

# Analyse et Conception des Systèmes

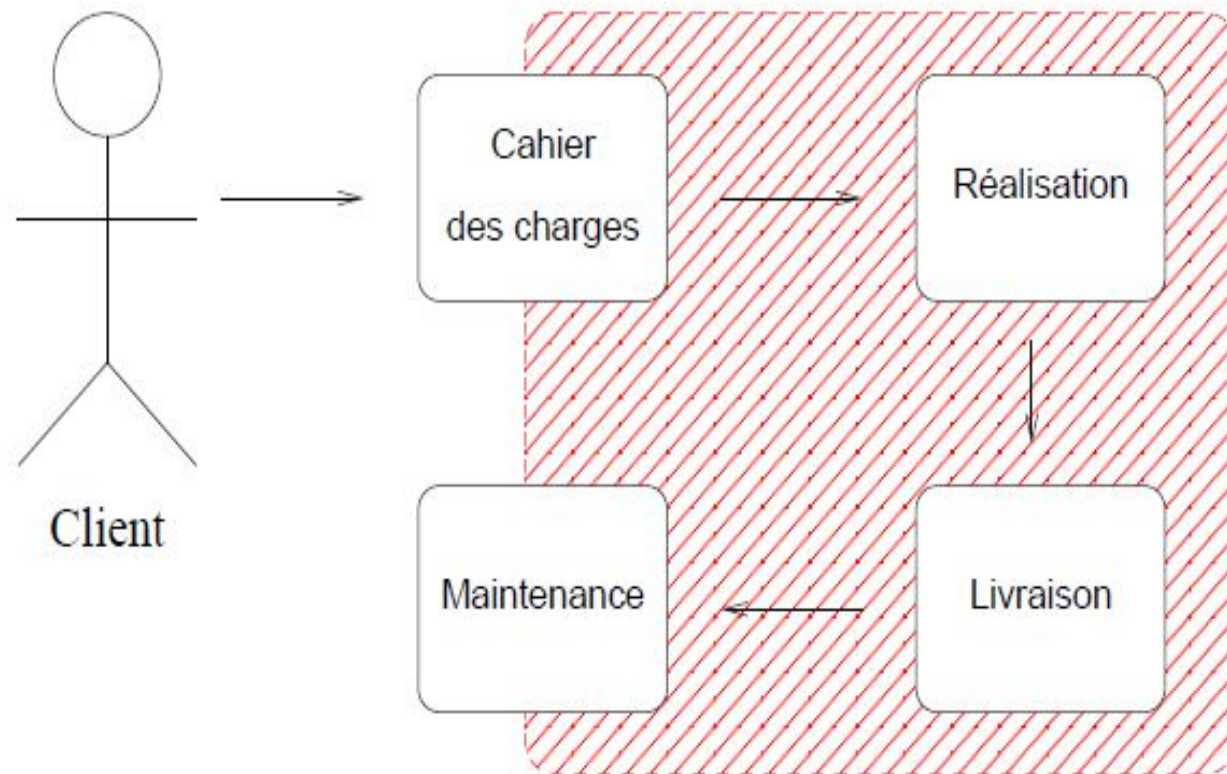


# Introduction



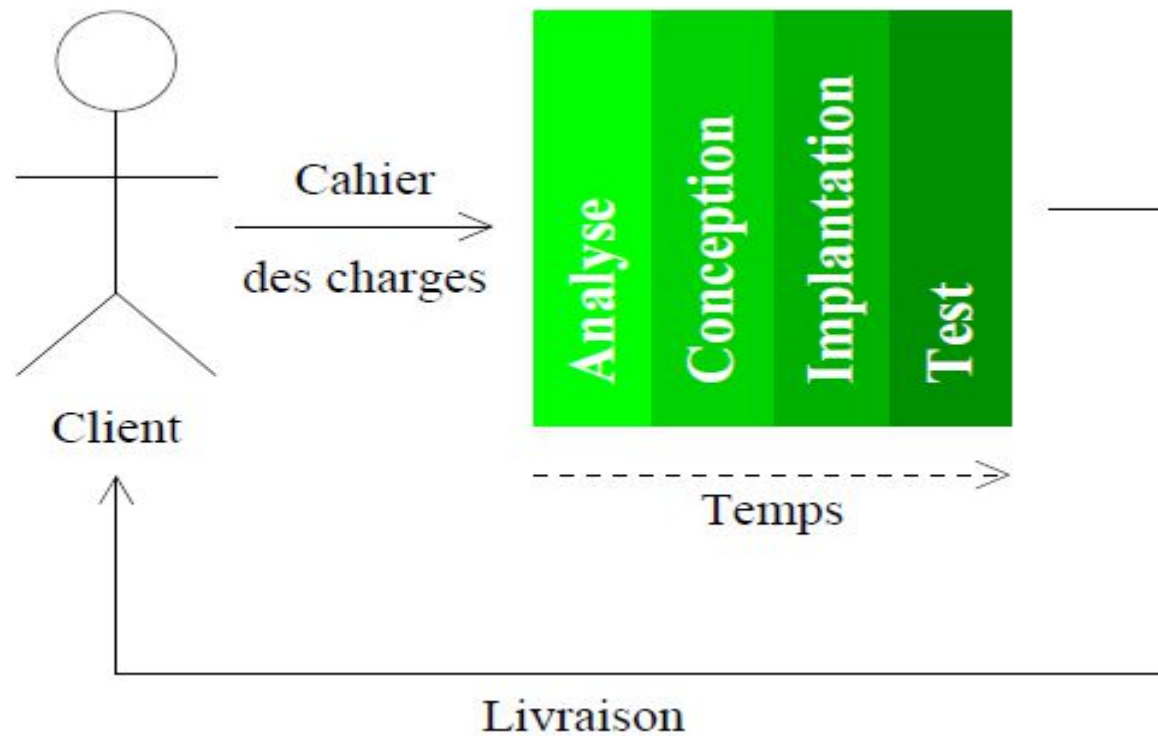
# Processus de développement logiciel

---



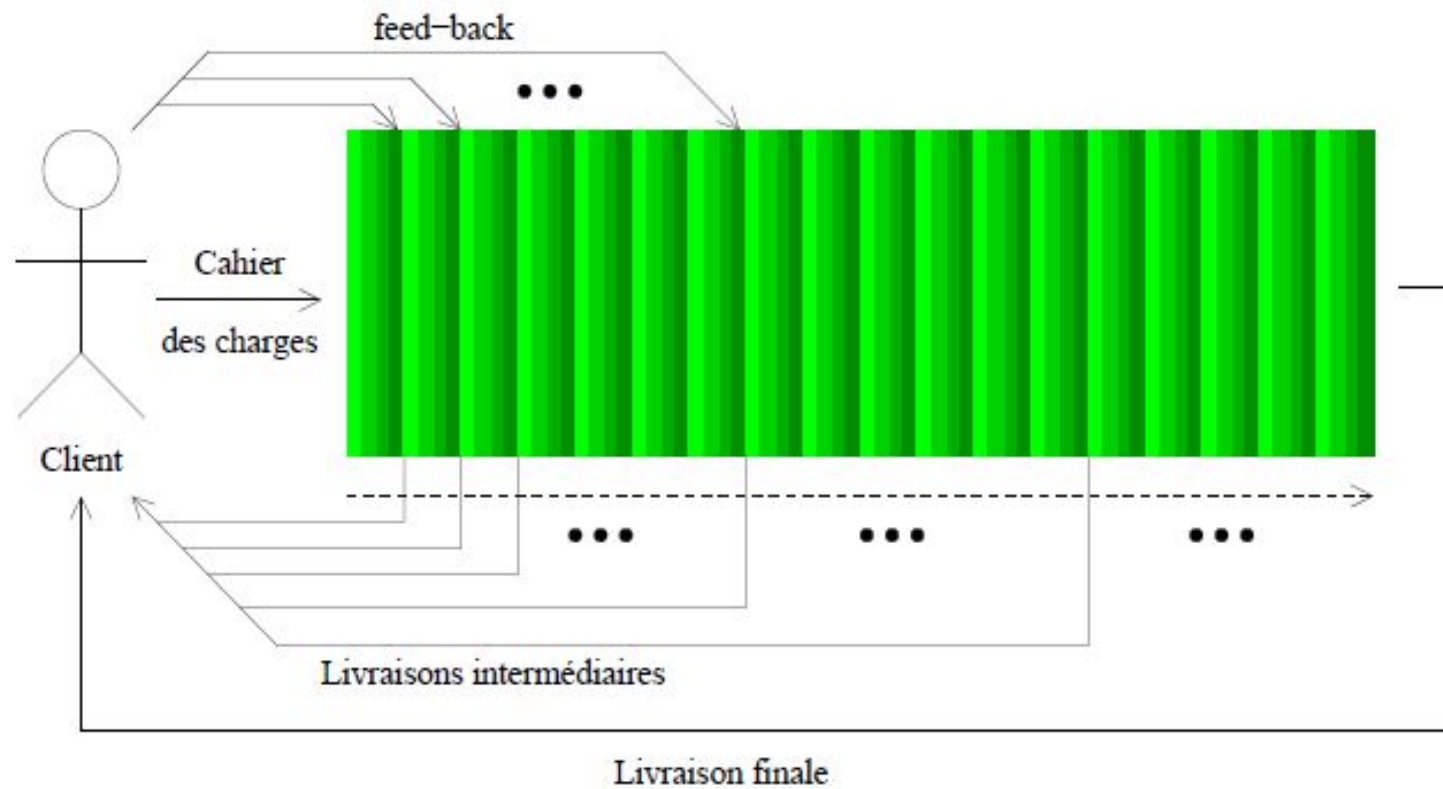
# Développement en cascade

---



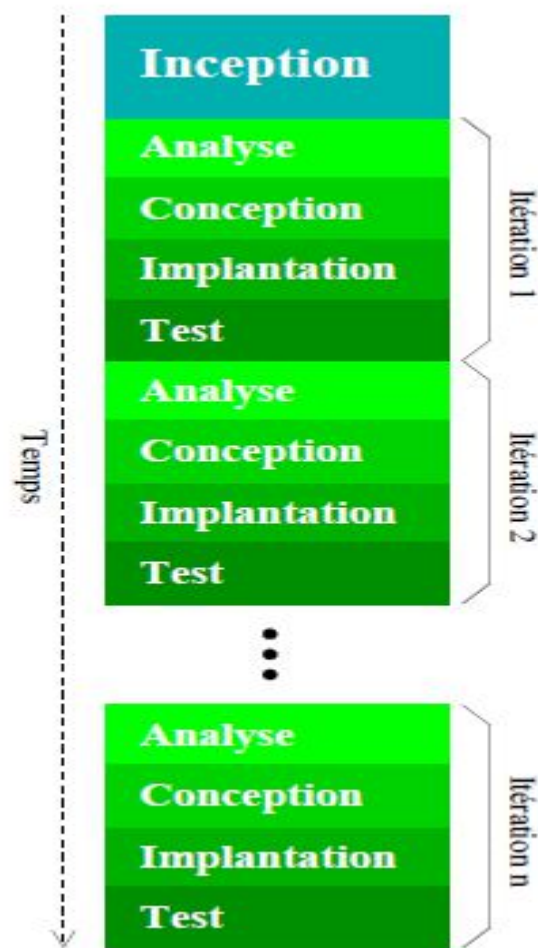
# Développement itératif

---

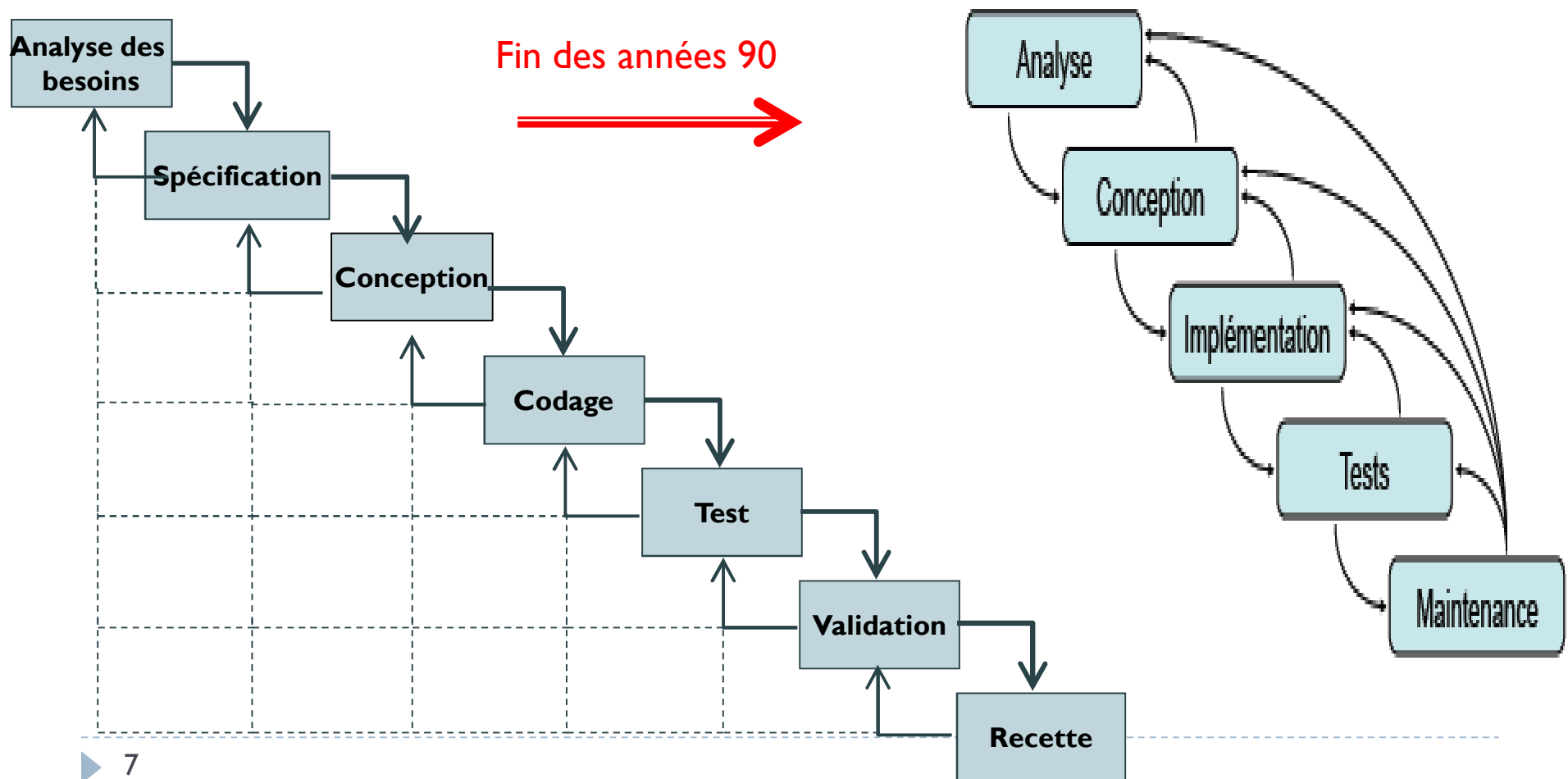


# Développement itératif

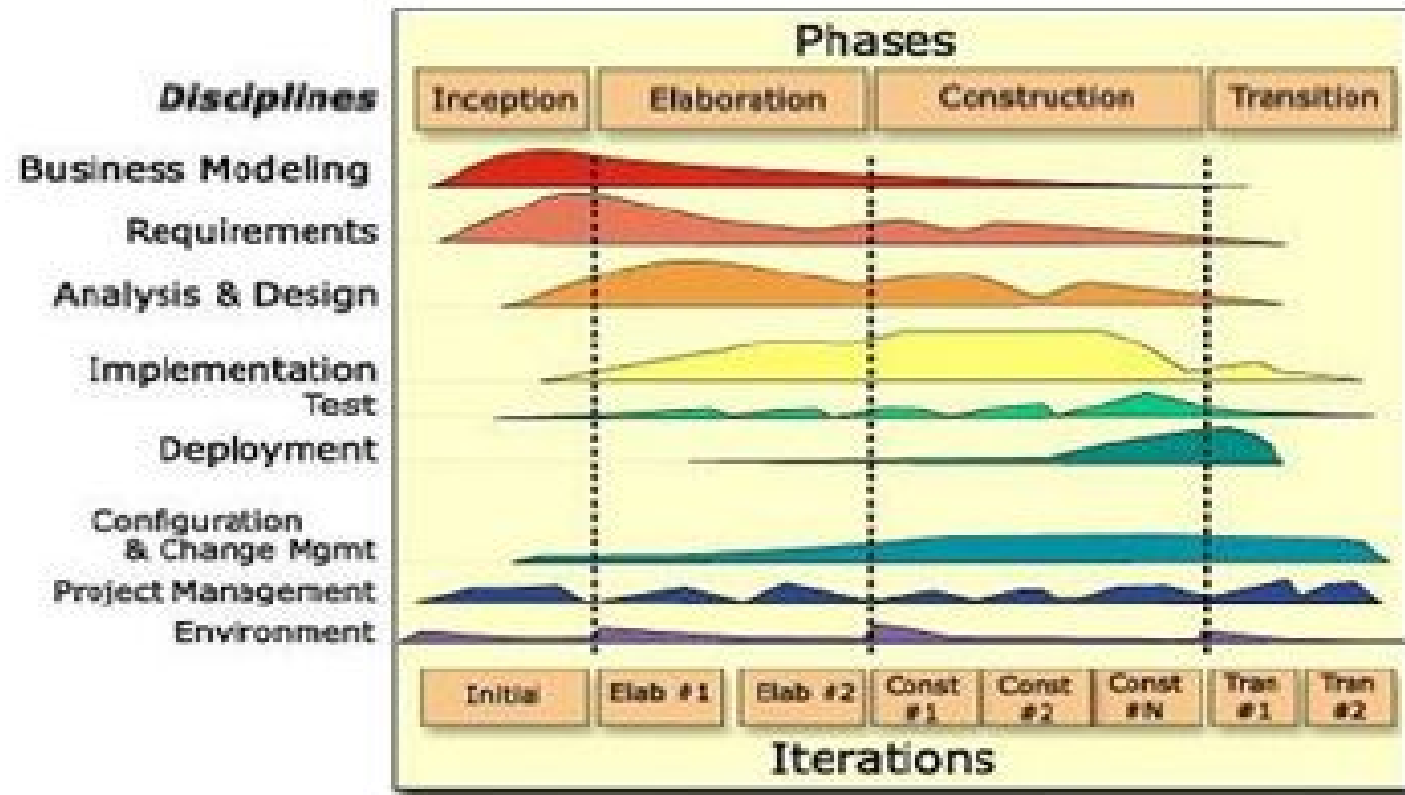
---



# Modèle de développement en cascade



# Processus unifié





# Les activités du projet

---

Elles sont identiques quelque soit le modèle:  
Cascade, incrémental....

# Analyse et Conception

---

## Objectifs de l'analyse :

- Étude du métier du client
