Smart Dam

Third Assignment – IOT
a.a. 2020/2021
Simone Ceredi – simone.ceredi@studio.unibo.it
Alessio Conti – alessio.conti3@studio.unibo.it

Il progetto vuole modellare un sistema per la valutazione real-time del livello di piena di un fiume/lago e gestire di conseguenza l'andamento delle acque tramite controllo di una diga.

L'elaborato si articola in diversi sistemi, cui di seguito verrà definito l'approccio utilizzato.

Remote Hydrometer (RH)

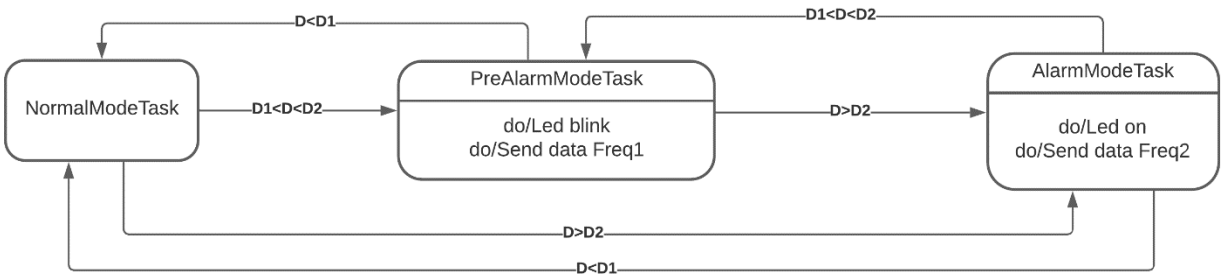
Sistema basato esp8266 il quale deve tenere traccia del livello dell'acqua tramite un sensore di prossimità, sonar.

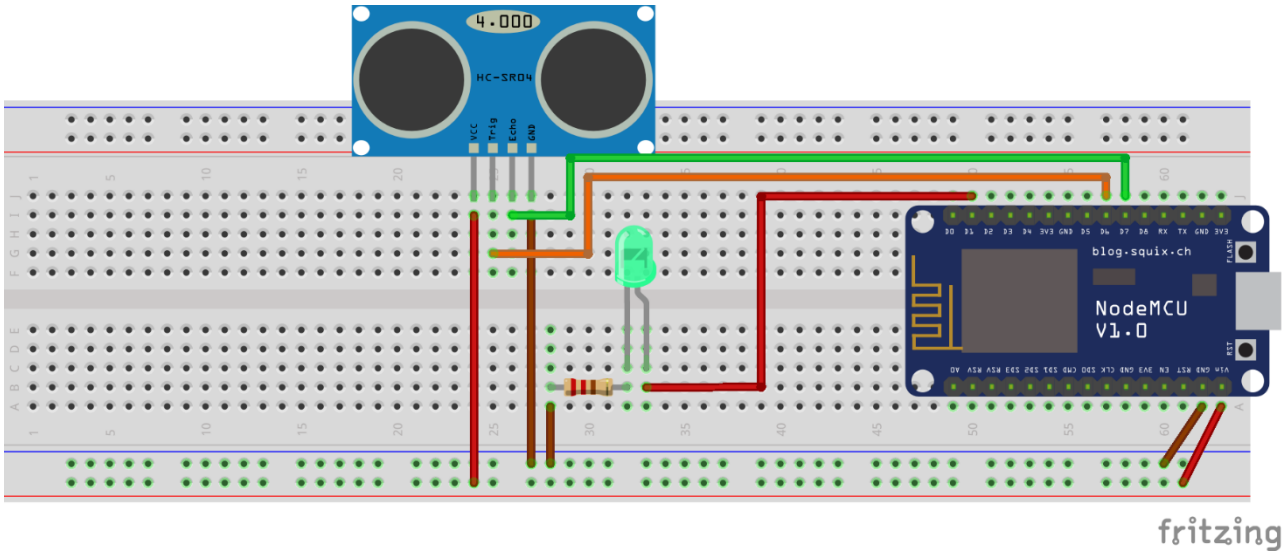
Per sviluppare il sistema abbiamo adottato un'architettura a task sincrona, in quanto le rilevazioni vengono effettuate a cadenze regolari ed eventuali comportamenti secondari sono particolarmente adatti ad un'implementazione a suddivisione di tempo come, ad esempio, il blinking del led.

I task principali sono tre, gli stessi che caratterizzano l'andamento dell'intero sistema: normale, preallarme ed allarme. A questi si aggiunge il task di gestione del blinking del led. Adottando il paradigma ad oggetti, ogni sensore ed attuatore è stato modellato tramite una classe c++ con rispettiva interfaccia.

