# Assignment #03 SmartGarden

La soluzione progettata e implementata è suddivisa in cinque sottosistemi, come richiesto dal testo del problema. Di seguito verranno illustrate brevemente le tecnologie e le strategie utilizzate per ogni sottosistema.

## Garden Service

Il garden service rappresenta il fulcro centrale del sistema e presenta le seguenti connessioni con gli altri sottosistemi:

* Garden Dashboard tramite http
* Garden Sensorboard tramite http
* Garden controller tramite comunicazione seriale

Il Garden Service è stato realizzato in Java, utilizzando la libreria vertex per rendere disponibile il server alla dashboard e alla sensorboard sulla porta 8080. In generale il suo comportamento presenta due parti fondamentali, la prima consiste in una reazione alle richieste http (POST per la sensorboard e GET per la dashboard) per ricevere i dati delle misurazioni e per condividere alla dashboard lo stato del sistema tramite oggetti JSON. La seconda invece consiste nell’inviare comandi al garden controller tramite seriale e ricevere da esso informazioni sul sistema. L’invio di comandi al garden controller avviene tramite la composizione di messaggi nel seguente formato: 1|1|1|1|-1|-1|-1. I valori contenuti tra i separatori rappresentano dei comandi per il garden controller, in particolare i primi due sono per i led digitali, il secondo e il terzo per i led analogici, il quarto e il quinto per l’accensione e la velocità del sistema di irrigazione e l’ultimo per la modalità.

Il comportamento generale del Garden Service può essere riassunto come segue: esso riceve le misurazioni, se il sistema è in modalità automatica calcola gli opportuni settaggi per il sistema delle luci e di irrigazione e invia di conseguenza i comandi al garden controller, contemporaneamente valuta se il sistema è nelle condizioni per entrare in allarme e infine mantiene aggiornata la dashboard.

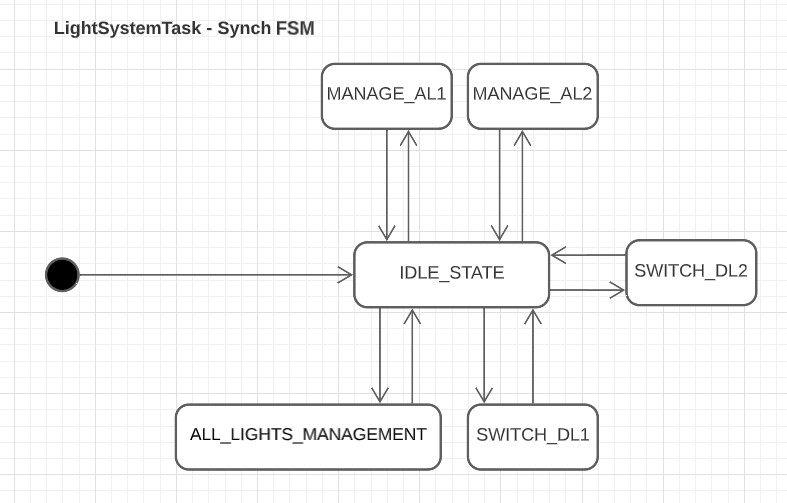
## Garden controller

Il garden controller rappresenta la parte contenente gli attuatori del sistema, esso ha due principali comunicazioni con gli altri sottosistemi:

* Garden Service tramite comunicazione seriale
* Garden App tramite Bluetooth

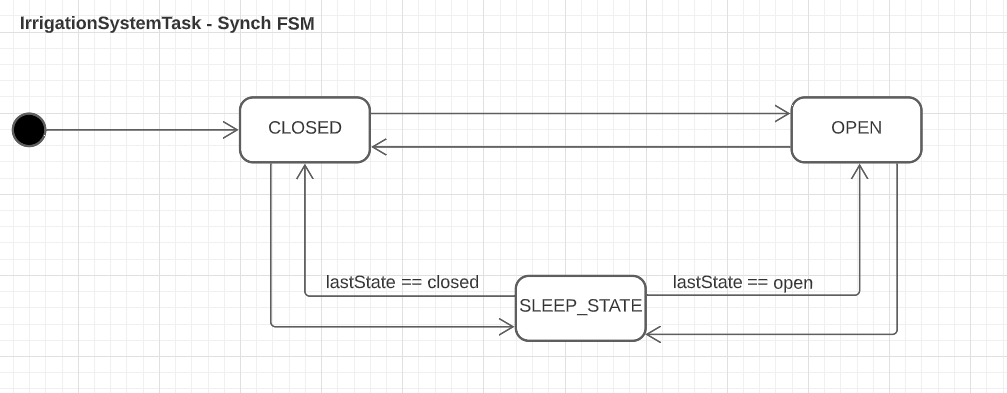
Questo sottosistema è stato organizzato in Task, in particolare sono stati individuati quattro Task a cui corrispondono quindi quattro FSM sincrone. I Task cooperano tramite l’utilizzo di variabili condivise che vengono alterate dai comandi automatici provenienti dal service oppure da quelli manuali provenienti dall’app.

