
ANNO ACCADEMICO 2023/2024

Sviluppo Applicazioni Software

Gestire i Turni

Luca Barra, Jasmine Bizzo



UNIVERSITÀ
DI TORINO

DIPARTIMENTO DI INFORMATICA

CAPITOLO 1	GESTIRE I TURNI	PAGINA 2
-------------------	------------------------	-----------------

CAPITOLO 2	MODELLO DI DOMINIO	PAGINA 7
-------------------	---------------------------	-----------------

CAPITOLO 3	SSD - SYSTEM SEQUENCE DIAGRAM	PAGINA 9
3.1	Scenario principale	9
3.2	Eccezioni ed estensioni - Passo 1	10
3.3	Eccezioni ed estensioni - Passo 5	11
3.4	Eccezioni ed estensioni - Passo 6	12

CAPITOLO 4	CONTRATTI	PAGINA 14
4.1	Passo 1	14
4.2	Passo 2	17
4.3	Passo 3	17
4.4	Passo 4	17
4.5	Passo 5	17
4.6	Passo 6	18

CAPITOLO 5	DCD - DESIGN CLASS DIAGRAM	PAGINA 21
-------------------	-----------------------------------	------------------

CAPITOLO 6	DSD - DESIGN SEQUENCE DIAGRAM	PAGINA 23
-------------------	--------------------------------------	------------------

1

Gestire i turni

Informazioni generali

Nome Caso d'Uso: Gestire i turni

Portata: Sistema

Livello: Obiettivo utente

Attore primario: Organizzatore

Parti interessate: Chef, Cuochi

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

Garanzie di successo o post-condizioni: I turni di cucina e di servizio sono stati impostati

Scenario principale di successo

#	Attore	Sistema
1	Aggiunge un turno di cucina e, opzionalmente, segna un luogo	Registra il turno della cucina.
	<i>Ripete il passo 1 finché non è soddisfatto</i>	
2	Opzionalmente, controlla gli eventi.	Visualizza gli eventi.
	<i>Se non vuole lavorare su nessun evento termina il Caso d'Uso</i>	
3	Sceglie un evento su cui lavorare.	Visualizza l'evento selezionato.
4	Aggiunge un turno di servizio, impostandone un orario.	Registra l'orario del turno di servizio.
5	Opzionalmente, aggiunge una scadenza oltre la quale non si può togliere la disponibilità.	Registra la scadenza.
6	Opzionalmente, cambia l'orario del turno di servizio.	Registra le modifiche.
	<i>Se vuole lavorare su altri servizi torna al punto 4</i>	

	<i>Se vuole lavorare su altri eventi torna al passo 3, altrimenti termina il Caso d'Uso</i>	
--	---	--

Estensione 1a

#	Attore	Sistema
1a.1	Modifica un turno di cucina.	Registra la modifica del turno di cucina.

Eccezione 1a

#	Attore	Sistema
1a.1	Aggiunge un turno di cucina e, opzionalmente, segna un luogo.	Non si hanno i permessi necessari.
	<i>Termina il Caso d'Uso</i>	

Eccezione 1b

#	Attore	Sistema
1b.1	Aggiunge un turno di cucina e, opzionalmente, segna un luogo.	Esiste già un turno in quell'orario.
	<i>Termina il Caso d'Uso</i>	

Eccezione 1a.1a

#	Attore	Sistema
1a.1a.1	Modifica un turno di cucina.	Non si hanno i permessi necessari.
	<i>Termina il Caso d'Uso</i>	

Estensione 1b

#	Attore	Sistema
---	--------	---------

1b.1	Cancella un turno di cucina.	Elimina il turno di cucina specificato.
------	------------------------------	---

Eccezione 1b.1a

#	Attore	Sistema
1b.1a.1	Cancella un turno di cucina.	Non si hanno i permessi necessari.
	<i>Termina il Caso d'Uso</i>	

Estensione 1c

#	Attore	Sistema
1c.1	Modifica tutti i turni di cucina in un determinato giorno, settimana.	Registra le modifiche di tutti i turni di cucina aventi determinate caratteristiche.

Eccezione 1c.1a

#	Attore	Sistema
1c.1a.1	Modifica tutti i turni di cucina in un determinato giorno, settimana.	Non si hanno i permessi necessari.
	<i>Termina il Caso d'Uso</i>	

Estensione 1d

#	Attore	Sistema
1d.1	Elimina tutti i turni di cucina in un determinato giorno, settimana.	Registra l'eliminazione di tutti i turni di cucina aventi determinate caratteristiche.

Eccezione 1d.1a

#	Attore	Sistema
---	--------	---------

1d.1a.1	Elimina tutti i turni di cucina in un determinato giorno, settimana.	Non si hanno i permessi necessari.
	<i>Termina il Caso d'Uso</i>	

