**Contratti per lo UC “Gestire i turni”**

Pre-condizione generale: l’attore è identificato con un’istanza *org* di Organizzatore

1. mostraTabellone ()

Precondizioni:

nessuna precondizione, il tabellone dei turni può essere aperto in qualsiasi momento

Post-condizioni:

il tabellone viene visualizzato da org

1a.1 visualizzazioneFiltrata (filtraData?: Data, filtraOrario?: Ora, filtraCuoco?: Cuoco, filtraEvento?: Evento)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*

Post-condizioni:

* [se è specificato filtraData] schemaTurni.data = filtraData
* [se è specificato filtraEvento] schemaTurni.evento = filtraEvento
* [se è specificato filtraOrario] schemaTurni.orario = filtraOrario
* [se è specificato filtraCuoco] schemaTurni.cuoco = filtraCuoco
* [se è specificato filtraData] schemaTurni.data = filtraData

1b.1 apriSchema (schemaTurni: insieme di turni)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* È richiesto uno schema turni *st* che **è parte del** *tdt* e che era stato precedentemente salvato

Post-condizioni:

* A *tdt* è applicato *st*

2. aggiungiTurnoCucina (cuoco?: Cuoco, data: Data, orarioInizio: Ora, orarioFine: Ora, evento: Evento, compito?: Compito)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*

Post-condizioni:

**compito non è stato assegnato o non è definito, e cuoco è disponibile o non definito**

* **è stata creata** una istanza di TurnoCucina *tc*
* [se è specificato cuoco] *tc.cuoco* **=** cuoco e viene creata una relazione **appartiene** tra cuoco e *tc*
* *tc.evento* **=** evento
* *tc.oraInizio* **=** oraInizio
* *tc.oraFine* **=** oraFine
* *tc.data* **=** data
* [se è specificato compito] *tc.compito =* compito e viene creata una relazione **si svolge in un** tra compito e *tc*
* TurnoCucina *tc* **è parte del** *tdt*

2a.1 eliminazioneTurno (turno: Turno)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* Esiste un turno *t* tale che *t* **è parte del** *tdt*

Post-condizioni:

* [se il cuoco è definito] viene eliminata la relazione **appartiene a** tra *t* e cuoco
* [se il compito è definito] viene eliminata la relazione **si svolge in un** tra *t* e compito
* Viene eliminata la relazione **è parte del** tra *t* e *tdt*

2b.1 applicoTurnoRicorrente (turno: Turno, insieme G di date)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* turno *t* **è parte del** *tdt*

Post-condizioni:

**[se non si ci sono turni con lo stesso orario e la stessa data]**

* per ogni data in G viene creato un turno
* t.data = data
* *t* **è parte del** *tdt*

3. raggruppaTurni (insieme T di Turni, cuoco?:Cuoco)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* **è richiesta** una lista di turni T *lt*
* T **è parte del** *tdt*

Post-condizioni:

**[se cuoco è disponibile o non definito]**

* È stata aggiunta una istanza di T**che è parte del** *tdt*
* T **appartengono a** Raggruppamento

4. modificaTurno (orarioInizio?: Ora, orarioFine?: Ora, turno:Turno, data?:Data, cuoco?: Cuoco)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* Turno *t* **è parte del** *tdt*

Post-condizioni:

**[se cuoco è disponibile o non definito]**

* [se è specificato orarioInizio] *t.orarioInizio =*  orarioInizio
* [se è specificato orarioFine] *t.orarioFine =*  orarioFine
* [se è specificato cuoco] *t.cuoco =* cuoco
* [se è specificato data] *t.data =* data
* Turnoè stato modificato

4a.1 modificaTurniRicorrenti (orarioInizio?: Ora, orarioFine?: Ora, insieme T di Turni, data?:Data, cuoco?: Cuoco)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* T**è parte del** *tdt*

Post-condizioni:

**se cuoco è disponibile o non definito**

* [se è specificato orarioInizio] *T.orarioInizio =*  orarioInizio
* [se è specificato orarioFine] *T.orarioFine =*  orarioFine
* [se è specificato cuoco] *T.cuoco =* cuoco
* [se è specificato data] *T.data =* data
* T *è* stato modificato

4b.1 eliminaTurniRicorrenti (turno:Turno)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* Turno *t* **è parte del** *tdt*

Post-condizioni:

[per ogni turno con stessa ora di inizio e fine e stesso cuoco , se definito, di *t]*

