Casi d'Uso Gestire Compiti della Cucina

Nome caso d'uso: Gestire compiti della cucina
Portata: Sistema
Livello: Obiettivo utente
Attore primario: Chef
Parti Interessate: Cuoco
Pre-condizioni: L'attore deve essere autenticato come chef.
Garanzie di successo o post-condizioni: Ogni preparazione è correttamente orgnizzata in turni

Scenario principale di successo

#	Attore	Sistema
1	Crea il foglio riepilogativo per un servizio di un evento	Precompila il foglio riepilogativo
	Se desidera prosegue con il passo 2, altrimenti termina il caso d'uso	
2	Opzionalmente aggiunge preparazioni o ricette fuori menù.	Registra aggiunta preparazioni o ricetta
3	Opzionalmente, ordina una o più attività di cucina	Restituisci l'elenco ordinato
4	Indica le tempistiche, le quantità/porzioni da preparare per una determinata attività di cucina del foglio riepilogativo.	Registra i dati inseriti per l'attività di cucina
5	Opzionalmente indica una rimanenza[1] da utilizzare per l'attività di cucina.	Registra la rimanenza scelta per l'attività di cucina
	Ripeti dal passo 4 per ogni ricetta o preparazione	
	Per l'assegnazione dei compiti prosegui altrimenti torna al passo 2 o termina il caso d'uso.	
6	Opzionalmente, consulta tabellone dei turni	Restituisce tabellone dei turni
7	Assegna un compito specificando quale attività di cucina del foglio riepilogativo deve essere svolta, in quali quantità e tempistiche, il turno e il cuoco.	Registra il compito

[1] La rimanenza sono delle quantità/porzioni di una mansione di cucina preparata e avanzata da un servizio o attività che si è concluso precedentemente

	Ripete dal passo 6 sinché non è soddisfatto *Se desidera torna al passo 2, se no termina il caso d'uso.*

Estensione 1a

#	Attore	Sistema
1a.1	Sceglie un foglio riepilogativo esistente per modificarlo	Registra le modifiche del foglio riepilogativo

Estensione 1b

#	Attore	Sistema
1b.1	Sceglie un foglio riepilogativo esistente per visualizzarlo	Fornisci il foglio riepilogativo scelto e le relativi dati aggiornati in tempo reale
	Termina caso d'uso	

Estensione 2a

#	Attore	Sistema
2a.1	Rimuovi un'attività di cucina dal foglio riepilogativo.	Rimuovi l'attività di cucina dal foglio riepilogativo
	Ripeti passo 2a.1 finché non soddisfatto	

Estensione 4a

#	Attore	Sistema
4a.1	Modifica attività di cucina per una determinata ricetta o preparazione del foglio riepilogativo.	Registra le modifica effettuata
	Ripeti passo 4a.1 finché non soddisfatto	

Estensione 7a

#	Attore	Sistema
7a.1	Modifica un compito specificando ricetta/preparazione, oppure quantità e tempistiche oppure il turno oppure il cuoco.	Fornisci il compito scelto
	Ripeti passo 7a.1 finché non soddisfatto	

Estensione 7b

#	Attore	Sistema
7b.1	Elimina un compito assegnato	Registra eliminazione del compito
	Ripeti passo 7b.1 finché non soddisfatto	

Estensione 7c

#	Attore	Sistema
7b.1	Imposta il turno come completo	Registra il turno come completo
	Ripeti passo 7b.1 finché non soddisfatto	

Eccezione 1a

#	Attore	Sistema
1a.1	Crea il foglio riepilogativo per un servizio di un evento (di cui ha ricevuto l'incarico)	Il servizio non è stato modificato
	Termina caso d'uso	

Eccezione 1a.1a

#	Attore	Sistema
1a.1a.1	Sceglie un foglio riepilogativo esistente per modificarlo	Non si hanno i permessi per poter modificare il foglio riepilogativo
	Termina caso d'uso	

Eccezione 1a.1b

#	Attore	Sistema
1a.1b.1	Sceglie un foglio riepilogativo esistente per modificarlo	Il servizio associato al foglio riepilogativo non è confermato
	Termina caso d'uso	

Eccezione 1b.1a

#	Attore	Sistema
1b.1a.1	Sceglie un foglio riepilogativo esistente per visualizzarlo	Non esiste un foglio associato al servizio
	Termina caso d'uso	

Eccezione 5a

#	Attore	Sistema
5a.1	Opzionalmente indica una rimanenza[2] da utilizzare per l'attività di cucina.	La rimanenza è già stata utilizzata da un'altra attività di cucina
	Termina caso d'uso	

[2] La rimanenza sono delle quantità/porzioni di una mansione di cucina preparata e avanzata da un servizio o attività che si è concluso precedentemente

Eccezione 7a

#	Attore	Sistema
7a.1	Assegna un compito specificando quale ricetta/preparazione del foglio riepilogativo, in quali quantità e tempistiche, il turno e il cuoco.	Il turno selezionato si è già concluso o è già completo
	Termina caso d'uso	

Eccezione 7a.1a

#	Attore	Sistema
7a.1 a.1	Modifica un compito specificando ricetta/preparazione, oppure quantità e tempistiche oppure il turno oppure il cuoco.	Il turno è già concluso o è già completo
	Termina caso d'uso	

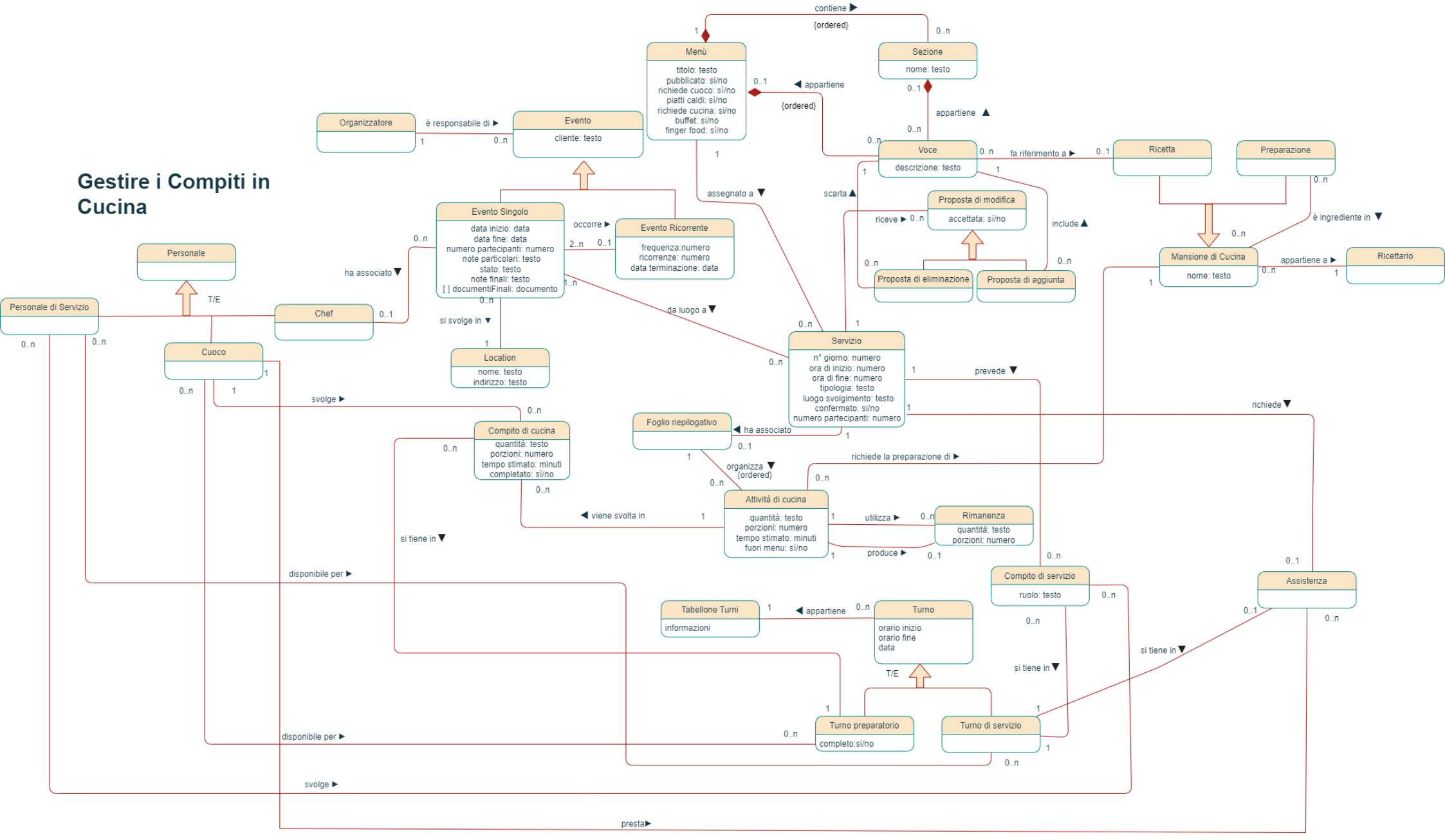
Eccezione 7a.1b

#	Attore	Sistema
7a.1 b.1	Modifica un compito specificando ricetta/preparazione, oppure quantità e tempistiche oppure il turno oppure il cuoco.	Il cuoco non ha abbastanza tempo a disposizione per eseguire il nuovo compito
	Termina caso d'uso	

