Introduction aux méthodes d'analyse et de modélisation

Emmanuel ADAM

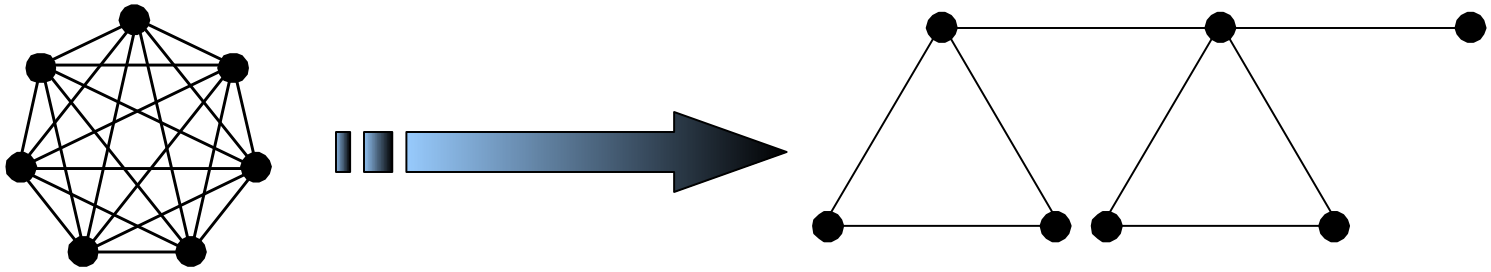
Institut des Sciences et Techniques de Valenciennes

Le Génie Logiciel, Pourquoi ?

- Apparu en 1968 pour
  - construire des logiciels fiables,
  - respecter les délais,
  - travailler en équipe,
  - décomposer la complexité,
  - assurer la qualité,
  - permettre la réutilisation,
  - ...

Le Génie Logiciel pour des problèmes complexes

- Ajouter un programmeur sans organisation ne fait que ralentir la mise en œuvre du projet.
  - Problème de nb de communications : (Loi de Brooks)



  - Problème de communication (compréhension)
    - malentendus, erreurs d'interprétation, ...

Quelques définitions

- le génie logiciel est l'art de ***spécifier***, de ***concevoir***, de ***réaliser***, et de faire ***évoluer***, avec des moyens et dans des délais raisonnables, des ***programmes***, des ***documentations*** et des ***procédures*** de ***qualité*** en vu d'utiliser un ordinateur pour résoudre certains problèmes.
  [MC Gaudel & co, 96]

- Le génie logiciel se caractérise par une approche ***rigoureuse*** et ***systématique*** de la construction de logiciels ne pouvant être maîtrisés par une seule personne.
  [JP Fournier 00]

Les étapes de développement

1 - Analyse des besoins

2 - Spécification

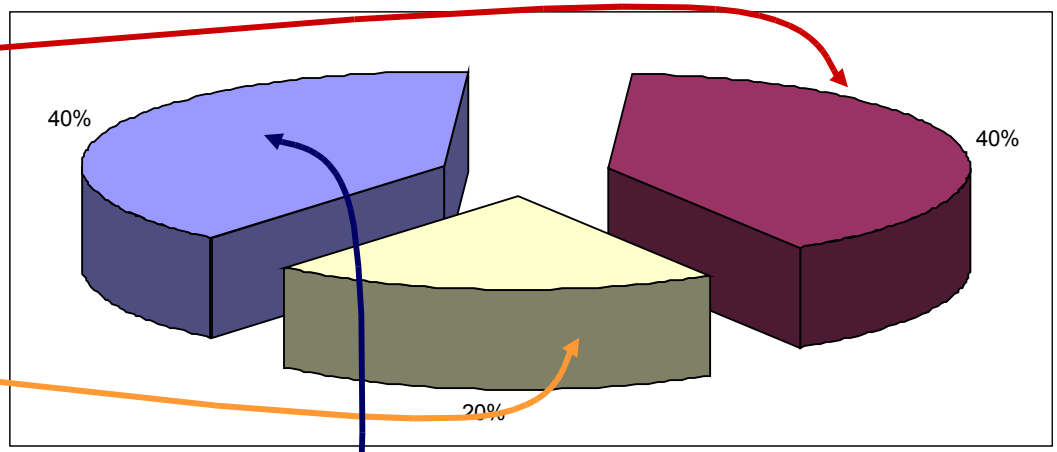
3 - Conception

4 - Codage

5 - Intégration

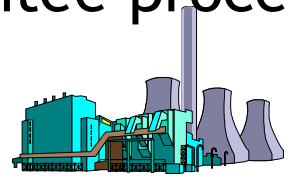
6 - Mise en œuvre

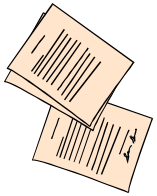
7 - Exploitation - Maintenance



Analyse des besoins

- Etude de faisabilité
- Type d'analyse
  - orientée processus, orientée rôle, orientée données





- Techniques d'acquisitions
  - entretiens, observations, questionnaires, ...

Spécification

(inspiré de IEEE Std 830 & [Jaulent 90])

- Première description du futur système (le quoi, Ce qu'il doit faire)
  - Les fonctions générales,
  - Les caractéristiques des utilisateurs,
  - Contraintes (développement, exploitation, maintenance),
  - Interfaces (homme-machine, logiciel/matériel, logiciel/logiciel),
  - Objets, entités constituant le système.
- Garder une trace des différentes spécifications (justification)
- Ne pas oublier le glossaire !
- Dans une annexe, placer les références aux documents utilisés
- Ne pas faire de choix d'implémentation à ce niveau...

Conception

- Augmenter la spécification pour se rapprocher de l'implémentation, du codage
- Conception Architecturale
  - Décomposer le système en sous-systèmes
  - Définir les interfaces, les liens entres les composants
- Conception détaillée
  - Détailler le fonctionnement des composants
    - définir quelques algos, la représentation des données, ...

Le codage et l'intégration

- Ne représente que 15 à 20% du temps dans un projet complexe et bien structuré
- Ne pas oublier de gérer les différentes versions des composants
- L'intégration nécessite une bonne définition préalables des interfaces entre composants

Les langages de programmation (1/2)

- Programmation impérative
  - programmation structurée,
  - basé sur des algos,
  - ex : fortran, pascal, c, ADA, …

- Programmation applicative
  - programmation fonctionnelle et déclarative,
  - programmation objet
  - ex : Lisp, Caml, Smalltalk, C++, Java, ...

Les langages de programmation (2/2)

- Programmation logique
  - issue de l'IA
  - ex : prolog, lisp, scheme…

- Programmation orientée agent ?
  - Agent = objet autonome communicant

Mise en œuvre

- Porter le logiciel sur le système client

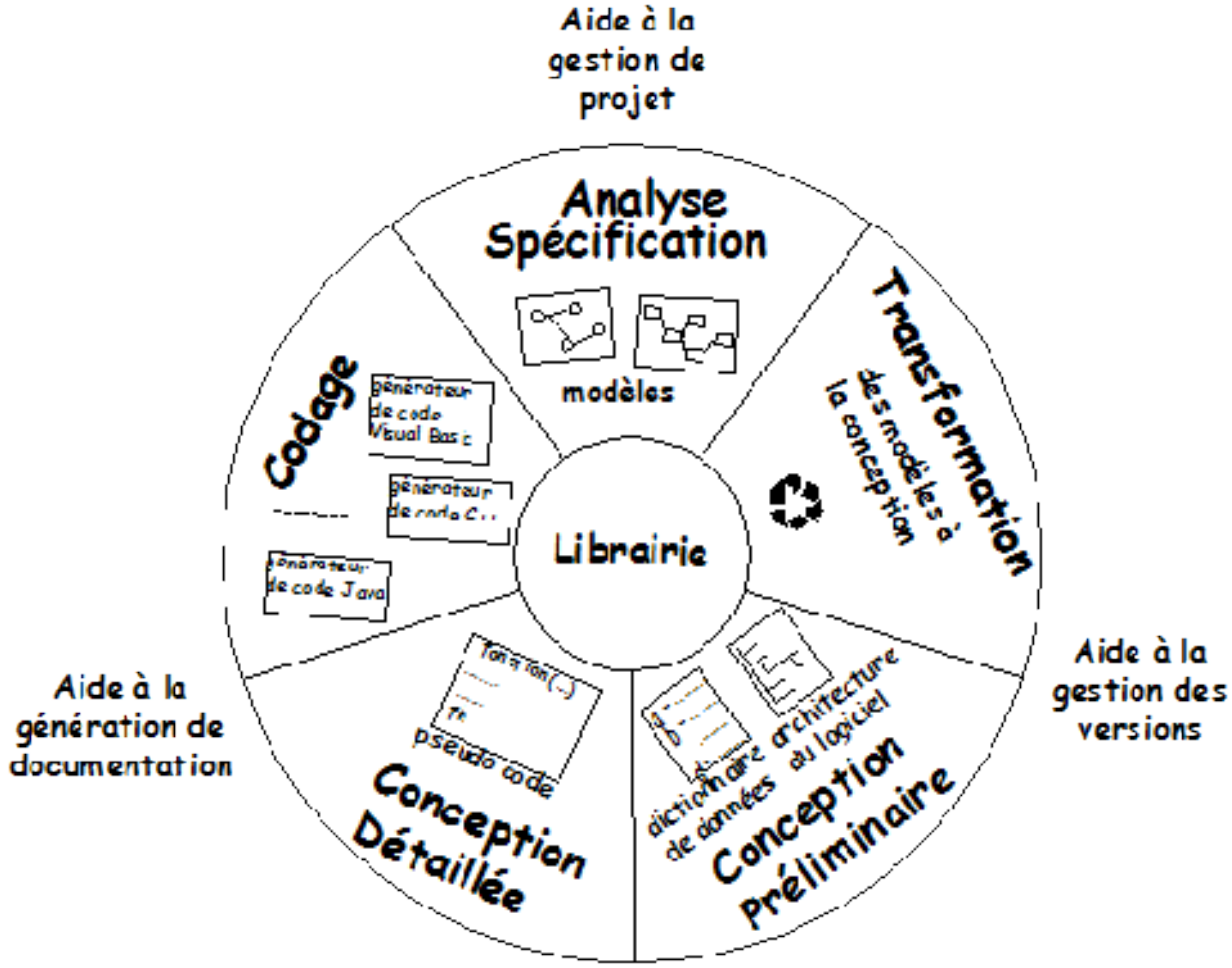
- Vérifier l'adéquation avec le système visé
  - Validation

- De manière générale, il faut tenter de valider à tout niveau (spécifications & conceptions)
  - test unitaires (composants)
  - test d 'intégration (logiciel complet)
  - test système (logiciel sur site)

Les AGL: des outils d'aide

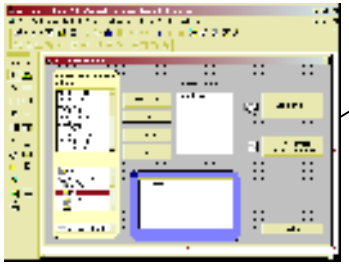
- AGL = Atelier de Génie Logiciel
  - Atelier de modélisation
    - Une méthode n'existe que si elle est supportée par un outil
  - Atelier de développement
    - facilite la programmation (Visual Basic, Visual C++, ...)

