Génie Logiciel 2 : Analyse (partie 2)

Christian Ingouff
Pierre-Alexandre Tyndal
Sonia Seddiki
Yann Charbonnier

EISTI

2013/2014
Semestre 2
Jalon 3

Table des matières

Partie 1

Présentation du jalon

Après avoir reformulé avec précision les contraintes et attentes définies par l'utilisateur, nous avons procédé à l'analyse des besoins. Dans une première partie, nous avons fait l'introduction de celle-ci avec le diagramme de cas d'utilisation et le diagramme de classes, posant ainsi une base structurelle à l'étude de ce projet. Ce livrable finalise l'analyse avec trois autres types de diagrammes : les diagrammes d'activité, les diagrammes de séquence, et les diagrammes d'états-transitions.

Le rapport suivant délivrera l'analyse dans son entièreté, c'est-à-dire qu'il inclura les cinq types de diagrammes présentés dans l'analyse. Ce choix a été déterminé d'une part pour la complétude et la cohérence du travail ainsi présenté, et d'autre part pour accueillir une mise à jour au niveau des deux premiers types de diagrammes.

Le modèle d'analyse choisi pour ce projet est l'analyse orientée objet, réalisée grâce au langage de modélisation UML (Unified Modeling Language).

Dans ce compte-rendu, nous détaillerons :

- Le diagramme de cas d'utilisation
- Le diagramme de classes
- Les diagrammes d'activité
- Les diagrammes de séquence
- Les diagrammes d'états-transitions

Ce livrable dipose de cette répartition des taches :

- Diagrammes d'activité : Christian, Sonia
- Diagrammes de séquence : Yann
- Diagrammes d'états-transitions : Pierre-Alexandre
- Rédaction du rapport : Sonia, Christian

Partie 2

Diagramme de cas d'utilisation

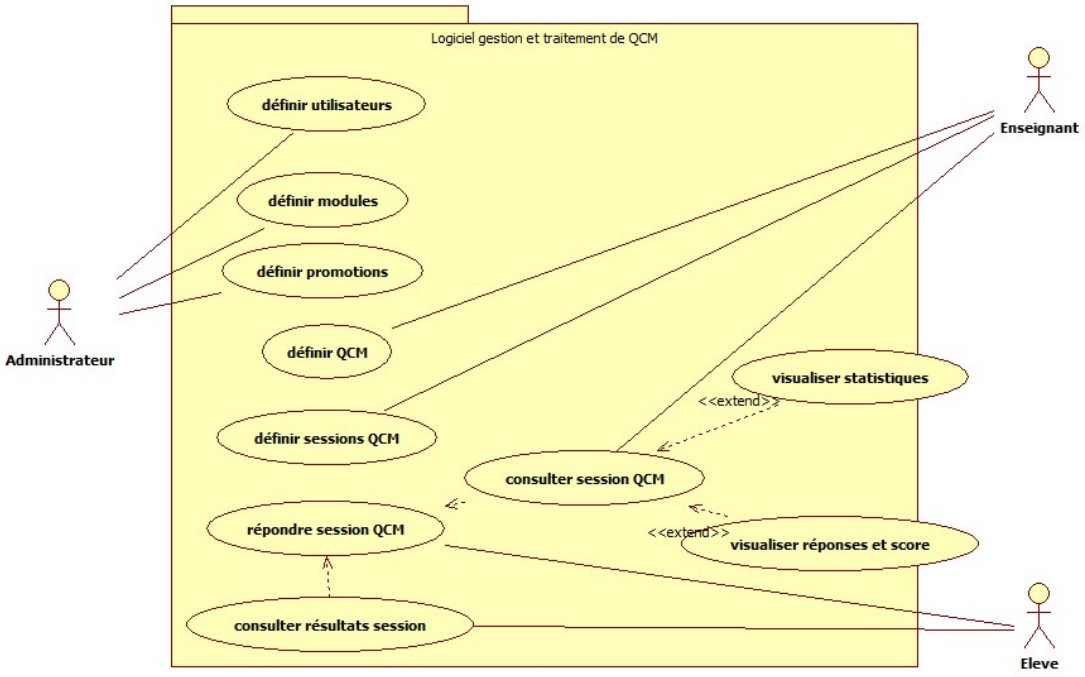

FIGURE 2.1 – Diagramme de cas d'utilisation

Ce diagramme décrit les différents cas d'utilisation de notre logiciel. Il le fait interagir avec des utilisateurs selon une liste d'opérations éventuellement liées entre elles. Ainsi, le diagramme présenté suit le raisonnement décrit ci-dessous.

