: Dagli esempi è semplice individuare la struttura ma non sempre questi sono chiari o sono alterati dagli sviluppatori.
    Argument 2.2: IntelliJ mette a disposizione il tool per la generazione di un nuovo progetto conforme ai requisiti Gradle.

2.4. **Solution 2**: Genereremo la struttura del progetto tramite IntelliJ così da essere sicuri di seguire la struttura conforme ai requisiti da Gradle, nulla esclude la consultazione di esempi reperibili sul web.

3.1.3 Impact Analysis

Al momento della proposta, DARTS, vi era una struttura delle directory simile ma non conforme a quella consigliata dalla documentazione di JetBrains. Quindi, cambiando la struttura delle directory abbiamo dovuto aggiornare gli statement *import* e gli statement *package* all'interno dei file java che compongono il progetto.

L'impatto della modifica verrà valutato utilizzando questa scala:

- **Debole**: se saranno necessarie solo modifiche marginali;
- **Medio**: se saranno necessarie modifiche all'artefatto, non facendo cambiare però la sua struttura in maniera eccessiva;
- **Forte**: se saranno necessarie sostanziali modifiche dell'artefatto o se l'artefatto dovrà essere completamente sostituito.

Artefatto	Impatto	Descrizione
Tutte le classi java contenute all'interno del package *main*.	Debole	Occorrerà modificare gli statement import/package in modo tale da rispecchiare la nuova struttura delle directory.

3.1.4 Studio di fattibilità

Identificazione, descrizione e valutazione dei costi

Identificazione	Valutazione	Motivazione
Modifica della struttura del progetto secondo standard Gradle.	Bassa	La struttura originale è molto vicina a quella consigliata dalla documentazione.

Identificazione, descrizione e valutazione dei benefici

Identificazione	Valutazione	Motivazione
Gestione semplificata delle dipendenze.	Media	Usando il file *gradle.build* è possibile dichiarare le dipendenze del progetto che saranno automaticamente risolte da Gradle stesso.

3.1.5 Analisi Post-Implementazione

Le modifiche apportate per il completamento della Change Request sono uguali a quelle previste nell'Impact Analysis. La gestione delle dipendenze è stata gestita in modo che il maggior numero possibile di componenti esterne fosse gestito da Gradle. Tuttavia, per quanto riguarda le componenti *TestFactorsPlugin.jar* e *TestSmellDiffusion.jar*, le quali non sono disponibili sul repository web, è stato deciso di inserirle manualmente includendo la libreria all'interno del progetto.

3.2 Separazione Infrastruttura Plugin dalla logica Core

3.2.1 Descrizione

Il progetto Gradle deve avere (almeno) due moduli ben distinti: uno per la sola logica legata all'infrastruttura plugin IntelliJ e l'altro contenente il cuore del tool, che è INDIPENDENTE dal fatto di essere un plugin per un IDE.

3.2.2 Problematiche affrontate

1. **Issue 1:** Quali componenti si occupano della logica del plugin e quali si occupano della logica core?
   1.1. **Proposal 1.1:** Ispezione del codice.
   Proposal 1.2: Studio della documentazione presente.
   1.2. **Criterion 1.1:** È bene separare le due componenti per poter facilitare l'estrazione e il riuso delle componenti che implementano l'analisi e la individuazione degli smell.
   1.3. **Argument 1.1:** L'ispezione del codice non per forza segue alcuni formalismi che è possibile trovare all'interno della documentazione.
   Argument 1.2: La documentazione, dando una visione generale di tutto il codice sorgente, può facilitare il lavoro di comprensione.

1.4. **Solution 1:** Verranno utilizzati entrambi gli approcci, quindi ispezione del codice e studio della documentazione presente.

3.2.3 Impact Analysis

La versione attuale del plugin presenta già una netta separazione tra quelle che sono le componenti che rappresentano l'infrastruttura del plugin e le componenti che implementano la logica core del plugin: analisi e individuazione degli smell.

3.2.4 Studio di fattibilità

Identificazione, descrizione e valutazione dei costi

Identificazione	Valutazione	Motivazione
Separazione Infrastruttura Plugin dalla logica Core	Bassa	La struttura originale presenta già questa netta separazione.

Identificazione, descrizione e valutazione dei benefici

Identificazione	Valutazione	Motivazione
Portabilità e riuso della logica core.	Media	Così facendo il tool è facilmente riutilizzabile all'interno di altri plugin o altri progetti.

3.2.5 Analisi Post-Implementazione

Nulla da specificare.

3.3 Statistiche e Misurazioni

3.3.1 Descrizione

Aggiungere un sottosistema che permetta il calcolo di alcune statistiche di utilizzo del tool (numero click, numero smell identificati, tempo esecuzione, smell scelti dall'utente ecc.). Bisogna mostrare con schermate dedicate un resoconto delle misure e che sia anche esportabile.

3.3.2 Problematiche affrontate

1. **Issue 1:** Quali statistiche bisogna individuare e quali misurazioni bisogna effettuare?
   1.1. **Proposal 1.1:** Tempo di esecuzione dell'analisi, testuale e strutturale.
   Proposal 1.2: Numero di esecuzioni del tool, testuali e strutturali.
   Proposal 1.3: Numero di smell individuati, suddivisi per tipo di smell.
   Proposal 1.4: Numero di classi o metodi affetti da smell.
   Proposal 1.5: Densità degli smell all'interno del progetto.
   Proposal 1.6: Azioni effettuate all'interno del tool.
   1.2. **Criterion 1.1:** Si vogliono effettuare misurazioni e raccogliere statistiche che siano quanto più utili agli sviluppatori del plugin originale, in modo da avere dati utili allo sviluppo e all'evoluzione di quest'ultimo.
   1.3. **Argument 1.1:** Stat utile all'identificazione di eventuali colli di bottiglia nella rilevazione degli smell, oppure problemi algoritmici che rallentano l'analisi.
   Argument 1.2: Stat utile nel comprendere quale delle due strategie di analisi viene più usata, potrebbe essere un punto di partenza per comprendere quale strategia far evolvere o ottimizzare e abbandonare l'altra.
   Argument 1.3: Stat non prettamente utile all'evoluzione del plugin ma di facile ottenimento.
   Argument 1.4: Come 1.3.
   Argument 1.5: Come 1.3
   Argument 1.6: Stat molto utile nel comprendere il grado di fiducia dell'utente finale nel plugin.
   1.4. **Solution 1:** Si è deciso di individuare, calcolare e raccogliere tutte le statistiche proposte. Dove per densità degli smell intendiamo l'applicazione delle seguenti tre formule:

$$GFD = \frac{SD}{TC}; LOCD = \frac{SD}{TC}; ETD = \frac{SD}{TM};$$ [1]

2. **Issue 2**: Quali parti del codice sorgente bisogna modificare per raggiungere l'obiettivo?
   2.1. **Proposal 2.1**: Consultazione della documentazione di DARTS.
   Proposal 2.2: Consultazione della documentazione dei plugin IntelliJ.
   Proposal 2.3: Effettuare reverse engineering.
   2.2. **Criterion 2.1**: Bisogna individuare tutte le componenti della logica core del plugin da modificare per andare ad inserire i metodi necessari alla raccolta e la misurazione dei dati.
   2.3. **Argument 2.1**: La documentazione di DART non descrive nel dettaglio il funzionamento del plugin, ma contiene lo schema architetturale del codice.
   Argument 2.2: La documentazione di IntelliJ non è molto utile, specialmente vista la scarsa

[1] Dove GFD = General Fixture Density, SD = Smell Detected, TC = Total Classes, TM = Total Methods, LOCD = Lack Of Cohesion Density, ETD = Eager Test Density.

documentazione. Utile solo a comprendere com'è strutturato un qualsiasi plugin, quindi andare ad individuare la componente da cui far partire lo studio.

Argument 2.3: La reverse engineering ci consente di effettuare information retrival a partire dal codice sorgente.

2.4. **Solution 2**: L'issue è stata risolta applicando tutte e tre le proposal.

3. **Issue 3**: Come raccogliere le statistiche?
   3.1. **Proposal 3.1**: Classe singleton.
   3.2. **Criterion 3.1**: Bisogna raccogliere tutte le statistiche nel modo più efficiente ed efficace possibile senza andare ad intaccare il normale funzionamento del plugin, sia nel comportamento che nella velocità di esecuzione.
   3.3. **Argument 3.1**: Una classe singleton, richiamata quando necessario, potrebbe raccogliere tutte le statistiche al proprio interno evitando problemi di inconsistenza e ridondanza dei dati.
   3.4. **Solution 3**: Risolta con la proposal 3.1.
4. **Issue 4**: Il conteggio del numero di classi di Test per il calcolo delle densità è definito dal nome delle classi. Le classi che vengono contate come TestClasses sono le classi che contengono la parola 'Test' all'interno del nome della classe.
   4.1. **Solution 4**: Dato che non sono state trovate soluzioni alternative migliori per poter differenziare una classe di test da una classe di produzione attraverso IntellijPSI, è stato deciso di mantenere la convenzione già definita su DARTS che solo le classi di Test possono contenere la stringa 'Test' all'interno del nome della classe.

3.3.3 Impact Analysis

Dopo aver letto e compreso la documentazione relativa al software "DARTS", tenendo conto della CR che prevede l'aggiunta di moduli atti al calcolo delle metriche descritte nella sezione precedente, è stato necessario effettuare l'analisi dell'impatto di questi ultimi per effettuare poi, l'aggiornamento dell'architettura.

Partendo dalla specifica della CR, dalla documentazione e dal codice sorgente, abbiamo identificato lo SIS *(Starting Impact Set)*, ovvero l'insieme di oggetti o componenti iniziali che molto probabilmente verranno impattati dalla CR:

- ***main.java.action.StructuralDetectionAction:***
  main.java.action.TextualDetectionAction: Alcune statistiche, vedi sezione precedente, devono essere calcolate dal momento in cui parte il plugin fino all'ottenimento dei risultati e quindi all'apertura della finestra principale del plugin.

- ***main.java.action.EagerTestStrategy***:
  main.java.refactor.GeneralFixtureStrategy:
  main.java.refactor.LackOfCohesionStrategy: classi che contengono la logica applicativa atta alla risoluzione degli smell. A valle di ciò, si ha il bisogno di calcolare le statistiche sugli smell.

Vista la specifica della CR che richiede un riscontro delle statistiche direttamente su UI del tool, sarà necessario costruire un Frame dedicato alle statistiche che sia collegato con il precedente Frame dei risultati. Quindi, sarà necessario costruire uno "StatsPanel" (mantenendo invariata l'Interface Layer) che sarà integrato (architetturalmente) nello strato relativo alla Interface Layer che avrà lo scopo di fornire un riepilogo delle statistiche *(main.java.windowCommitConstruction.statPanel.StatsPanel)* che quindi impatterà con la classe che ha lo scopo di mostrare l'analisi del progetto stesso:

- ***main.java.commitWindowConstruction.commitWindowFactory***: Classe che al proprio interno consente di aggiungere i JPanel relativi ad ogni Smell e, infine dovrà prevedere anche un Panel per le Statistiche che a sua volta utilizzerà il package ***main.java.stats*** stats per calcolare le statistiche.