Estensione 5a

#	Attore	Sistema
5a.1	Rimuove la scadenza.	Toglie la scadenza oltre la quale non si può più rimuovere la disponibilità.

Eccezione 5a.1a

#	Attore	Sistema
5a.1a.1	Rimuove la scadenza.	Il turno non ha una scadenza impostata.
	<i>Torna al passo 5</i>	

Estensione 5b

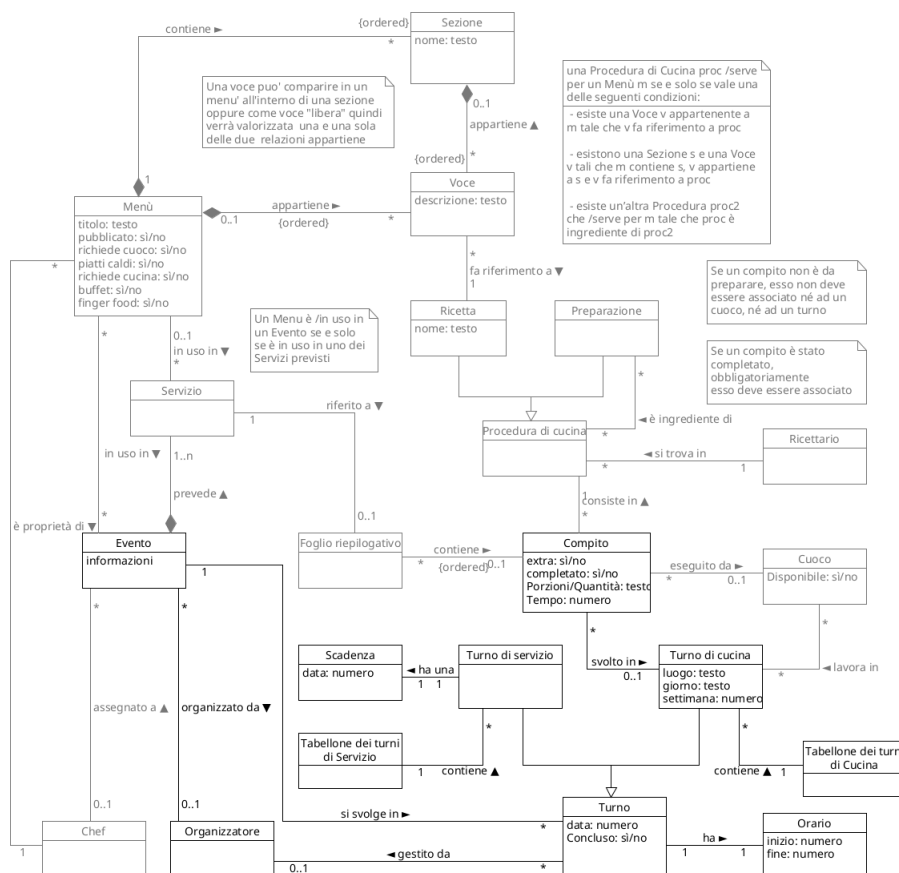
#	Attore	Sistema
5b.1	Modifica la scadenza.	Registra la nuova scadenza.