- Étude des besoins des utilisateurs
- Reformulation du cahier des charges sous une forme exploitable en conception

## Objectifs de la conception :

- Définition de l'architecture logicielle
- Définition du comportement de l'application

# Analyse et Conception

---

- ▶ Analyse:

- ▶ Processus consistant à se familiariser avec le domaine dans lequel le système sera intégré: Contexte, Utilisateurs, Contraintes, Coûts, performance, etc.

- ▶ Conception:

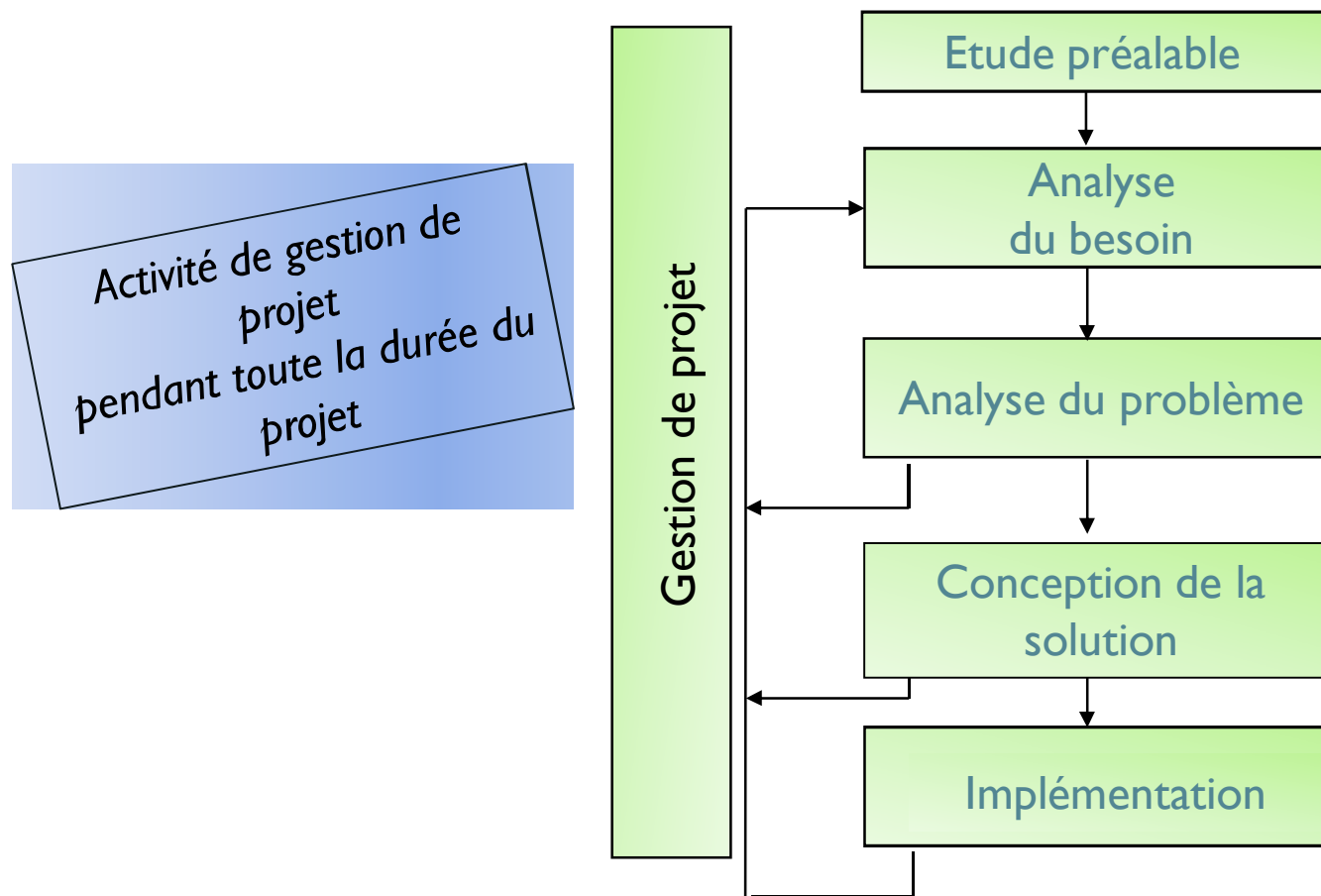
- ▶ Processus par lequel diverses techniques et principes sont appliqués dans le but de définir un système avec un niveau de détail suffisant pour permettre sa réalisation physique.



