LISV Laboratoire d'ingénierie des systèmes de Versailles
UVSQ université PARIS-SACLAY

Analyse – TD 2 : Gestion d'un projet informatique et cycle de vie

Halim Djerroud

révision 2.1

Exercice 1. [Lire et écrire un cahier des charges]

Nous (client) souhaitons un cœur d'application permettant de gérer l'ensemble des ouvrages d'une bibliothèque universitaire. Tous nos développements étant en Python, nous souhaitons donc une application développée en Python.

La bibliothèque est ouverte aux étudiants et enseignants de l'université. Elle permet d'emprunter des livres, de les réserver et de consulter le catalogue. Le personnel de la bibliothèque souhaite que le système lui permette de gérer l'ensemble des ouvrages disponibles, les prêts et les retours.

Les ouvrages peuvent être des **manuels**, des **revues** ou des **articles**. Pour chaque manuel, on considère le titre, l'auteur, l'ISBN et l'année. Pour chaque revue, on considère le titre, le numéro et la date de parution. Pour chaque article, on considère le titre, l'auteur et la revue d'origine.

1. Proposez une lecture de ce cahier des charges.
2. Listez les exigences par catégorie.
3. Quelles questions faudrait-il poser pour lever des ambiguïtés ?
4. Rédigez un cahier des charges consolidé.
5. Classez les exigences fonctionnelles (cf. cours).

Correction Exercice 1

1) Lecture du cahier des charges

Le cahier des charges décrit un système de gestion pour une bibliothèque universitaire. Les principaux **objets manipulés** sont les ouvrages (manuels, revues et articles). Les **acteurs** sont :

— les étudiants et les enseignants, qui utilisent le système pour consulter le catalogue, emprunter et réserver des ouvrages ;

— le personnel de la bibliothèque, qui gère l'ensemble du fonds documentaire ainsi que les prêts et retours.

Les **actions principales** consistent à gérer les ouvrages (ajout, modification, suppression), à permettre aux usagers de consulter le catalogue et d'effectuer des emprunts et des réservations, et à contrôler les retours.

Objet	Acteur	Action
Manuel / Revue / Article	Étudiant / Enseignant	Consulter, réserver, emprunter
Prêt	Personnel	Enregistrer un prêt / retour
Ouvrage	Personnel	Ajouter, modifier, supprimer

2) Exigences par catégorie

Les exigences **fonctionnelles** concernent la possibilité de consulter le catalogue, d'emprunter ou de réserver des ouvrages, et pour le personnel de gérer l'ensemble du cycle de vie des prêts et retours. Les exigences **non-fonctionnelles** incluent notamment le fait que l'application doit être développée en Python, qu'elle doit être simple d'utilisation et garantir un minimum de sécurité pour les données personnelles des usagers. Du point de vue **organisationnel**, l'accès est réservé aux étudiants et enseignants, tandis que le personnel assure la gestion quotidienne de la base documentaire. Enfin, des exigences **techniques** apparaissent : il faudra prévoir une base de données robuste, ainsi qu'une architecture évolutive permettant d'ajouter de futures fonctionnalités.

— **Fonctionnelles** : emprunter, réserver, consulter le catalogue, gérer prêts/retours, gérer ouvrages.

— **Non-fonctionnelles** : application en Python, sécurité des données, ergonomie simple.
— **Organisationnelles** : accès réservé aux étudiants/enseignants, personnel gérant le fonds documentaire.
— **Techniques** : intégration base de données, évolutivité.

3) Questions à poser pour lever les ambiguïtés

Plusieurs points restent à éclaircir avec le client :
— Comment gérer les retards et pénalités ? Le système doit-il bloquer automatiquement les usagers en retard, ou se limiter à un signalement ?
— Combien de prêts ou de réservations simultanés un utilisateur peut-il effectuer ?
— Les enseignants disposent-ils de privilèges particuliers (ex. durée de prêt plus longue) ?
— Quelle est la durée standard d'un prêt ?
— Le client souhaite-t-il des statistiques (ouvrages les plus empruntés, fréquentation, etc.) ?
— Le système doit-il gérer une seule bibliothèque ou plusieurs sites universitaires ?

