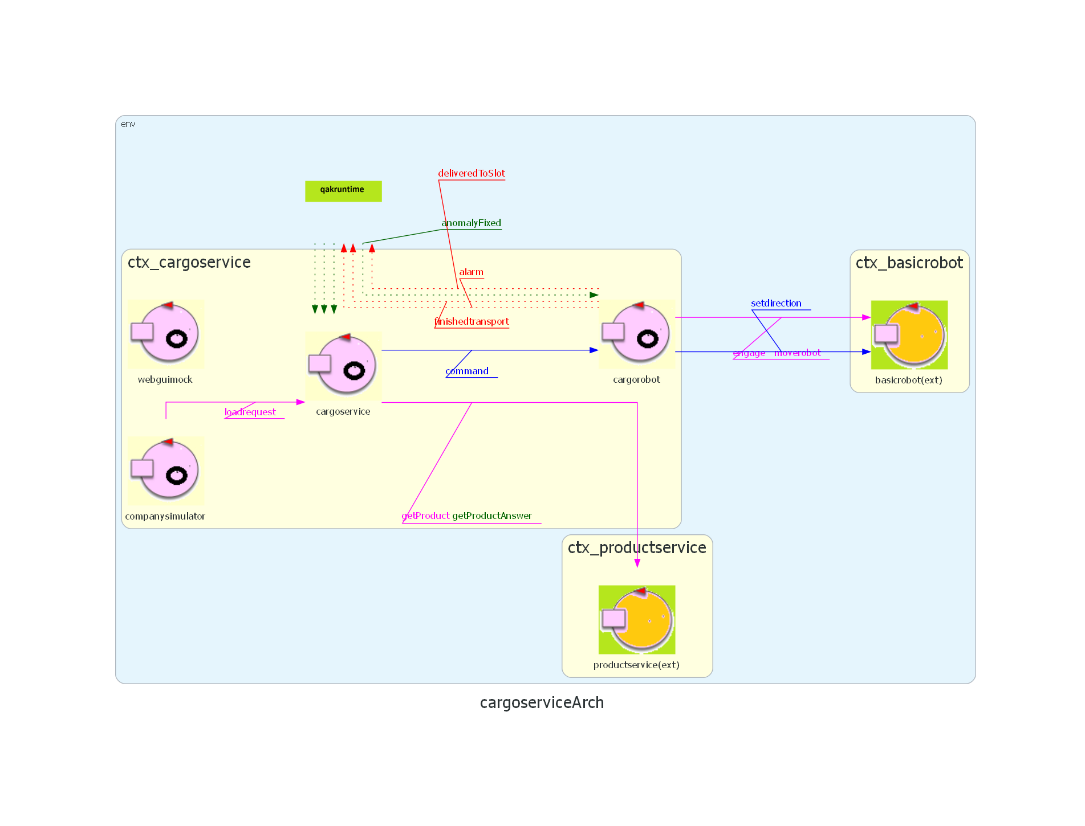