A far procedere il sistema è uno scheduler cooperativo implementato tramite una lista. A seconda del proprio periodo di funzionamento questo richiama all'azione i vari task "attivi" per i quali è trascorso un tempo congruo al loro periodo. Per lo sviluppo abbiamo cercato di mantenere minimale il tempo d'esecuzione di ogni task limitando al massimo il numero di operazioni che esso deve svolgere.

I task normal, prealarm e alarm lavorano in modalità similare: a seguito di una rilevazione tramite il sonar viene presa una decisione riguardo alla condizione in cui il sistema si trova in quel momento. Eventualmente viene postato tramite protocollo MQTT se vi è stato un cambio di stato del sistema oppure nel caso in cui sia passato un tempo compatibile con l'emissione di un nuovo dato di rilevazione.





Dam Service (DS)

Sistema backend che permette lo scambio di informazioni e dati tra i vari sottosistemi. Funge da elemento cardine e centrale in quanto coordina le varie operazioni dei sottosistemi a lui collegati. Sviluppato tramite java implementa un server HTTP che permette lo scambio di informazioni con Dam Dashboard e Dam Mobile App. Utilizza il protocollo MQTT per ricevere le informazioni dal Remote Hydrometer. Utilizza un canale seriale per lo scambio informativo con Dam Controller.

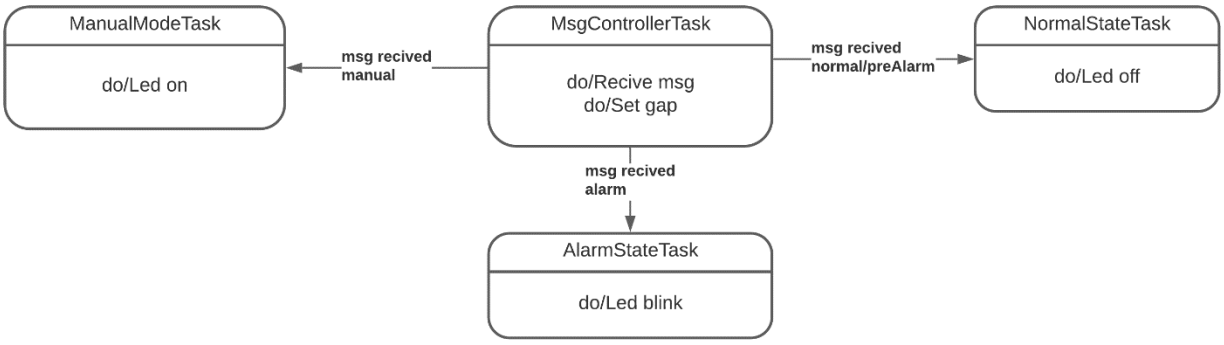
Elabora i dati provenienti da Remote Hydrometer e comanda direttamente tramite un canale seriale il livello di apertura della diga al Dam Controller (in caso in cui il sistema sia in automatico). Tutti i dati ricevuti vengono processati in modo da essere "comprensibili" agli altri sottosistemi. Tali dati vengono poi salvati all'interno della classe DsData, la quale funge da model per l'applicazione.

