Assignment-03 IoT report

Ingegneria e Scienze Informatiche

“Smart Room”

Filippo Gurioli – 0000971360

Silvia Furegato - 0000977475

Tommaso Turci - 0000971189

ANALISI

Durante la fase di analisi si è scelto di perseguire l'implementazione di una macchina a stati finiti asincrona. La scelta è dovuta alla volontà del gruppo di voler provare un'esperienza nuova rispetto a quella fatta nel precedente assignment, con sfide e logiche nuove.

L'obiettivo del gruppo è quindi gestire una Smart Room tramite l'utilizzo di diversi dispositivi che avrebbero dovuto comunicare tra di loro sfruttando la generazione e l'ascolto di eventi. In particolare, si vuole modellare un dispositivo mobile, in possesso dell'utente, che permetta allo stesso l'interazione con l'ambiente domotico, il quale viene rappresentato dal microcontrollore che emula le luci della Smart Room con un led e le tapparelle con un servo. Questo dovrà quindi modificarsi a seconda della volontà dell'utente espressa tramite l'app ma, oltre a ciò, dovrà anche essere in grado di essere modificato tramite una logica automatizzata gestita invece da un'applicazione web che sarà comunque interagibile tramite l'apposito sito. A completare il sistema sarà presente anche un ESP, le cui funzioni saranno quelle di monitorare le persone che entrano ed escono dalla stanza e al contempo misurare la luce solare proveniente dall'esterno della stanza.

ARCHITETTURA

Già dall'analisi si evince la nascita di conflitti tra quei componenti che, avendo la possibilità di lanciare comandi all'Arduino, avrebbero dovuto gestirsi un ordine di precedenza. Quest'ordine si è stabilito essere: frontend, mobile, backend. La scelta è dovuta dal fatto che si associa il ruolo di manager a chi accede ai controlli tramite la web app, e quindi necessita di massima precedenza; lo succede quindi mobile, che si presume essere un qualsiasi utente presente nella stanza; e per ultimo si lascia il posto al backend che, in assenza delle controparti, gestisce in automatico la stanza.

Dall'analisi si evidenziano quindi 5 sottostrutture: ESP, Arduino, app mobile, backend e frontend. Si andrà quindi a descrivere l'architettura intra - ed inter - struttura.

ESP

L'ESP, avendo un comportamento elementare, si è scelto di implementarlo con un loop classico in cui l'aggiornamento viene fatto solo in caso in cui venga effettivamente emesso un dato diverso da quello precedentemente inviato. Non elementare invece era il trasferimento del dato stesso, che si effettua tramite un collegamento HTTP direttamente con il backend. Data la caratteristica dell'ESP di essere solo una struttura di controllo, che non ha quindi bisogno di essere aggiornata, si è scelto di creare un collegamento unidirezionale verso il backend usando il metodo post dell'HTTP.

Frontend

Il frontend non è altro che una pagina web sviluppata con i classici tool CSS, PHP e JS. La sezione più interessante riguarda l'interazione di PHP e JS per l'invio e la ricezione dei dati di controllo. In entrata i dati sono gestiti in modo "lazy", ovvero richiesti solo al momento in cui lo user avrebbe veramente acceduto alla sezione che mostra quei particolari dati. In uscita, prima di ogni comando, si deve attivare la possibilità di interazione tramite l'apposita checkbox, che comunica al backend di rilasciare la modalità di gestione corrente (comandata in automatico o via mobile che sia) a favore della gestione del manager, ovvero i dati che verranno inviati in seguito dal frontend.

Mobile app

La mobile app sviluppata in Java deve servire all'utente per ricevere informazioni in tempo reale sullo stato delle periferiche dell'Arduino e al contempo permettergli di modificarle a suo gusto. Per la sua realizzazione si è scelto come canale di comunicazione il Bluetooth (BT), ottimo per le sue capacità di low power consumption e efficiente nello scambio di messaggi a corto raggio. Dopo aver instaurato una connessione BT con Arduino nell'apposita sezione, il thread dedicato alla ricezione delle informazioni sarà avvisato ogni qualvolta l'Arduino venga modificato (indipendentemente da chi lo stia facendo). Ancora una volta si è scelto di usare una checkbox per abilitare i comandi da mobile. La checkbox non sarà però attivabile se la mobile app fosse già stata avvisata della presa in gestione del frontend. In uno scenario normale, a checkbox attiva, il backend verrà avvisato della presa in gestione da parte del dispositivo mobile, che potrà quindi inviare i propri comandi ad Arduino che a sua volta, dopo l'esecuzione, riferirà i cambiamenti al backend.