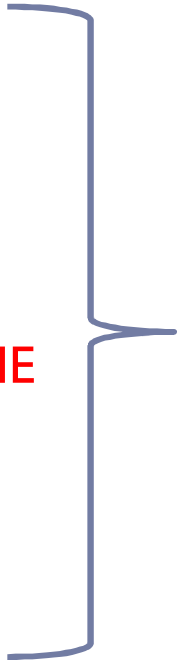
## Activités du projet

# Les activités du projet

---



# Les activités du projet

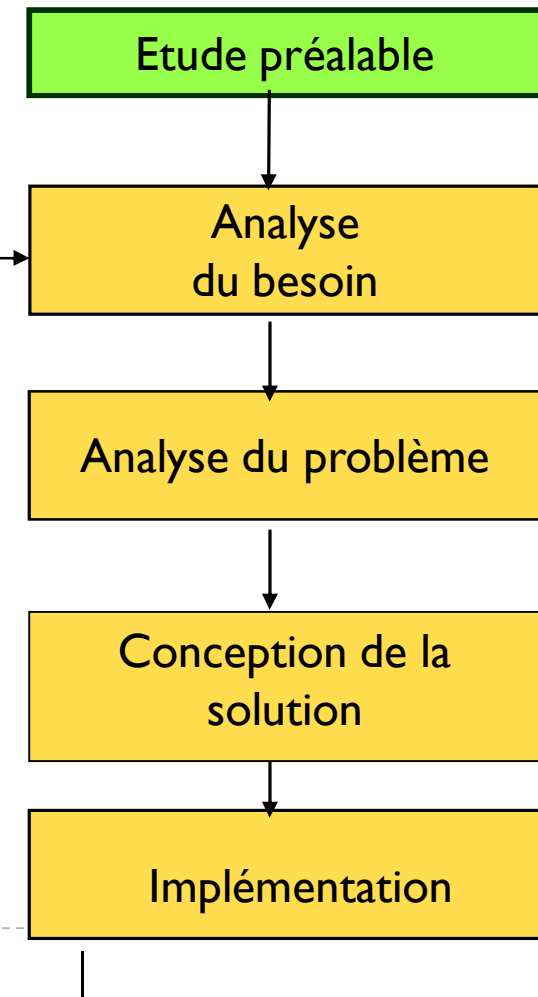
- ▶ **ETUDE PRÉALABLE : ANALYSE DU CONTEXTE**
  - ▶ **ANALYSE ET SPÉCIFICATION DES BESOINS**  
Fonctionnalités du système d'information
  - ▶ **ANALYSE DU PROBLÈME OU MÉTIER OU SYSTÈME**
    - ▶ Étude de la logique du système informatique  
(= Indépendamment des technologies)
    - ▶ Modélisation métier (vue logique)
  - ▶ **CONCEPTION DE LA SOLUTION**
    - ▶ Décisions technologiques
    - ▶ Affinement de la vue logique
    - ▶ Implémentation (Algorithmes, diagramme de composants)
    - ▶ Déploiement (Diagramme de déploiement)
- 
- Analyse**

# Etude préalable

On souhaite construire un système informatique pour **répondre à un besoin**

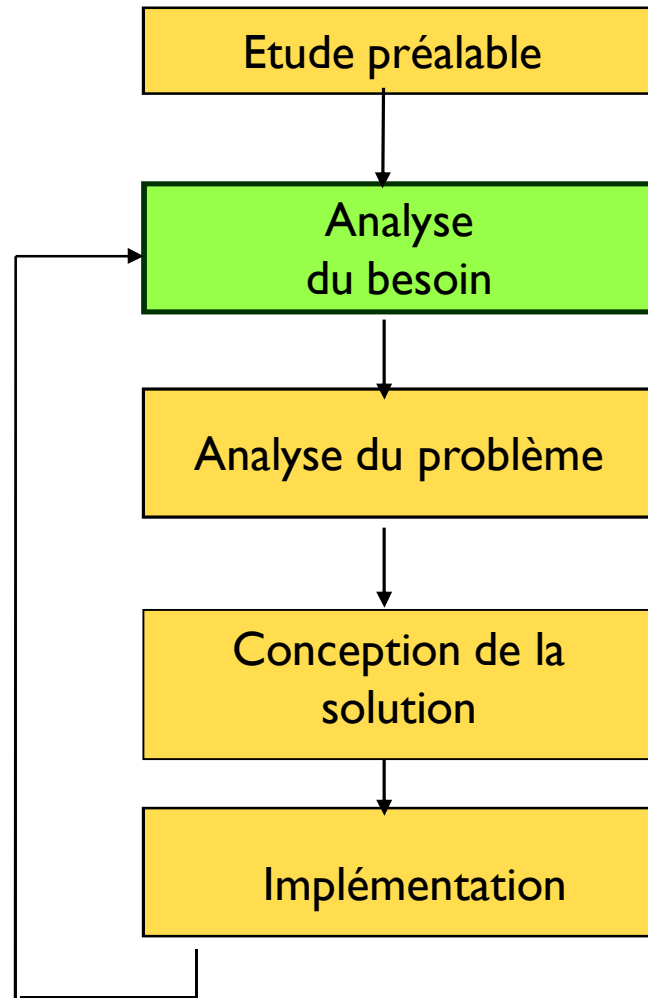
- ▶ Qui a ce besoin?
- ▶ Quel est ce besoin (à un niveau stratégique)?
- ▶ **Est-il justifié?**
- ▶ Activités principales du contexte étudié
- ▶ Bilan **gains-coûts** estimés

On modélise **le périmètre** du projet et son **contexte** : **Analyse du contexte**



# Analyse des besoins

---



Exprimer **les fonctionnalités demandées** (selon la vision du client)

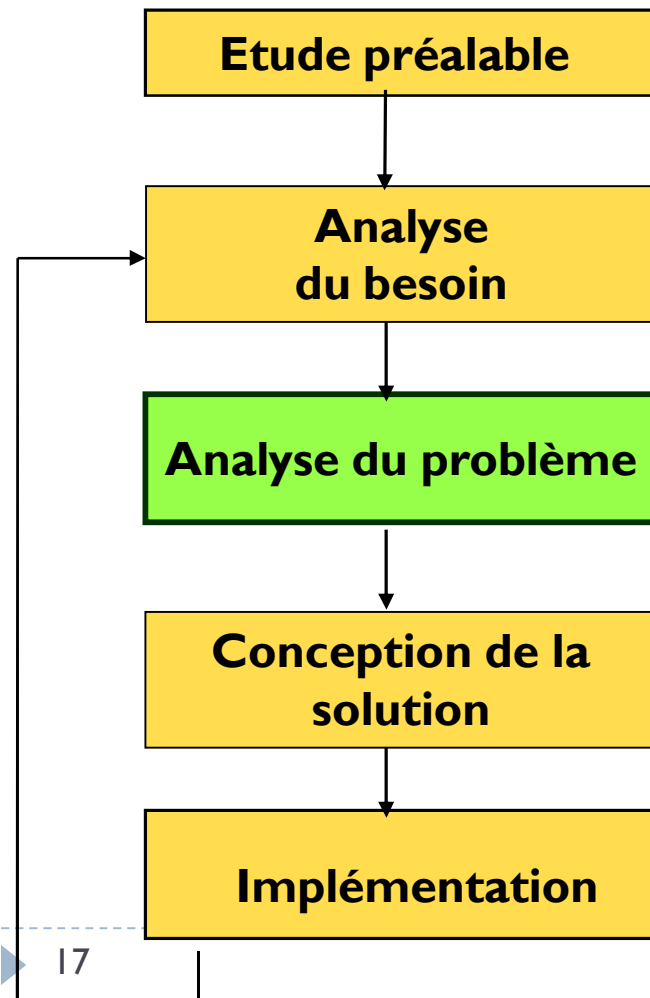
+

autres besoins non fonctionnels BNF  
(**performance, sécurité, flexibilité...**)



# Analyse du problème ou métier ou système

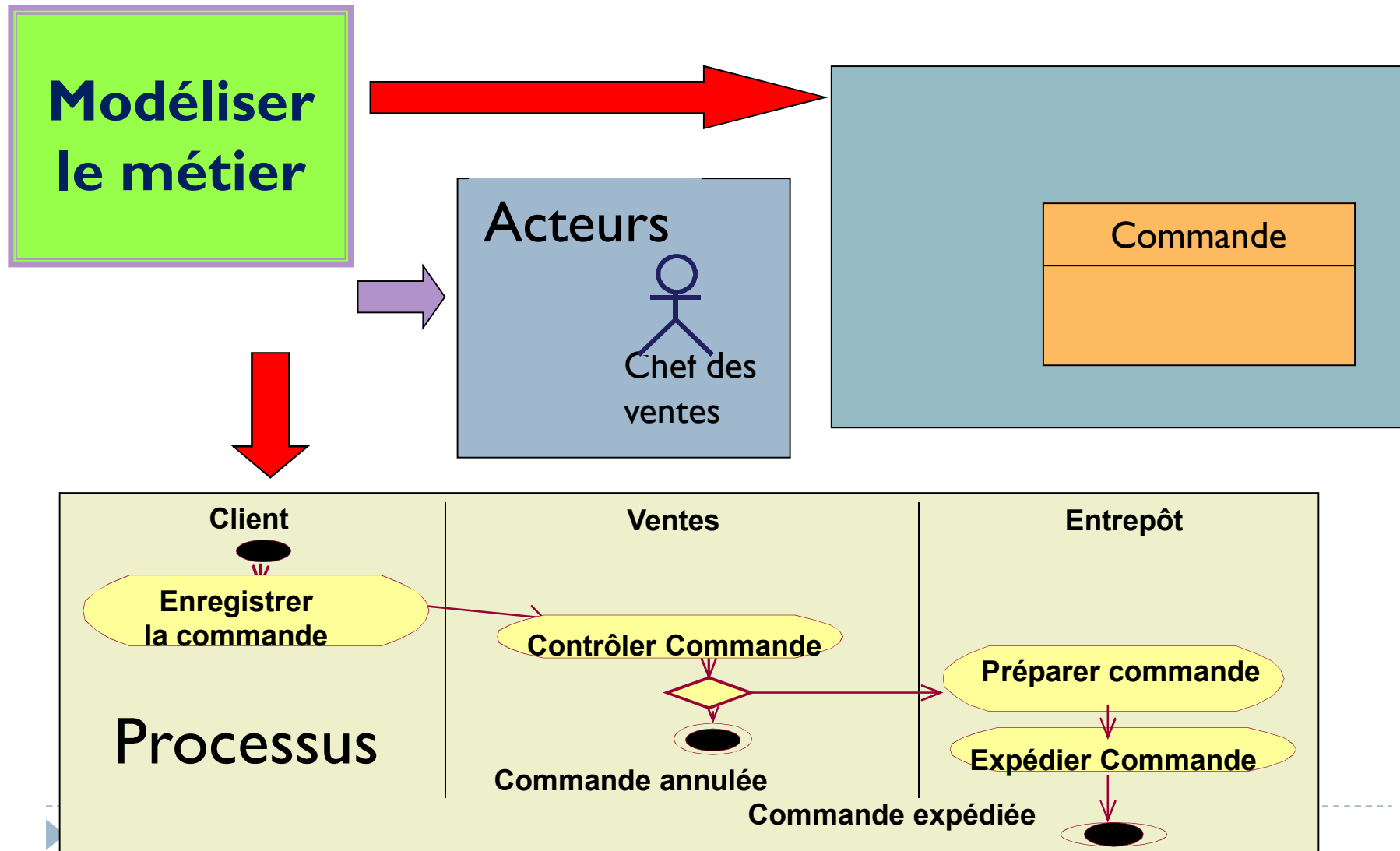
---



Exprimer **la structure** (Entités; données) et **la dynamique** ( Processus de traitements) du système désiré

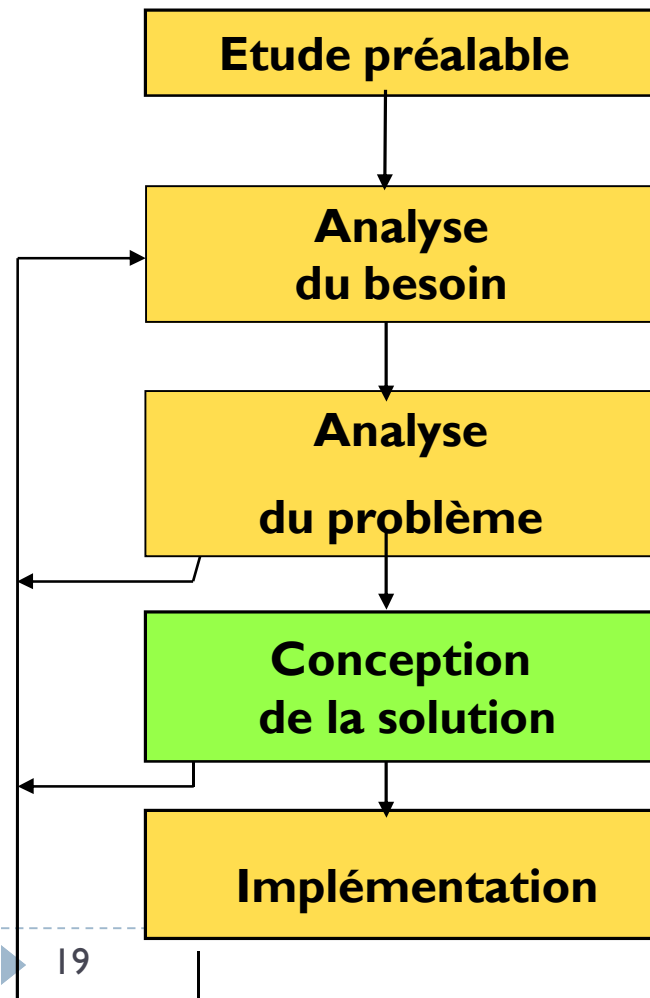
*Indépendamment de la technologie*

# Analyse du problème



# Conception de la solution

---



➤ Prendre en compte la technologie

➤ Conception structurelle et dynamique.

