Analisi

Abbiamo deciso di spendere buona parte del tempo a nostra disposizione per la progettazione del sistema, in maniera tale da poterci ben distribuire il carico di lavoro.
In questa fase abbiamo preso in considerazione le problematiche e i requisiti del progetto e le abbiamo strutturate seguendo una metodologia bottom up.
In particolare siamo partiti dal comportamento specifico di ogni task, e come collegare gli uni agli altri per poi raggrupparli in macro task che stavano a simboleggiare le modalità manual, single e auto.
In particolare abbiamo scelto di distribuire il carico di lavoro in tre parti.

Micelli Leonardo: Implementazione task su arduino.

Rossi Luca: Implementazione gui e scambio messaggi parte java

Schiaroli Davide: Scambio messaggi lato arduino e pair programming per parte dei task.

Implementazione

TASK:

Per progettare la parte software del sistema siamo partiti creando le interfacce delle componenti hardware, abbiamo poi implementato i vari task, che andavano ad estendere la classe task.
Ci siamo serviti delle librerie consigliate per il Servo e le abbiamo implementate così come la classe per lo scambio di messaggi.
La classe principale è EventCheck, un task che permette il cambio di modalità ed è responsabile di far partire/fermare i task corretti.
Impostando cosi il progetto abbiamo lasciato al file principale, il .ino solo l’inizializzazione dei componenti e lo scheduling di EventCheck

Leonardo Micelli
Luca Rossi
Davide Schiaroli

GUI:

La GUI è stata sviluppata cercando di seguire le il pattern mvc per dare maggior libertà di movimento e un grado maggiore di libertà su come sviluppare il sw.
Il concetto alla base è quello di disegnare la GUI, assegnando ad ogni pulsante un relativo metodo da richiamare tramite controller ad esempio per mandare le varie modalità, velocità o angoli (manual).
Il controller è inoltre incaricato di fare la sincronizzazione iniziale con arduino e di far partire un task per l'ascolto di messaggi da seriale.
Per lo scambio di messaggi si è deciso di usare la classe SerialCommChannel.

