Cours de M. Salem Dakhli

Salem.dakhli@acm.org

Introduction

Complexe car doit supporter les processus métier de l'organisation.
Organisation = objectifs + individus + tâches + technologie (de production, de biens et de services) + structure. Contrairement aux entreprises but pas forcément lucratif, ensemble de personnes.

Complexité -> caractéristique, tandis que Complication -> accidentelle.
Deux types de caractéristiques essentielles :

- Essentielle, on s'en accommode
- Accidentelle, on les supprime

Frederick Brooks Jr. « **No Silver Bullet** », 1987 (→ il n'y a pas de remède miracle, référence à l'inexistence de moyen universel pour remédier aux problèmes informatiques, entre autres la complexification croissante des besoins).
3 méthodes de programmation :

- Diviser pour mieux régner
- Merise/ systémique
- Objet

-> Mais aucune de ces méthodes ne répond vraiment à la problématique de la réalisation d'un logiciel.

Processus métier : Ensemble d'activités qui créent de la valeur.

Différence entre risques et incertitudes. Le risque, on le connaît et on peut l'évaluer (conséquences des risques et probabilité). Le problème, c'est que les SI sont intangibles et il est donc difficile d'en évaluer les risques.

On dit que le SI est situé, c'est-à-dire dépend du contexte organisationnel : deux problèmes peuvent avoir 2 solutions différentes.

La production d'un SI obéit à la **rationalité limité** des acteurs → les solutions obtenues doit être satisfaisante et non parfaite (Herbert Simon : théorie de la décision, intelligence artificielle, rationalité limitée).

Paradoxe de la productivité - Robert Solo 1987 : plus on informatise, plus la productivité baisse

- Il y a toujours une différence entre invention et rentabilisation de ladite invention.
- Problème de mesures, on ne sait pas définir les métriques.

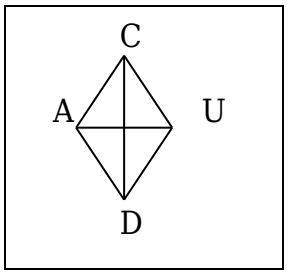
→ Il faut rendre les processus métiers efficients (faire les choses bien, en appliquant bien les règles) et efficaces (réaliser son objectif). La dimension de

l'objectif est multiple : but de l'objectif, contraintes économiques et temporelles … (cf. théorie des dimensions des logicielles).

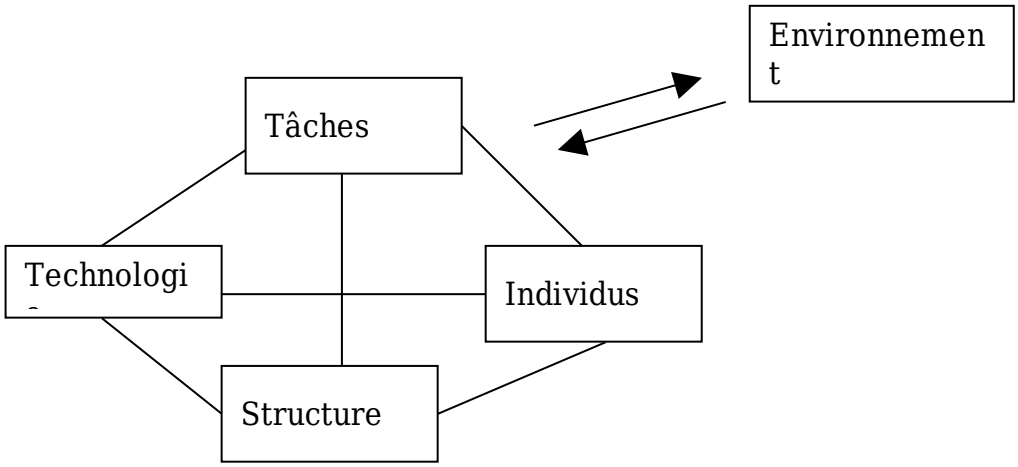
Un SI est un ensemble de moyens permettant de collecter, de stocker, de traiter et de diffuser de l'information.
Donnée : succession de signes et de codes.
Information : donnée + modèle d'interprétation, c'est une connaissance explicite, formalisée.
Connaissance : tacite (Savoir) et explicite (Savoir Faire) issue de l'information.

