Smart Bridge

Progetto Numero #2 del corso:
Embedded Systems and Internet of Things.

Nicola Montanari

Nicola.Montanari14@studio.unibo.it

0000970119

Novembre 2022

Chapter 1

Descrizione del progetto

Lo scopo di questo progetto era quello di costruire un prototipo che andasse a simulare la gestione di un ponte intelligente.
Il ponte è posizionato sopra un fiume ed è caratterizzato da due diverse funzionalità autonome:

- La prima funzionalità si basa sul monitoraggio del livello dell'acqua e la gestione di situazioni potenzialmente pericolose (livello dell'acqua troppo alto) andando a manovrare in modo automatico una valvola che possa far defluire l'acqua in eccesso in zone controllate.
- La seconda funzionalità si basa su una illuminazione automatica intelligente. Ovvero l'accensione e lo spegnimento delle luci del ponte in presenza di una o piu persone che attraversano quest'ultimo.

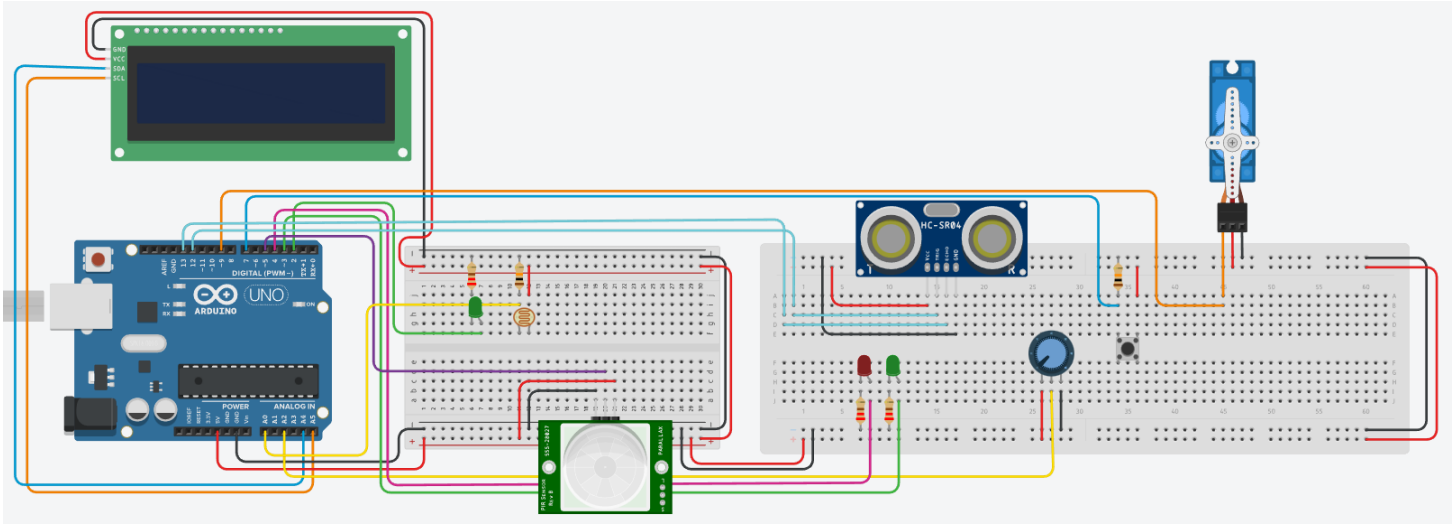
Ciascuna di queste funzionalità deve essere programmata e gestita in modo atomico.

Chapter 2

Schema Elettrico

Il prototipo di questo progetto è stato sviluppato con dei componenti hardware che simulano i sensori che verranno poi allocati sopra il ponte.

Questo è lo schema elettrico utilizzato nella creazione del prototipo:



2.1 Componenti hardware

A seguito un elenco di ciascun componente hardware con una piccola descrizione di ciascuno di essi.