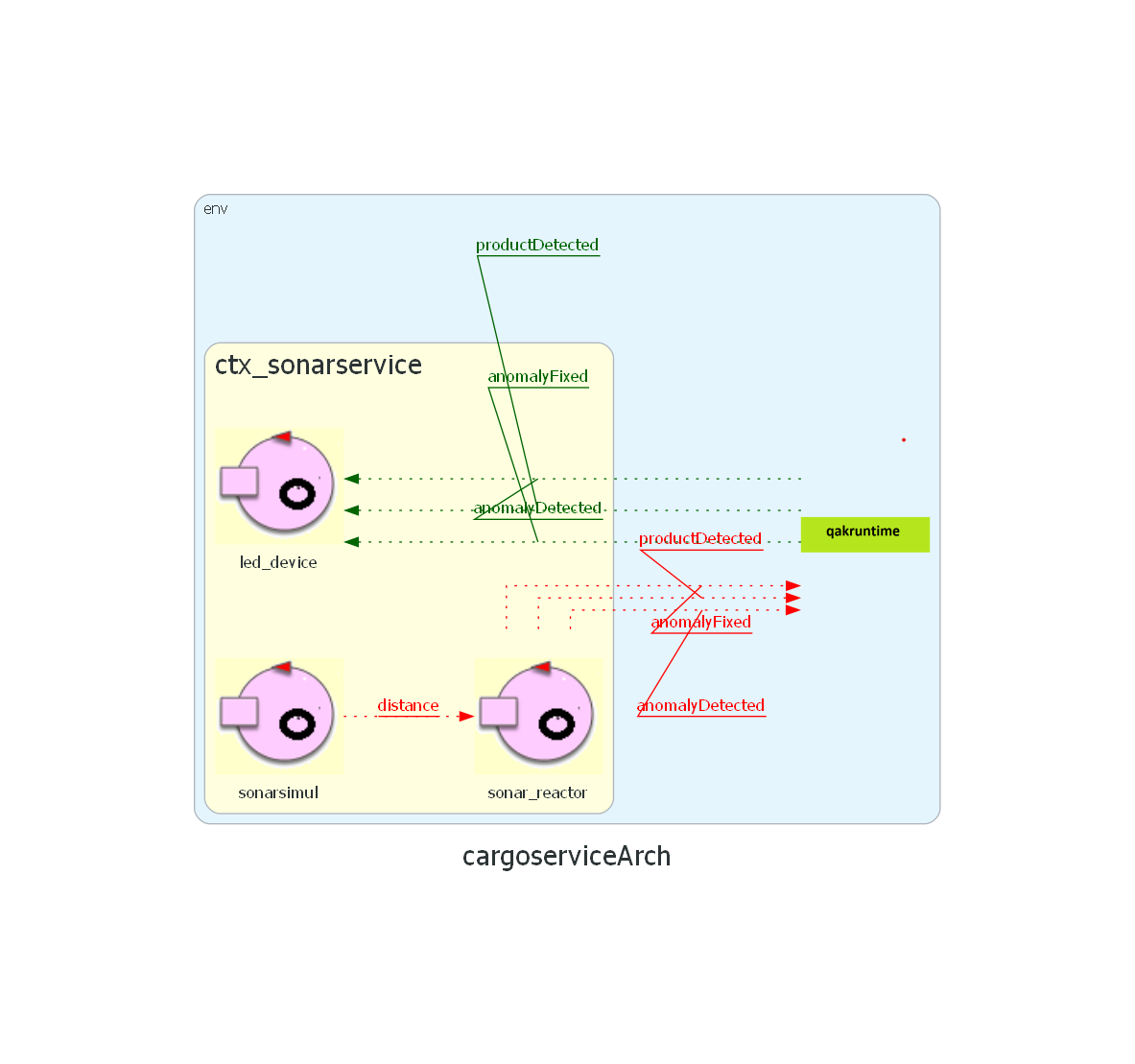
**SPRINT 3:**