Il y a 10 dimensions dans un logiciel. 4 types d'acteurs organisationnels concernés par le SI (Stakeholders) :

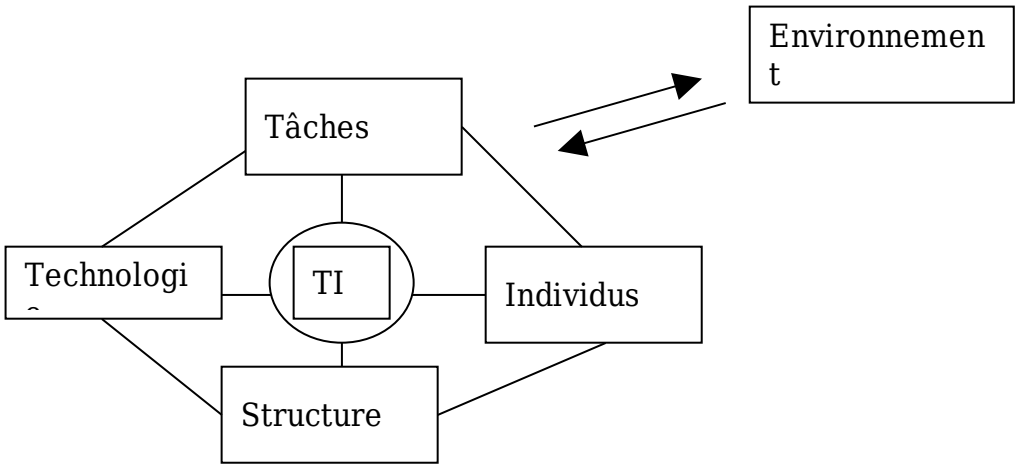
- Client : donneur d'ordre (espace des problèmes)
- Architecte : définit la solution informatique (espace des solutions)
- Développeur : construit, implémente (espace des constructions)
- Utilisateur final : utilise le logiciel développé (espace des opérateurs)



Théorie de l'agence : Il y a des relations conflictuelles au sein d'une même organisation (chacun pour sa peau).
PACO : méta modèle d'auto-amélioration par itérations successives (Problème, Architecture, Construction, Opération).
Le cœur du SI sont les acteurs organisationnels, liés par des contrats d'agence (producteur/consommateur).
Modèle de Leantt (1963) : le diamant organisationnel.



Puis modifié par Stohr et Koysinski (MIT, 1985).



Un SI est un nœud de contrat entre les différents acteurs organisationnels afin de produire, stocker et diffuser l'information. → C'est une organisation.
Informatiser (computerization), c'est fabriquer un SI.
Système informatique (PAS SI) = ensemble de logiciels, matériels et réseaux.

« Si un projet est en retard, rajouter des ressources humaines augmente le retard », Brooke.

Comment modélise-t-on un SI ?

Syndrome Whiscy (Why is not Sam Coding Yet?). Dans un SI, on n'est pas obligé de suivre les processus à la lettre, il y a toujours une grande partie de création.

3 courants :

- Début des années 60 : Paradigme (Tom de Marco : on divise en petites tâches/petits problèmes, Descartes)
- Méthode merise/ systémique : méthode d'analyse, de conception et de gestion de projet complètement intégrée.
- Objet : vient compléter les autres courants. On masque les données.

Objectif de la modélisation d'un SI :

- Spécifier les besoins et les exigences des acteurs.
- Spécifier le système (exigence de temps, de qualité, de contraintes de fonctionnement du système).
- Spécifier l'architecture globale (composants et connecteurs).
- Spécifier l'architecture Détaillée (algorithmique des composants et des connecteurs).

