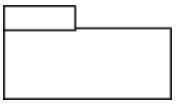
Support de cours en Systèmes d'information, Méthodes Avancées
Chapitre 5 : Analyse

1. Introduction

L'étape d'analyse est entamée à la suite de l'étape de capture de besoins fonctionnels. A la différence avec les approches classique (ex : Merise), l'analyse cible le système futur plutôt que le système existant. Elle est située sur la branche gauche du processus 2TUP et concert donc l'aspect métier. L'analyse permet d'approfondir la capture des besoins fonctionnels en détaillant l'aspect données par le développement du modèle statique et en détaillant l'aspect traitement en développant le modèle dynamique.

2. Découpage en catégories

La notion de catégorie est l'application du concept de package UML pour regrouper des classes identifiées dans les diagrammes de classes candidates. Une catégorie contient un ensemble de classes fortement couplées. Le couplage entre classes est issu de plusieurs situations (associations, agrégation, composition, héritage ou échange de messages).
Le découpage en catégorie permet une meilleure représentation de la structure statique du système sous forme de packages. Il permet en effet de répartir les responsabilités des catégories sur plusieurs équipes de développement, de réutiliser les catégories dans d'autres systèmes et de faciliter l'évolutivité, la réutilisation et la maintenance des composants du système.

Une catégorie est représentée graphiquement comme un paquetage 

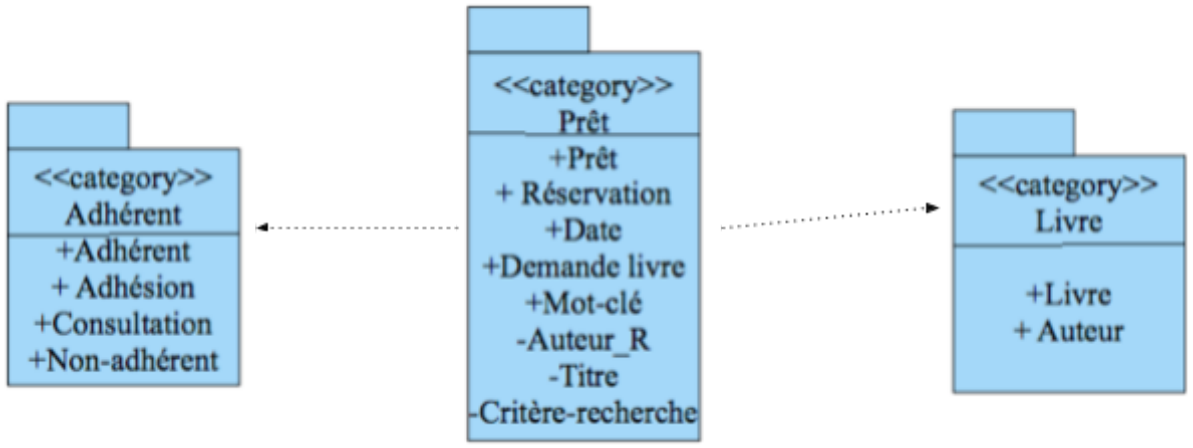
Critère du découpage : le point de départ est l'ensemble des diagrammes de classes. Il convient de regrouper les classes en catégories. Les critères de découpage peuvent être

- *la cohérence des classes* : dans le sens ou les classes sont liées sémantiquement
- *l'évolution* : en regroupant les classes a fort degré d'évolution ensemble et vice-versa
- *la durée de vie des objets* : on regroupe des classes dont les objets ont des durées de vies proches
- *la dépendance entre les classes* : la notion de dépendance est décrite plus loin

- *Notion de dépendance entre les catégories* : lors du découpage en catégorie des classes, il est naturel de se retrouver avec des liens entre des classes de catégories différentes. Ces liens traduits par des associations ou autres doivent être conservés. On utilise pour ce faire la notion de dépendance entre les catégories.

La notion de dépendance est liée à la notion d'utilisation de classes en dehors de leurs catégories d'origine. Lorsqu'une association existe entre deux classes A et B de catégories différentes, si à partir de la classe A on veut accéder à la classe B, cela se traduit par une dépendance entre les catégories. La catégorie de A dépend alors de la catégorie de B. cependant, l'utilisation des classes d'une catégorie en dehors de sa catégorie est conditionnée par sa visibilité à l'extérieur de la catégorie qui doit être publique. Dans ce cas, le nom de la classe est précédé par le symbole +. Par contre, lorsque la classe est précédée de – elle n'est pas visible en dehors de sa catégorie. L'utilisation de classe à partir dans les catégories autres que celle d'origine est symbolisée par le stéréotype <<import>>.

