TD 1: analyse et spécification des charges

GLA — Master 1 Informatique — Université Paris Diderot

Séances du 21 et 22 février 2012

1 Introduction

L'analyse des charges (ou besoins ou en anglais requirements) et leur spécification constituent la première étape du processus de développement d'un système (produit) logiciel. Son rôle est très important car elle définit ce que le système doit faire, ses propriétés indispensables ou désirées, les contraintes sur les fonctionnalités du système ou sur le processus de développement.

Le but de ce TD est de rédiger les parties préliminaires du cahier des charges pour plusieurs études de cas. Une description partielle de ces études de cas se trouve dans les sections suivantes. Vous devez utiliser votre connaissance des systèmes proposés pour compléter ces descriptions avec les charges manquants. Pour rédiger un cahier des charges, la norme IEEE/ANSI 830-1998 donne une recommandation du plan du document (voir section 3.1). Cette recommandation s'applique à tout type de système. Un plan plus spécifique aux systèmes logiciels se trouve en section 3.2 ; pour ce TD vous devrez vous concentrer sur les parties 1–4 et 6 de ce plan.

Comme exemples et références, vous disposez des trois ébauches de cahiers de charges. Lisez bien ces documents avant de passer aux études de cas suivantes.

Plusieurs études de cas sont fournis ci-dessous ; pour chacune, répondez aux questions suivantes :

1. Quelle est la frontière du système ?
2. Quels sont les charges des différents natures ? (voir "classification des charges" ci-dessous)
3. D'après vous, quels sont les acteurs principaux de ce système ?
4. Énumérer les scénarios plus importantes d'utilisation du système. Donner une description détaillée (contexte, flot normal, cas problématique, activités concurrente) d'au moins 3 scénarios.
5. Donnez au moins 2 diagramme de cas d'utilisation du système. Chacun doit regrouper au moins 3 scénarios.

2 Classification des charges — rappel

La classification des charges en fonction de leur niveau de détail :

- Charges d'utilisation (CU) : les services à fournir et les contraintes de fonctionnement a exprimer en langage naturel aidé par des diagrammes (par exemple les diagrammes des cas d'utilisation de UML).
- Charges du système (CS) : les fonctions du système spécifiées en détail ; c'est un contrat entre le client et le fournisseur du système. Les charges du système sont classifiées en fonction de leur type comme suit :
  - charges fonctionnels (CSF) : les services que le système doit fournir, comment le système doit réagir à des entrées ou des situations particulières.
  - Charges non-fonctionnels (CSNF) : contraintes qualitatives et quantitatives sur les services offerts comme les délais de réponse, les standards à respecter, le type de processus de développement à adopter, les contraintes de sécurité ou confidentialité, etc.
  - Charges dus au domaine (CSD) : contraintes imposés par le domaine d'utilisation du e système et qui reflètent les caractéristiques de ce domaine.

3 Plan du document de description des besoins

3.1 Recommandation IEEE/ANSI 830-1998

1. Introduction
   (a) Purpose of the requirements document
   (b) Scope of the product
   (c) Definitions, acronyms and abbreviations
   (d) References
   (e) Overview of the reminder of the document
2. General description
   (a) Product perspective
   (b) Product functions
   (c) User characteristics
   (d) General constraints
   (e) Assumptions and dependencies
3. Specific requirements
   Cover functional, non-functional and interface requirements. This is obviously the most substantial part of the document but because of the wide variability in organisatio- nal practice, it is not appropriate to define a standard structure for this section. The requirements may document external interfaces, describe system functionality and per- formance, specify logical database requirements, design constraints, emergent system properties and quality characteristics.
4. Appendices
5. Index

3.2 Recommandation spécifique au logiciel

Ce plan est recommandé dans le livre "Software engineering" par Ian Sommerville.

1. **Préface :** Décrire la structure du document, ses différentes versions, un résumé des raisons pour lesquelles les différentes versions du document ont été conçues.
2. **Introduction :** Indiquer à quel besoin le produit répond. Décrire brièvement les fonctionna litées du produit et ses interaction avec d'autres systèmes. Positionner le produit par rapport à la stratégie ou les objectifs du client.
3. **Glossaire :** Définir les termes techniques utilisés dans le document sans supposer un lecteur averti.
4. **Charges d'utilisation :** Décrire les fonctionnalités offertes à l'utilisateur ainsi que des charges non fonctionnelles du systèmes. Cette description peut utiliser le langage naturel, des diagrammes ou d'autres notation compréhensibles par les clients. Les standards qui doivent être respectés par le produit doivent être spécifiés.
5. **Architecture du système :** Présenter une vue de haut niveau de l'architecture préconisée du système et la distribution de fonctionnalités à travers les modules du système. Les composantes réutilisée de l'architecture doivent être soulignées.
6. **Charges du système :** Décrire les charges fonctionnelles en détail et d'autre charges non fonctionnelles, par exemple donner des détails sur les charges d'interface avec d'autres systèmes.
7. **Modèles du système :** Donner un ou plusieurs modèles du système et montrer la relation entre les composantes du système et son environnement.
8. **Évolution du système :** Décrire les hypothèses fondamentales sur lesquelles le système a été construit et anticiper les changements dus à l'évolution du matériel, aux changements des besoins d'utilisations, etc.

