IFT3912 - Développement et maintenance de logiciels

Examen intra - Hiver 2011

Solutions

Bruno Dufour

21 février 2011
Durée : 1 h 50

Instructions

- Toute forme de documentation ou communication est interdite.
- Inscrivez votre nom sur la page couverture du cahier de réponse.
- Répondez à toutes les questions dans le cahier de réponse. Indiquez clairement le numéro de la question dans le cahier (par exemple : 1-b) pour chaque réponse. L'ordre des réponses n'est pas important.
- Vous pouvez utiliser toutes les pages du cahier (recto et verso). Vos réponses ne doivent pas nécessairement débuter en haut d'une nouvelle page.
- D'autres cahiers sont disponibles si nécessaire.
- Prenez connaissance de toutes les questions en débutant, ne passez pas tout votre temps sur une seule question.
- Vérifiez l'orthographe et la grammaire.
- L'examen compte **7 pages** incluant celle-ci.
- Le nombre de points alloué à chaque question est indiqué dans l'énoncé, pour un total de 100 points.

Bonne chance !

PARTIE A - Gestion de projet

Question 1 (10 pts) - Considérez les tâches suivantes pour un projet de développement de moteur de recherche :

- A - Création d'un robot (*crawler*) pour la collecte de données (7 jours)
- B - Création de la base de données (4 jours)
- C - Collecte des données à indexer (10 jours)
- D - Entrée et validation des données à indexer (3 jours)
- E - Implémentation de l'algorithme de classement des résultats (9 jours)
- F - Implémentation de l'interface de requêtes de recherche (2 jours)
- G - Implémentation de l'interface d'affichage des résultats de recherche (4 jours)
- H - Évaluation de la qualité des résultats (5 jours)
- J - Tests de système (4 jours)
- K - Déploiement / installation (5 jours)

Proposez une décomposition hiérarchique **fonctionnelle** du projet de type *Work Breakdown Structure (WBS)* à partir de ces tâches. Justifiez vos choix.

Plusieurs solutions possibles, par exemple :

Moteur de recherche

1. Gestion des données
    1.1. Collecte de données
        1.1.1. Création d'un robot (crawler) pour la collecte de données (A)
        1.1.2. Collecte des données à indexer (C)
    1.2. Entrée des données
        1.2.1. Création de la base de données (B)
        1.2.2. Entrée et validation des données à indexer (D)
2. Recherche
    2.1. Implémentation de l'algorithme de classement des résultats (E)
    2.2. Interface
        2.2.1. Implémentation de l'interface de requêtes de recherche (F)
        2.2.2. Implémentation de l'interface d'affichage des résultats de recherche (G)
    2.3. Évaluation de la qualité des résultats (H)
3. Tests de système (J)
4. Déploiement / installation (K)