Nous observons 3 types d'utilisateurs :
— L'administrateur
— L'enseignant
— L'élève

Un administrateur peut :
— Définir des utilisateurs (administrateurs, enseignants ou élèves)
— Définir des promotions (listes d'élèves)
— Définir des modules (que gèrent les enseignants)

Un professeur peut :
— Définir des QCM
— Définir des sessions de QCM, auparavant créés
— Consulter le statut d'une session de QCM
  — Visualiser les réponses et le score d'un élève à cette session
  — Visualiser les statistiques globales de la session

Un élève peut :
— Répondre à des sessions de QCM, auparavant créées
— Consulter les résultats de sa session de QCM

Partie 3

Diagramme de classes

3.1 Diagramme

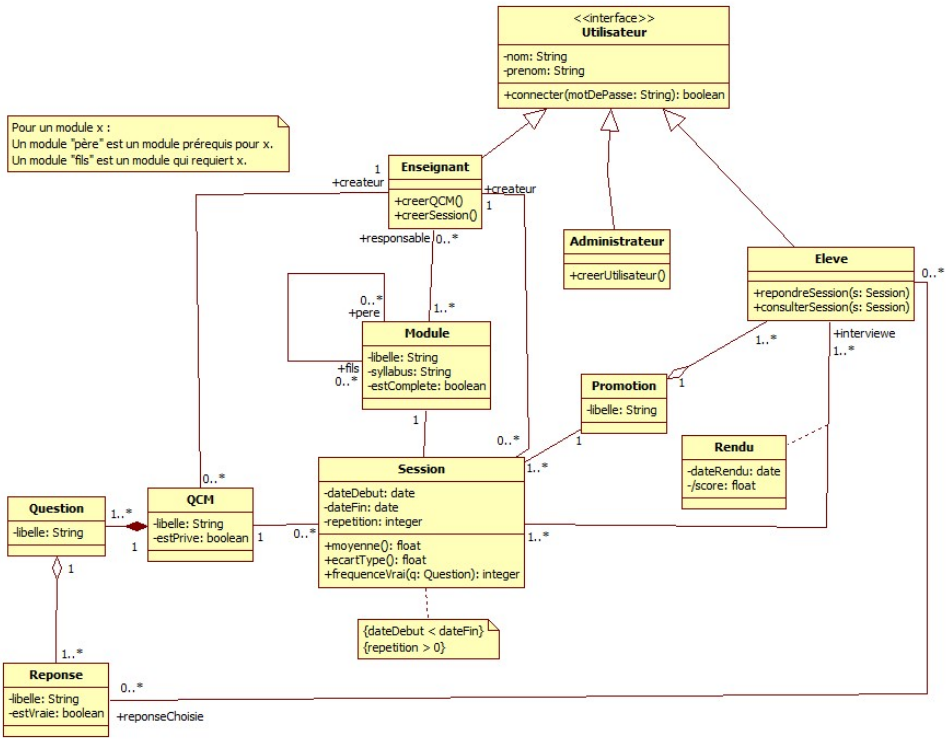

FIGURE 3.1 – Diagramme de classes

Ce diagramme décompose les besoins à l'aide de définitions de classes, ainsi que d'interactions entre ces classes. Le raisonnement ci-dessous explique le lien entre les classes et les interactions présentées et les besoins de notre projet.

Nous avons les classes suivantes :

- **Utilisateur** : défini par un nom et un prénom. Créé par un *Administrateur*.
- **Administrateur, Enseignant, Eleve** : les 3 types d'utilisateurs (généralisations de *Utilisateur*)
- **Module** : les enseignants doivent gérer un ou des modules (1..*). Un module peut être géré par un ou plusieurs enseignants (0..*). Défini par un libellé, un syllabus et un prédicat de complétion. Créé par un *Administrateur*.
- **Promotion** : liste d'élèves (agrégation de *Eleve*). Créé par un *Administrateur*.
- **QCM** : défini par un libellé et un prédicat de confidentialité. Il est créé par un *Enseignant*. Il est composé de :
  - **Question** : défini par un libellé. Contient un ensemble (agrégation) de :
    - **Réponse** : défini par un libellé et un prédicat de véracité. Peut être lié à un *Eleve* lorsqu'il est choisi lors d'une session de QCM.
- **Session** : définie par une date de début et de fin et par un nombre de répétition. Elle se passe dans le cadre d'un *Module* et est limitée à une *Promotion*. Elle est créée par un *Enseignant*.
- **Rendu** : un *Eleve* répond à une *Session* à l'aide d'un rendu. Il est défini par sa date. Le score sera obtenu en fonction des réponses données (lien entre *Eleve* et *Réponse*).

Pour un *Module* x, un module "père" est un module prérequis (nécessitant qu'il soit "complété") pour x et un module "fils" est un module qui requiert x.

Les statistiques globales pour une session consultables par l'enseignant sont la moyenne, l'écart-type et la fréquence de bonnes réponses par question. Le contenu de ces fonctions, ainsi que le calcul du score d'un élève à un QCM sera développé dans la partie Conception de notre projet.

