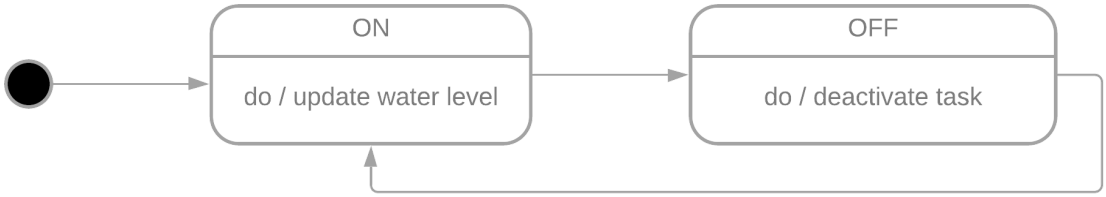
Relazione Assignment n.3

Autori: Acampora Andrea, Accursi Giacomo

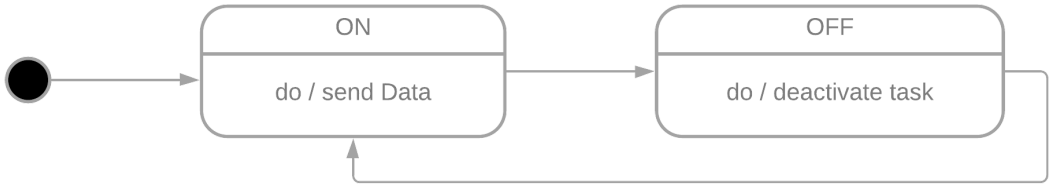
Remote Hydrometer

La rilevazione del livello dell'acqua è a carico del microcontrollore esp8266 che fa usa di un sonar.
Il sottosistema è stato implementato con logica in termini di macchine a stati finiti sincrone.
I task individuati sono:

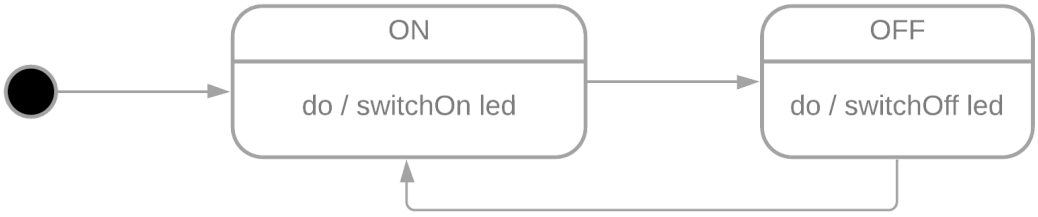
- Tracking Task: si occupa di monitorare e aggiornare il livello dell'acqua utilizzando il sonar. Il task rimane attivo per tutta l'esecuzione del sistema.



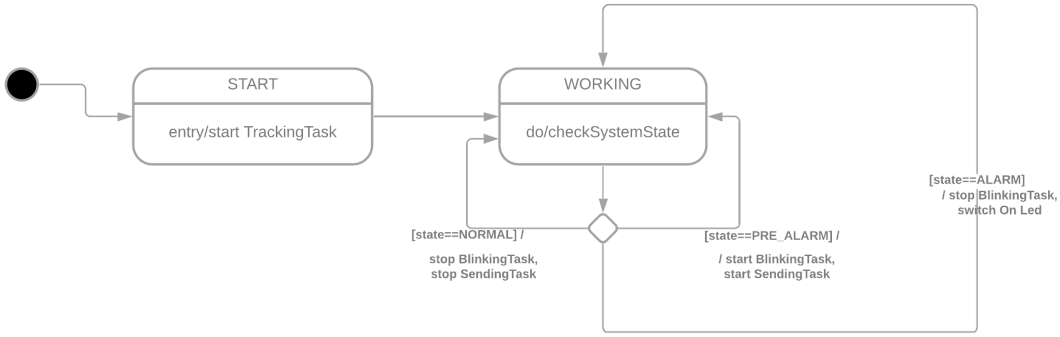
- Sending Task: il task è attivo quando il sistema si trova in stato di PRE_ALLARME o ALLARME, mentre viene disattivato quando si trova in stato NORMALE. Il suo compito è quello di comunicare i dati (livello dell'acqua e stato del sistema), attraverso chiamate Post al Dam Service utilizzando il protocollo HTTP.



- Blinking Task: viene attivato quando il sistema si trova in PRE_ALLARME. Ad ogni tick accende il LedAlarm se è spento e lo spegne se è acceso.



- WaterLevelControllerTask: si accorge di cambiamenti nello stato del sistema e attiva o disattiva gli altri task di conseguenza.