Atelier de modélisation



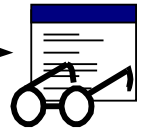
Atelier de développement

éditeur de code



éditeur d'interfaces homme-machine

compilateur



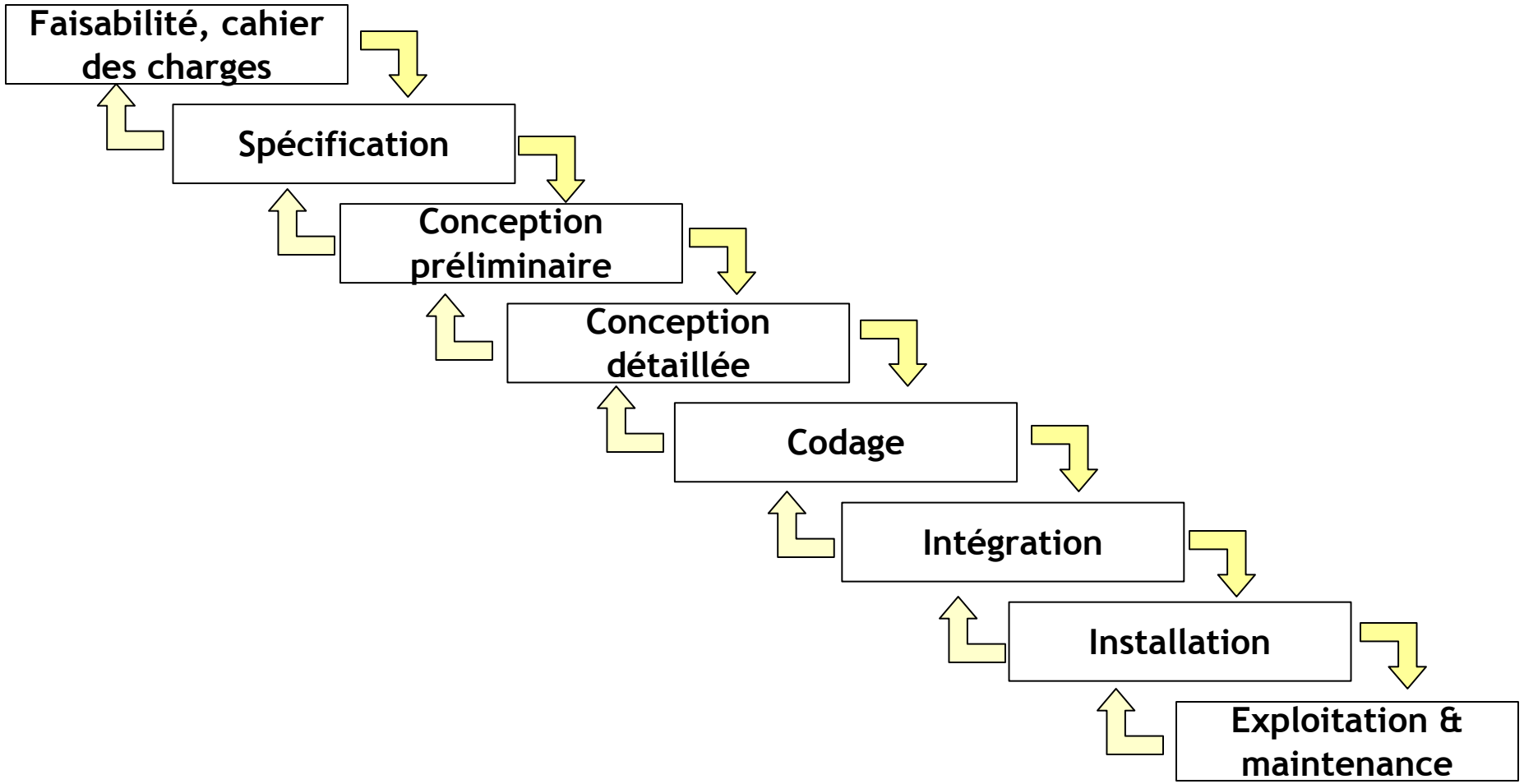
debugger

Les différents modèles de développement

- Enchaînement des étapes
  - Le modèle cascade
  - Le modèle en V
  - Le modèle évolutif
  - Le modèle spirale

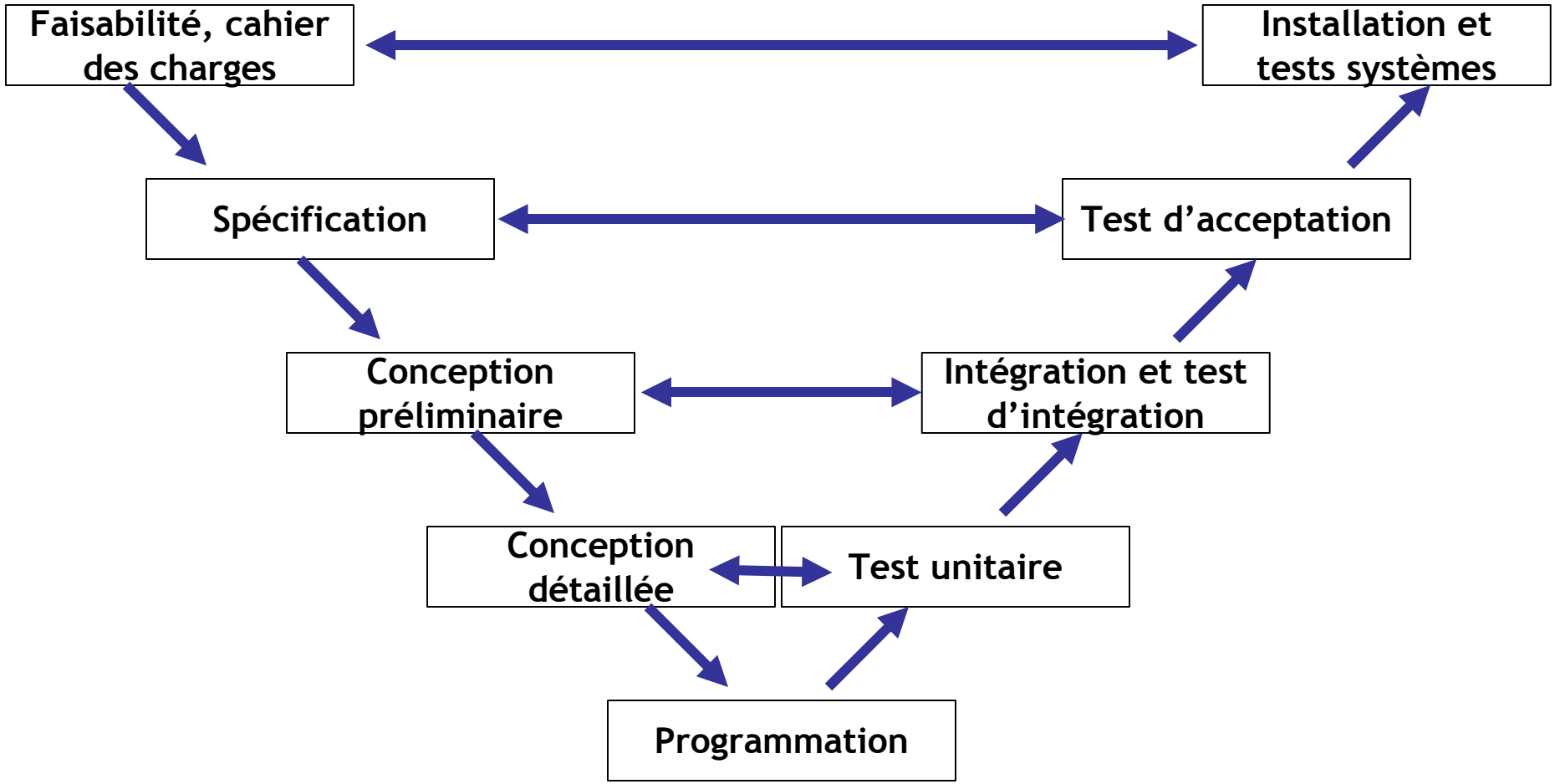
Les différents modèles de développement

- Le modèle cascade



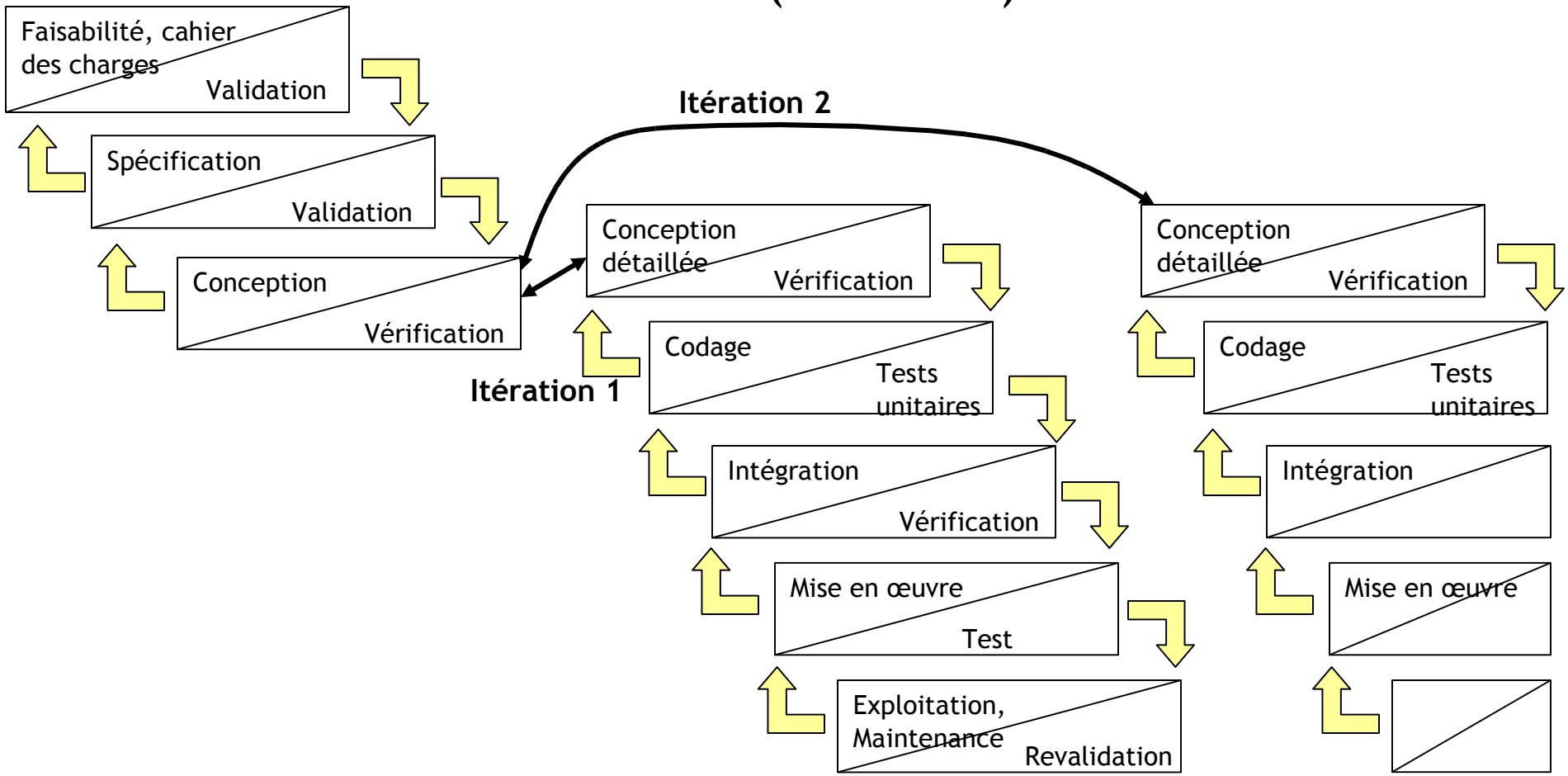
Les différents modèles de développement

- Le modèle en V



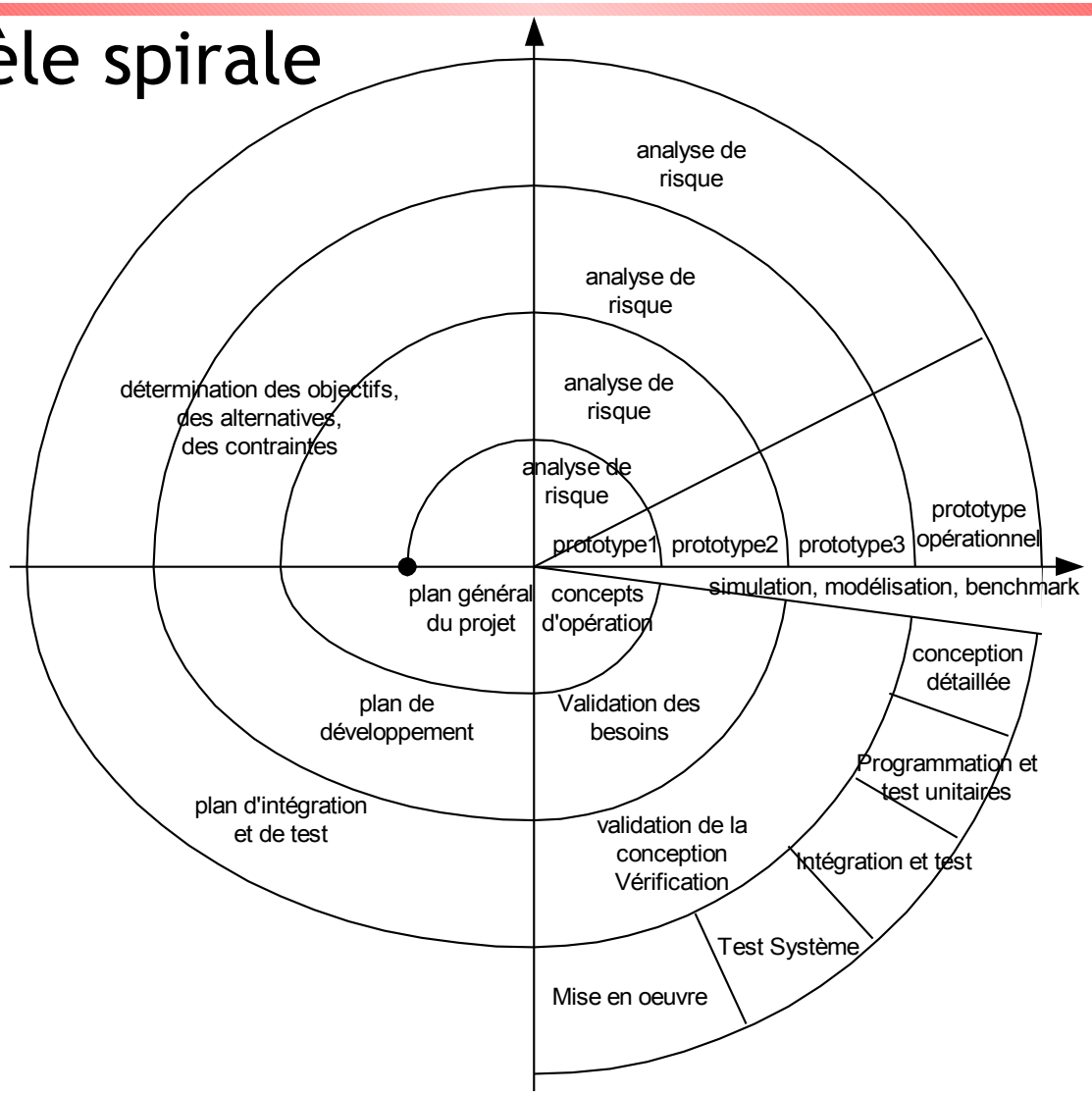
Les différents modèles de développement

- Le modèle incrémental (évolutif)



