

Progetto di Ingegneria del software

# **Smart Home**

Luca Milanesi 886279  
Bryan Anozie 885869

*Anno Accademico 2023-2024*

# Indice

<b>1 Introduzione</b>	<b>4</b>
1.1 Settori di Competenza . . . . .	4
1.2 Gestione dell'automazione . . . . .	5
<b>2 Requisiti</b>	<b>6</b>
2.1 Requisiti Funzionali . . . . .	6
2.2 Requisiti non Funzionali . . . . .	6
<b>3 Analisi</b>	<b>8</b>
3.1 Diagramma dei Casi d'Uso . . . . .	8
3.2 Attori . . . . .	8
3.3 Casi d'Uso . . . . .	8
3.3.1 Gestione Manuale . . . . .	8
3.3.2 Rileva Dati . . . . .	9
3.3.3 Gestione Automatica . . . . .	9
3.3.4 Gestisci Automatizzazione . . . . .	10
3.3.5 Controllo Luci . . . . .	10
3.3.6 Controllo Temperatura . . . . .	11
3.3.7 Gestisci Preferenze Temperatura . . . . .	13
3.3.8 Controllo Pulizia . . . . .	13
3.3.9 Controllo Protezione . . . . .	15
3.3.10 Chiamata di Emergenza . . . . .	16
3.4 Modello di dominio . . . . .	17
3.4.1 Spiegazione delle varie componenti . . . . .	17
3.5 Diagrammi di Sequenza di Sistema . . . . .	19
3.5.1 Gestione Manuale . . . . .	19
3.5.2 Rileva Dati . . . . .	20
3.5.3 Gestione Automatica . . . . .	21
<b>4 Progettazione</b>	<b>22</b>
4.1 Diagrammi di Sequenza di Progettazione . . . . .	22
4.1.1 mapeIteration . . . . .	22
4.1.2 automateSensorControl . . . . .	23
4.1.3 onDataChange . . . . .	24
4.2 Diagramma degli Stati . . . . .	27
4.2.1 Folletto per la Pulizia . . . . .	27
4.3 Diagramma delle Attività . . . . .	28

4.3.1	Controllo della Temperatura Automatico . . . . .	28
4.4	Diagramma delle Classi Software e Diagramma dei Package . . . . .	29
<b>5</b>	<b>Simulazione dell'Ambiente</b>	<b>30</b>
5.1	Spiegazione delle varie componenti . . . . .	30
<b>6</b>	<b>GUI</b>	<b>32</b>
<b>7</b>	<b>Conclusioni</b>	<b>33</b>
7.1	Gestione del Ciclo di Sviluppo del Software . . . . .	33
7.1.1	Metodologia Iterativa e Agile . . . . .	33
7.1.2	Divisione dei Compiti e Collaborazione . . . . .	33
7.1.3	Strumenti e Tecnologie . . . . .	33
7.1.4	Revisione del Codice, Test e Validazione . . . . .	34
7.2	Design Pattern . . . . .	35
7.2.1	Builder . . . . .	35
7.2.2	Observer . . . . .	35
7.2.3	Strategy . . . . .	35
7.2.4	Template Method . . . . .	35
7.2.5	State . . . . .	35
7.2.6	Singleton . . . . .	35
7.3	Pattern Architetturali . . . . .	36
7.3.1	Page Controller . . . . .	36
7.4	Design Principle . . . . .	37
7.4.1	Single Responsibility . . . . .	37
7.4.2	DRY - Don't Repeat Yourself . . . . .	37
7.4.3	Separation of Concerns . . . . .	37
7.4.4	Open Closure . . . . .	37
7.4.5	Common Closure . . . . .	37
7.4.6	Common Reuse . . . . .	38
7.4.7	Stable Abstraction . . . . .	38
7.5	Understand . . . . .	39
7.5.1	Controller Package . . . . .	39
7.5.2	ObservableElement . . . . .	40
7.6	Sviluppi Futuri . . . . .	41
7.6.1	Comunicazione Client-Server . . . . .	41
7.6.2	Sistema di Messaggistica Client-Server . . . . .	41
7.6.3	Protocollo TCP/IP . . . . .	41

# 1 Introduzione

Il progetto Smart Home mira a migliorare la qualità della vita dei residenti attraverso un sistema che fornisce varie funzionalità, sia manuali che automatiche, per vivere al meglio nella propria abitazione. Le funzionalità principali includono il controllo delle luci, la gestione della temperatura nelle varie stanze, il controllo della pulizia e la sicurezza dell'abitazione.

Il monitoraggio delle varie condizioni dell'abitazione avviene attraverso l'uso di sensori distribuiti nell'abitazione, che monitorano vari parametri ambientali da passare poi al sistema che li elabora e, a seconda della rilevazione effettuata, ne trae delle conclusioni finali per poter eseguire opportunamente delle operazioni.

Il sistema è anche in grado di prendere decisioni automatiche. Ad esempio, può regolare la temperatura in base alle preferenze dei residenti o attivare la sicurezza in risposta a movimenti sospetti rilevati da sensori.

Il progetto si propone di offrire una soluzione intelligente e adattabile per l'ambiente domestico, migliorando la comodità, la sicurezza e l'efficienza della vita quotidiana dei residenti.

## 1.1 Settori di Competenza

### 1. Controllo dell'illuminazione:

- *manuale* tramite attuatori.
- *automatico* tramite controlli che si appoggiano a rilevazioni effettuate tramite sensori di rilevamento presenza.

### 2. Controllo della temperatura

- *manuale* tramite comandi diretti ad un climatizzatore.
- *automatico* tramite controlli che si appoggiano a rilevazioni effettuate tramite sensori di temperatura.

### 3. Controllo della pulizia

- *manuale* tramite comandi diretti ad un folletto per la pulizia.

### 4. Controllo della sicurezza

- *manuale* tramite comandi diretti al sistema di sicurezza.
- *automatico* tramite controlli che si appoggiano a rilevazioni effettuate tramite sensori di rilevamento presenza e ad un allarme.

## 1.2 Gestione dell'automazione

Il sistema Smart Home per la gestione dei controlli automatici adotta una strategia avanzata attraverso l'utilizzo del **MAPE control feedback loop**. Questa tipologia di approccio consente una regolazione dinamica dei campi di interesse, assicurando un controllo automatizzato e ottimizzato in risposta alle condizioni rilevate dai sensori.

Questa strategia è un loop composto da 4 diverse principali operazioni:

- **Monitor:** Raccoglie dati dal sistema.
- **Analyze:** Interpreta e comprende i dati.
- **Planning:** Sviluppa un piano basato sull'analisi.
- **Execute:** Implementa il piano per influenzare il sistema.

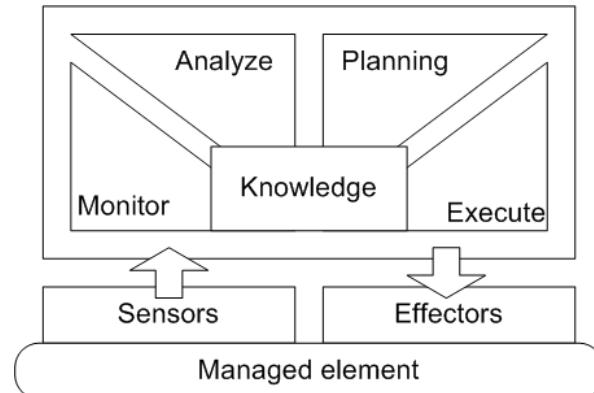


Figura 1: Diagramma del MAPE control feedback loop

## **2 Requisiti**

### **2.1 Requisiti Funzionali**

1. Possibilità di accendere o spegnere manualmente le luci di una specifica stanza manualmente.
2. Gestire automaticamente l'accensione e lo spegnimento delle luci in base ai sensori di rilevamento presenza. Se l'accensione automatica è attiva, il sistema accende la luce quando rileva la presenza e la spegne dopo un periodo predefinito di assenza nella stanza.
3. Possibilità di regolare manualmente la temperatura di una specifica stanza manualmente.
4. Regolare automaticamente la temperatura di una stanza alla temperatura ideale in base ai sensori di temperatura. Questa temperatura ideale può essere predefinita o impostata dal residente. Se la temperatura corrente supera una soglia prestabilita, il sistema, se la regolazione automatica della temperatura è attiva, interviene regolando la temperatura della stanza.
5. Possibilità di avviare o terminare manualmente il processo di pulizia di ogni stanza della casa attraverso un folletto per la pulizia.
6. Possibilità di armare e disarmare manualmente l'allarme per la sicurezza della casa.
7. Gestione della sicurezza della casa attraverso un sistema di sicurezza automatico che entra in azione quando un sensore di presenza rileva movimenti mentre l'allarme è attivo. Quando questo sistema di sicurezza entra in azione, viene attivata una sirena di emergenza e, se non disattivata entro un periodo preimpostato, il sistema avvia automaticamente una chiamata alle forze dell'ordine, disattivando contemporaneamente tutte le luci della casa.

### **2.2 Requisiti non Funzionali**

1. Progettare il sistema in modo che possa essere espanso facilmente per supportare un numero crescente di stanze, sensori o dispositivi senza compromettere le prestazioni.

2. Implementare un sistema robusto per gestire eventi bloccanti asincronamente, come ad esempio il controllo della pulizia o la rilevazione continua di dati da parte di un sensore.
3. Implementare dei test completi e approfonditi per verificare la correttezza del sistema. Questi test dovrebbero coprire scenari diversificati, situazioni di utilizzo comuni e borderline, assicurando che il sistema operi in modo affidabile, coerente e conforme alle specifiche funzionali stabilite.
4. Assicurare che il sistema sia facilmente manutenibile, consentendo aggiornamenti del software, correzioni di bug e l'aggiunta di nuove funzionalità.