3.2 OCL

Certains attributs sont soumis à des conditions que nous insérons ici à l'aide d'OCL.

```
context Session::répétition: integer
init: 1
```

Le nombre indiquant la répétition dans une session de QCM vaut 1 par défaut.

```
context Session
inv: (self.qcm.estPrivé) implies
     (self.créateur = self.qcm.créateur)
```

Si un QCM est privé, ce QCM ne peut être utilisé que par son créateur. Nous traduisons ceci par l'égalité entre le créateur de la session et le créateur du QCM.

```
context Session
inv: (self.promotion = self.interviewé.promotion)
```

Une session de QCM ne peut être accessible qu'à une seule promotion. Il faut donc vérifier l'appartenance de chaque interviewé qui désire répondre à la promotion correspondante.

```
context Session
inv: self.interviewé->count(r : Rendu | r.session = self) <= self.répétition
```

On ne peut au maximum répondre qu'au nombre de fois spécifié par le nombre indiquant la répétition. Cette condition vérifie si le nombre de rendus est bien inférieur à ce nombre.

```
context Rendu
inv: (self.dateRendu > self.session.dateDébut)
     AND (self.dateRendu < self.session.dateFin)
```

Une session n'est accessible qu'après la date de début et avant la date de fin.

Partie 4

Diagrammes d'activité

Les différents diagrammes d'activités présentés ci-dessous montrent les interactions entre le logiciel et l'utilisateur.

4.1 Définir utilisateurs

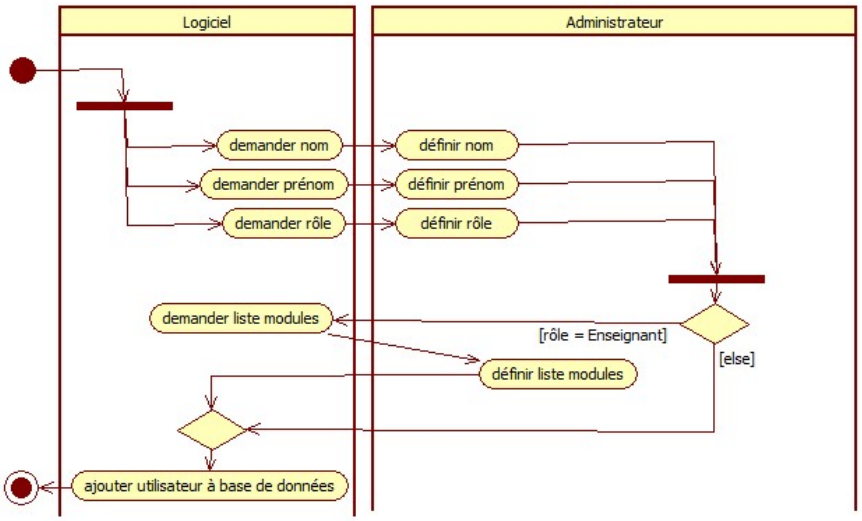

FIGURE 4.1 – Diagramme d'activités : définir utilisateurs

Ici, l'administrateur crée les différents utilisateurs. Si l'utilisateur créé est un enseignant, il renseigne également les modules auxquels il est rattaché. Puis il est enregistré dans la base de données.

