Utilisation d’UML à des fins de recueil des besoins et d’analyse 1/2.

Objectifs de l’analyse

- Formalisation des besoins utilisateur.
  - Besoins fonctionnels
  - Besoins non fonctionnels
- Identification des principaux artefacts qui constitueront la solution.

Besoins fonctionnels

- Il s’agit de mettre en évidence les comportements observables de l’application.
- Exemple :
  - L’élève réserve un livre
  - L’élève consulte le catalogue.

Les besoins non-fonctionnels

- Il s’agit :
  - Des contraintes que l’application doit prendre en compte :
    - Par exemple :
      - Des contraintes d’intégration
      - Des contraintes de déploiement.
  - De l’énoncé de caractéristique impératives de :
    - Performance,
    - Disponibilité
    - Coût de la solution
    - Par exemple :
      - Application disponible 7/7 24/24

Identification des besoins utilisateur

- Identifier les acteurs
- Identifier les principaux services rendus par l’application aux acteurs.
- Identifier les principales interactions entre l’application et les acteurs.

Identification des principaux artefacts

- Les classes :
  - Boundary
  - Entité
  - Contrôleur
- Les collaborations des classes.
- Les responsabilités des classes.

Formalisation du recueil des besoins

1. Identifier les acteurs
2. Identifier principaux services rendus par l'application
3. Identifier les cas d'utilisation
4. Trier les acteurs
5. Trier les cas d'utilisation
6. Produire :
    1. Les interactions
    2. Les scénarios.
7. Raffiner les cas d'utilisation.

Conseils

- A ce niveau ne sont considérés que :
  - Les acteurs
  - Les cas d’utilisation
  - Les interaction entre l’utilisateur et l’application
- Le « travail » réalisé par l’application pour réaliser les fonctions n’est pas abordé.

Qu’est ce qu’un acteur?

- C’est une entité qui interagit avec l’application :
  - Un utilisateur « humain »
  - Un système qui fournit des services
  - Un système qui utilise les services de l’application.
- Un acteur est l’abstraction d’un rôle.
  - Une personne peut jouer 2 rôles différents
  - Deux personnes peuvent avoir le même rôle.
- Un acteur est à l’extérieur de l’application
- Il existe 2 types d’acteurs :
  - Primaires
  - Secondaires

Qu’est ce qu’un acteur?

- Un acteur primaire :
  - Est le destinataire « privilégié » de l’application
  - Il est actif en ce sens que c’est lui qui initie les interactions.
- Un acteur secondaire :
  - Un acteur pour lequel l’application n’offre qu’un service marginal.
  - Il est passif en ce sens qu’il ne fait que répondre à des sollicitations du système.

Qu’est ce qu’un cas d’utilisation ?

- Selon Jacobson :
  - Un cas d’utilisation est une séquence de transactions avec le système dont le but est de fournir un résultat dont la valeur est mesurable par un acteur qui utilise le système.
- Une séquence de transactions :
  - Une séries d’interactions (échanges) entre un acteur et l’application .

Qu’est ce qu’un cas d’utilisation ?

- Un résultat dont la valeur est mesurable :
  - Un objectif dont la valeur est non triviale pour l’utilisateur.

Qu’est ce qu’un cas d’utilisation?

- Le processus de définition des cas d’utilisation ne met l’accent que sur ce que doit faire l’application pour satisfaire les besoins de l’utilisateur.
- Les choses sont vues à travers les yeux de l’utilisateur.
- On ne développe pas de cas d’utilisation pour les comportements marginaux.

Les limitations des cas d’utilisation

- Les cas d’utilisation ne capturent pas :
  - Les contraintes de performance
  - Les contraintes de fiabilité
  - Les règles de gestion
  - Les formules et les calculs
  - Les séquences d’utilisation des cas d’utilisation, de scénarios.
  - L’ergonomie de l’application.
  - Les contraintes légales applicables.

Les limitations des cas d’utilisation

- Les cas d’utilisation ne modélisent pas les interactions entre cas d’utilisation.
- Ils ne manifestent pas les conflits et incompatibilités entre cas d’utilisations.
- Ils ne peuvent exprimer aucune forme de concurrence.

Critères de validité d’un cas d’utilisation

- Il est lié à un acteur et un seul.
- Il se traduit par des échanges avec l’acteur.
- Il rend un service complet à l’acteur.

