

Exercice 5.1 : Cas Gestion de salles

Le cas d'étude concerne la gestion de locaux communaux. La commune de Coulos gère des salles de réunion et de spectacles. Elle souhaite automatiser les réservations de ses salles. Les salles font partie des bâtiments municipaux. Un bâtiment se trouve à une adresse donnée (numéro de rue, rue, code postal, ville). Il contient un certain nombre de salles. Chaque salle est caractérisée par un numéro, affiché au dessus de la porte et formé du numéro d'étage et du numéro de la salle dans l'étage, d'une superficie. Certaines salles possèdent du matériel (tableau, rétro-projecteur, moyens vidéo). D'autres équipements sont possibles et sont gérés à la demande par les services techniques de la mairie. Certaines salles comprennent une cuisine attenante munie de vaisselle. Les bâtiments sont tous équipés de commodités.

Les tarifs de location varient en fonction de la superficie, du type de salle demandé (cuisine, vaisselle uniquement, vidéo, tableau, salle amphithéâtre), de la durée (à la demi-journée ou à la soirée), de l'origine du demandeur (résident/non résident), de titre du demandeur (particulier, association, entreprise) et du type de manifestation (réunion, banquet, spectacle). On mémorise l'identité du demandeur.

Le système doit permettre d'effectuer les réservations et les annulations de salle, de calculer des tarifs de location, de fournir des plannings d'occupation, des factures hebdomadaires par demandeur, des taux d'occupation de salles.

Modéliser ce cas avec UML en respectant les règles de bon sens (une salle ne peut être louée à deux demandeurs en même temps).

- Analyse des besoins
- Analyse

Solution 5.1

Ce cas d'étude est un exemple classique de modélisation de données. Il s'agit donc essentiellement de modéliser les objets métier (la base de données) et les interfaces d'accès. Il s'agit d'un cas similaire à l'exemple du club vidéo. La méthode utilisée est similaire à ceci près que nous construisons un modèle du domaine (*Domain Model*) avant l'étude des cas d'utilisation et des scénarios principaux. En effet, les traitements étant imprécis (implicites ?) dans l'énoncé, définir les objets entités (ou métier) est un point d'appui pour la description des utilisations du système. Après l'expression du besoin, nous analyserons le système.

a) Expression du besoin.

Modèle du domaine Le contexte du système peut être défini par un modèle du domaine [RJB99]. Il s'agit de construire un modèle des objets métier et de leurs relations, sur lequel on s'appuiera pour préciser le besoin dans les cas d'utilisation. Le modèle du domaine constitue une sorte de glossaire des termes du métier. Il ne s'agit pas d'une modélisation du métier¹ mais simplement d'une modélisation classique des données comme dans Merise. Nous adoptons une démarche *ad hoc*, appelé génération spontanée dans le chapitre 1 du tome 1.

Le texte met en évidence des bâtiments contenant des salles, des demandeurs, du matériel, et des réservations. Les salles disposent de matériels fixes. La réservation porte sur une salle, pour une date et une durée donnée. Elle est faite par un demandeur.

Le tarif de réservation est fonction des caractéristiques de la salle, de l'origine et du titre du demandeur, du type de la salle, de la manifestation, de la durée, et du matériel loué. Nous avons choisi de représenter les différents composants du tarif de réservation dans les classes concernées. Pour cela, les tables de tarifs sont définies par une classe abstraite TableTarif, qui donne la structure générale d'une table : un code, un libellé et un tarif.

1. Processus d'ingénierie du métier au sens de Kettani [KMPRS98] ou *business model* au sens du processus unifié [EP00, RJB99].

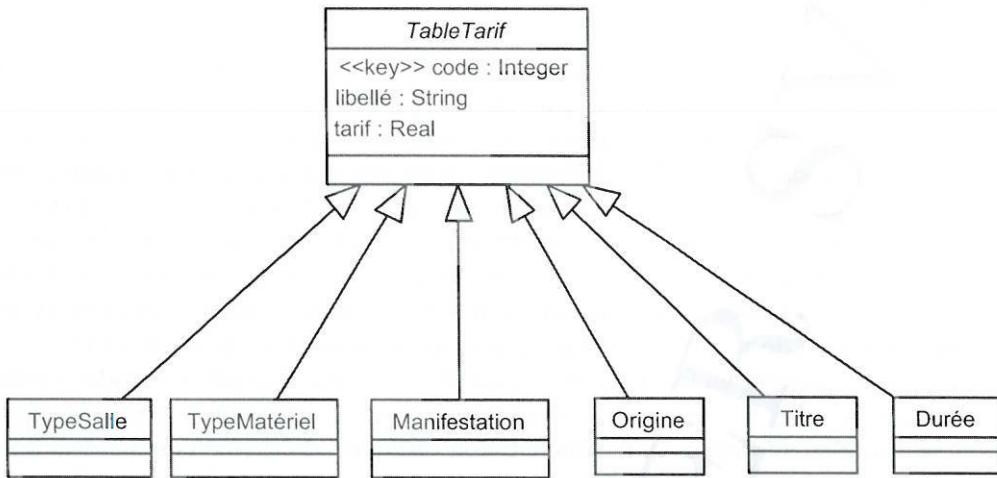


Figure 184 : Diagramme de classes, abstraction des tables de tarifs - Salles

La figure 185 représente le diagramme des classes métiers. Nous n'avons pas noté les rôles ou les noms d'associations lorsqu'il n'y a pas d'ambiguïté pour la navigation. S'agissant d'une modélisation de données, des clés candidates sont mises en évidence, pour l'accès aux informations (recherche par clé).

La réservation peut être représentée par une association n-aire. Mais une classe s'impose du fait de l'association optionnelle vers le matériel (*mobile*). Le montant est un attribut calculé en fonction des différents tarifs, selon une formule à déterminer.

La classe Adresse factorise le stockage des adresses lorsque plusieurs demandeurs ou plusieurs bâtiments ont même adresse. Elle peut être supprimée et remplacée par ses propriétés dans les classes Bâtiment et Demandeur. Noter que définir ces classes par héritage de la classe Adresse (au lieu d'une association) n'a aucun sens.

L'identifiant de la salle est relatif au bâtiment. Le type de matériel contient la description du matériel et son tarif. Le matériel est l'objet "réel", identifié par son numéro d'inventaire. Un matériel fixe n'est pas mobile et vice-versa, c'est une contrainte d'exclusion car le cardinal maximum des ensembles est de 1, définie en OCL par :

```

context Matériel
  self.matérielFixe.isEmpty xor self.matérielMobile.isEmpty
  
```

Les invariants portant sur les clés candidates sont à définir en OCL. Nous présentons uniquement le cas de la salle.

```

context Salle
  inv identifiant :
    Salle.allInstances->forAll(s1, s2 |
      s1 <> s2 implies (s1.no_étage <> s2.no_étage or
                           s1.no_salle <> s2.no_salle or s1.no_bat <> s2.no_bat))
  
```

Ce modèle du domaine peut être complété par une modélisation du métier (*Business Model*) qui précise les processus métier [EP00, RJB99] et par la description des besoins non fonctionnels (contraintes diverses relatives aux aspects interfaces, physiques, de conception ou légaux). Pour la suite, on s'appuiera sur le modèle du domaine pour trouver les traitements (les utilisations) implicites du système d'information.

