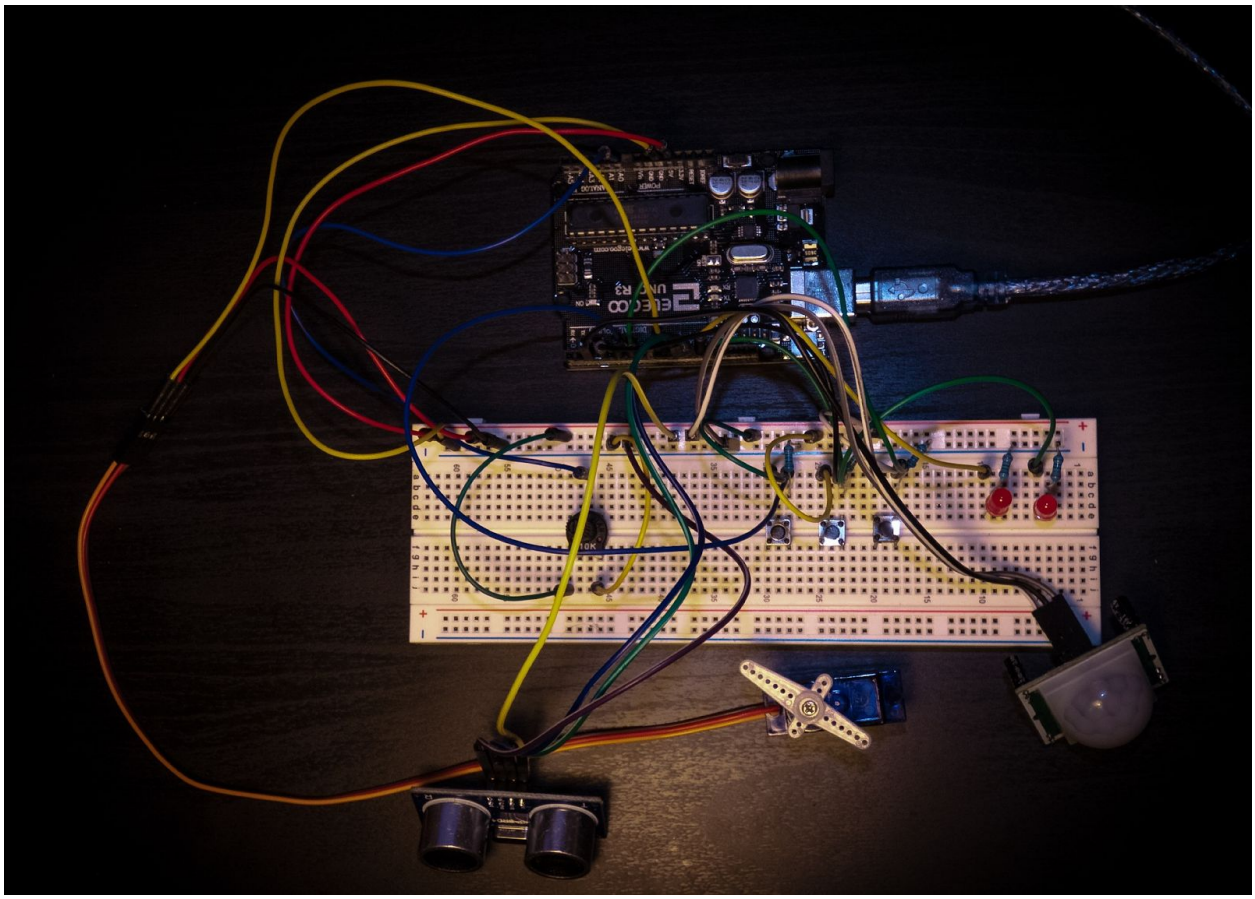
Sistemi embedded and Internet of things - a.a. 2019-2020

Relazione progetto #2 - Smart radar



Progetto realizzato dal gruppo formato da:

Abu Ismam

Hu Marco

Mattiussi Vlad

Indice

Capitolo 1: Introduzione

Il sistema è stato realizzato attraverso l'utilizzo di un Arduino basato sul microcontrollore ATmega328, del sensore pir HC-SR501, del servomotore, del sensore di prossimità HC-SR04, tre bottoni e due led. Il sistema comunica con l'utente mediante un'interfaccia grafica realizzata in Java.

