Leafy

Gabriele Basigli 0000977227
Tommaso Faggiano 0000975928
Muhamad H. Salman 0000969902

Settembre 2023

Contents

Abstract

Il progetto realizza un software in grado di gestire una serra automaticamente, tramite il monitoraggio preciso di diversi parametri vitali per le piante, come luce, temperatura, umidità dell'aria e del terreno, al fine di ottimizzare la crescita e la salute delle stesse.

Il software è dotato di un sistema di controllo che permette di regolare manualmente la gestione dei parametri, oppure di utilizzare dei set preimpostati in base al tipo di pianta selezionata. L'applicativo è inoltre in grado di generare dei resoconti che descrivono dettagliatamente lo stadio di crescita della pianta, eventualmente con annesse analisi statistiche al fine di migliorare nel tempo l'automatizzazione e la gestione.

Il sistema interagisce con la serra tramite un controllore, ha diversi sensori fisici per misurare le caratteristiche ambientali interne e dispone di vari attuatori che permettono di regolarle a piacimento.

1 Analisi dei requisiti

1.1 Requisiti del sistema

1.1.1 Raccolta dei requisiti

- La serra deve misurare, salvare ed elaborare i parametri di luce, temperatura e umidità del terreno.
- La serra deve gestire gli strumenti in base al tipo di pianta selezionata ed al suo stadio di crescita (scelta utente del preset).
- La serra deve attivare irrigatore, lampada UV e ventola quando necessario e nella misura corretta, in base ai parametri registrati.
- La serra deve tener traccia di data ed ora di ogni accensione di ogni strumento (resoconto).
- La serra deve tener traccia della durata di ogni accensione di ogni strumento (resoconto).
- Al termine di ogni giornata la serra deve mostrare un resoconto degli strumenti che si sono attivati ed il tempo complessivo per ogni strumento.
- Al termine di ogni settimana la serra deve produrre un resoconto dei parametri (mostrare che sono stati mantenuti stabili ai valori ottimali).
- L'amministratore deve poter aggiungere, bloccare e rimuovere utenti.
- L'amministratore deve poter assegnare e rimuovere i permessi di accesso ed utilizzo della serra.
- L'utente autorizzato può attivare direttamente gli attuatori.
- L'utente autorizzato può modificare il pre-set di controllo.
- L'utente autorizzato può creare un pre-set di controllo per una pianta non registrata.

1.1.2 Tabella dei requisiti

Table 1: Tabella dei requisiti

ID	Requisito	Tipo
R1F	Misurazione, salvataggio ed elaborazione dei vari parametri	Funzionale
R2F	Gestione degli strumenti in base al tipo di pianta selezionata ed al suo stadio di crescita (scelta utente del preset).	Funzionale
R3F	Attivazione di irrigatore, lampada UV e ventola quando necessario e nella misura corretta, in base ai parametri registrati.	Funzionale
R4F	Tracciamento di data ed ora di ogni accensione di ogni strumento (necessario per resoconto).	Funzionale
R5F	Tracciamento della durata di ogni accensione di ogni strumento (necessario per resoconto).	Funzionale
R6F	Resocontazione giornaliera degli strumenti che si sono attivati e dell tempo complessivo per ogni strumento.	Funzionale
R7F	Resocontazione settimanale dei parametri (mostrare che sono stati mantenuti stabili ai valori ottimali).	Funzionale
R8F	L'amministratore deve poter aggiungere, bloccare e rimuovere utenti.	Funzionale
R9F	L'amministratore deve poter assegnare e rimuovere i permessi di accesso ed utilizzo della serra.	Funzionale
R10F	L'utente autorizzato può attivare direttamente gli attuatori.	Funzionale
R11F	L'utente autorizzato può modificare il preset di controllo.	Funzionale
R12F	L'utente autorizzato può creare un preset di controllo per una pianta.	Funzionale
R13NF	Semplicità di utilizzo (navigabilità)	Non Funzionale

Table 1

ID	Requisito	Tipo
R14NF	Velocità di memorizzazione	Non Funzionale
R15NF	Velocità nel recupero dei dati	Non Funzionale
R16NF	Protezione dei dati sensibili	Non Funzionale
R17NF	Efficienza Energetica	Non Funzionale
R18NF	Controllo degli accessi	Non Funzionale
R19NF	Controllo dei permessi	Non Funzionale
R20NF	Precisione nel controllo degli attuatori	Non Funzionale

1.2 Analisi del Dominio

1.2.1 Vocabolario

Vengono qui definiti dei vocaboli che descrivono diversi aspetti del sistema.

Table 2: Vocabolario

Voce	Descrizione	Sinonimi
Parametro	Misura delle condizioni fisiche della pianta o delle condizioni ambientali della serra	Indice
Attuatore	Dispositivo necessario a regolare i parametri vitali della pianta (irrigatore, lampa UV, ventola).	Strumentazione, strumenti
Condizione ambientale	Stato climatico interno alla serra in un certo istante di tempo, ovvero i valori quantitativi di luce, umidità del terreno e dell'aria, e temperatura.	Stato
Luce	Quantità di luce solare a cui è esposta la serra.	
Umidità terreno	Livello di umidità del terreno della pianta.	
Umidità aria	Livello di umidità dell'aria interna della serra.	
Temperatura	Temperatura interna della serra.	
Soglia	Valore di un parametro oltre il quale è necessario attivare/disattivare l'attuatore corrispondente.	Limite
Preset	Insieme di soglie e di regole che, asssociate ad una specifica pianta, ne regolano il comportamento (controllando gli attuatori)	Set
Permesso	Permessi degli utenti ad accedere al sistema, visualizzare i dati, modificare il pre-set o agire sugli attuatori.	Privilegi, Abilitazione
Blocco	1. blocco dell'account utente 2. blocco del sistema per motivi di sicurezza	1. sospensione 2. stato di sicurezza

Table 2

Voce	Descrizione	Sinonimi
Resoconto	Documento salvato sul sistema che riassume l'attività delle serra in un certo lasso di tempo, comprendente i grafici	Report

1.3 Analisi dei Requisiti

1.3.1 Diagramma dei casi d'uso

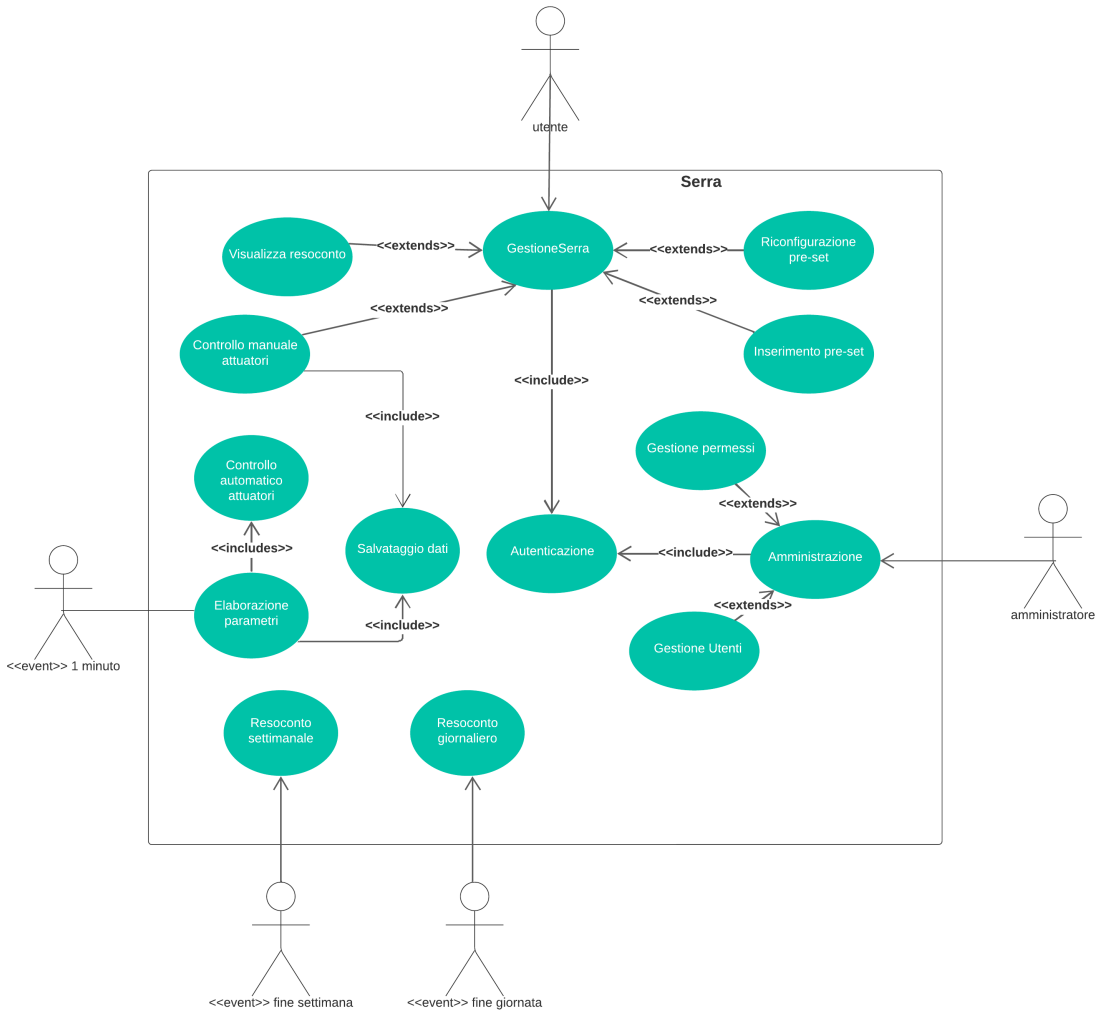


Figure 1: Diagramma dei casi d'uso

Nel diagramma non vengono riportati come attori i sensori e gli attuatori poichè essi hanno un ruolo passivo, in quanto le fasi di elaborazione dei parametri e successivo controllo della serra vengono scatenate da eventi generati da dei timer Ad esempio, si può supporre che il controllo della serra debba avvenire con granularità di 1 minuto. In tale caso ogni minuto si provvederà ad effettuare il controllo.
Eventi quali il termine di una settimana o di una giornata sono attori che scatenano la generazione di resoconti (rispettivamente settimanali e giornalieri).

Un aspetto importante risiede nella relazione tra ElaborazioneParametri e ControlloAutomaticoAttuatori. Difatti l'elaborazione dei parametri può spesso risultare in un solo salvataggio dei dati ricevuti dai sensori senza dover attivare alcun attuatore, se i valori rilevati sono consistenti con lo stato che si vuole mantenere. Inoltre, ElaborazioneParametri invocherà SalvataggioDati una prima volta per salvare i parametri letti, ed una seconda volta nel caso fosse richiesta l'attivazione degli attuatori.

1.3.2 Scenari

Table 3: GestioneSerra

Titolo	GestioneSerra
Descrizione	Offre all'utente autenticato la possibilità di selezionare un'operazione
Attori	Utente
Relazioni	VisualizzaResoconto, ControlloManualeAttuatori, InserimentoPreset, RiconfigurazionePreset, Autenticazione
Precondizioni	
Postcondizioni	
Scenario principale	1. Autenticazione (inclusione) 2. Visualizzare operazioni possibili 3. Reindirizzare l'utente all'operazione scelta
Scenari alternativi	1. L'utente non ha i permessi di accedere alle operazioni di controllo della serra. 2. L'utente non viene autenticato perchè non possiede un account 3. L'utente non viene autenticato perchè il suo account è stato sospeso
Requisiti non funzionali	Non deve succedere che un utente non autenticato o privo dei permessi acceda al sistema ed esegua delle operazioni
Punti aperti	

Table 4: VisualizzaResoconto

Titolo	VisualizzaResoconto
Descrizione	L'utente sceglie quale resoconto visualizzare
Attori	Utente
Relazioni	GestioneSerra
Precondizioni	Utente autenticato, resoconto già prodotto
Postcondizioni	
Scenario principale	1. Mostrare tutti i resoconti disponibili 2. Offrire la possibilità di filtrare i resoconti per data e tipologia 3. Visualizzare il resoconto selezionato dall'utente
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Table 5: ControlloManualeAttuatori

Titolo	ControlloManualeAttuatori
Descrizione	Il software permette di attivare o disattivare direttamente gli attuatori
Attori	Utente
Relazioni	GestioneSerra, SalvataggioDati
Precondizioni	Utente autenticato, Utente abilitato
Postcondizioni	Scheduling dell'operazione richiesta dall'utente
Scenario principale	1. Mostrare la lista degli attuatori disponibili ed il loro stato (acceso o spento) 2. Operazioni dell'utente sullo stato degli attuatori 3. Salvataggio dati per resoconto (inclusione)
Scenari alternativi	1. L'attuatore non è raggiungibile, segnalato errore
Requisiti non funzionali	
Punti aperti	

Table 6: RiconfigurazionePreset

Titolo	RiconfigurazionePreset
Descrizione	Riconfigurazione dei valori di soglia del preset per ogni attuatore
Attori	Utente
Relazioni	GestioneSerra
Precondizioni	Utente autenticato, Utente abilitato
Postcondizioni	Preset aggiornato sostituito al precedente
Scenario principale	1. Mostrare tutte le soglie del preset corrente utilizzate dal sistema. 2. Permettere all'utente di modificare il set dei valori di soglia per ogni attuatore. 3. Adottare la nuova configurazione di controllo della serra
Scenari alternativi	1. L'utente non inserisce dei valori validi, mostrato messaggio d'errore
Requisiti non funzionali	
Punti aperti	

Table 7: InserimentoPreset

Titolo	InserimentoPreset
Descrizione	Scelta di un preset per la pianta attuale o creazione di un nuovo prest
Attori	Utente
Relazioni	GestioneSerra
Precondizioni	Utente autenticato, Utente abilitato
Postcondizioni	Registrato un nuovo preset tra quelli esistenti
Scenario principale	1. Inserimento del nome del preset e pianta associata 2. Inserimento dei valori di soglia del nuovo preset per ogni attuatore 3. Salvataggio del preset tra quelli disponibili 4. Scelta dell'utente del preset da adottare tra quelli disponibili
Scenari alternativi	1. L'utente non vuole inserire un nuovo preset ma solo sceglierne un altro
Requisiti non funzionali	
Punti aperti	

Table 8: Amministrazione

Titolo	Amministrazione
Descrizione	Offre all'amministratore la possibilità di selezionare un'operazione
Attori	Amministratore
Relazioni	GestionePermessi, GestioneUtenti, Autenticazione
Precondizioni	
Postcondizioni	
Scenario principale	1. Autenticazione (inclusione) 2. Visualizzazione operazioni possibili 3. Reindirizzamento alla view all'operazione selezionata
Scenari alternativi	1. L'utente non ha i permessi per accedere alle operazioni di controllo della serra 2. L'utente non viene autenticato perchè non possiede un account 3. L'utente non viene autenticato perchè il suo account è stato sospeso
Requisiti non funzionali	
Punti aperti	

Table 9: GestionePermessi

Titolo	GestionePermessi
Descrizione	L'amministratore può assegnare e ritirare specifici permessi agli utenti
Attori	Amministratore
Relazioni	Amministrazione
Precondizioni	Utente autenticato, Utente abilitato
Postcondizioni	
Scenario principale	1. Mostrare lista di tutti gli utenti e rispettivi permessi 2. Aggiunta o rimozione di privilegi per uno o più utenti 3. Salvataggio della nuova configurazione degli account
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Table 10: GestioneUtenti

Titolo	GestioneUtenti
Descrizione	L'amministratore può registrare, bloccare o rimuovere gli account degli utenti
Attori	Amministratore
Relazioni	Amministrazione
Precondizioni	Utente autenticato, utente abilitato
Postcondizioni	
Scenario principale	1. Mostrata lista di tutti gli utenti ed i rispettivi permessi 2. Manipolazione degli account (inserimento, elimazione, sospensione) 3. Salvataggio nuova configurazione degli account
Scenari alternativi	1. L'utente non ha i permessi per accedere alle operazioni di controllo della serra 2. L'utente non viene autenticato perchè non possiede un account 3. L'utente non viene autenticato perchè il suo account è stato sospeso
Requisiti non funzionali	
Punti aperti	