Arduino

Per la natura del progetto, Arduino si è deciso di utilizzarlo come una sorta di “ripetitore” in quanto, da qualsiasi parte provengano le informazioni (backend o mobile) bisognerà avvisare l’altro estremo dei cambiamenti avvenuti, facendo così da ripetitore appunto. Nello specifico una macchina asincrona viene innescata dalla presenza di informazioni sulla linea seriale e dalla presenza di informazioni dal modulo BT. All’avvento di uno di questi eventi l’Arduino si comporta esattamente allo stesso modo: legge i dati in arrivo, esegue i comandi presenti nei dati ed invia in broadcast il risultato del comando.

Backend

Il backend è la struttura più complicata di tutto il progetto perché corrisponde al fulcro centrale di calcolo, in comunicazione con 3 sottosistemi ma che deve tenere in considerazione tutti e 5.

La sua realizzazione si compone di 3 thread, di cui 2 si occupano esclusivamente di gestire le comunicazioni con i 3 sottosistemi (uno dedicato alla seriale, comunicante con Arduino, e l’altro dedicato al protocollo HTTP, comunicante con ESP e frontend) mentre il principale gestisce la logica della modalità automatica, oltre che lo switch tra una modalità e l’altra. La modalità automatica viene sempre eseguita a meno che un dato comando non venga ricevuto da seriale, nel qual caso si passa il comando al BT, se poi però venisse recepito la volontà da parte del frontend di prendere la gestione questa passerebbe in primo piano, rilasciando il controllo del BT.

