ColdStorageSprint0V3

Introduction

Goal dello sprint0

- individuare un architettura logica iniziale che definisca le macro-entità del sistema e le loro interazioni
- definire un piano di lavoro iniziale

Requirements

I requisiti sono scritti dal committente TemaFinale23.

Requirements analysis

- **SERVICE AREA** area di lavoro del transport trolley
- **INDOOR** zona della service area nell'angolo in basso a sinistra della stanza
- **COLDROOM** entità caratterizzata da una porta con una posizione non precisabile dai requisiti e una capacità di carico
- **TRANSPORT TROLLEY** entità logica capace di spostarsi all'interno della service area
- **DDR ROBOT** soggetto fisico grazie al quale si implementano le azioni logiche compito del transport trolley
- Il committente ha predisposto il software per modellare il ddr robot tramite l'entità astratta BasicRobot. Le **API del componente basicrobot** sono le seguenti:
  - **Request moverobot : moverobot(TARGETX, TARGETY)**
  - **Reply moverobotdone : moverobotok(ARG)**
  - **Reply moverobotfailed: moverobotfailed(ARG)**
- Il transport trolley è quell'entità della applicazione che ha la responsabilità di controllare il basicrobot, il quale ci è stato già consegnato dal committente.
- ServiceAccessGUI è un'entità formalizzata come un attore che ha la responsabilità di inviare messaggi alla ColdStorageService e di interagire con l'utente
- ColdStorageService è l'entità core business del sistema e siccome il sistema è distribuito, allora ColdStorageService è modellata come un attore
- Cold Room viene modellata come un attore di modo da interagire con ColdStorageService.
- KEYPOINT: essa potrebbe essere modellata come un POJO all'interno di ColdStorageService stesso, ma riteniamo che modellando Cold Room come un attore porti ai seguenti vantaggi:
  - si alleggerisce ColdStorageService di un altro compito, dato che ColdStorageService è l'entità core business
  - si segue il principio di singola responsabilità
  - se verranno previste in futuro una entità di "Scarico della Cold Room", allora questa entità potrà interagire direttamente con la Cold Room, senza dover interagire con ColdStorageService

Dai requisiti possiamo asserire che:

- il sistema è **distribuito** su più nodi di elaborazione;

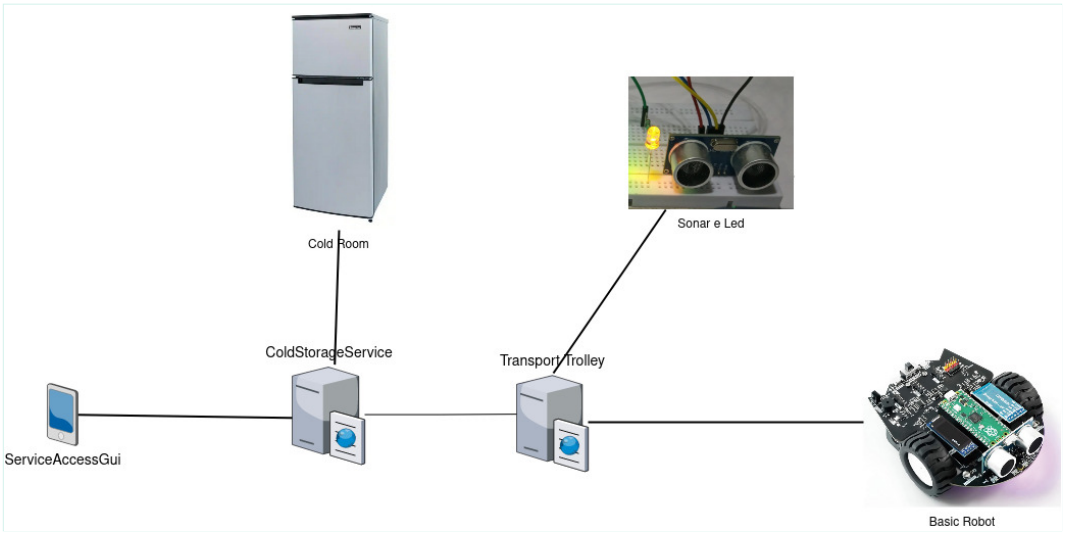
Dialogo con il committente

Dopo opportuni colloqui con il committente, possiano affermare che :

