UML – Unified Modeling Language 3/4 : diagrammes dynamiques et d'interaction

Yannick Prié
Département Informatique – Faculté des Sciences et Technologies
Université Claude Bernard Lyon 1
2009-2010

Objectifs de ce cours

- Apprendre la syntaxe et la sémantique des diagrammes dynamiques et d'interaction les plus importants
- Améliorer au passage la compréhension de différents principes objets

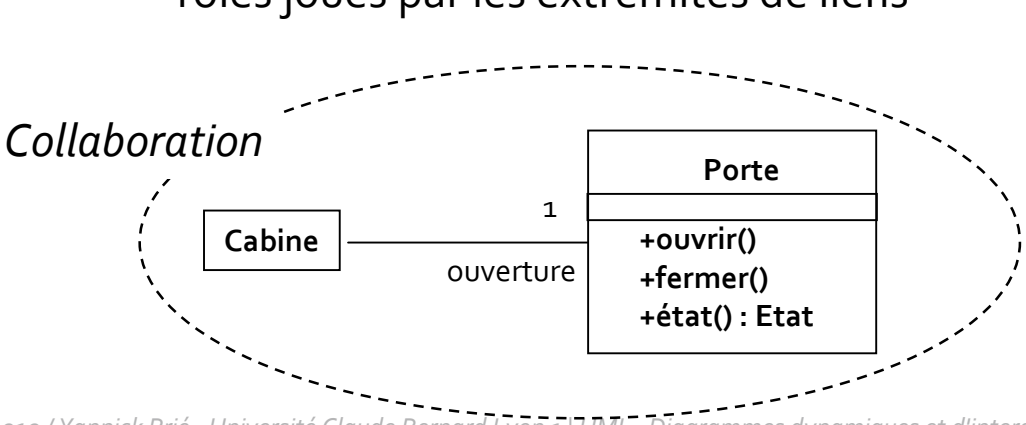
- Remarque
  - On ne traitera pas des cas d'utilisation ici, il y a un cours dédié

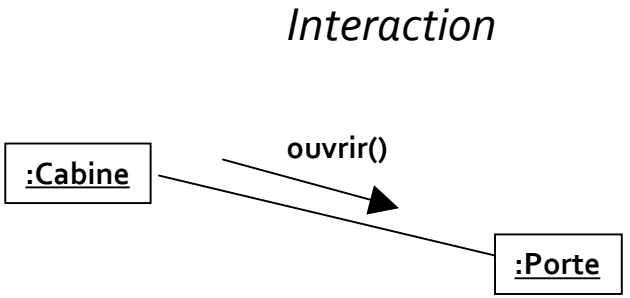
Plan

- **Diagrammes d'interaction**
  - diagrammes de séquences
  - diagrammes de communication
- Diagrammes d'activité
- Diagrammes de machines d'états
- Autres diagrammes UML
- Autres diagrammes non UML

Collaborations et interactions

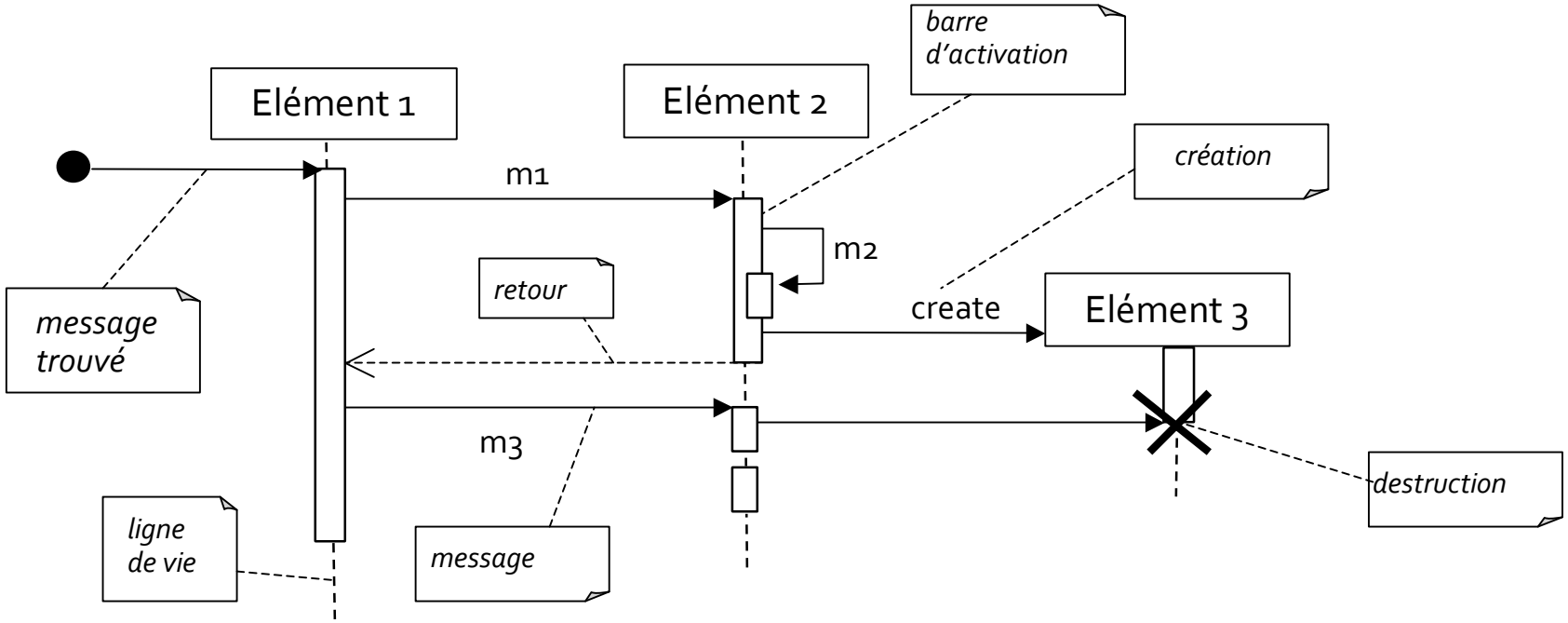
- Collaboration
  - ensemble de rôles joués par des classes fournissant un contexte d'interaction
- Interaction
  - communication entre instances des éléments d'une collaboration
  - ensemble partiellement ordonné de messages
  - plusieurs interactions possibles pour une même collaboration
- Éléments d'une interaction
  - participants (UML1 : objets, UML2 : souvent objets)
  - liens (supports de messages)
  - messages (déclenchant des opérations)
  - rôles joués par les extrémités de liens





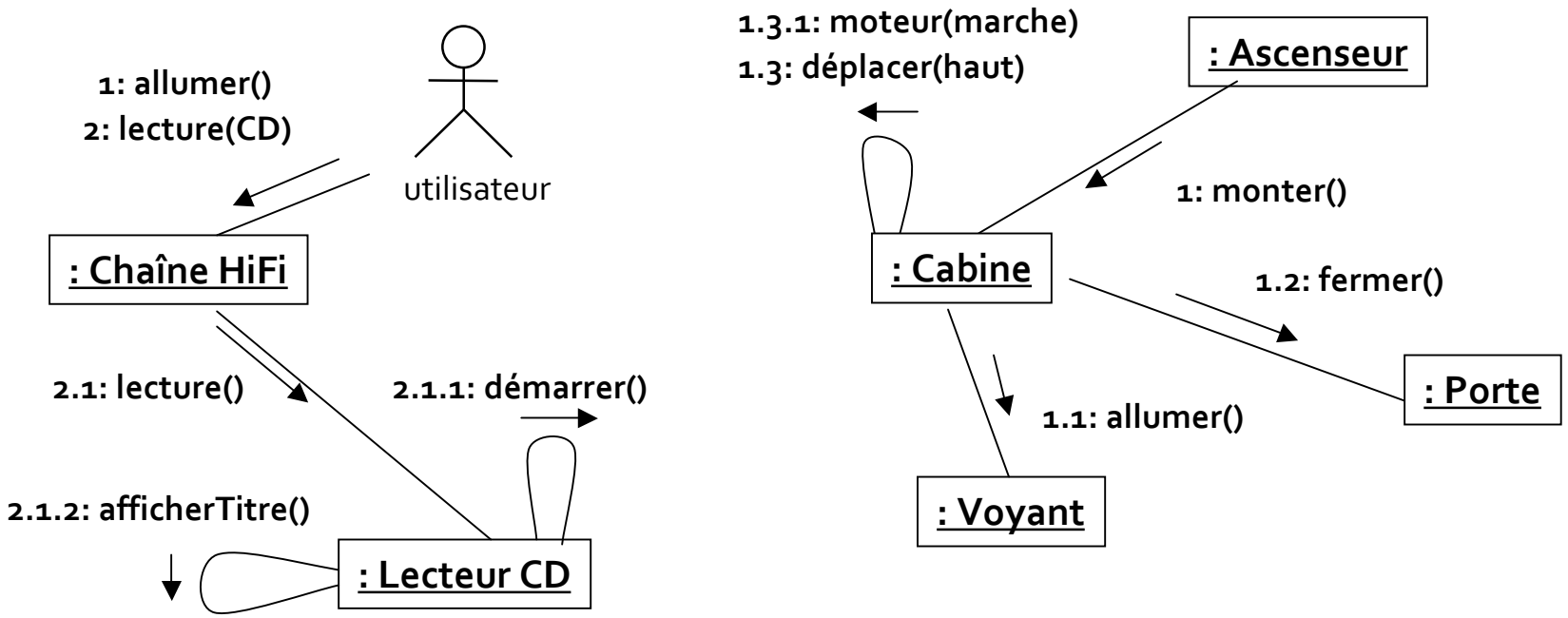
Diagrammes de séquences

- Interactions entre éléments dans une séquence temporelle
  - aspect chronologique ne rendant pas compte explicitement du contexte
  - permet de bien montrer qui fait quoi dans une interaction
- Description de scénarios typiques et des exceptions



Diagrammes de communication
(UML1 : diagrammes de collaboration)

- Diagramme d'objets rendant compte de la dynamique
  - structure spatiale permet la collaboration d'objets
  - dimension temporelle : ordre des messages
    - numérotation pointée



Petit exercice à faire en classe

- Dessiner un diagramme de communication impliquant le passage de la balle entre deux tortues d'équipes différentes.

Utilisation des interactions

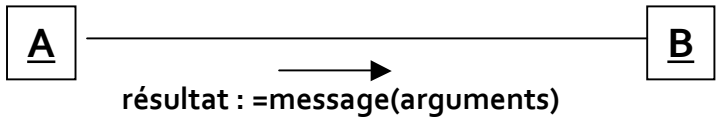
- Etudier/spécifier le comportement
  - du système dans sa globalité au sein d'un cas d'utilisation
    - se concentrer sur les événements du système considéré comme boîte noire
      - diagramme de séquence système (exemple plus loin)
  - de plusieurs objets au sein d'un cas d'utilisation
    - réalisations de CU comme des interactions dans une société d'objets
      - diagramme de séquence de fonctionnement
      - conseil : concevoir diagrammes de classes et d'interaction en même temps
- Illustrer/étudier un fonctionnement
  - diagramme qui traverse les couches : de l'IHM aux données
  - rétro-ingénierie

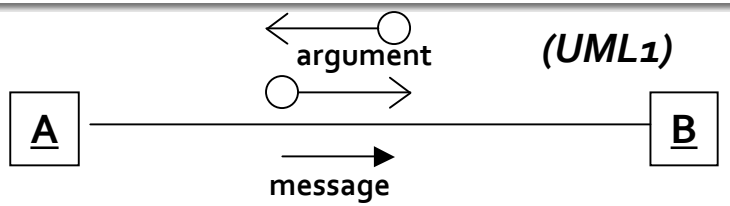
Messages

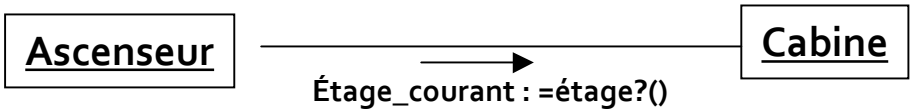
- Matérialisation d'une communication avec transmission d'information entre
  - émetteur (source)
  - récepteur (destination)
- Un message déclenche
  - une opération,
  - l'émission d'un signal
  - la création/destruction d'un objet
- Deux types principaux
  - appel de procédure ou flot de contrôle emboîté (appel standard de fonction)
    - déplacer()
  - flot de contrôle asynchrone (threads)
    - démarrer()
  - autres : à plat, dérobant (réception si attente), minuté (message actif pendant Dt)