Table 11: ElaborazioneParametri

Titolo	ElaborazioneParametri
Descrizione	Lettura dei dati dai sensori
Attori	Evento "fine minuto"
Relazioni	ControlloAutomaticoAttuatori, SalvataggioDati
Precondizioni	Necessità di aver scelto un preset in base alla pianta da controllare
Postcondizioni	il sistema ha salvato i dati
Scenario principale	1. Lettura di ogni parametro 2. Salvataggio dei parametri (inclusione) 3. ControlloAutomatico (inclusione) 4. Eventuale salvataggioDati (log) del cambio di stato degli attuatori (inclusione)
Scenari alternativi	1. Lettura di ogni parametro 2. Salvataggio dati (inclusione) 3. Analisi dei parametri 4. Il sistema rilva dei valori anomali, attiva il blocco di sicurezza e notifica chi di dovere
Requisiti non funzionali	
Punti aperti	

Table 12: ControlloAutomaticoAttuatori

Titolo	ControlloAutomaticoAttuatori
Descrizione	Analisi dei parametri ed attivazione/disattivazione di uno o più attuatori per modificare lo stato climatico della serra
Attori	ElaborazioneParametri
Relazioni	SalvataggioDati, ElaborazioneParametri
Precondizioni	Lettura dei parametri avvenuta con successo
Postcondizioni	Attuatore in funzione
Scenario principale	1. Mediante analisi del parametri, il sistema decide se modificare lo stato degli attuatori 2. Eventuale Attivazione/Disattivazione di attuatori 3. Notificare il cambiamento di stato di specifici attuatori per scatenare la generazione di un log
Scenari alternativi	
Requisiti non funzionali	Non deve succedere che uno strumento venga attivato quando non necessario o più tempo del necessario
Punti aperti	

Table 13: SalvataggioDati

Titolo	SalvataggioDati
Descrizione	Salva i dati dell'andamento dei parametri e del tempo di utilizzo degli attuatori, necessari a produrre resoconti
Attori	ElaborazioneParametri
Relazioni	ElaborazioneParametri, ControlloAutomaticoAttuatori
Precondizioni	Lettura dei parametri o controllo degli attuatori avvenuti con successo
Postcondizioni	Dati salvati persistentemente
Scenario principale	1. Salvataggio dei dati nel DB o inserimento di un log nel file appropriato
Scenari alternativi	
Requisiti non funzionali	
Punti aperti	

Table 14: ResocontoGiornaliero

Titolo	ResocontoGiornaliero
Descrizione	Produce il resoconto dell'andamento dei parametri e del tempo di utilizzo degli attuatori della giornata appena trascorsa.
Attori	Evento "fine giornata"
Relazioni	
Precondizioni	Sono stati salvati i dati della giornata appena trascorsa
Postcondizioni	Resoconto giornaliero salvato persistentemente e pronto alla visualizzazione su richiesta dell'utente.
Scenario principale	1. Recuperare dati relativi alla giornata appena trascorsa. 2. Generazione dei grafici e del report 3. Salvataggio in formato PDF del report
Scenari alternativi	Dati assenti o non sufficienti per un report, viene visualizzato un errore.
Requisiti non funzionali	
Punti aperti	

Table 15: ResocontoSettimanale

Titolo	ResocontoSettimanale
Descrizione	Produce il resoconto dell'andamento dei parametri e del tempo di utilizzo degli attuatori della settimana appena trascorsa.
Attori	Evento "fine settimana"
Relazioni	
Precondizioni	Sono stati salvati i dati della settimana appena trascorsa
Postcondizioni	Resoconto settimanale salvato persistentemente e pronto alla visualizzazione su richiesta dell'utente.
Scenario principale	1. Recuperati i dati relativi alla settimana appena trascorsa. 2. Generazione dei grafici e del report 3. Salvataggio in formato PDF del report
Scenari alternativi	Dati assenti o non sufficienti per un report, viene visualizzato un errore.
Requisiti non funzionali	
Punti aperti	

1.4 Analisi del rischio

1.4.1 Identificazione del bene

Table 16: Identificazione del bene

Bene	Valore	Esposizione
Sistema Informativo	Alto. Pilota gli attuatori, la configurazione delle soglie di attivazione e la visualizzazione dei resoconti.	Alta. Perdita d'immagine ed eventuale decadimento della pianta.
Sensore	Alto. Rilevazione dei parametri necessari alla gestione della serra.	Alta. Mancata o scorretta rilevazione di parametri significativi.
Microcontrollore	Alto. Lettura dei parametri dal sensore, trasmissione al software e pilotaggio degli attuatori.	Alta. Impossibilità di attivare o disattivare gli attuatori.
Attuatori	Medio. Dispositivi che concretano la gestione automatica della pianta.	Alta. Perdita del controllo sullo stato della pianta.
Resoconti	Basso. Contengono l'andamento dei parametri nel tempo e l'attività degli attuatori.	Bassa. Perdita di dati non necessari al funzionamento del sistema.
Credenziali	Alto. Credenziali degli utenti salvate dal sistema utilizzate in fase di autenticazione.	Alta. Rischio di privilege escalation a causa di un furto delle credenziali.

1.4.2 Identificazione delle minacce

Table 17: Identificazione delle minacce

Minaccia	Probabilità	Controllo	Fattibilità
Furto identità utente	Alta. Furto possibile se l'utente non custodisce le credenziali.	Controllo del numero di tentativi di accesso, log degli accessi e delle operazioni, salvataggio delle credenziali mediante crittografia	Basso costo di realizzazione
Violazione del sistema	Media. Il sistema non ha molte porte verso l'esterno	Utilizzo di crittografia per salvare credenziali utente.	Basso costo di realizzazione
Man in the middle attack	Bassa. Il sistema non esce dalla rete aziendale. La corruzione del traffico comporta la manipolazione del controllo.	Utilizzare canali di comunicazione dotati di protocolli di sicurezza.	Costo medio, protegge l'interoperabilità dei componenenti.
Manomissione (sensore, microcontrollore, attuatori, ambientale)	Bassa. Possibile solo con un accesso fisico alla serra o a causa di un fattore ambientale esterno (incendi ...)	Protezione fisica mediante alloggiamenti	Basso costo di realizzazione

1.4.3 Analisi delle tecnologie

Table 18: Analisi delle tecnologie

Tecnologia	Vulnerabilità
Accesso tramite credenziali	• Vengono sottratte all'utente le credenziali perchè custodite in modo non sicuro. • L'utente sceglie una password poco complessa e facilmente individuabile.

Table 18

Tecnologia	**Vulnerabilità**
Crittografia delle credenziali	• Esposizione della chiave crittografica. • Adottata chiave troppo corta. • Scelto metodo crittografico vulnerabile.
Protocolli di comunicaazione sicura	Furto della chiave privata utilizzata dal canale di sicurezza crittografato.
Alloggiamenti protettivi	Protezione da danni fisici.
Microcontrollore (protezioni elettriche e fisiche)	• Danni fisici • Condizioni ambientali dannose per l'hardware • Alimentazione elettrica con picchi di tensione.

1.4.4 Security use case & Misuse case

Figure 2: Security use/misuse case

Table 19: CanaleSicuro

<table>
<tr><td>Titolo</td><td colspan="2">CanaleSicuro</td></tr>
<tr><td>Descrizione</td><td colspan="2">La trasmissione sul canale deve essere sicura e permettere il controllo sull'integrità dei dati.</td></tr>
<tr><td>Misuse case</td><td colspan="2">Man in the middle</td></tr>
<tr><td>Relazioni</td><td colspan="2"></td></tr>
<tr><td>Precondizioni</td><td colspan="2">L'attaccante ha la possibilità di intercettare e modificare i dati trasmessi dal sensore al sistema o viceversa.</td></tr>
<tr><td>Postcondizioni</td><td colspan="2">Il sistema rileva la violazione all'integrità dei dati trasmessi.</td></tr>
<tr><td>Scenario principale</td><td>Attaccante</td><td>Sistema</td></tr>
<tr><td></td><td>Riesce ad accedere al canale sicuro e a manipolare i dati</td><td></td></tr>
<tr><td></td><td></td><td>Verifica l'integrità dei dati prima di utilizzarli, notificando un errore in caso di anomalie</td></tr>
<tr><td>Scenario di attacco avvenuto con successo</td><td>Attaccante</td><td>Sistema</td></tr>
<tr><td></td><td>Riesce a camuffare la manipolazione dei dati</td><td></td></tr>
<tr><td></td><td></td><td>Utilizza dei dati corrotti per gestire gli attuatori della serra</td></tr>
<tr><td></td><td></td><td>Effettua dei log dei dati ricevuti e effettua delle analisi periodicamente al fine di ricercare valori anomali</td></tr>
</table>

Table 20: ControlloAccessi

<table>
<tr><td>Titolo</td><td colspan="2">ControlloAccessi</td></tr>
<tr><td>Descrizione</td><td colspan="2">L'autenticazione prevede un limite al numero di tentativi di accesso</td></tr>
<tr><td>Misuse case</td><td colspan="2">Furto credenziali</td></tr>
<tr><td>Relazioni</td><td colspan="2"></td></tr>
<tr><td>Precondizioni</td><td colspan="2">L'attaccante è riuscito ad ottenere l'username di un utente legittimo.</td></tr>
<tr><td>Postcondizioni</td><td colspan="2">Il sistema blocca l'accesso all'account e segnala il tentativo malevolo.</td></tr>
<tr><td>Scenario principale</td><td>Attaccante</td><td>Sistema</td></tr>
<tr><td></td><td>Dopo aver scoperto un username, intraprende un attacco a dizionario per indovinare la password.</td><td></td></tr>
<tr><td></td><td></td><td>Blocca l'accesso dopo un certo numero di tentativi errati.</td></tr>
</table>

Scenario di attacco avvenuto con successo	**Attaccante**	**Sistema**
	Riesce ad indovinare la password prima di esaurire i tentativi.	
		Consente l'accesso perchè sono state inserite le credenziali corrette
	È libero di navigare tra le maschere del sistema e sfruttare i privilegi dell'utente per fini malevoli.	
		Effettua periodicamente un'analisi dei log per rilevare comportamenti insoliti

Table 21: CredenzialiCrittografate

Titolo	CredenzialiCrittografate
Descrizione	Il sistema memorizza le credenziali degli utenti in forma crittografata.
Misuse case	Violazione del sistema
Relazioni	
Precondizioni	L'attaccante riesce a penetrare nel sistema aggirando la fase di autenticazione
Postcondizioni	L'attaccante ha accesso alle credenziali in formato crittografato, non è dunque in grado di ottenere nessuna informazione sensibile

<table>
<tr><td>Scenario principale</td><td><table>
<tr><th>Attaccante</th><th>Sistema</th></tr>
<tr><td>Dopo essersi infiltrato nel sistema, legge le credenziali degli utenti.</td><td></td></tr>
<tr><td></td><td>Le credenziali erano state preventivamente crittate</td></tr>
</table></td></tr>
<tr><td>Scenario di attacco avvenuto con successo</td><td><table>
<tr><th>Attaccante</th><th>Sistema</th></tr>
<tr><td>l'attaccante riesce a scoprire la password partendo dal testo cifrato con tecniche di brute force.</td><td></td></tr>
<tr><td></td><td>Non può impedire che le credenziali vengano decifrate.</td></tr>
</table></td></tr>
</table>

Table 22: StatoSicurezza

Titolo	StatoSicurezza
Descrizione	Il sistema deve avvertire l'utente e/o l'amministratore di anomalie dovute a manomissioni o a fattori ambientali anomali.
Misuse case	Manomissione
Relazioni	
Precondizioni	L'attaccante ha accesso fisico alla serra e alla sensoristica
Postcondizioni	Il sistema blocca la gestione automatica della serra, notificando all'utente responsabile un'anomalia.

<table>
<tr><td>Scenario principale</td><td>
<table>
<tr><th>Attaccante</th><th>Sistema</th></tr>
<tr><td>Dopo aver aggirato le misure di sicurezza fisiche della serra, manomette i sensori o gli attuatori.</td><td></td></tr>
<tr><td></td><td>Nota una variazione di uno o più parametri che si discosta molto dal range prefissato e notifica l'utente responsabile.</td></tr>
<tr><td></td><td>Sospende il controllo automatico degli attuatori, entrando in uno stato di sicurezza.</td></tr>
<tr><td></td><td>Quando i parametri tornano a valori normali, il sistema si sblocca e riprendere la gestione automatica.</td></tr>
</table>
</td></tr>
<tr><td>Scenario di attacco avvenuto con successo</td><td>
<table>
<tr><th>Attaccante</th><th>Sistema</th></tr>
<tr><td>Un evento ambientale casuale e imprevisto provoca cambiamenti repentini delle condizioni ambientali della serra.</td><td></td></tr>
<tr><td></td><td>Nota una variazione di uno o più parametri che si discosta di molto dal range prefissato e notifica l'utente della necessità di agire al più presto al fine di ristabilire il normale funzionamento della serra.</td></tr>
</table>
</td></tr>
</table>

2 Analisi del problema

2.1 Analisi del documento dei requisiti

2.1.1 Analisi della funzionalità

Table 23: Tabella delle funzionalità

Funzionalità	Tipo	Grado complessità	Requisiti collegati
GestioneSerra	Interazione esterno, memorizzazione e gestione dati	Complessa	R2F, R10F, R11F, R12F
Amministrazione	Interazione esterno, gestione dati	Complessa	R8F, R9F
Autenticazione	Interazione esterno	Semplice	
ElaborazioneParametri	Gestione dati, controllo attuatori	Complessa	R1F, R3F
SalvataggioDati	Memorizzazione dati	Semplice	R4F, R5F
ResocontoGiornaliero	Interazione esterno, memorizzazione dati	Semplice	R4F, R5F, R6F
ResocontoSettimanale	Interazione esterno, memorizzazione dati	Semplice	R4F, R5F, R7F

Table 24: GestioneSerra: Tabella informazioni/flusso

Informazione	Tipo	Livello protezione/privacy	Input/output	Vincoli
Preset	Complesso	Medio	Input, output	Valori numerici finiti
Resoconto	Complesso	Medio	Output	
Tipo resoconto	Semplice	Basso	Input	Giornaliero o settimanale
Date filtro (input utente)	Semplice	Basso	Input	Intervallo di tempo non negativo

Table 24

Informazione	Tipo	Livello protezione/privacy	Input/output	Vincoli
Stato attuatore	Semplice	Alto	Input, Output	Acceso o spento
Pianificazione manuale attuatore	Complesso	Medio	Input	Intervallo di tempo non negativo

Table 25: Amministrazione: Tabella informazioni/flusso

Informazione	Tipo	Livello protezione/privacy	Input/output	Vincoli
Permessi	Complesso	Alto	Input, Output	Inserire permesso valido (enum)
Insieme utenti	Complesso	Alto	Output	
Nuovo utente	Complesso	Alto	Input	Fornire tutte le informazioni richieste
Rimuovi utente	Semplice	Alto	Input	L'utente deve esistere
Blocca utente	Semplice	Alto	Input	L'utente deve esistere

Table 26: Autenticazione: Tabella informazioni/flusso

Informazione	Tipo	Livello protezione/privacy	Input/output	Vincoli
Username	Semplice	Alto	Input	Non vuote
Password	Semplice	Alto	Input	Non vuote

Table 27: ElaborazioneParametri: Tabella informazioni/flusso

Informazione	**Tipo**	**Livello protezione/privacy**	**Input/output**	**Vincoli**
Temperatura	Semplice	Medio	Input	
Luce	Semplice	Medio	Input	
Umidità del terreno	Semplice	Medio	Input	Non negativo
Umidità dell'aria	Semplice	Medio	Input	

Table 28: SalvataggioDati: Tabella informazioni/flusso

Informazione	**Tipo**	**Livello protezione/privacy**	**Input/output**	**Vincoli**
Temperatura	Semplice	Medio	Input	
Luce	Semplice	Medio	Input	
Umidità del terreno	Semplice	Medio	Input	Non negativo
Umidità dell'aria	Semplice	Medio	Input	

Table 29: ResocontoGiornaliero: Tabella informazioni/flusso

Informazione	**Tipo**	**Livello protezione/privacy**	**Input/output**	**Vincoli**
Temperatura	Semplice	Medio	Input	
Luce	Semplice	Medio	Input	
Umidità del terreno	Semplice	Medio	Input	Non negativo
Umidità dell'aria	Semplice	Medio	Input	
Resoconto giornaliero	Complesso	Basso	Output	