Exemple : Dans la gestion de la bibliothèque, on peut distinguer les catégories et leur dépendances comme suit



3. Développement du modèle statique

Le découpage en catégorie génère un regroupement et répartition des classes candidates. Ces classes ne sont pas raffinées et ils convient de les examiner de près. C'est l'objectif de cette étape.

3.1. Traitement des anomalies

L'analyse de près des classes a pour objectif d'éliminer les anomalies. Ces anomalies se manifestent sous plusieurs formes : redondance, classes à la place de rôle, Classe représentant des acteurs, Classe techniques (de conception), classe de groupes ou listes d'objets et classe grosses. D'autres concepts de modélisation sont à vérifier : multiplicités, navigabilité, associations ayant le sens d'agrégation ou de composition. Aussi, il est possible d'ajouter des contraintes aux extrémités des

associations, comme les contraintes d'ordre, d'ajout / suppression d'instance…ainsi que des stéréotypes aux classes et aux liens.

3.2. Traitement des attributs

Les attributs doivent être analysés pour en éliminer ceux qui traduisent des rôles. Aussi, on peut, pour expliciter des contraintes entre des propriétés, ajouter des attributs dérivés, en les faisant précéder du symbole /. Un attribut dérivé peut être calculé à partir des attributs de la même classe ou de classes différentes associées. Aussi, on prévoit à cette étape les attributs jouant le rôle de qualificatifs.

3.3. Traitement des opérations

L'identification de certaines opérations se fait par analyse textuelle du cahier de charges mais aussi à partir des cas d'utilisation.

Opérations particulières : certaines opérations décrivant des aspects logiciels peuvent apparaître au niveau du cahier des charges ou cas d'utilisation. Il faut les écarter à ce niveau là. Aussi, il faut écarter à ce niveau les opérations implicites sur les classes et les objets :

- création et destruction d'instances
- lecture et mise à jour des valeurs d'attributs
- création et destruction des liens entre les instances d'associations
- recherche des instances d'associations

Enfin, les noms des opérations doivent être abrégés (pas en toutes lettres).

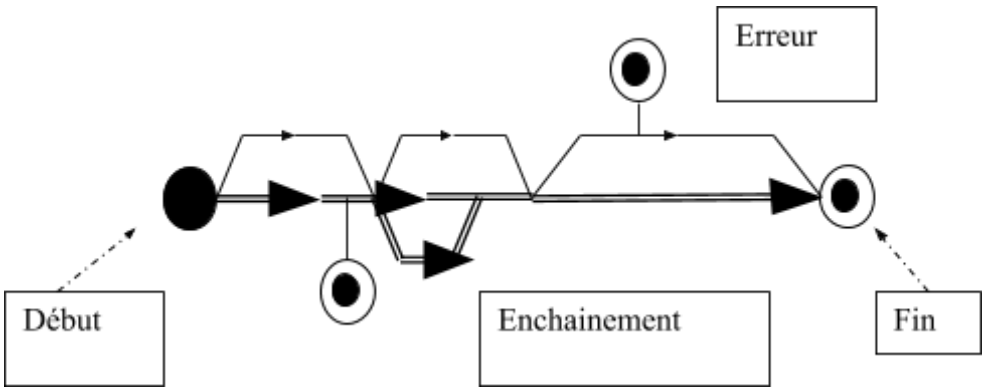
Ajout de contraintes : outre les règles de calculs, le diagramme de classes peut être enrichi, à ce niveau par des contraintes. **Exemple**: on peut exprimer la contrainte {date de recrutement > date de naissance} pour les employés dans le système de gestion du personnel. Les contraintes peuvent être écrites en langue naturelle ou en OCL (Object Constraint Language).

4. Développement du modèle dynamique

Le développement du modèle dynamique se fait en parallèle que le développement du modèle statique. Il concerne le détail de la description des cas d'utilisation par d'autres diagrammes, à savoir : les diagrammes de séquences et les diagrammes de collaboration.

4.1. Notion de scénario

La notion de scénario découle du fait que les enchaînements d'un cas d'utilisation ne sont pas exécutés tous à la fois, quoique l'ensemble des enchaînements constitue le cas complet.