Messages dans les diagrammes de communication

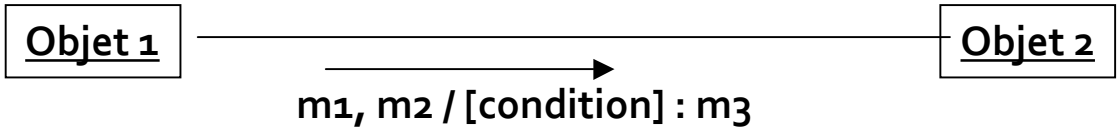
Résultats et arguments



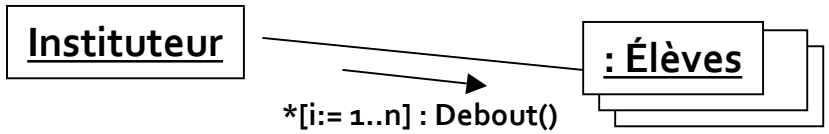




Synchronisation



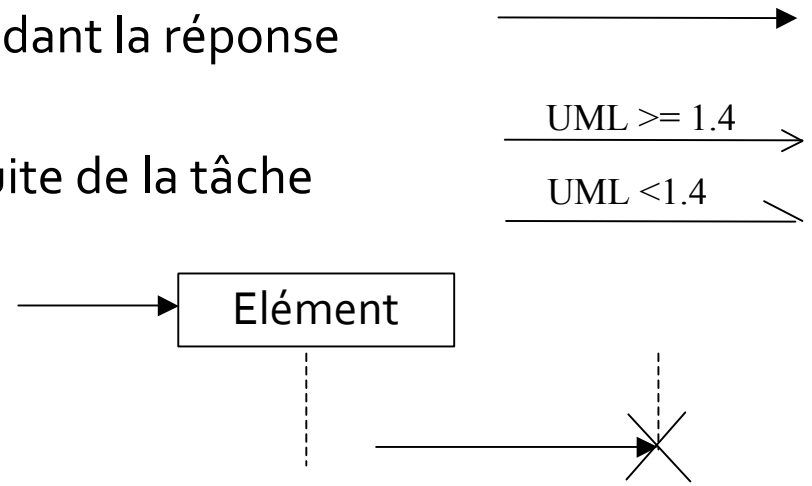
Itérateurs



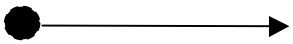
Garde

Messages dans les diagrammes de séquences

- Notation résultat = message(arguments)
- Echange de messages
  - flèches d'appel standard
    - blocage de l'émetteur en attendant la réponse
  - flèche d'appel asynchrone
    - pas d'attente du retour, poursuite de la tâche
  - Retour
  - Message de création
  - Message de destruction

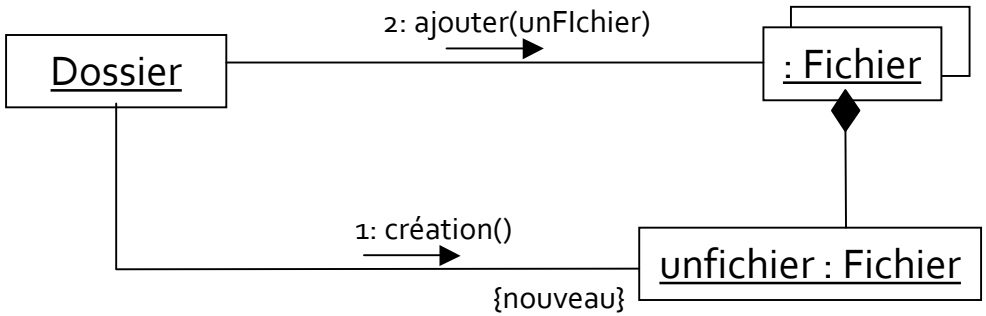


- Lancement de l'interaction venant de l'extérieur
  - 1er message = « message trouvé »

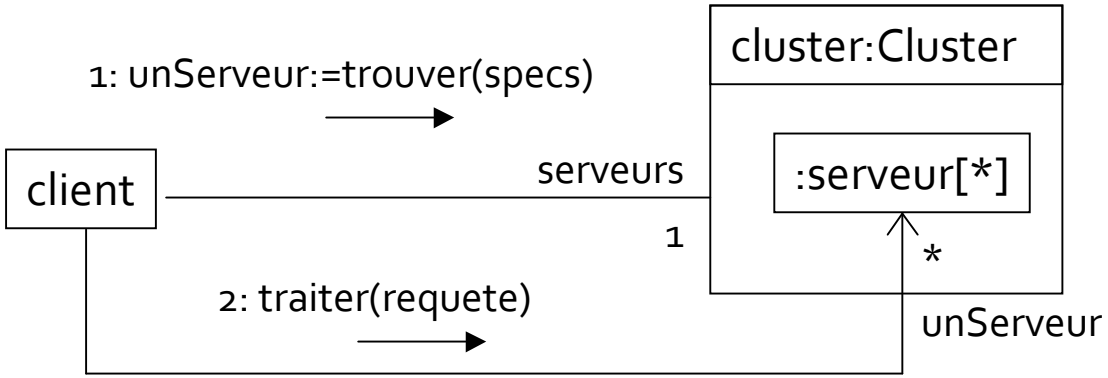


Gestion de collections

- Créer et ajouter (UML1 : multi-objet)



- Récupérer et utiliser (UML2 : structure composite)

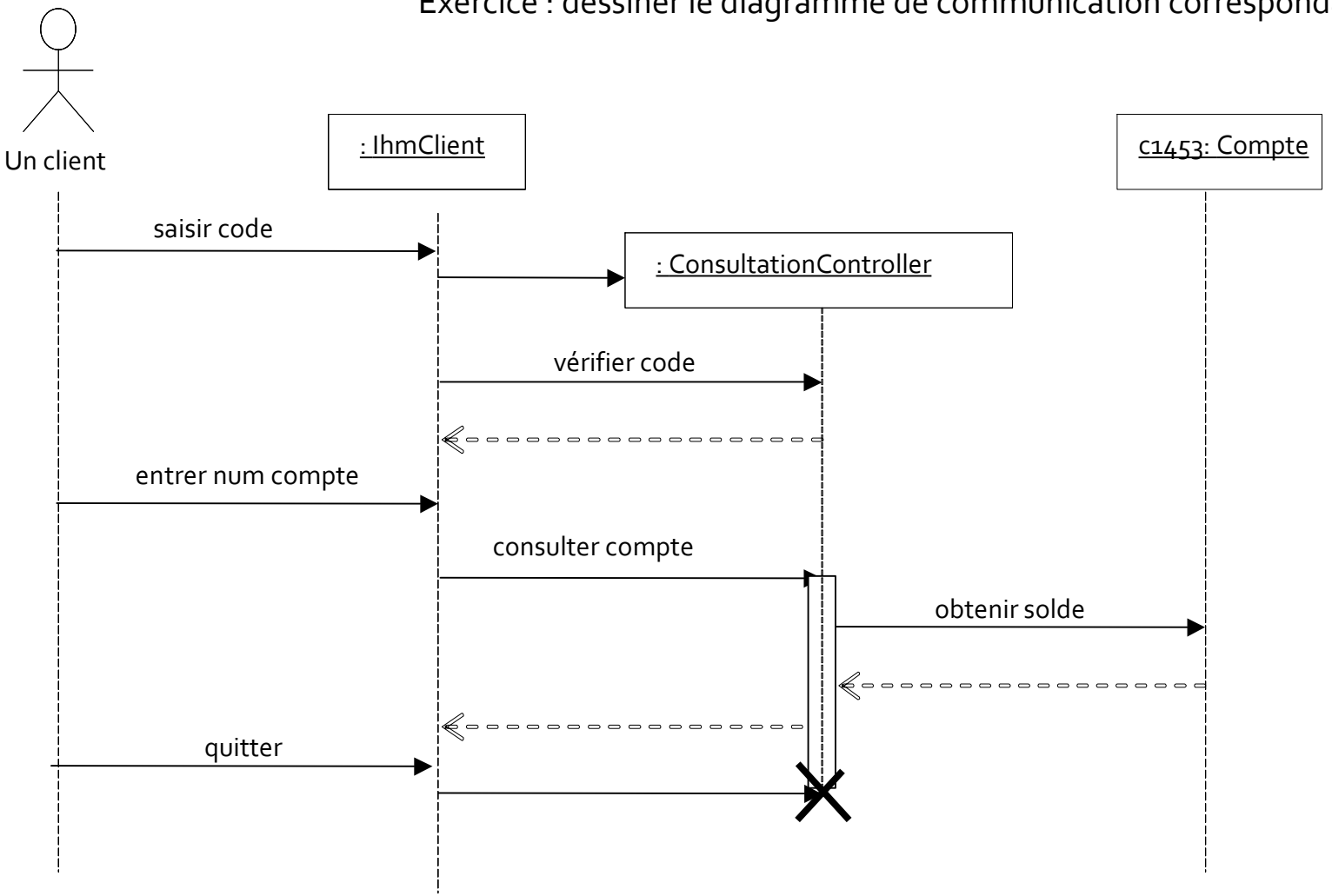


Diagrammes de séquences dans UML2

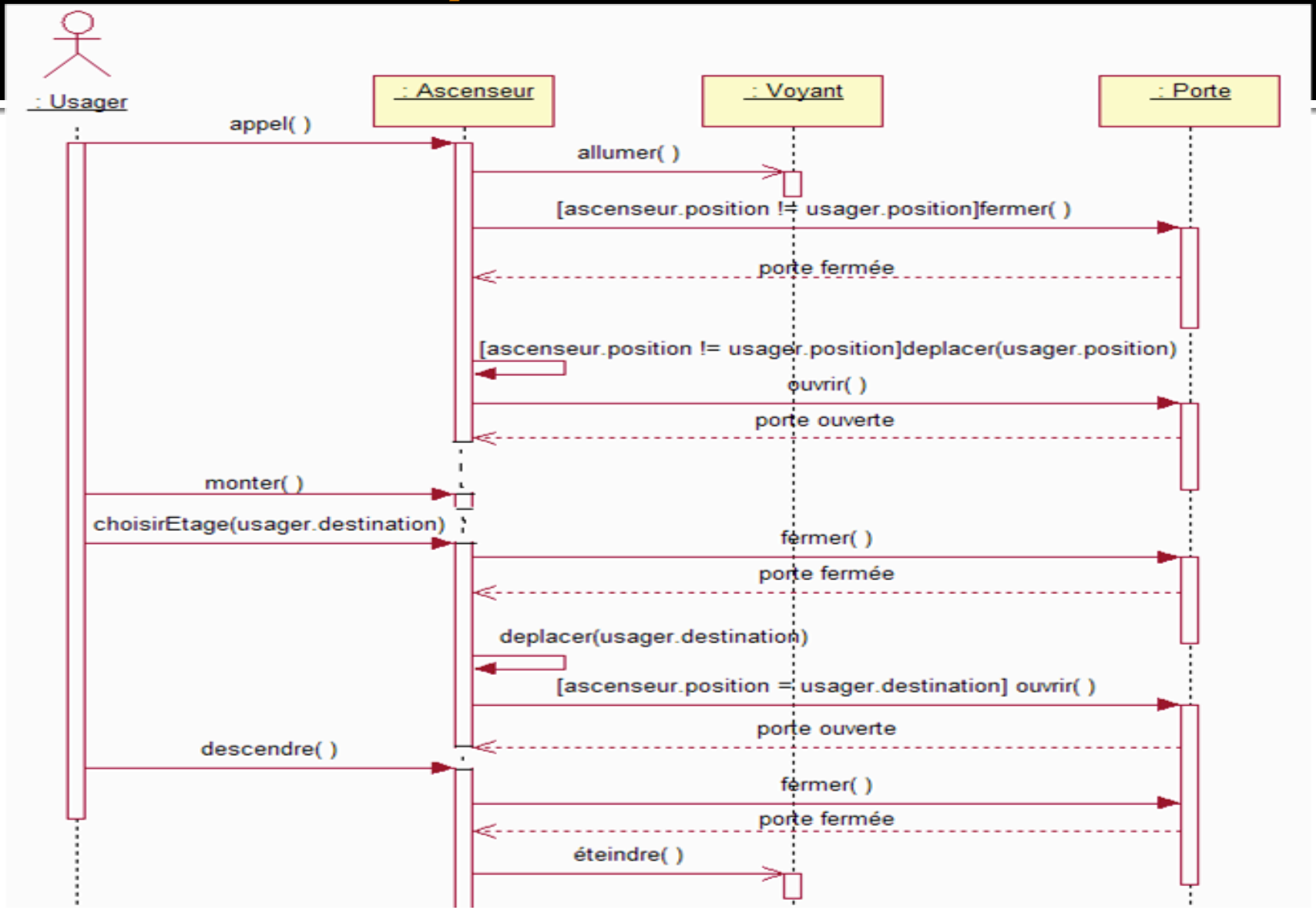
- En plus de objets participants (UML1), on ajoute
  - interfaces
    - pour spécifier quelle interface participe à l'interaction (la classe est peu importante)
  - classes
    - pour utiliser une méthode de classe
- Représentation polymorphisme / classe abstraite

