**Pre-condizioni generali:**

* L’attore è identificato con un’istanza di *org* di Organizzatore

**1 SelezionaCucina(cucina: Cucina)**

**Pre-condizioni:** ---

**Post-condizioni:**

* Se era in corso una precedente selezione di una Cucina, la selezione ha smesso di essere in corso
* E’ in corso la selezione di cucina

**2 CreaTurniCucinaCumulativamente(dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data, scadenzaDisponibilità?: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni Data d tale che:
  + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= d <= allaData
  + Non esiste un **Turno di cucina** *t* tale che:
    - *t* **è tenuto nella** *cucina*
    - *t* **è tenuto nella** *d*
    - dalleOre<= *t*.inizio<=alleOre oppure dalleOre<= *t*.fine<=alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(d) <=alGiornoSettimana

viene creato un Turno di cucina *t* tale che:

* + *t* **è tenuto nella** *d*
  + *t*.inizio *=* dalleOre
  + *t*.fine = alleOre
  + [Se specificata scadenzaDisponibilità]:

*t* **scade disponibilità alla** *d*

* + *org* **organizza** *t*
  + *t* **è tenuto nella** *cucina*

**2a.1 CreaTurnoCucina(dalleOre: Orario, alleOre: Orario, data: Data, scadenza?** : **Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* Non esiste un Turno di cucina *t* tale che:
  + *t* **è tenuto nella** *cucina*
  + *t* **è tenuto nella** data
  + dalleOre<= *t*.inizio<=alleOre oppure dalleOre<= *t*.fine<=alleOre

**Post-condizioni:**

* E’ stata creata un’istanza *t* di Turno di cucina
* *t* **è tenuto nella** *d*
* *t*.inizio = dalleOre
* *t*.fine = alleOre
* [Se specificata scadenza]:*t* **scade disponibilità alla** *d*
* *org* **organizza** *t*
* *t* **è tenuto nella** *cucina*

**2b.1.EliminaTurniCucinaCumulativamente(dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni Turno di cucina *t* tale che:
  + *t* **è tenuto nella** *cucina*
  + Non esiste un Cuoco *c* tale che *c* **è disponibile** nel *t*
  + *Esiste una data d tale che t* **è tenuto nella** *d*
  + *org* **organizza** *t*
  + *t*.inizio >= dalleOre
  + *t*.fine <= alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(*d*) <=alGiornoSettimana

* + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= *d* <= allaData

*t è stato eliminato*

**2c.1.EliminaTurnoCucina(turnoCucina: Turno di cucina)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* turnoCucina **è tenuto nella** *cucina*
* *org* **organizza** turnoCucina
* Non esiste un Cuoco *c* tale che *c* **è disponibile** nel turnoCucina

**Post-condizioni:** turnoCucina è stato eliminato

**2d.1.ModificaTurnoCucina(turnoCucina: Turno di cucina, dalleOre?: Orario, alleOre?: Orario, data?: Data, scadenzaDisponibilità?: Data, èValidaScadenza: si/no)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* turnoCucina **è tenuto nella** *cucina*
* *org* **organizza** turnoCucina
* Non esiste un Cuoco *c* tale che *c* **è disponibile** nel turnoCucina
* *Esiste una Data dataTurnoCucina tale che* turnoCucina **è tenuto nella** *dataTurnoCucina*
* [se specificato alleOre]:
  + [se specificato dalleOre]: alleOre>dalleOre

altrimenti: alleOre> turnoCucina.inizio

altrimenti:

* [se specificato dalleOre]: turnoCucina.fine > dalleOre
* Non esiste un Turno di cucina *t* tale che:
  + *t* non è turnoCucina
  + *t* **è tenuto nella** *cucina*
  + *Esiste una data d tale che t* **è tenuto nella** *d*
  + [se specificata data]: *d* = data

*altrimenti: d = dataTurnoCucina*

* + [se specificato dalleOre]:
    - [se specificato alleOre]:
      * dalleOre<= *t*.inizio<=alleOre oppure dalleOre<= *t*.fine<=alleOre
    - Altrimenti:
      * dalleOre<= *t*.inizio<=turnoCucina.fine oppure dalleOre<= *t*.fine<=turnoCucina.fine

*altrimenti:*

* + - [se specificato alleOre]:
      * turnoCucina.inizio<= *t*.inizio<=alleOre oppure turnoCucina.inizio<= *t*.fine<=alleOre
    - Altrimenti:
      * turnoCucina.inizio<= *t*.inizio<=turnoCucina.fine oppure turnoCucina.inizio<= *t*.fine<=turnoCucina.fine

