

## Il Treno, compito A

Nome _____		Cognome _____	
Matricola _____	Aula _____	Posizione nell'aula _____	
(come da schema, senza contare file e colonne vuote)			
La prova si svolge usando il solo libro UML@Classroom			

cattedra						
	A	B	C	D	E	...
1	A1	B1	C1	D1	E1	...
2	A2	B2	C2	D2	E2	...
...	...	...	...	...	...	...

**Domanda 1a** (Rispondere su questo foglio) chatGPT, alla domanda “Mi trovi: attori, principali e secondari, e casi d'uso per il seguente progetto? <testo progetto>” ha risposto come di seguito.

- Correggere eventuali allucinazioni o errori di chatGPT barrando le parti da cancellare.
- Aggiungere gli attori che eventualmente mancano, aggiungere eventuali casi d'uso mancanti.
- Per i casi d'uso: dire quali dovrebbero essere in relazione includes/extend con un altro caso d'uso (es. dire UC3 includes UC9) e quali dovrebbero essere un passo della sequenza degli eventi di un altro caso d'uso (in questo caso dire ad es. UC3 in UC9).

**Attori principali:** Utenti; Admin; Capostazione; ~~Sistema di Tracciamento (Il Treno);~~ **meglio Utente che Utenti, ma non importante**

**Attori secondari:** Sensori sui Binari; Applicazioni Client; (come detto le app client fanno parte del sistema)

**Casi d'uso:**

**UC 1) Consultazione Orari dei Treni:** Gli utenti accedono al sistema per ottenere informazioni su orari e binari → **Utente...**

UC 2) **Inserimento Orario Stagionale:** L'Admin inserisce l'orario stagionale dei treni

UC 3) **Controllo Capacità Stazioni:** Il sistema verifica la capacità delle stazioni e segnala problemi all'Admin.

UC 4) **Gestione Informazioni Temporanee:** L'Admin aggiunge informazioni temporanee come treni speciali etc

UC 5) **Rilevamento Arrivo/Partenza:** Il capostazione segnala l'arrivo e la partenza dei treni che vengono poi rilevati dai sensori sui binari.

UC 6) **Monitoraggio Ritardi:** Il sistema ~~monitora i ritardi~~ e attiva procedure di gestione dei ritardi.

UC 7) **Selezione Automatica Dati Fermata:** Il sistema seleziona automaticamente i dati associati a una fermata con ritardo e li visualizza per gli utenti.

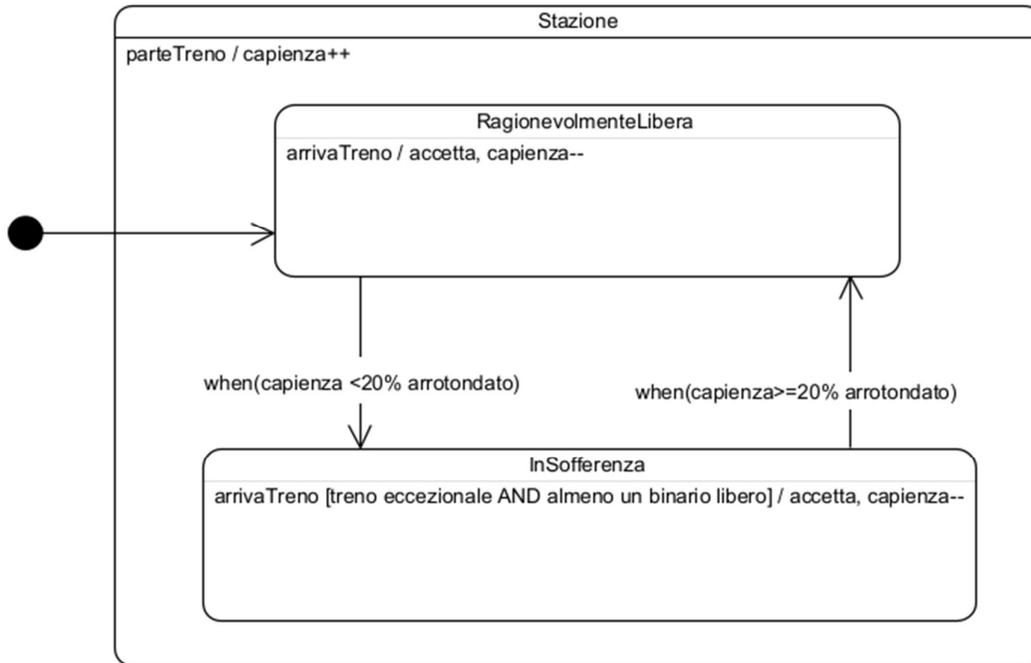
UC 8) **Memorizzazione Storico Indennità:** Il sistema memorizza lo storico delle indennità .