Les différents modèles de développement

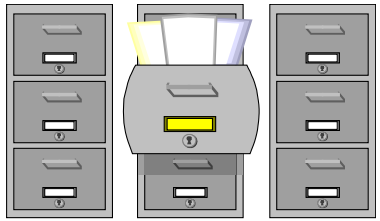
- Le modèle spirale



Les formalismes

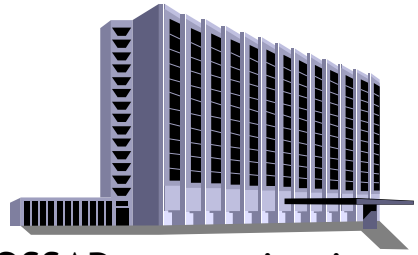
- Représenter le système (initial ou visé)
- 4 types de modèle :
  - modèle des données
  - modèle des flux de données
  - modèle de traitement
  - modèle dynamique

Quelques méthodes d'analyse, de spécification et de conception



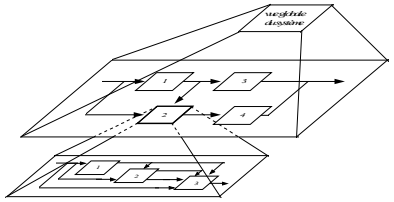
MERISE : orientée BdD

OMT : orientée objet
UML

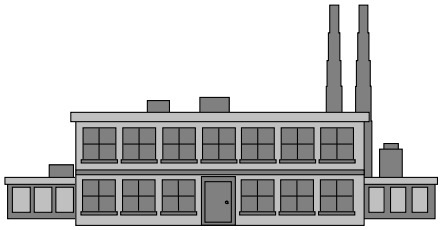


OSSAD : organisation

CISAD : coopération



SADT : méthode structurée



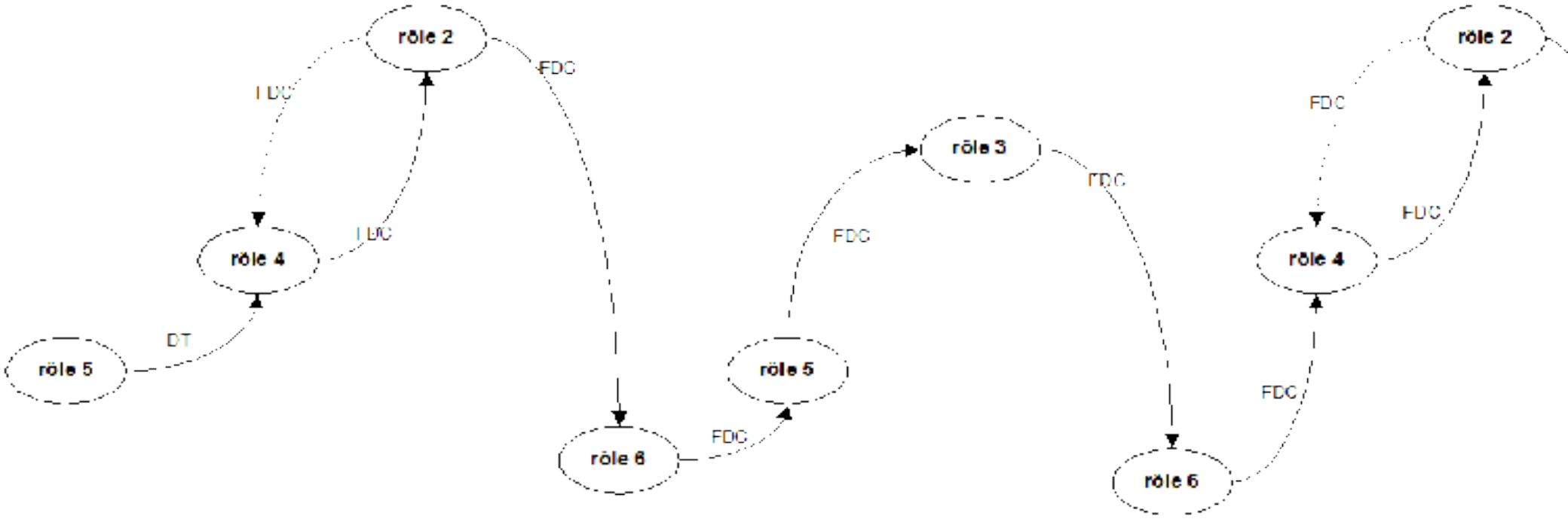
MKSM : gestion des connaissances

MERISE

- Orientée Base de données
- Cycle cascade
- Approche descendante
- Modélisation : données & activités, traitements
- Particularités :
  - modèles conceptuels, organisationnels, logique et physique

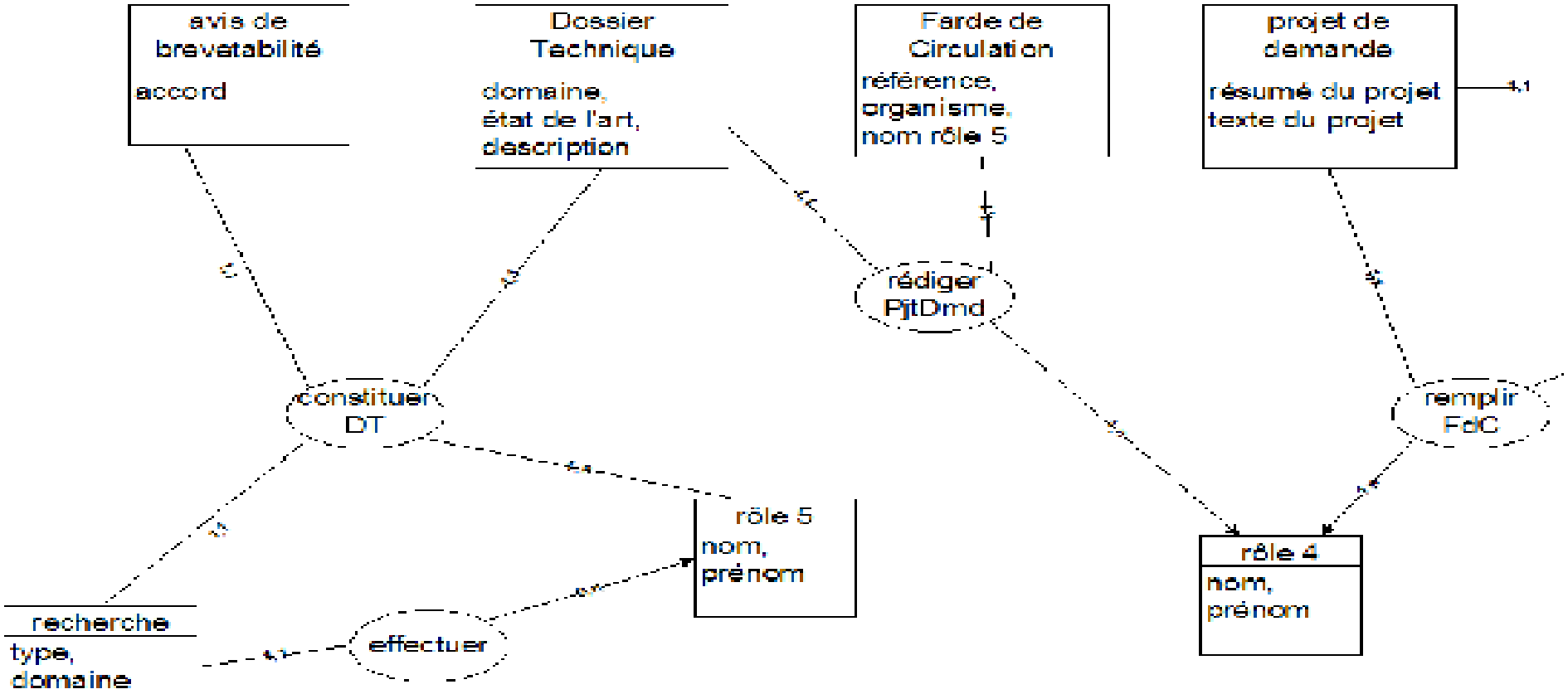
MERISE

Modèle Conceptuel de Communication



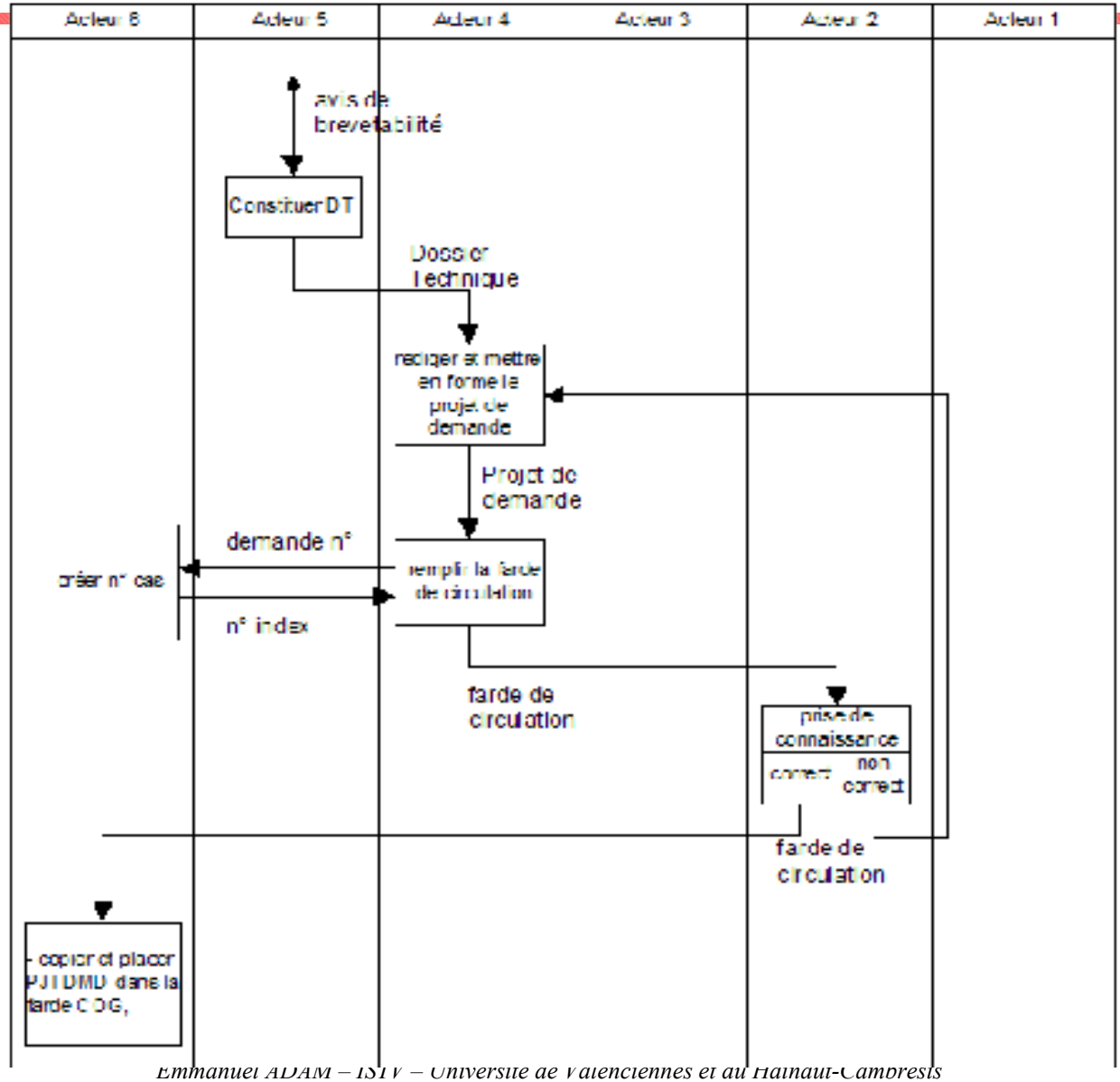
MERISE

Modèle Conceptuel des Données : Modèle Entité-Relation



MERISE

Modèle Organisationnel des traitements

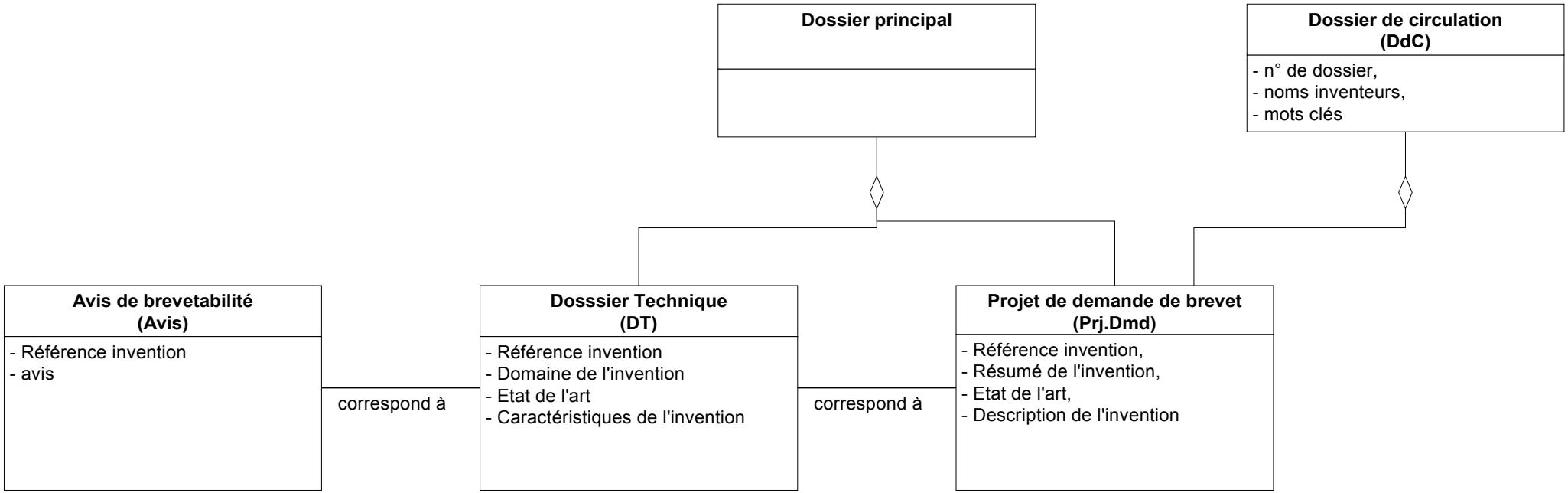


OMT : Object Modeling Technique

- Orientée Objet
- Cycle spirale
- Approche descendante
- Modélisation : données, dynamique, traitements
- Particularités :
  - Méthode objet utilisée dans qqs entreprises avant l'apparition d'UML