Capitolo 2: Architettura

Come architettura orientata agli oggetti per la progettazione del programma per Arduino si è utilizzata la decomposizione in task, grazie ad essa il programma risulta modulare, chiaro, estendibile e riusabile.

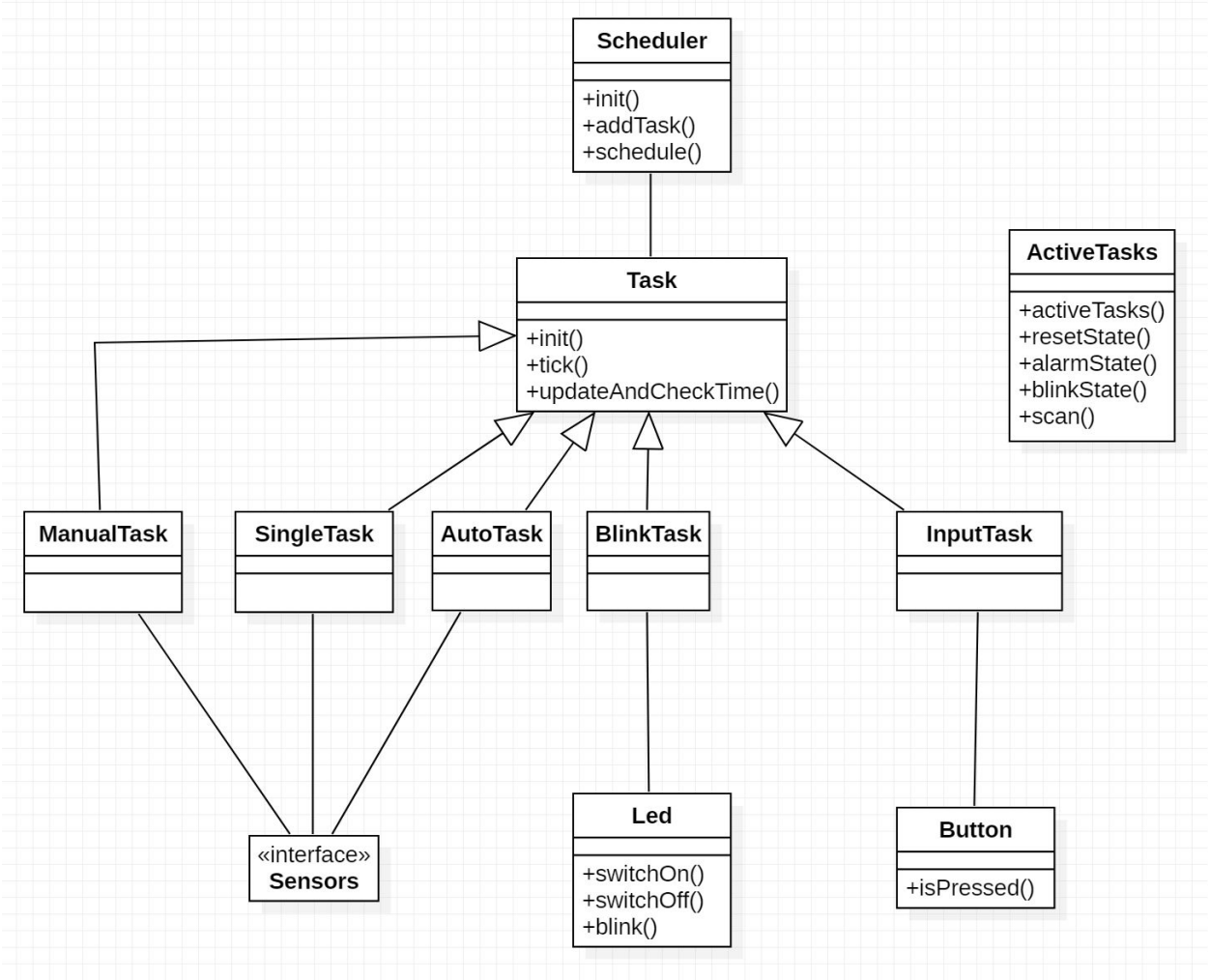
I tre task principali sono ManualTask, SingleTask e Autotask, ovvero le tre modalità di funzionamento del programma, in aggiunta a loro abbiamo anche un BlinkTask che gestisce i due led ed un InputTask che interagisce con tutti gli altri task e provvede alla loro attivazione.

InputTask è sempre attivo e grazie ai suoi metodi attiva opportunamente tutti gli altri task in maniera che il programma funzioni correttamente.