4 types de spécifications

- Type I : Spécifications des besoins et des exigences

Contrat d'agence entre la maîtrise d'ouvrage (MOA) et le maîtrise d'œuvre(MOE).
Porte sur les besoins fonctionnels et non fonctionnels (exigences)

- Type II : Spécifications du système

Contrat entre la MOA et la MOE.
Porte sur les spécifications du type I, les contraintes et la qualité de fonctionnement du système informatique. Cahier des charges. Base du contrat juridique entre MOA / MOE.

- Type III : Spécifications d'architecture globale

Contrat d'agence entre la MOE (l'architecte) et la MOE (autres architectures et développeurs).
Validées par la MOA. Définissent l'architecture globale du SI en termes de composants et de connecteurs. Basées sur les spécifications du type II.

- Type IV : Spécifications de l'architecture détaillée

Contrat entre la MOE (architecte) et la MOE (autres architectes et/ou développeurs).
Décrire chaque composant et chaque connecteur de manière détaillée (algorithmique).
L'architecture détaillée prépare l'implémentation des composants et des connecteurs.
Validée par l'architecte du projet.

COTS : Components off the shelf (composants sur étagère).

Rôles des acteurs organisationnels

1. Spécifications du type I : Rédigées par la MOA (ou le cas échéant par l'assistance à la MOA : AMOA).
2. Spécifications du type II : Rédigées conjointement par la MOE et la MOA.

« Nothing is certain, but death and taxes » Benjamin Franklin.

Les spécifications du type II servent de base pour la validation du SI par la MOA.
V/V (Vérification (a-t-on bien construit le SI ? efficience) / Validation (a-t-on construit le bon système ? efficacité)).
Vérification → MOE.
Validation → MOA.

3. Spécifications du type III :

Rédigées par la MOE (architecte du projet) et validées par la MOA.

4. Spécifications du type IV :

Rédigées par la MOE (architecte du projet ou développeur)
Validée par la MOE (autres architectes ou autres développeurs)

Difficultés de spécifier un SI issues :

- De la complexité du SI (qui a son tour résulte de la complexité des processus métiers supportés par le SI)

- De la volatilité des besoins des utilisateurs (qui résulte du caractère changeant du contexte organisationnel et de l'environnement externe).

Méthode de modélisation (spécifications d'un SI)

- 1er paradigme → paradigme structuré (fin des 50's) : Tom Demarco et Ed Yourdon

→ SA/SD (structured analysis, structured design)

→ D'autres méthodes ont été proposes : JSD (Jackson Software development).

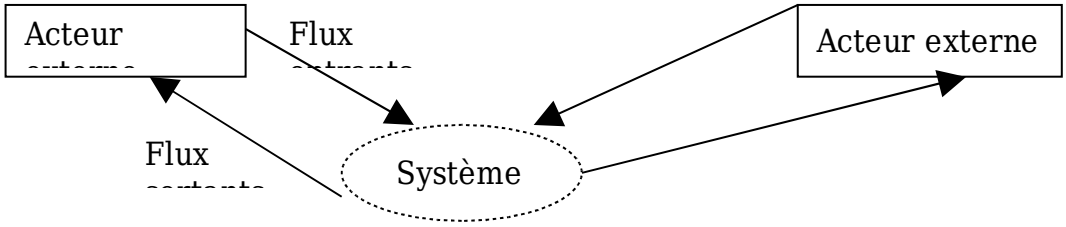
Principe : décomposer le problème d'informatisation en autant de sous-problèmes que nécessaire jusqu'à la maîtrise totale des sous-problèmes (description textuelle et algorithmique).

Avantages du paradigme structuré : Il permet de manager/gérer/maîtriser la complexité structurelle.
Inconvénients : Ne prend pas en compte la complexité systémique ou interactionnelle.