Relations entre les acteurs.

- Héritage :
  - Une spécialisation d’un acteur réalise les interactions de sa « super classe »
  - Il définit des interactions avec d’autres cas d’utilisation.

Relation entre les cas d’utilisation

- Spécialisation
- « Includes »
- « Extends »

Spécialisation de cas d’utilisation

- Il peut être envisagé lorsqu’un cas d’utilisation, pour une même situation de départ et un même service rendu se traduit par des interactions significativement différentes.
- Dans ce cas :
  - Le cas d’utilisation « principal » est abstrait.
  - Ses spécialisations traitent chacune un groupe d’interactions distinctes.

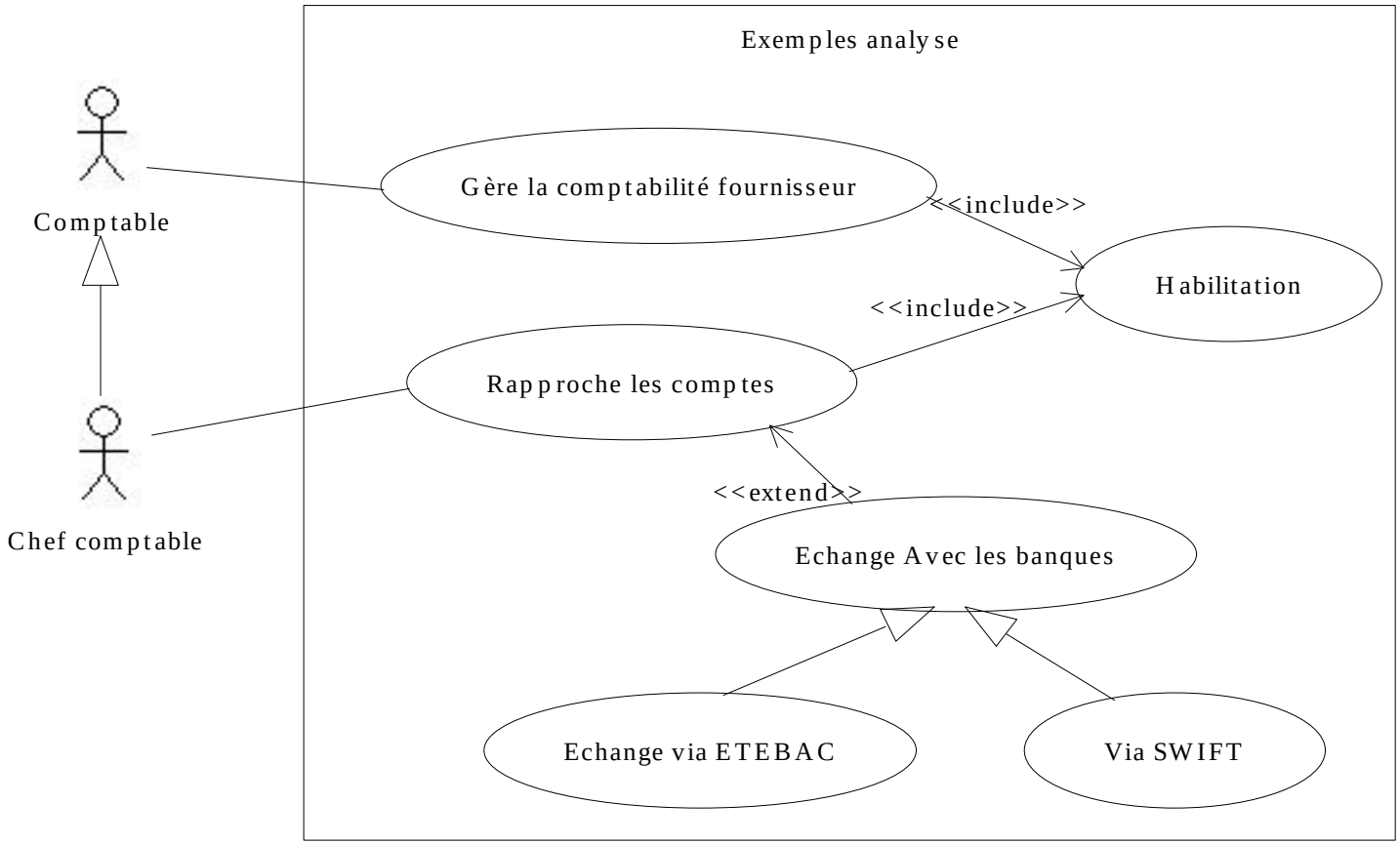
Includes

- C’est une relation entre 2 cas d’utilisations qui ne doivent pas être liés par un lien d’héritage.
- Il signifie que l’exécution de celui qui inclut l’autre passe obligatoirement par celui qui est inclus.

