

Sviluppo delle Applicazioni Software

Laboratorio

a.a 2023/24

Vito Fasulo 986492

Matteo Di Noia 979620

Gestire i turni

Informazioni generali

Nome caso d'uso: Gestire i turni

Portata: Sistema

Livello: Obiettivo Utente

Attore primario: Organizzatore

Parti Interessate: Personale

Pre-condizioni: L'attore deve essere autenticato come Organizzatore

Post-condizioni: L'inserimento/modifica dei turni sono visualizzabili dalle parti interessate

Scenario principale di successo

#	Attore	Sistema
1.	Organizzatore apre Evento esistente	Fornisce dati evento
2.	Apri orario di cucina o di servizio	Fornisce orario di cucina o servizio
3.	Opzionalmente ottiene lista dei turni	Fornisce lista dei turni
	<i>Prosegue a 4 oppure termina</i>	
4.	Crea e inserisce dati turno (data, ora, durata, num. persone richieste, ricorrenza, data chiusura disiscrizione, etc)	Registra nuovo turno
	<i>Ripete passi da 3 finché non soddisfatto</i>	

Estensioni

Eccezione 1a

#	Attore	Sistema
1a.1	Organizzatore apre Evento esistente	Evento non appartiene all'organizzatore.
	<i>Termina il caso d'uso</i>	

Eccezione 4a

#	Attore	Sistema
4a.1	Crea e inserisce dati turno (data, ora, durata, num. persone richieste, ricorrenza, data chiusura disiscrizione, etc)	Il turno creato è sovrapposto, anche parzialmente, ad un altro turno (in stesso luogo).

Eccezione 4b

#	Attore	Sistema
4b.1	Crea e inserisce dati turno (data, ora, durata, num. persone richieste, ricorrenza, data chiusura disiscrizione, etc)	Il turno creato ha dati non sensati (ora negativa, persone negativa, data chiusura minore di data, data prima di giornata attuale).

Estensione 4c

#	Attore	Sistema
4c.1	aggrega/disaggrega i turni	Registra aggregazione

Estensione 4d

#	Attore	Sistema
4d.1	Modifica i dati (data, ora,durata, num. persone richieste, ricorrenza) di uno o più turni	Registra modifica dei dati

Eccezione 4d.1a

#	Attore	Sistema
4d.1a.1	Modifica i dati (data, ora,durata, persone richieste, ricorrenza) di uno o più turni	Il turno modificato è ora sovrapposto, anche parzialmente, ad un altro turno (in stesso luogo).

Eccezione 4d.1b

#	Attore	Sistema
4d.1b.1	Modifica i dati (data, ora,durata, persone richieste, ricorrenza) di uno o più turni	Il turno modificato ha dati non sensati (ora negativa, persone negativa, data chiusura minore di data, data prima di giornata attuale).