L'étude (la modélisation) d'un SI consiste à le décomposer en sous-systèmes.
Outil : **diagramme de contexte,** noté **DC** et **diagramme de flux de données**, noté **DFD**.
Concepts :

1. Acteurs externe → symbolisé par Nom acteur
2. Flux informationnel → nom Flux
3. Stockage de données → Nom stockage
4. Traitement ou fonction → traitement
5. Système à informatiser → système

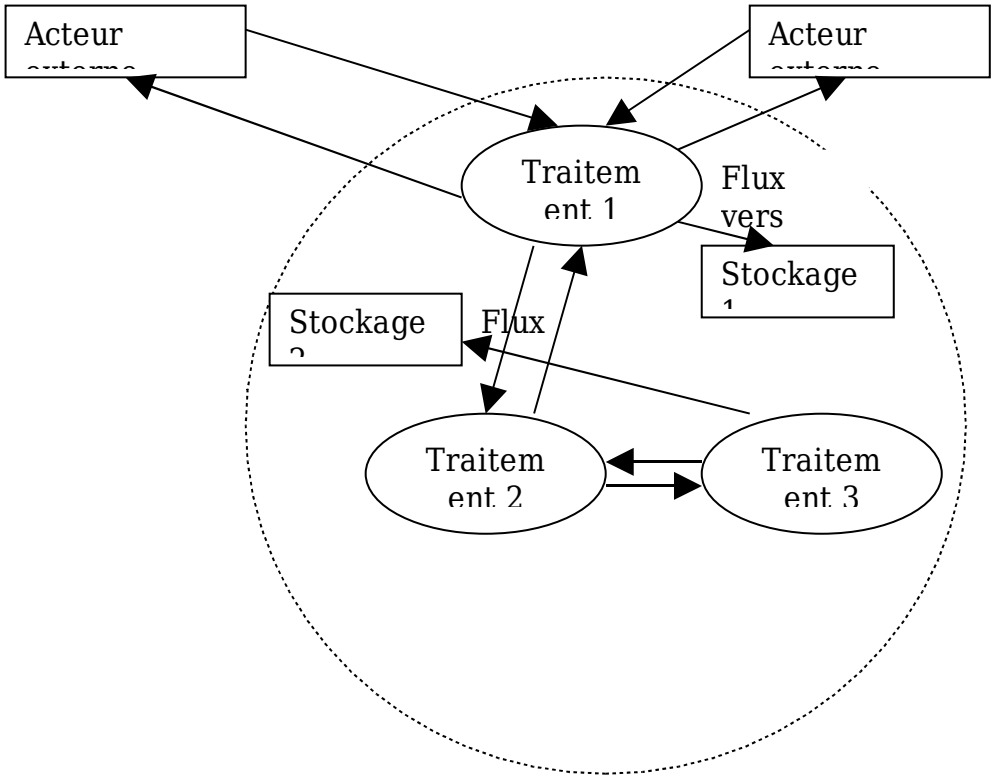
<u>Diagramme de contexte :</u>



Flux entrant : porteur d'un déclencheur d'un traitement supporté par le SI.
Flux sortant : porteur du résultat du traitement en réponse au flux entrant.

- 2eme paradigme : paradigme systémique, MERISE

<u>Diagramme du flux de données</u>
Le DFD1 : voir schéma.



Passage du DC au DFD1

1. Les flux externes entrants et sortants sont les mêmes.
2. Les traitements (fonctions) sont les acteurs internes.
3. Les acteurs internes échangent des flux internes.
4. Les acteurs internes échangent des flux avec les stockages de données.
5. Les stockages de données n'échangent pas de flux.
6. Les flux internes et externes doivent être nommés.
7. Les flux entre un acteur interne et un stockage de données ne portant pas de nom.
8. Pour des raisons liées à la lisibilité du DFD, on peut représenter les acteurs externes plusieurs fois. Dans ce cas, il faut rajouter (n-1) bandes obliques à la représentation d'un externe si cet acteur figure n fois dans le DFD.

Il en est de même pour les stockages de données mais il faut rajouter n barres verticales à la représentation d'un stockage de données si ce stockage figure n fois dans le DFD.

DFD (suite)

Rappel : le DFD est l'outil principal de modélisation des méthodes structurées (ou fonctionnelles, ou cartésiennes).

DFD : décrivent le QUOI mais ne décrivant pas le comment => Compléter la modélisation à l'aide des DFD par un processus de développement ou d'informatisation.

Le DFD ne résout pas tous les problèmes d'informatisation
→ Le compléter par d'autres outils de modélisation, y compris des outils appartenant à d'autres paradigmes.

