

Examen du cours "GÉNIE LOGICIEL AVANCÉ"

Master 1 – Université Paris Diderot Paris 7

Durée: 3 heures

Tout document autorisé.

Le soin apporté à la rédaction et à la présentation ainsi que la rigueur des réponses seront pris en compte dans la notation. Certaines questions n'ont pas de réponses tranchées : il est donc nécessaire d'**argumenter** pour justifier vos réponses. Veillez à ce que ces argumentations restent synthétiques et scientifiques (des hypothèses de raisonnement explicitées, un langage précis, une idée par paragraphe, une structuration logique ...).

Ce sujet porte sur la conception d'un système de gestion de "Questionnaires à Choix Multiples" (QCM). Il se décompose en 4 parties quasi-indépendantes. On indique le temps qu'il est conseillé d'allouer à chacune d'elle. Notez que le barème sera proportionnel à ce temps.

1 Spécification des besoins (30 minutes)

Une prospection a été menée auprès de plusieurs personnes de l'Université Paris Diderot en vue d'analyser leur utilisation potentielle d'un système de gestion de QCM :

- Mme K : appariteur.
- Mme A : enseignante de cours magistraux.
- M. O : étudiant.
- M. P : enseignant de travaux dirigés.
- Mme M : administrateur système.
- M. T : responsable pédagogique.

Ces personnes ont exprimé leurs besoins en des termes informels. Nous les avons compilés dans la figure 1.

Exercice 1 (Analyse des besoins)

1. *Quelle est la frontière du système ?*
2. *Quels sont les besoins fonctionnels ? Non fonctionnels ? Liés au domaine d'activité ?*
3. *D'après vous, quels sont les acteurs principaux de ce système ? Les acteurs secondaires ? Justifiez votre réponse.*
4. *Énoncez 3 problèmes qui sont d'après vous les sources de risques principales de ce projet.*
5. *Proposez un ou plusieurs diagramme(s) de cas d'utilisation. On rappelle que ces diagrammes sont plus informatifs si ils sont structurés, c'est-à-dire construits à partir de relations d'inclusions, d'extensions et de généralisations entre les cas d'utilisation et de relations de généralisation entre les acteurs. On gardera aussi à l'idée qu'il est important d'être le plus complet possible au sujet des cas d'utilisation principaux et généraux, avant de rentrer éventuellement dans les détails.*