E' stato creato un oggetto WaterLevel, nel quale viene salvato il livello dell'acqua corrente. E' l'unico che conosce la logica con cui cambiare lo stato del sistema.

Componenti utilizzati:

- Led verde
- Sonar
- Sensore di temperatura LM35
- Microcontrollore Esp8266

Dam Service

E' stato realizzato mediante un'applicazione Java che fa uso del toolkit Vert.x per gestire le chiamate HTTP provenienti da Remote Hydrometer, Dam Dashboard e Dam Mobile App.
Dam service è in grado di comunicare con gli altri sottosistemi grazie all'utilizzo dell'applicativo Ngrok che permette di raggiungere il server HTTP attraverso un indirizzo pubblico temporaneo.

Il sistema è in grado di far fronte a diverse richieste HTTP mettendo a disposizione un'API per ognuna.

Il controller dell'applicazione è responsabile di mettere in comunicazione il model con il verticle e interagisce via seriale con il DamController per comunicare l'apertura della diga.

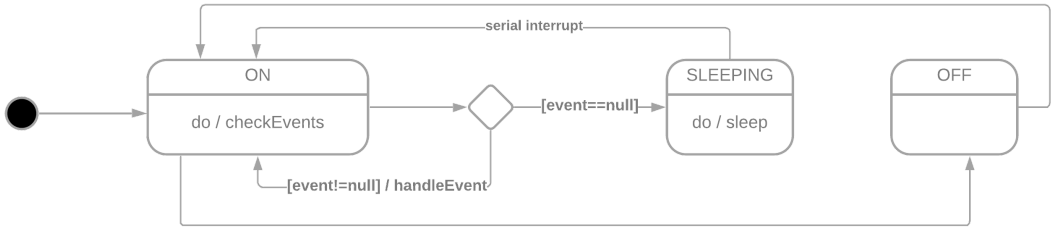
Il model contiene i metodi per accedere ai dati utili dell'applicazione e prende decisioni riguardo l'apertura della diga.

Per salvare lo storico delle rilevazioni è stato scelto di utilizzare un database. Per ognuna vengono salvati: livello dell'acqua, stato del sistema e momento della rilevazione.

La modalità (Manuale o Automatica) e il livello di apertura della diga non vengono salvati insieme alle rilevazioni perchè il loro cambiamento è asincrono rispetto a quest'ultime. E' il model ad occuparsi di queste informazioni, fornendole poi al controller su richiesta.

Dam Controller

Per la realizzazione è stato deciso di utilizzare un approccio ad eventi utilizzando una macchina a stati finiti asincrona. E' stato scelto questo approccio poichè permette di evitare di controllare continuamente la presenza di nuove comunicazione da parte del Dam Service e della Dam Mobile App.



Ogni interruzione viene comunicata al proprio eventSource il quale ha il compito di valutarla per poi generare un evento il quale viene aggiunto nella coda della macchina a stati finiti.

Gli eventi individuati sono:

- nuovo messaggio da parte del Dam Service sulla seriale
- nuovo messaggio della Mobile App tramite bluetooth.
- scadenza del timer utilizzato per il blink del led.

Le interruzioni sulla seriale, generate dall'arrivo dei caratteri, vengono notificate ad un event source che si occupa di leggerli e generare un evento in base al messaggio ricevuto.

Per la gestione del canale bluetooth è stata preferita la libreria AltSoftSerial rispetto a SoftwareSerial poichè quest'ultima causa problemi se usata insieme alla libreria EnableInterrupt. vedi qui

Per quanto riguarda la seriale hardware, la quale mette in comunicazione Dam Service e Dam controller, vengono abilitate le interruzioni sul pin RX di arduino, mentre per il canale bluetooth vengono abilitate le interruzioni sul pin 8, pin di ricezione usato da AltSoftSerial.

Il blink del Led viene effettuato attraverso l'utilizzo del timer0 che notifica la scadenza mediante interruzioni ad un event source per cambiare lo stato del led.
Sebbene venga sconsigliato l'utilizzo di timer0, poichè viene già utilizzato dalle funzioni millis(), delay() e micros() , era l'unico timer che avevamo a disposizione in quanto timer1 è utilizzato dalla libreria AltSoftSerial e timer2 dalla libreria ServoTimer2. Per evitare conflitti non abbiamo utilizzato le funzioni che utilizzano timer0.

