Université A/MIRA BEJAIA / Faculté des sciences exactes
Département : Informatique
Module : GL
Niveau : 3ème année Licence
Année universitaire : 2024/2025

Corrigé de la Série TD Nº 1

Exercice 1 :

1. Qui sont les principaux acteurs impliqués dans le développement de logiciels et quel est le rôle de chacun brièvement.

Les principaux acteurs impliqués dans la réalisation d'un projet logiciel, ainsi que les rôles de chacun sont les suivants :

1. **Les utilisateurs finaux** : Ce sont les personnes qui vont utiliser le produit logiciel une fois qu'il sera développé. Leurs besoins, leurs attentes et leurs commentaires sont essentiels pour orienter le processus de développement.

2. **Les propriétaires du produit** : Ce sont les personnes ou l'organisation qui ont la vision globale du produit et qui en définissent les objectifs et les priorités. Ils sont responsables de la prise de décision concernant les fonctionnalités à développer et de la définition de la stratégie globale du produit.

3. **Les analystes fonctionnels** : Ils sont chargés de comprendre les besoins métier des utilisateurs et de les traduire en spécifications fonctionnelles. Ils jouent un rôle clé dans la communication entre les utilisateurs finaux et l'équipe de développement.

4. **Les développeurs** : Ce sont les professionnels qui écrivent le code et développent le produit logiciel. Ils utilisent des langages de programmation, des frameworks et des outils de développement pour créer les fonctionnalités du produit.

5. **Les testeurs** : Ils sont responsables de la validation et de la vérification du produit logiciel. Ils effectuent des tests pour s'assurer que le logiciel fonctionne correctement, qu'il répond aux spécifications et qu'il est exempt de bugs ou d'erreurs.

6. **Les chefs de projet** : Ils sont chargés de coordonner et de gérer le processus de développement du produit logiciel. Ils veillent à ce que les objectifs soient atteints, les délais respectés et les ressources allouées de manière efficace.

7. **Les concepteurs d'interface utilisateur** : Ils se concentrent sur l'aspect visuel et l'expérience utilisateur du produit logiciel. Ils créent des maquettes et des prototypes d'interface utilisateur pour s'assurer que le logiciel est convivial et intuitif.

8. **Les architectes logiciels** : Ils conçoivent l'architecture globale du logiciel, définissent les structures et les composants clés, et veillent à ce que le logiciel soit évolutif, performant et sécurisé.

9. **Les responsables qualité** : Ils s'assurent que le produit logiciel respecte les normes de qualité définies, les bonnes pratiques de développement et les processus de gestion de la qualité.

Informations Supplémentaires

- **Variabilité de l'Équipe**
  La composition de l'équipe peut changer selon la taille du projet et l'organisation. Parfois, d'autres acteurs comme des experts métier, des responsables marketing ou des représentants du service client peuvent être impliqués.

- **Importance de la Communication**
  La collaboration et la communication entre tous les acteurs sont cruciales pour le succès du projet. Des outils de gestion de projet peuvent faciliter cette interaction.

- **Cycle de Vie du Développement**
  Chaque acteur peut jouer un rôle différent à chaque étape du cycle de vie du développement logiciel, de l'analyse des besoins à la maintenance post-lancement.

Remarque :

L'acteur qui rédige le cahier des charges est généralement **l'analyste fonctionnel**. Son rôle consiste à comprendre les besoins des utilisateurs et à les traduire en spécifications détaillées. Le cahier des charges sert de document de référence pour l'ensemble du projet, définissant les exigences fonctionnelles et non fonctionnelles du produit.
Dans certains cas, le **propriétaire du produit** ou un **chef de projet** peut également être impliqué dans la rédaction ou la validation du cahier des charges, surtout pour s'assurer qu'il correspond à la vision stratégique et aux objectifs du projet.

2. Quelles sont les qualités requises dans un cahier de charges ?
   - Un bon niveau de spécificité ;
   - Une formulation adéquate des besoins et une description claire du problème ;
   - Être précis et éviter toute ambiguïté, même lorsque le langage utilisé est naturel et non mathématique ;
   - Être complet afin d'éviter toute omission involontaire ;
   - Être cohérent, sans inférence de fonctionnalités non définies ;
   - Être vérifiable : définir des critères de validation pour évaluer la faisabilité des besoins, éventuellement en créant une maquette ou en réalisant une simulation ;
   - Être modifiable : faciliter l'expression de modifications ou d'ajouts de besoins.