4.2 Définir QCM

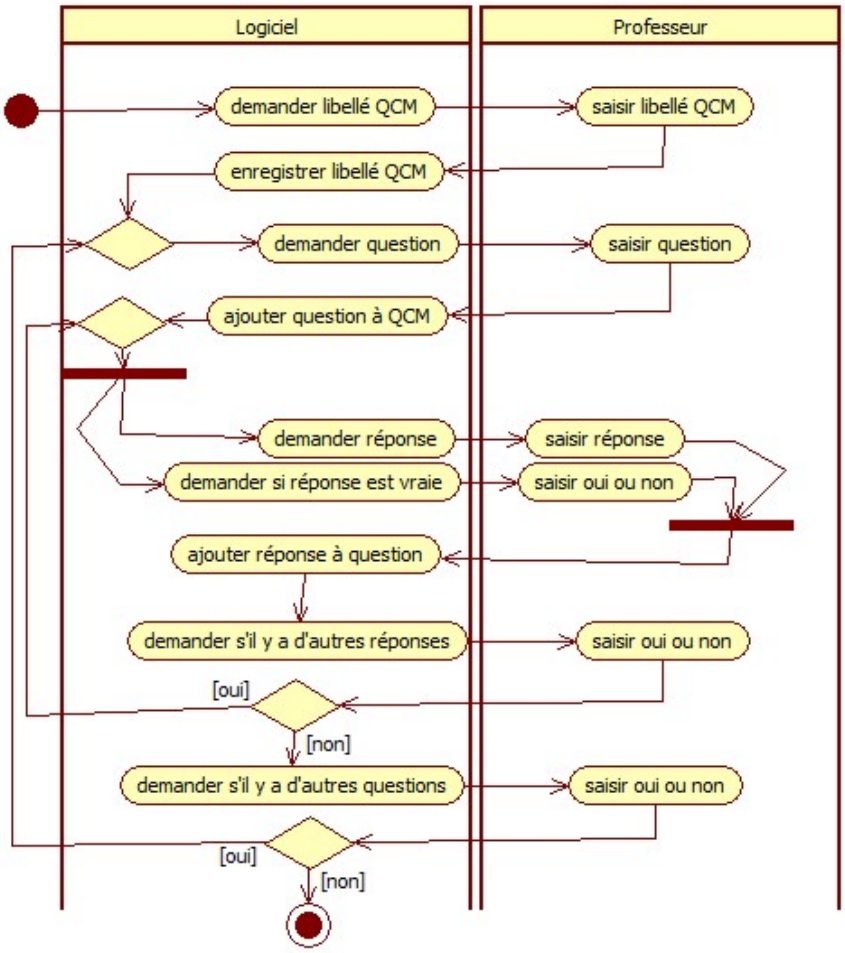

FIGURE 4.2 – Diagramme d'activités : définir QCM

Le professeur définit le QCM par son libellé. Puis il saisit la première question. Viennent ensuite les différentes propositions de réponse. Pour chaque réponse entrée, on précise si elle est vraie ou fausse. Quand il n'y a plus de réponses à saisir, on passe à la saisie de la question suivante, et ainsi de suite jusqu'à ce que toutes les questions soient rentrées.

4.3 Définir sessions QCM

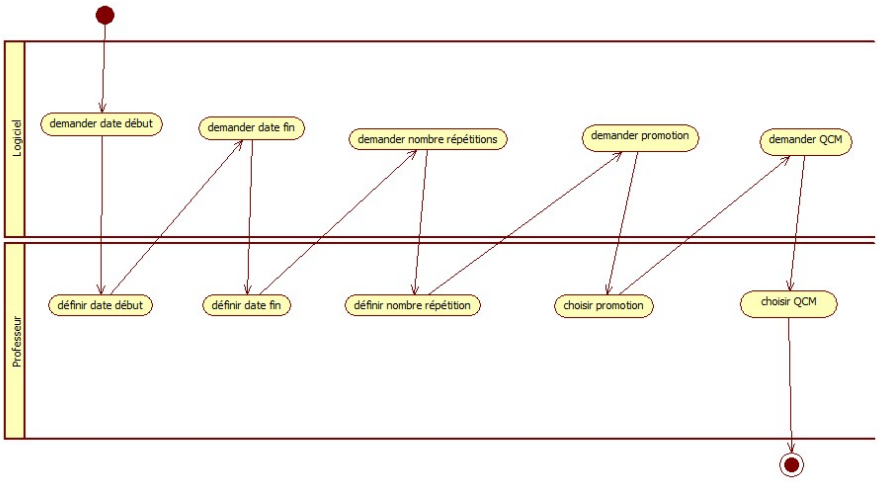

FIGURE 4.3 – Diagramme d'activités : définir sessions QCM

L'enseignant crée la session de QCM avec date de début, de fin, le nombre de répétitions, la promotion ainsi que le QCM proposé.

4.4 Consulter session QCM

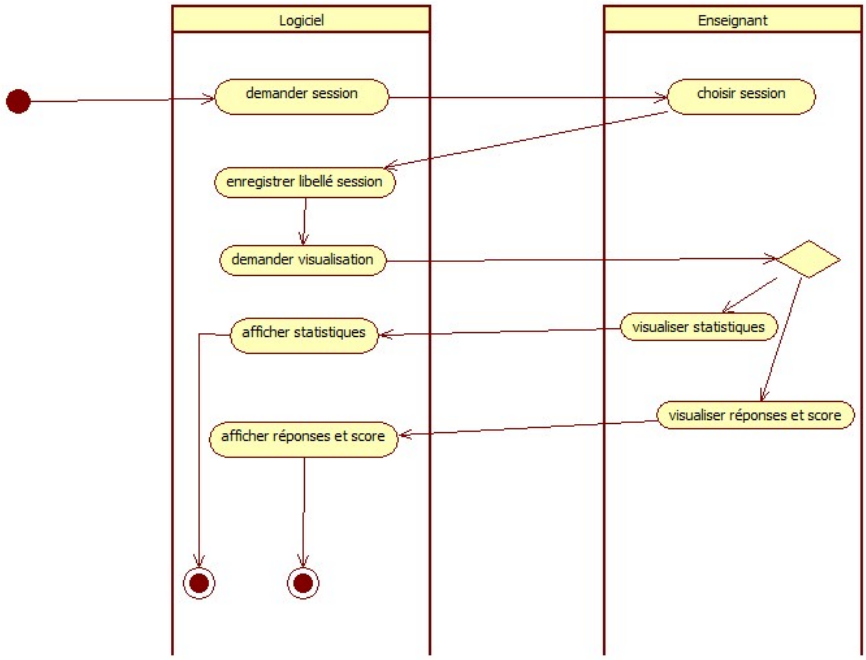

FIGURE 4.4 – Diagramme d'activités : consulter session QCM

L'enseignant consulte la session QCM et peut accéder aux scores et/ou aux statistiques.