Essendo il software di natura molto semplice e quindi essendo il codice di facile comprensione, non sono stati impattati altre componenti al di fuori di quelle previste; quindi, avremo SIS = CIS = AIS e DIS = FPIS = ∅.

L'impatto della modifica verrà valutato utilizzando questa scala:

- **Debole**: se saranno necessarie solo modifiche marginali;
- **Medio**: se saranno necessarie modifiche all'artefatto, non facendo cambiare però la sua struttura in maniera eccessiva;
- **Forte**: se saranno necessarie sostanziali modifiche dell'artefatto o se l'artefatto dovrà essere completamente sostituito.

Artefatto	Impatto	Descrizione
main *.java* *.action* *.StructuralDetectionAction*	BASSO	Sarà necessario introdurre la logica che consente di calcolare le statistiche per il conteggio del numero di esecuzione del tool nella modalità Strutturale e quindi la relativa creazione di una sessione per l'utente che sta eseguendo il tool.
Main *.java*	BASSO	Sarà necessario introdurre la logica che consente di calcolare le statistiche per il

.action *.TextualDetectionAction*		conteggio del numero di esecuzioni del tool in modalità Testuale e quindi la relativa creazione di una sessione per l'utente che sta eseguendo il tool.
Main *.java* *.refactor* *.EagerTestStrategy*	MEDIO	Dopo aver effettuato il refactoring, bisogna fare in modo che tutte le classi risultate positive all'Eager Test del progetto in esame, siano ottenute per poter racchiuderle nelle statistiche.
Main *.java* *.refactor* *.GeneralFixtureStrategy*	MEDIO	Dopo aver effettuato il refactoring, bisogna fare in modo che tutte le classi risultate positive all'Eager Test del progetto in esame, siano ottenute per poter racchiuderle nelle statistiche.
Main *.java* *.refactor* *.LackOfCohesionStrategy*	MEDIO	Dopo aver effettuato il refactoring, bisogna fare in modo che tutte le classi risultate positive al Lack Of Cohesion del progetto in esame, siano ottenute per poter racchiuderle nelle statistiche.
Main *.java* *.windowCommitConstruction* *.CommitWindowFactory*	BASSO	Le statistiche necessitano della loro parte grafica, di conseguenza deve essere aggiunto un modulo aggiuntivo che preveda l'interfaccia grafica, senza stravolgere tutto l'applicativo.

3.3.4 Studio di fattibilità

Identificazione, descrizione e valutazione dei costi

Identificazione	Valutazione	Motivazione
Modifica di alcune classi in modo tale da effettuare il calcolo delle classi totali e dei metodi totali	BASSA	Preferito il riuso del codice esistente, rispetto all'introduzione di nuove classi.

Identificazione, descrizione e valutazione dei benefici

Identificazione	Valutazione	Motivazione
Studio e analisi del progetto	ALTA	Consente allo sviluppatore di poter analizzare le metriche relative al progetto, e avere una visione totale di quanti smell impattano sulla qualità del codice.

3.3.5 Analisi Post-Implementazione

Per il completamento della CR3, è stato necessario quindi realizzare il sotto modulo *stats* che comprende le seguenti classi:

- *Action:* Rappresenta l'azione di refactoring effettuata su un test smell durante l'esecuzione di una sessione di analisi.
    - In particolare, l'azione viene definita completata quando viene utilizzato il Plugin di IntelliJ per effettuare l'operazione di refactoring sul codice.
    - L'azione invece viene definita cancellata quando invece, visualizzando l'operazione di refactoring attraverso il plugin, si annulla l'operazione e non viene effettuata.
- *Session:* Rappresenta l'insieme di statistiche che il plugin colleziona in una singola esecuzione di analisi testuale o strutturale.
- *Stats:* Rappresenta l'insieme di ogni sessione di esecuzione di analisi testuali e strutturali.

Successivamente, come già previsto prima dell'implementazione della modifica, è stato necessario realizzare il pannello che visualizza le statistiche: *StatsPanel*.

3.4 Invio statistiche a un server remoto [Bonus]

3.4.1 Descrizione

Bisogna trovare un meccanismo *lightweight* che permetta l'invio delle statistiche ad un server remoto. Abbiamo deciso di utilizzare Heroku, il quale offre un servizio gratuito di Hosting con CI da repository GitHub.

L'invio dei dati al server remoto è stato pensato in modo tale da inviare i dati al server dopo che l'utente avesse preso visione dei risultati dell'analisi del plug-in DARTS. Dunque, l'invio al server viene effettuato tramite richiesta POST con un oggetto *JSON (JavaScript Object Notation)* alla chiusura del plugin. I dati sul server saranno poi rappresentati attraverso un'interfaccia web semplice che mostri grafici colorati e user friendly.

3.4.2 Problematiche affrontate

1. **Issue 1:** Ogni quanto inviare i dati al server remoto?
   1.1. **Proposal 1.1:** Inviare i dati ad ogni statistica raccolta.
   Proposal 1.2: Inviare i dati solo al termine dell'esecuzione del plugin.
   1.2. **Criterion 1.1:** L'esecuzione del plugin deve essere quanto più "fluida" possibile, senza interruzioni.
   1.3. **Argument 1.1:** In caso di catastrofe, le statistiche raccolte saranno già state salvate, senza perdita di informazioni, ma il plugin potrebbe subire interruzioni causate dalla velocità di rete.
   Argument 1.2: I dati verranno inviati solo alla fine dell'esecuzione rendendo il plugin quanto più fruibile senza interruzioni, bisogna però trovare un metodo per il recovery delle statistiche precedenti in caso di imprevisti (Issue 3).
   1.4. **Solution 1:** Si preferisce non intaccare con la fruibilità del plugin; quindi, si procede con la Proposal 1.2.
2. **Issue 2:** Come generare il corpo della richiesta da inviare al server?
   2.1. **Proposal 2.1:** Creare una codifica delle statistiche raccolte sotto forma di stringa delimitata da caratteri separatori.
   Proposal 2.2: Utilizzare librerie esterne per la generazione di stringa formato JSON partendo dalla classe Stat. Nello specifico abbiamo trovato una libreria sviluppata da Google chiamata gson.
   2.2. **Criterion 2.1:** Bisogna semplificare quanto più possibile i dati inviati al server per facilitare le interpolazioni e il salvataggio delle statistiche raccolte.
   2.3. **Argument 2.2:** La generazione di un file JSON faciliterebbe la comunicazione tra client e server, conformandosi agli attuali standard di comunicazione client/server.
   2.4. **Solution 2:** Visto il Criterion 2.1, abbiamo deciso di adottare il formato JSON con l'ausilio della libreria gson.
3. **Issue 3**: Cosa accade nel momento in cui è assente la connessione ad Internet?
   3.1. **Proposal 3.1:** Lasciar perdere i dati non inviati al server.
   Proposal 3.2: Creare un file temporaneo contenente le statistiche non inviate al server e inviarle alla prossima esecuzione.
   3.2. **Criterion 3.1:** È importante cercare di inviare quante più statistiche al server senza trascurare alcun dato.
   3.3. **Argument 3.2:** Viene creato un file nella cartella del progetto *("nomeProgetto/stats.json")* e i dati relativi alle statistiche vengono salvate in locale. Non appena la connessione viene ripristinata, si effettua l'invio dei dati al server remoto.

3.4. **Solution 1:** In questo caso abbiamo adottato una specifica soluzione implementando quanto proposto in Proposal 3.2.

3.4.3 Impact Analysis

Per questa Change Request non sono presenti sostanziali cambiamenti al software già funzionante, poiché come si evince dall'architettura finale, è una componente esterna che viene chiamata dopo la presa visione dei risultati dell'analisi attraverso la GUI del plugin e quindi dopo la chiusura del plugin. Quindi l'unica componente che viene impattata fa riferimento alla GUI del plugin:

- ***main.java.windowCommitConstruction.CommitPrincipalFrame***

Artefatto	Impatto	Descrizione
main.java.windowCommitConstruction.CommitPrincipalFrame	BASSO	Sarà necessario prevedere degli action listener che, nel momento in cui l'utente decide di chiudere la GUI del plugin, effettua l'invio al server nel caso in cui la connessione sia presente, altrimenti i dati vengono salvati in un file locale del progetto e inviati non appena si ripristina la connessione.

3.4.4 Studio di fattibilità

Identificazione, descrizione e valutazione dei costi

Identificazione	Valutazione	Motivazione
Realizzazione del Server Remoto.	ALTA	Richiede la realizzazione di un'applicazione capace di collezionare e visualizzare i dati ricevuti.
Invio Statistiche in remoto	BASSA	Richiede l'invio delle statistiche già collezionate.

Identificazione, descrizione e valutazione dei benefici

Identificazione	Valutazione	Motivazione
Studio e Analisi di utilizzo del Tool	ALTA	La presente modifica consentirà di poter collezionare i dati di utilizzo del tool ed effettuare operazioni di analisi che aiuteranno all'evoluzione e al miglioramento del tool.

3.4.5 Analisi Post-Implementazione

Per completare la CR, è stato quindi necessario realizzare un nuovo server che potesse permettere la collezione e la visualizzazione dei dati.

Quindi è stata avviata la progettazione della infrastruttura esterna (descritta nella sezione *Architettura DartsStats*).

In seguito alla definizione della infrastruttura esterna si fa riferimento alla sezione successiva (*Architettura*) in cui vengono mostrate le nuove componenti derivanti dalle CR descritte sopra. In particolare, notiamo per la presente CR l'aggiunta della componente esterna *DartsStats* che si interfaccia con le componenti definite nel package *utility*.