Equivalence entre diagrammes de séquence et de communication

Exercice : dessiner le diagramme de communication correspondant

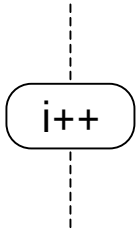


Autre exemple

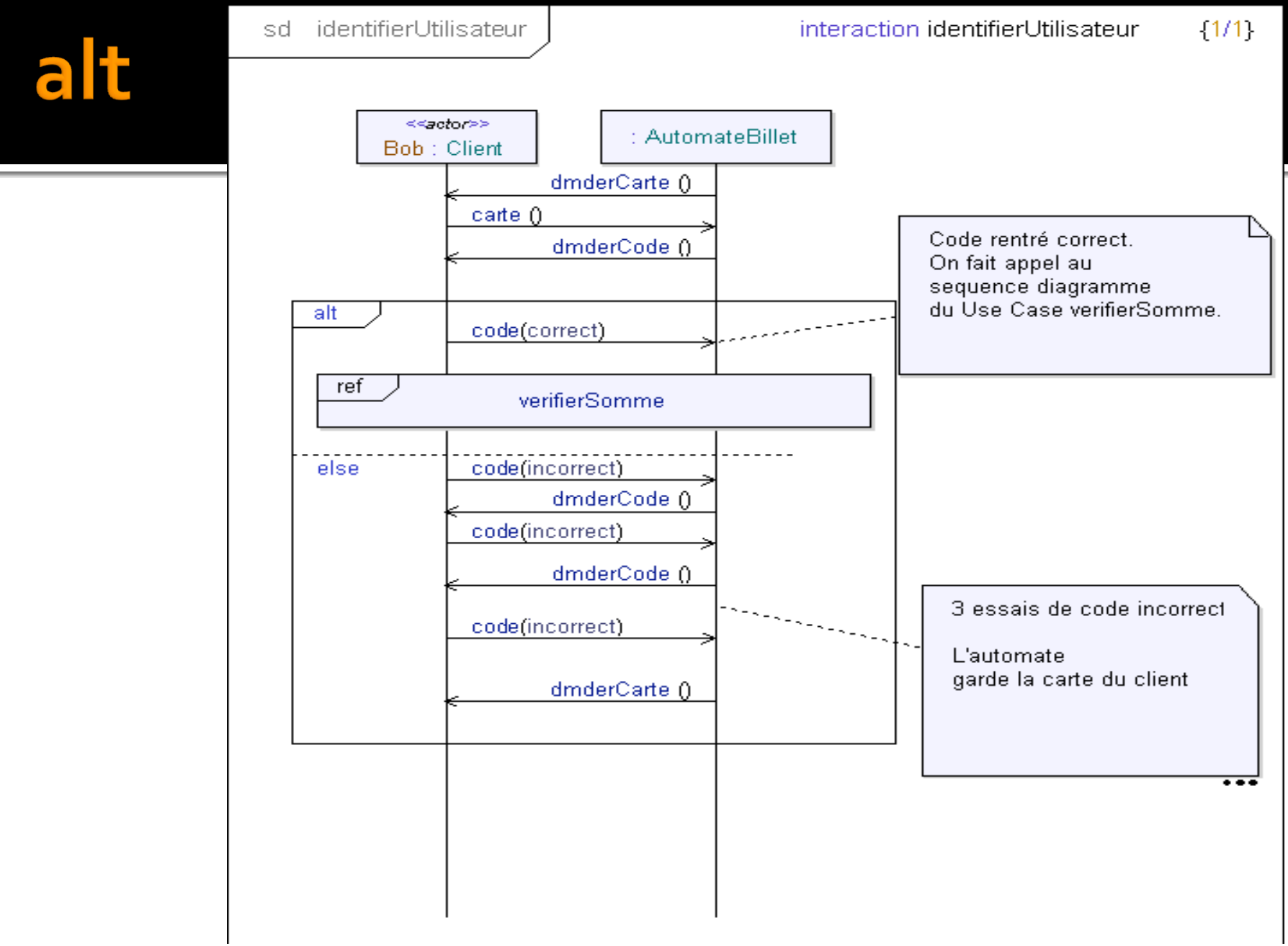


Cadre d'interaction

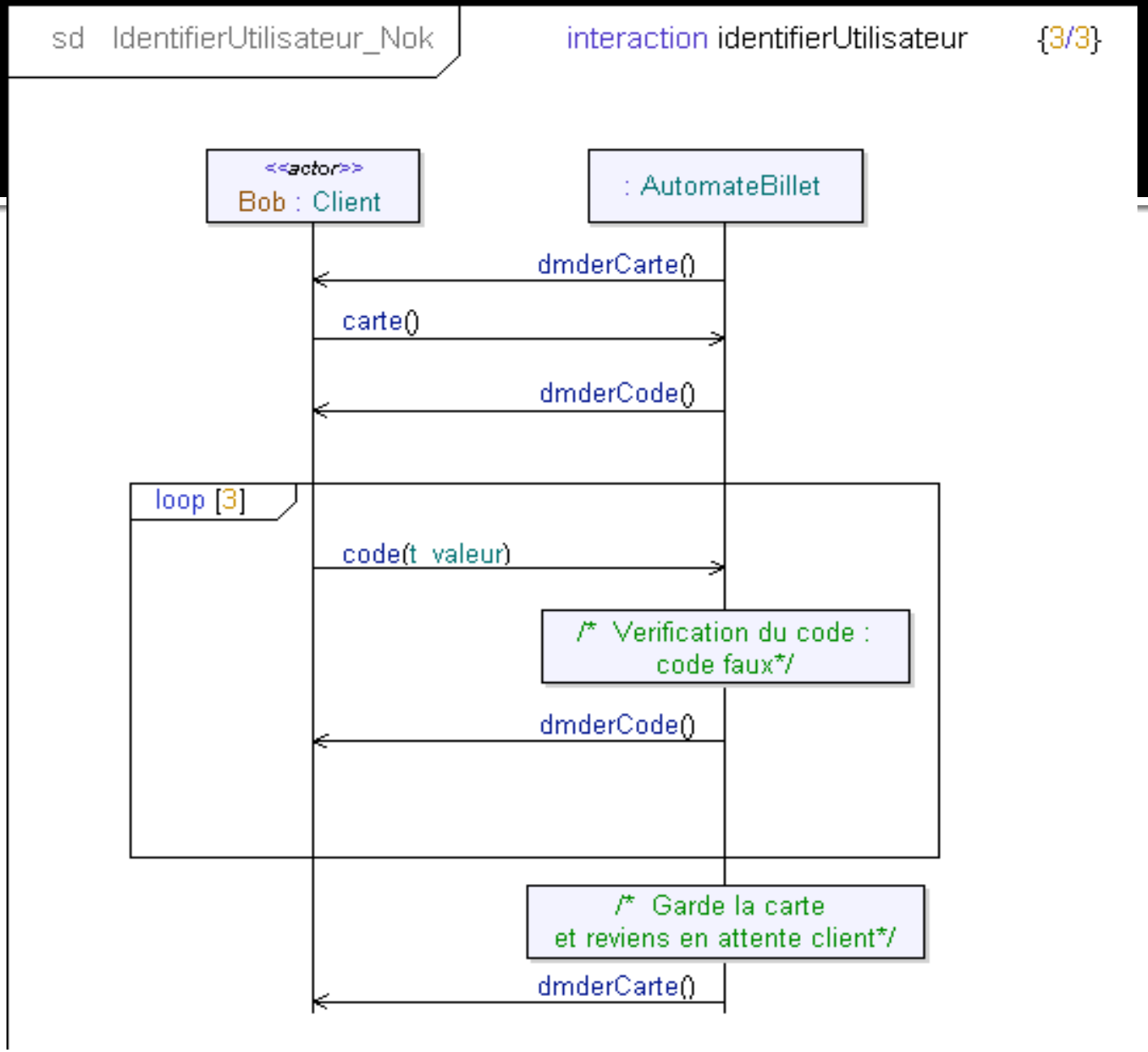
- Cadre nommé par un opérateur qui entoure un fragment critique du DS
  - alt
    - fragment alternatif, conditions dans les gardes
  - loop
    - fragment à répéter tant que la condition de garde est vrai
    - notion de boîte d'action avec itérateur
  - opt
    - fragment optionnel exécuté si la garde est vraie
  - par
    - fragments qui s'exécutent en parallèle
  - region
    - region critique dans laquelle un seul thread doit s'exécuter
  - ref
    - passage à un autre diagramme de séquence
- Attention
  - ne pas représenter des algorithmes : trop compliqué



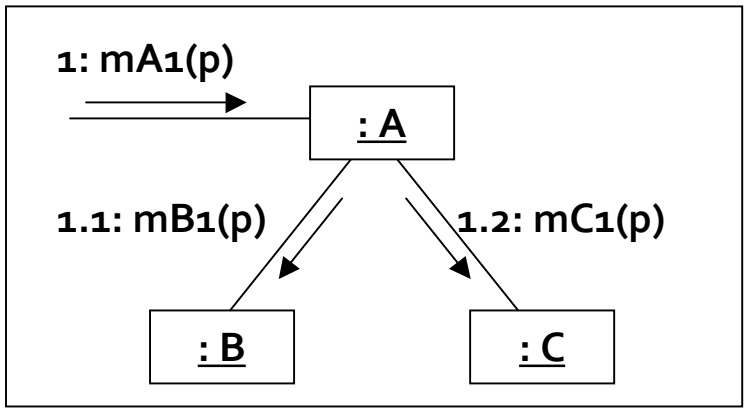
alt

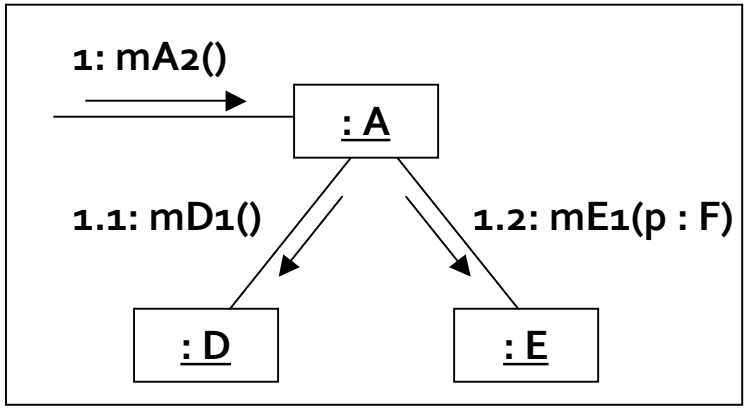

sd identifierUtilisateur
interaction identifierUtilisateur {1/1}
<<actor>>
Bob : Client
: AutomateBillet
dmderCarte ()
carte ()
dmderCode ()
Code rentré correct.
On fait appel au
sequence diagramme
du Use Case verifierSomme.
alt
code(correct)
ref
verifierSomme
else
code(incorrect)
dmderCode ()
code(incorrect)
dmderCode ()
code(incorrect)
dmderCarte ()
3 essais de code incorrect
L'automate
garde la carte du client

loop

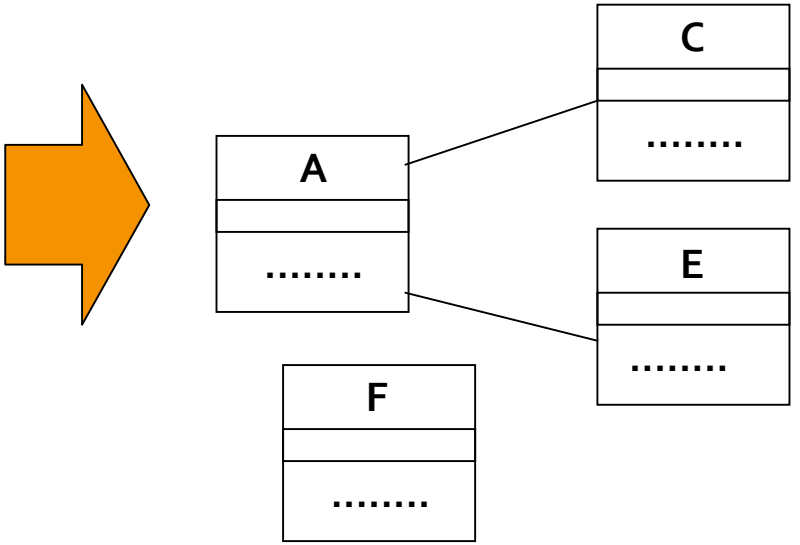

sd IdentifierUtilisateur_Nok
interaction identifierUtilisateur {3/3}
<<actor>>
Bob : Client
: AutomateBillet
dmderCarte()
carte()
dmderCode()
loop [3]
code(t valeur)
/* Verification du code : code faux*/
dmderCode()
/* Garde la carte et reviens en attente client*/
dmderCarte()