□

1. P : « Je veux que les corrections soient faites automatiquement. »
2. A : « Le système doit permettre la constitution collaborative de base de questions/réponses. »
3. A : « La notation doit être sûre : si le système a le moindre doute, il doit demander son avis au responsable du questionnaire. »
4. A : « Pour moi, un questionnaire, c'est un formulaire présentant une collection de questions dont les réponses sont à choisir parmi un ensemble fini. »
5. T : « Aucune note incertaine et non validée ne doit être divulguée à étudiant. »
6. A : « On doit pouvoir associer un questionnaire à un étudiant. »
7. P : « La triche doit être impossible. »
8. A : « Si un enseignant organise plusieurs séances de QCM, on ne veut pas qu'une question soit repesée à un même étudiant. »
9. K : « Le travail de distribution et de ramassage des copies doit être minimal. »
10. A : « Les règles de notation doivent être redéfinissables (même *a posteriori*, une fois que la correction a été faite, on doit pouvoir la refaire avec de nouveaux critères d'évaluation). »
11. A : « Je dois pouvoir écrire les énoncés des questions à l'aide du générateur de document de mon choix (\LaTeX , OPENOFFICE, MATHML, ...) et adjoindre des figures ou des images. »
12. T : « Les étudiants ne doivent pas pouvoir accéder à la base de questions-réponses. »
13. O : « Je veux recevoir mes notes de QCM par SMS. »
14. K : « Une copie de QCM, c'est un questionnaire rempli par un étudiant, il faut d'ailleurs veiller à ce que l'étudiant n'ait pas oublié de noter son numéro d'étudiant sur cette copie ! »
15. T : « Pour constituer un ensemble de questionnaires, on doit pouvoir spécifier des critères de sélection de questions (suivant la matière, le thème, des notions abordées, le niveau du public). »
16. A : « Les étudiants et moi-même devons pouvoir accéder à des statistiques de réussite aux questions. »
17. T : « Les questionnaires doivent être anonymisés. »
18. T : « On peut attendre quelques heures pour obtenir les résultats d'une interrogation, mais pas quelques jours. »
19. A : « En fonction d'un planning des interrogations et les thèmes à aborder, j'aimerais savoir si il y a suffisamment de questions dans la base ou si je dois en inventer de nouvelles. »
20. A : « Je ne veux pas que les redoublants s'ennuient. »
21. P : « On doit pouvoir associer une correction à une question, cette correction peut contenir des informations complémentaires et des pointeurs vers les sections du cours associées. »
22. A : « Un processus de validation croisée doit être possible : si je soumetts un couple question/réponse, je peux demander à ce qu'il ne soit intégré à la base qu'une fois validé par un collègue. »
23. T : « Les notes des QCM sont automatiquement intégrées dans la base de données de notes des étudiants. »
24. T : « Je veux pouvoir transmettre aux étudiants le planning des sessions de QCM et les thèmes sur lesquels ils portent. Avant chaque session, je veux leur envoyer un rappel. »
25. M : « Le gestionnaire de QCM doit s'intégrer facilement au système d'information de l'Université Paris Diderot. »
26. A : « Ce serait intéressant d'avoir des questionnaires portant sur plusieurs cours à la fois ... »
27. P : « Si jamais la machine à corriger les copies est en panne, je dois pouvoir corriger à la main. »
28. P : « Suite à une session de QCM, j'aimerais connaître les points incompris par les étudiants de mon groupe de travaux dirigés. »

FIG. 1 – Résultats de la prospection.

2 Spécification informelle du système (1 heure)

Exercice 2 (Architecture)

1. Hiérarchisez les services du système suivant les problèmes qu'ils adressent.
2. Proposez une architecture générale pour le système en exhibant les modules et leurs sous-modules ainsi que leur relation de dépendances. On rappelle qu'une bonne architecture maximise la cohérence des modules et minimise leur interdépendance. N'oubliez pas d'argumenter et d'essayer d'être le plus complet possible (encore une fois, non pas en termes de détails traités mais en termes de problèmes généraux pris en compte).
3. Dans quel ordre développeriez-vous les fonctionnalités de ce système ? Pourquoi ?

□

Exercice 3 (Modélisation Objet)

1. Quelles seraient les classes abstraites principales de ce système ? En langage naturel structuré, donnez pour chacune d'elle son nom, son rôle, le module dans lequel elle apparaît, ses invariants, ses méthodes principales ainsi que les pré-conditions et les post-conditions de ces dernières.
2. Proposez un ou plusieurs diagrammes statiques de classe représentant les relations entre les classes abstraites explicitées par la question précédente. Cette modélisation statique devra permettre les extensions suivantes :
 - (a) Fournir un service de simulation d'épreuves de QCM (on génère des questionnaires pour de faux étudiants, on tire des réponses au hasard, on génère les rapports, ...).
 - (b) Permettre un autre type de sessions de QCM dans lesquels ce sont des groupes d'étudiants qui répondent aux formulaires et non pas un étudiant isolé.
 - (c) Autoriser des sessions de QCM où l'étudiant répond via une page WEB.
 - (d) Notifier l'auteur d'un couple question/réponse de la base si ce couple a été modifié par une tierce personne.
 - (e) Changer le type des réponses à un QCM : plutôt qu'une réponse binaire (je coche, je ne coche pas), on autorise des réponses numériques utiles pour les questions de la forme : "Classez les éléments suivants par rapport à..."