Una volta che il server è disponibile per la collezione dei dati, abbiamo previsto l'aggiunta di due nuove classi:

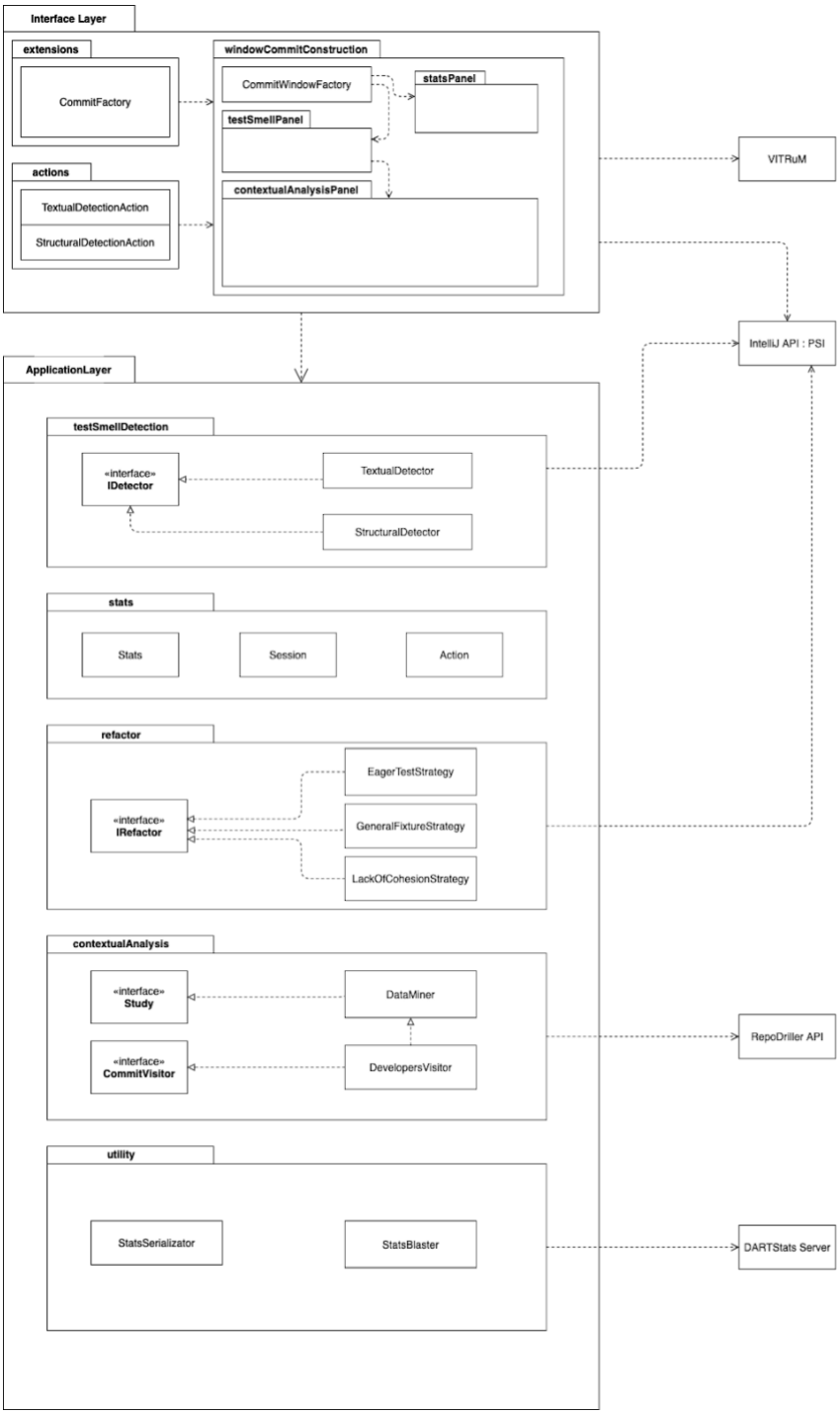
- ***utility.StatsBlaster.java:*** procede con l'invio delle informazioni al server.
- ***utility.StatsSerializator.java:*** procede con la serializzazione e il salvataggio delle statistiche.

L'utilizzo di queste classi impatterà, come previsto, sulla classe *CommitPrincipalFrame* che dovrà richiamarle alla chiusura del frame.

In definitiva quindi, avremo SIS = CIS = AIS e DIS = FPIS = ∅.

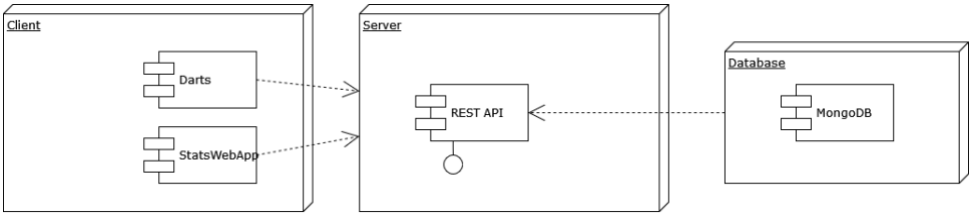
4 Architettura DARTS e DARTSStat

4.1 Architettura DARTS



4.2 Architettura DARTSStat

L'architettura progettata utilizza uno stile architetturale *REST,* la quale permette di poter effettuare le operazioni tramite l'utilizzo delle *REST API.*



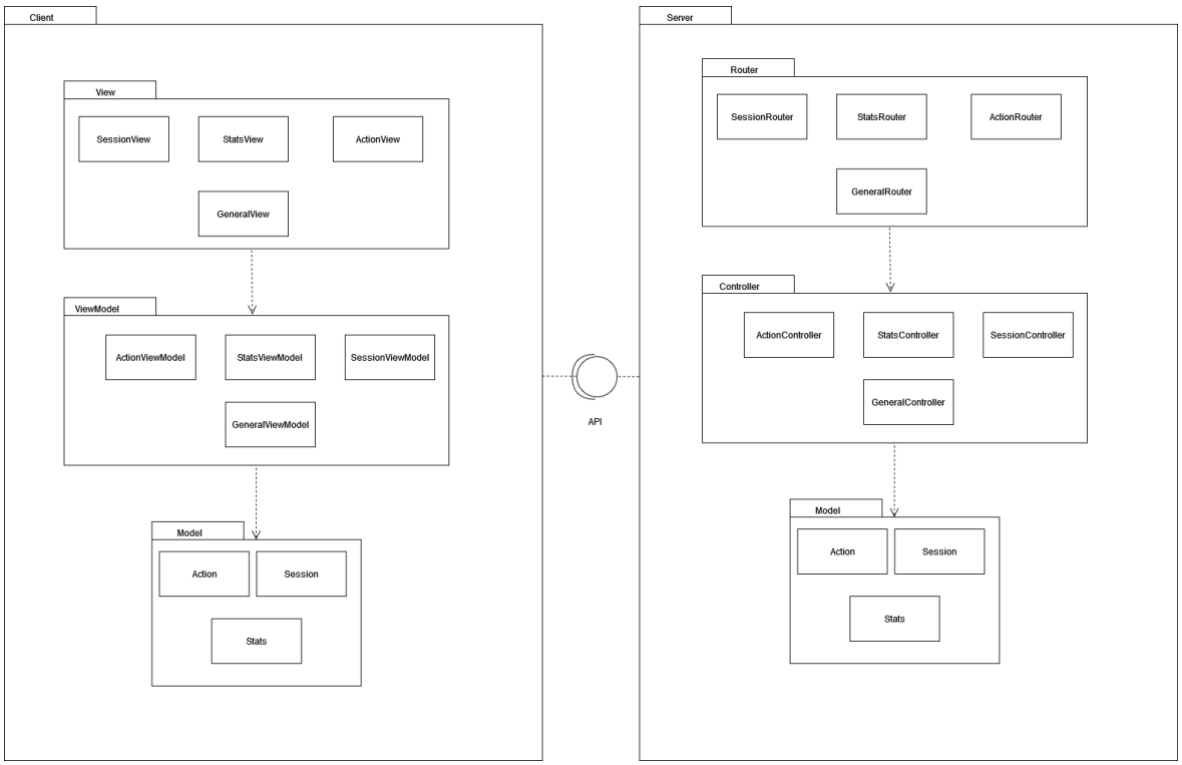
A seguire dall'architettura abbiamo quindi le seguenti componenti per il server:

- **Router**: Permette il collegamento via API delle operazioni di business accessibili al server
- **Controller**: Permette di gestire la logica di business del server e di effettuare le operazioni sulle statistiche da salvare e renderle persistenti
- **Model:** Permettono di definire il modello degli oggetti e delle entità del sistema

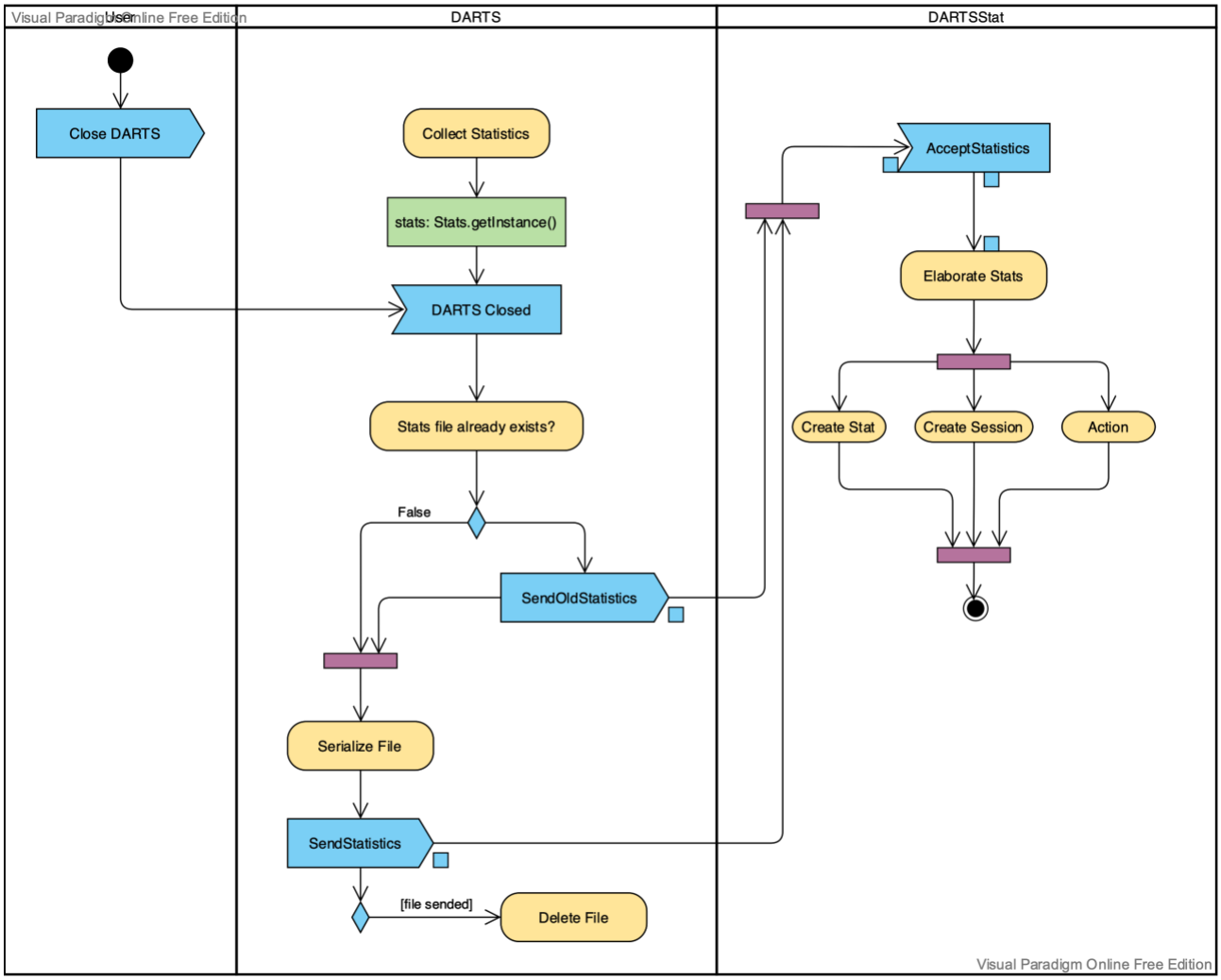
In seguito per la progettazione dell'applicazione client che definisce la visualizzazione delle statistiche, è stato utilizzato il pattern architetturale *MVVM (Model-view-ViewModel)*, ovvero una variante del Presentation-Model Design, la quale permette di astrarre la presentazione del modello su diverse architetture.