Déduire structure et responsabilité des diagrammes d'interaction



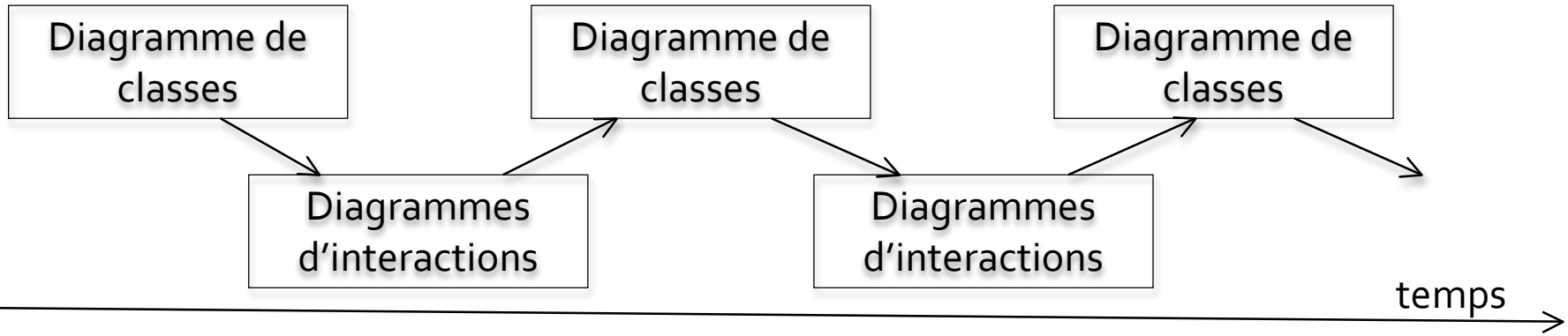


- Liens
  - ➜ associations
- Messages
  - ➜ opérations
  - ➜ dépendances



Co-conception des classes et des interactions

- Les objets utilisés dans les interactions pour réaliser les scénarios proviennent
  - des classes déjà décrites dans le diagramme de classes
  - des besoins nouveaux en objets pour l'interaction spécifique
- A partir des diagrammes d'interaction, on complète le diagramme de classes
  - précisions (attribut, méthodes)
  - nouvelles classes
- etc.
- On essaye de réaliser tous les scénarios en convergeant vers un diagramme de classes stables

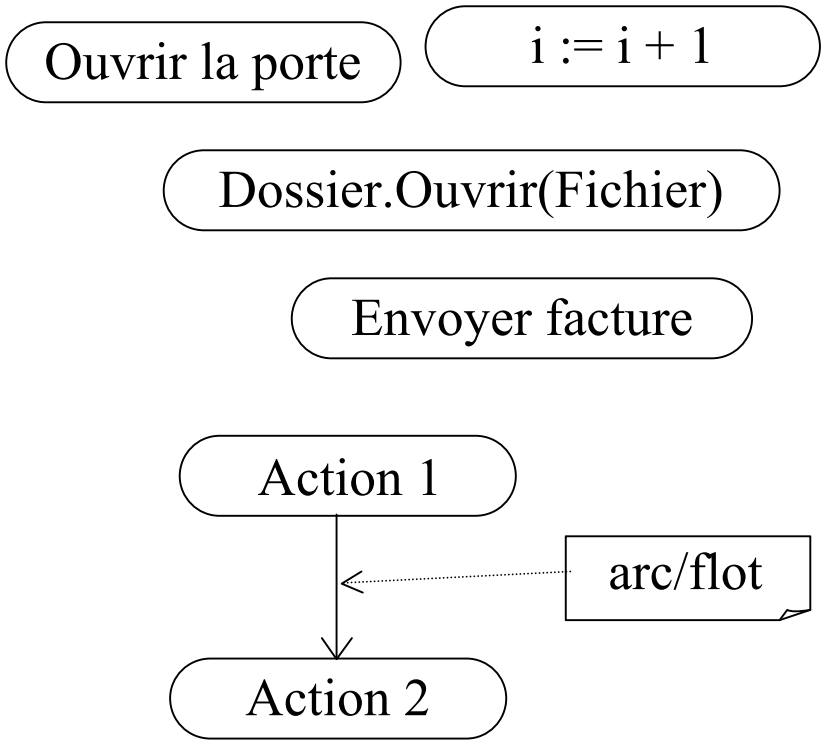


Plan

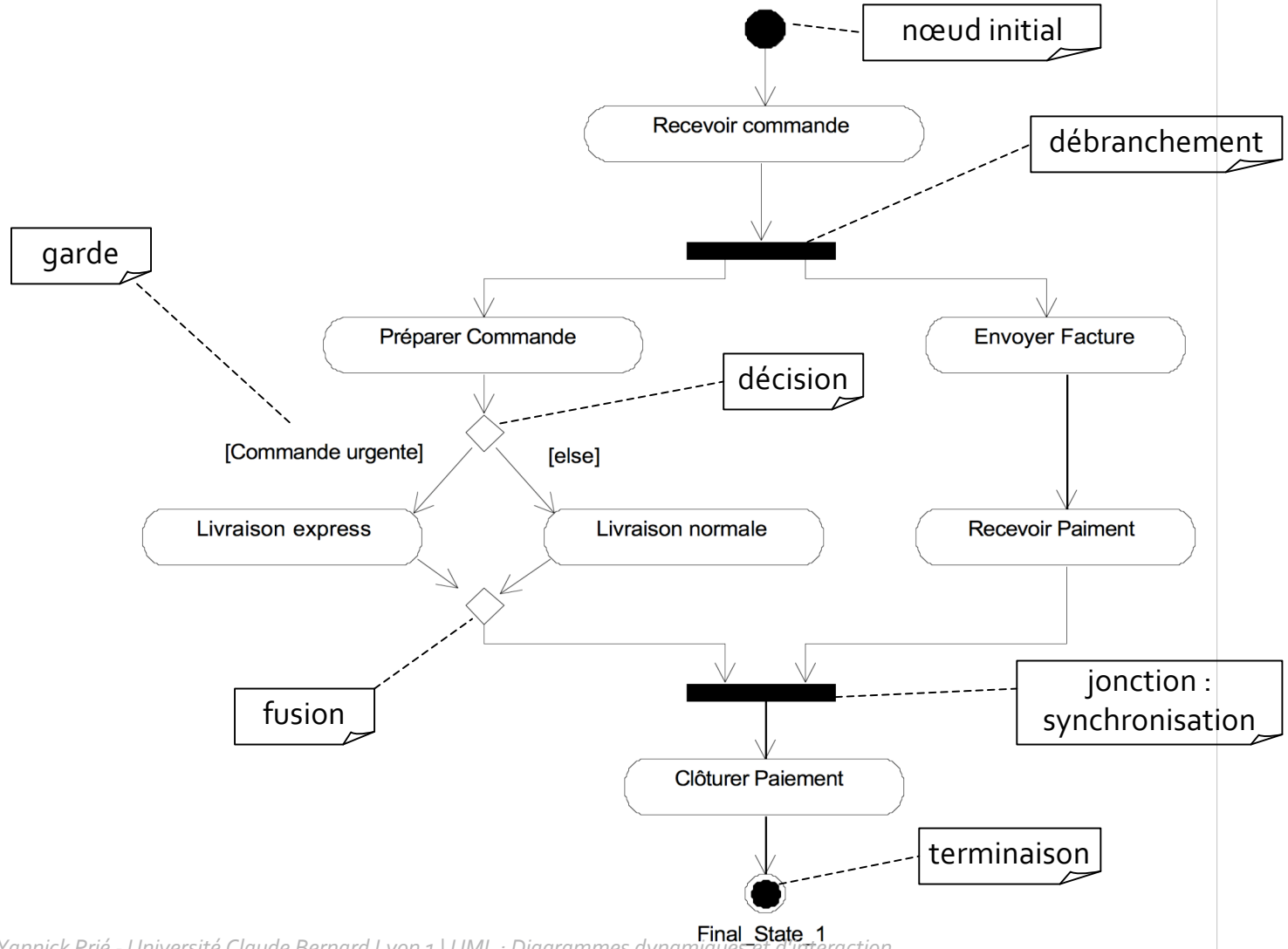
- Diagrammes d'interaction
  - diagrammes de séquences
  - diagrammes de communication
- **Diagrammes d'activité**
- Diagrammes de machines d'états
- Autres diagrammes UML
- Autres diagrammes non UML

Diagrammes d'activité

- Objectif
  - présenter les activités séquentielles d'un processus
  - activité = suite d'actions
- Action
  - travail à réaliser
  - nœud du graphe
- Transition
  - contrainte d'enchaînement
  - relation du graphe
- Raffinements
  - débranchements / jointures
  - décisions / fusions
  - entrée / terminaison
  - ressources utilisées (objets)



Exemple


nœud initial
Recevoir commande
débranchement
garde
Préparer Commande
Envoyer Facture
décision
[Commande urgente]
[else]
Livraison express
Livraison normale
Recevoir Paiment
fusion
jonction : synchronisation
Clôturer Paiement
terminaison
Final_State_1

Petit exercice à faire en classe

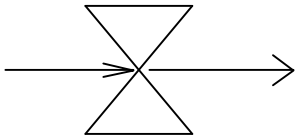
- Modéliser les activités autour d'un enseignement de l'UFR informatique.

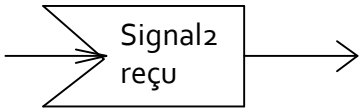
Diagrammes d'activité pour modéliser …

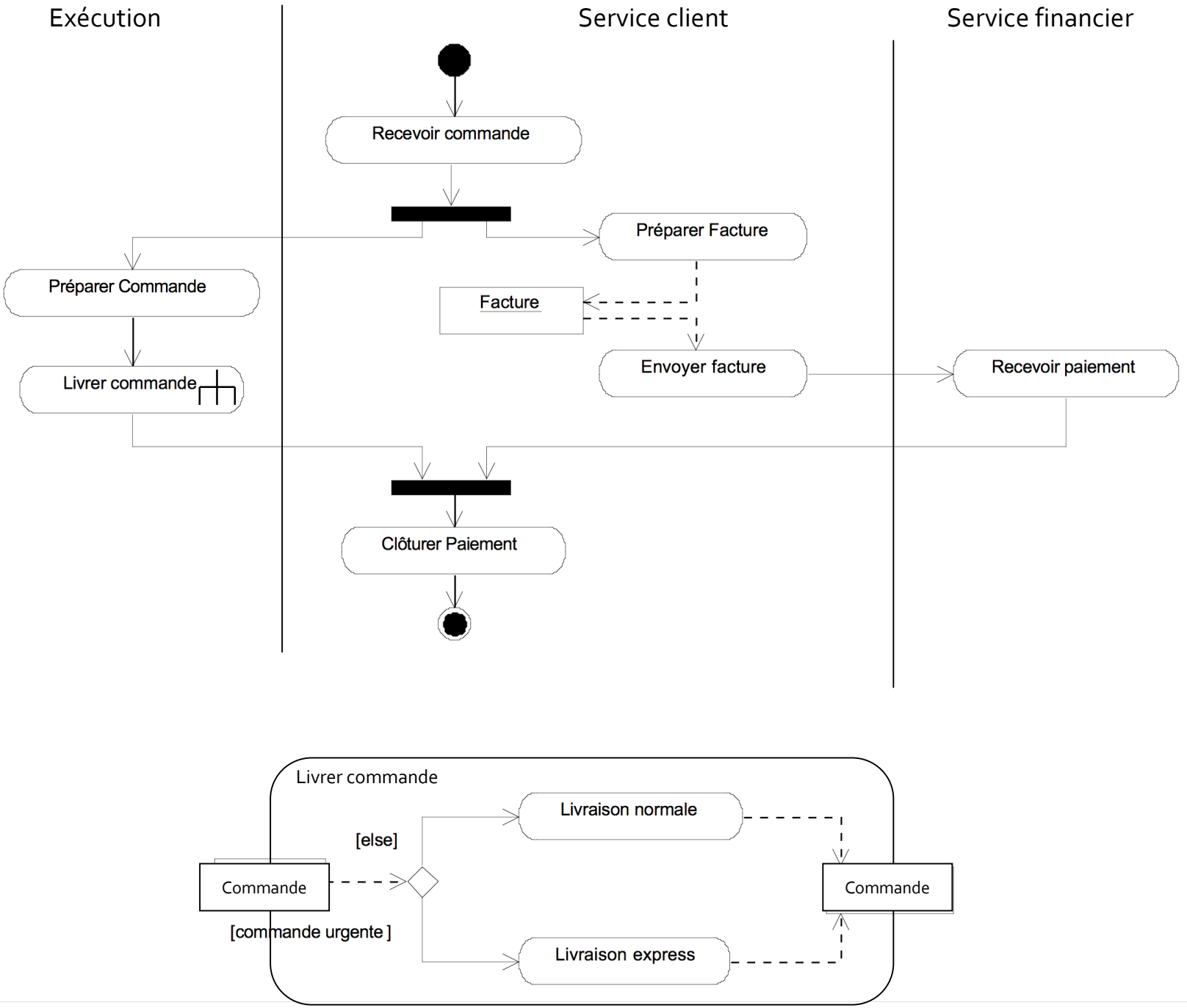
- Les processus métier de l'organisation
  - qui fait quoi, où
  - les enchaînement d'activité (workflow)
- Les flots de données
  - DFD (Data Flow Diagram) en UML
- La logique procédurale
  - algorithmes complexes, parallèles
  - organisation séquentielle globale des activités de plusieurs objets
    - vs. diag. machines d'états : un objet