ActiveTasks è una variabile globale. Permette di far comunicare i task tra loro.. Attenzione ActiveTasks non è un task, quindi non estende la classe task.

Lo scheduler, ogni volta che viene chiamato, tramite l'array di Task al suo interno esegue ogni task. Ogni volta che scatta il timer dello scheduler ad ogni task viene eseguito un tick. Nel nostro caso ogni 50ms viene chiamato tick di ogni task.

Il periodo dello scheduler è il massimo comune divisore tra tutti i periodi dei task.



Nell'immagine un diagramma che mostra le principali entità del programma.

L'interfaccia "Sensors" nell'immagine rappresenta pir, potenziometro e il sensore ad ultrasuoni.

Nel file "Config.h" sono presenti tutti i define dei vari pin e costanti del programma.

MsgService è una classe utilizzata per la comunicazione seriale tra Java e Arduino.

Capitolo 3: Sviluppo

3.1) Sviluppo Arduino

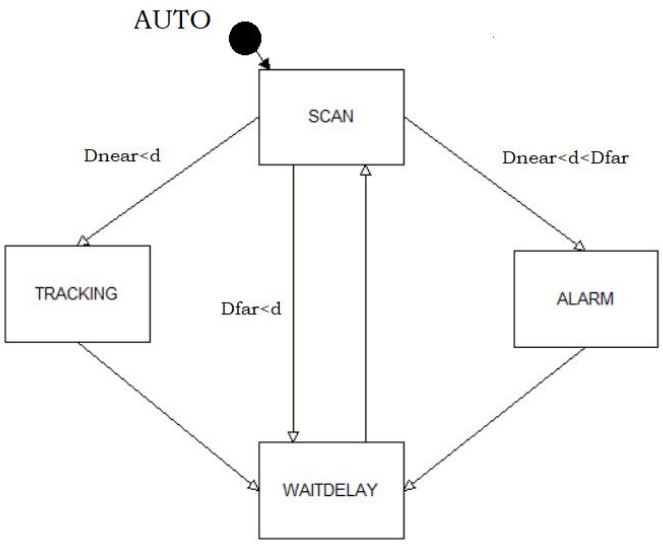
La nostra è una macchina a stati finiti, i tre principali task possiedono i propri stati.

Ogni task durante la sua esecuzione si troverà in un determinato stato, il passaggio di stato all'interno dei task si è realizzato grazie a condizioni e variabili globali condivise ("activeTask").

3.1.1) AutoTask:

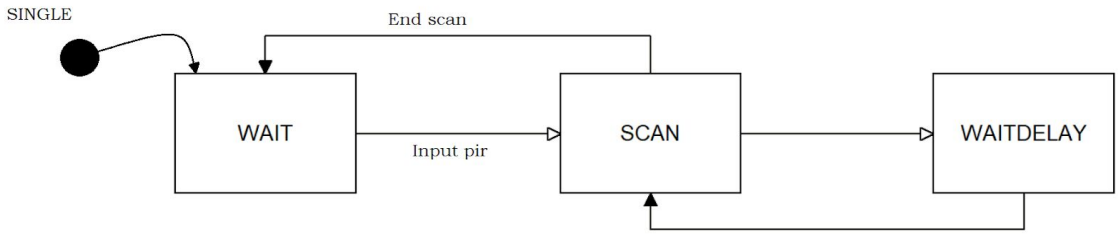
- SCAN: Scansione, una volta finito vado in uno dei tre casi successivi
  TRACKING: Se trovo qualcosa sotto Dnear
- ALARMING: Se trovo qualcosa tra Dnear e Dfar.
- WAITDELAY: Se non trovo nulla prima di Dfar. Aspetto lo sliceTime.

Sia SCAN, che ALARM che TRACKING una volta terminati passano a WAITDELAY. Tracking ha un autoanello che fa cambiare lo stato da TRACKING a TRACKING finché la condizione sia vera, ovvero che sia presente un oggetto sotto Dnear.



