Relazione finale fase 1: Conway

Ingegneria dei Sistemi Software M

Chergui Jacopo

Link github

1. **Finalità:** apprendere i principali concetti dell'Ingegneria del Software ed applicarli ad un sistema complesso, mescolando concetti teorici con la pratica di implementazioni tecniche al fine di consolidare tali concetti, sottolineando l'importanza di 'pensare' ai giusti principi prima di iniziare ad implementare.
2. **Cosa ho realizzato/sperimentato:** fin'ora ho avuto modo di realizzare e testare a fondo il codice fornito dal docente, implementando personalmente alcune classi, comprendendone logica e funzionamento. Il focus è stato sull'analisi e sull'esecuzione, sull'apprendimento delle best practice.
3. **Cosa ho imparato:** gradle, springboot, MQTT, Docker.
4. **Perché conway:** le regole sono semplici, ma il comportamento che emerge è complesso e non prevedibile, fatto comune nello sviluppo del software. Rappresenta inoltre una buona palestra per design e modularità (strutturazione del codice e manutenibilità). Possibile uso di pattern come MVC, singleton e observer.
5. **Perché Java e Springboot:** Java è un linguaggio ad oggetti che consente facilmente di implementare tutti i concetti chiave dell'ingegneria del software e del clean code. Springboot facilita lo sviluppo e necessita di poche configurazioni.
6. **Concetti swe:** principi SOLID, sistema distribuito a microservizi, design pattern.
7. **Si tratta di un sistema distribuito?** Al momento siamo ancora in una fase di transizione verso il distribuito con due soli microservizi, ma con la struttura del codice e la modularità sarà immediata la distribuzione. Al momento però la computazione 'core' è centralizzata.
8. **Perché librerie custom:** la realizzazione di librerie custom permette all'ingegnere del software di alzare il livello di astrazione per sviluppare un approccio più umano alla risoluzione di problemi, i quali spesso si ripetono e sono risolvibili con soluzioni già pronte al riutilizzo.
9. **Linguaggio come salto evolutivo:** come evidenziato nel punto precedente, il linguaggio può spesso essere customizzato per lo specifico problema da risolvere ma al tempo stesso essere adatto a risolvere problemi comuni ed essere riutilizzato.