

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Constantine 2



Faculté des Nouvelles Technologies pour l'Information et la Communication
Département des Technologies des Langages et Systèmes d'Information

Projet de fin d'études pour l'obtention du diplôme de
Licence en Informatique

Option : Génie Logiciel

Thème

Application Web pour la gestion des séances
d'enseignement

Dirigé par :

DOUIBI Halima

Réalisé par :

ZITOUNI Bessem

KHAMADJ Nasereddine

HIBA Saddam Hocine

Sommaire

INTRODUCTION GÉNÉRALE.....	5
CHAPITRE 1: ETAT DE L'ART	8
1.INTRODUCTION	9
2.LA PRESENTATION DE L'UML	9
1.Un peu d'Histoire	9
2.Définition	10
3.Pourquoi le choix d'UML ?	11
4.Modélisation UML	11
5.Les diagrammes d'UML	12
3.LA PRESENTATION DE L'UP	15
1.Définition	15
2.Un peu d'histoire	15
3.Les caractéristiques de l'UP	16
4.Les principes de l'UP	16
5.Cycle de vie de l'UP	17
6.L'architecture bidirectionnelle	17
7.Les activités de l'UP	19
8.Les phases de l'UP	20
4.CONCLUSION	21
CHAPITRE 2 : ETUDE PRELIMINAIRE ET SPECIFICATION DES BESOINS	22
1.INTRODUCTION	23
2.CAHIER DE CHARGE	23
2.1.Présentation de projet	23
2.2.Les fonctionnalités	24
3.DIAGRAMMES DE CAS D'UTILISATION	27
3.1.Introduction	27
3.2.Les Acteur Principaux	27
3.3Les digrammes de cas d'utilisation (par acteur)	28
4.DESCRPTION TEXTUEL DES CAS D'UTILISATIONS	33
4.1.Introduction	33
4.2.L'utilisateur	34
4.3.L'enseignant	43
4.4.L'étudiant	48
4.5.L'administrateur	49
6.LES DIAGRAMMES DE SEQUENCES	51
6.1.Introduction	51
6.2.Les diagrammes	52

7.LES DIAGRAMMES D'ACTIVITES	66
7.1.Introduction	66
7.2.Les diagrammes d'activité	67
8.CONCLUSION	79
CHAPITRE 3: ANALYSE ET CONCEPTION	80
1.INTRODUCTION	81
2.LES DIGRAMMES DE SEQUENCES DETAILLEES	81
2.1.Introduction	81
2.2.Les diagrammes	82
3.LES DIAGRAMMES DE NAVIGATIONS	96
3.1.Les diagrammes	96
4.LES DIAGRAMMES DE CLASSE	100
4.2.Les diagrammes de classes de métier	100
4.3.Les diagrammes de classe (métier-contrôle-dialogue)	104
5.CONCLUSION	111
CHAPITRE 4: IMPLÉMENTATION	112
1.INTRODUCTION	113
2.PRESENTATION DE L'ENVIRONNEMENT	113
2.1.Donnée	113
2.2.Traitement	116
2.2.1.Les classes contrôles	118
2.2.2.Les classes dialogues	121
3.PLATEFORME	126
3.1.Plateforme matérielle	126
3.2.Plateforme logicielle (les outils)	127
4.DEPLOIEMENT DE L'APPLICATION	127
4.1.Diagramme de déploiement	128
4.2.Test d'exécution	Error! Bookmark not defined.
5.MODE D'EXECUTION DE L'APPLICATION	129
5.1.L'architecture client-serveur	129
5.2.Modèle MVC	130
6.MESURE DE SECURITE	131
6.1.Authentification	131
6.2.Cryptage des données	131
6.3.Utilisation des adresses relative	132
7.INTERFACES UTILISATEUR: (PAGE WEB)	133
8.L'APPLICATION MOBILE	150
9.QUELQUE PROBLEME DANS L'IMPLEMENTATION	151
10.CONCLUSION	151

CONCLUSION GÉNÉRALE ET PERSPECTIVES	152
TABLE DES FIGURES	155
TABLE DES REFERENCES :	156

INTRODUCTION GÉNÉRALE

1. INTRODUCTION

Avant le Web 2.0, l'informatique était surtout un « lieu de stockage » de données, et un moyen de les médiatiser. Des « paquets de connaissance » étaient ainsi délivrés en classe, par disciplines et par « niveaux de formation », avec des didacticiels améliorés. L'informatique était aussi un moyen d'évaluer et noter le niveau de connaissance de l'étudiant, par son professeur ou de permettre via les mails, des échanges plus rapide et plus interactifs.

2. PROBLEMATIQUE

L'un des problème de l'éducation en Algérie et surtout qui concerne les université est l'absence de la technologie (informatique) qui donne le lieu aux plusieurs problèmes ,on peut prend par exemple notre université : qu'est normalement une département de l'informatique mais ça est seulement en théorique, pour que un étudiant peuvent consulter sa note aux pour poser des question sur un cours il faut recentrer le responsable de module , l'utilisations des outils de l'internet comme le mails peut résoudre ces problèmes mais ces outils ont rapidement montré des limites éducatives, notamment en termes de déperdition, temps passé et de surcharge informationnelle , nécessitant une pédagogie de la gestion des messageries électronique .

3. OBJECTIF

Notre objectif est de réaliser une application web pour répondre aux problèmes vue dans la problématique, cette application ne vise d'ailleurs pas à remplacer les fonctions de l'université (lecture, calcul, apprendre à apprendre...etc.), mais à mettre des solutions a des problèmes de l'éducation traditionnelle, le problème de transports est l'un de ces problèmes. Cette application et vue comme une autre façon d'apprentissage en ligne dans un cadre pédagogique, les services de publication des cours, des séries des exercices ou des interrogations permettent aux enseignants d'évaluer et de noter le niveau de l'étudiant, On peut citer quelques avantages de cette application :

- ✓ accès facile et peu couteux au contenu de formation (on se forme à partir de n'importe où, si on a accès à l'Internet)
- ✓ la possibilité de revenir en arrière quand il faut et au niveau requis, ce qui permet d'apprendre à son propre rythme.
- ✓ économies de temps.

- ✓ économies de transports,...etc.

4. STRUCTURE DE LA MEMOIRE

Ce mémoire est organisé en quatre chapitres :

- ❖ **1^{er} chapitre** : présent la démarche UML et e processus de développement UP.
- ❖ **2^{eme} chapitre** : présente le résultat de l'étude effectue sur le système.
- ❖ **3^{eme} chapitre** : présente l'analyse et la conception de notre application selon le processus de développement UP, on utilisant UML comme mécanisme de modalisation.
- ❖ **4^{eme} chapitre** : dans ce dernier chapitre nous présentons l'implémentation et la réalisation pratique de l'application ainsi que les outils de développement et les langages de programmation.
- ❖ **Conclusion générale** : clôture notre modeste travail.

CHAPITRE 1: ETAT DE L'ART

1. INTRODUCTION

Dans ce chapitre, nous décrivons les différents outils mis en œuvres pour l'analyse et conception de notre application de fin d'étude.

Ainsi, nous donnons un aperçu sur le langage UML (Unified Modeling Language) qui est utilisé ultérieurement dans le chapitre dédié à l'analyse et la conception.

2. LA PRESENTATION DE L'UML

En dix ans, UML (Unified Modeling Language) que l'on peut traduire par « Langage de Modélisation Unifier » s'est imposé comme un standard en matière des langages de modélisation des systèmes informatiques objets.

Ce langage est né de la fusion de plusieurs méthodes existant auparavant, et est devenu désormais la référence en termes de modélisation objet.

1. Un peu d'Histoire

La technologie orientée objet est issue du monde de la programmation pour répondre à des besoins dans l'industrie du logiciel tel que la rapidité du développement, la réutilisation et la modularité. Ainsi des langages tels que SMALTALK, C++, ADA ont eu du succès dans ce domaine.

Dès lors, aux années 80 on commença à réfléchir à des langages de modélisation ou conception graphique. Les premiers ouvrages sur le sujet parurent entre les années 1980 à 1992, proposant des démarches ou méthodes de modélisation orientées objet, tels que :

- **GRADY BOOCH** : Méthode OOAD « Object Oriented Analysis and Design » en français « Analyse et Modélisation Orientée Objet ».
- **PETER HOOD** : Méthode OOD « Object Oriented Design » en français « Modélisation orienté Objet ».
- **IVAR JACOBSON** : Méthode OOSE « Oriented Object Software Engineering » en français « Génie Logiciel Orienté Objet ».
- **JAMES RUMBAUGH** : Méthode OMT «Object Modeling Technique » en français « Technique de Modélisation Objet ».

Tous les autres cités ci-dessus et bien d'autres proposaient des méthodes très proches les unes des autres puisque elles étaient toutes basées sur les concepts issus de l'orienté objet. Cependant, chacun utilisait son propre formalisme ou notation. L'OMG (Object Management

Group), posa, alors, le problème de standardisation de ces formalismes. L'OMG est un consortium d'entreprises fondé pour construire des standards pour les systèmes orientés objet dans le but d'en assurer leur interopérabilité.

En **1995**, **GRADY BOOCH** et **JAMES RUMBAUGH**, à l'époque chercheurs chez Rational Software (société américaine spécialisée en software), ont préparé description de leur méthode unifiée sous la version v8.0. La même année, les autres sont rejoints par **IVAR JACOBSON** pour travailler sur le même projet.

En **1997**, Rational propose la version 1.0 d'UML à l'OMG. Les auteurs cités sont considérés aujourd'hui comme les géniteurs d'UML et continuent à œuvrer pour son amélioration à travers ses différentes versions. L'évolution des versions d'UML peut être récapitulée dans le tableau suivant :

Année	Version
1995	Méthodes unifiée UML 0.8 (intégrant la méthode de BOOCH) puis UML 0.9 (intégrant la méthode OOSE et OMT).
1996	UML 1.0 proposé à l'OMG.
1997	UML 1.0 standardisé par l'OMG.
1998	UML 1.2
1999	UML 1.3
2000	UML 1.4
2003	UML 1.5 puis UML 2.0

Figure1 : Tableau des différentes versions d'UML

2. Définition

La modélisation orientée objet dans tous les domaines est aujourd'hui dominée par le langage UML (Unified Modeling Language).

UML est un langage de modélisation unifié résultant de l'unification d'un ensemble de concepts pris des méthodes orientés objet dans le but de modéliser d'une façon claire et

Précise la structure et le comportement d'un système indépendamment de toutes méthodes et de tout langage de programmation.

3. Pourquoi le choix d'UML ?

UML a été choisi comme langage de modélisation dans notre projet ; parmi les points forts suivantes :

- ❖ **UML est un langage formel et normalisé** : il permet
 - gain de précision.
 - gage de stabilité.
 - encourage l'utilisation d'outils.
- ❖ **UML est un support de communication performant**
 - Il cadre l'analyse.
 - Il facilite la compréhension de représentations abstraites complexes.
 - Son caractère polyvalent et sa souplesse en font un langage universel.

4. Modélisation UML

4.1. Qu'est-ce qu'un modèle ?

La modélisation consiste à créer une représentation simplifiée d'un problème: **le modèle**. Grâce au modèle il est possible de représenter simplement un problème, un concept et le simuler.

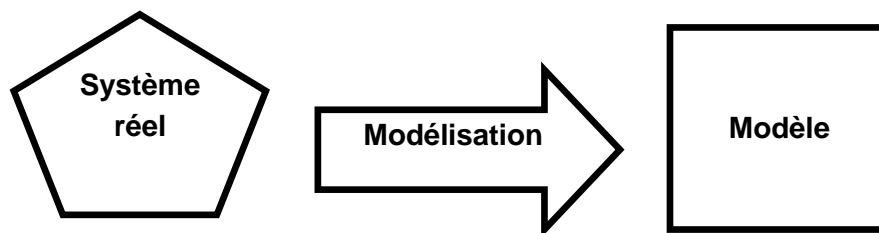


Figure 2 : Présentation d'un modèle

La modélisation apparaît dans les deux premières phases d'un processus de développement l'analyse et la conception. Ce sont deux étapes importantes :

- L'analyse : permet de se familiariser avec le domaine d'étude et de pointer le problème et ce en représentant ce qui existe.
- La conception : permet de proposer une solution à la problématique et aussi de représenter cette solution.

Le modèle constitue donc, une représentation possible du système pour un point de vue donné

4.2. La modélisation UML

Le langage UML fournit à l'aide de ses outils des représentations pour l'ensemble des éléments du monde objet (classes, objets, ...) ainsi que les liens qui les relient.

Toutefois, étant donné qu'une seule représentation est trop subjective, UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux **vues**.

Une vue est constitué d'un ou plusieurs **diagrammes** de sorte qu'ils sont des supports graphiques de modélisation qui met en œuvre un ensemble d'éléments de visualisation représentant les concepts du modèle.

5. Les diagrammes d'UML

Les diagrammes UML peuvent être classés sous deux types de vues :

- **Les vues statiques** : représentant le système physiquement.
- **Les vues dynamiques** : montrant le fonctionnement du système.

Diagramme de vue statique	Diagramme de vue dynamique
<ul style="list-style-type: none">• Diagramme d'objet• Diagramme de classe• Diagramme de cas d'utilisation• Diagramme de composant• Diagramme de déploiement	<ul style="list-style-type: none">• Diagramme de collaboration• Diagramme d'activité• Diagramme de séquence• Diagramme d'état-transition

Figure 3 : tableau de différent diagramme d'UML

5.1. Diagrammes de la vue statique

Qui sont au nombre de cinq(5) :

5.1.1. Diagrammes de cas d'utilisation (use case)

Les use cases permettent de structurer les besoins des utilisateurs et les objectifs correspondants d'un système. Ils centrent l'expression des exigences du système sur ses utilisateurs : ils partent du principe que les objectifs du système sont tous motivés.

La portée des cas d'utilisation dépasse largement la définition des besoins du système. Les cas d'utilisation interviennent tout au long du cycle de vie du projet, depuis le cahier des charges jusqu'aux tests.

5.1.2. Diagrammes de classes

Sont sans doute les diagrammes les plus utilisés d'UML. Ils décrivent les types des objets qui composent un système et les différents types de relations statiques qui existent entre eux. Ils font abstraction du comportement du système.

Le diagramme de classe permet de représenter l'ensemble des informations finalisées qui sont gérées par le domaine. Ces informations sont structurées, c'est-à-dire qu'elles ont regroupées dans des classes. Le diagramme met en évidence d'éventuelles relations entre ces classes.

5.1.3. Diagrammes d'objets

Le diagramme d'objets permet de mettre en évidence des liens entre les objets. Les objets, instances de classes, sont reliés par des liens, instances d'associations. A l'exception de la multiplicité, qui est explicitement indiquée, le diagramme d'objet utilise les mêmes concepts que le diagramme de classes. Ils sont essentiellement utilisés pour comprendre ou illustrer des parties complexes d'un diagramme de classes.

5.1.4. Diagramme de composant

Les diagrammes de composants décrivent les composants et leurs dépendances dans l'environnement de réalisation.

En général, ils ne sont utilisés que pour des systèmes complexes.

5.1.5. Diagrammes de déploiement

Les diagrammes de déploiement montrent la disposition physique des différents matériels (les nœuds) qui entrent dans la composition d'un système et la répartition des instances de composants, processus et objets qui « vivent » sur ces matériels.

Les diagrammes de déploiement sont donc très utiles pour modéliser l'architecture physique d'un système.

5.2. Diagrammes de la vue dynamique

Qui sont au nombre de quatre(4) :

5.2.1. Diagramme d'activité

Le diagramme d'activité est attaché à une catégorie de classe et décrit le déroulement des activités de cette catégorie. Le déroulement s'appelle "**flot de contrôle**".

Il indique la part prise par chaque objet dans l'exécution d'un travail. Il sera enrichi par les **conditions** de séquençages.

5.2.2. Diagramme de collaboration

Le diagramme de collaboration permet de mettre en évidence les interactions entre les différents objets du système. Dans le cadre de l'analyse, il sera utilisé

- pour préciser le contexte dans lequel chaque objet évolue.
- pour mettre en évidence les dépendances entre les différents objets impliqués dans l'exécution d'un processus ou d'un cas d'utilisation.

Un diagramme de collaboration fait apparaître les interactions entre des objets et les messages qu'ils échangent.

5.2.3. Diagramme d'état-transition

Ils ont pour rôle de représenter les traitements (opérations) qui vont gérer le domaine étudié. Ils définissent l'enchaînement des états de classe et font donc apparaître l'ordonnancement des travaux.

Le diagramme d'états-transition est associé à une classe pour laquelle on gère différents états : il permet de représenter tous les états possibles ainsi que les événements qui provoquent les changements d'état.

5.2.4. Diagramme de séquence

Les diagrammes de séquence possèdent intrinsèquement une dimension temporelle mais ne représente pas explicitement les liens entre les objets. Ils privilégient ainsi la représentation

Temporelle à la représentation spatiale et sont plus aptes à modéliser les aspects dynamiques du système.

Le diagramme de séquence permet de visualiser les messages par une lecture de haut en bas. L'axe vertical représente le temps, l'axe horizontal les objets qui collaborent. Une ligne verticale en pointillé est attachée à chaque objet et représente sa durée de vie.

Remarque :

Dans notre conception nous allons utiliser le diagramme de séquence de la version 2.0 d'UML.

3. LA PRESENTATION DE L'UP

L'UML n'est pas une méthode mais seulement un langage, donc il nous faut un processus de développement pour passer des besoins vers un code. Nous avons opté pour le processus UP (Unified Process).le choix de ce processus est motivé par le fait que c'est un standard et qu'il s'adapte bien aux domaines tels que le E-business ou le E-commerce.

1. Définition

Le processus unifié (UP) est un processus de développement logiciel itératif, centré sur l'architecture, piloté par des cas d'utilisation et orienté vers la diminution des risques.

C'est un canevas de processus pouvant être adapté à une large classe de systèmes logiciels

- À différents domaines d'application,
- À différents types d'entreprises,
- À différents niveaux de compétences
- À différentes tailles de l'entreprise.

2. Un peu d'histoire

Le terme “**Unifié**” est très approprié puisqu'il s'agit de la fusion des travaux d'**Ivar Jacobson**, **Grady Booch** (au départ chez Objectory) et de **James Rumbaugh**, enrichie de nombreux apports issus d'UML (développé en parallèle) et du produit commercial RUP, sorti

En 1998 et toujours mis à jour par IBM (après le rachat de Rational qui avait lui-même acheté Objectory en 1995). D'autres ont permis de peaufiner le processus, notamment Walker Royce et Philippe Kruchten pour la planification et la gestion de projet.

3. Les caractéristiques de l'UP

Les caractéristiques essentielles du processus unifié :

- Le processus unifié est à base de composants.
- Le processus unifié utilise le langage UML (ensemble d'outils et de diagramme).
- Le processus unifié est piloté par les cas d'utilisation.
- Centré sur l'architecture.
- Itératif et incrémental.

4. Les principes de l'UP

Le Processus Unifié (UP) est **piloté par les cas d'utilisation** (eux même guidés par les risques) **centré sur l'architecture, itératif et incrémental**.

4.1. Le processus unifié est piloté par les cas d'utilisation

L'objectif principal d'un système logiciel est de rendre service à ses utilisateurs ; il faut par conséquent bien comprendre les désirs et les besoins des futurs utilisateurs. **Le processus de développement sera donc centré sur l'utilisateur**. Le terme utilisateur ne désigne pas seulement les utilisateurs humains mais également les autres systèmes. L'utilisateur représente donc une personne ou une chose dialoguant avec le système en cours de développement.

Les cas d'utilisation font apparaître les besoins fonctionnels et leur ensemble constitue le modèle des cas d'utilisation qui décrit les fonctionnalités complètes du système.

4.2. Le processus unifié est itératif

L'itération est une répétition d'une séquence d'instructions ou d'une partie de programme un nombre de fois fixé à l'avance ou tant qu'une condition définie n'est pas remplie, dans le but de reprendre un traitement sur des données différentes.

Elle qualifie un traitement ou une procédure qui exécute un groupe d'opérations de façon répétitive jusqu'à ce qu'une condition bien définie soit remplie.

Une itération prend en compte un certain nombre de cas d'utilisation et traite en priorité les risques majeurs.

Ph.Kruchten propose différentes perspectives, indépendantes et complémentaires, qui permettent de définir un modèle d'architecture (publication IEEE, 1995).

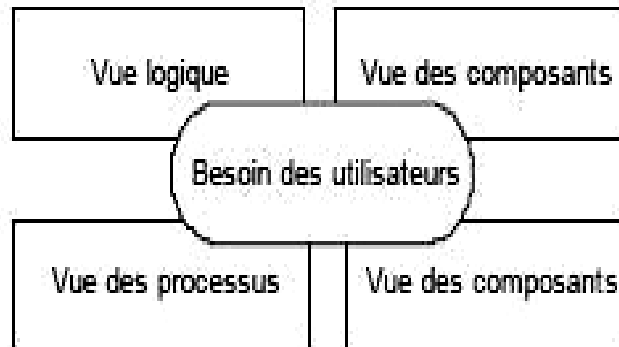


Figure 4 : la vue ("4+1")

5. Cycle de vie de l'UP

L'objectif d'un processus unifié est de maîtriser la complexité des projets informatiques en diminuant les risques.

UP est un ensemble de principes génériques adapté en fonctions des spécificités des projets. UP répond aux préoccupations suivantes :

- **QUI** participe au projet ?
- **QUOI**, qu'est-ce qui est produit durant le projet ?
- **COMMENT** doit-il être réalisé ?
- **QUAND** est réalisé chaque livrable ?

6. L'architecture bidirectionnelle

UP gère le processus de développement par deux axes.

L'axe vertical : représente les principaux enchaînements d'activités, qui regroupent les activités selon leur nature. Cette dimension rend compte l'aspect statique du processus qui s'exprime en termes de composants, de processus, d'activités, d'enchaînements, d'artefacts et de travailleurs.

L'axe horizontal_: représente le temps et montre le déroulement du cycle de vie du processus; cette dimension rend compte de l'aspect dynamique du processus qui s'exprime en terme de cycles, de phases, d'itérations et de jalons.

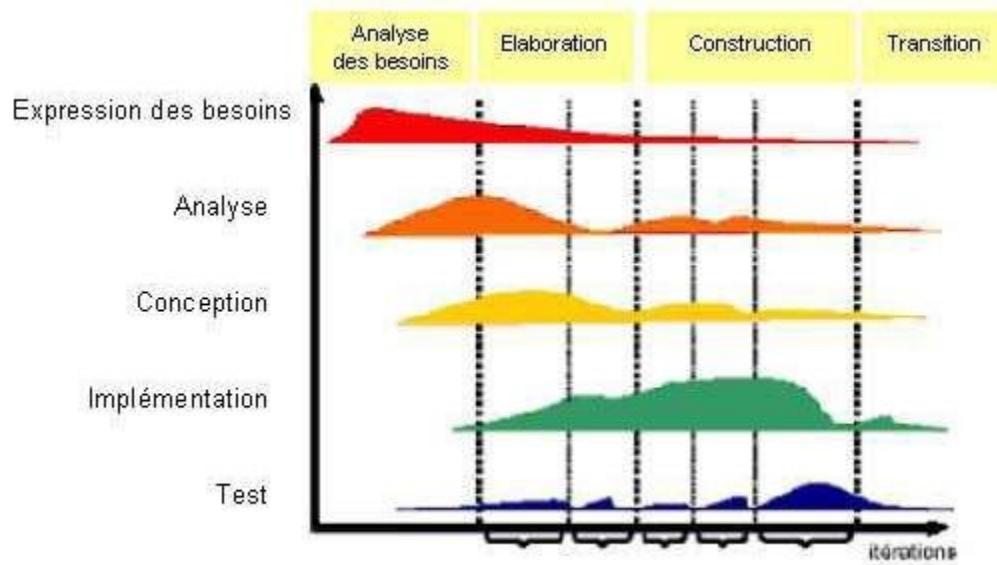


Figure 5 : Présentation de cycle de vie du processus UP

UP répète un certain nombre de fois une série de cycle qui s'articule autour de 4 phases :

- analyse des besoins
- élaboration
- construction
- transition

Pour mener efficacement un tel cycle, les développeurs ont besoins de toutes les représentations du produit logiciel

- un modèle de cas d'utilisation
- un modèle d'analyse : détailler les cas d'utilisation et procéder à une première répartition du comportement
- un modèle de conception : finissant la structure statique du système sous forme de sous-systèmes, de classes et interfaces.
- un modèle d'implémentation : intégrant les composants
- un modèle de déploiement : définissant les nœuds physiques des ordinateurs.
- un modèle de test : décrivant les cas de test vérifiant les cas d'utilisation.
- une représentation de l'architecture

7. Les activités de l'UP

IL existe quatre type d'activités, se sont :

7.1.Expression des besoins

L'expression des besoins comme son nom l'indique, permet de définir les différents besoins :

- inventorier les **besoins principaux** et fournir une liste de leurs fonctions
- recenser les **besoins fonctionnels** (du point de vue de l'utilisateur) qui conduisent à l'élaboration des modèles de cas d'utilisation.
- appréhender les **besoins non fonctionnels** (technique) et livrer une liste des exigences.

Le modèle de cas d'utilisation présente le système du point de vue de l'utilisateur et représente sous forme de cas d'utilisation et d'acteur, les besoins du client.

7.2.Analyse

L'objectif de l'analyse est d'accéder à une compréhension des besoins et des exigences du client. Il s'agit de livrer des spécifications pour permettre de choisir la conception de la solution.

Un modèle d'analyse livre une spécification complète des besoins issus des cas d'utilisation et les structure sous une forme qui facilite la compréhension (scénarios), la préparation (définition de l'architecture), la modification et la maintenance du futur système.

Il s'écrit dans le langage des développeurs et peut être considéré comme une première ébauche du modèle de conception.

7.3.Conception

La conception permet d'acquérir une compréhension approfondie des contraintes liées au langage de programmation, à l'utilisation des composants et au système d'exploitation.

Elle détermine les principales interfaces et les transcrit à l'aide d'une notation commune. Elle constitue un point de départ à l'implémentation :

- elle décompose le travail d'implémentation en sous-système.
- elle crée une abstraction transparente de l'implémentation.

7.4.Implémentation

L'implémentation est le résultat de la conception pour implémenter le système sous formes de composants, c'est-à-dire, de code source, de scripts, de binaires, d'exécutables et d'autres éléments du même type.

Les objectifs principaux de l'implémentation sont de planifier les intégrations des composants pour chaque itération, et de produire les classes et les sous-systèmes sous formes de codes sources.

7.5.Test

Les tests permettent de vérifier des résultats de l'implémentation en testant la construction. Pour mener à bien ces tests, il faut les planifier pour chaque itération, les implémenter en créant des cas de tests, effectuer ces tests et prendre en compte le résultat de chacun.

8. Les phases de l'UP

8.1.Phase de création

Traduit une idée en vision de produit fini et présente une étude de rentabilité pour ce produit. Elle répond aux questions suivantes :

- Que va faire le système pour les utilisateurs ?
- A quoi peut ressembler l'architecture d'un tel système ?
- Quels sont l'organisation et les coûts du développement de ce produit ?

On fait apparaître les principaux cas d'utilisation.

L'architecture est provisoire, identification des risques majeurs et planification de la phase d'élaboration.

8.2.Phase d'élaboration

Permet de préciser la plupart des cas d'utilisation et de concevoir l'architecture du système. L'architecture doit être exprimée sous forme de vue de chacun des modèles (émergence d'une **architecture de référence**).

A l'issue de cette phase, le chef de projet doit être en mesure de prévoir les activités et d'estimer les ressources nécessaires à l'achèvement du projet.

Les tâches à effectuer dans la phase élaboration sont les suivantes :

- créer une architecture de référence.

- identifier les risques, ceux qui sont de nature à bouleverser le plan, le coût et le calendrier.
- définir les niveaux de qualité à atteindre.
- formuler les cas d'utilisation pour couvrir les besoins fonctionnels et planifier la phase de
- Construction.
- élaborer une offre abordant les questions de calendrier, de personnel et de budget.

8.3.Phase de construction

Moment où l'on construit le produit. L'architecture de référence se métamorphose en produit complet, elle est maintenant stable. Le produit contient tous les cas d'utilisation que les chefs de projet, en accord avec les utilisateurs ont décidé de mettre au point pour cette version .Celle-ci doit encore avoir des anomalies qui peuvent être en partie résolue lors de la phase de transition.

8.4.Phase de transition

Le produit est en version bêta. Un groupe d'utilisateurs essaye le produit et détecte les anomalies et défauts. Cette phase suppose des activités comme la fabrication, la formation des utilisateurs clients, la mise en œuvre d'un service d'assistance et la correction des anomalies constatées. (Où le report de leur correction à la version suivante).

4. CONCLUSION

Ce chapitre a été consacré à la présentation des différents outils de développement employés pour l'analyse et la conception de notre application web. Nous avons introduit le maximum des concepts relatifs au langage UML et le processus UP.

Dans le prochain chapitre, nous allons entamer les étapes de l'analyse et de la conception, en appliquant les concepts théoriques illustrés dans le présent chapitre.

CHAPITRE 2 : ETUDE PRELIMINAIRE ET SPECIFICATION DES BESOINS

1. INTRODUCTION

Dans cette chapitre on présente le résultat de l'étude effectuée sur le système avec :

- Le cahier de charge.
- Les diagrammes de cas d'utilisation.
- Les descriptions textuelles.
- Les diagrammes de séquences.
- Les diagrammes d'activités.

2. CAHIER DE CHARGE

2.1. Présentation de projet

Dans le cadre de notre projet de fin d'étude, il nous est demandé de concevoir et de réaliser une application web en utilisant les technologies JEE.

Le but de ce projet est de créer un espace de connaissance permet aux étudiants et enseignants inscrits (application web) d'échanger des connaissances entre eux.

On peut voir L'application comme une simulation des sciences de td et de cours.

L'application offre à ses utilisateurs une bibliothèque de livres de divers catégories, la bibliothèque permette à l'utilisateur de consulter ou de télécharger des livres ou des documents. L'application offre aussi des services de discussion et de communication (chat).

Les étudiants et les enseignants peuvent communiquer entre eux en différents manières soit par l'envoi et la réception des messages ou bien par la participation dans les espaces de discussion (chat), l'autre manière c'est les cours et les td publiés par les enseignants avec des questions posées, des réponses, des commentaires ou des observations. Chaque utilisateur aura une page personnelle (profile) qui contient toutes ses informations.

L'application est divisée en des filières (créées par l'administrateur) chaque filière procède des modules (créés par l'administrateur), dans chaque module on trouve les cours (créés par les enseignants), les cours peuvent aussi procéder des TDs (travaux dirigés), les modules peuvent avoir des tests (interrogation proposée par l'enseignant), les cours sont limités à la création par un délai, et alloués à des groupes spécifiques, à l'inscription l'utilisateur doit choisir son type (étudiant ou enseignant) et pour accéder aux services de l'application il doit valider son compte par le code de confirmation, ce code sera envoyé (à l'email de l'utilisateur) par administrateur de l'application si l'utilisateur est de type enseignant sinon il est envoyé automatiquement, l'enseignant peut ajouter des cours seulement dans ses modules, les étudiants peuvent répondre, aux TDs.

Et aux interrogations avant la fin de délai des TDs et des interrogations ce délai sera fixé à la création par l'enseignant, les réponses des étudiants seront consultées et notées par l'enseignant responsable (celui qui a créé le td ou bien l'interrogation).

L'application offre à l'enseignant la possibilité de créer et de gérer des groupes d'étudiants, ces groupes sont créés par l'enseignant et validés par l'administrateur.

L'étudiant peut ajouter des commentaires au cours ou des réponses aux TDs ou interrogation seulement s'il appartient à l'un des groupes alloués au cours ou interrogation.

L'étudiant peut demander de participer dans l'un des groupes et son demande sera validée par l'enseignant responsable (celui qui a créé le groupe).

2.2. Les fonctionnalités

2.2.1. L'utilisateur

- ✓ Inscrire dans le site.
- ✓ S'authentifier.
- ✓ Confirmer son compte (enseignant, étudiant).
- ✓ modifier son profile.
- ✓ consulter les cours.
- ✓ consulter les observations.
- ✓ consulter les TD.
- ✓ Consulter les commentaires.
- ✓ Consulter un profile.
- ✓ Gérer son messagerie.
- ✓ Consulter les réponses.
- ✓ Consulter un livre.
- ✓ Ajouter commentaires.
- ✓ Ajouter des livres.
- ✓ télécharger un cours.
- ✓ télécharger un TD.
- ✓ Télécharger un livre.
- ✓ télécharger une interrogation.
- ✓ télécharger un corrigé type.
- ✓ imprimer un cours.
- ✓ imprimer un TD.

- ✓ imprimer une interrogation.
- ✓ imprimer un corrigé type.
- ✓ Imprimer un livre.
- ✓ modifier ses commentaires.
- ✓ recherche sur un cours.
- ✓ recherche sur un TD.
- ✓ recherche sur un livre.
- ✓ recherche sur une interrogation.
- ✓ Participer dans le service de chat.
- ✓ quitter l'application.

2.2.2. L'enseignant

- ✓ Ajouter des cours.
- ✓ Ajouter des observations.
- ✓ Ajouter des interrogations.
- ✓ Ajouter des TD.
- ✓ Ajouter des groupes
- ✓ ajouter un corrigé type.
- ✓ Gérer ses cours
- ✓ Gérer ses séries d'exercices.
- ✓ Gérer ses interrogations.
- ✓ Gérer ses observations.
- ✓ Gérer ses corrigés type.
- ✓ Gérer ses groupes.
- ✓ Noter les réponses.

2.2.3. Les étudiants

- ✓ Ajouter une réponse.
- ✓ Modifier son groupe

2.2.4. L'administrateur

- ✓ Supprimer des commentaires.
- ✓ Supprimer des cours.
- ✓ Supprimer des TDs.

- ✓ Supprimer des interrogations.
- ✓ Supprimer des corrigés type.
- ✓ Valide un livre.
- ✓ Gérer les filières.
- ✓ Gérer les modules.
- ✓ Gérer les groupes.
- ✓ Bloquer un utilisateur.

3. DIAGRAMMES DE CAS D'UTILISATION

3.1. Introduction

Un diagramme de cas d'utilisation capture le comportement d'un système, d'un sous-système, d'une classe ou d'un composant tel qu'un utilisateur extérieur le voit. Il scinde la fonctionnalité du système en unités cohérentes, les cas d'utilisation, ayant un sens pour les acteurs. Les cas d'utilisation permettent d'exprimer le besoin des utilisateurs d'un système, ils sont donc une vision orientée utilisateur de ce besoin au contraire d'une vision informatique.

Il ne faut pas négliger cette première étape pour produire un logiciel conforme aux attentes des utilisateurs. Pour élaborer les cas d'utilisation, il faut se fonder sur des entretiens avec les utilisateurs.

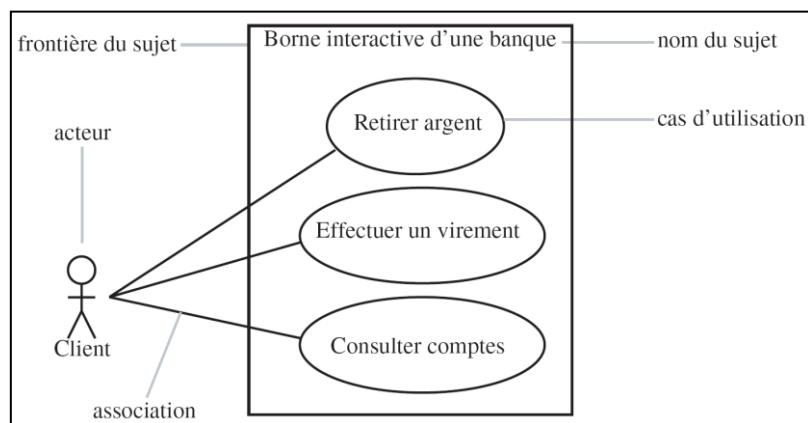


Figure 6 : Exemple simplifié de diagramme de cas d'utilisation modélisant une borne d'accès à une banque.

3.2. Les Acteur Principaux

Dans notre application il y a 3 acteurs se sont:

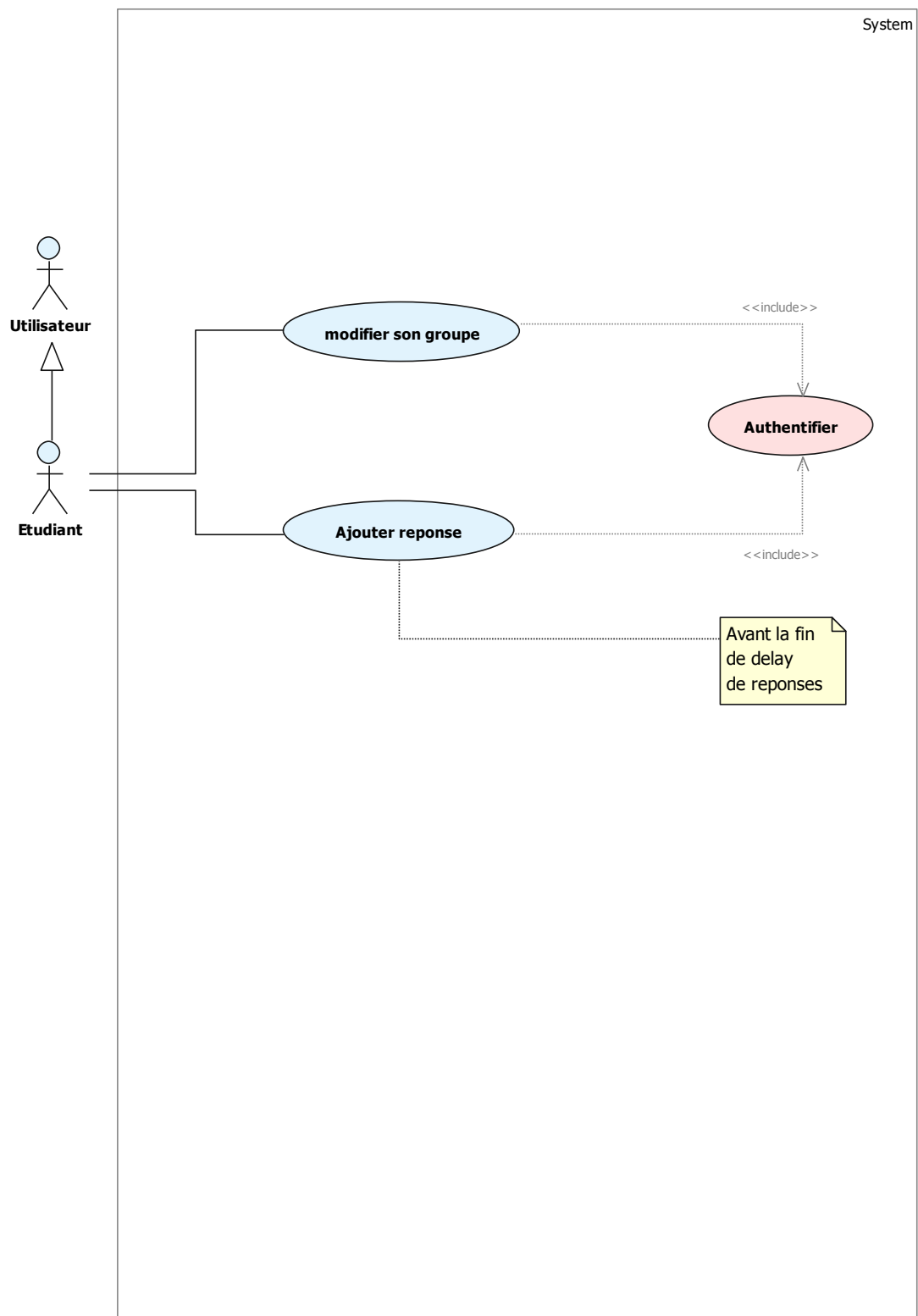
- L'étudiant.
- L'enseignant.
- L'Administrateur.

Remarque1 : l'utilisateur est acteur qui prend les cas répétés dans les autres acteurs.

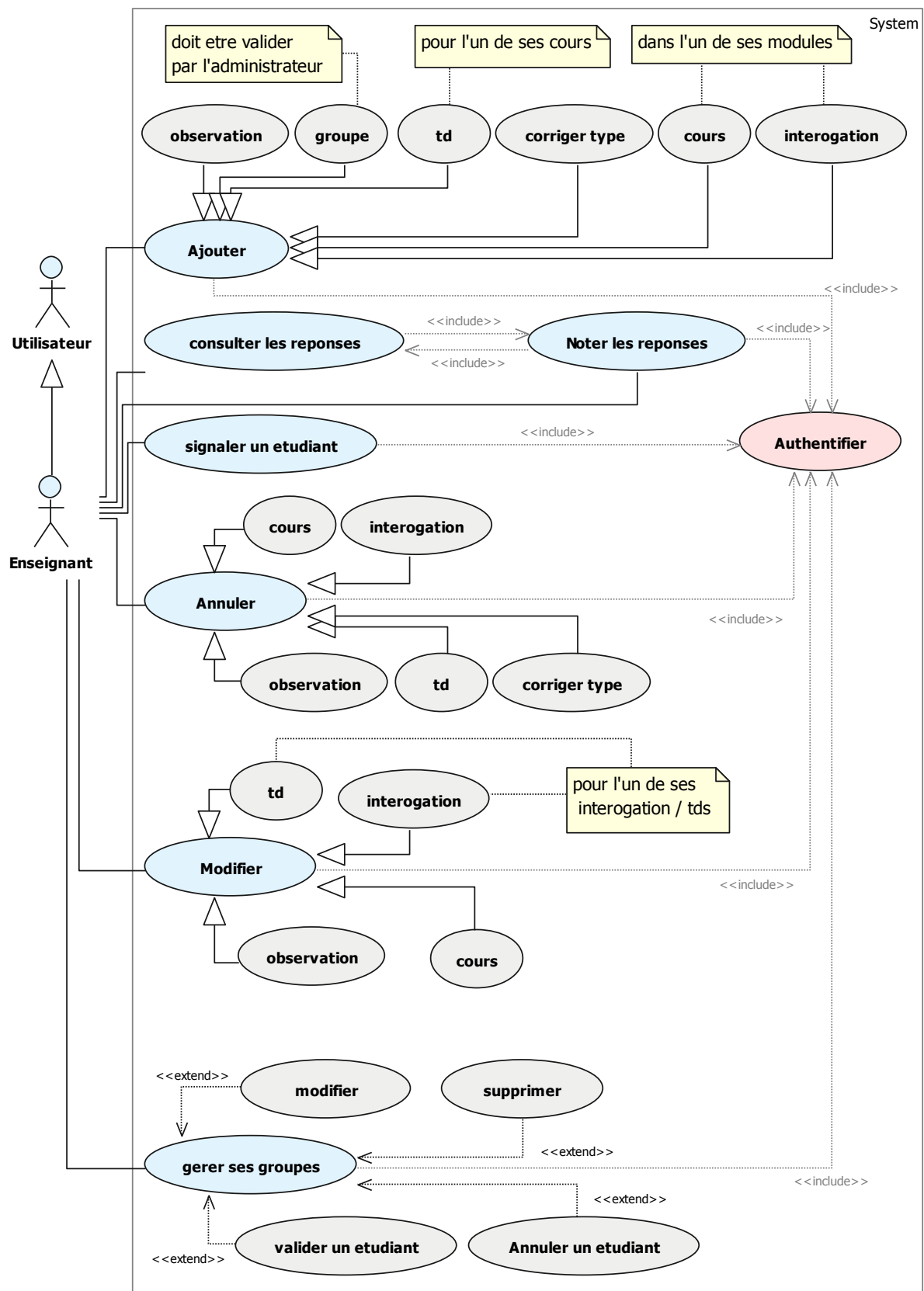
Remarque2 : pour la visibilité de tous les cas de l'utilisateur on a partitionné le diagramme aux deux parties.