## 3 Analisi

### 3.1 Diagramma dei Casi d'Uso

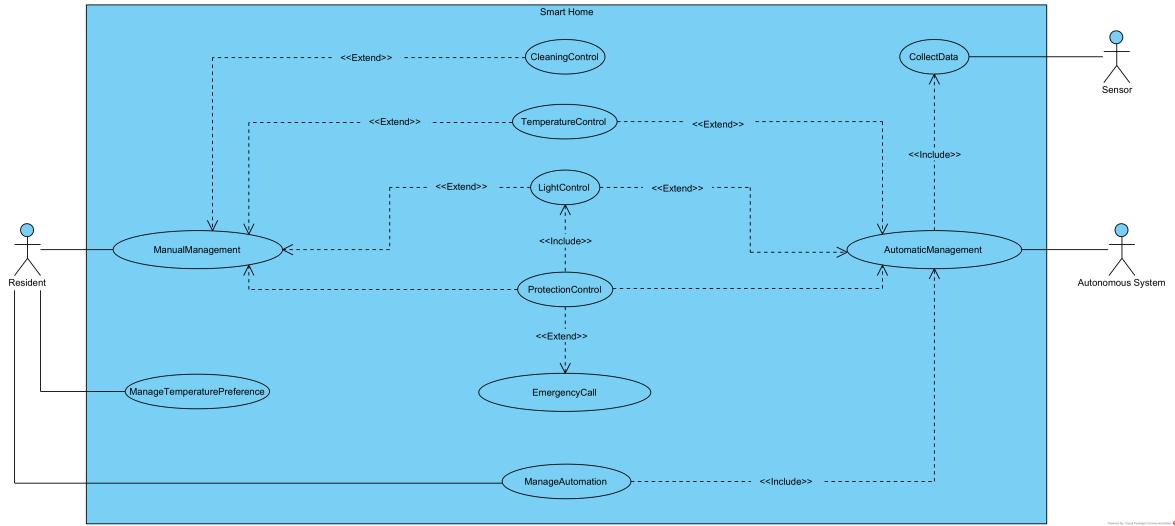


Figura 2: Diagramma dei casi d'uso

### 3.2 Attori

- **Residente:** Il residente dell'abitazione che ha accesso al sistema Smart Home.
- **Sistema Automatico:** Sottosistema dedicato alla gestione automatica dei controlli in base ai dati rilevati dai sensori a cui si appoggia.
- **Sensore:** Un sensore generale presente in una stanza della casa, responsabile della rilevazione di dati specifici relativi al suo campo di interesse.

### 3.3 Casi d'Uso

#### 3.3.1 Gestione Manuale

##### Attori

- Residente

**Descrizione** Il Residente può eseguire manualmente un'azione su una componente della casa, controllando direttamente le operazioni.

**Precondizioni** -

### Scenario principale

1. Il Residente desidera eseguire un'azione su una componente della casa.
2. Il Residente interagisce con un dispositivo per controllare il componente della casa.
3. Il dispositivo per controllare il componente della casa riceve la richiesta.
4. Il Sistema esegue l'azione richiesta sul componente della casa.

### 3.3.2 Rileva Dati

**Attori**

- Sensore

**Descrizione** Il Sensore presente in una stanza, attuando il comportamento del MAPE control feedback loop, ha la capacità di rilevare dati rilevanti in base alla sua funzione, contribuendo alla raccolta di informazioni utili per il sistema.

**Precondizioni** -

### Scenario principale

1. Il Sensore inizia il rilevamento di eventi nel suo campo di interesse.
2. Il Sistema esegue una iterazione del MAPE control feedback loop notificando i suoi osservatori in base ai dati rilevati dal sensore.

*Ritorna al passo 2 creando il loop del MAPE control feedback loop.*

### 3.3.3 Gestione Automatica

**Attori**

- Sistema Automatico

**Descrizione** Il Sistema Automatico può attivare automaticamente un'azione su una componente del sistema basandosi sui dati attuali rilevati dai sensori, permettendo un controllo automatizzato delle operazioni.

**Precondizioni** -

**Scenario principale**

1. Il Sistema Automatico rimane in ascolto delle rilevazioni fatte da un sensore tramite un dispositivo di controllo.
2. Quando il Sensore rileva un nuovo evento, lo notifica al Sistema Automatico.
3. Il Sistema effettua delle operazioni sulla base dei dati rilevati dal sensore.

### 3.3.4 Gestisci Automatizzazione

**Attori**

- Residente

**Descrizione** Il Residente può attivare o disattivare il processo automatico di risposta a input esterni o a cambiamenti di stato dei sensori.

**Precondizioni** -

**Scenario principale**

1. Il Residente desidera attivare o disattivare l'automazione di un dispositivo di controllo della casa.
2. Il Residente interagisce con il dispositivo di controllo in questione.
3. Il Sistema attiva o disattiva l'automazione del dispositivo di controllo in questione.

### 3.3.5 Controllo Luci

**Attori**

- Residente
- Sistema Automatico

**Descrizione** Il Residente o il Sistema Automatico possono accendere o spegnere la/le luce/i di una stanza, il Sistema Automatico può farlo automaticamente in base alle condizioni rilevate dai sensori di rilevamento presenza.

### Precondizioni

- La stanza deve essere equipaggiata con almeno una fonte di illuminazione controllabile.

#### Scenario 1: Gestione manuale delle luci da parte del Residente

1. Il Residente desidera accendere o spegnere la/le luce/i nella stanza.
2. Il Residente interagisce con un dispositivo di controllo delle luci.
3. Il dispositivo di controllo delle luci riceve la richiesta.
4. Il Sistema accende o spegne la/le luce/i della stanza.

#### Scenario 2: Gestione automatica delle luci da parte del Sistema Automatico

1. Il Sistema Automatico rimane in ascolto delle rilevazioni fatte dal sensore di rilevamento presenza della stanza tramite un dispositivo di controllo.
2. Quando il Sensore rileva un nuovo evento, lo notifica al Sistema Automatico.
3. Se l'allarme della casa è disarmato e il controllo automatico delle luci della stanza è attivo:
  - 3.1. Se è stata rilevata la presenza di qualcuno.
    - 3.1.1. Il Sistema attiva le luci della stanza.
  - 3.2. Se non è stata rilevata nessuna presenza
    - 3.2.1. Il Sistema fa partire un timer temporale di una durata preimpostata.
    - 3.2.2. Terminato il timer, il Sistema spegne le luci della stanza.

### 3.3.6 Controllo Temperatura

#### Attori

- Residente
- Sistema Automatico

**Descrizione** Il Residente o il Sistema Automatico possono regolare manualmente la temperatura di una stanza, il Sistema Automatico può farlo automaticamente in base ai dati rilevati dai sensori di temperatura.

### Precondizioni

- La stanza deve essere equipaggiata con un climatizzatore controllabile.

#### **Scenario 1: Gestione manuale del climatizzatore da parte del Residente**

1. Il Residente desidera accendere o spegnere il climatizzatore nella stanza.
2. Il Residente interagisce con un dispositivo di controllo della temperatura.
3. Il dispositivo di controllo della temperatura riceve la richiesta.
4. Il Sistema accende o spegne il climatizzatore della stanza.

#### **Scenario 2: Gestione manuale della temperatura da parte del Residente**

1. Il Residente desidera aumentare o diminuire la temperatura della stanza.
2. Il Residente interagisce con un dispositivo di controllo della temperatura.
3. Il dispositivo di controllo della temperatura riceve la richiesta.
4. Se il climatizzatore della stanza è spento:
  - 4.1. Il Sistema accende il climatizzatore.
5. Il Sistema aumenta o diminuisce la temperatura del climatizzatore.

#### **Scenario 3: Gestione automatica della temperatura da parte del Sistema Automatico**

1. Il Sistema Automatico rimane in ascolto delle rilevazioni fatte dal sensore di temperatura della stanza tramite un dispositivo di controllo..
2. Quando il Sensore rileva un nuovo evento, lo notifica al Sistema Automatico.
3. Se il controllo automatico della temperatura della stanza è attivo:

- 3.1. Se la temperatura rilevata supera o scende al di sotto della temperatura ideale di un valore pari al threshold stabilito:
  - 3.1.1. Il Sistema accende il climatizzatore della stanza se non lo è già e imposta la temperatura al valore della temperatura ideale.
- 3.2. Se la temperatura rilevata è uguale alla temperatura ideale:
  - 3.2.1. Il Sistema spegne il climatizzatore della stanza se non lo è già.

### **3.3.7 Gestisci Preferenze Temperatura**

#### **Attori**

- Residente

**Descrizione** Il Residente può gestire le preferenze di temperatura di una stanza. Questo include la configurazione della temperatura ideale e della soglia di temperatura (threshold).

#### **Precondizioni** -

#### **Scenario principale**

1. Il Residente desidera modificare la temperatura ideale o la soglia di temperatura di una stanza.
2. Il Residente comunica al sistema le preferenze di temperatura da aggiornare.
3. Il Sistema aggiorna le preferenze di temperatura della stanza sulla base dei dati passati dal residente.

### **3.3.8 Controllo Pulizia**

#### **Attori**

- Residente

**Descrizione** Il Residente può avviare o interrompere manualmente il processo di pulizia della casa attraverso un folletto per la pulizia.

## Precondizioni

- La casa deve essere equipaggiata con un folletto per la pulizia controllabile.

### Scenario 1: Avvio della pulizia da parte del Residente

1. Il Residente desidera avviare il processo di pulizia della casa.
2. Il Residente interagisce con un dispositivo di controllo della pulizia.
3. Il dispositivo di controllo della pulizia riceve la richiesta.
4. Se il folletto per la pulizia è nello stato di ricarica:
  - 4.1. Il Sistema avvia il folletto per la pulizia.
  - 4.2. Il folletto per la pulizia inizia a pulire la casa, spostandosi da una stanza all'altra.
  - 4.3. Il folletto per la pulizia pulisce la stanza i cui si trova attualmente.
  - 4.4. Il folletto per la pulizia si sposta nella stanza successiva.  
*I passi 4.3. e 4.4. vengono ripetuti finché non vengono pulite tutte le stanze della casa.*
  - 4.5. Il folletto per la pulizia ritorna alla base di ricarica.
5. Se il folletto per la pulizia non è nello stato di ricarica:
  - 5.1. Il Sistema genera un'eccezione.

