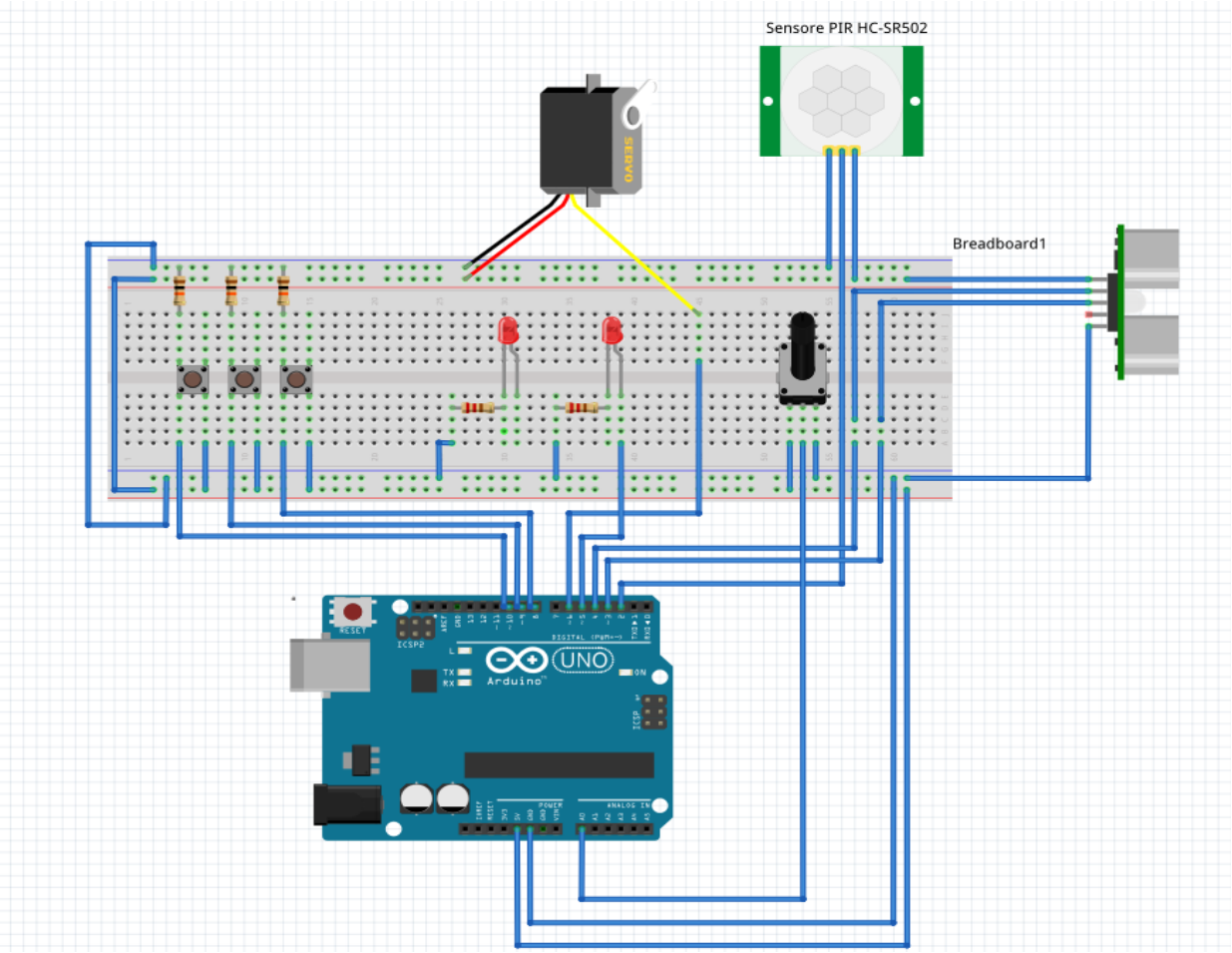
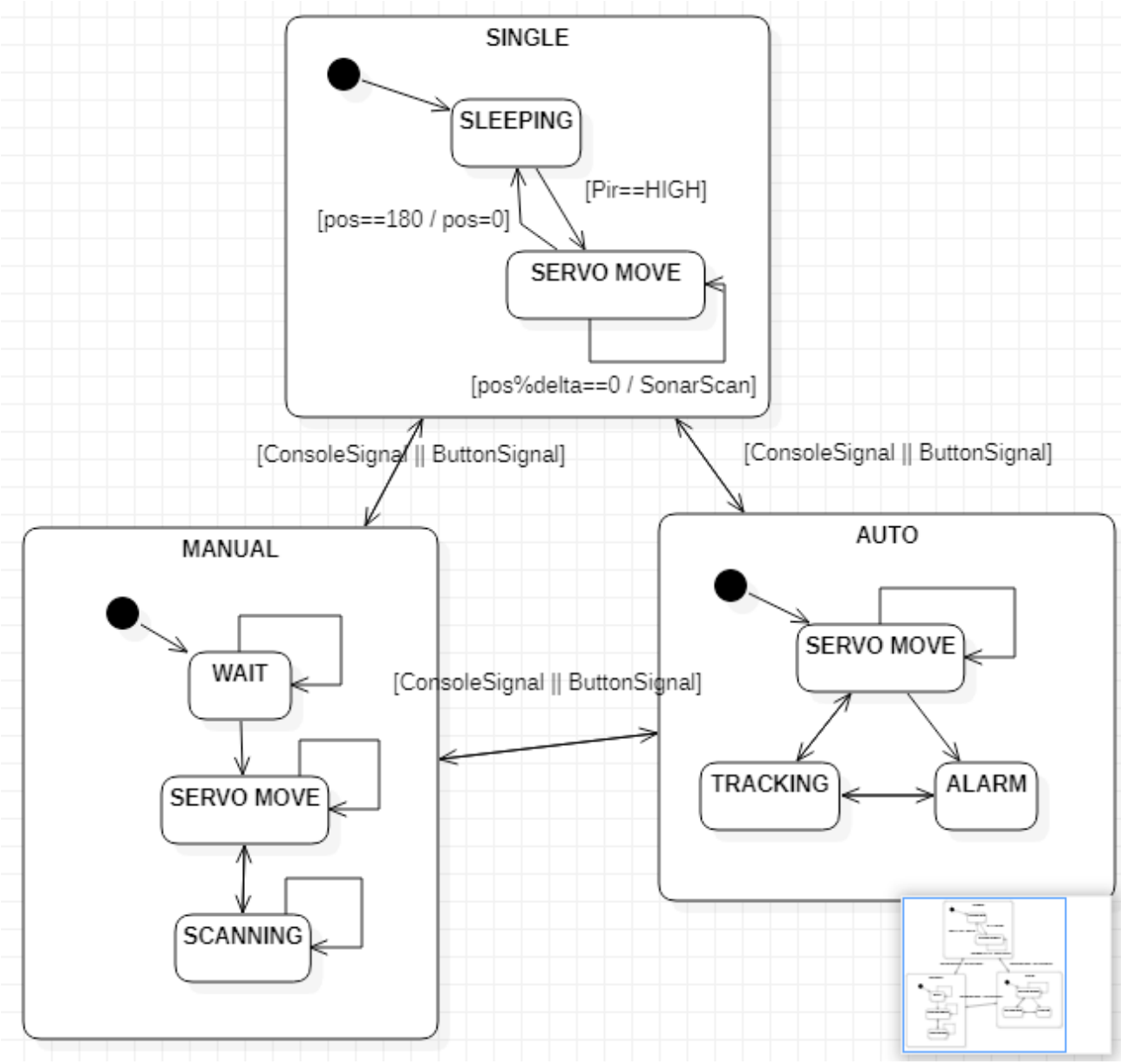
Analisi e Progettazione