Estensione 6a

#	Attore	Sistema
6a.1	Cancella un turno di servizio.	Registra la cancellazione del turno di servizio..

2

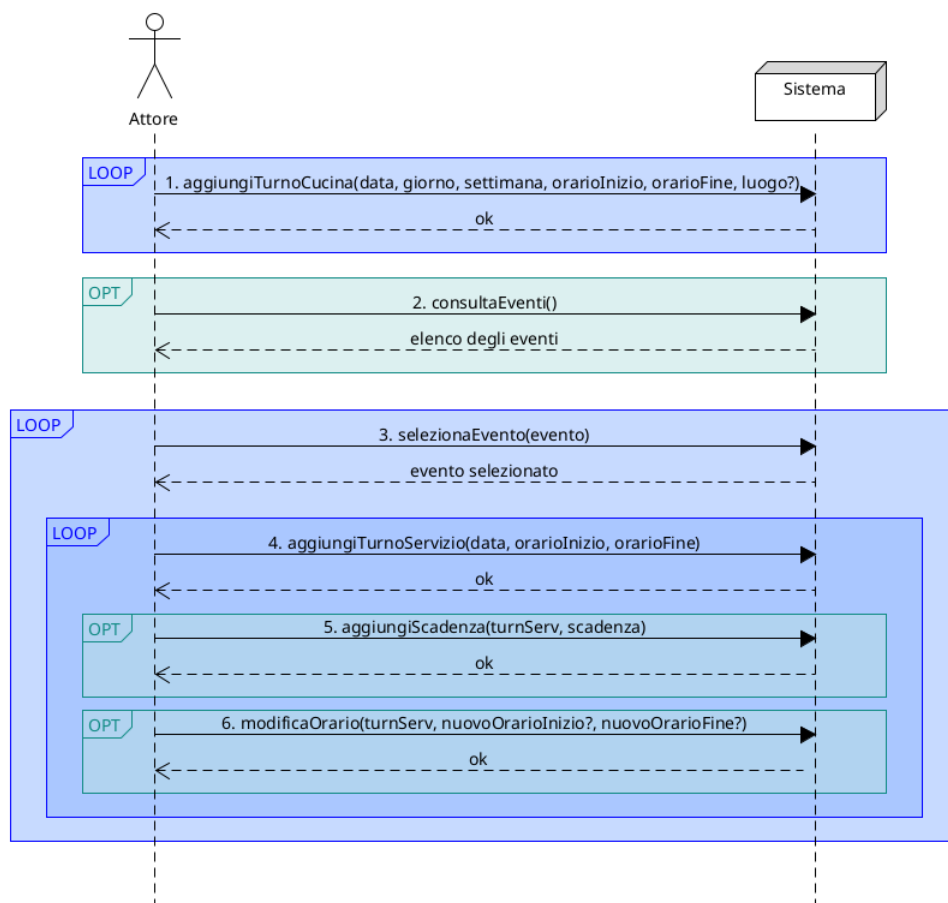
Modello di Dominio



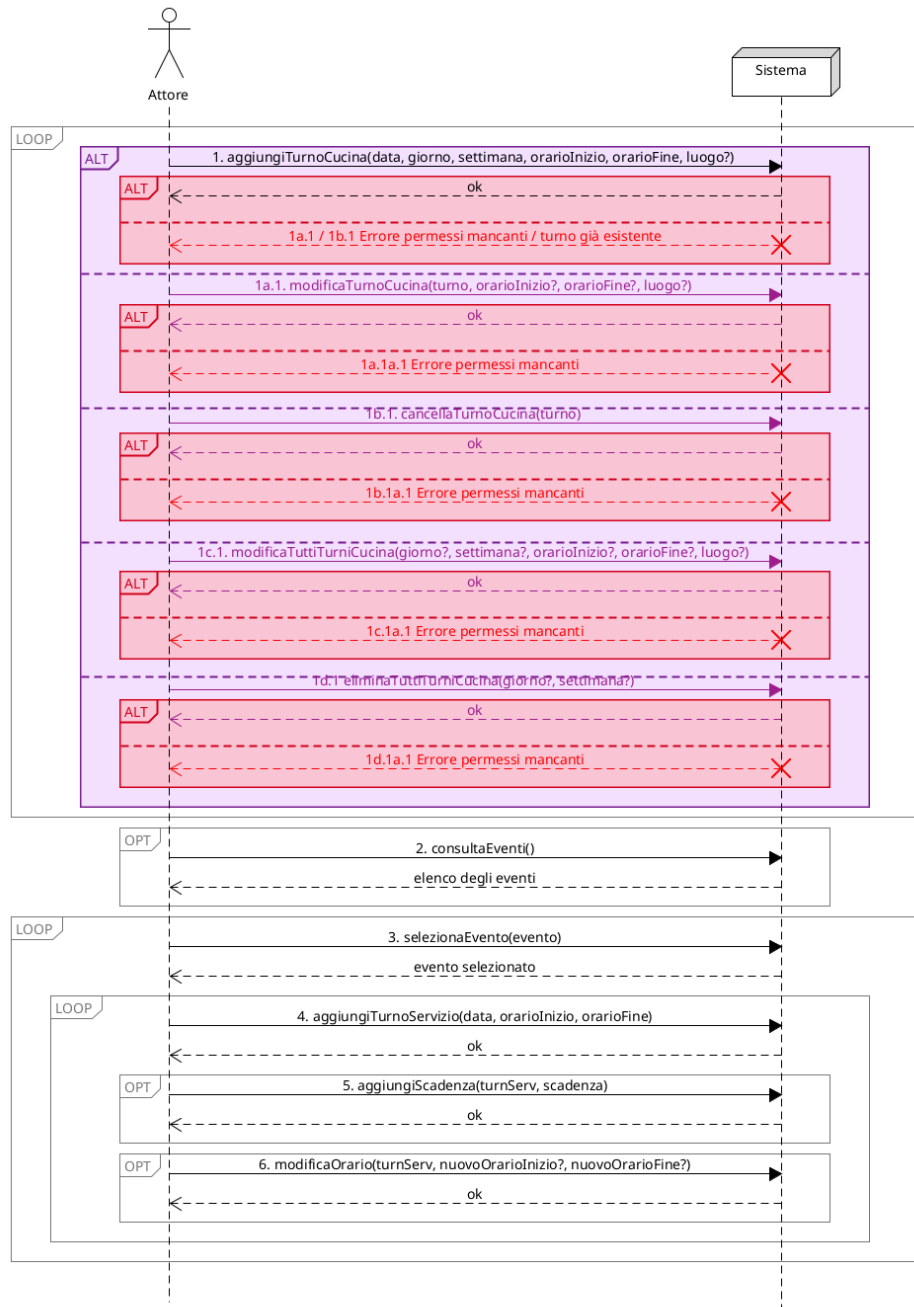
3

SSD - System Sequence Diagram

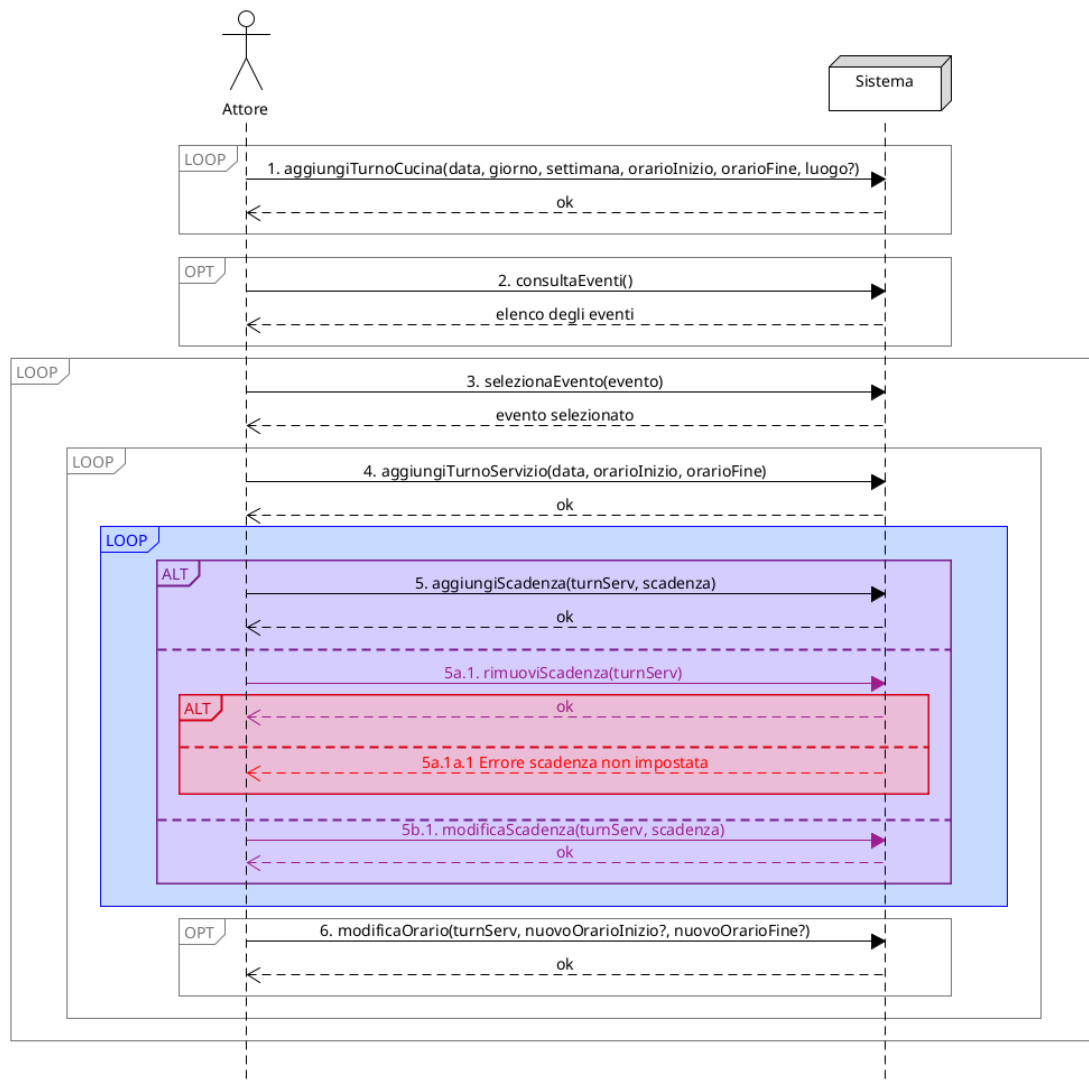
3.1 Scenario principale



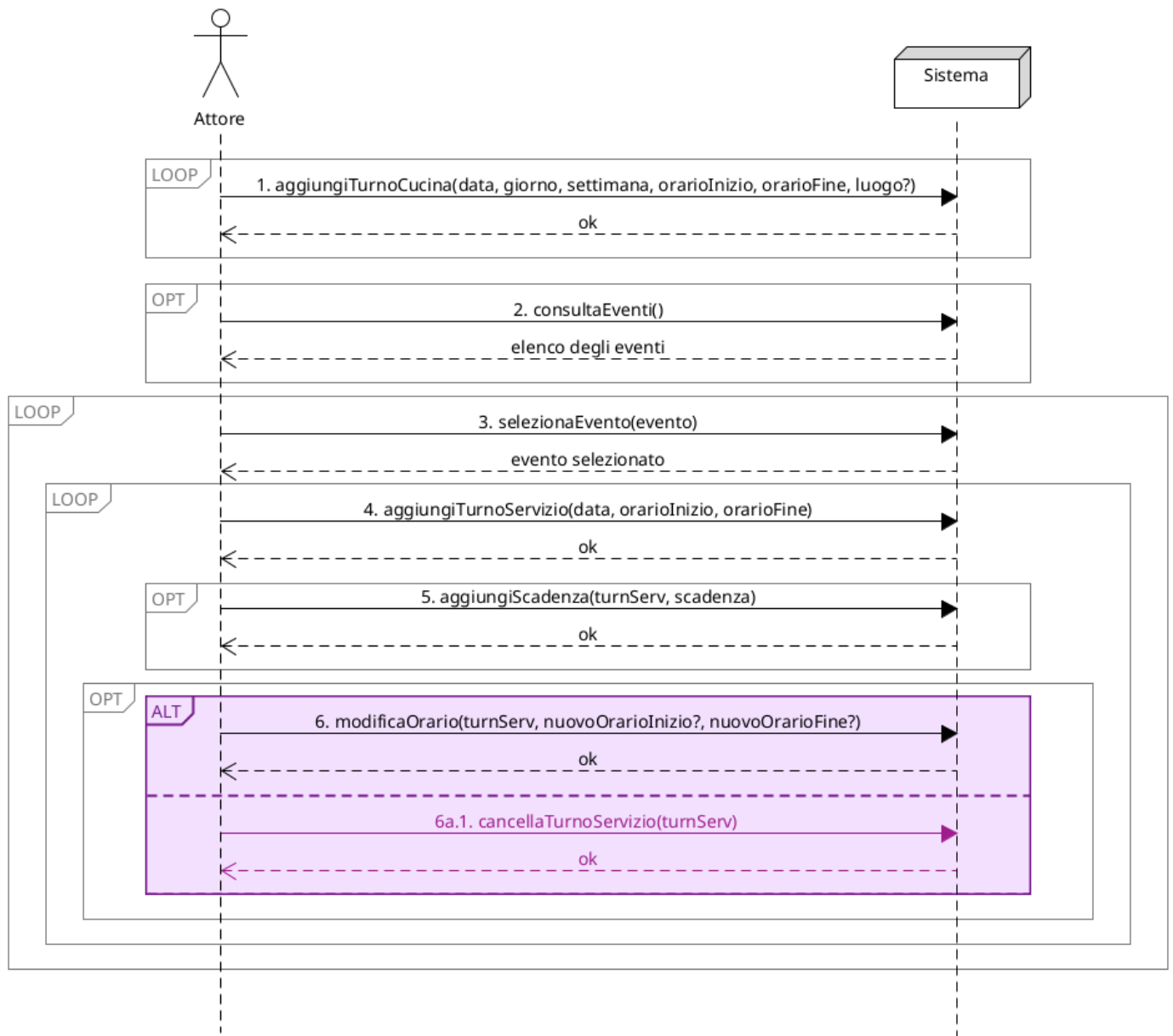
3.2 Eccezioni ed estensioni - Passo 1



3.3 Eccezioni ed estensioni - Passo 5



3.4 Eccezioni ed estensioni - Passo 6



4

Contratti

Pre-condizione generale: L'attore è identificato con un'istanza *org* di Organizzatore.

4.1 Passo 1

1. **aggiungiTurnoCucina**(turno: TurnoCucina, data: numero, giorno: testo, settimana: numero, orarioInizio: numero, orarioFine: numero, luogo?: Testo):

Pre-condizioni:

Post-condizioni: Se turno è gestito da *org* e non esiste un altro turno con data == turno.data, orarioInizio >= turno.orario.orarioInizio e orarioFine <= turno.orario.orarioFine

- è stata creata un'istanza *tc* di TurnoCucina;
- è stata creata un'istanza *o* di Orario;
- è stata creata un'associazione **ha** turnoCucina e *o*;
- è stata creata un'associazione **contiene** turnoCucina e *TabelloneTurniCucina*;
- turno.data = data;
- turno.giorno = giorno;
- turno.settimana = settimana;
- [se luogo è specificato] turno.luogo = luogo;
- turno.concluso = no;
- o.inizio = orarioInizio;
- o.fine = orarioFine.

1a.1 modificaTurnoCucina(turno: TurnoCucina, orarioInizio?: numero, orarioFine?: numero, luogo?: Testo):

Pre-condizioni:

- TabelloneTurniCucina *ttc* **contiene** turno;
- [se orarioInizio o orarioFine è specificato] non esiste un *tc* tale che *tc*.orario.inizio > orario e *tc*.orario.fine < orario.

