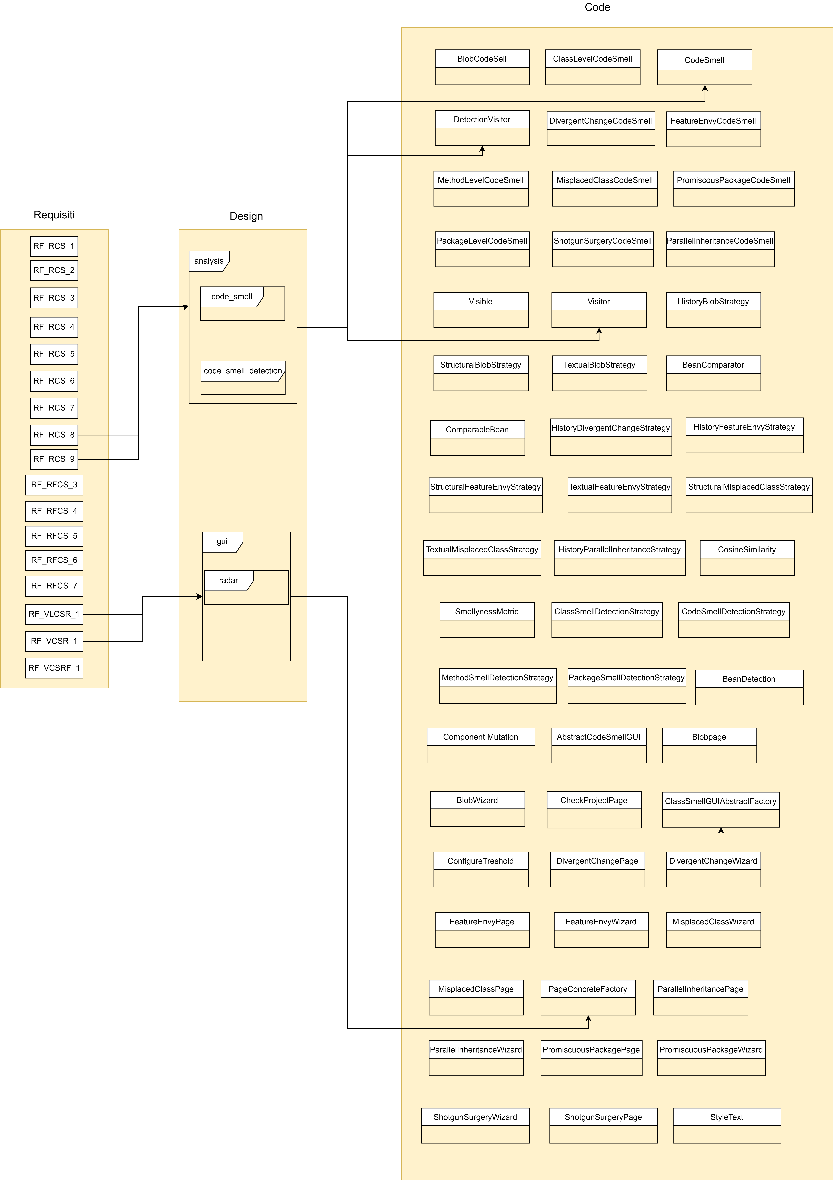
Impact Analysis

* **Identificazione Start impact set (SIS)**

L’impact analysis è iniziata col costruire un grafo della tracciabilità andando ad analizzare gli artefatti disponibili di cASpER. Il grafo risultante e il seguente (Per una miglior visualizzazione è disponibile nella cartella /Immagini/TraceabilityGraph):



Siamo partiti dai requisiti inserendo nell’omonima pila tutti i requisiti funzionali offerti da cASpER compresi i nuovi requisiti da implementare. Dai nuovi requisiti da implementare inizia la tracciabilità orizzontale. Pertanto nella pila dedicata al design sono stai inseriti solamente i moduli che gestiscono tali requisiti, ovvero, Analysis e GUI. La tracciabilità orizzontale è stata mantenuta tra **RF\_VLCSR\_1**,**RF\_VCSR\_1** e l’intero modulo GUI. Mentre per **RF\_RCS\_8** e **RF\_RCS\_9** la tracciabilità orizzontale è stata mantenuta col modulo analysis. In quanto tale modulo gestisce la detection dei code smell.

