1. Contratti UC Gestione Eventi

Scenario principale

Pre-condizioni generali:

-L’utente è un Organizzatore or

Passo 1

* creazioneEvento(noteIniziali: Testo, cl: Cliente, dataInizio: Testo, dataFine: Testo, periodo?: Testo, numOccorrenze?: Numero, dataFineRic?: Testo)

Pre-condizioni: Nessuna

Post-condizioni:

* È stato creato un nuovo Evento Capofila ev
* ev.NoteIniziali = noteIniziali
* ev.Stato = “Creazione”
* ev.DataInizio = dataInizio
* ev.DataFine = dataFine
* or organizza ev
* cl comissiona ev
* [Se è specificato periodo]
  + Ev.periodo = periodo
  + [se è specificato numOccorrenze]
    - Ev.Numero occorrenze = numOccorrenze
    - Sono state create numOccorrenze Occorrenze oc, per ciascuna delle quali vale
      * ev è Composto da oc
      * oc.Note Iniziali = ev.Note Iniziali
      * oc.Stato = “Creazione”
      * Se oc è l’i-esima occorrenza, oc.DataInizio = ev.DataInizio + (periodo \* I) (\*)
      * Se oc è l’i-esima occorrenza, oc.DataFine = ev.DataFine + (periodo \* I)
      * cl comissiona ev
      * or organizza ev
* [Se è specificato dataFineRic]
  + Ev.DataFineRicorrenza = dataFineRic
  + Per ogni data della forma dataInizio + (periodo \* k nell’intervallo [dataInizio, dataFine], è stata creata un’Occorrenza oc, tale che
    - ev è Composto da oc
    - oc.Note Iniziali = ev.Note Iniziali
    - oc.Stato = “Creazione”
    - oc.DataInizio = ev.DataInizio + (periodo \* k)
    - oc.DataFine = ev. DataFine + (periodo \* k)
    - cl comissiona ev
    - or organizza ev

modificaEvento(noteIniziali?: Testo, cl?: Cliente, dataInizio?: Testo, dataFine?: Testo, periodo?: Testo, numOccorrenze?: Numero, dataFineRic?: Testo, propMod: Sì/No)

Pre-condizioni:

* or Organizza ev

Post-condizioni:

* [Se è specificato noteIniziali]
  + ev.NoteIniziali = noteIniziali
* [Se è specificato cl]
  + [Se esiste un Cliente cl’ tale che cl’ organizza ev]
    - l’associazione tra cl’ ed ev è stata eliminata
    - cl organizza ev
* [Se è specificato dataInizio]

// le date delle occorrenze sono state adeguate ed il personale

// impegnato in altri eventi è stato liberato

* [Se è specificato dataFine]

// le date delle occorrenze sono state adeguate ed il personale

// impegnato in altri eventi è stato liberato

* [Se ev è un Evento Capofila] // adeguamento ricorrenza
  + [Se periodo è specificato]
  + [Se numeroOccorrenze è specificato]
    - [numeroOccorrenze > ev.NumeroOccorrenze]
      * Sono state create numeroOccorrenze - ev.NumeroOccorrenze nuove occorrenze oc con le stesse caratteristiche di ev e sono state associate ad ev
    - [ev.NumeroOccorrenze > numeroOccorrenze]
      * Sono state mantenute solo le prime numeroOccorrenze occorrenze di cui è composto ev
    - ev.NumeroOccorrenze = numeroOccorrenze
  + [Se dataFineRic è specificato]
    - [dataFineRic > ev.DataFineRiccorrenza]
      * Sono state create nuove occorrenza, a distanza ev.Periodo l’una dall’altra, fino a raggiungere dataFineRic e sono state associate ad ev
    - [altrimenti]
      * Sono state eliminate tutte le occorrenze che compongono ev con data di inizio maggiore di dataFineRic
    - ev.DataFineRicorrenza = dataFineRic
* [Se ev è un’Occorrenza]
  + [Se propMod == Sì]