Eccezione 7b.1a

#	Attore	Sistema
7a.1 b.1	Elimina un compito assegnato	Il turno è già concluso
	Termina caso d'uso	

Gestire i Compiti in Cucina



Organizzatore

SSD per l'UC Gestire Compiti Cucina
Scenario principale

:Sistema

1. creaFoglioRiepilogativo(evento, servizio)

ALT

foglio precompilato

foglio precompilato

LOOP

LOOP

2. aggiungiFuoriMenu(mansione)

ok

OPT

3. ordinaFoglioRiepilogativo(ordinamento)

ok

LOOP

4. indicaStime(attivitàDiCucina, quantità?, porzione?, tempoStimato)

ok

OPT

5. utilizzaRimanenza(attività di cucina, rimanenza)

ALT

ok

ok

LOOP

OPT

6. tabelloneTurni()

tabellone turni

7. assegnaCompitoDiCucina(attivitàDiCunina, quantità?, porzione?,
tempoStimato, turno, cuoco)

ALT

ok

ok

SSD per l'UC Gestire Compiti Cucina
Estensioni ed eccezioni del passo 1

Organizzatore

:Sistema

ALT

1. creaFoglioRiepilogativo(evento, servizio)

ALT

foglio precompilato

foglio precompilato

1a.1 il servizio non è stato modificato

1a. apriFoglioRiepilogativo(foglioRiepilogativo)

ALT

foglio precompilato scelto

1a.1a.1 non si hanno i permessi

1a.1b.1 il servizio associato al foglio riepilogativo non è confermato

1b. visualizzaFoglioRiepilogativo(foglioRiepilogativo)

ALT

foglio precompilato scelto

1b.1a.1 non esiste un foglio associato al servizio

LOOP

ALT

OPT

2. aggiungiFuoriMenu(mansione)

ok

LOOP

2a.1 rimuoviAttivitaDiCucina(attivitàDiCucina)

ok

OPT

3. ordinaFoglioRiepilogativo(ordinamento)

ok

LOOP

ALT

4. indicaStime(attivitàDiCucina, quantità?, porzione?, tempoStimato)

ok

4a.1 modificaQuantità(attivitàDiCucina, quantità?, porzione?, tempoStimato?)

ok

OPT

5. utilizzaRimanenza(attivitàDiCucina, rimanenza)

ALT

ok

ok

LOOP

OPT

6. tabelloneTurni()

tabellone turni

7. assegnaCompitoDiCucina(attivitàDiCunina, quantità?, porzione?,
tempoStimato, turno, cuoco)

ALT

ok

ok

Organizzatore

SSD per l'UC Gestire Compiti Cucina
Estensioni ed eccezioni del passo 2, 4 e 5

:Sistema

1. creaFoglioRiepilogativo(evento, servizio)

ALT

foglio precompilato

foglio precompilato

LOOP

ALT

LOOP

2. aggiungiFuoriMenu(mansione)

ok

LOOP

2a.1 rimuoviAttivitaDiCucina(attivitàDiCucina)

ok

OPT

3. ordinaFoglioRiepilogativo(ordinamento)

ok

LOOP

ALT

4. indicaStime(attivitàDiCucina, quantità?, porzione?, tempoStimato)

ok

4a.1 modificaQuantità(attivitàDiCucina, quantità?, porzione?, tempoStimato?)

ok

OPT

5. utilizzaRimanenza(attività di cucina, rimanenza)

ALT

ok

ok

5a.1 La rimanenza è già stata utilizzata da un'altra attività di cucina

LOOP

OPT

6. tabelloneTurni()

tabellone turni

7. assegnaCompitoDiCucina(attivitàDiCunina, quantità?, porzione?,
tempoStimato, turno, cuoco)

ALT

ok

ok

SSD per l'UC Gestire Compiti Cucina
Estensioni ed eccezioni del passo 7

Organizzatore

:Sistema

1. creaFoglioRiepilogativo(evento, servizio)

ALT
- foglio precompilato
- foglio precompilato

LOOP

ALT

OPT
- 2. aggiungiFuoriMenu(mansione)
- ok

LOOP
- 2a.1 rimuoviAttivitaDiCucina(attivitàDiCucina)
- ok

OPT
- 3. ordinaFoglioRiepilogativo(ordinamento)
- ok

LOOP

ALT
- 4. indicaStime(attivitàDiCucina, quantità?, porzione?, tempoStimato)
- ok
- 4a.1 modificaQuantità(attivitàDiCucina, quantità?, porzione?, tempoStimato?)
- ok

OPT
- 5. utilizzaRimanenza(attivitàDiCucina, rimanenza)

ALT
- ok
- ok

LOOP

OPT
- 6. tabelloneTurni()
- tabellone turni

ALT
- 7. assegnaCompitoDiCucina(attivitàDiCucina, quantità?, porzione?, tempoStimato, turno, cuoco)

ALT
- ok
- ok
- 7a.1 Turno selezionato già concluso o è già completo

- 7a. modificaCompitoDiCucina(compitoDiCucina, quantità, porzione?, tempoStimato?, turno?, cuoco?)

ALT
- ok
- ok
- 7a.1a.1 Turno selezionato già concluso o è già completo
- 7a.1b.1 cuoco selezionato non disponibile

- 7b. eliminaCompitoDiCucina(compitoDiCucina)

ALT
- ok
- ok
- 7b.1a.1 Turno selezionato già concluso

- 7c. settaTurnoComeCompleto(turno)

ALT
- ok
- ok

Contratti Gestire Compiti della Cucina

1) creaFoglioRiepilogativo(evento: Evento, servizio: Servizio)

Pre-condizioni:

- è in corso la creazione di un foglio riepilogativo.
- servizio è un servizio appartenente all'evento evento
- servizio ha già il menù assegnato.

Post-condizioni:

- è stata creata un'istanza di Foglio Riepilogativo *foglio*
- *foglio* viene **associato** al servizio
  Per ogni mansione di cucina *mansione* nel menu del servizio
  - è stata creata un'istanza di Attività Di Cucina *adc*
  - *adc* **richiede la preparazione di** *mansione*
  - *adc.fuoriMenu* = no
  - *foglio* **comprende** *adc*

1a) apriFoglioRiepilogativo(foglioRiepilogativo: FoglioRiepilogativo)

Pre-condizioni: -

Post-condizioni: -

1b) visualizzaFoglioRiepilogativo(foglioRiepilogativo: FoglioRiepilogativo)

Pre-condizioni: -

Post-condizioni: -

2) aggiungiFuoriMenu(mansione: MansioneCucina)

Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo *foglio*

Post-condizioni:

- è stata creata un'istanza di Attività Di Cucina *adc*
- *adc*.fuoriMenu = si
- *adc* **richiede la preparazione di** mansione
- *foglio* **comprende** *adc*

2.a) rimuoviAttivitaDiCucina(adc: AttivitàDiCucina)

Pre-condizioni:

- è in corso la modifica del Foglio Riepilogativo *foglio*
- L' Attività di Cucina adc **richiede la preparazione di** *mansione* di MansioneCucina
- *foglio* **organizza** adc

Post-condizioni:

- L'associazione **richiede la preparazione di** tra *mansione* e adc è eliminata
- tutte le istanze di Compito in cui adc **viene svolta** sono state eliminate
- L'istanza adc viene eliminata

3) ordinaFoglioRiepilogativo(ordinamento: Ordine)

Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo *foglio*

Post-condizioni:

- l'associazione **organizza** tra *foglio* e le Attività Di Cucina viene modificata secondo il criterio di ordinamento

4) indicaStime(adc: Attività Di Cucina, quantità?: testo, porzione?: numero, tempoStimato: minuti)

Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo *foglio.*
- *foglio* **organizza** adc.