- Sonar
  - Sensore che rileva la distanza tra il ponte e il livello superficiale dell'acqua.
- LCD
  - Schermo che informa i passanti dello stato attuale del ponte, del livello dell'acqua e i gradi di apertura della valvola.
- PIR
  - Sensore di movimento per passanti.
- Servo Motore
  - Valvola per far defluire l'acqua in eccesso.
- Sensore di luminosità
  - Verifica che sia buio per attivare o meno le luci per i passanti.
- Pulsante
  - Un operatore può premere questo pulsante per attivare la modalità manuale e gestire la valvola a piacimento.
- Potenziometro
  - Un operatore manuale può aprire la valvola a piacimento usando questo sensore dopo aver attivato la modalità manuale.
- Vari tipi di Illuminazione
  - Un led di colore verde simula le luci per i passanti.
  - Un led di colore verde e uno di colore rosso indicano lo stato attuale del ponte.

Chapter 3

Funzionamento Programma Arduino

Uno degli scopi principali lato programmazione era quello di gestire le varie funzionalità in modo atomico e concorrente.
Ovvero entrambe le funzionalità dovevano operare nello stesso periodo e ciascuna di essa non doveva influenzare il funzionamento dell'altra.
Per garantire tutto ciò si è utilizzata una programmazione orientata ad oggetti oltre alla suddivisione dell'intero programma in task.
La gestione delle varie task è stata poi affidata a uno scheduler Multi-tasking, un processo che analizza e sceglie quali task eseguire in base alle varie situazioni.

Grazie a queste scelte, il programma risulterà molto piu semplice da programmare, da gestire e da riusare.

3.1 Organizzazione ad oggetti

Per il corretto funzionamento del programma e per massimizzare la riusabilità, è stata implementata una coppia classe-interfaccia per ogni singolo componente hardware.

Così facendo durante la programmazione non c'è mai stato bisogno di utilizzare DIRETTAMENTE funzioni di Arduino quali per esempio "digitalRead()" o "analogRead()", siccome sono sempre stati istanziati degli oggetti.

Questo garantisce la totale divisione tra programmazione ad oggetti e hardware.

3.2 Scelta delle Task

La scelta di quali e quante task gestire in questo caso era fondamentale siccome l'intero programma di Arduino è stato basato su questa struttura.
Dopo un'attenta analisi ai requisiti è stata sviluppata questa soluzione.

Le task **istanziate** saranno 5, ovvero:

- NormalState
  - Il ponte risulta essere in uno stato NORMALE e analizza il livello dell'acqua con un certo intervallo di tempo. Il sistema intelligente di luci è attivato.
- PreAlarmState
  - Il ponte risulta essere in uno stato di PREALLARME, analizza il livello dell'acqua con un intervallo di tempo inferiore a quello normale. Il sistema intelligente di luci è attivato.
- AlarmState
  - Il ponte risulta essere in uno stato di ALLARME, analizza il livello dell'acqua con un intervallo di tempo inferiore a quello di PreAllarme. Il sistema intelligente di luci è disattivato. La valvola viene aperta automaticamente in base al livello dell'acqua oppure utilizzando l'apertura manuale tramite strumentazione.
- LightTask
  - Analizza la presenza di passanti e accende o spegne le luci di conseguenza.

- StateActivator
  - Attiva o disattiva le varie task in base all'ultimo dato ricevuto dal sonar. Questa task si assicura che solo una delle prime 3 task sia attivata alla volta. Gesitisce anche l'attivazione e disattivazione della task per le luci dei passanti. Viene eseguita ad ogni ciclo dello scheduler.

Le task "NormalState", "PreAlarmState" e "AlarmState" verranno tutte istanziate da una classe comune chiamata "State", che estende la classe "Task".
Per differenziare ciascuno stato verranno passati come parametri vari dati che definiranno il comportamento di ciascuna task.

3.3 Task nel dettaglio

3.3.1 State

La classe "State" risulta essere la classe principale per definire i vari stati del ponte.
Tutti gli stati del ponte vengono istanziati da questa classe, programmata in modo da essere il piu generale possibile, affinchè sia facile generare nuovi stati per il ponte o gestire in modo semplice e efficace le modifiche per stati già esistenti.
Questa classe presenta numerosi parametri nel costruttore proprio per definire nel minimo dettaglio lo stato del ponte partendo da una classe generica.

I parametri utilizzati sono:

- name
  - Nome dello stato.
- led1 & led2
  - Due istanze di led che indicano lo stato del ponte.
- sonar
  - Sensore che eseguirà le misurazioni
- motor
  - Valvola per far defluire l'acqua.
- lcd
  - Schermo per avvisare i pedoni dello stato del ponte.
- pot
  - Istanza della classe potenziometro per controllare la valvola.
- btn
  - Bottone per attivare la modalità Manuale.