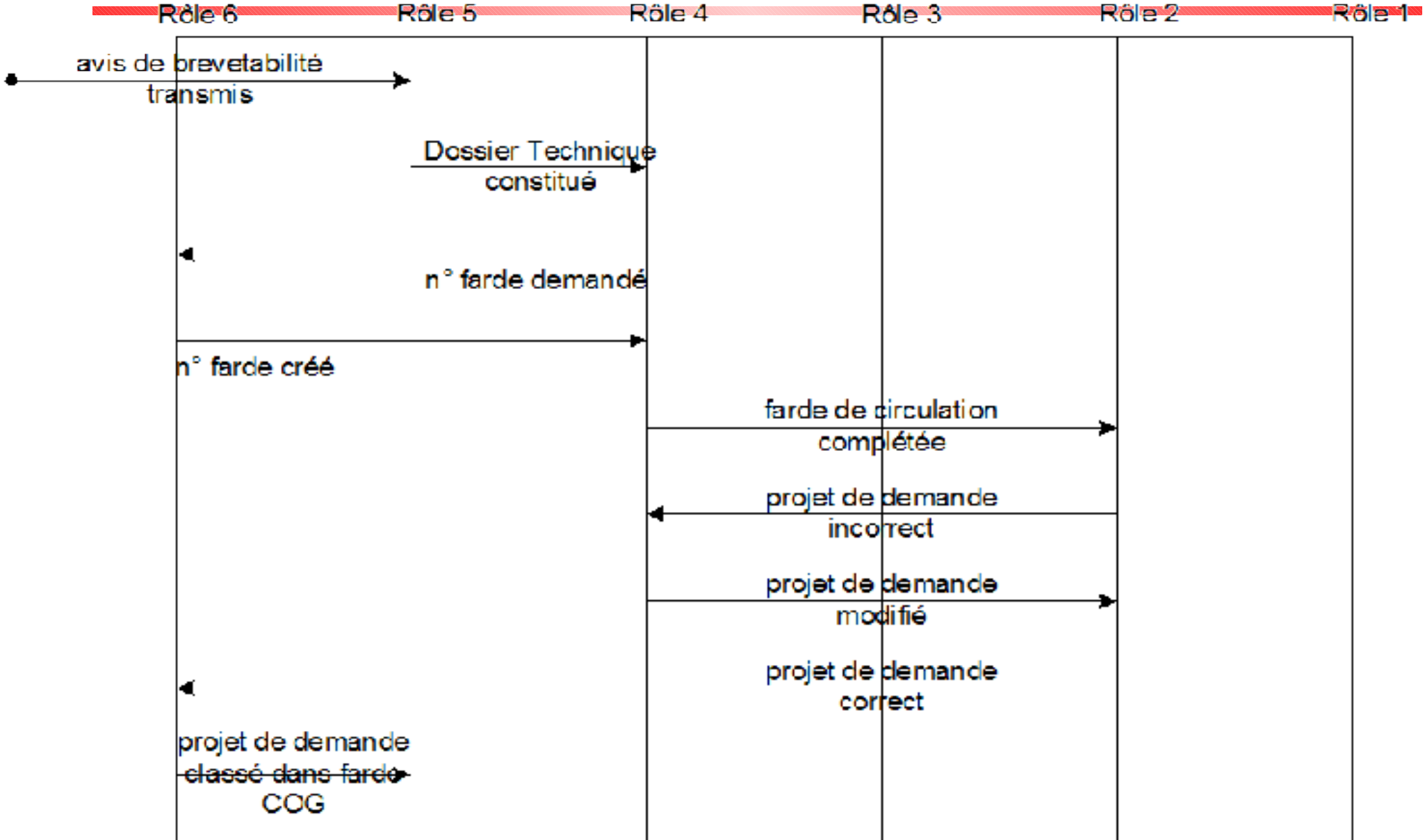
OMT

Modèle de données



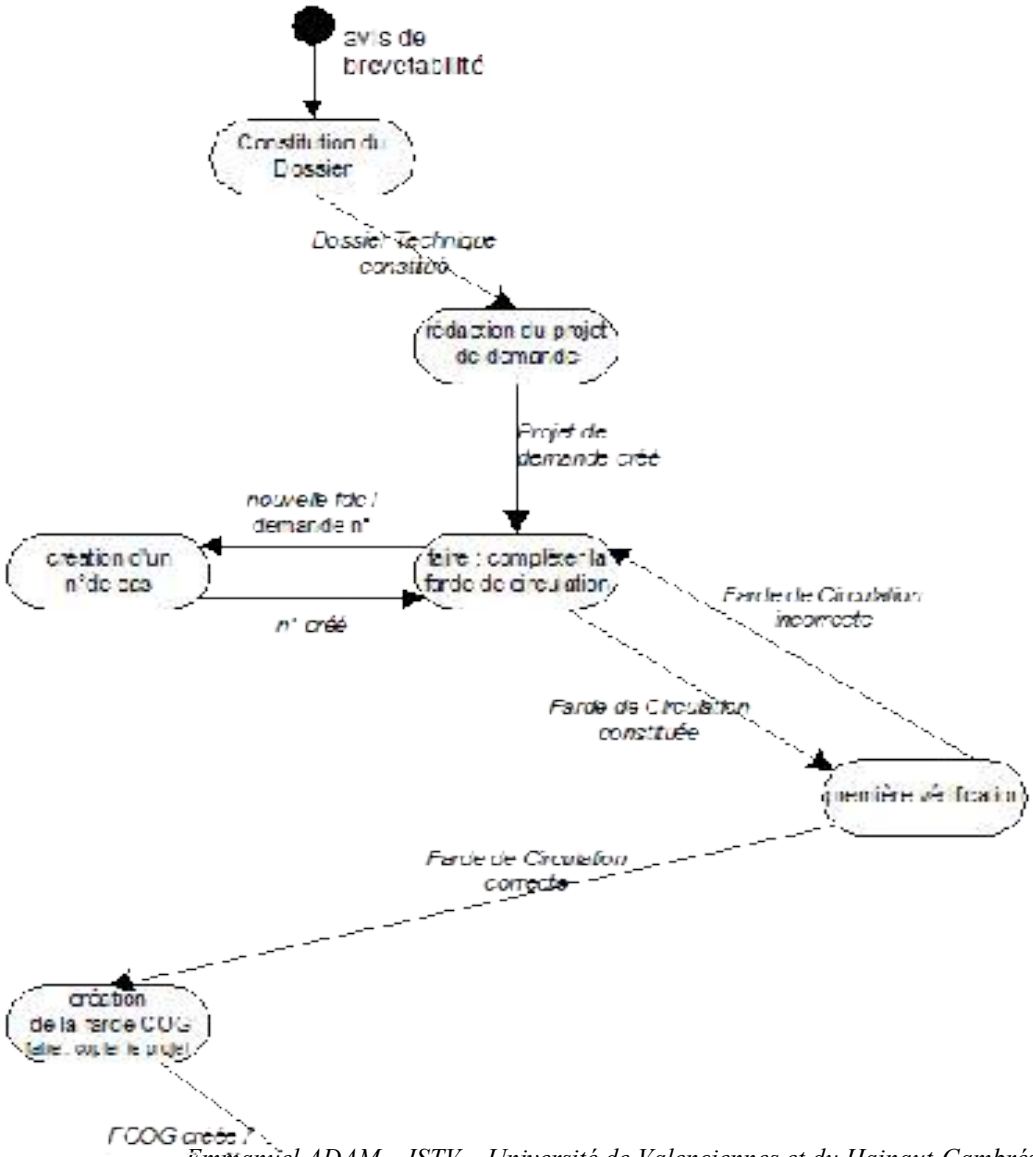
OMT

Diagramme d'événements



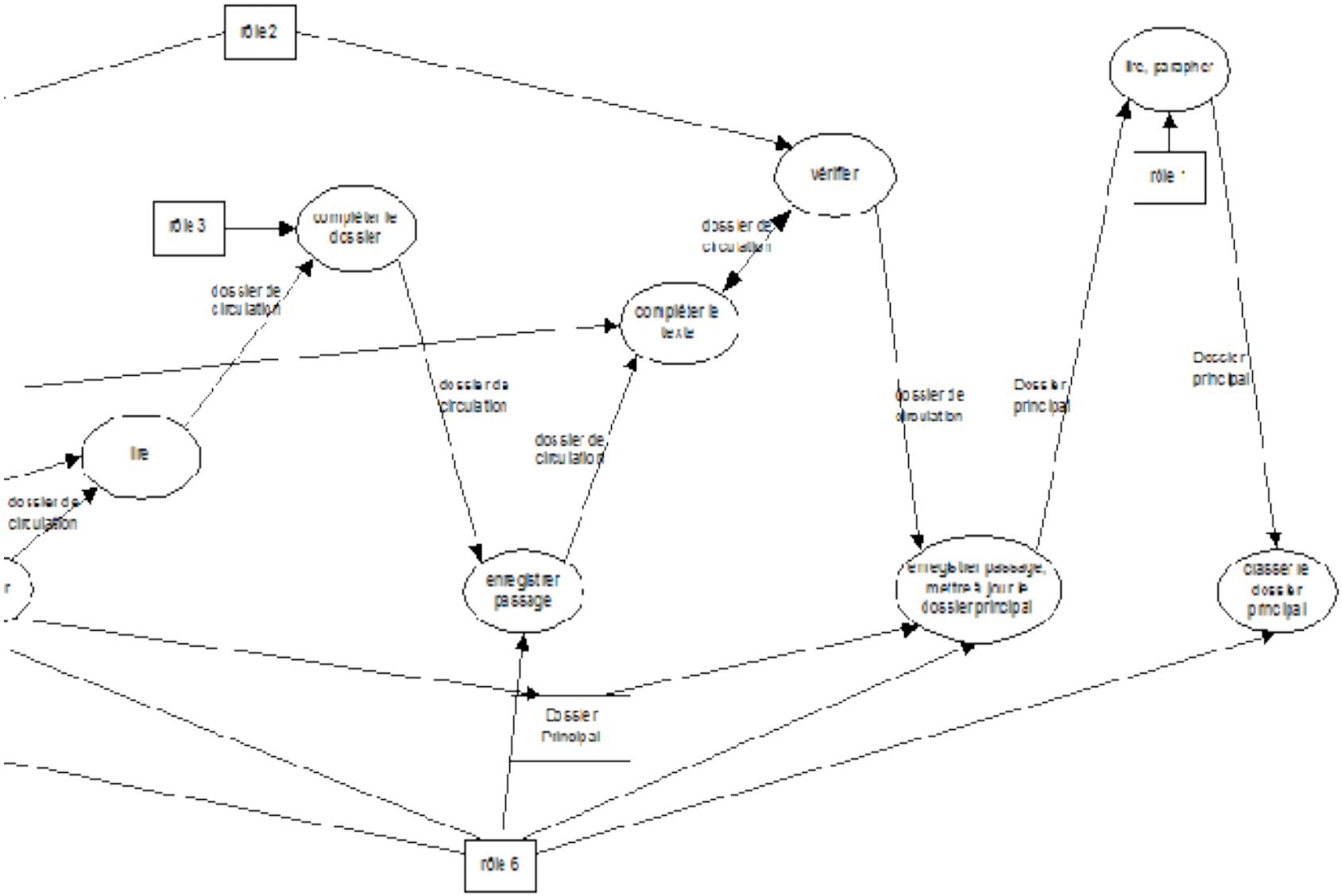
OMT

Diagramme d'état



OMT

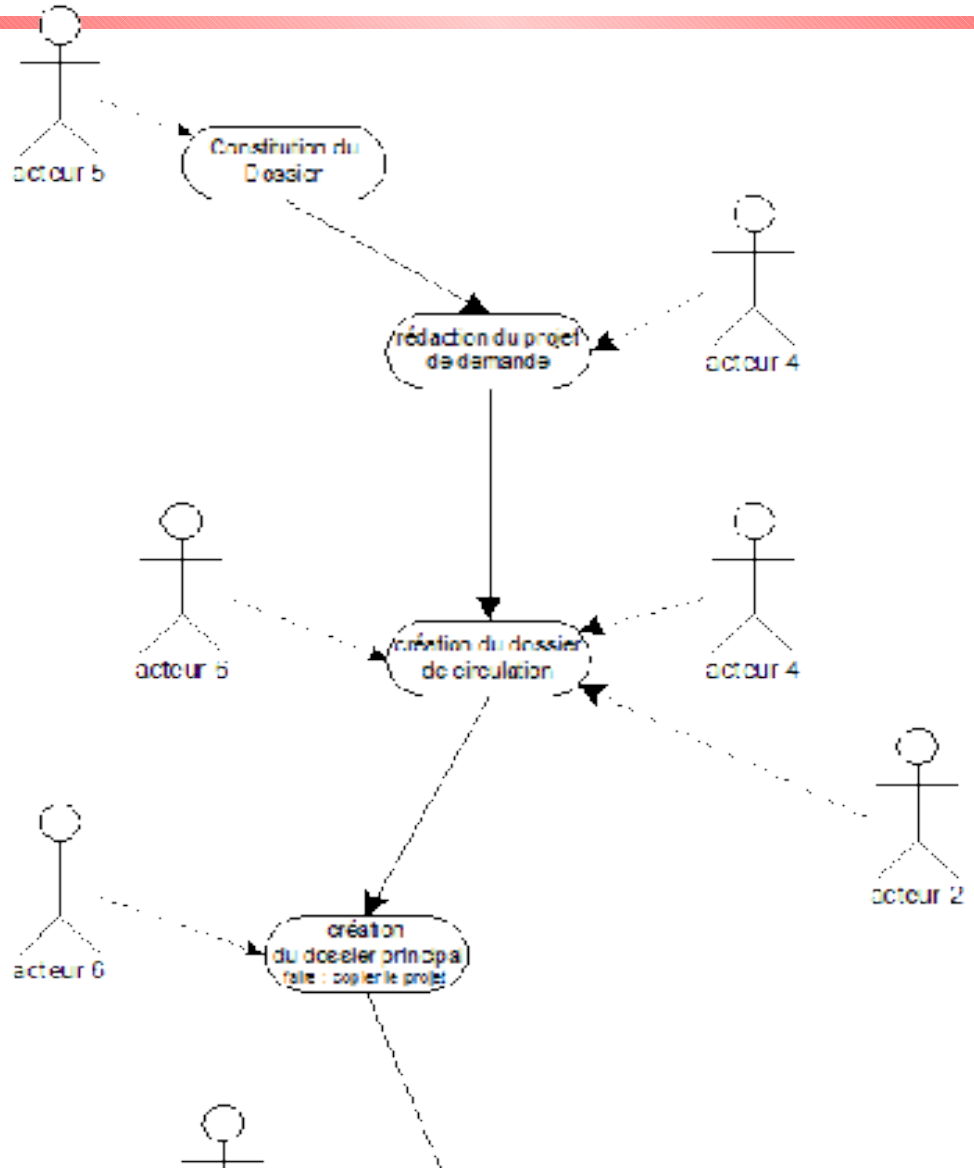
Diagramme de flux de données



UML : Unified Modelling Language

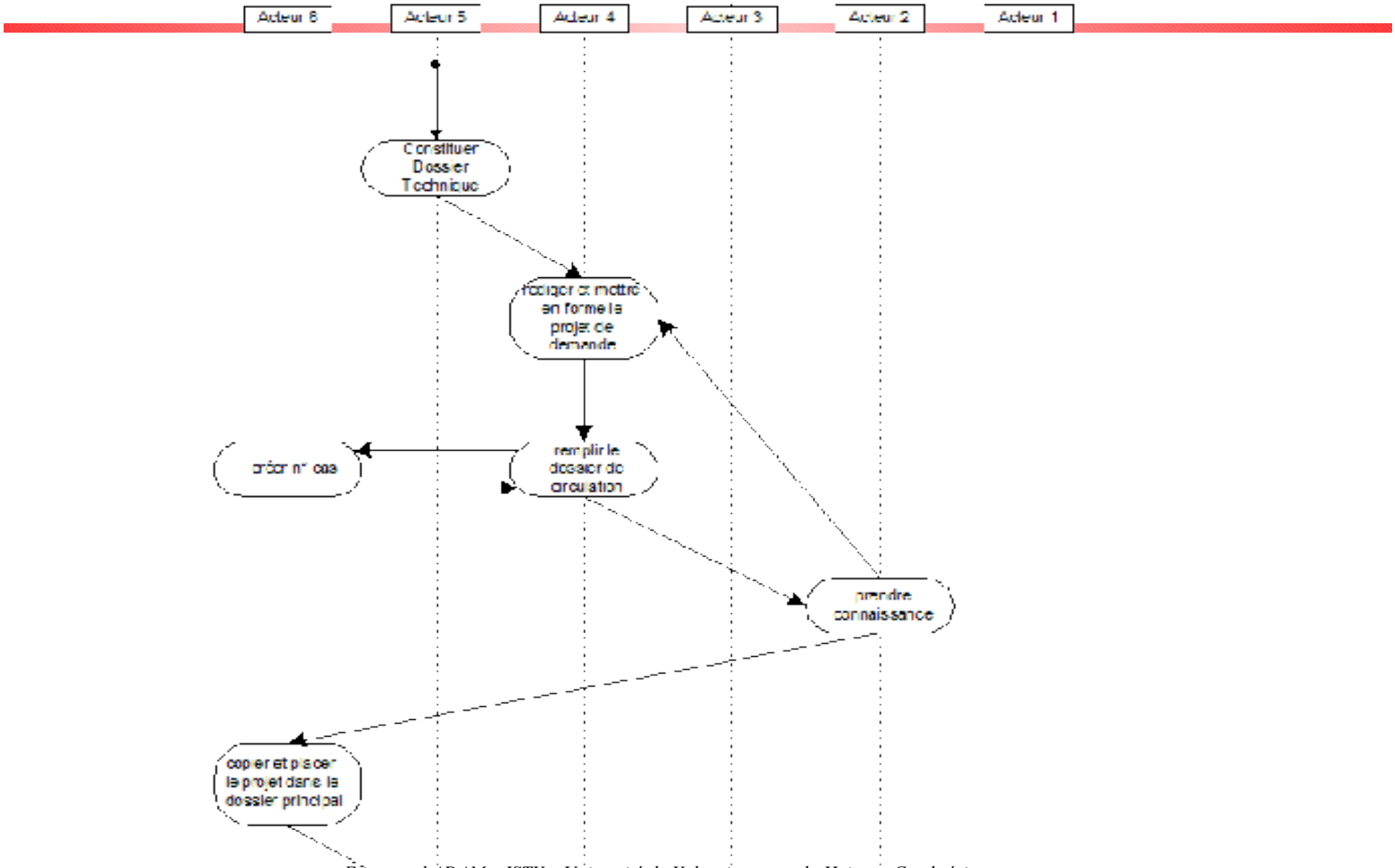
- Langage, fusion des différentes méthodes orientées objets