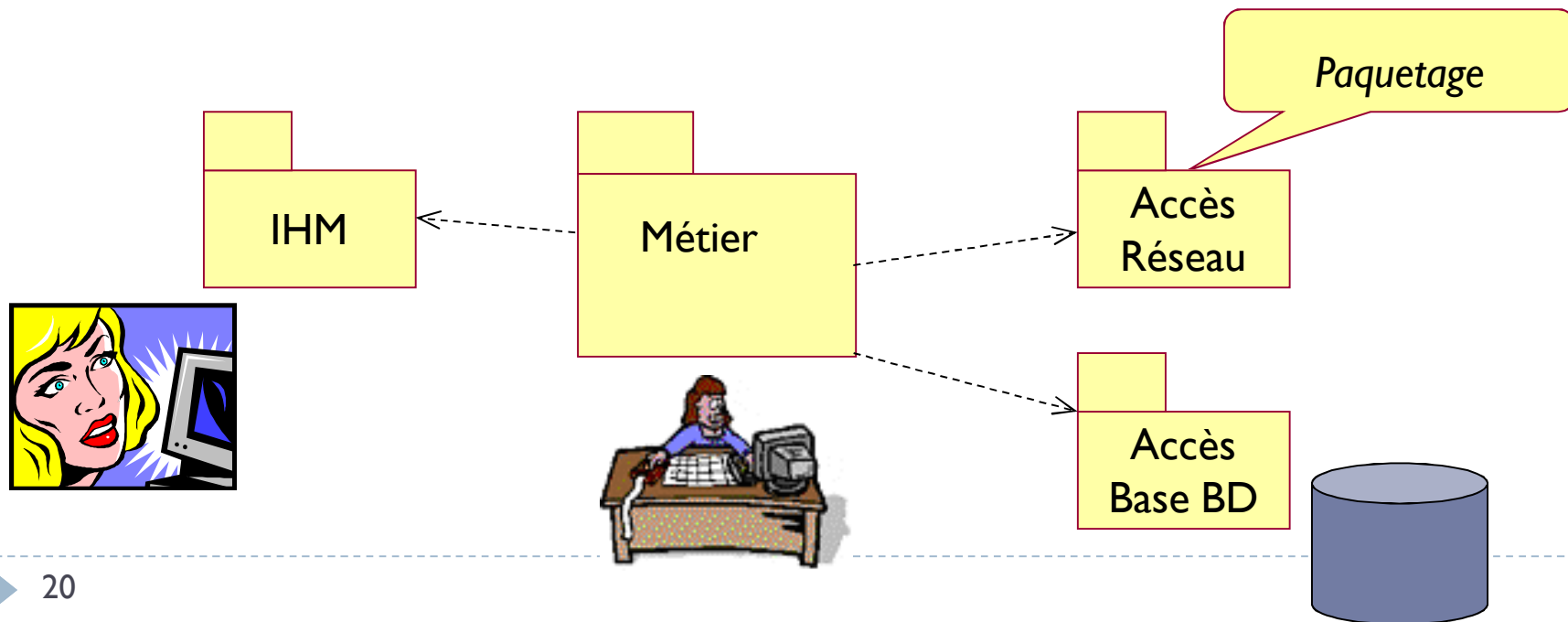
**Ne jamais concevoir avant d'analyser**

# Conception de la solution

## Conception architecturale

Modéliser l'architecture technique ou physique (structurer le logiciel)

La **visibilité** entre les paquetages est limitée.

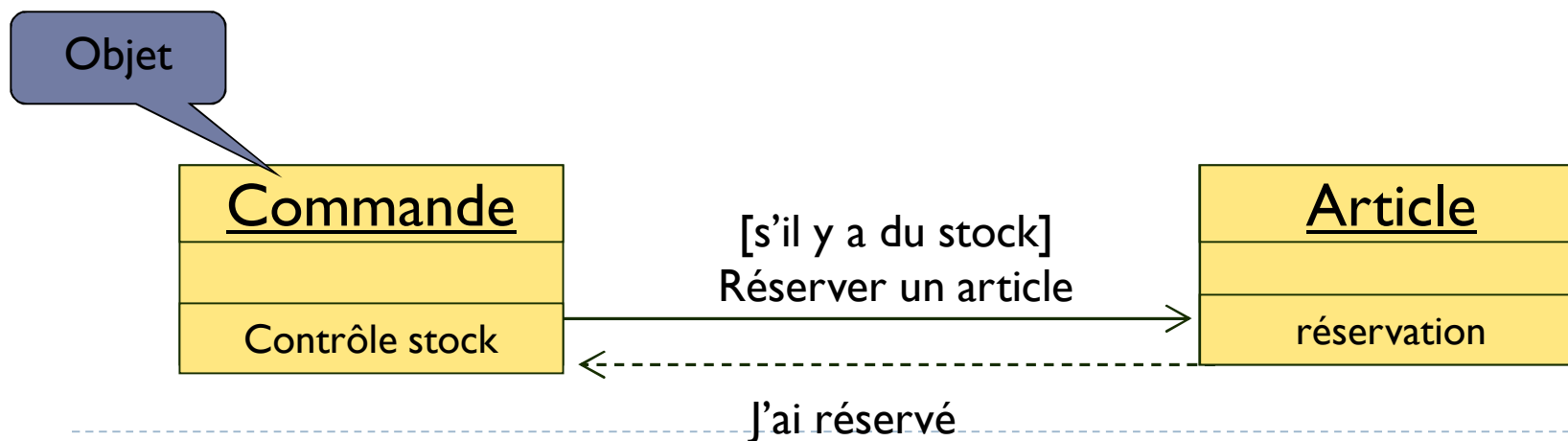


# Conception de la solution

## Conception détaillée

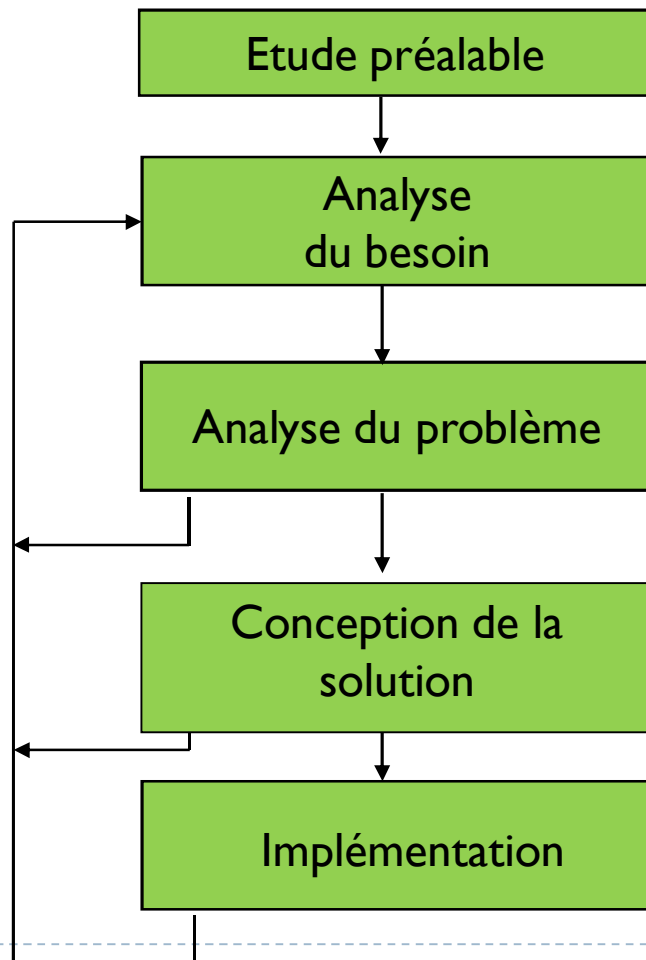
Que fait le système informatique?

- Comportement des objets
- Demandes de service



# Activités du projet: Modélisation

---



- Toutes ces activités du projet ont une part de modélisation.

- **La modélisation** est au centre de la démarche d'Analyse et de Conception.

- La modélisation intervient tout au long du processus de développement, comme un outil de description et de communication entre les acteurs.

# La notion de modèle

---

- Un modèle est une **représentation abstraite et non-ambiguë** de la réalité dans un langage donné.
- *Une maquette, un plan, une photo, un organigramme sont des modèles.*
- Les modèles servent à :
  - **Communiquer** : vérifier que l'analyste a bien compris les besoins des utilisateurs grâce à des modèles du problème (**modèles d'analyse**),
  - **Préparer la réalisation** : grâce à des modèles de la solution (**modèles de conception**).

# La modélisation

---

- ❑ La modélisation d'une manière générale permet de faciliter la communication entre humains.
- ❑ Un modèle est une abstraction de la réalité.
- ❑ Il doit faciliter la compréhension du phénomène ou système étudié: il réduit la complexité.
- ❑ Il doit permettre de simuler le phénomène ou système étudié: il reproduit ses comportements.