3.3. Les digrammes de cas d'utilisation (par acteur)

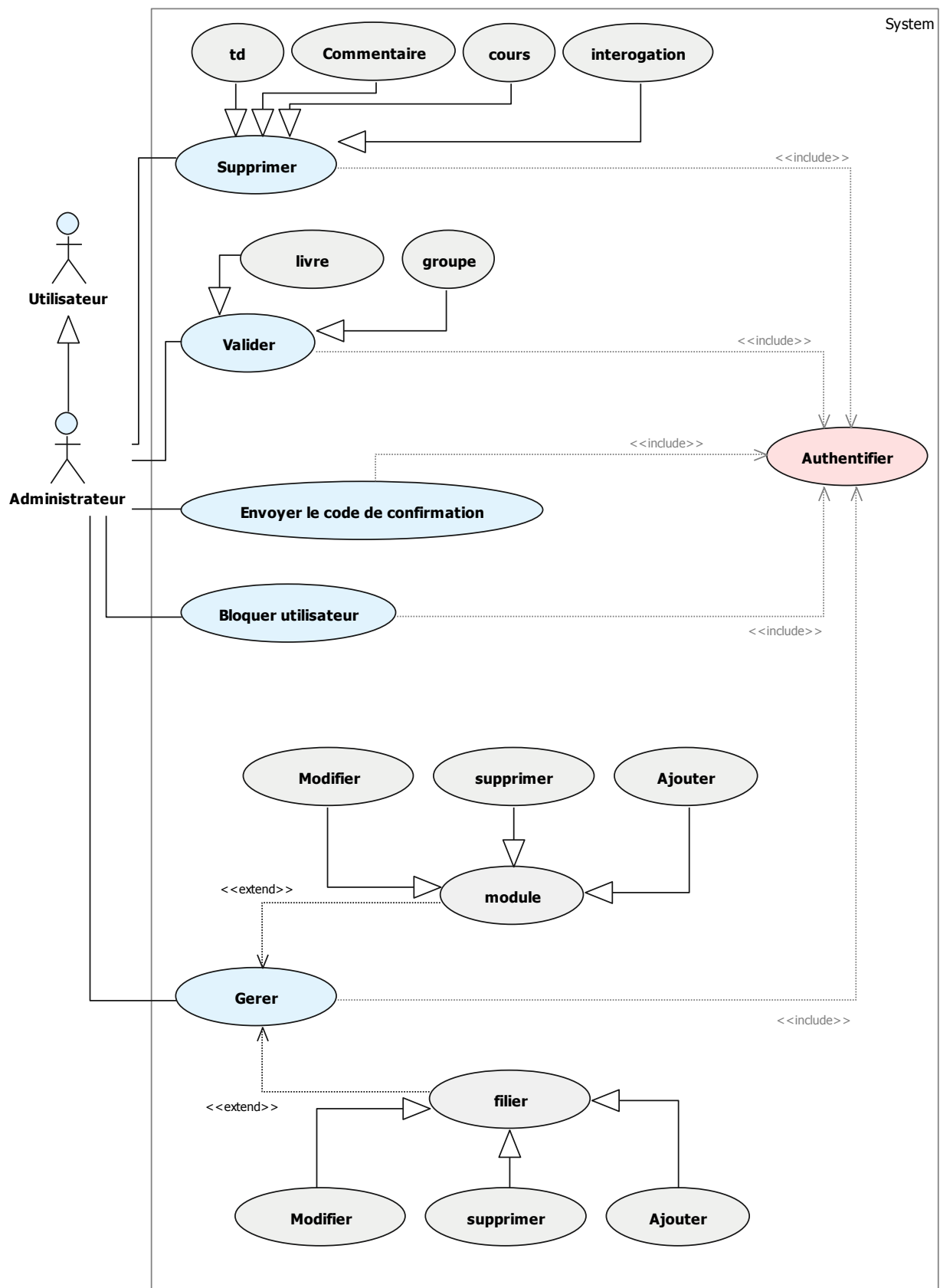
3.3.1. L'étudiant



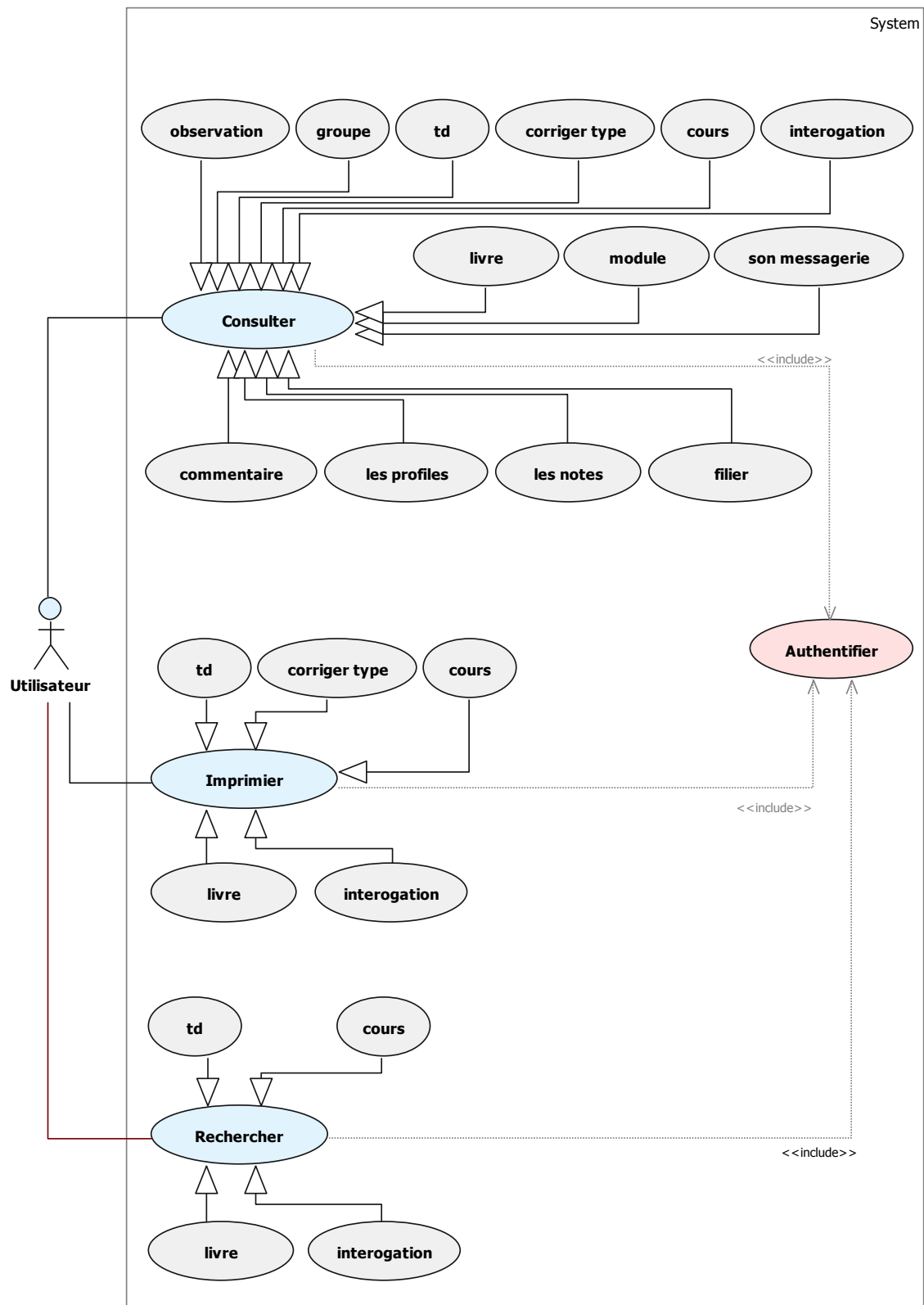
3.3.2. L'enseignant



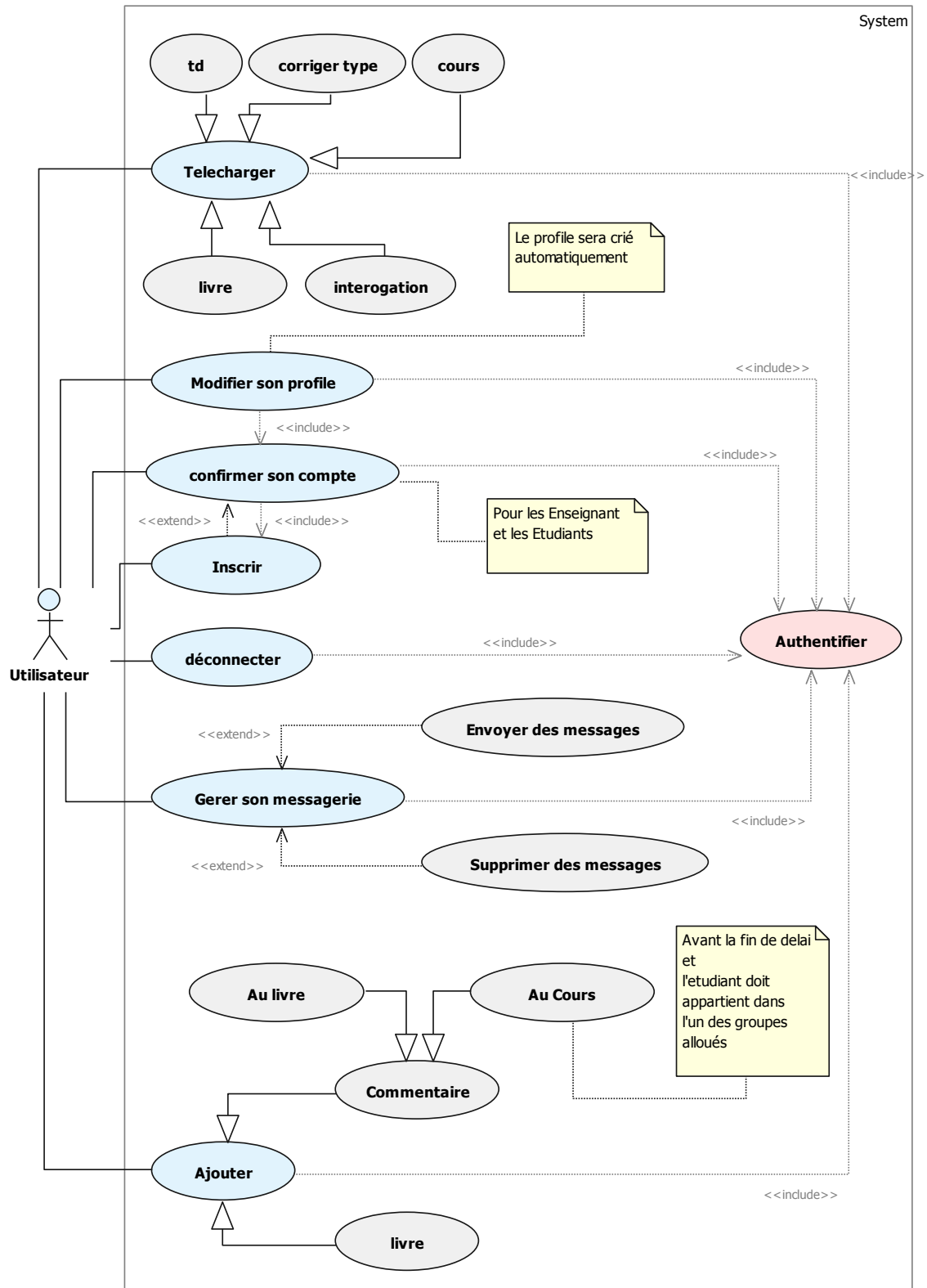
3.2.3. L'administrateur



3.2.4. L'utilisateur



Remarque : les cas des recherches (TD, Cours, Livre, Interrogation) sont des options de la recherche.



Remarque : le cas d'inscription est pour les visiteurs non inscrits.

Remarque

Il n'est pas possible de décrire tous les scénarios d'un cas d'utilisation, il faudra choisir les principaux cas d'utilisation (ceux qui sont fréquent) et quelques exceptions, pour les cas :

- Les cas d'ajout (cours, TD, interrogation, commentaire, livre, module, filière, groupe, réponse, observation, questionnaire) on a choisi le cas d'ajouter un cours.
- Les cas de modification (cours, TD, interrogation, commentaire, livre, module, filière, profile) on a choisi le cas de modifier un cours.
- Les cas de suppression (cours, TD, interrogation, commentaire, livre, module, filière, groupe, réponse, observation, questionnaire, utilisateur) on a choisi le cas de supprimer un cours.
- Les cas de validation (Etudiant, livre, groupe) on a choisi le cas valider un livre.
- Les cas d'impression (cours, TD, interrogation, livre) on a choisi le cas d'imprimer un cours.
- Les cas de téléchargement (cours, TD, interrogation, livre, CV) on a choisi le cas de télécharger un cours.

4. DESCRIPTION TEXTUEL DES CAS D'UTILISATIONS

4.1. Introduction

Le diagramme de cas d'utilisation décrit les grandes fonctions d'un système du point de vue des acteurs, mais n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation. Bien que de nombreux diagrammes d'UML permettent de décrire un cas, il est recommandé de rédiger une description textuelle car c'est une forme souple qui convient dans bien des situations.

Une description textuelle couramment utilisée se compose de trois parties.

1. La première partie permet d'identifier le cas, elle doit contenir les informations qui suivent : Nom du cas d'utilisation, Acteurs principaux, Acteurs secondaire, Résumé.
2. La deuxième partie contient la description du fonctionnement du cas sous la forme d'une séquence de messages échangés entre les acteurs et le système. Elle contient : Les pré-conditions, Des scénarios nominaux et alternatifs, Des post-conditions.
3. La troisième partie de la description d'un cas d'utilisation est une rubrique optionnelle. Elle contient généralement des spécifications non fonctionnelles (spécifications techniques, ...).

4.2. L'utilisateur

4.2.1. Cas d'utilisation: inscrire

Cas d'utilisation :	inscrire
Acteurs Principaux :	L'étudiant, L'enseignant, L'administrateur
Acteurs Secondaire :	aucuns
Résumé :	Pour accéder aux services de l'application il faut inscrire.
Pré-condition :	<ul style="list-style-type: none"> a) Session fermée b) l'utilisateur choisit l'option 'inscrire'
Post-condition :	- l'utilisateur sera inscrit dans l'application
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système affiche un formulaire d'inscription 2. l'utilisateur saisit ses informations. 3. l'utilisateur envoie ses informations. 4. le système vérifie le formulaire. 5. le système vérifie l'existence de nom d'utilisateur ou l'email. 6. le système ajoute l'utilisateur. 7. Le système envoie le code de confirmation à l'email de l'utilisateur. 8. le système affiche un champ pour saisir le code de confirmation.
Scenarios Alternatif :	<p>l'enseignant annule l'opération: L'enchaînement démarre au point 1 jusqu'à 8 scénarios nominaux.</p> <ol style="list-style-type: none"> 9. Le système affiche la page d'index <p>formulaire invalide:</p> <ol style="list-style-type: none"> 10. L'enchaînement démarre au point 4 du scénario nominal. 11. le système affiche l'erreur - Le scénario nominal reprend au point 1. <p>email ou nom d'utilisateur déjà existe: L'enchaînement démarre au point 5 du scénario nominal.</p> <ol style="list-style-type: none"> 12. le système affiche l'erreur - Le scénario nominal reprend au point 1.

4.2.2. Cas d'utilisation : s'authentifier

Cas d'utilisation :	s'authentifier
Acteurs Principaux	L'étudiant, L'enseignant, L'administrateur
Acteurs Secondaire	aucuns
Résumé :	l'utilisateur de l'application peut s'authentifier pour accéder à ses services
Pré-condition :	<ul style="list-style-type: none"> a) Session fermée b) l'utilisateur choisit l'option 's'authentifier'
Post-condition :	session sera ouverts
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système affiche un formulaire d'authentification 2. l'utilisateur saisit son nom et mot de passe. 3. l'utilisateur envois son nom et mot de passe. 4. le system vérifie le formulaire. 5. le système vérifie l'existence de nom d'utilisateur et le mot de passe. 6. le système vérifie si son compte est vérifié. 7. système ouvre la session. 8. le système affiche la page d'index.
Scenarios Alternatif :	<p>l'enseignant annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ul style="list-style-type: none"> - l'opération sera annulée <p>9. le système affiche la page d'index.</p> <p>formulaire invalide:</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <p>10. le system affiche l'erreur</p> <ul style="list-style-type: none"> - Le scénario nominal reprend au point 1. <p>utilisateur introuvable:</p> <p>L'enchaînement démarre au point 5 du scénario nominal.</p> <p>11. le système affiche l'erreur</p> <ul style="list-style-type: none"> - Le scénario nominal reprend au point 1. <p>Compte non confirmer:</p> <p>L'enchaînement démarre au point 6 du scénario nominal.</p> <p>12. le system affiche un champ pour saisir le code de confirmation.</p> <ul style="list-style-type: none"> - Le scénario nominal reprend au point 1.

4.2.3. Cas d'utilisation : confirmer son compte

Cas d'utilisation :	Confirmer son compte
Acteurs Principaux :	L'étudiant, l'enseignant
Acteurs Secondaire :	aucuns
Résumé :	L'utilisateur peut confirmer son compte pour être prêt à l'utilisé
Pré-condition :	<ul style="list-style-type: none"> a) le compte de l'utilisateur n'est pas confirmé. b) L'utilisateur choisit l'option 'authentifier'
Post-condition :	son compte sera confirmé.
Scenarios Nominal :	<ul style="list-style-type: none"> 1. le système affiche un champ pour saisir le code de confirmation 2. l'utilisateur saisit le code de confirmation. 3. l'utilisateur envoyer le code de confirmation 4. Le système vérifie le code. 5. le système ouvre la session. 6. le système une page pour indiquer que l'opération est réussite
Scenarios Alternatif :	<p>l'enseignant annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ul style="list-style-type: none"> 7. le système affiche la page d'index <p>Code de confirmation erroné:</p> <p>L'enchaînement démarre au point3 du scénario nominal.</p> <ul style="list-style-type: none"> 8. le system affiche l'erreur - Le scénario nominal reprend au point 1.

4.2.4. Cas d'utilisation : télécharger (cours, livre, td, interrogation, corrigé type)

Cas d'utilisation :	imprimer
Acteurs Principaux :	L'étudiant, l'enseignant, l'administrateur
Acteurs Secondaire :	aucuns
Résumé :	L'utilisateur peut télécharger le contenu de cours, td, interrogation, corrigé type
Pré-condition :	<p>A. Livre :</p> <p>a) Consulter le livre.</p> <p>B. interrogation :</p> <p>a) consulter le cours</p> <p>C. corrigé type :</p> <p>a) Consulter le livre.</p> <p>D. td :</p> <p>a) consulter le cours</p> <p>- l'utilisateur choisit l'option 'télécharger'</p>
Post-condition :	- le téléchargement sera démarré.
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système recherche sur le fichier correspondant. 2. le system envoi le fichier au navigateur. 3. l'utilisateur démarre le téléchargement.
Scenarios Exceptionnel:	<p>Le fichier est endommagé :</p> <p>L'enchaînement démarre au point 1 du scénario nominal.</p> <ol style="list-style-type: none"> 4. le system affiche l'erreur

4.2.5. Cas d'utilisation : imprimer (cours, livre, td, interrogation, corrigé type)

Cas d'utilisation :	imprimer
Acteurs Principaux :	L'étudiant, l'enseignant, l'administrateur
Acteurs Secondaire :	aucuns
Résumé :	L'utilisateur peut imprimer le contenu de livre, td, cours, interrogation, corrigé type
Pré-condition :	<p>A. Livre :</p> <p>a) Consulter le livre.</p> <p>B. interrogation :</p> <p>a) consulter l'interrogation.</p> <p>C. corrigé type :</p> <p>a) Consulter le corrigé type.</p> <p>D. td :</p> <p>a) consulter le td.</p> <p>E. cours :</p> <p>a) consulter le cours.</p> <p>- l'utilisateur choisit l'option 'imprimer'</p>
Post-condition :	- l'impression sera démarrée.
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système recherche sur le fichier correspondant. 2. le system envoi le fichier au navigateur. 3. l'utilisateur démarre l'impression.
Scenarios Exceptionnel:	<p>Le fichier est endommagé :</p> <p>L'enchaînement démarre au point 1 du scénario nominal.</p> <ol style="list-style-type: none"> 4. le system affiche l'erreur

4.2.6. Cas d'utilisation : rechercher (cours, livre, td, interrogation)

Cas d'utilisation :	rechercher
Acteurs Principaux :	L'étudiant, l'enseignant, l'administrateur
Acteurs Secondaire :	aucuns
Résumé :	L'utilisateur peut rechercher sur un livre, td, cours, interrogation
Pré-condition :	<ul style="list-style-type: none"> - session ouverte - l'utilisateur choisit l'option 'rechercher' <p>A. Livre :</p> <ul style="list-style-type: none"> a) choisit l'option 'livre' <p>B. interrogation :</p> <ul style="list-style-type: none"> a) choisit l'option 'interrogation' <p>C. td :</p> <ul style="list-style-type: none"> a) choisit l'option 'td' <p>D. cours :</p> <ul style="list-style-type: none"> a) choisit l'option 'cours'
Post-condition :	<ul style="list-style-type: none"> - le résultat de recherche sera affiché.
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système affiche un champ de recherche. 2. l'utilisateur saisir le mot de recherche. 3. l'utilisateur envoi le mot de recherche. 4. le système vérifie le mot de recherche saisit 5. le système vérifie l'existence des éléments correspondants au mot de recherche. (td, cours, interrogation, livre) 6. le système affiche le résultat.
Scenarios Exceptionnel:	<p>L'utilisateur annule l'opération:</p> <p>L'enchaînement démarre au point 3 du scénario nominal.</p> <ul style="list-style-type: none"> - L'opération sera annulée. <p>mot de recherche invalide:</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <ol style="list-style-type: none"> 7. le system affiche l'erreur <ul style="list-style-type: none"> - Le scénario nominal reprend au point 1.

4.2.7. Cas d'utilisation : participer dans le service de chat.

Cas d'utilisation :	participer dans le service de chat.
Acteurs Principaux :	L'étudiant, l'enseignant, l'administrateur
Acteurs Secondaire :	aucuns
Résumé :	L'utilisateur peut accéder et participer dans le service de chat.
Pré-condition :	a) Session ouverte. b) l'utilisateur choisit l'option 'chat'
Post-condition :	
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système affiche la liste de l'utilisateur dans l'espace de chat. 2. l'utilisateur choisit un des utilisateurs dans la liste. 3. le system affiche un formulaire de chat. 4. l'utilisateur démarre l'envoi et la réception du message.
Scenarios Exceptionnel:	<p>L'utilisateur déconnecté :</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <ol style="list-style-type: none"> 5. le système affiche un message d'erreur. <ul style="list-style-type: none"> - Le scénario nominal reprend au point 1. <p>L'utilisateur annule l'opération:</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <ol style="list-style-type: none"> 6. le system affiche la page d'accueil. <p>List vide:</p> <p>L'enchaînement démarre au point 1 du scénario nominal.</p> <ol style="list-style-type: none"> 7. le system affiche la liste vide.

4.2.8. Cas d'utilisation : quitter l'application.

Cas d'utilisation :	quitter l'application.
Acteurs Principaux :	L'étudiant, l'enseignant, l'administrateur
Acteurs Secondaire :	aucuns
Résumé :	L'utilisateur peut quitter l'application.
Pré-condition :	a) Session ouverte. b) l'utilisateur choisit l'option 'quitter'
Post-condition :	
Scenarios Nominal :	<ol style="list-style-type: none">1. le système affiche une demande de confirmation.2. l'utilisateur envoie la confirmation.3. le système ferme la session de l'utilisateur.4. le système affiche la page d'accueil.
Scenarios Exceptionnel:	L'utilisateur annule l'opération: L'enchaînement démarre au point 2 du scénario nominal. - L'opération sera annulée.

4.2.9. Cas d'utilisation : consulter un cours.

Cas d'utilisation :	consulter un cours
Acteurs Principaux :	l'administrateur, l'enseignant, l'étudiant
Acteurs Secondaire :	Aucuns
Résumé :	l'utilisateur peut consulter un cours
Pré-condition :	a) l'utilisateur consulte le module b) la List des cours de ce module n'est pas vide
Post-condition :	- le cours sera affiché
Scenarios Nominal :	1. l'utilisateur sélectionne un cours. 2. le system affiche le contenue de cours.

4.3. L'enseignant

4.3.1. Cas d'utilisation : ajouter un cours.

Cas d'utilisation :	ajouter un cours.
Acteurs Principaux :	L'enseignant
Acteurs Secondaire :	Aucuns
Résumé :	l'enseignant peut ajouter des nouveaux cours à l'un de ses modules.
Pré-condition :	<ul style="list-style-type: none"> a) l'enseignant consulte un de ses modules b) l'enseignant choisit l'option 'ajouter cours'.
Post-condition :	- Le cours sera ajouté et affiché.
Scenarios Nominal :	<ul style="list-style-type: none"> 1. le système affiche un formulaire d'ajout de cours. 2. L'enseignant remplit le formulaire. 3. L'enseignant envoie les informations de cours. 4. le system vérifie le formulaire. 5. le système ajoute le cours. 6. le système affiche le nouveaux cours.
Scenarios Alternatif :	<p>l'enseignant annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ul style="list-style-type: none"> 7. le système affiche le module. <p>formulaire invalide:</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <ul style="list-style-type: none"> 8. le system affiche l'erreur - Le scénario nominal reprend au point 1.

4.3.2. Cas d'utilisation : supprimer un cours.

Cas d'utilisation :	supprimer un cours.
Acteurs Principaux :	L'enseignant
Acteurs Secondaire :	Aucuns
Résumé :	l'utilisateur de l'application peut supprimer un de ses cours.
Pré-condition :	a) l'enseignant consulte un de ses cours b) l'enseignant choisit l'option 'supprimer'
Post-condition :	- le cours sera supprimé.
Scenarios Nominal :	<ol style="list-style-type: none">1. le système affiche une demande de confirmation2. l'utilisateur envoi la confirmation.3. le système supprime le cours.4. le système affiche le module.
Scenarios Alternatif :	l'enseignant annule l'opération: L'enchaînement démarre au point 2 du scénario nominal. <ol style="list-style-type: none">5. le système affiche le contenu de cours.

4.3.3. Cas d'utilisation : noter les réponses

Cas d'utilisation :	noter les réponses
Acteurs Principaux :	L'enseignant
Acteurs Secondaire :	Aucuns
Résumé :	l'enseignant peut noter les réponses des étudiants concernant ses interrogations ou ses TDs
Pré-condition :	<ul style="list-style-type: none"> a) l'enseignant consulte la réponse b) l'enseignant choisit l'option 'noter'
Post-condition :	- la réponse sera notée
Scenarios Nominal :	<ul style="list-style-type: none"> 1. le système affiche champs pour saisir la note. 2. l'enseignant saisit la note. 3. l'enseignant envoie la note. 4. le système vérifie la note. 5. Le système ajoute la note. 6. le système affiche la List des réponses.
Scenarios Alternatif :	<p>l'enseignant annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ul style="list-style-type: none"> 7. le système affiche la List des réponses. <p>note invalide:</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <ul style="list-style-type: none"> 8. le system affiche l'erreur - Le scénario nominal reprend au point 1.

4.3.4. Cas d'utilisation : modifier un cours.

Cas d'utilisation :	modifier
Acteurs Principaux :	L'enseignant
Acteurs Secondaire :	aucuns
Résumé :	L'enseignant peut modifier le contenu de l'un de ses cours.
Pré-condition :	<ul style="list-style-type: none"> a) l'enseignant consulte un de ses cours. b) L'enseignant choisit l'option 'modifier'.
Post-condition :	- La modification sera appliquée au cours.
Scenarios Nominal :	<ul style="list-style-type: none"> 1. le système affiche un formulaire de modification. 2. L'enseignant saisit les modifications. 3. l'utilisateur envoie les modifications. 4. le system vérifie le formulaire. 5. le système appliquer les modifications. 6. le système affiche le cours.
Scenarios Alternatif :	<p>l'enseignant annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ul style="list-style-type: none"> 7. le système affiche le cours. <p>formulaire invalide:</p> <p>L'enchaînement démarre au point 4 du scénario nominal.</p> <ul style="list-style-type: none"> 8. le system affiche l'erreur - Le scénario nominal reprend au point 1.

4.3.5. Cas d'utilisation : valider un étudiant.

Cas d'utilisation :	valider un étudiant.
Acteurs Principaux :	L'enseignant.
Acteurs Secondaire :	L'étudiant.
Résumé :	L'enseignant peut valider des nouveaux étudiants dans l'un de ses groupes
Pré-condition :	<ul style="list-style-type: none"> a) Session ouverte. b) l'enseignant consulte l'un de ses groupes
Post-condition :	- l'étudiant sera validé
Scenarios Nominal :	<ul style="list-style-type: none"> 1. le système affiche la liste des étudiants de groupe. 2. l'enseignant sélectionne un étudiant. 3. l'enseignant choisit l'option 'valider'. 4. le système valide l'étudiant.
Scenarios Exceptionnel:	<p>List vide:</p> <p>L'enchaînement démarre au point 1 du scénario nominal.</p> <ul style="list-style-type: none"> 5. le système affiche une liste vide. <p>L'étudiant sélectionné est déjà validé:</p> <p>L'enchaînement démarre au point 3 du scénario nominal.</p> <ul style="list-style-type: none"> - L'option 'valider' sera être invisible

4.4. L'étudiant**4.4.1. Cas d'utilisation : modifier son groupe.**

Cas d'utilisation :	modifier son groupe
Acteurs Principaux :	L'étudiant
Acteurs Secondaire :	aucuns
Résumé :	l'étudiant peut modifier le contenu de l'un de ses cours.
Pré-condition :	a) l'étudiant consulte son profile. b) L'étudiant choisit l'option 'modifier groupe'.
Post-condition :	- Le groupe de l'étudiant sera modifié.
Scenarios Nominal :	1. affiche la liste des groupes disponibles 2. l'étudiant choisit un groupe. 3. le système change le groupe de l'étudiant. 4. le système affiche le profile.

4.5. L'administrateur

4.5.1. Cas d'utilisation : bloquer un utilisateur

Cas d'utilisation :	bloquer un utilisateur
Acteurs Principaux :	L'administrateur
Acteurs Secondaire :	Aucuns
Résumé :	L'administrateur de l'application peut bloquer un utilisateur
Pré-condition :	<ul style="list-style-type: none">a) session ouverteb) l'administrateur sélectionne un utilisateur d'après la List des utilisateurs
Post-condition :	<ul style="list-style-type: none">- l'utilisateur sera bloqué
Scenarios Nominal :	<ul style="list-style-type: none">1. le système affiche une demande de confirmation2. L'administrateur envois la confirmation.3. Le system bloque l'utilisateur.4. le système affiche la liste des utilisateurs.
Scenarios Alternatif :	<p>I'enseignant annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ul style="list-style-type: none">5. le système affiche la liste les utilisateurs.

4.5.2. Cas d'utilisation : valider (livre / groupe).

Cas d'utilisation :	Valider.
Acteurs Principaux :	l'administrateur
Acteurs Secondaire :	aucuns
Résumé :	L'administrateur peut valider des groupes ou des livres.
Pré-condition :	<ul style="list-style-type: none"> - Session ouverte. A. livre <ul style="list-style-type: none"> a) l'administrateur consulter le livre B. livres. C. groupe <ul style="list-style-type: none"> a) l'administrateur consulter le groupe. - l'administrateur choisit l'option 'valider'
Post-condition :	<ul style="list-style-type: none"> - Livre / groupe sera validé
Scenarios Nominal :	<ol style="list-style-type: none"> 1. le système affiche une demande de confirmation. 2. l'utilisateur envoie la confirmation. 3. le système valide le livre / groupe. 4. le système affiche la liste des livre / groupe.
Scenarios Exceptionnel:	<p>L'utilisateur annule l'opération:</p> <p>L'enchaînement démarre au point 2 du scénario nominal.</p> <ol style="list-style-type: none"> 5. le système affiche la liste des livre / groupe.

6. LES DIAGRAMMES DE SEQUENCES

6.1. Introduction

Les principales informations contenues dans un diagramme de séquence sont les messages échangés entre les lignes de vie, présentés dans un ordre chronologique. Ainsi, contrairement au diagramme de communication, le temps y est représenté explicitement par une dimension (la dimension verticale) et s'écoule de haut en bas.

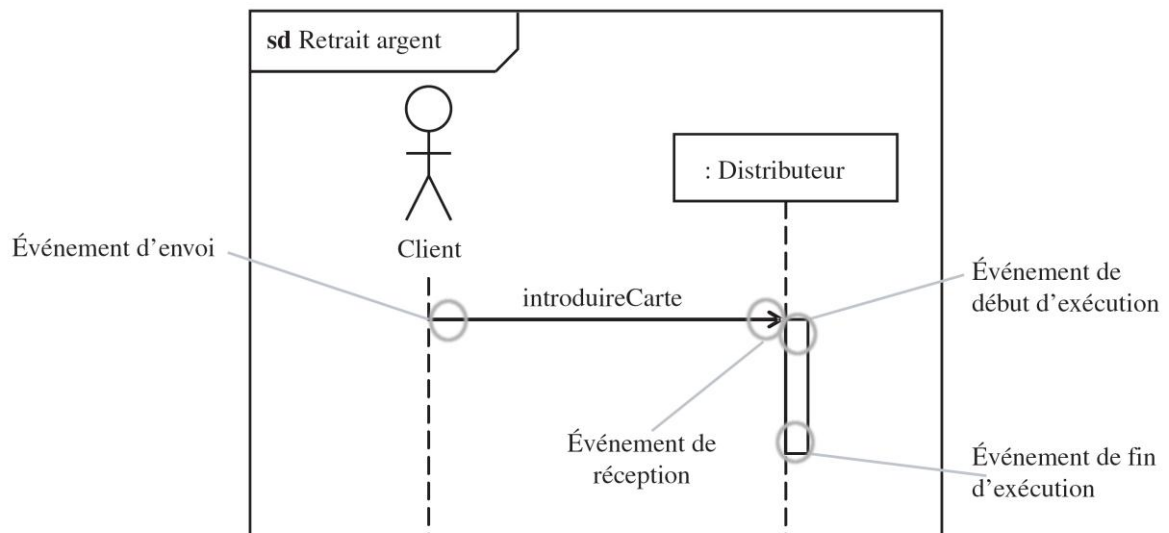
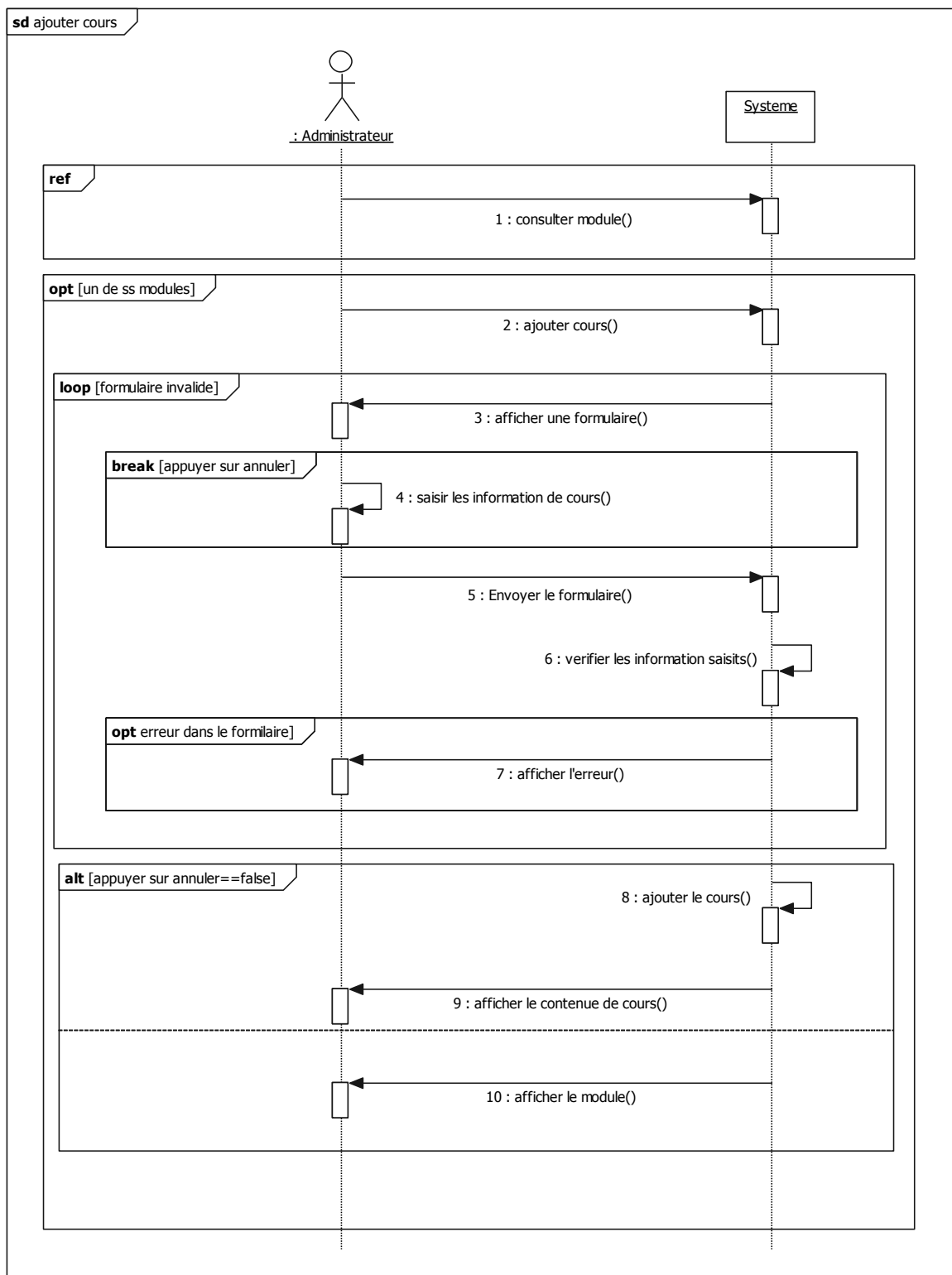


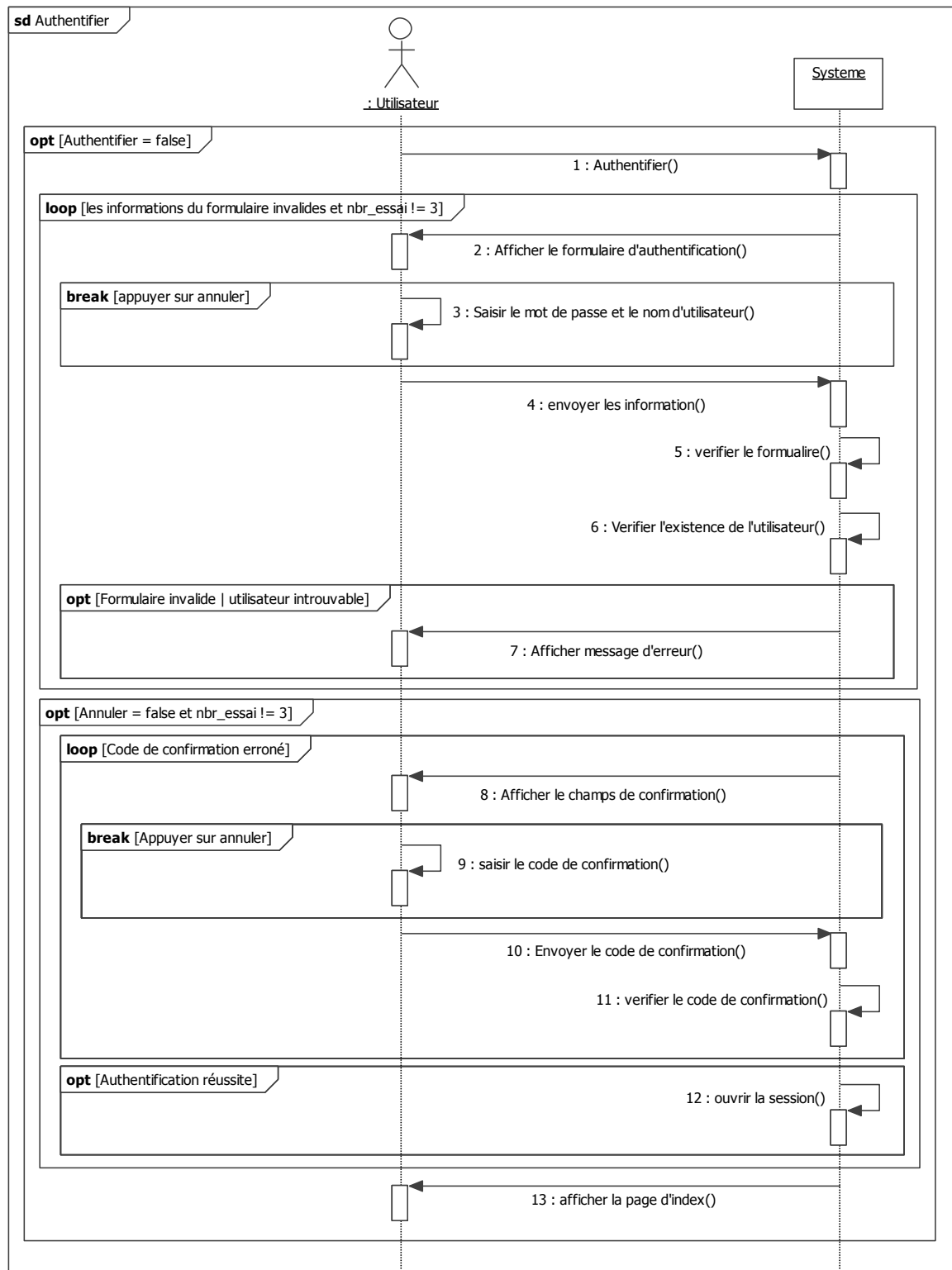
Figure 7 : Les différents évènements correspondant à un message asynchrone.