### Scenario 2: Interruzione della pulizia da parte del Residente

1. Il Residente desidera interrompere il processo di pulizia della casa.
2. Il Residente interagisce con un dispositivo di controllo della pulizia.
3. Il dispositivo di controllo della pulizia riceve la richiesta.
4. Se il folletto per la pulizia è nello stato di pulizia:
  - 4.1. Il folletto per la pulizia termina la pulizia della casa.
  - 4.2. Il folletto per la pulizia ritorna alla base di ricarica.
5. Se il folletto per la pulizia non è nello stato di pulizia:
  - 5.1. Il Sistema genera un'eccezione.

### **3.3.9 Controllo Protezione**

#### **Attori**

- Residente
- Sistema Automatico

**Descrizione** Il Residente o il Sistema Automatico possono gestire la sicurezza della casa, il Sistema Automatico può farlo automaticamente tramite i sensori rilevatori di presenza.

#### **Precondizioni**

- La casa deve essere equipaggiata con un sistema di allarme controllabile.
- Il Sistema deve prevedere una procedura di emergenza anti-intrusione.

#### **Scenario 1: Gestione manuale della protezione da parte del Residente**

1. Il Residente desidera armare o disarmare l'allarme della casa.
2. Il Residente interagisce con un dispositivo di controllo della protezione.
3. Il Residente comunica il pin di sicurezza.
4. Il dispositivo di controllo della protezione riceve la richiesta.
5. Il Sistema arma o disarma l'allarme della casa.
6. Se l'allarme viene disarmato e la sirena acustica della casa è attiva:
  - 6.1. Il Sistema disattiva la sirena acustica della casa.

#### **Scenario 2: Gestione automatica della protezione da parte del Sistema Automatico**

1. Il Sistema Automatico rimane in ascolto delle rilevazioni fatte dai sensori di rilevamento presenza delle stanze tramite dei dispositivi di controllo.
2. Quando un Sensore rileva un nuovo evento, lo notifica al Sistema Automatico.

3. Se l'allarme della casa è armato, il controllo automatico della protezione della stanza è attivo ed è stata rilevata la presenza di qualcuno:
  - 3.1. Il Sistema attiva la sirena acustica della casa.
  - 3.2. Il Sistema fa partire un timer temporale di una durata preimpostata.
  - 3.3. Terminato il timer, se l'allarme è ancora armato e non è stato disattivato da nessun Residente:
    - 3.3.1. Il Sistema esegue una procedura di emergenza anti-intrusione.

### **3.3.10 Chiamata di Emergenza**

#### **Attori**

- Sistema Automatico

**Descrizione** Il Sistema Automatico può effettuare automaticamente una chiamata di emergenza agli enti preposti in situazioni critiche.

#### **Precondizioni**

- Il Sistema deve prevedere un servizio apposito per effettuare chiamate di emergenza.

#### **Scenario principale**

1. Il Sistema Automatico desidera effettuare una chiamata di emergenza.
2. Il Sistema Automatico tramite un servizio apposito fa partire una chiamata di emergenza agli enti preposti.

### 3.4 Modello di dominio

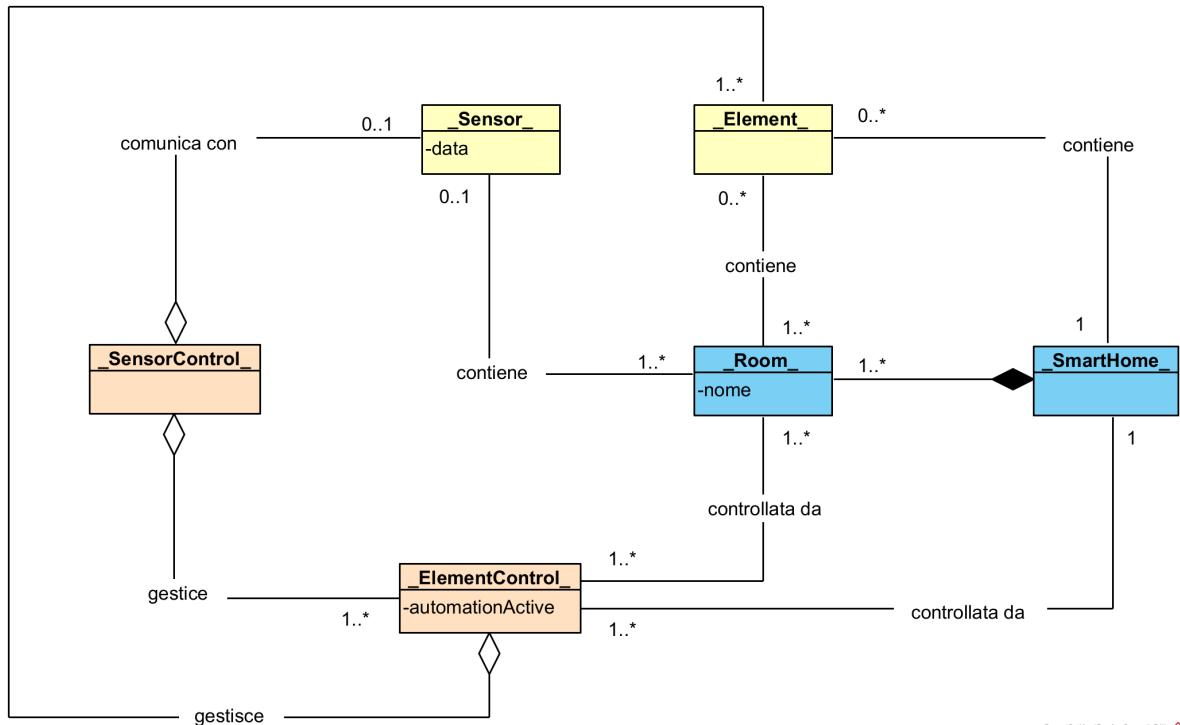


Figura 3: Modello di dominio

#### 3.4.1 Spiegazione delle varie componenti

**SmartHome** È la classe che rappresenta il sistema generale della Smart Home. È composta da più stanze **Room**, può contenere degli elementi generali della casa **Element**, come ad esempio un allarme o un folletto per la pulizia. Il comportamento degli elementi della casa sono controllati da dei controllori appositi **ElementControl**.

**Room** È la classe che rappresenta una generica stanza della casa. Può contenere degli elementi generali della stanza **Room** e dei sensori di vario genere **Sensor**. Il comportamento degli elementi della stanza sono controllati da dei controllori appositi **ElementControl**.

**Element** È un generico elemento presente in una specifica stanza **Room** o nella casa **SmartHome**.

**Sensor** È un generico sensore presente in una stanza **Room** che effettua dei rilevamenti in base al suo campo di interesse.

**ElementControl** È un controllore a cui è affidata la gestione di un determinato elemento **Element** presente in una specifica stanza **Room** o nella casa **SmartHome**.

**SensorControl** È un controllore a cui è affidata la gestione automatica del sistema in base ai rilevamenti che vengono effettuate dal sensore **Sensor** a cui è affidato appoggiandosi ai controllori degli elementi **ElementControl**.

## 3.5 Diagrammi di Sequenza di Sistema

### 3.5.1 Gestione Manuale

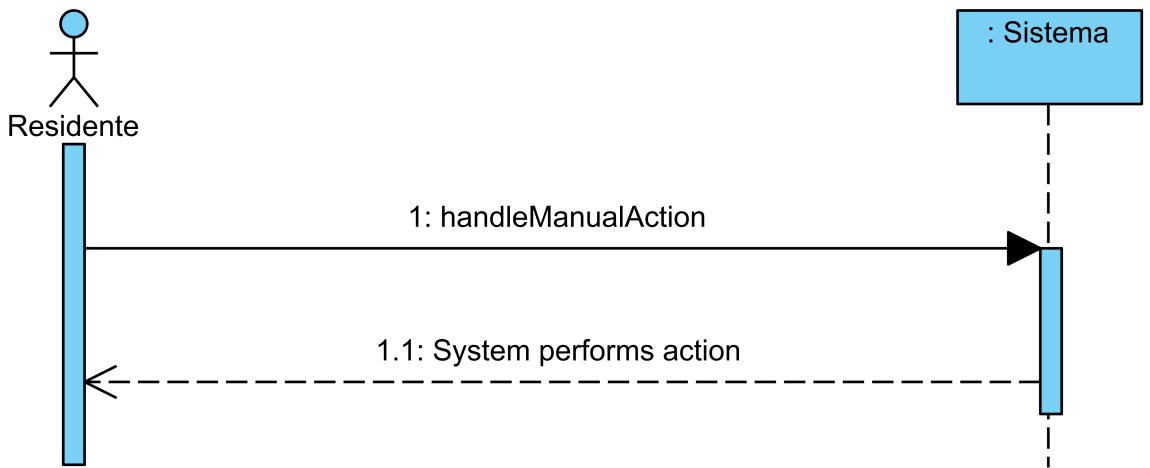


Figura 4: SSD Gestione Manuale

Diagramma di sequenza generale per la gestione manuale di una componente della casa da parte di un Residente. Il metodo `handleManualAction` è un metodo generico per riferirsi a un qualsiasi metodo per la gestione manuale di una determinata componente della casa.