Cas d'utilisation On suppose deux acteurs : un demandeur et un administrateur. Le demandeur apparaît explicitement dans l'énoncé, il demande ou annule des réservations. L'administrateur est un rôle générique que nous donnons aux personnes qui gèrent la base, modifient

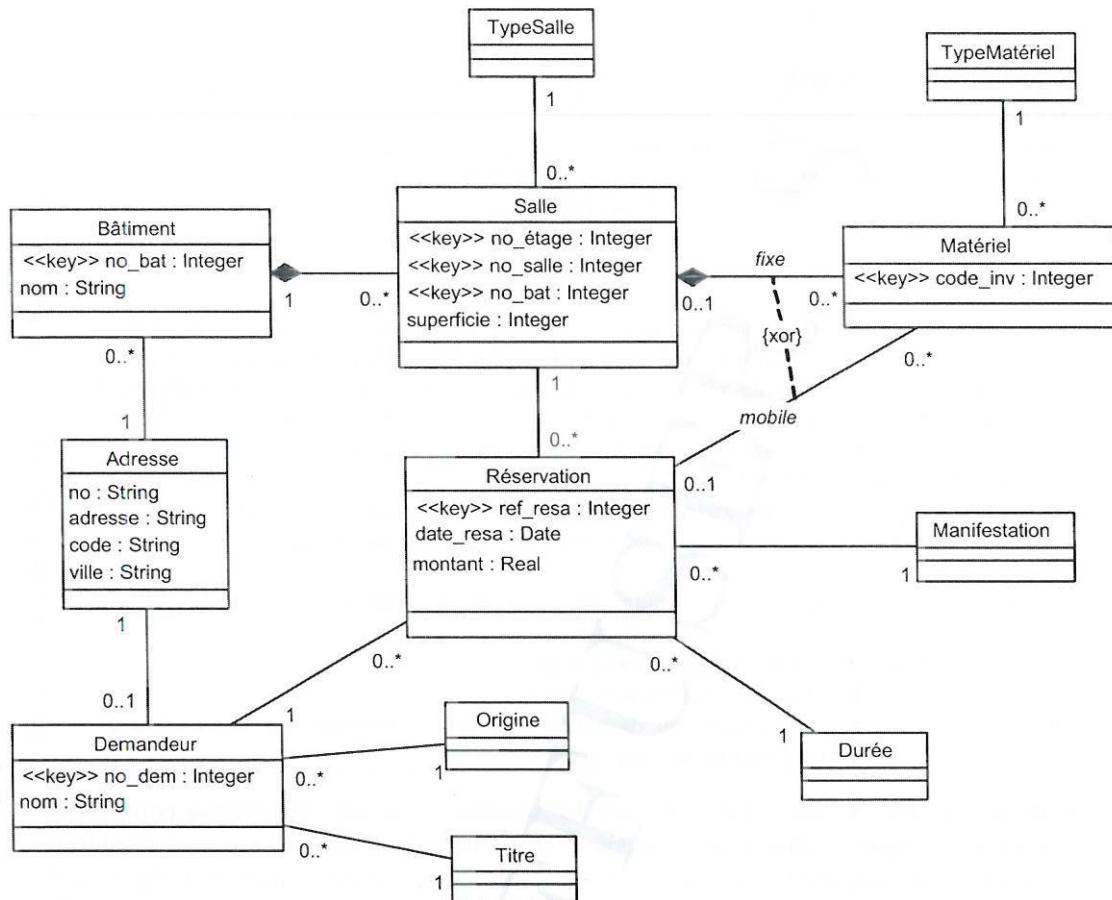


Figure 185 : Diagramme de classes du domaine - Salles

les données et réalisent les traitements généraux. Ces deux acteurs induisent deux cas d'utilisation : Réservations et Administration. La gestion des informations sur les demandeurs fait partie de l'administration, mais comme elle peut être invoquée dans l'UC de réservation, on la représente par un UC à part. Une première version du diagramme de cas d'utilisation est alors la suivante.

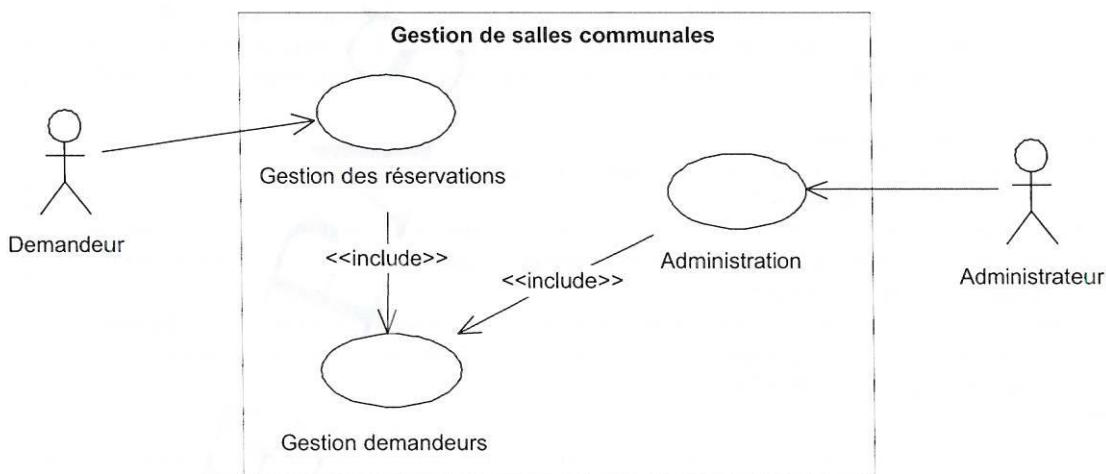


Figure 186 : Cas d'utilisation, version préliminaire - Salles

Détaillons la gestion des réservations.

Cas d'utilisation : Gestion des réservations	
acteurs primaires :	Demandeur, Administrateur
invariant :	Unicité de réservation
	Une salle n'est pas réservée pour deux demandeurs différents au même moment.
	description
	La gestion des réservations comprend la réservation des salles, la consultation des réservations, l'annulation des réservations.
cas : Réservation	
	Les éléments de la réservations sont saisis et recherchés dans la base en fonction de critères donnés : salle, demandeur, matériel, durée, manifestation, date. À tout moment, il est possible de consulter le planning des réservations en cours. Si tous les éléments sont corrects et qu'il n'y a pas de conflit de réservation, le montant est calculé et la réservation confirmée. Le numéro de la réservation est fourni par le système au demandeur.
cas : Consultation des réservations	
	La consultation prend plusieurs formes : recherche d'une réservation par son numéro, par demandeur, par date ou par salle, consultation du planning des réservations.
cas : Annulation d'une réservation	
	Après recherche de la réservation, le demandeur confirme sa suppression.
	exceptions
cas : Réservation avec un demandeur inexistant	
précondition :	Le demandeur n'est pas inscrit.
résultat :	Il y a création du demandeur (voir UC Gestion des demandeurs) avant d'établir la réservation.

Lorsqu'on souhaite décrire le cas d'utilisation Administration, on découvre sa complexité. Pour la gérer, on souhaite organiser plus finement cette utilisation du système, en considérant différents aspects de l'administration des locaux communaux, comme le montre la figure 187. Nous choisissons cette seconde version car elle offre une meilleure structuration pour les scénarios.

Détaillons l'utilisation relative à la gestion des locaux.