**LightSystemTask**

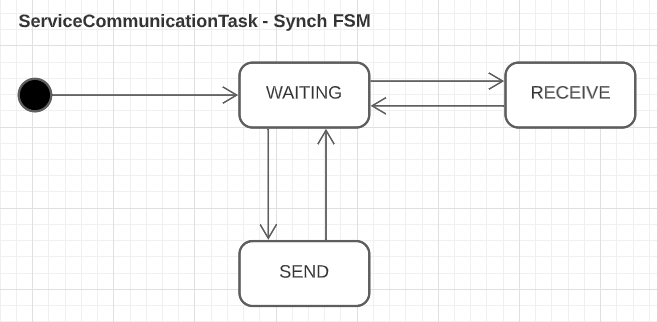


Questo task si occupa del sistema di gestione delle luci. Presenta uno stato centrale (IDLE\_STATE) in cui valuta le guardie per eventuali cambi di stato. Presenta inoltre per ogni led uno stato di gestione singolo, utilizzato tipicamente durante il controllo manuale e infine uno stato per la gestione complessiva di tutti i led utilizzato durante la modalità automatica del sistema.

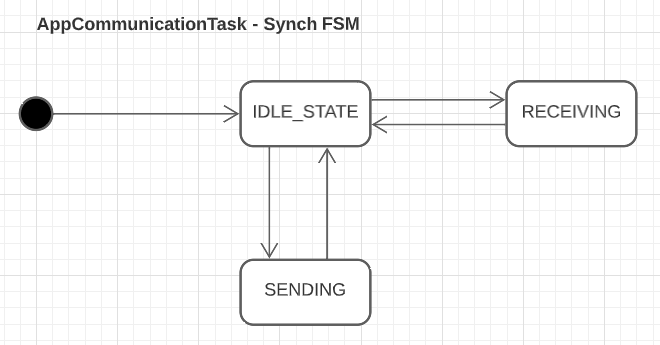
**IrrigationSystemTask**

****

Questo task incapsula la gestione del sistema di irrigazione. Presenta tre stati fondamentali: OPEN, CLOSED e SLEEP\_STATE. Il task permette quindi di aprire e chiudere il sistema e modificarne la velocità sia quando è in funzione che quando è spento. Sia dallo stato OPEN che da CLOSED il sistema può entrare in una condizione di sleep in cui non è utilizzabile e una volta tornato in funzione effettua una transizione verso l’ultimo stato in cui era prima di entrare in sleep.

**ServiceComunicationTask**

Questo task si occupa della comunicazione seriale con il Garden Service, ha una struttura tipica di un task di comunicazione, quindi può essere in WAITING, oppure inviare messaggi in SEND o infine riceverli in RECEIVE. Quando riceve un pacchetto dal service, estrae i comandi e li salva nelle variabili condivise, ciò porterà gli altri task a modificare il loro comportamento di conseguenza. Se riceve come comando -1 significa che non deve alterare la variabile corrispondente.

**AppComunicationTask**

Questo task incapsula la comunicazione con la Garden App. Strutturalmente è molto simile al precedente task di comunicazione, presenta quindi i classici tre stati IDLE\_STATE, SENDING e RECEIVING. Ha il medesimo comportamento in ricezione del ServiceComunicationTask, ma riceve i comandi manuali dall’applicazione tramite Bluetooth.

## Garden sensorboard

Questo sottosistema si collega tramite http al Garden service ed effettua due richieste http, una POST per inviare al service i dati delle misurazioni di luce e temperatura e una GET per ricevere dal service la modalità corrente del sistema, potendo gestire così il led di conseguenza.

