**PROGETTO “SMART HOME”**

INDICE

1. Introduzione 2
2. Elaborati Analisi 3
   1. Casi D’uso dettagliati 3
      1. Configurazione 3
      2. Creazione Scenario 4
   2. Diagramma Casi D’uso 4
   3. Diagrammi di sequenza di sistema 6

2.3.1. Crea Scenario 6

2.3.2. Configurazione 7

* 1. Modello di Dominio 8
  2. Diagramma Attività: HSC 9

3. Elaborati Progettazione

* 1. Modello di Progettazione 10
  2. Diagramma di Sequenza 11
  3. Diagramma Macchina a Stati 12
  4. Diagramma Architettura SW 13

4. Pattern 14

5 Strumenti Utilizzati 15

# INTRODUZIONE

Il sistema Smart Home permette all’utente di gestire determinati oggetti di una casa attraverso il monitoraggio semi-automatizzato di un sistema di controllo.

IL sistema prevede le seguenti funzionalità principali:

* CONFIGURAZIONE:

L’utente tramite la configurazione inserisce tutti gli “elementi” che fanno parte dell’abitazione a partire dalle stanze fino alle singole tipologie di oggetti ed i relativi sensori.

Nella configurazione viene considerata anche la programmazione del sistema di riscaldamento e del settaggio del sistema di sicurezza.

* SCENARIO:

Lo scenario rappresenta un insieme di azioni scelte dall’utente ed eseguite dal sistema; l’utente potrà selezionare una serie di elementi sui quali, a seconda della azione e dall’orario scelto dall’utente, il sistema si occuperà di eseguirne l’eventuale cambiamento di stato di ogni oggetto facente parte dello scenario.

Inoltre, è presente uno scenario particolare legato alla sicurezza dell’abitazione.

* MODIFICA DELLO STATO DEGLI OGGETTI:

Il sistema è in grado, in base alla configurazione dell’utente, di modificare lo stato degli oggetti appartenenti al sistema grazie ai segnali inviati dai sensori al sistema.

L’utente può in qualsiasi momento modificare lo stato degli oggetti indipendentemente da scenari o settaggi definiti dal sistema.

Ora procediamo alla descrizione dettagliata delle scelte progettuali legate a ciascun elaborato:

# 2. ELABORATI ANALISI

## 2.1 CASI D’USO IN FORMATO DETTAGLIATO

### 2.1.1 Configurazione

|  |  |
| --- | --- |
| Nome del caso d’uso | Configurazione |
| Portata | Sistema di Smart-Home |
| Livello | Obiettivo utente |
| Attore Primario | Utente |
| Parti Interessate e Interessi | Utente: Vuole inserire i dati di valore per il sistema. |
| Pre-condizioni | Nessuna |
| Garanzia di successo | Il Sistema aggiunge le informazioni inserite nella sua base di conoscenza. |
| Scenario Principale di Successo | 1. L’Utente seleziona l’opzione di configurazione ambiente. 2. Il Sistema apre un’interfaccia che permette all’Utente di aggiungere una stanza. 3. L’Utente assegna un nome alla stanza. 4. L’Utente aggiunge gli oggetti contenuti nella stanza.   L’Utente ripete i passi 3 e 4 fino al termine.  5. Il Sistema mostra il riepilogo dei dati inseriti dall’Utente.  6. L’Utente conferma le scelte fatte.  7. Il Sistema salva le informazioni ricevute. |
| Estensioni | \*a.  In qualsiasi momento il sistema fallisce:   1. Il Sistema gestisce l’errore lanciando un’eccezione.          4a.  L’oggetto inserito necessita di un accoppiamento manuale con  un sensore:   1. L’Utente sceglie il sensore da accoppiare tra quelli compatibili con l’oggetto.          4b.  L’oggetto inserito non è compatibile con nessun sensore  esistente e necessita di un accoppiamento manuale:   1. Il Sistema genera un sensore accoppiato all’oggetto inserito.       6a.  L’Utente non conferma le scelte fatte.   1. Il Sistema ritorna al punto 2 dello scenario principale.          7a.  Il salvataggio dei dati non va a buon fine.   1. Il Sistema notifica all’Utente il mancato salvataggio dei dati. 2. Il Sistema ritorna al punto 2 dello scenario principale. |
| Requisiti speciali | * Il testo deve essere visibile da una distanza di 50 cm. * L’interfaccia Utente è di tipo touch screen. |
| Elenco delle variabili tecnologiche e dei dati | Nessuna. |
| Frequenza di ripetizione | Una sola volta. |
| Problemi aperti | Il tipo di interfaccia Utente non è ancora definito. |

