SISTEMI EMBEDDED E IoT – a.a 2021-22

**Progetto #2 - Smart Coffee Machine**

Il sistema è composto da due sottosistemi:

* Sottosistema su Arduino
* Sottosistema su PC

I due sottosistemi comunicano mediante scambio di messaggi via Seriale.

**Sottosistema Arduino**

La soluzione proposta organizza il comportamento in 6 task:

* WELCOME: è la modalità iniziale in cui si trova la macchina.

All’avvio viene prodotto un messaggio di benvenuto e viene inizializzato il numero di prodotti.

* IDLE: è la modalità generale di funzionamento della smart coffee machine.

In questa fase attraverso i bottoni è possibile selezionare il prodotto desiderato ed attraverso il potenziometro è possibile scegliere il livello dello zucchero. Il prodotto è disponibile solo se il numero di articoli è > 0, altrimenti entra in modalità MAINTENANCE.

* MAKE: Gestione della fase di produzione e di recupero di un caffè.

In questa fase viene prodotto a video un messaggio che indica quale tipologia di bevanda la macchina sta facendo.

Internamente questo task è diviso in due stati: make e remove.

Make viene utilizzato per simulare tutto il processo di produzione, mentre remove viene utilizzato per simulare la (eventuale) rimozione da parte dell’utente.

Il processo di realizzazione viene simulato attraverso il motore. Al termine del processo di produzione viene visualizzato il messaggio che indica che il prodotto è pronto.

La fase di rimozione invece è articolata come segue: una volta pronto il prodotto deve essere rimosso dall’utente, cioè se la distanza misurata dal sonar è > 40 cm il prodotto è considerato rimosso, altrimenti la macchina attende la rimozione. Se l’utente non preleva il prodotto entro 5 secondi o se la distanza rilevata dal sonar è sempre < 40 cm, il motore viene riportato a 0.

* STAND BY: questa modalità gestisce lo stand-by della macchina.

La macchina entra in questa modalità se la macchina è rimasta inattiva per più di 60 sec ed il pir non rileva nessun utente.

* CHECK: gestione della fase di autotest della macchina.

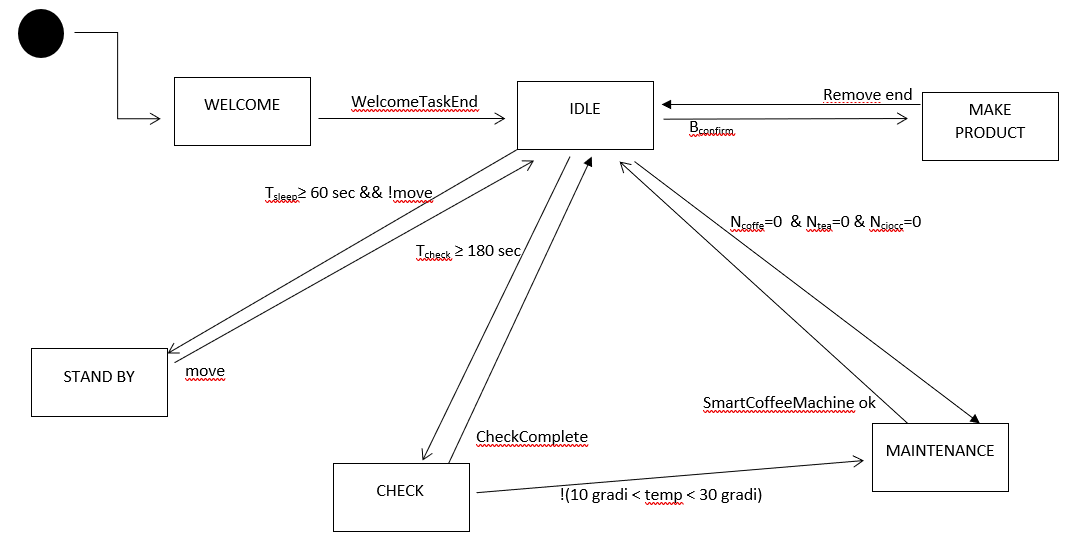
La macchina esegue una rotazione completa del motore e verifica il valore della temperatura attraverso il sensore di temperatura. Se la temperatura rilevata dal sensore non è compresa tra [10,30] gradi la macchina entra in modalità MAINTENANCE. Alternativamente la macchina torna allo stato IDLE se il check è stato completato correttamente.

* MAINTENANCE: Gestione della fase di manutenzione.

La macchina entra in questa modalità se nessuno dei tre prodotti è presente (viene prodotto un messaggio che indica la necessità di ricevere assistenza con un refill) oppure se il valore del sensore di temperatura non è compreso tra 10 e 30 gradi.

In questa modalità non è possibile realizzare prodotti finché non si riceve l’assistenza necessaria. Il normale comportamento della macchina viene ripreso tramite l’applicazione Coffee Machine Manager che permette di effettuare la ricarica oppure il recover in caso di temperatura non compresa nel range [10,30].

Il comportamento dei task è dato dalle FSM sincrone in figura:



Risorse usate dai task:

* 3 bottoni: Bup, Bdown, Bmake per selezione e confermare il prodotto nella fase di Idle.
* Potenziometro: per sezionare il livello di zucchero desiderato nella fase di Idle.
* Display: per visualizzare cosa sta facendo la macchina.
* Pir: per rilevare un movimento da parte dell’utente nella fase di Stand-by.
* Sonar: viene utilizzato per la fase di rimozione del prodotto, rileva rispettivamente se la distanza è maggiore o minore di 40cm.
* Servo motore: per simulare il processo di realizzazione nella fase di Make, Remove e Check.
* Sensore di temperatura: per rilevare la temperatura della macchina nelle fasi di Check e Maintenance.

------------------------------------------------------

**Sottosistema Java/PC**

Il sottosistema è composto dai seguenti elementi:

* SmartCoffeeMachine.java
  + Contiene il main e i listener dei bottoni.
* SCMWindow.java
  + Contiene l’implementazione della GUI e dei vari metodi utilizzati.
* CommChannel.java & SerialCommChannel.java
  + Contengono l’interfaccia e la classe che la implementa per poter analizzare il flusso di dati nella seriale e controllarne le interazioni.