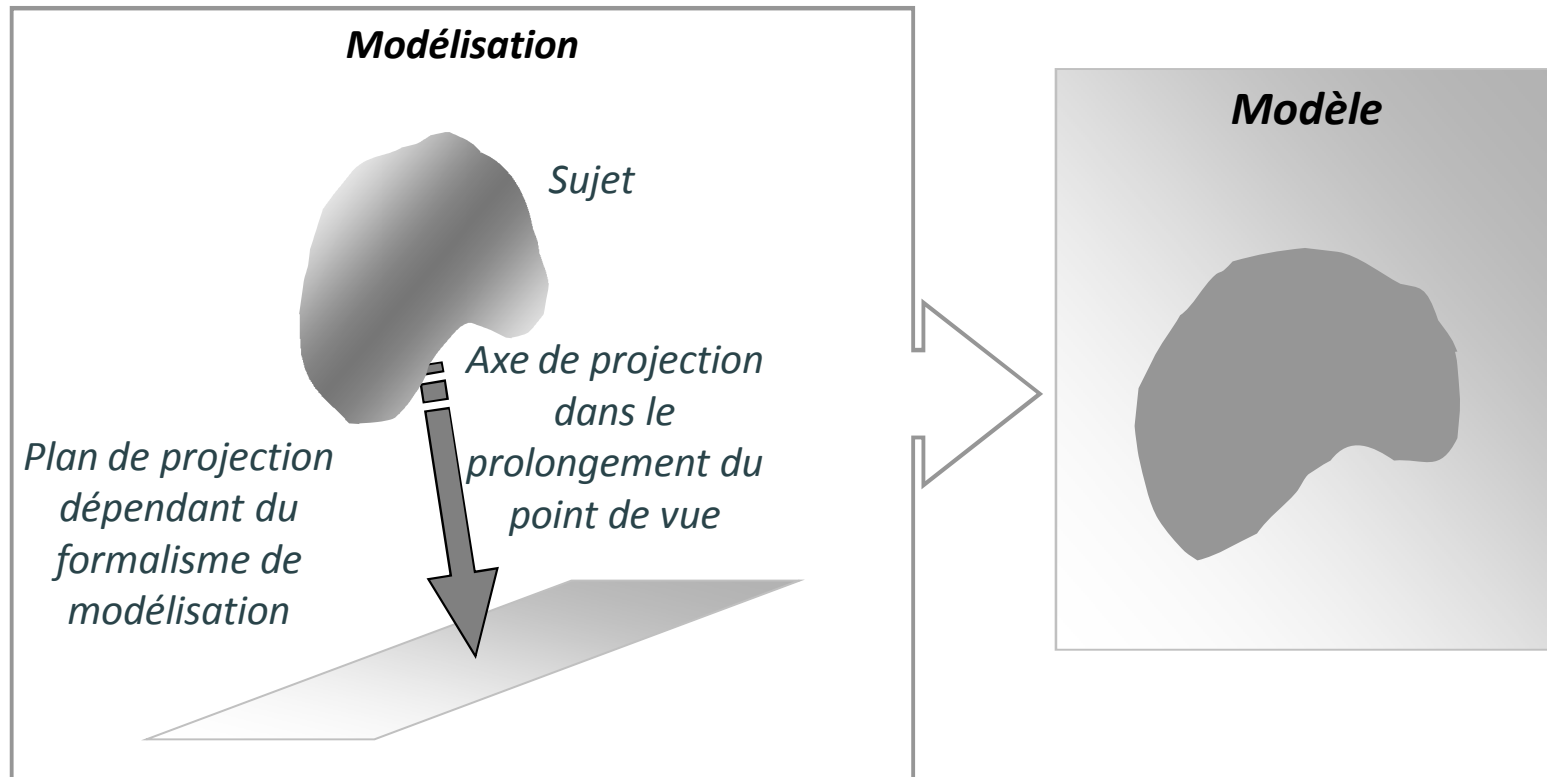
# La modélisation

---

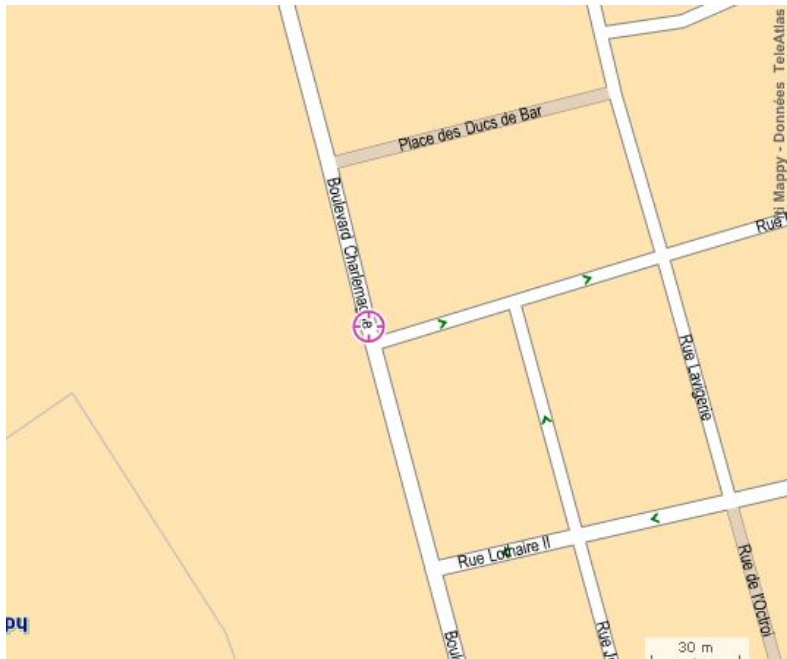
- ▶ La modélisation est finalement une activité **de projection** :
  - ▶ d'un sujet réel,
  - ▶ sur le plan d'un langage de modélisation,
  - ▶ **selon un angle de considération** résultant de l'utilisation attendue du modèle,
  - ▶ pour obtenir **une vision abstraite, partielle et formalisée** du sujet : **le modèle**.

# Principe de la modélisation

---

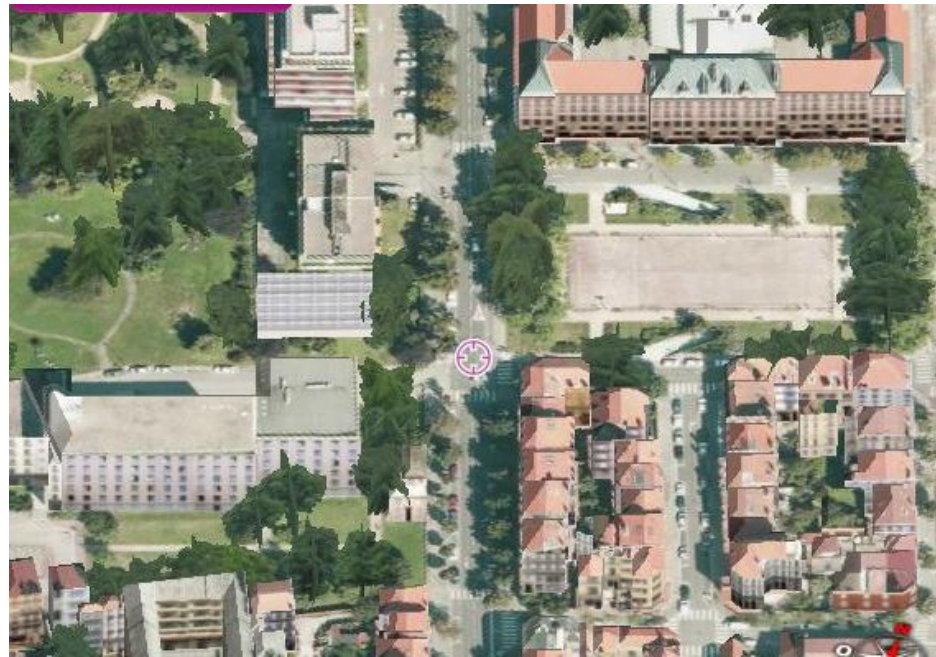


# Un modèle...



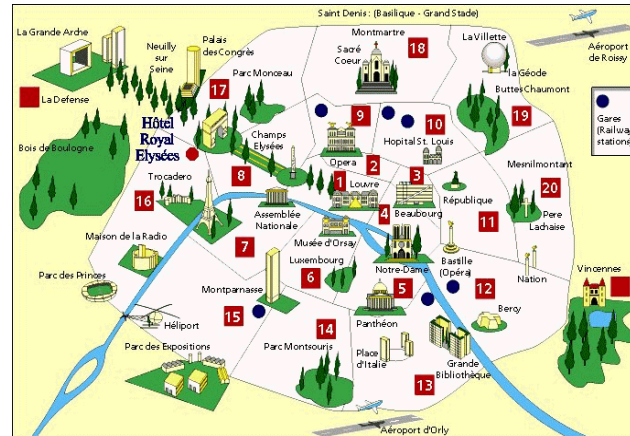
**But ?**  
**Lecteurs ?**  
**Notation ?**

# La réalité ?

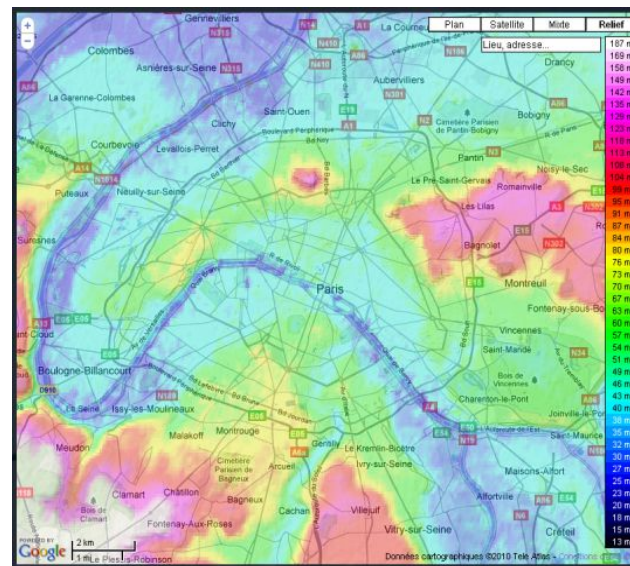


Autant de modèles que de buts, de lecteurs, de notations ... de modélisateurs.

Modèle pour touriste



Modèle pour technicien



même  
réalité

# Historique: Approches de modélisation

---

## ▶ **Approche fonctionnelle (SADT)**

- ▶ 1960 – fin 1970
- ▶ **l'important c'est les traitements**
- ▶ Séparation nette des données et traitements

## ▶ **Approche systémique (Merise)**

- ▶ (années 80)
- ▶ **Approche conceptuelle globale du système**
- ▶ Basée sur la recherche d'éléments pertinents du système: données, actions, évènements.

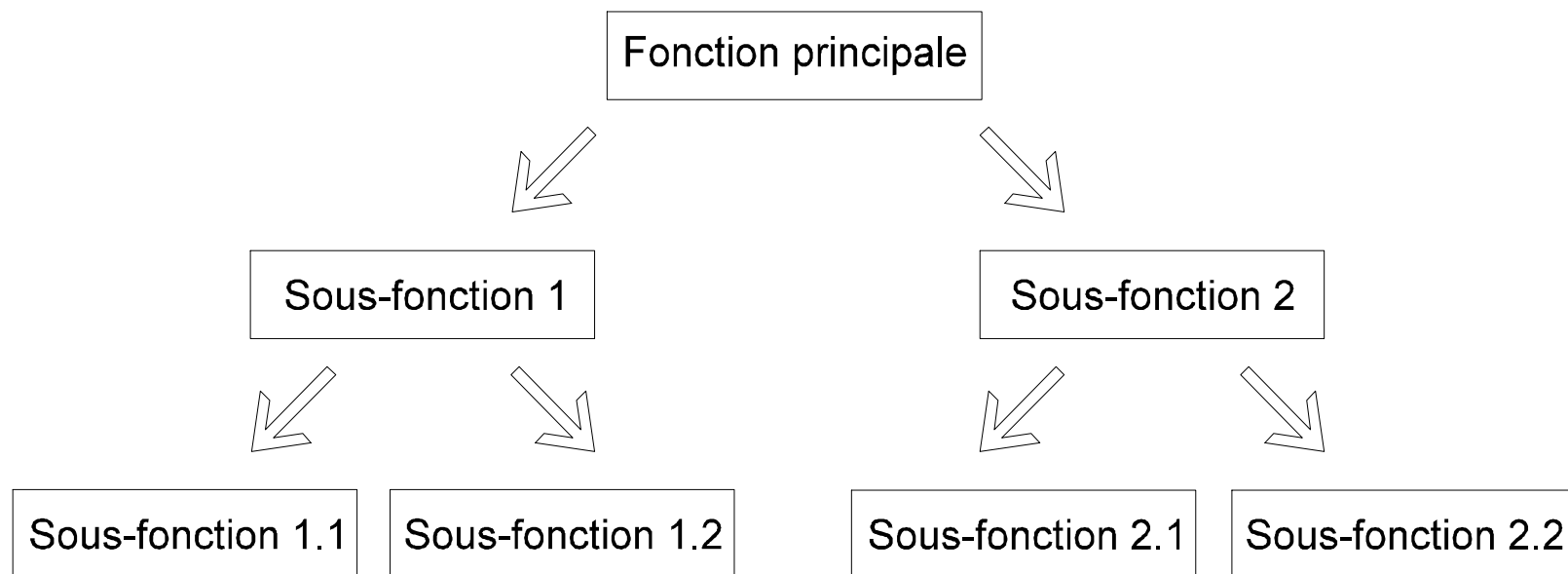
## ▶ **Emergence des méthodes objet**

- ▶ 1980 – début 1990 : première génération
- ▶ Plus de 50 méthodes objets (dont OMT, OOSE)
- ▶ **L'important c'est l'objet**
- ▶ **Objet = données + traitements**
- ▶ **Langage de modélisation orienté objet: UML**

# Approches de modélisation: **Approche fonctionnelle**

---

- La fonction donne la forme du système



## Approches de modélisation: **Approche objet**

---

- ▶ Les activités « orientées-objet » reposent, comme leur nom l'indique, sur **le concept d'objet**.
- ▶ Dans ce contexte, l'objet constitue la brique élémentaire à partir de laquelle ces activités se construisent. Dans une activité « orientée-objet », **tout est objet**.
- ▶ Un objet est donc une entité identifiée qui possède un **comportement propre** (des fonctions spécifiques) dépendant de son **état interne** et avec laquelle on peut **interagir** (échange de messages).