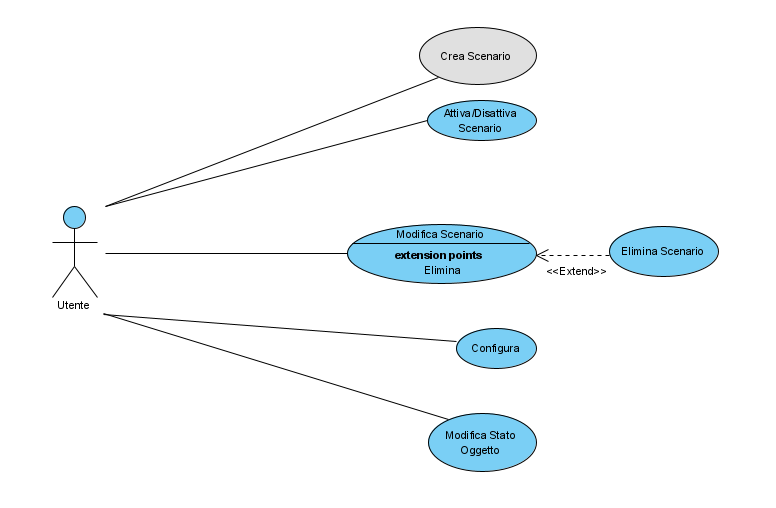
### 

### 2.2.2 Creazione Scenario

|  |  |
| --- | --- |
| Nome del caso d’uso | Crea scenario |
| Portata | Sistema di Smart-Home |
| Livello | Obiettivo utente |
| Attore Primario | Utente |
| Parti Interessate e Interessi | Utente: Vuole creare una schedule di azioni che verranno eseguite dal sistema rispettando una programmazione oraria oppure attivate manualmente. |
| Pre-condizioni | Il sistema deve essere già configurato. |
| Garanzia di successo | Il Sistema aggiunge la schedule alla lista degli scenari esistenti. |
| Scenario Principale di Successo | 1. L’Utente seleziona l’opzione di creazione scenario dall’interfaccia grafica. 2. Il Sistema apre un’interfaccia che permette all’Utente di scegliere la tipologia di programmazione oraria. 3. L’Utente seleziona la stanza desiderata. 4. L’Utente seleziona le azioni, che devono essere eseguite dal sistema, relative agli oggetti della stanza selezionata.   L’Utente ripete i passi 3 e 4 fino al termine.   1. Il Sistema mostra il riepilogo delle azioni precedentemente selezionate dall’Utente. 2. L’Utente conferma le scelte fatte. 3. Il Sistema salva lo scenario creato. |
| Estensioni | \*a. In qualsiasi momento il sistema fallisce:                 1.  Il Sistema gestisce l’errore lanciando un’eccezione.        3a.  La stanza selezionata è già presente in uno                 scenario che si attiva nello stesso orario:   1. Il Sistema notifica all’Utente la presenza di un conflitto. 2. Il Sistema ritorna al punto 3 dello scenario principale.          6a.  L’Utente non conferma le scelte fatte.   1. Il Sistema ritorna al punto 2 dello scenario principale.          7a.  Il salvataggio dello scenario non va a buon fine.   1. Il Sistema notifica all’Utente il mancato salvataggio dello scenario. 2. Il Sistema ritorna al punto 2 dello scenario principale. |
| Requisiti speciali | * Il testo deve essere visibile da una distanza di 50 cm. * L’interfaccia Utente è di tipo touch screen. |
| Elenco delle variabili tecnologiche e dei dati | Nessuna. |
| Frequenza di ripetizione | Ogni volta che L’Utente vuole creare uno scenario. |
| Problemi aperti | Il tipo di interfaccia Utente non è ancora definito. |

2.2 DIAGRAMMA CASI D’USO

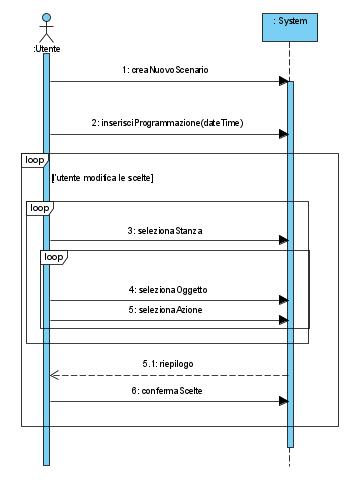
L’attore principale, l’unico presente, che interagisce con il sistema è l’utente, il quale potrà raggiungere i seguenti obiettivi:

* Configurazione del sistema SmartHome
* Creazione/Modifica ed Eliminazione di Scenario
* Attivazione/Disattivazione di Scenario
* Modifica Stato di Oggetto

2.3 DIAGRAMMI DI INTERAZIONE

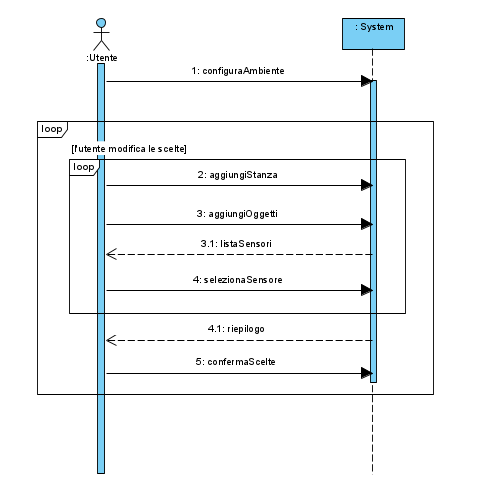
### 2.3.1 Creazione Scenario

Per creare un nuovo scenario l’utente può inserire, se vuole, la data e l’orario in cui lo scenario verrà attivato dal sistema, per poi selezionare le stanze coinvolte con le relative azioni sugli oggetti appartenenti alla stanza selezionata.

Alla fine della procedura l’utente ottiene un riepilogo delle azioni selezionate con eventuale conferma da parte di quest’ultimo.**2.3.2 Configurazione**

Alla prima interazione con il sistema, l’utente si preoccupa di configurare il sistema tramite l’inserimento dei seguenti dati:

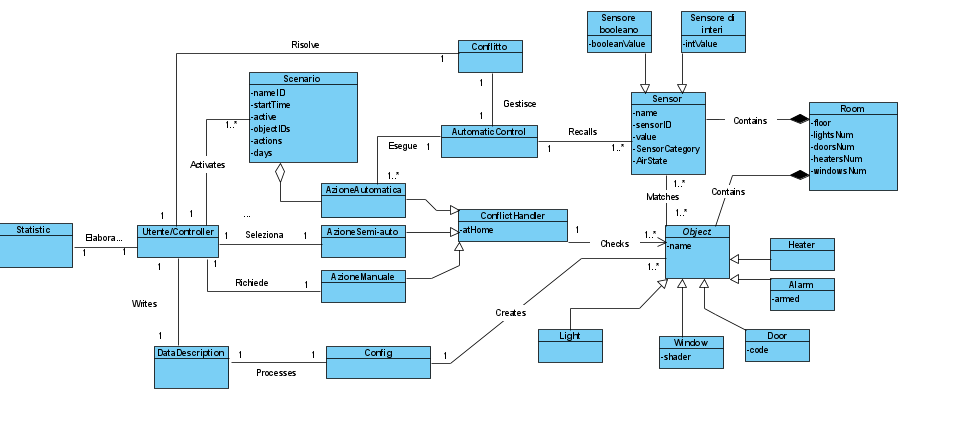
* Configurazione generale dell’ambiente tramite l’inserimento della presenza dell’allarme e degli orari relativi alla fascia oraria giornaliera e notturna
* Inserimento della stanza: per ognuna viene chiesto l’inserimento degli oggetti e dei sensori, queste operazioni vengono ripetute per ogni stanza inserita dall’utente.



2.4 MODELLO DI DOMINIO

Nel modello di dominio viene rappresentato il sistema mediante l’utilizzo di classi che identificano i concetti della realtà.

Il seguente diagramma inquadra la prima trasposizione delle classi da realtà a schema (anche quelle non implementate nella applicazione finale).