### 3.5.2 Rileva Dati

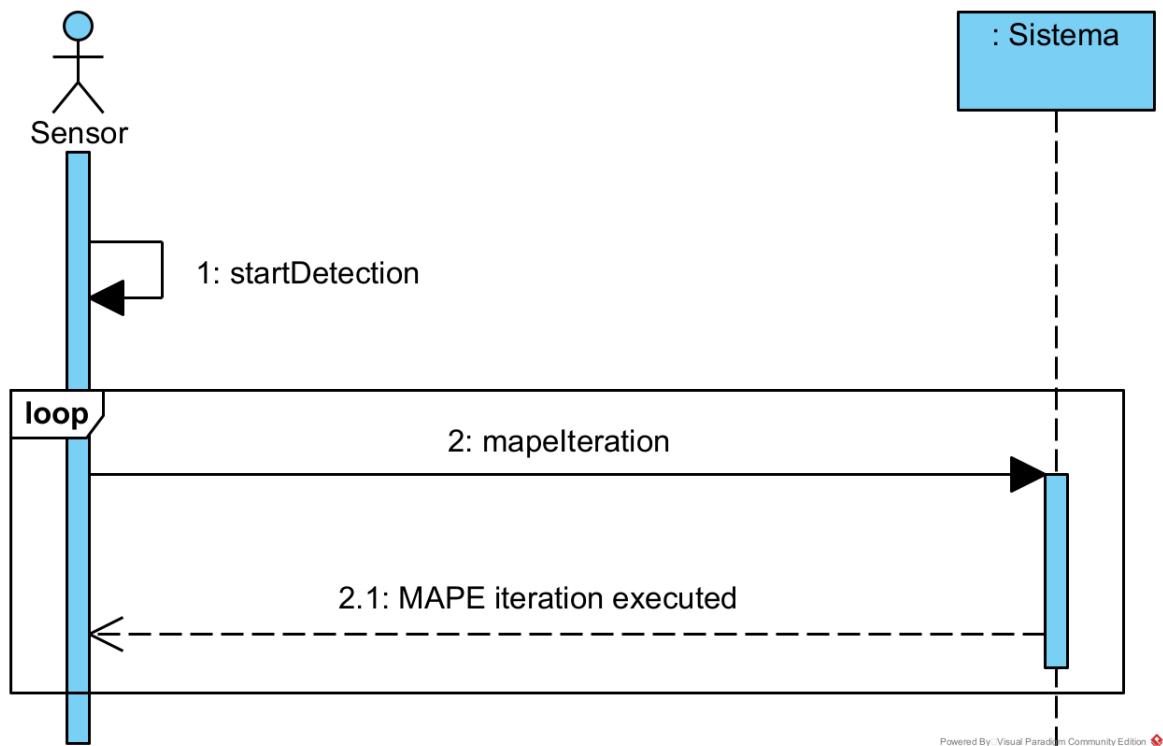


Figura 5: SSD Gestione Manuale

Diagramma di sequenza generale per la rilevazione dei dati da parte di un Sensore con l'utilizzo del **MAPE control feedback loop**.

### 3.5.3 Gestione Automatica

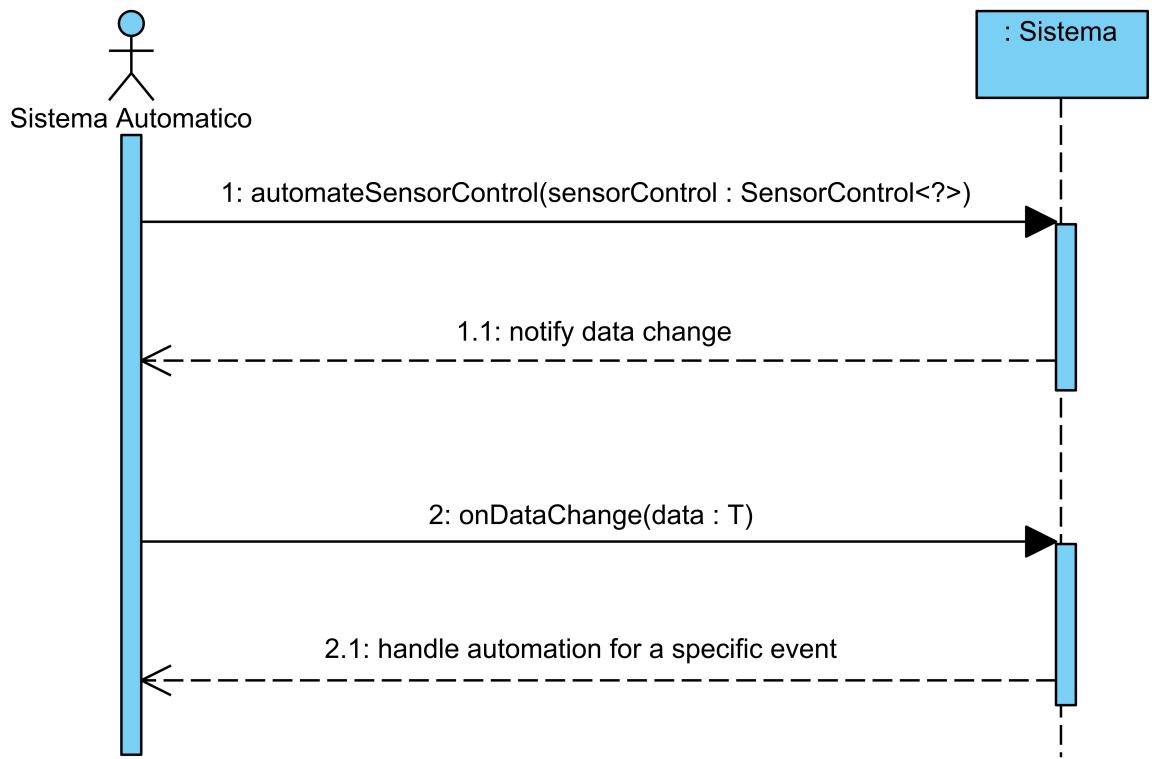


Figura 6: SSD Gestione Automatica

Diagramma di sequenza generale per la gestione automatica di una componente della casa da parte del Sistema Automatico. Il metodo `onDataChange` è un metodo generico per riferirsi a un metodo per la gestione automatica di un determinato evento rilevato e notificato da un sensore.

## 4 Progettazione

### 4.1 Diagrammi di Sequenza di Progettazione

#### 4.1.1 mapeIteration

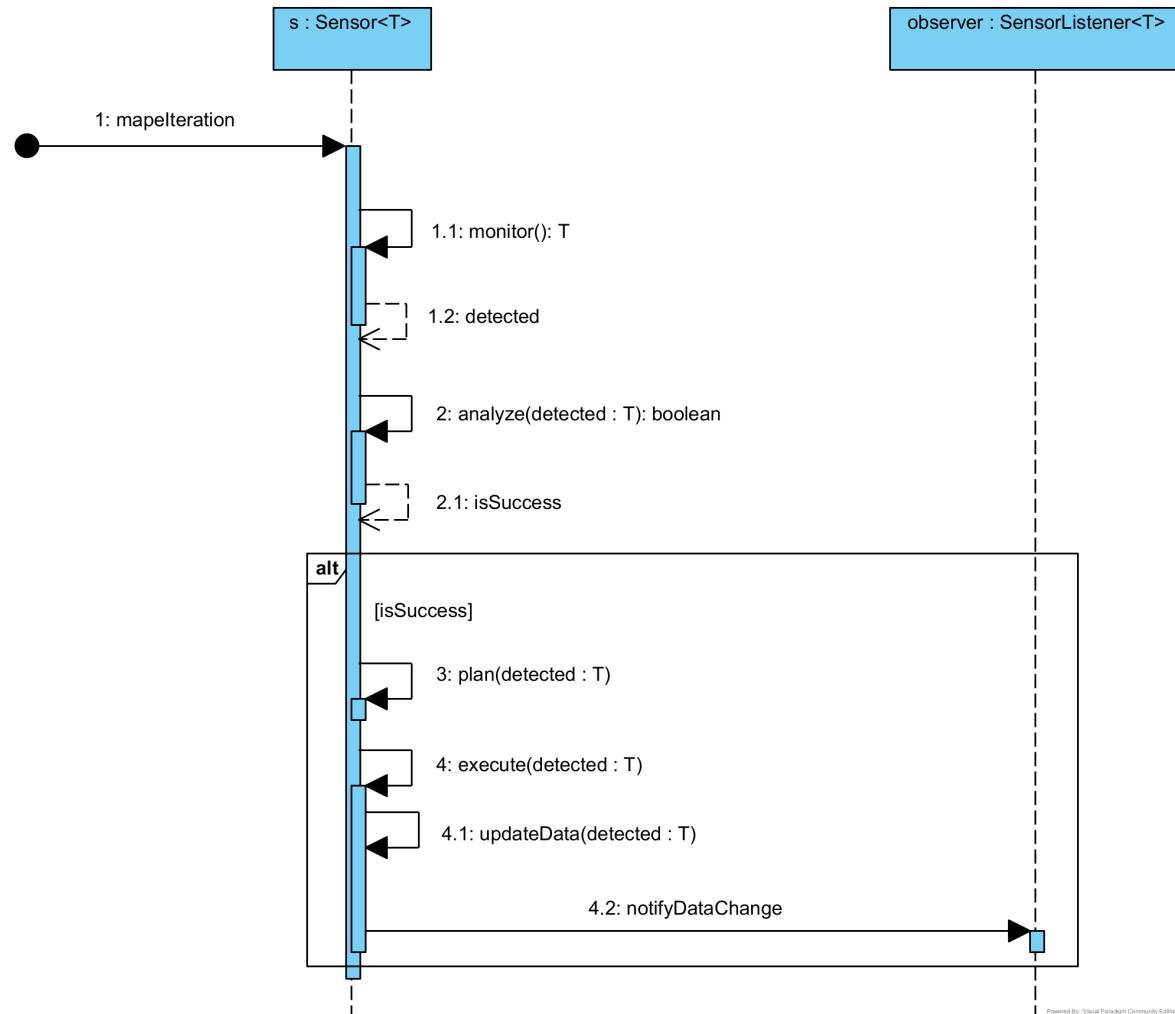


Figura 7: SD mapeIteration

Diagramma di sequenza di progettazione di una iterazione del **MAPE control feedback loop** eseguita da un Sensore.