Cas d'utilisation : Gestion des locaux (partie 1)	
acteurs primaires :	Administrateur
	description
	La gestion des locaux est l'ensemble des traitements de la base de données relatifs aux bâtiments, aux salles et à leur type : saisie, mise-à-jour, consultation, suppression.
cas : Ajout d'une salle	
	Le numéro du bâtiment est saisi ou choisi dans une liste de bâtiments existant. Si le bâtiment n'existe pas, on doit d'abord le créer [†] . Le type est ensuite donné (numero) ou choisi dans une liste. Si le type n'existe pas, on doit d'abord le créer [†] . Lorsque le bâtiment et le type sont corrects, le numéro de la salle et de l'étage sont saisis, ainsi que la superficie. Une nouvelle salle est créée.
cas : Recherche d'une salle	
	La recherche se fait par bâtiment, par numéro d'étage ou de salle ou par type.
cas : Mise à jour du matériel	
	Après recherche de la salle, on peut lui ajouter/modifier/supprimer du matériel fixe, en choisissant ce matériel dans une liste. Si le matériel n'existe pas, on doit d'abord le créer [†] .
cas : Suppression d'une salle	
	Après recherche de la salle, l'occurrence trouvée est doit être supprimée. La suppression d'une salle entraîne la suppression de ses réservations. Avant de supprimer la salle, il faut confirmer la suppression des réservations. [‡]
cas : ...	

[†] Deux alternatives sont possibles : le traitement en cours (A) est soit interrompu soit suspendu pour créer l'élément manquant (B). Dans le premier cas, l'existence de l'élément pouvait être mise en précondition du cas. Dans le second cas, à la fin de l'invocation du cas d'utilisation de B, on doit reprendre le cas d'utilisation de A, au point où il en était. La représentation adaptée est le cas d'exception.

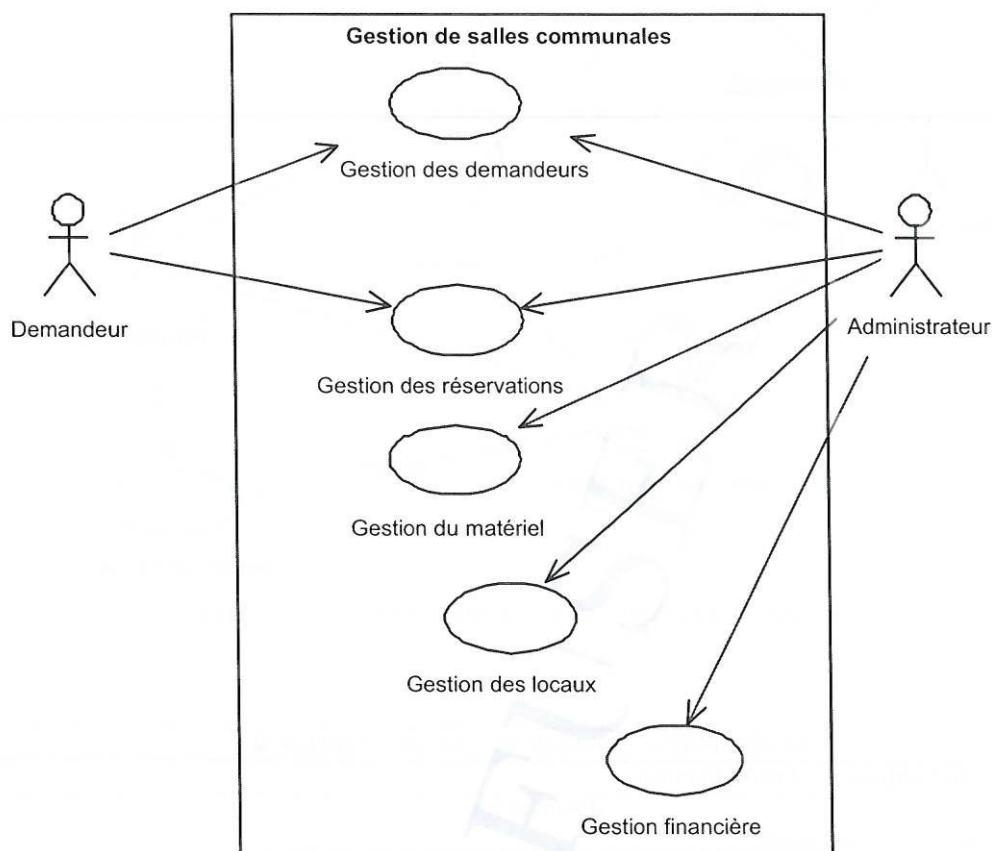


Figure 187 : Cas d'utilisation - Salles

[‡] Remarque similaire à [†] pour la suppression, conditionnée par l'inexistence d'un élément.

Le choix des alternatives peut être décidé au niveau de l'analyse des besoins ou dans la conception détaillée. Il peut aussi être représenté sous forme d'exceptions dans le formalisme des cas d'utilisation. Tout dépend du niveau de détails que souhaite introduire l'analyste du besoin.

Ce cas d'utilisation met aussi en évidence des relations de dépendance entre différents cas d'un cas d'utilisation ([†]) et entre cas d'utilisation ([‡]). Ce schéma de dépendance est aussi valable pour la gestion des bâtiments, des types, des durées, des demandeurs, des matériaux. Nous n'avons pas représenté toutes ces dépendances au niveau du diagramme de cas d'utilisation, sinon nous aurions obtenu un diagramme de flots de données (figure 188).

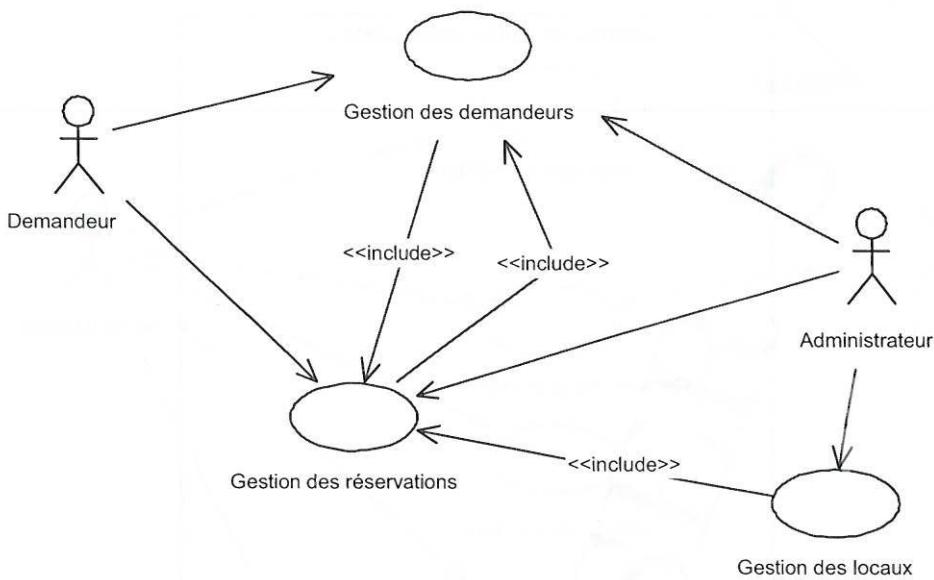


Figure 188 : Cas d'utilisation, version préliminaire - Salles

Cas d'utilisation : Gestion des locaux (partie 2)	
acteurs primaires :	Administrateur
	description
cas : Ajout d'un bâtiment	Le nom du bâtiment est saisi. L'adresse est soit choisie dans une liste, en fonction du code postal, soit entrée directement (numéro de rue, rue, code postal, ville).
cas : Recherche d'un bâtiment	La recherche se fait sur le nom, le numéro ou l'adresse du bâtiment.
cas : Modification d'un bâtiment	Après recherche du bâtiment, les informations du bâtiment sont modifiées.
cas : Suppression d'un bâtiment	Après recherche du bâtiment, l'occurrence trouvée est doit être supprimée. La suppression d'un bâtiment entraînant la suppression de ses salles, une confirmation est demandée.
cas : Ajoute/supression/consultation d'un type de salle	La description est similaire à celle des bâtiments.