// le altre occorrenze sono state adeguate in modo che le date rispettino periodo e data di fine ricorrenza

Cose da cambiare in modifica evento:

menù: comporta probabilmente una modifica dei compiti e quindi del personale

conviene cancellare e ricreare

chef

organizzatore

cliente

occorrenze: come previsto sul documento

documentazione

note

stato

note finali

data inizio (controllare coerenza con eventi)

data fine (controllare coerenza con eventi)

Anche qui, conviene una soluzione ibrida, come per la modifica del servizio

eliminazioneEvento(ev: Evento, propMod: Sì/No)

Pre-condizioni:

* or Organizza ev
* ev.stato = “Creazione”

Post-condizioni:

* Per ogni Servizio srv tale che ev prevede srv
  + eliminaServizio è stato applicato ad srv
* [Se il Menù m è in uso in ev]
  + È stata eliminata l’associazione tra m ed ev
* [Se ev è assegnato a Chef c]
  + È stata eliminata l’associazione tra ev e c
* [Se propMod = “Sì”]
  + [Se ev.periodo è specificato] // sto agendo sul capofila
    - Per ogni Occorrenza oc tale che ev composto da oc
      * eliminazioneEvento(oc, propMod)
  + [altrimenti] // sto agendo su un’occorrenza qualsiasi
    - Se ev’ è composto da ev, per ogni Occorrenza oc tale che ev’ composto da oc
      * eliminazioneEvento(oc, propMod)
* È stato eliminato l’evento ev

annullaEvento(ev: Evento, propMod: Sì/No)

Pre-condizioni:

* or Organizza ev

Post-condizioni:

* ev.stato = “Annullato”
* Per ogni servizio srv tale che ev prevede srv // cancelliamo solo la prenotazione
  + Per ogni disponibilità disp in srv
    - disp.prenotato = “No”
    - disp.ruolo = “ “
    - l’associazione tra disp ed srv è stata eliminata
* [Se propMod = Sì]
  + [ev è un Evento Capofila]
    - Per ogni Occorrenza oc tale che e composto da oc
      * è stata applicata annullaEvento(oc, propMod)
  + [altrimenti]
    - Se ev’ è composto da ev, per ogni Occorrenza oc tale che ev’ composto da oc
      * è stata applicata annullaEvento(oc, propMod)

chiusuraEvento(ev: Evento, noteFin?: Testo, propMod: Sì/No)

Pre-condizioni:

* or Organizza ev

Post-condizioni:

* Ev.stato = “Terminato”
* [Se è specificato noteFin] ev.note finali = noteFin
* [Se propMod == Sì]
  + [Se ev è un Evento Capofila]
    - Per ogni Occorrenza oc da cui è composto ev
      * chiusuraEvento(oc, noteFine, propMod)
  + [altrimenti]
    - Se ev’ è composto da ev, per ogni Occorrenza oc tale che ev’ composto da oc
      * chiusuraEvento(oc, noteFine, propMod)

caricamentoEvento(ev: Evento)

Pre-condizioni:

* or organizza ev

Post-condizioni: nessuna

Passo 2

aggiuntaServizio(tipo: Testo, data: Testo, luogo: Testo, numPersone: Testo, menu: Menù. Propaga: Sì/No)

Pre-condizioni:

* È in corso la modifica di un evento e
* e.Stato = “Creazione” (non elenchiamo proprio gli eventi in corso oppure completati tra quelli ai queli si possa aggiungere un servizio)

Post-condizioni:

* È stato creato un nuovo Servizio srv
* Srv.Tipo = tipo
* [e.DataInizio <= data && e.DataFine >= data]
  + Srv.Data = data
* Srv.Luogo = luogo
* Srv.Numero di persone = numPersone
* menu In uso in srv
* e Prevede srv
* [Se non esistono altri Servizi pevisti da e, con la stessa data di srv]
  + e.DataFine = data
* Srv.approvato = No
* [Se e.periodo è specificato e Propaga = Sì]
  + Per ogni Occorrenza oc tale che e composto da oc
    - È stato creato un nuovo Servizio srv’
    - srv’.Tipo = tipo
    - Se oc è l’i-esima occorrenza
      * [oc.DataInizio <= data && oc.DataFine >= data]
        + srv’.Data = data + (e.periodo \* i)
    - srv’.Luogo = luogo
    - srv’.NumeroPersone = numPersone
    - menu In uso in srv’
    - oc Prevede srv’
    - srv’.approvato = No
    - [Se non esistono altri Servizi pevisti da oc, con la data maggiore di srv’.data]
      * e.DataFine = srv’.Data
    - [Se non esistono altri Servizi pevisti da oc, con la data maggiore di srv’.data]
      * e.DataInizio = srv’.Data

modificaServizio(srv: Servizio, tipo?: Testo, data?: Testo, luogo?: Testo, numPartecipanti?: Testo, evento?: Evento, menu?: Menù, propagaModifica: Sì/No)

Pre-condizioni:

* È in corso la modifica di un evento e
* e prevede srv
* ev.Stato = “Creazione” // questo va messo, perchè non è detto che al passo 1 venga aperto un evento in fase di creazione

Post-condizioni:

* [Se è specificato tipo] srv.Tipo = tipo
* [Se è specificato data]
  + è stata applicata aggiuntaServizio(srv.tipo, data, srv.luogo, srv.numPersone, srv.menu. PropMod)
  + è stata applicata eliminaServizio(srv, propMod)
* [Se è specificato luogo] srv.Luogo = luogo
* [Se è specificato numPartecipanti] srv.Numero di persone = numPartecipanti
* [Se è specificato evento]
  + è stata applicata aggiuntaServizio(srv.tipo, srv.data, srv.luogo, srv.numPersone, srv.menu. PropMod), creando un nuovo servizio srv’
  + è stata applicata eliminaServizio(srv, propMod)
  + è stata eliminata l’associazione tra e ed srv’
  + evento prevede srv’
* [Se è specificato menu]
  + è stata applicata aggiuntaServizio(srv.tipo, srv.data, srv.luogo, srv.numPersone, menu. PropMod)
  + è stata applicata eliminaServizio(srv, propMod)
* [Se propagaModifica == Sì]
  + [se e è un Evento Capofila]
    - Per ogni Occorrenza oc tale che e composto da oc
      * modificaServizio è stata applicata, con gli stessi parametri, all’evento previsto da oc equivalente ad srv
  + [altrimenti]
    - Se ev è l’Evento Capofila tale che ev composto da e
      * Per ogni Occorrenza oc tale che ev composto da oc
      * modificaServizio è stata applicata, con gli stessi parametri, all’evento previsto da oc equivalente ad srv

eliminaServizio(srv: Servizio, propMod: Sì/No)

Pre-condizioni:

* ev prevede srv
* ev.Stato = “Creazione”

Post-condizioni:

* [Se m è il Menù in uso in srv]
  + l’associazione tra m ed srv è stata eliminata
* L’associazione tra srv ed ev è stata eliminata
* Per ogni proposta di modifica m tale che m menù di srv
  + l’associazione tra m ed srv è stata eliminata
  + l’associazione tra m e Ricetta è stata eliminata
  + m è stata eliminata
* [Se f è il foglio riepilogativo che contiene compiti di srv]
  + l’associazione tra f ed srv è stata eliminata
  + Per ogni compito c tale che f composto da c
    - l’associazione tra c e turno è stata eliminata
    - l’associazione tra c e Cuoco è stata eliminata
    - [Se c’ è prosecuzione di c]
      * c’ è stato eliminato, così come ogni altro compito sua prosecuzione, diretta od indiretta
    - c è stato eliminato
  + l’associazione tra Chef ed f è stata eliminata
  + f è stato eliminato
* Per ogni Disponibilità d in srv
  + l’associazione tra d ed srv è stata eliminata
  + d.prenotato = “No”
  + d.ruolo = “”
* [Se propMod = “Sì”]
  + [Se ev è un Evento Capofila]
    - Per ogni Occorrenza oc tale che ev Composto da oc
      * eliminaServizio è stata applicata anche al servizio di oc avente le stesse caratteristiche
  + [altrimenti]
    - Se ev’ è l’Evento Capofila tale che ev’ composto da ev]
      * Per ogni Occorrenza oc tale che ev’ Composto da oc
        + eliminaServizio è stata applicata anche al servizio di oc avente le stesse caratteristiche
* ev.DataFine = massima data tra quelle dei servizi associati
* ev.DataFine = minima data tra quelle dei servizi associati