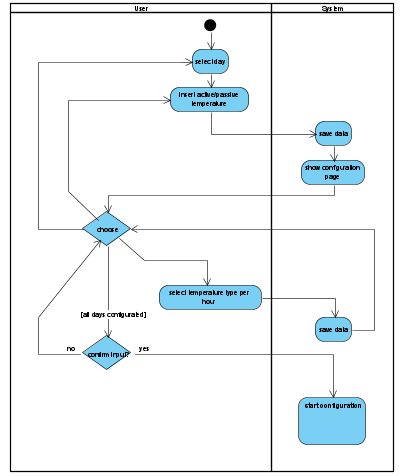


2.5 DIAGRAMMI DI ATTIVITA

Diagramma di attività Heat System Config:

Il diagramma seguente descrive il percorso di attività legate alla configurazione del sistema di riscaldamento.

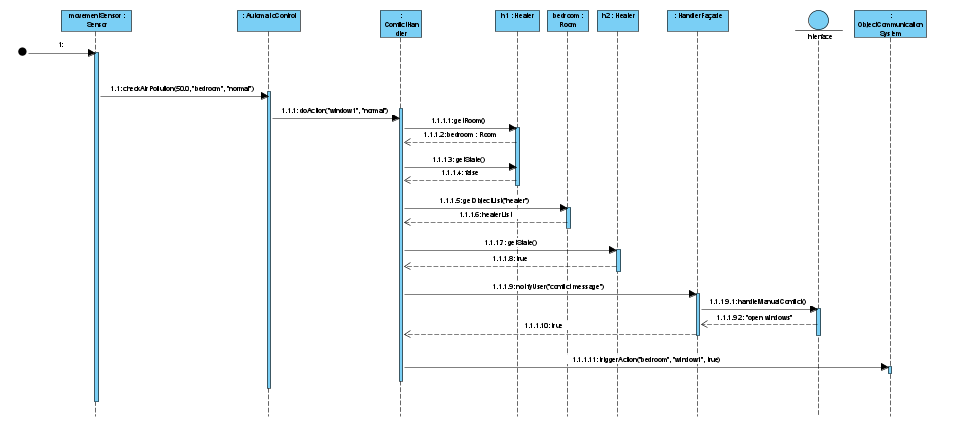
Per il giorno selezionato, l’utente inserisce la temperatura minima garantita(passiva) e quella ottimale (attiva) per poi impostare la programmazione desiderata per tutte le ore del giorno selezionato. L’ utente può decidere di confermare oppure cambiare le scelte precedentemente fatte.



## 3.1 MODELLO DI PROGETTO

3.2 DIAGRAMMI DI SEQUENZA

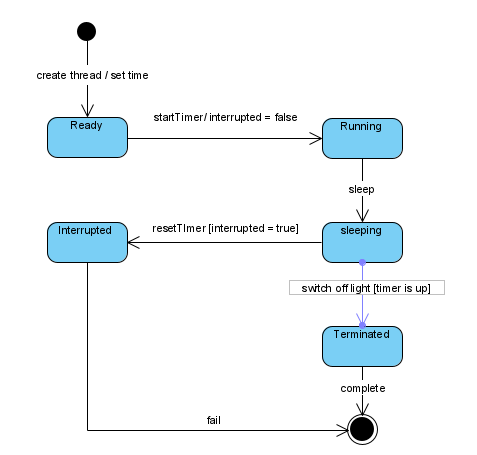
Il seguente diagramma di sequenza di progettazione mostra le operazioni svolte dal sistema alla chiamata dell’azione di controllo della qualità dell’aria.



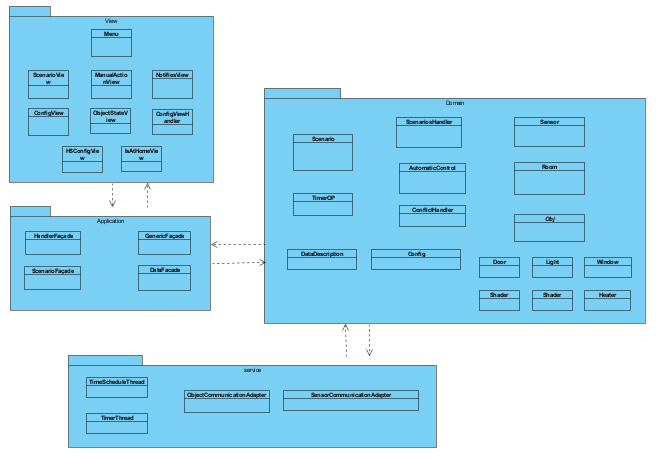
3.3 DIAGRAMMA MACCHINA A STATI

Diagramma macchina a stati relativo al thread di tipo light:

lo schema mostra il cambiamento dello stato di quest’ultimo in relazione agli eventi, in particolare si evidenzia il cambiamento di stato in relazione al segnale ricevuto dal sensore. Nel caso non dovesse ricevere alcun segnale per un periodo di tempo prefissato (time), il thread termina con lo spegnimento della luce, altrimenti passa nello stato di interrupt e termina.