6.2. Les diagrammes

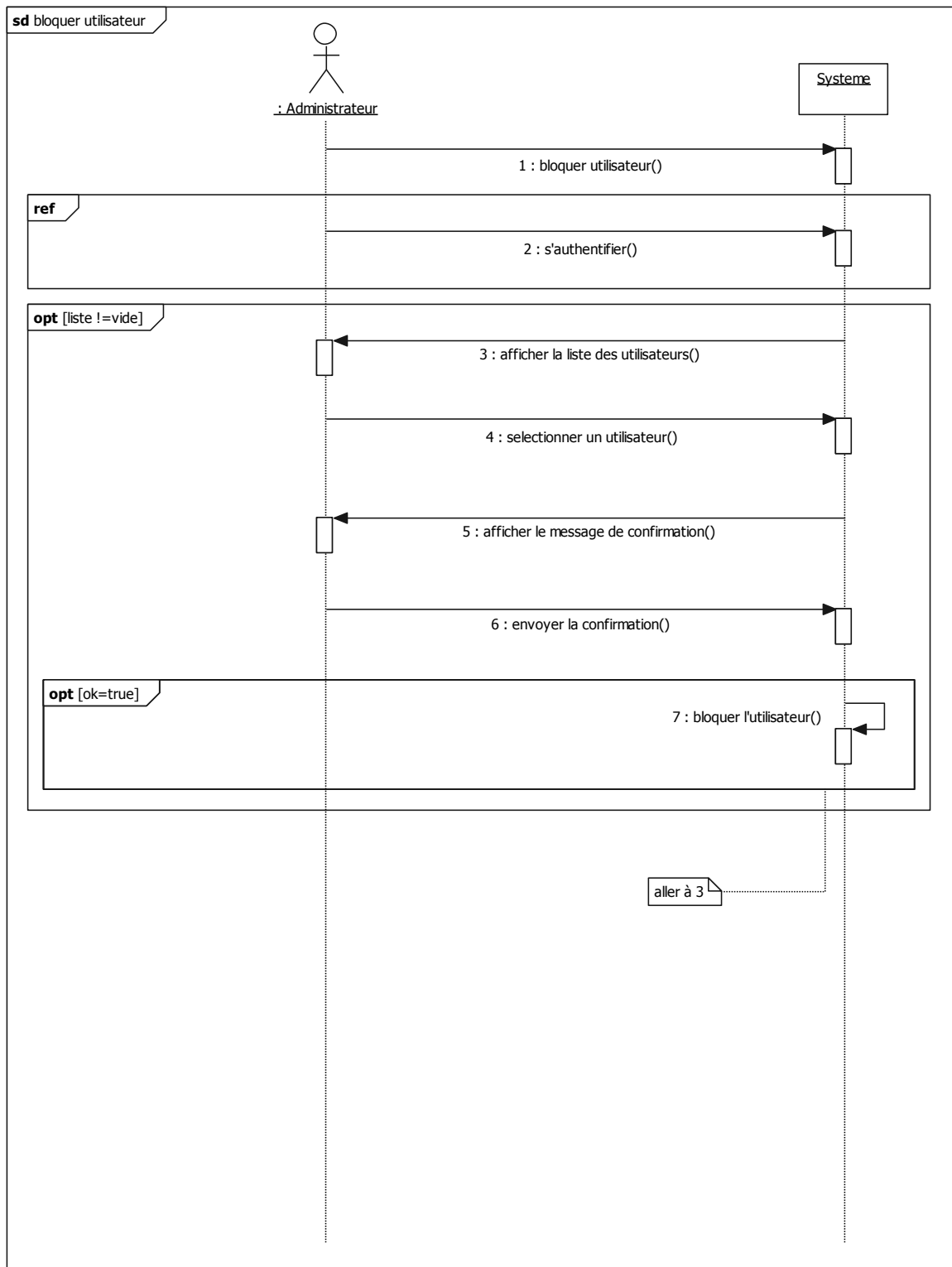
6.2.1. Ajouter un cours



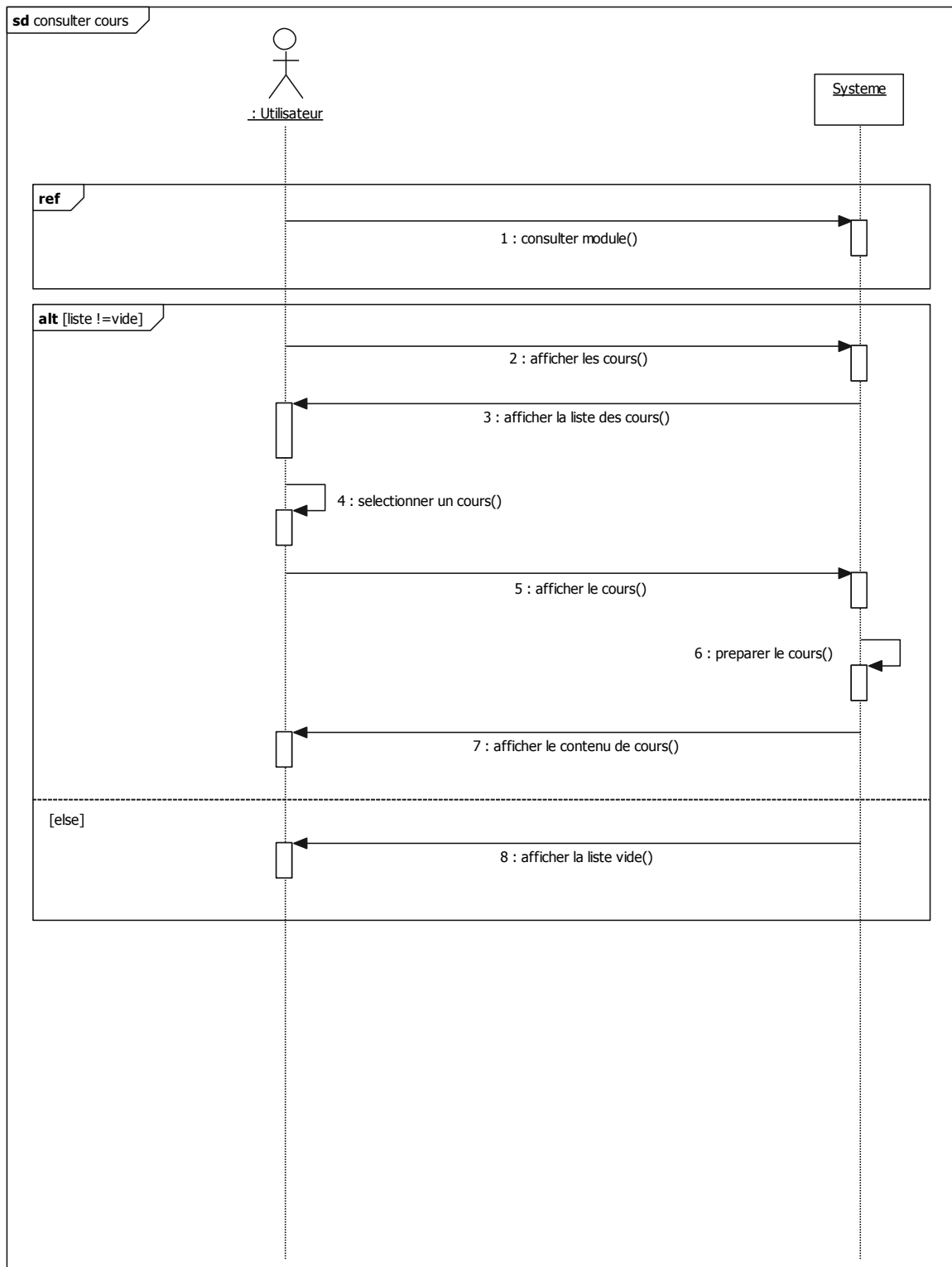
6.2.2. Authentifier



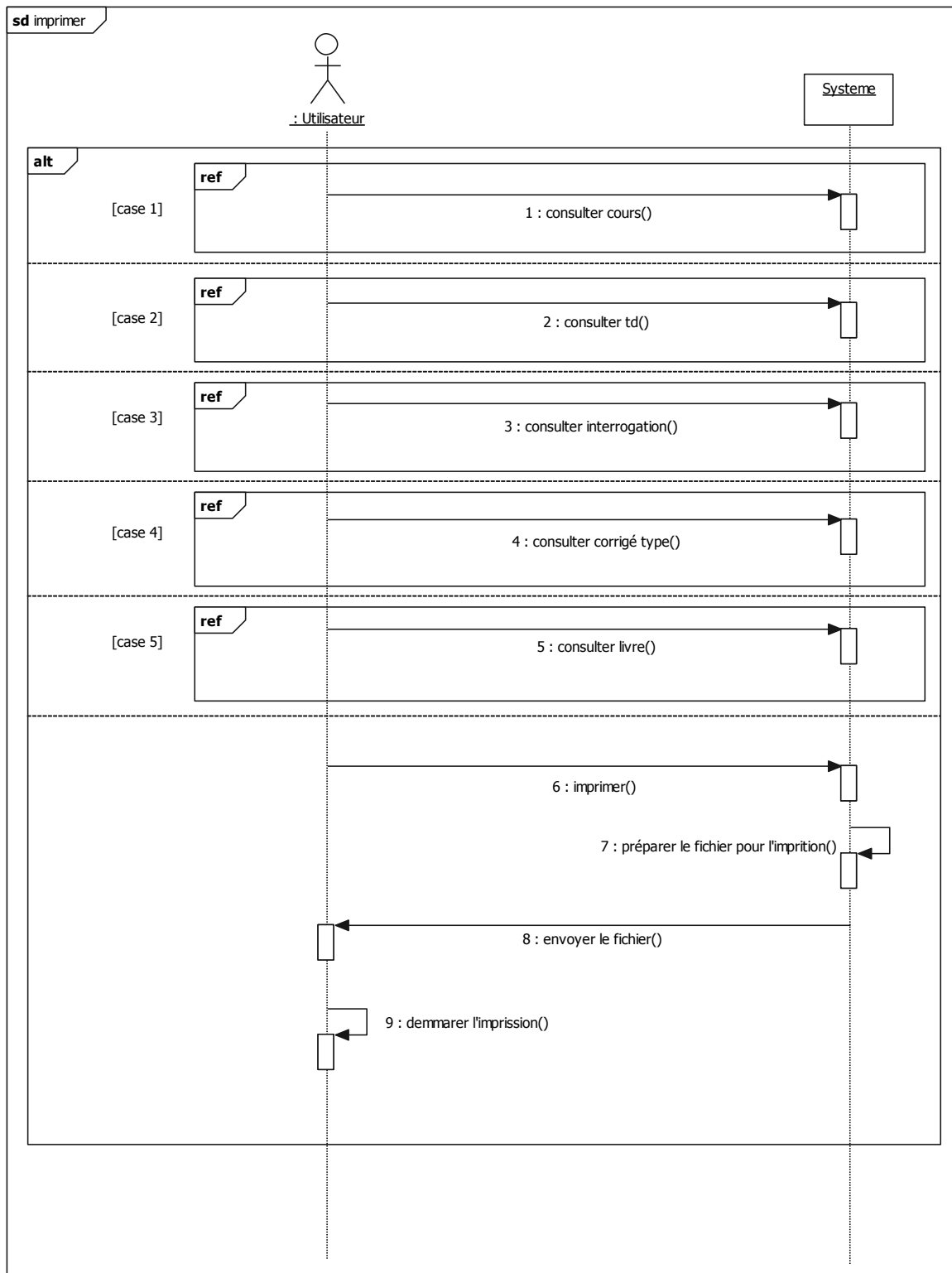
6.2.3. Bloquer utilisateur



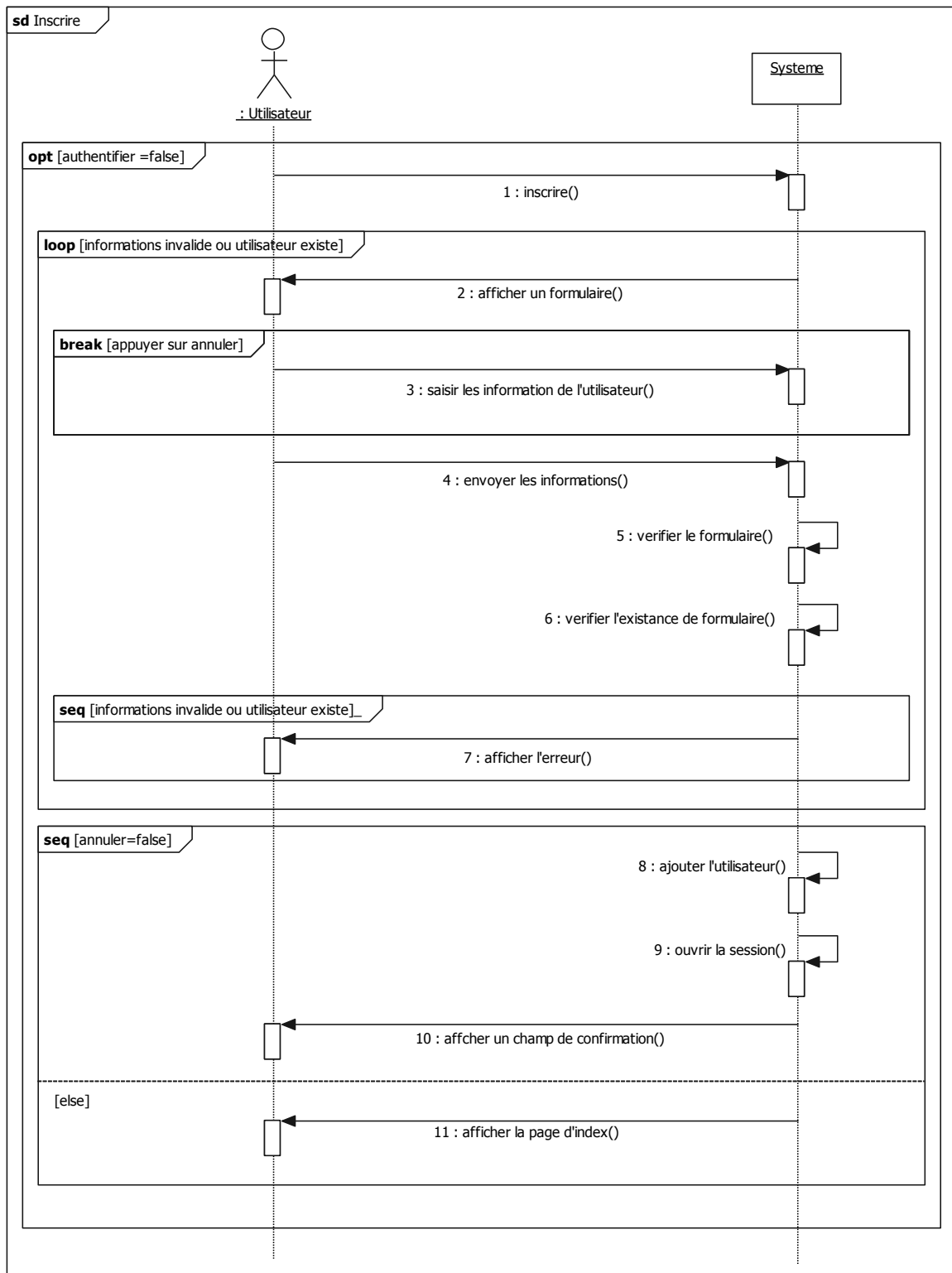
6.2.4. Consulter un cours



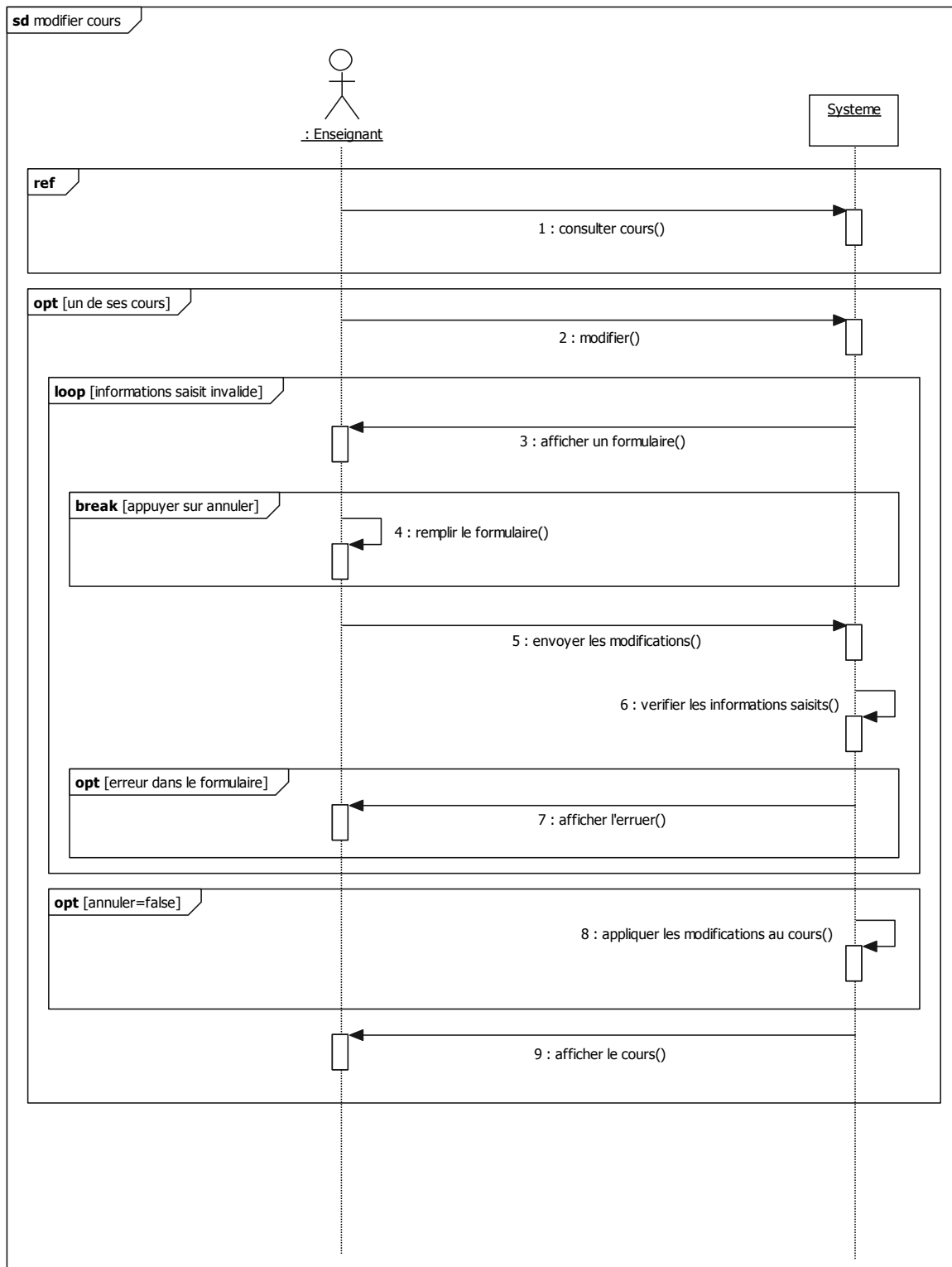
6.2.5. Imprimer



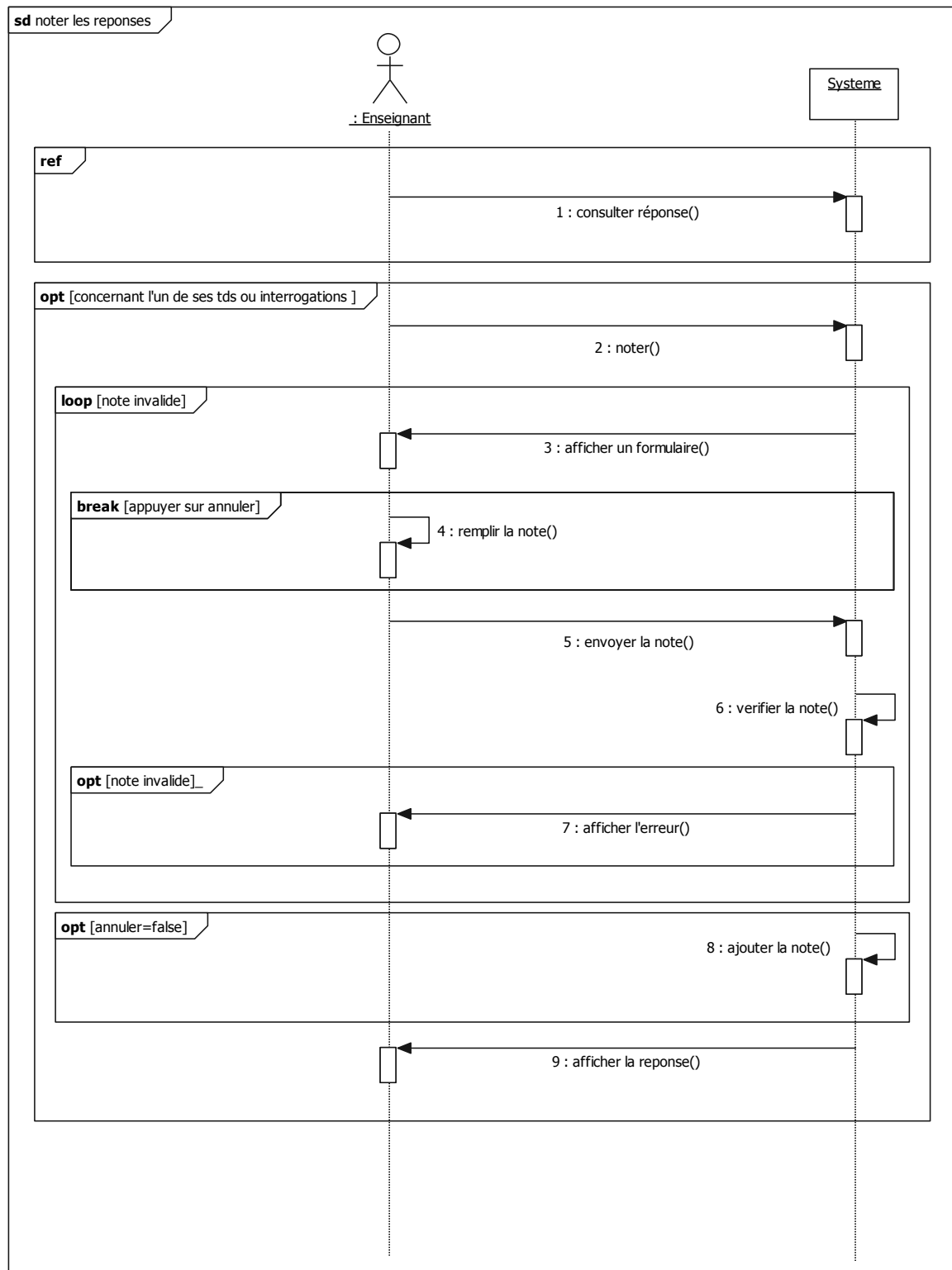
6.2.6. Inscrire



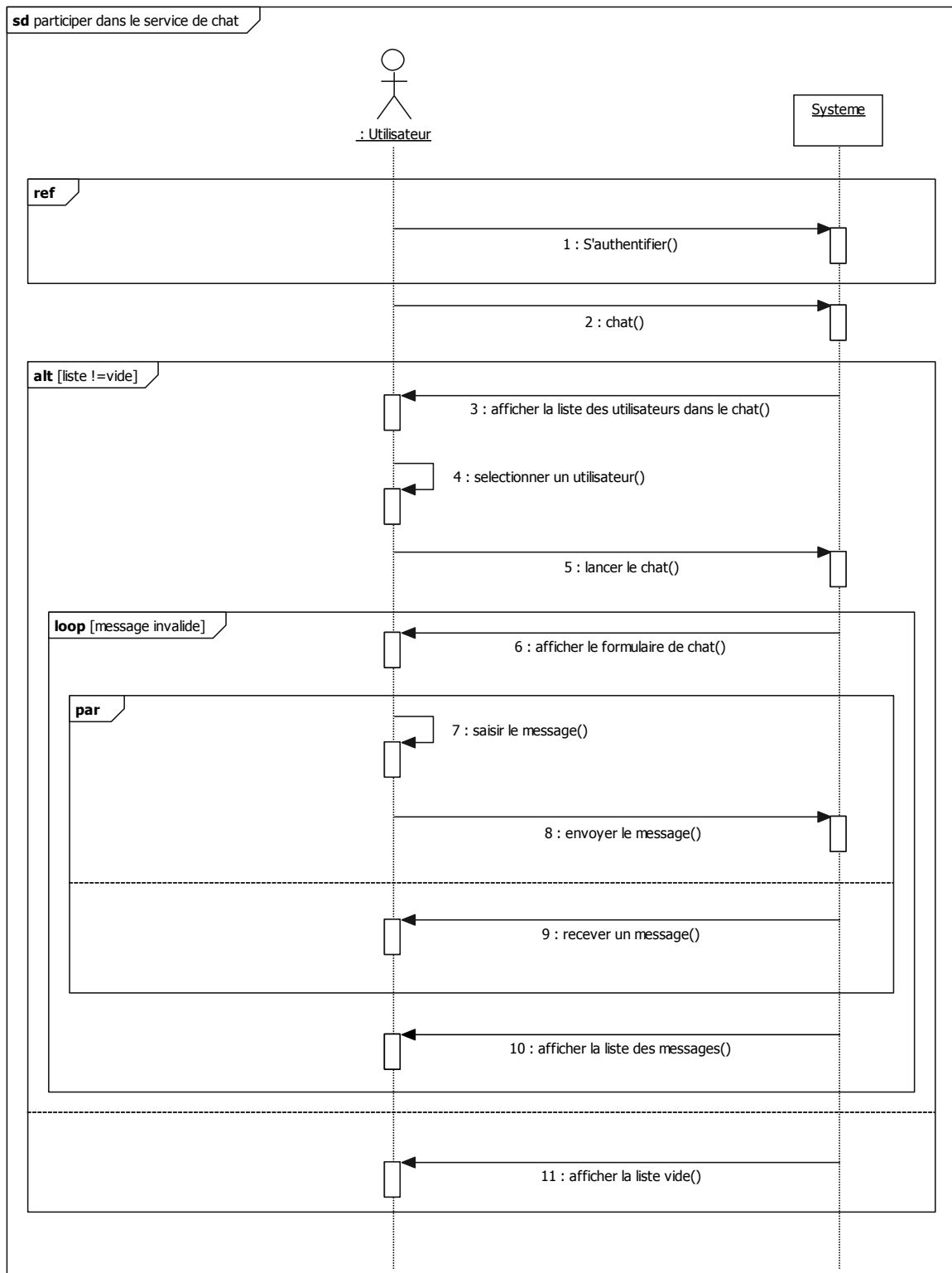
6.2.7. Modifier un cours



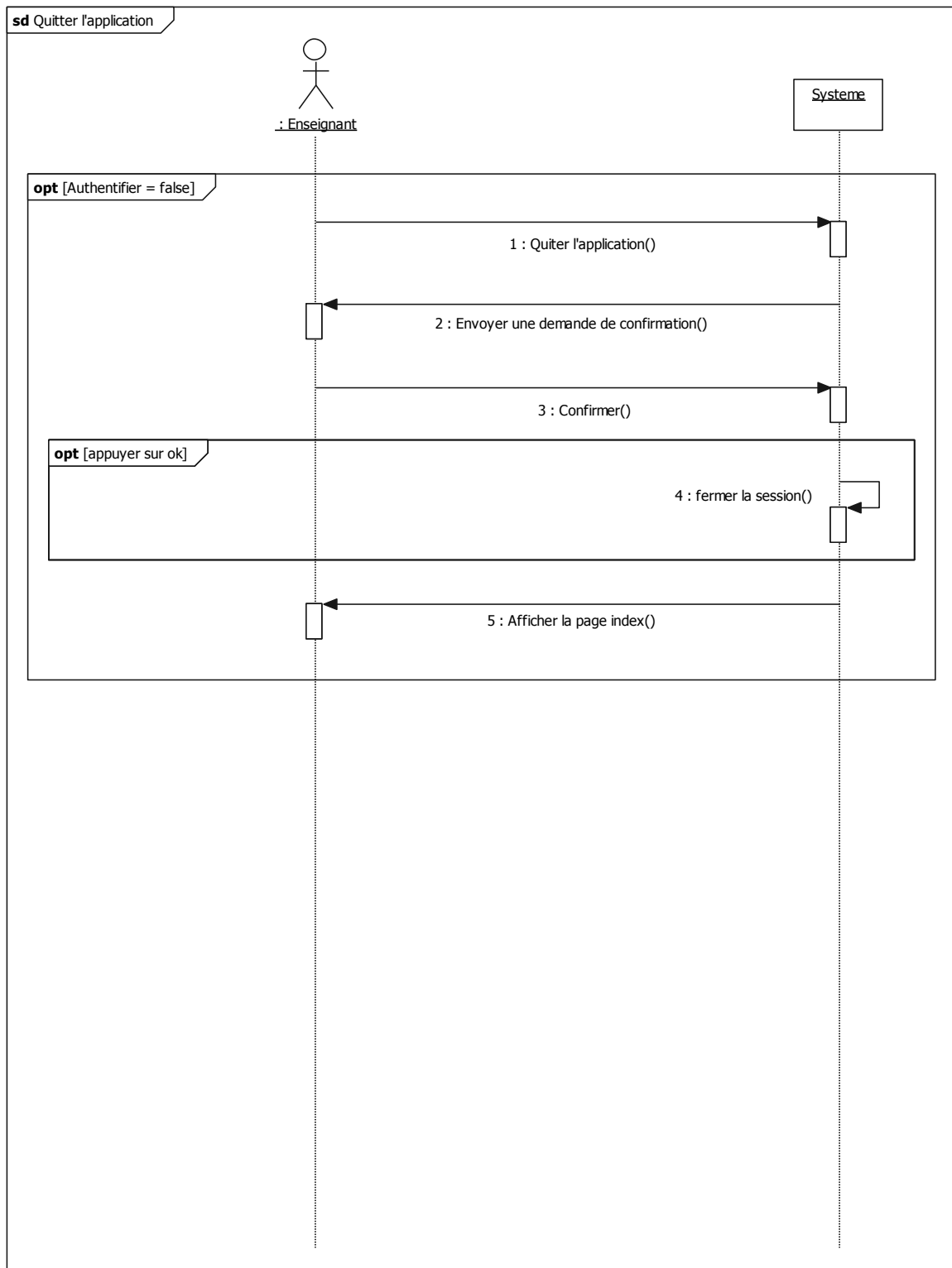
6.2.8. Noter les réponses



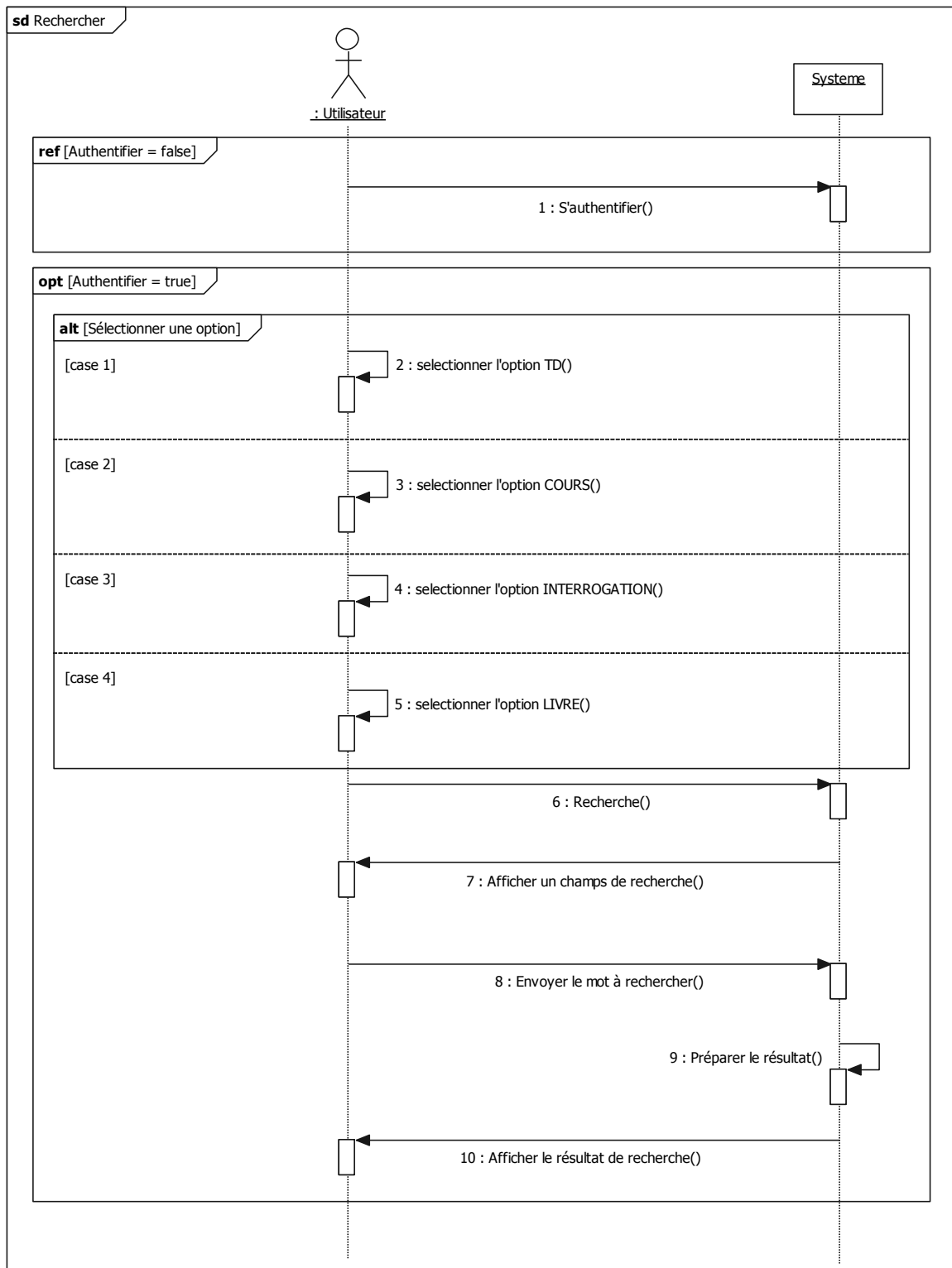
6.2.9. Participer dans le service de chat



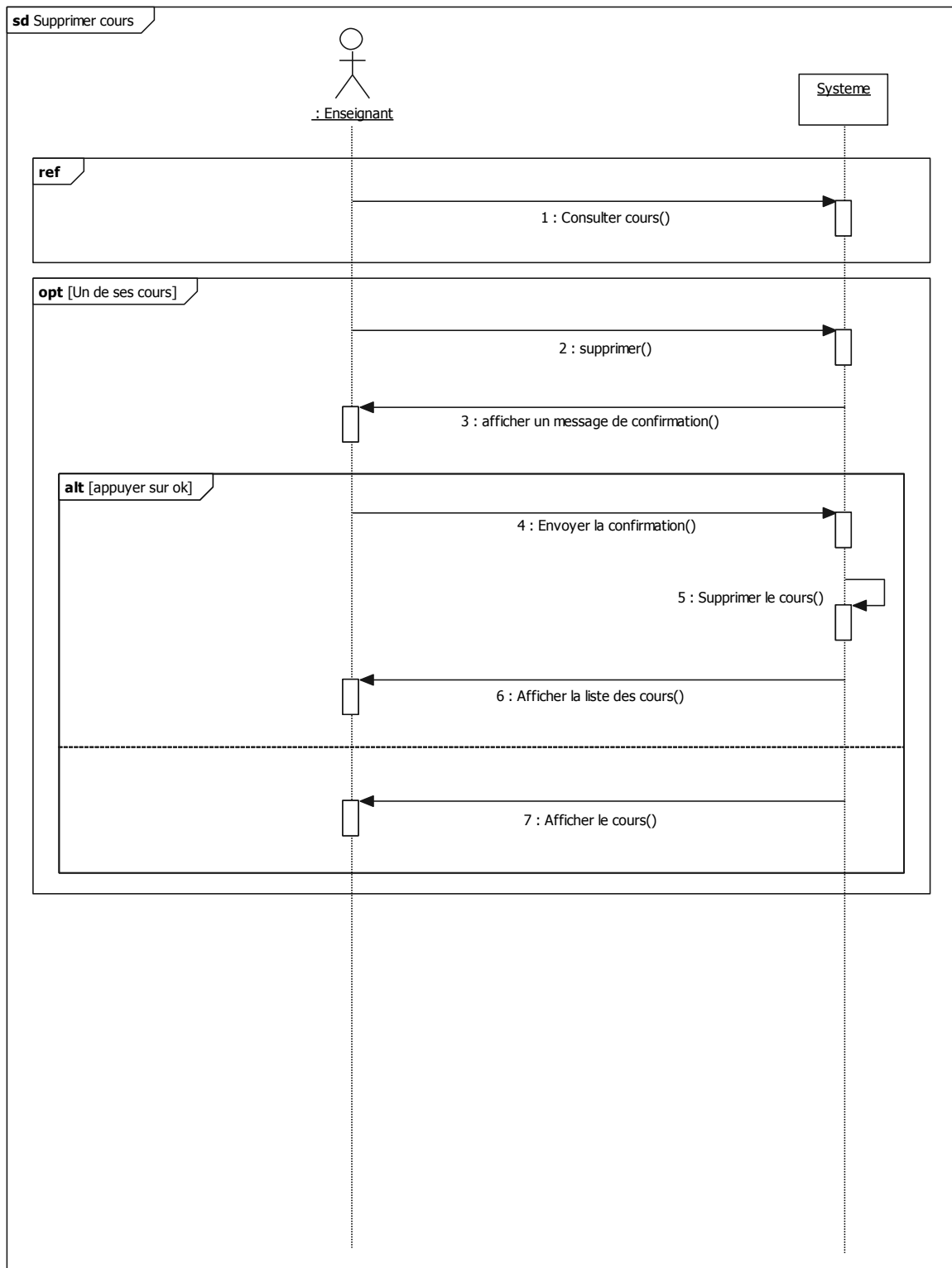
6.2.10. Quitter l'application



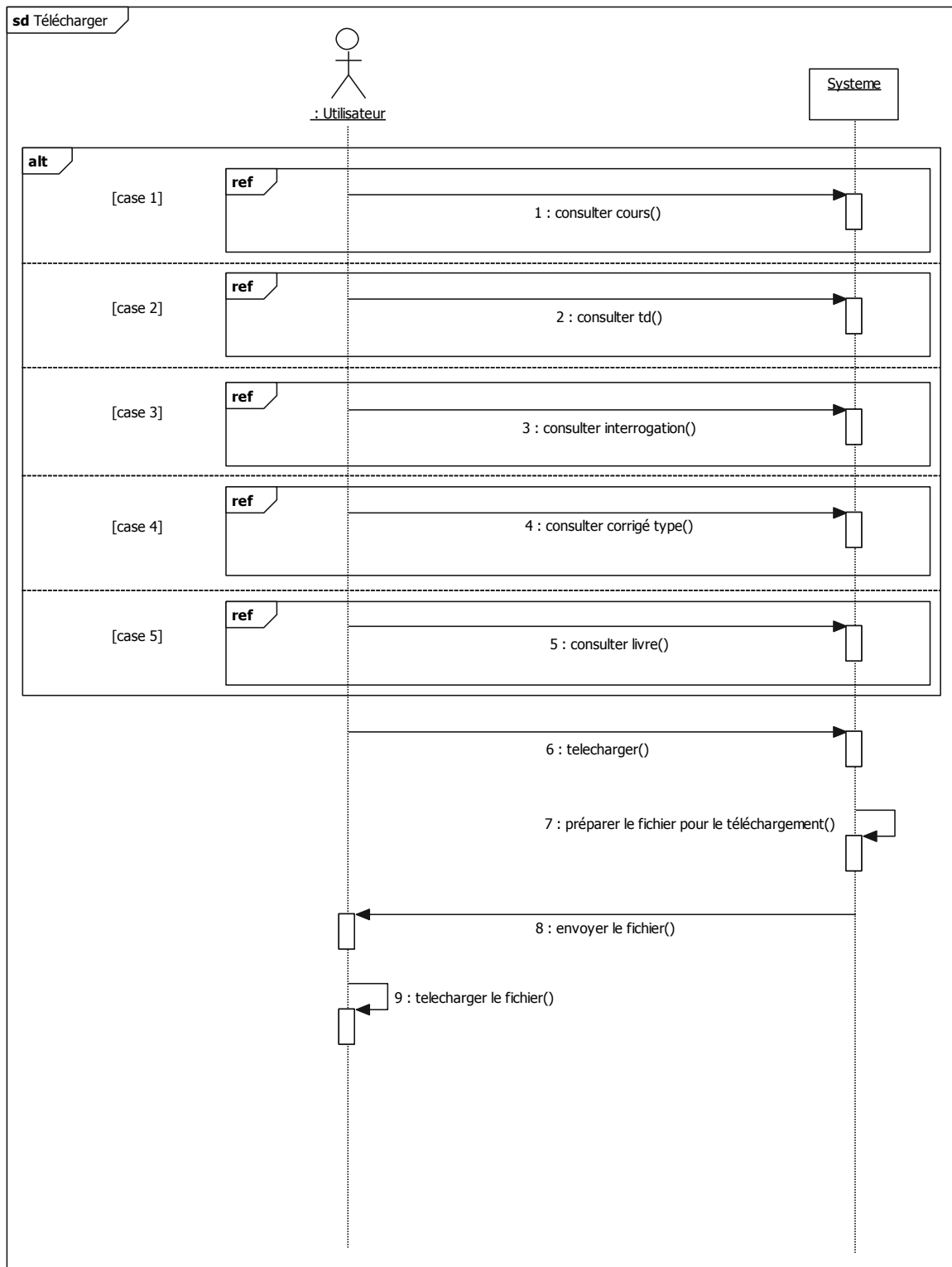
6.2.11. Rechercher



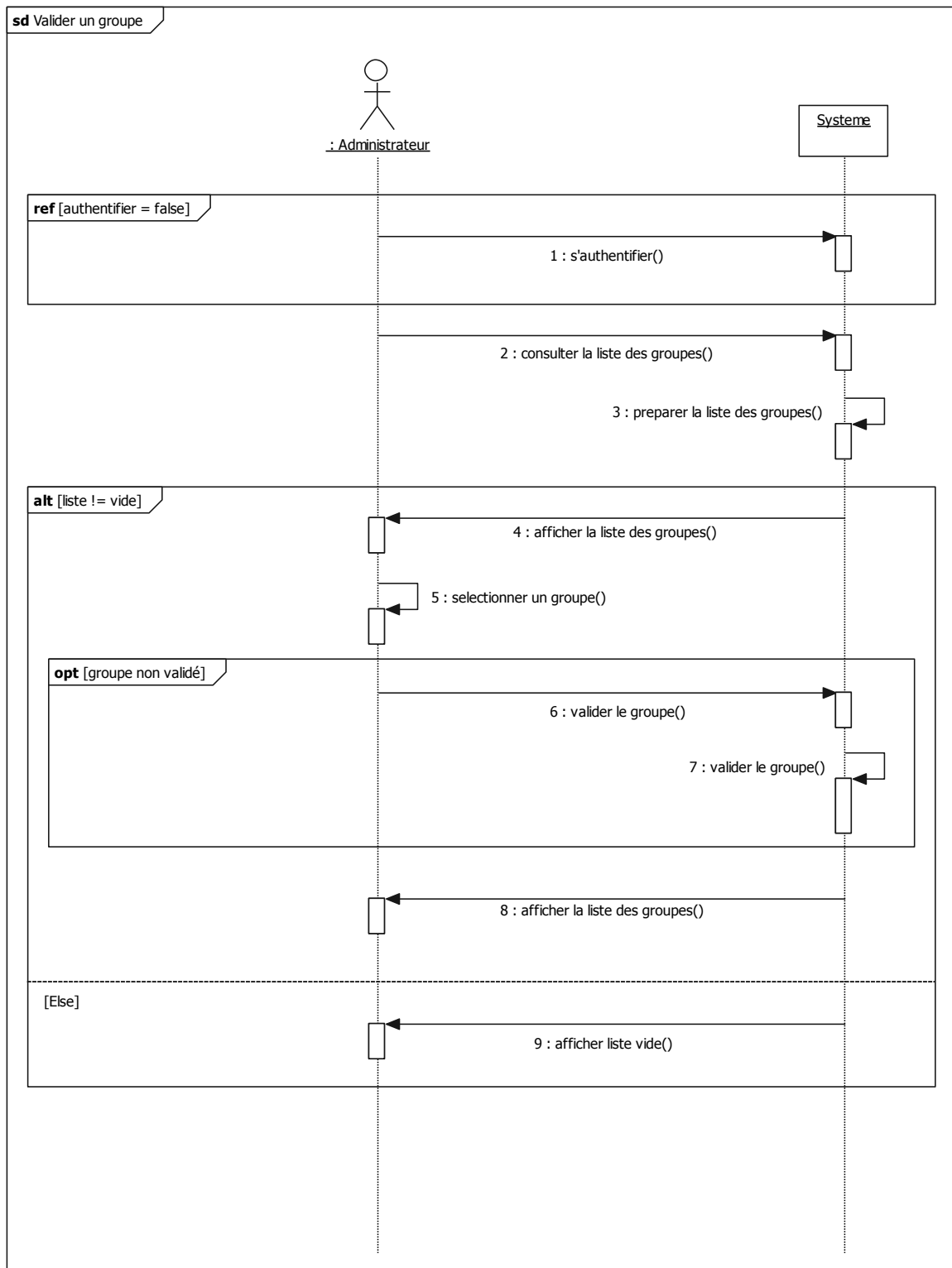
6.2.12. Supprimer un cours



6.2.13. Télécharger



6.2.14. Valider un groupe



7. LES DIAGRAMMES D'ACTIVITES

7.1. Introduction

Les diagrammes d'activités permettent de mettre l'accent sur les traitements. Ils sont donc particulièrement adaptés à la modélisation du cheminement de flots de contrôle et de flots de données. Ils permettent ainsi de représenter graphiquement le comportement d'une méthode ou le déroulement d'un cas d'utilisation.

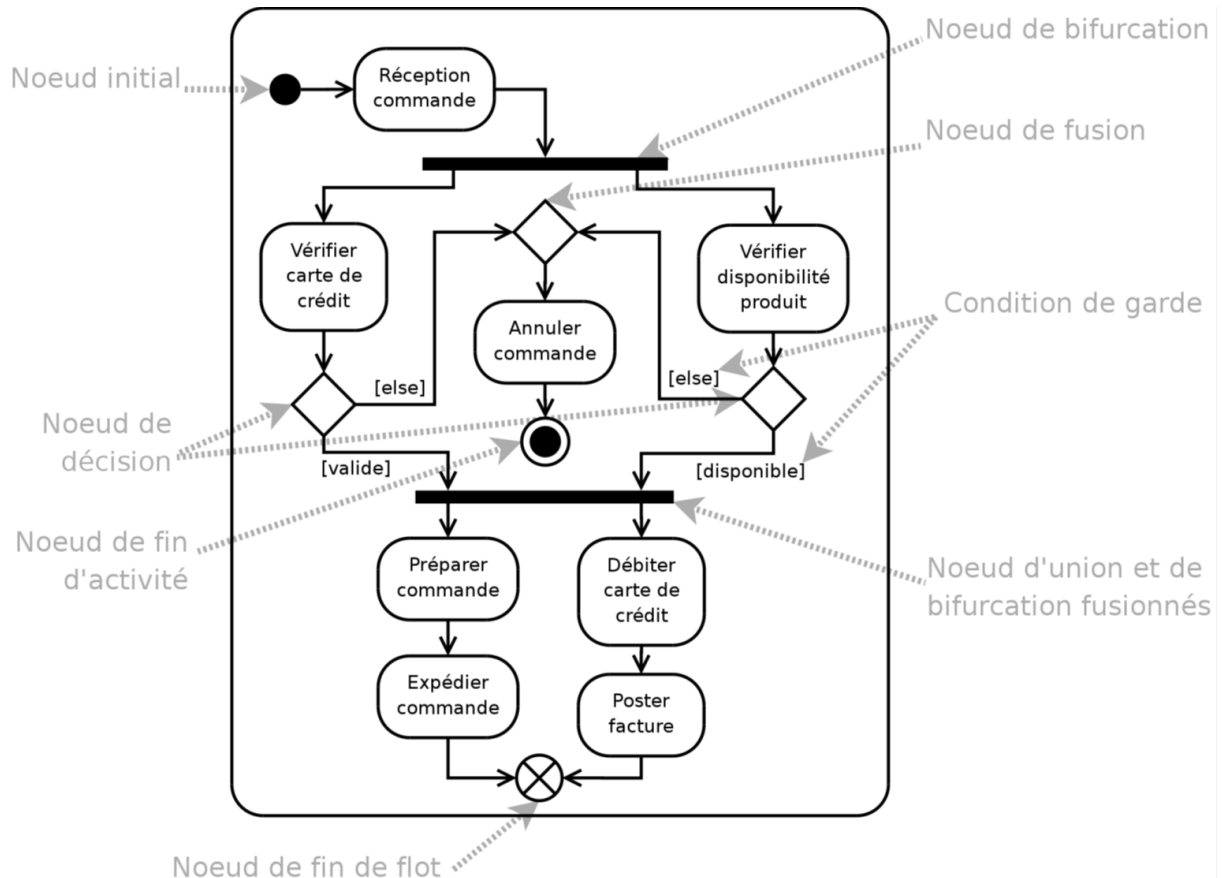


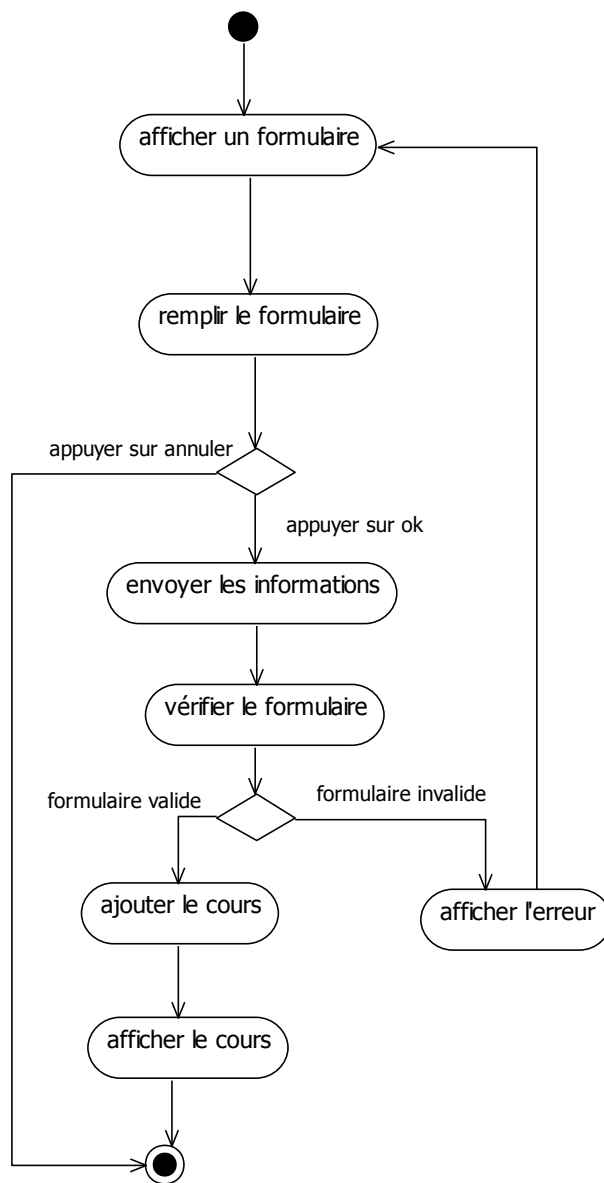
Figure 8 : Exemple de diagramme d'activité illustrant l'utilisation de nœuds de contrôle.

Ce diagramme décrit la prise en compte d'une commande.