**Punto di Partenza:**

****Nello sprint2 il team ha implementato il componente sonarservice, sostituendolo al mock object corrispondente creato durante lo sprint1, ottenendo la seguente architettura:

Si vuole fornire per comprensione un vocabolario riassuntivo di termini definiti nei precedenti sprint: **Vocabolario:**

|  |  |
| --- | --- |
| Termine | Significato attribuito |
| Container | Contenitore in cui viene inserito il prodotto |
| loadrequest /richiesta di carico | Richiesta mandata dalla compagnia, specificando un PID |
| prodotto | Nel sistema è l’oggetto trasportato dal robot, la cui presenza può innescare diversi eventi |
| prodotto registrato | Prodotto conosciuto da ProductService a cui è associato un PID e un peso(Weight) |
| Microservizio | Componente software progettato per svolgere una specifica funzione del sistema. Ogni microservizio comunica con gli altri tramite messaggi, rendendo il sistema flessibile e scalabile. |
| GUI (Graphical User Interface) /WebGUI | Interfaccia grafica utente accessibile via web, che consente di visualizzare in tempo reale lo stato della stiva e interagire in modo intuitivo con il sistema. |
| Bounded Context | Il "bounded context" (**contesto limitato**) è un concetto fondamentale nel Domain-Driven Design (DDD) e si riferisce a un ambito applicativo ben definito e autonomo all'interno del quale vengono definite entità, regole e logiche di business in modo univoco e chiaro. All'interno di un bounded context, il significato di ogni entità o concetto è inequivocabile e specifico per quel contesto, evitando ambiguità e conflitti con altri contesti. |
| IOPort | Punto fisico (porta) attraverso il quale i contenitori dei prodotti entrano o escono dalla nave. È il punto in cui il sonar rileva la presenza di un prodotto. |
| Sonar | Sensore a ultrasuoni che misura la distanza tra sé e un oggetto. Nel nostro sistema serve per rilevare se un contenitore è presente all’IOPort. |
| DDRobot | è un robot che utilizza due motori indipendenti per muovere le ruote o i cingoli. È il supporto fisico che viene comandato da cargorobot. |
| PID (Product Identifier) | Numero intero univoco assegnato a ciascun prodotto registrato, usato per tracciarne l'identità all'interno del sistema. |
| Slot | Spazio fisico nella stiva della nave dove può essere posizionato un contenitore. Esistono 4 slot disponibili; uno è sempre occupato (slot5). |
| Cargorobot | Robot mobile autonomo (a guida differenziale) incaricato di trasportare i contenitori dall’IOPort fino allo slot assegnato e poi tornare alla posizione HOME. |
| Stiva | Area rettangolare della nave in cui i contenitori vengono caricati. Contiene gli slot e l’IOPort. |
| ProductService | Microservizio che gestisce la registrazione dei prodotti. Verifica i dati e assegna un PID univoco |
| CargoService | Microservizio che riceve richieste di carico, controlla i vincoli, assegna gli slot e coordina il caricamento tramite cargorobot. |
| SonarService | Microservizio che rileva la presenza di un contenitore all’IOPort tramite i dati forniti dal sonar. |
| DFREE | Distanza soglia usata dal sonar: se la distanza misurata è maggiore di DFREE per 3 secondi, si ipotizza un malfunzionamento del sensore. |
| MaxLoad | Peso massimo complessivo che la nave può sopportare. Il sistema rifiuta richieste che farebbero superare questo limite. |
| Worker | Persona che colloca fisicamente i contenitori sull’IOPort dopo che sono stati registrati. |
| Sistema logico di riferimento | Rappresentazione concettuale dell’intero sistema, con attori, componenti e interazioni, usata come base per l’architettura e la progettazione tecnica. |
| attore | Entità che svolge un ruolo attivo nel sistema, eseguendo azioni e comunicando con gli altri attori attraverso messaggi |
| Linguaggio QAK | Linguaggio modellistico usato per descrivere e simulare il comportamento dei componenti del sistema come “attori” |
| POJO | Plain Old Java Object: un oggetto di una classe in java |
| Anomalia | Nel documento è inteso come un comportamento inatteso di un componente hardware, tale da compromettere il normale funzionamento del sistema. |

**Goal dello Sprint3:**

* Enunciazione esplicita dei requisiti della webgui
* Analisi del problema
* Definizione dell’architettura logica con modello eseguibile
* Progetto e realizzazione della webgui

**Enunciazione esplicita dei requisiti della webgui:**

Il sistema deve fornire un’interfaccia grafica accessibile via web (**webgui**) che consenta di monitorare lo stato della stiva (**hold**) in tempo reale. In particolare:

**RF1. Visualizzazione dello stato della stiva**

* + L’interfaccia deve mostrare lo stato corrente di ciascun **slot** (occupato/libero).
  + Deve essere mostrato anche il **peso complessivo** del carico presente nella stiva.

**RF2. Aggiornamento dinamico**

* + La WebGUI deve aggiornarsi automaticamente al variare dello stato della stiva, senza necessità di ricaricare manualmente la pagina.
  + L’aggiornamento avviene tramite notifiche push, ottenute osservando la risorsa CoAP hold e inoltrate ai client via WebSocket.

**RF3. Accessibilità via Web**

* + La WebGUI deve essere consultabile da un browser attraverso un endpoint HTTP dedicato.
  + Deve essere disponibile almeno una pagina HTML responsiva che consenta all’utente di monitorare lo stato della stiva in maniera chiara e intuitiva.

**Analisi del problema:**

**Modello dei dati da rappresentare**:

Per poter mostrare correttamente lo stato della stiva, è necessario associare all’interfaccia un modello dedicato, **hold**, il cui scopo è quello di mantenere consistenti le informazioni che la riguardano.

Il componente **hold** terrà traccia:

* del peso complessivo degli elementi presenti nella stiva
* dello stato (occupato/libero) di ciascuno slot

Nello **Sprint 1**, cargoservice era stato sviluppato in modo da mantenere direttamente queste informazioni. Con l’aggiunta del componente hold, si è ritenuto opportuno effettuare un **refactoring**, spostando la responsabilità della gestione degli slot su di esso.

In questa nuova architettura, cargoservice si limita a inviare messaggi a hold per richiedere l’aggiornamento dello stato o per ottenere informazioni sugli slot, delegando a quest’ultimo la gestione dei dati.

Verranno utilizzate le classi java definite già nello sprint1 per la gestione degli slot:

* Slot

**Definizione dell’architettura logica con modello eseguibile:**

**Progetto e realizzazione della webgui:**