Report assigment 2

Smart coffee machine

Nicolò Malucelli
11/05/2022

Schema generale dell'applicazione

Nella figura 1.1 è mostrato lo schema generale dell'applicazione e di come i diversi task interagiscono tra loro attraverso l'uso di variabili condivise. In seguito, saranno mostrate in dettaglio le singole macchine a stati finiti. In questa figura sono inoltre mostrati gli input e gli output della *"Smart coffee machine"*.

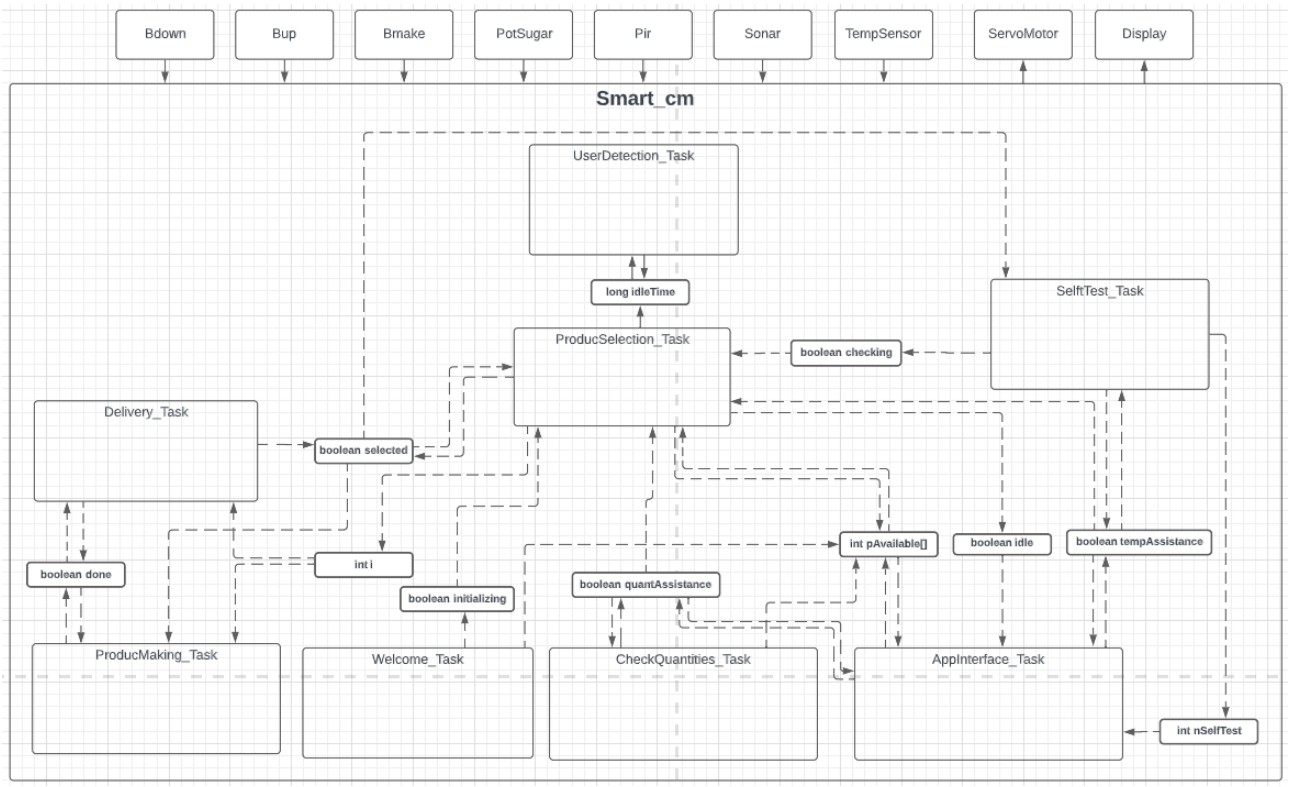

Fig.1.1 interazioni tra task

Analisi dettagliata dei task

Task di benvenuto

Il task di benvenuto (fig.2.1) è un semplice task che ha il compito di inizializzare il numero di quantità disponibili delle varie bevande e mostrare un messaggio di benvenuto per i primi secondi dall'avvio dell'applicazione.