Table 30: ResocontoSettimanale: Tabella informazioni/flusso

Informazione	**Tipo**	**Livello protezione/privacy**	**Input/output**	**Vincoli**
Temperatura	Semplice	Medio	Input	
Luce	Semplice	Medio	Input	
Umidità del terreno	Semplice	Medio	Input	Non negativo
Umidità dell'aria	Semplice	Medio	Input	
Resoconto settimanale	Complesso	Basso	Output	

2.1.2 Analisi dei vincoli

Table 31: Tabella dei vincoli

Requisito	Categorie	Impatto	Funzionalità
Semplicità di utilizzo	Usabilità	Migliora la navigabilità	GestioneSerra, Autenticazione, Amministrazione
Velocità di memorizzazione	Tempo di risposta	Migliora il tempo di risposta	SalvataggioDati, ResocontoGiornaliero, ResocontoSettimanale
Velocità di recupero dei dati	Tempo di risposta	Migliora il tempo di risposta	GestioneSerra, Amministrazione, Autenticazione
Precisione del controllo degli attuatori	Esattezza	Migliora il controllo della pianta	ElaborazioneParametri
Consumo energetico	Efficienza	Peggiora il tempo di risposta, migliora l'efficienza energetica (riduce costi di consumo, diminuisce l'impatto ambientale)	ElaborazioneParametri, GestioneSerra
Controllo degli accessi	Sicurezza	Peggiora il tempo di risposta e usabilità, migliora la privacy dei dati	Autenticazione, GestioneSerra, Amministrazione
Controllo dei permessi	Sicurezza	Peggiora il tempo di risposta, migliora la sicurezza	GestioneSerra, Amministrazione
Protezione dei dati sensibili	Sicurezza	Peggiora tempo di risposta e usabilità, migliora la privacy dei dati	GestioneSerra, Autenticazione, Amministrazione

2.1.3 Analisi delle interazioni

Table 32: Tabella delle maschere

Maschera	Informazioni	Funzionalità
HomeGestioneSerra	Visualizzazione informazioni sullo stato del sistema, scelta delle funzionalità	GestioneSerra
ViewVisualizzaResoconto	Date filtro, tipo resoconto	GestioneSerra
ViewControlloManualeAttuatori	Stato attuatore, pianificatore manuale attuatore	GestioneSerra
ViewRiconfigurazionePreset	Preset	GestioneSerra
ViewInserimentoPreset	Preset	GestioneSerra
HomeAmministrazione	Scelta delle funzionalità	Amministrazione
ViewGestionePermessi	Lista utenti con rispettivi permessi, aggiunta e rimozione di permessi	Amministrazione
ViewGestioneUtenti	Lista degli utenti, aggiunta, rimozione, blocco utenti	Amministrazione
ViewAutenticazione	Username, password	Autenticazione

Table 33: Tabella sistemi esterni

Sistema	Descrizione	Protocollo di interazione	Livello di sicurezza
Sensori	Sistemi hardware che inviano dati su temperatura e umidità al microcontrollore.	Ogni sensore invia al microcontrollore un flusso di byte rappresentante uno specifico parametro.	Medio. Il sensore non è attaccabile a livello informatico ma è fragile fisicamente
Microcontrollore	Sistema hardware che trasmette le informazioni dai sensori al sistema	Il microcontrollore trasmette tramite Wi-Fi i dati al sistema.	Medio. Il microcontrollore è vulnerabile fisicamente

2.2 Analisi dei ruoli e delle responsabilità

Table 34: Tabella dei ruoli

Ruoli	Responsabilità	Maschere	Riservatezza	Numerosità
Utente	Può visualizzare e/o modificare lo stato della serra (preset e attivazione degli attuatori) sulla base dei propri permessi.	HomeGestioneSerra, ViewVisualizzaResoconto, ViewControlloManualeAttuatori, ViewRiconfigurazionePreset, ViewInserimentoPreset	È richiesto un alto grado di riservatezza.	Il numero di utenti non ha un limite massimo, dipende dall'organizzazione e dal cliente.
Amministratore	Gestione degli utenti (aggiunta, rimozione e sospensione), rispettivi permessi.	HomeAmministrazione, ViewGestionePermessi, ViewGestioneUtenti, ViewAutenticazione	È richiesto un notevole grado di riservatezza.	2-3 persone considferando i turni di lavoro ed i giorni liberi

Table 35: Utente: Tabella ruolo-informazioni

Informazione	Tipo di accesso
Preset	Lettura/Scrittura
Resoconto	Lettura
Tipo resoconto	Scrittura
Stato attuatore	Lettura/Scrittura
Date filtro	Scrittura

Table 36: Amministratore: Tabella ruolo-informazioni

Informazione	Tipo di accesso
Lista utenti con rispettivi permessi	Lettura/Scrittura
Lista degli utenti	Lettura/Scrittura
Username	Scrittura
Password	Scrittura

2.3 Scomposizione del problema

Table 37: Tabella scomposizione funzionalità

Funzionalità	Scomposizione
GestioneSerra	VisualizzaResoconto, RiconfigurazionePreset, InserimentoPreset, ControlloManualeAttuatori
Amministrazione	AggiungiPermessi, RimuoviPermessi, AggiungiUtente, RimuoviUtente, BloccaUtente
ElaborazioneParametri	ControlloAutomaticoAttuatori

Table 38: GestioneSerra: Tabella sotto-funzionalità

Sotto-funzionalità	Sotto-funzionalità	Legame	Informazioni
RiconfigurazionePreset	InserimentoPreset	Non si può modificare un preset che non è ancora stato creato	Preset

Table 39: Amministrazione: Tabella sotto-funzionalità

Sotto-funzionalità	**Sotto-funzionalità**	**Legame**	**Informazioni**
RimuoviUtente	AggiungiUtente	Non è possibile rimuovere un account utente se non è stato inserito nel sistema	Dati identificativi del'account utente (username)
BloccaUtente	AggiungiUtente	Non è possibile bloccare un account utente se non è stato inserito nel sistema	Dati identificativi dell'account utente (username)
AggiungiPermessi	AggiungiUtente	Non è possibile aggiungere un permesso se non è stato inserito nel sistema	Dati identificativi dell'account utente, permessi da aggiungere
RimuoviPermessi	AggiungiPermessi	Non è possibile rimuovere un permesso senza averlo prima assegnato	Dati identificativi dell'account utente, Permessi da rimuovere

Table 40: ElaborazioneParametri: Tabella sotto-funzionalità

Sotto-funzionalità	**Sotto-funzionalità**	**Legame**	**Informazioni**
ControlloAutomatico Attuatori	Elaborazione Parametri	Non è possibile attivare un attuatore senza aver prima letto i dati dal sensore	Temperatura, luce, umidità del terreno