Cas d'utilisation : Gestion des demandeurs	
acteurs primaires :	Demandeur, Administrateur
	description
	La gestion des demandeurs est l'ensemble des traitements de la base de données relatifs aux demandeurs, aux origines et aux titres : saisie, mise-à-jour, consultation, suppression. Le comportement est similaire à la gestion des locaux. Il n'est pas détaillé pour des raisons de taille du tome.

Cas d'utilisation : Gestion du matériel	
acteurs primaires :	Administrateur
	description
	La gestion des matériels est l'ensemble des traitements de la base de données relatifs aux matériels fixes ou mobiles : saisie, mise-à-jour, consultation, suppression. Le comportement est similaire à la gestion des locaux. Il n'est pas détaillé pour des raisons de place.

Cas d'utilisation : Gestion financière	
acteurs primaires :	Administrateur
	description
<p>La gestion financière concerne les aspects financiers : gestion des tarifs, facturation. Nous y plaçons aussi la gestion de diverses tables (durées, manifestations).</p> <p>Le comportement n'est pas détaillé pour des raisons de place.</p>	

Nous définissons aussi des contraintes fonctionnelles. *Les objets sont identifiés par leur clé candidate. Un matériel fixe n'est pas mobile. Une salle n'est pas réservée pour deux demandeurs différents au même moment.* La liste n'est pas exhaustive.

Scénarios Les scénarios sont rangés par cas d'utilisation. Pour des raisons pédagogiques, liées à l'explication de la structure du diagramme de cas d'utilisation, nous les présentons à part. Les scénarios sont relativement simples.

Dans le cadre d'un tel exercice, nous ne cherchons pas la modélisation complète et exhaustive du système mais uniquement la démarche utilisée et les cas représentatifs. C'est pourquoi nous illustrons uniquement les cas d'utilisation Gestion des locaux et Gestion des réservations par quelques scénarios. Les scénarios des autres cas sont très similaires.

Dans le scénario de l'ajout d'une salle, il n'est pas précisé ce qui se passe si le bâtiment ou le type n'existe pas : abandon, création ou les deux possibilités. La spécification est complétée soit par des scénarios d'exception soit en groupant différents scénarios avec des structures de contrôles (commentaires).

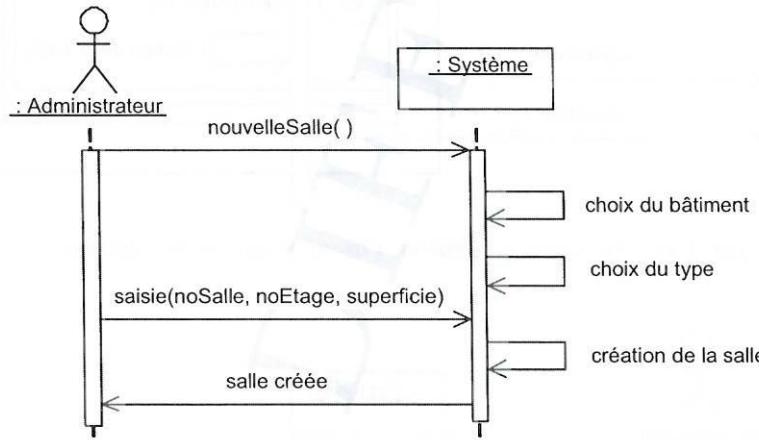


Figure 189 : Scénario de l'ajout d'une salle - Salles

La modification de salles commence par le choix de la salle puis le choix des caractéristiques à modifier. Elle se termine par une validation.

La suppression d'une salle implique la suppression de ses réservations, et la libération des matériels fixes, qui deviennent implicitement mobiles. Les types et bâtiments de la salle sont inchangés, sauf bien sûr leur relation avec la salle supprimée. Si des réservations sont en cours, une confirmation est demandée.

Les éditions (planning, taux d'occupation, listes des salles, etc) sont des scénarios triviaux : requête-réponse, il n'est pas utile de les détailler sous forme de scénarios. On précisera pour chaque édition les informations nécessaires et la partie du domaine accédée.

La gestion des réservations comprend l'ajout, la modification ou l'annulation de réservations. Lors de l'ajout, les différents éléments de la réservation sont fournis, ensuite, une vérification de cohérence est engagée avec le planning des réservations.

Si le demandeur n'existe pas, il s'agit d'une exception, on invoque le cas d'utilisation Gestion des demandeurs avant de reprendre le traitement en cours.

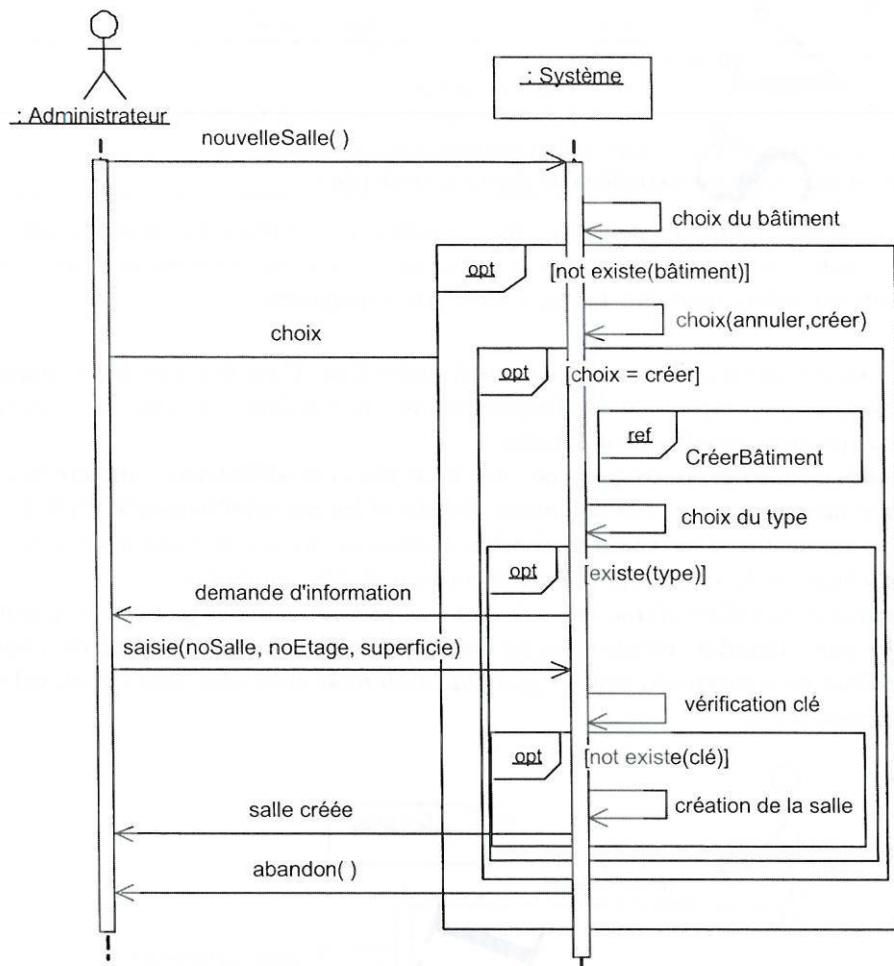


Figure 190 : Scénario général de l'ajout d'une salle - Salles

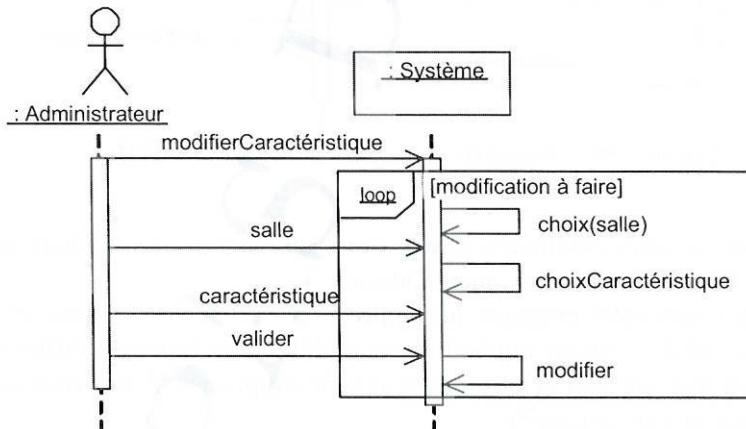


Figure 191 : Scénario de la modification d'une salle - Salles

L'annulation d'une réservation comprend la recherche de la réservation puis la demande de confirmation d'annulation. A l'issue de ces deux étapes, la réservation est annulée.

