SAPIENZA Università di Roma
Facoltà di Ingegneria dell'Informazione, Informatica e Statistica
Corso di Laurea in Ingegneria Informatica ed Automatica
Corso di Progettazione del Software
Esame del **18 settembre 2020**
Tempo a disposizione: 3:30 ore

DG

Requisiti. L'applicazione da progettare riguarda un sistema di prenotazione di auto private.

Sono di interesse gli autisti che hanno un nome ed una matricola (una stringa). Ogni autista possiede diverse auto (almeno una) e le auto sono possedute da un unico autista. Alcuni autisti possono guidare limousine; di questi interessa sapere quale è la lunghezza massima delle limousine che possono guidare. Delle auto interessa la targa (una stringa). Sono poi di interesse i clienti con nome e numero di telefono. I clienti effettuano prenotazioni e ogni prenotazione è effettuata da un solo cliente. Le prenotazioni hanno un codice (una stringa) ed una data, e coinvolgono un cliente (che ha effettuato la prenotazione), un autista ed una macchina (che deve ovviamente essere in possesso dell'autista stesso).

DST

Siamo interessati al comportamento dell'autista. L'autista è inizialmente a riposo. Quando riceve una richiesta da parte del cliente (con la data), se non ha altre prenotazioni per la stessa data, mostra le macchine che ha disponibili e si mette in attesa della scelta del cliente. Se invece ha già una prenotazione per la data indicata risponde che è occupato rimanendo a riposo. Quando l'autista è in attesa ed arriva il messaggio da parte del cliente che aveva richiesto la prenotazione con la macchina scelta, crea una prenotazione e la memorizza nel sistema (vedi diagramma UML) e torna a riposo; se il messaggio per errore arriva da un altro cliente, l'autista risponde segnalando l'errore senza cambiare stato.

DA

Siamo interessati alla seguente attività principale. L'attività prende un insieme di prenotazioni e verifica se ci sono prenotazioni il cui autista non possiede l'auto indicata dalla prenotazione e in caso segnala l'errore terminando. Se la verifica non da errore, allora procede eseguendo in parallelo due sottoattività complesse A e B i cui dettagli non interessano se non per il fatto che entrambe le sottoattività A e B producono un risultato una stringa. Al termine di entrambe A e B, i due risultati prodotti vengono integrati (producendo una nuova stringa) e stampati in output.

Domanda 1. Basandosi sui requisiti riportati sopra, effettuare l'analisi producendo lo schema concettuale in UML per l'applicazione, comprensivo di: diagramma delle classi (inclusi eventuali vincoli non esprimibili in UML); diagramma stati e transizioni per la classe *Autista*; diagramma delle attività principale; specifica del diagramma stati e transizioni, riportando solo gli stati, e le variabili di stato ausiliarie, ma non la specifica delle transizioni; la segnatura delle attività complesse, delle attività atomiche e dei segnali di input/output. Motivare, qualora ce ne fosse bisogno, le scelte di progetto.

Domanda 2. Effettuare il progetto, illustrando i prodotti rilevanti di tale fase e motivando, qualora ce ne fosse bisogno, le scelte di progetto. In particolare definire SOLO le responsabilità sulle associazioni del diagramma delle classi (nella tabella, inserire anche il motivo di ognuna delle responsabilità).

Domanda 3. Effettuare la realizzazione, producendo un programma JAVA e motivando, qualora ce ne fosse bisogno, le scelte di progetto. In particolare realizzare in JAVA SOLO i seguenti aspetti dello schema concettuale:

- La classe `Autista` con la classe `AutistaFired`, le eventuali sottoclassi e le classi JAVA per rappresentare le *associazioni* di cui la classe `Autista` ha responsabilità (basta un riportare un solo "TipoLink" e "ManagerAssociazione").
- L'*attività principale*, l'*attività atomica di verifica*, ma non le sotto-attività A e B e i segnali di input-output.

1)