3. Quelles mesures peuvent-être prises pour diminuer l’écart entre les besoins réels d’un client et ceux formulés dans le cahier des charges ?

   Les mesures qui peuvent être prises pour réduire l’écart entre les besoins réels d’un client et ceux formulés dans le cahier des charges sont les suivantes :

   1. **Communication régulière** : Établir un dialogue ouvert avec le client via des réunions et des entretiens pour bien comprendre ses besoins.

   2. **Analyse des besoins** : Effectuer une analyse approfondie en identifiant les objectifs, les cas d'utilisation et les contraintes spécifiques.

   3. **Implication des parties prenantes** : Engager les utilisateurs finaux et experts dès le début pour recueillir diverses perspectives et assurer une compréhension précise.

   4. **Modélisation visuelle** : Utiliser des diagrammes et des wireframes pour clarifier les exigences et réduire les ambiguïtés.

   5. **Prototypage et itérations** : Créer des prototypes pour valider les exigences et recueillir des retours, permettant des ajustements successifs.

   6. **Validation continue** : Confirmer régulièrement les exigences avec le client à travers des démonstrations et des sessions de rétroaction.

   7. **Documentation claire** : Rédiger un cahier des charges précis et compréhensible pour éviter les malentendus.

   8. **Suivi des changements** : Établir un processus pour gérer l'évolution des besoins tout au long du projet.

4. Expliquer pourquoi il est utile de faire la distinction entre la définition des besoins et spécification des besoins.

   Faire la distinction entre la définition des besoins et la spécification des besoins est crucial pour la réussite d'un projet. La définition des besoins consiste à identifier et comprendre les attentes des parties prenantes, clarifiant ainsi le "pourquoi" du projet, tandis que la spécification des besoins se concentre sur le "quoi" et non pas sur le "comment", en détaillant les exigences fonctionnelles et non fonctionnelles. Cette distinction aide à éviter les malentendus, à gérer les attentes, et facilite la conception et le développement en fournissant des critères clairs. De plus, elle améliore la traçabilité des besoins tout au long du cycle de vie du projet et contribue à réduire les risques en permettant d'identifier et d'atténuer les problèmes potentiels dès le départ.

Remarque :

Ces deux étapes, bien que liées, jouent des rôles distincts dans le processus de gestion de projet. Une distinction claire entre la définition et la spécification des besoins permet d'optimiser la communication, de faciliter le développement et d'assurer un meilleur alignement avec les attentes des parties prenantes.

5. Quels sont les domaines d'application les plus adaptés aux trois modèles de cycle de vie étudiés en cours.

Les modèles de cycle de vie en cascade, en V et en spirale sont appliqués dans divers domaines en fonction de leurs caractéristiques spécifiques. Voici quelques exemples de domaines d'application adaptés à chaque modèle :

- **Modèle en cascade :**
  - Développement de logiciels où les exigences sont bien définies et stables, comme les applications de bureau ou les logiciels d'entreprise standardisés.
  - Projets de développement de logiciels pour lesquels la planification préalable est essentielle et où les étapes doivent être réalisées séquentiellement, comme les projets gouvernementaux ou les systèmes administratifs.

- **Modèle en V :**
  - Développement de logiciels qui nécessitent une validation et une vérification rigoureuses, tels que les logiciels critiques pour la sécurité, les systèmes de contrôle industriels ou les applications médicales.
  - Projets de développement de logiciels qui exigent une traçabilité rigoureuse entre les exigences, les tests et les activités de développement, tels que les logiciels réglementés ou les systèmes intégrés.

- **Modèle en spirale :**
  - Projets de développement de logiciels complexes où l'adaptabilité et l'atténuation des risques sont essentielles, comme les systèmes de défense, les systèmes financiers ou les systèmes de gestion de la qualité.
  - Développement de logiciels où les exigences sont évolutives et nécessitent des itérations successives, comme les projets de recherche et développement ou les logiciels innovants.

Il est important de noter que ces modèles de cycle de vie peuvent être adaptés et combinés en fonction des besoins spécifiques de chaque projet. Le choix du modèle approprié dépendra des caractéristiques du projet, des exigences du domaine et des contraintes spécifiques auxquelles il est confronté.