La description des besoins s'arrête ici, mais il faut bien noter qu'elle n'est pas complète. Dans un contexte de développement réel, le besoin doit être exprimé plus précisément, sans

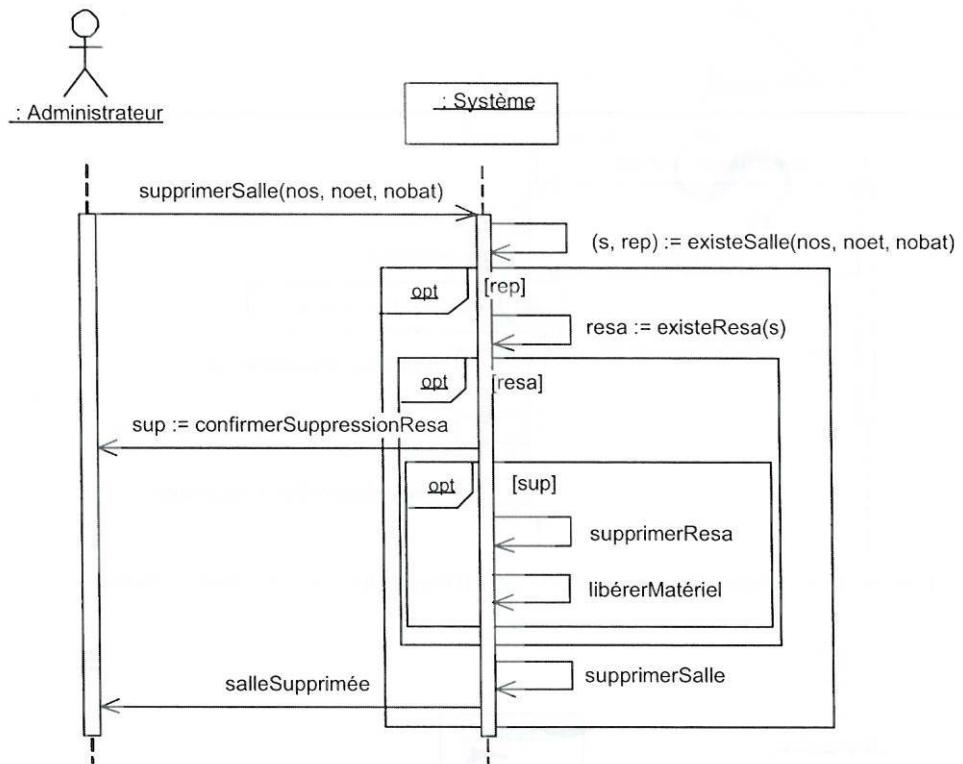


Figure 192 : Scénario général de la suppression d'une salle - Salles

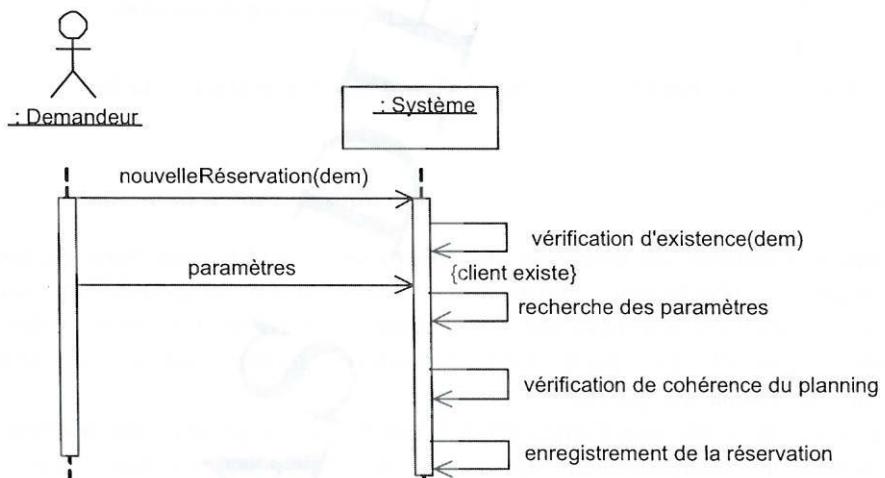


Figure 193 : Scénario normal de l'ajout d'une réservation - Salles

oublis ou sous-entendus. L'ensemble des cas d'utilisation et des scénarios doit être affiné et approfondi pour mettre à jour toutes les interrogations sur le modèle. Les choix "métier" doivent être fixés à ce niveau. Par exemple, si lors de l'ajout d'une réservation pour un demandeur inexistant on décide si le traitement est interrompu (arrêt) ou suspendu (le temps de faire appel à l'UC Gestion des demandeurs). Dès lors, il devient possible de faire valider le besoin par les "utilisateurs". Pour la suite, on suppose que le besoin est stable.

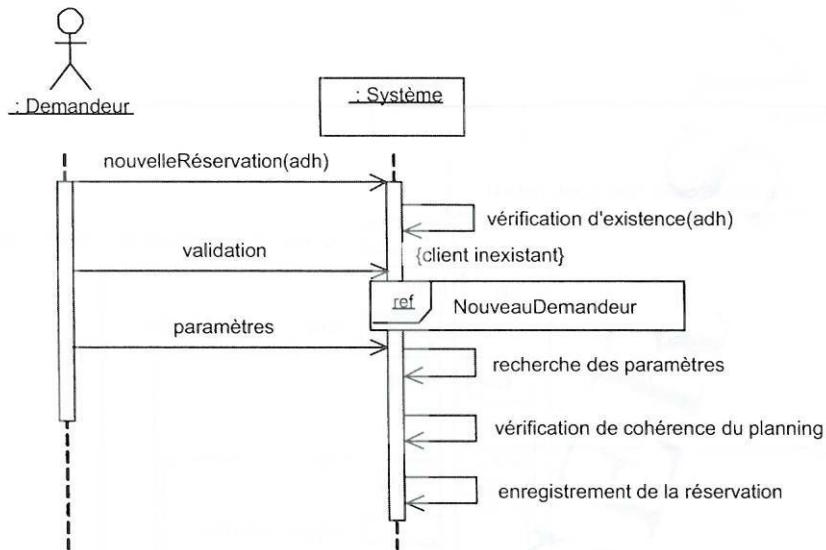


Figure 194 : Scénario d'exception de l'ajout d'une réservation - Salles

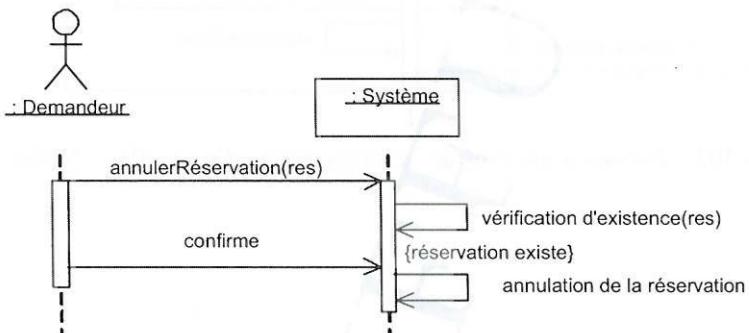


Figure 195 : Scénario de l'annulation d'une réservation - Salles

b) Analyse.