- le operazioni di carico e di scarico della ColdRoom potrebbero essere effettuate in parallelo oppure in maniera sequenziale. Per semplicità di realizzazione, dato che il committente non ha espresso riflessioni in materia, vengono effettuate in maniera sequenziale, ma nel caso realistico esse verrebbero fatte in parallelo.
- Non deve succedere che un camion, ricevuto il proprio ticket si veda rifiutata l'operazione di scarico una volta arrivato in INDOOR.
- E' possibile che il transport trolley non riesca a scaricare un intero truck tutto in un solo viaggio. Il committente ha affermato che non è una casistica da prendere in considerazione.
- la richiesta di un ticket può avvenire mentre sono ancora in corso operazioni di scarico precedenti

Macrocomponenti

- Le macroentità sono:
  - *ColdStorageService*
  - *Transport Trolley*
  - *Sonar e Led*
  - *ServiceAcessGUI*
  - *ColdRoom*



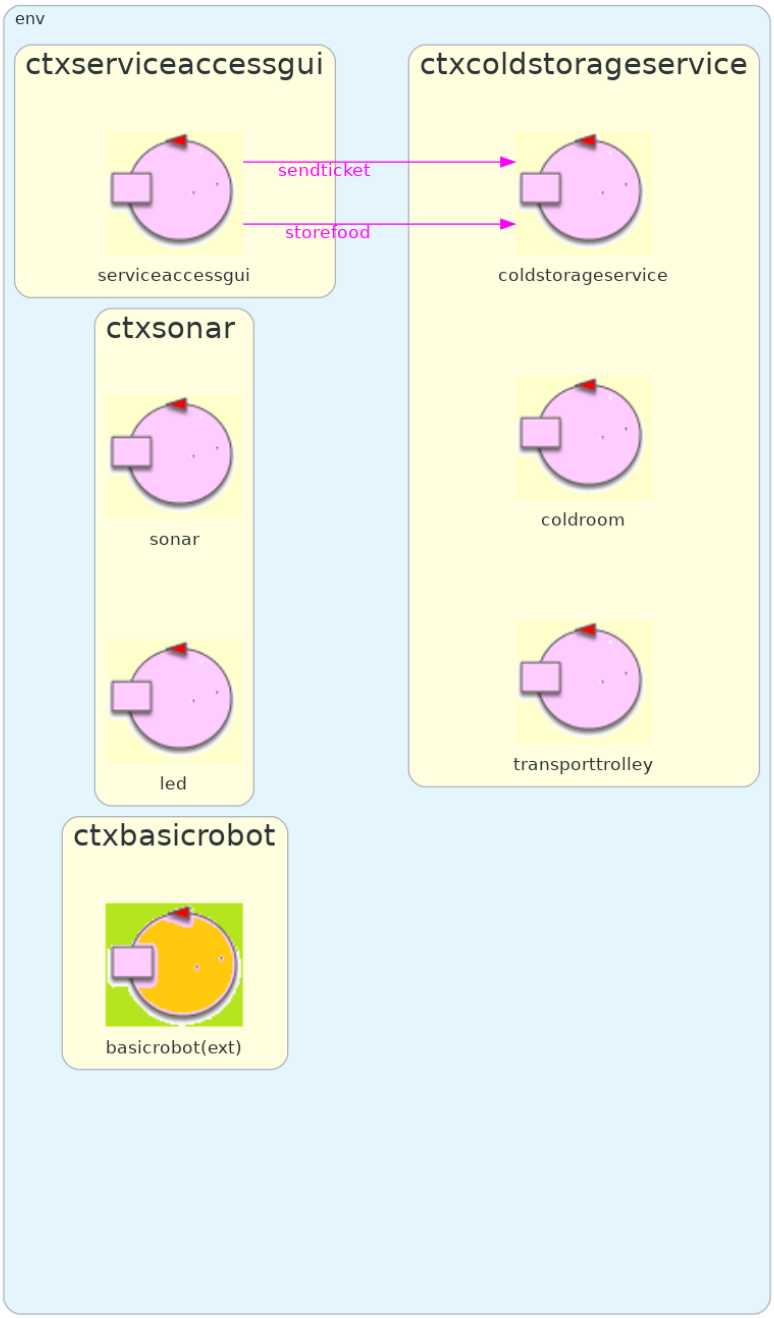
Architettura logica

Il sistema è composto da:

- ***ColdStorageService***: prende in carico richieste di generazione ticket e richieste di invio di un camion; si interfaccia con la ColdRoom per saperne lo stato; fa partire le deposit action; E' inserita in un suo contesto perchè è la core business.
- ***Transport Trolley***: invia al Basic Robot la sequenza di comandi necessari per effettuare una deposit action
- ***Basic Robot***: esegue i comandi del Transport Trolley. E' in un contesto esterno dato che è un servizio datoci dal committente.
- ***Sonar***: effettua la misurazione e emette eventi. E' inserito in un contesto diverso perchè è specificato nei requisiti che è presente un Raspberry alla quale sarà collegato.
- ***ServiceAccessGui***: si interfaccia con ColdStorageService per la richiesa di ticket. E' inserita in un suo contesto
- ***Cold Room***: aggiorna lo stato della quantità di kg
  Come si legge nei requisiti, le interazioni tra le entità sono

Sender	Receiver	Description	Messaggio	Notes
ServiceAccessGui	ColdStorageService	send to the ColdStorageService a request to store new **FW** kg of food	Request storefood : storefood(KG)	frase letterale dei requisiti
ColdStorageService	ServiceAccessGui	**Reply to storefood** If the request is accepted, the services return a ticket [with the ticket number]	Reply ticketaccepted : ticketaccepted(TICKETCODE)	frase letterale dei requisiti
ColdStorageService	ServiceAccessGui	**Reply to storefood** Quando la Cold Room ha finito lo spazio disponibile, allora non può essere emesso il ticket	Reply ticketdenied : ticketdenied(ARG)	il committente ha specificato che l'utente non può rimanere bloccato su questa richiesta
ServiceAccessGui	ColdStorageService	the driver uses the ServiceAcessGUI to enter the ticket number and waits until the message **charge taken**	Request sendticket: sendticket(TICKETCODE)	frase letterale dei requisiti
ColdStorageService	ServiceAccessGui	messaggio che permette al truck di presentarsi ad INDOOR	Reply chargetaken : chargetaken(ARG)	frase letterale dei requisiti

il codice della seguente architettura puo essere visionato al seguente link: https://github.com/RootLeo00/robot-coldstorage/blob/sprint0/src/coldstorageservice.qak



coldstorageserviceArch

Piano di lavoro

prevediamo di suddividere in sprint lo sviluppo del sistema secondo il seguente elenco

SPRINT	GOAL	TEMPO STIMATO	NOTE
SPRINT 1	a) prototipazione del core business ColdStorageService	2 giorni	si prevede di svolgere la fase a e b) in parallelo in quanto non fortemente dipendenti tra loro
	b) creazione di una infrastruttura containerizzata per facilitare il testing	3 ore	si prevede di svolgere la fase a e b) in parallelo in quanto non fortemente dipendenti tra loro
	c) testing	2 ore	
SPRINT 2	a) introduzione della la GUI nel sistema	1 giorno	
	b) test della gui	2 ore	
	c) test totale del sistema	2 ore	
SPRINT 3	a) introduzione del alarm requirement nel prototipo sviluppato allo sprint1	1 giorno	
	b) test alarm requirement	2 ore	
	c) test totale del sistema	2 ore	
SPRINT 4	a) deployment su robot fisico	3 ore	

Lo sprint 3 e lo sprint 4 possono essere realizzati in parallelo.

By Caterina Leonelli email: caterina.leonelli2@studio.unibo.it, GIT repo: https://github.com/RootLeo00/sw-eng.git

By Matteo Longhi email: matteo.longhi5@studio.unibo.it GIT repo: https://github.com/carnivuth/iss_2023_matteo_longhi.git