Dopo aver ottenuto la tracciabilità orizzontale tra la pila dei requisiti e quella del design siamo passati alla pila del codice, inserendo in essa tutte le classi del modulo analysis e GUI. Le uniche classi che vengono impattate dall’aggiunta di un nuovo code smell risultano essere CodeSmell, DetectionVisitor, Visitor, ClassSmellGUIAbstractFactory e PageConcreteFactory.

Dopo la pila del codice dovrebbe essere presente la pila dei test per capire quali test vengono impattati dalla CR, tuttavia in cASpER non è implementato nessun test relativo alle classi impattate, pertanto la pila dei test risulta essere vuota.

In conclusione lo start impact set (SIS) risulta essere il seguente, {**CodeSmell, DetectionVisitor, Visitor, ClassSmellGUIAbstractFactory, PageConcreteFactory**}

* **Identificazione Candidate Impact Set (CIS)**

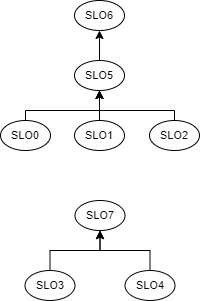
In tale fase integriamo il SIS con gli SLO, che possono cambiare a causa di modifiche agli elementi del SIS, analizzando gli impatti diretti e indiretti

|  |  |
| --- | --- |
| * SLO0 - CodeSmell * SLO1 - DetectionVisitor * SLO2 - Visitor * SLO3 - ClassSmellGUIAbstractFactory * SLO4 - PageConcreteFactory * SLO5 - PsiParser * SLO6 - ClassBean * SLO7 - CheckProjectPage |  |

Aggiungendo un nuovo CodeSmell andremo ad impattare direttamente sulla classe PsiParser, in quanto essa avvia la detection degli smell. Pertanto dovrà essere inserita anche la detection dei nuovi smell da implementare. Tuttavia per la rilevazione di SpaghettiCode e SwissArmyKnife, bisogna controllare se implementano più interfacce o estendono qualche classe. Attualmente la classe ClassBean costruita sulla classe PsiClass non mantiene traccia di tali informazioni.

Andremo a modificare la classe PsiParser in modo che quando si esegue il parse da PsiClass a ClassBean vengono salvate le classi implementate e le classi estese. Tale modifica quindi impatterà in maniera diretta sulla classe ClassBean poichè dovrà essere aggiunto un attributo per conservare le classi implementate ed uno per conservare le classi estese.

Inoltre anche la classe CheckProjectPage sarà impattata direttamente per l’aggiunta di una nuova page.



Dopo aver costruito i vari grafi di connettività è stata costruita la matrice di tracciabilità. Essa è disponibile nel file excel Matrice Connettività.

In conclusione il candidate impact set previsto per l’implementazione di tale change request risulta essere il seguente, {**CodeSmell, DetectionVisitor, Visitor, ClassSmellGUIAbstractFactory, PageConcreteFactory, PsiParser, ClassBean, CheckProjectPage** }

* **Actual Impact Set (AIS), Discovered Impact Set (DIS) e False Positive Impact Set (FPIS)**
* AIS

|  |  |
| --- | --- |
| CodeSmell,  DetectionVisitor,  Visitor,  ClassSmellGUIAbstractFactroy  PageConcreteFactroy,  PsiParser,  ClassBean  CheckProjectPage  WizardConcreteFactory |  |

* DIS

|  |
| --- |
| WizardConcreteFactory |

**Recall Impact Analysis:** |CIS ∩ AIS| /|AIS| = 8/9 = 0,88

**Precision Impact Analysis:** |CIS ∩ AIS| /|CIS| = 8/8 = 1