SCHEMI

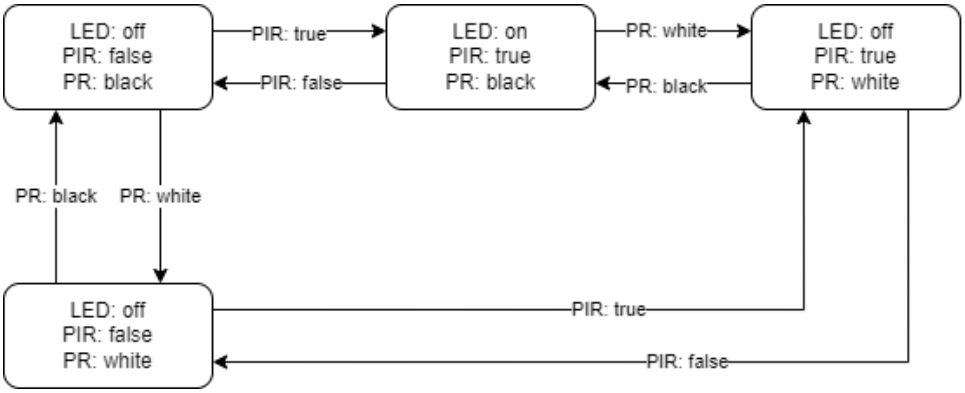

Diagramma 1.1: gestione automatica del LED

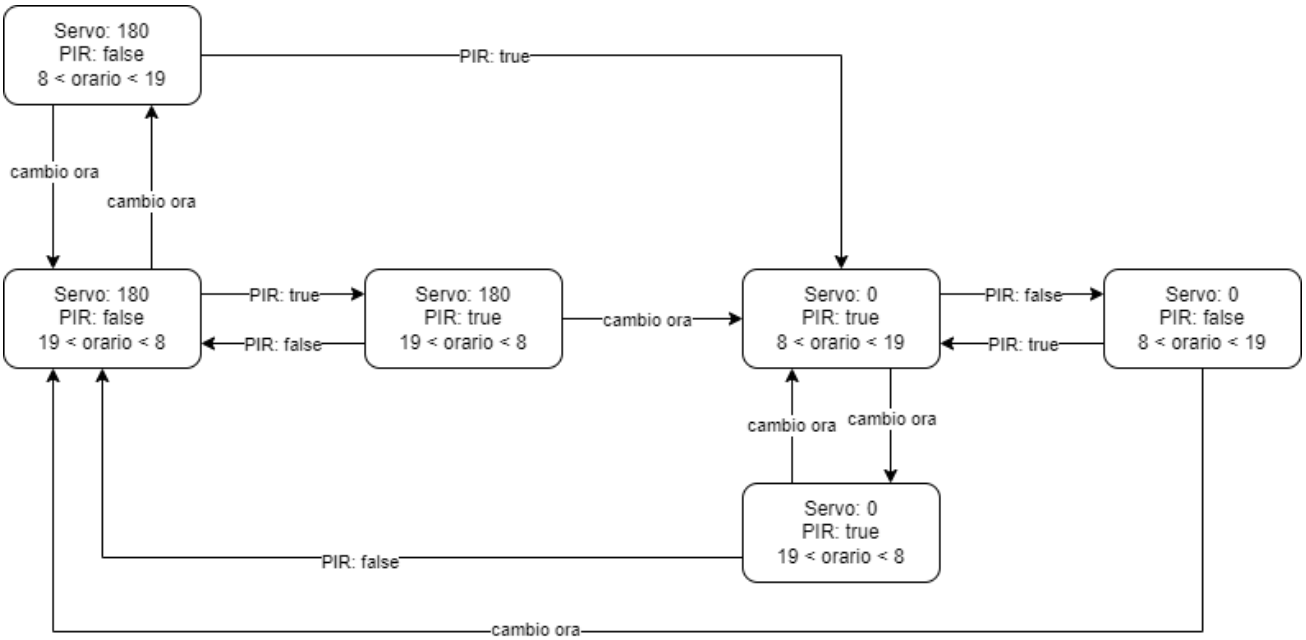

Diagramma 1.2: gestione automatica del Servo

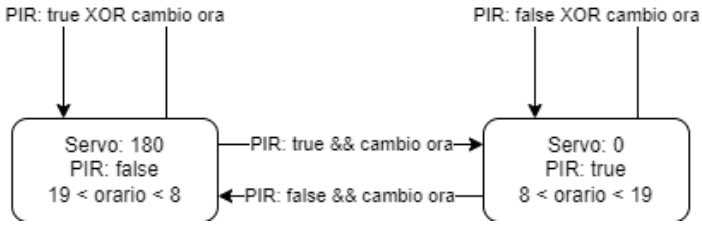

Diagramma 1.3: gestione automatica del Servo fattorizzata