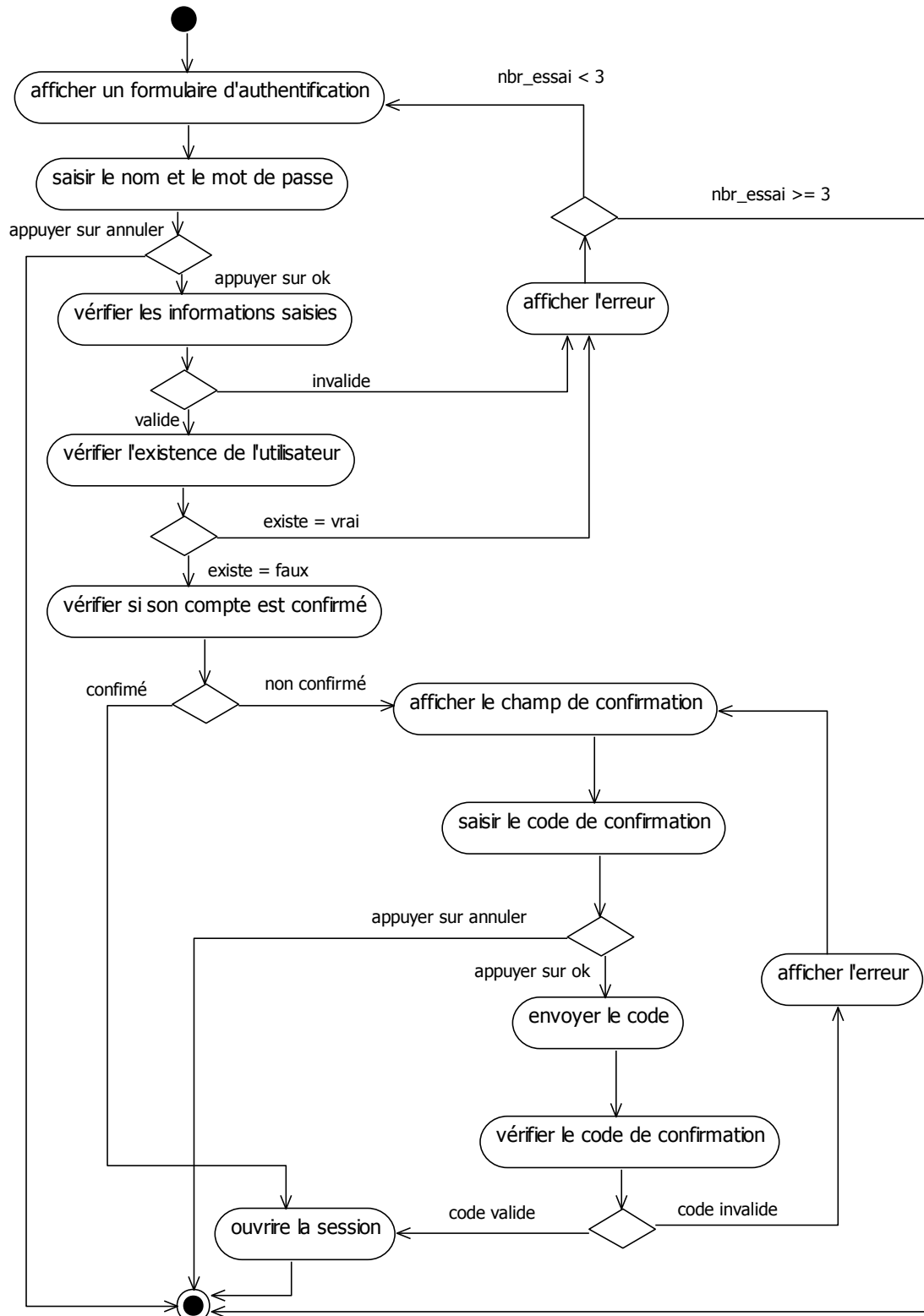
Dans cette section et pour chaque cas d'utilisation, nous allons présenter un diagramme d'activité qui permet de représenter le déroulement d'un cas d'utilisation.

7.2. Les diagrammes d'activité

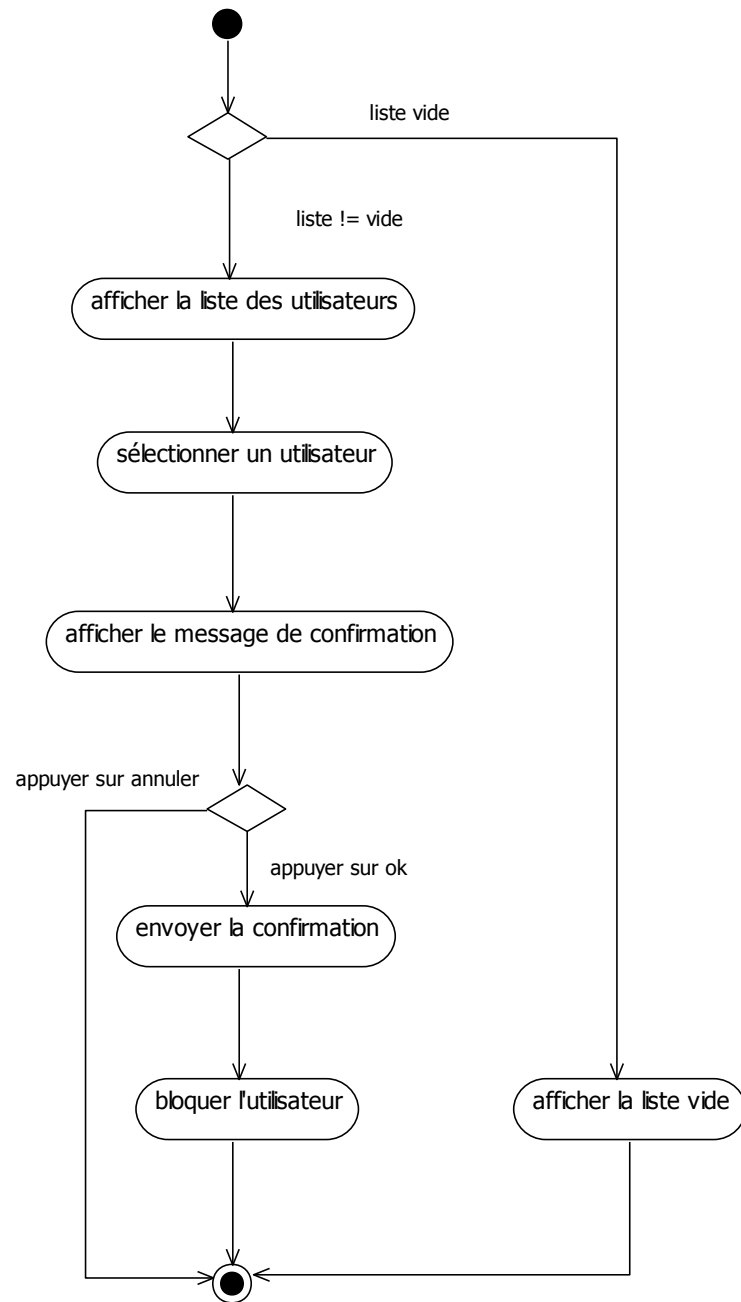
7.2.1. Ajouter un cours



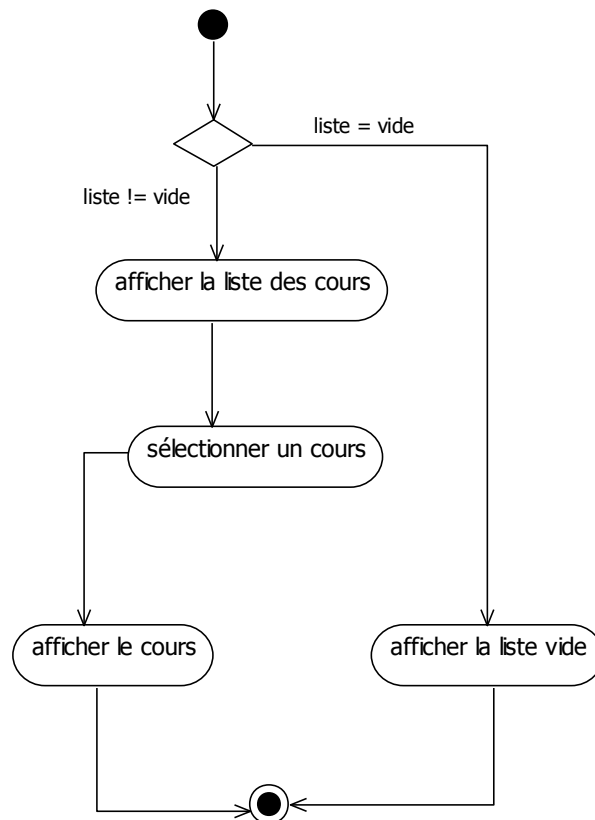
7.2.2. Authentifier



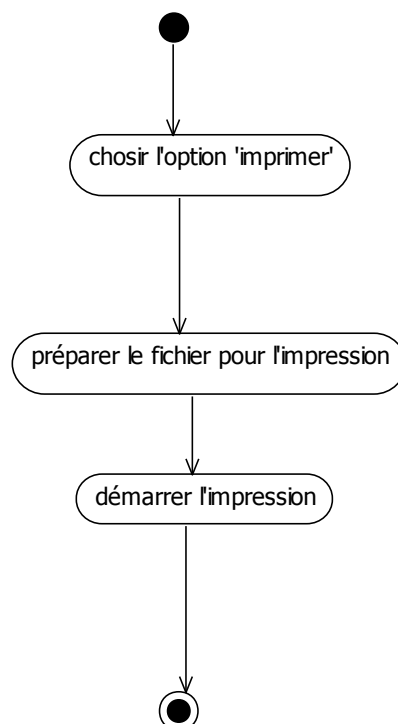
7.2.3. Bloquer utilisateur



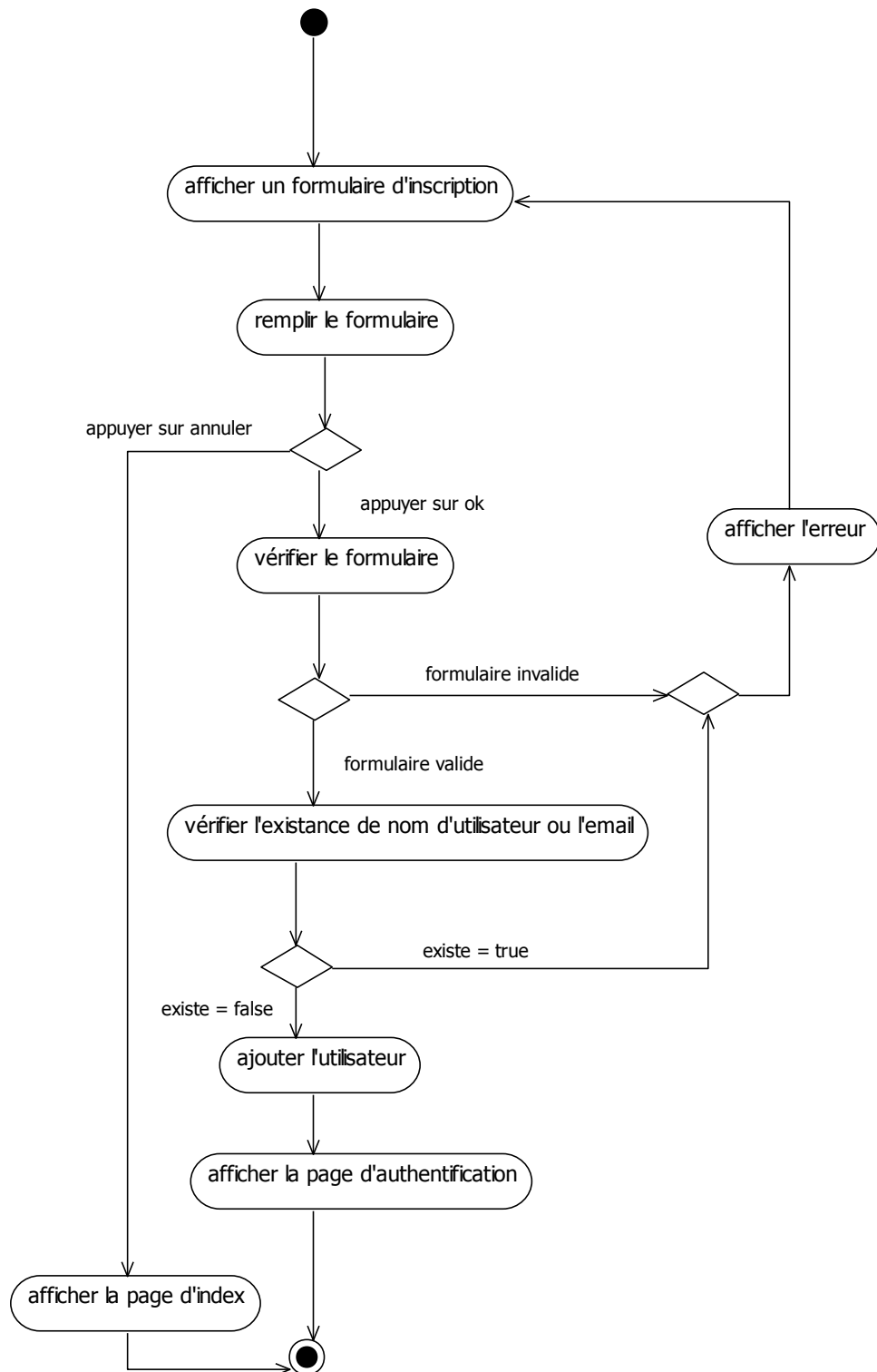
7.2.4. Consulter un cours



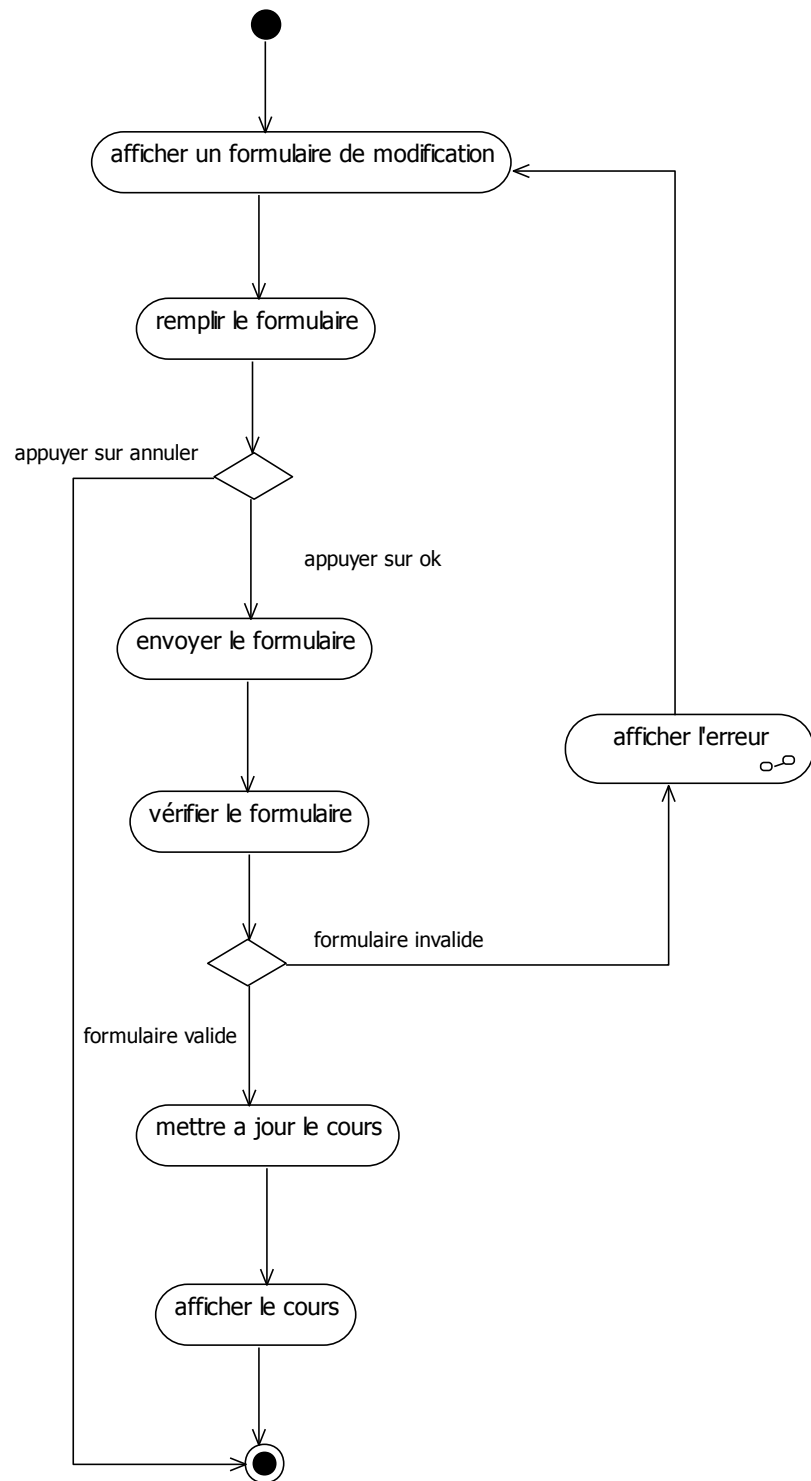
7.2.5. Imprimer



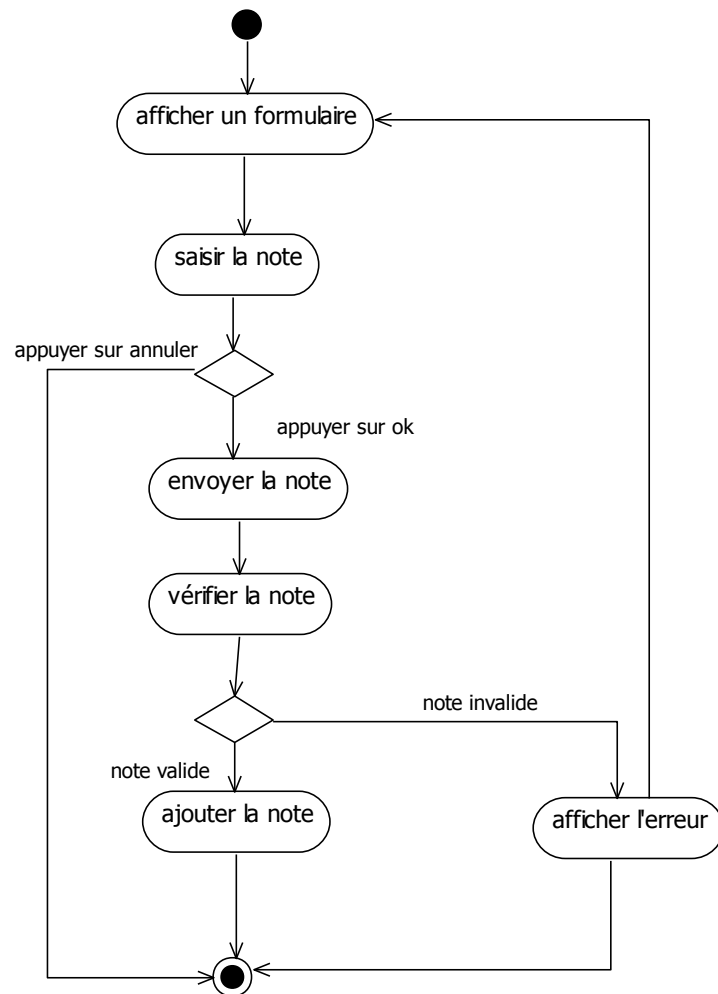
7.2.6. Inscrire



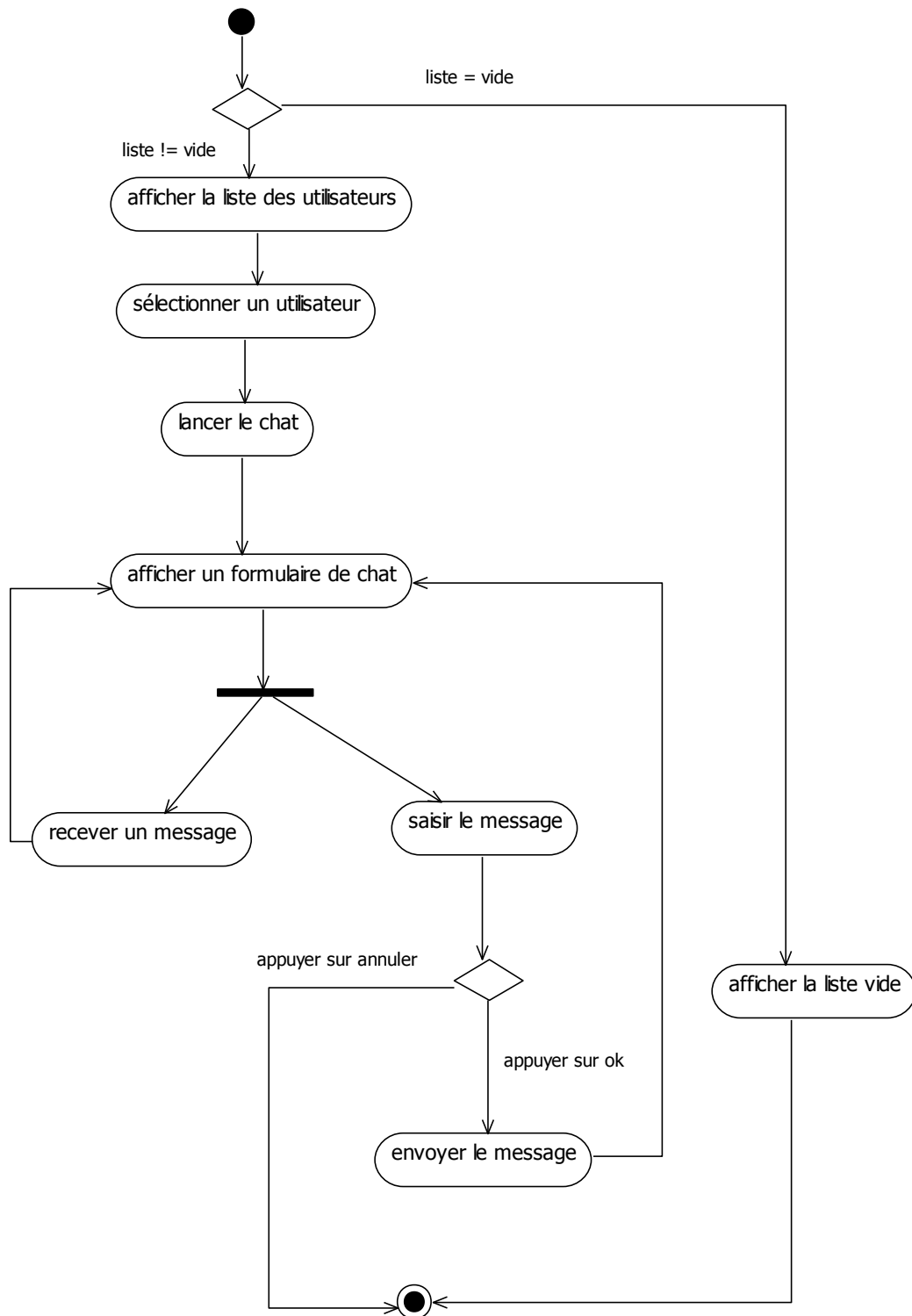
7.2.7. Modifier un cours



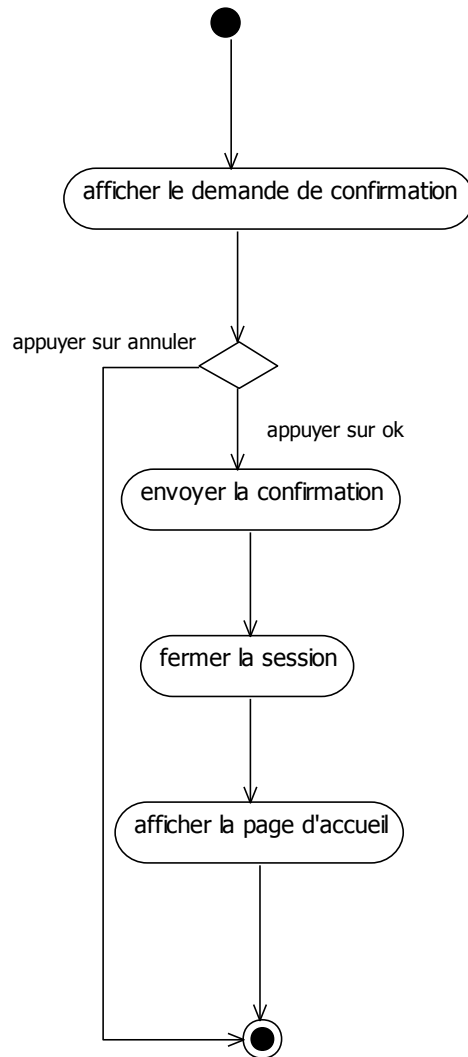
7.2.8. Noter les réponses



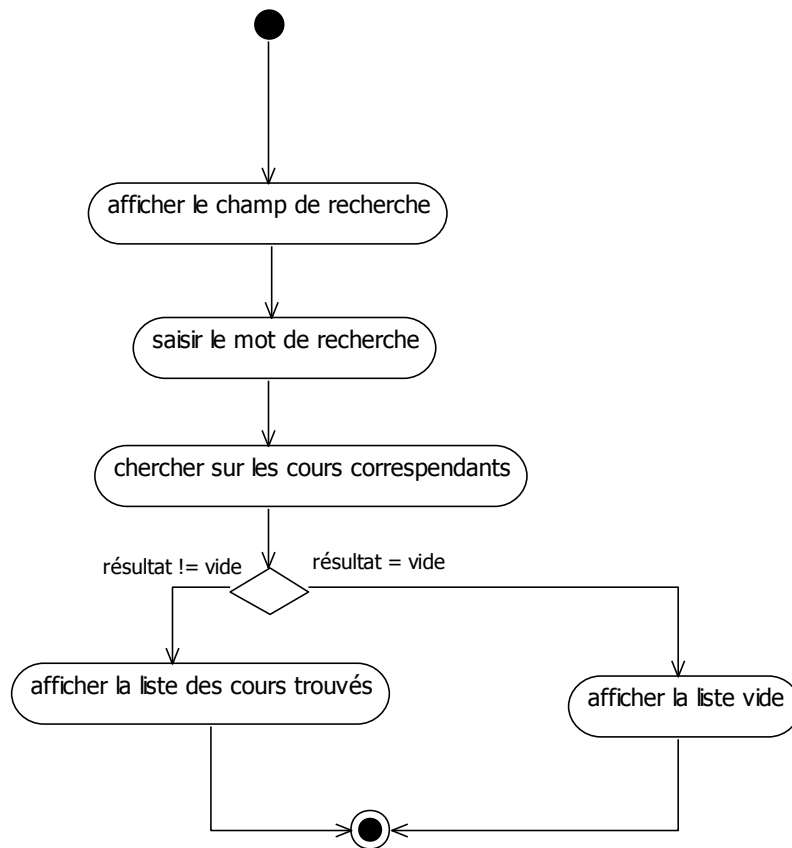
7.2.9. Participer dans le service de chat



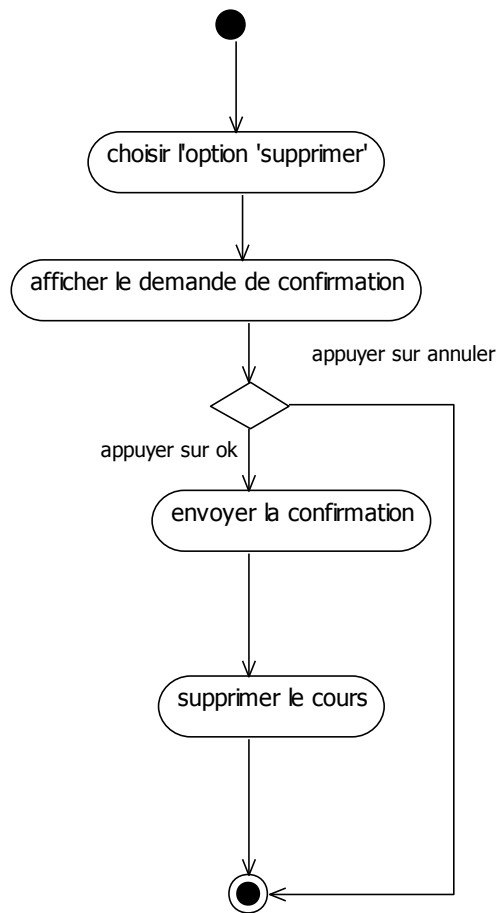
7.2.10. Quitter l'application



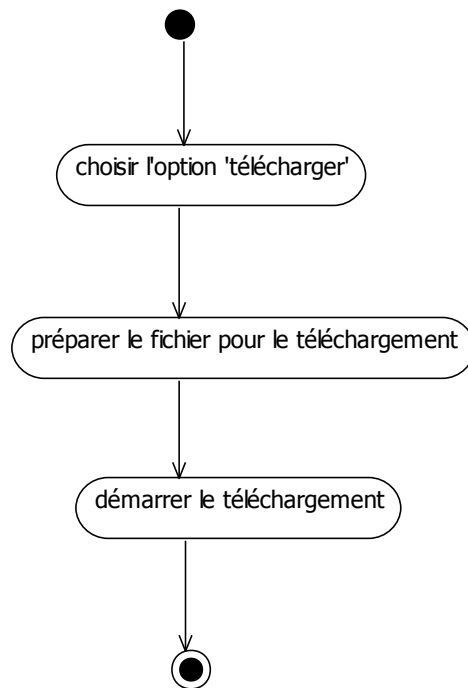
7.2.11. Rechercher



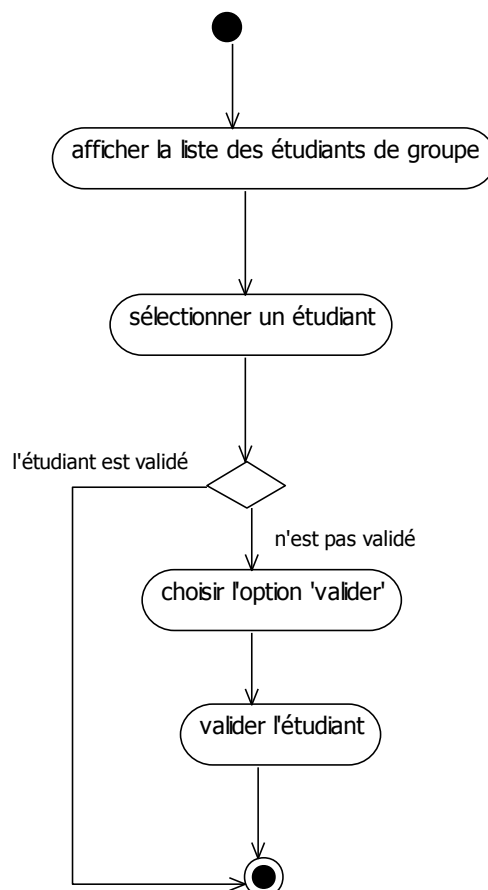
7.2.12. Supprimer un cours



7.2.13. Télécharger



7.2.14. Valider un Etudiant



8. CONCLUSION

L'étape d'étude préliminaire et de spécification des besoins est difficile à mettre en place mais quand elle est bien faite, elle nous aide à bien compris les besoins de l'utilisateur dans un cadre bien organisé et de minimaux des détails possibles.

Dans le chapitre suivant on va aller en plus des détails dans ces fonctionnalités.

CHAPITRE 3: ANALYSE ET CONCEPTION

• INTRODUCTION

Dans ce chapitre on décrit dans la première section les diagrammes des séquences détaillées correspondants à chaque diagramme de séquence système qui on a vue dans le chapitre précédant, dans la deuxième section on donne les diagrammes de navigation pour chaque utilisateur du site,

Et on termine le chapitre par les diagrammes des classes déduits à partir des diagrammes des séquences.

2. LES DIGRAMMES DE SEQUENCES DETAILLEES

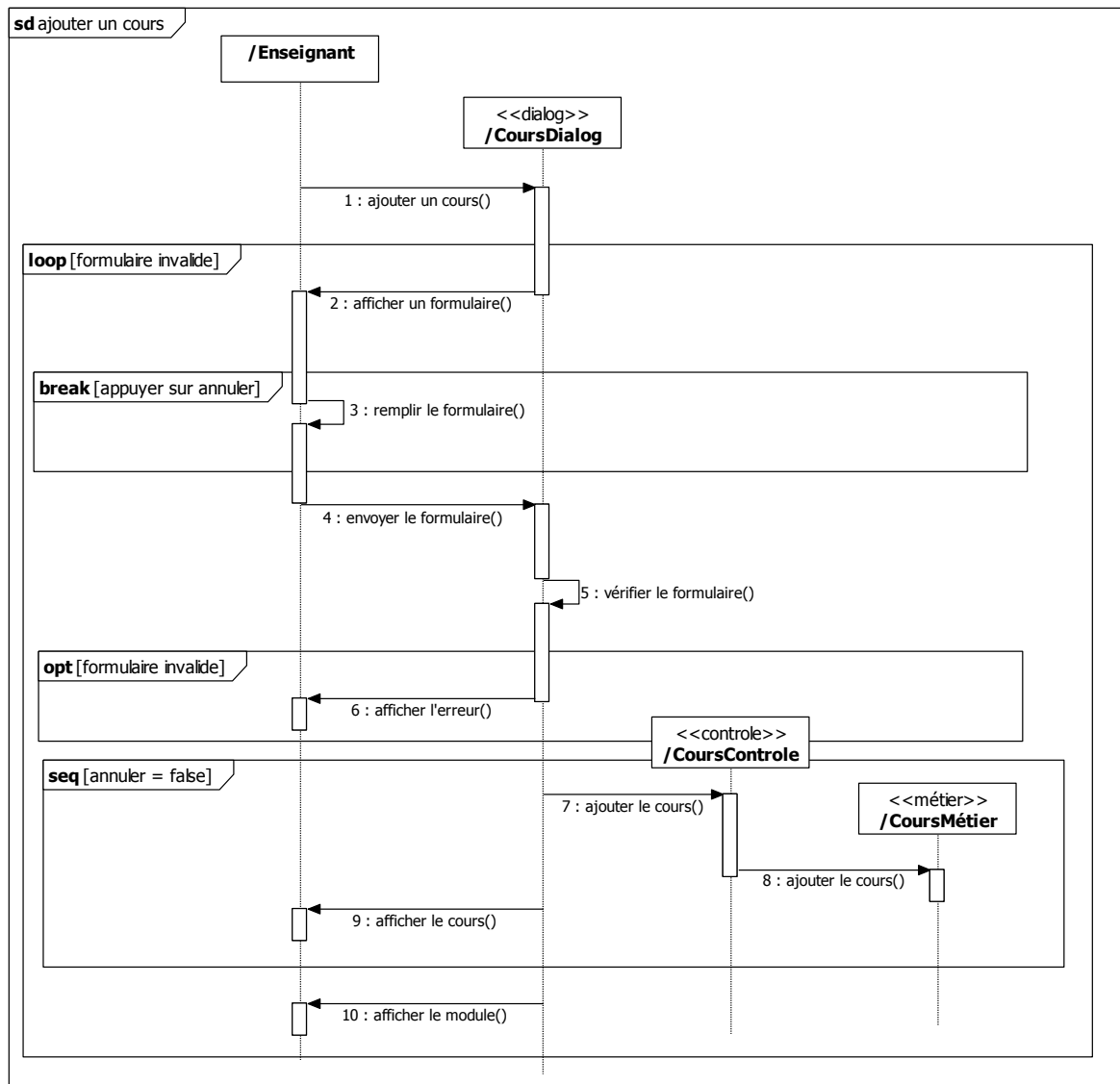
2.1. Introduction

Chaque cas d'utilisation se traduit par un certain nombre de scénarios. Chaque scénario peut être spécifié par un diagramme de séquence.

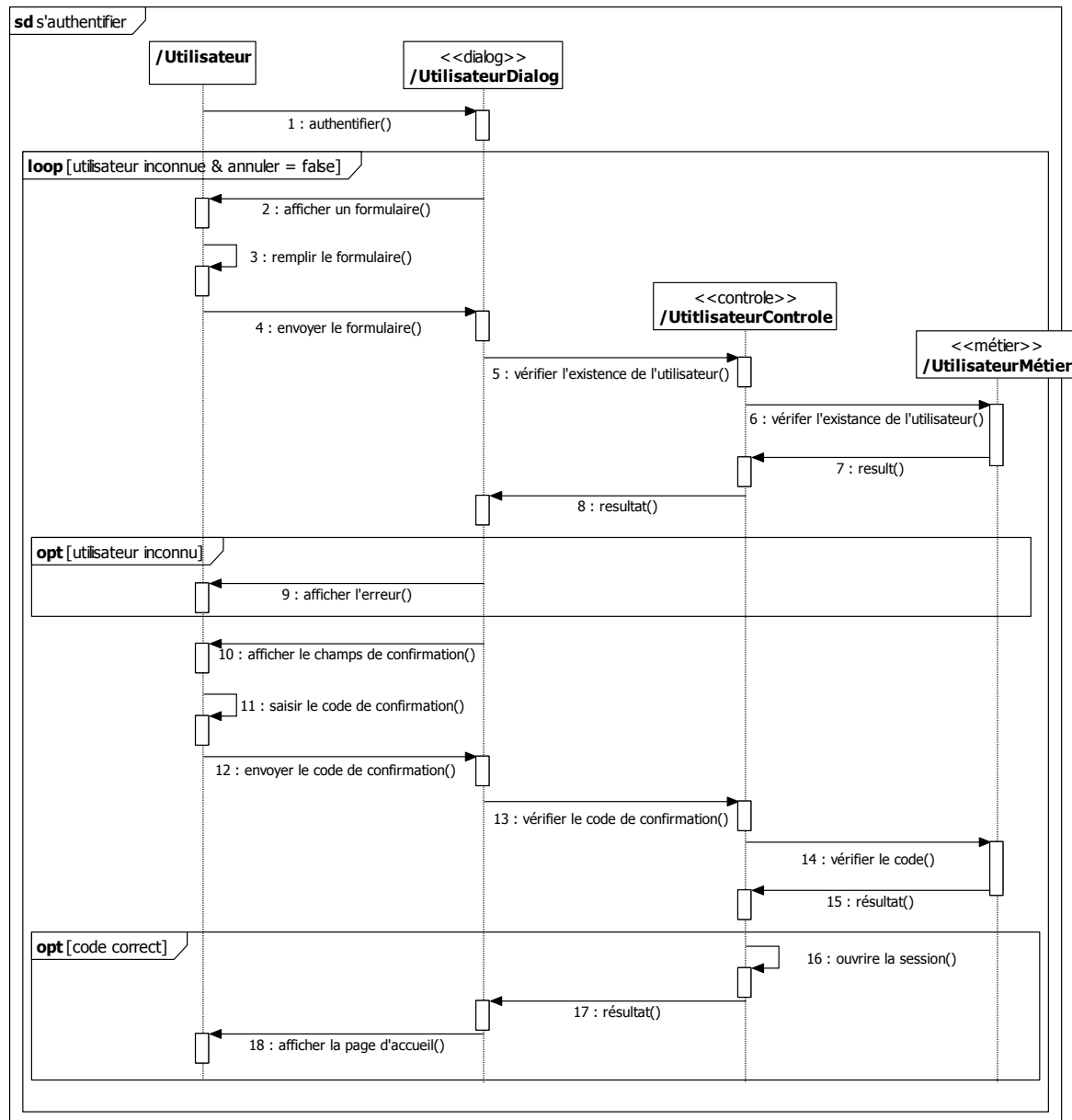
IL faut savoir qu'un scénario décrit une extension particulière d'un cas d'utilisation du début à la fin. Il correspond à un enchaînement du cas d'utilisation se terminant par une fin normale ou erreur.

