Conseils supplémentaires – Cas d'utilisation

Ces informations complètent le contenu des textes 2.3 **Cas d'utilisation** (chapitre 5 et 6, pp. 61-110) et 2.4 **Compléments sur les cas d'utilisation** (chapitre 24, pp. 465-476) ou mettent en évidence certaines idées présentes dans ces textes.

Définitions

Cas d'utilisation

- Les cas d'utilisation permettent d'instaurer un dialogue avec le client et de constituer un support pour l'analyse de qualité.
- Les cas d'utilisation sont des besoins fonctionnels qui décrivent le système en termes de responsabilités.
- Un cas d'utilisation est constitué d'un ensemble de séries d'actions réalisées par le système qui produit un résultat tangible et observable ayant de la valeur pour un acteur.
- Un cas d'utilisation peut aussi être vu comme une collection de scénarios de réussite ou d'échec qui décrit la façon dont un acteur utilise un système pour atteindre un but.

Scénario

- Un scénario est une suite d'actions et d'interactions entre les acteurs et le système à développer. Il est aussi possible de considérer un scénario comme un chemin logique unique au sein d'un cas d'utilisation.

Acteur

- Un acteur est une abstraction d'un rôle joué par des entités externes humaines, techniques ou informatiques, qui interagissent directement avec le système à développer.
- C'est une entité qui a un comportement (comme une personne, un système ou une entreprise).

Relation entre cas d'utilisation

- Relation *include* : le cas d'utilisation de base en incorpore explicitement un autre, à un moment spécifié de son déroulement. Cette relation permet de factoriser un comportement commun à plusieurs cas d'utilisation. Cette relation permet de déléguer une partie du déroulement d'un cas d'utilisation à un autre cas d'utilisation.
- Relation *extends* : le cas d'utilisation en incorpore implicitement un autre, à un moment spécifié. Cette relation permet de séparer le comportement obligatoire dans un cas d'utilisation, des comportements optionnels. Cette relation exprime une délégation conditionnelle d'une partie du déroulement d'un cas d'utilisation à un autre cas d'utilisation.
- Ni la relation *include*, ni la relation *extends* n'indique une spécialisation des cas d'utilisation, formes alternatives de cas d'utilisation, ou encore décomposition des cas d'utilisation en sous-cas d'utilisation.

Rappel sur la fonction des trois entités de base d'un diagramme de cas d'utilisation

- Qui intervient dans le cas? Les acteurs.
- Pourquoi interviennent-ils? Les cas.
- Qui fait quoi? Les associations entre acteurs et cas.

Qu'est-ce qu'ils ne sont pas?

- Un cas d'utilisation n'est pas une description du domaine mais il décrit des besoins.
- Un cas d'utilisation ne décrit pas comment le système fait les traitements, mais ce qu'il fait.
- Un cas d'utilisation n'est pas infaillible, mais il reste une approximation.
- Un cas d'utilisation ne répond pas à la question : « Que fait l'utilisateur? », mais répond plutôt à la question « Quels sont vos buts? ».
- Un acteur ne décrit pas un état, mais un comportement.
- Un acteur ne représente pas une personne particulière, mais un rôle.

Comment distinguer un acteur principal d'un acteur secondaire?

Les acteurs secondaires sont intéressés par la réalisation du cas d'utilisation, mais c'est l'acteur principal qui déclenche la réalisation du cas d'utilisation et qui est la personne la plus intéressée à ce que le cas d'utilisation se réalise.

Conseils pratiques

Pour identifier les acteurs, vous pouvez vous poser les questions suivantes :

- Pour qui réalise-t-on le système?
- À qui s'adresse le système?
- Qui doit interagir directement avec le système?
- Qui supervise le fonctionnement du système, qui en fait l'administration?

- Est-ce que les acteurs identifiés communiquent bien directement avec le système, par émission et (ou) réception de messages?
- Dans la situation où l'acteur pourrait être un système informatique, est-ce que l'acteur se situe bien à l'extérieur du système?
- Quels sont les rôles que jouent les acteurs par rapport au système informatique à développer? Gardez en tête qu'une même personne peut jouer successivement plusieurs rôles par rapport au système et, par conséquent, être modélisée par plusieurs acteurs. Réciproquement, le même rôle peut être joué simultanément par plusieurs entités externes concrètes (personnes ou systèmes), qui seront alors modélisées par le même acteur.

Pour identifier les cas d'utilisation, vous pouvez vous poser les questions suivantes :

- Quelles sont les tâches principales d'un acteur?
- Qu'est-ce que l'on attend du système?
- Quelle est l'information que doit fournir chaque acteur?
- Quel objectif de l'acteur le cas d'utilisation permet-il d'atteindre?
- Est-ce que le cas d'utilisation représente bien une fonction visible par l'acteur?
- Est-ce que le cas d'utilisation est nommé avec un verbe ayant un sens et un sens univoque?

En vrac

- Identifier l'ensemble des cas d'utilisation pour décrire les fonctionnalités du système.
- Utiliser le patron de description d'un cas d'utilisation et l'appliquer à 10 des cas d'utilisation identifiés précédemment en incluant obligatoirement les cas d'envois de messages; ce qui signifie qu'il faut décrire au minimum 10 cas d'utilisation de manière détaillée.
- Vous pouvez généraliser des acteurs. Si un ensemble d'acteurs communiquent de la même façon avec certains cas d'utilisation, on peut créer un acteur généralisé (souvent abstrait) qui permettra de factoriser les rôles qu'ils ont en commun.
- Regroupez vos cas d'utilisation en paquetages (ou package) (par métier, par acteur ou par lot de livraison).
- Demandez au client d'affecter une priorité fonctionnelle à chaque cas d'utilisation.
- Concernant les relations entre cas d'utilisation, développez plutôt les cas factorisés (include) avant ceux qui les utilisent; développez plutôt les cas qui étendent (extend) après les cas de base.

Quelques pièges à éviter

- Répertorier en tant qu'acteurs des entités externes qui n'interagissent pas directement avec le système, mais uniquement par l'intermédiaire d'un des vrais acteurs.
- Recenser des acteurs qui correspondent en fait à des composants internes au système étudié, voire à des futures classes.
- Privilégier les acteurs physiques à la place des acteurs logiques.
- Un cas d'utilisation n'est pas une fonction atomique. Une erreur fréquente concernant les cas d'utilisation consiste à vouloir descendre trop bas en termes de granularité.

- Confondre les interfaces graphiques du système et le fonctionnel. Une erreur fréquente concernant les cas d'utilisation consiste à rendre dépendants les cas d'utilisation à un choix prématuré d'interface usager.

Références

Roque, P., Vallée, F. 2000. *UML en action*. Éditions Eyrolles.

Pender, T. 2000. *Weekend Crash Course*. Wiley Publishing, Inc.