- supporté par des outils
- +, cf. prochains cours

UML
Cas d'utilisation



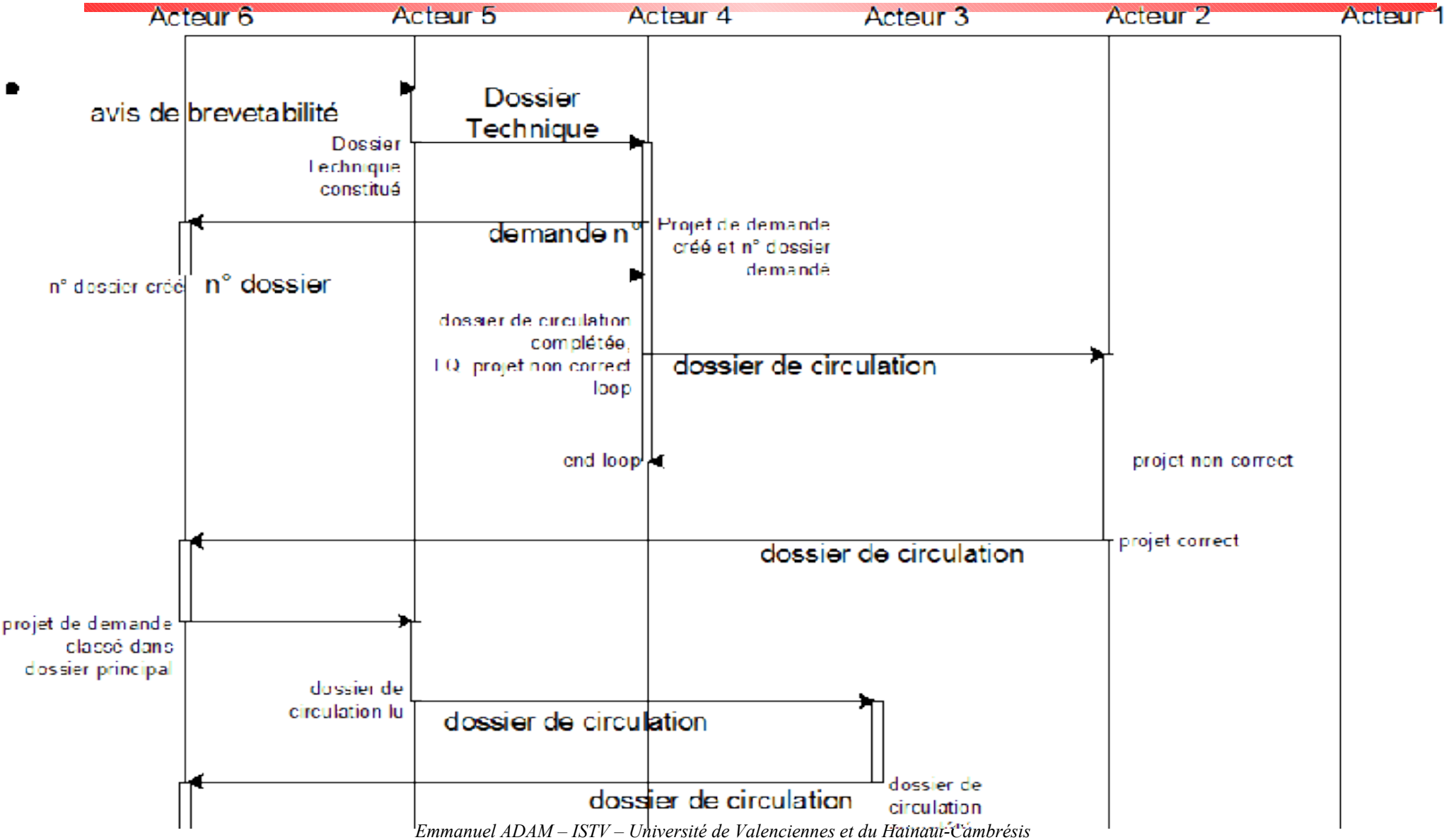
UML

Scénario



UML

Diagramme de séquence

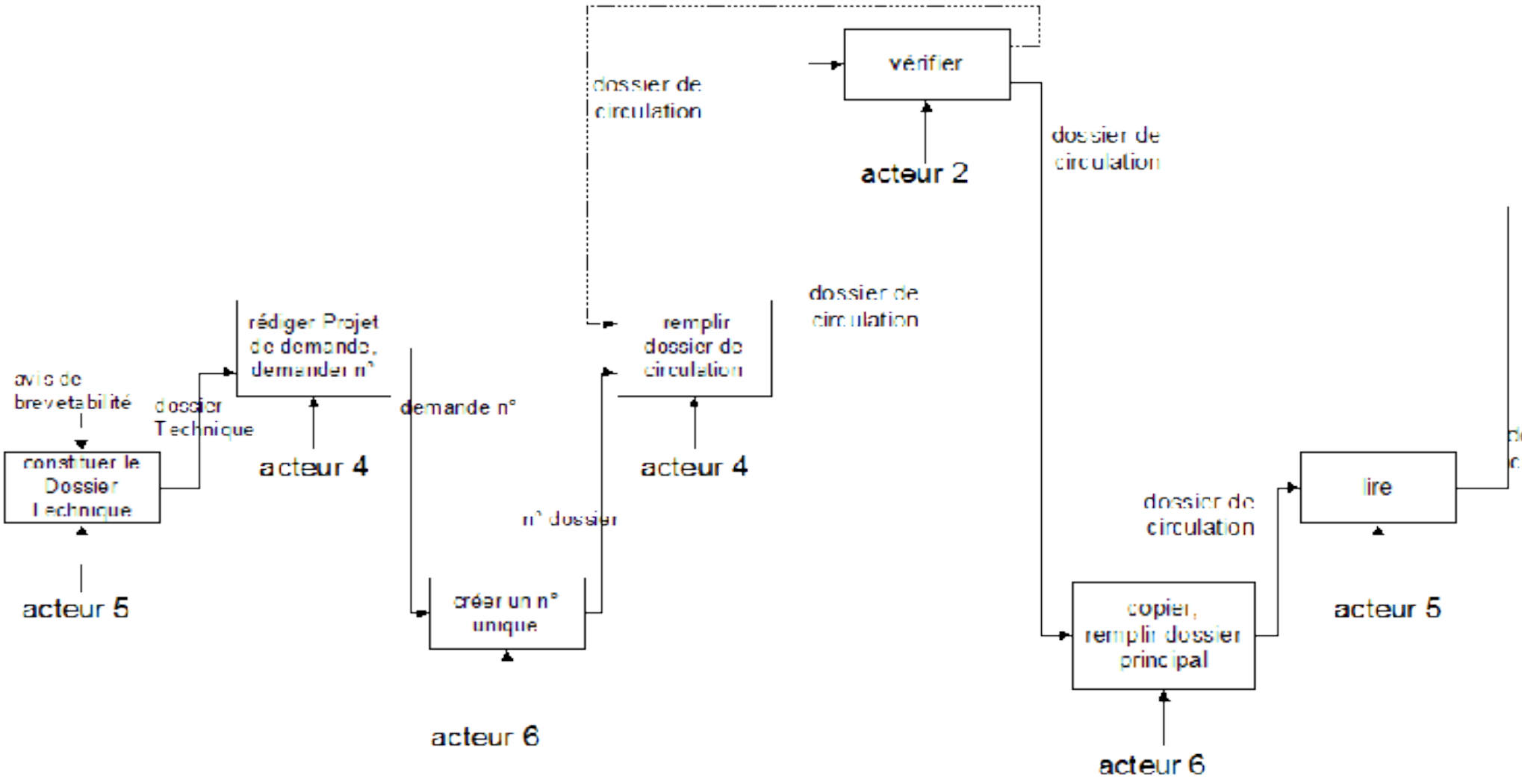


SADT : Structured Analysis and Design Technique

- Méthode structurée
- Cycle cascade
- Approche descendante
- Modélisation : données, activités
- Particularités :
  - Actigrammes fortement utilisés