* [se un cuoco *c* **appartiene a** *t*] viene eliminata la relazione **appartiene** tra *c* e *t*
* Viene eliminata la relazione **è parte del** tra T e *tdt*

5. aggiungiScadenza (insieme T di Turni, dataScadenza: Data)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* T **è parte del** *tdt*

Post-condizioni:

* *T.dataScadenza* = dataScadenza

5a.1 modificaScadenza (insieme T di Turni, dataScadenza: Data)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* T **è parte del** *tdt*

Post-condizioni:

**se la data di scadenza non oltrepassa la data del turno**

* *T.dataScadenza* = dataScadenza

6 mostraTabellone ()

Precondizioni:

nessuna precondizione, il tabellone dei turni può essere aperto in qualsiasi momento

Post-condizioni:

il tabellone viene visualizzato da org

6a.1 aproSchemaEvento (evento: Evento)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* È richiesto un evento
* Esiste un’istanza di Servizio *s* che **si svolge durante** evento

Post-condizioni:

* Viene restituita la lista dei turni lt tale che lt **appartiene** **a** *s*

7. aggiungoTurnoServizio (servizio: Servizio, oraInizio?: Orario, oraFine?: Orario, addettoServizio?: testo, luogo?: testo, data: Data, cuoco?:Cuoco)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* esiste un Evento *ev* tale che servizio **si svolge durante** *ev* e *org* **ha in carico** *ev*

Post-condizioni:

* **è stata creata** una istanza di turno di servizio *ts*
* [se è specificato orarioInizio] *ts.orarioInizio =*  ora Inizio
* [se è specificato orarioFine] *ts.orarioFine =*  oraFine
* [se è specificato data] *ts.data =* data
* [se è specificato addettoServizio] *ts.addettoServizio =* addettoServizio
* [se è specificato luogo] *ts.luogo =* luogo
* *ts* **appartiene a** servizio *s*
* *ts* **è parte del** tabellone dei turni *tdt*

7a.1 modificaTurnoServizio (turnoServizio: TurnoServizio, oraInizio?: Orario, oraFine?: Orario, addettoServizio?: testo, luogo?: testo, data?: Data)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* esiste un Evento *ev* tale che Servizio *sr* **si svolge durante** *ev* e turnoServizio **appartiene** *sr*

Post-condizioni:

* [se è specificato orarioInizio] *ts.orarioInizio =*  oraInizio
* [se è specificato orarioFine] *ts.orarioFine =*  oraFine
* [se è specificato data] *ts.data =* data
* [se è specificato addettoServizio] *ts.addettoServizio =* addettoServizio
* [se è specificato luogo] *ts.luogo =* luogo
* Una istanza di *ts* è stata modificata

7b.1 eliminaTurnoServizio (turnoServizio: TurnoServizio, servizio: Servizio, tabelloneDeiTurni: TabelloneTurni )

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* esiste un Evento *ev* tale che Servizio *sr* **si svolge durante** *ev* e turnoServizio **appartiene** *sr*
* turnoServizio **è parte del** *tdt*

Post-condizioni:

* viene eliminata la relazione **appartiene a** tra *ts* e addetto al servizio
* viene eliminata la relazione **appartiene a** tra *ts* e s
* viene eliminata la relazione **è parte del** tra *ts* e *tdt*
* [se è specificato luogo] *ts.luogo =* luogo

7c.1 impostaTempoPrePostTurno (turnoServizio: TurnoServizio, tempoPre?: numero, tempoPost?: numero)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* esiste un Evento *ev* tale che Servizio *sr* **si svolge durante** *ev* e turnoServizio **appartiene** *sr*

Post-condizioni:

* [se è specificato tempoPre] *turnoServizio.tempoPre =*  tempoPre
* [se è specificato tempoPost] *turnoServizio.tempoPost =*  tempoPost

7d.1 applicoRicorrenza (servizio: Servizio, insieme G di date, evento: Evento)

Precondizioni:

* E’ in corso la compilazione del Tabellone dei Turni *tdt*
* Per ogni elemento di G esiste un Evento *evv tale evv.data != ev.data*
* esiste un Evento *ev* tale che servizio **si svolge durante** *ev* e *org* **ha in carico** *ev*

Post-condizioni:

* per ogni elemento di G viene creato un servizio
* *servizio.data = data*
* *servizio appartiene ….*

8. salvaSchemaTurniEvento(evento:Evento)

Precondizioni:

* Esiste un Evento *ev* tale che *org* **ha in carico** *ev*

Post-condizioni:

* La sessione è terminata