2.4 Creazione del modello di dominio

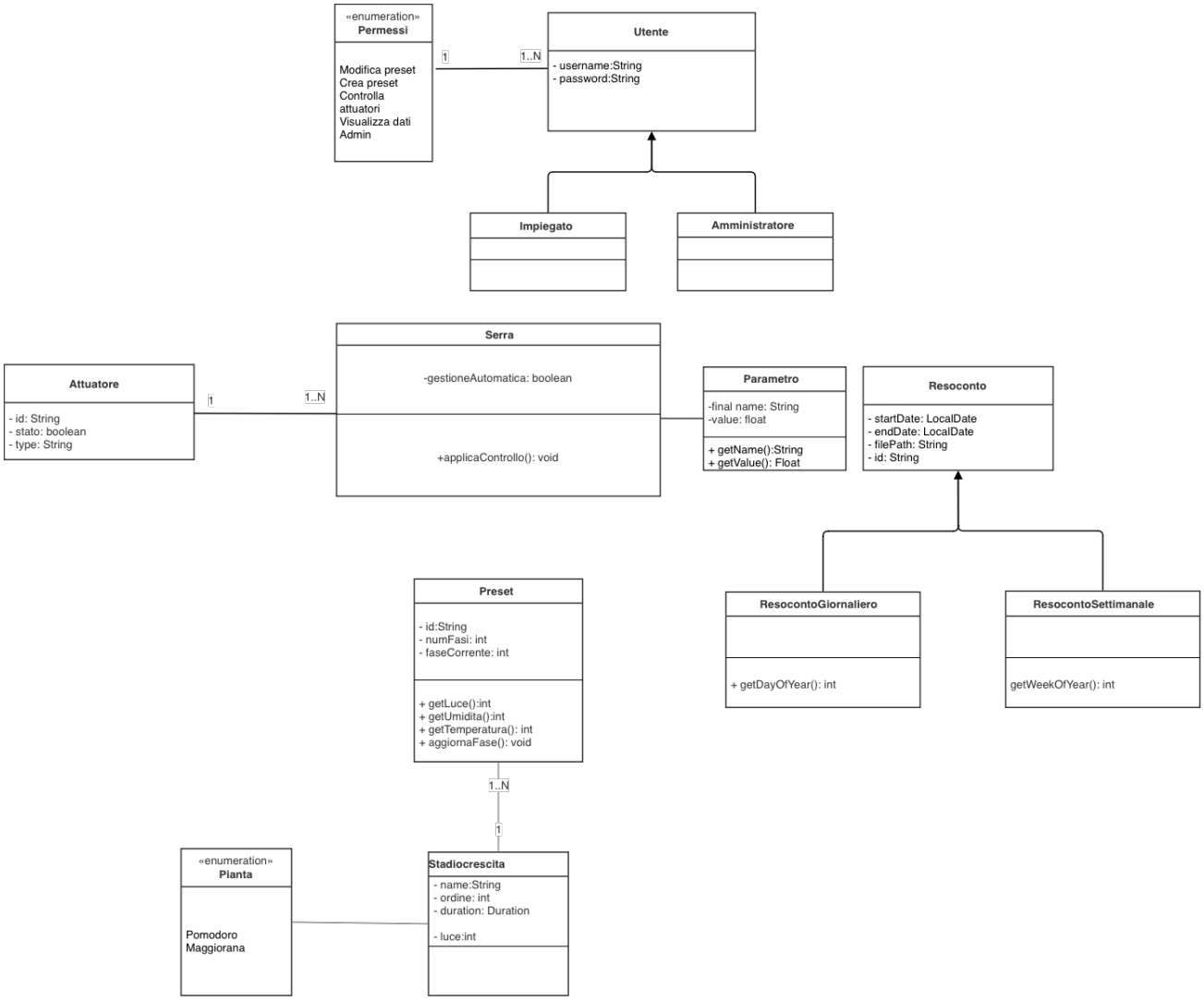


Figure 3: Modello del dominio

2.5 Architettura logica

2.5.1 Struttura

Diagramma dei Package

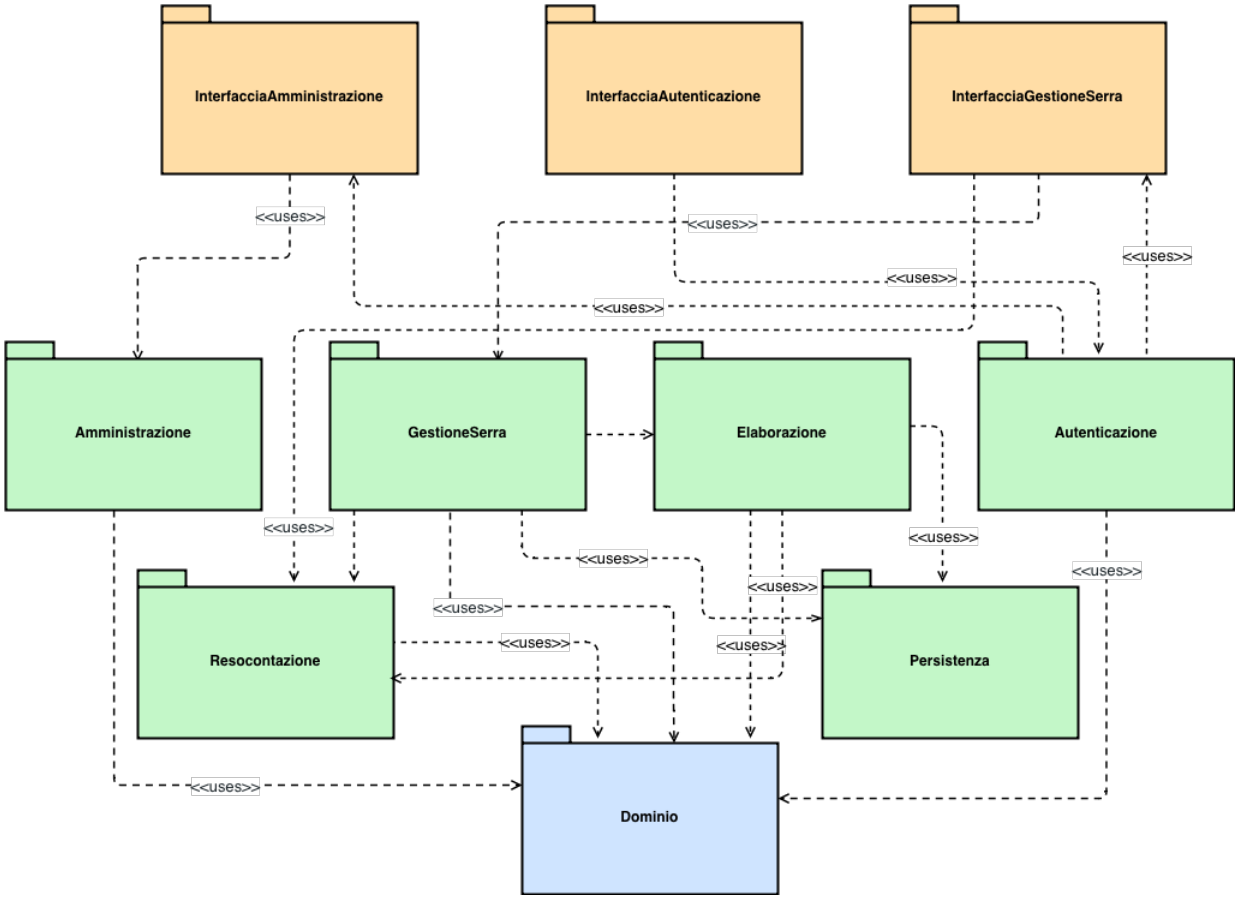


Figure 4: Diagramma dei package

Diagramma delle classi

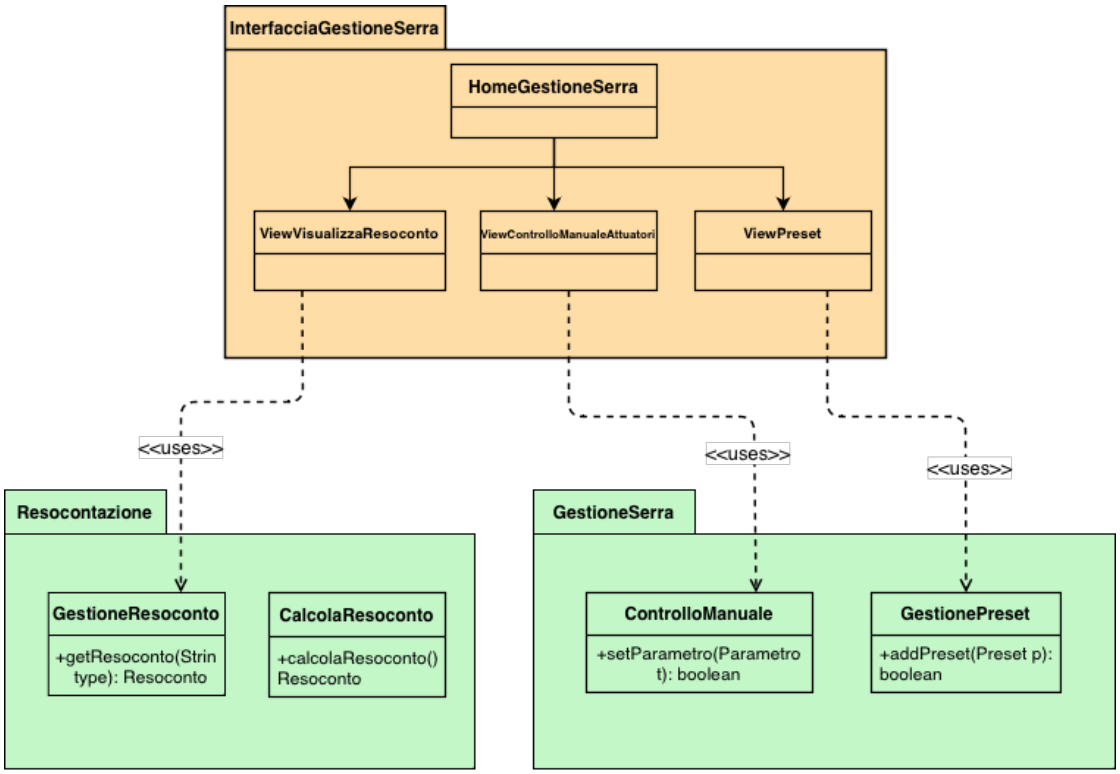


Figure 5: Diagramma delle classi

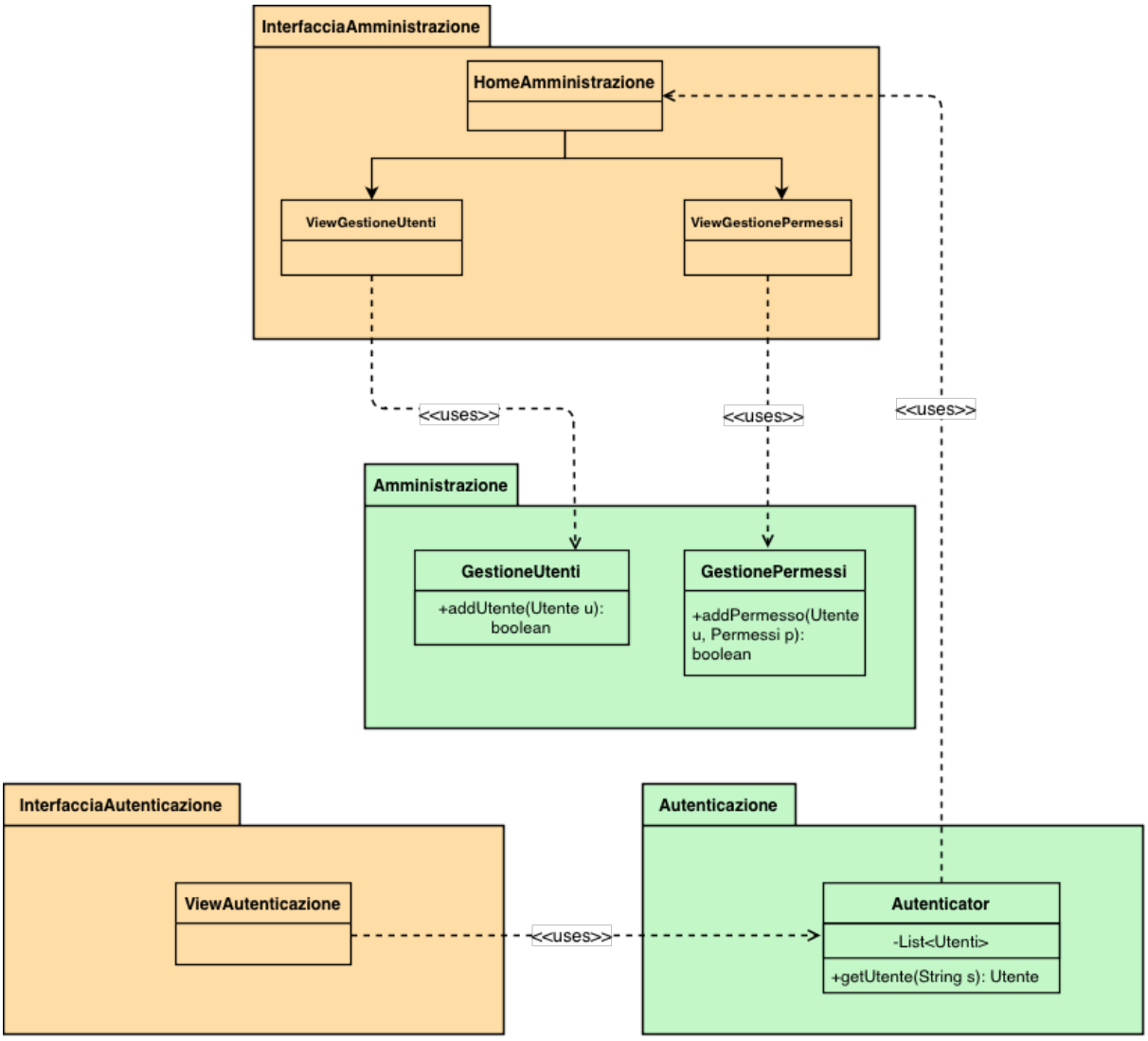


Figure 6: Diagramma delle classi

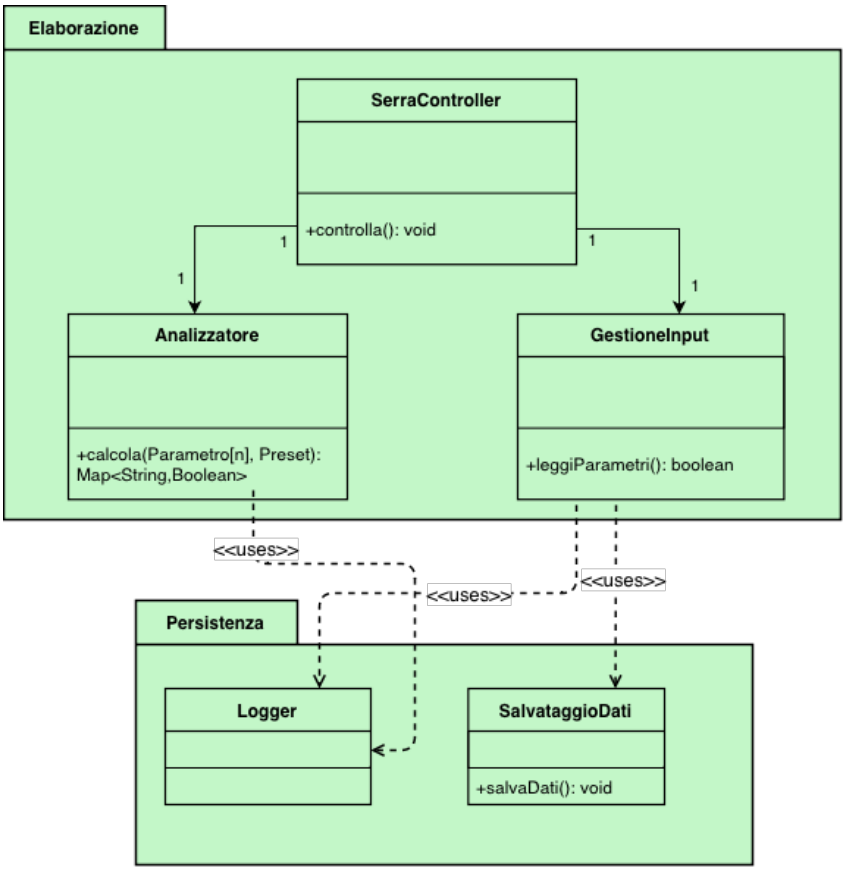


Figure 7: Diagramma delle classi

2.5.2 Interazione

Gestione utente e gestione permessi

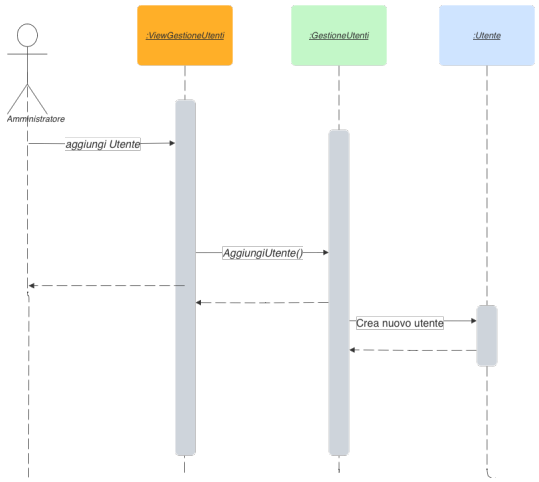


Figure 8: Sequenza interazioni Gestione utente

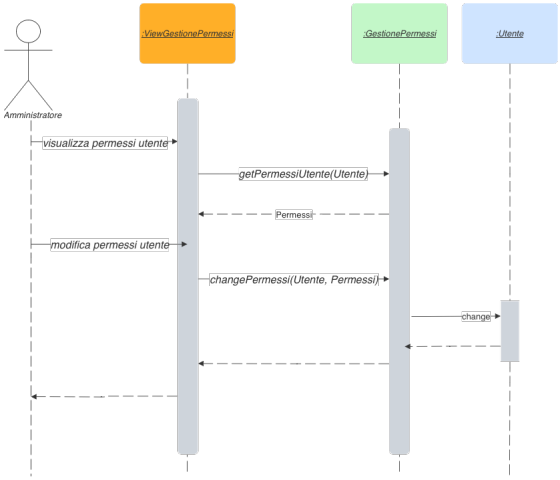


Figure 9: Sequenza interazioni Gestione permessi

Controllo automatico attuatori

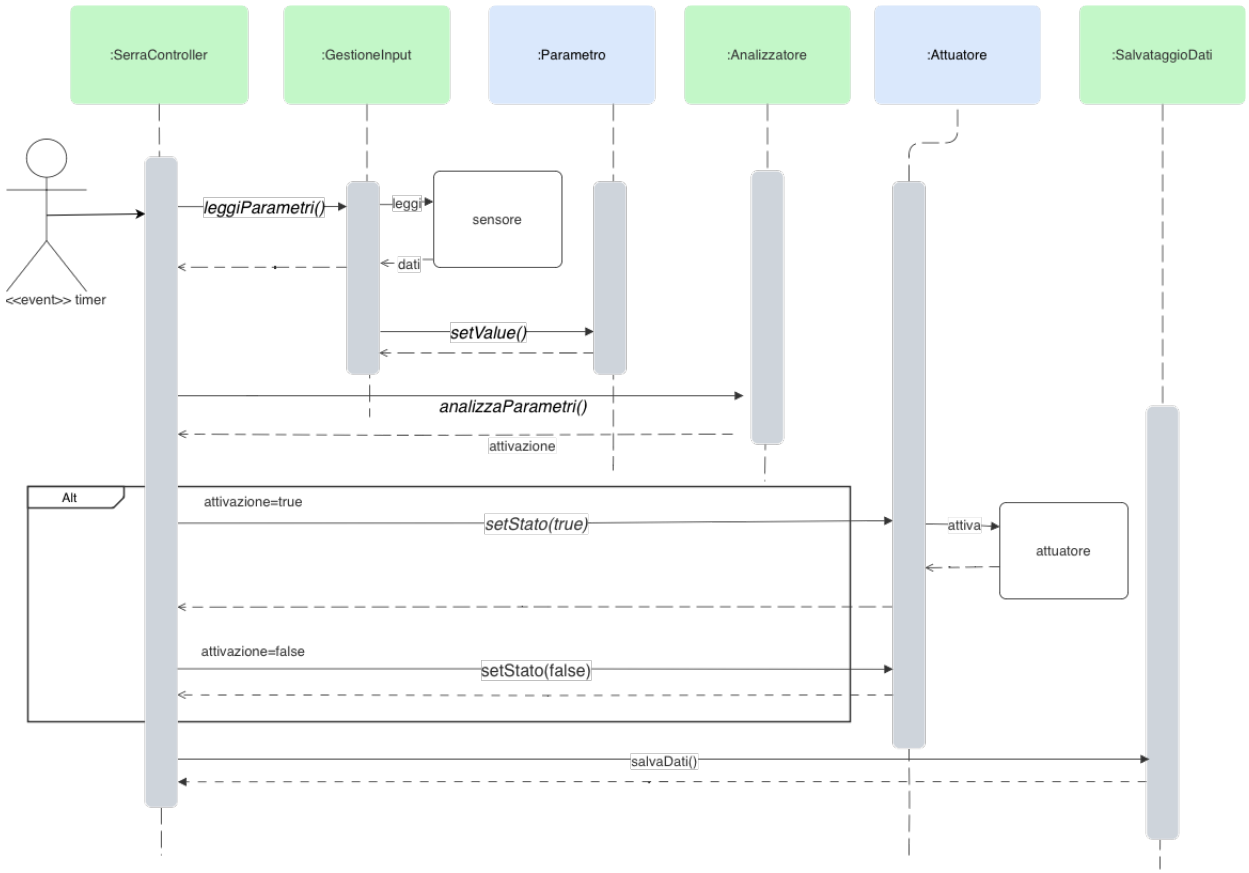


Figure 10: Sequenza interazioni controllo automatico attuatori

Viene qui mostrato l'eventuale attivazione di un attuatore dopo aver letto da un sensore, ed il successivo salvataggio dei dati. Nel sistema saranno sempre presenti più sensori e più attuatori, ma possono essere gestiti singolarmente come illustrato dal diagramma.

Controllo manuale attuatori

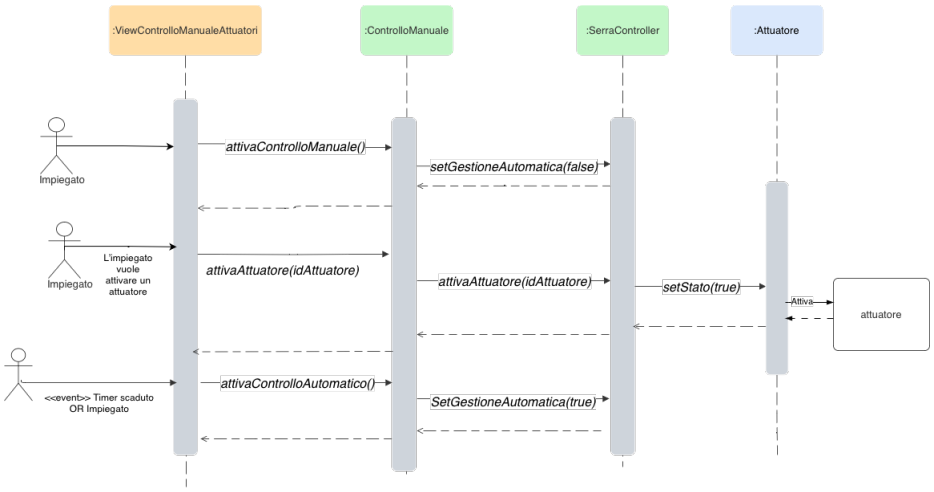


Figure 11: Sequenza interazioni controllo manuale attuatori

Il diagramma evidenzia il passaggio dalla gestione automatica alla gestione manuale della serra. Dopodiché è possibile attivare manualmente uno specifico attuatore o effettuare altre operazioni, come modificare i valori dei parametri del preset, il che avrà effetto solo dopo essere ritornati alla gestione automatica.