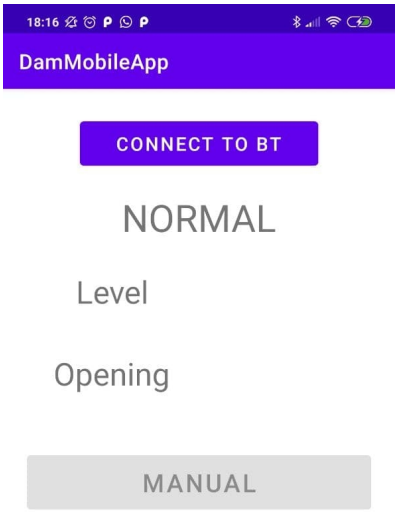
Il sottosistema Dam Controller non è a conoscenza dello stato del sistema in quanto si tratta di una entità passiva che reagisce in base alle comunicazioni che arrivano. Il blink del led è attivo quando vengono ricevuti messaggi in modalità automatica.

Componenti utilizzati:

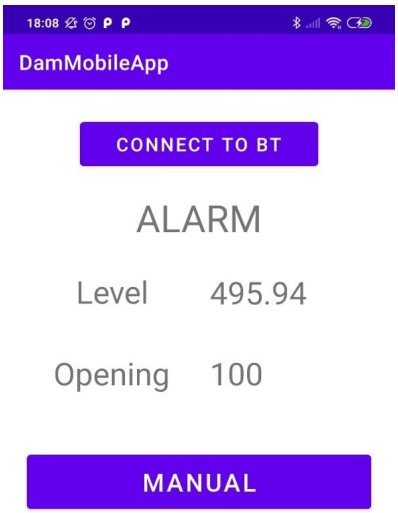
- Microcontrollore Arduino Uno
- Modulo bluetooth HC06
- Servo Motore
- Led verde

Dam Mobile App

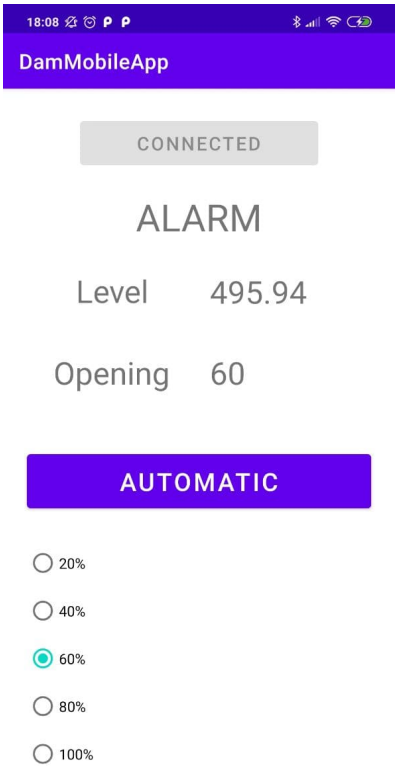
Sono stati utilizzati gli esempi relativi a richieste HTTP e connessione Bluetooth forniti dal prof. Croatti.
All'avvio dell'applicazione verrà mostrata la seguente schermata:



inizialmente il bottone per entrare in modalità manuale è disabilitato e verrà abilitato quando il sistema si troverà in stato di allarme. In qualunque momento è possibile connettersi ad Arduino tramite bluetooth utilizzando l'apposito bottone.



Una volta cliccato il bottone per entrare in modalità manuale, questo verrà segnalato al Dam Service mediante chiamata POST e apparirà la seguente schermata:



Cliccando su uno dei Radio-Button presenti verrà inviato un messaggio sul canale Bluetooth ad Arduino contenente il livello di apertura della diga scelto e verrà effettuata una chiamata POST al Dam Service per aggiornare i dati nel model dell'applicazione. Questo è necessario per fare in modo che anche la dashboard venga a conoscenza dell'attuale livello di apertura.

E' possibile tornare in modalità automatica cliccando nuovamente sul bottone . I radio-button a questo punto scompariranno, e l'app comunicherà al Dam Service la decisione.