Exercice 2 :

1. Risques et techniques de résolution pour chaque approche :

a) Acheter un système de gestion de bases de données et développer son propre système basé sur cet outil :

- **Risques possibles :**
  - **Risque de coût élevé :** L'achat d'un système de gestion de bases de données peut être coûteux, et le développement ultérieur du système peut également nécessiter des ressources financières importantes.
  - **Risque de délai de mise en œuvre :** Le développement d'un système personnalisé peut prendre du temps, retardant, ainsi la mise en place du système intégré.

- **Techniques de résolution de risque :**
  - Faire une analyse financière approfondie pour évaluer les coûts réels de l'achat et du développement du système.
  - Établir un calendrier de développement réaliste en tenant compte de toutes les étapes nécessaires et des ressources disponibles.

b) Acheter un système comparable à celui d'une autre université et le modifier pour ses propres besoins :

- **Risques possibles :**
  - **Risque de compatibilité :** Le système acheté peut ne pas être parfaitement compatible avec les besoins spécifiques de l'université.
  - **Risque de dépendance :** En modifiant le système existant, l'université peut devenir dépendante du fournisseur initial pour les mises à jour et le support technique.

- **Techniques de résolution de risque :**
  - Effectuer une évaluation approfondie du système avant l'achat pour vérifier sa compatibilité avec les besoins spécifiques de l'université.
  - Négocier des accords contractuels solides avec le fournisseur, garantissant un support technique à long terme et des mises à jour régulières.

c) Se joindre à un groupe d'autres universités, établir un cahier des charges commun, contacter une société de logiciels qui développera un seul système pour tous :

- **Risques possibles :**
  - **Risque de compromis :** Les besoins spécifiques de chaque université peuvent ne pas être entièrement satisfaits dans le système développé en commun.
  - **Risque de coordination :** La coordination entre les différentes universités peut être complexe et nécessiter des efforts supplémentaires.

- **Techniques de résolution de risque :**
  - Établir un processus de communication et de collaboration clair entre les universités pour garantir que les besoins de chacune sont pris en compte.
  - La désignation d'une équipe de projet dédiée, comprenant des représentants de chaque université partenaire, peut faciliter la coordination et la communication. La mise en place d'une structure de gouvernance claire et l'utilisation d'outils

de gestion de projet collaboratifs peuvent également contribuer à la coordination efficace des activités.

2. Quelle est la meilleure approche à adopté ?
La meilleure approche dépendra de la culture et des priorités de l'université Abderrahmane-Mira de Bejaia. Si elle valorise la personnalisation et dispose des ressources nécessaires, l'approche de l'achat d'un système performant et du développement personnalisé pourrait être privilégiée. Si elle souhaite tirer parti de l'expérience d'une autre université renommée, l'approche de l'achat et de la personnalisation d'un système similaire pourrait être envisagée. Si elle accorde une grande importance à la collaboration et au partage des meilleures pratiques, l'approche de la collaboration avec d'autres universités prestigieuses pourrait être la plus adaptée.

3. La liste de fonctionnalités que pourrait contenir cette application, est comme suit :

Espace administration :
+ Gestion des étudiants (nom, prénom, carteIdentNatio, dateNaissance, adresse, niveau, filière, groupe, etc.)

- Ajout d'un étudiant
- Modification d'un étudiant
- Suppression d'un étudiant
- Consulter la liste des étudiants / Chercher
- Importer une liste des étudiants
- Gestion des filière/parcours (nom, abréviation)
- Gestion niveau (titre, filière)
- Gestion des groupes (nom, niveau)
- Gestion des matières (nom, coefficient, volume horaire, crédit, enseignant)
- Gestion des enseignants (nom, prénom, téléphone, email, grade, spécialité)
- Gestion des grades (titre, charge d'enseignement)
- Gestion des spécialités (titre)
- Calcul des moyennes et des classements des étudiants.
- Génération de relevés de notes et de rapports d'évaluation.
- Gestion des horaires des cours et des emplois du temps des étudiants.
- Gestion des demandes de diplômes et des certifications.

Espace enseignant :

- Authentification
- Mise à jour des informations personnelles
- Consulter les matières / étudiants
- Consulter les étudiants
- Gestion des notes (saisi / Mise à jour)
- Communication avec les étudiants (messages, notifications, etc.).

- Gestion des ressources pédagogiques (documents, supports de cours, etc.).

Espace étudiant :

- Authentification
- Mise à jour des informations personnelles
- Consulter les matières / enseignants
- Consulter les notes

Possibilité d'extension future pour inclure de nouvelles fonctionnalités. Ainsi, cette liste n'est pas exhaustive et peut varier en fonction des besoins spécifiques de chaque université.