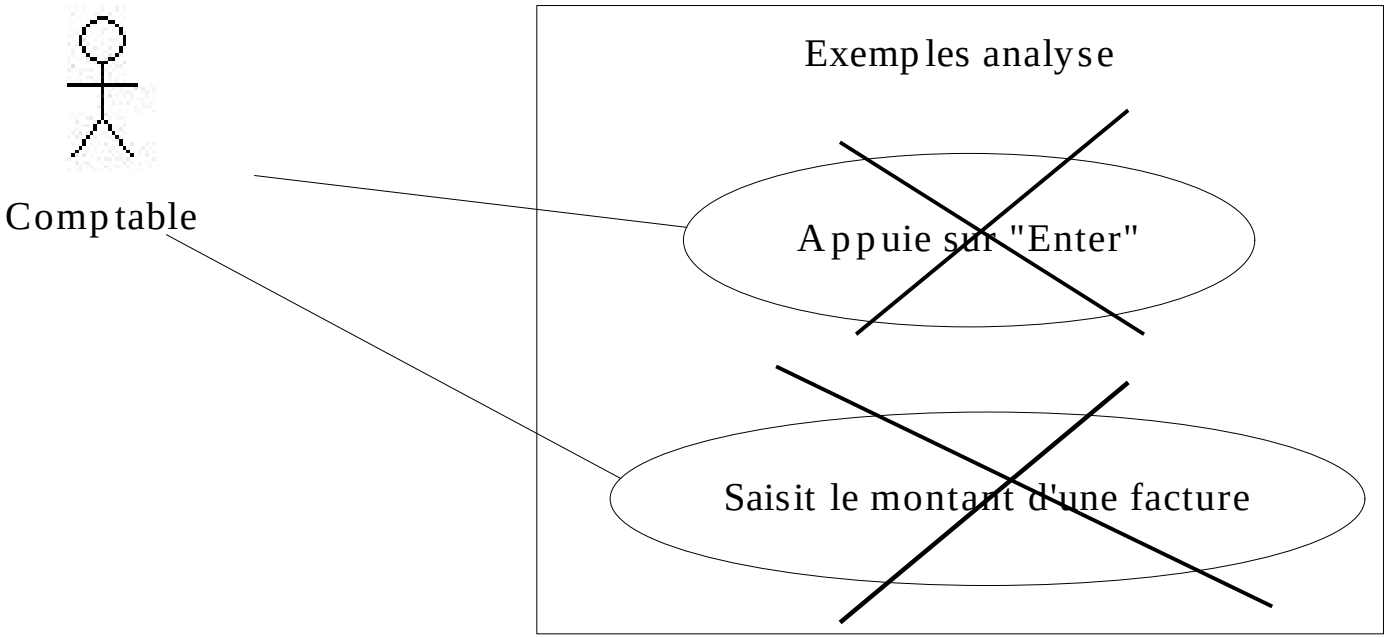
Extends

- Cela signifie que le cas d’utilisation qui «étend » l’autre peut optionnellement être utilisé .
- C’est une relation entre 2 cas d’utilisations qui ne doivent pas être liés par un lien d’héritage.

Exemple


Exemples analyse
Comptable
Chef comptable
Gère la comptabilité fournisseur
Habilitation
Rapproche les comptes
Echange Avec les banques
Echange via ETEBAC
Via SWIFT
<<include>>
<<include>>
<<extend>>

Contre exemple



Utilisation de scénarios

- Les scénarios ne sont pas des modèles normalisés par UML. Ce sont des descriptions textuelles rédigées à l'aide d'un traitement de texte.
- Ils sont par exemple le résultat d'entretiens avec la MOA ou de l'analyse des documents de spécifications.
- Ils complètent les cas d'utilisation et les diagrammes de séquence.
- Leur utilisation n'est pas obligatoire, toutefois elles est conseillée.