Diagrammes avancés

- Actions liées à des signaux
  - délai
  - envoi / réception
- Utilisation d'objets
  - en entrée ou sortie d'action
- Partitions (UML1 : swimlanes, travées)
  - montrer les responsabilités au sein du mécanisme ou d'une organisation
- Décomposition des actions
  - appeler une sous-activité (un autre diagramme d'activité) dans une action

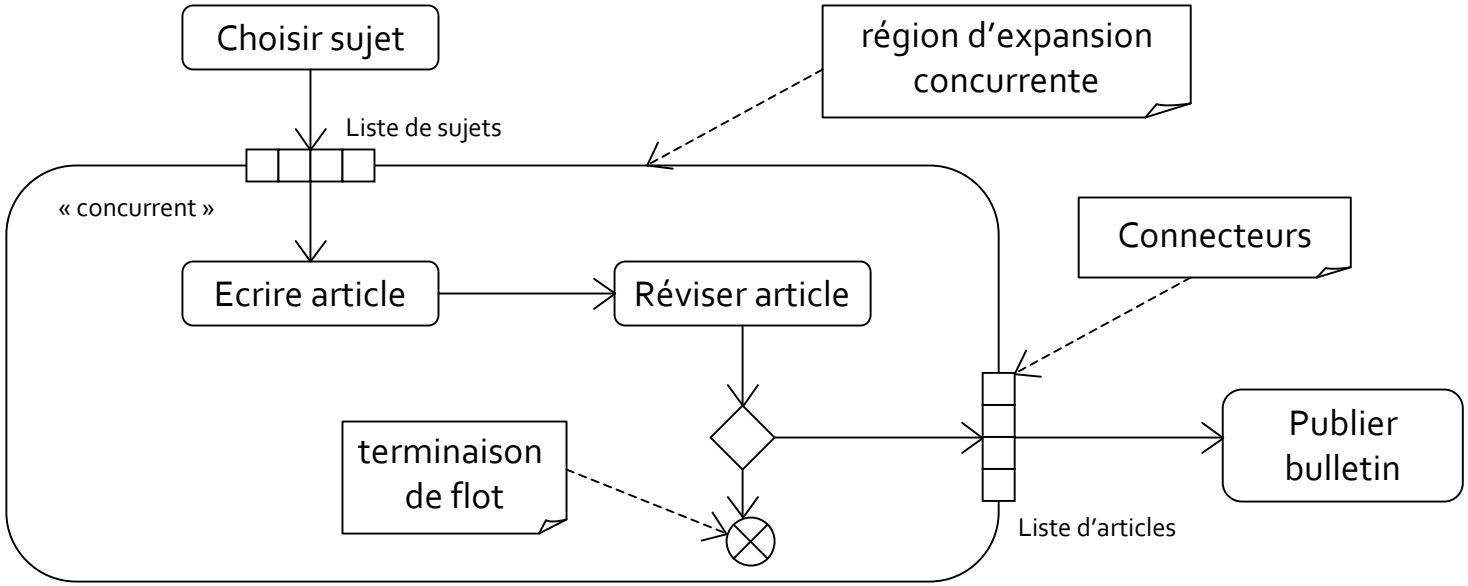





Exécution
Service client
Service financier
Recevoir commande
Préparer Facture
Préparer Commande
Facture
Livrer commande
Envoyer facture
Recevoir paiement
Clôturer Paiement
Livrer commande
Livraison normale
[else]
Commande
Commande
[commande urgente]
Livraison express

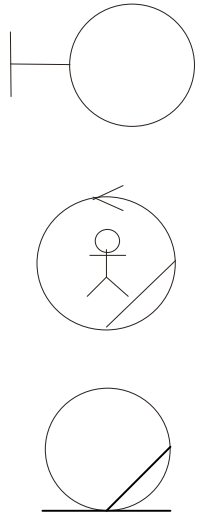
Connecteurs, régions d'expansion, terminaison de flots (UML2)

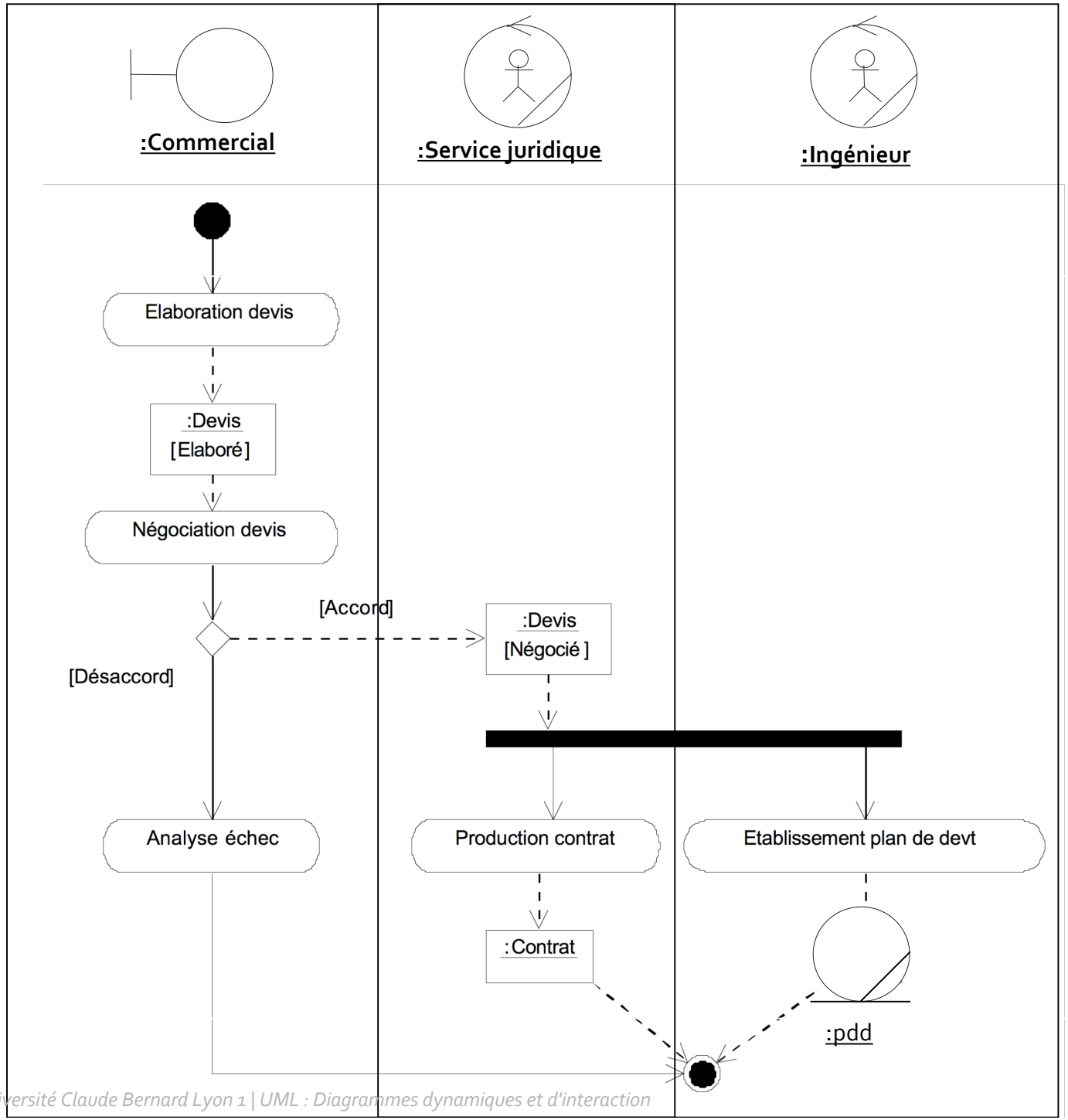
- Connecteurs
  - cf. objets paramètres entrée/sortie actions
- Régions d'expansion
  - actions qui se passent pour plusieurs éléments de même type (itératif ou concurrent)



Modélisation de processus métier

- Modéliser le fonctionnement d'une organisation
- Trois stéréotypes d'objets
  - *Case worker*
    - interaction avec l'ext. de l'entreprise
  - *Internal worker*
    - travail à l'intérieur de l'entreprise
  - *Entity*
    - objet passif : stockage d'informations
- Partitions / couloirs d'activité



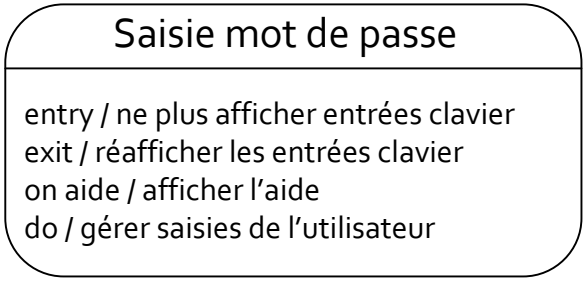

:Commercial
:Service juridique
:Ingénieur
Elaboration devis
:Devis
[Elaboré]
Négociation devis
[Accord]
[Désaccord]
:Devis
[Négocié]
Analyse échec
Production contrat
Etablissement plan de devt
:Contrat
:pdd

Plan

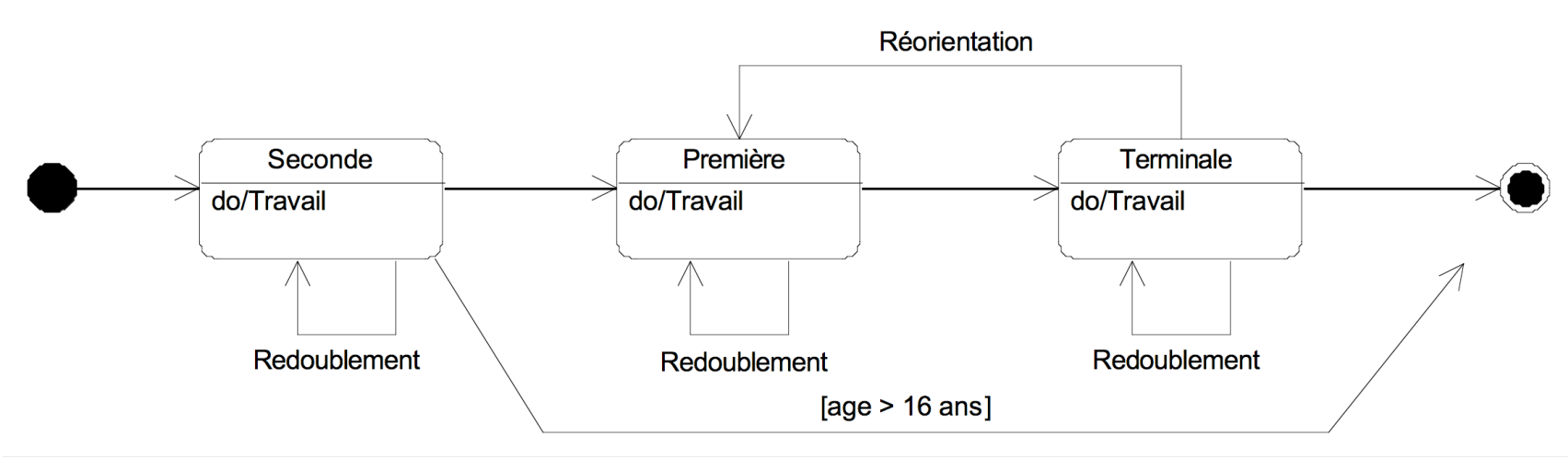
- Diagrammes d'interaction
  - diagrammes de séquences
  - diagrammes de communication
- Diagrammes d'activité
- **Diagrammes de machines d'états**
- Autres diagrammes UML
- Autres diagrammes non UML

