

UML

Diagramme de cas d'utilisation
(Use Case Diagram)

Emmanuel Pichon

2013

Introduction au diagramme de cas d'utilisation

- ⊙ Technique de recueil des besoins sous forme textuelle
- ⊙ Besoin = comportement attendu d'un système du point de vue de son utilisateur (humain ou système)
- ⊙ Les besoins servent de référence pour l'ensemble des phases de fabrication logicielle
- ⊙ UML ne propose rien pour les exigences non fonctionnels
 - besoins techniques : performance, sécurité, disponibilité, ...
 - besoins projet : délais, lotissements, ...

Présentation du diagramme de cas d'utilisation (*use case diagram*)

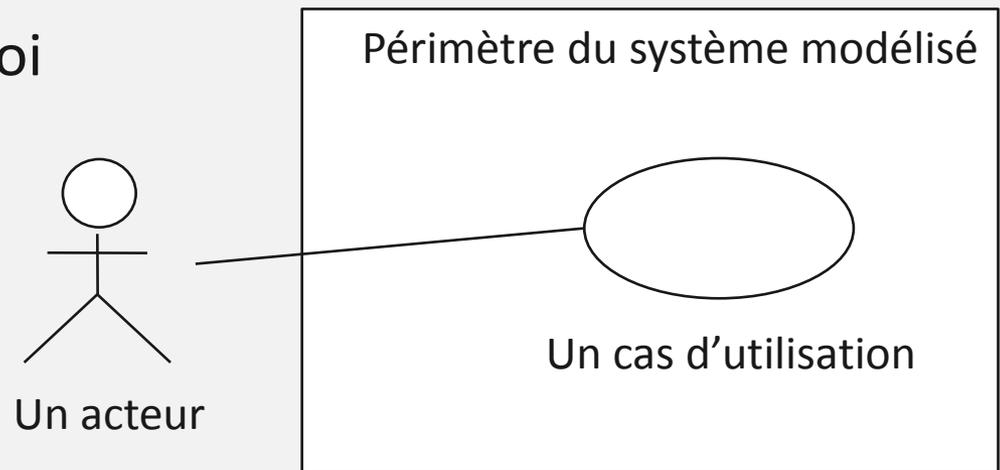
◎ Usage

- Définir un périmètre du recueil des besoins et du système à mettre en œuvre

◎ Signification

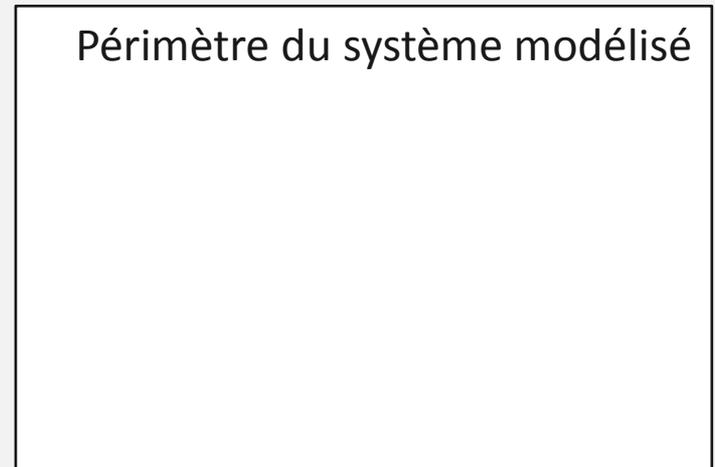
- Qui a besoin de faire quoi avec le système ?

◎ Notation UML



Périmètre du système modélisé (*boundary*)

- ◎ Usage
 - Identifier le système à modéliser et son périmètre
- ◎ Signification
 - Frontière entre le système à modéliser et l'extérieur
- ◎ **NB : toute la modélisation dépend de ce périmètre**
- ◎ Notation UML (optionnelle)
 - Un rectangle contenant le nom du système modélisé

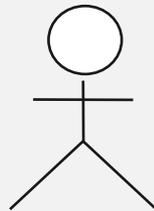


Acteur (*actor*)

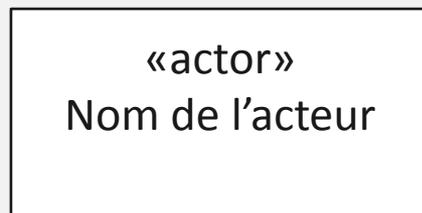
◎ Signification

- Entité (humain ou système) externe au système qui interagit avec celui-ci

◎ Notations UML



Nom de l'acteur



◎ Identification

- Rôle lié à l'utilisation du système
- (les acteurs humains ne reflètent pas l'organisation de l'entreprise)