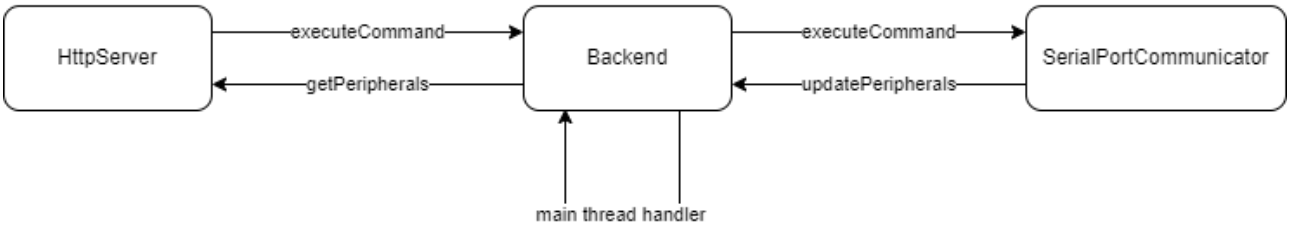

Diagramma 1.4: Backend async FSM

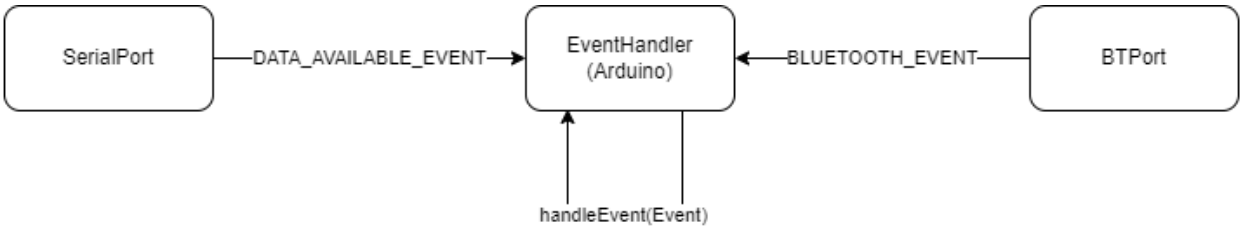

Diagramma 1.5: Arduino async FSM

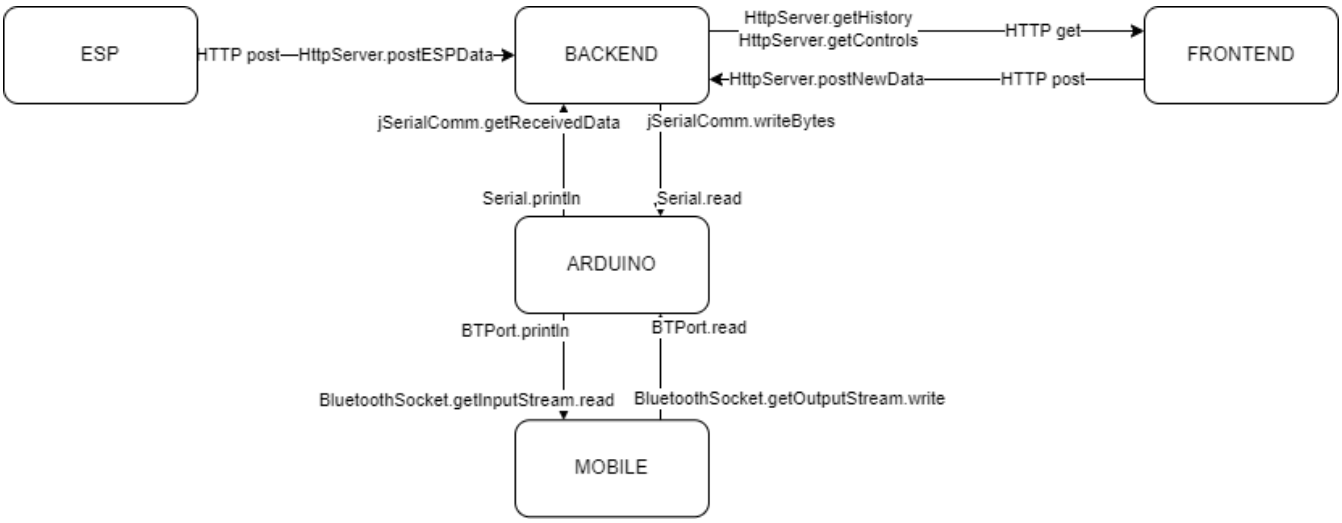

Diagramma 1.6: comunicazioni

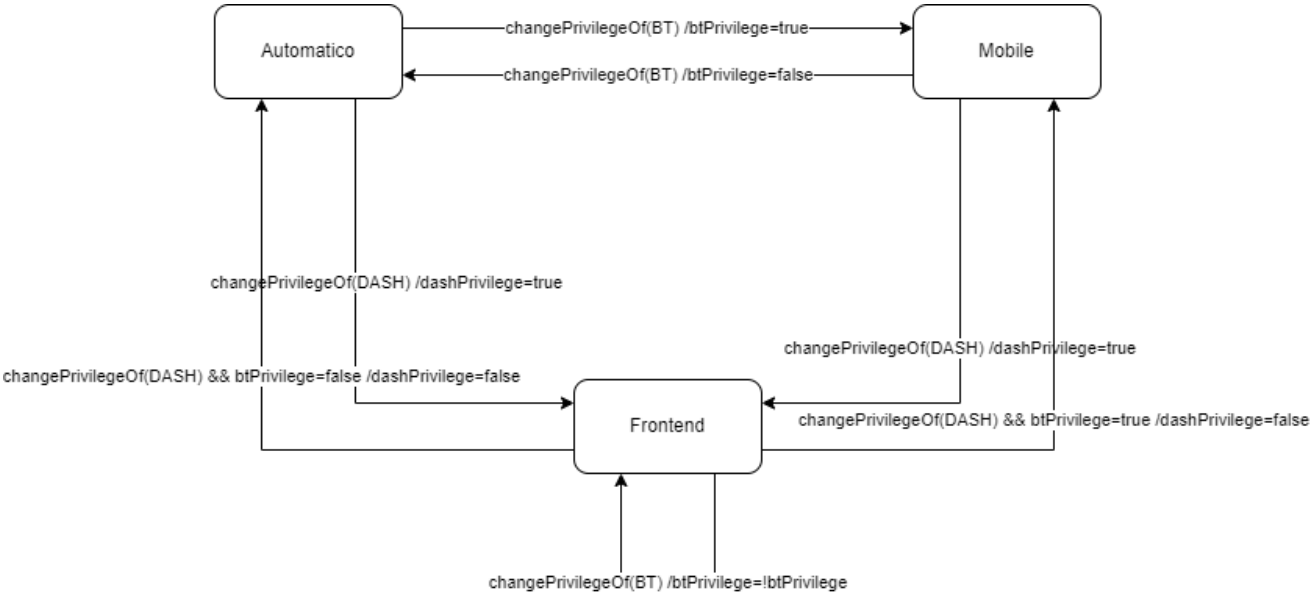

Diagramma 1.7: precedenze