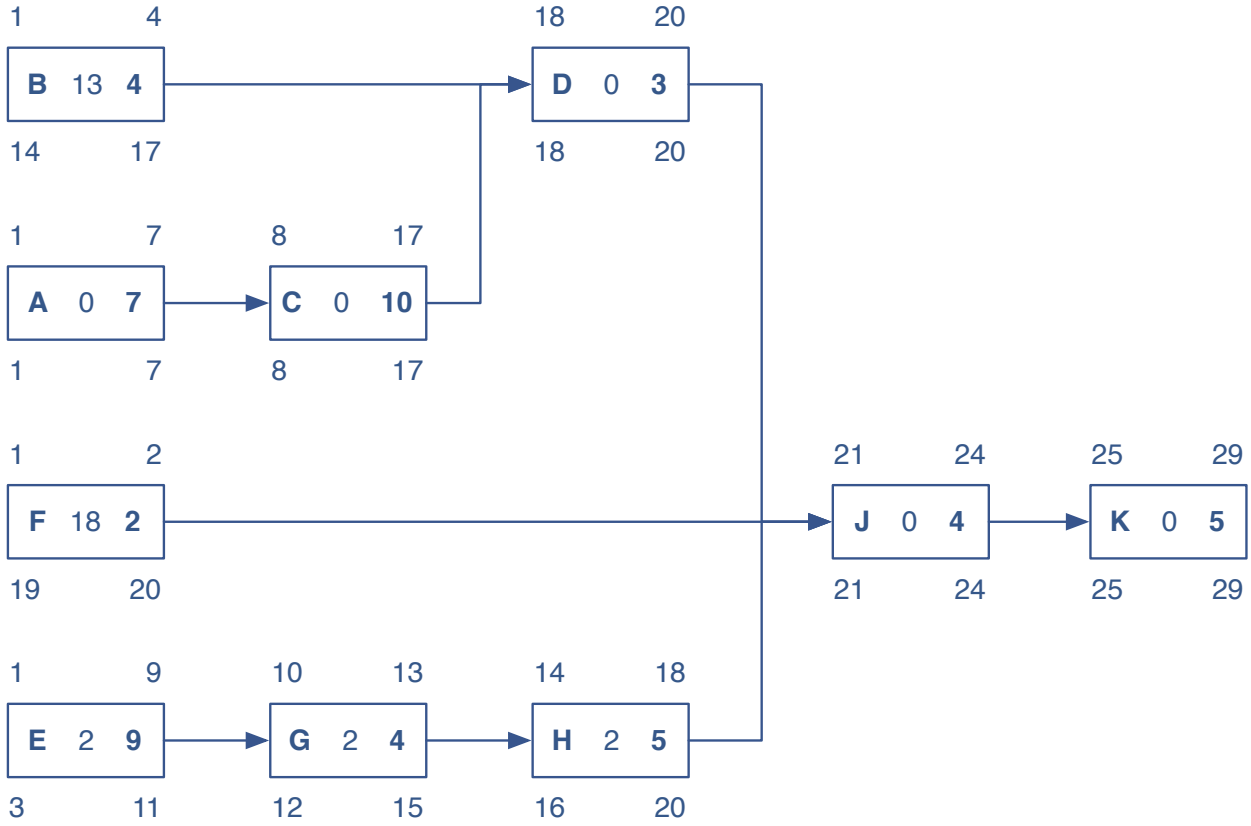
Justification :

Le système entier comprend deux principales activités (qui correspondent naturellement à deux sous-systèmes) : la gestion de données à indexer, et la recherche des données à partir de l'index. Les tâches sont donc organisés de façon hiérarchiques selon leur appartenance à ces deux activités de premier niveau. Les tâches qui traitent du système en entier sont aussi placées au premier niveau. D'autres niveaux sont créés pour refléter une décomposition modulaire possible de chaque sous-système.

Question 2 (38 pts)

a) Donnez le diagramme de réseau (activités sur les nœuds) qui correspond au WBS de la question 1. Utilisez les dépendances suivantes entre les tâches :

A → C, B → D, C → D, D → J, E → G, F → J, G → H, H → J, J → K

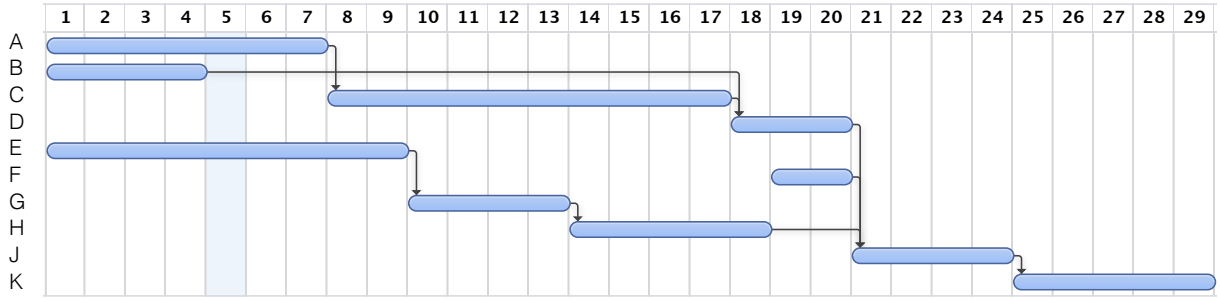


b) À partir du diagramme de réseau en a), calculez les valeurs suivantes pour chaque nœud :