4.5 Répondre session QCM

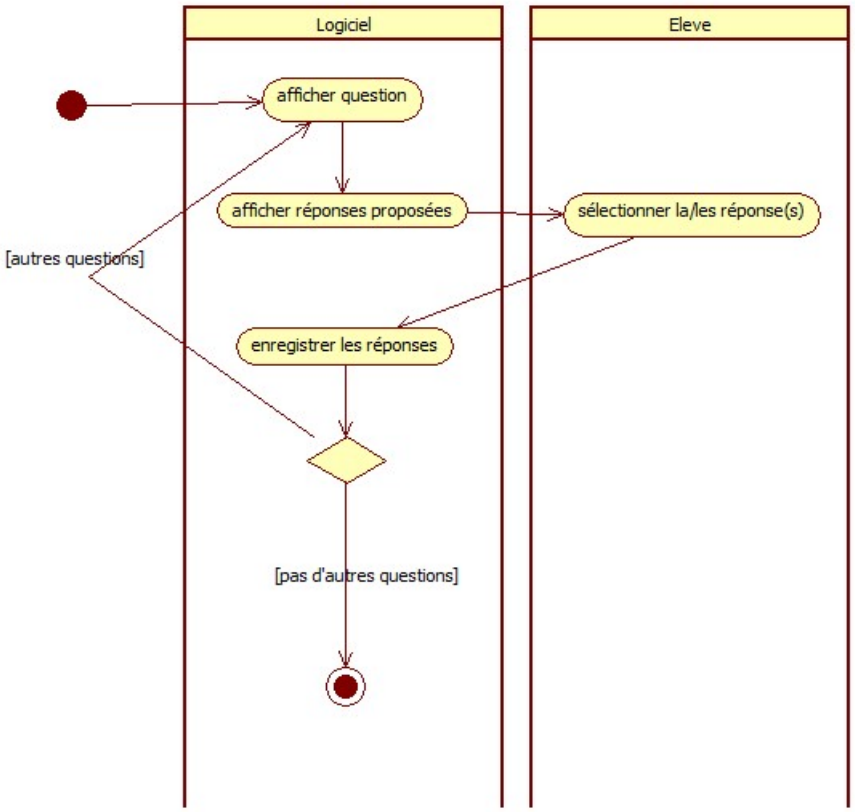

FIGURE 4.5 – Diagramme d'activités : répondre session QCM

Le logiciel affiche la question et les réponses proposées. L'étudiant sélectionne la ou les réponses qu'il souhaite cocher, celles-ci sont ensuite enregistrées par le logiciel (après validation préalable de l'étudiant par exemple). L'opération se répète jusqu'à la dernière question du QCM.

4.6 Consulter résultats session

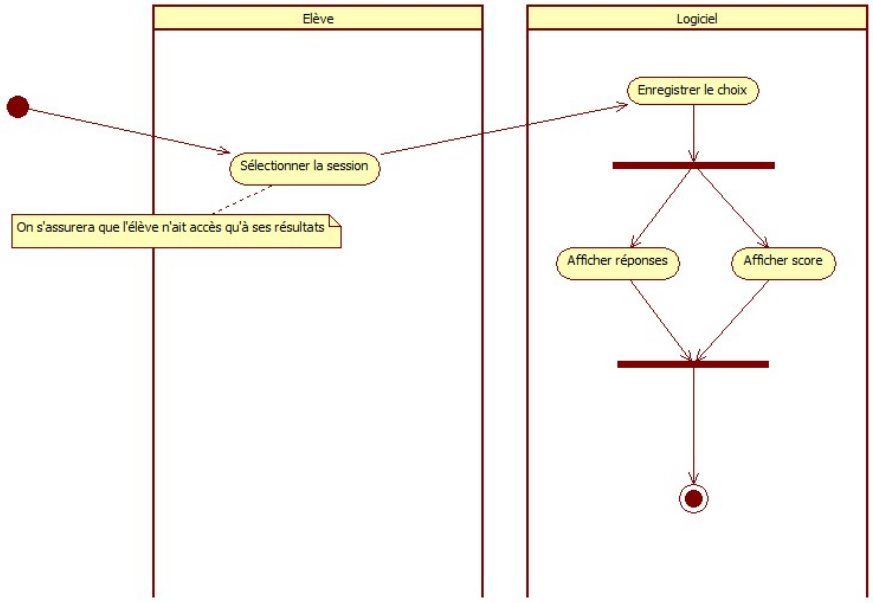

FIGURE 4.6 – Diagramme d'activités : consulter résultats session

L'élève sélectionne la session qu'il veut consulter (on s'assurera d'ailleurs qu'il ne puisse accéder qu'aux sessions auxquelles il a participé). Puis le logiciel affiche les différentes réponses sélectionnées par l'élève (avec pour chaque question, la réponse de l'élève et la bonne réponse dans le cas où il a mal répondu) ainsi que le score total.