Post-condizioni:

- adc.tempoStimato = tempoStimato
  [Se quantità è passato come parametro] :
  - adc.quantità = quantità

  [Se porzione è passato come parametro] :
  - adc.porzione = porzione

4a) modificaQuantità(adc: Attività Di Cucina, quantità?: testo, porzione?: numero, tempoStimato?: minuti)

Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo *foglio.*
- *foglio* **organizza** adc.

Post-condizioni:

[Se tempoStimato è passato come parametro] :
- adc.tempoStimato = tempoStimato

[Se quantità è passato come parametro] :
- adc.quantità = quantità
  [Se adc.porzione era stato settato in precedenza] :
  - adc.porzione viene settato con valore di default

[Se porzione è passato come parametro] :
- adc.porzione = porzione

- [Se adc.quantità era stato settato in precedenza] :
    - adc.quantità viene settato con valore di default

5) utilizzaRimanenza(adc: Attività di Cucina, rimanenza: Rimanenza)

Pre-condizioni:

- è in corso la definizione di una Attività Di Cucina *adc*

Post-condizioni:

-E' stata creata l'associazione adc **utilizza** rimanenza

6) tabelloneTurni()

Pre-condizioni: -

Post-condizioni: -

7) assegnaCompitoDiCucina(adc: Attività Di Cucina, quantità?: testo, porzione?: numero, tempoStimato: minuti, turno: Turno, cuoco: Cuoco)

Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo il quale **comprende** *adc*
- cuoco ha dato la **disponibilità** per il turno

Post-condizioni: Se turno.completo = no

- è stata creata un'istanza di Compito Di Cucina *cdc.*
- *cdc* **viene svolta in** adc
- cuoco **svolge** *cdc*
- *cdc* **si tiene in** turno
- *cdc*.completato = no
- *cdc*.tempoStimato = tempoStimato
- [Se quantità è passato come parametro] :
    - *cdc*.quantità = quantità

  [Se porzione è passato come parametro] :
    - *cdc*.porzione = porzione

7a) modifcaCompitoDiCucina(cdc: Compito Di Cucina, quantità?: testo, porzione?: *numero*, tempoStimato?: minuti, turno?: Turno, cuoco?: Cuoco)

Pre-condizioni:

- cuoco ha dato la **disponibilità** per il turno.

Post-condizioni: Se turno viene passato e turno.completo = no

- [Se quantità è passato come parametro ed è diverso dal precedente valore di cdc.quantità] :
    - cdc.quantità = quantità
    - [Se cdc.porzione era stato settato in precedenza] :

        - cdc.porzione viene settato con valore di default
- [Se porzione è passato come parametro ed è diverso dal precedente valore di cdc.porzione] :
    - cdc.porzione = porzione
      [Se *cdc*.quantità era stato settato in precedenza] :
        - cdc.quantità viene settato con valore di default

7b) eliminaCompitoDiCucina(adc: Attività di Cucina, cdc: CompitoDiCucina)

Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo il quale **comprende** l'Attività di Cucina adc
- *adc* **viene svolta in** cdc
- Il Cuoco *cuoco* **svolge** cdc
- *cuoco* ha dato la **disponibilità** per il Turno *turno* in cui **si tiene** cdc

Post-condizioni:

- l'associazione **svolge** tra *cuoco* e cdc è eliminata
- l'associazione **si tiene in** tra *turno* e cdc è eliminata
- l'associazione **viene svolta in** tra adc e cdc è eliminata
- l'istanza cdc è eliminata.

7c) settaTurnoComeCompleto(turno: Turno)

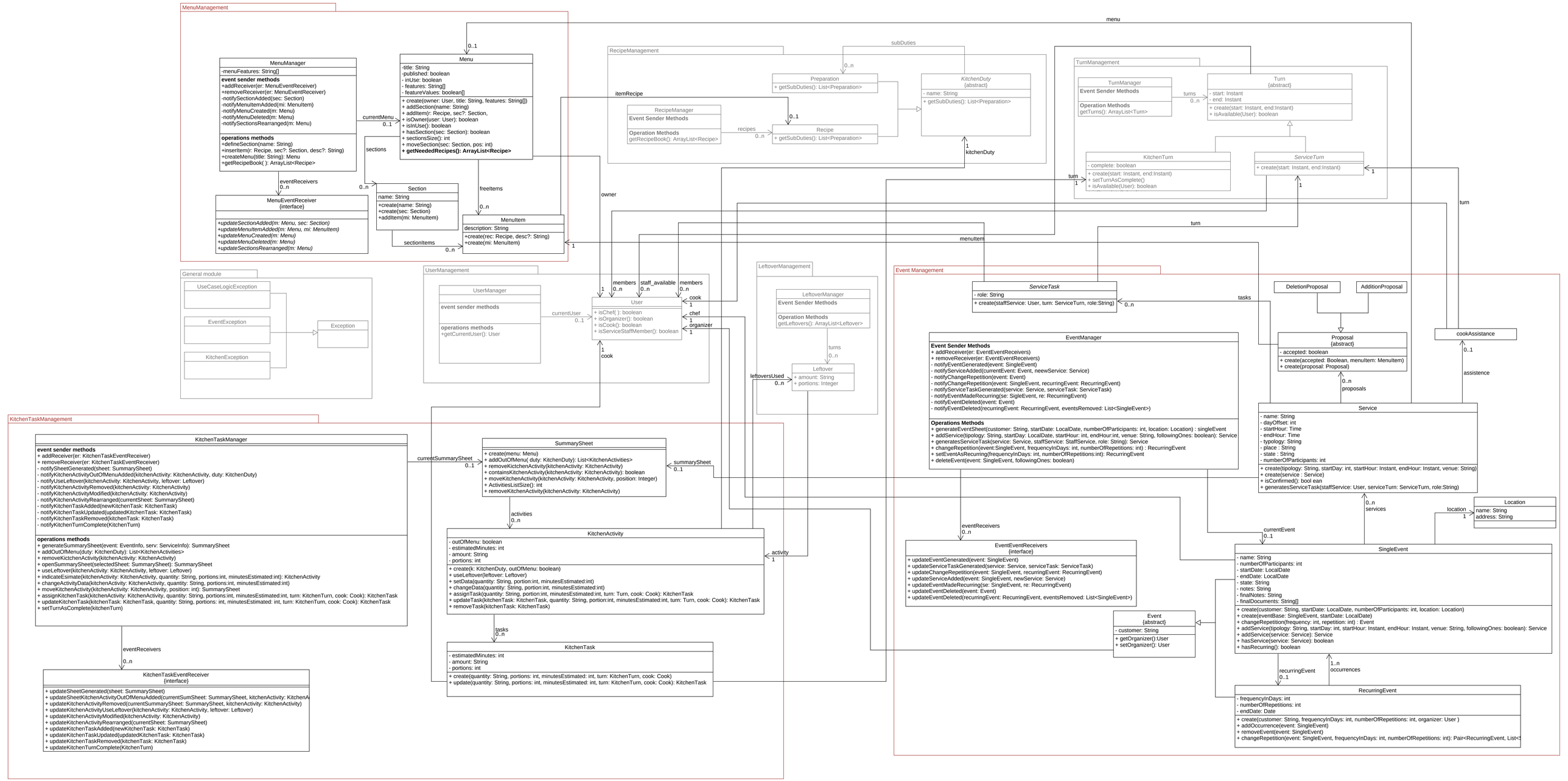
Pre-condizioni:

- è in corso la definizione del Foglio Riepilogativo *foglioRiepilogativo*

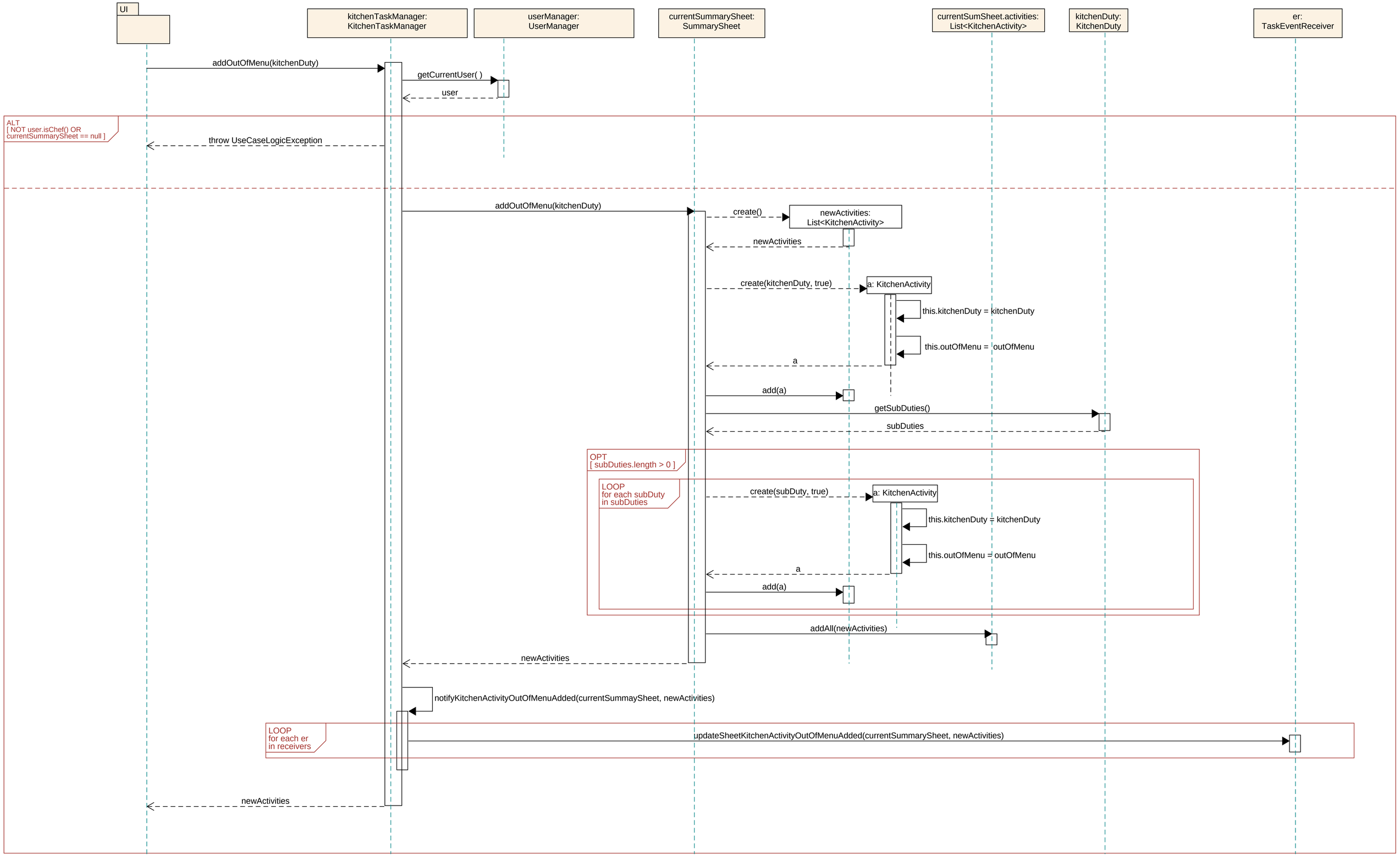
Post-condizioni:

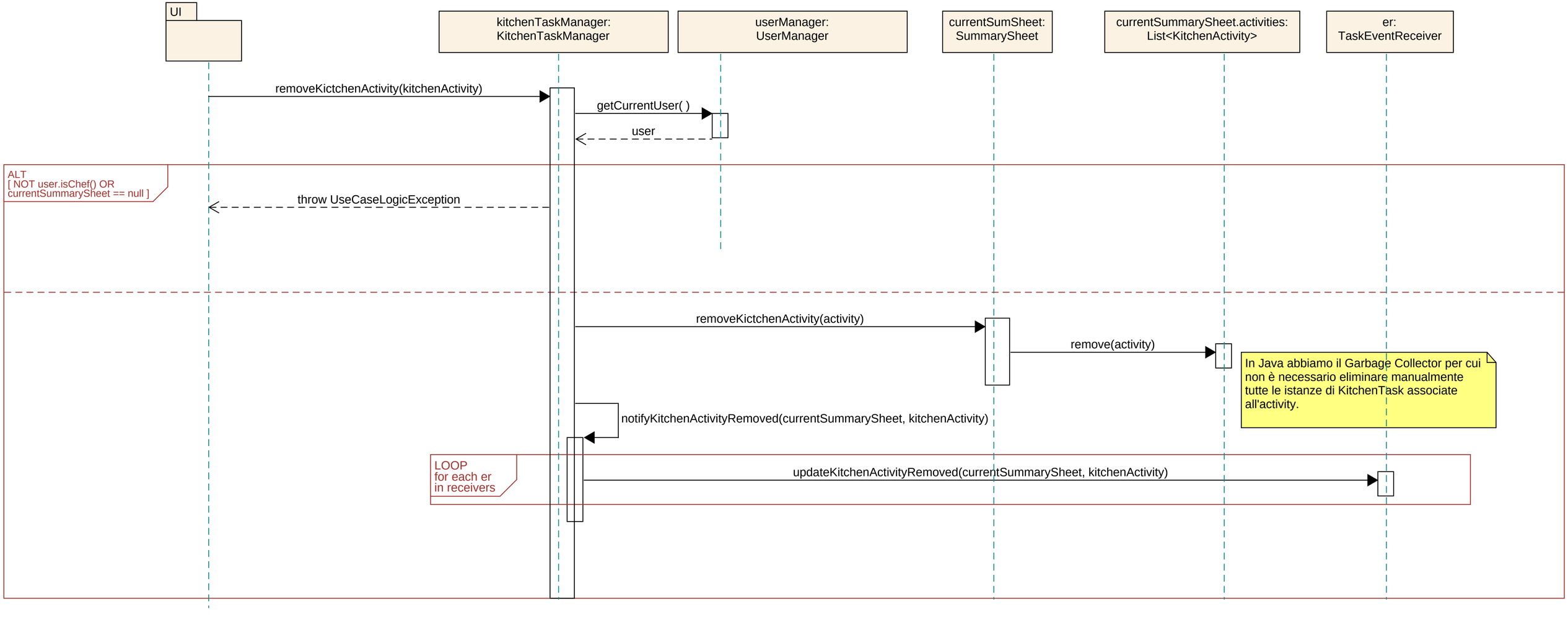
- turno.completo = si

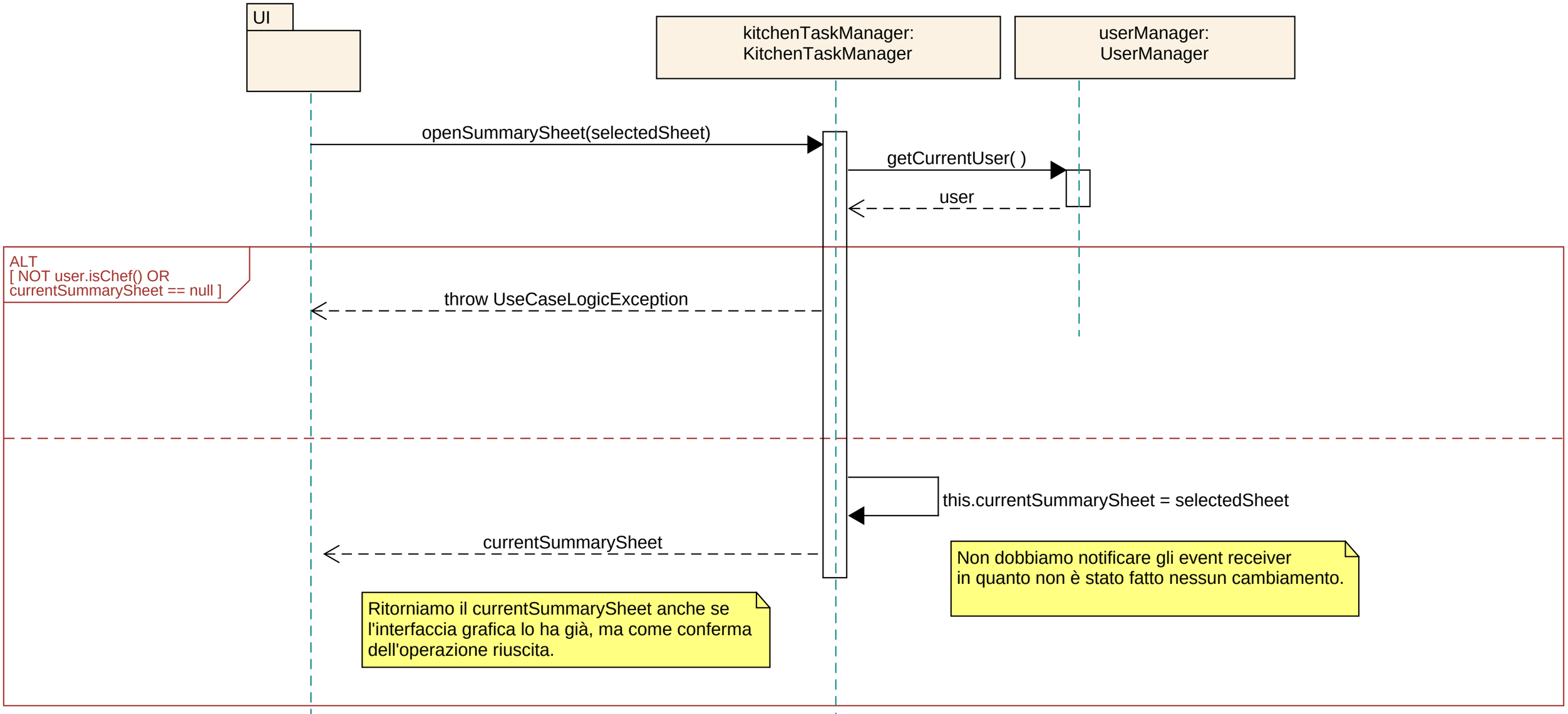
Design Class Diagram (DCD)

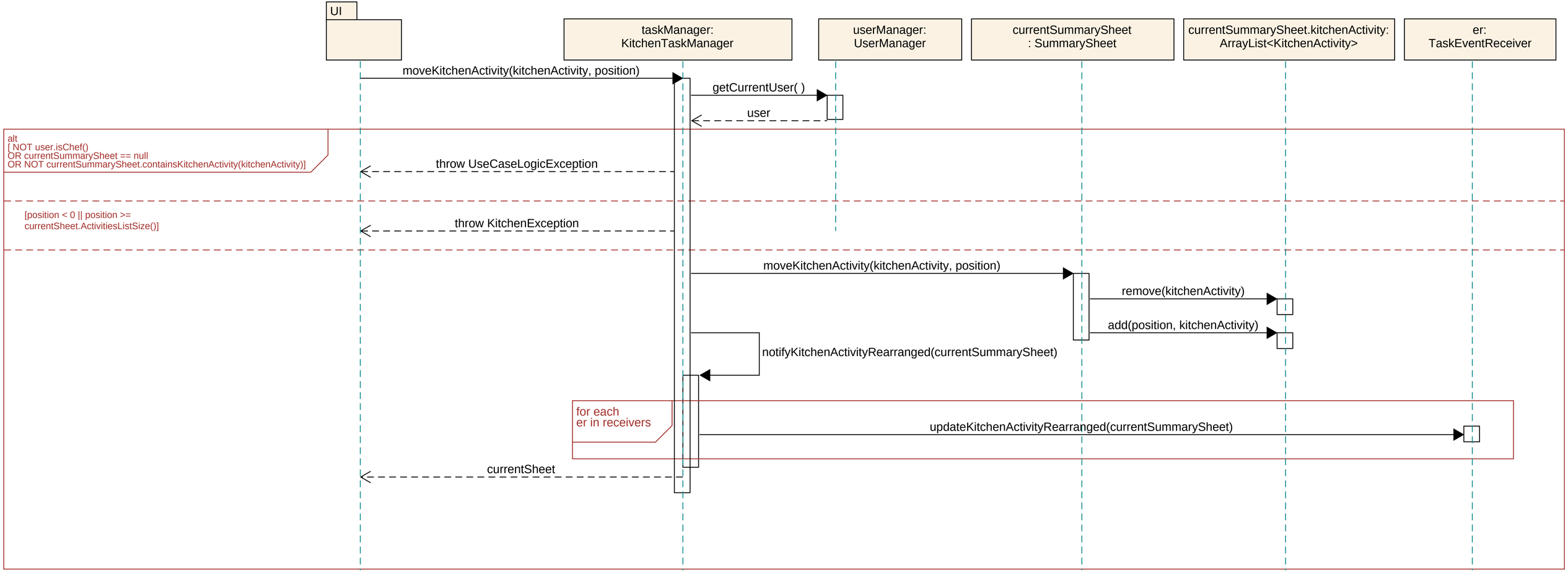


UI
taskManager: KitchenTaskManager
userManager: UserManager
service: Service
menuItem.itemReceipe: Preparation
er: TaskEventReceiver
generateSummarySheet(event, serv)
getCurrentUser( )
user
ALT
[(NOT user.isChef())]
throw UseCaseLogicException
[(NOT event.containsService(serv)) OR (service.getMenu() == null) OR !(service.isConfirmed())]
throw KitchenException
createSummarySheet()
create(this.getMenu())
summarySheet: SummarySheet
create()
activities: List<KitchenActivity>
create()
menuItems: List<MenuItem>
LOOP
for each sec
in menu.sections
LOOP
for each menuItem
in sec.sectionItems
add(menuItem)
LOOP
for each menuItem
in menu.freeItems
add(menuItem)
LOOP
for each menuItem
in menuItems
create(menuItem.itemReceipe, false)
activity: KitchenActivity
this.kitchenDuty = kitchenDuty
this.outOfMenu = false
create()
leftoversUsed: List<Leftover>
create()
tasks: List<KitchenTasks>
activity
add(activity)
getSubDuties()
subDuties
OPT
[subDuties.length > 0]
LOOP
for each subDuty
in subDuties
create(subDuty, false)
activity: KitchenActivity
this.kitchenDuty = kitchenDuty
this.outOfMenu = false
create()
leftoversUsed: List<Leftover>
create()
tasks: List<KitchenTasks>
activity
add(activity)
newSumSheet
this.sheet = newSumSheet
newSumSheet
this.currentSummarySheet = newSheet
notifySheetGenerated(currentSummarySheet)
for each
er in receivers
updateSheetGenerated(currentSummarySheet)
newSumSheet


UI
kitchenTaskManager: KitchenTaskManager
userManager: UserManager
currentSummarySheet: SummarySheet
currentSumSheet.activities: List<KitchenActivity>
kitchenDuty: KitchenDuty
er: TaskEventReceiver
addOutOfMenu(kitchenDuty)
getCurrentUser( )
user
ALT
[NOT user.isChef() OR currentSummarySheet == null]
throw UseCaseLogicException
addOutOfMenu(kitchenDuty)
create()
newActivities: List<KitchenActivity>
newActivities
create(kitchenDuty, true)
a: KitchenActivity
this.kitchenDuty = kitchenDuty
this.outOfMenu = outOfMenu
a
add(a)
getSubDuties()
subDuties
OPT
[subDuties.length > 0]
LOOP
for each subDuty in subDuties
create(subDuty, true)
a: KitchenActivity
this.kitchenDuty = kitchenDuty
this.outOfMenu = outOfMenu
a
add(a)
addAll(newActivities)
newActivities
notifyKitchenActivityOutOfMenuAdded(currentSummaySheet, newActivities)
LOOP
for each er in receivers
updateSheetKitchenActivityOutOfMenuAdded(currentSummarySheet, newActivities)
newActivities