2.2. Les diagrammes

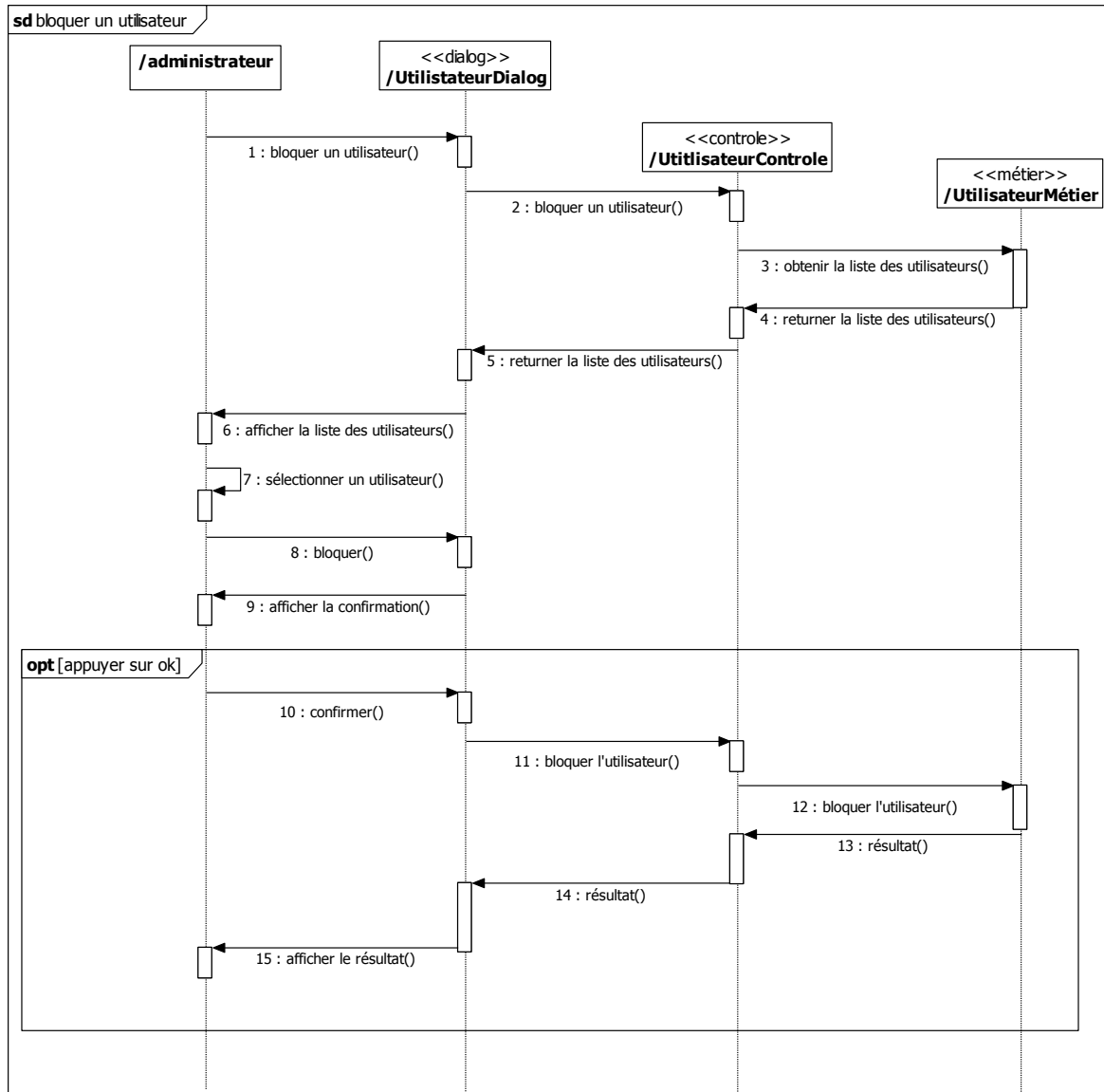
2.2.1. Ajouter un cours



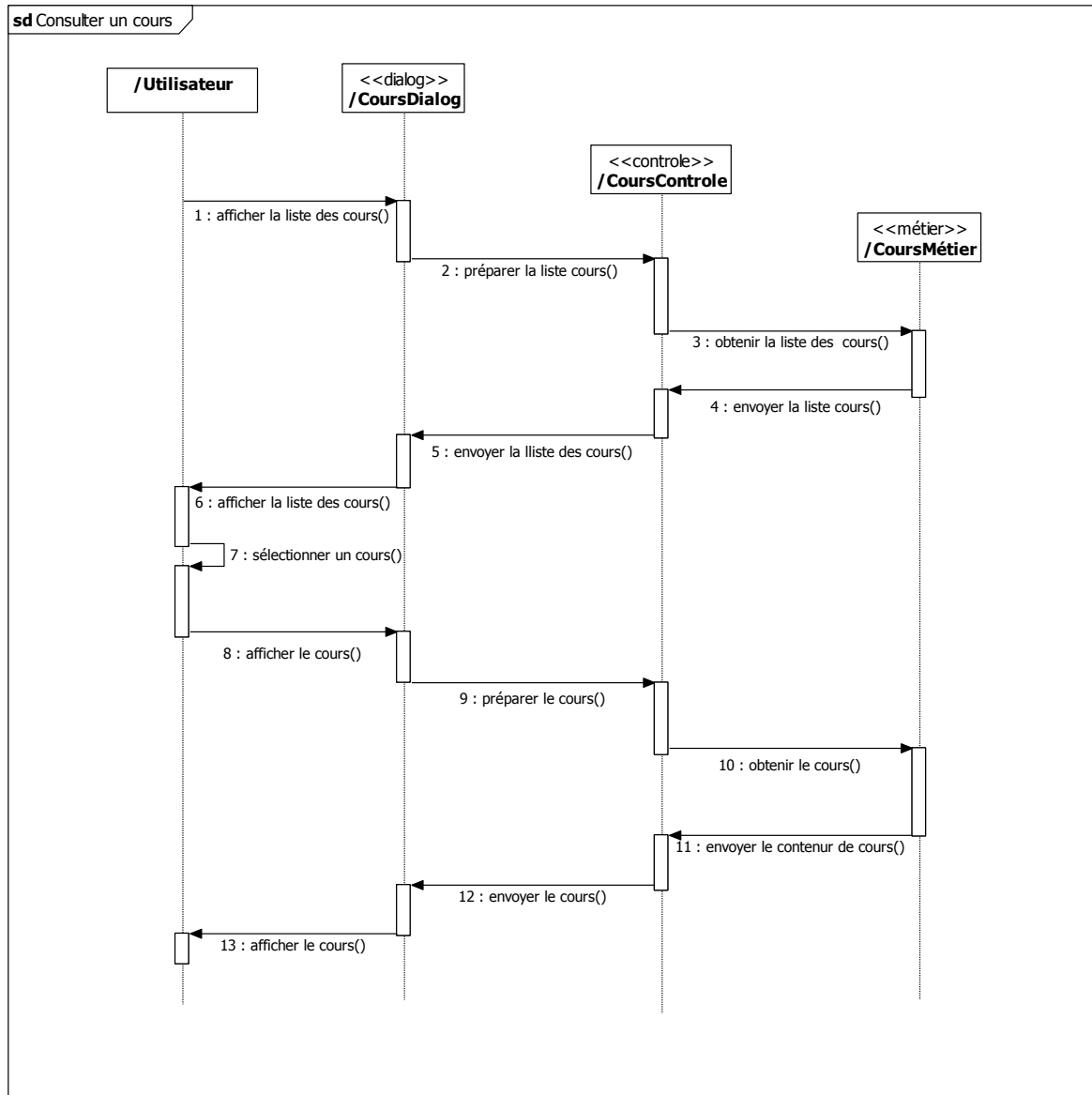
2.2.2. Authentifier



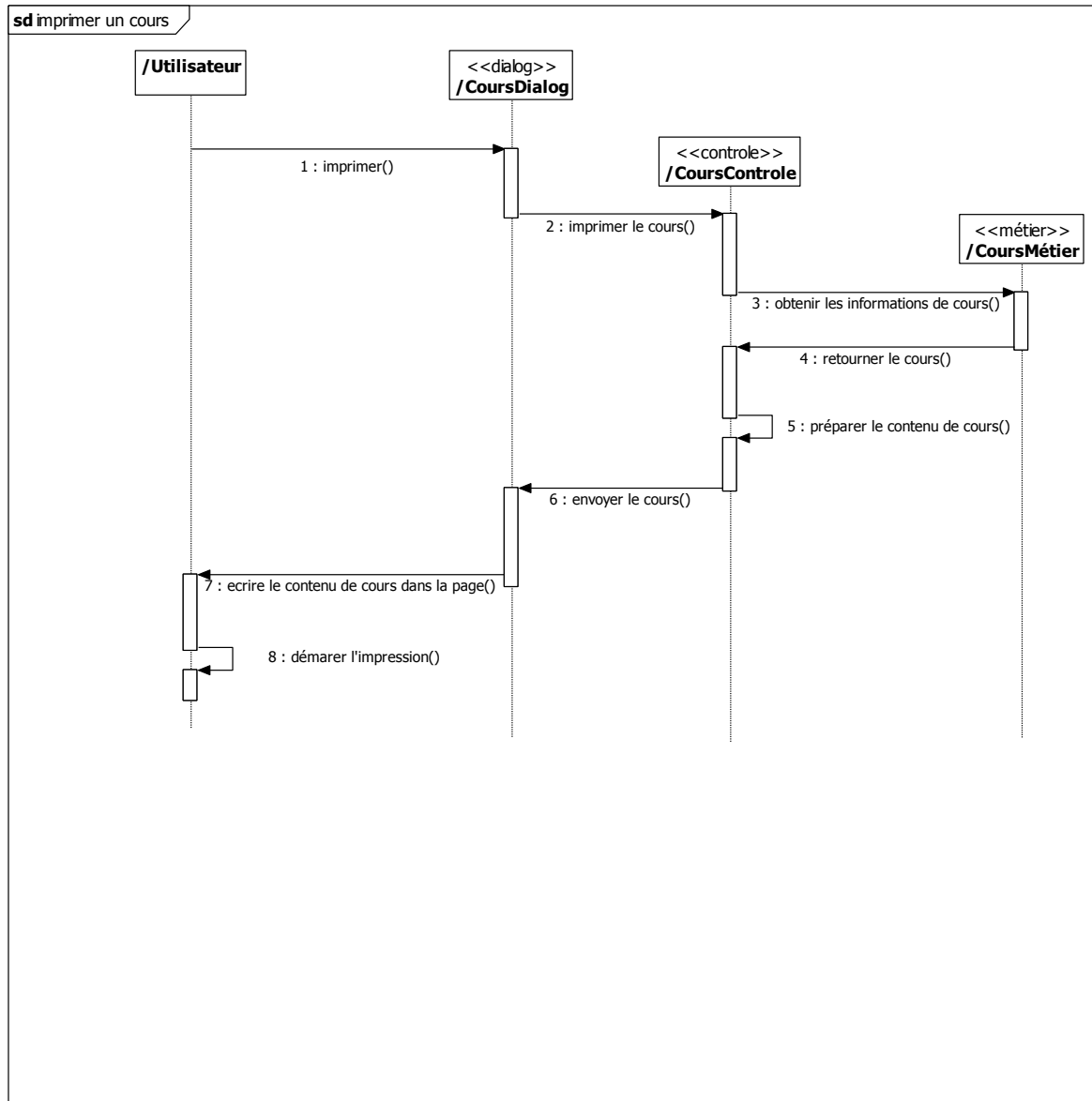
2.2.3. Bloquer utilisateur



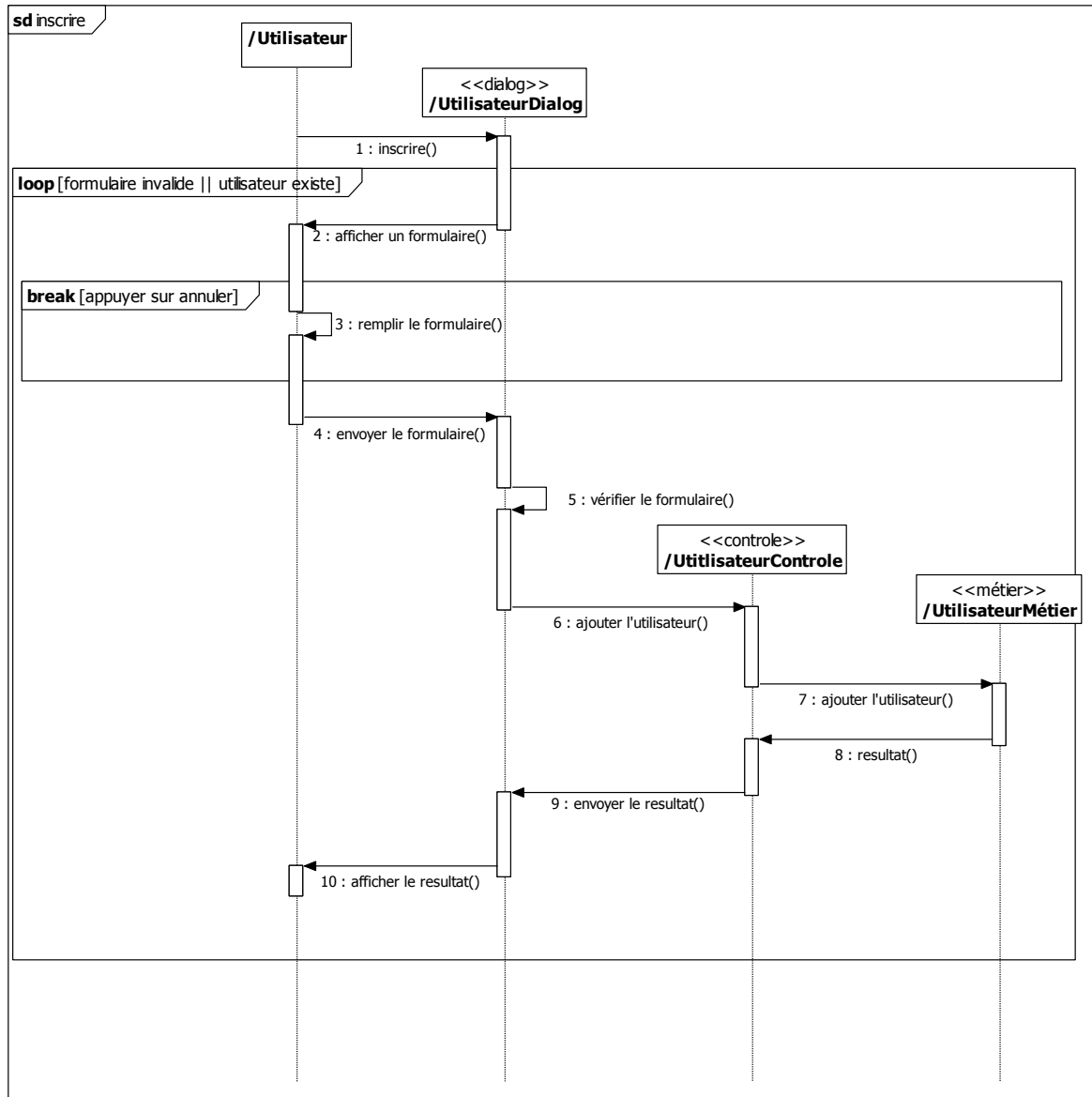
2.2.4. Consulter un cours



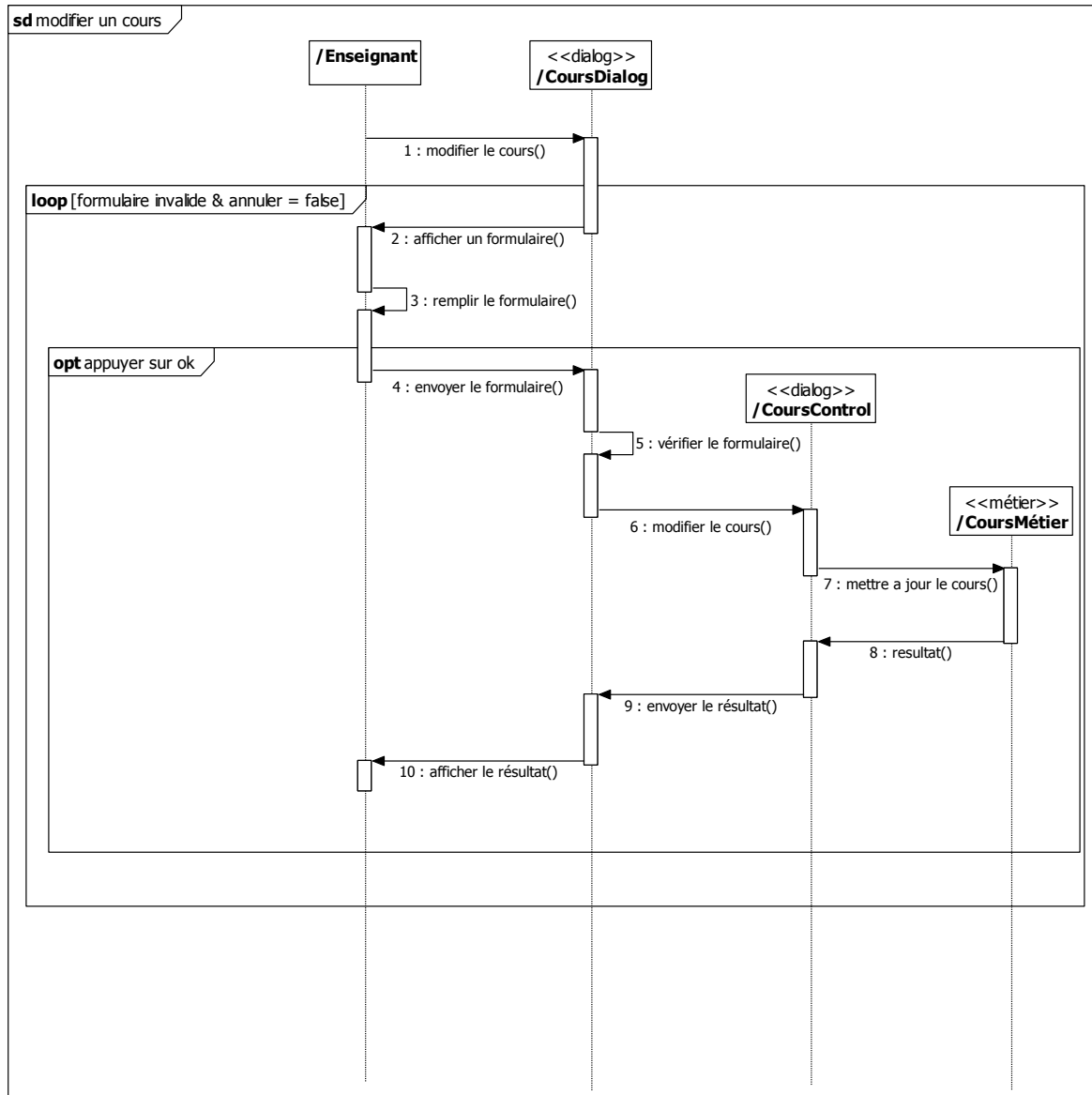
2.2.5. Imprimer



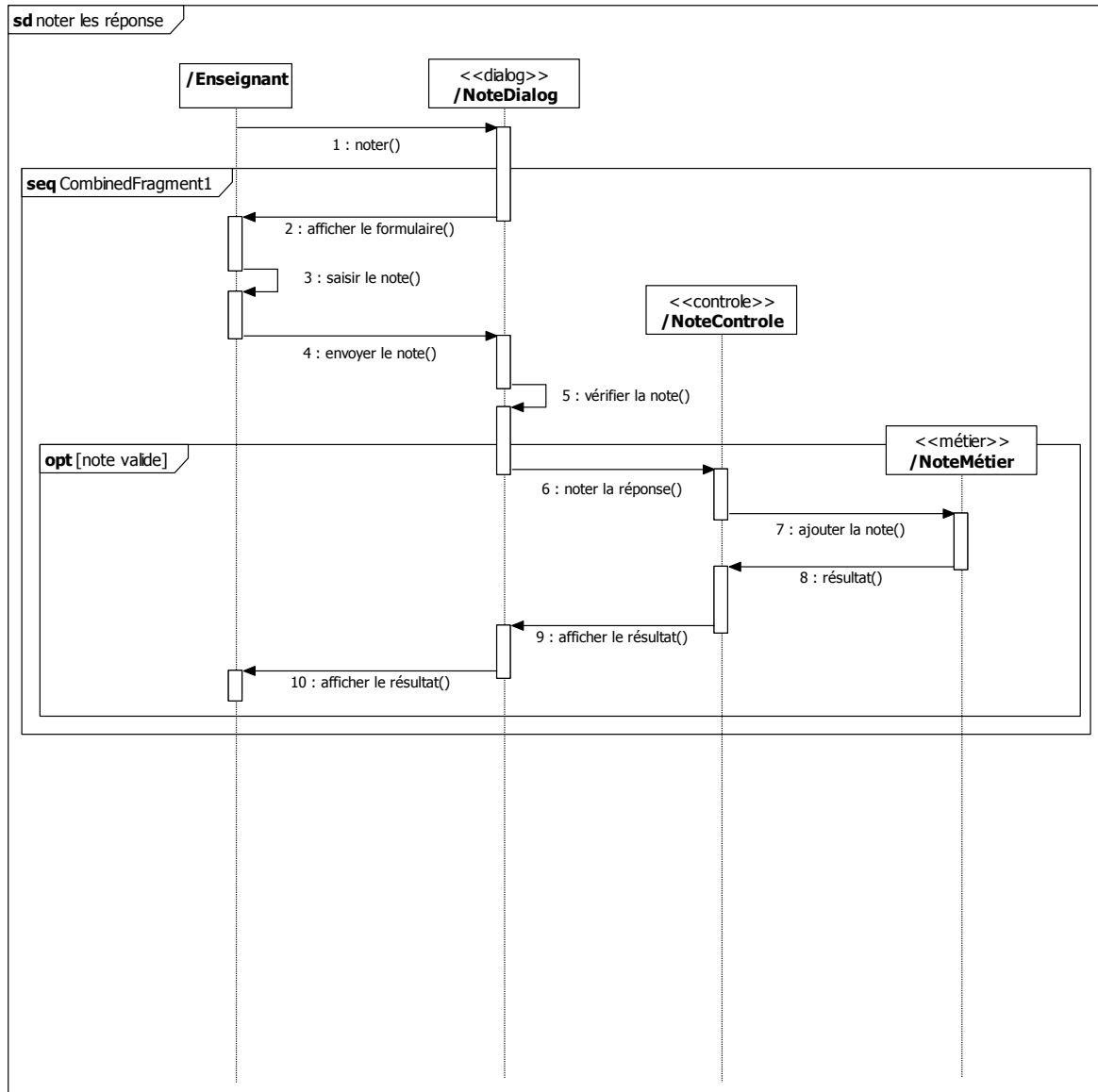
2.2.6. Inscrire



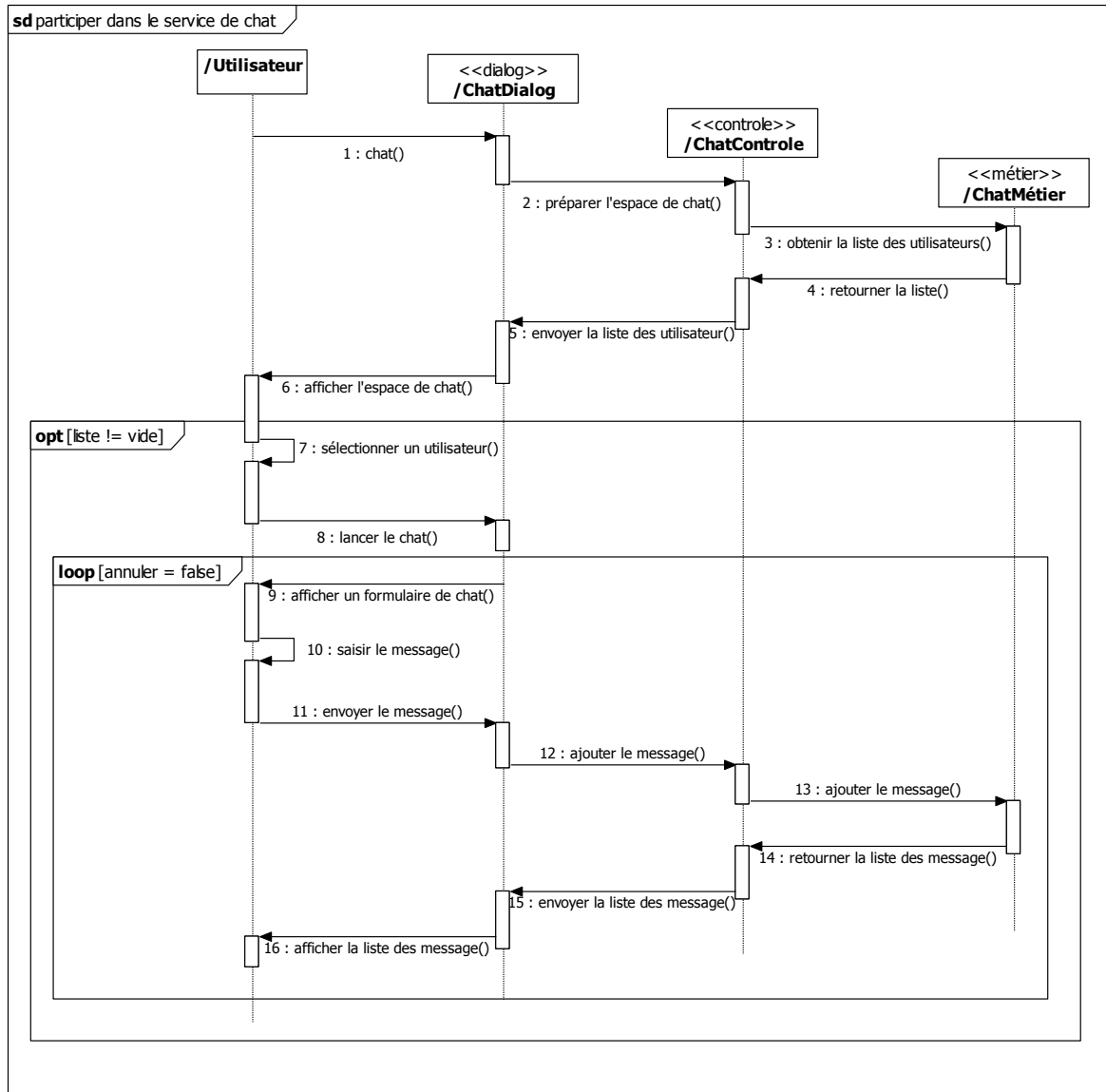
2.2.7. Modifier un cours



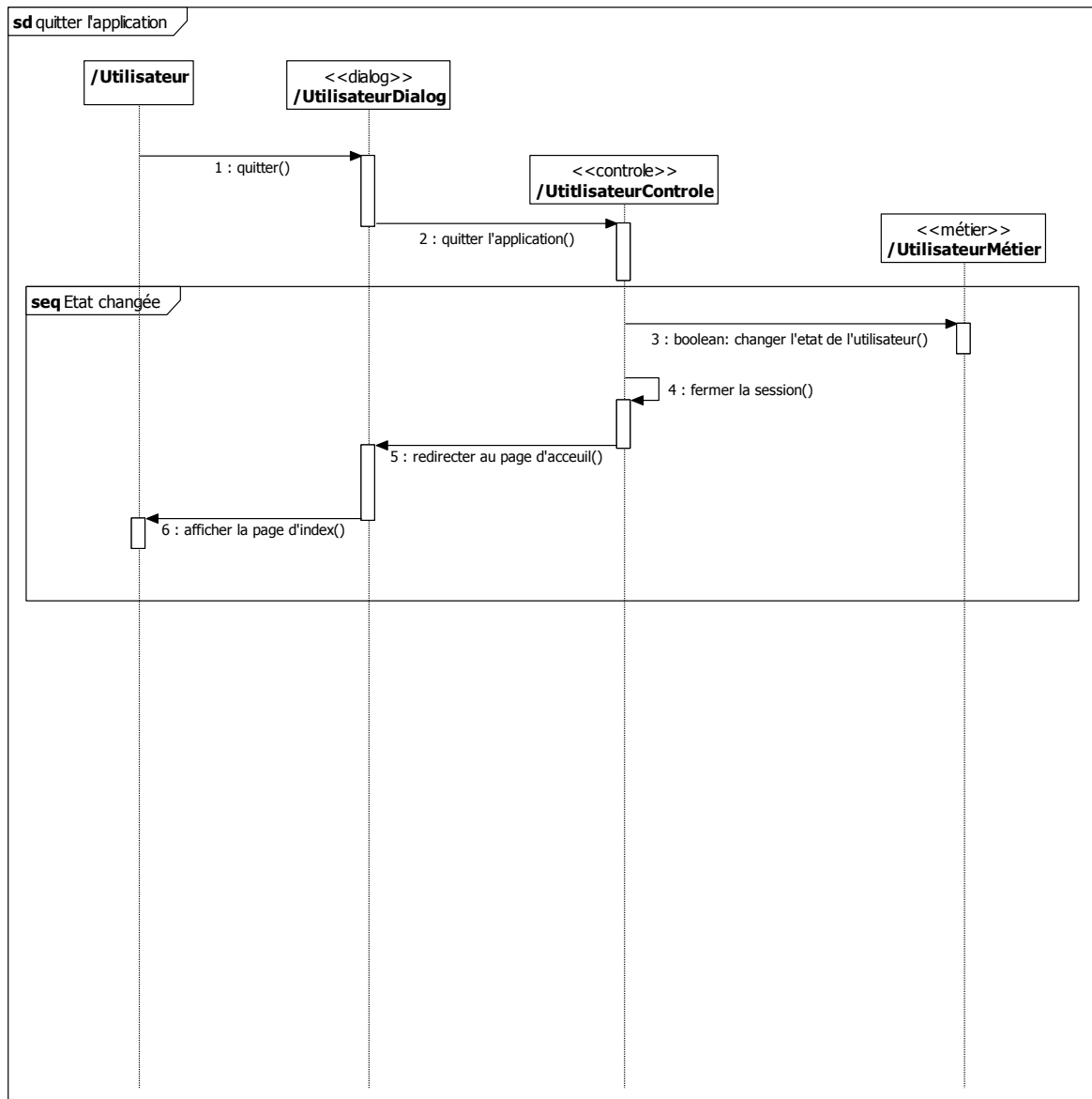
2.2.8. Noter les réponses



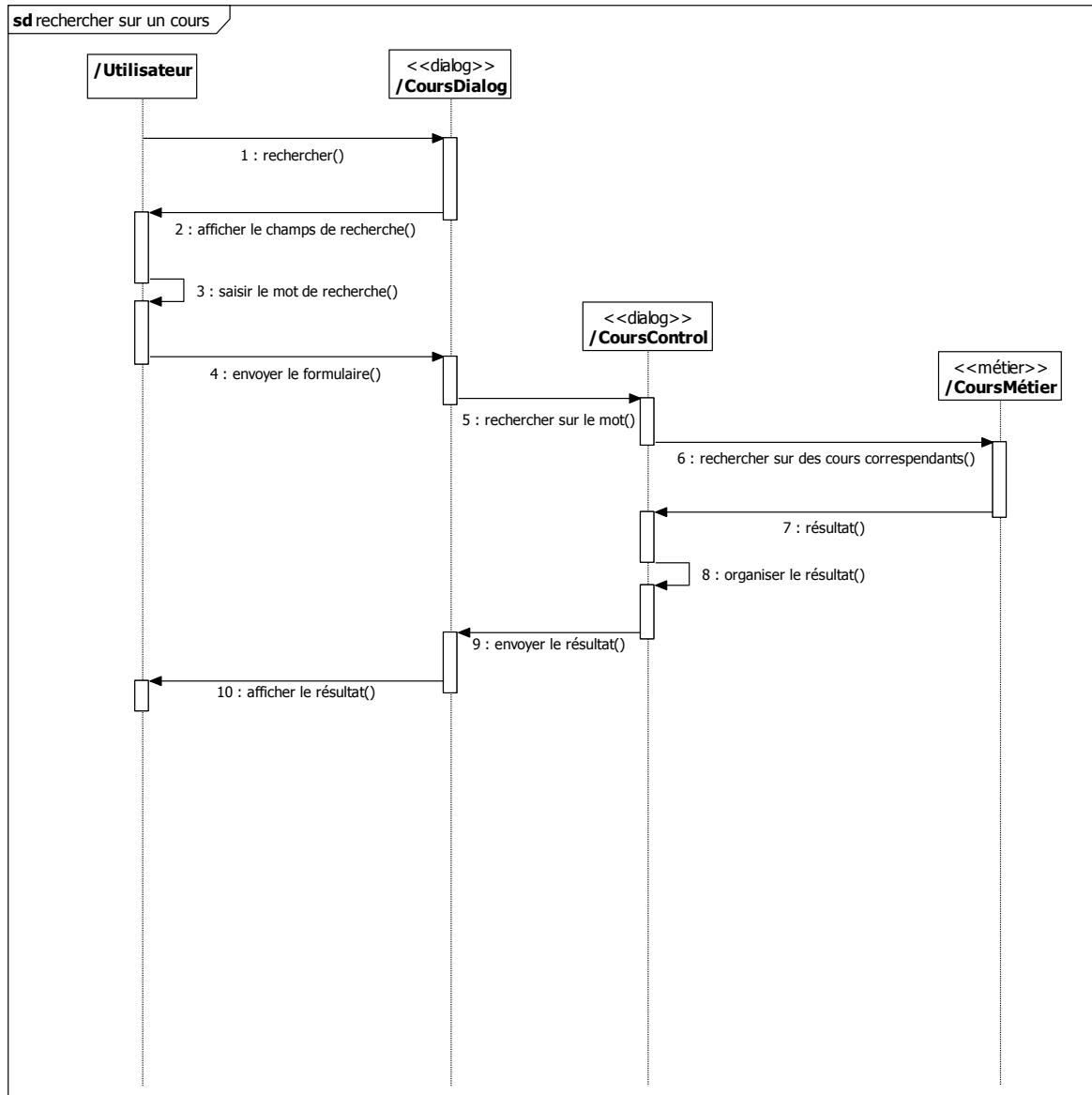
2.2.9. Participer dans le service de chat



