??/??/2020

**PIAZZA\_SPRINT 2**

## Sprint Goal

-

## Requisiti

-

## Analisi dei Requisiti

## Analisi del Problema

Quali problematiche sorgono?

Controlla sempre l’overview iniziale perché alcune potrebbe essere necessario riprenderle taggandole.

Problematiche ancora aperte derivanti dall’overview :

sincronizzare il comportamento del cliente con quello del waiter

Al momento non è richiesto di implementare un’applicazione per il cliente 🡪 per stare sul semplice lo simuliamo. Come? Immaginiamo il cliente come un semplice pacco da prendere, portare al tavolo ecc.. la palla ce l’ha il waiter.

Abbandonare la soluzione del modello a stati del cliente, per passare a un client\_simulator più semplice.

Una soluzione può essere la seguente:

*Un attore, chiamato simulator, che non fa altro che stare in ascolto di messaggi provenienti dal (waiter?) dove viene detto:*

*-table\_reached(CID) 🡪 quando lo riceve, dopo tot manda un messaggio readyToOrder(CID)*

*-tea\_served(CID) 🡪 quando lo riceve, dopo tot manda un messaggio readyToPay(CID), solo se non è già stato cacciato per aver superato maxstaytime (come controllarlo?)*

*-NON fare tempi random altrimenti perdi il controllo della situazione in debug.*

*L’interazione con la smartbell verrebbe poi totalmente rimossa e sostituita con degli auto-msg ring che si manda da sola la smartbell: OK!*

Interazioni Wi-Fi

Stato logico del waiter

Stato dei teatable

Il waiter, o qualcuno per lui, deve tenere traccia della situazione corrente della TeaRoom

Momento in cui occupare un tavolo

Countdown del maxStayTime

Stima del maxWaitingTime