Quindi i componenti principali dell'architettura sono:

- **View**: Permette di definire il layout e la struttura della visualizzazione delle componenti all'utente.
- **Model**: Permettono di definire il modello degli oggetti e delle entità del sistema
- **ModelView**: Permette di definire la presentazione e la visualizzazione dei model all'utente.



Di seguito l'Activity Diagram del funzionamento di DARTS con DARTSStat.



5 Testing

Abbiamo progettato delle attività di testing per verificare la presenza di errori e, in caso fossero presenti, risolverli. In questo capitolo vengono descritte le modalità di testing, le funzionalità da testare e gli strumenti per condurre il testing.

5.1 DARTS

5.1.1 Obiettivi del testing

Come anticipato, lo scopo del testing è quello di individuare la presenza di faults all'interno del sistema e condurre attività di debug. Il testing ha due "parametri" fondamentali, che sono gli input e l'oracolo, ovvero il risultato che vogliamo ottenere. A valle di ciò ricordiamo che, il testing ha successo nel caso in cui, dato un input al sistema, il risultato ottenuto è diverso dal valore che ci aspettiamo, ovvero dall'oracolo. Il sistema DARTS è un plugin per l'ambiente di sviluppo IntelliJ IDEA e in quanto tale questo vuol dire che in fase di identificazione dei test smell, sia nel refactoring, vengono utilizzati strumenti forniti dalla stessa piattaforma che sfruttano l'ambiente di esecuzione; ambiente che non può essere facilmente emulato in uno scenario di testing automatizzato. Questo però non ci impedisce del tutto di effettuare test di regressione. Infatti, visto la mancata possibilità di scrivere casi di test per il plugin, è stato concepito in precedenza un "test pilota", ovvero un test con delle istanze di test smell create a mano e che quindi potessero simulare l'oracolo. Lo scopo era far eseguire DARTS su questo progetto pilota e verificare che il plugin riuscisse a trovare ogni istanza di test smell inserita manualmente. Dopo il testing di regressione iniziale, che ci assicura che il plugin funzioni correttamente come descritto dalle documentazioni precedenti, si inizia a sviluppare le CR descritte. Dopo ogni task portato a termine, si passa al testing della CR e in caso di fault, si aggiusta la classe di produzione fino a quando il risultato del testing sia uguale all'oracolo. Dopo il testing della CR bisogna necessariamente eseguire di nuovo il testing di regressione, per avere la certezza che la nuova modifica effettuata non abbia intaccato il funzionamento di tutto il plugin.

5.1.2 Approccio

I sottosistemi di DARTS aggiuntivi implementati per la realizzazione delle change request sono:

- **stats**: per la definizione delle statistiche
- **utility**: per l'invio e il salvataggio delle statistiche

5.1.3 Componenti di testing

Le componenti di test realizzate si trovano nel modulo test del sistema DARTS e sono le seguenti:

- Sottosistema **stats**:
  - **StatsTest**: componente di test relativa alla componente di produzione *Stats*
  - **SessionTest**: componente di test relativa alla componente di produzione *Session*
- Sottosistema **utility**:
  - **StatsBlasterTest***:* componente di test relativa alla componente di produzione *StatsBlaster*
  - **StatsSerializatorTest***:* componente di test relativa alla componente di produzione ***StatsSerializator***
  - **StatsUtilityTest***:* componente di test relativa alla componente di produzione *StatsUtility*

5.1.4 Test Case Plan

Avendo le tre componenti delle interfacce identiche, verrà effettuato un solo category partition. La nomenclatura segue la seguente struttura: **D_SS_XX**

- **D**: Iniziali del sistema DARTS
- **SS**: Sub System (**ST: stats; UT: utility)**
- **XX**:
  - **SE**: Session
  - **ST**: Stats
  - **STB**: StatsBlaster
  - **STS**: StatsSerializator
  - **STU**: StatsUtility

5.1.4.1 D_ST_SE

5.1.4.1.1 Densità Smell Eager Test (Density ET)

- Parametri:
  - NTM (Number of Total Method):
    - 0 [property No Instance]
    - 1 [property Single Instance]
    - 2 [property More Instance]
  - NOET (Number Of Eager Test):
    - 0 [property No Instance]

- 1 [property Single instance]
- 2 [property More Instance]

Parametri:	
Parametro: Numero totali di metodi del test	
Valore [NTM]	1. Non ci sono metodi nella classe di test, valore uguale a 0 [NTM_NI]. 2. C'è al più un metodo nella classe di test, valore uguale a 1 [NTM_SI]. 3. Ci sono più metodi nella classe di test, valore uguale a più di 1 [NTM_MI].
Numero totale di istanze di Eager Test	
Valore [NOET]	1. Non ci sono istanze di Eager Test, valore uguale a 0 [NOET_NI]. 2. C'è un'istanza di Eager Test, valore uguale a 1 [NOET_SI]. 3. Ci sono istanze di Eager Test, valore uguale a più di 1 [NOET_MI].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_0_0	NOET_NI, NTM_NI	Corretto: il metodo fornisce -1.
TC_0_1	NOET_NI, NTM_SI	Corretto: il metodo fornisce 0.
TC_0_2	NOET_SI, NTM_SI	Corretto: il metodo restituisce NOET / NTM
TC_0_3	NOET_MI, NTM_SI	Corretto: il metodo restituisce NOET / NTM

TC_0_4	NOET_SI, NTM_MI	Corretto: il metodo restituisce NOET / NTM
TC_0_5	NOET_MI, NTM_MI	Corretto: il metodo restituisce NOET / NTM

5.1.4.1.2 Densità Smell General fixture (Density GF)

- Parametri:
  - NTC (Number Of Total Classes):
    - 0 [property No Instance]
    - 1 [property Single Instance]
    - 2 [property More Instance]
  - NOGF (Number Of General Fixture):
    - 0 [property No Instance]
    - 1 [property Single instance]
    - 2 [property More Instance]

Parametri:	
Parametro: Numero totali di classi	
Valore [NTC]	1. Non ci sono classi, valore uguale a 0 [NTC_NI]. 2. C'è al più una classe, valore uguale a 1 [NTC_SI]. 3. Ci sono più classi, valore uguale a più di 1 [NTC_MI].
Numero totale di istanze di General Fixture	
Valore [NOGF]	1. Non ci sono istanze di General Fixture, valore uguale a 0 [NOGF_NI]. 2. C'è un'istanza di General Fixture, valore uguale a 1 [NOGF_SI].

	3. Ci sono istanze di General Fixture, valore uguale a più di 1 [NOGF_MI].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_1_0	NOGF_NI, NTC_NI	Corretto: il metodo fornisce -1.
TC_1_1	NOGF_NI, NTC_SI	Corretto: il metodo fornisce 0.
TC_1_2	NOGF_SI, NTC_SI	Corretto: il metodo restituisce NOGF / NTC
TC_1_3	NOGF_MI, NTC_SI	Corretto: il metodo restituisce NOGF / NTC
TC_1_4	NOGF_SI, NTC_MI	Corretto: il metodo restituisce NOGF / NTC
TC_1_5	NOGF_MI, NTC_MI	Corretto: il metodo restituisce NOGF / NTC

5.1.4.1.3 Densità Smell Lack Of Cohesion (Density LOC)

- Parametri:
  - NTC (Number of Total Classes):
    - 0 [property No Instance]
    - 1 [property Single Instance]
    - 2 [property More Instance]
  - NOLOC (Number of Lack of Cohesion:
    - 0 [property No Instance]
    - 1 [property Single instance]
    - 2 [property More Instance]

Parametri:

Parametro: Numero totali di classi	
Valore [NTC]	1. Non ci sono classi, valore uguale a 0 [NTC_NI]. 2. C'è al più una classe, valore uguale a 1[NTC_SI]. 3. Ci sono più classi, valore uguale a più di 1[NTC_MI].
Numero totale di istanze di Lack Of Cohesion	
Valore [NOLOC]	1. Non ci sono istanze di Lack Of Cohesion, valore uguale a 0 [NOLOC_NI]. 2. C'è un'istanza di Lack Of Cohesion, valore uguale a 1 [NOLOC_SI]. 3. Ci sono istanze di Lack Of Cohesion, valore uguale a più di 1 [NOLOC_MI].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_2_0	NOLOC_NI, NTC_NI	Corretto: il metodo fornisce -1.
TC_2_1	NOLOC_NI, NTC_SI	Corretto: il metodo fornisce 0.
TC_2_2	NOLOC_SI, NTC_SI	Corretto: il metodo restituisce NOLOC / NTC
TC_2_3	NOLOC_MI, NTC_SI	Corretto: il metodo restituisce NOLOC / NTC
TC_2_4	NOLOC_SI, NTC_MI	Corretto: il metodo restituisce NOLOC / NTC
TC_2_5	NOLOC_MI, NTC_MI	Corretto: il metodo restituisce NOLOC / NTC

5.1.4.1.4 GetAction Method

- Parametri:

- ArrayList size:
  - 0 [property No Instance]
  - 1 [property Single Instance]

Parametri:	
Parametro: ArrayList size	
Valore [ALS]	1. Non Ci sono elementi nell'arrayList, valore size 0. [ALS_NI] 2. ci sono elementi nell'arrayList, valore size almeno 1.[ALS_SI]
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_3_0	ALS_NI	Corretto: il metodo fornisce una size uguale a zero.
TC_3_1	ALS_SI	Corretto: il metodo fornisce una size maggiore di zero.