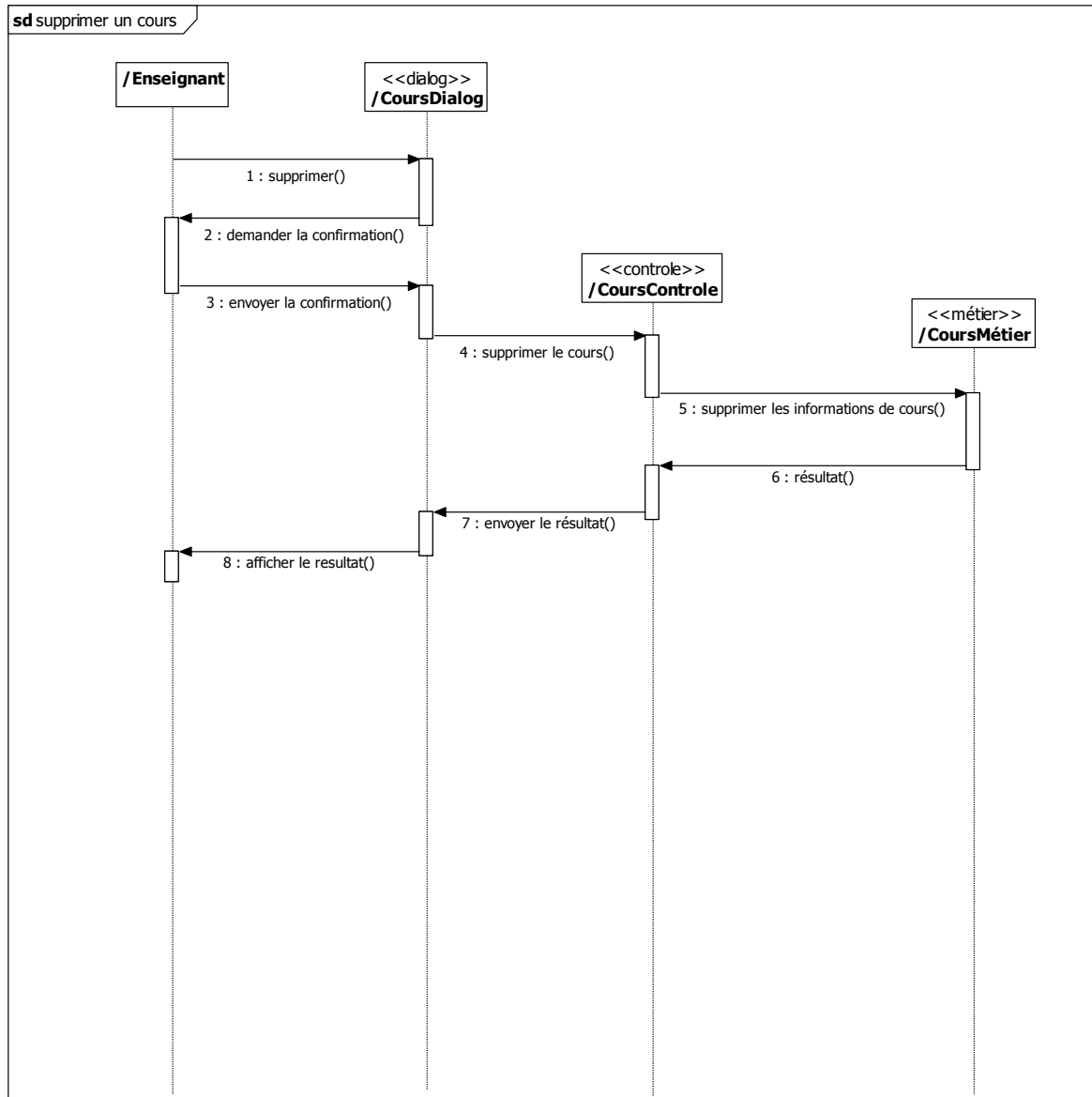
2.2.10. Quitter l'application



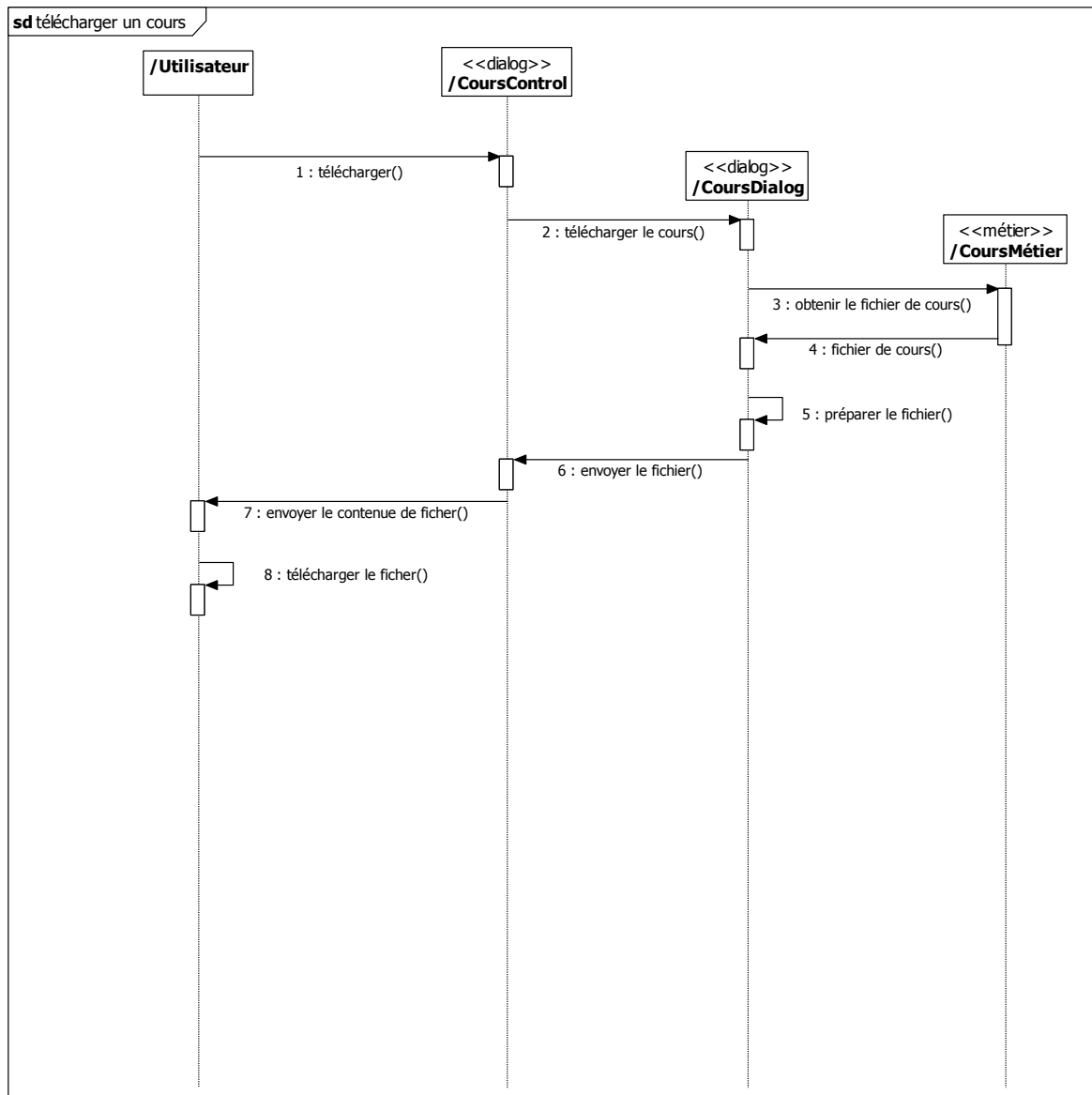
2.2.11. Rechercher



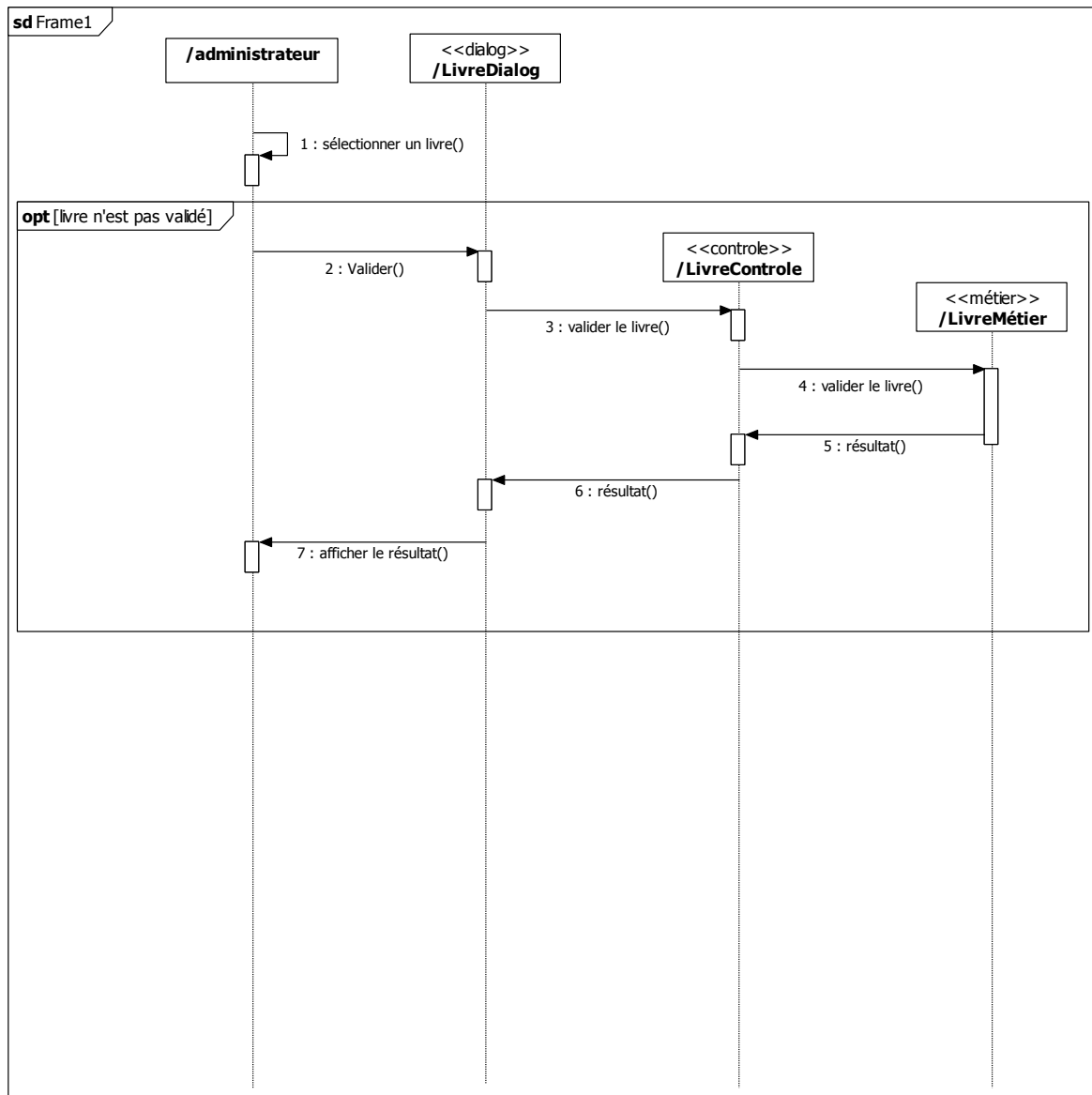
2.2.12. Supprimer un cours



2.2.13. Télécharger



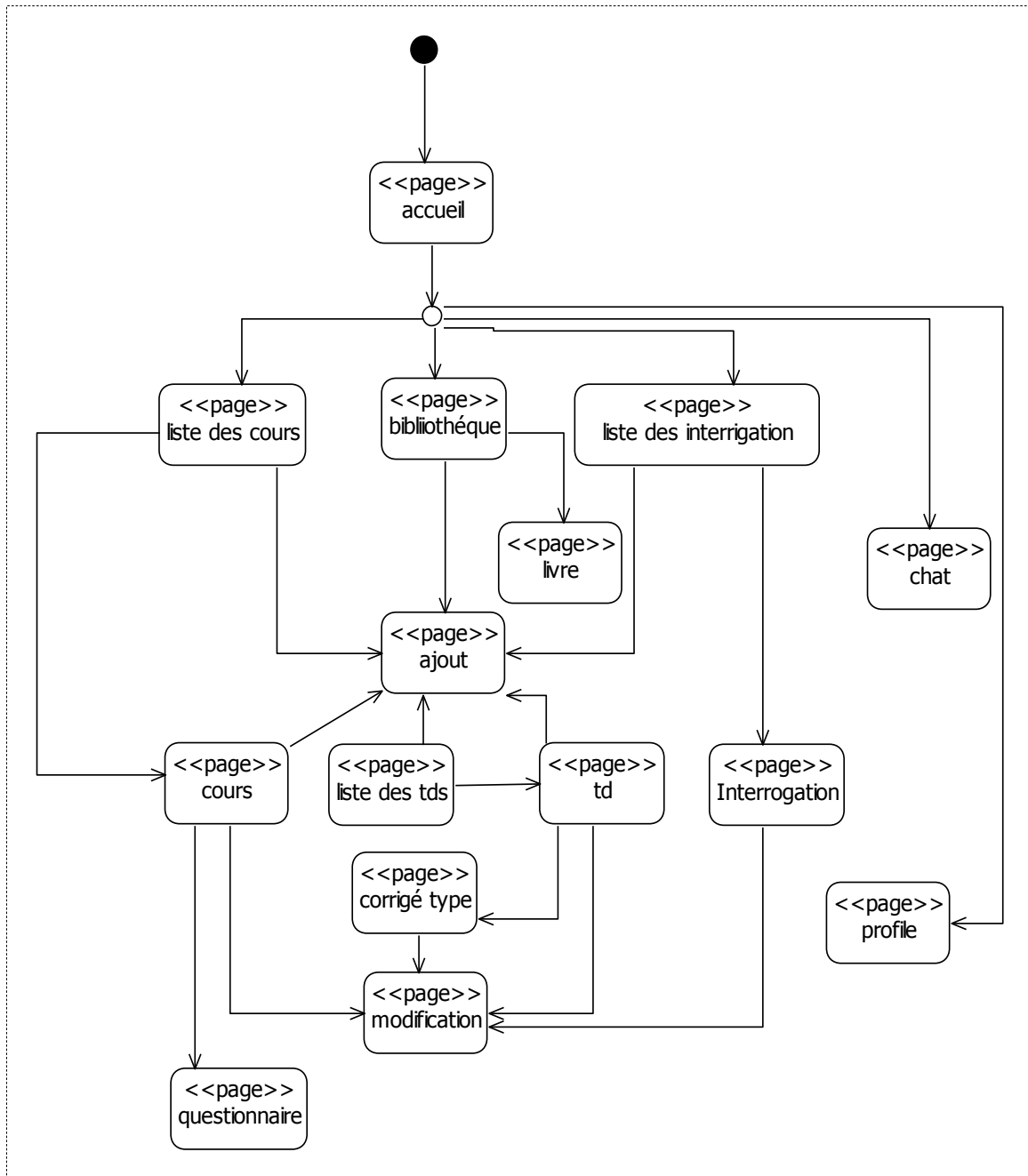
2.2.14. Valider un Livre



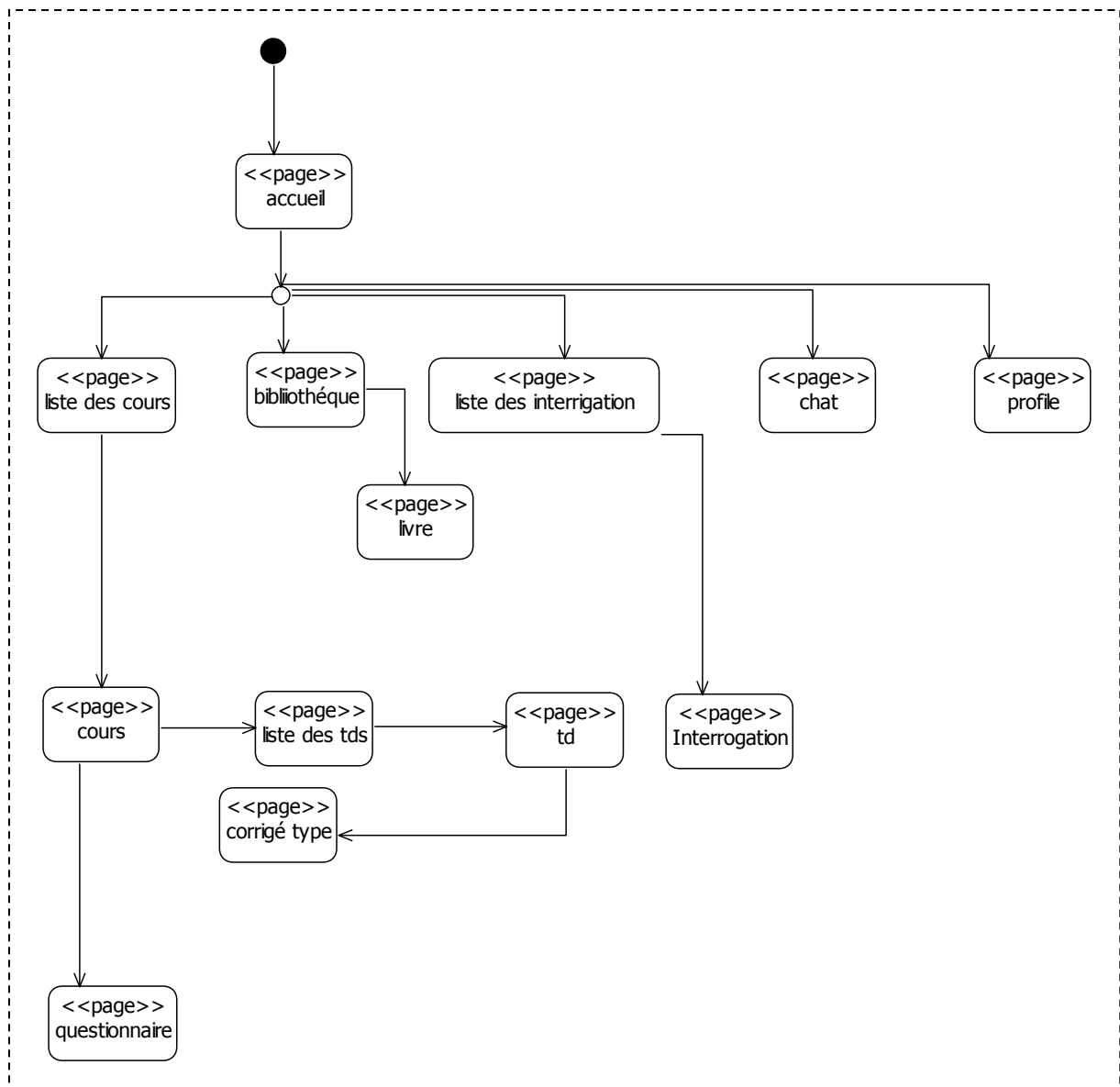
3. LES DIAGRAMMES DE NAVIGATIONS

3.1. Les diagrammes

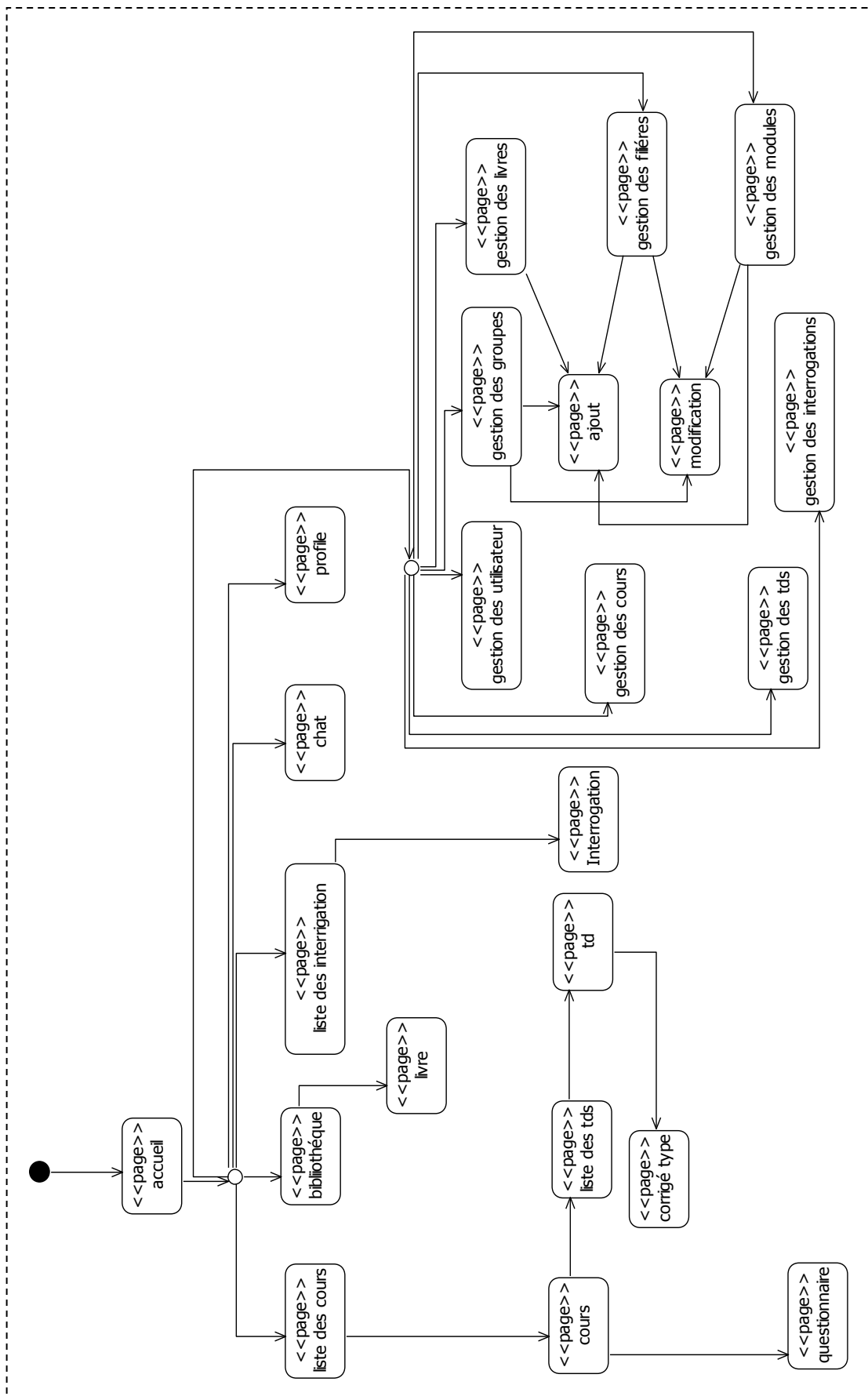
3.1.1. Enseignant



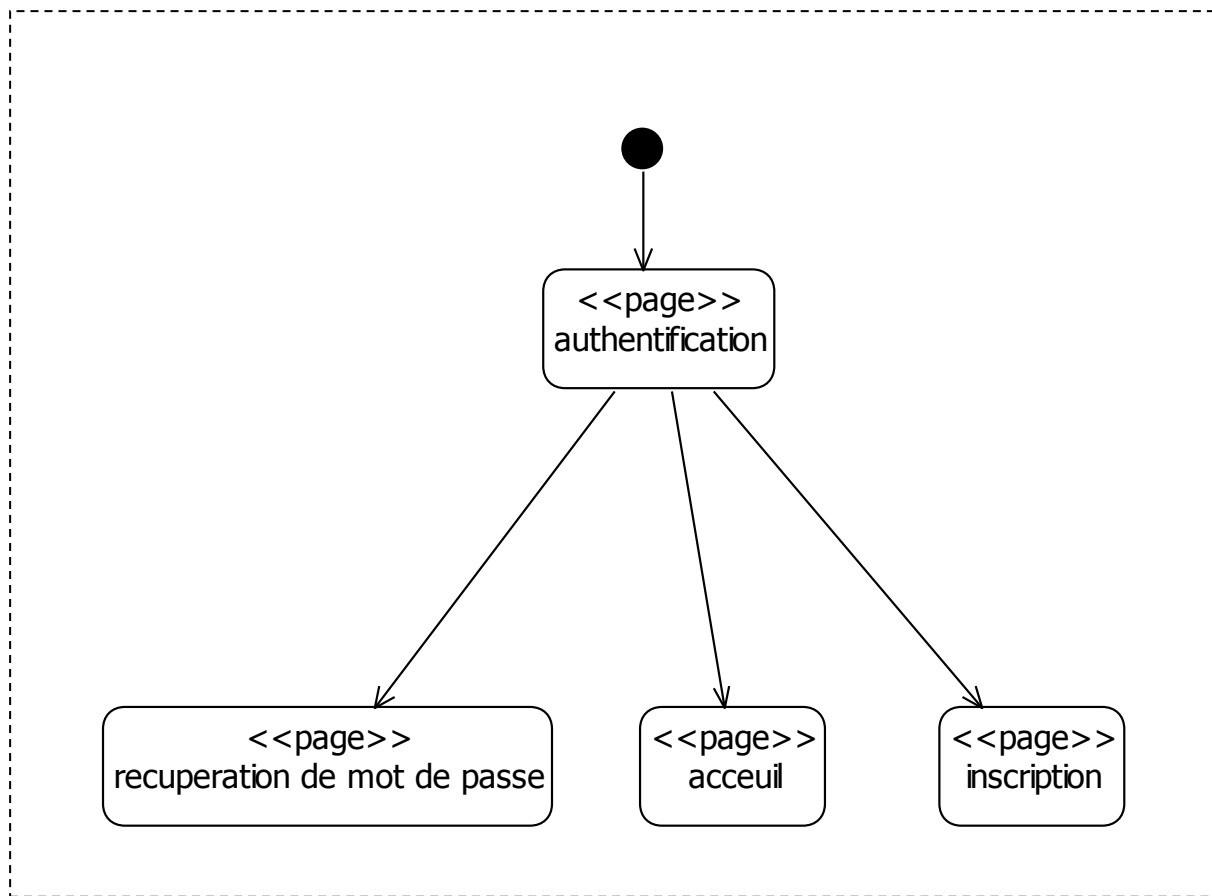
3.1.2. Etudiant



3.1.3. Administrateur



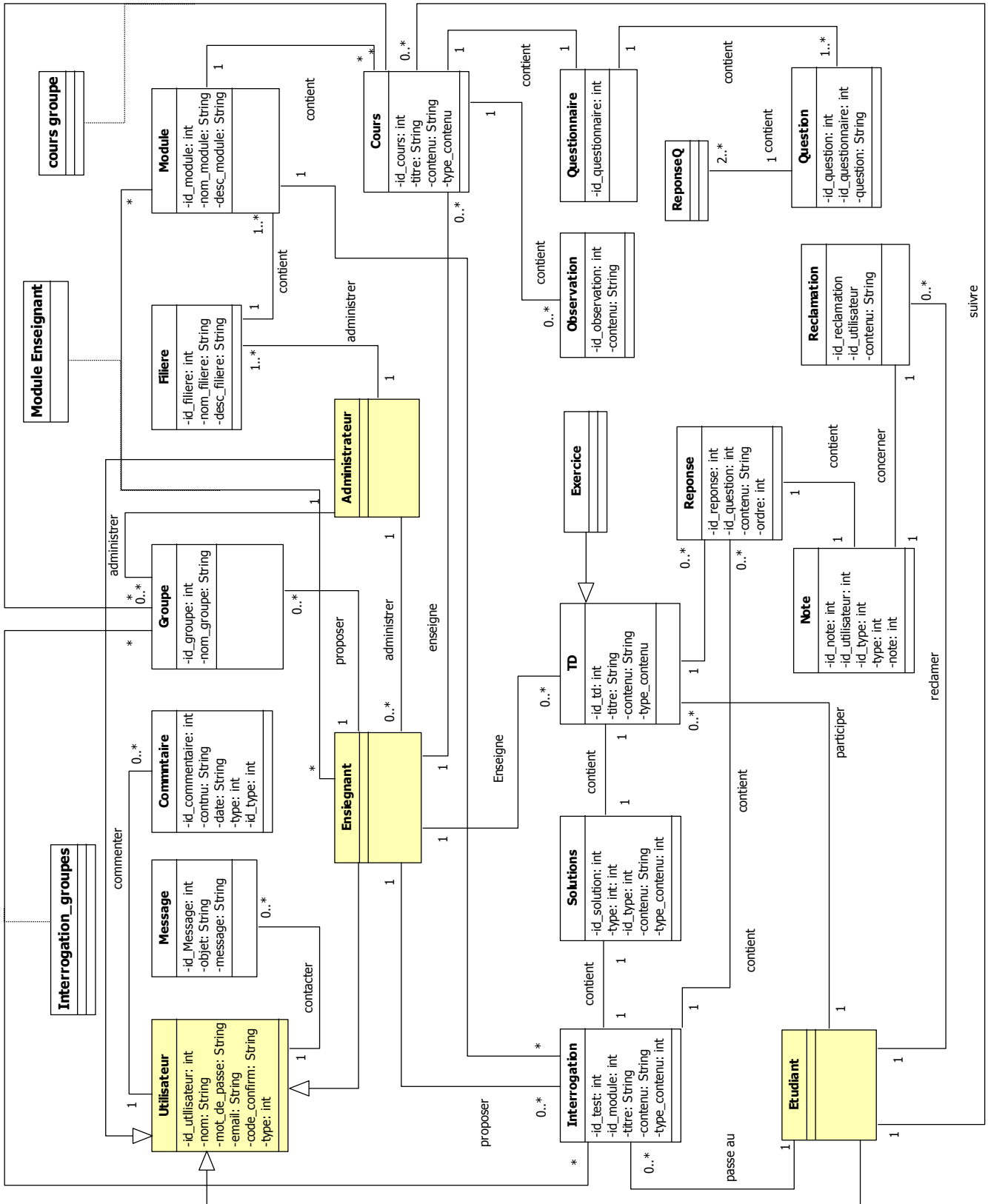
3.1.4. Visiteur



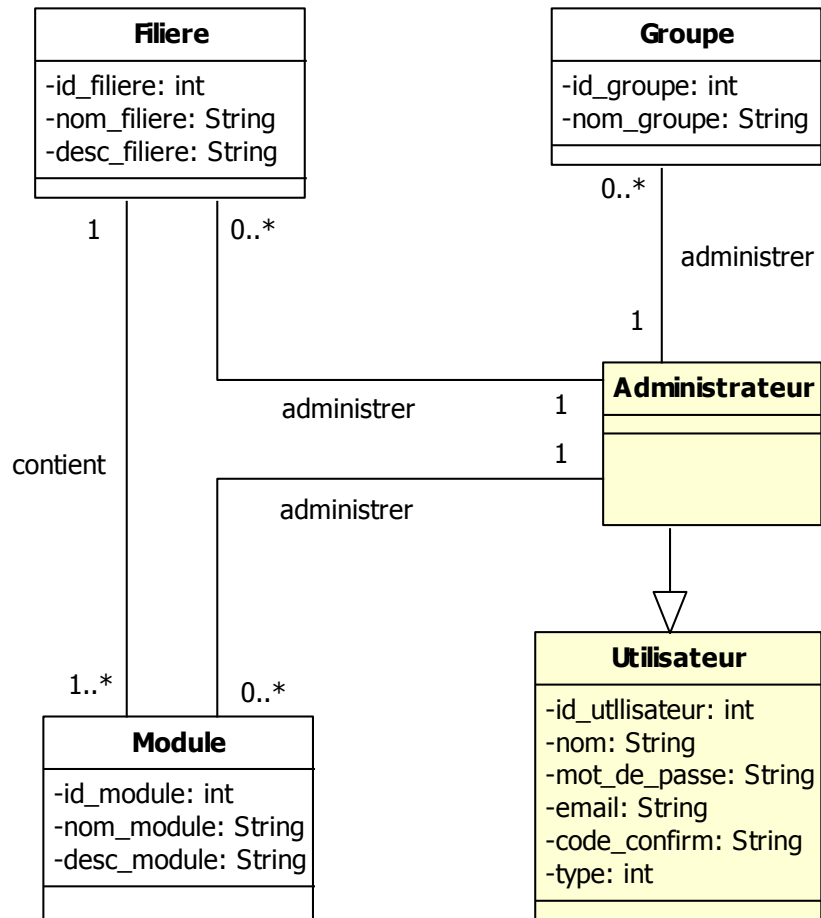
4. LES DIAGRAMMES DE CLASSE

4.1. Les diagrammes de classes de métier

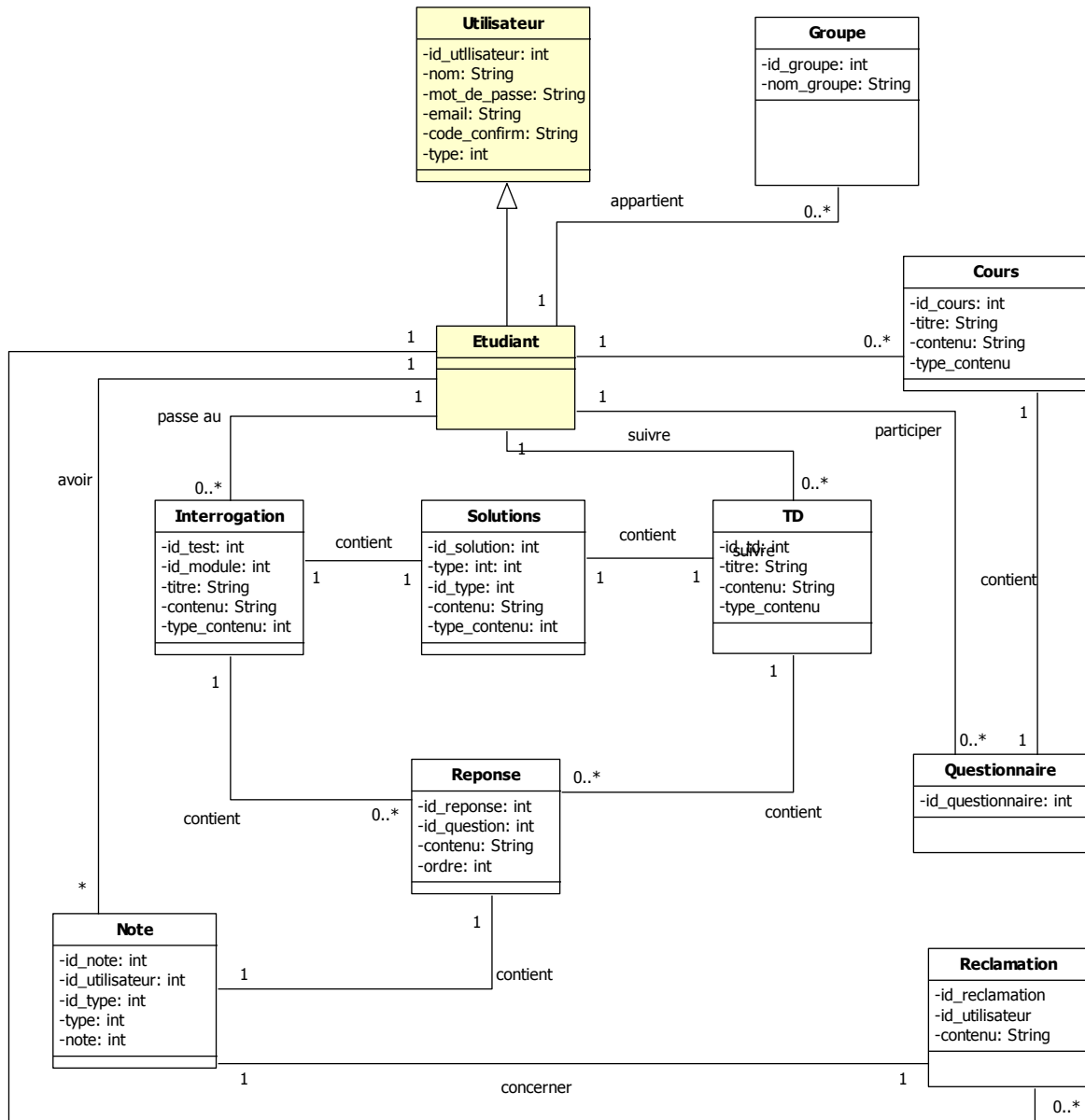
4.1.1. Générale



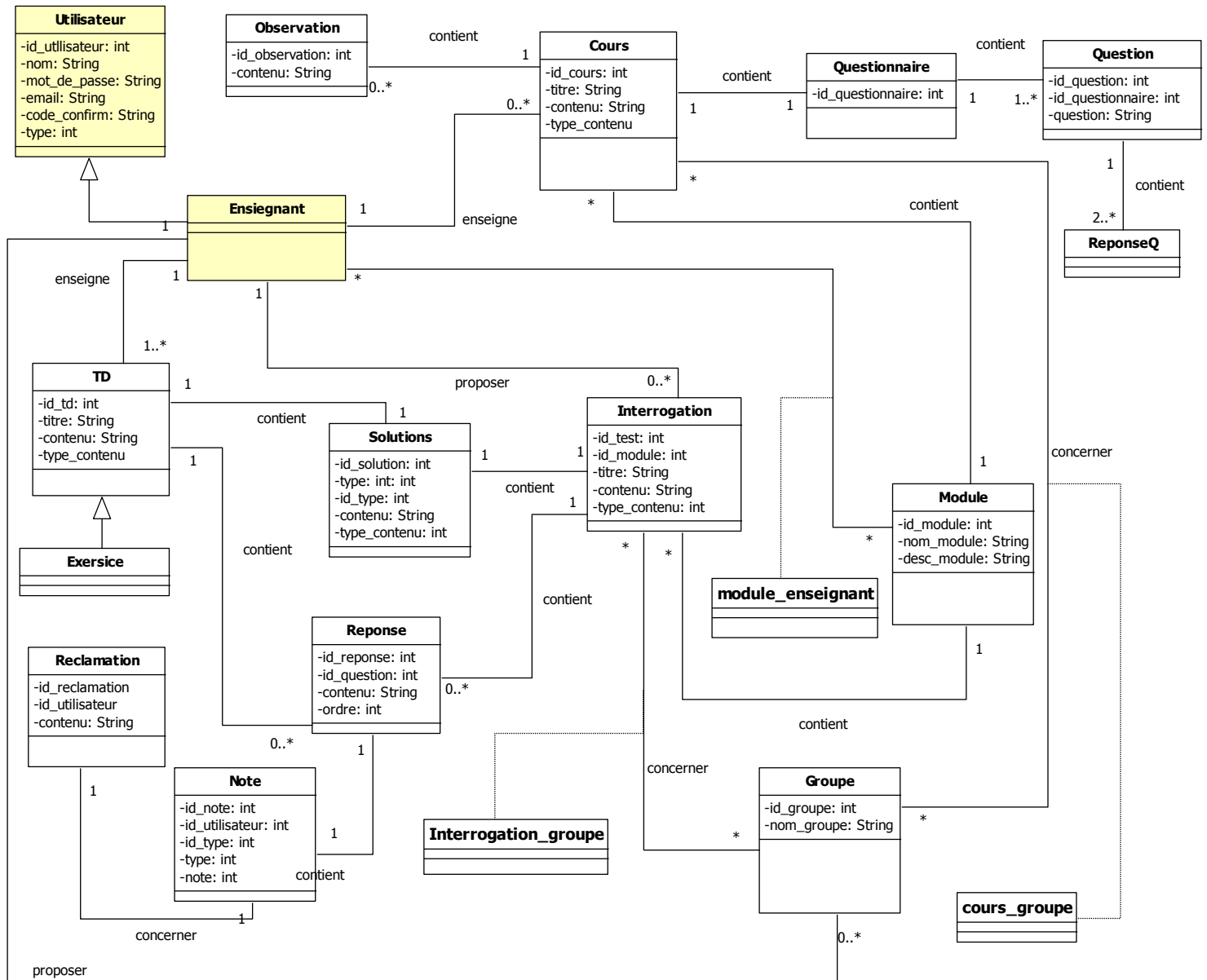
4.1.2. Administrateur



4.1.3. Etudiant

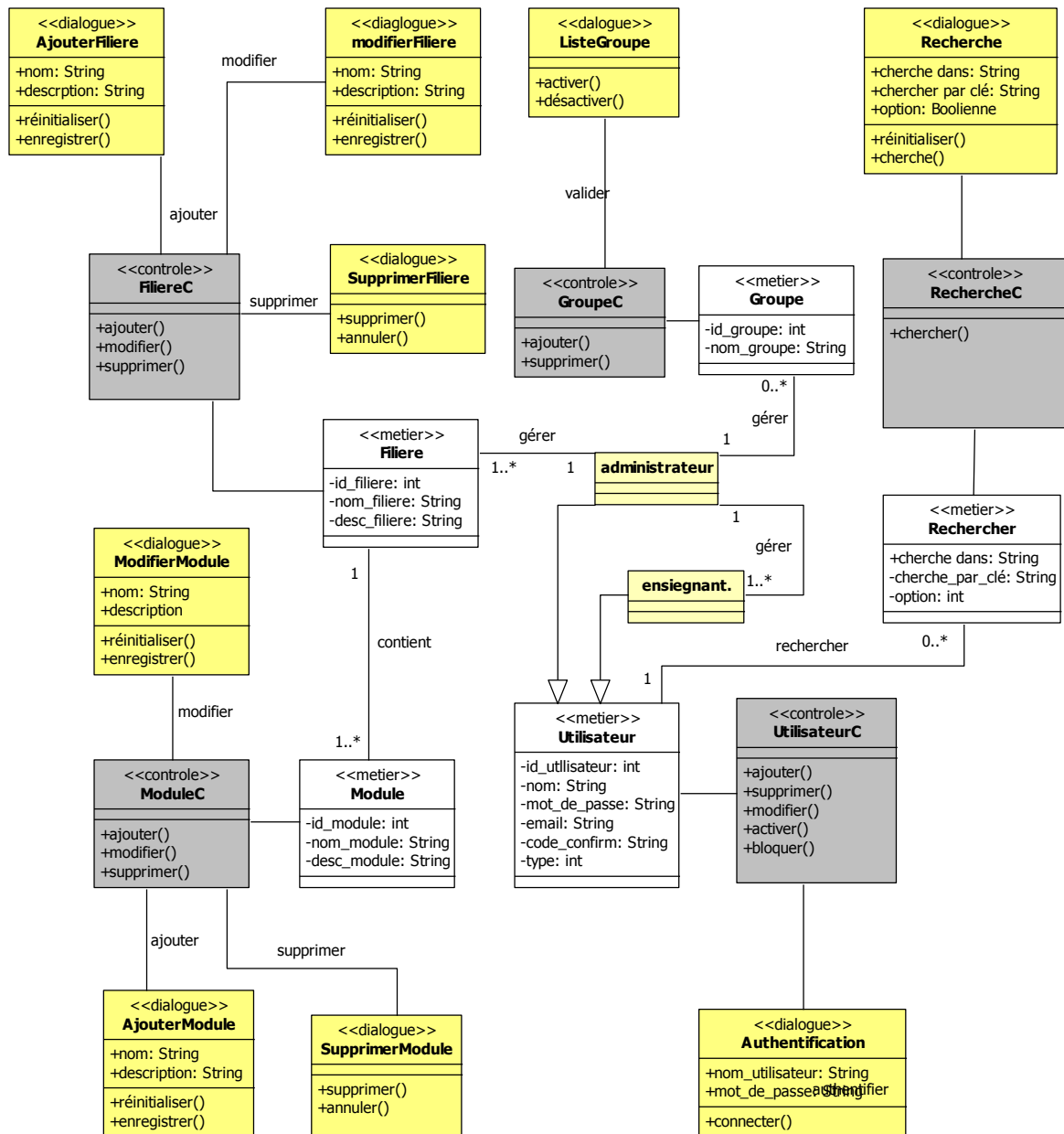


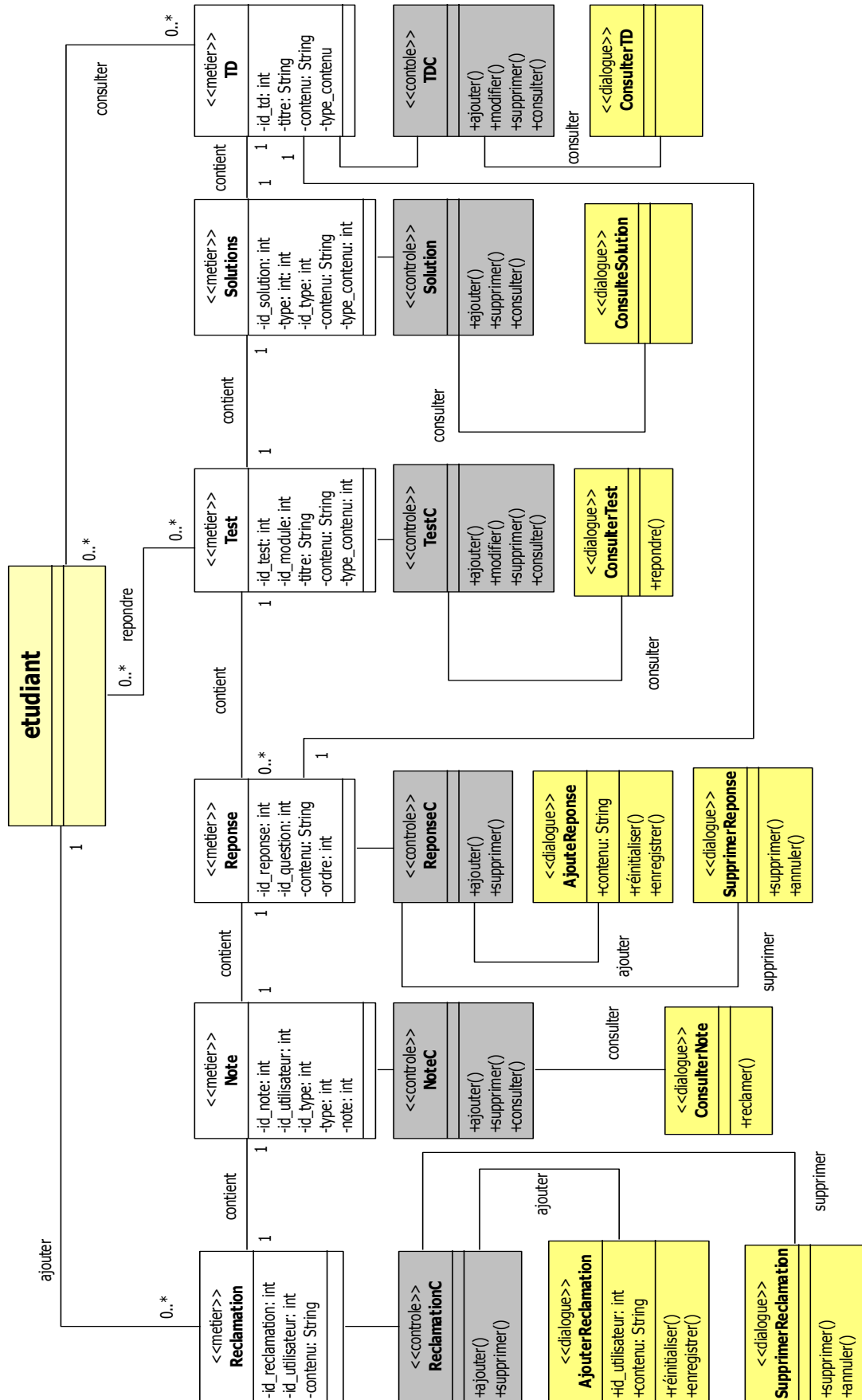
4.1.4. Enseignant



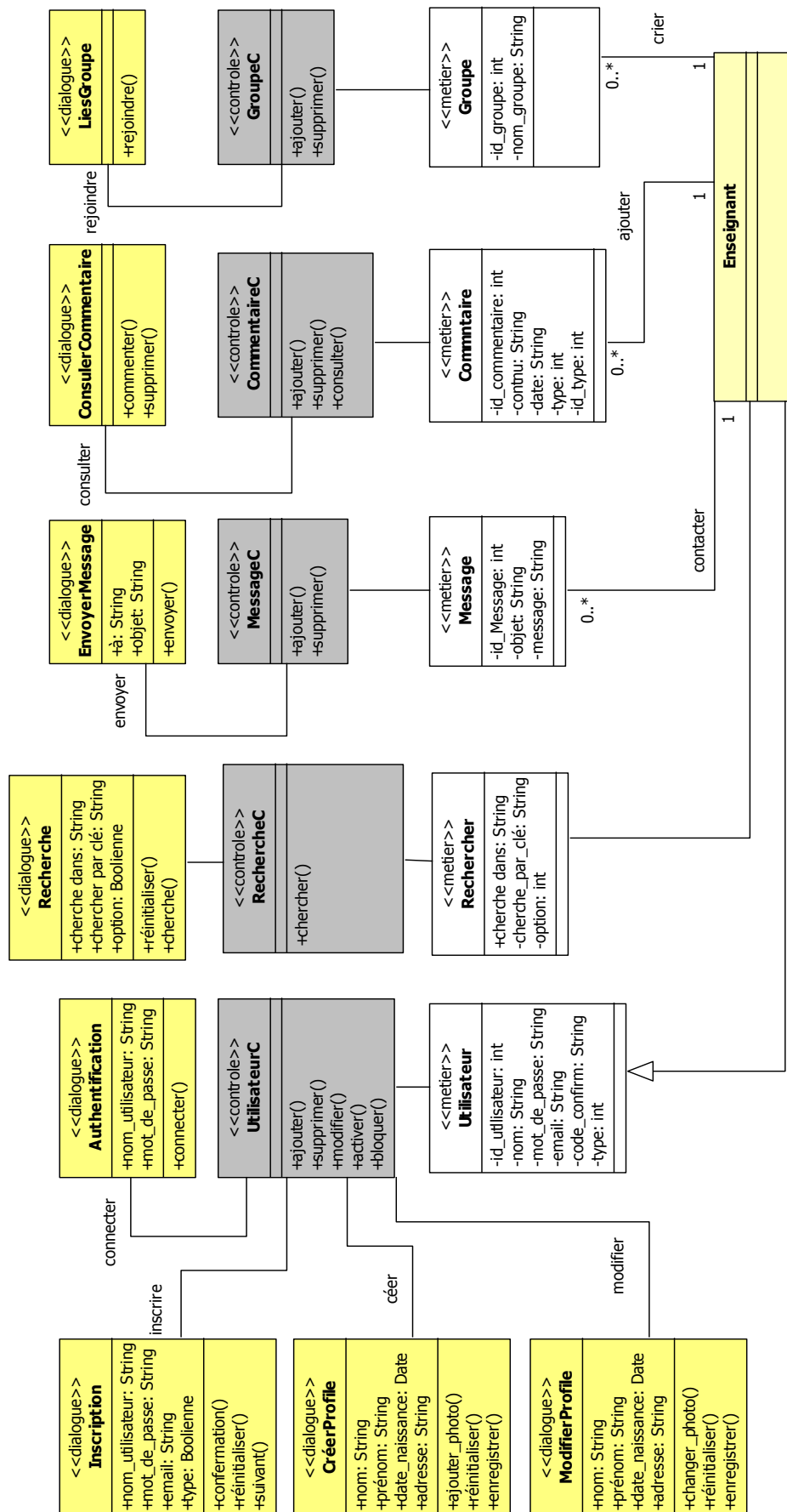
4.2. Les diagrammes de classe (métier-contrôle-dialogue)

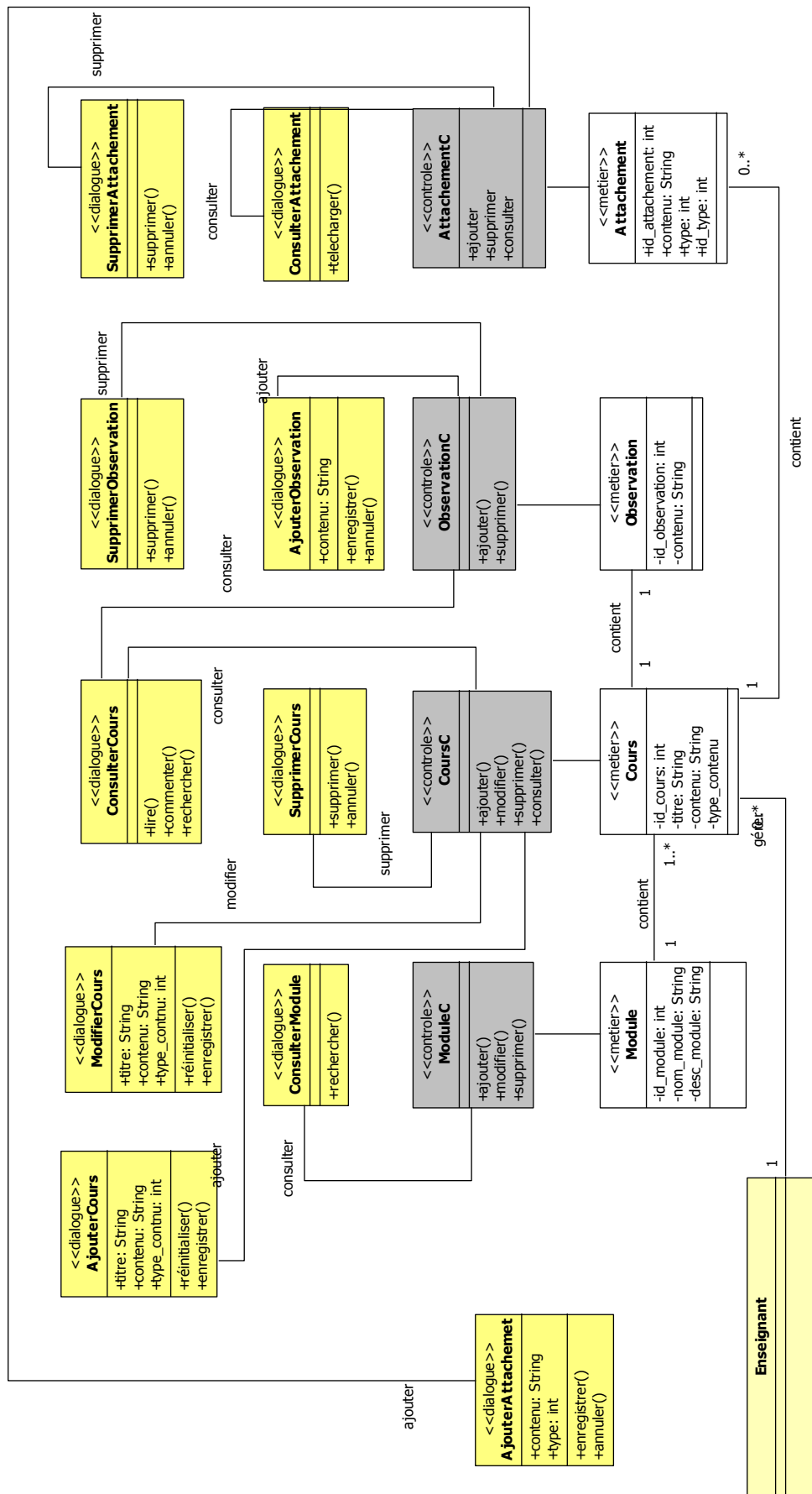
4.2.1. Administrateur

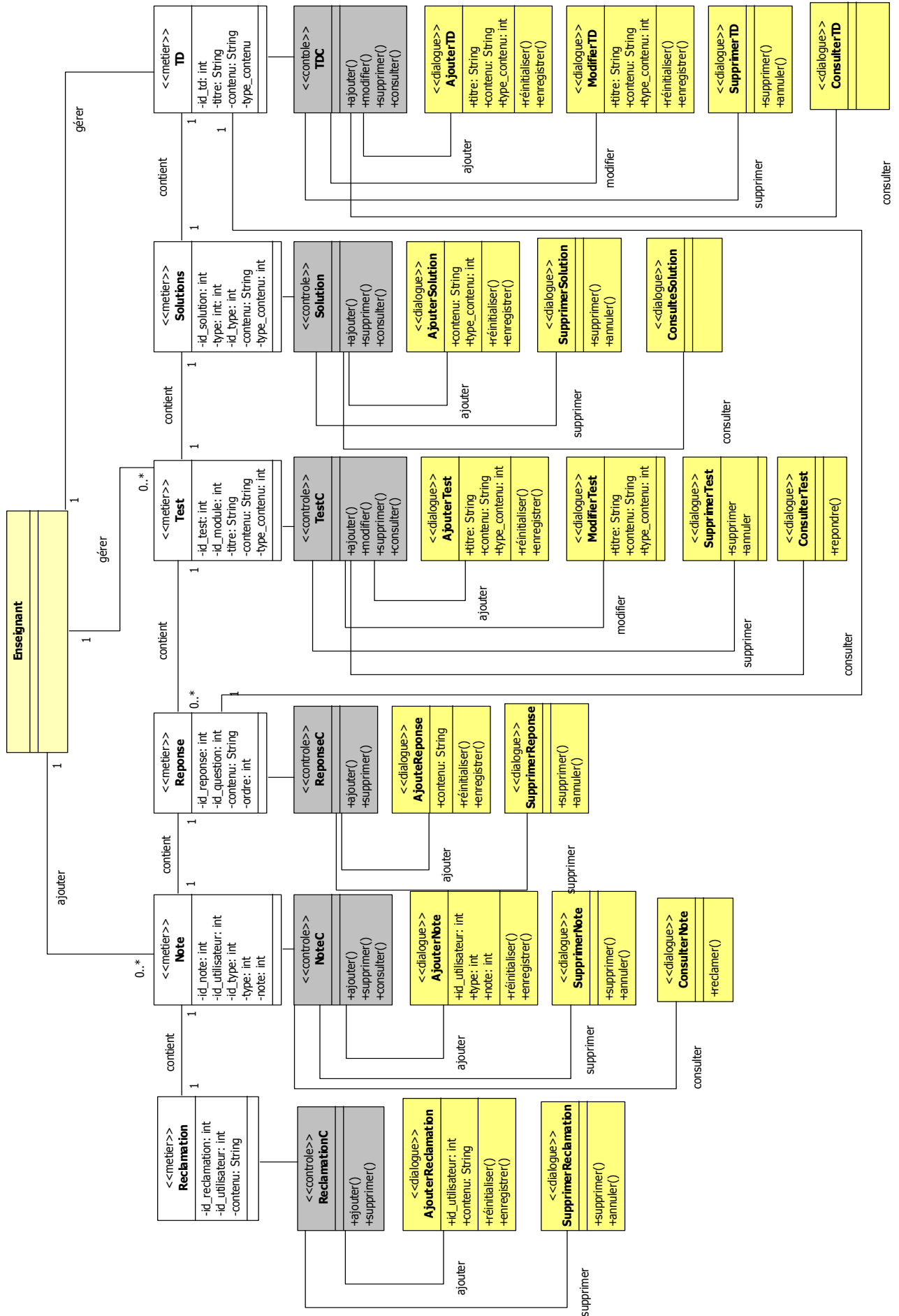


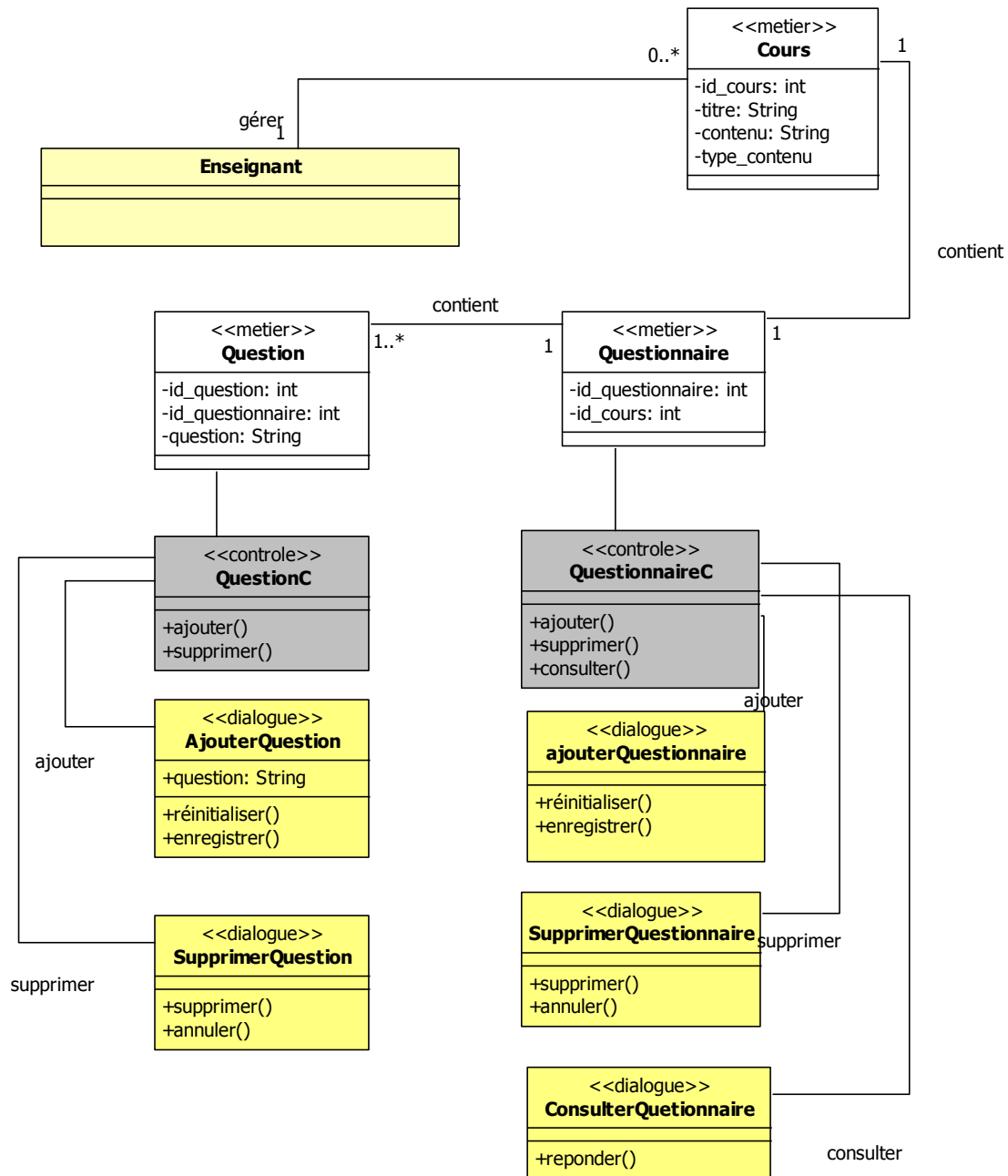


4.2.3. Enseignant









5. CONCLUSION

L'étape de d'analyse et de conception est l'étape la plus importante dans le cycle de développement, Car c'est le résultat de cette étape qui sera nous guidé dans l'étape d'implémentation.

Dans le chapitre suivant on va montrer comment nous implémenter le résultat de ce chapitre pour réaliser notre application.

CHAPITRE 4: IMPLÉMENTATION

1. INTRODUCTION

Après avoir analysé les besoins et défini la méthodologie de conception, nous allons implémenter notre conception via un environnement adéquat.

Dans ce chapitre, on va présenter les outils utilisés et les langages de programmation qu'on a utilisée pour l'implémentation et la réalisation de notre application web ainsi que présenterons quelques exemples de codes et des interfaces utilisateurs représentant les pages.

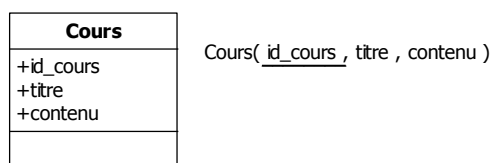
2. PRESENTATION DE L'ENVIRONNEMENT

2.1. Donnée

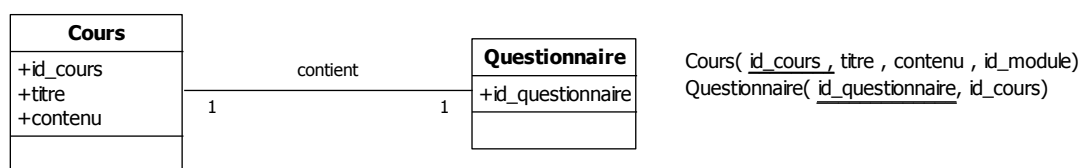
Dans cette partie on est en train de montrée les étapes qu'on a suivre pour traduire les classes métier qu'il obtenir dans chapitre 3 vers base des données relationnel, et pour cela on à respecter une méthodologie adapté.

2.1.1. Les règles de passages

- Classes avec Attribut

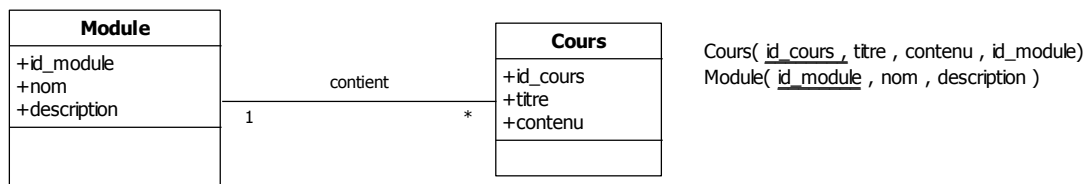


- Association 1 vers 1



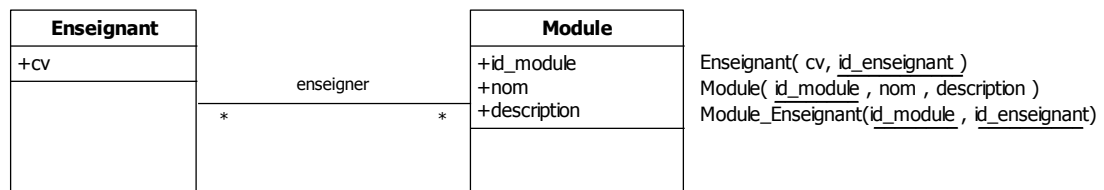
Remarque : id_cours dans la table Questionnaire est une clé étrangère.

- Association 1 vers plusieurs



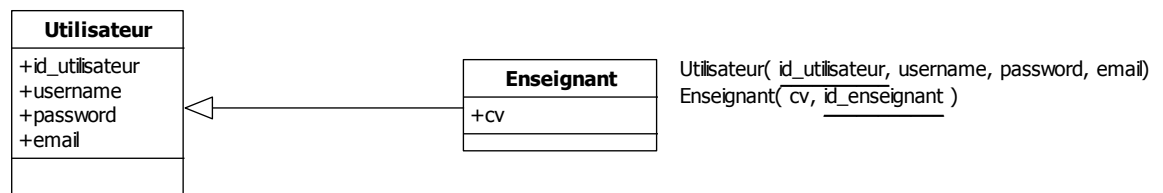
Remarque : id_module dans la table Cours est une clé étrangère.

- Association plusieurs vers plusieurs



Remarque : id_module et id_enseignant dans la table Module_Enseignant sont des clés étrangères.