Dans la phase d'analyse, les scénarios précédents sont étendus pour faire apparaître les objets du système. Ces objets sont ensuite abstraits au niveau du diagramme de classes. On suppose l'existence d'objets interface, liés aux utilisations particulières : gestion des locaux, gestion des réservations, etc. Ces objets interface, notés sous forme de fenêtres, seront à affiner dans l'étape de conception du système.

Il est important de noter que l'application n'induit pas de parallélisme intrinsèque, les objets sont passifs. Le comportement dynamique des objets est standard : création-vie-suppression. Il n'y a donc pas lieu de modéliser le comportement dynamique par des diagrammes états-transitions. Les situations particulières sont modélisables simplement via les précondition des opérations. De ce fait, on n'utilise pas d'événements autres que les envois de messages et d'actions autres que les opérations. Cela facilite la validation (voir le chapitre 6).

Diagrammes d'interactions Les diagrammes d'interactions (communications ou séquences) sont des extensions des scénarios de la partie a. Nous utilisons le formalisme des séquences lorsque plusieurs cas sont regroupés dans un diagramme avec des "structures de contrôle" sous forme de commentaires et le formalisme des communications dans le cas contraire.

Pour ajouter une nouvelle salle (scénario de la figure 189), on choisit un bâtiment dans la liste des bâtiments, et un type dans la liste des types. On suppose donc que le bâtiment et

le type de la salle à ajouter existent. Dans le cas contraire, la procédure est abandonnée (cas d'exception). Après le bâtiment et le type, les informations propres à la salle sont fournies. On vérifie qu'il n'existe pas de salle ayant même identifiant. Si c'est le cas, on crée une nouvelle salle et ajoutée à l'ensemble des salles. Sinon la procédure est abandonnée.

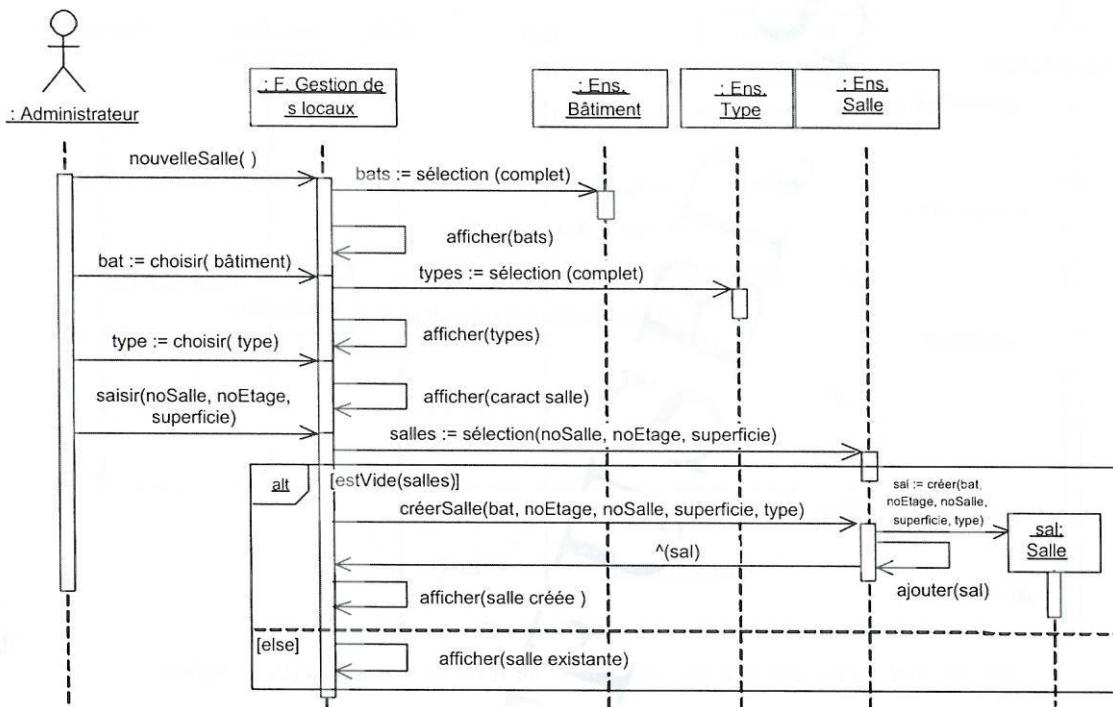


Figure 196 : Diagramme de séquences, ajout d'une salle - Salles

Dans cet exemple, nous avons mis en évidence la structure générale des interactions : une fenêtre d'interface qui dirige l'interaction, des ensembles d'objets métier et des objets métier. C'est la présentation adoptée pour le club vidéo dans le tome 2. D'autres formes d'organisation des instances des classes métier sont abordées dans l'exercice 7.7.

Pour que la description soit complète, on peut préciser la sémantique des opérations en OCL. La désignation de l'ensemble Ens. Classe des instances de la classe Classe est notée Classe.allInstances en OCL. Seule l'opération de création de salle nécessite une spécification, cette opération d'intanciation ajoute l'instance créée à l'ensemble des instances de la classe.

```

context Salle::créerSalle(bat, noEtage, noSalle, superficie, type) : Salle
pre: -- le bâtiment et la salle existent
    Bâtiment.allInstances->includes(bat) and Type.allInstances->includes(type)
post: -- soit sal l'objet créé
    let sal : Salle in
        result = sal and Salle.allInstances@pre->excludes(sal) and
            sal.no_étage = noEtage and sal.no_salle = noSalle and
            sal.no_bat = bat.no_bat and sal.superficie = superficie and
            sal.typeSalle = type and sal.bâtiment = bat and
            -- ajout explicite dans l'ensemble des instances
            Salle.allInstances = Salle.allInstances@pre->including(sal)

```

Les exceptions sont simplement une sortie du traitement si le bâtiment ou le type n'existent pas. Il s'agit d'une variante du diagramme de séquences de la figure 196.

La recherche d'une salle est une sélection dans l'ensemble des salles. La mise à jour du matériel d'une salle donnée consiste à sélectionner un ou plusieurs matériels disponibles (pas fixés dans une salle ou réservés) et à les affecter à la salle. Il s'agit d'une séquence simple.

La suppression d'une salle entraîne la suppression des réservations de la salle et la libération des matériels (scénario figure 192). L'abandon est une sortie de l'interaction.