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
currentSumSheet:
SummarySheet
currentSummarySheet.activities:
List<KitchenActivity>
er:
TaskEventReceiver
removeKictchenActivity(kitchenActivity)
getCurrentUser( )
user
ALT
[NOT user.isChef() OR
currentSummarySheet == null]
throw UseCaseLogicException
removeKictchenActivity(activity)
remove(activity)
In Java abbiamo il Garbage Collector per cui
non è necessario eliminare manualmente
tutte le istanze di KitchenTask associate
all'activity.
notifyKitchenActivityRemoved(currentSummarySheet, kitchenActivity)
LOOP
for each er
in receivers
updateKitchenActivityRemoved(currentSummarySheet, kitchenActivity)


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
openSummarySheet(selectedSheet)
getCurrentUser( )
user
ALT
[NOT user.isChef() OR
currentSummarySheet == null]
throw UseCaseLogicException
this.currentSummarySheet = selectedSheet
currentSummarySheet
Non dobbiamo notificare gli event receiver
in quanto non è stato fatto nessun cambiamento.
Ritorniamo il currentSummarySheet anche se
l'interfaccia grafica lo ha già, ma come conferma
dell'operazione riuscita.


UI
taskManager:
KitchenTaskManager
userManager:
UserManager
currentSummarySheet
: SummarySheet
currentSummarySheet.kitchenActivity:
ArrayList<KitchenActivity>
er:
TaskEventReceiver
moveKitchenActivity(kitchenActivity, position)
getCurrentUser( )
user
alt
[NOT user.isChef()
OR currentSummarySheet == null
OR NOT currentSummarySheet.containsKitchenActivity(kitchenActivity)]
throw UseCaseLogicException
[position < 0 || position >=
currentSheet.ActivitiesListSize()]
throw KitchenException
moveKitchenActivity(kitchenActivity, position)
remove(kitchenActivity)
add(position, kitchenActivity)
notifyKitchenActivityRearranged(currentSummarySheet)
for each
er in receivers
updateKitchenActivityRearranged(currentSummarySheet)
currentSheet

UI

taskManager:
KitchenTaskManager

userManager:
UserManager

kitchenActivity: KitcenActivity

er:
TaskEventReceiver

indicateEstimate(kitckenActivity, amount, portions, minutesEstimated)

getCurrentUser( )

user

ALT
NOT user.isChef()
OR currentSummarySheet == null]

throw UseCaseLogicException

setData(amount, portions, minutesEstimated)

this.minutesEstimated = minutesEstimated

OPT
amount != null

this.amount = amount

OPT
[portions != null]

this.portions = portions

kitchenActivity

notifyKitchenActivityModified(kitchenActivity)

for each
er in receivers

updateKitchenActivityModified(kitchenActivity)

kitchenActivity

UI

taskManager:
KitchenTaskManager

userManager:
UserManager

kitchenActivity:
KitcenActivity

er:
TaskEventReceiver

changeData(kitckenActivity, amount, portion, minutesEstimated)

getCurrentUser( )

user

ALT
[NOT user.isChef()
OR currentSheet == null]

throw UseCaseLogicException

changeData(amount, portion, minutesEstimated)

OPT
[minutesEstimated != null]

this.minutesEstimated = minutesEstimated

OPT
[amount != null]

this.amount = amount

OPT
[portion != null]