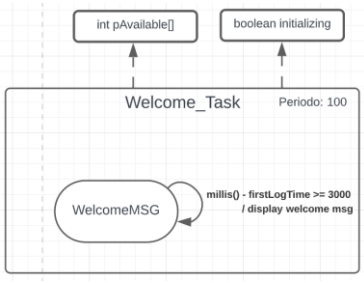

Fig 2.1 welcome task

Selezione, produzione e consegna

I tre principali task che governano il funzionamento della Smart coffee machine sono quelli di selezione, produzione e consegna. Ogni task è rappresentato da una *FSM* e interagisce con gli altri attraverso l'uso di variabili condivise che permettono di sincronizzare correttamente le varie operazioni. (fig.2.2)

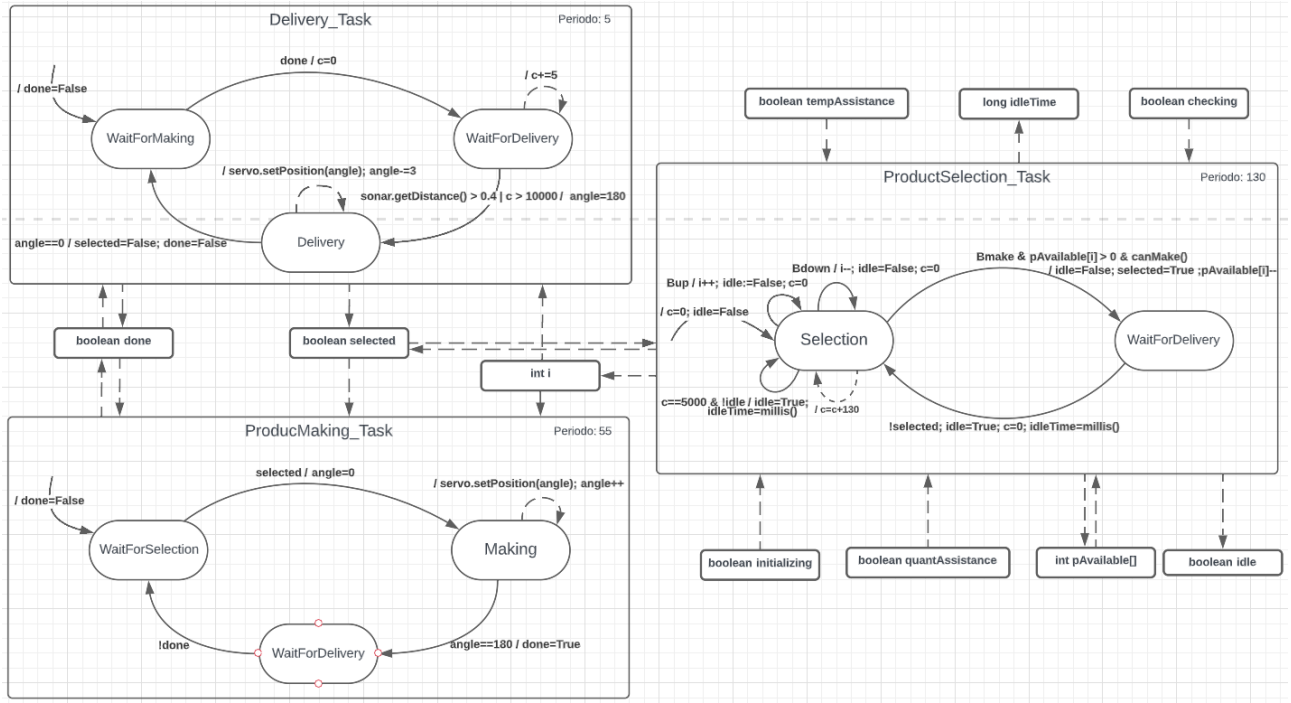

Fig2.2 FSM dei Task di selezione, produzione e consegna

Task di selezione

Il task di selezione è l'unico che interagisce fisicamente con l'utente attraverso i pulsanti (*Bup*, *Bdown* e *Bmake*) ed il potenziometro (*PotSugar*). Dovendo questo task gestire diversi input è senza dubbi quello più complesso dal punto di vista logico, poiché deve valutare lo stato di diverse variabili globali per poter gestire ogni interazione con l'utente. All'avvio della macchina l'*FSM* di questo task si trova in stato di *selection*. È possibile scorrere le bevande attraverso i *Bup* e *Bdown* e richiedere la bevanda attraverso *Bmake* nel caso in cui quest'ultima sia disponibile. Una volta ordinata la produzione di una bevanda, la *FSM* entra in stato *waitForDelivery*, da cui potrà uscire soltanto quando lo permetterà il task di consegna, come descritto in seguito.