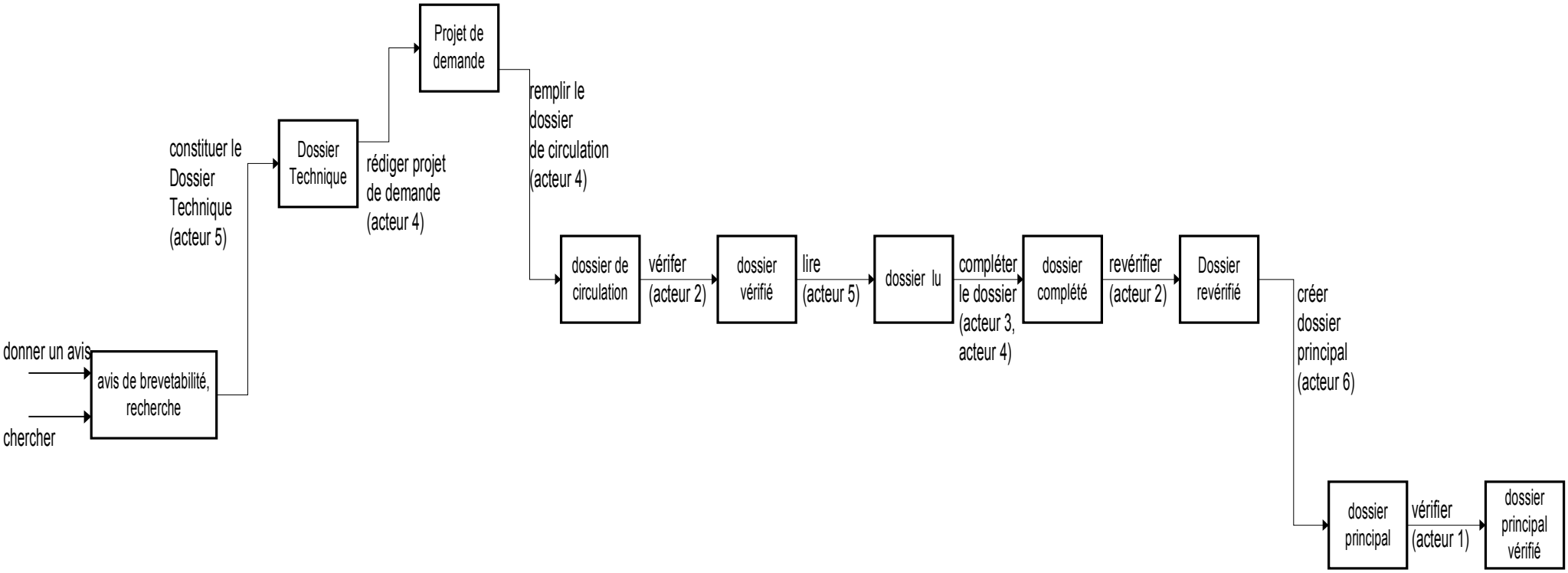
SADT

Actigrammes



SADT

Datagrammes



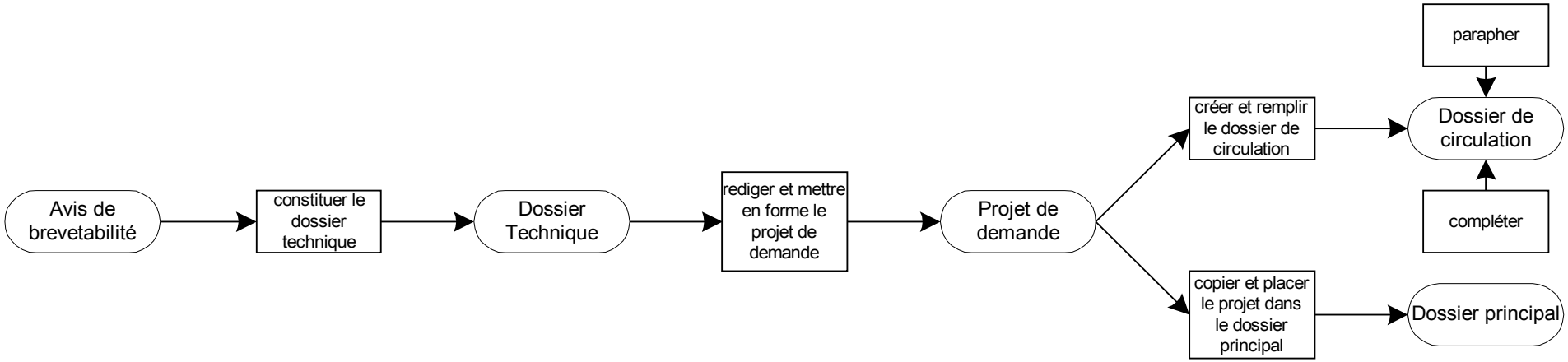
OSSAD : Office Support System Analysis and Design

- Méthode pour modélisation des organisations
- Cycle V
- Approche descendante
- Modélisation : activités, traitement, données
- Particularités :
  - 8 formalismes proposés !

OSSAD

Diagramme A1

Relations entre les fonctions de l'organisme



OSSAD

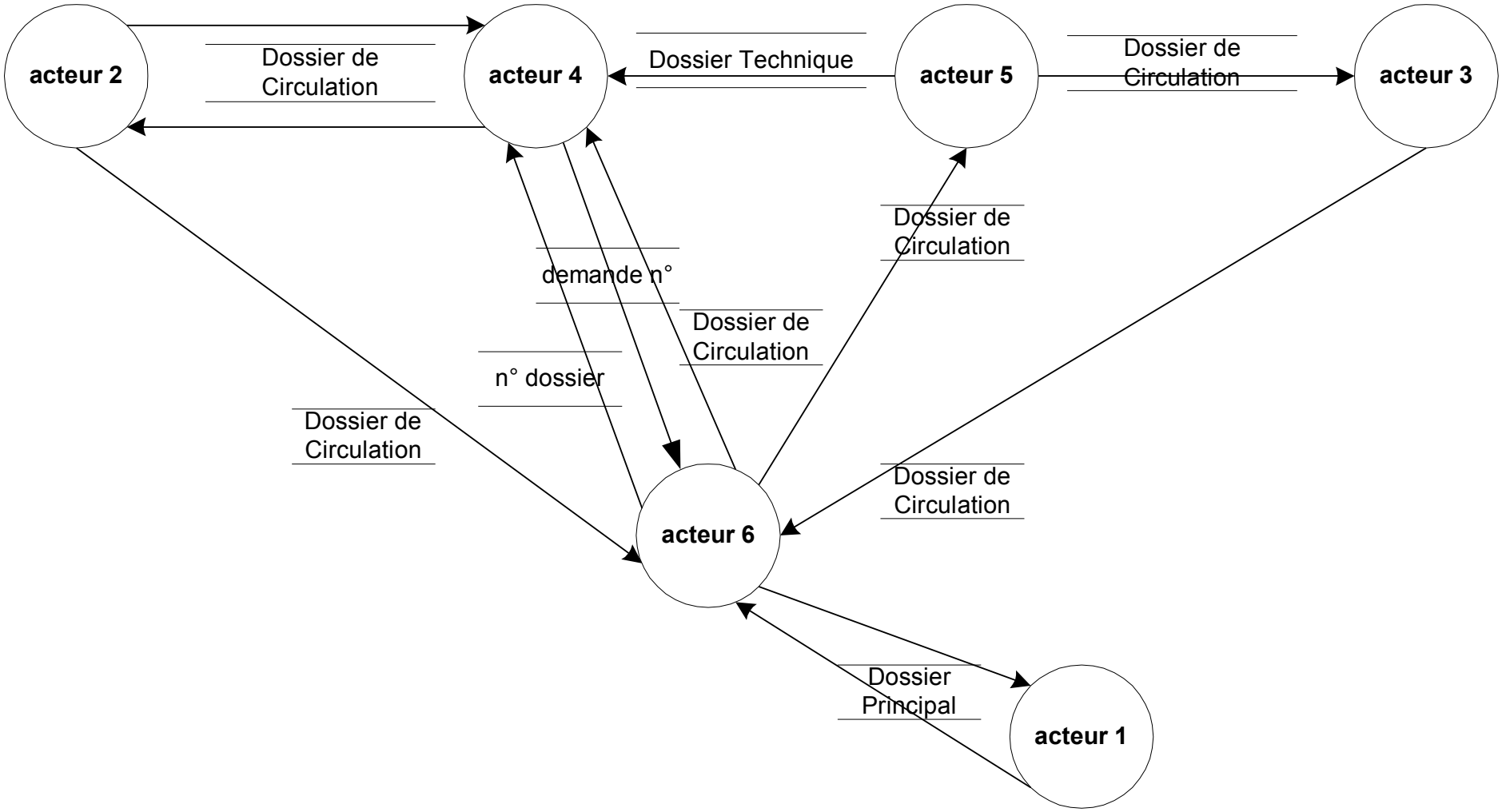
Diagramme A2

Matrice Activité/Rôle

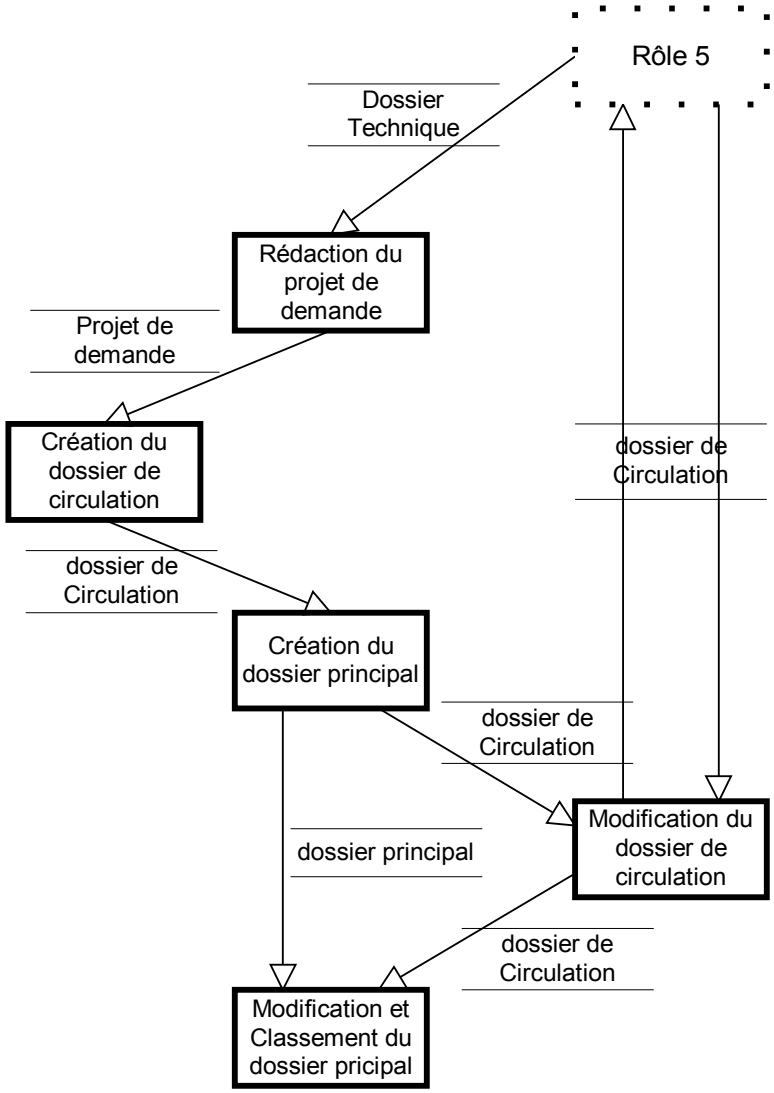
	Acteur 6	Acteur 5	Acteur 4	Acteur 3	Acteur 2	Acteur 1
Constitution du dossier		X				
Rédaction du projet de demande			X			
Création du dossier de circulation	X		X		X	
Création du dossier principal	X					
Modification du dossier de circulation	X	X				
Modification du texte du brevet	X		X			
Modification et envoi du dossier principal	X					X