Structure type d’un scénario

- Sommaire, identification
- Pré conditions
- Enchaînement nominal
- Enchaînements alternatifs
- Exceptions
- Post conditions

Sommaire, identification

- Référence au cas d’utilisation
- But
- Résumé
- Acteurs
- Trace des évolutions :
- Date de création
- Date de modification,
- Auteur
- Version…

Pré conditions

- Ensemble des conditions qui doivent être vérifiées pour le le cas d’utilisation puisse être exécuté.

Enchaînement nominal

- Représente la suite d’actions utilisateur typiques qui réalise le cas d’utilisation.
- Il se matérialise par une succession d’étapes qui précisent :
  - Les actions de l’acteur
  - Les réponses du système.

Enchaînements alternatifs

- La structure est la même que pour l’enchaînement nominal.
- Toutefois ces enchaînements précisent des étapes qui sont parfois utiles à la réalisation du cas d’utilisation.
  - Penser aux « Extends »

Exceptions

- Tout échange entre l’acteur et l’application peut se solder par une impossibilité ou une erreur.
- Ces conditions sont des «exceptions ».
- Chaque étape de chaque scénario précise les exceptions possibles.
- Cette section détaille, pour chacune les actions que doit prendre l’application, les conséquences sur le déroulement du scénario.

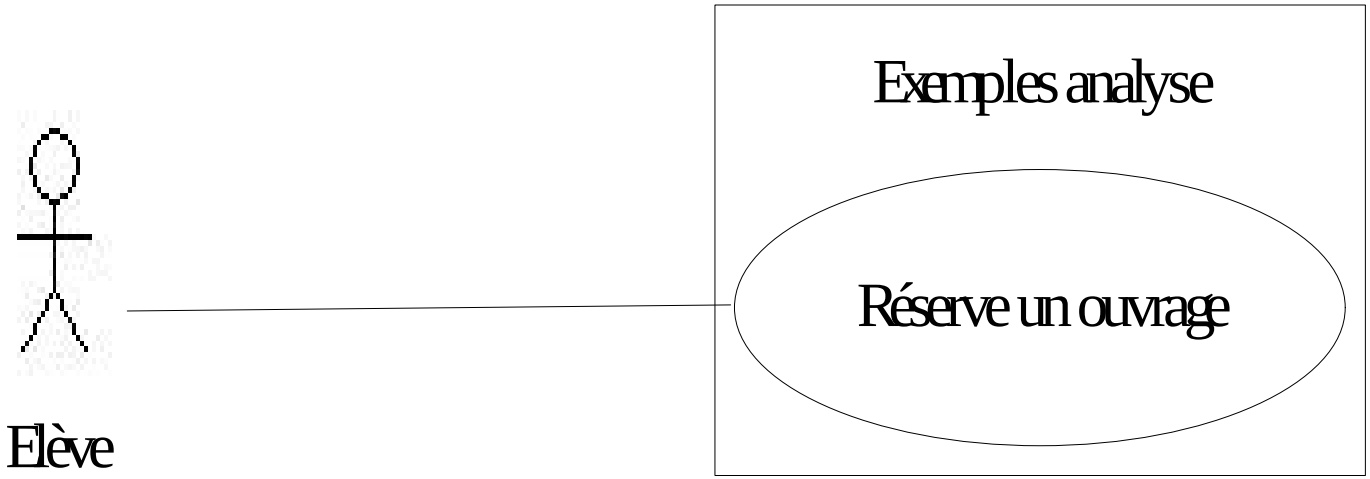
Post Conditions

- Elle détaille l’état de l’application à l’issu de l’exécution du scénario.