Task di produzione

Il task di produzione si trova all'avvio nello stato di *waitForSelection*. La *FSM* può uscire da questo stato soltanto quando viene ordinata una bevanda, ovvero quando il product selection task setta a *true* la variabile *selected*. Una volta nello stato di *making* il servo comincia a ruotare ed una volta raggiunti i 180 gradi entra in fase di *waitForDelivery*, impostando la variabile *done* a *true* e sbloccando così il task di Delivery.

Task di consegna

Il task di delivery si comporta in modo simile al task di produzione, con la differenza che esce dallo stato di *waitForMaking* in base al valore della variabile *done*. Una volta in stato di *waitForDelivery* controlla periodicamente il valore della distanza attraverso il sonar. Non appena una delle due condizioni descritte nello schema è soddisfatta, il task entra in fase di delivery ruotando il servo da 180 gradi a 0. All'uscita da questo stato, il task setta a *false* le variabili *selected* e *done* permettendo così ai task di selezione e produzione di ricominciare la loro attività dal loro stato iniziale.

Modalità assistenza

Gli unici due task che permettono alla *Smart coffee machine* di entrare in stato di assistenza sono il *SelfTest_Task* e il *CheckQuantities_Task.*

Task di self-test

Il task di self test (fig.2.3) effettua un test ogni 3 minuti. Il test è diviso in due fasi, *checkA* e *checkB*. Durante il *checkA* il servo viene fatto ruotare da 0 a 180 gradi mentre durante il *checkB* ruota da 180 a 0. Terminata la seconda fase di test viene misurata la temperatura attraverso l'apposito sensore. Nel caso in cui la temperatura sia fuori dal range prefissato, il task setta a *true* la variabile *tempAssistance* bloccando così il normale funzionamento della macchina.

Per come è stata strutturata l'applicazione non è possibile effettuare un test mentre un prodotto è in fase di produzione, così come non è possibile produrre una bevanda mentre la macchina sta effettuando un self-test.

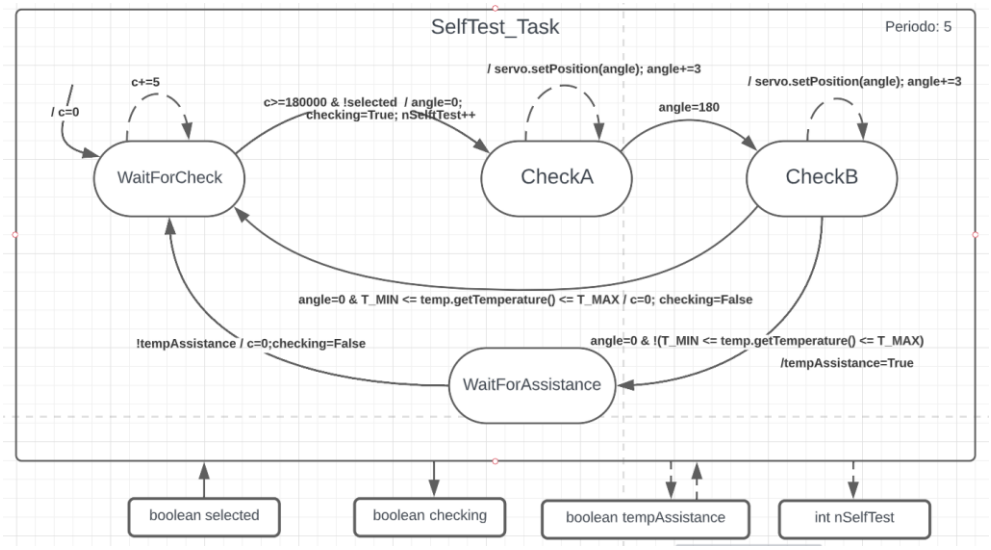

Fig 2.3 FSM del task di selft-test nel dettaglio

Task di controllo delle quantità

Questo task (fig.2.4) a differenza di altri non è propriamente una *FSM* poiché al suo interno svolge una singola operazione ogni periodo di tempo. Il periodo di tempo è lo stesso del task di selezione poiché è solo quest'ultimo che può ridurre il numero di prodotti disponibili. Sarebbe stato possibile inserire quindi il controllo delle quantità all'interno del task di selezione, tuttavia, si è preferito separare le due attività che sono in realtà ben distinte.