Post-condizioni: Se turno è gestito da *org*:

- [se orarioInizio è specificato] turno.orario.inizio = orarioInizio;
- [se orarioFine è specificato] turno.orario.fine = orarioFine;
- [se luogo è specificato] turno.luogo = luogo.

1b.1 cancellaTurnoCucina(turno: TurnoCucina):

Pre-condizioni:

- TabelloneTurniCucina *ttc* **contiene** turno.

Post-condizioni: Se turno è gestito da *org*

- ⟨ per ogni Compito *c* **presente in** turno ⟩:
 - l'associazione **svolto in** tra *c* e turno è stata eliminata.
- l'associazione **contiene** tra *tc* e *ttc* è stata eliminata;
- l'associazione **gestito da** tra *tc* e *org* è stata eliminata;
- l'associazione **si svolge in** tra *tc* e *Evento* è stata eliminata;
- l'associazione **ha** tra *tc* e *o* è stata eliminata;
- l'istanza *o* di Orario cessa di esistere;
- l'istanza *tc* di TurnoCucina cessa di esistere.

1c.1 modificaTuttiTurniCucina(giorno?: Testo, settimana?: numero, orarioInizio?: numero, orarioFine?: numero, luogo?: Testo):

Pre-condizioni:

Post-condizioni: Se turno è gestito da *org*

- [se giorno e settimana sono specificati] ⟨per ogni TurnoCucina *tc* presente in *ttc* con *tc*.data.giorno == giorno e *tc*.data.settimana == settimana ⟩:
 - [se orarioInizio è specificato] turno.orario.inizio = orarioInizio;
 - [se orarioFine è specificato] turno.orario.fine = orarioFine;
 - [se luogo è specificato] *tc*.luogo = luogo.
- [se giorno è specificato] ⟨per ogni TurnoCucina *tc* presente in *ttc* con *tc*.data.giorno == giorno ⟩:
 - [se orarioInizio è specificato] turno.orario.inizio = orarioInizio;

- [se orarioFine è specificato] turno.orario.fine = orarioFine;
- [se luogo è specificato] tc.luogo = luogo.
- [se settimana è specificato] ⟨per ogni TurnoCucina *tc* presente in *ttc* con *tc.data.settimana* == settimana⟩:
 - [se orarioInizio è specificato] turno.orario.inizio = orarioInizio;
 - [se orarioFine è specificato] turno.orario.fine = orarioFine;
 - [se luogo è specificato] tc.luogo = luogo.

1d.1 eliminaTuttiTurniCucina(giorno: Testo, settimana: Testo):

Pre-condizioni:

Post-condizioni:

- [se giorno e settimana sono specificati] ⟨per ogni TurnoCucina *tc* presente in *ttc* con *tc.data.giorno* == giorno e *tc.data.settimana* == settimana⟩:
 - ⟨ per ogni Compito *c* **presente in** *t*⟩:
 - * l'associazione tra *c* e *tc* è stata eliminata.
 - l'associazione **contiene** tra *tc* e *ttc* è stata eliminata;
 - l'associazione **gestito da** tra *tc* e *org* è stata eliminata;
 - l'associazione **si svolge in** tra *tc* e *Evento* è stata eliminata;
 - l'associazione **ha** tra *tc* e *o* è stata eliminata;
 - l'istanza *o* di Orario cessa di esistere;
 - l'istanza *tc* di TurnoCucina cessa di esistere.
- [se giorno è specificato] ⟨per ogni TurnoCucina *tc* presente in *ttc* con *tc.data.giorno* == giorno⟩:
 - ⟨ per ogni Compito *c* **presente in** *t*⟩:
 - * l'associazione tra *c* e *tc* è stata eliminata.
 - l'associazione **contiene** tra *tc* e *ttc* è stata eliminata;
 - l'associazione **gestito da** tra *tc* e *org* è stata eliminata;
 - l'associazione **si svolge in** tra *tc* e *Evento* è stata eliminata;
 - l'associazione **ha** tra *tc* e *o* è stata eliminata;
 - l'istanza *o* di Orario cessa di esistere;
 - l'istanza *tc* di TurnoCucina cessa di esistere.
- [se settimana è specificata] ⟨per ogni TurnoCucina *tc* presente in *ttc* con *tc.data.settimana* == settimana⟩:
 - ⟨ per ogni Compito *c* **presente in** *t*⟩:
 - * l'associazione tra *c* e *tc* è stata eliminata.
 - l'associazione **contiene** tra *tc* e *ttc* è stata eliminata;
 - l'associazione **gestito da** tra *tc* e *org* è stata eliminata;
 - l'associazione **si svolge in** tra *tc* e *Evento* è stata eliminata;
 - l'associazione **ha** tra *tc* e *o* è stata eliminata;
 - l'istanza *o* di Orario cessa di esistere;
 - l'istanza *tc* di TurnoCucina cessa di esistere.

4.2 Passo 2

2. consultaEventi():

Pre-condizioni:

Post-condizioni:

Note:-

consultaEventi() è un'interrogazione di sistema quindi non ha pre e post condizioni.

4.3 Passo 3

3. selezionaEvento(evento: Evento):

Pre-condizioni:

- evento è organizzato da *org*.

Post-condizioni:

4.4 Passo 4

4. aggiungiTurnoServizio(turnoServizio: TurnoServizio, data: numero, orarioInizio: numero, orarioFine: numero):

Pre-condizioni:

- è in corso la modifica di *tts* TabelloneTurniServizio;
- è in corso la definizione dei turni di servizio in *e* Evento e servizio è previsto da *e*.

Post-condizioni:

- è stata creata un'istanza *ts* di TurnoServizio;
- è stata creata un'istanza *o* di Orario;
- è stata creata un'associazione **ha** turnoServizio e *o*;
- è stata creata un'associazione **contiene** tra turnoServizio e TabelloneTurniServizio;
- turnoServizio.data = data;
- o.inizio = orarioInizio;
- o.fine = orarioFine.
- turnoServizio.concluso = no.