9. **Annexes :** Fournir des informations détaillées et spécifiques au produit développe. Exemples d'annexes qui peuvent être données : description du matériel et de la base de donnée utilisée. Pour le matériel, définir la configuration minimale et optimale pour utiliser le produit. Pour la base de données, définir le modèle relationnel.
10. **Index :** Plusieurs indexations du document peuvent y paraître : l'index des termes utilisés, l'index des diagrammes, l'index des fonctions, etc.

4 Étude de cas : VELIV

Ce sujet porte sur la conception d'un logiciel permettant de gérer l'emprunt des livres dans une bibliothèque libre-service, nommé VELIV. Le système physique s'appuie sur :

- des cartes magnétiques distribuées aux abonnés
- les livres
- une borne d'entrée, située à l'extérieur de la bibliothèque
- une borne de consultation (à l'intérieur)
- une borne d'emprunt (à l'intérieur)
- une boite à livre pour retourner les livres empruntés (à l'extérieur)

Chaque carte magnétique est identifiée par un code unique abonné à 8 chiffres, qui identifie l'abonné ; chaque carte est aussi associé à un code PIN secret, connu seulement par l'abonné, et à une période de validité. Pour entrer, les abonnés doivent présenter leur cartes auprès de la borne d'entrée et entrer leur code PIN. La borne d'entrée contrôle l'ouverture des portes d'entrée et est reliée à un ordinateur central qui contient le système de gestion VELIV. Les bornes de consultation et emprunt sont elles aussi reliées au système de gestion VELIV ; pour les utiliser est nécessaire de s'identifier avec la carte magnétique abonné. Chaque livre est équipé avec un radio-identificateur (RFID), utilisé à la fois pour identifier le livre à l'intérieur de la bibliothèque et comme dispositifs antivol. Entre la sortie de la bibliothèque et l'espace livres, des barrières RFID surveillées par un employée sont présentes, pour éviter la sortie avec livres qui n'ont pas été régulièrement empruntés.

Chaque radio-identificateur de livre est associé à un code unique à 13 chiffres (l'ISBN du livre). Pour emprunter un ou plusieurs livres, l'abonné se rende à la borne d'emprunt ou, après identification, il pourra valider les livres choisis avant de sortir de la bibliothèque. Pour retourner un livre, il est suffisant de le glisser dans la boite à livre, sans besoin de s'identifier ou d'entrer dans la bibliothèque.

Chaque abonné peut emprunter jusqu'à un maximum des 3 livres à la fois. Chaque livre peut être emprunté pour un maximum de 45 jours. Si un livre n'est pas retourné avant le délai maximum, l'accès à tous services de VELIV sont suspendus pour l'abonné, qui devra se rendre au service client pour être habilité à nouveau.

Chaque abonné peut utiliser la borne de consultation pour consulter le catalogue de livres de la bibliothèque. Le catalogue montre les livres possédés pas la bibliothèque, leur état (déjà emprunté ou pas), leur collocation physique (si disponibles) et leur meta-donnés littéraires typiques (titre, auteur, genre, éditeur, etc.).

Votre tache est de *concevoir le système de gestion VELIV* en respectant les meilleurs pratiques de réutilisation de code et d'extensibilité. Le système de gestion sera développé dans un langage de programmation à objets.

5 Étude de cas : FlyBleu

Ce sujet porte sur la conception d'un logiciel pour gérer les réservations de billets d'avion pour la compagnie aérienne FlyBleu. Le logiciel à concevoir est un site web qui permet de :

- s'enregistrer comme nouvel utilisateur du système, en donnant un nom d'utilisateur (unique entre tous les utilisateurs) et un mot de passe (pas "trop simple" et dont la longueur doit être comprise entre 4 et 10 caractères)
- accéder au site avec son propre nom utilisateur et mot de passe
- rechercher des vols avec les paramètres habituels (destination, date, préférence de siège, etc.)
- réserver des billets pour les vols trouvés et les payer
- effectuer le *check-in* et imprimer la carte d'embarquement

– annuler une réservation

Le logiciel de réservation doit interagir avec d'autres logiciels, plus particulièrement :

– Les payements sont effectué grâce à des logiciels externes, fourni par plusieurs banques, qui vérifient les données de la carte bleu de l'utilisateur et informent FlyBleu si les payements ont été acceptés (ou pas). Chaque banque fourni un *API* bien défini pour l'utilisation de son logiciel, mais différente pour chaque banque.
– Certains vols sont opérés par des compagnies aériennes partenaires de FlyBleu. Ces vols sont affichés dans les résultats de recherche du site FlyBleu, mais leur réservation passe par un logiciel externe—un pour chaque partenaire—accessible via API comme dans le cas des payements externes.

Votre tâche est de *concevoir le site web et le système de réservation FlyBleu* en respectant les meilleures pratiques de réutilisation du code et d'extensibilité. Le système sera développé dans un langage de programmation à objets.