**UC 9) Sottoscrizione** Un Utente si può sottoscrivere al servizio notifica ritardi

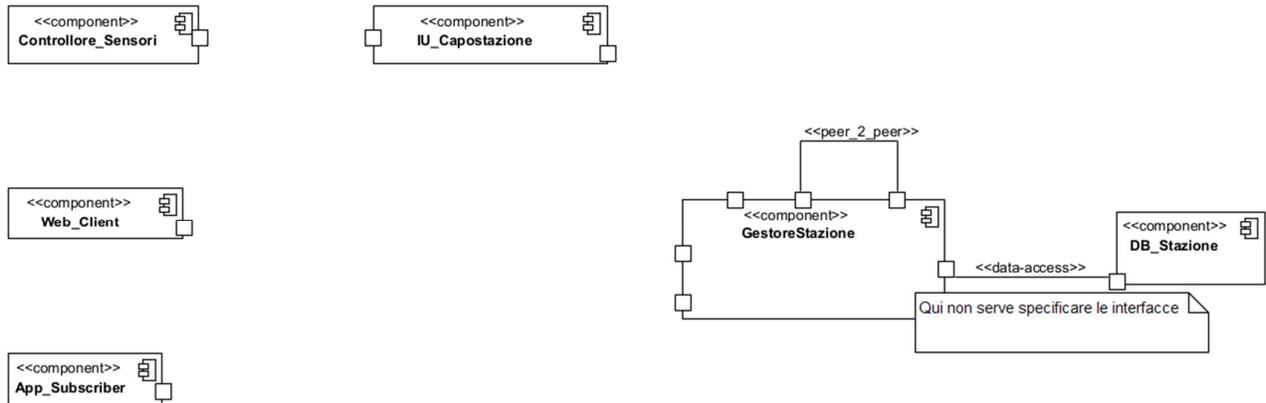
**UC 10) Richiesta indennità ritardi** Un utente può chiedere un'indennità per un ritardo (da discutere)

**Eventuali “in”, includes o extends** UC3 in UC5; UC6 modificato extends (oppure in) UC5; UC7 in UC5; UC8 in UC10; UC2 includes UC3

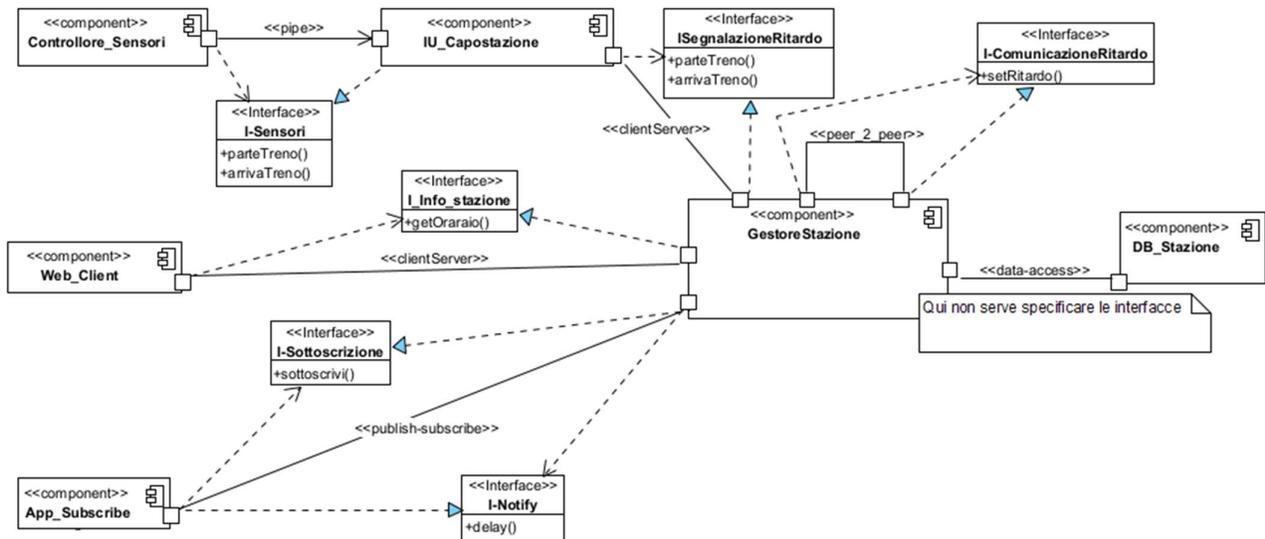
**Domanda 2.a** Dare un diagramma di macchina a stati che rappresenti gli stati in cui si trova una stazione ferroviaria relativamente all'occupazione dei binari per gestire eventuali conflitti dovuti a treni in ritardo (quindi non durante la definizione dell'orario stagionale). Nella valutazione dell'accettazione di un treno in arrivo, è richiesto che almeno il 20% dei binari della stazione (arrotondando all'unità superiore) sia mantenuto libero. Se il treno in arrivo è classificato come *eccezionale* si ignora il vincolo del 20%.