◎ Nommage

- Nom court ayant un sens métier

◎ Documentation

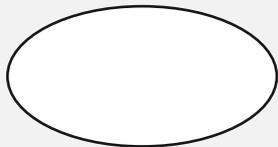
- Description du rôle de l'acteur par rapport au périmètre du système

Cas d'utilisation (*use case*)

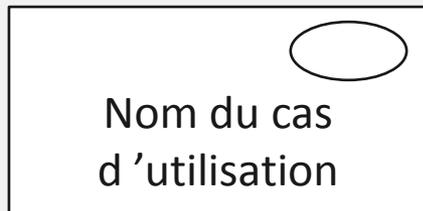
◎ Signification

- Un comportement attendu d'un système du point de vue d'un acteur

◎ Notations UML



Nom du cas d'utilisation



◎ Identification

- Finalité métier pour laquelle un acteur utilise le système
- Ce n'est pas une fonction

◎ Nommage

- Phrase courte commençant par un verbe à l'infinitif (éviter le verbe « gérer »)

◎ Documentation (cf. page suivante)

Cas d'utilisation – documentation (1/3)

⦿ Usage

- Définir de manière textuelle le comportement du système attendu et perçu par les acteurs

⦿ Signification principale

- Description des interactions entre les acteurs et le système
- Suivant 1 ou n scénarios

⦿ NB

- Ne pas décrire le fonctionnement interne du système (principe d'encapsulation appliqué au niveau du système)
- Description centrée sur les acteurs et leurs buts
- Plusieurs niveaux de détails sont possibles

Cas d'utilisation – documentation (2/3)

- ⊙ Exemple d'organisation de la documentation (assez complet)
 - Pré conditions
 - Événement déclencheur
 - Interactions entre les acteurs et le système
 - Résultat final
 - Post conditions
 - Variantes
 - Scénarios optionnels
 - Situations exceptionnelles
- ⊙ NB : la manière de documenter n'est pas imposée par UML

Cas d'utilisation – documentation (3/3)

- ⊙ Exemple de description typique des interactions
 - Un acteur réalise une action sur le système
 - Le système réalise une action en réponse (visible par un acteur)
 - (et ainsi de suite)
- ⊙ Conseils de rédaction
 - Rédiger au temps présent dans un style simple
 - Eviter deux actions de suite pour le même acteur ou le système
- ⊙ NB : la description des interactions justifie l'identification des cas d'utilisation et de leurs relations
 - Il faut donc documenter les cas d'utilisation avant de construire le diagramme de cas d'utilisation (qui n'est qu'une synthèse)

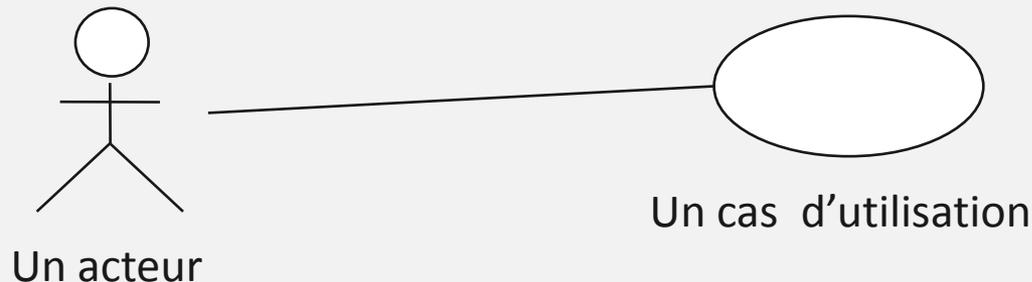
Relation acteur – cas d'utilisation (*association*)

◎ Signification

- Un acteur est en interaction avec le système dans le cadre d'un cas d'utilisation

◎ Notation UML

- Un trait simple entre un acteur et un cas d'utilisation



- On peut distinguer les acteurs principaux, secondaires, ... à l'aide d'un stéréotype au niveau de l'association
Exemple : «principal»

Illustration (source OMG)

