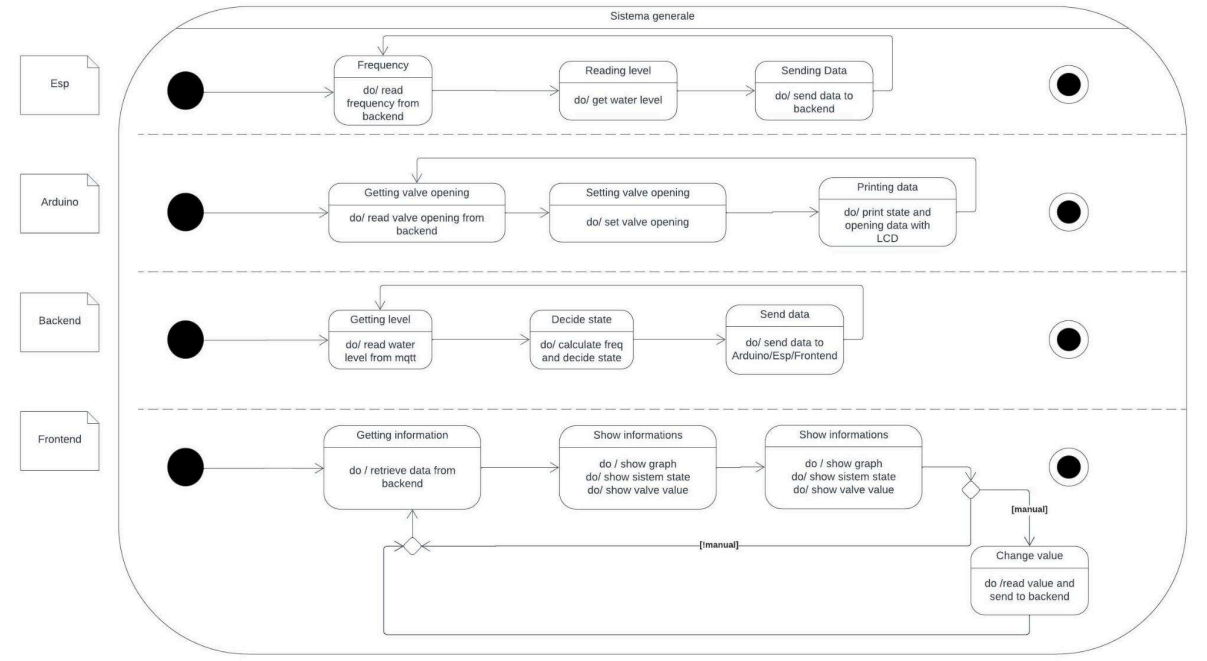
Relazione Assignment 3

Annibalini Lorenzo
Bacchini Lorenzo
Sanchi Emanuele

Descrizione del sistema

Il sistema si compone di quattro parti: water level monitoring subsystem, water channel controller, river monitoring service e river monitoring dashboard. Tra di essi la comunicazione avviene in modo differente, ma ognuna passa sempre attraverso il river monitoring service.
Per le connessioni MQTT abbiamo utilizzato mosquitto v.1.6.9 come broker sulla porta 1883, come topic “water-level” e “frequency” e come librerie per intefcciarsi paho.mqtt.client; per le connessioni HTTP abbiamo usato le librerie di python requests e http.server; per la connessione seriale abbiamo usato la libreria python serial.tools.list_ports.
Per semplicità di realizzazione e debug l’intero sistema è stato realizzato collegando tutti i dispositivi di rete alla stessa rete, in particolare alla rete creata dall’hotspot di un telefono. SSID e password sono quindi state inserite all’interno dei codici. Inoltre sia la dashboard che mosquitto che il river monitoring service sono stati avviati nello stesso PC permettendo così l’utilizzo dell’indirizzo di rete come localhost.

Diagrammi di stato



Water Level Monitoring Subsystem

Il water level monitoring subsystem è il responsabile della lettura del livello dell'acqua.
Il sistema è realizzato tramite un ESP al quale sono collegati due led e un sonar. L'ESP dopo aver letto la distanza dall'acqua tramite il sonar, calcola l'altezza effettuando una sottrazione tra l'altezza del ponte e la distanza rilevata. Invia poi il risultato al river monitoring service utilizzando il protocollo MQTT con topic "water-level". I due led sono utilizzati per verificare il corretto funzionamento della rete e quindi quello rosso viene acceso se uno tra connessione Wi-Fi e connessione al broker MQTT sono assenti; quello verde si accende quando entrambe le connessioni funzionano e l'ESP comunica in modo corretto. Per realizzare le connessioni Wi-Fi e MQTT abbiamo usato rispettivamente le librerie WiFi.h e PubSubClient.h come visto a lezione.
Il sistema è anche iscritto al topic "frequency" dal quale legge la frequenza con cui campionare l'altezza dell'acqua. Questa frequenza viene pubblicata sul broker dal river monitoring service.

Water Channel Controller

Il water channel controller si occupa della gestione dell'apertura della valvola del fiume e della stampa dei dati su un display LCD.
Secondo la nostra implementazione ad ogni esecuzione del superLoop arduino (water channel controller) riceve i dati sulla seriale dal river monitoring service e agisce di conseguenza regolando l'apertura della valvola e mostrando sullo schermo LCD quelli che sono i dati relativi allo stato del sistema e appunto all'apertura della valvola, è inoltre presente sulla breadboard di arduino un bottone, che consente al water channel controller di cambiare la sua modalità operativa tra automatica in cui i livelli di apertura della valvola sono gestiti in base allo stato del sistema e manuale in cui la valvola è direttamente comandata dal river monitoring dashboard grazie ad uno slider.