#### 4.1.2 automateSensorControl

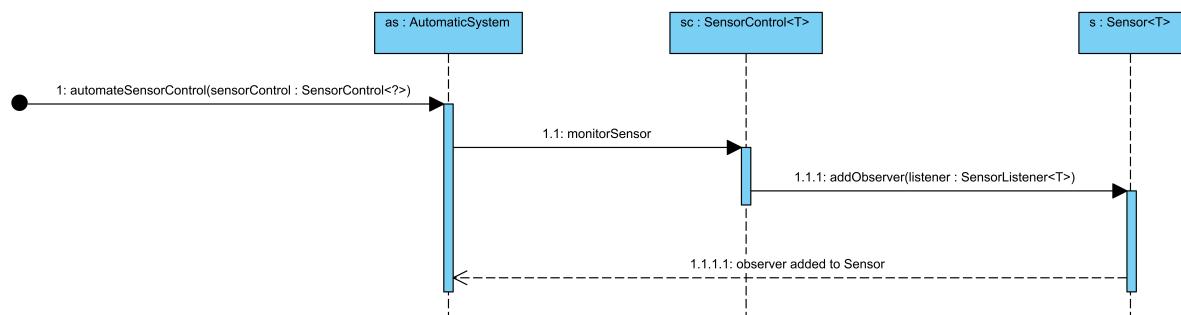


Figura 8: SD `automateSensorControl`

Diagramma di sequenza di progettazione è responsabile della configurazione e dell'avvio del monitoraggio di un sensore tramite controllore di competenza.

#### **4.1.3 onDataChange**

Questo metodo è ereditato in ogni controllore automatico che ha la responsabilità di gestire un determinato evento rilevato da un sensore di riferimento, in questo caso abbiamo due tipologie di eventi di rilevazione per i nostri sensori.

- Rilevazione di presenza
- Rilevazione della temperatura

Di seguito sono riportati i due metodi per la gestione di queste due tipologie di eventi da parte di un controllore automatico di competenza del sistema.

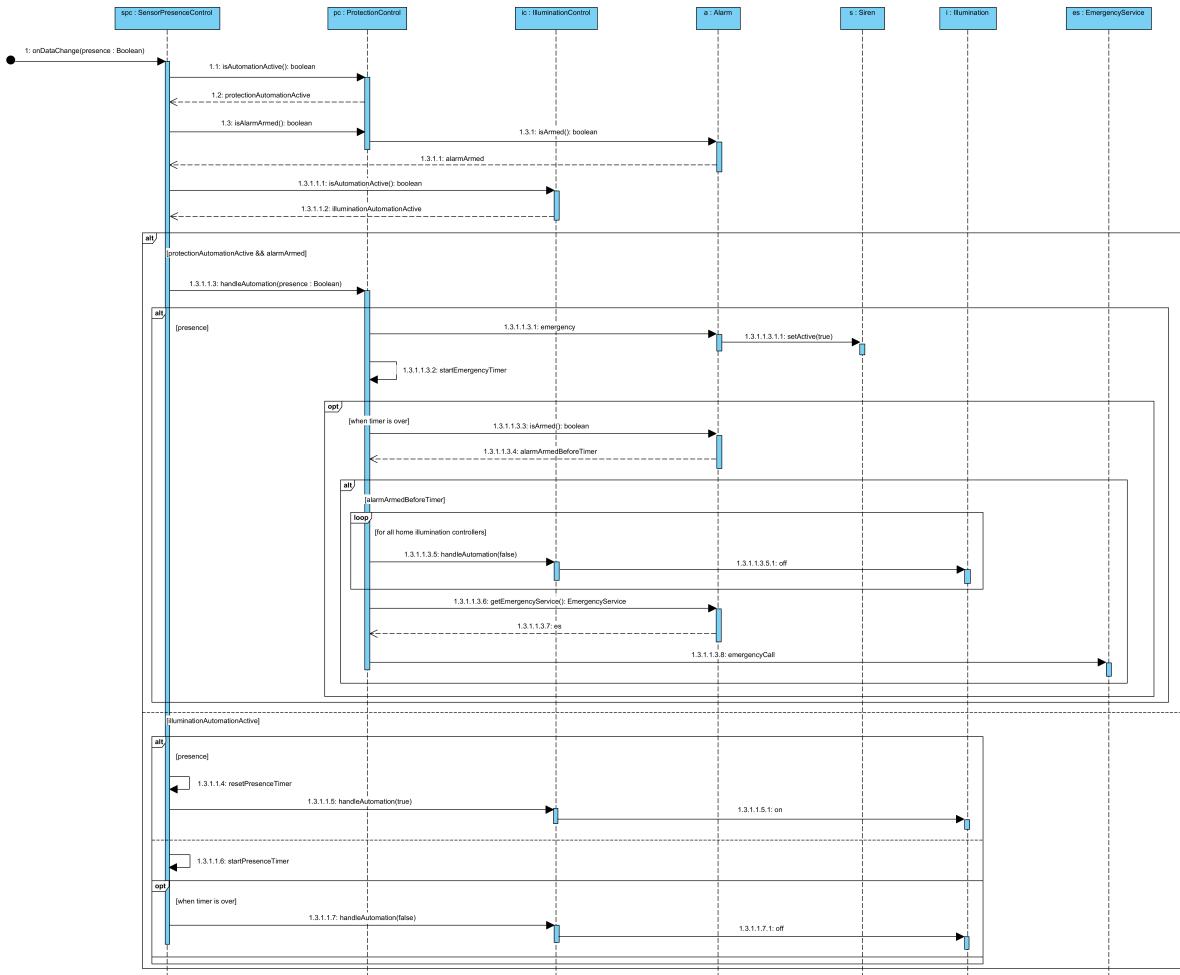


Figura 9: SD `onDataChange` per l'evento di rilevazione di presenza

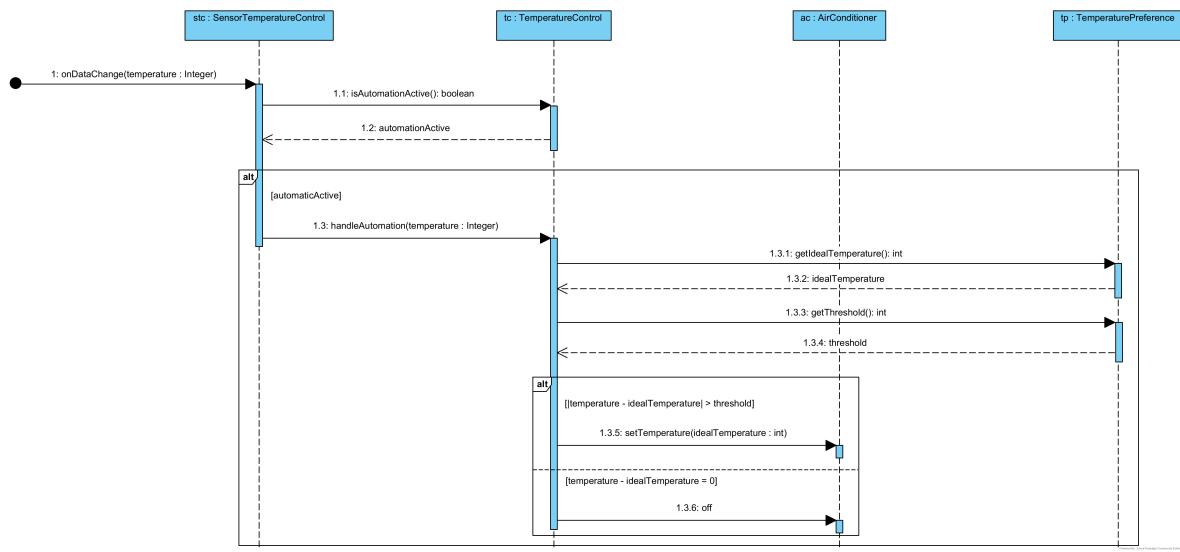


Figura 10: SD `onDataChange` per l'evento di rilevazione della temperatura

## 4.2 Diagramma degli Stati

### 4.2.1 Folletto per la Pulizia

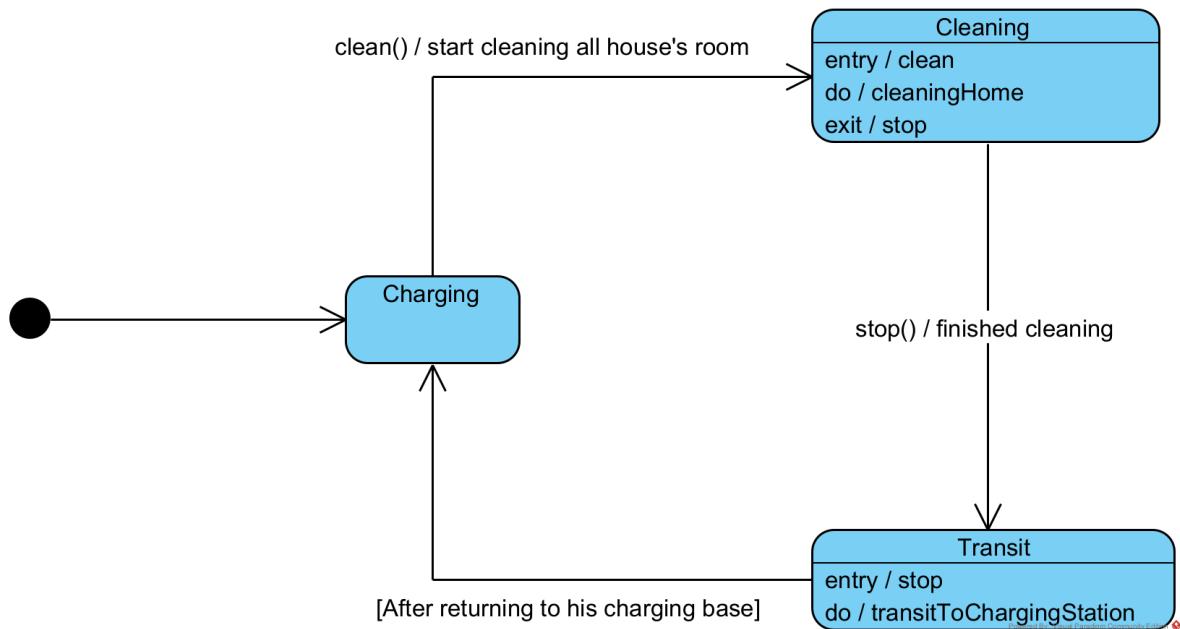


Figura 11: Diagramma degli stati del folletto per la pulizia.