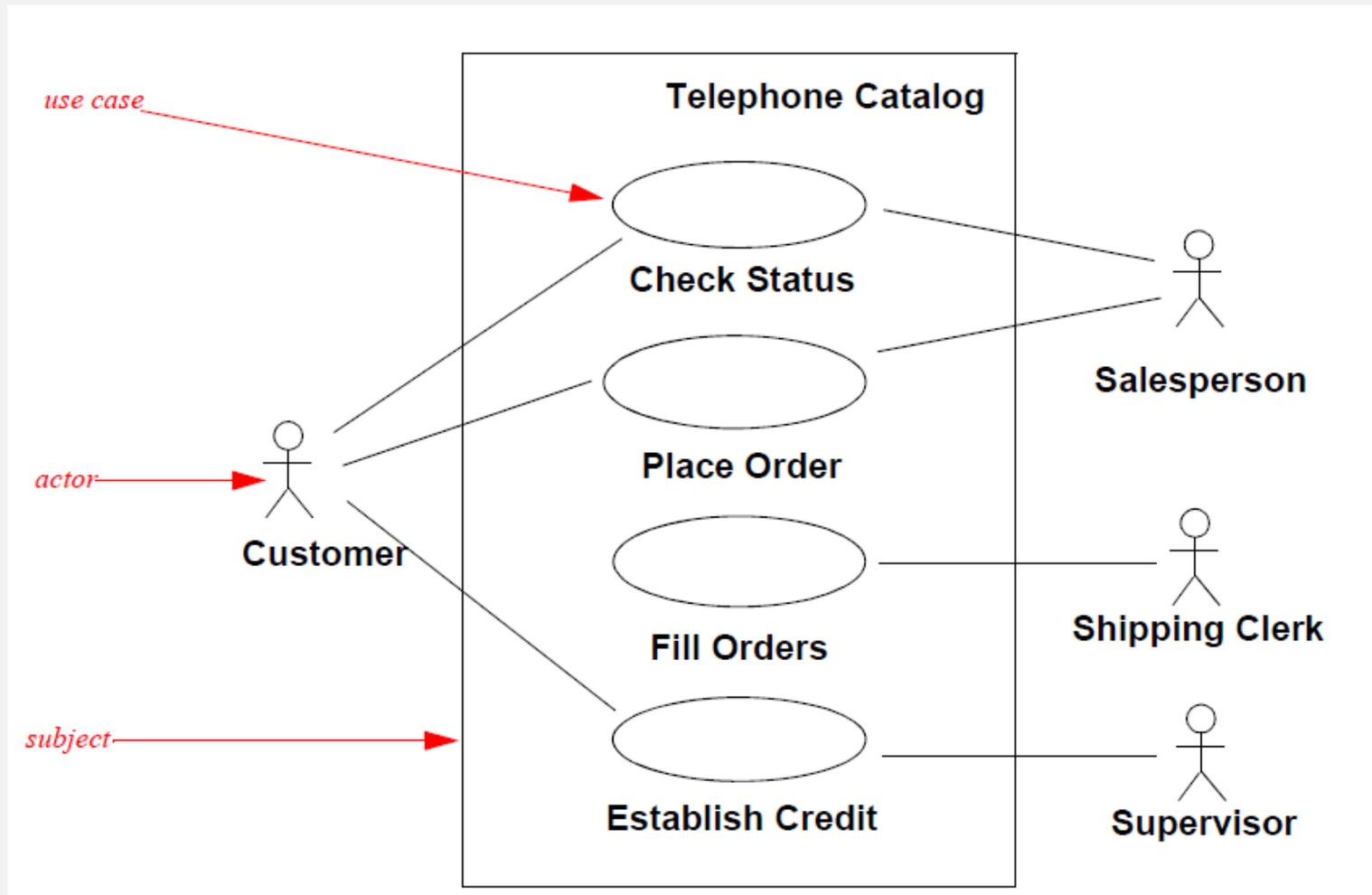


Diagramme de cas d'utilisation

Synthèse n°1

- ⦿ Le diagramme de cas d'utilisation
 - Répond à la question « Qui veut faire quoi avec le système ? »
 - N'est qu'un sommaire / une synthèse de la documentation
- ⦿ Certains projets n'utilisent que la partie textuelle des cas d'utilisation (pas de diagramme)
- ⦿ Éviter trop de relations dans les diagrammes de cas d'utilisation
 - En général, les relations d'association entre acteur et cas suffisent
 - Les autres relations sont à utiliser avec précaution / parcimonie (cf. pages suivantes)

Relation de généralisation entre acteurs (*generalization*)