**Domanda 3.a** Completare la vista C&C data (sul sottosistema che realizza i seguenti requisiti--attenzione alle modifiche rispetto al progetto). Indicare tutte le interfacce delle componenti (tranne con DB). *L'arrivo e la partenza di un treno da una stazione sono segnalati opportuni sensori sui binari e confermati dal capostazione.* Se il sistema identifica un ritardo nella partenza di un treno da una stazione (notate che il controllo è fatto localmente in stazione), trasmette l'informazione alle stazioni successive. Una volta che la stazione successiva riceve questi dati, valuta se cambiare binario, seleziona automaticamente i dati associati a quella fermata di quel treno. **Inoltre memorizza ritardo ed eventuale nuovo binario per interrogazioni via web e informa le applicazioni client sottoscritte.**



(Nel diagramma mancano le molteplicità, ma sono ovvie)



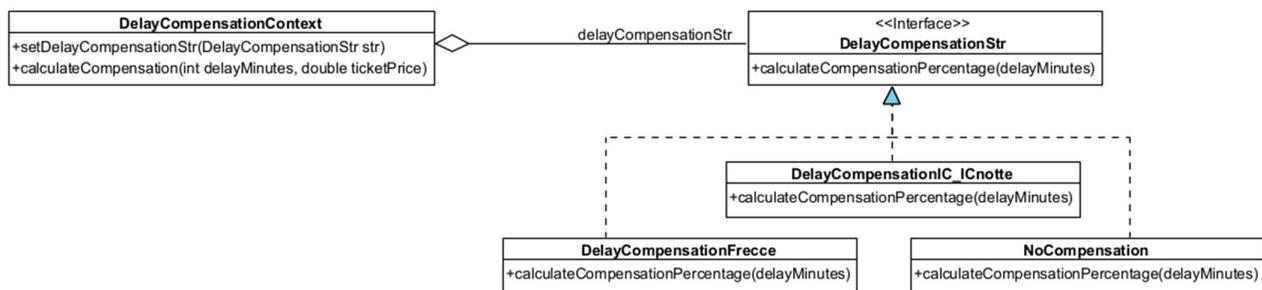
**Domanda 4.a** Seguendo il design pattern opportuno, dare il diagramma delle classi che comprende l'interfaccia DelayCompensationStr e le classi che la implementano.

```

public class DelayCompensationContext {
    private DelayCompensationStr delayCompensationStr;
    public DelayCompensationContext(DelayCompensationStr str) { this.delayCompensationStr = str;}
    public void setDelayCompensationStr(DelayCompensationStr str) {this.delayCompensationStr = str;}
    public double calculateCompensation(int delayMinutes, double ticketPrice) {
        double bonusPercentage = delayCompensationStr.calculateCompensationPercentage(delayMinutes);
        return bonusPercentage * ticketPrice;
    }
}

```

Strategy



**Domanda 5.a** Le indennità per ritardi sono calcolate come segue: 50% del biglietto per ritardi tra 30 e 59 minuti per le Frecce e del 70% per ritardo maggiore di 59 minuti per Frecce, Intercity e Intercity Notte. Usando criteri a scatola chiusa, dare 6 casi di test per il metodo: calculateCompensation(int delayMinutes, double ticketPrice) dell'esercizio 4.

	Input	Output	Ambiente	Criterio usato
Caso 1	(29,10)	0	Frecce	Classe equiv e frontiera
Caso 2	(30,50)	25	Frecce	Classe equiv e frontiera
Caso 3	(59,100)	50	Frecce	Classe equiv e frontiera
Caso 4	(59, 33)	0	IC	Classe equiv e frontiera
Caso 5	(60,100)	70	Frecce	Classe equiv e frontiera
Caso 6	(60,20)	14	IC_notte	Classe equiv e frontiera