this.portion = portion

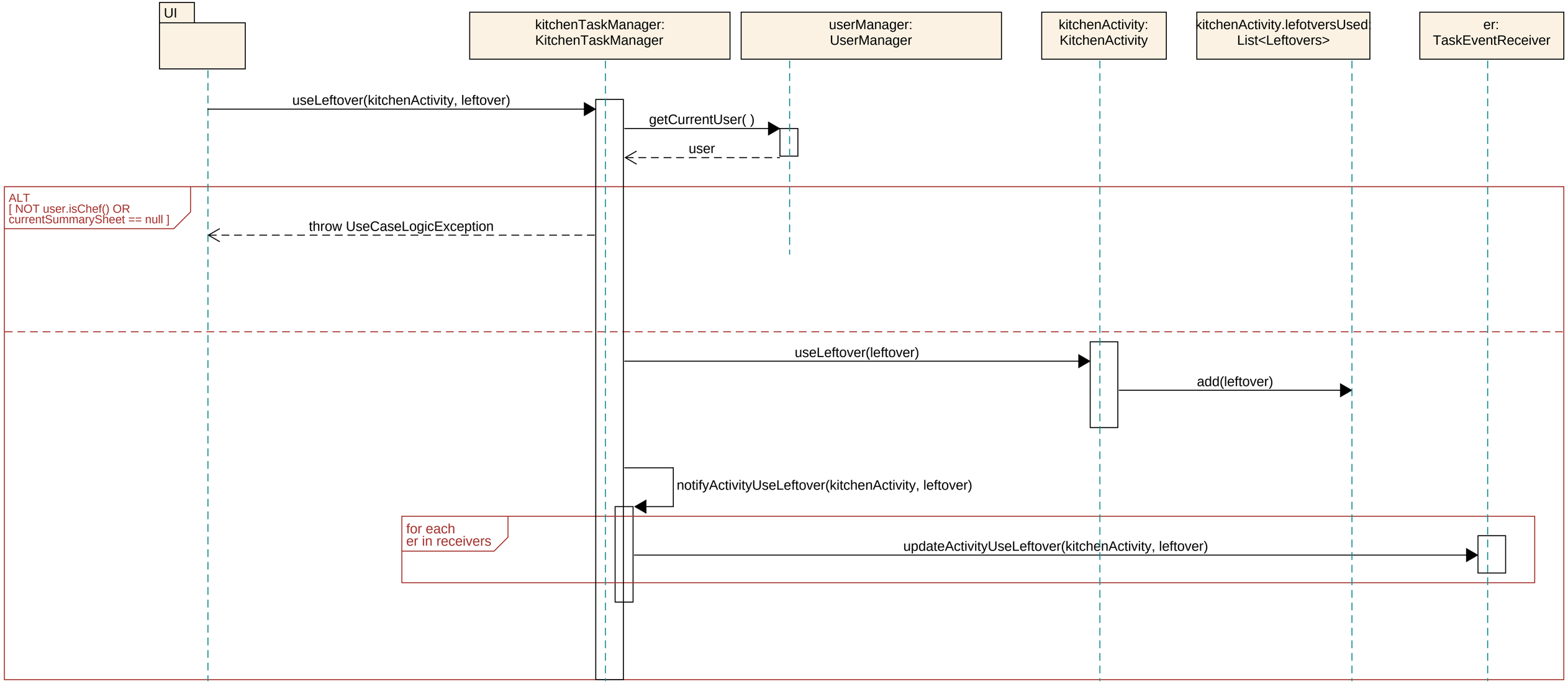
kitchenActivity

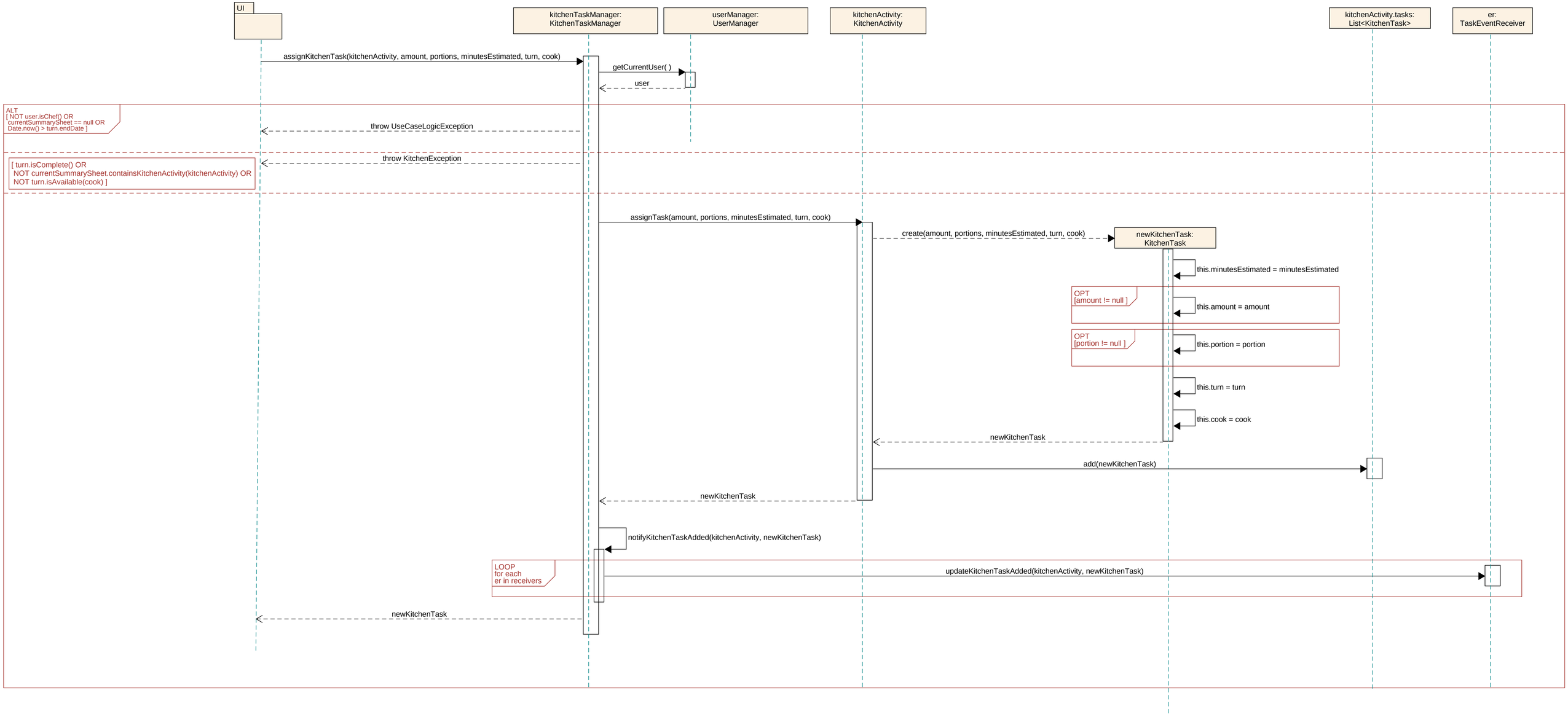
notifyActivityModified(kitchenActivity)

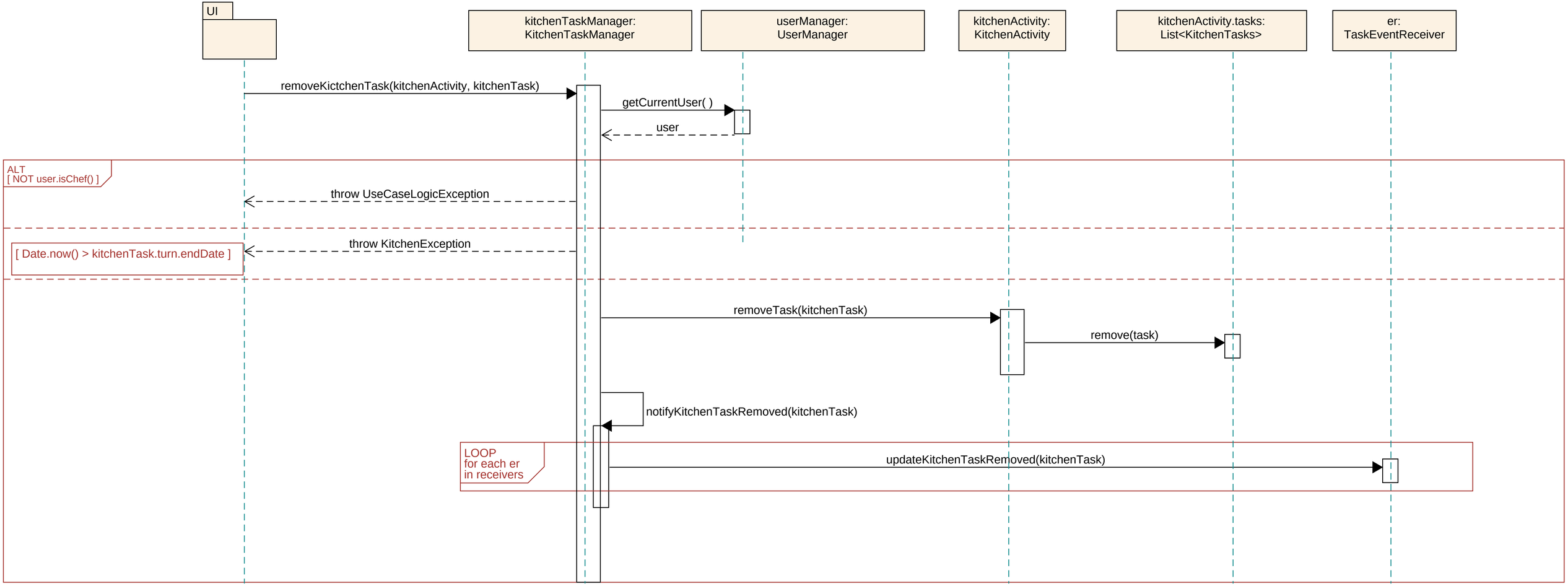
LOOP
for each
er in receivers

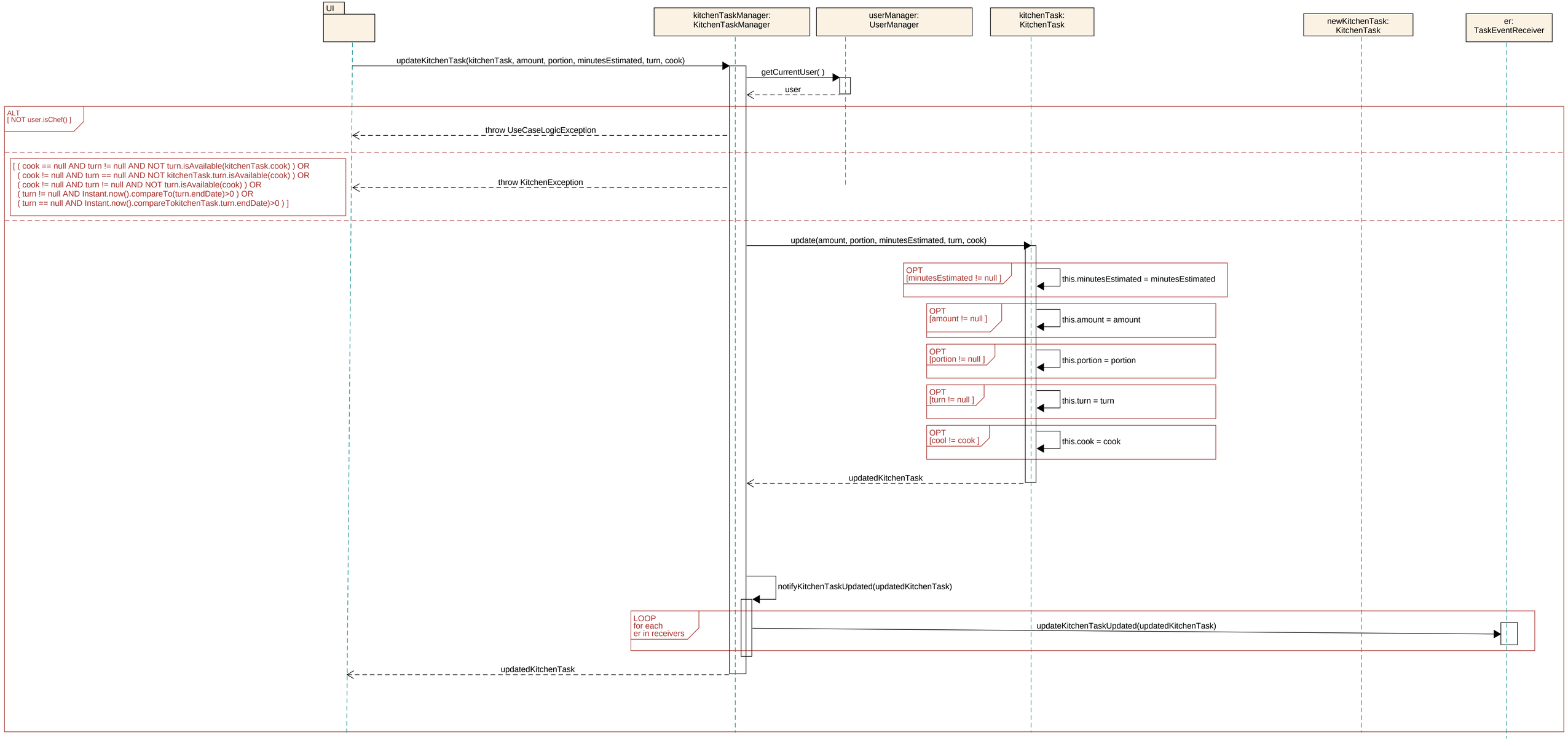
updateKitchenActivityModified(kitchenActivity)