4.5 Passo 5

5. aggiungiScadenza(turnoServizio: TurnoServizio, scadenza: Testo):

Pre-condizioni:

- *tts* **contiene** turnoServizio;
- turnoServizio è gestito da *org*;
- turnoServizio.concluso = no.

Post-condizioni:

- è stata creata un'istanza s di Scadenza;
- $s.data = scadenza$;
- è stata creata l'associazione **ha una** tra turnoServizio e s .

5a.1 rimuoviScadenza():**Pre-condizioni:**

- è in corso la modifica di ts TurnoServizio;
- tts **contiene** turnoServizio;
- turnoServizio è **gestito da** org ;
- turnoServizio.concluso = no.

Post-condizioni: Se esiste l'associazione **ha una** tra turnoServizio e un'istanza s di Scadenza:

- l'associazione **ha una** tra turnoServizio s è stata eliminata;
- l'istanza s di Scadenza cessa di esistere.

5b.1 modificaScadenza(turnoServizio: TurnoServizio, scadenza: Testo):**Pre-condizioni:**

- è in corso la modifica di tts TabelloneTurniServizio;
- tts **contiene** turnoServizio;
- turnoServizio è **gestito da** org ;
- turnoServizio.concluso = no.

Post-condizioni:

- $s.data$ = scadenza.

4.6 Passo 6**6. modificaOrario(turnoServizio: TurnoServizio, nuovoOrarioInizio?: numero, nuovoOrarioFine?: numero,):****Pre-condizioni:**

- è in corso la modifica di tts TabelloneTurniServizio;
- tts **contiene** turnoServizio;
- turnoServizio è **gestito da** org ;
- turnoServizio.concluso = no.

Post-condizioni:

- [se orarioInizio è specificato] turnoServizio.Orario.inizio = orarioInizio;
- [se orarioFine è specificato] turnoServizio.Orario.fine = orarioFine;

6a.1 cancellaTurnoServizio(turnoServizio: TurnoServizio):

Pre-condizioni:

- è in corso la modifica di *tts* TabelloneTurniServizio;
- *tts* **contiene** turnoServizio;
- turnoServizio è **gestito da** *org*;
- turnoServizio.concluso = no.

Post-condizioni:

- [se esiste l'associazione **ha una** tra turnoServizio e un'istanza *s* di Scadenza]:
 - questa è stata eliminata;
 - l'istanza *s* di Scadenza cessa di esistere.
- l'associazione **contiene** tra *ts* e *tts* è stata eliminata;
- l'associazione **gestito da** tra *ts* e *org* è stata eliminata;
- l'associazione **si svolge in** tra *ts* e *Evento* è stata eliminata;
- l'associazione **ha** tra *ts* e *o* è stata eliminata;
- l'istanza *o* di Orario cessa di esistere;
- l'istanza *tc* di TurnoServizio cessa di esistere.