## Approches de modélisation: Approche objet

---

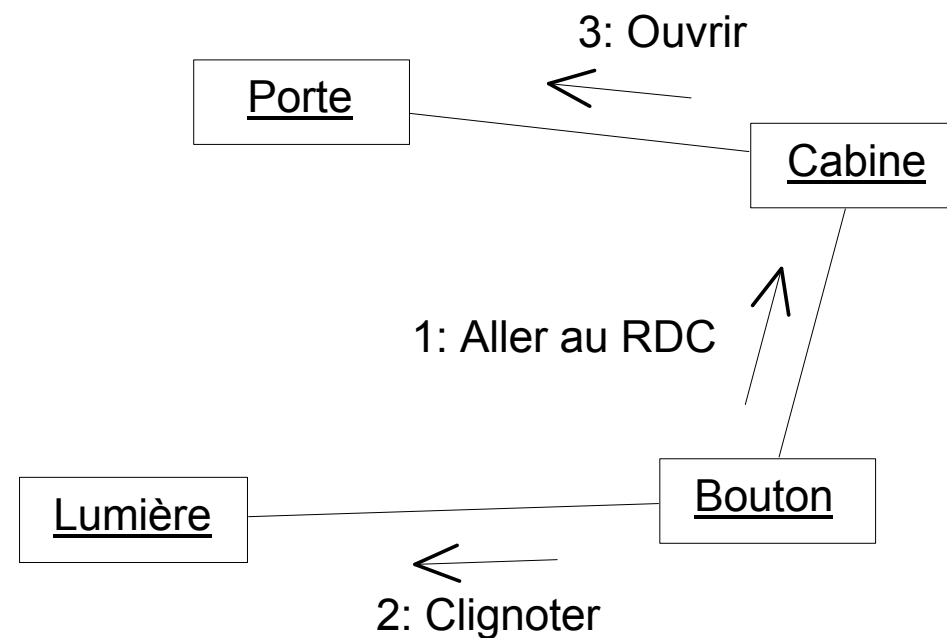
- ▶ Chaque module représente un objet du domaine de l'application.
- ▶ Les objets sont des entités autonomes qui collaborent afin de réaliser un projet global



# Approches de modélisation

---

- La fonction est réalisée par **des objets collaborants**



# Approche objet

---

- ▶ En fait, deux avantages prépondérants sont mis en général en avant lorsque l'on choisit une approche objet :
  - ▶ **La modularité** : Par construction, étant donné que l'on conçoit des classes.
  - ▶ **La réutilisabilité**
  - ▶ **La maintenance** de chaque classe est en soi plus simple à réaliser que celle d'un logiciel unique traitant toutes les données d'un système.
    - ▶ Il importe bien entendu **dans l'approche objet** de construire son système en veillant **à minimiser le nombre de relations entre classes**.

# Approche objet

---

- ▶ Au-delà de ces deux avantages majeurs et compte tenu de la plus grande modularité dans la construction d'une application à l'aide d'objets, **la maintenance** élémentaire de chaque classe est en soi plus simple à réaliser que celle d'un logiciel unique traitant toutes les données d'un système.
- ▶ Il importe bien entendu dans l'approche objet de construire son système en veillant **à minimiser le nombre de relations entre classes.**

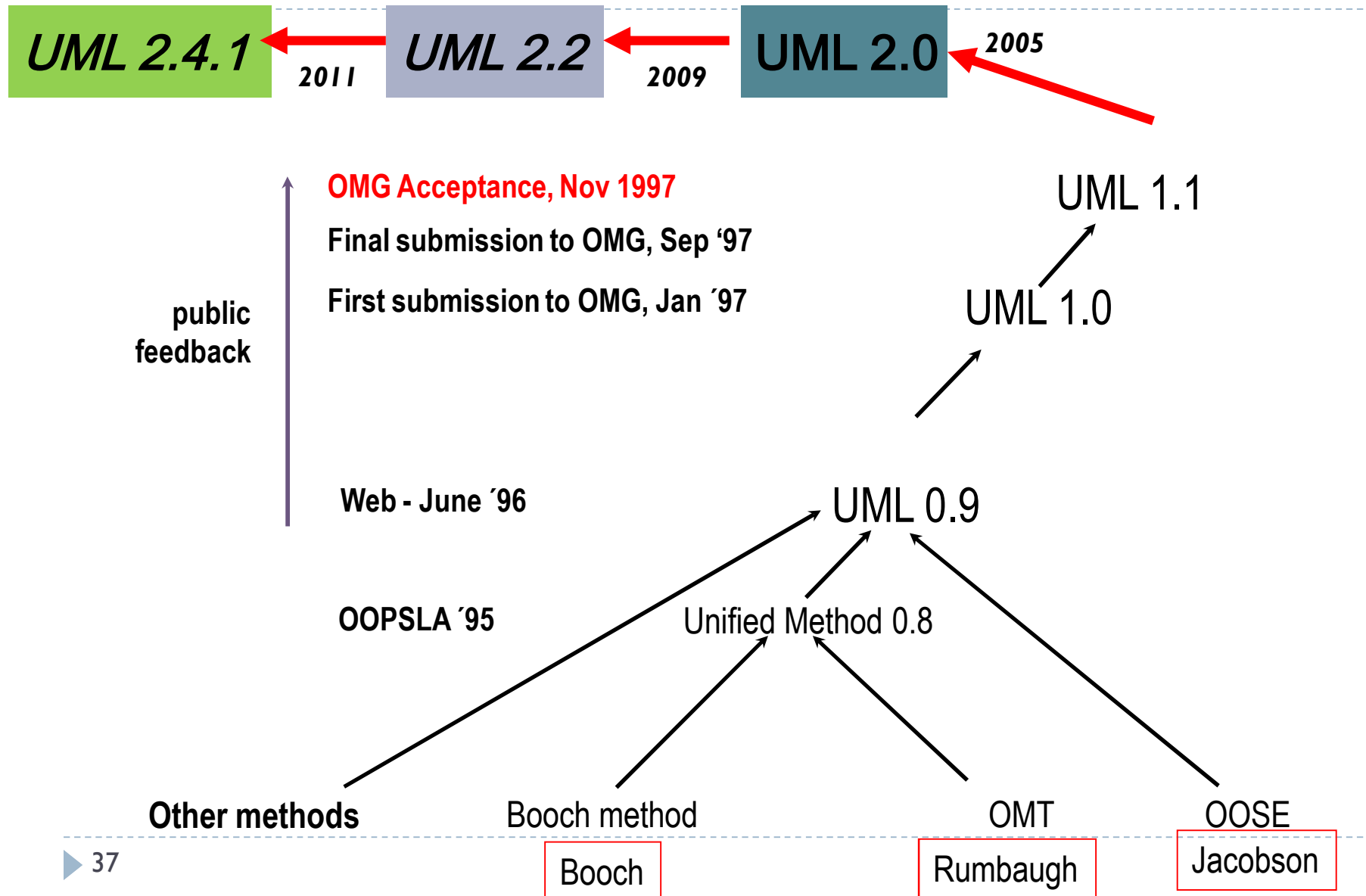
# UML

---

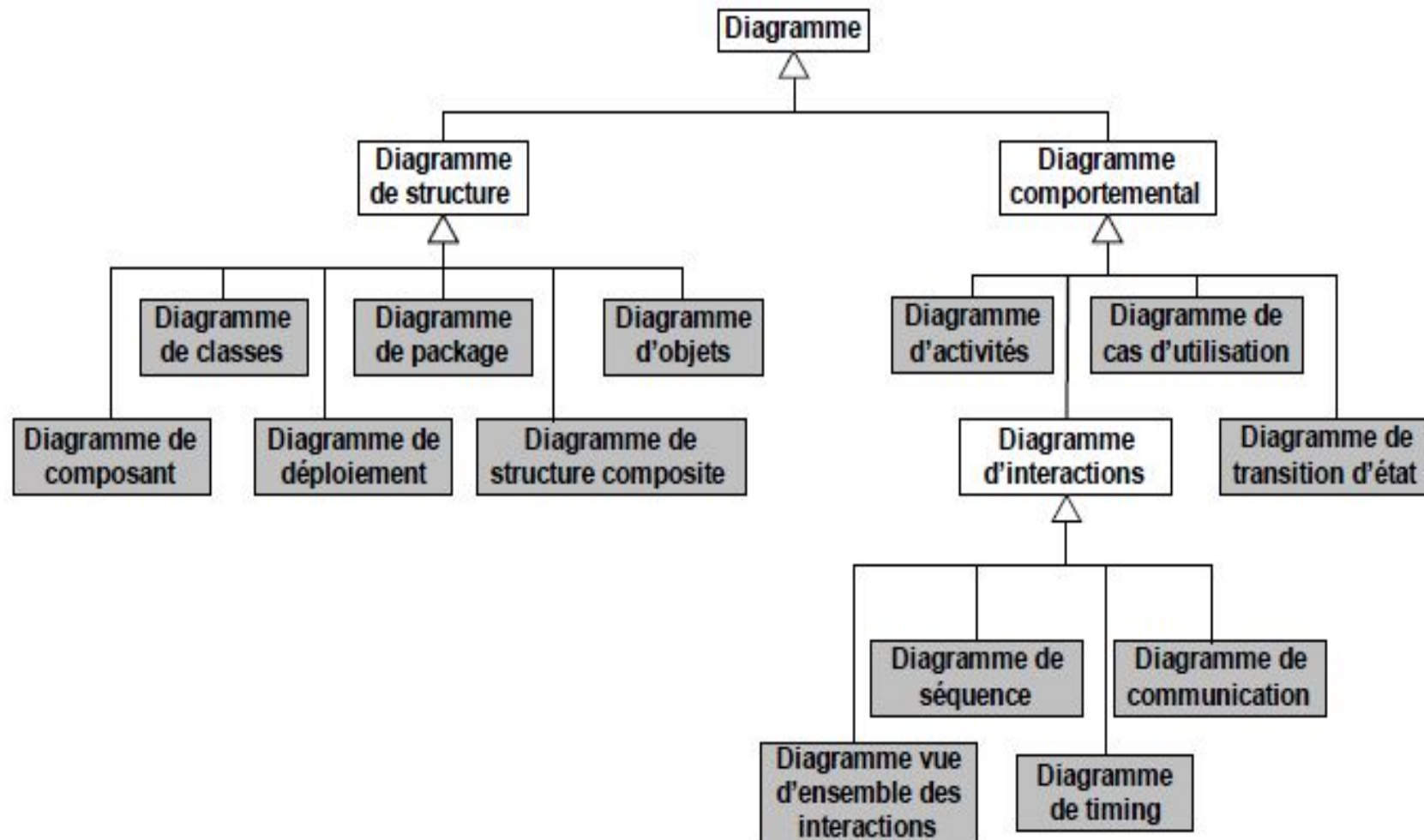
- ▶ *UML est une notation, pas une méthode*
- ▶ *UML est un langage de modélisation objet*
- ▶ *UML convient pour toutes les méthodes objet*
- ▶ *UML est un standard*