**Post-condizioni:**

* [se specificato dalleOre]: turnoCucina.inizio = dalleOre
* [se specificato alleOre]: turnoCucina.fine = alleOre
* [se specificata data]:
  + turnoCucina **è tenuto nella** *data*
* [se èValidaScadenza = si]
  + [se specificata scadenzaDisponibilità]:
    - turnoCucina **scade disponibilità alla** scadenzaDisponibilità

altrimenti:

* Se esiste una Data *d* tale che turnoCucina **scade disponibilità alla** *d* allora il legame d’associazione tra le due istanze viene eliminato

**3.RaggruppaTurniCucinaCumulativamente(dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni coppia di Turni di cucina *t1, t2* tale che:
  + *t1* **è tenuto nella** *cucina*
  + *t2* **è tenuto nella** *cucina*
  + *Esiste una Data d1 tale che t1* ***è tenuto nella*** *d1*
  + *Esiste una Data d2 tale che t2* ***è tenuto nella*** *d2*
  + *org* **organizza** *t1*
  + *org* **organizza** *t2*
  + *t1*.inizio >= dalleOre
  + *t2*.inizio >= dalleOre
  + *t1*.fine <= alleOre
  + *t2*.fine <= alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(d1) <=alGiornoSettimana

* dalGiornoSettimana <= F(d2) <=alGiornoSettimana
  + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= *d1* <= allaData
  + dallaData <= *d2* <= allaData
  + Non è vero che *t1* **è raggruppato con** *t2*

*t1* **è raggruppato con** *t2*

**3a.1.RaggruppaTurniCucinaEsplicitamente(insieme turniCucina di Turni di cucina)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Per ogni Turno di cucina *t* appartenente a turniCucina:
  + *t* **è tenuto nella** *cucina*
  + *org* **organizza** *t*

**Post-condizioni:**

* Per ogni coppia di Turni di cucina *t1, t2* appartenenti a turniCucina tale che:
  + Non è vero che *t1* **è raggruppato con** *t2*

*t1* **è raggruppato con** *t2*

**3b.1.** **EliminaRaggruppamentoTurniCucinaCumulativamente (dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni coppia di Turni di cucina *t1, t2* tale che:
  + *t1* **è tenuto nella** *cucina*
  + *t2* **è tenuto nella** *cucina*
  + *Esiste una Data d1 tale che t1* ***è tenuto nella*** *d1*
  + *Esiste una Data d2 tale che t2* ***è tenuto nella*** *d2*
  + *org* **organizza** *t1*
  + *org* **organizza** *t2*
  + *t1*.inizio >= dalleOre
  + *t2*.inizio >= dalleOre
  + *t1*.fine <= alleOre
  + *t2*.fine <= alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(d1) <=alGiornoSettimana

* dalGiornoSettimana <= F(d2) <=alGiornoSettimana
  + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= *d1* <= allaData
  + dallaData <= *d2* <= allaData
  + *t1* **è raggruppato con** *t2*

Il legame d’associazione *t1* **è raggruppato con** *t2* è stato eliminato

**3c.1.** **EliminaRaggruppamentoTurniCucinaEsplicitamente (insieme turniCucina di Turni di cucina)**

**Pre-condizioni:**

* E’ in corso la selezione di *cucina*
* Per ogni Turno di cucina *t* appartenente a turniCucina:
  + *t* **è tenuto nella** *cucina*
  + *org* **organizza** *t*

**Post-condizioni:**

* Per ogni coppia di Turni di cucina *t1, t2* appartenenti a turniCucina tale che:
  + *t1* **è raggruppato con** *t2*

Il legame d’associazione *t1* **è raggruppato con** *t2* è stato eliminato

**(1-3)a.1.SelezionaEvento(evento: Evento)**

**Pre-condizioni:** *org* **organizza** evento

**Post-condizioni:**

* Se era in corso una precedente selezione di un Evento, la selezione ha smesso di essere in corso
* E’ in corso la selezione di evento

DA RIVEDERE E EVENTUALEMENTE MODIFICARE MODELLO DI DOMINIO

**(1-3)a.2.ImpostaRicorrenzaEvento(insieme dateRicorrenza di Date)**

**Pre-condizioni:** E’ in corso la selezione di *evento*

**Post-condizioni:** Per ogni Data *d* appartenente a dateRicorrenza: *evento* **occorre nella** d

**(1-3)a.3.SelezionaServizio(servizio: Servizio)**

**Pre-condizioni:**

* E’ in corso la selezione di *evento*
* *evento* **prevede** servizio

**Post-condizioni:**

* Se era in corso una precedente selezione di un Servizio, la selezione ha smesso di essere in corso
* E’ in corso la selezione di servizio

**(1-3)a.4.CreaTurniServizioCumulativamente(dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data, scadenzaDisponibilità?: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* *servizio*.inizio <= dalleOre
* *servizio*.fine <= alleOre
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni Data d tale che:
  + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= d <= allaData
  + Non esiste un Turno di servizio *t* tale che:
    - *t* **è riferito a** *servizio*
    - *t* **è tenuto nella** *d*
    - dalleOre<= *t*.inizio<=alleOre oppure dalleOre<= *t*.fine<=alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(d) <=alGiornoSettimana

viene creato un Turno di servizio *t* tale che:

* + *t* **è tenuto nella** *d*
  + *t*.inizio *=* dalleOre
  + *t*.fine = alleOre
  + [Se specificata scadenzaDisponibilità]:

*t* **scade disponibilità alla** *d*

* + *org* **organizza** *t*
  + *t* **è riferito a** *servizio*

**(1-3)a.4a.1.CreaTurnoServizio(dalleOre: Orario, alleOre: Orario, data: Data, scadenza?** : **Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* *servizio*.inizio <= dalleOre
* *servizio*.fine <= alleOre
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* Non esiste un Turno di servizio *t* tale che:
* *t* **è riferito a** *servizio*
  + *t* **è tenuto nella** data
  + dalleOre<= *t*.inizio<=alleOre oppure dalleOre<= *t*.fine<=alleOre

**Post-condizioni:**

* E’ stata creata un’istanza *t* di Turno di servizio
* *t* **è tenuto nella** *d*
* *t*.inizio = dalleOre
* *t*.fine = alleOre
* [Se specificata scadenza]:*t* **scade disponibilità alla** *d*
* *org* **organizza** *t*
* *t* **è riferito a** *servizio*

**(1-3)a.4b.1.EliminaTurniServizioCumulativamente(dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni Turno di servizio *t* tale che:
  + *t* **è riferito a** *servizio*
  + Non esiste un’istanza di Personale di servizio *s* tale che *s* **è disponibile** nel *t*
  + *Esiste una data d tale che t* **è tenuto nella** *d*
  + *org* **organizza** *t*
  + *t*.inizio >= dalleOre
  + *t*.fine <= alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(*d*) <=alGiornoSettimana

* + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= *d* <= allaData

*t è stato eliminato*

**(1-3)a.4c.1.EliminaTurnoServizio(turnoServizio: Turno di servizio)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* turnoServizio **è riferito a** *servizio*
* *org* **organizza** turnoServizio
* Non esiste un’istanza di Personale di servizio s tale che *s* **è disponibile** nel turnoServizio

**Post-condizioni:** turnoServizio è stato eliminato

**(1-3)a.4d.1.ModificaTurnoServizio(turnoServizio: Turno di servizio, dalleOre?: Orario, alleOre?: Orario, data?: Data, scadenzaDisponibilità?: Data, èValidaScadenza: si/no)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* [Se specificato dalleOre] *servizio*.inizio <= dalleOre
* [Se specificato alleOre] *servizio*.fine <= alleOre
* turnoServizio **è riferito a**  *servizio*
* *org* **organizza** turnoServizio
* Non esiste un’istanza di Personale di servizio s tale che *s* **è disponibile** nel turnoServizio
* *Esiste una Data dataTurnoServizio tale che* turnoServizio **è tenuto nella** *dataTurnoServizio*
* [se specificato alleOre]:
  + [se specificato dalleOre]: alleOre>dalleOre

altrimenti: alleOre> turnoServizio.inizio

altrimenti:

* [se specificato dalleOre]: turnoServizio.fine > dalleOre
* Non esiste un Turno di servizio *t* tale che:
  + *t* non è turnoServizio
  + *t* **è riferito a** *servizio*
  + *Esiste una data d tale che t* **è tenuto nella** *d*
  + [se specificata data]: *d* = data

*altrimenti: d = dataTurnoServizio*

* + [se specificato dalleOre]:
    - [se specificato alleOre]:
      * dalleOre<= *t*.inizio<=alleOre oppure dalleOre<= *t*.fine<=alleOre
    - Altrimenti:
      * dalleOre<= *t*.inizio<=turnoServizio.fine oppure dalleOre<= *t*.fine<=turnoServizio.fine

*altrimenti:*

* + - [se specificato alleOre]:
      * turnoServizio.inizio<= *t*.inizio<=alleOre oppure turnoServizio.inizio<= *t*.fine<=alleOre
    - Altrimenti:
      * turnoServizio.inizio<= *t*.inizio<=turnoServizio.fine oppure turnoServizio.inizio<= *t*.fine<=turnoServizio.fine

**Post-condizioni:**

* [se specificato dalleOre]: turnoServizio.inizio = dalleOre
* [se specificato alleOre]: turnoServizio.fine = alleOre
* [se specificata data]:
  + turnoServizio **è tenuto nella** *data*
* [se èValidaScadenza = si]
  + [se specificata scadenzaDisponibilità]:
    - turnoServizio **scade disponibilità alla** scadenzaDisponibilità

altrimenti:

Se esiste una Data *d* tale che turnoServizio **scade disponibilità alla** *d* allora il legame d’associazione tra le due istanze viene eliminato

**(1-3)a.5.AssegnaMansioneServizio(turnoServizio: Turno di servizio, personaleServizio: Personale di servizio)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* turnoServizio **è riferito a**  *servizio*
* personaleServizio **è disponibile nel** turnoServizio

**Post-condizioni:**

* E’ stata creata un’istanza di Mansione di servizio *m*
* *m* **è svolta nel** turnoServizio
* *m* **è svolta da** personaleServizio

**(1-3)a.5a.1.EliminaMansioneServizio(mansioneServizio: Mansione di servizio)**

**Pre-condizioni:**

* Esiste un Turno di servizio *t* tale che mansioneServizio è svolta nel *t*
* *org* **organizza** *t*

**Post-condizioni:** mansioneServizio è stata eliminata

**(1-3)a.6.1.RaggruppaTurniServizioCumulativamente(dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni coppia di Turni di servizio *t1, t2* tale che:
  + *t1* **è riferito a** *servizio*
  + *t2* **è riferito a** *servizio*
  + *Esiste una Data d1 tale che t1* ***è tenuto nella*** *d1*
  + *Esiste una Data d2 tale che t2* ***è tenuto nella*** *d2*
  + *org* **organizza** *t1*
  + *org* **organizza** *t2*
  + *t1*.inizio >= dalleOre
  + *t2*.inizio >= dalleOre
  + *t1*.fine <= alleOre
  + *t2*.fine <= alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(d1) <=alGiornoSettimana

* dalGiornoSettimana <= F(d2) <=alGiornoSettimana
  + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= *d1* <= allaData
  + dallaData <= *d2* <= allaData
  + Non è vero che *t1* **è raggruppato con** *t2*

*t1* **è raggruppato con** *t2*

**(1-3)a.6a.1.RaggruppaTurniServizioEsplicitamente(insieme turniServizio di Turni di servizio)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* Per ogni Turno di servizio *t* appartenente a turniServizio:
  + *t* **è riferito a** *servizio*
  + *org* **organizza** *t*

**Post-condizioni:**

* Per ogni coppia di Turni di servizio *t1, t2* appartenenti a turniServizio tale che:
  + Non è vero che *t1* **è raggruppato con** *t2*

*t1* **è raggruppato con** *t2*

**(1\_3)a.6b.1.EliminaRaggruppamentoTurniServizioCumulativamente (dalleOre: Orario, alleOre: Orario, dalGiornoSettimana: Numero, alGiornoSettimana: Numero, dallaData: Data, allaData: Data)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* Considerando un ordinamento temporale effettuabile sugli orari: alleOre > dalleOre
* 1<= dalGiornoDellaSettimana <= 7
* 1<= alGiornoDellaSettimana <= 7
* alGiornoDellaSettimana >= dalGiornoDellaSettimana
* Considerando un ordinamento temporale effettuabile sulle date: dallaData <= allaData

**Post-condizioni:**

* Per ogni coppia di Turni di cucina *t1, t2* tale che:
  + *t1* **è riferito a** *servizio*
  + *t2* **è riferito a** *servizio*
  + *Esiste una Data d1 tale che t1* ***è tenuto nella*** *d1*
  + *Esiste una Data d2 tale che t2* ***è tenuto nella*** *d2*
  + *org* **organizza** *t1*
  + *org* **organizza** *t2*
  + *t1*.inizio >= dalleOre
  + *t2*.inizio >= dalleOre
  + *t1*.fine <= alleOre
  + *t2*.fine <= alleOre
  + Considerata la funzione F(x) che prende in input una data e restituisce il numero equivalente all’n-esimo giorno della settimana con: 1 <= n <= 7, allora:

dalGiornoSettimana <= F(d1) <=alGiornoSettimana

* dalGiornoSettimana <= F(d2) <=alGiornoSettimana
  + Considerando un ordinamento temporale effettuabile sulle date: dallaData <= *d1* <= allaData
  + dallaData <= *d2* <= allaData
  + *t1* **è raggruppato con** *t2*

Il legame d’associazione *t1* **è raggruppato con** *t2* è stato eliminato

**(1-3)a.6c.1.EliminaRaggruppamentoTurniServizioEsplicitamente (insieme turniServizio di Turni di servizio)**

**Pre-condizioni:**

* E’ in corso la selezione di *servizio*
* Per ogni Turno di servizio *t* appartenente a turniServizio:
  + *t* **è riferito a** *servizio*
  + *org* **organizza** *t*

**Post-condizioni:**

* Per ogni coppia di Turni di servizio *t1, t2* appartenenti a turniServizio tale che:
  + *t1* **è raggruppato con** *t2*

Il legame d’associazione *t1* **è raggruppato con** *t2* è stato eliminato