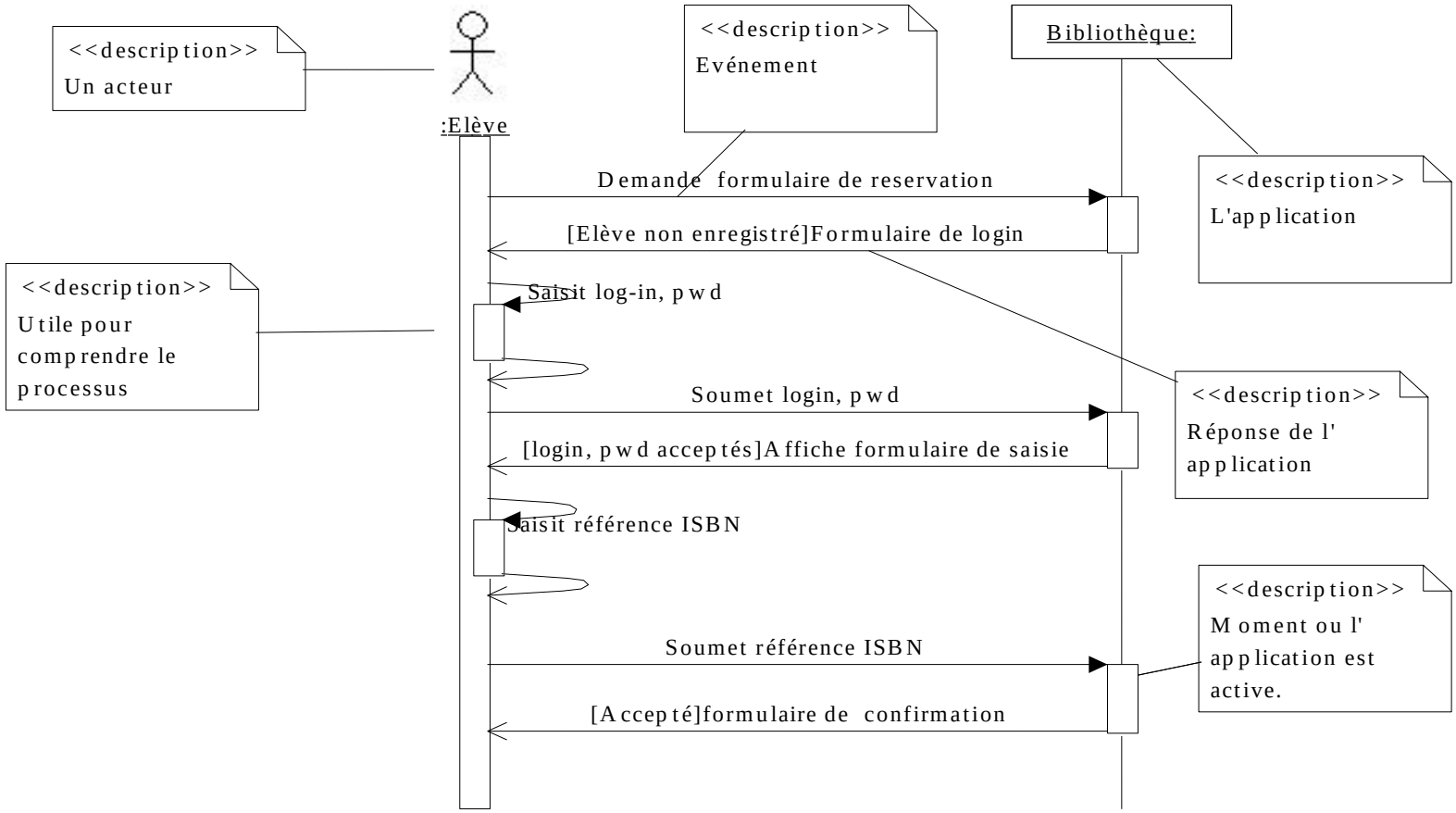
Les diagrammes d’interaction

- Ils mettent en évidence les échanges entre l’acteur d’un cas d’utilisation et le système.
- En particulier sont exprimés :
  - Les événements
  - Les réponses
  - Les gardes.
- Dans le cas de le formalisation du recueil des besoins leur utilisation reste informelle.
- Ils synthétisent le contenu des scénarios.

Cas d’utilisation identifié



Exemple de diagramme



A l’issue de la formalisation des besoins

- La totalité des cas d’utilisation
  - Leurs relations
  - Les acteurs concernés
- Les scénarios de chaque cas d’utilisation
- Les diagrammes d’interactions

Les objectifs de l’analyse

- Formalisation des besoins utilisateurs
- Identification des principaux artefacts de l’application (analyse proprement dite)

Les diagrammes utilisés

- Les diagrammes d’activité
- Les diagrammes de séquence
- Les diagrammes de classe « BEC »