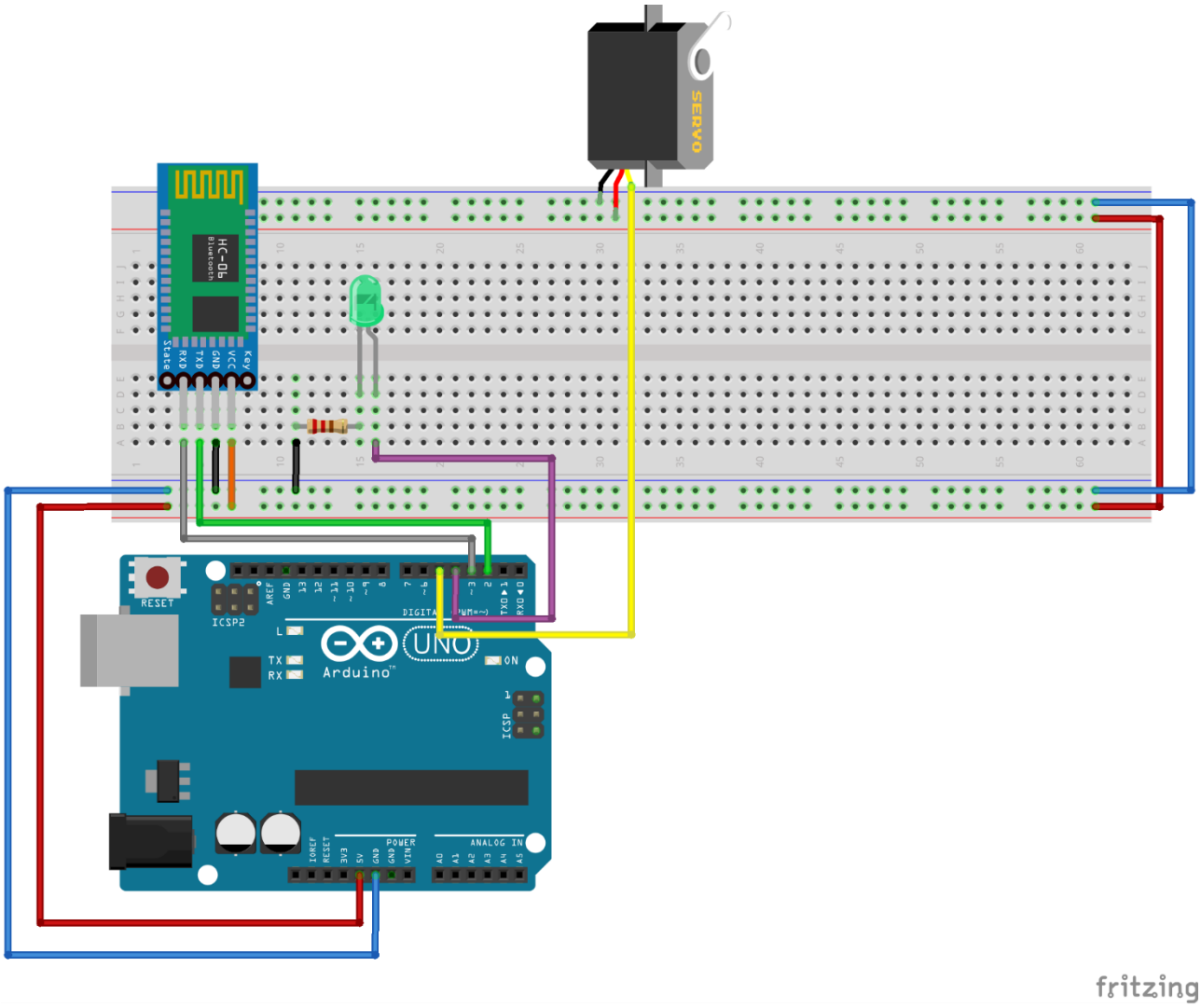
Dam Controller (DC)

Sistema basato Arduino UNO il quale a seconda degli ordini imposti dal Dam Service oppure da un operatore fisico tramite Dam Mobile App deve attuare l'apertura o chiusura di una diga grazie all'utilizzo di un servomotore.

Come per il Remote Hydrometer anche in questo caso a far procedere il sistema è uno scheduler cooperativo implementato tramite una lista. Soluzione particolarmente adatta in quanto la maggior parte degli stati in cui si trova il sistema durante l'esecuzione richiedono una temporizzazione (blinking del led, apertura/chiusura della diga).

Il fulcro di tale sistema si trova nel task di controllo dei messaggi. Alla ricezione di nuove notifiche via Bluetooth oppure da seriale le decodifica e attiva il task necessario per svolgere il comando imposto, a seconda dello stato in cui si trova il sistema (NormalStateTask, AlarmStateTask, ManualModeTask).





Dam Mobile App (DM)

Mobile app Android che comunica tramite bluetooth con il Dam Controller e richiede dati tramite HTTP dal Dam Service.

Implementa un thread secondario grazie ad un async task in modo da permettere di attuare richieste HTTP da un Worker Thread, mentre allo stesso tempo permette di demandare tutte le operazioni inerenti all'interfaccia utente al Main Thread.

La logica di gestione viene gestita da una classe separata con relativa interfaccia: Strategy e StrategyImpl in modo da mantenere il più possibile separati i concetti di recezione dati e logica di gestione e controllo.

Dam Dashboard (DD)

Front-end sviluppato come client su pc tramite java.

Periodicamente la dashboard attua richieste HTTP verso il Dam Service richiedendo lo stato del sistema. Elaborando i dati ricevuti, l'applicativo procede con eventuali ulteriori richieste HTTP per ottenere tutti i dati di cui necessita in quel preciso istante.

Il programma è stato suddiviso in tre componenti principali: Main, View, DataCollector che possono essere ricondotti ad una semplificazione di MVC.

La classe Main esemplifica il controller, questa ciclicamente richiama il metodo pubblico "render" della classe View.

La logica di controllo è incapsulata e mantenuta separata con l'utilizzo di una classe che implementa il pattern Strategy, così da mantenere il sistema modulare e componibile.

La classe DataCollector funge da model mantenendo e gestendo i dati ricevuti da Dam Service.

Per quanto riguarda la resa grafica abbiamo utilizzato la libreria esterna "XChart" per la creazione di grafici real-time che rappresentassero fedelmente il livello d'acqua rilevato in funzione del tempo.