Questo è il diagramma degli stati del folletto per la pulizia della casa, composto da 3 principali stati:

- **Charging**: il folletto è nella sua stazione di ricarica presente in una stanza della casa.
- **Cleaning**: il folletto sta pulendo la casa passando per tutte le stanze.
- **Transit**: il folletto sta transitando da una posizione qualsiasi della casa verso la sua postazione di ricarica.

## 4.3 Diagramma delle Attività

### 4.3.1 Controllo della Temperatura Automatico

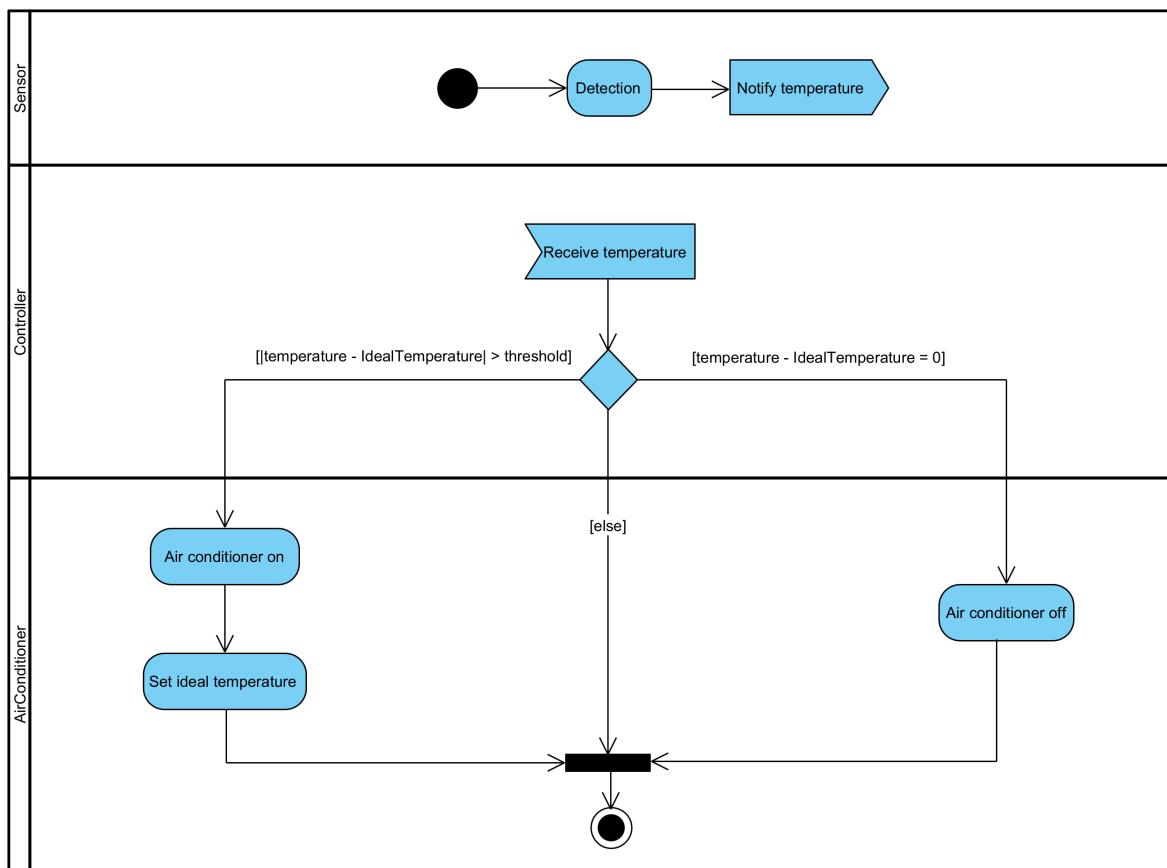


Figura 12: Diagramma delle attività per il controllo della temperatura automatico.

Questo diagramma delle attività descrive il comportamento del sistema per la regolazione automatica della temperatura di una stanza tramite un sensore per la temperatura, un controller e un climatizzatore.

## 4.4 Diagramma delle Classi Software e Diagramma dei Package

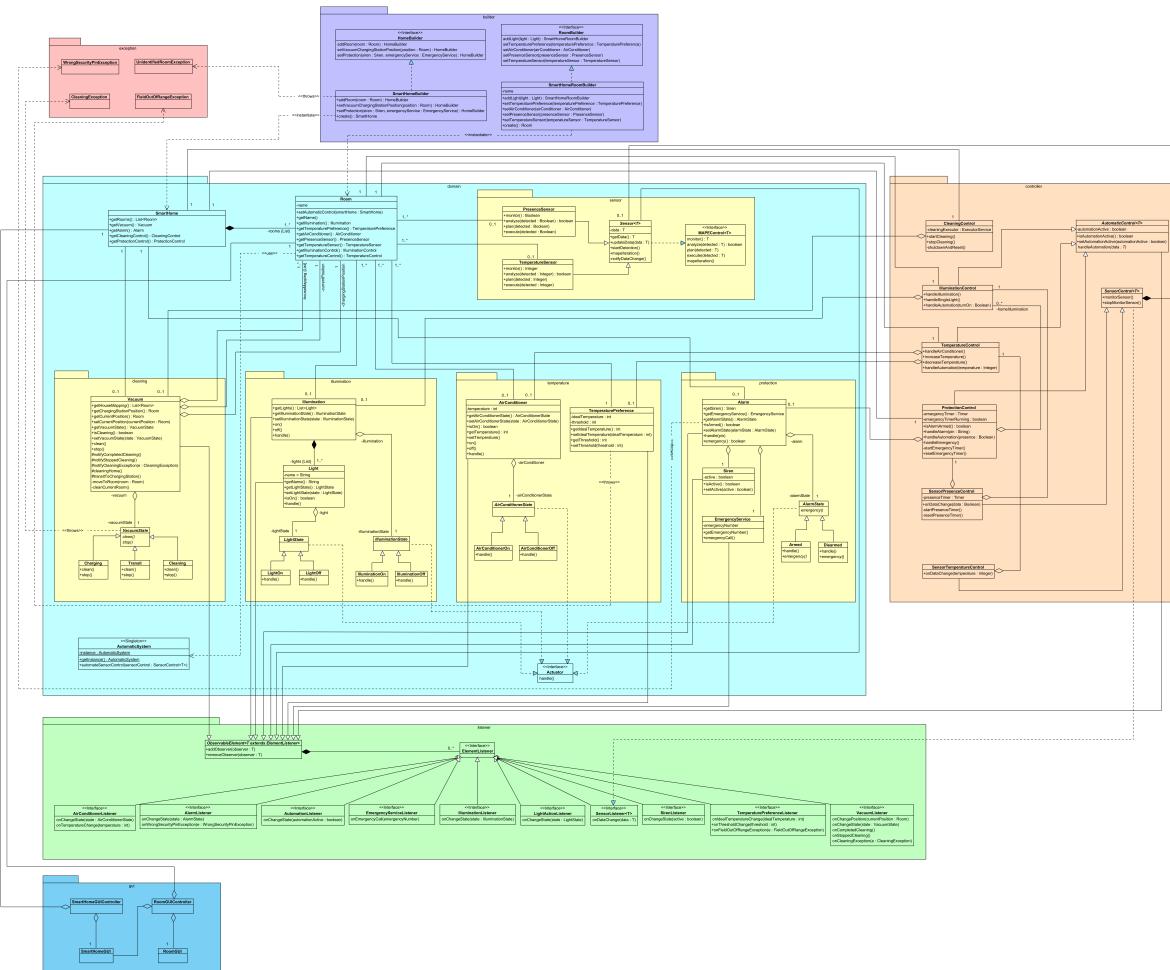


Figura 13: Diagramma delle Classi Software incorporato nel Diagramma dei Package

## 5 Simulazione dell'Ambiente

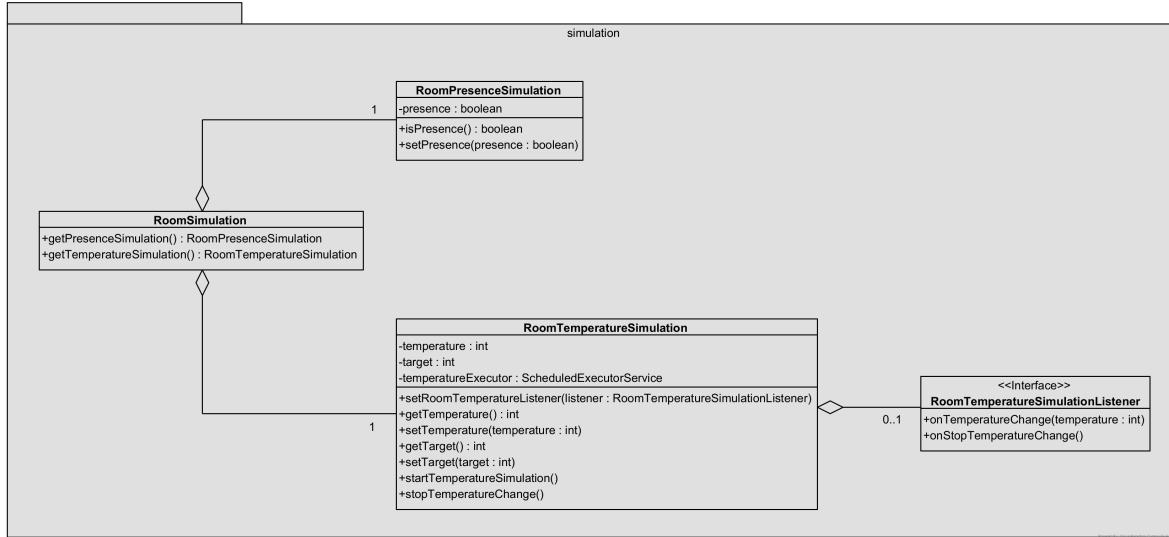


Figura 14: Modello delle classi software per la simulazione dell’ambiente

Questo set di classi è stato progettato per la simulazione del comportamento dell’ambiente all’interno della casa, hanno uno scopo esclusivamente simulativo e non sono destinate all’implementazione reale nel sistema di una casa fisica.