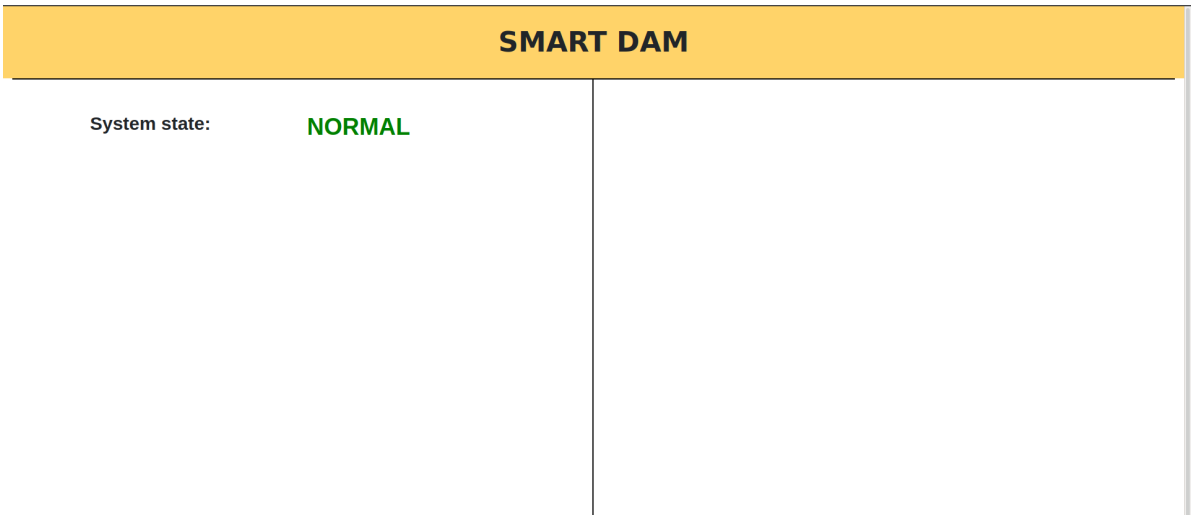
Tutte le chiamate HTTP vengono effettuate facendo uso di un Async-Task il quale viene eseguito su un thread separato rispetto al MainThread. E' stato utilizzato l'Async-Task poichè le chiamate HTTP sono per loro natura asincrone. Una volta terminata la chiamata HTTP l'aggiornamento della view è a carico di un thread separato.

L'applicazione riporta in tempo reale i dati forniti forniti dal Dam Service con una frequenza di aggiornamento di 5 secondi.

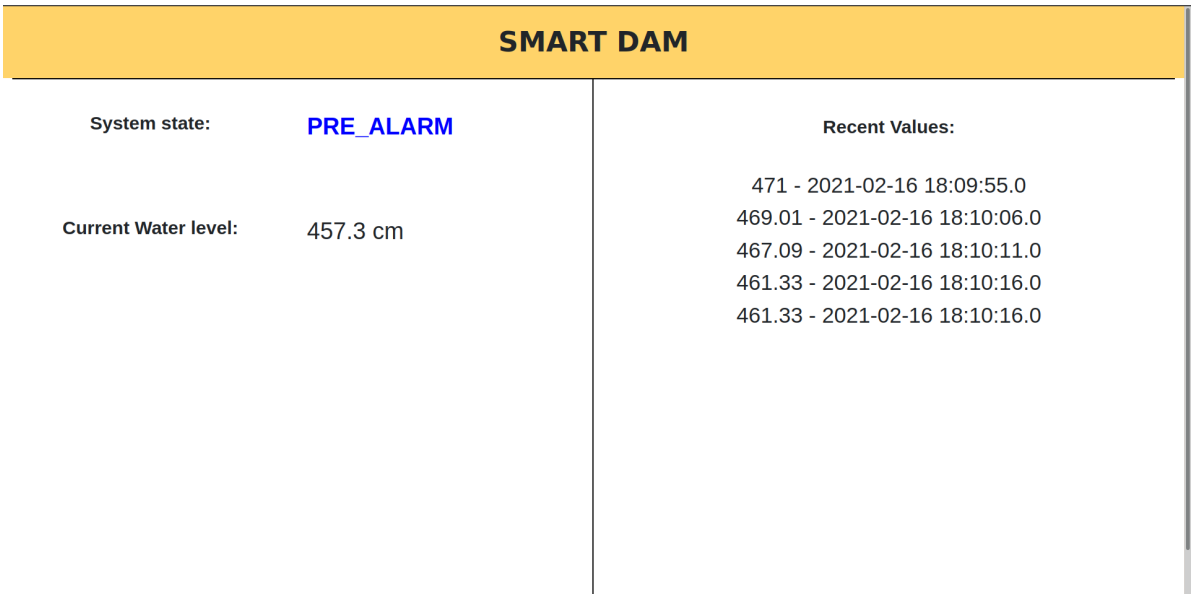
Dam Dashboard

SI tratta di una semplice Web App che effettua richieste GET al Dam Service ogni 5 secondi.
Il sistema non prende decisioni su quali dati visualizzare ma si limita a mostrare quelli che il Dam Service decide di esporre e che gli invia tramite un oggetto JSON.

Vista del sistema quando il sistema è in stato normale:



Vista del sistema quando il sistema è in stato pre_allarme:



Vista del sistema quando il sistema è in stato allarme:

SMART DAM

System state:	**ALARM**
Current Water level:	495.94 cm
Dam Opening level:	100 %
Modality:	MANUAL

Recent Values:

495.9 - 2021-02-16 18:08:31.0
495.94 - 2021-02-16 18:08:42.0
495.9 - 2021-02-16 18:08:48.0
495.94 - 2021-02-16 18:08:53.0
495.9 - 2021-02-16 18:08:58.0