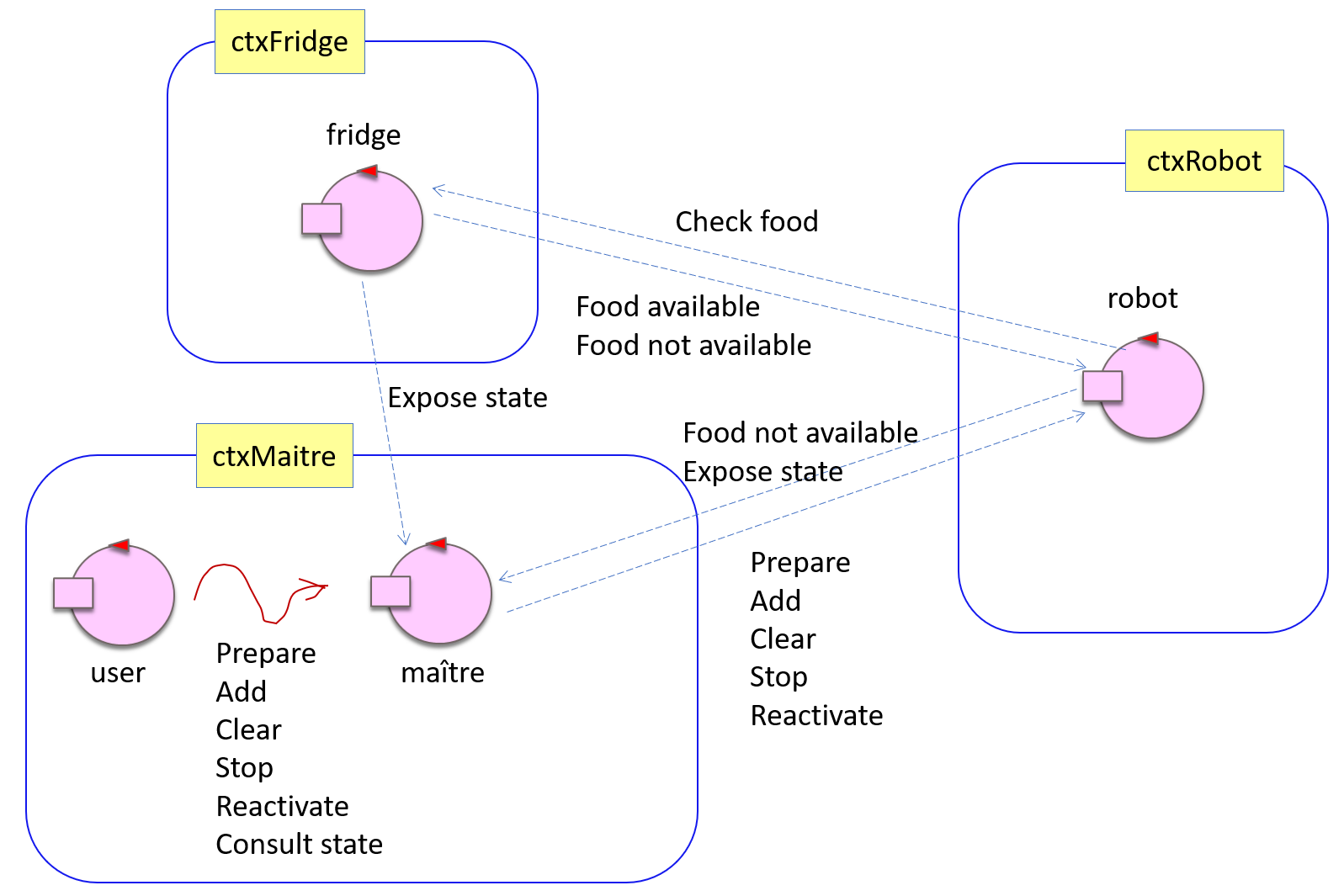
**ANALISI DEL PROBLEMA**

L’**architettura logica** del sistema è composta da quattro entità attive, modellabili come attori: il robot, il frigo, l’utente (maître) e l’applicazione sullo smartphone del maître.



* A questo livello il robot è un’entità logica che nasconde sia i dettagli relativi al funzionamento dei task che deve compiere, sia i dettagli relativi al dispositivo (fisico o virtuale) che rappresenta.
* L’entità maitre rappresenta il software che deve ricevere i comandi dell’utente e inoltrarli al robot.
* Il modello implementa un attore mock che rappresenta l’utente e i comandi che invia all’applicazione.
* Il requisito “Consult room state” implica la presenza di un’entità che mantenga lo stato delle risorse non smart (tavolo, dispensa e lavastoviglie), o in alternativa l’integrazione di questa conoscenza in una delle entità già esistenti.

Avendo a disposizione il modello dei QActor non c’è **abstraction gap** e le risorse necessarie per la costruzione del modello sono limitate.

**TestPlan**

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