5.1.4.2 D_ST_ST

5.1.4.2.1 AddSession Method

- Parametri:
  - Session value:
    - Null [property Null Instance]
    - Not Null [property Not Null Instance]

Parametri:
Parametro: Session Value

Valore [SSV]	1. Si tenta di inserire una sessione con valore NULL. [SSV_NI] 2. Si tenta di inserire una sessione con valore diverso da NULL. [SSV_NNI].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_4_0	SSV_NI	Errato: il metodo inserisce una sessione di tipo NULL.
TC_4_1	SSV_NNI	Corretto: il metodo inserisce una sessione diversa da NULL

5.1.4.3 D_UT_STB

5.1.4.3.1 Blast Method

- Parametri:
  - Stats value
    - Null [property Null Instance]
    - Not Null [property Not Null Instance]

Parametri:	
Parametro: Stats value	
Valore [SV]	1. Si tenta di inserire una Stats con valore NULL. [SV_NI] 2. Si tenta di inserire una Stats con valore diverso da NULL. [SV_NNI].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_5_0	SV_NI	Corretto: Il metodo non invia al server una statistica di tipo NULL.
TC_5_1	SV_NNI	Corretto:Il metodo invia al server una statistica di tipo diverso da NULL.

5.1.4.4 D_UT_STS

5.1.4.4.1 DeleteJsonFile Method

- Parametri:
  - Path
    - Path Not Exist [property PATH_NE]
    - Path Exist [property PATH_E]

Parametri:	
Parametro: Path	
Valore [PATH]	1. Si tenta di cancellare un file avente un URL che non esiste. [PATH_NE] 2. Si tenta di cancellare un file avente URL esistente. [PATH_E].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_6_0	PATH_NE	Corretto: Il metodo non cancella il file.
TC_6_1	PATH_E	Corretto: Il metodo cancella correttamente il file.

5.1.4.4.2 FileExist Method

- Parametri:

- Path
  - Path Not Exists [property PATH_NE]
  - Path Exist [property PATH_E]

Parametri:	
Parametro: Path	
Valore [PATH]	1. Il metodo controlla se esiste un file che non esiste [PATH_NE] 2. Il metodo controlla se esiste un file che esiste.. [PATH_E].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_7_0	PATH_NE	Corretto: Il metodo non è in grado di trovare il file.
TC_7_1	PATH_E	Corretto: Il metodo riesce a trovare il file correttamente.

5.1.4.4.3 Serialize Method

- Parametri:
  - Stats
    - Stats Null [property SS_NL]
    - Stats Not Null [property SS_NN]
- Parametri:
  - Path
    - Path Not Exists [property PATH_NE]
    - Path Exist [property PATH_E]

Parametri:	
Parametro: Stats	
Valore [SS]	1. Il metodo prende una statistica NULL. [SS_NL] 2. Il metodo prende una statistica NOT NULL. [SS_NN]
Parametro: Path	
Valore[PATH]	1. Il metodo prende un path che non esiste. [PATH_NE] 2. Il metodo prende un path che esiste. [PATH_E]
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_8_0	SS_NN, PATH_E	Corretto: il metodo serializza correttamente.
TC_8_1	SS_NL, PATH_NE	Errato: il metodo non serializza il file
TC_8_2	SS_NL, PATH_E	Errato: il metodo non serializza il file
TC_8_3	SS_NN, PATH_NE	Errato: il metodo serializza

5.1.4.5 D_UT_STU

5.1.4.5.1 MD5 Method

- Parametri:
  - Type
    - Generic Type [Property T_GT]
    - Null [Property T_N]
    - Empty [Property T_E]

Parametri:	
Parametro: Type	
Valore [TT]	1. Il metodo prende un parametro di tipo generico ed effettua l'MD5. [T_GT] 2. Il metodo prende un parametro NULL. [T_N]. 3. Il metodo prende un parametro Empty [T_E].
Ambiente	
Nessuno	

Codice	Combinazione	Esito
TC_9_0	T_GT	Corretto: il metodo effettua l'Hash in maniera corretta.
TC_9_1	T_N	Errato: Il metodo effettua l'Hash in maniera errata
TC_9_2	T_E	Errato: Il metodo effettua l'Hash in maniera errata

5.1.5 Integration Test Case

Il testing di integrazione ha lo scopo di identificare errori e malfunzionamenti nel sistema, eseguendo vari test su tutte le funzionalità del software e controllando che l'esito atteso e quello risultato dal testing siano lo stesso. La strategia di testing di integrazione adottata è di tipo "Bottom-up", secondo la quale vengono prima collaudati i moduli a livello più basso della gerarchia, passando poi al livello superiore e testandolo sfruttando le funzionalità del livello precedente risalendo in questo modo fino all'intero sistema. Il Framework utilizzato per effettuare il test di integrazione è Mockito, che consente di creare un "mock" di una classe, dichiarandone il comportamento e facilitando alcuni tipi di test. Il diagramma che ci ha consentito di individuare i moduli da testare è presente all'interno del repository e prende il nome di *"testingUML"*.

I moduli da testare quindi sono:

- *main.java.stats.Session.java*

- *main.java.stats.Stats.java*

5.1.6 Test Case Specification

5.1.6.1 Eager Test Density

5.1.6.1.1 TC_0_0_DensityET

Test Case ID:	TC_0_0
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Eager Test	0
Number Of Test Method	0

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Eager Test Density: -1

5.1.6.1.2 TC_0_1_DensityET

Test Case ID:	TC_0_1
Pre-Condition:	

L'utente avvia l'analisi del plugin su un progetto da lui scelto

Flow of events:

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Eager Test	0
Number Of Test Method	1

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:

Eager Test Density: 0

5.1.6.1.3 TC_0_2_DensityET

Test Case ID:	TC_0_2
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Eager Test	1
Number Of Test Method	1

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Eager Test Density: Number Of Eager Test / Number Of Test Method

5.1.6.1.4 TC_0_3_DensityET

Test Case ID:	TC_0_3
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Eager Test	2
Number Of Test Method	1

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Eager Test Density: Number Of Eager Test / Number Of Test Method

5.1.6.1.5 TC_0_4_DensityET

<table>
<tr><td>Test Case ID:</td><td>TC_0_4</td></tr>
<tr><td colspan="2">Pre-Condition:</td></tr>
<tr><td colspan="2">L'utente avvia l'analisi del plugin su un progetto da lui scelto</td></tr>
<tr><td colspan="2">Flow of events:</td></tr>
<tr><td colspan="2">1. L'utente lancia l'esecuzione dell'analisi sul progetto
<table><tr><td>Input</td><td>Valore</td></tr><tr><td>Number Of Eager Test</td><td>1</td></tr><tr><td>Number Of Test Method</td><td>2</td></tr></table>2. L'Utente visualizza le statistiche sulla GUI del plugin</td></tr>
<tr><td colspan="2">Oracle:</td></tr>
<tr><td colspan="2">Eager Test Density: Number Of Eager Test / Number Of Test Method</td></tr>
</table>

5.1.6.1.6 TC_0_5_DensityET

Test Case ID:	TC_0_5
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Eager Test	2
Number Of Test Method	2

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Eager Test Density: Number Of Eager Test / Number Of Test Method

5.1.6.2 General Fixture Density

5.1.6.2.1 TC_1_0_DensityGF

Test Case ID:	TC_1_0
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of General Fixture	0
Number Of Test Class	0

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
General Fixture Density: -1

5.1.6.2.2 TC_1_1_DensityGF

Test Case ID:	TC_1_1
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	
1. L'utente lancia l'esecuzione dell'analisi sul progetto	

Input	Valore
Number Of General Fixture	0
Number Of Test Class	1

2. L'Utente visualizza le statistiche sulla GUI del plugin
Oracle:
General Fixture Density: 0

5.1.6.2.3 TC_1_2_DensityGF

Test Case ID:	TC_1_2
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of General Fixture	1
Number Of Test Class	1

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
General Fixture Density: Number Of General Fixture / Number Of Test Classes

5.1.6.2.4 TC_1_3_DensityGF

Test Case ID:	TC_1_3
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of General Fixture	2
Number Of Test Class	1

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
General Fixture Density: Number Of General Fixture / Number Of Test Classes

5.1.6.2.5 TC_1_4_DensityGF

Test Case ID:	TC_1_4
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of General Fixture	1
Number Of Test Class	2

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
General Fixture Density: Number Of General Fixture / Number Of Test Classes

5.1.6.2.6 TC_1_5_DensityGF

Test Case ID:	TC_1_5
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	
1. L'utente lancia l'esecuzione dell'analisi sul progetto	

Input	Valore
Number Of General Fixture	2
Number Of Test Class	2

2. L'Utente visualizza le statistiche sulla GUI del plugin
Oracle:
General Fixture Density: Number Of General Fixture / Number Of Test Classes

5.1.6.3 Lack Of Cohesion Density

5.1.6.3.1 TC_2_0_DensityLOC

Test Case ID:	TC_2_0
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Lack Of Cohesion	0
Number Of Test Class	0

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Lack Of Cohesion Density: -1

5.1.6.3.2 TC_2_1_DensityLOC

Test Case ID:	TC_2_1
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	
1. L'utente lancia l'esecuzione dell'analisi sul progetto	

Input	Valore
Number Of Lack Of Cohesion	0
Number Of Test Class	1

2. L'Utente visualizza le statistiche sulla GUI del plugin
Oracle:
Lack Of Cohesion Density: 0

5.1.6.3.3 TC_2_2_DensityLOC

Test Case ID:	TC_2_2
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	
1. L'utente lancia l'esecuzione dell'analisi sul progetto	

Input	Valore
Number Of Lack Of Cohesion	1
Number Of Test Class	1

2. L'Utente visualizza le statistiche sulla GUI del plugin
Oracle:
Lack Of Cohesion Density: Number Of Lack Of Cohesion / Number Of Test Classes

5.1.6.3.4 TC_2_3_DensityLOC

Test Case ID:	TC_2_3
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	
1. L'utente lancia l'esecuzione dell'analisi sul progetto	

Input	Valore
Number Of Lack Of Cohesion	2
Number Of Test Class	1

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Lack Of Cohesion Density: Number Of Lack Of Cohesion / Number Of Test Classes

5.1.6.3.5 TC_2_4_DensityLOC

<table>
<tr><td>Test Case ID:</td><td>TC_2_4</td></tr>
<tr><td colspan="2">Pre-Condition:</td></tr>
<tr><td colspan="2">L'utente avvia l'analisi del plugin su un progetto da lui scelto</td></tr>
<tr><td colspan="2">Flow of events:</td></tr>
<tr><td colspan="2">1. L'utente lancia l'esecuzione dell'analisi sul progetto
<table><tr><td>Input</td><td>Valore</td></tr><tr><td>Number Of Lack Of Cohesion</td><td>1</td></tr><tr><td>Number Of Test Class</td><td>2</td></tr></table>2. L'Utente visualizza le statistiche sulla GUI del plugin</td></tr>
<tr><td colspan="2">Oracle:</td></tr>
<tr><td colspan="2">Lack Of Cohesion Density: Number Of Lack Of Cohesion / Number Of Test Classes</td></tr>
</table>