OSSAD

Exemple de Diagramme D1 : Relations entre rôles



OSSAD
Exemple de diagramme D2 : Relations entre procédures



OSSAD

Fiche représentant une tâche

Fiche Tâche

Nom de la tâche + (Id) :	remplir Dossier de Circulation par acteur 4 (Ta402)
(Decription) :	
Liens (arborescence) :	
/ asc. (Procédure)	Création du dossier de Circulation (Pr003)
\ desc. (Opérations)	Demander n° dossier (Op402), Remplir (Op403)
Relations avec tâches :	Rédaction du Projet / Acteur 4, Création du dossier principal / Acteur 6
par ressources entrantes	Projet de demande (Doc001)
par ressources sortantes	Farde de Circulation (Doc002)

OSSAD

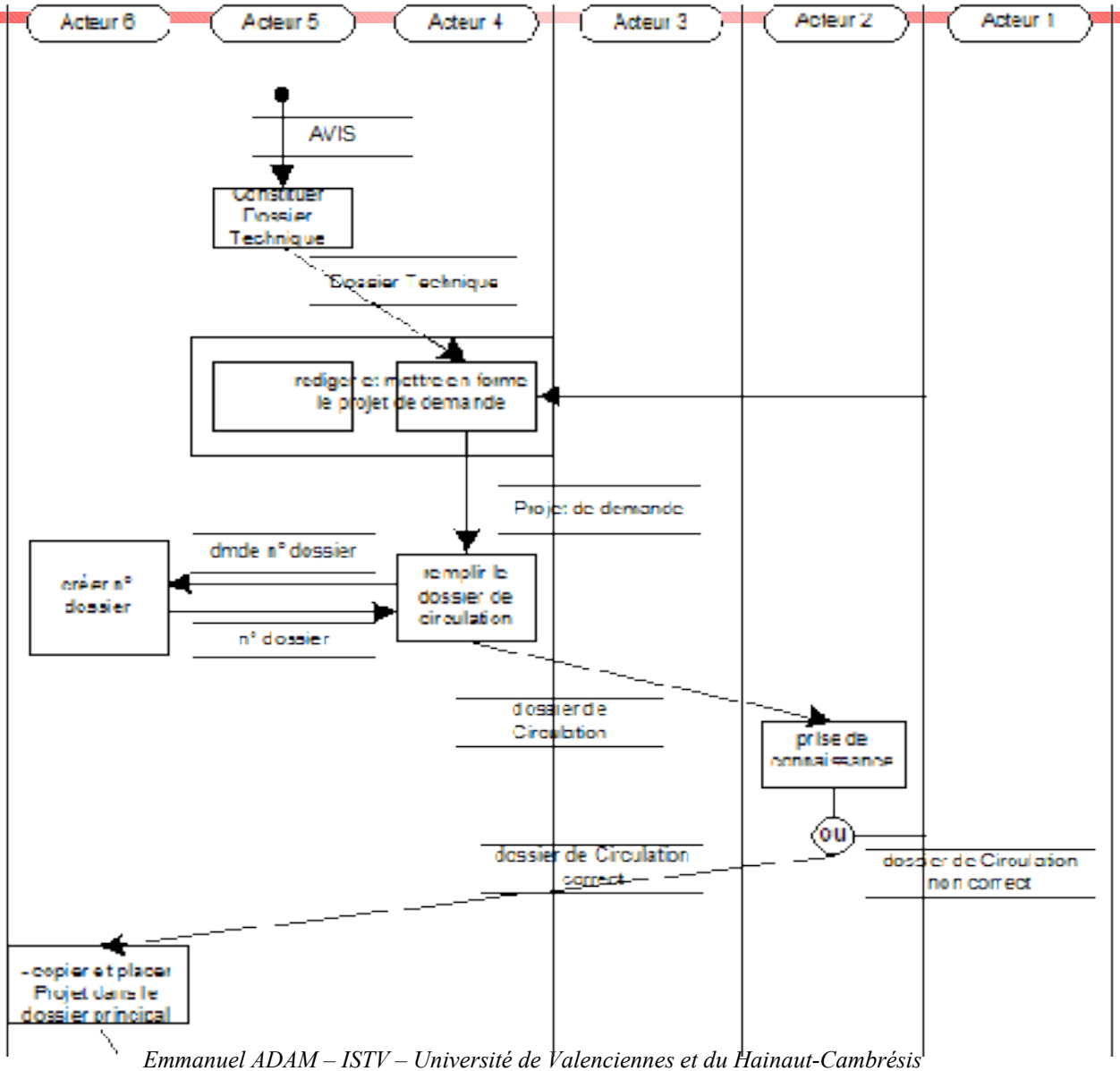
Fiche représentant une donnée

Fiche Ressource

Nom de la ressource + (Id) :	Projet de demande (doc001)
(Decription) :	projet de demande de dépôt de brevet
Liens (arborescence) :	
/ asc. (Super Ress.)	néant
\ desc. (Rubriques)	référence invention, résumé, état de l'art, description de l'invention
Relations avec opérations :	
Origine :	Rédiger le projet de demande (Op451)
Destination :	Mettre en forme le projet (Op452)

OSSAD

Diagramme d ’une procédure

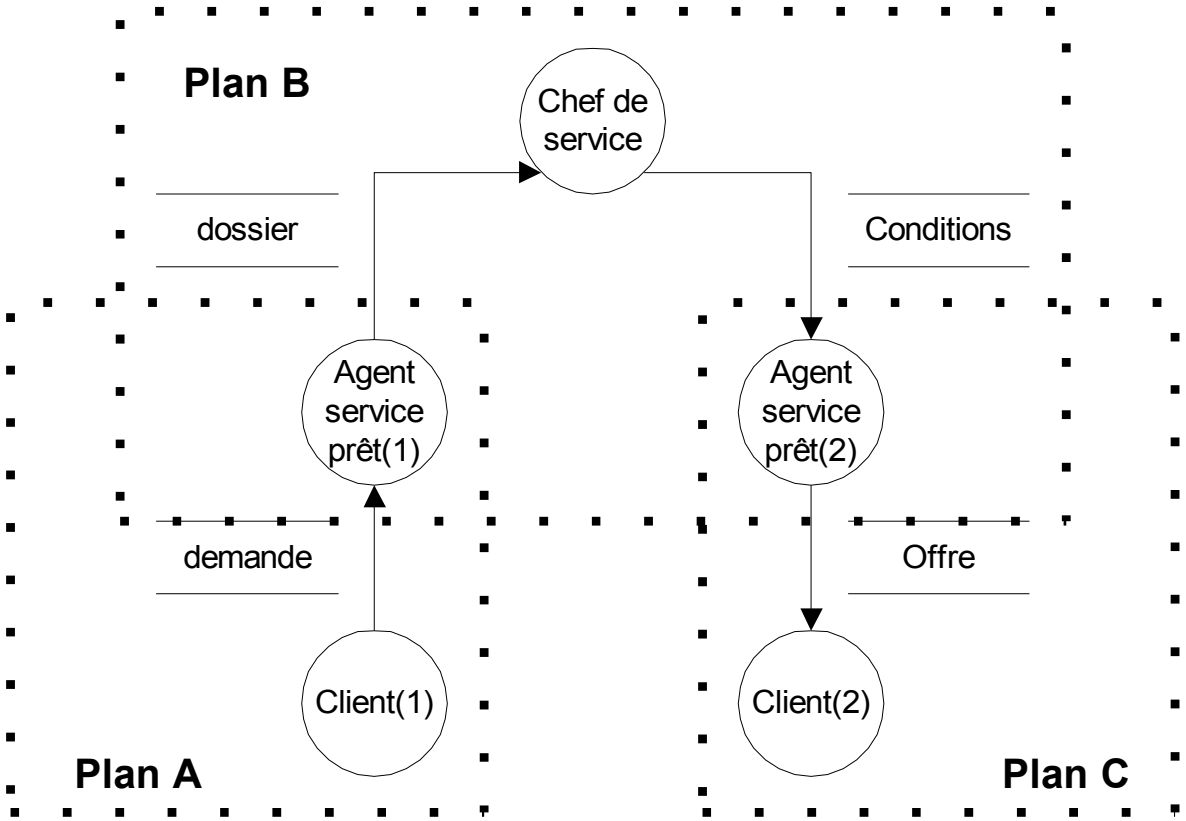


CISAD : Cooperative Information System Analysis and design

- Étend OSSAD en permettant la prise en compte de la coopération

CISAD

Exemple de modèle descriptif de rôle adapté (type D1)



CISAD

Matrice de Grudin pour la modélisation des communications

		site 1			...			site imprévisible		
		rôle A	rôle B	...	rôle A	rôle B	...	rôle A	rôle B	...
site 1	rôle A									
	rôle B									
	...									
site ...	rôle A									
	rôle B									
	...									
site imprévisible	rôle A									
	rôle B									
	...									