Visualizzazione resoconto

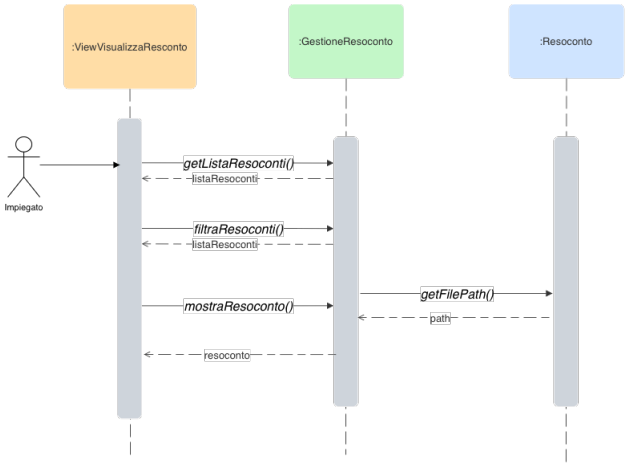


Figure 12: Sequenza interazioni visualizzazione resoconto

Gestione Preset

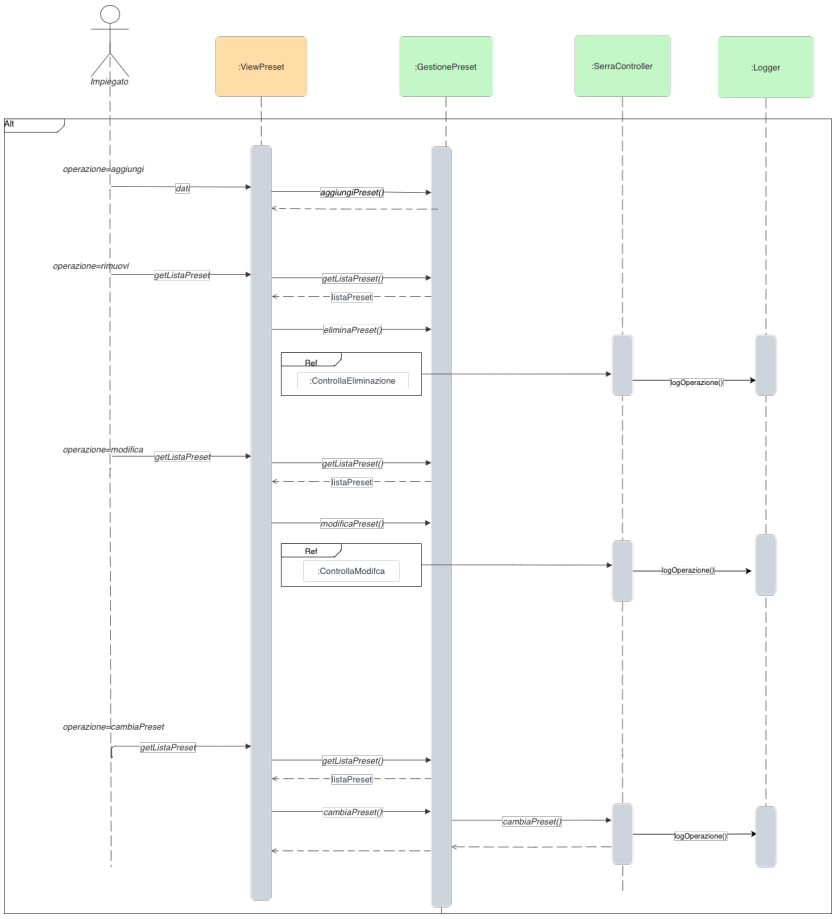


Figure 13: Sequenza interazioni gestione Preset

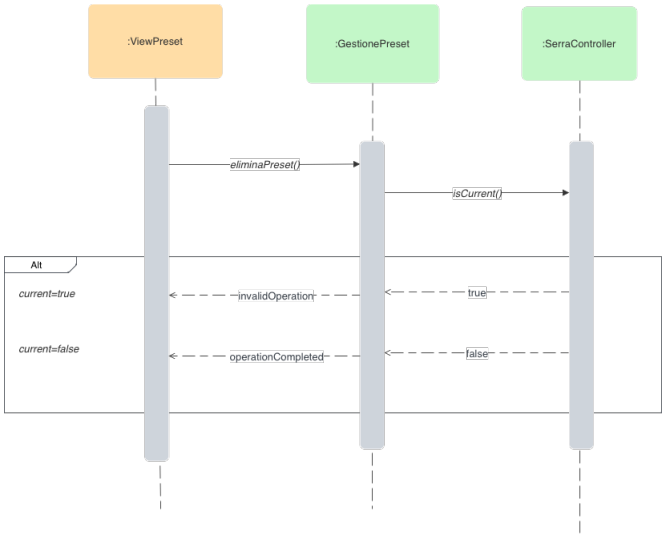


Figure 14: Sequenza interazioni eliminazione Preset (REF controllaEliminazione)

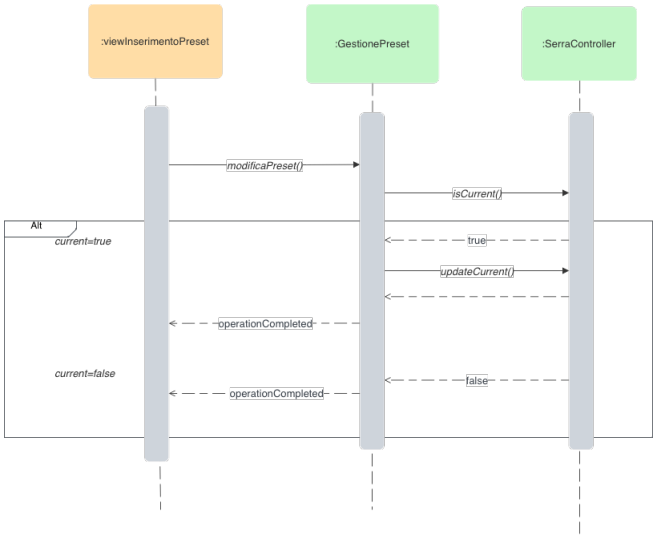


Figure 15: Sequenza interazioni modifica preset (REF controllaModifica)

2.5.3 Comportamento

Il seguente diagramma di stato mostra brevemente come si alternano le diverse modalità di funzionamento della serra sulla base delle differenti situazioni.

I due stati principali del sistema sono legati al normale funzionamento della serra, in cui si applica un controllo manuale o automatico, a discrezione dell'utente. Il terzo stato rappresenta un blocco di sicurezza degli attuatori quando si rilevano per un certo periodo di tempo dei parametri ambientali anonimi, che segnalano la presenza di una condizione non prevista all'interno della serra, in quel caso il controllo automatico viene sospeso.

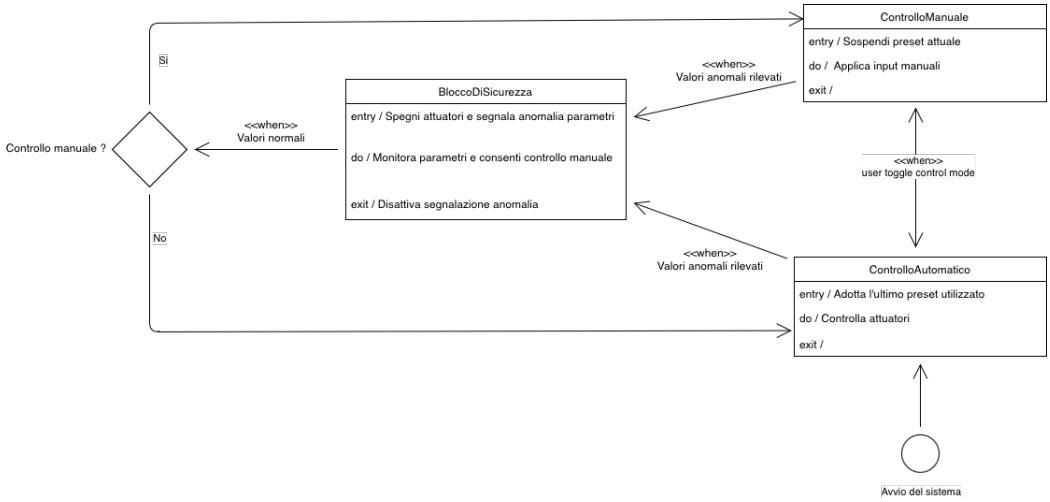


Figure 16: Diagramma di stato

2.6 Piano di lavoro

Table 41: Piano di lavoro

Package	Progetto	Sviluppo
Dominio	Basigli, Faggiano, Salman	Basigli, Faggiano, Salman
GestioneSerra	Basigli, Faggiano, Salman	Faggiano, Salman
Elaborazione	Basigli, Faggiano, Salman	Faggiano, Salman
Amministrazione	Basigli, Faggiano, Salman	Basigli, Salman
Autenticazione	Basigli, Faggiano, Salman	Basigli, Salman

Table 41

Package	Progetto	Sviluppo
InterfacciaGestioneSerra	Basigli, Faggiano, Salman	Faggiano, Salman
InterfacciaAutenticazione	Basigli, Faggiano, Salman	Basigli, Salman
InterfacciaAmministrazione	Basigli, Faggiano, Salman	Basigli, Salman

I tempi di rilascio previsti sono i seguenti:

- Progettazione in 2 settimane dalla data odierna;
- Sviluppo delle singole parti con collaudo unitario entro una settimana dall fine edella progettazione;
- Intergrazione e test dell'intero sistema entro una settimana rispetto alla fine dello sviluppo.

Sviluppi futuri

Il committente ha richiesto di estendere il sistema e renderlo distribuito, in modo che sia accessibile da client remoti e che sia in grado di gestire più serre della stessa pianta in modo centralizzato.

2.7 Piano del collaudo

Vengono riportati i testi unitari e di integrazione cruciali per il sistema.

Listing 1: Stadio Test

```
import org.junit.Assert;
import org.junit.Before;
import org.junit.Test;
import java.time.Duration;

public class StadioTest {
    private Stadio stadio;

    @Before
    public void setUp() {
        stadio = new Stadio("Germoglio", 1, Duration.ofHours(3));
    }

    @Test
    public void testGetName() {
        String expected = "Germoglio";
        String actual = stadio.getName();
        Assert.assertEquals(expected, actual);
    }

    @Test
    public void testGetOrdine() {
        int expected = 1;
        int actual = stadio.getOrdine();
        Assert.assertEquals(expected, actual);
    }

    @Test
    public void testGetDuration() {
        Duration expected = Duration.ofHours(3);
        Duration actual = stadio.getDuration();
        Assert.assertEquals(expected, actual);
    }

    @Test
    public void testSetLuce() {
        int luce = 80;
        stadio.setLuce(luce);
        int actual = stadio.getLuce();
        Assert.assertEquals(luce, actual);
    }

    @Test
    public void testSetUmidita() {
        int umidita = 60;
```

```
        stadio.setUmidita(umidita);
        int actual = stadio.getUmidita();
        Assert.assertEquals(umidita, actual);
    }

    @Test
    public void testSetTemperatura() {
        int temperatura = 25;
        stadio.setTemperatura(temperatura);
        int actual = stadio.getTemperatura();
        Assert.assertEquals(temperatura, actual);
    }
}
```

Listing 2: Preset test

```
import org.junit.Assert;
import org.junit.Before;
import org.junit.Test;

public class PresetTest {
    private Preset preset;

    @Before
    public void setUp() {
        preset = new Preset("preset1", 3, Pianta.Pomodoro, new Stadio[1], 0);
    }

    @Test
    public void testGetId() {
        String expected = "preset1";
        String actual = preset.getId();
        Assert.assertEquals(expected, actual);
    }

    @Test
    public void testGetNumFasi() {
        int expected = 3;
        int actual = preset.getNumFasi();
        Assert.assertEquals(expected, actual);
    }

    @Test
    public void testGetPlant() {
        Pianta plant = new Pianta();
        preset.setPlant(plant);
        Pianta actual = preset.getPlant();
        Assert.assertEquals(plant, actual);
    }
```

```
    @Test
    public void testGetFasi() {
        Stadio[] fasi = {new Stadio("Germoglio", 1, Duration.ofHours(3), 80, 60, 25);};
        preset.setFasi(fasi);
        Stadio[] actual = preset.getFasi();
        Assert.assertArrayEquals(fasi, actual);
    }

    @Test
    public void testGetFaseCorrente() {
        int expected = 0;
        int actual = preset.getFaseCorrente();
        Assert.assertEquals(expected, actual);
    }

    @Test
    public void testSetFaseCorrente() {
        int faseCorrente = 2;
        preset.setFaseCorrente(faseCorrente);
        int actual = preset.getFaseCorrente();
        Assert.assertEquals(faseCorrente, actual);
    }
}
```

Listing 3: Impiegato test

```
import org.junit.Assert;
import org.junit.Before;
import org.junit.Test;
import java.util.HashMap;
import java.util.Map;

public class ImpiegatoTest {
    private Impiegato impiegato;
    private Preset preset;
    private Stadio stadio;
      private Map<String, Preset> presets;

    @Before
    public void setUp() {
        stadio = new Stadio("Germoglio", 1, Duration.ofHours(3), 80, 60, 25);
        preset = new Preset("preset1", 3, Pianta.Pomodoro, {stadio}, 0);
         presets = new HashMap<String, Preset>("P1", Preset)
        impiegato = new Impiegato({Permessi.ModificaPreset, Permessi.CreaPreset}, presets);
    }

    @Test
    public void testGetPrivilegies() {
        Permessi[] expected = {Permessi.ModificaPreset, Permessi.CreaPreset,
            Permessi.ControllaAttuatori, Permessi.VisualizzaDati, Permessi.Admin};
```

```
        Permessi[] actual = impiegato.getPrivilegies();
        Assert.assertArrayEquals(expected, actual);
    }

    @Test
    public void testSetPrivilegies() {
        Permessi[] privilegies = {Permessi.Admin, Permessi.VisualizzaDati};
        impiegato.setPrivilegies(privilegies);
        Permessi[] actual = impiegato.getPrivilegies();
        Assert.assertArrayEquals(privilegies, actual);
    }

    @Test
    public void testPianifica() {
        // Test implementation
    }

    @Test
    public void testCreaPreset()
            Preset p2 = new Preset("preset2", 4, Pianta.Basilico, {stadio}, 0))
        impiegato.creaPreset(p2);
        Assert.assertEquals(p2, impiegato.getPreset("P2"));
    }

    @Test
    public void testModificaPreset() {
            preset.setLuce(30);
        boolean result = impiegato.modificaPreset("P1", preset);
        Assert.assertTrue(result);
    }

    @Test
    public void testModificaPreset_NonEsistente() {
        boolean result = impiegato.modificaPreset("P5", preset);
        Assert.assertFalse(result);
    }

    @Test
    public void testVisualizzaResoconto() {
        impiegato.creaPreset("P1", preset);
        Resoconto resoconto = impiegato.visualizzaResoconto("P1");
        Assert.assertNotNull(resoconto);
    }
}
```

3 Progettazione

3.1 Progettazione architetturale

Requisiti non funzionali
Dall'analisi dei requisiti emerge che aspetti critici da tenere in considerazione sono:

- Semplicità di utilizzo;
- Velocità di memorizzazione e recupero dei dati;
- Precisione ed efficienza della gestione dei controllori;
- Protezione e controllo degli accessi;
- Efficienza energetica.

È dunque fondamentale progettare il sistema in modo tale da garantire delle performance soddisfacenti mantenendo tuttavia livelli di sicurezza efficaci. Difatti tali requisiti tendono ad andare in conflitto tra loro dal momento in cui aggiungere strumenti di controllo degli accessi lede alla semplicità di utilizzo e alla velocità di interazione con il sistema.
Un ulteriore aspetto che merita attenzione è l'efficienza energetica. Oltre ad analisi che riguardano caratteristiche fisiche della serra (come ad esempio una corretta ed efficiente coibentazione interna) sarà anche importante definire degli algoritmi più o meno complessi che, usando dati storici della serra riescano a diminuire lo spreco di energia, tendendo alla minimizzazione delle variazioni di stato degli attuatori (dovrebbero passare on/off o off/on il meno possibile). Nonostante non viene qui fatta tale analisi, risulta meritevole di attenzione.
Infine, evidenziamo l'utilità dei resoconti, che avranno compito di mostrare nella maniera più esplicativa possibile l'andamento delle condizioni ambientali e l'operatività della sistema nel tempo, al fine di monitorare con precisione l'efficienza della gestione automatica su ampi lassi di tempo.

Scelta dell'architettura
L'architettura più idonea per questo tipo di sistema è di tipo MVC con un database per la persistenza. Considerando le specifiche attuali, non è necessario che il sistema sia distribuito o si connetta alla rete, dunque per motivi di sicurezza evitiamo di aggiungere questa caratteristica e limitiamo il sistema ad un MVC in locale. Verranno presi in considerazione accorgimenti per rendere il sistema facilmente estendibile al distribuito, per essere accessibile da remoto o gestire più serre in una rete locale, in modo da poter soddisfare eventuali sviluppi futuri richiesti dal cliente.

Il model si occuperà di rappresentare i dati legati alla serra, tenendo traccia dello stato interno (quali attuatori sono attivi/disattivi, quale preset è in funzione, ...). Tali dati saranno analizzati dal controller che si preoccuperà di gestire l'accensione degli attuatori e la creazione dei report. Il tutto sarà visualizzato all'utente tramite le interfacce della view, che offriranno anche la possibilità di intervenire sul sistema di controllo.