- Démarrage au plus tôt (*early start - ES*)
- Fin au plus tôt (*early finish - EF*)
- Démarrage au plus tard (*late start* - LS)
- Fin au plus tard (*late finish - LF*)
- Jeu (*slack, float)*

(Voir diagramme en 2-a)

c) Donnez le diagramme de Gantt qui correspond au diagramme de réseau en a).



d) Combien de temps total est nécessaire à la réalisation du projet ? Justifiez votre réponse.

Le chemin critique A → C → D → J → K impose une durée minimale de 29 jours de travail pour le projet.

e) Combien de membres l'équipe doit-elle compter pour que le projet soit réalisable selon l'échéancier le plus court ? Justifiez votre réponse.

3 personnes sont nécessaires, puisque le jeu de 18 jours pour la tâche F implique qu'elle peut débuter au 19ème jour sans que la durée du projet ne soit affectée.

PARTIE B - Méthodologies agiles

Question 3 (12 pts)

a) Quel est l'objectif principal des méthodologies de développement agiles ?

L'objectif principal commun à toutes les méthodologies agiles est de faire face au changement d'une façon plus rapide et flexible qu'avec les méthodologies de développement traditionnelles.

b) Décrivez 3 particularités de la programmation extrême qui supportent cet objectif. Justifiez vos réponses.

Plusieurs solutions possibles, par exemple :

- Développement piloté par les tests : permet d'effectuer des changements majeurs au système avec assurance.
- "User stories" : contiennent peu de détails, et peuvent être manipulées facilement lorsqu'un changement survient.
- Le client fait partie de l'équipe : permet d'obtenir la rétroaction du client rapidement, et donc de détecter les changements requis le plus tôt possible dans le but de s'adapter rapidement.

c) Identifiez 2 différences importantes entre les phases de développement des méthodologies agiles et celles du modèle de développement en cascade traditionnel.

- Les phases de développement agiles contiennent toutes les activités de développement.
- Les phases de développement agiles sont composées d'itérations de courte durée.

PARTIE C - Tests

Question 4 (15 pts) - Considérez le code suivant :

```
public static void tri(int a[]) {
    for (int i = 1; i < a.length; i++){
        int j = i;
        int v = a[i];
        while ((j > 0) && (a[j-1] > v)) {
            a[j] = a[j-1];
            j--;
        }
        a[j] = v;
    }
}
```

Pour la fonction "`tri`", donnez un ensemble **minimal** de tests qui accomplissent la couverture complète :

a) des instructions

Exemple de solution : [2,1] → [1,2]