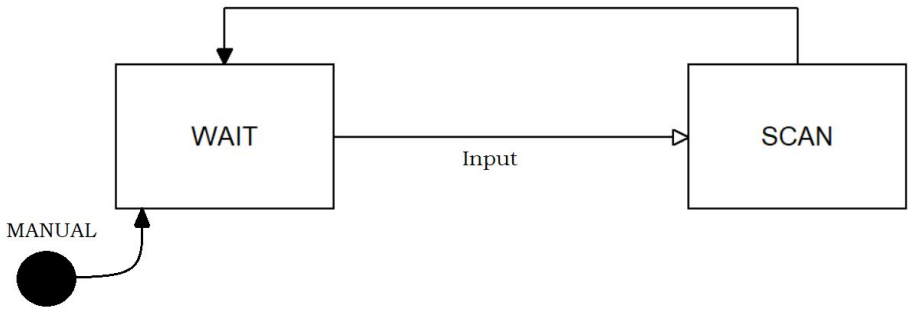
3.1.2) SingleTask:

- WAIT: In questo stato si attende finché il pir non rileva qualche movimento.
- SCAN: Una volta che il pir ha rilevato un movimento il servo inizia a girare e vengono fatte le scansioni, dopo ogni scansione si va in WAITDELAY.
- WAITDELAY: Aspetta un delay, ovvero lo sliceTime che non è altro che il tempo di attesa tra una scansione e l'altra, in pratica il tempo per tra uno spostamento e l'altro.



3.1.3) ManualTask:

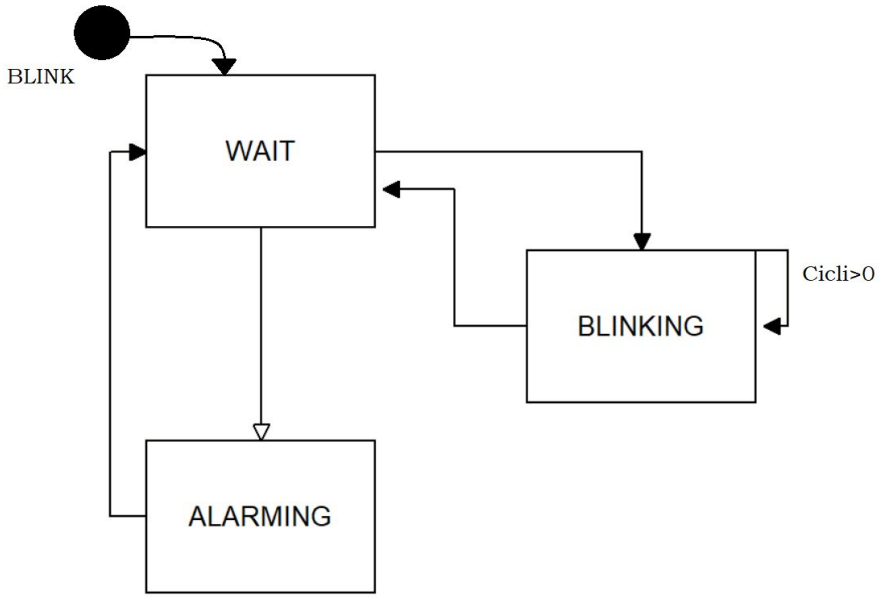
- WAIT: Si rimane in attesa finché non si riceve un segnale dall'utente mediante GUI
- SCAN: Il servo si posiziona nella posizione richiesta dall'utente ed esegue una scansione, una volta terminata si ritorna in WAIT.



3.1.4) BlinkTask

- WAIT: Attende finché non viene richiamato e può andare sia in BLINKING che in ALARMING
- BLINKING: Esegue il blinking e continua finché non viene fermato
- ALARMING: Esegue l'alarming e continua finché non viene fermato

Una volta terminati ALARMING e BLINKING tornato nello stato di WAIT



3.2) Sviluppo Java

La GUI di Java è stata realizzata mediante JavaFX e creando la grafica tramite SceneBuilder.

Come libreria per poter connettere arduino e inviare/ricevere dati viene utilizzata la JSSC.

Capitolo 4) Guida utente

La porta seriale di arduino va scelta dalla checkbox e successivamente va aperto il seriale tramite l'apposito bottone.

Una volta aperto il seriale si attiveranno i bottoni per selezionare la modalità di funzionamento, si attiva inoltre anche il bottone per chiudere il seriale.

Attivando la modalità Manual permette di rendono cliccabili il bottone e l'annessa textField per ruotare servo mentre le attivando modalità Auto e Single si rendono operativi il bottone e le textFiled per cambiare il tempo del ciclo.

Chiudendo la finestra la connessione seriale viene chiusa.

Nella console o sul seriale di Java vengono stampati i vari messaggi in arrivo da Arduino.