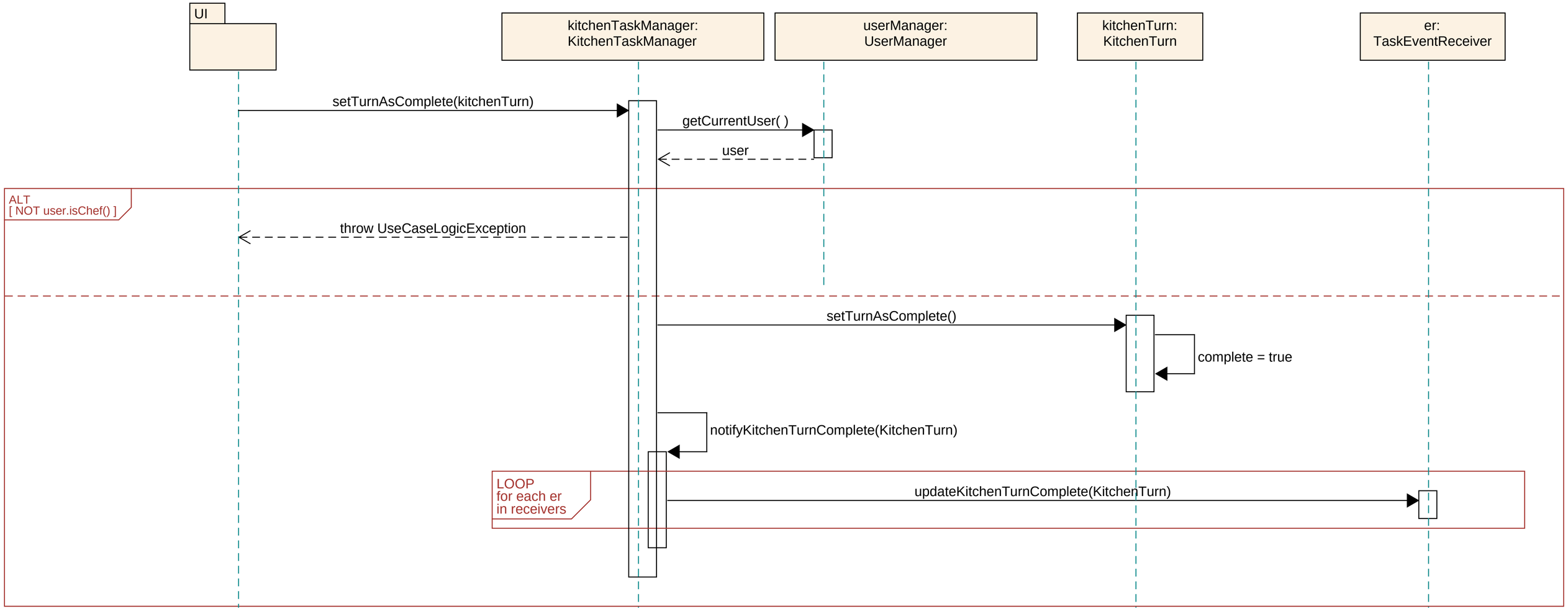
kitchenActivity


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
kitchenActivity:
KitchenActivity
kitchenActivity.lefotversUsed
List<Leftovers>
er:
TaskEventReceiver
useLeftover(kitchenActivity, leftover)
getCurrentUser( )
user
ALT
[NOT user.isChef() OR
currentSummarySheet == null]
throw UseCaseLogicException
useLeftover(leftover)
add(leftover)
notifyActivityUseLeftover(kitchenActivity, leftover)
for each
er in receivers
updateActivityUseLeftover(kitchenActivity, leftover)


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
kitchenActivity:
KitchenActivity
kitchenActivity.tasks:
List<KitchenTask>
er:
TaskEventReceiver
assignKitchenTask(kitchenActivity, amount, portions, minutesEstimated, turn, cook)
getCurrentUser( )
user
ALT
[NOT user.isChef() OR
currentSummarySheet == null OR
Date.now() > turn.endDate]
throw UseCaseLogicException
[turn.isComplete() OR
NOT currentSummarySheet.containsKitchenActivity(kitchenActivity) OR
NOT turn.isAvailable(cook)]
throw KitchenException
assignTask(amount, portions, minutesEstimated, turn, cook)
create(amount, portions, minutesEstimated, turn, cook)
newKitchenTask:
KitchenTask
this.minutesEstimated = minutesEstimated
OPT
[amount != null]
this.amount = amount
OPT
[portion != null]
this.portion = portion
this.turn = turn
this.cook = cook
newKitchenTask
add(newKitchenTask)
newKitchenTask
notifyKitchenTaskAdded(kitchenActivity, newKitchenTask)
LOOP
for each
er in receivers
updateKitchenTaskAdded(kitchenActivity, newKitchenTask)
newKitchenTask


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
kitchenActivity:
KitchenActivity
kitchenActivity.tasks:
List<KitchenTasks>
er:
TaskEventReceiver
removeKictchenTask(kitchenActivity, kitchenTask)
getCurrentUser( )
user
ALT
[NOT user.isChef()]
throw UseCaseLogicException
[Date.now() > kitchenTask.turn.endDate]
throw KitchenException
removeTask(kitchenTask)
remove(task)
notifyKitchenTaskRemoved(kitchenTask)
LOOP
for each er
in receivers
updateKitchenTaskRemoved(kitchenTask)


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
kitchenTask:
KitchenTask
newKitchenTask:
KitchenTask
er:
TaskEventReceiver
updateKitchenTask(kitchenTask, amount, portion, minutesEstimated, turn, cook)
getCurrentUser( )
user
ALT
[NOT user.isChef()]
throw UseCaseLogicException
[(cook == null AND turn != null AND NOT turn.isAvailable(kitchenTask.cook)) OR
(cook != null AND turn == null AND NOT kitchenTask.turn.isAvailable(cook)) OR
(cook != null AND turn != null AND NOT turn.isAvailable(cook)) OR
(turn != null AND Instant.now().compareTo(turn.endDate)>0) OR
(turn == null AND Instant.now().compareTokitchenTask.turn.endDate)>0)]
throw KitchenException
update(amount, portion, minutesEstimated, turn, cook)
OPT
[minutesEstimated != null]
this.minutesEstimated = minutesEstimated
OPT
[amount != null]
this.amount = amount
OPT
[portion != null]
this.portion = portion
OPT
[turn != null]
this.turn = turn
OPT
[cool != cook]
this.cook = cook
updatedKitchenTask
notifyKitchenTaskUpdated(updatedKitchenTask)
LOOP
for each
er in receivers
updateKitchenTaskUpdated(updatedKitchenTask)
updatedKitchenTask


UI
kitchenTaskManager:
KitchenTaskManager
userManager:
UserManager
kitchenTurn:
KitchenTurn
er:
TaskEventReceiver
setTurnAsComplete(kitchenTurn)
getCurrentUser( )
user
ALT
[NOT user.isChef()]
throw UseCaseLogicException
setTurnAsComplete()
complete = true
notifyKitchenTurnComplete(KitchenTurn)
LOOP
for each er
in receivers
updateKitchenTurnComplete(KitchenTurn)