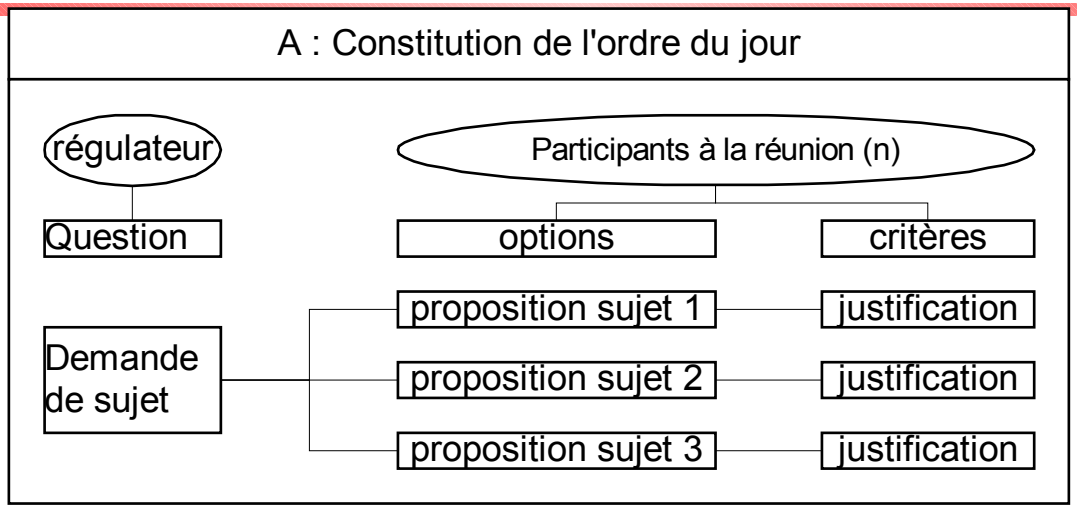
Synchrone

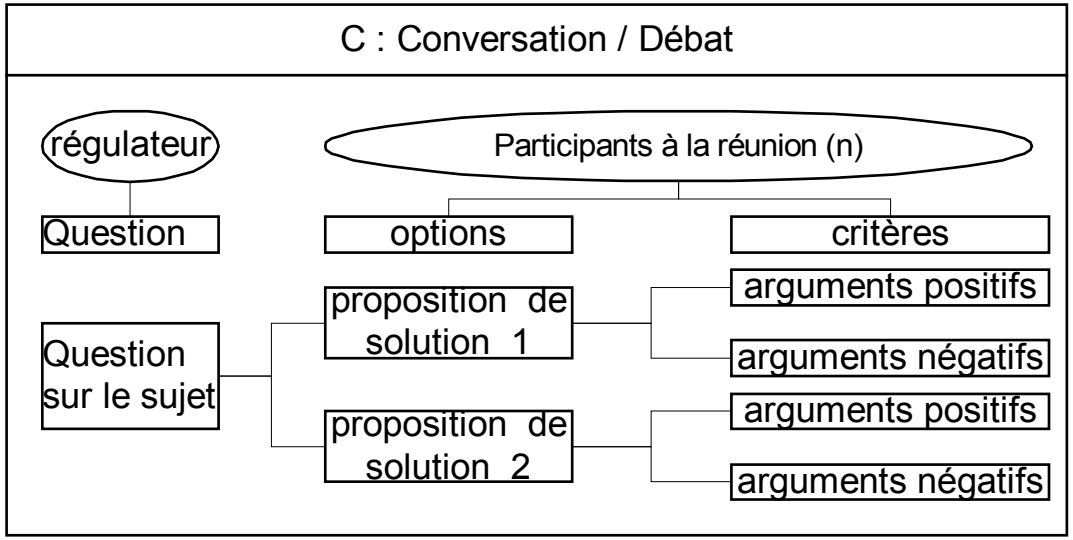
Asynchrone et moment prévisible

Asynchrone et moment imprévisible

CISAD

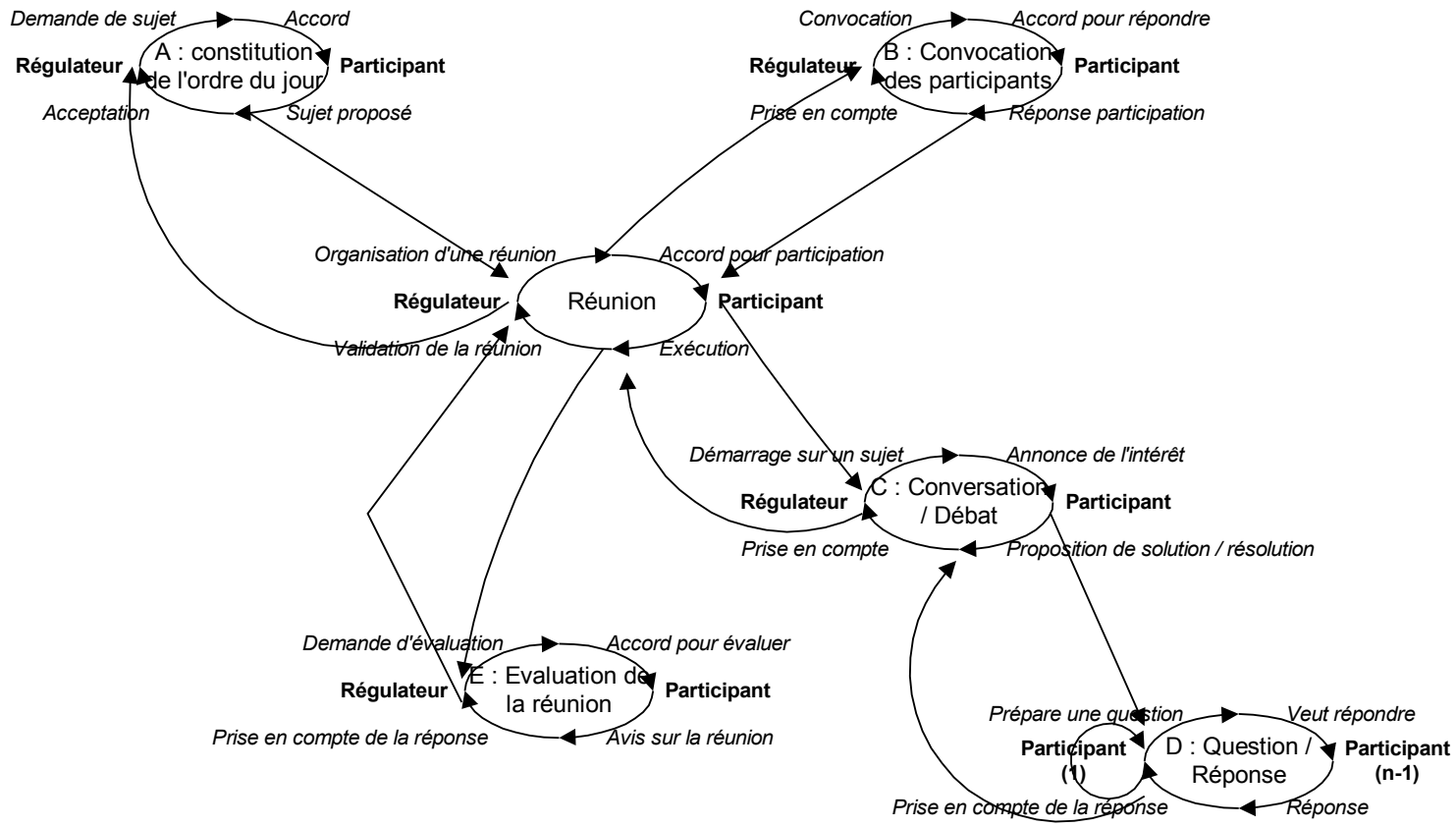
Exemple de modèles d'argumentation





CISAD

Un exemple de représentation de la conversation

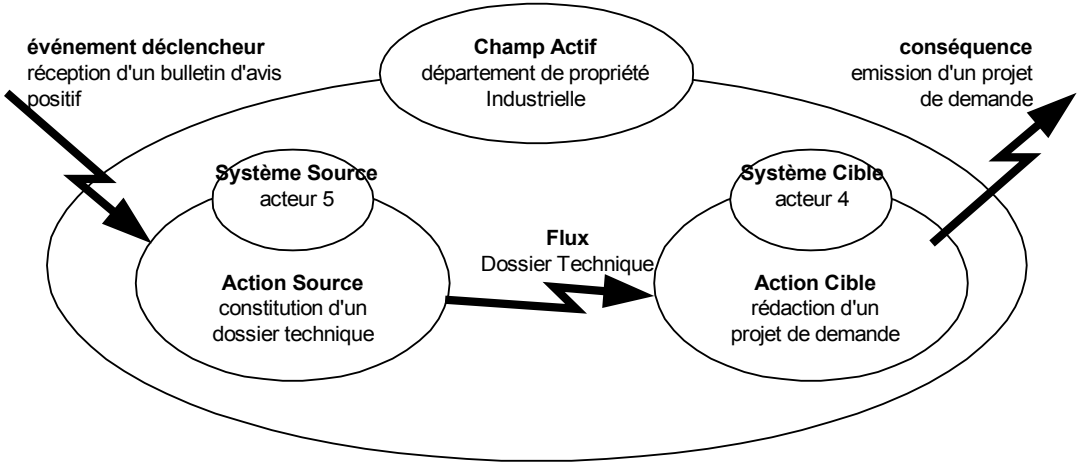


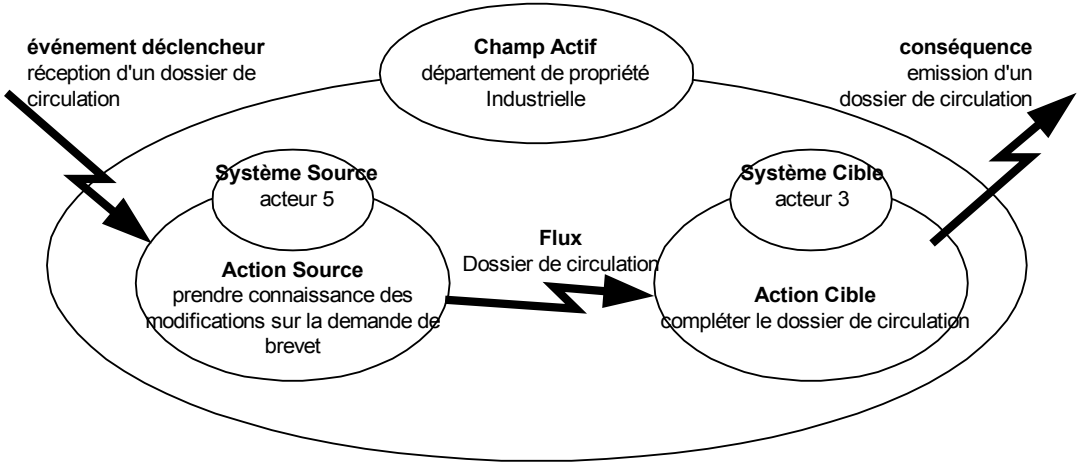
MKSM : Methodology for Knowledge System Management

- Méthode pour la gestion des connaissances
- Cycle V
- Approche descendante
- Modélisation : traitement, activités, données
- Particularités :
  - s'arrête à la modélisation du système réel,
    passe la main à OMT pour la spécification du système informatique.

MKSM

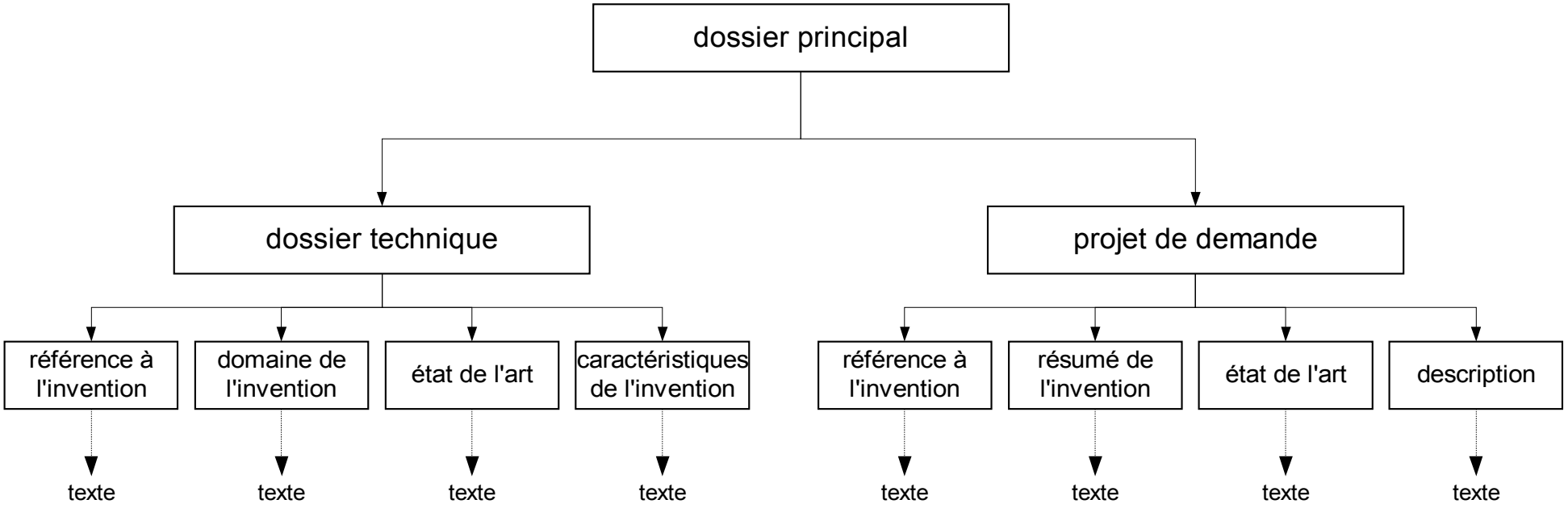
Modèle du domaine: représentation d ’un processus





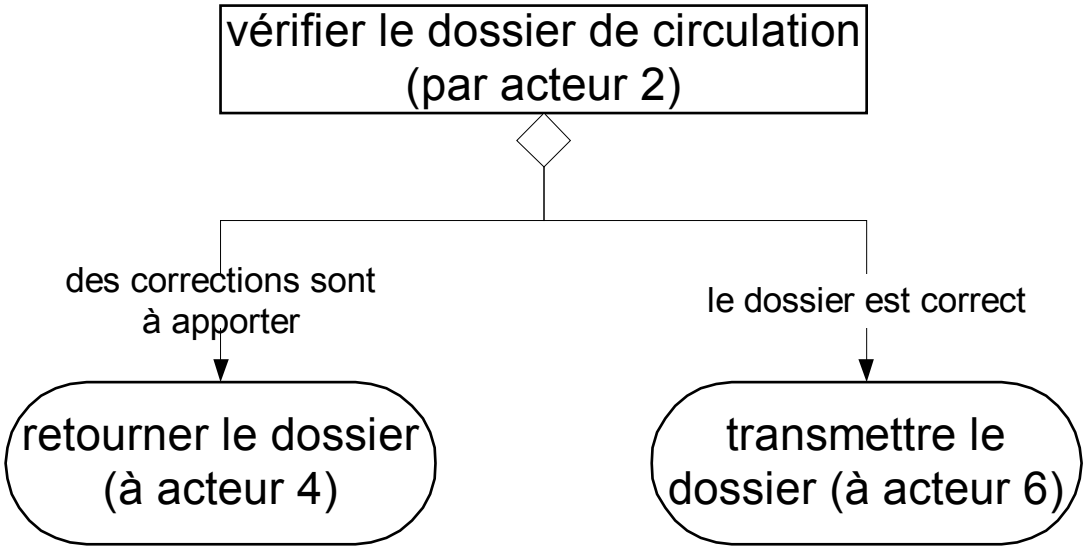
MKSM

Modèle de concepts : Exemple d ’un réseau de concepts



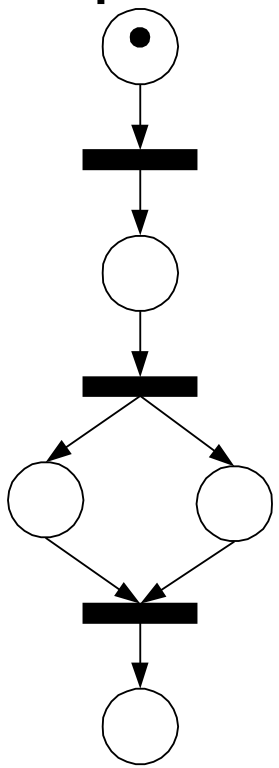
MKSM

Modèle de tâches



Autre modèle : le réseau de Petri

- Outil mathématique pour la représentation de la dynamique de systèmes à événements discrets
- Il existe de nombreuses extensions (temporisés, colorés, objets, …)



Conclusion

- Il existe de nombreuses méthodes
- Les méthodes doivent être choisies en fonction du besoin, du domaine
- et surtout suite à une « bonne » analyse devant contenir TOUTES les informations utiles.
- L'analyse est la base de tout projet,
- Ne pas hésiter à coupler les méthodes **si cela est nécessaire** et de façon rigoureuse.