Si riporta il diagramma dell'architettura del sistema:

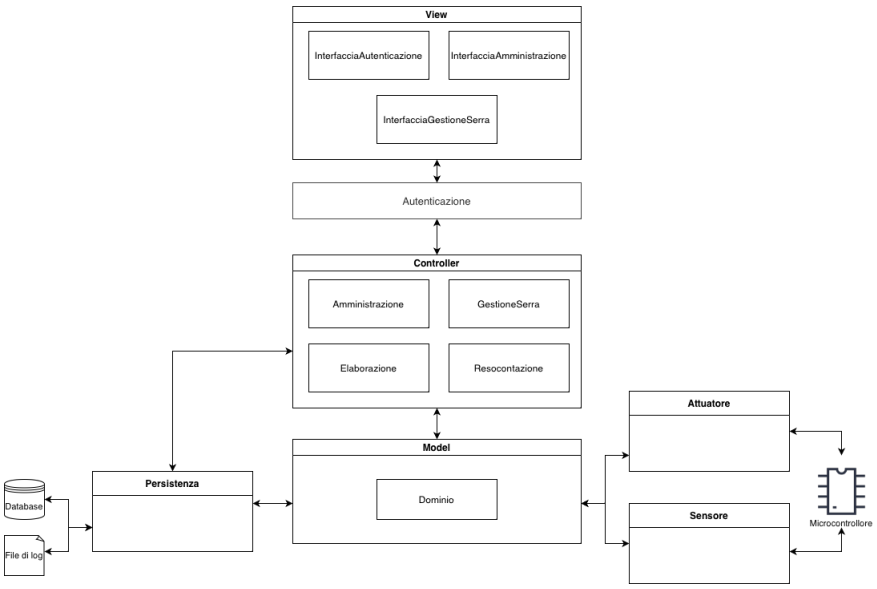
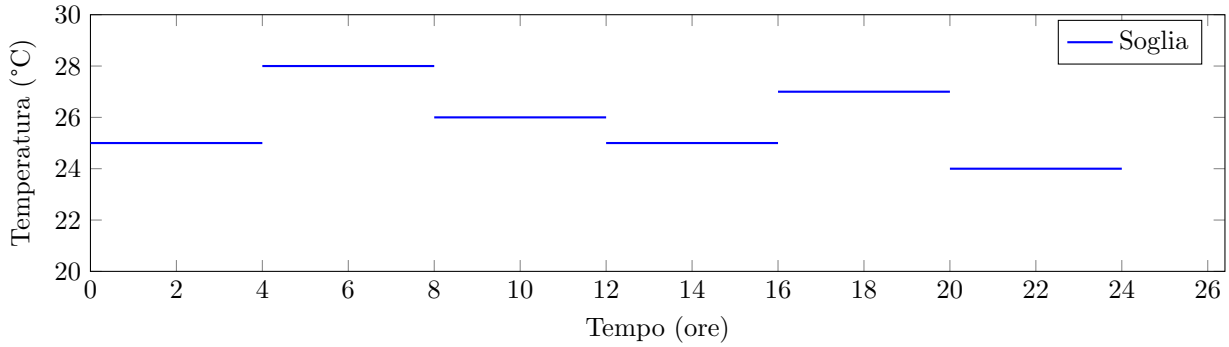
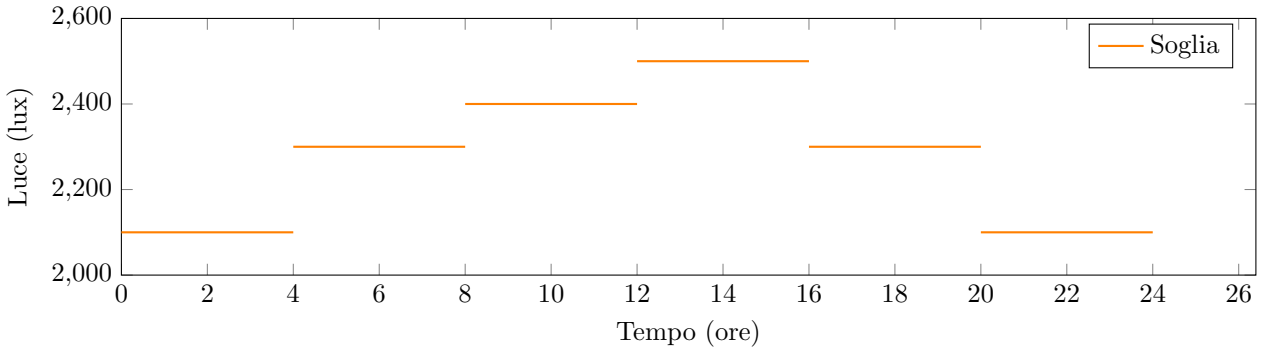


Figure 17: Architettura del sistema

3.2 Progettazione di dettaglio

Si vuole qui descrivere più dettagliatamente il concetto di preset e di stadio di crescita di una pianta, al fine di giustificare le scelte progettuali fatte.
Un preset viene definito come un insieme ordinato di fasi, dette stadioCrescita, ognuna delle quali descrive quali dovranno essere le soglie per ogni parametro della serra in un intervallo di tempo dalla durata ben definita nella fase stessa. Lo scopo è rappresentare la crescita di una pianta come una funzione nel tempo per ogni parametro ambientale.





Questi grafici mostrano un esempio di preset per la temperatura e la luce (tali osservazioni vengono estese a tutti i restanti parametri), in cui la durata di vita della pianta (supposta di 24 ore) è stata suddivisa in 6 stadi da 4 ore. In ogni fase sono definite soglie per ogni parametro, rendendo la configurazione della serra personalizzabile in base alle necessità.
Il preset manterrà informazioni sulla fase corrente e verrà aggiornata da un evento che gli permetterà di tener conto del passare del tempo.

Si vuole porre inoltre particolare risalto ai resoconti, che descriveranno l'andamento della serra in specifici range temporali (noi abbiamo considerato range settimanali e giornalieri). In particolare, il ResocontoGiornaliero avrà informazioni quali:

- variazione oraria tra parametri ricevuti dai sensori e valori di soglia ideali impostati nel preset;
- attivazione media degli attuatori (passaggio da on/off e viceversa)

Il ResocontoSettimanale invece avrà informazioni su:

- giorno della settimana più dispendioso in termini di efficienza energetica (giorno in cui l'attivazione/disattivazione è stata più frequente);

Tali informazioni descritte rappresentano solo un punto di partenza per una futura espansione, nella quale ogni resoconto potrà descrivere analisi più complesse, anche al fine di predirre l'andamento della serra basandosi su informazioni storiche.
Infine, il report verrà anche generato in un formato scaricabile (es. file pdf) che conterrà tutte le analisi effettuate.

3.2.1 Struttura: Dominio

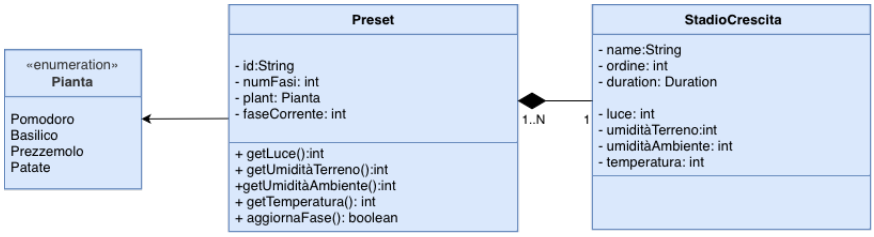


Figure 18: Dominio: Preset

I vari metodi getter nel Preset restituiranno i valori di soglia tenendo presente lo stadioCrescita corrente, il cui stato verrà salvato in faseCorrente.

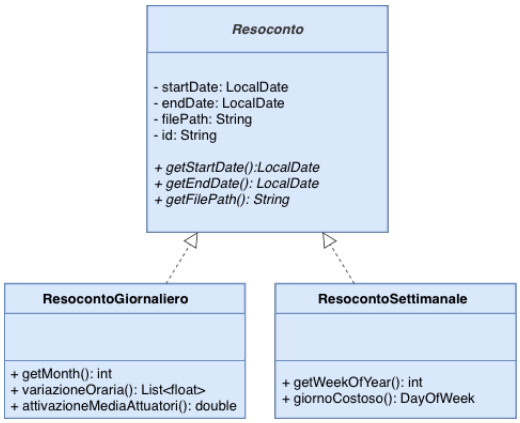


Figure 19: Dominio: Resoconto

La classe astratta Resoconto viene concretizzata da ResocontoGiornaliero e ResocontoSettimanale i quali permettono di rappresentare i risultati delle analisi che vengono effettuate dalle classi del Controller (i.e. variazione oraria dei parametri, attivazione media, ...). Ogni resoconto avrà un percorso dove verrà salvato il file generato contenente le analisi formattate correttamente.

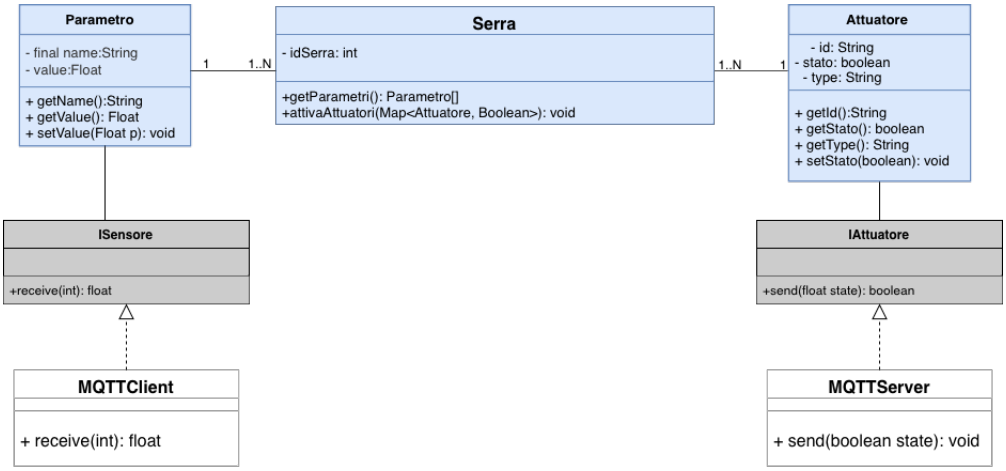


Figure 20: Dominio: Serra

La serrra salva lo stato degli attuatori e dei parametri, in particolare si interfaccia con i sensori e gli attuatori tramite le interfacce ISensore e IAttuatore. Tali interfacce si preoccuperanno di gestire la comunicazione con il microcontrollore, garantendo anche che la comunicazione sia cifrata. Abbiamo qui concretizzato tali interfacce con delle classi legate alle fase di implementazione, e si basano su MQTT, un protocollo utilizzato per inviare e ricevere dati crittografati e semplici da analizzare. Tuttavia le interfacce sono pensate per offrire le funzionalità fondamentali a chi le utilizza, dunque sarà possibile estenderle con specifiche classi concrete coerentemente con le necessità progettuali future.

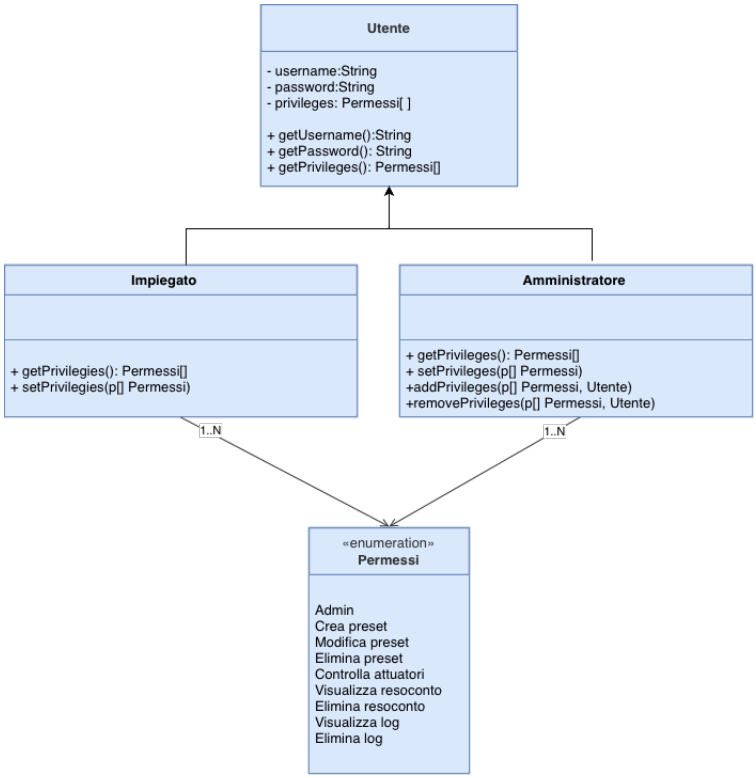


Figure 21: Dominio: Utente

La classe Utente viene estesa da Impiegato e Amministratore, il quale ha funzionalità come aggiungere e rimuovere permessi ad un generico Utente.

3.2.2 Struttura: Controller

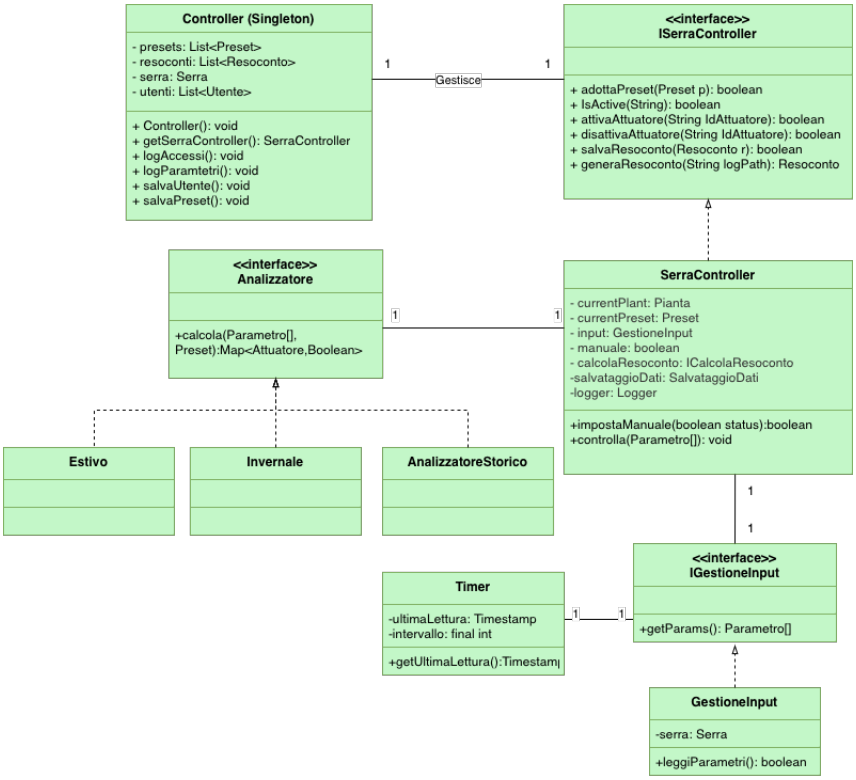


Figure 22: Controller: Elaborazione

La classe SerraController è gestita tramite il design pattern Singleton, in quanto si vuole garantire che sia presente un'unica istanza che permetta l'accesso e la modifica ai dati del model, che sono condivisi da più classi, in modo da garantirne la consistenza e sincronizzazione. Si è ritenuto importante inserire una interfaccia di serraController, per evitare rigidità, fragilità e immobilità. Essa ha dei metodi per cambiare stato da manuale ad automatico e per effettuare il controllo che porterà alla possibile accensione degli attuatori.

La classe Analizzatore ha lo scopo di implementare la logica di calcolo per definire quali attuatori dovranno essere attivati. Tale logica potrà essere più o meno complessa, nelle prime fasi abbiamo supposto solo delle logiche basate sul periodo dell'anno o su analisi partendo da risultati passati per predirre più accuratamente l'eventuale accensione degli attuatori. Tale classe segue il design pattern strategy.

La classe GestioneInput si occupa di leggere i parametri dai sensori (attraverso il model) e fornirli a SerraController per la gestione automatica. La lettura avverà periodicamente, scandita da un Timer, dando vita ad un pattern di tipo Observer, dove il timer è il soggetto e gestioneInput l'observer.

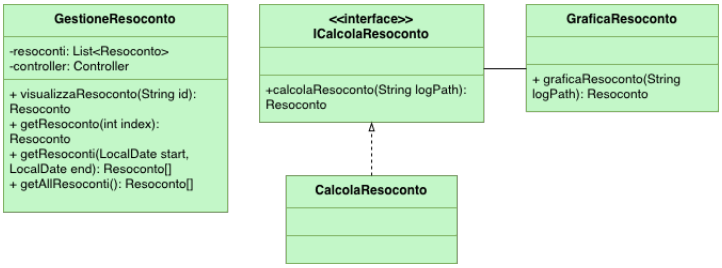


Figure 23: Controller: Resocontazione

CalcolaResoconto si occupa di calcolare tutti i valori quantitativi utili per la resocontazione, generando un'istanza di Resoconto. Esso può inoltre generare un resoconto scaricabile (es. pdf) utilizzando la classe GraficaResoconto, la quale permette inoltre di realizzare i diagrammi ed altri elementi grafici presenti nei report.