3.4 DIAGRAMMA ARCHITETTURA SW



4 PATTERN

# **ARCHITECTURAL PATTERN**

* MVC:

Abbiamo utilizzato il pattern Model-View-Controller per dividere il sistema in tre blocchi dalle funzionalità ben distinte:

1. Il package domain è la parte centrale del software dove vengono gestiti tutti i calcoli (Model)
2. Il package view è l’interfaccia di comunicazione con l’utente (View)
3. Il package application gestisce la delegazione delle azioni da effettuare sul sistema, rendendo possibile la comunicazione tra gli altri due package(controller).

# **DESIGN PATTERN**

* Observer:

Il pattern Observer si è reso utile in quelle situazioni dove diversi oggetti (Observer) avevano la necessità di conoscere lo stato di un oggetto (Subject). Nel nostro caso abbiamo usato la classe Sensor come observer e le classi Object e AutomaticControl come subjects.

* Singleton (creazionale)

Abbiamo utilizzato il pattern Singleton per garantire la presenza di un’unica istanza relativamente ad alcune classi. Nello specifico:

1. Alarm: in ogni casa può esserci un unico allarme e tutti i sensori relativi si devono collegare a quest’ultimo.
2. AutomaticControl: questa classe deve conoscere tutti i sensori, per cui devono riferirsi sempre alla stessa istanza.
3. Config: la configurazione effettuata dall’utente deve essere effettuata una sola volta, quindi deve esserci un’unica istanza.
4. ConflictHandler: similmente di AutomaticControl, ConflictHandler deve conoscere tutti gli oggetti, i quali devono essere sempre riferiti alla stessa istanza.

* Facade (strutturale):

L’obiettivo di questo pattern è quello di nascondere la complessità strutturale di un sistema, mostrando solamente una “facciata” che nasconde la parte di back-end. Nel nostro caso lo abbiamo utlizzato per delegare e organizzare le operazioni relative al dominio richieste dall’interfaccia.

5 STRUMENTI UTILIZZATI

# 

# Github

Nel corso dello sviluppo abbiamo utilizzato prevalentemente il branch developement per l’implementazione delle principali funzionalità, lasciando il main libero per l’upload dei vari diagrammi di analisi e progettazione. Abbiamo inoltre creato un branch interface per l’implementazione dell’interfaccia di comunicazione con l’utente. Nella parte finale abbiamo lavorato quasi interamente sul branch main tranne che durante la fase di refactoring per la quale abbiamo creato un branch apposito.

# Sonarcloud

Abbiamo utilizzato la versione cloud della piattaforma per l’analisi del nostro codice. Questo ci ha aiutato a risolvere tutti i bug e alcuni dei code smell presenti, oltre a tenere sotto controllo la fase di test.

Coverage:

Abbiamo utlizzato il plugin jacoco per maven che ci ha permesso di generare dei report sulla coverage e successivamente integrarli nell’analisi sonarcloud.

Abbiamo deciso di escludere dal conteggio della coverage il package view, comprensivo del codice riguardante l’interfaccia, svincolato quindi dalla logica applicativa del programma.