- lcdState
  - Enumerazione che indica la modalità di visualizzazione per l'lcd
- statusLed1
  - Indica lo stato del led1 quando si è in questo stato: 1 = ACCESO, 0 = SPENTO.
- statusLed2
  - Indica lo stato del led2 quando si è in questo stato: 1 = ACCESO, 0 = SPENTO. Inoltre è possibile inserire un numero maggiore di zero, in questo caso il led inizierà a lampeggiare con un periodo pari a N millisecondi.
- minWaterDistance
  - Minima distanza dall'acqua che ci deve essere per entrare in questo stato.
- maxWaterDistance
  - Massima distanza dall'acqua che ci deve essere per rimanere in questo stato.
- WL_MIN
  - Distanza tra il ponte e il fondo del fiume.
- manualOperations
  - Variabile booleana che indica se in questo stato è possibile attivare la modalità manuale della valvola o meno.
- minValve
  - Gradi minimi della valvola di scolo, quando la distanza dell'acqua risulterà essere uguale a "minWaterDistance" la valvola sarà aperta di una gradazione pari a "minValve"

- maxValve
  - Gradi massimi della valvola di scolo, quando la distanza dell'acqua risulterà essere uguale a "maxWaterDistance" la valvola sarà aperta di una gradazione pari a "maxValve"

Ad ogni tick, questa task effettuerà una misurazione del livello dell'acqua.
Nel caso questa misurazione rientri nel range prestabilito, verrò effettuato un aggiornamento dello stato delle luci, dello schermo e della valvola sempre in maniera coerente ai parametri inseriti in fase di creazione dell'istanza.
Infine, queste task hanno il compito di inviare messaggi all'applicazione JAVA inviando informazioni riguardante il nome dello stato attuale e del livello attuale dell'acqua appena misurato.

3.3.2 LightTask

LightTask è la task che gestisce il sitema di illuminazione automatico per i pedoni che attraversano il ponte.
Ad ogni tick, nel metodo principale ci si accerta che le condizioni per accendere o spegnere le luci siano verificate, ovvero si analizza lo stato della luce tramite LightSensor e la presenza o meno di movimento tramite PIR.
Una volta che le luci vengono accese e il pir non rileva piu del movimento, le luci resteranno accese per T1 tempo, di default settato a 5 secondi.
Nel caso in cui il PIR rilevi del movimento mentre le luci sono ancora accese, allora verrà resettato il timer di spegnimento.
Nella casistica in cui la luminosità aumenti sopra il treshold mentre le luci risultano essere accese, il timer di spegnimento viene ignorato e le luci vengono spente.
Questa task ha anche il compito di inviare un messaggio per notificare un eventuale cambiamento di stato delle luci dei passanti all'applicazione JAVA.

3.3.3 StateActivator

StateActivator è una task che ha il compito di analizzare i dati raccolti dal sonar e di attivare o disattivare i vari stati del ponte (sotto forma di task istanziate).
Ad ogni tick, questa task si assicura che solo una delle prime 3 task sia attiva alla volta e che quindi per ogni valore misurato dal sonar venga assegnato solamente uno stato del ponte.
Bisogna specificare che NON viene eseguita nuova misurazione del livello dell'acqua, ma viene analizzata semplicemente l'ultima misurazione eseguita dal sensore.
Inoltre, questa task ha il compito di attivare o disattivare la funzionalità di illuminazione intelligente nel caso si entri in uno stato specifico (esempio: Stato di ALLARME).
StateActivator è stata progettata in modo che venga eseguita ad ogni ciclo dello scheduler.
Per fare in modo che questa classe possa funzionare come descritto, è necessario inserire come parametri nel costruttore tutte e tre le task-stati del ponte, oltre che l'istanza della classe LightTask e il sonar.
Infine, i 3 parametri finali sono delle variabili booleane (vero/falso) che servono a indicare allo StateActivator se attivare o meno la funzionalità delle luci automatiche quando il ponte è in uno stato particolare.
Esempio: Se "lightsEnabled1" risulta essere "TRUE", allora le luci intelligenti saranno attive quando il ponte risulta essere nello stato N1, in questo caso, lo stato "Normale".