5.1.6.3.6 TC_2_5_DensityLOC

Test Case ID:	TC_2_5
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto da lui scelto	
Flow of events:	

1. L'utente lancia l'esecuzione dell'analisi sul progetto

Input	Valore
Number Of Lack Of Cohesion	2
Number Of Test Class	2

2. L'Utente visualizza le statistiche sulla GUI del plugin

Oracle:
Lack Of Cohesion Density: Number Of Lack Of Cohesion / Number Of Test Classes

5.1.6.4 GetActionMethod

5.1.6.4.1 TC_3_0_DensityLOC

<table>
<tr><td>Test Case ID:</td><td>TC_3_0</td></tr>
<tr><td colspan="2">Pre-Condition:</td></tr>
<tr><td colspan="2">L'utente avvia l'analisi del plugin su un progetto da lui scelto</td></tr>
<tr><td colspan="2">Flow of events:</td></tr>
<tr><td colspan="2">1. L'utente interagisce con la GUI ed effettua delle azioni all'interno di essa, ad esempio selezionando la Refactoring Preview
<table>
<tr><td>Input</td><td>Valore</td></tr>
<tr><td>ArrayList size</td><td>0</td></tr>
</table>
2. L'Utente visualizza le statistiche sulla GUI del plugin</td></tr>
<tr><td colspan="2">Oracle:</td></tr>
<tr><td colspan="2">Number Of Action: 0</td></tr>
</table>

5.1.6.4.2 TC_3_1_DensityLOC

<table>
<tr><td>Test Case ID:</td><td>TC_3_1</td></tr>
<tr><td colspan="2">Pre-Condition:</td></tr>
<tr><td colspan="2">L'utente avvia l'analisi del plugin su un progetto da lui scelto</td></tr>
<tr><td colspan="2">Flow of events:</td></tr>
<tr><td colspan="2">1. L'utente interagisce con la GUI ed effettua delle azioni all'interno di essa, ad esempio selezionando la Refactoring Preview
<table>
<tr><td>Input</td><td>Valore</td></tr>
<tr><td>ArrayList size</td><td>1</td></tr>
</table>
2. L'Utente visualizza le statistiche sulla GUI del plugin</td></tr>
<tr><td colspan="2">Oracle:</td></tr>
<tr><td colspan="2">Number Of Action: 1</td></tr>
</table>

5.1.6.5 D_ST_ST: AddSession Method

Test Case ID:	TC_4_0
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto	
Flow of events:	
Avviata l'analisi il sistema prova a creare una session vuota Input: Session — Valore: Object == null 2. Il sistema prova a inserire la session nella lista	
Oracle:	
addSession restituisce null	

Test Case ID:	TC_4_1
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto	
Flow of events:	

Avviata l'analisi il sistema crea una sessione

Input	Valore
Session	Object != null

2. Il sistema inserisce la sessione nella lista delle sessioni

Oracle:
Session List contiene la session

5.1.6.6 D_UT_STB: Blast Method

Test Case ID:	TC_5_0
Pre-Condition:	
L'utente avvia l'analisi del plugin su un progetto	
Flow of events:	
L'utente chiude il plugin e il sistema prova a inviare Stats null	

Input	Valore
Stats	Object == null

2. Il sistema prova a inviare la stats

Oracle:
L'oggetto Stats non viene inviato al server

Test Case ID:	TC_5_1
Pre-Condition:	
L'utente chiude il plugin e il sistema invia un oggetto stats al server	
Flow of events:	
Avviata l'analisi il sistema crea una stats	

Input	Valore

Session	Object != null

2. Il sistema invia la stats al server

Oracle:
Il server contiene l'oggetto stats

5.1.6.7 D_UT_STS: DeleteJsonFile Method

Test Case ID:	TC_6_0
Pre-Condition:	
L'utente chiude il plugin e il sistema cancella il file json preesistente	
Flow of events:	
Il sistema cerca il file json attraverso il path	

Input	Valore
Path	Path non esistente

2. Il sistema prova a cancellare il file

Oracle:
Il metodo restituisce false

Test Case ID:	TC_6_1
Pre-Condition:	
L'utente chiude il plugin e il sistema cancella il file json preesistente	
Flow of events:	

Il sistema cerca il file json attraverso il path

Input	Valore
Path	Path esistente

2. Il sistema prova a cancellare il file

Oracle:
Il metodo restituisce true e il file non è presente nel sistema

5.1.6.8 D_UT_STS: FileExist Method

Test Case ID:	TC_7_0
Pre-Condition:	
L'utente chiude il plugin e il sistema cerca il file	
Flow of events:	
Il sistema cerca il file attraverso il path	

Input	Valore
Path	Path non esistente

2. Il sistema restituisce false

Oracle:
Il metodo restituisce false

Test Case ID:	TC_7_1
Pre-Condition:	
L'utente chiude il plugin e il sistema cerca il file	
Flow of events:	
Il sistema cerca il file json attraverso il path	

Input	Valore

Path	Path esistente

2. Il sistema restituisce true

Oracle:
Il metodo restituisce true

5.1.6.9 D_UT_STS: Serialize Method

Test Case ID:	TC_8_0
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	
Il sistema colleziona l'oggetto stats e cerca di salvare il file json	

Input	Valore
Stats	Oggetto != Null
Path	Path Esistente

2. Il sistema restituisce true

Oracle:
Il metodo restituisce true e il file json contiene l'oggetto stats

Test Case ID:	TC_8_1
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	

Il sistema colleziona l'oggetto stats e cerca il file json

Input	Valore
Stats	Oggetto Null
Path	Path Non Esistente

2. Il sistema restituisce false

Oracle:
Il metodo restituisce false

Test Case ID:	TC_8_2
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	

Il sistema colleziona l'oggetto stats e cerca il file json

Input	Valore
Stats	Oggetto Null
Path	Path Esistente

2. Il sistema restituisce false

Oracle:

Il metodo restituisce false

Test Case ID:	TC_8_3
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	

Il sistema colleziona l'oggetto stats e cerca il file json

Input	Valore
Stats	Oggetto !=Null
Path	Path Non Esistente

2. Il sistema restituisce false

Oracle:
Il metodo restituisce false

5.1.6.10 D_UT_STU

5.1.6.11 MD5 Method

Test Case ID:	TC_9_0
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	
Il sistema colleziona la stringa e calcola l'hash MD5 **Input** / **Valore** **Type** / Stringa non vuota 2. Il sistema restituisce true	
Oracle:	
Il sistema genera una stringa MD5 relativa all'oggetto	

Test Case ID:	TC_9_1
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	

Il sistema colleziona la stringa e calcola l'hash MD5

Input	Valore
Type	Stringa null

2. Il sistema restituisce null

Oracle:
Il sistema genera null

Test Case ID:	TC_9_2
Pre-Condition:	
L'utente chiude il plugin	
Flow of events:	

Il sistema colleziona la stringa e calcola l'hash MD5

Input	Valore
Type	Stringa vuota

2. Il sistema restituisce null

Oracle:
Il sistema genera null

5.1.7 Test Execution Report

5.1.7.1.1 TER_0_0_DensityET

ID Test Case	TC_0_0
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Eager Test Density c'è un risultato uguale a -1.
Output Atteso	-1
Output Sistema	-1
Anomalie	Nessuna anomalia

5.1.7.1.2 TER_0_1_DensityET

ID Test Case	TC_0_1
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Eager Test Density c'è un risultato uguale a 0.
Output Atteso	0
Output Sistema	0
Anomalie	Nessuna anomalia

5.1.7.1.3 TER_0_2_DensityET

ID Test Case	TC_0_2
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Eager Test Density c'è un risultato uguale a NOET / NTM.
Output Atteso	1
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.4 TER_0_3_DensityET

ID Test Case	TC_0_3
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Eager Test Density c'è un risultato uguale a NOET / NTM.
Output Atteso	2
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.5 TER_0_4_DensityET

ID Test Case	TC_0_4
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Eager Test Density c'è un risultato uguale a NOET / NTM.
Output Atteso	1
Output Sistema	2
Anomalie	Nessuna anomalia

5.1.7.1.6 TER_0_5_DensityET

ID Test Case	TC_0_5
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Eager Test Density c'è un risultato uguale a NOET / NTM.
Output Atteso	2
Output Sistema	2
Anomalie	Nessuna anomalia

5.1.7.1.7 TER_1_0_DensityGF

ID Test Case	TC_1_0
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce General Fixture Density c'è un risultato uguale a -1.
Output Atteso	0
Output Sistema	0
Anomalie	Nessuna anomalia

5.1.7.1.8 TER_1_1_DensityGF

ID Test Case	TC_1_1
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce General Fixture Density c'è un risultato uguale a 0.
Output Atteso	0
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.9 TER_1_2_DensityGF

ID Test Case	TC_1_2
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce General Fixture Density c'è un risultato uguale a NOGF / NTC.
Output Atteso	1
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.10 TER_1_3_DensityGF

ID Test Case	TC_1_3
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce General Fixture Density c'è un risultato uguale a NOGF / NTC.
Output Atteso	2
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.11 TER_1_4_DensityGF

ID Test Case	TC_1_4
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce General Fixture Density c'è un risultato uguale a NOGF / NTC.
Output Atteso	1
Output Sistema	2
Anomalie	Nessuna anomalia

5.1.7.1.12 TER_1_5_DensityGF

ID Test Case	TC_1_5
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce General Fixture Density c'è un risultato uguale a NOGF / NTC.
Output Atteso	2
Output Sistema	2
Anomalie	Nessuna anomalia

5.1.7.1.13 TER_2_0_DensityLOC

ID Test Case	TC_2_0
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Lack Of Cohesion Density c'è un risultato uguale a -1.
Output Atteso	0
Output Sistema	0
Anomalie	Nessuna anomalia

5.1.7.1.14 TER_2_1_DensityLOC

ID Test Case	TC_2_1
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Lack Of Cohesion Density c'è un risultato uguale a 0.
Output Atteso	0
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.15 TER_2_2_DensityLOC

ID Test Case	TC_2_2
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Lack Of Cohesion Density c'è un risultato uguale a NOLOC / NTC.
Output Atteso	1
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.16 TER_2_3_DensityLOC

ID Test Case	TC_2_3
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Lack Of Cohesion Density c'è un risultato uguale a NOLOC / NTC.
Output Atteso	2
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.1.17 TER_2_4_DensityLOC

ID Test Case	TC_2_4
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Lack Of Cohesion Density c'è un risultato uguale a NOLOC / NTC.
Output Atteso	1
Output Sistema	2
Anomalie	Nessuna anomalia