DIAGRAMMA DELLE CLASSI

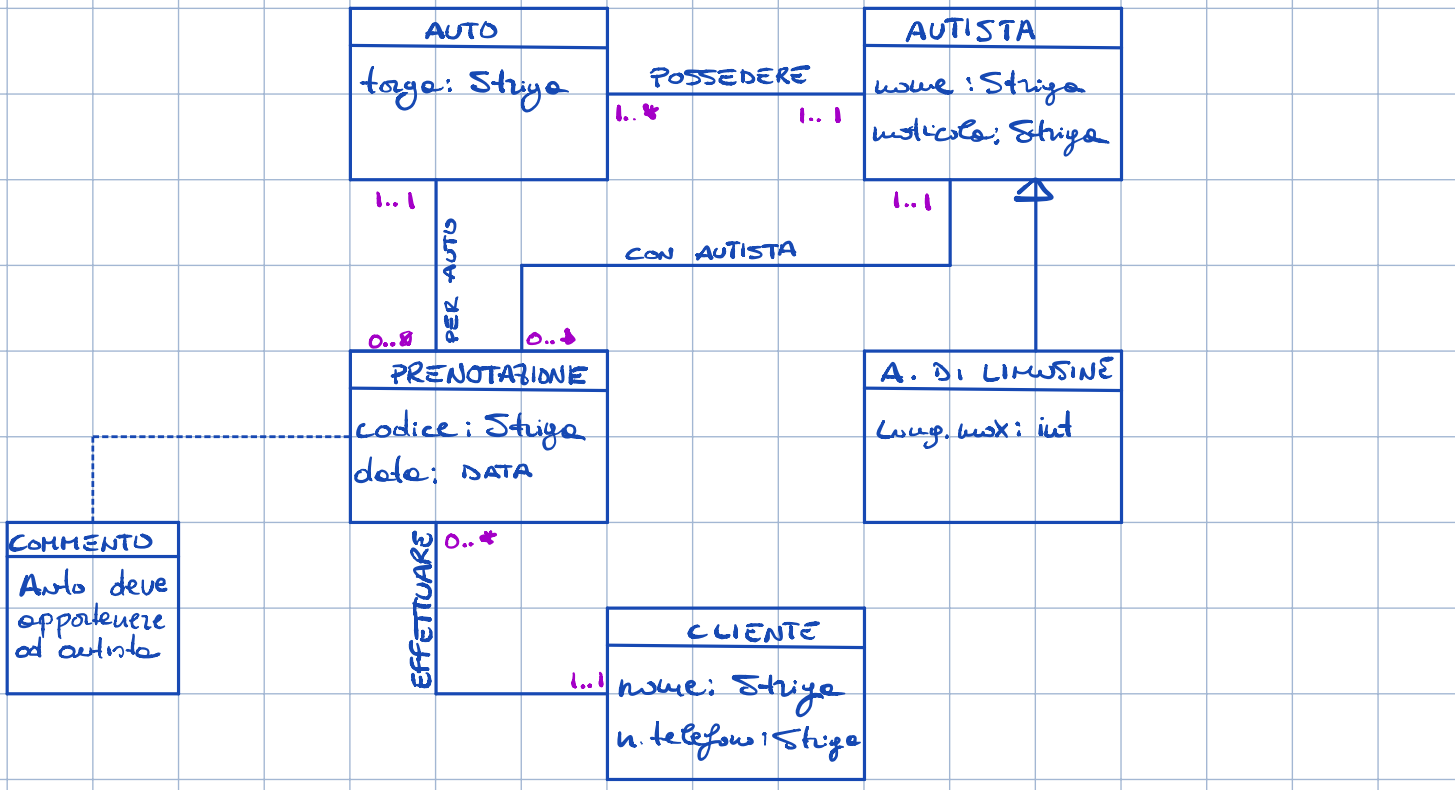
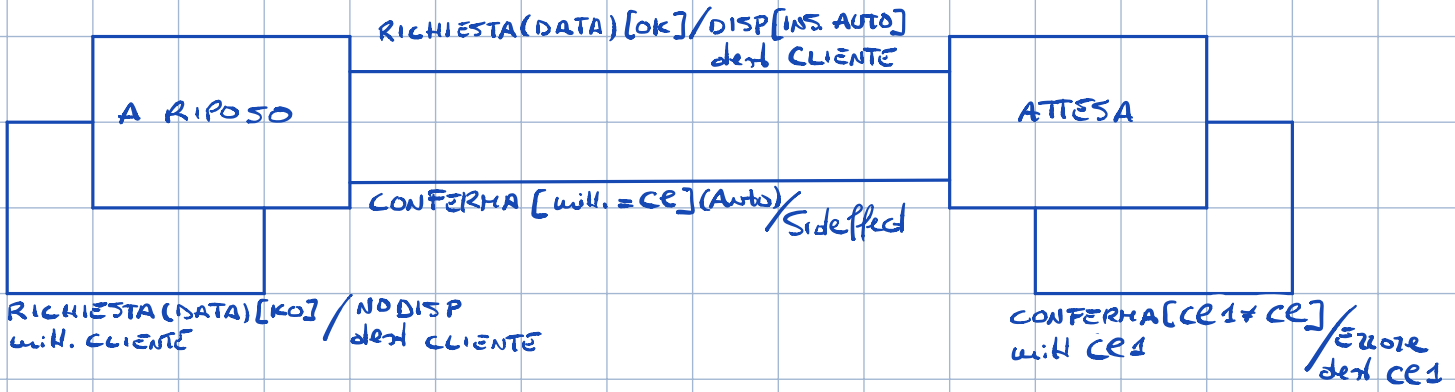