◎ Usage

- Factorisation des besoins communs
- Séparation des besoins spécifiques

◎ Signification

- Un acteur A a les mêmes besoins qu'un autre acteur B
- L'acteur A peut avoir d'autres besoins supplémentaires

◎ Notation UML

- Une flèche pleine



Relation de généralisation entre cas d'utilisation (*generalization*)

◉ Usage

- Spécialiser ou généraliser un besoin

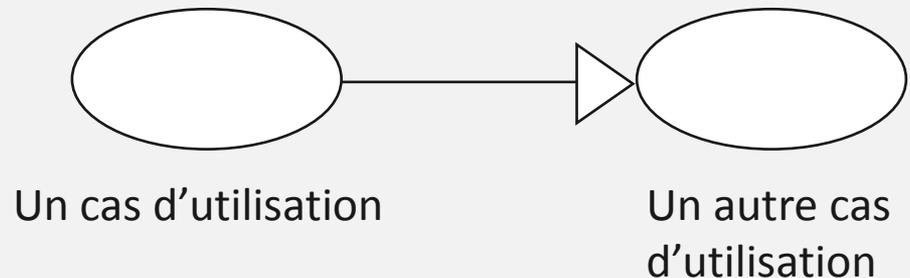
◉ Signification

- Un besoin est un cas particulier d'un autre besoin

- ◉ NB : il est possible de définir des variantes dans un cas d'utilisation sans utiliser cette relation

◉ Notation UML

- Une flèche pleine



Relation d'inclusion de cas d'utilisation (*include*)

◎ Usage

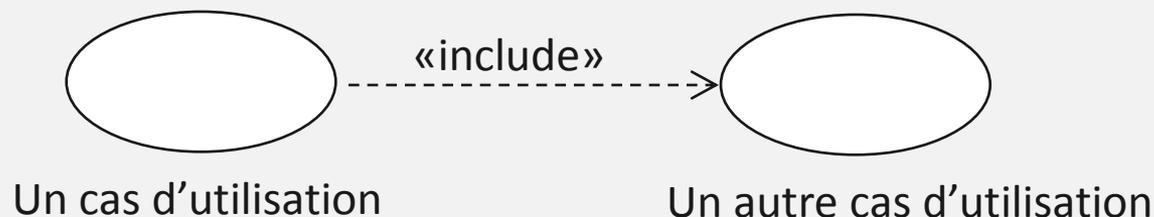
- Factoriser une partie des besoins
- Décomposer un cas d'utilisation trop complexe
 - NB : le diagramme d'activités est également utilisable

◎ Signification

- Un besoin inclut un autre besoin ou une partie du comportement du système

◎ Notation UML

- Une flèche en pointillé stéréotypée «include»



Relation d'extension de cas d'utilisation (*extend*)

◎ Usage

- Isoler des besoins supplémentaires (conditionnels)

◎ Signification

- Un besoin étend un autre besoin (il l'enrichit)

◎ Notation UML

- Une flèche en pointillé stéréotypée «extend»
- Les conditions peuvent être notées et liées à la flèche
- Il est possible d'indiquer des points d'extension

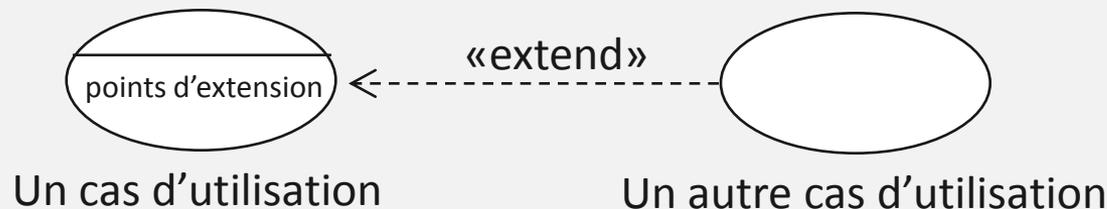


Illustration (source OMG)