- Héritage



Remarque : id_enseignant dans la table Enseignant est une clé étrangère.

2.1.3. Les tables de la base de données

- **Attached** (id_attached, content, content_type, type, type_id)
- **banned** (id_banned, id_user, type, id_type)
- **book** (id_book, name, author, pages, nbr_visits, nbr_downloads, photo, url, date_add, is_confirmed)
- **chat** (id_message, to, from, is_groupe, message, date_add, date_read)
- **comment** (id_comment, content, type, id_type, date, id_user, is_attached, id_quote)
- **cours** (id_cours, title, content, id_module, id_enseignant, is_attached, date_add, nbr_comment, nbr_vistors, nbr_visit, is_observed, is_questionary, content_type, date_comment, is_archived, nbr_td)
- **cours_groupe** (id_cours, id_groupe)
- **enseignant** (id_user, cv)
- **etudiant** (id_user, user_groupe)
- **filier** (id_filier, nom_filier, desc_filier, is_archived)
- **groupe** (id_groupe, name, id_user, nbr_users, is_activated)
- **message** (id_message, from, to, object, message, is_read, date, is_attached, is_deleted_from, is_deleted_to)
- **module** (id_module, nom_module, desc_module, id_filier, is_archived, nbr_cour)
- **module_enseignant** (id_enseignant, id_module)
- **note_td** (id_note, id_td, id_user, note)
- **note_test** (id_note, id_test, id_user, note)
- **observation** (id_observation, content, id_cour)
- **question** (id_question, id_questionary, question, note)
- **questionary** (id_questionary, id_cours)
- **question_response** (id_question, id_response)
- **reclamation** (id_reclamation, id_user, id_test, content, date)
- **response** (id_response, id_question, content, order)
- **solution_td** (id_solution, id_td, content, content_type)
- **solution_test** (id_solution, id_test, content, content_type)
- **td** (id_TD, title, content, id_cours, is_attached, date_add, date_comment, nbr_comment, nbr_visit, content_type, is_archived, has_solutions)

- **test** (id_test, title, id_module, content, content_type, date_start, delay, id_user, dely_reclamation, has_solutions, nbr_answers, date_reclamation)
- **test_groupe** (id_test, id_groupe)
- **user** (id_user, username, password, email, confirm_code, user_state, type, last_visite_date, nbr_added, block_date)
- **userinfo** (id_user, firstName, lastName, address, birthday, birthday_type, photo, sex)

2.2. Traitement

Dans cette partie de chapitre on parle sur l'implémentation des classes contrôle et classes dialogue en utilisant la technologie JAVA pour les traitements et pour la présentation en utilise des autre langages.

2.2.1. Les classes contrôles

2.2.1.1. Java

Java est un langage de programmation et une plate-forme informatique créée par Sun Microsystems en 1995. Il s'agit de la technologie sous-jacente qui permet l'exécution de programmes dernier cri. Java est utilisée sur plus de 850 millions d'ordinateurs de bureau et un milliard de périphériques dans le monde.

2.2.1.2. Servlet

Programme Java qui s'exécute dynamiquement sur le serveur Web et permet l'extension des fonctions de ce dernier, typiquement : accès à des bases de données, etc. Un Servlet peut être chargé automatiquement lors du démarrage du serveur Web ou lors de la première requête du client. Une fois chargés, les servlets restent actifs dans l'attente d'autres requêtes du client.

2.2.1.3. Utilisation de la technologie java dans les classes contrôles

L'implémentation des classes contrôles se fait en utilisant les servlets.

Ex

```
package servlets;
public class UserServlet extends HttpServlet{
    private static final long serialVersionUID = 1L;
    public static synchronized void addUser( beans.User new_User) throws SQLException{}
    public static synchronized void deleteUsers(String condition) throws SQLException {}
    public void updateLastVisit(int id_User, String date)throws SQLException {}
    public void activateChat(int id_User, int chat) throws SQLException{}
    public static synchronized void deleteUser(int id_User) throws SQLException{}
    public String[] getUserInfo(String id_User) throws SQLException{}
    public void editUserInfo(int id_User, String [] toChange , String [] new_Info)
throws SQLException{}
    public beans.User getUser(int id_User) throws SQLException {}
    public ArrayList<String> getUsersListNames(String q, int id_user) throws
SQLException{}
    public beans.User login(String userName , String password )throws SQLException{}
    public void doGet(HttpServletRequest request , HttpServletResponse response)throws
ServletException, IOException{}
    public void doPost(HttpServletRequest request , HttpServletResponse response)throws
IOException ,ServletException{}
}
```

Figure 10 Code source d'une classe de controle

2.2.1.4. Les filtres

2.2.1.4.1. Définition

Physiquement un filtre c'est un composant qui intercepte les requêtes aux ressources, dans une application web, il existe comme une partie d'une chaîne dont le dernier élément dans la chaîne c'est le ressource demandé, le filtre peut passer la requête à l'élément suivant dans la chaîne si c'est le dernier filtre alors passe la requête au ressource demandé.

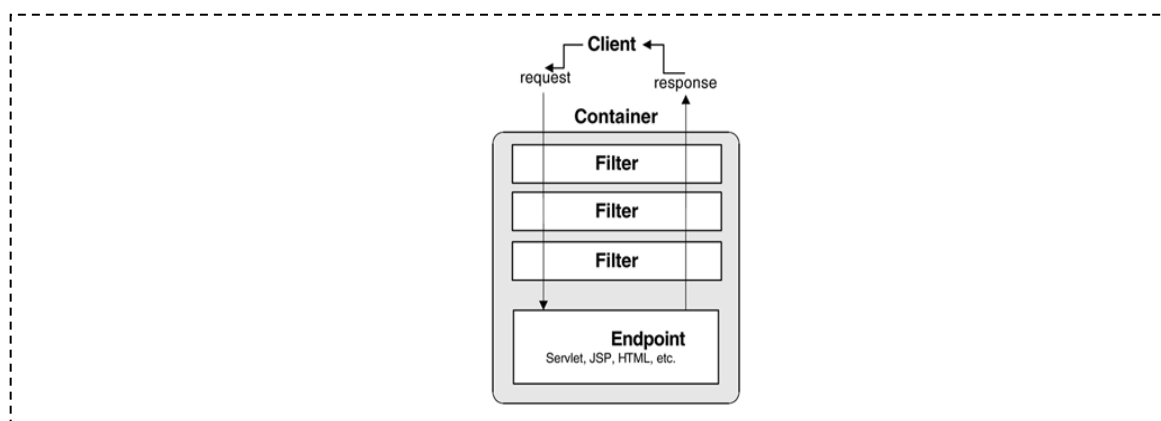


Figure11 Les filtres

2.2.1.4.2. Les filtres dans notre application

a. Le filtres de Connexion

Dans la programmation traditionnel (sans utiliser les filtres) pour vérifier que un visiteur est authentifié ou non , généralement on va inclure la page de vérification dans tous les page web de l'application , mais avec l'utilisation des filtres cette vérification sera faite d'une manier très simple , on a crié un filtres de connexion , le rôle de ce filtre est de vérifier à chaque requête, si l'utilisateur est authentifié ou non , dans le cas où il est authentifié le filtres passe la requête a l'élément suivant sinon il fait une redirection vers la page d'authentification.

b. Le filtres de cache

Le problème avec les pages dynamique est que au contraire des pages statique, les pages dynamique doit être générer à chaque demande de celui-ci, ce qui prend un temps, généralement c'est la meilleur technique pour répondre aux demandes variantes des utilisateurs, mais la demande de la même ressource par des déférents utilisateurs n'implique pas que chaque utilisateur doit avoir une propre réponse mais elle peut être la même réponse.

Dans ce cas la génération des pages à chaque demande est une perte de temps, la solution qu'on a adopté pour résoudre ce problème est de créer des filtres. Appelés les filtres de cache.

Le rôle de ces filtres est de mémoriser les pages les plus demandées, à chaque demande de cette page le filtre doit vérifier si la page existe déjà dans le l'emplacement des caches, si il existe il envoie la page au client sans la générer sinon il doit générer la page et sauvegarder une copie pour la prochaine demande, le filtre modifie les pages cachés dans deux cas :

- Si la durée de cache est écoulée.
- Si le page source (Servlet) a été modifié.

c. Le filtre de compression

Une autre technique que on a adopter dans notre site pour accélérer l'envoi des réponses est de les compresser avant les envoyer aux clients, le but de cette technique est de minimiser le plus possible la taille des réponses envoyées au client, le filtre de compression qu'on a utilisé, il doit vérifier initialement si le navigateur de client supporte cette technique si c'est le cas il envoie une réponse compressée au client (la plupart des navigateurs web supporte cette technique) sinon il envoie une réponse non compressée. La taille de la réponse envoyée peut être minimisée jusqu'à 80%.

Exemple : test de filtre avec l'outil « HttpWatch Basic ».

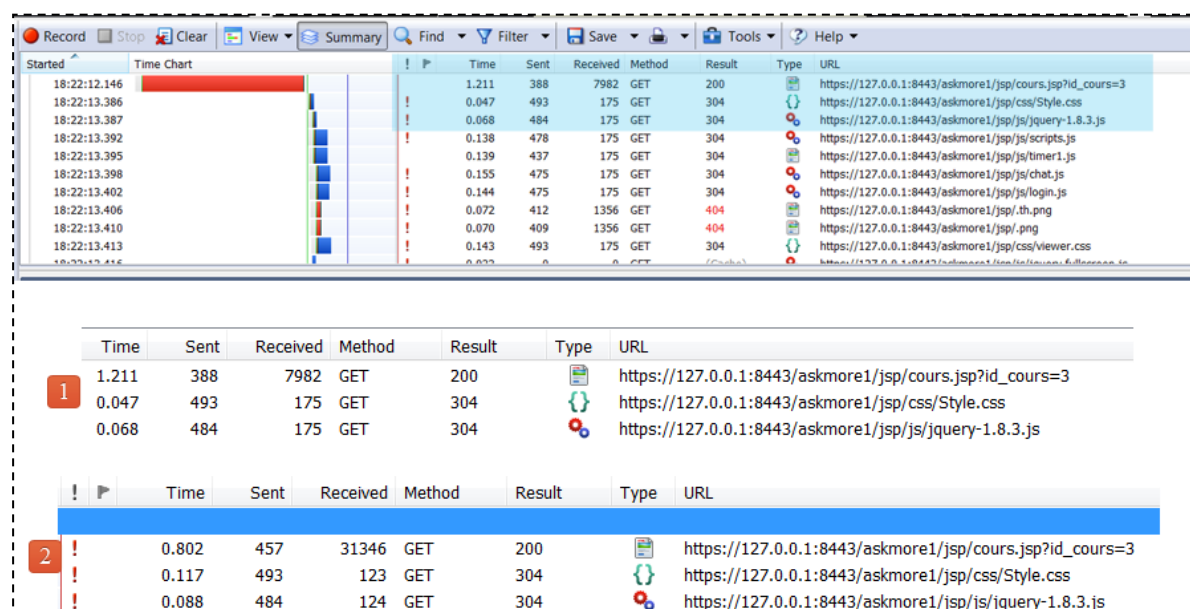


Figure 12 exemple de filtre de cache et de compression

- 2- Avec l'utilisation du filtre.
- 3- Sans l'utilisation du filtre.

2.2.1.5. SQL (Structured Query Language)

SQL est langage de communication standard avec les SGBDR. Il a été conçu par IBM en 1976, puis normalisé en 1986, 1992, 1999 et 2003.

Le succès que connaissent les éditeurs de SGBD relationnels a plusieurs origines et repose notamment sur SQL :

- Le langage est une norme depuis 1986, qui s'enrichit au fil du temps.
- SQL peut s'interfacer avec des langages de troisième génération comme C ou Cobol, mais aussi avec des langages plus évolués comme C++, Java ou C#. Certains considèrent ainsi que le langage SQL n'est pas assez complet (le dialogue entre la base et l'interface n'est pas direct).
- Les SGBD rendent indépendants programmes et données (la modification d'une structure de données n'entraîne pas forcément une importante refonte des programmes d'application).
- Ces systèmes sont bien adaptés aux grandes applications informatiques de gestion (architectures type client-serveur et Internet) et ont acquis une maturité sur le plan de la fiabilité et des performances.
- Ils intègrent des outils de développement comme les pré-compilateurs, les générateurs de code, d'états, de formulaires.

Exemple des requêtes SQL :

```
/*
 * creation prepared Statement
 * Insertion d'un nouveau TD dans la Base de données
 */
PreparedStatement state = connection.prepareStatement("INSERT
INTO `td` (`title`,`content`,`id_cours`,`is_attached`
`,`date_add`,`date_comment`,`content_type`) VALUES (?, ?, ?, ?, ?,
?, ?) ");
state.setString(1, new_TD.getTitle());
state.setString(2, new_TD.getContent());
state.setInt(3, new_TD.getId_cours());
state.setInt(4, new_TD.isIs_attached());
state.setString(5, new_TD.getDate_add());
state.setInt(6, new_TD.getDelay_comment());
state.setInt(7, new_TD.getContent_type());
state.executeUpdate();
```

2.2.2. Les classes dialogues

2.2.2.1. JSP (Java Server Page)

Extension de la technologie Java Servlet de Sun qui permet de programmer simplement l'affichage de contenus dynamiques sur le Web. JSP consiste en une page HTML incluant du code Java qui s'exécutera soit sur le serveur Web, soit sur le serveur d'application. Un exemple d'utilisation de JSP dans notre application web :

```
<%@page contentType="text/html" pageEncoding="utf-8" %>
<jsp:include page="content-top.jsp" >
    <jsp:param name="title_page" value="ASK-MORE : INDEX" />
</jsp:include>
<!-- Contenu de la page -->
.
.
.
<!-- La fin de Contenu de la page -->
<%@include file="content-bottom.jsp" %>
```

Figure 13 Code source Page JSP (Classe dialogue)

2.2.2.2. Java script

Langage de rédaction de scripts, destiné aux utilisateurs non spécialistes, qui permet d'intégrer des instructions Java préprogrammées dans la construction d'un document Web. Alors que le langage Java permet aux programmeurs de créer de nouveaux objets et des mini-applications, le langage JavaScript permet aux concepteurs de pages Web de fusionner des applets ou de les programmer de façon personnalisée, sans avoir à manipuler de code Java. Le langage JavaScript, qui est lui-même une application écrite en langage Java, a été développé par Sun et Netscape, originellement sous le nom de LiveScript.

Et on l'utilise pour la vérification des champs du formulaire avant l'envoi au serveur.

```
function Verifier_Email(mail) {
    var emailRegEx = /^[A-Z0-9._%+-]+@[A-Z0-9.-]+\.[A-Z]{2,4}$/i;
    if (mail.search(emailRegEx) == -1) {
        return false;
    } else {
        return true;
    }
};
```

Figure 14 Code source JavaScript

First Name * 4 type first Name X field is vide !! 1
e.g : Bessem

Last Name * type last Name 2
write your last name here

Sex * Male Ok 3

Figure 15 Résultat de Vérification d'un formulaire

- 1- Champ non valide avec l'affichage de l'erreur.
- 2- Champ pendant la modification.
- 3- Champ accepté.
- 4- Champ obligatoire.

2.2.2.3. Ajax (Asynchronous JavaScript and XML)

Acronyme d'Asynchronous JavaScript and XML, Ajax est une technique de développement d'applications Web interactives. Elle compile un ensemble de technologies couramment utilisées sur le Web dont HTML, CSS, XML, XSLT et JavaScript - pour afficher dynamiquement les informations.

En utilise cette technique dans le but d'améliorer la réactivité d'affichage de site Web en autorisant ce dernier à n'échanger que de petites quantités de données avec le serveur central plutôt que de rafraîchir la globalité de la page.

Par exemple dans l'envoi des fichiers au serveur, noter les réponses, dans le service de chat et ... etc.

```
$.ajax({  
  url: "../note.do", // Lien  
  type: "POST", // Methode  
  data: {  
    "request": "addNote" // La liste des données  
  },  
  success: function (res) {  
    callBack();  
  },  
  error: function (res) {  
  },  
  complete: function (res) {  
  }  
});
```

Figure 16 Code source AJAX

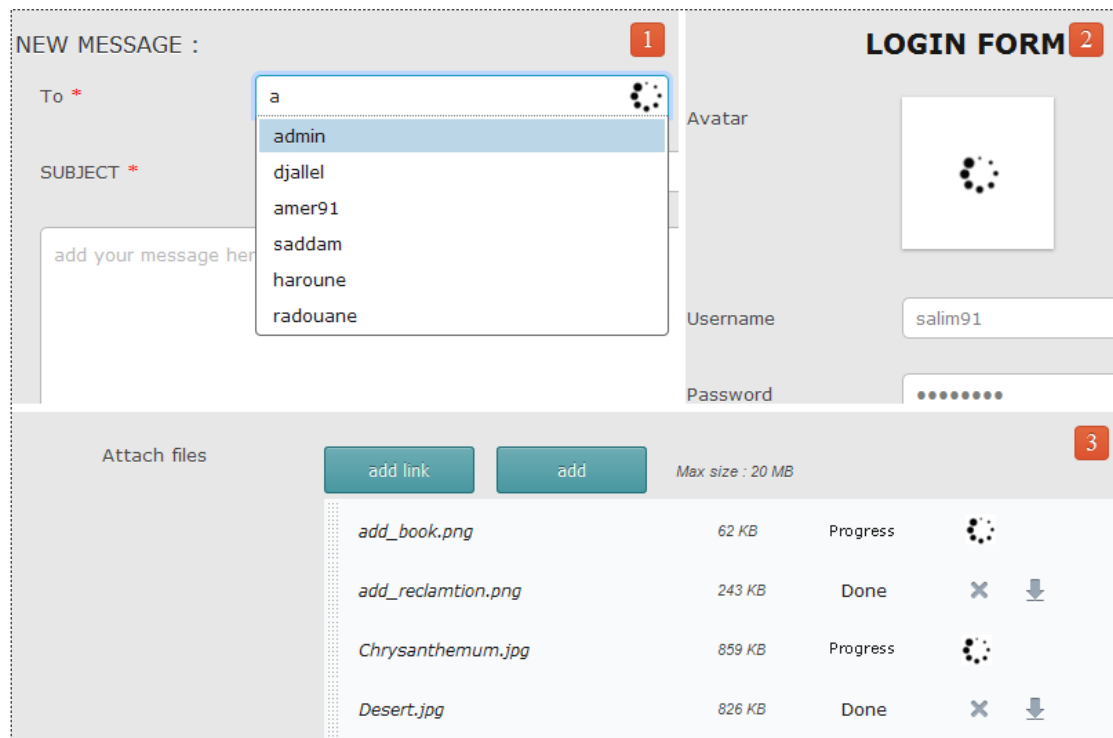


Figure 17 Exemple de l'utilisation de l'AJAX

- 1- L'auto-complète des noms des utilisateurs dans l'envoi des messages.
- 2- L'obtention de l'image de profile dans l'authentification automatiquement.
- 3- Uploader des fichiers au serveur (Ajouter des attachements).

2.2.2.4. JQuery

JQuery est une bibliothèque JavaScript, Le Framework JavaScript JQuery va vous permettre de coder rapidement et simplement des traitements à base de code JavaScript pour dynamiser et améliorer l'ergonomie de vos sites internet.

JQuery vous permet notamment :

```
// 1
$('.overlay:visible').hide();
// 2
$('a[href=#login]').click(function(){
    $('.overlay.login').slideDown();
    $('.overlay.login input[name=username]').focus();
    return false;
});
// 3
$('a[href=#formation]').click(function(){
    $(this).find('li').toggleClass('selected');
    $('.formation').slideToggle();
    if( $('.SiteSearch').css('display') != 'none' ){
        $('.SiteSearch').slideUp();
        $('a[href=#SiteSearch]').find('li').removeClass('selected');
    }
    return false;
});
// 4
self.after(ok).css('display','inline-block').addClass('has-success');
```

Figure 18 Exemple des codes sources JQuery

- 1- parcourir et manipuler le DOM (l'arbre des éléments HTML).
- 2- gérer des événements utilisateurs (clic souris, survole, ...).
- 3- ajouter des effets et animations visuels (fondu, disparation,...).
- 4- gérer les styles CSS et attributs des balises HTML.
- 5- gérer des interactions AJAX (communication asynchrone avec le serveur).

2.2.2.5. CSS (Cascading Style Sheets)

Les CSS, appelés *Cascading Style Sheets*, soit "Feuilles de style en cascades" en français, permettent de créer la mise en page d'un site. Avec les CSS, on peut faire toute la mise en page: Les couleurs de fond, de texte, les polices, les bordures ...etc.

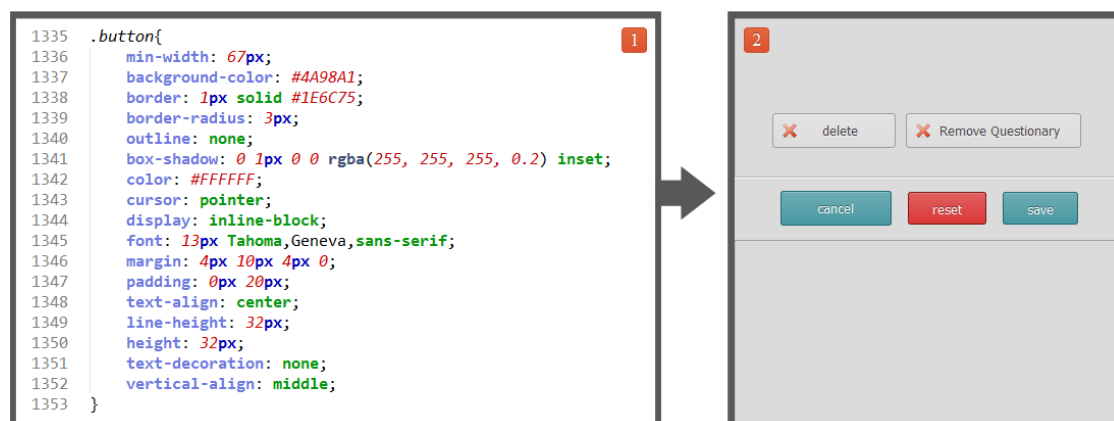


Figure 19 Exemple d'un code CSS

- 1- Le code source.
- 2- Résultat.

3. PLATEFORME

3.1. Plateforme matérielle

Notre travail a été réalisé sur 3 PC, dont les caractéristiques sont les suivant:

- PC 1 (Serveur):
 - **Nom** : HP Pavilion dv7.
 - **Microprocesseur** : Intel® Core™ 2 Duo CPU P8600 @ 2.40 GHz 2.40 GHz
 - **RAM** : 3.00 GB.
 - **Système d'exploitation** : Windows 7.
- PC 2 (Client):
 - **Nom** : Asus K50c.
 - **Microprocesseur** : Intel® Celeron® D CPU 220 @ 1.20GHz 1.50 GHz
 - **RAM** : 3.00 GB.
 - **Système d'exploitation** : Unix Ubuntu (Back track 5 R3).
- PC 3 (Client):
 - **Nom** : Lenovo.
 - **Microprocesseur** : Intel® Core™ 2 Duo CPU P8600 @ 2.00 GHz 2.30 GHz
 - **RAM** : 3.00 GB.
 - **Système d'exploitation** : Windows 8.

3.2. Plateforme logicielle (les outils)

Notre travail a été réalisé en utilisant plusieurs application en peut citer les exemples suivants:

Icone	Nom de logiciel	Rôle
	Eclipse	pour implémenter les codes JAVA et JSP.
	Apache Tomcat	serveur web.
	Mysql	(Xampp) pour la gestion de la base des données.
	Sublime	pour la création et l'implémentation des pages web (JavaScript, JQuery, Ajax).
	Adobe Dream Weaver	pour la mise en page (Html5, Css3).
	Adobe PhotoShop	Pour la création des interfaces, modification des photos, création des icones, organisation des couleurs.
	Les navigateurs	(Mozilla FireFox, Google Chrome, Opera). Pour tester et exécuter l'application.
	StarUML	Pour réaliser la conception (les diagrammes).

Figure 20 La liste des outils

4. DEPLOIEMENT DE L'APPLICATION

Un diagramme de déploiement décrit la disposition physique des ressources matérielles qui composent le système et montre la répartition des composants sur ces matériels. Chaque ressource étant matérialisée par un nœud, le diagramme de déploiement précise comment les composants sont répartis sur les nœuds et quelles sont les connexions entre les composants ou les nœuds. Les diagrammes de déploiement existent sous deux formes : spécification et instance.

4.1. Diagramme de déploiement

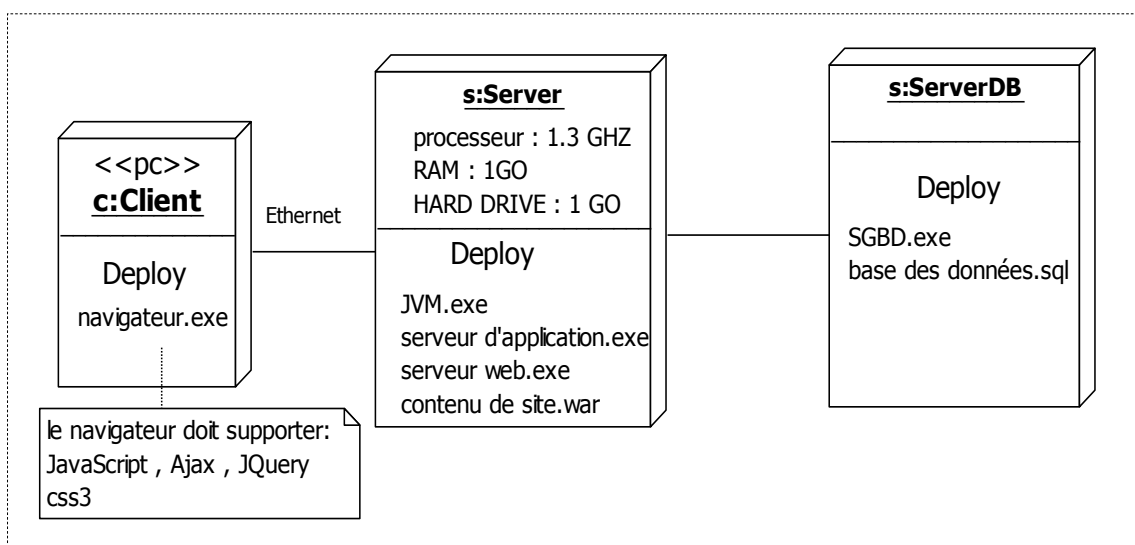


Figure 21 Diagramme de déploiement

4.2. Test d'exécution

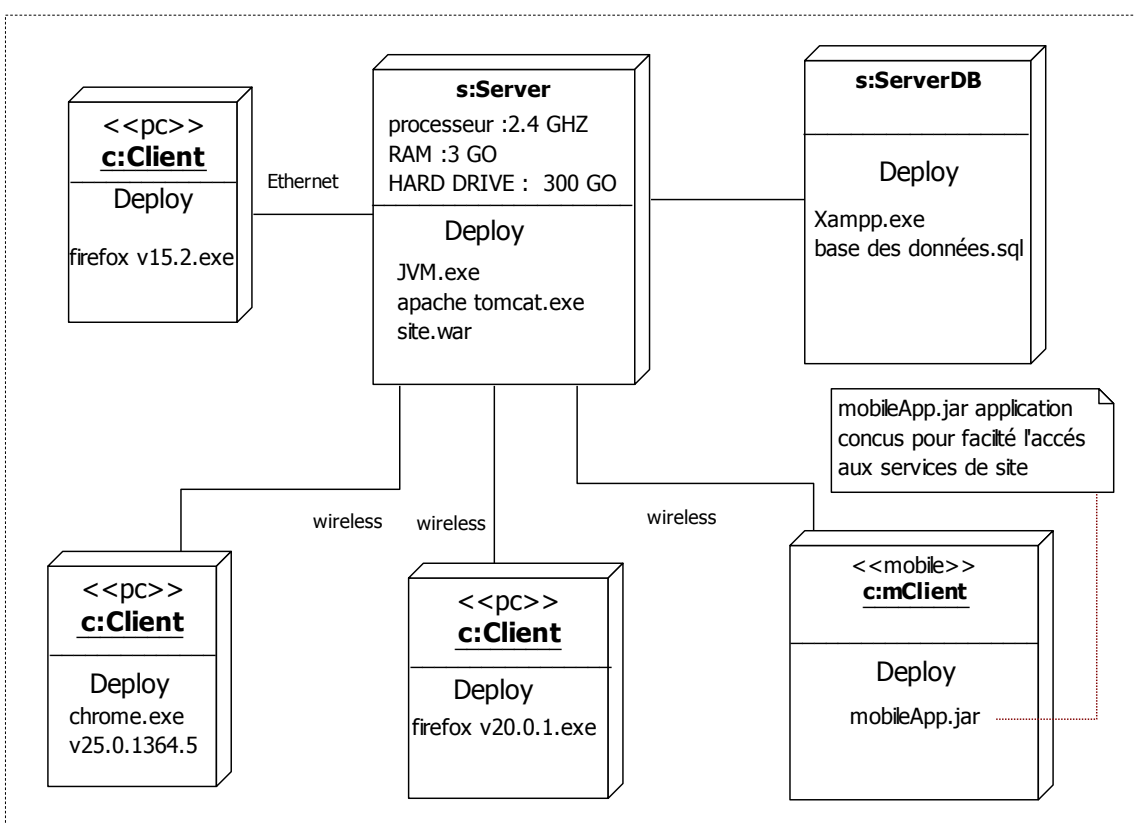


Figure 22 Exemple de teste

5. MODE D'EXECUTION DE L'APPLICATION

5.1. L'architecture client-serveur

5.1.1. Définition

L'architecture client-serveur est un modèle de fonctionnement logiciel qui peut se réaliser sur tout type d'architecture matérielle (petites ou grosses machines), à partir du moment où ces architectures peuvent être interconnectées.

5.1.2. Présentation de l'architecture à 3 niveaux

Dans l'architecture à 3 niveaux (appelée *architecture 3-tiers*), il existe un niveau intermédiaire, c'est-à-dire que l'on a généralement une architecture partagée entre :

- Un client, c'est-à-dire l'ordinateur demandeur de ressources, équipée d'une interface utilisateur (généralement un navigateur web) chargée de la présentation ;
- Le serveur d'application (appelé également **middleware**), chargé de fournir la ressource mais faisant appel à un autre serveur
- Le serveur de données, fournissant au serveur d'application les données dont il a besoin.

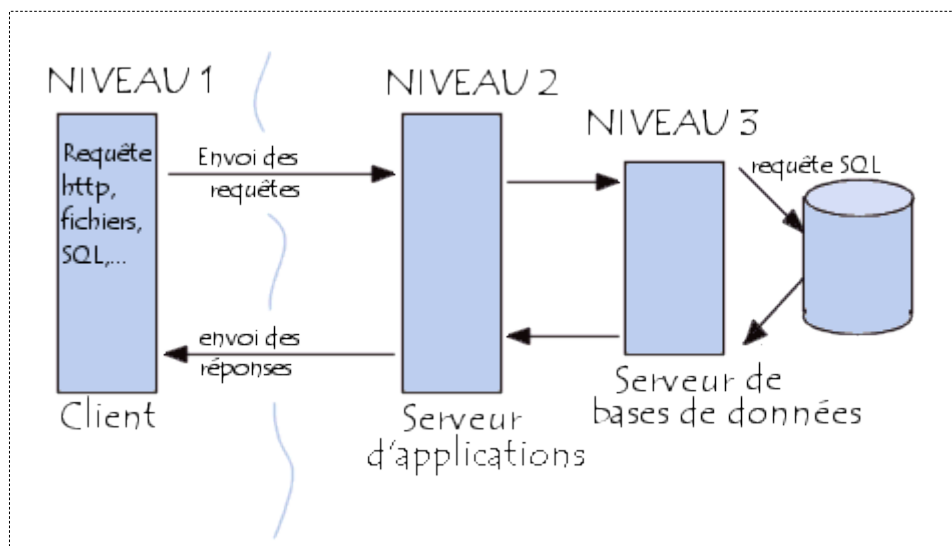


Figure 23 l'architecture Client/serveur à 3 niveaux

Etant donné l'emploi massif du terme d'architecture à 3 niveaux, celui-ci peut parfois désigner aussi les architectures suivantes :

- Partage d'application entre client, serveur intermédiaire, et serveur d'entreprise.

- Partage d'application entre client, serveur d'application, et serveur de base de données d'entreprise.

5.2. Modèle MVC

5.2.1. Définition

Très souvent en programmation Internet, les requêtes HTTP sont gérées par des composants Web qui reçoivent les requêtes, créent les réponses et les retournent aux clients. On a donc un seul composant responsable de la logique d'affichage, du logique métier et de la logique de persistance. Il existe une autre architecture appelée MVC ou Modèle Vue Contrôleur qui permet de séparer clairement les trois activités des composants impliqués.

L'architecture MVC sépare le logique métier de l'affichage. Dans ce modèle, un composant est chargé de recevoir les requêtes (Servlets), un autre traite les données (Classes) et un troisième gère l'affichage (JSP). Si l'interfaçage entre ces trois composants est clairement défini, il devient plus simple de modifier un composant sans toucher aux deux autres.

5.2.2. Le modèle MVC en pratique

a. Modèle : des traitements et des données

Dans le modèle, on trouve à la fois les données et les traitements à appliquer à ces données. Ce bloc contient donc des objets Java d'une part, qui peuvent contenir des attributs (données) et des méthodes (traitements) qui leur sont propres, et un système capable de stocker des données d'autre part. Rien de bien transcendant ici, la complexité du code dépendra bien évidemment de la complexité des traitements à effectuer par votre application.

b. Vue : des pages JSP

Une page JSP est destinée à la vue. Elle est exécutée côté serveur et permet l'écriture de gabarits (pages en langage "client" comme HTML, CSS, JavaScript, XML, etc.). Elle permet au concepteur de la page d'appeler de manière transparente des portions de code Java, via des balises et expressions ressemblant fortement aux balises de présentation HTML.

c. Contrôleur : des servlets

Une Servlet est un objet qui permet d'intercepter les requêtes faites par un client, et qui peut personnaliser une réponse en conséquence. Il fournit pour cela des méthodes permettant de scruter les requêtes HTTP. Cet objet n'agit jamais directement sur les données, il faut le voir comme un simple aiguilleur : il intercepte une requête issue d'un client.

Appelle éventuellement des traitements effectués par le modèle, et ordonne en retour à la vue d'afficher le résultat au client.

6. MESURE DE SECURITE

Toutes logiciels ont besoin de sécurisé leurs informations surtout les plus critique comme les informations de base de données et les informations les plus confidentiel. Notre application web est l'un de ces logiciels pour cela on a utilisé les mesures de sécurité pour la protéger.

6.1. Authentification

C'est le moyen ou l'utilisateur se connecté à son compte il a un nom de l'utilisateur et un mot de passe qui l'identifie.

6.2. Cryptage des données

6.2.1. SSL

Le Transport Layer Security (que l'on abrège « TLS »), anciennement nommé Secure Socket Layer (soit « SSL »), constitue un protocole de sécurisation des échanges sur le réseau Internet, développé à l'origine par Netscape. Le protocole sera ensuite renommé en Transport Layer Security (« TLS ») par l'IETF après le rachat du brevet de Netscape par l'IETF en 2001... Le groupe de travail qui correspond à l'IETF permettra la création de la RFC 2246. Par un abus de langage courant, on emploie l'expression « SSL » pour désigner indifféremment « SSL » ou « TLS »

• A quoi sert un certificat SSL ?

La technologie SSL (Secure Sockets Layer) est la norme mondiale dans le domaine de la sécurité Web. Un certificat SSL permet de garantir aux utilisateurs de votre site Web qu'ils ont accès à un site légitime et évite que les données sensibles soient interceptées ou altérées de manière frauduleuse.

6.2.2. MD5

MD5 (Message Digest 5) est une fonction de hachage cryptographique qui calcule, à partir d'un fichier numérique, son empreinte numérique (en l'occurrence une séquence de 128 bits ou 32 caractères en notation hexadécimale) avec une probabilité très forte que deux fichiers différents donnent deux empreintes différentes.

On a utilisé cette fonction pour crypter les mots des passes des utilisateurs.

6.3. Utilisation des adresses relative

L'utilisation des chemins relatifs pour interdire les accès direct aux ressources de l'application est l'une des techniques les plus utilisées pour protéger les ressources de l'application contre les attaques des pirates (hackers), les chemins réels peuvent être utilisés par les pirates pour ouvrir des failles.

Dans l'application on a utilisé les chemins relatifs dans deux façons :

6.3.1. Cacher les liens de téléchargement

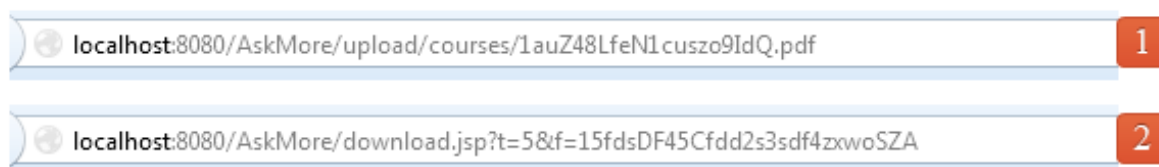


Figure 24 cacher les liens de téléchargement

- 1- Le chemin de fichier est clair.
- 2- Une page JSP avec des paramètres cryptés remplace le chemin de fichier

6.3.2. Les adresses relatives

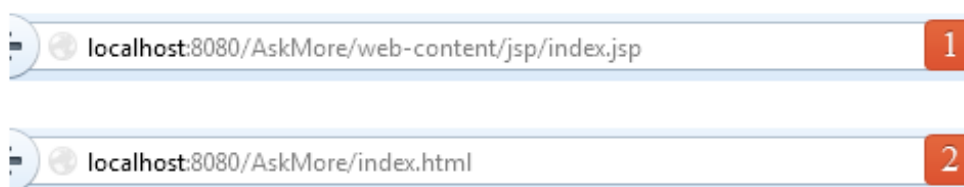


Figure 25 Les adresses relatives

- 1- Le chemin n'est pas relatif.
- 2- Le chemin est relatif.

7. INTERFACES UTILISATEUR: (PAGE WEB)

7.1. Page d'accueil

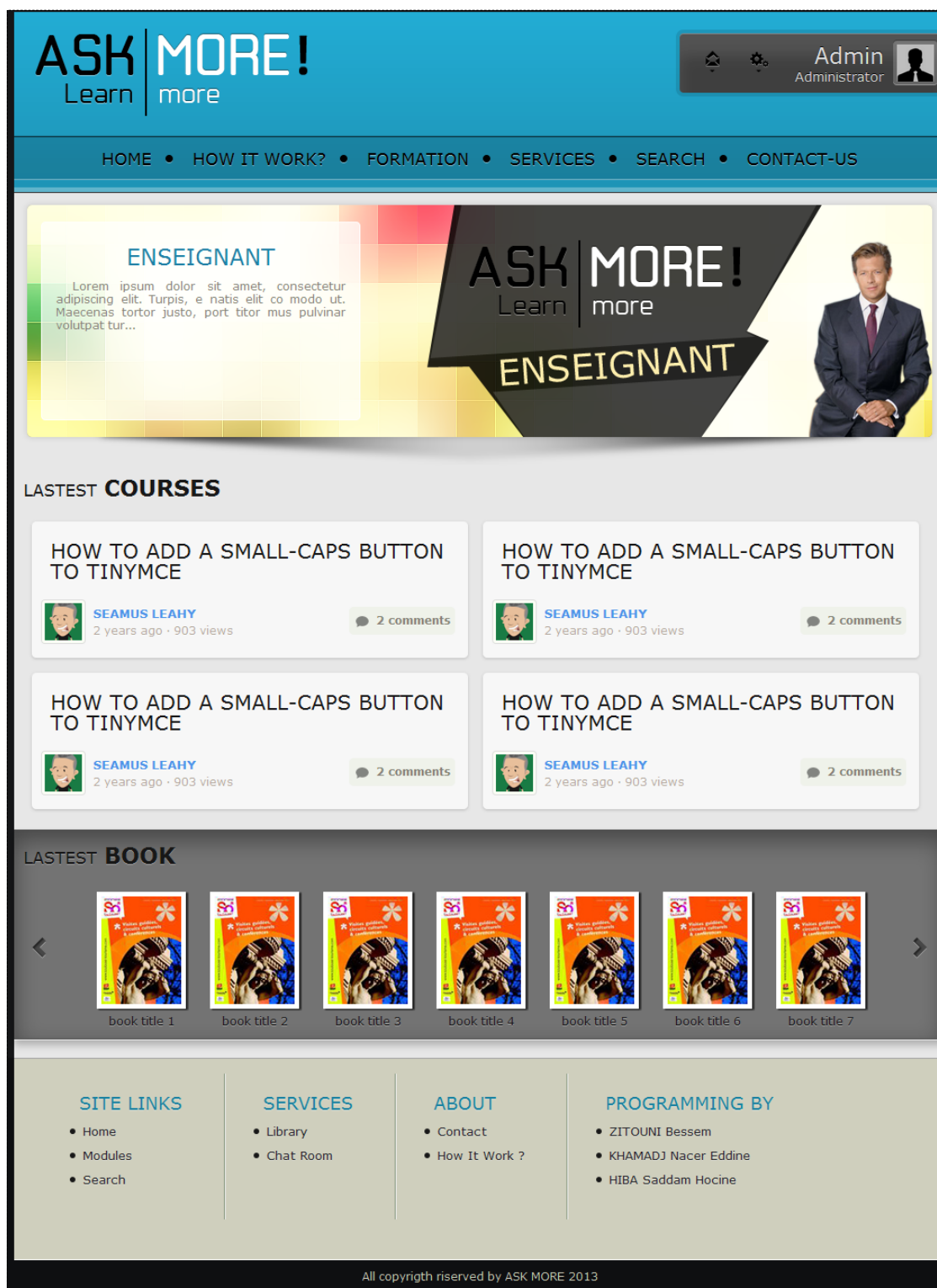


Figure26 Page d'accueil

7.2. La liste des formations

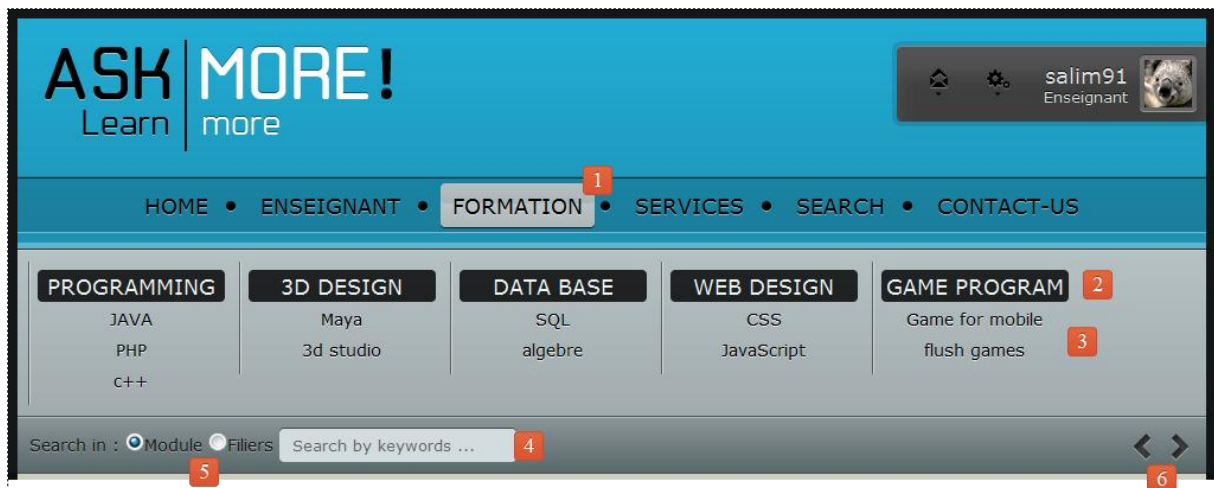


Figure 27 La liste des formations

- 1- Bouton d'accès rapide aux formations existes.
- 2- Nom de formation.
- 3- La liste des modules.
- 4- Recherche rapide dans la liste des formations.
- 5- Les options de recherche.
- 6- Des boutons pour le déplacement dans la liste.

7.3. Pages de Module

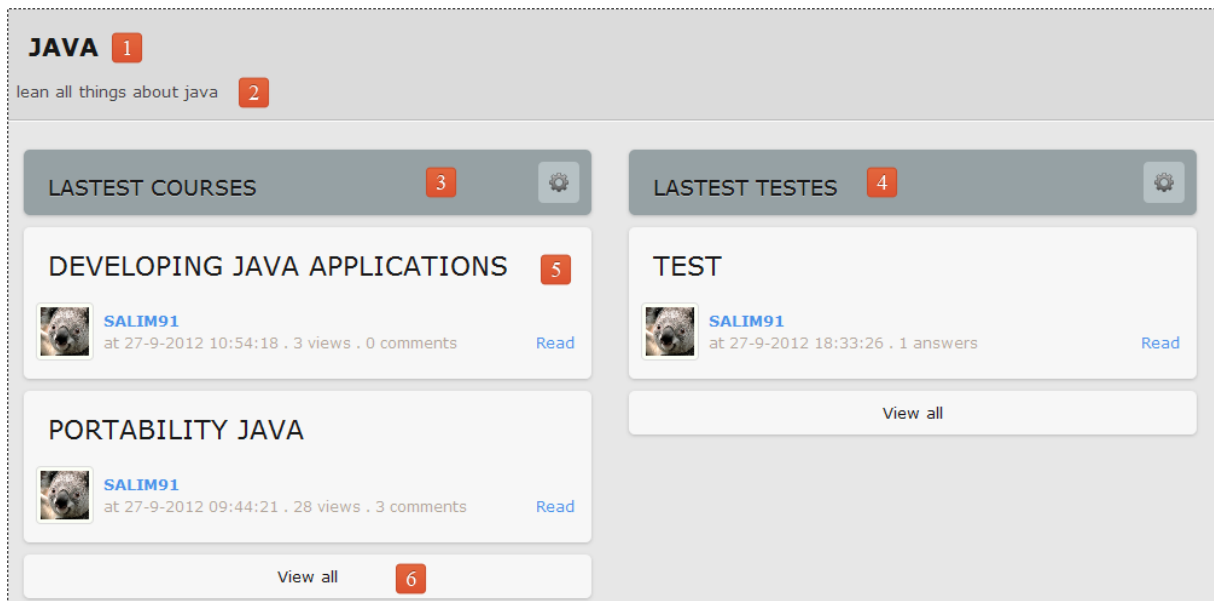


Figure 28 Page de Module

- 1- Le nom de module.
- 2- La description de module.
- 3- La liste des dernières cours de ce module.
- 4- La liste des dernières interrogations de ce module.
- 5- Un cours.
- 6- Afficher tous les cours de ce module.