Estensione 4e

#	Attore	Sistema
4e.1	sblocca forzatamente o toglie sblocco forzato su gruppo turni	Registra modifica dei dati

Eccezione 4e.1a

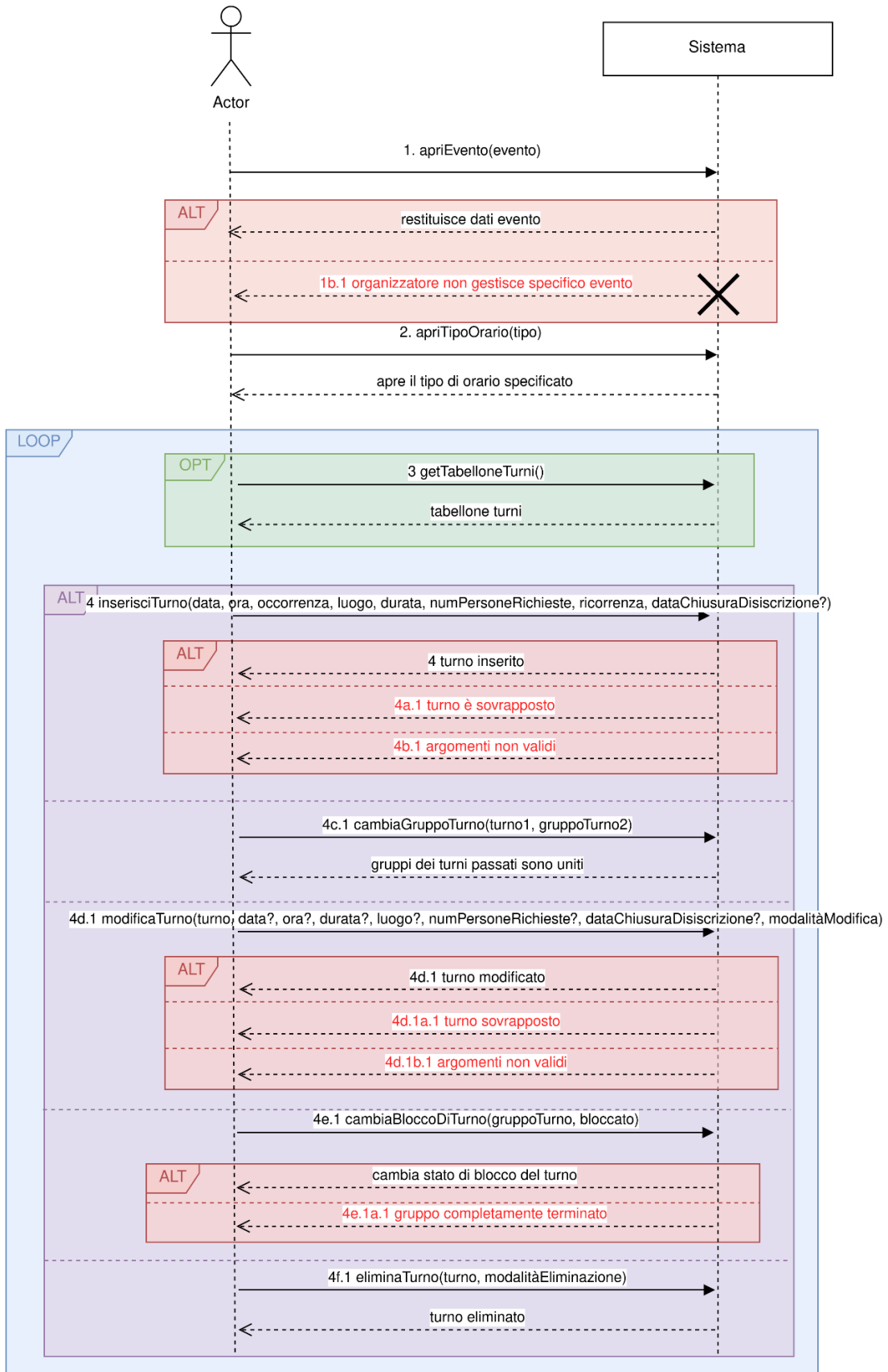
#	Attore	Sistema
4e.1a.1	sblocca forzatamente o toglie sblocco forzato su gruppo turni	Il gruppo di turni è compreso di soli turni scaduti.

Estensione 4f

#	Attore	Sistema
4f.1	elimina turno/i selezionato/i	Registra eliminazione

Modello di Dominio

SSD



Contratti

Pre-condizione generale: L'utente deve essere autenticato come Organizzatore.

1-ApriEvento(evento: Evento)

Pre-condizioni:

- L'evento esiste ed è **organizzato** dall'organizzatore correntemente autenticato.

Post-condizioni: _

2-apriTipoOrario(tipo: "servizio" | "cucina")

Pre-condizioni: _

Post-condizioni: _

3-getTabelloneTurni()

Pre-condizioni: _

Post-condizioni: _

4-inserisciTurno(data: Data, ora: Ora, occorrenza: Occorrenza, luogo: Testo, durata: Durata, numPersoneRichieste: numero, ricorrenza: si/no, dataChiusuraDisiscrizione?: Data)

Pre-condizioni:

- data e ora rappresentano un orario futuro
- dataChiusuraDisiscrizione è posteriore a data
- E' aperto l'Evento evento ed è aperto il Tipo tipo
- Il turno non esiste un altro turno $t2$ tale che $t1 \neq t2$ con $t2.tipo = turno.tipo$ e $t2.luogo = turno.luogo$

Post-condizioni:

- E' stata creata istanza *turno* di Turno
- *turno* è **relativo** a occorrenza
- [se ricorrenza è impostata] E' stata creata istanza TurnoRipetuto *turnoRipetuto*
 - *turno* è **istanza di** *turnoRipetuto*
 - Per ogni istanza *occorrenzaSucc* di Occorrenza che **avviene in** evento, ed è posteriore ad occorrenza, è stato creato turno *turnoSucc*
 - *turnoSucc* è **relativo** ad *occorrenzaSucc*
 - *turnoSucc* è **istanza di** *turnoRipetuto*
 - È stato creato *gruppoTurnoSucc* istanza di GruppoTurno
 - *turnoSucc* è raggruppato in *gruppoTurnoSucc*
- È stato creato *gruppoTurno* istanza di GruppoTurno
- *turno* è raggruppato in *gruppoTurno*