DIAGRAMMA STATI TRANSIZIONI



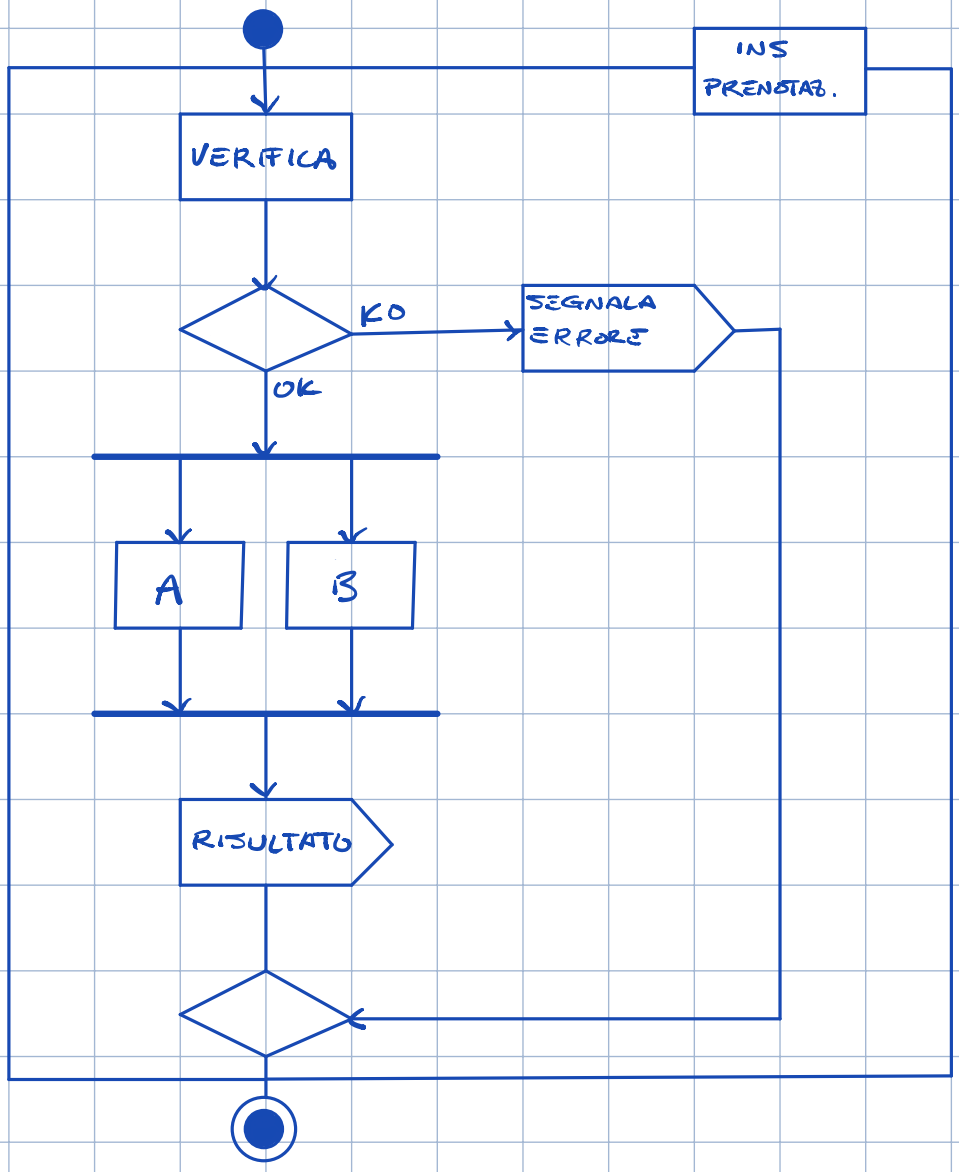
InizioSpecificaStati Antista

Variabili di stato ausiliarie
cl : Cliente
data: Data

Stati: {RIPOSO, ATTESA}
Stato iniziale:
statoCorrente = RIPOSO;
cl = -- ; Data = --;

FineSpecifica

DIAGRAMMA ATTIVITA'



Segnatura attività:

AttivitàPrincipale (ps insiemePrenotazioni):() // att. complessa

Verifica (ps insiemePrenotazioni):(OK: Boolean) // att. atomica

AttivitàA (ps insiemePrenotazioni):(risA: Stringa) // att. complessa

AttivitàB (ps insiemePrenotazioni):(risB: Stringa) // att. complessa

Errore():() // regola I/O

Stampa(risA: Stringa, risB: Stringa):() // regola I/O

2) TABELLA RESPONSABILITÀ:

POSSIEDE	AUTISTA	SI (1,2)
	AUTO	SI (1)
CONAUTISTA	PRENOTAZIONE	SI (1,2)
	AUTISTA	SI (2)
PERAUTO	PRENOTAZIONE	SI (1,2)
	AUTO	NO
EFFETTUA	PRENOTAZIONE	SI (1,2)
	CLIENTE	NO oppure (3)

1: Metodo esplicito

2: Operazioni

3: requisiti

3) REALIZZAZIONE