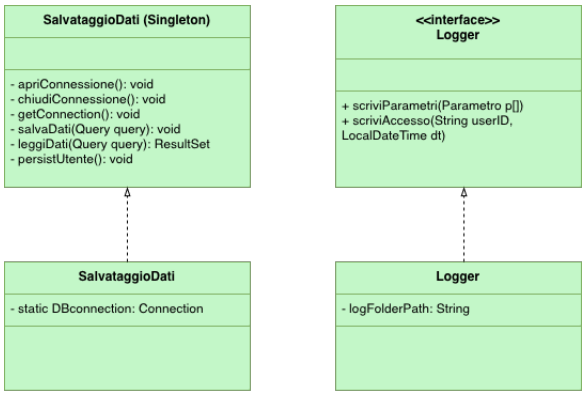


Figure 24: Controller: Persistenza

La classe salvataggioDati incapsula la connessione con il database ed i metodi CRUD sui dati. Nonostante i moderni DBMS siano in grado di gestire transazioni concorrenti con diversi livelli di isolamento, abbiamo adotatto il pattern singleton per istanziare un'unica connessione con il database ed avere un unico punto di accesso per tutte le classi. Difatto il presente sistema non è "data intensive" perché i dati memorizzati sono soggetti a poche letture o scritture, pertanto sequenzializzare le operazioni non dovrebbe degradare le performance complessive.

3.2.3 Struttura: Autenticazione e View

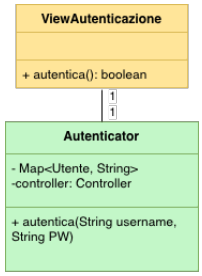


Figure 25: View e autenticazione

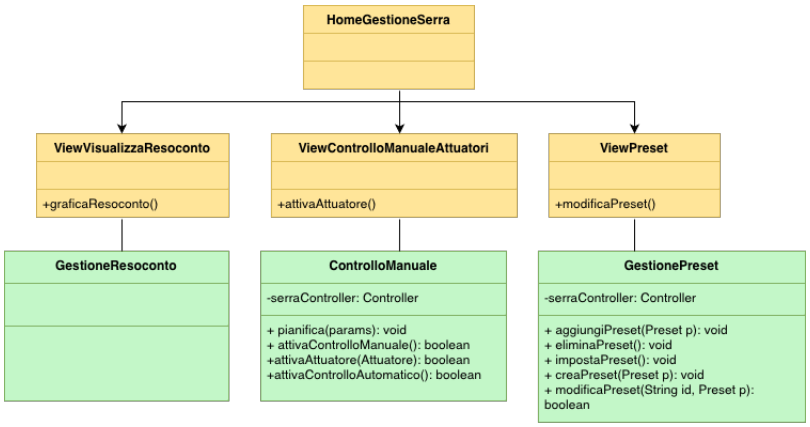


Figure 26: View e controller: Home gestione serra

ControlloManuale permette di gestire manualmente la serra quando possibile, il controller verificherà se ci sono le condizioni ottimali per effettuare il passaggio automatico-manuale e viceversa.
GestionePreset permette di aggiungere/rimuovere/modificare dei preset se lecito, in quanto il controller impedirà di eliminare il preset attualmente in uso e simili operazioni inconsistenti.

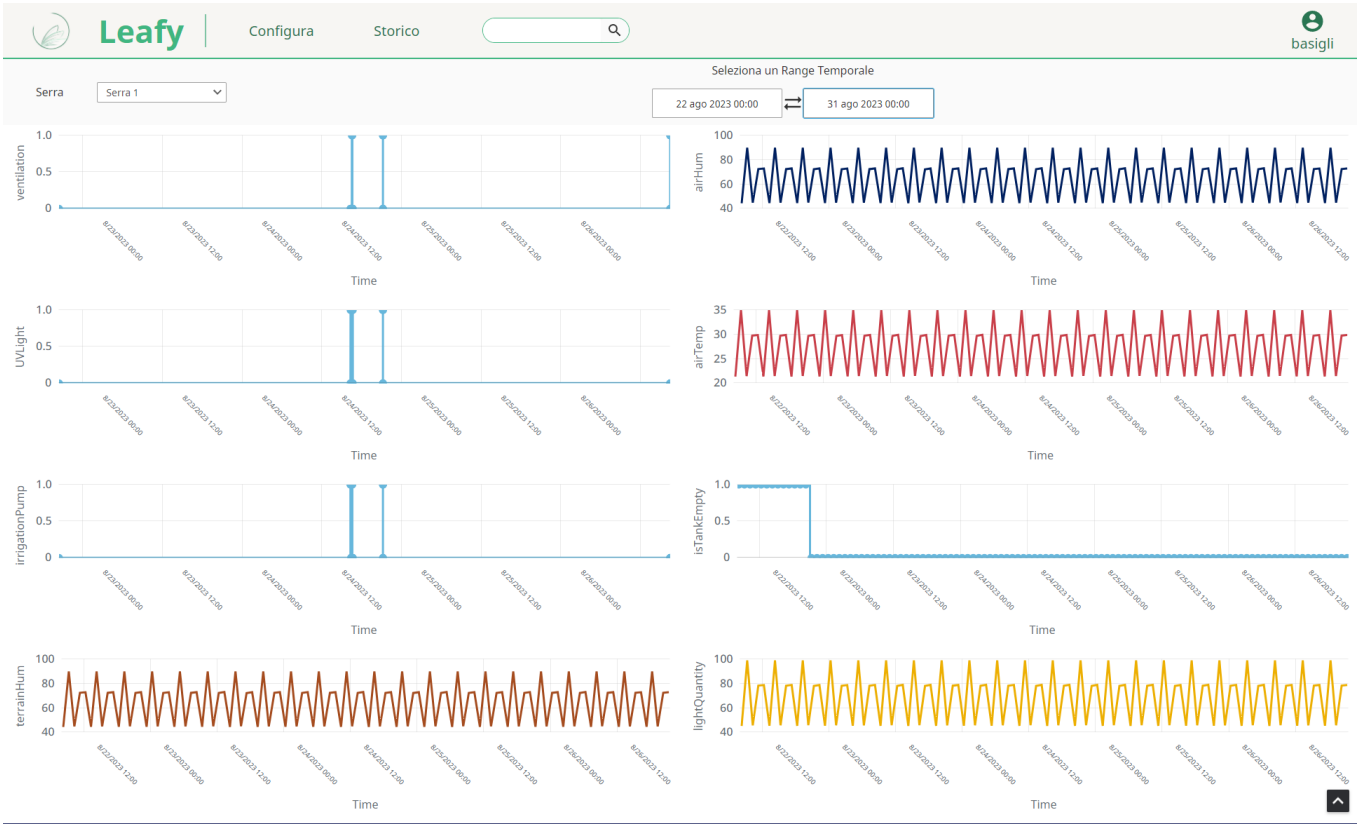


Figure 27: Esempio di view per la visualizzazione di un resoconto

Figure 28: Esempio di view per il controllo manuale attuatori

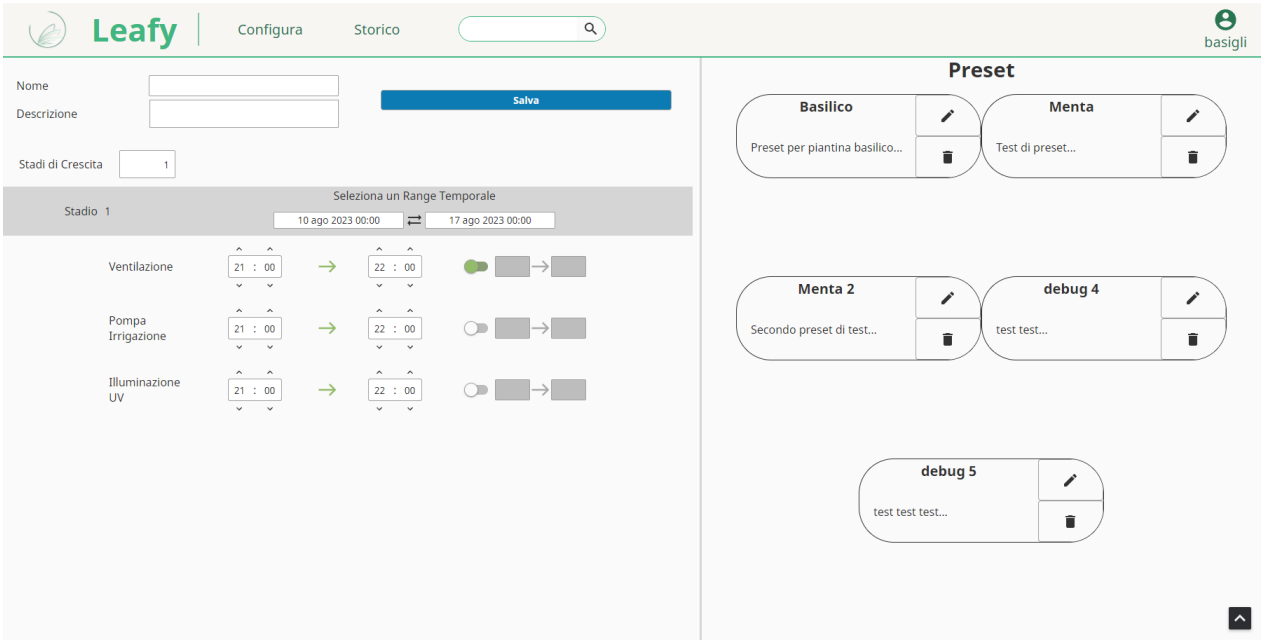


Figure 29: Esempio di view per la gestione preset

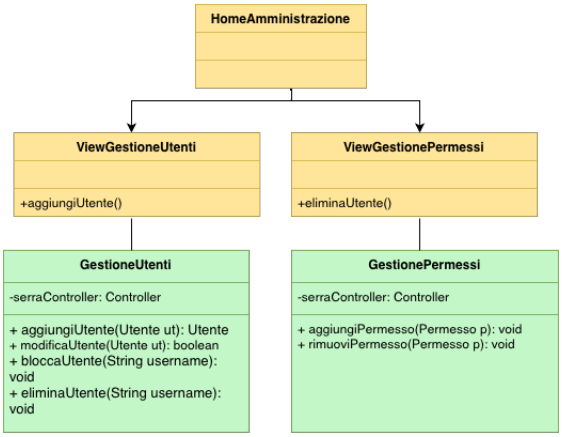


Figure 30: View e controller: Home amministrazione

GestioneUtenti permette ad utenti con i giusti permessi di modificare/creare/eliminare utenti, mentre GestionePermessi permette di modificare i permessi di tali utenti.

3.2.4 Interazione

Si vuole qui dettagliare l'interazione che si avrà con i sensori e gli attuatori, tenendo ora in considerazione della gestione dell'unica istanza di controller da parte del singleton.

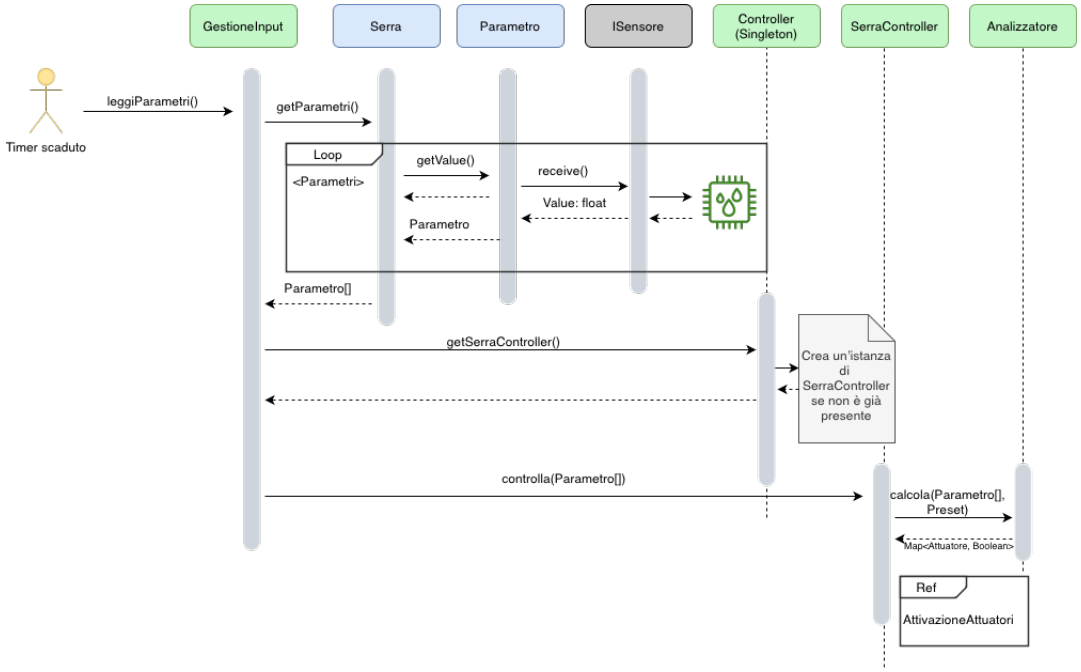


Figure 31: Interazione

Nei diagrammi è stata data importanza all'interazione con i sistemi esterni, tralasciando il salvataggio dati, le cui interazioni sono poco variabili rispetto al diagramma di analisi, e altre interazioni tra varie classi del model (la serra gestisce informazioni sul tempo trascorso, al fine di modificare quando necessario lo stadio corrente del preset) che renderebbero il diagramma meno leggibile.

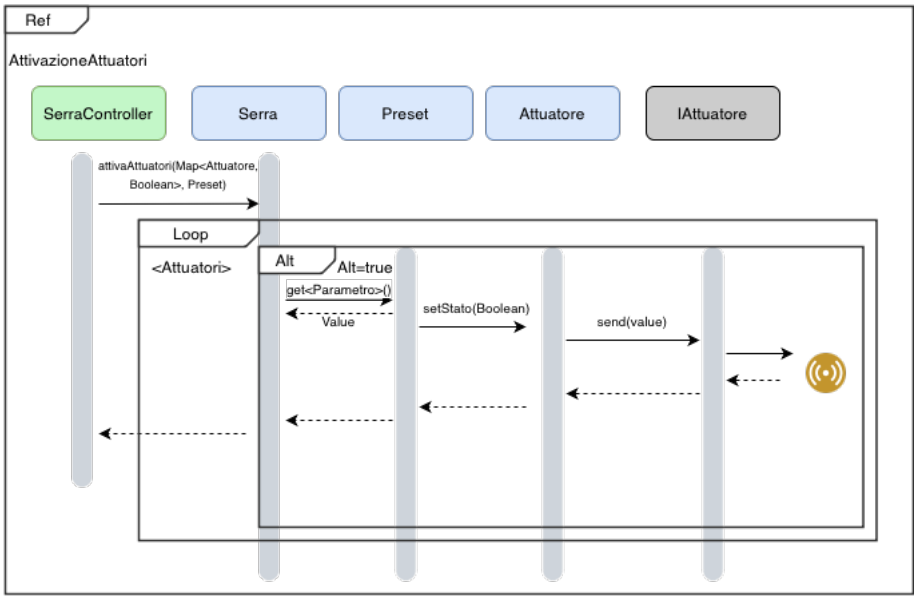


Figure 32: Interazione

3.2.5 Comportamento

Il comportamento non presenta sostanziali differenze rispetto alla fase di analisi, motivo per il quale non è stato riportato qui.

3.3 Progettazione della persistenza

3.3.1 Database

Le entità che sono state definite sono:

- *Preset*: Modella ogni preset definito dall'utente
- *StadioCrescita*: Dettaglia il preset con informazioni aggiuntive; Ogni preset può contenere in generale più stadi.
- *Parametro*: Modella le informazioni lette dai sensori. Contiene un timestamp che definisce l'istante temporale in cui è avvenuta la misurazione. Tali informazioni sono di fondamentale importanza per la resocontazione.
- *Attuatore*: Modella le informazioni riguardo l'attuazione degli attuatori. Viene definito un timestamp che descrive l'istante temporale in cui è avvenuto il controllo.
- *Permesso*: Definisce l'insieme dei permessi che ogni utente può avere.
- *Utente*: Definisce l'insieme degli utenti, ognuno dei quali può possedere più permessi. Si ricorda inoltre che le password vengono salvate dopo aver effettuato un hashing su di esse, per evitare che una eventuale fuga di dati sia efficace.