Diagrammes de machines d'états

- Abstraction des comportements possibles pour une classe
  - automate à états finis décrivant les chemins possibles dans le cycle de vie d'un objet
- Etat d'un objet
  - situation nommée d'un objet qui répond à certaines conditions de durée et de stabilité
  - Ex. : activité continue (sonnerie), tâche de fond (pagination), attente, suite d'actions…
- Transition entre états
  - réponse de l'objet dans un certain état à l'occurrence d'un événement
    - passage d'un état à un autre sur événement + condition respectée,
    - action à exécuter



Exemple de diagramme de machines d'états



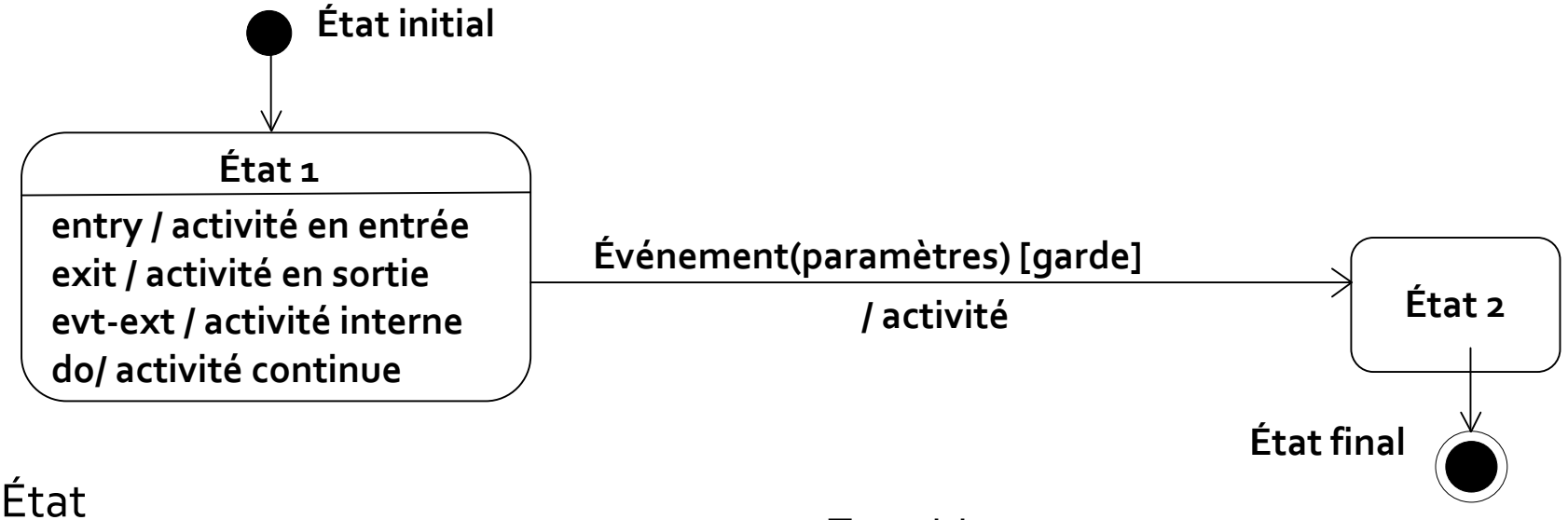
Petit exercice à faire en classe

- Tracer un diagramme de machines d'états pour un objet « TP-Etudiant ».

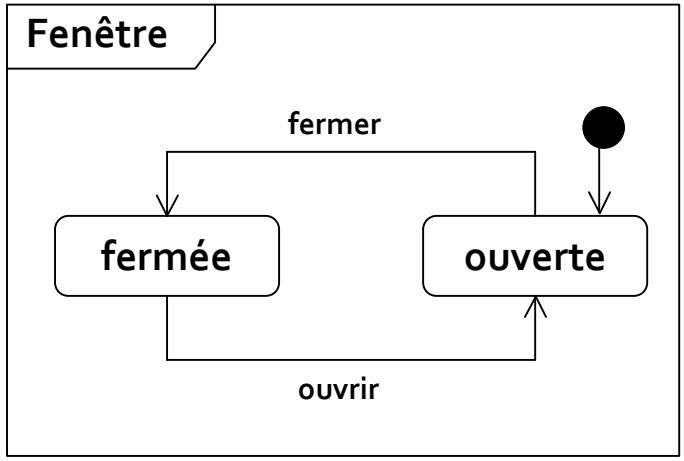
Utilisation

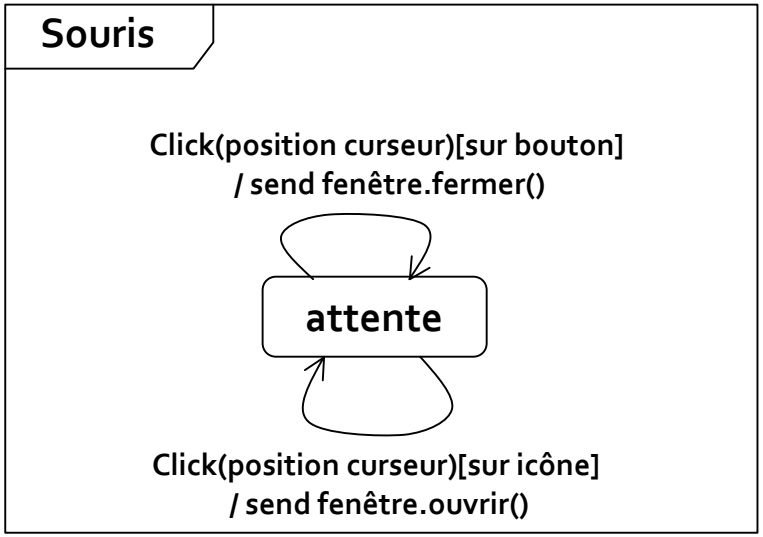
- Pour se concentrer sur le fonctionnement d'une classe
  - décrire / fixer le comportement concret de la vie d'une objet lié à un ou plusieurs scénarios
- Pour les classes complexes
  - objets réactifs complexes (objets métier…)
  - protocole et séquences légales (sessions…)
  - en général pas plus de 10% des classes d'une application
    - plus en télécommunication / moins en informatique de gestion
- Larman
  - navigation dans un site web, IHM
    - enchaînement de pages/fenêtres

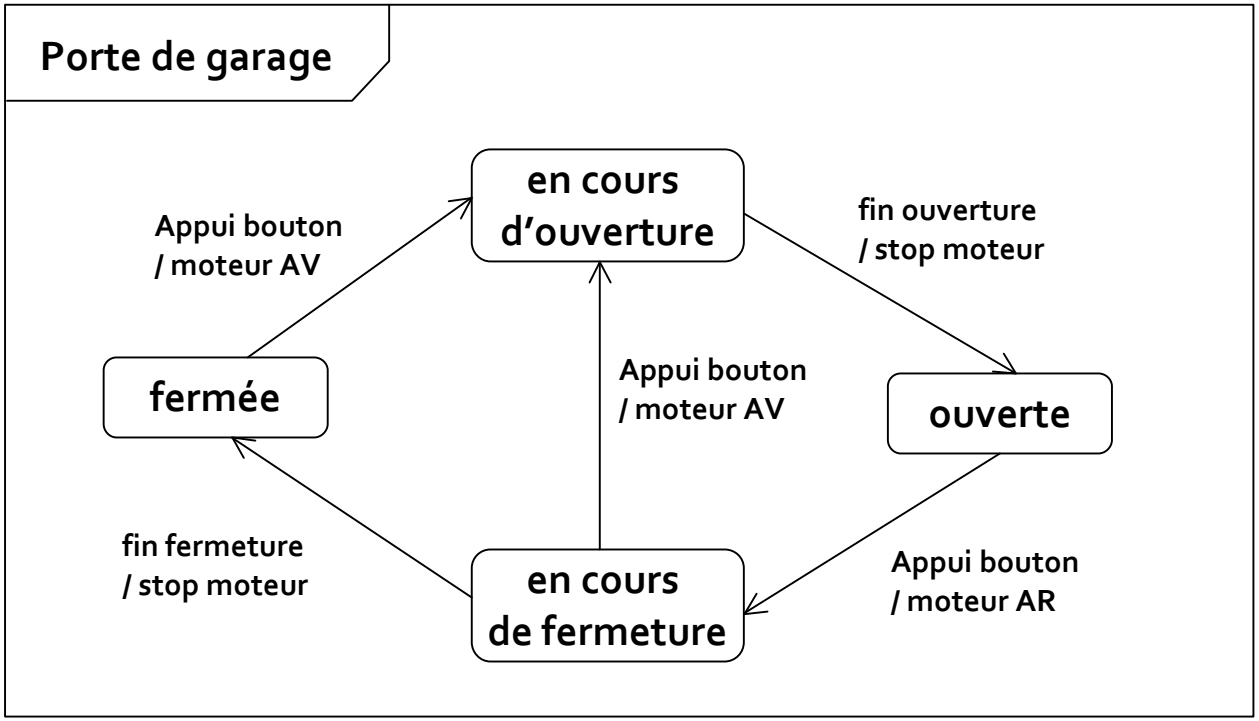
Syntaxe générale



- État
  - activité continue : peut être interrompue
    - do / activité
  - activités internes instantanées
    - auto-transition sur événement extérieur, instantanée, qui ne nécessite pas changement d'état
    - deux activités spéciales : sur entrée et sortie
- Transition
  - tout est facultatif mais absence d'événement rare
  - événements
    - résultants de messages entre objets
    - internes : when(maximum atteint)
    - temporels : after(3 jours)
  - activité classique : envoyer un message à une cible
    - send cible.message(arguments)

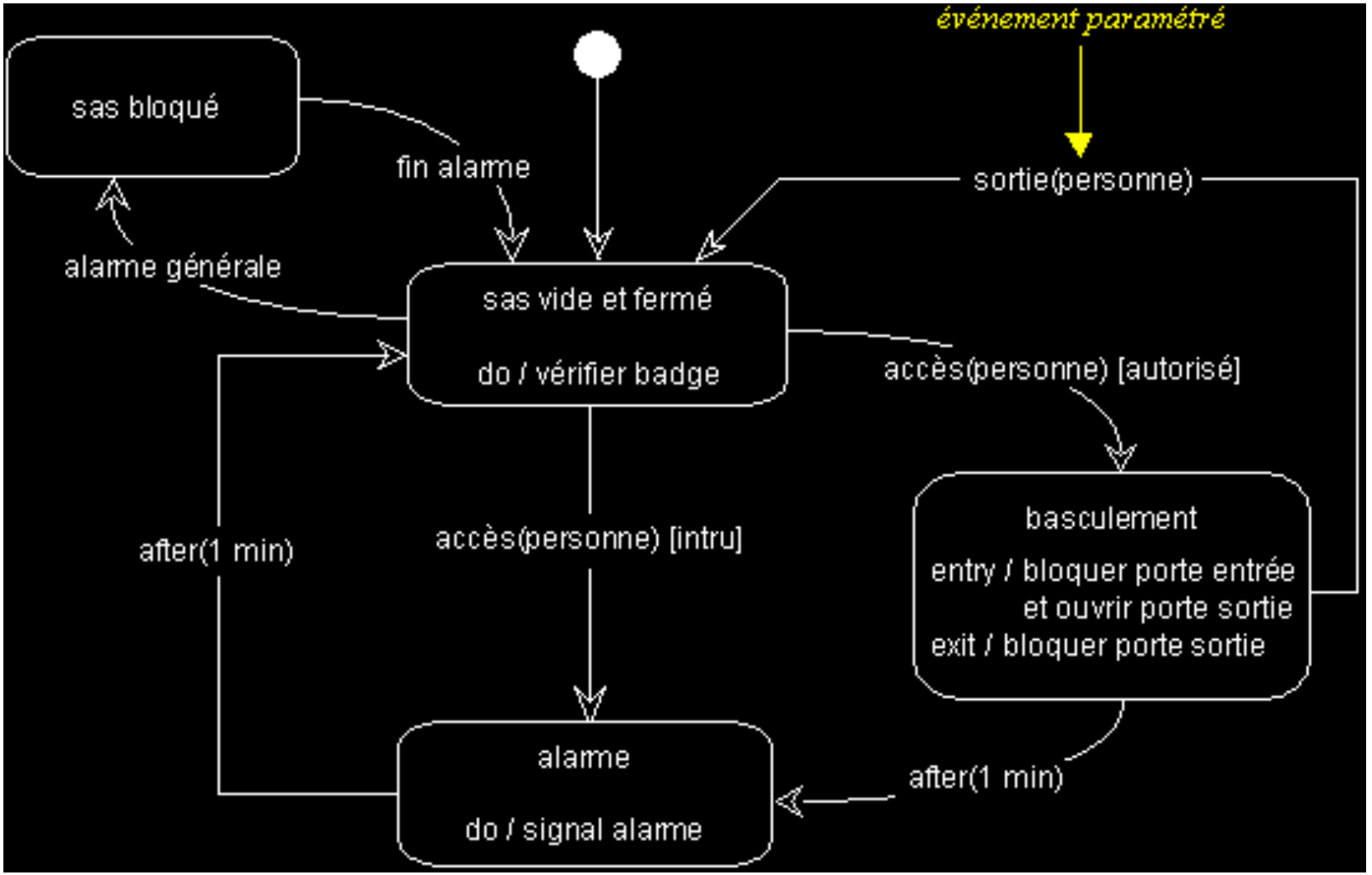

Fenêtre
fermer
fermée
ouverte
ouvrir


Souris
Click(position curseur)[sur bouton]
/ send fenêtre.fermer()
attente
Click(position curseur)[sur icône]
/ send fenêtre.ouvrir()