Sono utilizzate per comprendere il comportamento del sistema in risposta a varie condizioni ambientali e rappresentano uno strumento utile e fondamentale durante lo sviluppo e la fase di test del progetto.

### 5.1 Spiegazione delle varie componenti

**RoomSimulation** rappresenta la simulazione di una stanza all’interno di una casa. Contiene due sotto-componenti principali per la simulazione della presenza e della temperatura.

**RoomPresenceSimulation** simula la presenza o l’assenza di persone nella stanza.

**RoomTemperatureSimulation** simula la temperatura ambiente nella stanza. Utilizza un esecutore pianificato asincrono per simulare variazioni di temperatura nel tempo.

**RoomTemperatureSimulationListener** fornisce un’interfaccia per ricevere notifiche sui cambiamenti della temperatura, consentendo la gestione di eventi correlati alla simulazione.

## 6 GUI

Per poter testare e provare le funzionalità implementate per il sistema di Smart Home è stata creata una GUI dove è possibile gestire in modo simulativo il comportamento di una Smart Home predefinita tramite tutte le funzionalità offerte dal sistema.

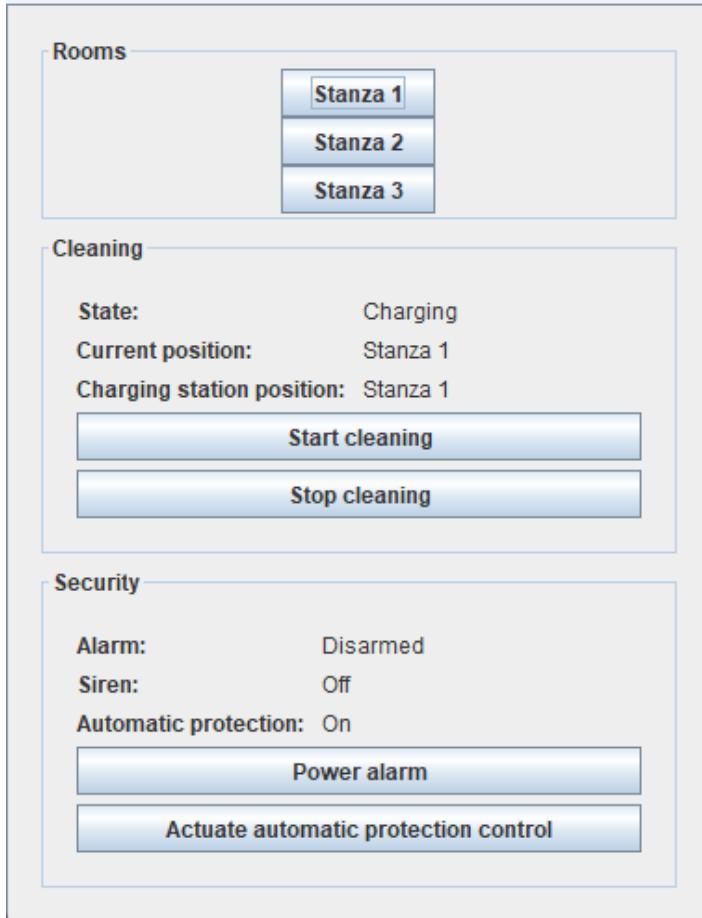


Figura 15: Schermata principale della Smart Home

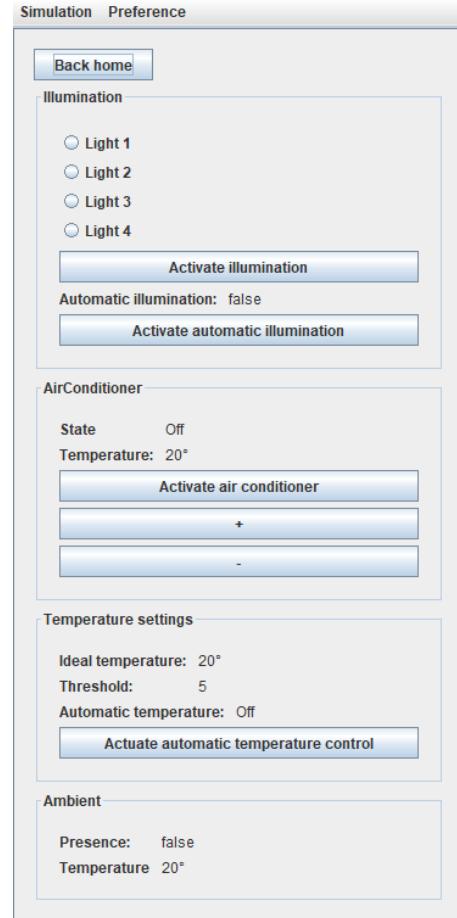


Figura 16: Schermata di una stanza della Smart Home

## 7 Conclusioni

### 7.1 Gestione del Ciclo di Sviluppo del Software

Nel corso del progetto, ci siamo organizzati in modo flessibile e collaborativo, basato su una metodologia di sviluppo iterativa, integrando i principi della metodologia Agile.

#### 7.1.1 Metodologia Iterativa e Agile

Abbiamo implementato un approccio iterativo nello sviluppo del software, suddividendo il lavoro in iterazioni affrontando ciascuna fase incrementalmente, integrando i principi della metodologia Agile, utilizzando sprints, durante il quale un team lavorava su una quantità specifica di lavoro, con una pianificazione iterativa.

#### 7.1.2 Divisione dei Compiti e Collaborazione

Per lavorare in modo più efficiente, abbiamo diviso i compiti tra noi dopo aver fatto un'analisi iniziale del progetto insieme. La collaborazione è stata effettuata attraverso chiamate regolari su Discord, per un monitoraggio costante del lavoro svolto, per la risoluzione di problemi e per la divisione di ulteriori compiti.

#### 7.1.3 Strumenti e Tecnologie

Per la gestione del progetto, abbiamo adottato un set di strumenti e tecnologie essenziali. Ecco un breve elenco di quelli principali:

- **GitHub con GitActions:** Abbiamo sfruttato GitHub insieme a GitActions per automatizzare l'analisi della build del progetto.
- **IntelliJ:** Abbiamo utilizzato IntelliJ come ambiente di sviluppo principale. Per la collaborazione in tempo reale, abbiamo sfruttato la funzione "Code with Me" di IntelliJ che è stata utilizzata per la collaborazione in tempo reale, consentendo di lavorare simultaneamente sullo stesso progetto, facilitando la risoluzione collaborativa di problemi.
- **SonarQube e SonarCloud tramite GitHub Actions:** Per il controllo della qualità del codice, abbiamo integrato SonarQube e SonarCloud tramite GitHub Actions. Questa integrazione ha automatizzato l'analisi del codice.
- **Understand:** Abbiamo utilizzato il tool Understand per individuare eventuali antipattern strutturali.

- **Discord:** Discord è stato utilizzato come piattaforma di comunicazione principale.
- **LaTeX con documentazione condivisa:** Per la documentazione del progetto, abbiamo utilizzato LaTeX con documentazione condivisa.

#### 7.1.4 Revisione del Codice, Test e Validazione

Abbiamo implementato sessioni regolari di revisione del codice, assieme all'implementazione di test unitari durante lo sviluppo per garantire una buona qualità del software.

## 7.2 Design Pattern

### 7.2.1 Builder

Il Builder Pattern è utilizzato in `SmartHomeBuilder` e `SmartHomeRoomBuilder` per la costruzione degli oggetti `SmartHome` e `Room` rispettivamente. Questo pattern consente la costruzione step-by-step di oggetti complessi.

### 7.2.2 Observer

L'Observer Pattern è implementato nelle classi che estendono `ObservableElement` per gestire la comunicazione tra oggetti osservabili e osservatori, utilizzando le classi che estendono `ElementListener` per la comunicazione dei cambiamenti agli osservatori.

### 7.2.3 Strategy

Lo Strategy Pattern è utilizzato nei controllori automatici che estendono `AutomaticControl`. Ogni controller implementa la strategia specifica attraverso il metodo `handleAutomation`.

### 7.2.4 Template Method

Il Template Method Pattern è utilizzato nei sensori per definire la struttura generale del MAPE control feedback loop. L'interfaccia `MAPEControl` definisce i metodi necessari, mentre le classi concrete come `PresenceSensor` e `TemperatureSensor` implementano i dettagli di ciascuna fase del ciclo: `monitor`, `analyze`, `plan` ed `execute`.

### 7.2.5 State

Lo State Pattern è utilizzato nelle classi che possono assumere diversi stati durante il loro ciclo di vita, come ad esempio `Vacuum`, `Light`, `Alarm`, `AirConditioner` ecc., comportandosi diversamente rispetto ad una determinata azione in base allo stato in quel momento.

### 7.2.6 Singleton

Il Singleton Pattern è utilizzato in `AutomaticSystem` per garantire l'esistenza di una singola istanza della classe.

## 7.3 Pattern Architetturali

### 7.3.1 Page Controller

`RoomGUIController` e `SmartHomeGUIController` agiscono come controller per le rispettive pagine `RoomGUI` e `SmartHomeGUI`. Gestiscono gli eventi degli elementi dell'interfaccia utente e gestiscono il flusso di controllo della pagina associata.

## 7.4 Design Principle

### 7.4.1 Single Responsibility

Il principio di singola responsabilità afferma che una classe dovrebbe avere una singola responsabilità. Questo principio è stato adottato nella maggior parte delle classi.

### 7.4.2 DRY - Don't Repeat Yourself

Questo principio indica che si dovrebbero evitare ripetizioni nelle informazioni. Questo principio è stato osservato nella totalità del codice del progetto.