7.4. Page Ajouter un cours

MODULE : **JAVA**

ADD NEW COURSE

Title * 1
e.g : jQuery and Ajax

Content type * TEXT PDF IMAGES SWF 2

3

4 **B** *I* U ABC | ↶ ↷ | ↵ | ☰ ☷

Comments delay * 5
e.g : 5 (jours)

Privacy 6 GROUPE1 GROUPE3
ex : choose groupes
important : don't choose any groupe for public

Attach files add link add 7 8 Max size : 20 MB

Important : * Required field

reset save 9

Figure29 Ajouter un cours

- 1- Champ de texte (titre de cours).
- 2- Type de contenu de cours (texte/html, fichier PDF, présentation des images, fichier flash).
- 3- Le contenu de cours (cas de texte/html).
- 4- Bar d'outils de traitement du texte.
- 5- Champ Numérique pour fixer un délai par jour au l'ajout des commentaires dans ce cours.
- 6- Attacher un lien.
- 7- Attacher un fichier.
- 8- Les boutons sauvegarder et annuler.

7.5. Page de cours

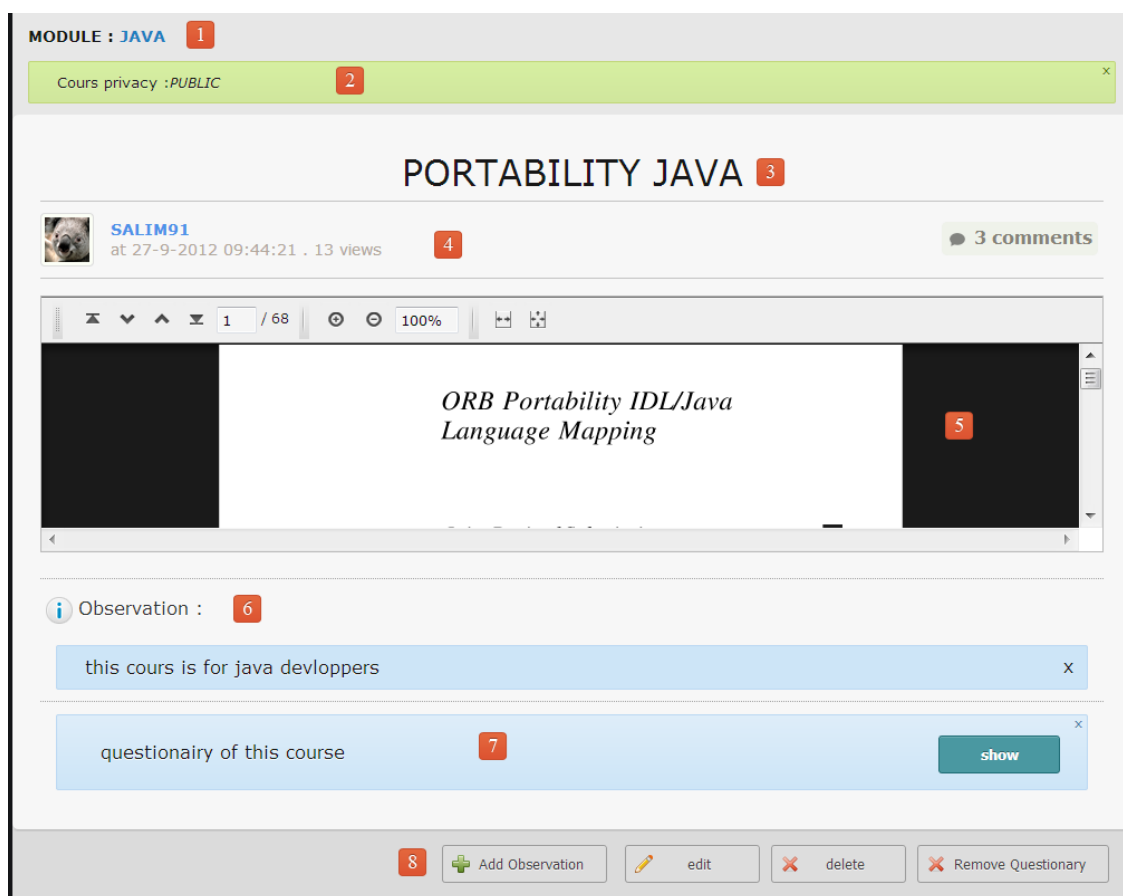


Figure30 Page de cours

- 1- Le module de cours (dans ce cas est JAVA).
- 2- Notification si le cours est privée ou disponible pour des groupes préféré.
- 3- Le titre de cours.
- 4- Les informations de l'enseignant qui ajout cette cours.
- 5- Le contenu de cours.

Remarque : le contenu possible de trouver on 4 types:

Texte/html, PDF comme ce cas, liste des images ou un Fichier de flash (vidéo).

- 6- La liste des observations de cours.
- 7- Le questionnaire de cours.
- 8- La barre d'outils de cours:
 - a. Ajouter ou supprimer le questionnaire.
 - b. Supprimer le cours.
 - c. Modifier le cours.
 - d. Ajouter une observation.

7.6. Ajouter un commentaire

The screenshot shows a 'LEAVE REPLY' form. At the top left is a speech bubble icon and the text 'LEAVE REPLY :'. Below this is a user profile picture of a koala. To the right of the profile picture is a large text input field with the placeholder text 'add your comment here ...'. Below the text field is a file upload area with a link icon, a paperclip icon, and the text 'Max size : 20 MB'. To the right of the file upload area is a checkbox labeled 'Send with (ENTRE)' and a blue 'Comment' button.

Numbered annotations:

- 1: 'LEAVE REPLY :'
- 2: 'add your comment here ...'
- 3: File upload area (link and paperclip icons)
- 4: 'Send with (ENTRE)' checkbox
- 5: 'Comment' button

Figure31 Ajouter un commentaire

- 1- Ajouter un commentaire.
- 2- Champ de texte (pour le contenu de commentaire).
- 3- Attacher un fichier ou un lien au commentaire.
- 4- Option pour envoyer le formulaire avec le bouton "Entrée".
- 5- Bouton de validation.

7.7. La liste des commentaires

The screenshot shows a 'COMMENTS' list. At the top left is a speech bubble icon and the text 'COMMENTS :'. Below this is a list of three comments. Each comment has a user profile picture, a username, a date, and a time. The first comment is from 'salim91 (Enseignant)' at '27-9-2012 09:47:12' with the text 'are there any comments.....'. The second comment is from 'toto (Etudiant)' at '27-9-2012 10:08:40' with the text 'why we should use a spring model?'. The third comment is from 'salim91 (Enseignant)' at '27-9-2012 10:10:07' with the text 'because is the best way to use a MVC model'. The third comment is highlighted in blue and contains a quote from 'toto at 27-9-2012 10:08:40' with the text 'why we should use a spring model?'. To the right of each comment are icons for replying, reporting, and deleting.

Numbered annotations:

- 1: 'COMMENTS :'
- 2: First comment text 'are there any comments.....'
- 3: Second comment text 'why we should use a spring model?'
- 4: Second comment date and time '27-9-2012 10:08:40'
- 5: Report icon (triangle with exclamation mark)
- 6: Reply icon (speech bubble)
- 7: Delete icon (trash can)
- 8: Third comment text 'because is the best way to use a MVC model'
- 9: Quote text 'why we should use a spring model?'


Figure32 La liste des commentaires

- 1- La liste des commentaires.
- 2- Un commentaire.
- 3- Contenu de commentaire.
- 4- Des informations sur le commentaire (date d'ajout et qui ajout).
- 5- Signaler le commentaire.
- 6- Reprendre au commentaire.
- 7- Supprimer le commentaire.
- 8- Modifier le commentaire.
- 9- Un emprunt sur un commentaire précédent.

7.8. Page d'interrogation

MODULE : JAVA 2

TEST

 **SALIM91**
at 27-9-2012 18:33:26 1

0 answer 4

Université Mentouri de Constantine Année 2007-2008
Faculté des Sciences de l'ingénieur 3ème LMD STIC
Département d'informatique Durée : 1H30 3

Correction du Contrôle Management de Projet

Partie 1 :(10 points)

★ NOTES : 5

NAME	NOTE	ACTION
toto	80 / 100	reclamer 6

👉 RECLAMATIONS : 7

toto at 27-9-2012 18:50:36 x

there is an error in the total please check it.

Figure33 Page d'interrogation

- 1- Les informations de l'enseignant qui ajout cette cours.
- 2- Le module de l'interrogation.
- 3- Le contenu de l'interrogation (format texte/html).
- 4- Barre d'outils de contenu :
Remarque : la barre contient les services : impression, téléchargement, modifier la taille de texte et afficher on plein d'écran.
- 5- La liste des notes.
- 6- Bouton pour réclamer sur la note.
- 7- La liste de des réclamations.

7.9. Noter une réponse

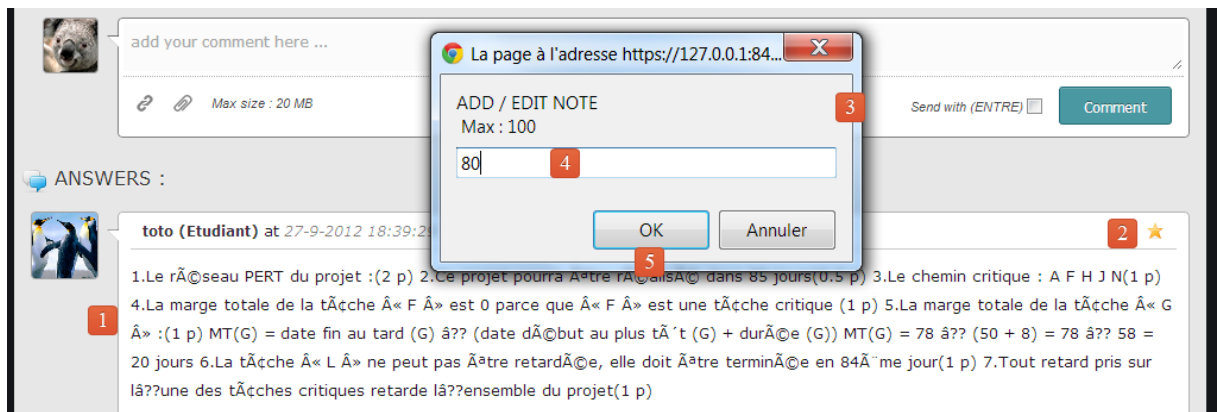


Figure 34 Noter une réponse

- 1- La réponse.
- 2- Bouton de ajouter ou modifier la note (si l'existe).
- 3- Formulaire d'ajouter un note (par JavaScript).
- 4- Champ de texte pour la note.
- 5- Valider la note.

7.10. Page d'Authentification

The screenshot shows a login interface with a dark theme. At the top, a navigation bar contains links: HOME • FORMATION • SERVICES • SEARCH • CONTACT-US. The main content area is divided into two sections. The left section, titled 'FORM', contains an 'Avatar' placeholder, a 'USERNAME' field with the text 'toto' (annotated with a red box 1), a 'PASSWORD' field with masked characters (annotated with a red box 2), and a profile picture of two penguins (annotated with a red box 3). Below these are 'login' (annotated with a red box 4) and 'cancel' (annotated with a red box 5) buttons. The right section, titled 'LOGIN', features a blue padlock icon, the word 'LOGIN', a 'forget your password ?' link (annotated with a red box 7), and a 'Register' link (annotated with a red box 6). At the bottom left, there is a 'forget your password?' link.

Figure35 Authentification

- 1- Champ de pseudo d'utilisateur.
- 2- Champ de mot de passe.
- 3- Image de profile.
- 4- Bouton d'authentification.
- 5- Bouton pour annuler l'authentification.
- 6- Lien de la page d'inscription.
- 7- Lien pour récupérer le mot de passe oublié.

7.11. Page d'inscription (étape 1)

REGISTER

1 Create your username 2 Confirm your email and activate your profile 3 Create your own profile **1**

Username * **2**
type username , minimum length is 5 character

Password * **3**
type password , min length is 5 character

Password Confirm * **4**
confirm your password

Email * **5**
type email , e.g : mail@server.dotr

Type * **6**
choose your type student or teacher

Important : * Required field

7

Figure 36 Page d'inscription étape 1

- 1- Barre de progression de l'inscription.
- 2- Champ de texte pour le nom de l'utilisateur.
- 3- Champ de texte pour le mot de passe.
- 4- Champ de texte pour confirmer le mot de passe.
- 5- Champ de texte pour l'adresse mail électronique.
- 6- Liste déroulante pour choisir le type de l'utilisateur (Enseignant/Étudiant).

7.12. Page d'inscription (étape 2 et 3)

The image shows two screenshots of a web application interface. The top screenshot is titled 'EMAIL VERIFICATION' and contains the following text: 'Thanks for signing up... Now it's time to get engaged!', 'Please check your email account for the **verification email**.', 'Once you've copy the code of confirmation in that email and typed in this form, you'll be able to login to **AskMore** web-app'. Below this is a form with a label 'Confirm code *', a text input field with placeholder text 'paste confirm code here ...', and a red square icon with the number '2'. Below the input field is the text 'type code you recived in your email'. At the bottom of the form are two buttons: 'next' and 'logout'. The bottom screenshot shows a progress bar with three steps: '1 Create your username', '2 Confirm your email and activate your profile', and '3 Create your own profile'. The second step is highlighted in green. Below the progress bar is the heading 'CONGRATULATIONS' followed by a red square icon with the number '4'. The text below reads: 'your account is activated Now you can create your [Profile](#)', where the 'Profile' link is followed by a red square icon with the number '5'. Below this is the text 'get started with the following:' followed by a list of three items: '1. Make connections !', '2. Simply fill out your personal profile to build the link with other members sharing your interests.', and '3. Start exploring! Use the search field or formation list.'. At the bottom of the page is the text 'Welcome to **AskMore** !!'. At the very bottom are two buttons: 'Profile' (with a red square icon with the number '5' to its left) and 'logout'.

Figure 37 Page d'inscription étape 2 et 3

- 1- Etape numéro 2 (confirmation de compte).
- 2- Champ de texte pour le code de confirmation.

Remarque : le code de confirmation a été envoyé à l'adresse mail.

- 3- Etape 3 création de profile.
- 4- Félicitation le compte a été confirmé.
- 5- Lien de profile pour complété la création de profile.

7.13. Les pages de profile et modifier compte

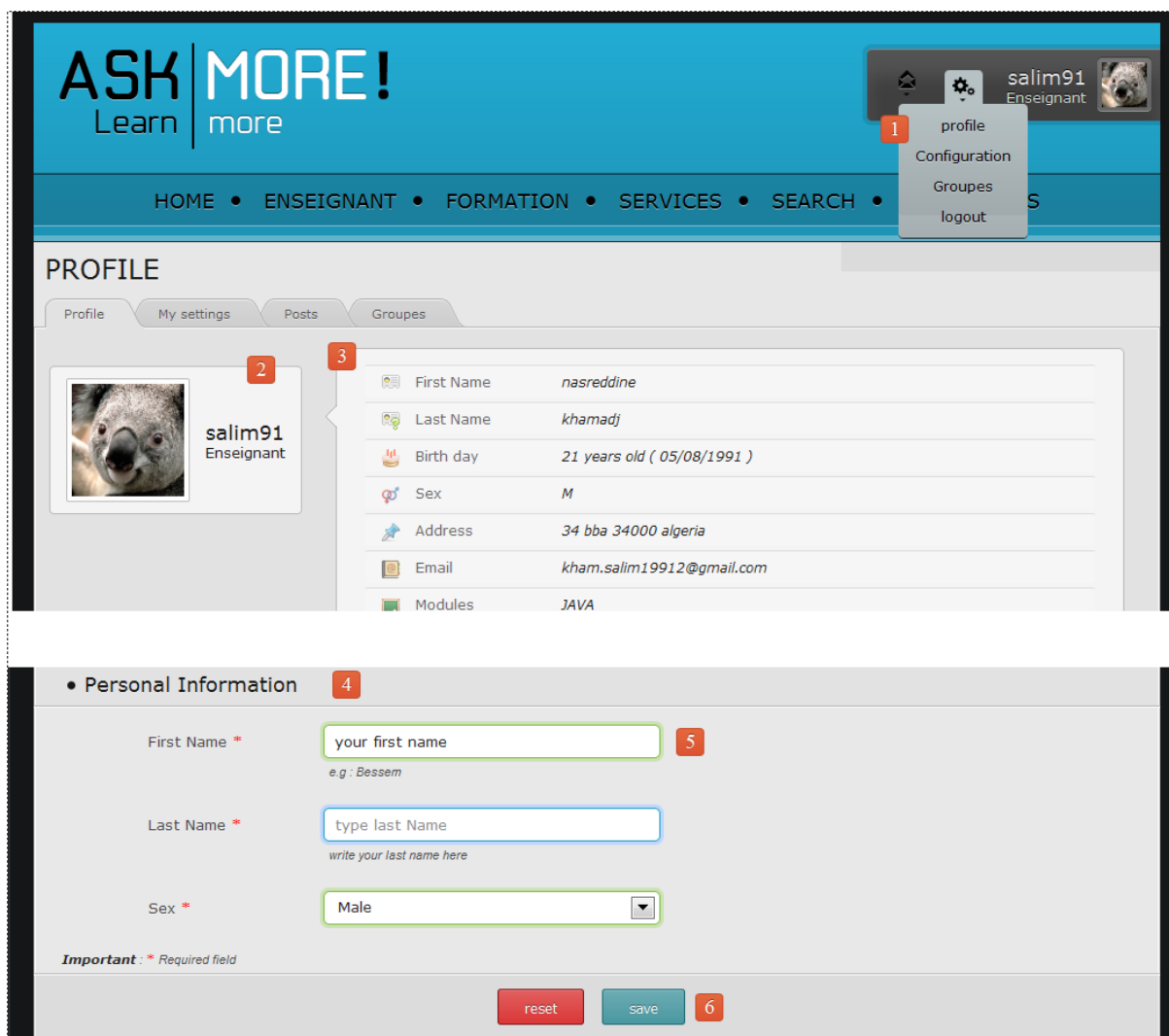


Figure 38 Le profil de l'utilisateur

- 1- Une liste déroulante pour les options de profile.
- 2- Le nom et la photo de l'utilisateur.
- 3- Les informations personnelles de l'utilisateur.
- 4- Une partie de la modification de profile.
- 5- Formulaire de la modification des informations personnelles.
- 6- Les boutons de l'envoi et reset de formulaire.

7.14. La messagerie (la page des messages reçus)

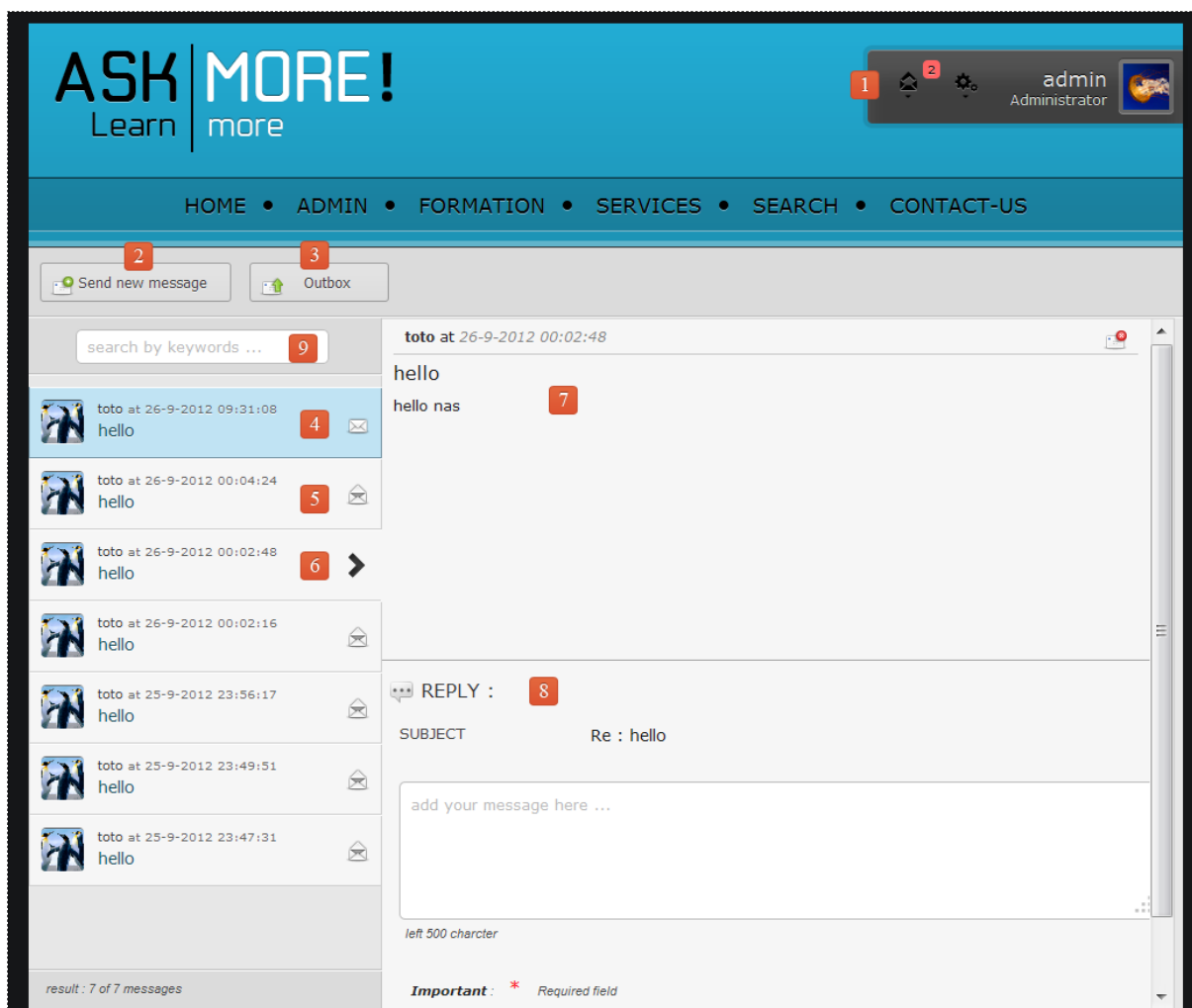


Figure 39 La messagerie

- 1- Bouton d'accès rapide à la messagerie avec une notification des nouveaux messages.
- 2- Envoyer un nouveau message.
- 3- Ouvrir la boîte des messages envoyés.
- 4- Un nouveau message ne pas lus.
- 5- Un message déjà lus.
- 6- Un message pendant la lecture.
- 7- Contenu de message.
- 8- Formulaire pour reprendre au message.

7.15. L'espace de discussion

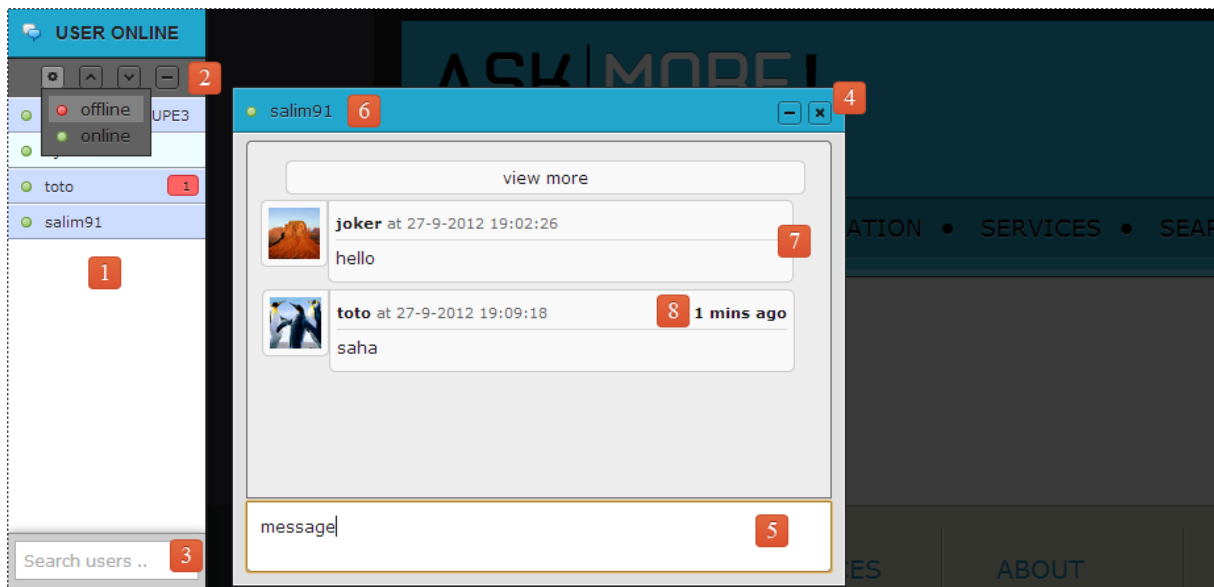


Figure 40 l'espace de discussion (CHAT)

- 1- La liste des utilisateurs connectés et les groupes des utilisateurs.
- 2- La configuration de service (cacher, activé, désactivé).
- 3- La recherche dans la liste (rapide).
- 4- Block de discussion avec un utilisateur.
- 5- Champ de texte (le contenu de message à envoyer).
- 6- Le nom de l'utilisateur et son état.
- 7- Un message.
- 8- Un message déjà vu avant 1 minute.

7.16. Les pages de la bibliothèque et de livre

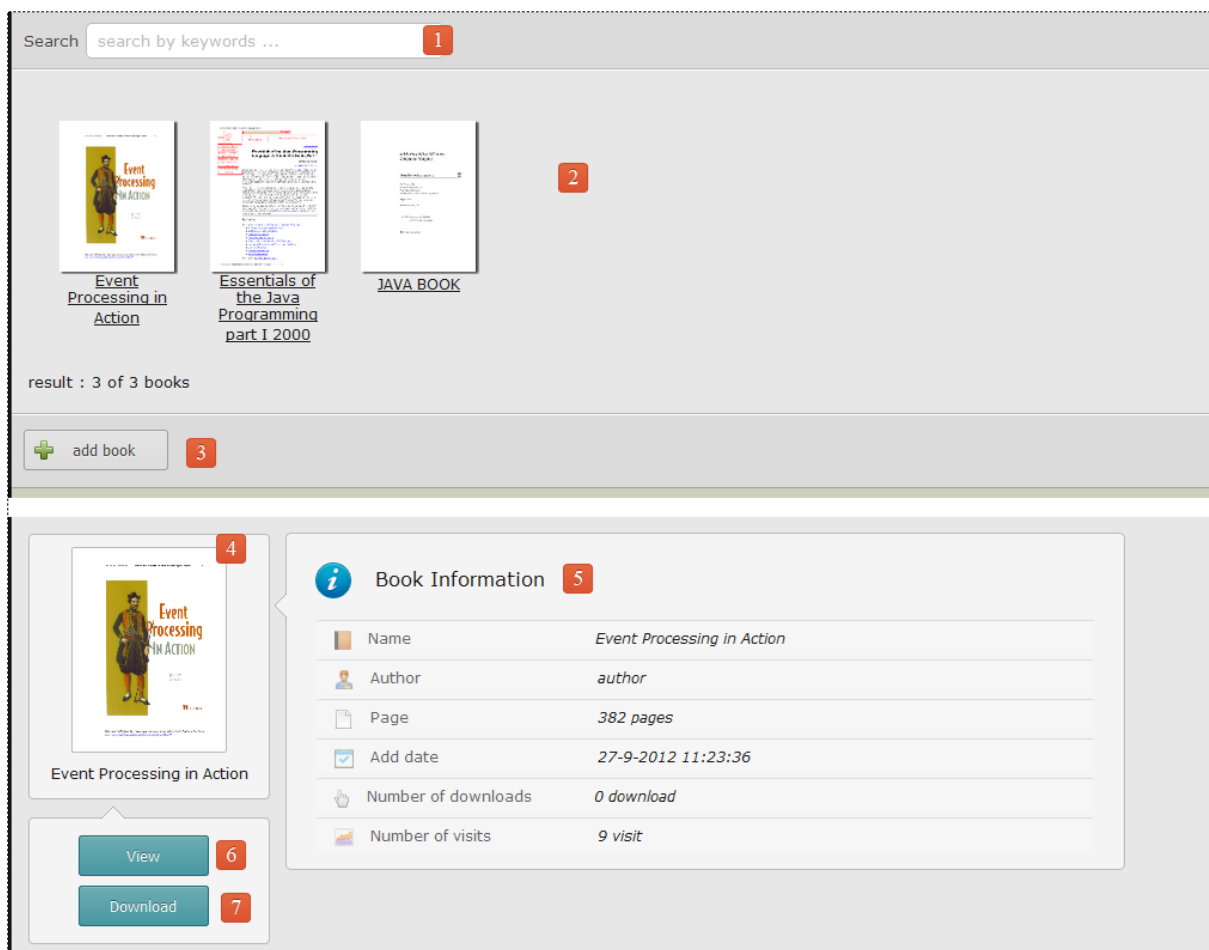


Figure 47 Bibliothèque

- 1- La recherche dans la bibliothèque.
- 2- La liste des livres.
- 3- Proposer un nouveau livre.
- 4- La premier page de livre et son titre.
- 5- Des informations sur le livre.
- 6- Consulter le livre.
- 7- Télécharger le livre.

7.17. La page d'administration des utilisateurs

The screenshot shows a web application interface for user management. At the top is a navigation bar with links: HOME, ADMIN, FORMATION, SERVICES, SEARCH, and CONTACT-US. Below this is a search bar labeled 'Search' with the placeholder text 'search by keywords ...' and a red callout '1'. To the right of the search bar are three checkboxes: 'Administrateur' (checked), 'Enseignant', and 'Etudiant'. Below the search bar is a table of users. The table has columns: ID, USERNAME / TYPE, EMAIL, ACTIVATION, BLOCK, and ACTION. The table contains five rows of user data. A red callout '2' points to the table header. A red callout '3' points to the 'Acivate' button in the 'ACTIVATION' column for user 'Djallel'. A red callout '4' points to the 'Deactivate' button in the 'ACTIVATION' column for user 'Toto'. A red callout '5' points to the 'Deny' button in the 'BLOCK' column for user 'Djallel'. A red callout '6' points to the 'Delete' button (represented by a trash icon) in the 'ACTION' column for user 'Djallel'. At the bottom of the table, there is a status bar that says 'result : 5 of 5 Users' with a red callout '7'.

ID	USERNAME / TYPE	EMAIL	ACTIVATION	BLOCK	ACTION
5	Djallel Etudiant	djallel@gmail.com	<button>Acivate</button> 3	<button>Deny</button> 5	<button>Delete</button> 6
4	Joker Etudiant	kham.salim3321@gmail.com	<button>Acivate</button>	<button>Deny</button>	
3	Toto Etudiant	kham.salim12@yahoo.fr	<button>Deactivate</button> 4	<button>Deny</button>	
2	Salim91 Enseignant	kham.salim19912@gmail.com	<button>Deactivate</button>	<button>Deny</button>	
1	Admin Administrateur	kham.salim@gmail.com	<button>Deactivate</button>	<button>Deny</button>	

result : 5 of 5 Users 7

Figure 42 L'administration des utilisateurs

- 1- Champ de recherche un utilisateur par le nom.
- 2- La liste des utilisateurs.
- 3- Bouton pour activer le compte de l'utilisateur.
- 4- Bouton pour désactiver le compte d'utilisateur.
- 5- Bouton pour bloquer l'utilisateur.
- 6- Bouton pour supprimer l'utilisateur.
- 7- Le nombre des utilisateurs affichés dans la page.

7.18. Page de statistique des cours

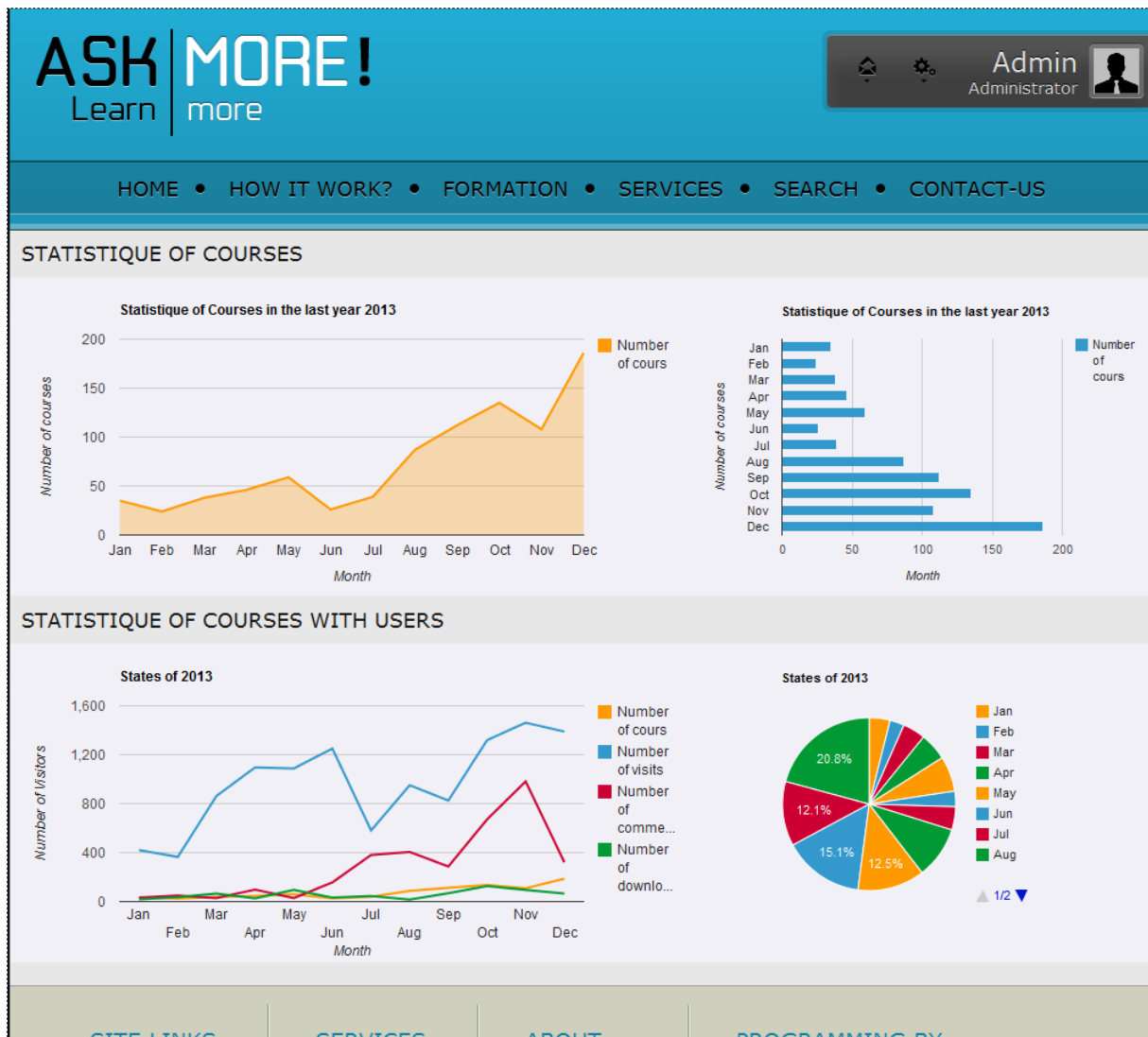


Figure 43 Page des statistiques

8. L'APPLICATION MOBILE

Pour faciliter l'accès aux services de notre application web, on a développé une application mobile utilisant la technologie J2ME (java 2 mobile édition), cette application permet à l'utilisateur inscrit dans notre application web d'accéder au service de chat instantané et au service de messagerie à partir son téléphone portable (ce téléphone doit supporter la technologie J2ME).

Pour la création de cette application mobile on a utilisé l'architecteur client / serveur qu'est implémenté en JAVA par des sockets.

Quelque capture d'écran :

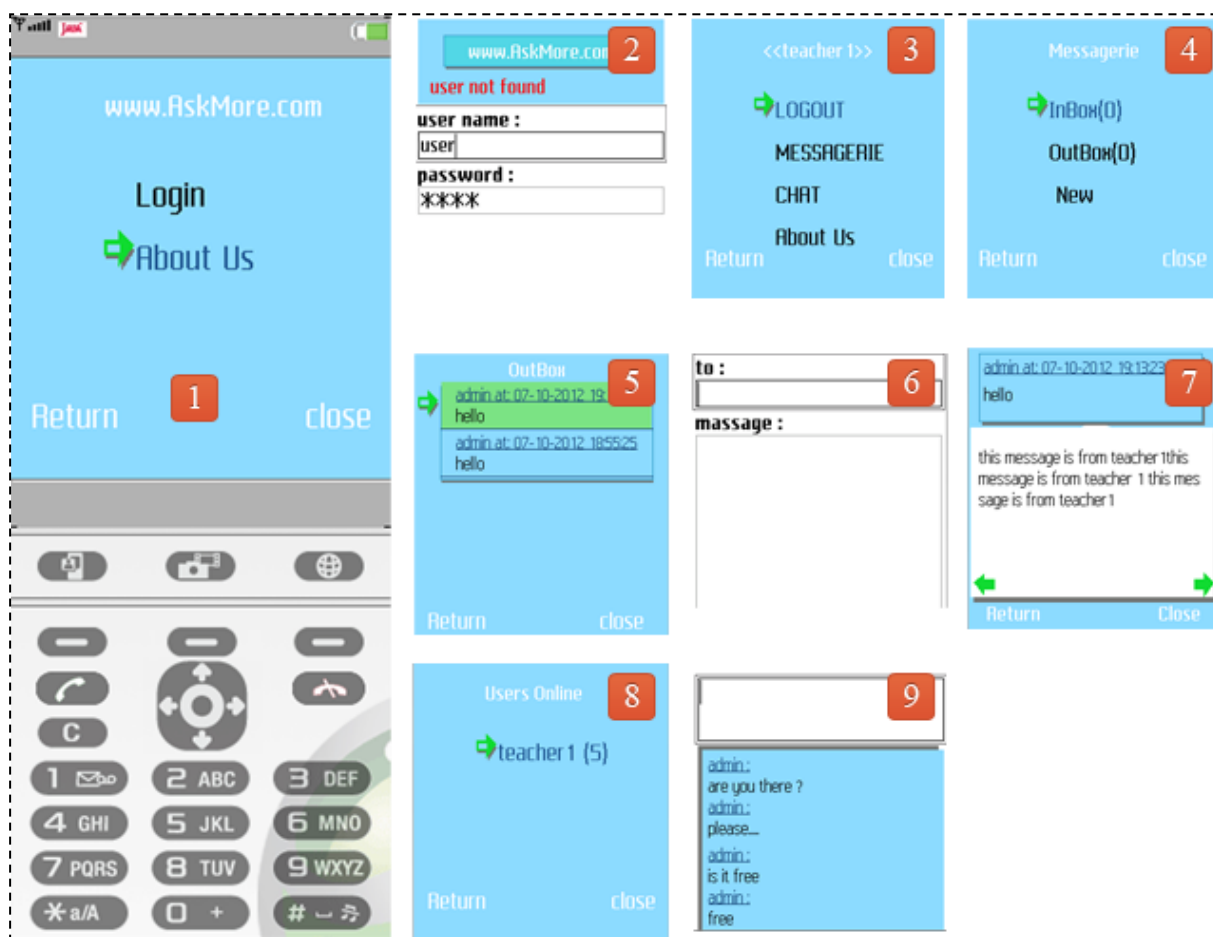


Figure 44 l'application mobile

- 1- L'interface principale de l'application.
- 2- Formulaire de l'authentification.
- 3- Menu principale.
- 4- Messagerie.

- 5- Page de l'outbox (les messages envoyés).
- 6- Formulaire d'envoi d'un nouveau message.
- 7- Lire (ouvrir) un message.
- 8- La liste des utilisateurs connectés dans le service de chat.
- 9- Formulaire de discussion.

9. QUELQUE PROBLEME DANS L'IMPLEMENTATION

9.1. Lecture d'un document PDF sur la page web

La majorité des documents qu'on peut trouver dans le web ayant le format PDF, c'est pour ça on a vu que l'offre de la possibilité d'ajouter des documents PDF est une nécessité. Mais le problème qui se pose ici, c'est comment garantir que tous les navigateurs des clients peuvent consulter ces documents sans être besoin d'installer des plugins ou d'autre programme ?

✓ La solution

L'idée est simple, on a développé une lecture PDF propre à notre site en utilisant le langage JAVA son rôle est de convertir les documents PDF a une série d'images est les envoyés aux clients (navigateur), dans ce cas, quel que soit le navigateur utilisé par le client, il peut afficher le contenu de document tant que il support les codes JavaScript.

9.2. Envoyer des champs textes et des fichiers dans le même formulaire

Le problème qu'on a rencontré ici c'est que on ne peut pas envoyer des fichiers et des champs texte dans le même formulaire, car le 'enctype' utilisé pour l'envoi des fichiers n'est pas la même que sel utilisé pour l'envoi des champs textes.

✓ Solution

La solution est d'utiliser le code Ajax pour l'envoi des fichiers.

10.CONCLUSION

La phase de l'implémentation est l'étape la plus importante dans le cycle de vie d'un site web. Ce chapitre a été consacré à la présentation des techniques de traitement et des outils utilisé pour la réalisation de notre application, dans la fin de ce chapitre on a montré les différentes parties de ce travail par quelques captures d'écran.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

1. CONTEXTE

Le projet qu'on a présenté dans ce mémoire est dans le but d'améliorer les niveaux de l'éducation dans les universités algériennes. c'est pour ça on ait sollicité de développer une application web , qui prend en charge la facilité de communication et le rapprochement entre les étudiants et leur enseignants , et pour atteindre ce but l'application doit offre aux étudiants et enseignants des services divers (publication des cours, chat ,réclamations ...) , les services de l'application sont facile à accéder , le mélange de l'éducation classique et moderne donne le lieu a un nouveau genre d'éducation inventif et plus efficace.

2. DESCRIPTION

Pour la réalisation de ce projet on a choisi comme approche de conception l'approche UP pour la modélisation. Pour l'implémentation on a basé sur le langage de programmation java (J2EE), qui devenu le langage le plus utilisé, grâce aux services de sécurité et l'indépendance de plateforme, plus de la facilité et la simplicité de codage etc..., on a utilisé aussi des différents outils de développements comme (StartUML pour la conception, Eclipse pour les code java...etc.).

Ce projet à bénéficier nous a plusieurs titres, il nous permis:

- D'appliquer les théorique qu'on a vue dans notre cycle d'étude.
- de travail en équipe sur une durée limitée
- d'enrichir nos connaissances dans des domaines très variés comme : *L'Orienté Objet, UML, UP, le langage JAVA*, le modèle MVC ...etc.

Après le passage par les différentes étapes de développement, l'application a abouti à un logiciel qui répond globalement besoins définis au chapitre 2.

3. PERSPECTIVES

Cette application peut être amélioré dans le futur par des nouvelles fonctionnalités on peut citer quelques fonctionnalités qu'elles peuvent être ajouté à l'application :

- l'application offre aux étudiants et aux enseignants la possibilité d'ajouter des nouveaux sujets de discussions.
- On peut améliorer l'application à une plateforme Multi-langue (Arabe, Français, Anglais) pour respecter la langue du système scolaire.

- Développer une application de Live Streaming.
- développer une application Android pour le système Android.

On souhaite que notre projet soit utile pour notre département, compréhensible et surtout serra un modèle de référence à ceux qui opteront pour la réalisation de sites web.

TABLE DES FIGURES

Figure1 : Tableau des différentes versions d'UML	10
Figure 2 : Présentation d'un modèle	11
Figure 3 : tableau de différent diagramme d'UML	12
Figure 4 : la vue ("4+1").....	17
Figure 5 : Présentation de cycle de vie du processus UP	18
Figure 6 : Exemple simplifié de diagramme de cas d'utilisation.	27
Figure7 : Les différents événements correspondant à un message asynchrone.	51
Figure 8 : Exemple de diagramme d'activité.	66
Figure 9 Schéma de la base de données	115
Figure10 Code source d'une classe de controle.....	118
Figure11 Les filtrés	119
Figure 12 exemple de filtré de cache et de compression.....	120
Figure 13 Code source Page JSP (Classe dialogue)	122
Figure 14 Code source JavaScript	122
Figure 15 Résultat de Vérification d'un formulaire	123
Figure 16 Code source AJAX	123
Figure 17 Exemple de l'utilisation de l'AJAX.....	124
Figure 19 Exemple d'un code CSS	125
Figure 18 Exemple des codes sources JQuery	125
Figure 20 La liste des outils.....	127
Figure 21 Diagramme de déploiement	128
Figure 22 Exemple de teste.....	128
Figure 23 l'architecture Client/serveur à 3 niveaux	129
Figure 24 cacher les liens de téléchargement.....	132
Figure 25 Les adresses relatives	132
Figure26 Page d'accueil	133
Figure 27 La liste des formations	134
Figure28 Page de Module	135
Figure29 Ajouter un cours.....	136
Figure30 Page de cours	137
Figure31 Ajouter un commentaire	138
Figure32 La liste des commentaires	138
Figure33 Page d'interrogation.....	139
Figure34 Noter une réponse	140
Figure35 Authentification	141
Figure 36 Page d'inscription étape 1	142
Figure 37 Page d'inscription étape 2 et 3.....	143
Figure38 Le profile de l'utilisateur	144
Figure 39 La messagerie.....	145
Figure40 l'espace de discussion (CHAT).....	146
Figure41 Bibliothèque	147
Figure42 L'administration des utilisateurs	148
Figure 43 Page des statistiques.....	149
Figure 44 l'application mobile.....	150

TABLE DES REFERENCES

- UML 2.0 - Laurent AUDIBERT 2007.
- Apprendre SQL avec MySQL - Christian Soutou - Eyrolles 2006.
- Rudiments SQL pour Oracle - Cyril Gruau – 2005.
- <http://www.journaldunet.com/>
- <http://www.yoja-web.com/fr/javascript/jquery-presentation>
- <http://www.gdawj.com/projetbetaboutiquejavaee/>
- <http://www.siteduzero.com/informatique/>
- <http://sylvain.sab.free.fr/cestquoi/web/css.php>
- <http://www.html5-css3.fr/css3/introduction-css3>
- https://www.java.com/fr/download/whatis_java.jsp