UML est **la notation** pour  
documenter les modèles objets

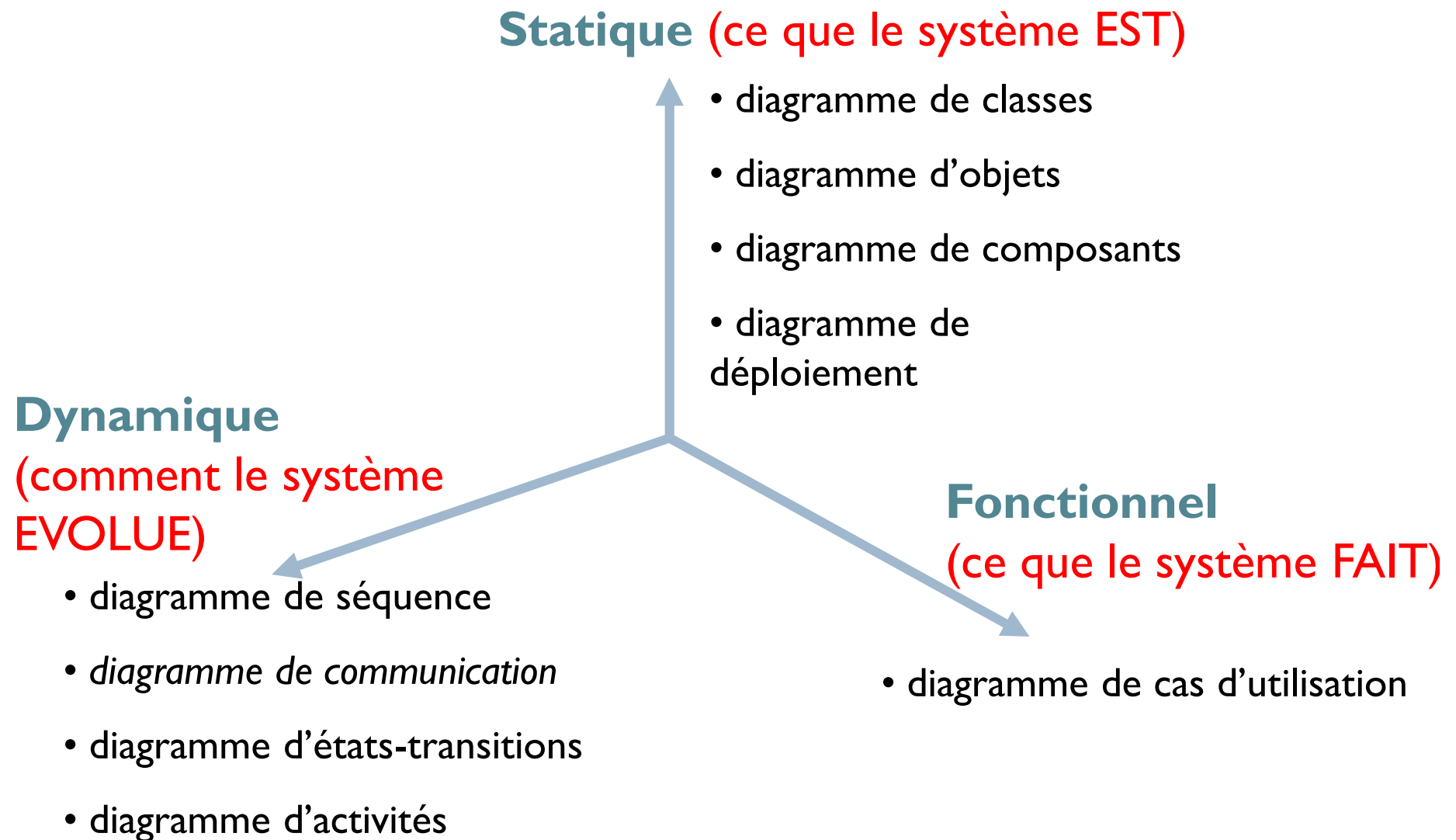
# UML



# UML 2



### 3 Axes de modélisation d'un système



# Définitions: Normes et Standards

---

- ❑ Ces deux termes sont souvent utilisés l'un à la place de l'autre alors qu'ils relèvent d'instances fort différentes.
- ❑ Cette confusion est essentiellement liée au fait **qu'en anglais, il n'existe qu'un seul mot**, le terme "**standard**" pour désigner les deux concepts.

## **Norme**

- ❑ Les normes sont des ensembles de règles approuvées par **des instances officielles** (un organisme, national ou international) **en charge de la normalisation**:
  - ❑ ISO, AFNOR, INNORPI, etc.
- ❑ Elles offrent une certaine garantie de stabilité et de pérennité.





# Normes et Standards

---

## Standards

- ▶ sont définis par des groupes qui n'ont pas de mandats officiels des gouvernements.
- ▶ Ces groupes peuvent être :
  - ▶ Industriel ou commerciaux : par exemple PostScript ou PDF sont des standards de fait qui sont définis par la société Adobe,
  - ▶ Groupe d'experts :
    - le W3C (World Wide Web Consortium)
    - le consortium Unicode,
    - OMG (Object Management Group),
- ▶ Un standard est aussi un procédé ou un service qui est **largement répandu**.

## Diapositive 41

---

**A1**

Admin; 14/10/2014

# International Organization for Standardization (ISO)

---

- Créé en 1946
- **118 pays**, siège à Genève
- La Tunisie représentée par l' **INNORPI** (Institut National de la Normalisation et de la Propriété Industrielle )
- La France représentée par l' **AFNOR** (**A**ssociation **F**rançaise de **NOR**malisation)  
*l'organisation officielle en charge des normes en France.*
- Couvre tous les secteurs à l'exception de l'électricité et de l'électronique
- Plus de 10000 normes

## Chapitre 2:

### Diagramme comportementaux:

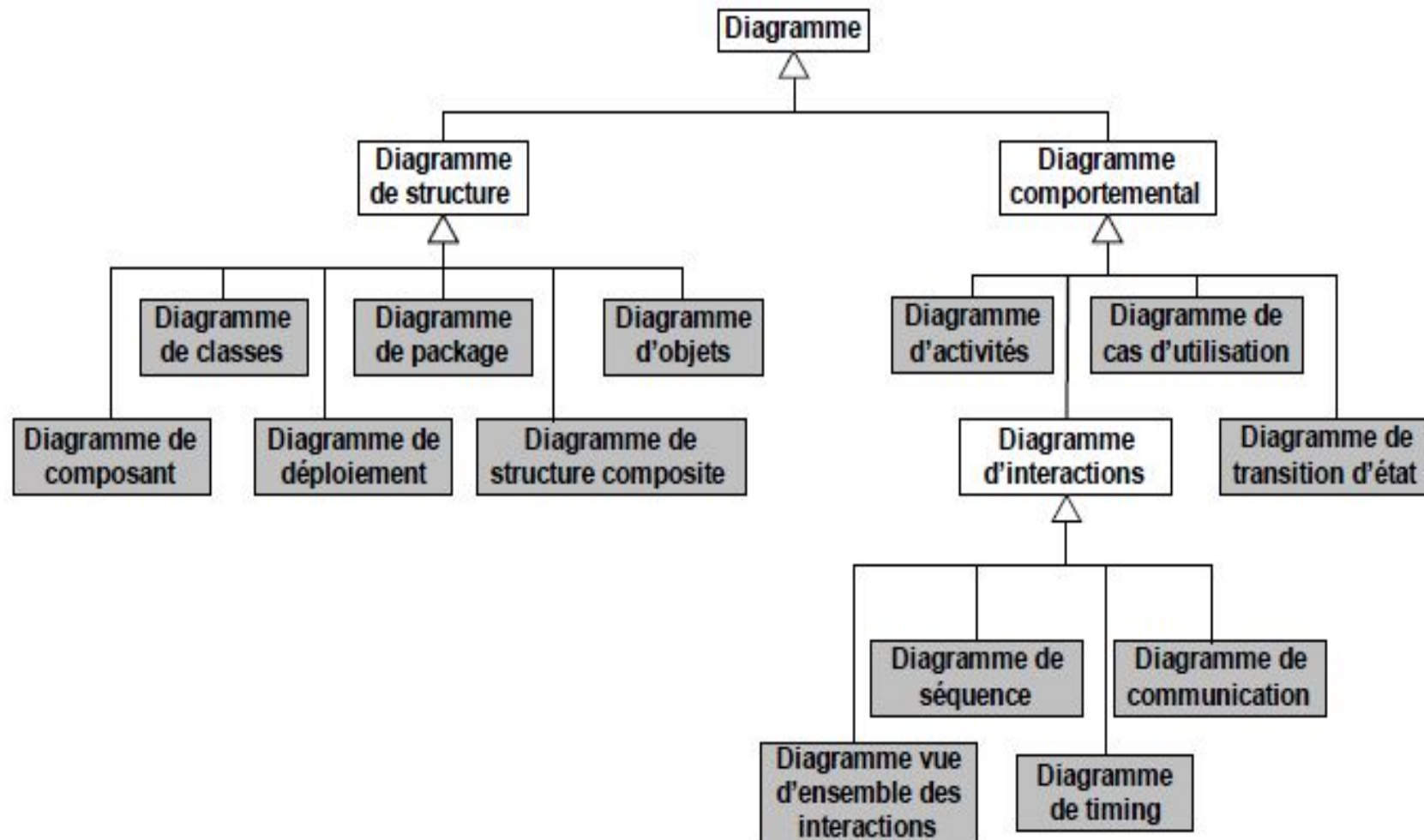
Diagramme de cas d'utilisation

# Les activités du projet

---

- ▶ Etude préalable : Analyse du contexte
- ▶ Analyse et spécification des besoins
- ▶ Analyse du problème ou métier ou système
- ▶ Conception de la solution

# UML 2



# Diagramme des cas d'utilisation

---

- ▶ Permet de définir **les limites du système et ses relations** avec l'environnement.
- ▶ **Utilisé pour modéliser les exigences (besoins) du client.**

# Concepts de base: Cas d'utilisation

---

- ▶ Les cas d'utilisation constituent un moyen **de recueillir et de décrire** les besoins **des acteurs du système**.
- ▶ Ils peuvent être aussi utilisés ensuite comme **moyen d'organisation du développement du logiciel**:
  - ▶ structuration et le déroulement de **la conception et des tests logiciels**.

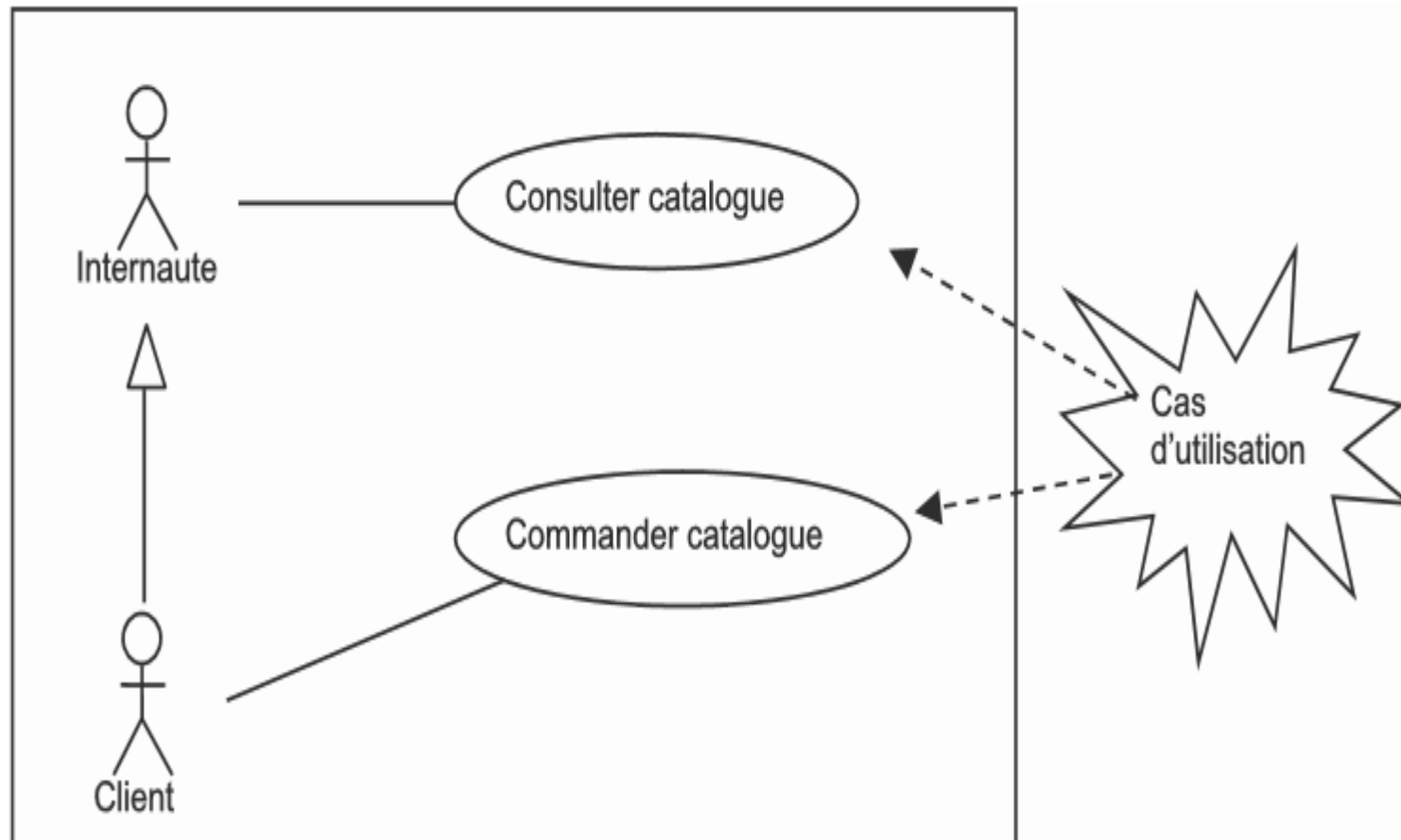


## Objectifs: Cas d'utilisation

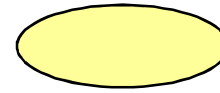
---

- Capturer **le comportement** désiré du système,
- **Servir d'entente** entre les différents intervenants (développeurs, utilisateurs et experts) sur **les fonctions disponibles** et sur **la façon d'interagir avec le système**.
- Spécifier **ce que le système fait** (fonctions), **mais pas comment il le fait**.

