CU03-Transférer un séjour de chambre

```
@startuml
left to right direction
package "Application de gestion hotelière" {
 usecase "CU01–Noter une reservartion d'une seule chambre" as N
 usecase "CU02–Noter une reservartion de plusieurs séjour" as S
 usecase "CU03–Transférer un séjour de chambre" as T

}
Commis --> N
Commis --> S
Commis --> T

@enduml
```

CU03-Transférer un séjour de chambre

Précondition(s) :

- Le commis est authentifié.

Postcondition(s) :

- Le séjour se poursuit dans une autre chambre.

Acteur principal : Le commis à la réception

Scénario principal

1. Un client désire occuper une chambre différente pour le reste de son séjour.
2. Le commis entre le numéro de chambre dans laquelle le client séjourne.
3. Le système affiche le séjour ainsi que toutes les chambres disponibles de la même catégorie.
4. Le commis informe le client des chambres disponibles puis sélectionne celle que le client préfère.
5. Le système modifie la ligne courante du séjour et inscrit une nouvelle ligne au séjour pour la nouvelle chambre.
6. Le commis remet la clé de la chambre au client et lui indique l'emplacement de la chambre dans l'hôtel.

Scénarios alternatifs 3a. Il ne reste plus de chambres disponibles dans la catégorie réservée. Le système affiche les chambres disponibles pour toutes les catégories. Retour à 4

MDD CU01 + CU02 + CU03

LigneRéservation correspond à Sejour. Il faut aussi modifier les multiplicité entre Séjour et Chambre pour 1 – 1.

```
@startuml
skinparam style strictuml
hide methods
class Commis
```

```
class Client {
  nom
telephone
}
class Hotel{
telephone
}
class Reservation {
confirmation:Strgin
}
class Sejour {
dateArrive: DateHeure
dateDepart: DateHeure
}
class Chambre {
no
}
class Clé
class Categorie {
nom: String
}
Commis "1" -- "*" Reservation : Effectue
Client "1" -- "*" Reservation : Demande
Reservation "1" -- "*" Sejour : Contient
Sejour "1" -- "*" Chambre : Reserve
Chambre "*" -- "1" Categorie : sont-décrites-par
Hotel "1" -- "*" Commis : Emploie
Hotel "1" -- "*" Chambre : Contient
Chambre "1" -- "*" Clé : est-ouverte*par
Chambre "*" -- "1" Emplacement : sont-situé-à
@enduml
```

DSS CU03-Transférer un séjour de chambre

```
@startuml
skinparam style strictuml
Actor ":Commis" as C
participant ":System" as S

C->S: demarrerTransferChanbre(string noChambre)
C<<--S: information du séjour, chambres disponibles

C->S: transfererChambre(string noChambreActuel, string noNouvelleChambre)
C<<--S: confirmation

@enduml
```

Contrat CU03-demarrerTransfert

Opération: demarrerTransfer(numeroChambre) **Présconditions: Postconditions:**

- Aucune

Contrat CU03-transférerChambre

Opération: transférerChambre(noChambreActuel, noNouvelleChambre) **Présconditions: Postconditions:**

- Une instance sn:Sejour a été créée
- sn.dateArrive est devenu maintenant
- sn.dateDepart est devenu Chambre.sejour.dateDepart sur la base de correspondance avec noChambreActuel
- Une association a été créée entre sn et Chambre.sejour.réservation sur la base de correspondance avec noChambreActuel
- Une association a été créée entre sn et Chambre sur la base de correspondance avec noNouvelleChambre
- Chambre.Sejour.dateDepart et devenu maintenant sur la base de correspondance avec noChambreActuel

RDCU CU03-demarrerTransfert

```
@startuml
skinparam style strictuml
title DémarrerRéservation

participant ":SystemeReservation" as sr
participant ":Reservation" as r

note left of sr : Controleur de facade\n (système)
->sr : demarrerReservation()
activate sr

note right of sr : Par créateur
create r
sr-->r : create
activate r

participant "llr:List<LigneReservation>" as llr

note right of r: Par créateur
create llr
r --> llr: create

participant "lr:List<Reservation>" as lr

note right of sr : Expert en\n information
sr --> lr: add(r:reservation)
deactivate sr
deactivate r
@enduml
```

RDCU CU03-transférerChambre

```
@startuml
skinparam style strictuml
title - réserverCatégorie(arrivée : date, départ : date, nom : string,
quantité:int)

participant ":SystemeReservation" as sr
participant ":Reservation" as r
participant "lr:LigneReservation" as lr
participant ":CatalogueCategorie" as cc
participant "lc:List<Categorie>" as lc
participant "llr:List<ligneResevation>" as llr

note left of sr : Controleur de facade\n (système)
->sr : réserverCatégorie(arrivée : date, départ : date, nom : string,
quantité:int)

activate sr
note right of sr: expert en information
sr -> cc : c = getCategorie(nom:String)
note right of cc : expert en information\nCohésion
cc -> lc : c=get(nom:String)

note right of sr : Par expert en information
sr -> r : lr = addReservation(arrivée : date,\n départ : date,\n
quantité:int,c:Categorie)

create lr
note right of r: par createur
r--> lr : lr = create(arrivée : date,\n départ : date,\n
quantité:int,c:Categorie)

note right of r : Par expert en information
r->llr : add(lr:LigneReservation)
deactivate sr
@enduml
```