Pre-condizioni:

* Ev prevede srv
* Or organizza ev
* Menu in uso in srv
* Fr contiene compiti di srv
* Pm menù di srv

Post-condizioni:

* Menu non è più associato ad srv
* Fr non è più associato ad srv
* Pm non è più associato ad srv

recuperoElencoServizi(ev: Evento, propMod?: Sì/No)

(Così può vedere solo I servizi dei suoi eventi, ha senso?)

Pre-condizioni:

* Ev prevede srv
* Or organizza ev

Post-condizioni: Nessuna (è un’interrogazione)

Passo 3

assegnazioneChef(ev: Evento, chef: Chef)

Pre-condizioni:

* ev.stato = “Creazione”
* Non esiste un Evento ev’, tale che
  + Ev' assegnato a chef
  + Ev'.dataInizio >= ev.dataInizio ed ev’.dataFine <= ev.dataFine

Post-condizioni:

* ev assegnato a chef

Passo 4

prenotazionePersonale(disp: Disponibilità, srv: Servizio)

Pre-condizioni:

* disp.Prenotato = No
* è in corso la modifica di un evento e
* e prevede srv

Post-condizioni:

* disp.Prenotato = Sì
* Disp in srv
* or prenotazione disp

cancellazionePrenotazione(disp: Disponibilità)

Pre-condizioni

* è in corso la modifica di un evento e
* e prevede srv
* disp in srv
* disp.prenotato = Sì

Post-condizioni:

* disp.prenotato = No
* disp.ruolo = “”
* disp non è più associata a srv

Passo 5

approvazioneMenu(srv: Servizio)

Pre-condizioni:

* Ev prevede srv

Post-condizioni:

* srv.approvato = Sì
* ev.stato = “In corso”

propostaModifica(srv: Servizio, ric: Ricetta, cm: Commento)

Pre-condizioni:

* è in corso la modifica di un evento ev
* ev prevede srv
* ev.stato == “Creazione”

Post-condizioni:

* È stata create una Proposta Modifica pm
* Pm.commento = cm
* Pm.accettata = No
* Pm menù di srv
* Pm riferita a ric

Passo 6

assegnazioneRuolo(disp: Disponibilità, ruolo: Testo)

Pre-condizioni:

* Or prenotazione disp
* Se mp inserimento disp, allora mp è un Membro del personale di servizio

Post-condizioni:

* Disp.Ruolo = ruolo

Cose da fare

* Fare in modo che al passo 1 venga fissato l’evento su cui si lavora
* controllare che l’evento su cui si lavora sia settato: altro (es. Or è l’organizzatore non serve, è già stato verificato al passo 1)
* l’associazione tra organizzatore e prenotazione è ridondante, ma conviene metterla per semplificare e velocizzare la verifica del passo 6
* Anche l’associazione tra prenotazione e membro del personale riservato è ridondante, ma semplifica la verifica delle post-condizioni

Note: (NON CANCELLARE!!!)

* Per definire la semantica delle creazione dell'evento ricorrente, si è assunto che il periodo sia in giorni. È impreciso, perchè potrebbe non esserlo, ma rende l'idea di cosa dovrebbe succedere.