4) Cahier des charges consolidé (extrait)

L'objectif est de développer en Python une application de gestion des ouvrages d'une bibliothèque universitaire. Cette application permettra aux usagers (étudiants et enseignants) de consulter le catalogue, de réserver des ouvrages et d'effectuer des emprunts. Le personnel disposera d'outils pour gérer les ouvrages (ajout, suppression, mise à jour des informations) et pour suivre les prêts et les retours. Le système devra en outre assurer la sécurité des données des usagers et offrir une interface adaptée à chacun des deux profils (usager et personnel).
— **Objectif** : développer en Python un système de gestion des ouvrages d'une bibliothèque universitaire.
— **Fonctionnalités principales** :
  — Consultation du catalogue par étudiants et enseignants.
  — Réservation et emprunt d'ouvrages.
  — Gestion par le personnel (ajout, suppression, suivi prêts/retours).
— **Contraintes** :
  — Développement en Python.
  — Sécurité des données utilisateurs.
  — Interface adaptée aux usagers et au personnel.

5) Classification des exigences fonctionnelles

— **Niveau stratégique** : gérer efficacement la bibliothèque universitaire grâce à un outil centralisé.
— **Niveau utilisateur** :
  — Consulter le catalogue
  — Réserver un ouvrage
  — Emprunter / retourner un ouvrage
— **Niveau sous-fonction** :
  — Enregistrer un ouvrage
  — Identifier un utilisateur
  — Gérer les attributs spécifiques (ISBN pour les manuels, revue d'origine pour les articles, date de parution pour les revues)

Exercice 2. [Classer les exigences fonctionnelles]

On souhaite développer un logiciel permettant de gérer un service de rendez-vous médicaux. L'application devra permettre à un médecin de consulter son planning et de planifier un rendez-vous pour un patient. Ainsi, le médecin enregistre le patient, planifie le rendez-vous et génère un rappel automatique pour le patient.

Pour cela, le système doit permettre :
— d'enregistrer un patient,
— d'identifier un médecin,
— d'associer une date et une heure de rendez-vous,
— d'envoyer un rappel (mail/SMS) au patient.

1. Lire le cahier des charges.
2. Identifier les exigences fonctionnelles.
3. Détecter les niveaux utilisateurs, stratégique et sous-fonctions.
4. Représenter les cas d'utilisation en fonction de leur niveau.

Correction Exercice 2

1) Lecture du cahier des charges

Le cahier des charges porte sur un logiciel de gestion des rendez-vous médicaux. Les principaux **acteurs** sont les médecins, qui consultent leur planning et planifient des rendez-vous, et les patients, qui reçoivent un rappel de leur rendez-vous. Les **objets** manipulés sont le patient, le médecin et le rendez-vous. Les **actions principales** consistent à enregistrer un patient, identifier un médecin, planifier un rendez-vous avec une date et une heure, et générer automatiquement un rappel envoyé au patient.

Objet	Acteur	Action
Patient	Médecin	Enregistrer un nouveau patient
Médecin	Système	Identifier le médecin / accéder au planning
Rendez-vous	Médecin	Associer une date et une heure
Notification	Système	Envoyer un rappel (mail/SMS) au patient

2) Identification des exigences fonctionnelles

Les exigences fonctionnelles explicites sont les suivantes :
- Enregistrer un patient.
- Identifier un médecin.
- Associer une date et une heure de rendez-vous.
- Envoyer un rappel automatique au patient.

On peut également en déduire des exigences implicites, comme la consultation du planning par le médecin ou la mise à jour d'un rendez-vous existant.

3) Détection des niveaux (stratégique, utilisateur, sous-fonctions)

- **Niveau stratégique** : gérer l'ensemble des rendez-vous médicaux.
- **Niveau utilisateur** :
  - Consulter le planning d'un médecin.
  - Planifier un rendez-vous pour un patient.
  - Générer un rappel automatique.
- **Niveau sous-fonction** :
  - Enregistrer un patient.
  - Identifier un médecin.
  - Associer une date et une heure de rendez-vous.
  - Envoyer un rappel (mail/SMS).

4) Cas d'utilisation par niveau

- **Cas d'utilisation stratégique** : « Gérer les rendez-vous médicaux ».
- **Cas d'utilisation utilisateur** : « Consulter le planning », « Planifier un rendez-vous », « Gérer les rappels ».
- **Cas d'utilisation sous-fonctions** : « Enregistrer un patient », « Identifier un médecin », « Associer une date/heure », « Envoyer un rappel ».

Ces cas d'utilisation peuvent être représentés par un diagramme hiérarchisé (comme vu en cours), où le cas d'utilisation stratégique englobe les cas utilisateurs, eux-mêmes composés de sous-fonctions.