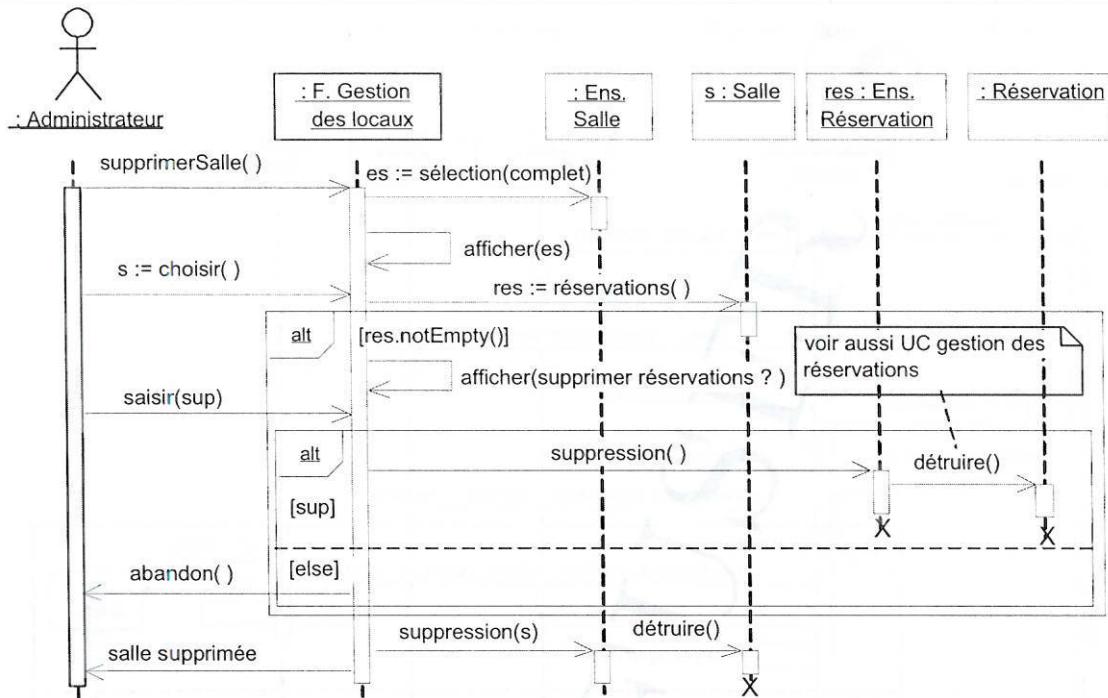


Figure 197 : Diagramme de séquences, suppression d'une salle - Salles

La gestion des demandeurs est représentée ici par une séquence d'ajout du demandeur (figure 198). Cette séquence est utilisée dans l'ajout d'une réservation.

La gestion des réservations comprend essentiellement la réservation et l'annulation. La séquence de réservation (figure 200) s'écrit dans le même esprit que pour la gestion des salles. On vérifie l'existence de l'adhérent. Dans la négative, il est ajouté (appel de la séquence nouvel adhérent). Ensuite, la date est entrée et la durée choisie. La recherche de la salle se fait en fonction du type de la salle et de sa disponibilité. On choisit les différents éléments de la réservation dans les ensembles existants. Nous avons mis en évidence deux situations : la salle est choisie ou pas. Lorsque tous les éléments sont valides et la salle choisie, la réservation est effective. Si la salle n'est pas choisie, la réservation est abandonnée. En fait, ce scénario d'abandon est possible à chaque choix, mais nous ne l'avons pas noté pour ne pas surcharger le schéma. On pourra définir ces alternatives dans d'autres diagrammes de séquences. L'objectif de cet exercice n'est pas de définir exhaustivement les collaborations mais d'en donner les exemples représentatifs. L'annulation d'une réservation consiste simplement à rechercher la réservation puis à l'annuler après confirmation (figure 194).

Dans tous ces diagrammes, les messages peuvent être affinés par une description OCL des opérations, en se basant pour l'instant sur le diagramme de classes du domaine (figure 185).

Les éditions (planning, taux d'occupation, listes des salles, etc.) sont des interactions simples : requête-réponse. Pour des raisons de place, nous ne les détaillons pas sous forme de collaboration. Il s'agit essentiellement de navigations dans le diagramme de classes métier.

Bilan des diagrammes d'interactions Les diagrammes de séquences précédents mettent en évidence le fonctionnement du système : on dispose d'une base d'information sous forme d'objets métier, les opérations consistent à consulter ou mettre à jour cette base (ajout / modification / suppression) via une interface graphique (fenêtres). Les ajouts ou suppres-

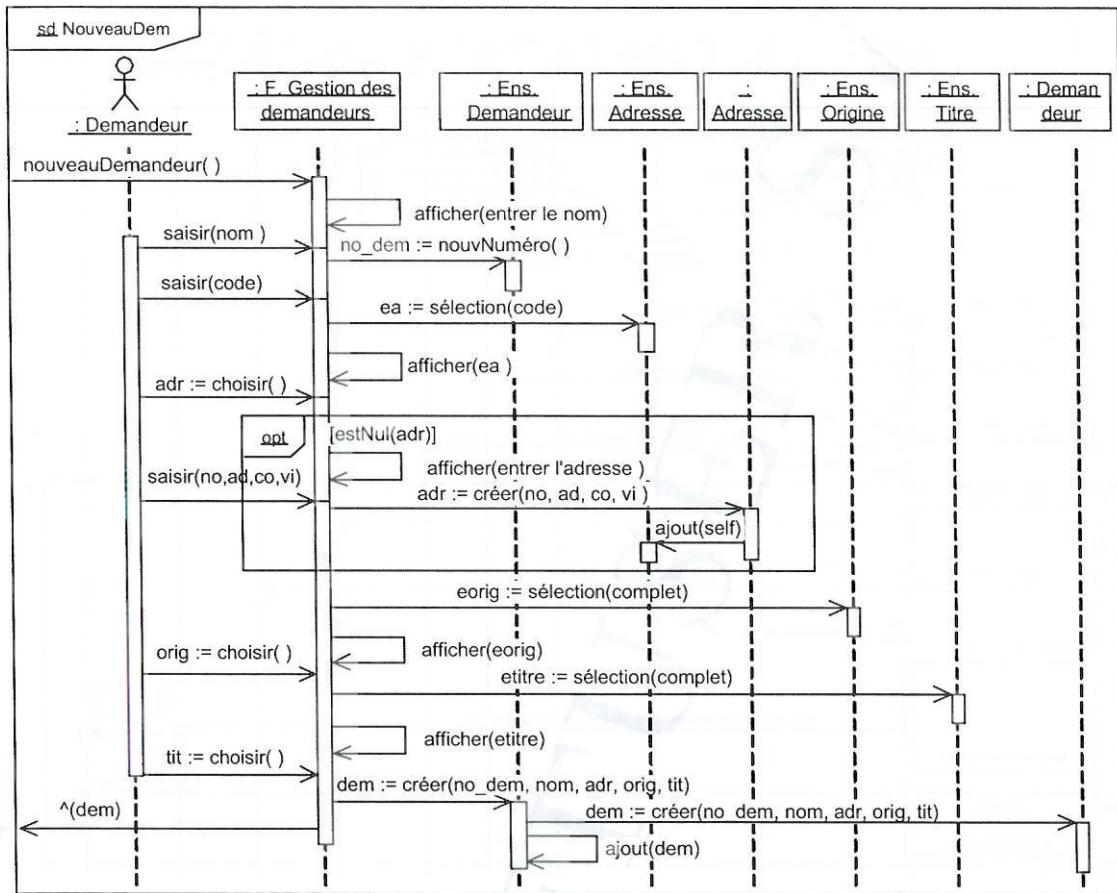


Figure 198 : Diagramme de séquences, ajout d'un demandeur - Salles

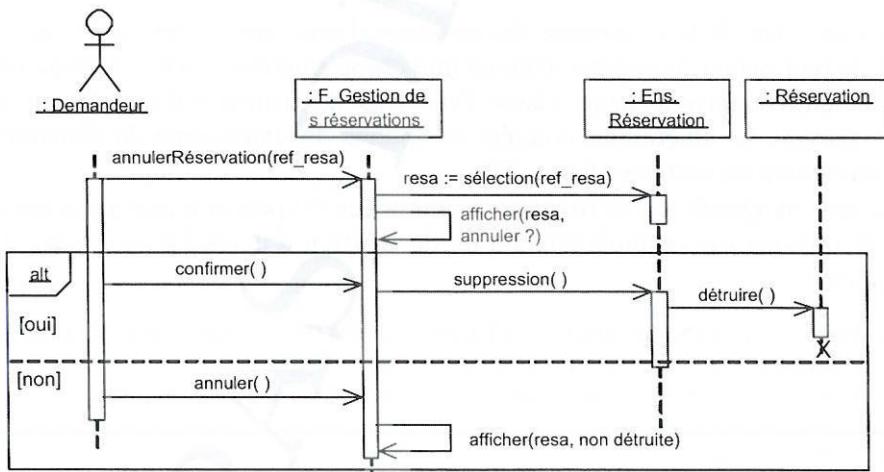


Figure 199 : Diagramme de séquences, annulation d'une réservation - Salles

sions sont plus ou moins complexes en fonction des contraintes du diagramme de classes (cardinalités, contraintes, etc.).