5.1.7.1.18 TER_2_5_DensityLOC

ID Test Case	TC_2_5
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce Lack Of Cohesion Density c'è un risultato uguale a NOLOC / NTC.
Output Atteso	2
Output Sistema	2
Anomalie	Nessuna anomalia

5.1.7.1.19 TER_3_0_GetActionMethod

ID Test Case	TC_3_0
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce azioni effettuate non viene specificato nulla
Output Atteso	0
Output Sistema	0
Anomalie	Nessuna anomalia

5.1.7.1.20 TER_3_1_GetActionMethod

ID Test Case	TC_3_1
Tester	Dario Di Dario
Risultati della Procedura	L'utente, tramite GUI nota che alla voce azioni effettuate, viene specificato l'azione di Refactoring preview
Output Atteso	1
Output Sistema	1
Anomalie	Nessuna anomalia

5.1.7.2 D_ST_ST: AddSession Method

ID Test Case	TC_4_0
Tester	Dario Di Dario
Risultati della Procedura	Il sistema inserisce all'interno della lista la sessione desiderata
Output Atteso	null
Output Sistema	null
Anomalie	N/A

ID Test Case	TC_4_1
Tester	Dario Di Dario
Risultati della Procedura	Il sistema inserisce all'interno della lista la sessione desiderata
Output Atteso	Object != null
Output Sistema	Object != null
Anomalie	N/A

5.1.7.3 D_UT_STB: Blast Method

ID Test Case	TC_5_0

Tester	Dario Di Dario
Risultati della Procedura	Il sistema invia il file al server
Output Atteso	true
Output Sistema	true
Anomalie	N/A

ID Test Case	TC_5_1
Tester	Dario Di Dario
Risultati della Procedura	Il sistema non invia il file al server
Output Atteso	false
Output Sistema	false
Anomalie	N/A

5.1.7.4 D_UT_STS: DeleteJsonFile Method

ID Test Case	TC_6_0
Tester	Gilberto Recupito

Risultati della Procedura	Il sistema non cancella il file json
Output Atteso	false
Output Sistema	false
Anomalie	N/A

ID Test Case	TC_6_1
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema cancella il file json
Output Atteso	true
Output Sistema	true
Anomalie	N/A

5.1.7.5 D_UT_STS: FileExist Method

ID Test Case	TC_7_0
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema controlla l'esistenza del file

Output Atteso	false
Output Sistema	false
Anomalie	N/A

ID Test Case	TC_7_1
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema controlla l'esistenza del file
Output Atteso	true
Output Sistema	true
Anomalie	N/A

5.1.7.6 D_UT_STS: Serialize Method

ID Test Case	TC_8_0
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema salva il file json contenente l'oggetto Stats
Output Atteso	true

Output Sistema	true
Anomalie	N/A

ID Test Case	TC_8_1
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema salva il file json contenente l'oggetto Stats
Output Atteso	false
Output Sistema	false
Anomalie	N/A

ID Test Case	TC_8_2
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema salva il file json contenente l'oggetto Stats
Output Atteso	false
Output Sistema	false
Anomalie	N/A

ID Test Case	TC_8_3
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema salva il file json contenente l'oggetto Stats
Output Atteso	false
Output Sistema	false
Anomalie	N/A

5.1.7.7 D_UT_STU: MD5 Method

ID Test Case	TC_9_0
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema calcola l'hash MD5
Output Atteso	2E4CB449D08B2757D0E8C7FC100C8987
Output Sistema	2E4CB449D08B2757D0E8C7FC100C8987
Anomalie	N/A

ID Test Case	TC_9_1
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema calcola l'hash MD5

Output Atteso	null
Output Sistema	null
Anomalie	N/A

ID Test Case	TC_9_2
Tester	Gilberto Recupito
Risultati della Procedura	Il sistema calcola l'hash MD5
Output Atteso	null
Output Sistema	null
Anomalie	N/A

5.2 DARTSStat

5.2.1 Obiettivi del testing

Come anticipato, lo scopo è di individuare la presenza di *faults* all'interno del sistema e condurre attività di debug. Diciamo che un test ha successo quando, dato un input al sistema, il risultato ottenuto è diverso dal valore previsto, chiamato oracolo.

L'approccio per la definizione dei test frame sarà il *category partition*, quindi, al fine di minimizzare il numero di test case, gli *input* e gli *environment* saranno partizionati in classi di equivalenza.

5.2.2 Approccio

I sottosistemi di DARTStat sono tre: Stat, Session e Action; in più abbiamo il sottosistema General. Ogni sottosistema ha la rispettiva componente all'interno dei moduli del sistema: *model*, *controller* e *router*.

Questi, verranno testati seguendo questo ordine: model, controller e router.

5.2.3 Componenti di testing

Le componenti di test di DARTSStat si trovano all'interno della cartella tests contenuta nella root del progetto.

All'interno abbiamo:

- **controllers/**
  - Contiene gli script di test dei controller del sistema.
- **models/**
  - Contiene gli script di test dei model del sistema.
- **routers/**
  - Contiene gli script di test dei router del sistema.
- **db.mock.data.js**
  - Questo file contiene quattro differenti stati del database:
    - Tutti gli elementi presenti
    - Nessuna action e nessuna session
    - Nessuna Action
    - Nessun elemento
- **scaffold.js**
  - Contiene i metodi utili per la creazione della struttura di *scaffold*
- **serverScaffold.js**
  - Contiene i metodi utili per la creazione della struttura di *scaffold* e del server di test

5.2.4 Test Case Plan

Avendo le tre componenti delle interfacce identiche, verrà effettuato un solo category partition. La nomenclatura segue la seguente struttura: **DS_SS_XX**

- **DS**: Iniziali del sistema DARTSStat
- **SS**: Sub System
- **XX**:

- **AC**: Action
- **GEN**: General
- **SE**: Session
- **ST**: Stat

Vediamo una prima specifica:

5.2.4.1 DS_SS_AC/SE: Sottosistema Action/Session

5.2.4.1.1 Find All

- Parametri:
  - Nessuno
- Ambiente:
  - Numero di action/session nel DB
    - nessuna [property NotPresent]
    - una o più [property Present]

Parametri	
Nessuno	
Ambiente	
Ambiente: Numero di action/session nel DB	
Presente [PR]	1. Nessuna action/session presente all'interno del Database [PR_NP] 2. Una o più action/session presente all'interno del Database [PR_P]

Codice	**Combinazione**	**Esito**
TC_1_1	PR1	Corretto: lista vuota
TC_1_2	PR2	Corretto: lista di item

5.2.4.1.2 Find One

- Parametri:
  - ID Valido:
    - formato non valido [error]
    - formato valido [property Valid]
- Ambiente:
  - Numero di action/session corrispondente nel DB
    - nessuna [error]
    - una [if Valid] [property Present]

Parametri	
Parametro: ID	
Validità [VA]	1. Formato non valido [error] 2. Formato valido [VA_V]
Ambiente	
Ambiente: Numero di action/session corrispondente nel DB	
Presente [PR]	1. Action/Session non presente all'interno del Database [error] 2. Action/Session presente all'interno del Database [if VA_V] [PR_P]

Codice	Combinazione	Esito
TC_2_1	VA1	Non Corretto: ID non valido
TC_2_2	VA2, PR1	Non Corretto: Non esiste alcun item con quell'ID
TC_2_3	VA2, PR2	Corretto: Item

5.2.4.1.3 Delete All

- Parametri:
  - Nessuno
- Ambiente:
  - Numero di action/session nel DB
    - nessuna [property ZeroDeleted]
    - una o più [property DeletedSome]

Parametri	
Nessuno	
Ambiente	
Ambiente: Numero di action/session nel DB	
Presente [PR]	1. Action/Session non presenti all'interno del Database [PR_ZD] 2. Action/Session presenti all'interno del Database [PR_DS]

Codice	Combinazione	Esito
TC_3_1	PR1	Corretto: Nessuna action/session rimossa
TC_3_2	PR2	Corretto: Rimosse una o più action/session

5.2.4.2 DS_SS_ST: Sottosistema Stat

5.2.4.2.1 Create

- Parametri:
  - STAT:
    - body non valido [error]
    - body valido [property BodyValid]

- Ambiente:
    - Nessuno

Parametri	
Parametro: STAT	
Valido [VA]	1. Body della richiesta non valido [error] 2. Body della richiesta valido [VA_V]
Ambiente	
Nessuno	

Codice	**Combinazione**	**Esito**
TC_4_1	VA1	Non Corretto: Item non aggiunto al DB
TC_4_2	PR2	Corretto: Item aggiunto al DB

5.2.4.2.2 Find All

- Parametri:
    - Nessuno
- Ambiente:
    - Numero di stat nel DB
        - nessuna [property NotPresent]
        - una o più [property Present]

Parametri
Nessuno
Ambiente

Ambiente: Numero di stat nel DB	
Presente [PR]	1. Nessuna stat presente all'interno del Database [PR_NP] 2. Una o più stat presente all'interno del Database [PR_P]

Codice	Combinazione	Esito
TC_5_1	PR1	Corretto: lista vuota
TC_5_2	PR2	Corretto: lista di item

5.2.4.2.3 Find One

- Parametri:
  - ID Valido:
    - formato non valido [error]
    - formato valido [property Valid]
- Ambiente:
  - Numero di stat corrispondente nel DB
    - nessuna [error]
    - una [if Valid] [property Present]

Parametri	
Parametro: ID	
Validità [VA]	1. Formato non valido [error] 2. Formato valido [VA_V]
Ambiente	
Ambiente: Numero di action/session corrispondente nel DB	

Presente [PR]	1. Stat non presente all'interno del Database [error] 2. Stat presente all'interno del Database [if VA_V] [PR_P]

Codice	Combinazione	Esito
TC_6_1	VA1	Non Corretto: ID non valido
TC_6_2	VA2, PR1	Non Corretto: Non esiste alcun item con quell'ID
TC_6_3	VA2, PR2	Corretto: Item

5.2.4.2.4 Delete All

- Parametri:
  - Nessuno
- Ambiente:
  - Numero di stat nel DB
    - nessuna [property ZeroDeleted]
    - una o più [property DeletedSome]

Parametri	
Nessuno	
Ambiente	
Ambiente: Numero di action/session nel DB	
Presente [PR]	1. Stat non presenti all'interno del Database [PR_ZD] 2. Stat presenti all'interno del Database [PR_DS]

Codice	Combinazione	Esito
TC_7_1	PR1	Corretto: Nessuna action/session rimossa
TC_7_2	PR2	Corretto: Rimosse una o più action/session

5.2.4.3 DS_SS_GE: Sottosistema General

5.2.4.3.1 General

- Parametri:
  - Nessuno
- Ambiente:
  - Numero di stat nel DB
    - nessuna [property StatNotPresent]
    - una o più [property Present]
  - Numero di session nel DB
    - nessuna [property SessionsNotPresent]
    - una o più [property Present]
  - Numero di action nel DB
    - nessuna [property ActionsNotPresent]
    - una o più [property Present]