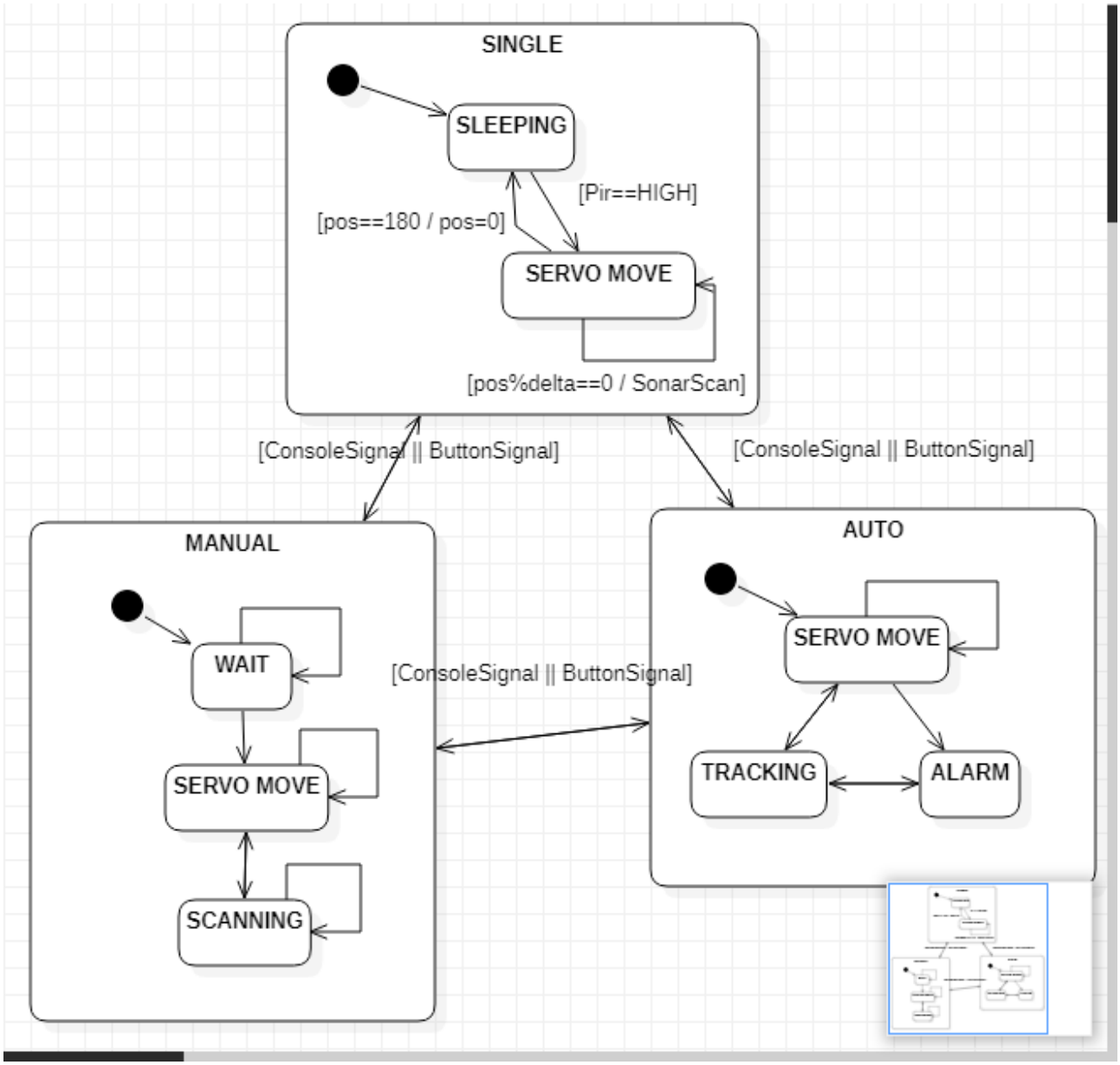
Conclusione

Durante il collaudo abbiamo affrontato diverse problematiche, ad esempio abbiamo dovuto cambiare la libreria che gestiva il servomotore poiché utilizzando un timer hardware di Arduino, lo stesso che utilizza lo Scheduler, la scheda andava incontro a malfunzionamenti.
Anche il thread che gestiva la lettura dei messaggi ha dato problemi, bloccandosi a volte nella ricezione dei messaggi da seriale.

Complessivamente il carico di lavoro si è rivelato equilibrato e c'è stata una buona comunicazione all'interno del gruppo che ci ha permesso di venire a capo dei principali problemi in un tempo ragionevole.

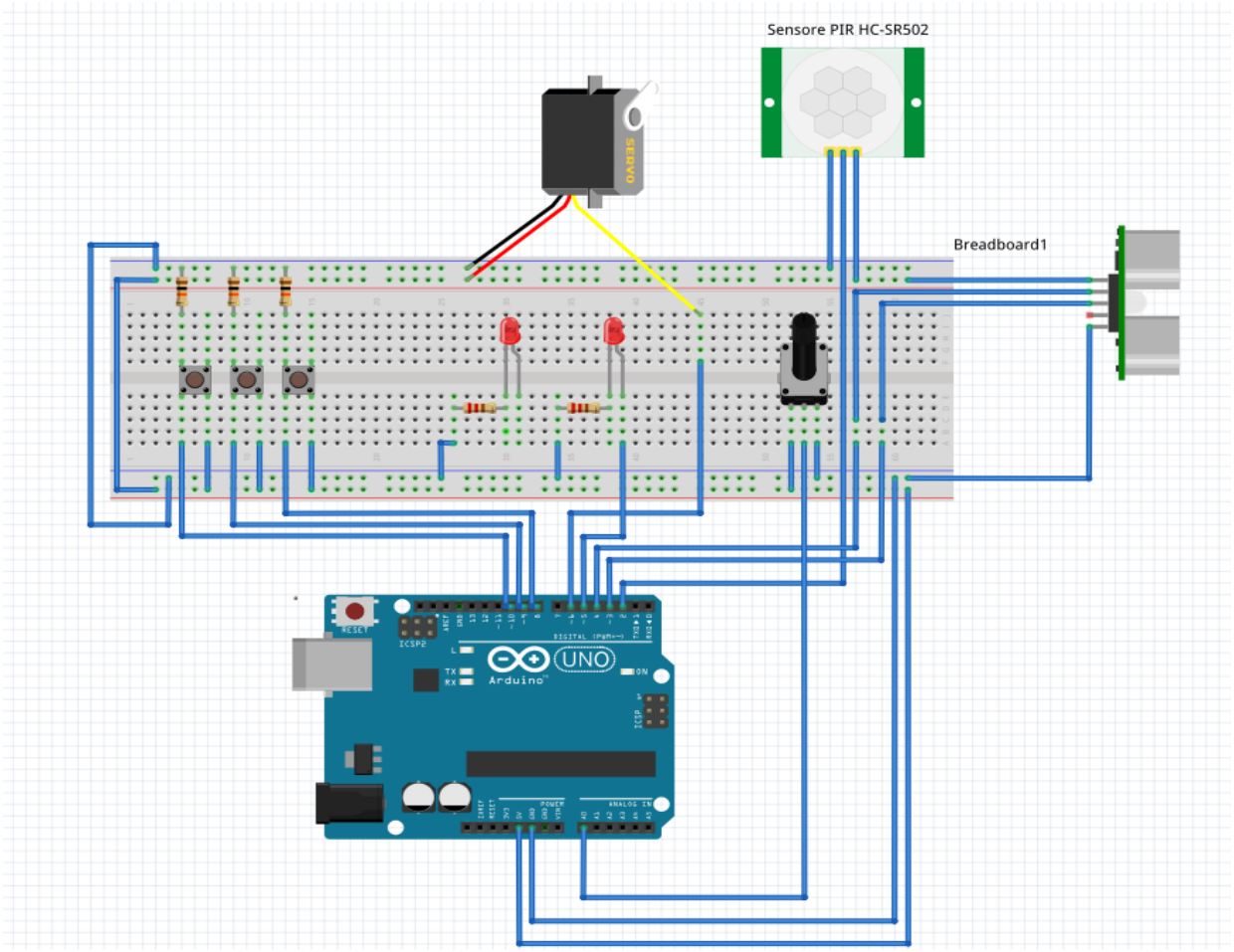
Leonardo Micelli
Luca Rossi
Davide Schiaroli

Schema a stati



Leonardo Micelli
Luca Rossi
Davide Schiaroli

Schema di fritzing



Leonardo Micelli
Luca Rossi
Davide Schiaroli