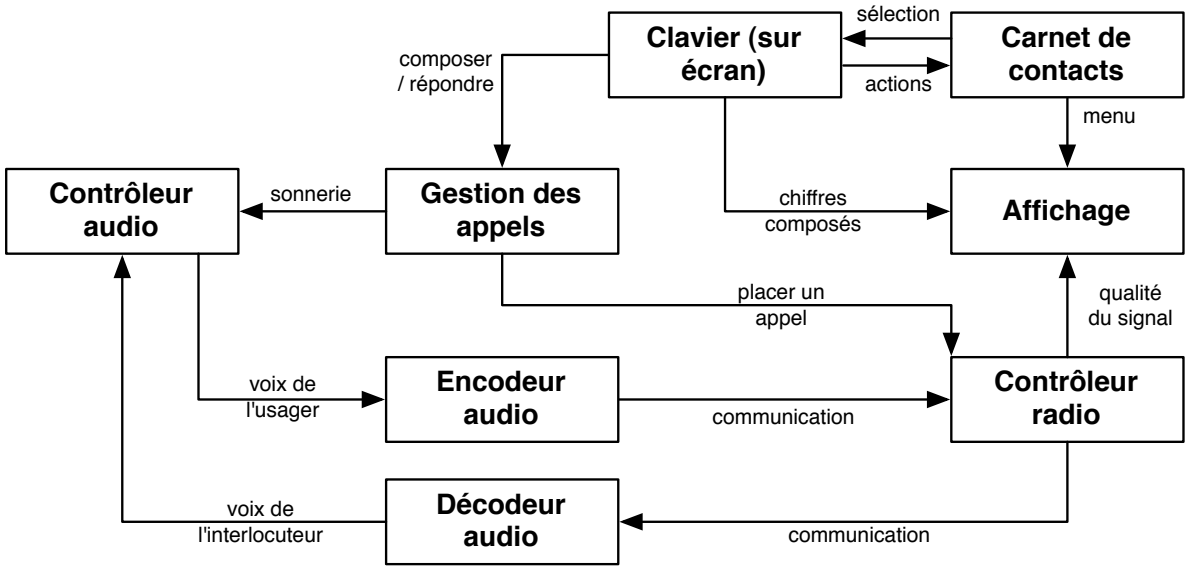
b) des branches

Exemple de solution : [2,1] → [1,2]

c) des conditions

Exemple de solution : [2,1,3] → [1,2,3]

Question 5 (25 pts) - Un système d'exploitation de téléphone cellulaire simple est composé des modules suivants :



a) Décrivez une stratégie de test incrémentale pour cette application qui tente de minimiser l'effort de test. Donnez l'ordre d'intégration de chacun des modules. Justifiez vos choix en fonction des faux modules nécessaires aux tests.

Comme un système partiel n'est pas vraiment utile, on favorise une approche de bas en haut. Il faudra donc créer des pilotes pour chaque module à tester. Puisque le système contient des cycles de dépendances entre modules, il faudra inévitablement écrire des pilotes et des modules souches. Comme le module "Affichage" ne dépend d'aucun autre module, il devrait être testé en premier. Ceci permet aussi de tester les autres modules plus facilement puisque leur sortie peut être observée directement. Le module "Contrôleur radio" interagit directement avec le matériel, il est donc tout indiqué pour la création d'un module souche qui le remplace. Avec la création de ce faux module, il devient donc possible de tester en séquence les modules "Encodeur audio", "Contrôleur audio" et "Décodeur audio". À ce point, le vrai module "Contrôleur audio" peut être testé puisque toutes ses dépendances sont satisfaites. Les tests peuvent aussi continuer avec le module "Gestion des appels" (avec ou sans le faux module créé précédemment, pour accroître le parallélisme des tests). Comme les modules "Clavier" et "Carnet de contacts" sont mutuellement dépendants, il faudra choisir l'un deux pour la création d'un deuxième module souche (par exemple, le module "Carnet de contacts"). L'autre module pourra ensuite être testé, et le module souche remplacé par le vrai module puis testé à son tour.

b) Donnez 3 types de tests de système (autre que les tests de fonctionnalité) qui sont particulièrement importants pour ce type d'application. Pour chacun de types de tests, décrivez son importance et mentionnez un exemple concret de test pour le système en question.

Plusieurs solutions possibles, par exemple :

- Tests de facilité d'utilisation : un téléphone doit être facilement utilisable sans longue période d'apprentissage par un grand nombre de personnes ayant des connaissances techniques variées. Il est donc important de tester, par exemple, la simplicité des options de l'interface, son uniformité, ou encore si l'interface réagit promptement aux actions d'un utilisateur.
- Tests de stockage : un système d'exploitation de téléphone est un système embarqué qui doit répondre à des contraintes strictes d'utilisation des ressources, qui sont souvent très limitées. Il faut donc tester, par exemple, si l'utilisation de la mémoire correspond bien aux limites imposées.
- Tests de récupération : un téléphone qui rencontre une défaillance doit pouvoir revenir à son état normal sans avoir recours à un réparateur ou sans avoir à suivre une procédure complexe. Par exemple, il faudrait tester que le système peut reprendre l'exécution normale des opérations après un redémarrage en injectant des fautes dans le système ou en forçant un état interne erroné pour déterminer comment il se comporte dans ces situations.