River Monitoring Service

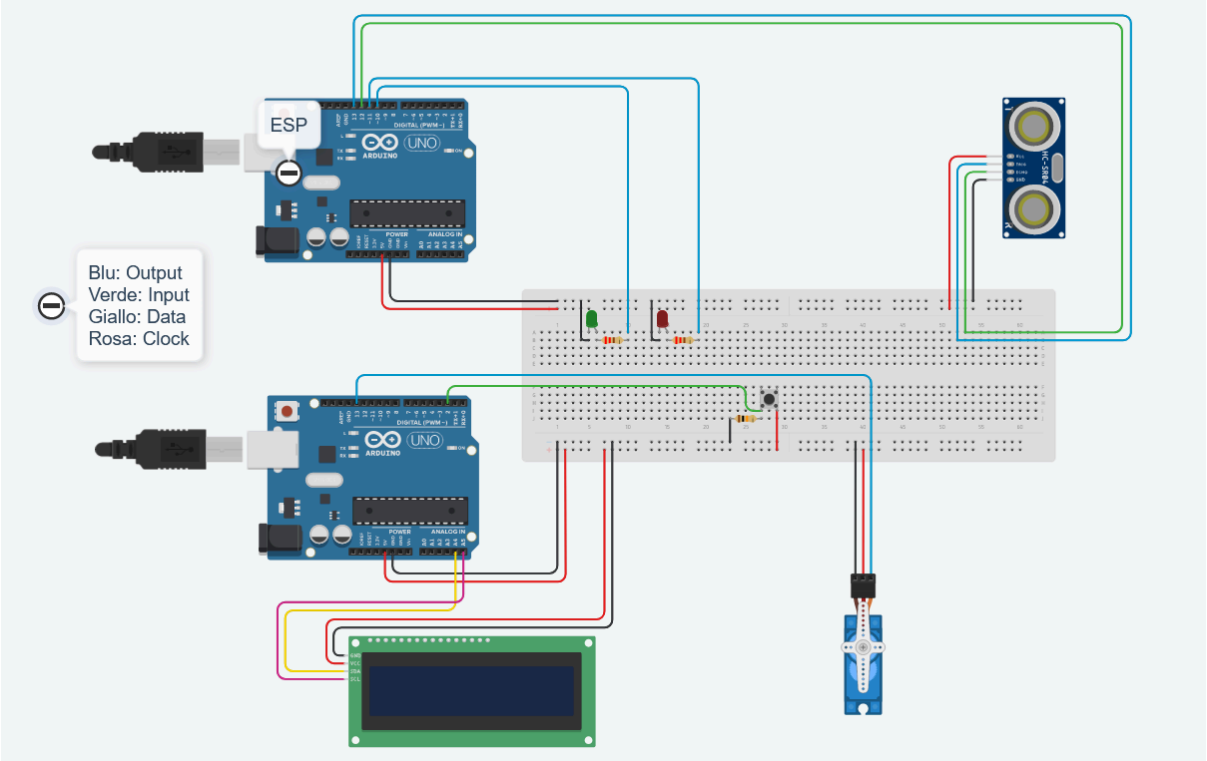
Il river monitoring service è la mente dell'intero sistema. Qui infatti è realizzata la logica del funzionamento dell'intero sistema e da qui partono i segnali per comunicare con i vari sottosistemi (ESP, Arduino e Dashboard).
Il river monitoring service è realizzato tramite python e al suo interno svolge tre azioni principali: comunica con l'ESP tramite il broker MQTT, comunica con Arduino tramite la porta seriale e comunica con la dashboard tramite HTTP.
All'interno è presente una Enum che rappresenta gli stati di allerta del livello dell'acqua.
Alla partenza del sistema vengono lette tutte le porte seriali collegate e stampate a schermo così che l'utente possa scegliere quella giusta. Poi il sistema si connette al broker MQTT iscrivendosi al topic "water-level" per poter ricevere le letture dell'ESP. Successivamente, in un thread separato, apre una connessione HTTP come server per poter permettere alla Dashboard di collegarsi come client. Dal punto di vista della connessione HTTP vengono gestite due richieste: una sul path generico "/", per leggere l'attuale livello dell'acqua, e una sul path "/set_valve_value?value=n" con cui prende il valore passato in GET dalla Dashboard e invia di conseguenza ad Arduino per un'apertura manuale.

Infine il sistema entra in un loop infinito in cui, leggendo il valore dell'altezza dell'acqua, decide in quale stato entrare e quindi cosa inviare in seriale ad Arduino e cosa mettere a disposizione in HTTP.

River Monitoring Dashboard

La river monitoring dashboard scritta in python è l'interfaccia che permette di visualizzare e comandare l'intero sistema.
Al suo interno troviamo: il grafico altezza acqua / tempo per la visualizzazione dell'andamento nel tempo del livello all'interno del fiume, lo stato del sistema (normal, alert, ecc..) e lo slider per l'apertura manuale della valvola di sfiato.
Dal punto di vista delle comunicazioni il frontend comunica in HTTP con il river monitoring service al quale ogni n secondi invia una request (GET) all'url per riceve una stringa con tutte le informazioni del sistema; se nel mentre si modifica lo slider per l'apertura manuale della valvola allora viene fatta una richiesta HTTP ad un altro path e viene quindi attivata la modalità automatica.
Per semplificare la costruzione grafica della dashboard abbiamo deciso di affidarci a PySimpleGUI e MatPlotLib con Pyplot rispettivamente per la creazione delle finestre, bottoni e slider e del grafico.

Schema tinkercad



Link video:[video.mp4]