4c.1-cambiaGruppoTurno(turno1: Turno, gruppoTurno2?: GruppoTurno)

Pre-condizioni:

- turno1 è raggruppato in *gruppoTurno1* istanza di GruppoTurno

Post-condizioni:

- [Se gruppoTurno2 è definito]
 - Per ogni *cuoco* istanza di Cuoco disponibile per gruppoTurno2 o *gruppoTurno1*, cuoco **non è più disponibile** per gruppoTurno2 o *gruppoTurno1*
 - Per ogni *t* istanza di Turno **raggruppato in *gruppoTurno1***, allora *t* **non è più raggruppato a *gruppoTurno1* ma è raggruppato a gruppoTurno2**
 - L'istanza *gruppoTurno1* è stato eliminata
- [Se gruppoTurno2 non è definito e *gruppoTurno1* **raggruppa** più di un Turno] E' stato istanziato *nuovoGruppo* di tipo GruppoTurno
 - turno1 **non è più raggruppato** in *gruppoTurno1* ma è **raggruppato** in *nuovoGruppo*

4-modificaTurno(turno: Turno, data?: Data, ora?: Ora, luogo?: Testo, durata?: Durata, numPersoneRichieste?: numero, dataChiusuraDisiscrizione?: Data, modalitàModifica: “singolo” | “successivi” | “tutti”)

Pre-condizioni:

- turno esiste
- [Se esiste data] *diffData* è la differenza tra la data e turno.data
- [Se non esiste data] *diffData* = 0
- [Se esiste dataChiusuraDisiscrizione] *diffDataChius* è la differenza tra la dataChiusuraDisiscrizione e turno.dataChiusuraDisiscrizione
- [Se non esiste data] *diffDataChius* = 0

Post-condizioni:

- [Se modalitàModifica è singolo o *turno* **non è istanza** di un TurnoRipetuto] *turniInteressati* contiene solo turno
- [Se modalitàModifica non è singolo] dato *turnoRipetuto* di cui *turno* è **istanza**, *turniInteressati* contiene tutti i turni che **sono istanza** di *turnoRipetuto* successivi o uguali a *turno* se modalitàModifica="successivi" o tutti altrimenti
- Per ogni *turno* contenuto di *turniInteressati*
 - turno.data = turno.data + *diffData*
 - turno.dataChiusuraDisiscrizione = turno.dataChiusuraDisiscrizione + *diffDataChius*
 - [Se durata esiste] turno.durata = durata
 - [Se ora esiste] turno.ora = ora
 - [Se numPersoneRichiesta esiste] turno.numPersoneRichieste = numPersoneRichieste

4e.1-cambiaStatoSbloccoForzato(gruppoTurno: GruppoTurno, sbloccato: sì/no)

Pre-condizioni:

- Esiste Turno *turno* **raggruppato in gruppoTurno**, tale per cui *turno.data* e *turno.ora* è posteriore all'attuale data e ora

Post-condizioni: _

- GruppoTurno.forzatamenteSbloccato = forzatamenteSbloccato

4f.1-eliminaTurno(turno: Turno, modalitàEliminazione: “singolo” | “successivi” | “tutti”)

Pre-condizioni: _

Post-condizioni:

- [Se non esiste TurnoRipetuto di cui è **istanza** turno oppure modalitàModifica="singolo"] *turniInteressati* contiene solo turno
- [Se il turno è **istanza di** TurnoRipetuto *turnoRipetuto* e modalitàModifica!="singolo"] *turniInteressati* contiene ogni Turno **istanza** di *turnoRipetuto*, se modalitàModifica="successivi" allora vengono considerati solo quelli successivi a turno; tutti altrimenti
- Per ogni *turnoAltro* in *turniInteressati*
 - Data *turnoAltro* che è **relativo all'Occorrenza** *occorrenza*, *turnoAltro* **non è più relativo a occorrenza**
 - Per ogni CompitiDiCucina compito il quale è **svolto** in *turnoAltro*, *turnoAltro* **non svolge più compito**
 - Dato *GruppoTurno* gruppo in cui *turno* è **raggruppato**, *turnoAltro* **non è più raggruppato** in *gruppo*
 - [Se non c'è Turno che è **raggruppato in gruppo**] per ogni Cuoco *cuoco* **disponibile per gruppo**, *cuoco* **non è più disponibile** per *gruppo* ed è stata eliminata l'istanza *gruppo*.
 - [Se non c'è Turno che è **istanza di** *turnoRipetuto*] è stata eliminata l'istanza *turnoRipetuto*
 - E' stata eliminata l'istanza *turnoAltro*

DCD

DSD