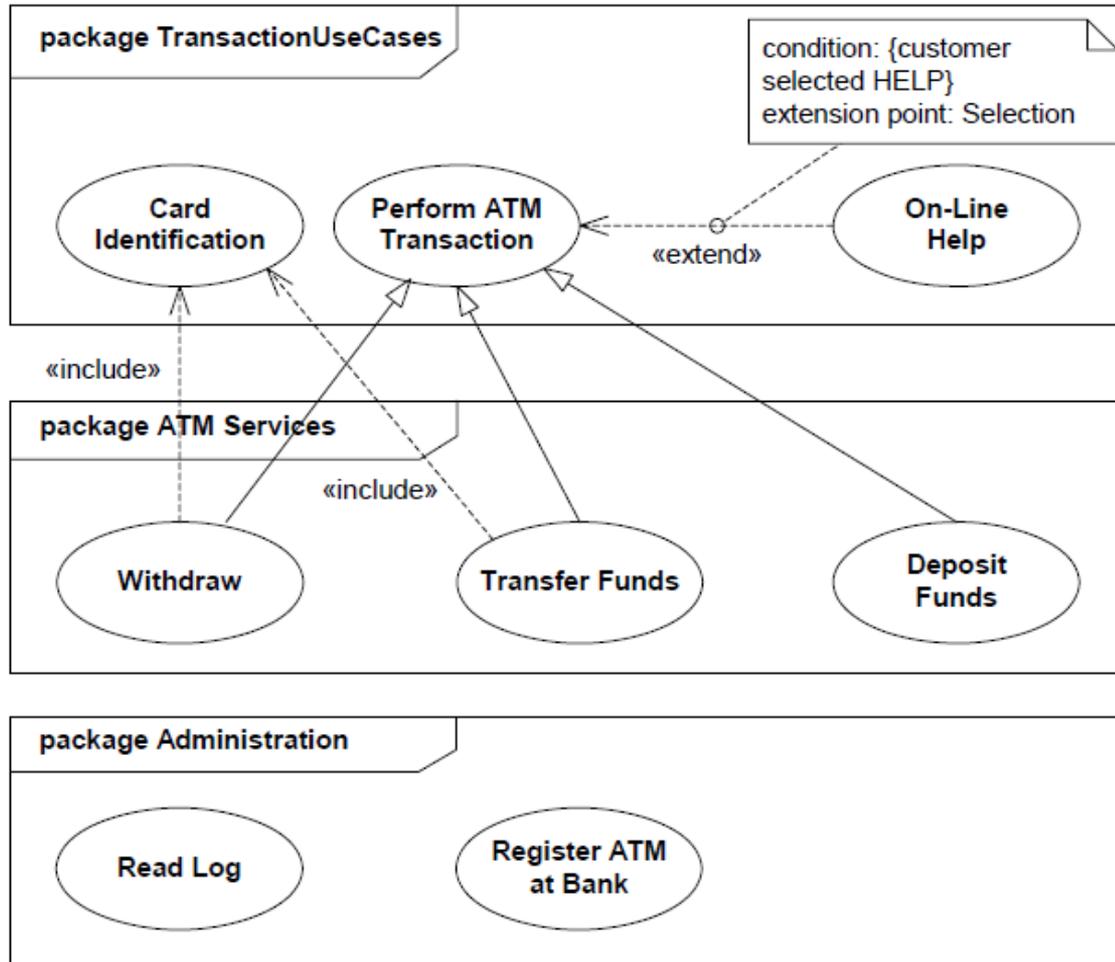


Diagramme de cas d'utilisation

Synthèse n°2

- ◎ Lien avec les autres activités de fabrication des logiciels
 - Alimenter l'analyse des besoins et la conception du système
 - Alimenter les tests afin de vérifier que le système réalisé répond aux besoins exprimés
- ◎ L'usage des relations (hors association) est souvent confondu avec une décomposition fonctionnelle
 - Le recueil des besoins n'identifie pas des fonctions
 - UML ne préconise pas la décomposition fonctionnelle

Conseils de mise en œuvre

- ⦿ Identifier le système ou le périmètre à modéliser
- ⦿ Identifier et documenter les acteurs
- ⦿ Lister les cas d'utilisation à décrire
- ⦿ Décrire les cas d'utilisation (cf. détail page suivante)
- ⦿ Construire le diagramme de cas d'utilisation

Description du déroulement des cas d'utilisation

- ⦿ Limiter la description au contenu utile pour la suite du projet
 - notamment les tests d'intégration externe et de recette
- ⦿ Veiller à ne pas décrire deux fois une partie de scénario
- ⦿ Mettre à jour la liste des cas d'utilisation à décrire
 - Supprimer les cas d'utilisation en trop (si description en double)
 - Ajouter des cas d'utilisation pour factoriser un besoin commun à plusieurs cas d'utilisation (relation «include»)
 - Identifier les éventuelles relations «extend» si un cas d'utilisation enrichit un autre cas d'utilisation