Porte de garage
en cours
d'ouverture
Appui bouton
/ moteur AV
fin ouverture
/ stop moteur
fermée
Appui bouton
/ moteur AV
ouverte
fin fermeture
/ stop moteur
en cours
de fermeture
Appui bouton
/ moteur AR

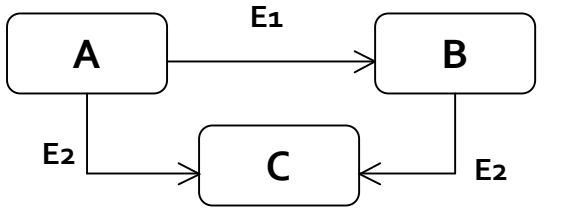
http://uml.free.fr/cours/p20.html

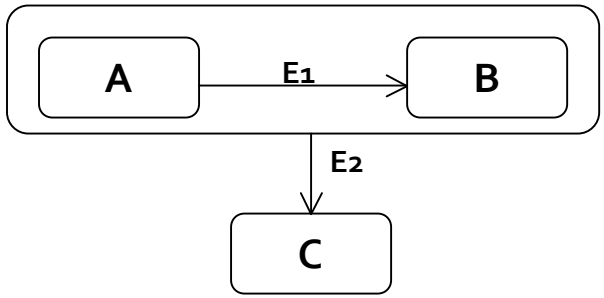
Autre exemple

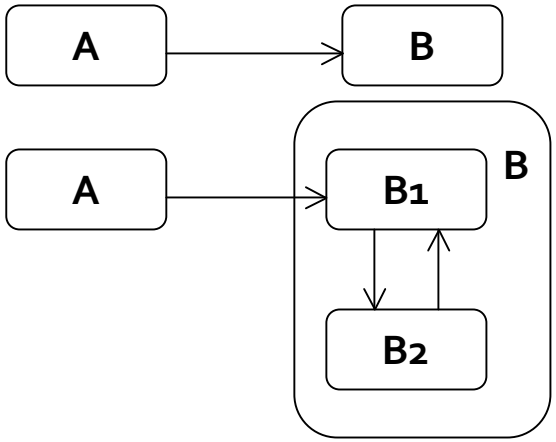


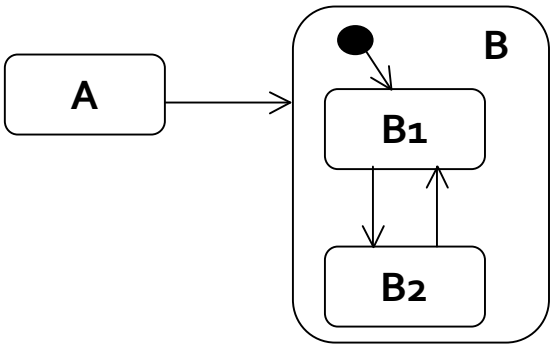
Super-états / états composites

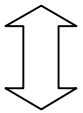
- Pour factoriser un comportement
  - transitions déclenchées par le même événement, conduisant au même état
- Transition interne
  - couple événement / activité sans effet sur l'état courant
  - permet de ne pas « réinitialiser » l'état en revenant à l'état de départ
  - se note en haut du super état



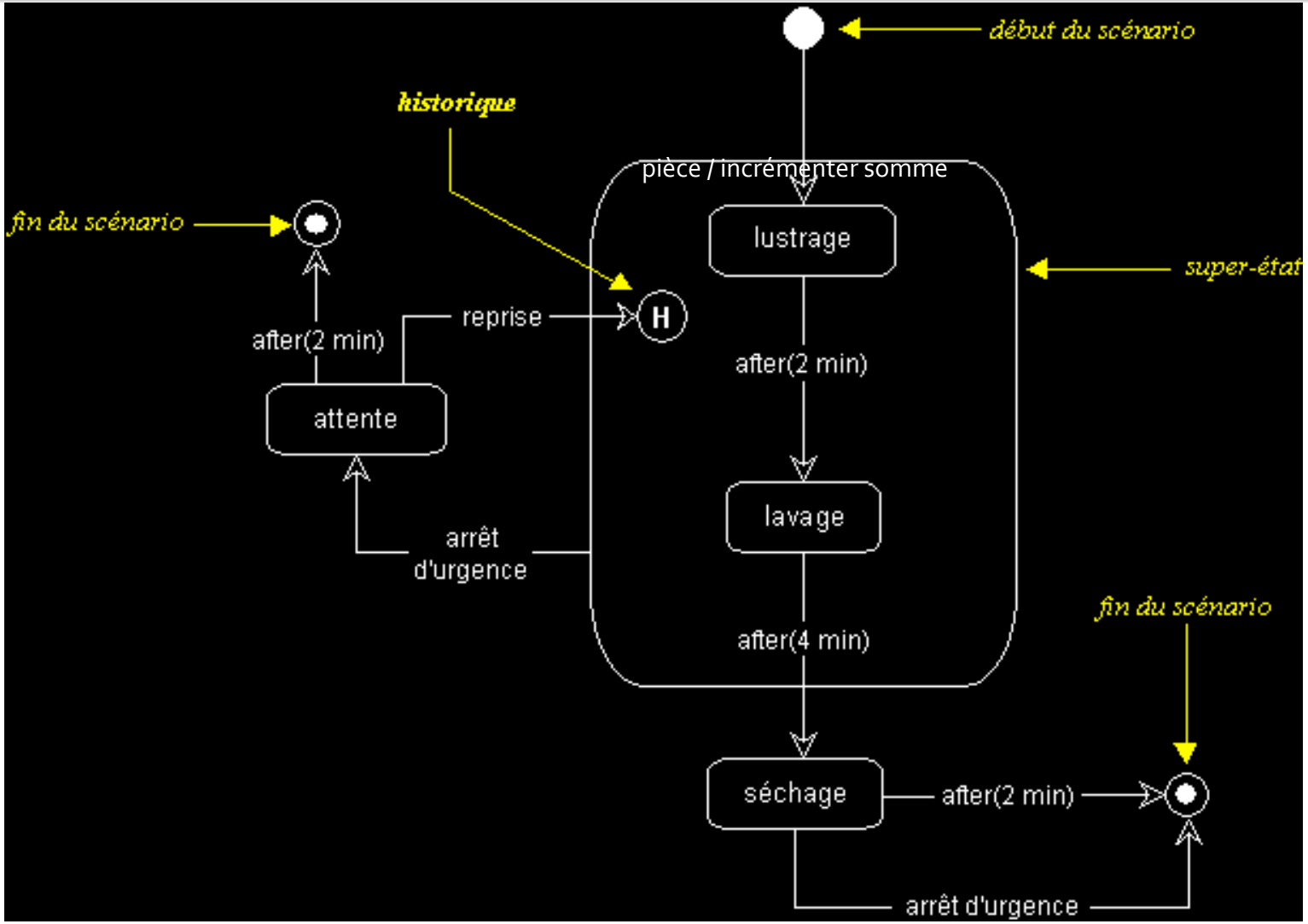






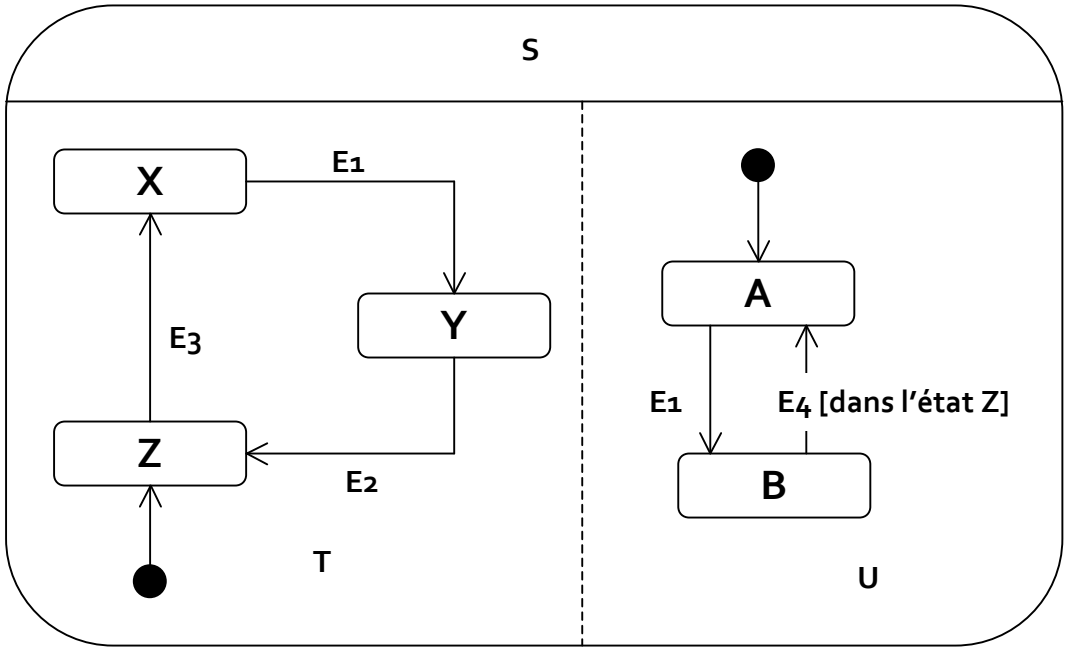


Exemple super-état



États concurrents

- Pour décomposer des états complexes
- Exercice : trouver le diagramme d'état « à plat » équivalent



Implémentation des états

- Utilisation de case … switch
  - déconseillé
- Utilisation du pattern état
  - arborescence de classes états
  - délégation de la gestion de l'état
- Tables d'états
  - représentation tabulaire du diagramme
    - état source, cible, événement, garde, procédure à exécuter
  - permet de « paramétrer » le comportement de la classe

Plan

- Diagrammes d'interaction
  - diagrammes de séquences
  - diagrammes de communication
- Diagrammes d'activité
- Diagrammes de machines d'états
- **Autres diagrammes UML**
- Autres diagrammes non UML

Vue d'ensemble des interactions

- Mixte diagramme activité / diagrammes de séquences
  - les actions sont remplacées par des diagrammes de séquence
- Utilisé ?