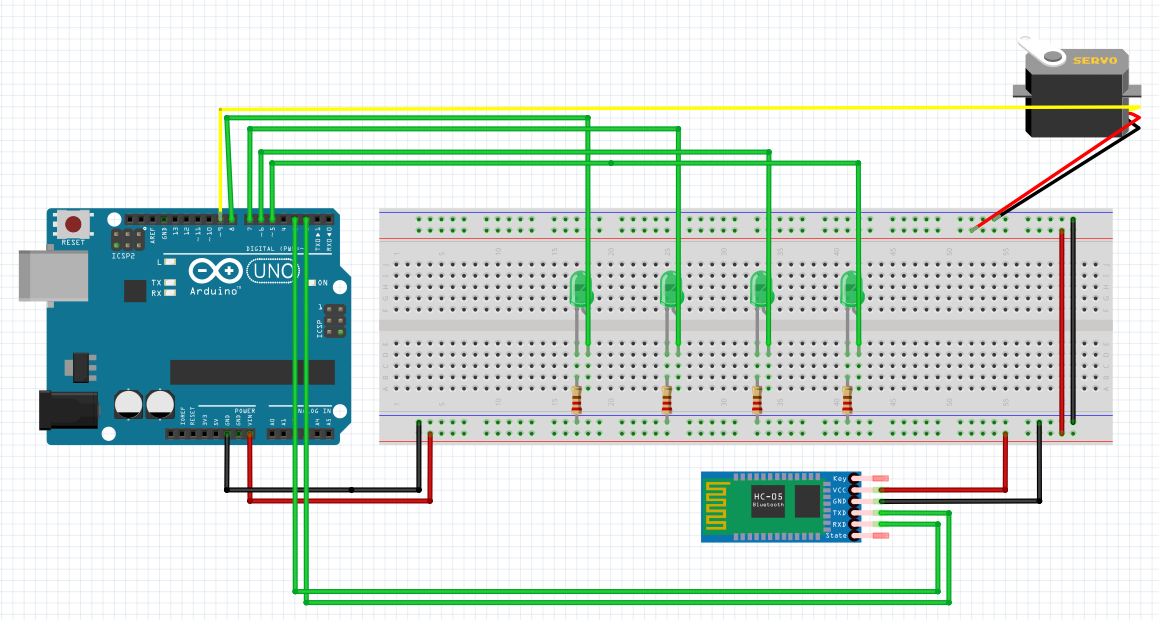
## Garden app

L’applicazione comunica solamente con il garden controller tramite una connessione bluetooth, che viene instaurata al click dell’apposito pulsante per richiedere il controllo manuale. Tramite i pulsanti all’interno dell’app vengono mandati dei pacchetti con lo stesso formato di quelli inviati dal service, essi contengono comandi che vengono letti dal controller e provocano gli effetti desiderati sugli attuatori. Contemporaneamente anche l’app riceve dal controller pacchetti che hanno lo scopo di aggiornare la UI con lo stato corrente del sistema.

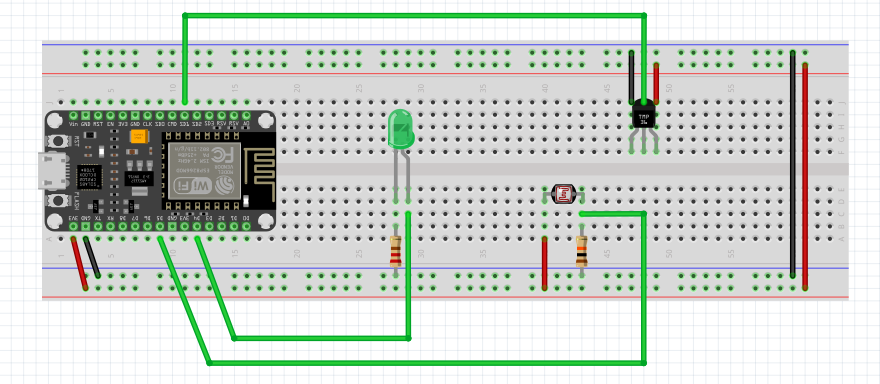
## Garden dashboard

Questo è sicuramente il sottosistema più semplice, riceve dal service tramite richieste http GET le informazioni sullo stato del sistema e le mostra in output tramite una semplice GUI.

## Circuito



*Garden Controller*



*Garden Sensorboard*