Diagramme de classes Les fenêtres sont des sous-classes d'une classe abstraite Fenêtre, comme dans l'exemple du club vidéo (figure 171). Les ensembles sont des collections d'ins-

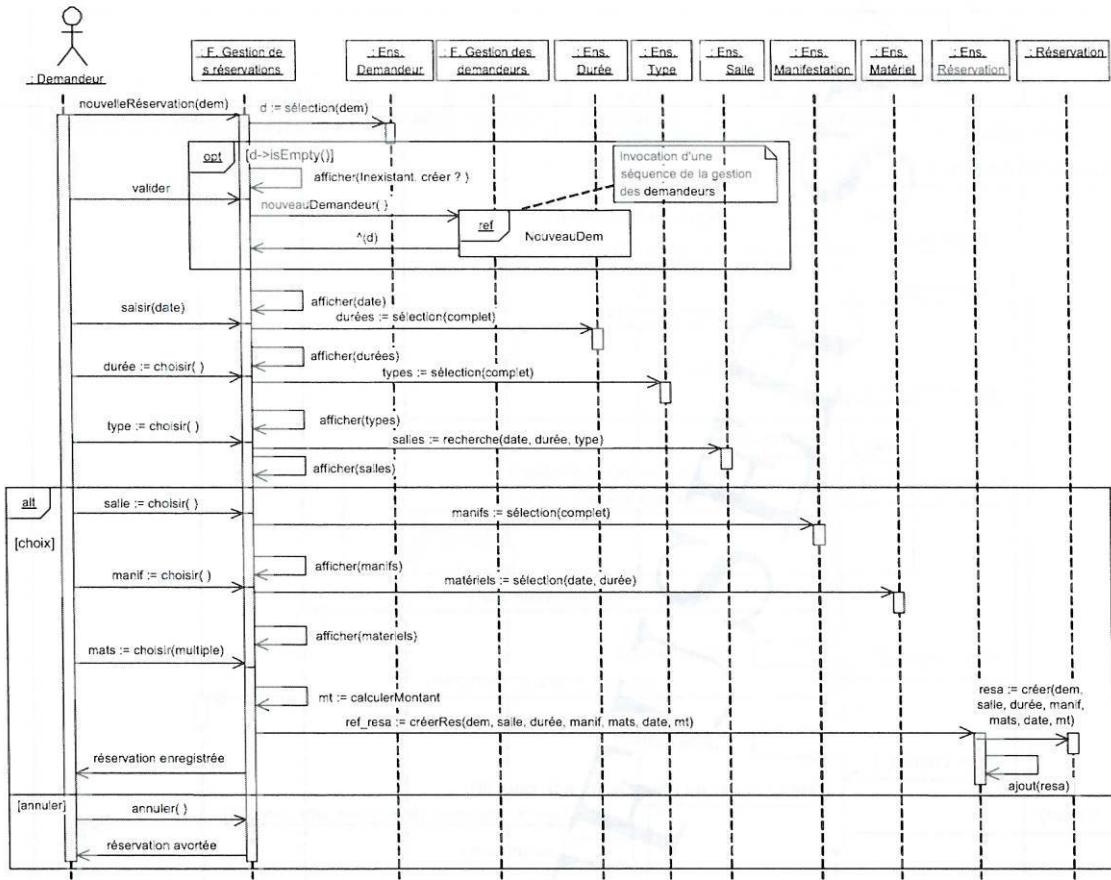


Figure 200 : Diagramme de séquences, ajout d'une réservation - Salles

tances d'une classe. En OCL, l'ensemble des instances d'une classe C est noté C.allInstances. Si on choisit de représenter le système comme une classe singleton (self : Système) associée directement ou indirectement à chaque classe, l'ensemble des instances d'une classe C est noté self.c par navigation. Le diagramme complet est l'union du diagramme du domaine (figure 185) et du diagramme du système (figure 201).

Chaque opération globale sur la base peut maintenant s'exprimer à partir des navigations du système. Reprenons par exemple l'opération de création d'une salle de la page 237 sous l'angle du système.

```

context Système::créerSalle(bat, noEtage, noSalle, superficie, type) : Salle
pre: -- le bâtiment et la salle existent
    self.bâtiment->includes(bat) and self.type->includes(type)
post: -- soit sal l'objet créé
    let sal : Salle in
        self.salle@pre->excludes(sal) and
        sal.no_étage = noEtage and sal.no_salle = noSalle and
        sal.no_bat = bat.no_bat and sal.superficie = superficie and
        sal.typeSalle = type and sal.bâtiment = bat and
        -- ajout explicite dans l'ensemble des instances
        self.salle = self.salle@pre->including(sal) and result = sal

```

La solution proposée dans cet exercice doit être affinée pour mettre en évidence chaque traitement. La description des traitements comprend une description des documents ou écrans de saisie utilisés ou produits et la spécification OCL des opérations invoquées. Ce travail est

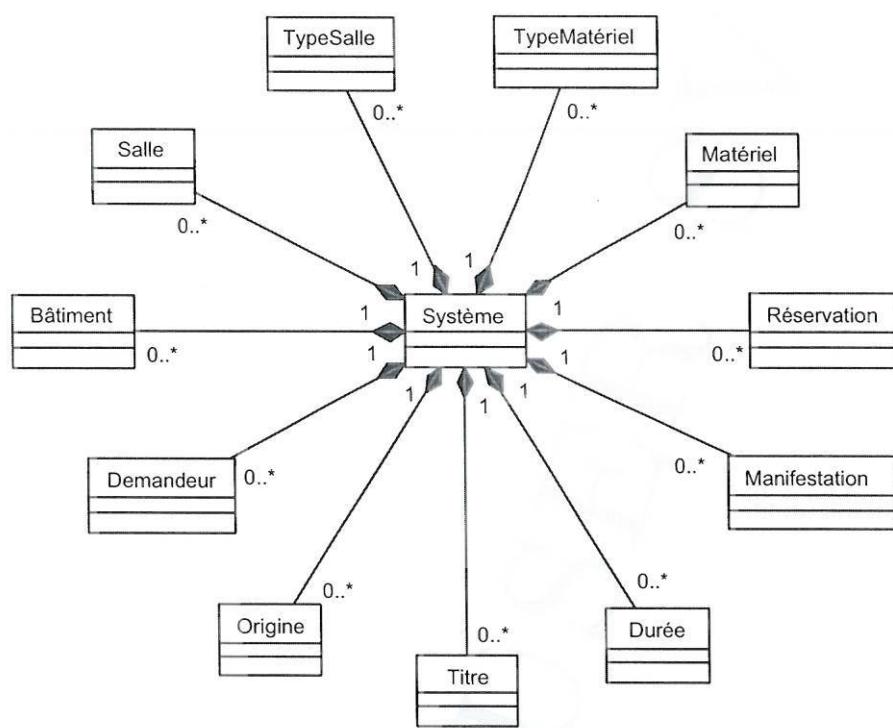


Figure 201 : Diagramme de classes métier, vue de composition - Salles

laissé à la charge du lecteur.