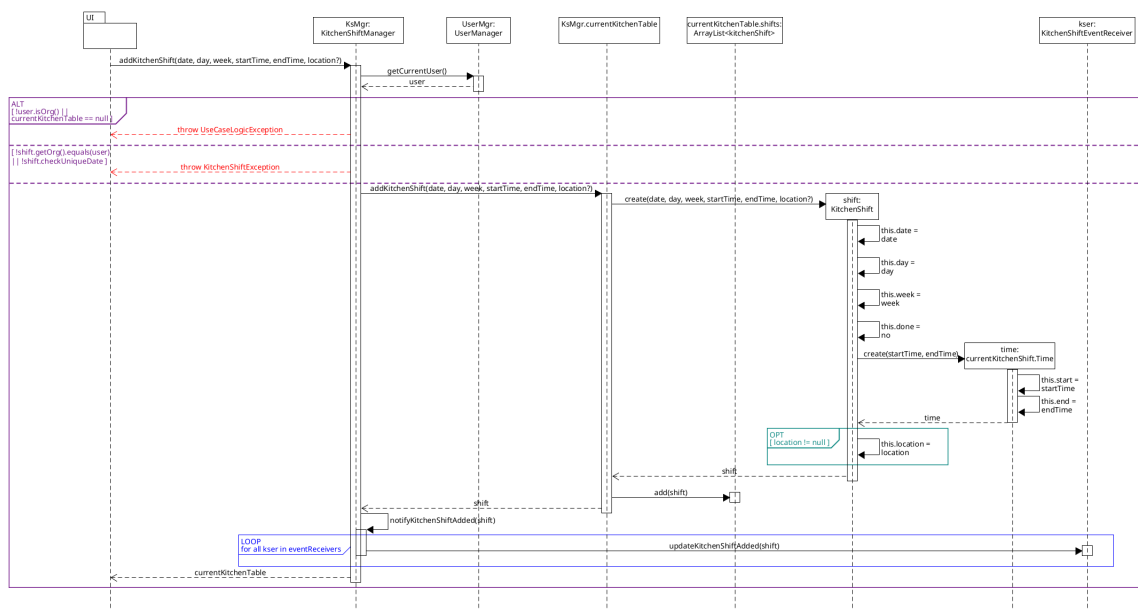
DCD - Design Class Diagram

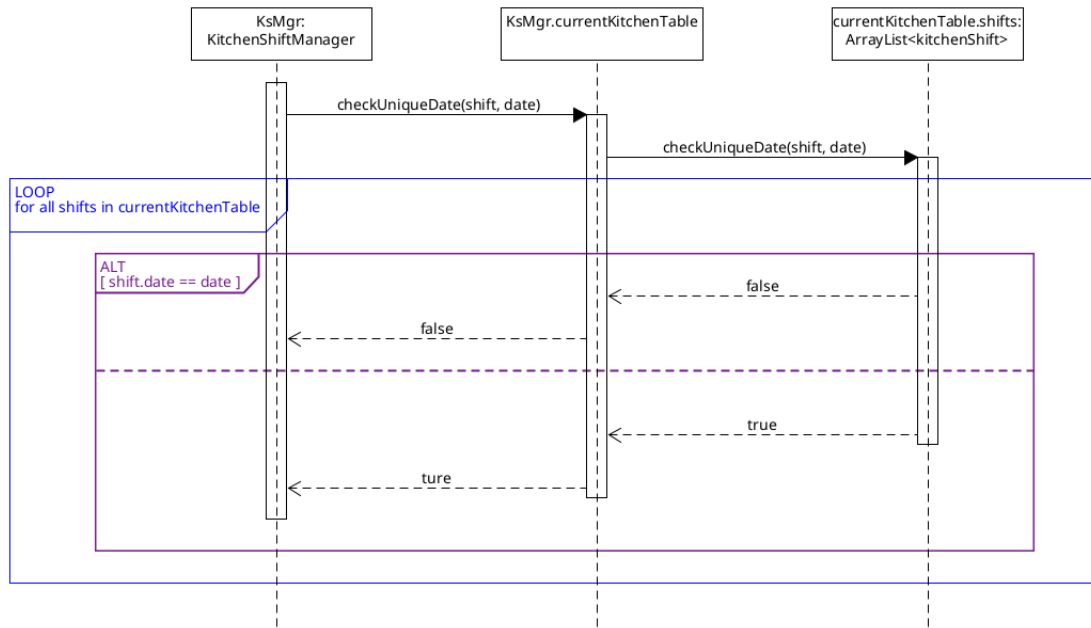


6

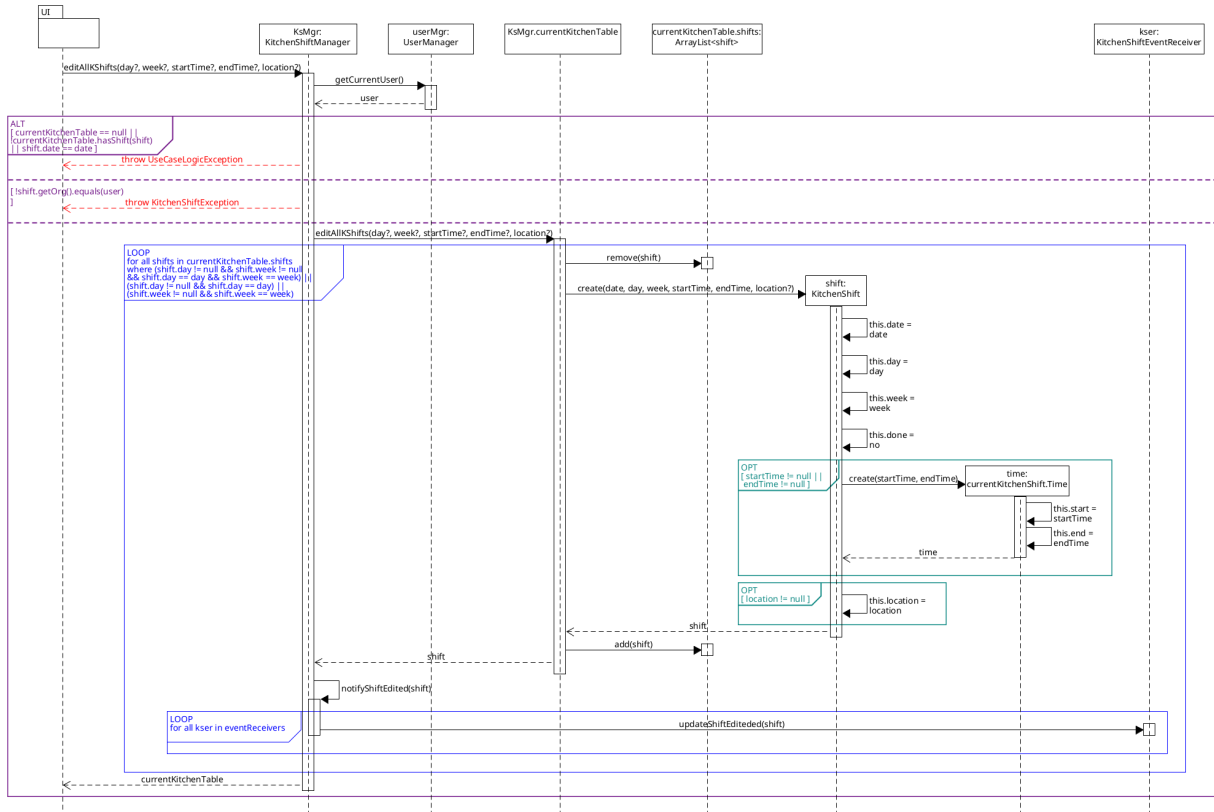
DSD - Design Sequence Diagram

1. addKitchenShift(shift, date, day, week, startTime, endTime, location?)