4.7 Définir modules

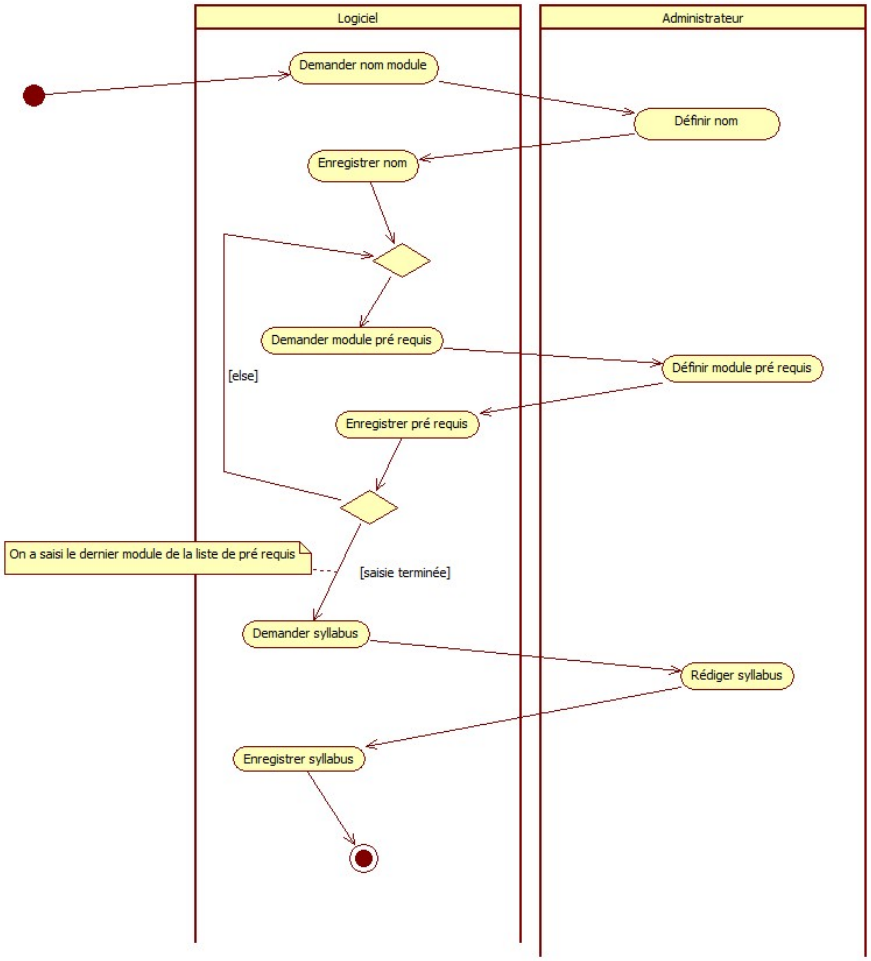

FIGURE 4.7 – Diagramme d'activités : définir modules

L'administrateur définit le nom du module et, le cas échéant, les différents modules prérequis. Une fois cette saisie terminée, il saisit le syllabus.

4.8 Définir promotions

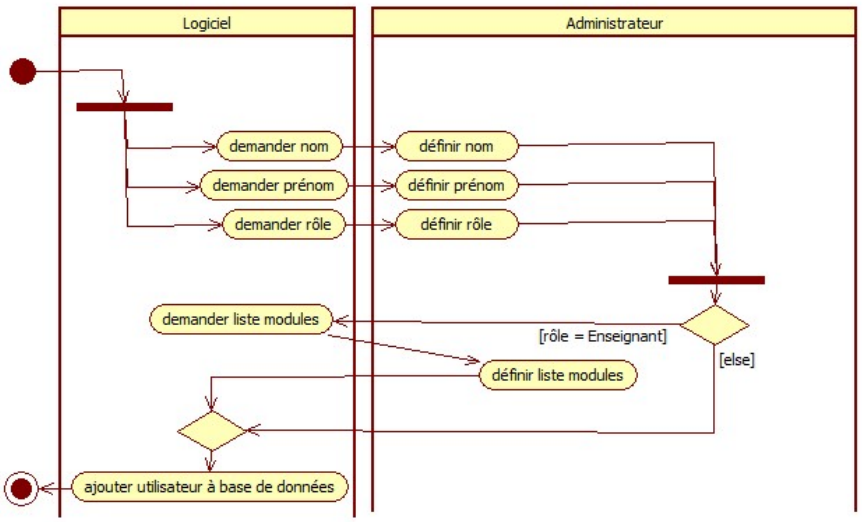

FIGURE 4.8 – Diagramme d'activités : définir promotions

L'administrateur définit la promotion en renseignant le cycle et l'année scolaire.

Partie 5

Diagrammes de séquence

Les diagrammes de séquence sont une version améliorée des diagrammes d'activité. Ils sont un intermédiaire entre ceux-ci et le code, puisqu'ils utilisent directement les classes et leurs méthodes.