3.4 Struttura Setup e SuperLoop

Grazie alla programmazione ad oggetti è stato possibile mantenere le due funzioni principali di Arduino molto ridotte in termini di codice.

Inizialmente sono state elencate le varie costanti che descrivono il dominio del ponte, oltre a quelle che definiscono i vari PIN dei componenti hardware.

Nel "Setup()" sono stati istanziati i componenti hardware dalle loro relative classi.
Infine, dopo aver istanziato e inizializzato lo scheduler, vengono create e inizializzate le varie task, passando come parametri iniziale tutto l'occorrente tra componenti hardware e parametri di dominio.
In conclusione ogni task viene aggiunta allo scheduler.

Il superloop non contiene altro che il metodo principale "schedule()" dello scheduler.

Chapter 4

Programma Java

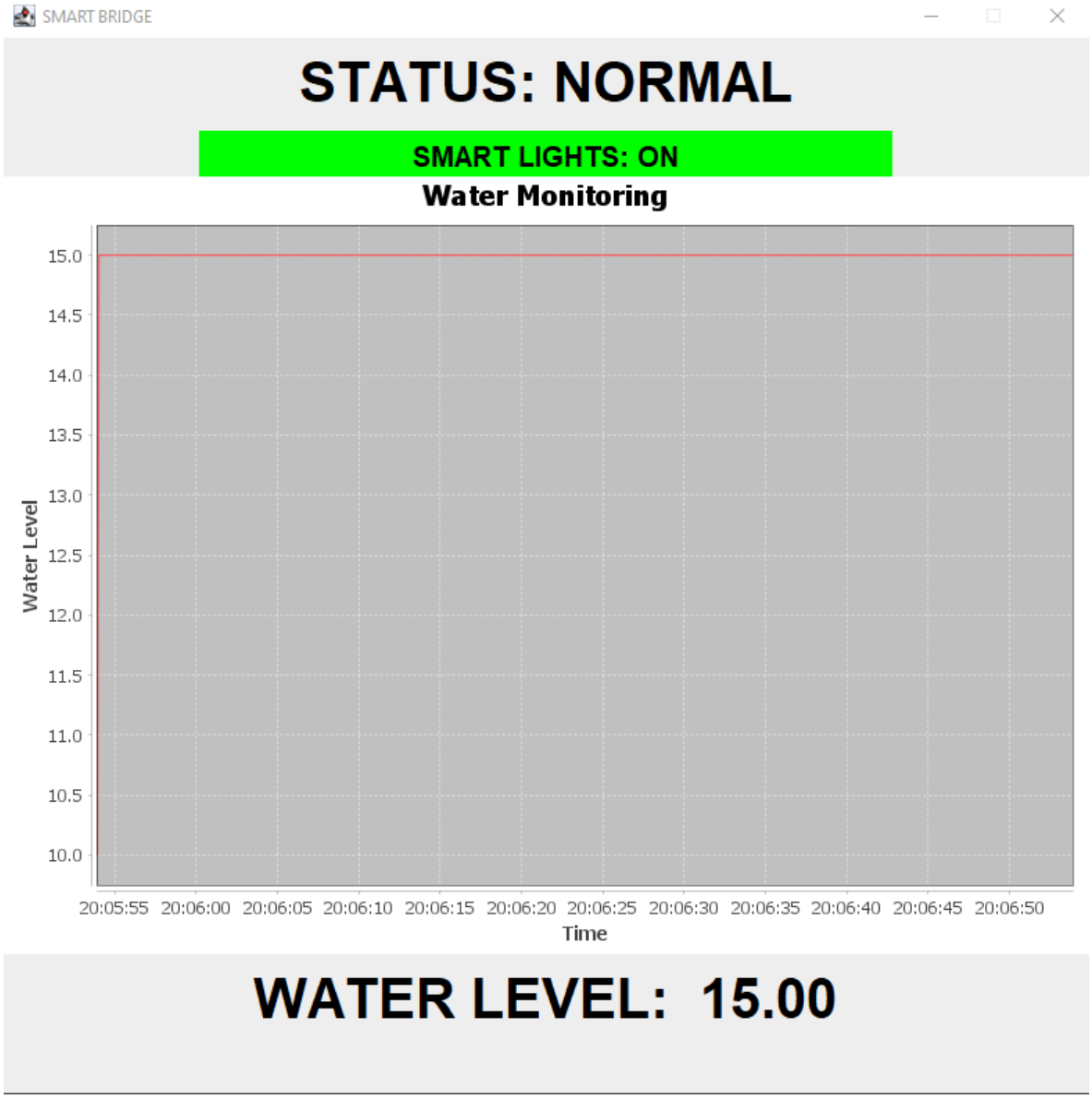
E' stato poi sviluppato un programma in Java che consente all'utente di:

- Visualizzare lo Stato **attuale** del ponte.
- Visualizzare se le luci intelligenti siano accese o spente.
- Visualizzare il livello **attuale** dell'acqua.
- Visualizzare un grafico che mostra il livello dell'acqua nel corso del tempo. Il grafico viene aggiornato in tempo reale.

I dati necessari vengono inviati direttamente da Arduino via Seriale ad ogni ciclo.

Per disegnare il grafico è stata utilizzata la libreria chiamata "JFreeChart".

4.1 Screenshot dell'applicazione JAVA



Chapter 5

Come avviene il dialogo tra Arduino e l'Applicazione JAVA

L'applicazione JAVA necessita di dati in tempo reale per aggiornare il grafico e per far si che la sua visualizzazione rispetti quello che accade nella realtà.
Per garantire ciò, Arduino ha il compito di inviare a tale applicazione ogni aggiornamento che viene effettuato.
Ogni volta che la luce dei pedoni viene attivata o disattivata, Arduino invierà un messaggio contenente "Smart Light:" e lo stato delle luci: 0 o 1.
Ogni volta che viene effettuata una rilevazione del livello dell'acqua, Arduino invierà un messaggio contenente lo stato del ponte e il valore appena misurato.

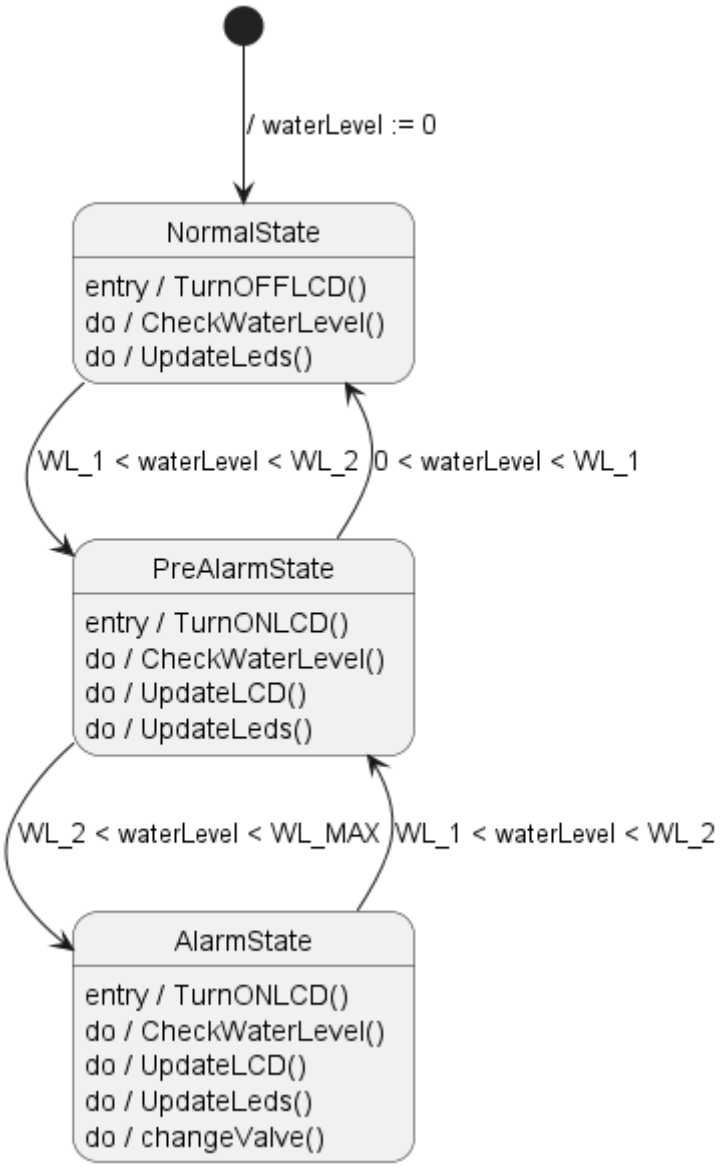
Tutti questi dati vengono inviati tramite Seriale grazie alla classe "MsgService" e alla sua relativa interfaccia.