L'utilisation de patrons de conception est non obligatoire mais conseillée.

□

3 Spécification formelle du système (45 minutes)

Exercice 4 (Modèle relationnel) Il s'agit maintenant de définir formellement une partie de la base de données pour les **questions** des QCM. Pour simplifier, on considère que pour chaque question on associe :

- un sous-ensemble non vide de mots clés ; l'ensemble de tous les mots clés est \mathbb{M} ;
- une liste de réponses possibles ; l'ensemble de toutes les réponses est \mathbb{R} ;
- pour chaque élément de la liste de réponse on lui associe un nombre entier dans l'intervalle $[-100, 100]$ représentant le pourcentage du barème qui est utilisé par la réponse pour calculer la note pour une question. Par exemple, soit q une question ayant trois réponses possible r_1 , r_2 et r_3 . Soient -20 , 50 respectivement 50 les pourcentages associés avec chaque réponse et n le nombre de points pour cette question. Si les réponses r_2 et r_3 sont cochées, le nombre de points obtenu est maximum, c'est-à-dire n . Si r_1 est cochée, le nombre de points est $-0,2 \times n$, donc négatif. Si r_1 et r_2 sont cochées, le nombre de points obtenu est $0,3 \times n$.

1. En utilisant les expressions d'ensemble du langage B , définir le type ensemble correspondant à la base de données de QCM définie ci-dessus. Justifier que toutes les contraintes de cette définition sont satisfaites par votre type.
2. L'ensemble de tous les mots clés \mathbb{M} est évidemment infini. Pour définir un ensemble fini de mots clé à utiliser dans la base de données de questions, on définit une table des mots clés à utiliser pour les questions. Cette table peut être modifiée par des opérations d'insertion, d'élimination et de changement d'un mot clé. Donner la définition de type, son invariant et des trois opérations sur cette table en langage B . Préciser à chaque fois la section de la spécification B dans laquelle se trouve chaque définition.
3. Ecrire en langage B la partie de l'invariant sur la base de données de questions qui relie l'ensemble fini de mots clés reconnus (la table des mots clés) et la base de données des questions.

□

4 Validation et vérification par test (45 minutes)

Exercice 5 (Test de validation) Soit la spécification suivante pour la fonction/méthode qui calcule la note d'une question dans une copie QCM :

```
/** Calcule la note pour la question quest en utilisant le barème total n et
 * la sélection des réponses rep.
 * @param quest    séquence de réponses possibles avec leur pourcentage sur le barème
 * @param n        entier représentant le barème de la question quest
 * @param rep      séquence de booléens représentant le choix des réponses dans la copie
 * @return         un entier si le calcul peut être fait, NaN sinon
 * Pre-condition:  n > 0 et size(quest)=size(rep)
 * Post-condition: le calcul peut être fait et on renvoie la note, sinon on renvoie NaN
 */
int calculNote(ReponseSeq quest, int n, BoolSeq rep);
```

Écrire une suite de test pour la validation de cette méthode en partant de sa spécification. Justifier, pour chaque test de votre séquence, l'intérêt par rapport à la spécification donnée. □

Exercice 6 (Test de défaut) Considérer l'implémentation suivante de la fonction `calculNote` spécifiée ci-dessus. (On utilise un pseudo-Java.)

```
1: int calculNote(ReponseSeq quest, int n, BoolSeq rep) {
2:   if (n <= 0 || (quest.length() != rep.length()))
3:     return NaN;
4:   int note = 0;
5:   int i = 0;
6:   while (i < quest.length() && note != NaN) {
7:     if (rep.getAt(i))
8:       note += quest.getAt(i).getPourcentage() * n;
9:     i++;
10:  }
11:  return note;
}
```

Écrire une suite de test pour la vérification de cette méthode en utilisant les critères de couverture *S* (toutes les instructions), *D* (tous les arcs), *C* (toutes les conditions atomiques) et *DC* (toutes les combinaisons arc et condition atomique). □