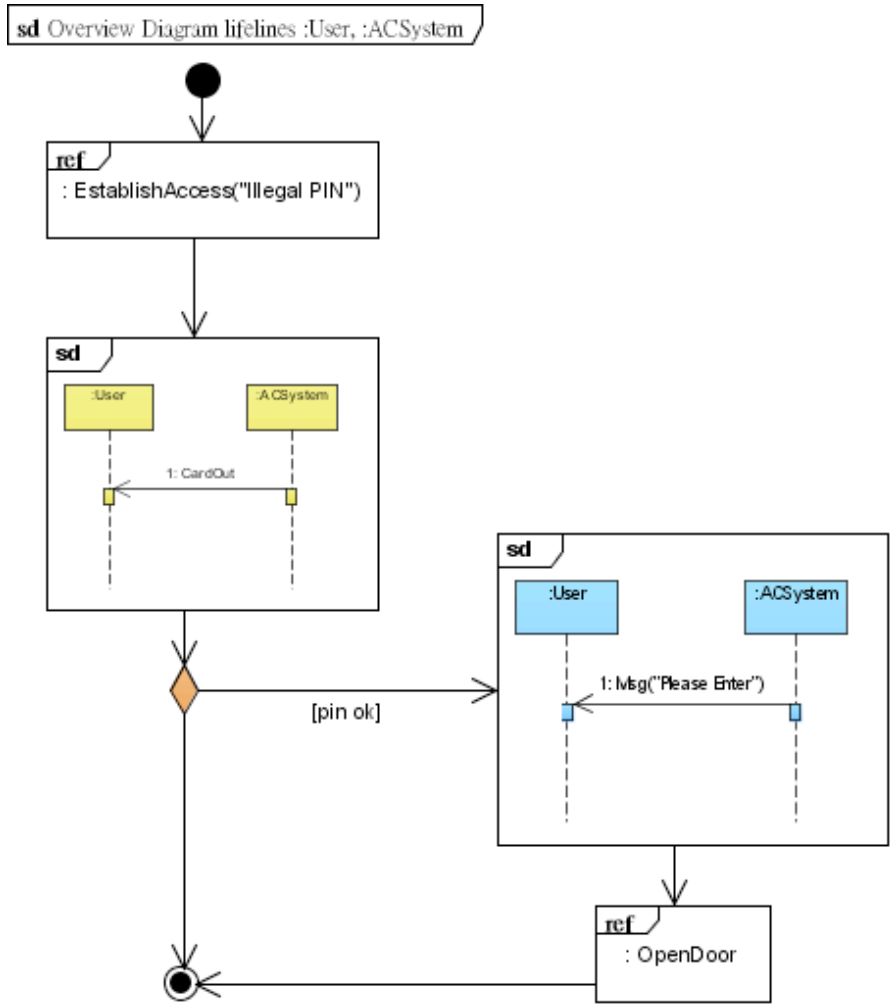
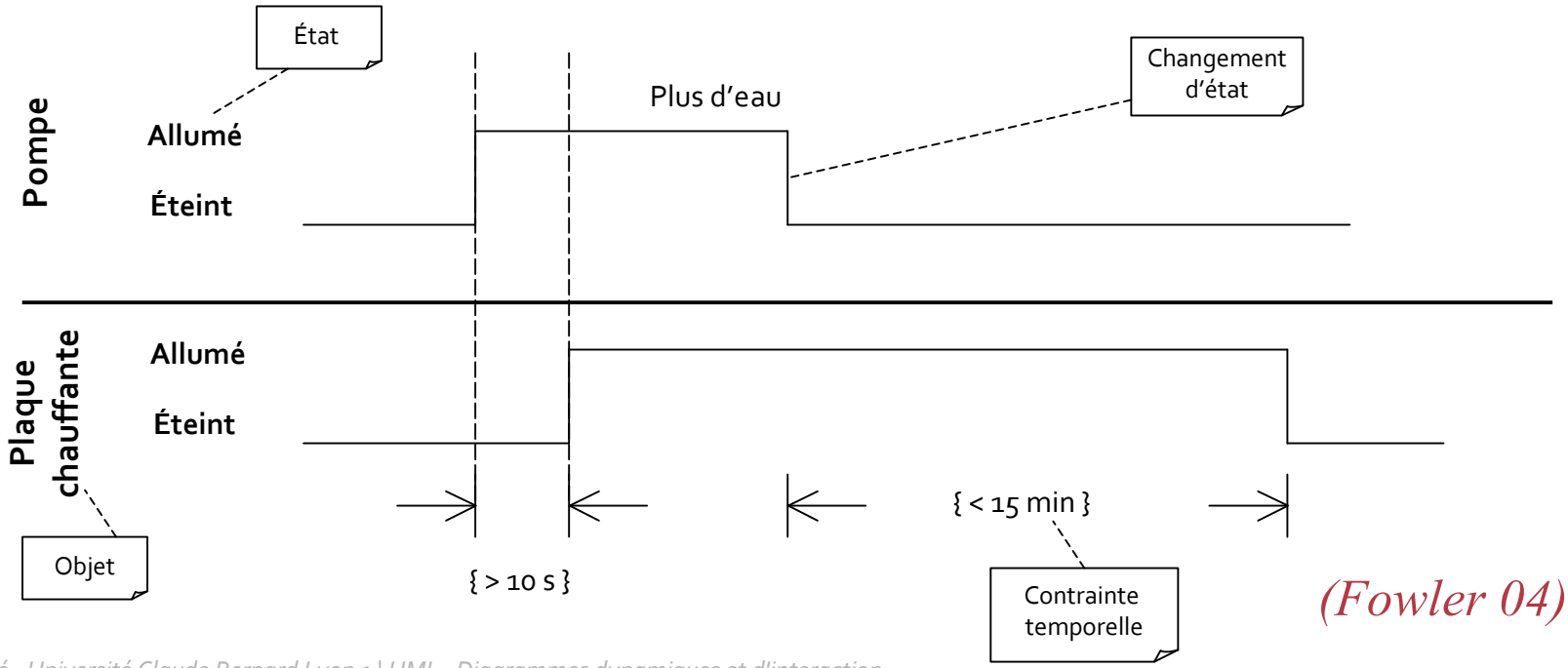


Diagramme de timing

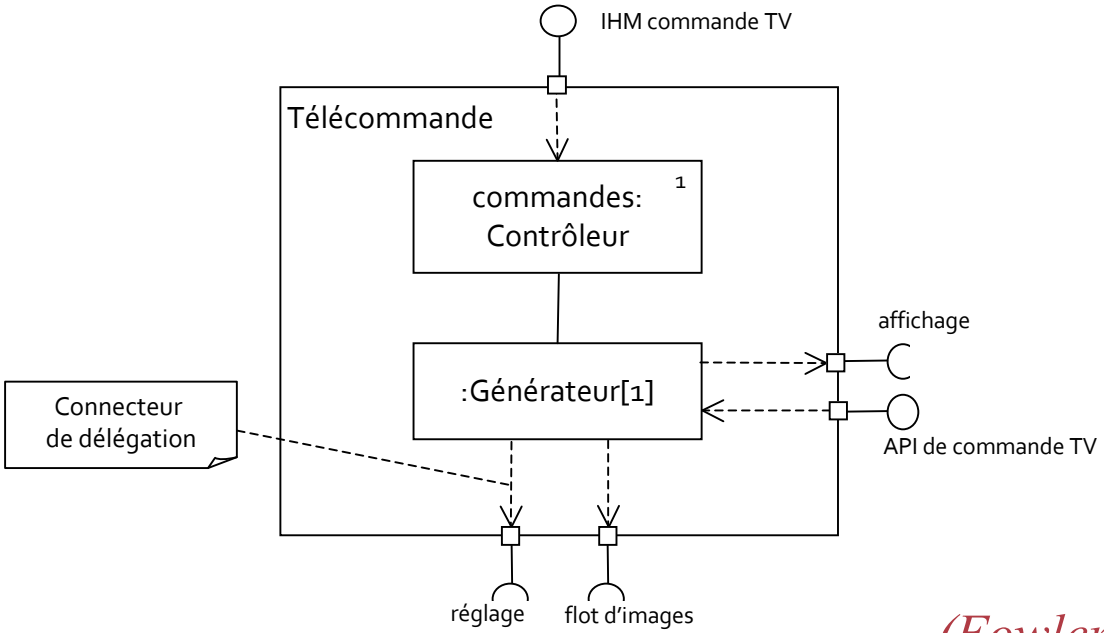
- Interactions avec focus sur les changements d'états d'objets et les contraintes temporelles associées
  - ligne de vie horizontale
- Utilisé surtout dans les applications temps réel



(Fowler 04)

Structures composites (classeurs structurés)

- Pour décomposer structurellement une classe
  - parties, connecteurs
- Montrer les réalisations / utilisations d'interfaces
  - ports, interfaces
- Adaptés pour les composants



(Fowler 04)

Diagrammes de collaborations

- Non officiels dans UML2, se rapprochent des diagrammes de structure composite
- Permettent de présenter les éléments impliqués dans une collaboration, et le rôle qu'ils y jouent
  - fixer les éléments et les rôles pour les diagrammes d'interaction
- En théorie utilisés pour représenter des patterns

Plan

- Diagrammes d'interaction
  - diagrammes de séquences
  - diagrammes de communication
- Diagrammes d'activité
- Diagrammes de machines d'états
- Autres diagrammes UML
- **Autres diagrammes non UML**

Diagrammes de contexte (Roques, 2004)

- Diagramme de contexte statique
  - diagramme de classe
    - une classe système
    - tous les acteurs autour
- Diagramme de contexte dynamique
  - diagramme de communication qui résume les messages entre système et acteurs (pas de numérotation)
- Diagramme de contexte statique étendu
  - diagramme de contexte statique avec
    - Attributs et opérations de hauut niveau pour le système et les acteurs non humains
- Remarque
  - « diagrammes de classes avec messages »
    - diagramme de classe avec résumé des messages entre classes

Diagramme de flux d'écrans informel
(Fowler, 2004)

- Un rectangle par écran
- Des flèches pour la navigation
  - éventuellement un nom signifiant le lien

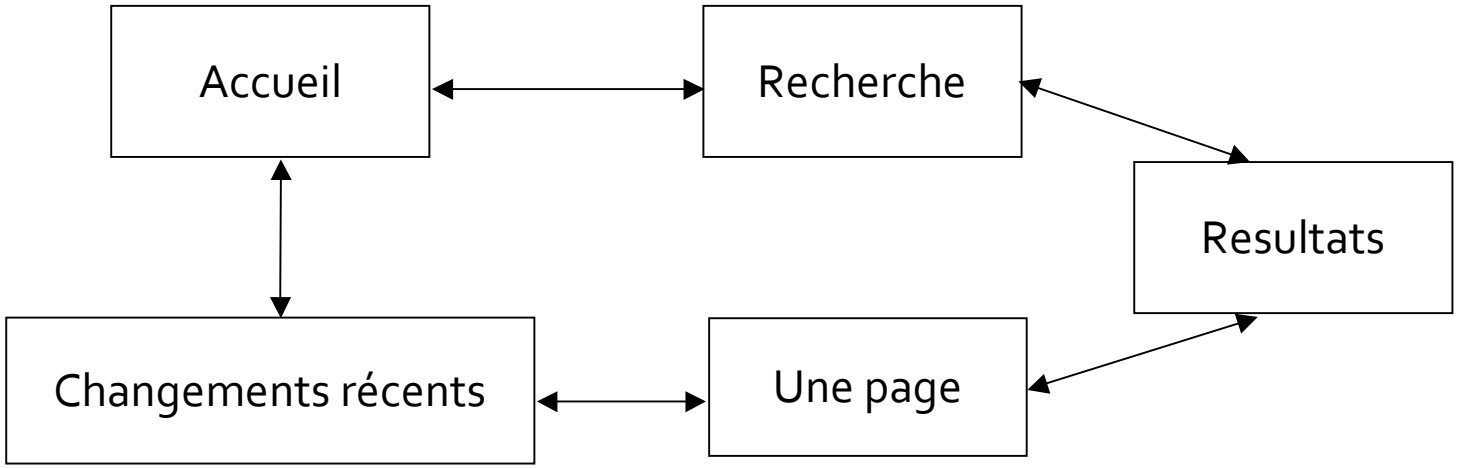


Table de décision (Fowler)

- Pour représenter des conditions logiques complexes
- Deux parties
  - conditions
  - conséquences

Client privilégié	X	X	O	O	N	N
Commande prioritaire	O	N	O	N	O	N
International	O	O	N	N	N	N
Prix	150	100	70	50	80	60
Alerte	oui	oui	oui			

Cartes CRC

Classe	
Responsabilite1	Coll1,2

- Classes - Responsabilités - Collaborateurs
  - à la base inventé pour l'enseignement
- Jouer des scénarios avec des cartes
  - 5-6 participants
- Une carte
  - nom de classe
  - tableau à deux colonnes
    - responsabilité de la classe : quelque chose d'un objet doit faire
    - collaborateurs : classes avec lesquelles il faut collaborer pour assurer la responsabilité
- Jeu
  - déterminer des classes de départ avec responsabilités évidentes
  - jouer les scénarios, ajouter les responsabilités, créer de nouvelles classes, etc.
- Voir par exemple
  - http://www.csc.calpoly.edu/~dbutler/tutorials/winter96/crc_b/

A suivre

- UML 4/4 : concepts avancés