5.1 Définir QCM

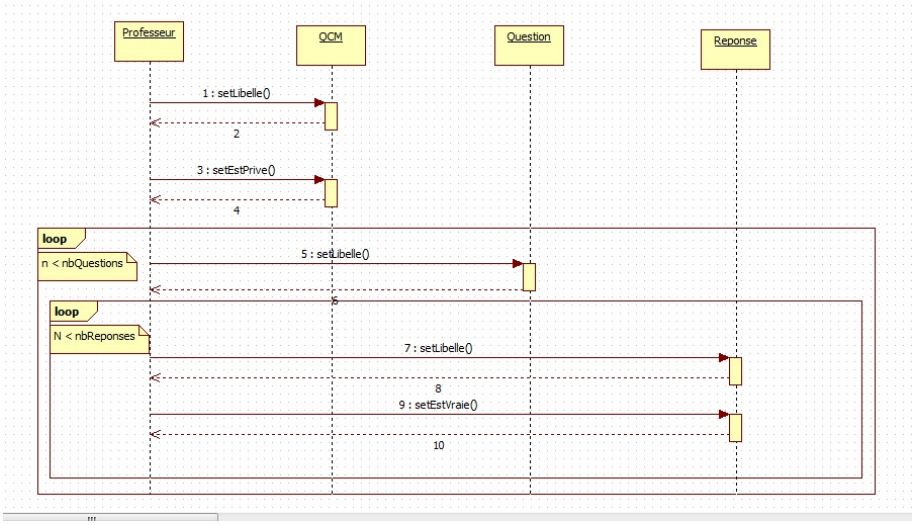

FIGURE 5.1 – Diagramme de séquences : définir QCM

On retrouve le même cheminement que dans le diagramme d'activité. Toutefois, la description se fait avec les méthodes des classes QCM, Question et Réponse (ici, les setters). L'encadré "loop" indique que l'on répète ce qui est dans le cadre tant que la condition décrite est vraie. Par exemple, pour la boucle avec la condition "n $<$ nbReponses", on répète ce qui est à l'intérieur (i.e. "setLibelle" et "setEstVrai") tant que le nombre de boucles n est inférieur au nombre total de réponses nbReponses de la question. Autrement dit, on fixe le libellé et la valeur de véracité de la réponse autant de fois qu'il y a de réponses à la question.

5.2 Répondre session QCM

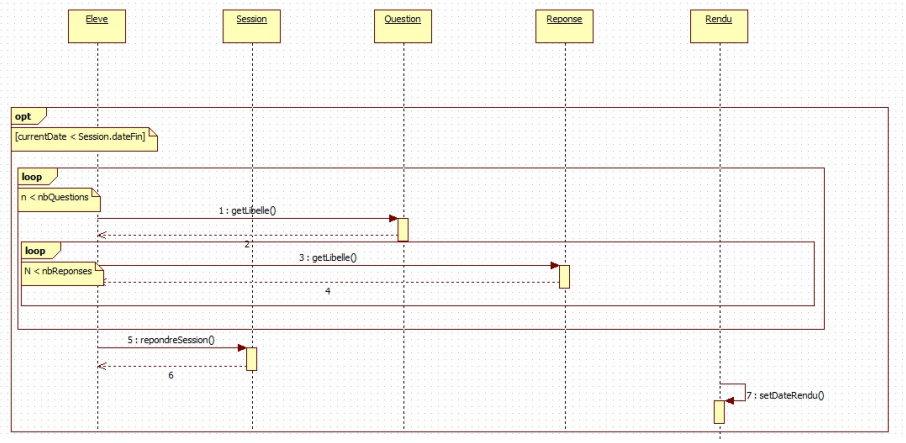

FIGURE 5.2 – Diagramme de séquences : répondre session QCM

Avant de laisser l'élève répondre à la session, deux choix sont possibles :
— soit il y répond pour la première fois
— soit il y souhaite revenir sur ses réponses après sauvegarde

Dans tous les cas, il faut s'assurer que l'étudiant ne peut répondre qu'avant la date de fin de la session. Le bloc "opt" est utile dans notre cas, puisqu'il symbolise que le contenu du cadre "opt" ne peut être effectué que si la condition associée est vérifiée (ici, que la date actuelle est inférieure à la date de fin).L'élève peut ainsi récupérer les questions et réponses associées, et ensuite répondre au questionnaire.