L'Applicazione JAVA è sempre in attesa di nuovi messaggi, alla ricezione di ogni messaggio, quest'ultimo viene filtrato e utilizzato per aggiornare il relativo campo.

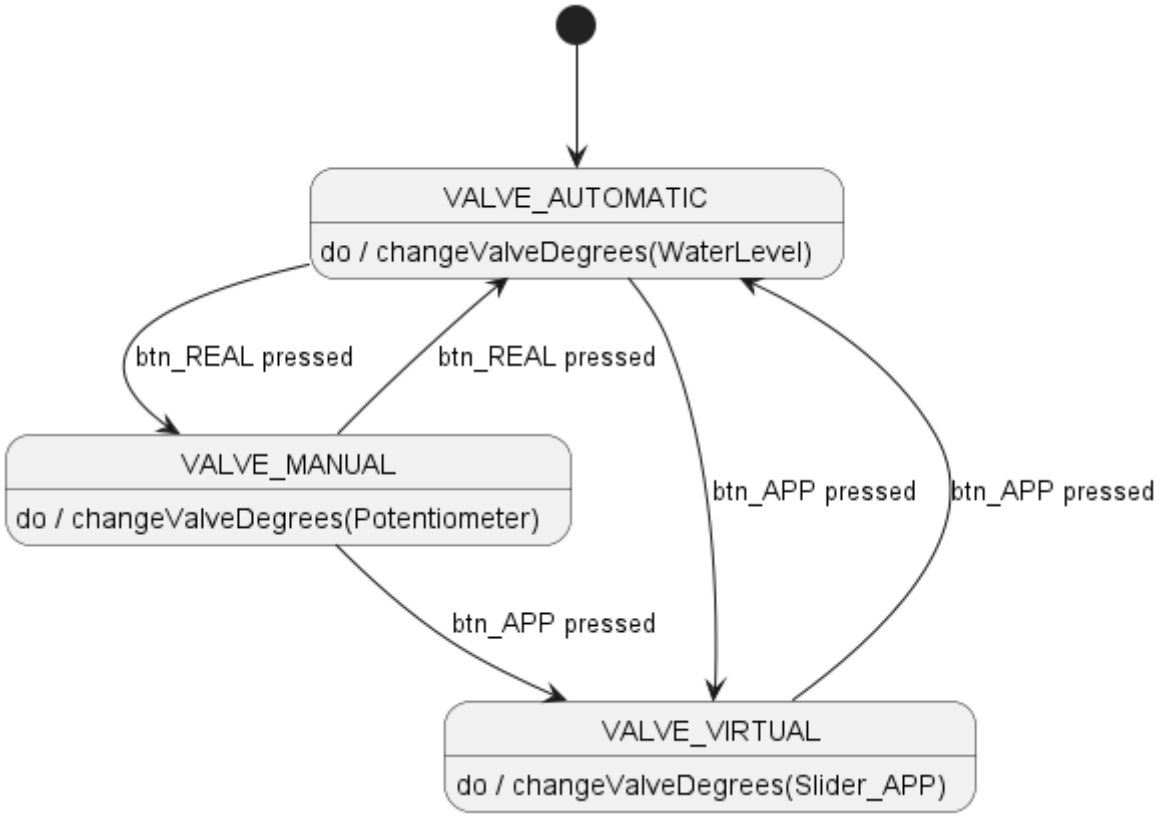
Chapter 6

Schemi degli stati

6.1 Stati del ponte



6.2 Valvola


VALVE_AUTOMATIC
do / changeValveDegrees(WaterLevel)
btn_REAL pressed
btn_REAL pressed
VALVE_MANUAL
do / changeValveDegrees(Potentiometer)
btn_APP pressed
btn_APP pressed
btn_APP pressed
VALVE_VIRTUAL
do / changeValveDegrees(Slider_APP)

6.3 Illuminazione Automatica