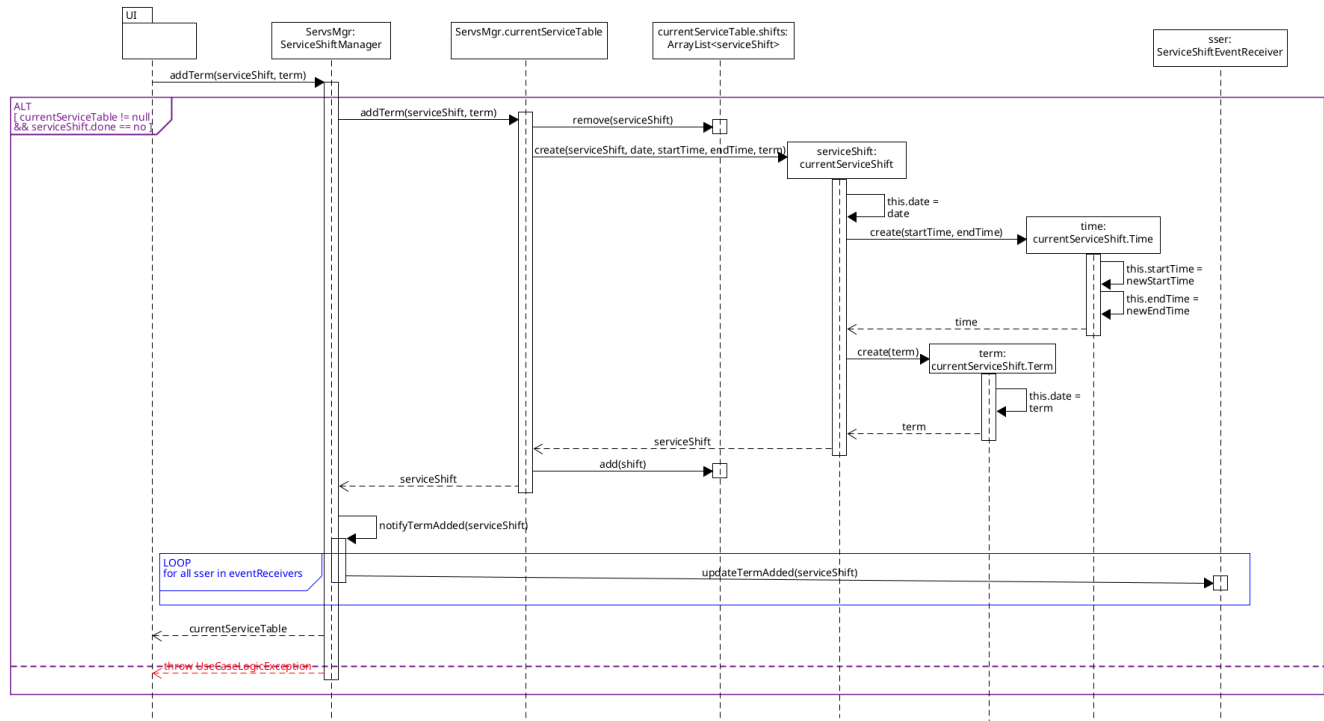




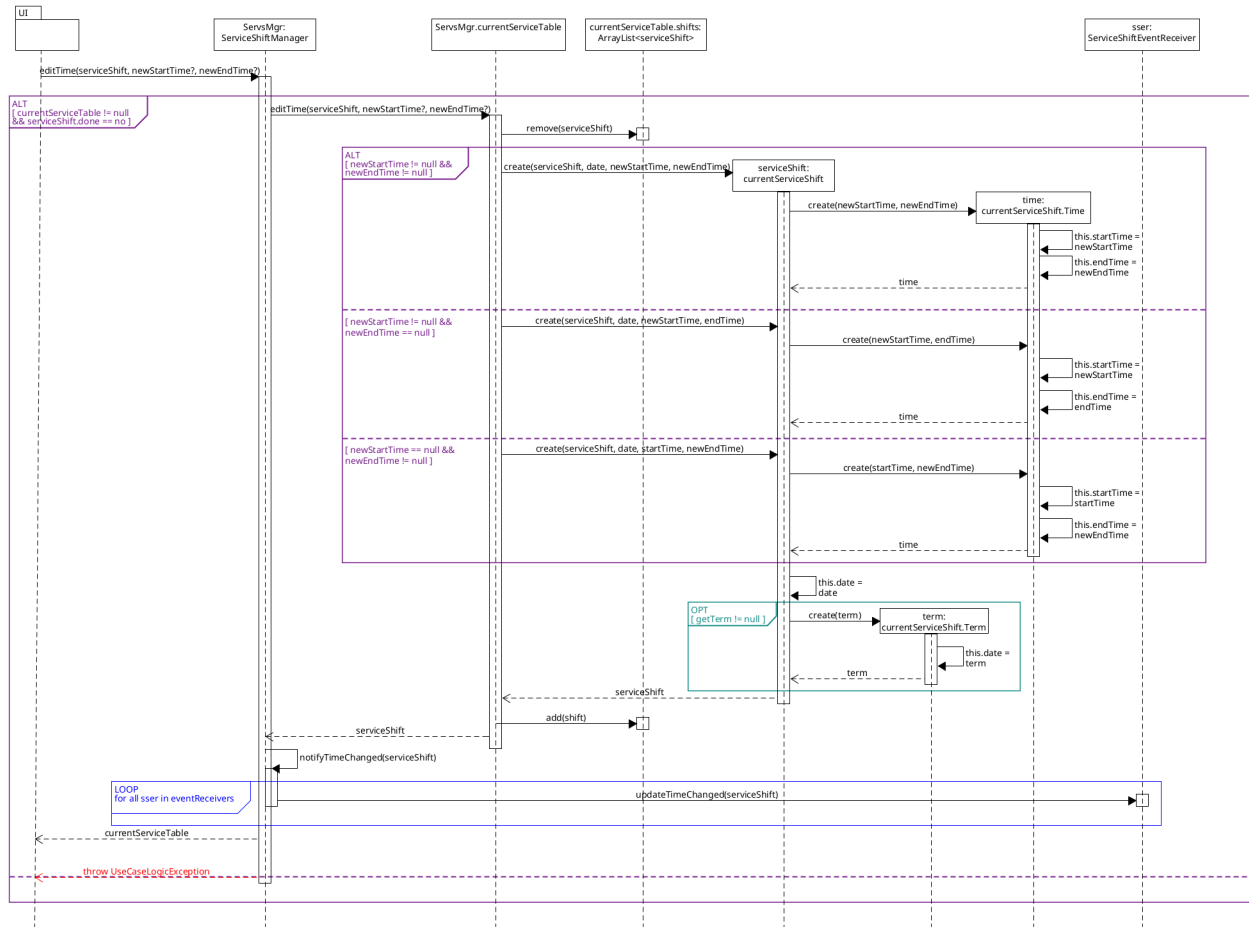
1c. editAllShifts(day?, week?, startTime?, endTime?, location?)



5. addTerm(serviceShift, term)



6. editTime(serviceShift, newTime)



6a. deleteServiceShift(serviceShift)