# Exemple: Cas d'utilisation



# Cas d'utilisation



- ▶ Un cas d'utilisation (*use case*) décrit:
  - ▶ Une **fonctionnalité** du système suivant le point de vue de l'utilisateur.
  - ▶ **Les interactions** entre les acteurs et le système,
  - ▶ Un **comportement** attendu **du système** du point de vue d'un ou de plusieurs acteurs,
  - ▶ Un **service** rendu par le système.

# Cas d'utilisation: Scénario

---

- ▶ **Un cas d'utilisation** = Ensemble de « chemins d'exécution » possibles
- ▶ **Un scénario** = Un chemin particulier d'exécution
- ▶ **Un scénario** = Instance de cas d'utilisation
- ▶ Exemple:
- ▶ Cas d'utilisation : *Acheter un objet sur internet*
- ▶ Mais il peut y avoir des scénarios tels que:
  - *échec lors du paiement*
  - *Article non disponible*

## Cas d'utilisation

---

- ▶ La représentation d'un cas d'utilisation met en jeu trois concepts :
  - ▶ l'acteur,
  - ▶ le cas d'utilisation,
  - ▶ l'interaction entre l'acteur et le cas d'utilisation.

# Diagramme de cas d'utilisation

---

- ▶ Un diagramme de cas d'utilisation :

- ▶ décrit

- ▶ les **acteurs**



- ▶ les **cas d'utilisation**

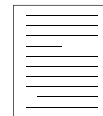
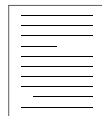
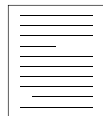
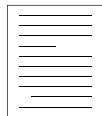


- ▶ le **système**



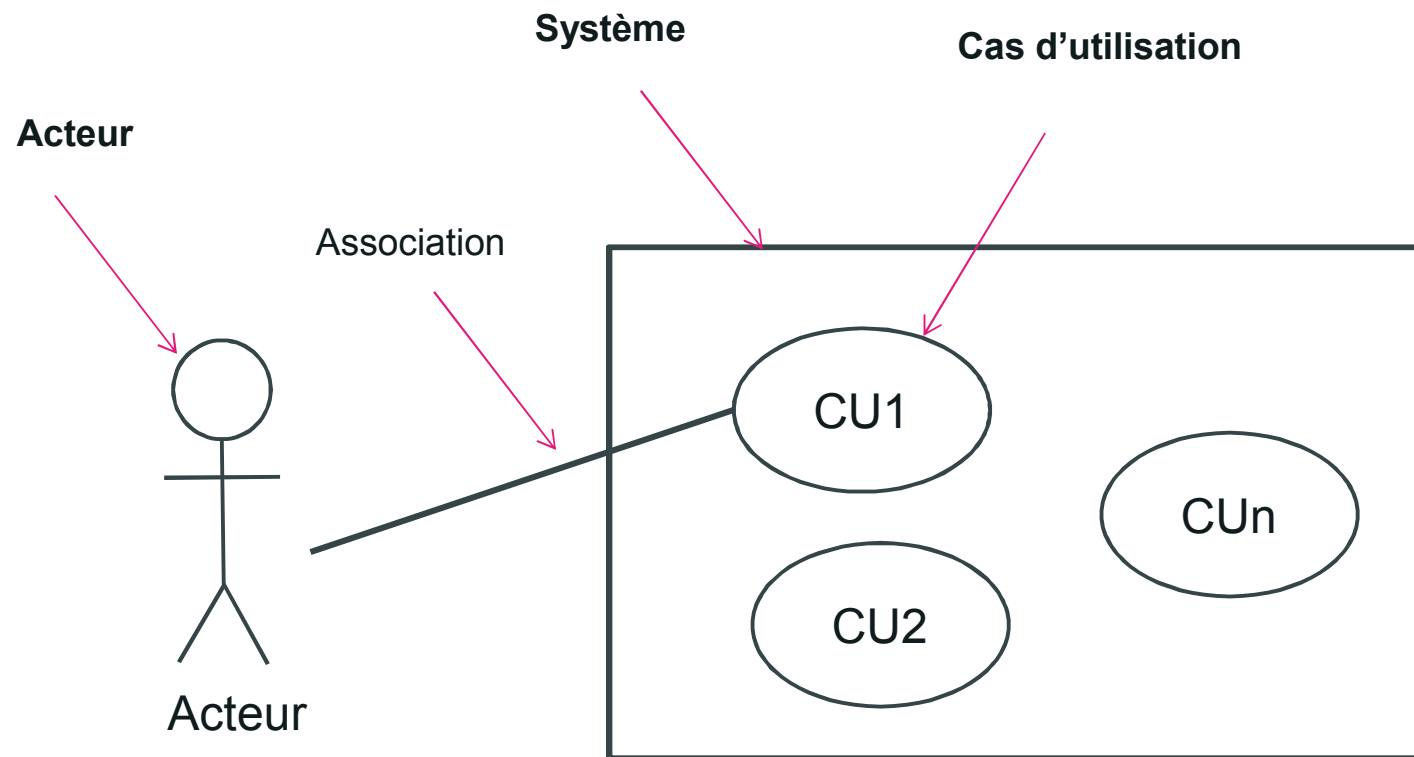
- ▶ contient

- ▶ des **descriptions** textuelles



# UML : Diagramme De Cas D'utilisation

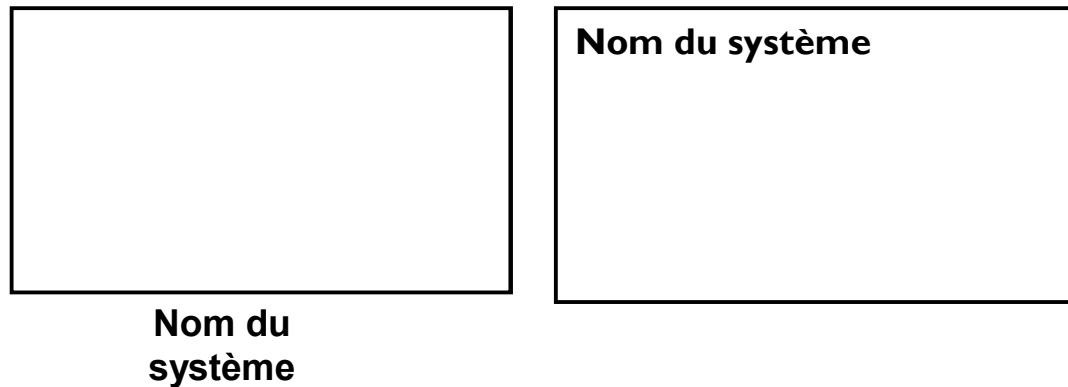
---



# Le système

---

- ▶ Le système est **un ensemble de cas d'utilisation**
- ▶ Le système ne comprend pas les acteurs.





# Cas d'utilisation

---

- ▶ Use Case :
  - ▶ Ensemble de séquences d'actions réalisées par le système et qui produisent un résultat **intéressant** pour un acteur particulier.

# Acteurs

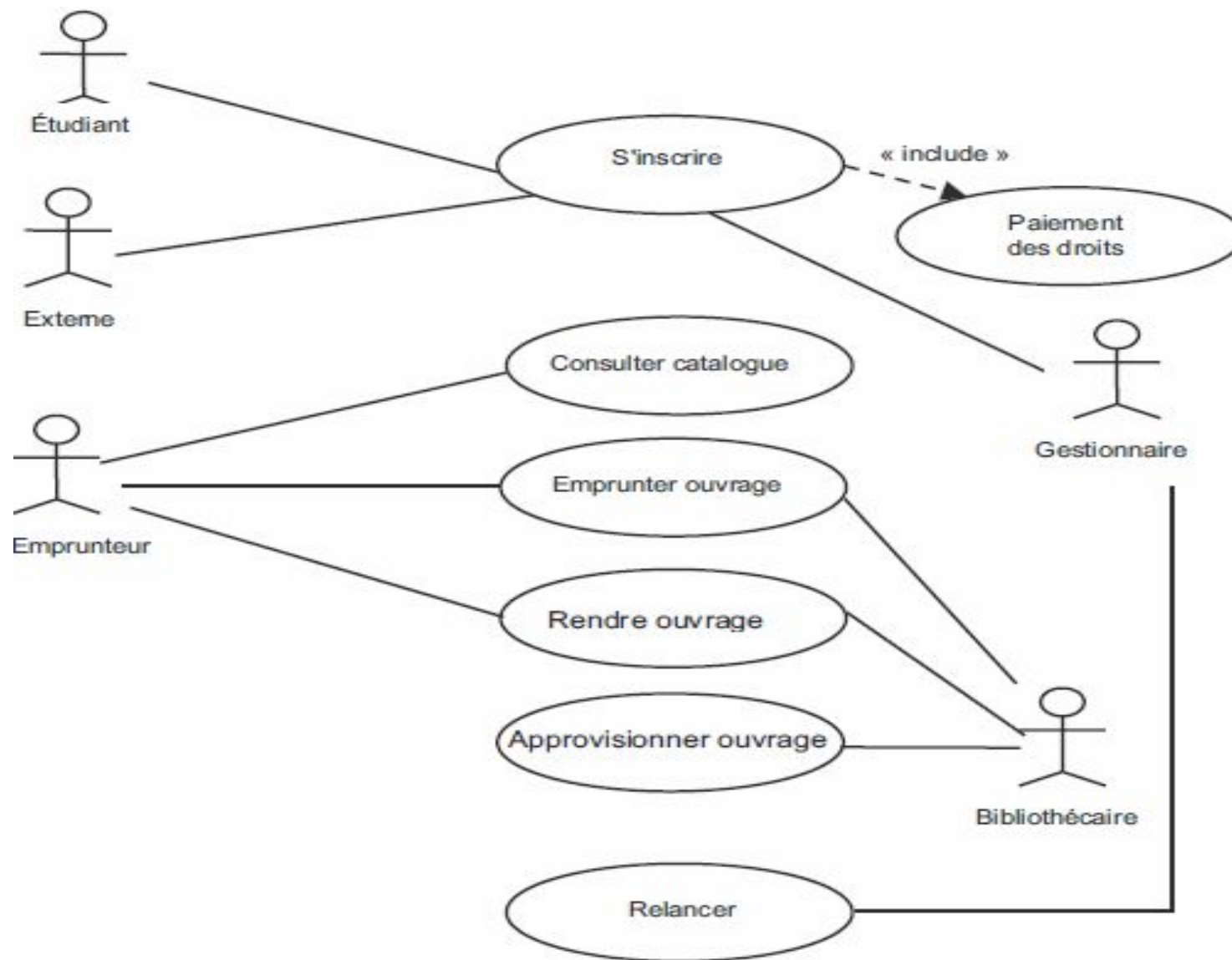
---

- ▶ UML n'emploie pas le terme d'utilisateur mais d'acteur.
- ▶ Le terme acteur ne désigne pas seulement des utilisateurs humains mais également les autres systèmes (machines, programmes, ...)

# Acteurs

---

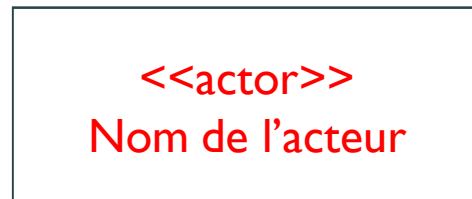
- ▶ Un acteur est un rôle joué par une entité externe qui est en interaction avec le système (échange de l'information en entrée et/ou en sortie):
  - ▶ des utilisateurs humains,
  - ▶ matériels,
  - ▶ logiciels.



# Acteur

---

- ▶ Est représenté par:
  - ▶ Un petit bonhomme (*stick man*) avec son nom dessous **ou**
  - ▶ Par un rectangle contenant le mot-clé **<< actor >>** avec son nom dessous **ou**
  - ▶ Par un mélange de ces 2 représentations.



# UML : diagramme de cas d'utilisation

---

## Acteurs vs Personnes :

Ne pas confondre **acteur** et **personne** utilisant le système :

- Une même personne peut jouer plusieurs rôles,
- Plusieurs personnes peuvent jouer un même rôle,
- Un acteur n'est pas forcément une personne physique.