Parametri	
Nessuno	
Ambiente	
Ambiente: Numero di stat nel DB	
Stat Presente [PR_St]	1. Nessuna stat presente all'interno del Database [PR_St_NP]

	2. Una o più stat presente all'interno del Database [PR_St_P]
Ambiente: Numero di session nel DB	
Action Presente [PR_Se]	1. Nessuna action presente all'interno del Database [PR_Se_NP] 2. Una o più action presente all'interno del Database [PR_Se_P]
Ambiente: Numero di action nel DB	
Session Presente [PR_Ac]	1. Nessuna session presente all'interno del Database [PR_Ac_NP] 2. Una o più session presente all'interno del Database [PR_Ac_P]

Codice	Combinazione	Esito
TC_8_1	*	Corretto

5.2.5 Test Case Specification

5.2.5.1 DS_SS_AC/SE: Sottosistema Action/Session

5.2.5.1.1 Find All

Test Case ID:	TC_1_1
Pre-Condition:	
All'interno del DB non ci sono action/session	
Flow of events:	
1. HTTP GET /api/{action/session} 2. Il sistema restituisce lista vuota con status code 200 OK	
Oracolo:	

Lista vuota con status code 200 OK

Test Case ID:	TC_1_2
Pre-Condition:	
All'interno del DB ci sono una o più action/session	
Flow of events:	
1. HTTP GET /api/{stat/session} 2. Il sistema restituisce lista contenente action/session con status code 200 OK	
Oracolo:	
Lista contenente action/session con status code 200 OK	

5.2.5.1.2 Find One

Test Case ID:	TC_2_1
Pre-Condition:	

Flow of events:	

1. GET /api/{action/session}/:id

Input	Valore
id	123

2. Il sistema restituisce 500 INTERNAL SERVER ERROR

Oracolo:

Il sistema restituisce 500 INTERNAL SERVER ERROR

Test Case ID:	TC_2_2
Pre-Condition:	
All'interno del DB non c'è la action/session richiesta	
Flow of events:	

1. GET /api/{action/session}/:id

Input	Valore
id	41224d776a326fb40f000001

2. Il sistema restituisce 404 NOT FOUND

Oracolo:
Il sistema restituisce 404 NOT FOUND

Test Case ID:	TC_2_3
Pre-Condition:	
All'interno del DB c'è la action/session richiesta	
Flow of events:	

1. GET /api/{action/session}/:id

Input	Valore
id	41224d776a326fb40f000001

2. Il sistema restituisce la action/session associata a quell'ID con status code 200 OK
Oracolo:
Il sistema restituisce la action/session associata a quell'ID con status code 200 OK

5.2.5.1.3 Delete All

Test Case ID:	TC_3_1
Pre-Condition:	
All'interno del DB non ci sono action/session	
Flow of events:	
1. HTTP DELETE /api/{action/session} 2. Il sistema non rimuove action/session con status code 200 OK	
Oracolo:	
Il sistema non rimuove action/session con status code 200 OK	

Test Case ID:	TC_3_2
Pre-Condition:	
All'interno del DB ci sono una o più action/session	
Flow of events:	
1. HTTP DELETE /api/{action/session} 2. Il sistema rimuove una o più action/session con status code 200 OK	
Oracolo:	
Il sistema rimuove una o più action/session con status code 200 OK	

5.2.5.2 DS_SS_ST: Sottosistema Stat

5.2.5.2.1 Create

Test Case ID:	TC_4_1
Pre-Condition:	

Flow of events:	

1. POST /api/stat

Input	Valore
body	[JSON Object]

2. Il sistema crea la stat
3. Il sistema crea la/le session
4. Il sistema crea la/le action
5. Il sistema restituisce la stat appena creata con status code 200 OK

Oracolo:
Il sistema crea le nuove entità e restituisce 200 OK

Test Case ID:	TC_4_2
Pre-Condition:	

Flow of events:	

1. POST /api/stat

Input	Valore
body	[invalid or empty JSON Object]

2. Il sistema restituisce errore con 500 INTERNAL SERVER ERROR

Oracolo:
Il sistema restituisce errore con 500 INTERNAL SERVER ERROR

5.2.5.2.2 Find All

Test Case ID:	TC_5_1
Pre-Condition:	
All'interno del DB non ci sono stat	
Flow of events:	
1. HTTP GET /api/stat 2. Il sistema restituisce lista vuota con status code 200 OK	
Oracolo:	
Lista vuota con status code 200 OK	

Test Case ID:	TC_5_2
Pre-Condition:	
All'interno del DB ci sono una o più stat	
Flow of events:	
1. HTTP GET /api/stat	

2. Il sistema restituisce lista contenente stat con status code 200 OK
Oracolo:
Lista contenente stat con status code 200 OK

5.2.5.2.3 Find One

Test Case ID:	TC_6_1
Pre-Condition:	

Flow of events:	

1. GET /api/stat/:id

Input	Valore
id	123

2. Il sistema restituisce 500 INTERNAL SERVER ERROR

Oracolo:
Il sistema restituisce 500 INTERNAL SERVER ERROR

Test Case ID:	TC_6_2
Pre-Condition:	
All'interno del DB non c'è la stat richiesta	
Flow of events:	

1. GET /api/stat/:id

Input	Valore
id	41224d776a326fb40f000001

2. Il sistema restituisce 404 NOT FOUND

Oracolo:
Il sistema restituisce 404 NOT FOUND

Test Case ID:	TC_6_3
Pre-Condition:	
All'interno del DB c'è la action/session richiesta	
Flow of events:	

1. GET /api/stat/:id

Input	Valore
id	41224d776a326fb40f000001

2. Il sistema restituisce la stat associata a quell'ID con status code 200 OK

Oracolo:
Il sistema restituisce la stat associata a quell'ID con status code 200 OK

5.2.5.2.4 Delete All

Test Case ID:	TC_7_1
Pre-Condition:	

All'interno del DB non ci sono action/session
Flow of events:
1. HTTP DELETE /api/stat 2. Il sistema non rimuove stat con status code 200 OK
Oracolo:
Il sistema non rimuove stat con status code 200 OK

Test Case ID:	TC_7_2
Pre-Condition:	
All'interno del DB ci sono una o più stat	
Flow of events:	
1. HTTP DELETE /api/stat 2. Il sistema rimuove una o più stat con status code 200 OK	
Oracolo:	
Il sistema rimuove una o più stat con status code 200 OK	

5.2.5.3 DS_SS_GE: Sottosistema General

5.2.5.3.1 General

Test Case ID:	TC_8_1
Pre-Condition:	
All'interno del DB non ci sono stat/action/session	
Flow of events:	

1. HTTP GET /api/general 2. Il sistema restituisce un oggetto con status code 200 OK
Oracolo:
Il sistema restituisce un oggetto con status code 200 OK

Test Case ID:	TC_8_2
Pre-Condition:	
All'interno del DB ci sono stat/action/session	
Flow of events:	
1. HTTP GET /api/general 2. Il sistema restituisce un oggetto con status code 200 OK	
Oracolo:	
Il sistema restituisce un oggetto con status code 200 OK	

5.2.6 Test Execution Report

L'intero report delle esecuzioni dei test per il sistema DARTSSTat è presente all'interno di TravisCI al seguente link: https://app.travis-ci.com/github/darioTecchia/DARTStats/builds.

6 Test di Regressione

In assenza di un'automazione dell'operazione di testing all'interno del tool, è stato necessario effettuare il test manualmente analogamente a quanto riportato nella documentazione della versione precedente di DARTS.

Quindi, è stato utilizzato il progetto fantoccio TestProjectForDarts, il quale presenta le seguenti classi di test per ogni tipologia di test smell:

6.1 Eager Test

- EagerNotPresentTest: Classe che non contiene il test smell 'Eager Test'
- EagerSingleInstanceTest: Classe che contiene una singola istanza di 'Eager Test'
- EagerMoreThanOneTest: Classe che contiene più di una istanza di 'Eager Test'

6.2 General Fixture

- GeneralFixtureNotPresentTest: Classe che non contiene il test smell 'General Fixture'
- GeneralFixtureSingleInstanceTest: Classe che contiene una singola istanza di 'General Fixture'
- GeneralFixtureMoreThanOneTest: Classe che contiene più di una istanza di 'General Fixture'

6.3 Lack Of Cohesion of methods

- LackOfCohesionNotPresentTest: Classe che non contiene il test smell 'Lack Of Cohesion of methods'
- LackOfCohesionSingleInstanceTest: Classe che contiene una singola istanza di 'Lack Of Cohesion of methods'
- LackOfCohesionMoreThanOneTest: Classe che contiene più di una istanza di 'Lack Of Cohesion of methods'

Quindi è stato eseguito il test di regressione che ha riportato i seguenti risultati:

L'esecuzione dei test ha riportato un riscontro positivo e il sistema è stato capace di rilevare gli smell definiti.

7 Sviluppi Futuri

7.1 Nuovo Metodo per il calcolo delle classi di test e di produzione in DARTS

Il tool è limitato dalla convenzione che il riconoscimento tra una classe di test e una classe di produzione è derivabile dal nome del file, ovvero se la classe contiene la stringa *"Test"* all'interno del nome. Sarebbe interessante ricavare il riconoscimento della classe secondo altre caratteristiche, in modo da non limitare l'utente a dover nominare una classe di test con il nome di test.

7.2 Nuovo Metodo per l'assegnazione di un identificativo dell'utente e del progetto al Server

Attualmente il tool utilizza per la collezione delle statistiche un identificativo che corrisponde al MAC address del dispositivo utilizzato (cifrato con md5). Sarebbe interessante utilizzare altre tecniche per il riconoscimento del progetto e delle statistiche, come l'assegnazione di un ID univoco da parte del server e che poi venga utilizzato dal tool lato client.

7.3 Diff check degli artefatti rifattorizzati

Il server attualmente permette di visualizzare se un'operazione di refactoring è stata effettuata oppure no, senza specificare quale modifica è stata effettuata. Sarebbe interessante dare la possibilità agli sviluppatori del tool, consultando le statistiche raccolte, poter visualizzare lo snippet di codice pre e post rifattorizzazione.

7.4 Separazione della logica applicativa da quella di plugin con sotto progetti Gradle

Ci è stata indicata la possibilità di poter creare sotto progetti all'interno di un progetto Gradle.

Quindi, data questa possibile, sarebbe l'ideale separare ulteriormente i due layer, applicativa e plugin, tramite sotto progetti Gradle.

Purtroppo, vista la mancanza di tempo e vista la nostra poca esperienza in materia Gradle, abbiamo deciso di rimandare questa modifica alla prossima attività di maintenance del tool.