In ogni entità è definita una chiave primaria con un numero progressivo.

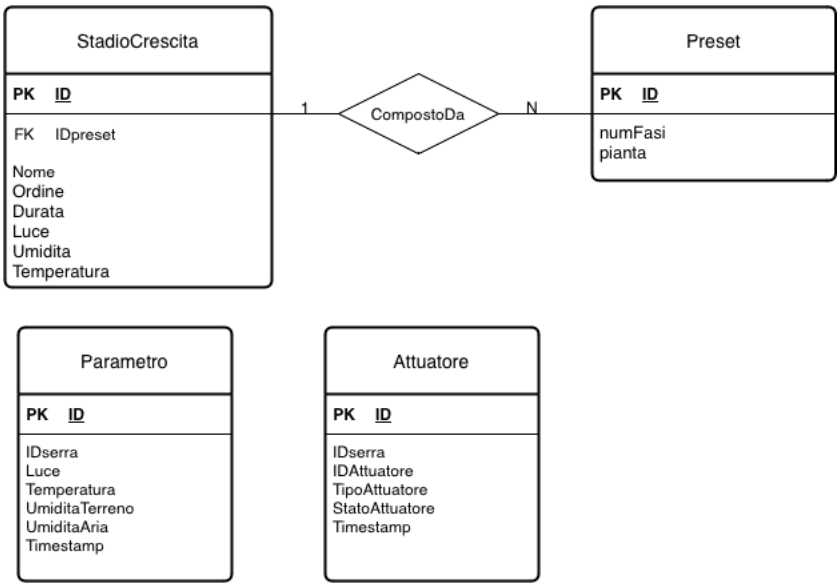


Figure 33: Persistenza

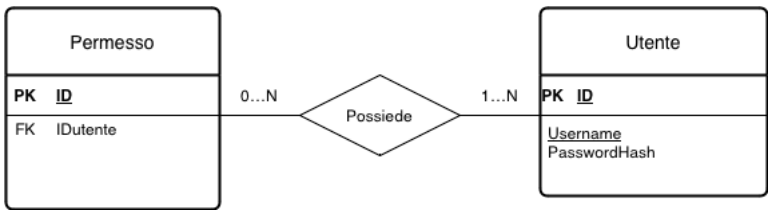


Figure 34: Persistenza

3.3.2 Log

Vengono salvati dei log per ottenere informazioni sugli accessi e sulle operazioni effettuate dagli utenti. Il formato dei log è il seguente:

- *Log accessi*: <timestamp><username>
- *Log operazioni*: <timestamp><tipoOperazione><tipoOggetto>

Viene inoltre salvato, quando richiesto, un resoconto. Esso potrà essere settimanale o giornaliero e in generale il formato sarà: *resoconto<Tipo><ID>.<formato>*

3.4 Progettazione del collaudo

Vengono qui riportati i principali test per collaudare l'applicazione, dando maggior rilievo ai punti più importanti del sistema.

Listing 4: Preset Test

```
public class PresetTest{

@Before
public static void initPreset(){
    Pianta p = Pianta.POMODORO;
    List<StadioCrescita> stadi = new ArrayList<StadioCrescita>();

    //creo 10 stadi crescita della durata di 24 ore ciascuno assegnando valori fittizi alle
        soglie
    for(int i=0; i<10; i++)
        stadi.append(new StadioCrescita(i,Duration.ofHours(24),12000, 30, 30, 23+(i*0.3)));

    Preset mypreset = new Preset("Estivo",p,stadi);

}

@Test
public static void presetPiantaEParametri(){

    assertEquals(mypreset.getPianta(), Pianta.POMODORO);
    assertEquals(mypreset.getLuce(), 12000);
    assertEquals(mypreset.getTemperatura(), 23);

}

@Test
public static void presetFaseENumFasi(){

    mypreset.aggiornaFase();
    assertEquals(mypreset.getFaseCorrente(),1);
    assertEquals(mypreset.getNumFasi(), 10);

}


}
```

Listing 5: SerraController Test

```
public class SerraControllerTest{

@Before
public static void initController(){
```

```
        Pianta p = Pianta.POMODORO;
        List<StadioCrescita> stadi = new ArrayList<StadioCrescita>();

        //creo 10 stadi crescita della durata di 24 ore ciascuno assegnando valori fittizi alle
            soglie
        for(int i=0; i<10; i++)
            stadi.append(new StadioCrescita(i,24,12000, 30, 30, 23+(i*0.3)));

        Preset mypreset = new Preset("Estivo",p,stadi);
        IGestioneInput gestioneInput;
        Analizzatore analizzatore = new Estivo();

        ISerraController serra = new SerraController(p,mypreset, gestioneInput, false,
            analizzatore);

    }

    @Test
    public static void impostaManualeTest(){

        serra.impostaManuale(true);
        assertTrue(serra.getManuale());

    }

}
```

Listing 6: Controller Singleton Test

```
    public class ControllerSingletonTest(){

    @Before
    public static void initController(){

        Pianta p = Pianta.BASILICO;
        List<StadioCrescita> stadi = new ArrayList<StadioCrescita>();

        //creo 10 stadi crescita della durata di 24 ore ciascuno assegnando valori fittizi alle
            soglie
        for(int i=0; i<10; i++)
            stadi.append(new StadioCrescita(i,24,12000, 30, 30, 23+(i*0.3)));

        Preset mypreset = new Preset("Estivo",p,stadi);
        IGestioneInput gestioneInput;
        Analizzatore analizzatore = new Estivo();

        ISerraController serra = new SerraController(p,mypreset, gestioneInput, false,
            analizzatore);
        Controller c;
        Controller.Controller(serra);
```

```
    }

    @Test
    public static void istanzaTest(){
        ISerraController serra2 = new SerraController(p,mypreset, gestioneInput, false,
            analizzatore);
        c.Controller(serra2);

        //assegnando ad un controller un'istanza diversa di serraController, essa non deve essere
            salvata se ne esiste
        //gia' una
        AssertEquals(c.getSerraController(), serra);
    }

    @Test
    public static void addPreset(){

        Preset mypreset2 = new Preset("Invernale",p,stadi);

        c.addPreset(mypreset2);

        assertEquals(c.getPresets().get(0), mypreset);
        assertEquals(c.getPresets().get(1), mypreset2);
    }
}
```

Listing 7: Resoconto Test

```
    public class ResocontoTest(){

    @Before
    public static void initResoconto(){

        Controller contr;
        IcalcolaResoconto calres = new CalcolaResoconto();
        GestioneResoconto gestres = new GestioneResoconto(c);
    }

    @Test
    public static void typeResoconto(){

        assertTrue( calres.calcolaResoconto("res") is Resoconto);
    }

    @Test
    public static void typeResoconti(){
        List<Resoconto> res = gestres.getResoconti(LocalDate.of(2023,2,2), LocalDate.of(2023,2,15));

        if( res != null )
            assertTrue(res.get(0) is Resoconto);
```

```
    }
}
```

Listing 8: Analizzatore Test

```
    public class AnalizzatoreTest{

    @Before
    public static void initAnalizzatore(){

        Pianta p = Pianta.POMODORO;
        List<StadioCrescita> stadi = new ArrayList<StadioCrescita>();

        //creo 10 stadi crescita della durata di 24 ore ciascuno assegnando valori fittizi alle
            soglie
        for(int i=0; i<10; i++)
            stadi.append(new StadioCrescita(i,24,12000, 30, 30, 23+(i*0.3)));

        Preset mypreset = new Preset("Estivo",p,stadi);
    }

    @Test
    public static void estivoTest(){

        Analizzatore estivo = new Estivo();
        Serra serra = new Serra("0101");
        Map<Attuatore, boolean> mymap = estivo.calcola(serra.getParametri(), mypreset);

        AssertEquals(mymap.get(serra.getAttuatore("irrigatore")), true);
        AssertEquals(mymap.get(serra.getAttuatore("lampada")), false);

    }

    @Test
    public static void InvernaleTest(){

        Analizzatore invernale = new Estivo();
        Serra serra = new Serra("0101");
        Map<Attuatore, boolean> mymap = invernale.calcola(serra.getParametri(), mypreset);

        AssertEquals(mymap.get(serra.getAttuatore("irrigatore")), false);
        AssertEquals(mymap.get(serra.getAttuatore("lampada")), true);

    }
}
```

3.5 Progettazione del deployment

3.5.1 Artefatti

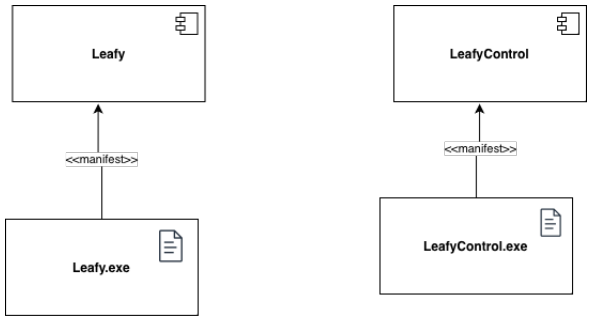


Figure 35: Artefatti

3.5.2 Deployment type-level

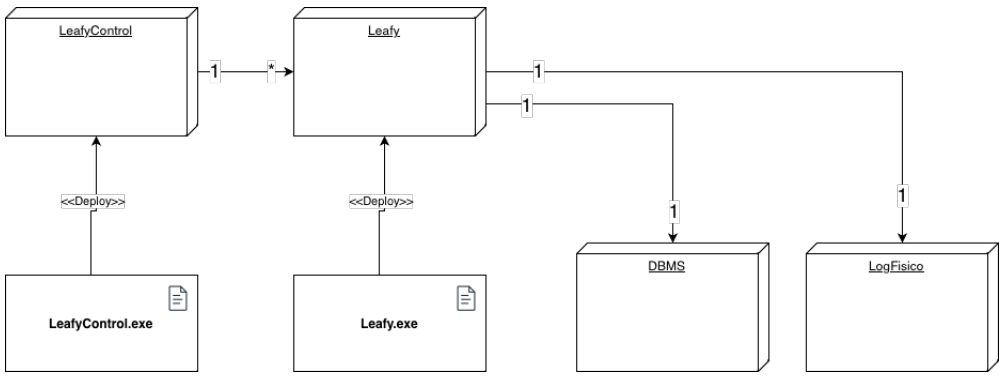


Figure 36: Deployment type-level

4 Implementazione

Si vogliono qui dettagliare le scelte implementative che sono state fatte nell'interazione tra il sistema e il microcontrollore. Il microcontrollore scelto è un arduino che utilizza MQTT per inviare e ricevere dati dal sistema. È stato scelto MQTT poichè un protocollo che utilizza poca banda per inviare e ricevere informazioni e pensato per funzionare con dispositivi remoti leggeri. Inoltre è possibile implementare una versione sicura (MQTTS) che si appoggia su un canale TLS per garantire cifratura end to end. Viene utilizzato un modulo WiFi che permette di trasferire informazioni via etere.
MQTT è di tipo publish/subscribe, con un broker che riceve i messaggi e li inoltra. Nel nostro caso abbiamo due situazioni che si possono verificare:

- *Arduino vuole inviare informazioni sui parametri letti al sistema*: qui Arduino sarà il publisher, mentre il sistema farà da subscriber.

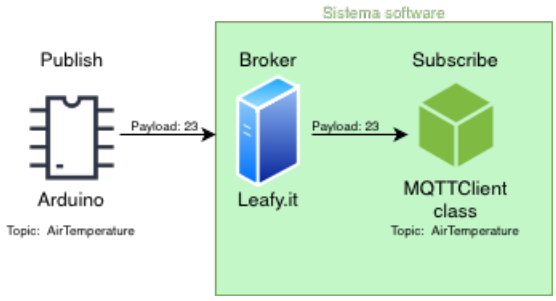


Figure 37: Esempio di interazione nel quale arduino legge la temperatura dal sensore e la invia al sistema

- *Il sistema vuole inviare ad arduino informazioni su un attuatore da attivare*: qui sarà il sistema ad essere publisher, mentre Arduino sarà subscriber.

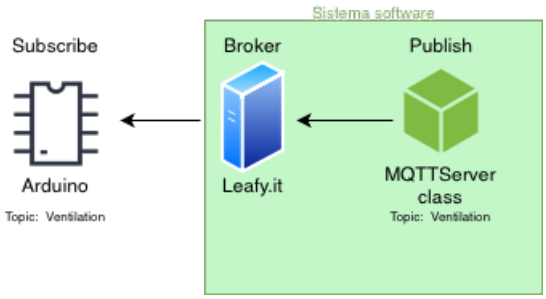


Figure 38: Esempio di interazione nel quale il sistema vuole attivare l'attuatore di ventilazione

MQTT funziona con dei topics ai quali il subscriber può "iscriversi" per ricevere i suoi specifici messaggi; nel nostro caso abbiamo definito dei topics per leggere dai sensori e dei topics per attivare gli attuatori.

Listing 9: Codice arduino che descrive un esempio di interazione MQTT

```
//crea un client WiFi e lo collega al client MQTT
WiFiClient wifiClient;
MqttClient mqttClient(wifiClient);

...

//il broker e' il sistema software che agisce da server -> leafy.it:1883
const char broker[] = "leafy.it";
int       port    = 1883;

//topics ai quali il sistema si iscrive per ricevere informazioni dai sensori
const char topic1[] = "1.AirHumidity";
const char topic2[] = "1.AirTemperature";
const char topic3[] = "1.IsTankEmpty";
const char topic4[] = "1.LightQuantity";
const char topic5[] = "1.TerrainHumidity";

//topics ai quali arduino si iscrive per capire quando attivare un attuatore
const char topic6[] = "1.IrrigationPump";
const char topic7[] = "1.UVLight";
const char topic8[] = "1.Ventilation";

void setup() {

...

  //connessione al client MQTT
  if (!mqttClient.connect(broker, port)) {
    Serial.print("MQTT connection failed! Error code = ");
    Serial.println(mqttClient.connectError());

    while (1);
  }

  //associazione di una funzione di handler che capisce quale attuatore attivare quando arriva un
      messaggio di attivazione di un attuatore
  mqttClient.onMessage(onMqttMessage);

  //arduino si iscrive ai soli topic che gli interessano in posizione di subscriber
  mqttClient.subscribe(topic6);
  mqttClient.subscribe(topic7);
  mqttClient.subscribe(topic8);

}

void loop(){

  //tale funzione controlla continuativamente che sia passato sufficiente tempo e poi pubblica un
```

```
      messaggio per ogni parametro letto
  readAndSendParams();

}

void readAndSendParams() {

  ...

  mqttClient.poll();

  //ogni "interval" millisecondi il client MQTT pubblica dei messaggi sui parametri che ha letto
      (il sistema software sara' subscriber su questi topics)
  if (currentMillis - previousMillis >= interval) {

    ...

    mqttClient.beginMessage(topic1);
    mqttClient.print(HUM);
    mqttClient.endMessage();

    mqttClient.beginMessage(topic2);
    mqttClient.print(TEMP);
    mqttClient.endMessage();

    mqttClient.beginMessage(topic3);
    mqttClient.print(isTankEmpty);
    mqttClient.endMessage();

    mqttClient.beginMessage(topic4);
    mqttClient.print(lightQuantity);
    mqttClient.endMessage();

    mqttClient.beginMessage(topic5);
    mqttClient.print(terrainHum);
    mqttClient.endMessage();

  }


}

void onMqttMessage(int messageSize) {

  //ricevuto un messaggio, si capisce di che topic si tratta e si attiva l'attuatore corrispondente
  const char topic[] = mqttClient.messageTopic()

  ...

  if (topic == topic6) {
```

```
    pin = pinPump;
  }
  else if (topic == topic7) {
    pin = pinUVLight;
  }
  else if (topic == topic8) {
    pin = pinVentilation;
  }


  while (mqttClient.available()) {
    bool value = (bool) mqttClient.read();
    digitalWrite(pin, value);
  }
}
```

Inseriamo infine un'immagine del prototipo di serra creato.