### 7.4.3 Separation of Concerns

Questo principio afferma che un sistema dovrebbe essere suddiviso in componenti separati, ognuno dei quali si occupa di una singola preoccupazione o aspetto del sistema. Nel contesto del progetto abbiamo applicato questo principio soprattutto nei seguenti aspetti:

- **Controller:** Le classi controller, come `CleaningControl`, `ProtectionControl`, `TemperatureControl` e altri, gestiscono specifiche aree di funzionalità, mantenendo una separazione chiara tra le responsabilità.
- **Classi di Dominio:** Le classi come `SmartHome`, `Room`, `Vacuum`, `Alarm`, `Illumination`, `Light`, `PresenceSensor`, `TemperatureSensor`, e altre, si concentrano su aspetti specifici e separati del dominio.

### 7.4.4 Open Closure

Questo principio sottolinea che le classi dovrebbero essere aperte all'estensione ma chiuse alla modifica. Nel nostro codice è stato adottato questo principio per i sensori (`TemperatureSensor` e `PresenceSensor`), infatti queste due classi estendono la classe generale `Sensor<T>` che implementa già la logica generale del MAPE feedback control loop. Per creare un nuovo sensore infatti basta creare una classe che estenda `Sensor<T>` e che implementi solamente la logica dei singoli passaggi di MAPE.

### 7.4.5 Common Closure

Questo principio indica che le classi che cambiano insieme dovrebbero essere raggruppate insieme. Questo principio è stato adottato per tutte le classi del nostro progetto inerenti ad una specifica area di interesse.

#### 7.4.6 Common Reuse

Questo principio indica che le classi che non sono riutilizzate insieme non dovrebbero essere raggruppate insieme. Questo principio, come in Common Closure, è stato adottato per tutte le classi del nostro progetto inerenti ad una specifica area di interesse.

#### 7.4.7 Stable Abstraction

Questo principio indica che le astrazioni, come interfacce e classi base, dovrebbero essere stabili nel tempo e meno soggette a modifiche frequenti. Nel contesto del progetto abbiamo applicato questo principio soprattutto nei seguenti aspetti:

- **Utilizzo di Interfacce:** Le interfacce vengono impiegate per definire astrazioni stabili, permettendo l'estensione del sistema senza modificare le interfacce esistenti.
- **Dipendenza da Astrazioni:** Le classi dipendono da astrazioni piuttosto che da implementazioni concrete, aumentando la stabilità in caso di modifiche alle implementazioni.
- **Riutilizzo delle Astrazioni:** Le astrazioni sono progettate per essere riutilizzabili in contesti diversi.

## 7.5 Understand

Abbiamo utilizzato il tool Understand per rilevare la presenza di possibili antipattern strutturali.

### 7.5.1 Controller Package

Dall'analisi delle dipendenze tra i package del progetto, abbiamo riscontrato una forte dipendenza tra i package **controller** e **domain**. Questa struttura è data dal fatto che i controller hanno la responsabilità di controllare e gestire il comportamento degli elementi del dominio nella loro totalità, questa tipologia di struttura genera un **External Butterfly**, non è necessariamente un problema, ma se si dovesse modificare un suo componente si avrà un impatto globale significativo.

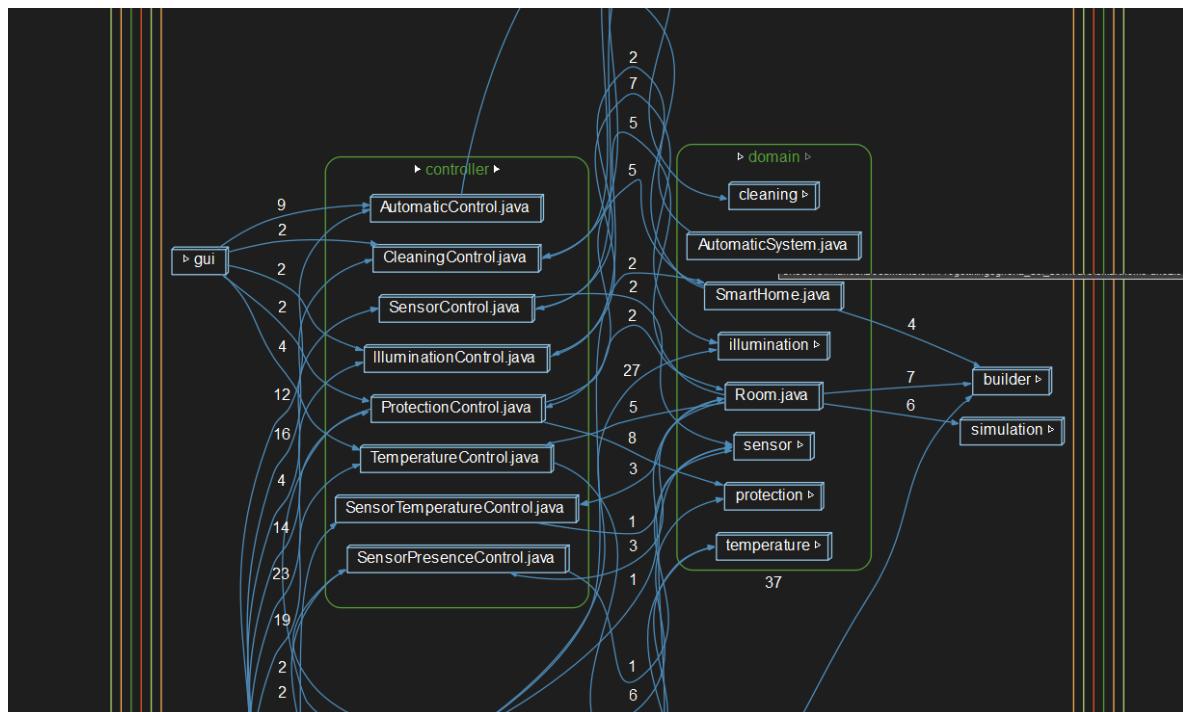


Figura 17: Grafico delle dipendenze tra i package **controller** e **domain**

### 7.5.2 ObservableElement

Dall'analisi del grafico butterfly della classe `ObservableElement` abbiamo notato molte dipendenze verso la medesima classe da parte di altri elementi del sistema. Questo è dato dal fatto che ogni elemento osservabile nel sistema estende la classe `ObservableElement` che implementa le funzioni necessarie per implementare il design pattern Observer, questa tipologia di struttura genera un **Global Butterfly**, non è necessariamente un problema, ma se si dovesse modificare un suo componente si avrà un impatto globale significativo.

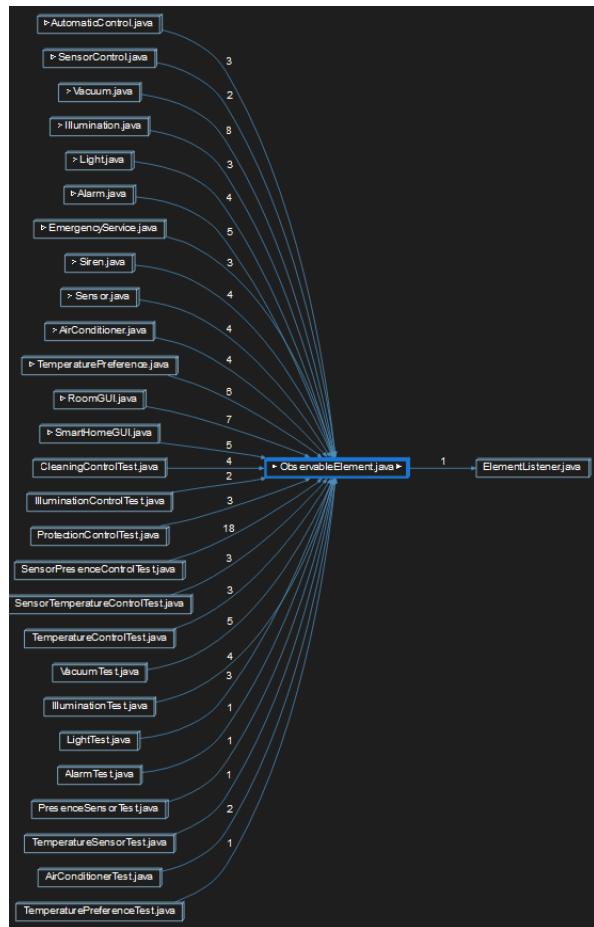


Figura 18: Grafico butterfly per la classe `ObservableElement`

## 7.6 Sviluppi Futuri

Si potrebbero considerare sviluppi futuri che includano una comunicazione client-server tramite il protocollo TCP/IP con l'utilizzando delle socket.

### 7.6.1 Comunicazione Client-Server

Integrare una comunicazione client-server consentirebbe l'interazione con il sistema da dispositivi esterni che fungeranno da client, come ad esempio il dispositivo mobile del residente. Il server, posizionato all'interno dell'abitazione e connesso alla rete domestica, rimarrà in ascolto, su una porta logica conosciuta dai client, delle richieste inviate ad esso per eseguire operazioni specifiche nel sistema Smart Home.

### 7.6.2 Sistema di Messaggistica Client-Server

Implementare un sistema di messaggistica consentirebbe ai client di interagire direttamente con il server. I messaggi predefiniti potrebbero indicare azioni specifiche da eseguire, semplificando l'interazione con il sistema. Ad esempio, un messaggio inviato al server potrebbe essere interpretato come una richiesta per attivare un'illuminazione o avviare un processo di pulizia.

### 7.6.3 Protocollo TCP/IP

L'utilizzo del protocollo TCP/IP e delle socket permetterà una connessione affidabile tra il client e il server. I messaggi inviati tramite questo protocollo saranno gestiti dal server per eseguire azioni specifiche nel sistema. Ad esempio, il client potrebbe inviare messaggi predefiniti come:

- `[room name].illumination.handle`: Per gestire l'illuminazione di una specifica stanza.
- `[room name].temperature.airConditioner.setTemperature.[val]`: Per impostare la temperatura del climatizzatore di una specifica stanza al valore `[val]`
- `[room name].temperature.automation.on`: Per attivare il controllo automatico della temperatura
- `cleaning.start`: Per avviare il processo di pulizia della casa.