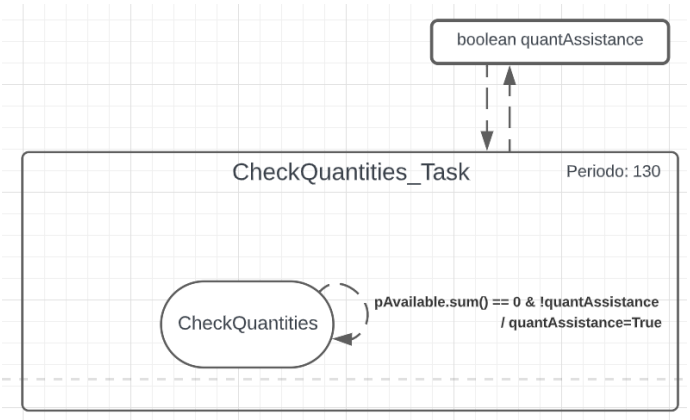

Fig 2.4 task di controllo delle quantità

Task di user detection

Così come i task di benvenuto e controllo delle quantità, anche il task di user detection (fig.2.5) non è una vera *FSM*. Questo task manda in sleep *Arduino* nel caso in cui l'applicazione si trovi in modalità idle per più di 60 secondi e nel caso in cui il pir non rilevi la presenza di un utente nelle vicinanze. Questo task è l'unico tra tutti a fare l'uso degli *interrupt*. Va sottolineato che il task effettua il *detatch* dell'*interrupt* non appena il sistema esce dalla modalità di *sleep* ed effettua l'*attach* soltanto quando strettamente necessario per evitare che il normale funzionamento dello *scheduler* sia interrotto da interrupt inutili.

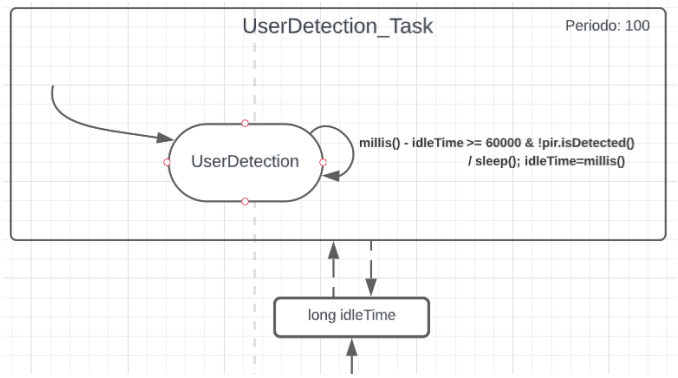

Fig 2.5 task di user detection nel dettaglio

Task di interfacciamento con l'applicazione desktop

Questo task si occupa di aggiornare l'applicazione desktop in seguito ad un cambiamento di stato nella *Smart coffee machine* e aggiornare la *Smart coffee machine* a seguito di un'azione svolta nell'applicazione desktop (*refill* o *resume*). In particolare, questo task permette di uscire dallo stato di assistenza, settando a *false* l'opportuna variabile di assistenza a seguito di un messaggio ricevuto dall'applicazione desktop.

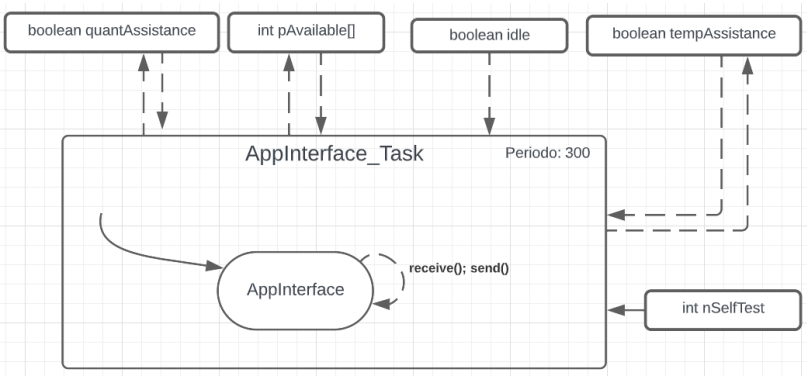

Fig 2.6 task di interfacciamento con l'applicazione desktop

Schema fisico dell’applicazione