Notion d'enchaînements et de scénarios(ref. UML en Action, P. Roques)

Un ensemble d'enchaînements ordonnées et exécutées d'emblée est appelé scénario. Il s'agit d'une configuration ou d'une exécution particulière du cas d'utilisation. Toutefois, il est possible de différencier les scénarios selon leur affectation du système, c'est-à-dire la manière dont ils aboutissent aux post-conditions du cas d'utilisation. Les types de cas d'utilisation sont les suivants :

- *Les scénarios nominaux :* il s'agit de scénarios qui permettent d'atteindre la fin du cas (et en conséquent réaliser la postcondition) de manière naturelle.
- *Les scénarios alternatifs :* il s'agit d'enchaînements qui aboutissent à la fin du cas mais dans des situations rares car il s'agit d'alternatives en face à des exceptions du traitement normal.
- *Les scénarios aux limites :* ces scénarios définissent les limites d'exécution du cas de telle sorte qu'une autre exécution du cas provoquerait une erreur.
- *Les scénarios d'erreur :* il s'agit de sortie anormales du cas d'utilisation qui n'aboutit pas aux post-conditions.

Exemple: prenons le cas d'utilisation "*gestion d'un adhérent de la bibliothèque*". Ce cas inclut la création de l'adhérent et la consultation de ses informations. Comme scénario alternatif, on peut citer la modification d'informations le concernant.

- scénarios nominaux: création de l'adhérent, consultation des informations sur un adhérent

- scénarios alternatifs : modification des informations sur un employé

- scénarios aux limites : s'il y a un nombre limité d'adhérents de la bibliothèque, l'ajout du dernier adhérent est un scénario à la limite.
- scénarios d'erreur : suppression d'un adhérent ayant des adhésions en cours.

4.2. Formalisation des cas d'utilisation

4.2.1. *Notion de message*

La notion de message est centrale dans la formalisation d'un scénario car un message est l'élément échangé entre les objets (y compris les acteurs). Un message est une spécification de communication entre les objets. Cette communication englobe l'ensemble des paramètres valorisés passés de l'émetteur vers le récepteur. Il existe deux catégories de messages :

- les signaux : il s'agit de messages asynchrones ne nécessitant pas de retour d'informations vers l'émetteur à partir du récepteur
- les appels d'opérations : il s'agit de messages synchrones qui nécessitent un retour d'informations vers l'émetteur à partir du récepteur

4.2.2. *Situations exceptionnelles*

- Messages destinés à un groupe d'objets : dans certains cas, il est possible concerne plusieurs instances d'objets. Par exemple, lors de l'affectation des diplômes à un employé, plusieurs instances de diplômes sont concernées.

- Messages de création / destruction d'objets : les instances auxquelles sont destinés les messages ne sont pas initialement pré-existant. Dans ce cas, un message particulier permet de créer l'instance. Cette opération peut être spécifiée par le stéréotype « create ». De même, un envoi de message peut être la destruction d'une instance d'objet.

- Les auto-messages : dans certaines situations, un message peut s'auto envoyer un message. Par exemple, on peut l'ajout de la date de recrutement d'un employé peut provoquer un message de demande de comparaison de la date de recrutement avec la date de naissance et peut produire une exception.

4.2.3. *Les différentes formalisations possibles*

Les diagrammes qui interviennent dans la formalisation des cas d'utilisation sont les quatre diagrammes vus dans la partie une du cours, à savoir le diagramme de séquences, le diagramme de collaboration, le diagramme d'états-transition et le diagramme d'activité. Ci-après quelques recommandations relatives aux diagrammes qui accompagnent les cas d'utilisation :

- **Diagrammes d'états-transitions**

    - Diagrammes facultatifs. Pas toutes les classes sont concernées par ce diagramme mais seulement celles qui ont un comportement dynamique complexe.
    - Au moins trois états par diagrammes

- Les événements induisant des changements d'état peuvent être (1) la réception de signal (2) la réception d'opération (3) le temps (on utilise le mot-clé *after*) et (4) la satisfaction d'une condition (on utilise le mot-clé *when*).

- **Diagrammes de séquences**

  - Recommandé.
  - Utiliser des notes pour spécifier les exceptions. Les notes peuvent être dessinées en marge et au même niveau que la flèche décrivant l'exception.
  - Utiliser les auto-messages sous la forme de flèche réflexive partant et revenant au même couloir de l'objet en question.

5. Conclusion

L'analyse est une étape importante qui permet de développer les modèles statiques (diagrammes de classes) et les modèles dynamiques (diagramme des cas d'utilisation). Pour le modèle statique, il s'agit de corriger les anomalies avant de compléter les diagrammes dans les étapes suivantes. Pour le modèle dynamique, le système est remplacé par l'ensemble des classes manipulées par le cas d'utilisation. Dans le chapitre suivant, nous abordons les étapes de conception (conception générique, conception préliminaire et conception détaillée).