**TESTPLAN**

**SPRINT 00 – Analisi dei requisiti**

Obiettivi:

* testare la ricezione di messaggi da parte del maître
* testare la comunicazione tra maître e robot

**Pre condizioni**: il maître è in attesa di ricevere un comando

* il maître una volta ricevuto il comando lo inoltra al robot, che lo esegue

**Post condizioni**: il maître e il robot sono nello stato relativo all’esecuzione del comando inviato, alla fine di ogni task il robot è nello stato “ended”

Nello specifico, in questo primo Test Plan vogliamo verificare i comandi:

* prepare
* stop
* reactivate
* roomstate
* add
* check\_food
* clear

**SPRINT 01 – Analisi del problema**

Obiettivi:

* testare la ricezione di eventi da parte del maître
* testare la comunicazione tra maître e robot
* testare la comunicazione tra robot e fridge

**Pre condizioni**: il maître è in attesa di ricevere un evento

* il maître una volta ricevuto l’evento lo inoltra al robot, che lo esegue e quando è necessario coinvolge anche il frigo.

**Post condizioni**: il maître e il robot sono nello stato relativo all’esecuzione del comando inviato, quando il comando è “add” il frigo è nello stato di “checkfood”

Nello specifico, in questo primo Test Plan vogliamo verificare l’invio di eventi:

* prepare\_button
* stop\_button
* reactivate\_button
* roomstate
* add\_button
* fridgestate
* clear\_button

**SPRINT 02 – Frigo Smart**

Obiettivi:

* testare il funzionamento dell’interazione con il frigo

**ADD**

**Pre condizioni**: maître, robot e fridge sono in attesa di comandi

* si invia un evento “add\_button(pizza,4)” al maître

**Post condizioni**: il maître si deve trovare in uno stato di “add”, il robot in “add\_started” e il frigo in “check\_food”

**EXPOSE FRIDGE STATE**

**Pre condizioni**: il fridge è in attesa di comandi

* si invia un evento “exposefridgestate” al frigo

**Post condizioni**: il fridge si deve trovare in uno stato di “exposestate”

**TAKE FOOD**

**Pre condizioni**: il fridge è in attesa di comandi

* si invia un evento “take\_food(pizza,4)” al frigo

**Post condizioni**: il maître si deve trovare in uno stato di “takefood”

**SPRINT 03 – Prepare**

Obiettivi:

* testare la funzionalità “prepare”

**Pre condizioni**: robot, robotexecutor e robotmover sono in attesa di comandi

* si invia un messaggio di “prepare” al robot

**Post condizioni**: il robotexecutor, dopo aver eseguito tutti i passi, ritorna allo stato “athome” e il robotmover è in stato “notmoving”.

Nello specifico viene testato che il robot si trovi nello stato “handlingprepare” e che il robotxecutor e il robotmover facciano i seguenti step:

Da RH a pantry => robotexecutor: startprepare

robotmover: moving

At pantry => robotexecutor: atpantry

robotmover: moving

Da pantry a table => robotexecutor: attable\_dishes

robotmover: moving

Da table a fridge => robotexecutor: atfridge

robotmover: moving

Da fridge a table => robotexecutor: attable\_food

robotmover: moving

Da table a RH => robotexecutor: athome

robotmover: notmoving

**SPRINT 04 – Add**

Obiettivi:

* testare la funzionalità “add”

**Pre condizioni**: robot, robotexecutor e robotmover sono in attesa di comandi

* si invia un messaggio di “add(pizza, 4)” al robot

**Post condizioni**: il robotexecutor, dopo aver eseguito tutti i passi, ritorna allo stato “athome” e il robotmover è in stato “notmoving”.

Nello specifico viene testato che il robot si trovi nello stato “handlingadd” e che il robotexecutor e il robotmover facciano i seguenti step:

Da RH a fridge => robotexecutor: startadd

robotmover: moving

At fridge => robotexecutor: atfridge

robotmover: moving

Da fridge a table => robotexecutor: attable\_food

robotmover: moving

Da table a RH => robotexecutor: athome

robotmover: notmoving

**SPRINT 05 – Clear**

Obiettivi:

* testare la funzionalità “clear”

**Pre condizioni**: robot, robotexecutor e robotmover sono in attesa di comandi

* si invia un messaggio di “clear” al robot

**Post condizioni**: il robotexecutor, dopo aver eseguito tutti i passi, ritorna allo stato “athome” e il robotmover è in stato “notmoving”.

Nello specifico viene testato che il robot si trovi nello stato “handlingclear” e che il robotexecutor e il robotmover facciano i seguenti step:

Da RH a table => robotexecutor: startclear

robotmover: moving

At table => robotexecutor: attable

robotmover: moving

Da table a fridge => robotexecutor: atfridge

robotmover: moving

Da fridge a table => robotexecutor: attable

robotmover: moving

Da table a dishwasher => robotexecutor: atdishwasher

robotmover: moving

Da dishwasher a RH => robotexecutor: athome

robotmover: notmoving

**SPRINT 06 – Roomstate**

Obiettivi:

* testare la funzionalità “exposeroomstate”

**Pre condizioni**: il roomstate è in attesa di comandi

* si invia un messaggio di “exposeroomstate” al roomstate

**Post condizioni**: il roomstate si trova nello stato exposeroomstate

**SPRINT 08 – Movimento con planner**

Obiettivi:

* testare che il robotmover arrivi alla posizione corretta (cella della mappa), grazie al planner

**DESTINAZIONE CORRETTA**

**Pre condizioni**: il robotmover è in attesa di comandi, la posizione del robot nel planner è (0,0)

* si invia un messaggio "goto(fridge)" al robotmover

**Post condizioni**: la posizione del planner è (5,0) e il robotmover si trova nello stato “notmoving”

**DESTINAZIONE ERRATA**

**Pre condizioni**: il robotmover è in attesa di comandi, la posizione del robot nel planner è (0,0)

* si invia un messaggio "goto(casamia)" al robotmover

**Post condizioni**: il robotmover si trova nello stato “notmoving”

**SPRINT 09 – Maître and frontend**

Obiettivi:

* testare la ricezione di eventi da parte del maître

**Pre condizioni**: maître e robot sono in attesa di comandi

* si invia un evento “prepare\_button” al maître

**Post condizioni**: il maître si trova nello stato “prepare” e il robot in “handlingprepare”

**SPRINT 10 – Stop e riattiva**

Obiettivi:

* testare la funzionalità stop
* testare la funzionalità riattiva

**STOP**

**Pre condizioni**: il robot è in attesa di comandi

* si invia un messaggio “prepare” al robot (in modo che ci sia un task in esecuzione)
* si invia un messaggio “stop” al robot

**Post condizioni**: il robot si trova nello stato “stopped”

**RIATTIVA**

**Pre condizioni**: il robot si trova nello stato “stopped”

* si invia un messaggio “reactivate” al robot

**Post condizioni**: il robot si trova nello stato “reactivated”