Il sistema prodotto è stato costruito seguendo le specifiche raccolte, in una prima fase di analisi, dal documento relativo al porogetto a noi fornito. Il sistema è costituito da Arduino, collegato ad un circuito ,formato da una breadboard collegata a sensori (un sonar, un pir) e attuatori (due led, un servo motore, al quale è stato collegato il sonar), un potenziometro e tre tasti , e ad un' applicazione Console che si interfaccia via seriale al sistema, fungendo da controller.



Leonardo Micelli
Luca Rossi
Davide Schiaroli

In questa prima 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 modellare il comportamento specifico del sistema in ogni sua distinta modalità di esecuzione, rappresentandolo attraverso una macchina a stati finiti come in seguito riportata:



Leonardo Micelli
Luca Rossi
Davide Schiaroli

Suddivisione del carico di lavoro

Micelli Leonardo: Implementazione task su arduino.

Rossi Luca: Implementazione gui e scambio messaggi parte java

Schiaroli Davide: Scambio messaggi lato arduino

Implementazione

ARDUINO:

Per implementare la parte software del sistema siamo partiti creando le interfacce delle componenti hardware, abbiamo poi implementato i vari task. Ci siamo serviti di librerie per la gestione delle periferiche (come ad esempio il Servo) e dello scambio seriale di messaggi fra Arduino e la console.
L'organizzazione in task del programma prevede una struttura gerarchica nella quale il task EventCheck è incaricato di ricevere messaggi dalla console o dal circuito e di attivare il macro task corrispondente alla modalità di esecuzione più appropriata. Infine ogni macro task è incaricato di attivare e coordinare i task relativi alle periferiche. Al modulo Arduino rimane solo il compito di istanziare e di inizializzare in fase di setup i vari task e di schedularli in fase di loop.

CONSOLE:

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.

Leonardo Micelli
Luca Rossi
Davide Schiaroli

Conclusione e Commenti

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