# ordinaPer

Nella funzione ordinaPer della classe RisultatiRicette è necessario, per la corretta esecuzione del programma, che prima dell’esecuzione del metodo sia impostato almeno un filtro. In caso contrario non sarebbe necessario chiamare la funzione. Possiamo descrivere questa condizione con il seguente codice OCL.

context RisultatiRicette::ordinaPer(filtro)

pre : filtro != NULL

# cercaRicette

Nella classe RicercaRicette è presente il metodo cercaRicette. Per permettere a questo metodo di venire chiamato correttamente dobbiamo imporre che uno dei parametri presi in input, in particolare il parametro dispensa, non sia nullo. Questa limitazione è espressa formalmente come segue:

context RicercaRicette::cercaRicette(nomePiatto, filtri, dispensa)

pre : dispensa != NULL

# cercaPerDispensa

Sempre appartenente alla classe sopracitata RicercaRicette, il metodo cercaPerDispensa richiede una limitazione simile a quella dell’analogo cercaRicette poiché il funzionamento di questi due è quasi del tutto identico. Viene quindi espressa l’impossibilità del parametro ricetta di essere nullo in questo modo:

context RicercaRicette::cercaPerDispensa(nomePiatto, filtri, dispensa)

pre : dispensa != NULL

# rimuoviIngrediente

La funzione rimuoviIngrediente della classe Dispensa richiede che la quantità dell’ingrediente che si desidera rimuovere sia minore della quantità che si possiede (attributo della classe Dispensa). Inoltre si prevede che ogni volta che venga chiesto di rimuovere una certa quantità di un certo ingrediente questa debba essere almeno un unità, al contrario infatti il metodo perderebbe di significato. Per descrivere queste condizioni si usa il seguente codice:

context Dispensa::rimuoviIngrediente(ingrediente, quantità)

pre : ingrediente -> count(ingredienti) > 1 && count(ingredienti)<=quantità

inv : quantità[ingrediente] >= 1

# isApproved

Nella classe GestioneNuoveRicette il metodo isApproved è usato per aggiornare lo stato di una richiesta di inserimento di una nuova ricetta. Si impone il che dopo l’esecuzione del metodo il parametro booleano approved abbia uno stato definito. Di seguito il codice per descrivere questa condizione.

context GestoreNuoveRicette::isApproved()

post : approved == true || approved == false

# Segnalazione

Nella classe Segnalazione si decide di imporre all’utente un vincolo, ovvero che il commento legato a una determinata segnalazione non ecceda i 251 caratteri. Questa imposizione utile alla semplicità e mantenibilità del sistema è implementata con il codice OCL:

context Segnalazione inv commento.size() < 251

# isPrimaVisita

Il metodo isPrimaVisita della classe PaginaHome permette al sistema di capire se l’utente ha già visitato il sito in precedenza. Per ottenere un metodo affidabile, si pone come condizioni che la variabile primavisita sia falsa prima appunto di aver eseguito il metodo, e sia true alla fine dell’esecuzione. Per descrivere queste condizioni in OCL si usa il seguente codice:

context PaginaHome::isPrimaVisita(cookie)

pre : primavisita == true

post : primavisita == false