# Maven

Abbiamo utlizzato maven per la fase di build del progetto

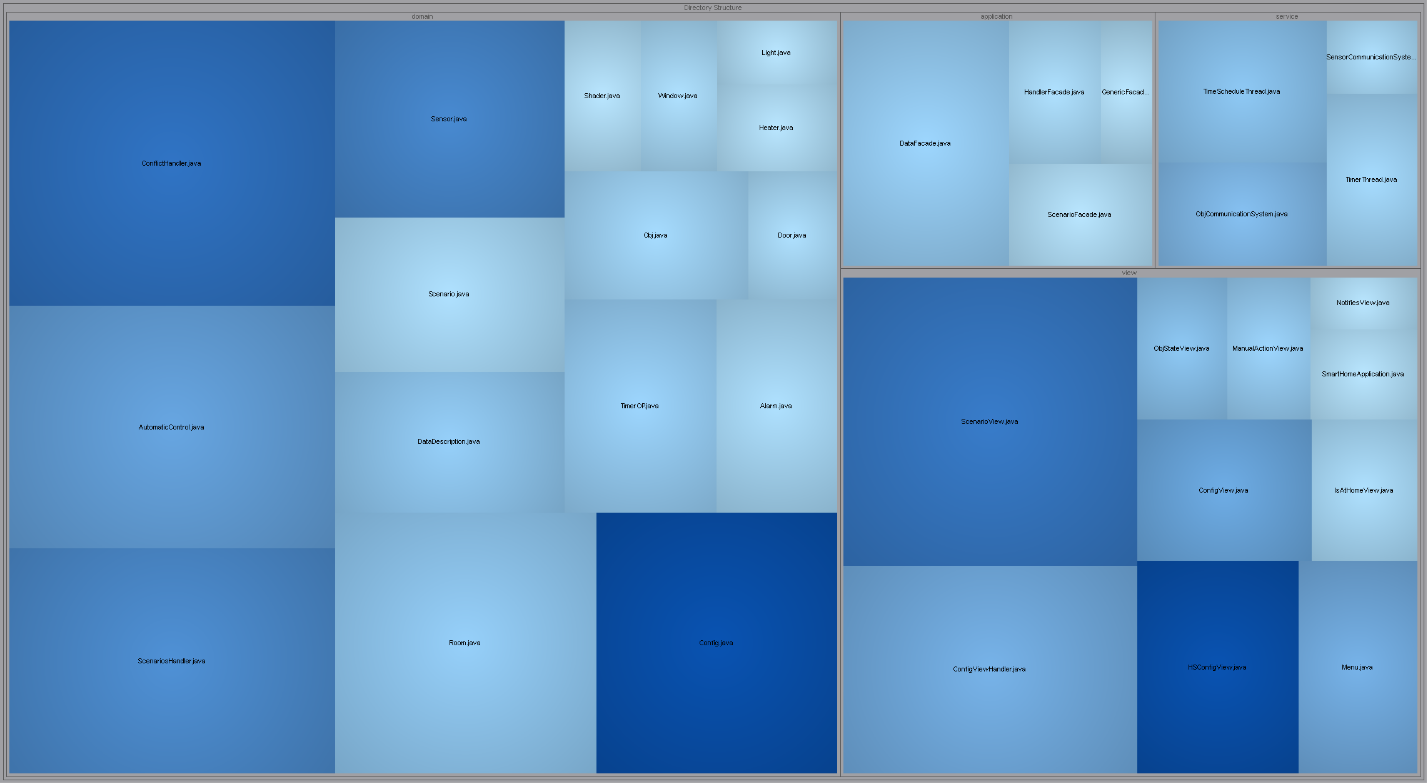
# Junit

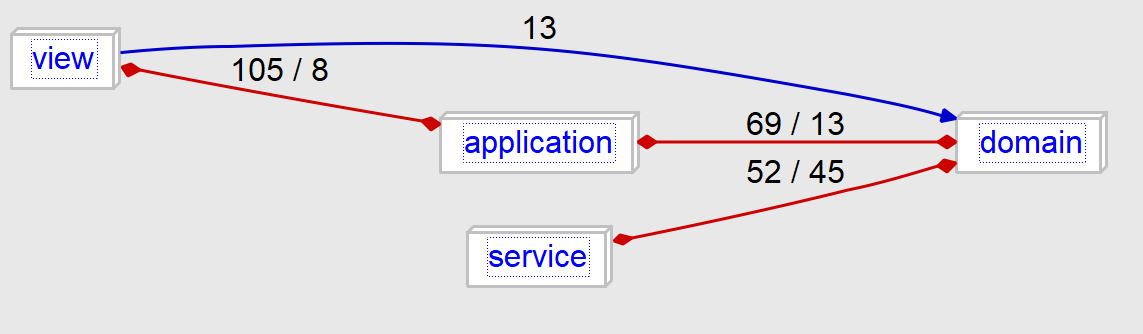
Junit si è rivelato utile per effettuare i test relativi al nostro codice

# Install4j

Tramite Install4j abbiamo creato un wizard di installazione per il progetto, in modo tale da facilitarne l’esecuzione da parte di un utente esterno

# Understand





*Barzio Davide 844556*

*Basavecchia Lorenzo 845111*

*Benaglia Federico 845138*