5.3 Définir modules

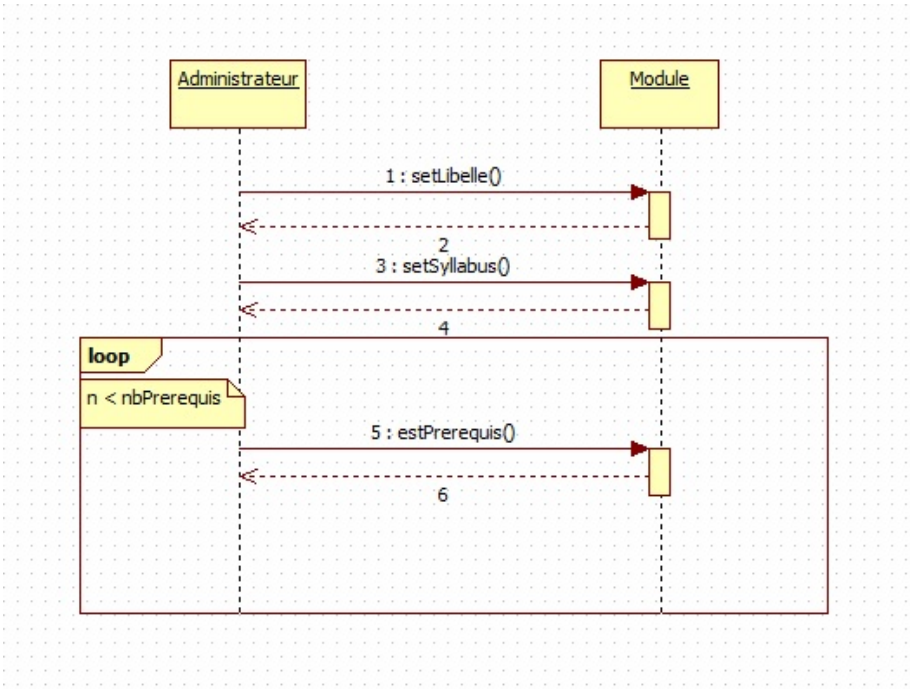

FIGURE 5.3 – Diagramme de séquences : définir modules

Après avoir défini le libellé et le syllabus du module, l'administrateur définit quels modules sont des prérequis pour le module considéré tant que la liste n'est pas totalement établie.

Partie 6

Diagrammes d'états-transitions

Suite à l'étude du cahier des charges et donc, par ce biais, l'étude des classes et objets de ce projet, nous avons conclu que seule la session de QCM a une vie suffisamant intéressante pour susciter un étude sous la forme de diagramme.

6.1 Session de QCM

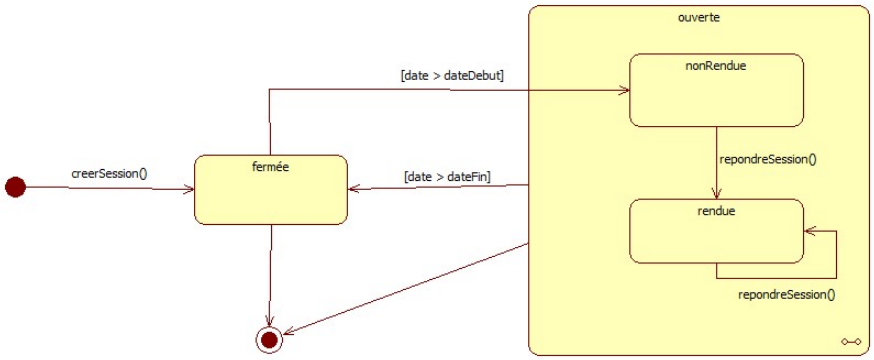

FIGURE 6.1 – Diagramme d'états-transitions : Session de QCM

Ce diagramme résume la vie de la session de QCM. Lors de sa création, la session de QCM est fermée, et selon sa date d'ouverture, la session sera ouverte puis refermée. Il est possible de rendre cette session uniquement pendant la periode où cette dernière est ouverte. Une fois la date de fermeture atteinte, la session est fermée qu'elle soit rendue ou non. Le professeur ayant créé la session de QCM est le seul pouvant supprimer ce dernier, et cela à n'importe quel moment.

Conclusion

Toute cette partie d'analyse nous permet désormais d'avoir une vision d'ensemble sur le projet. Cette vision transitionnera notre travail vers la phase de conception, à savoir le détaillage de fonctions complexes pour enfin concrétiser la théorie en l'implémentant dans le programme en Java.

Tout comme les deux premiers diagrammes nous ont permis de nous représenter la structure du programme, les derniers types de diagrammes cadrent le fonctionnement des entités définies auparavant entre elles. Elles font ainsi l'exposé de comment les classes définies dans le diagramme de classes vont interagir pour pouvoir se plier aux besoins de l'utilisateur.

Les diagrammes présentés sont flexibles, et sujets à d'autres modifications durant la phase d'implémentation. En effet, la phase d'analyse n'effectue pas obligatoirement l'étude des limitations d'un langage tel que le Java, et il peut également y avoir d'autres litiges tels que la sécurité du programme ou le souci d'intuitivité parmi d'autres.

En conclusion, cette analyse constitue la charpente du résultat final : elle sera définitivement le point d'appui du fonctionnement du programme résultant, mais ne sera pas exclusive dans sa composition. C'est sur cette analyse que s'enchaîne les jalons suivants, c'est-à-dire les phases de conception et de programmation.