Une moulinette : terme argotique pour désigner un processus de traitement de documents.

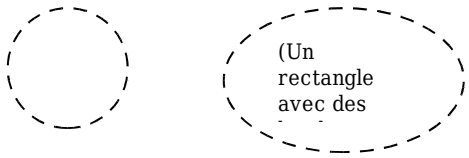
« Quand on passe d'une DFD n à une DFD n+1, on explose un traitement. » Dahkli.
En gros, on décompose les traitements en sous-traitements.

- ⇨ Les traitements non explosés deviennent des acteurs externes par rapport au niveau système basé sur les traitements explosés.
- ⇨ Les flux externes, les flux internes, les stockages de donnés et les acteurs externes changent.

Les traitements d'exception doivent être omis du DFD1. On ne peut les modéliser qu'à partir du DFD2 (pour des raisons de simplicité).

Certains flux sont porteurs d'un message particulier permettant de déclencher ou de contrôler un phénomène. Ces flux s'appellent « triggers » et sont représentés à l'aide de flèches

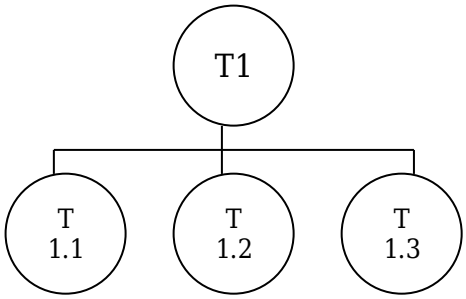
Certains traitements sont destinés à produire un service particulier, contrôle d'un phénomène. On les note



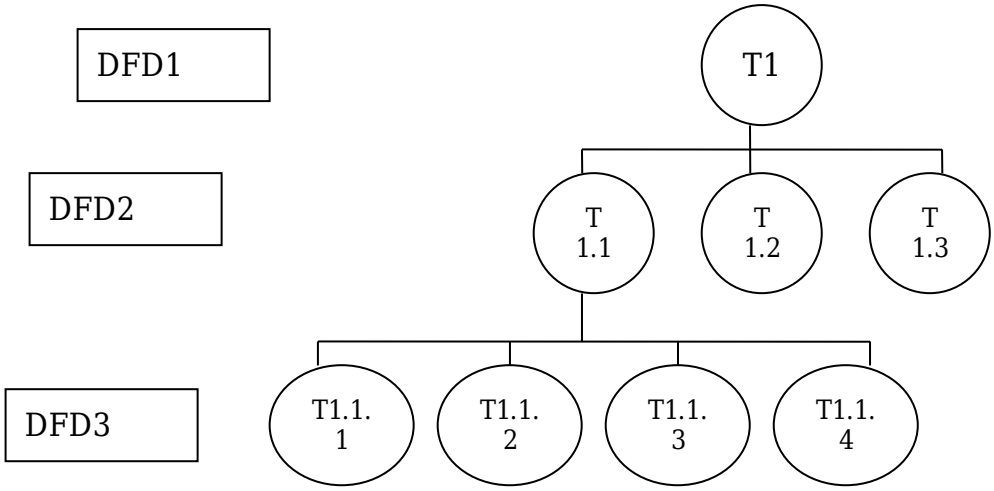
Quelques règles de construction d'un DFD.

1. Numéroter les traitements.

Soit T1 un traitement identifié dans le DFD1. Supposons que l'on explose T1 en 3 sous traitements au niveau du DFD2 => On obtient alors le schéma suivant :



Supposons qu'au niveau du DFD3, on explose le sous traitement T1.1 en 4 sous traitements :



2. Nommer les flux externes
3. Nommer les traitements. Certains acteurs recommandent l'utilisation d'un verbe d'action pour désigner un traitement. Toutefois, on peut déroger à cette règle dans les cas où il est difficile de trouver un verbe pour qualifier le traitement.
4. Nommer, dans la mesure du possible, les flux internes entre traitements.
5. Les flux internes entre traitements et stockages de données n'ont pas de nom en général. Toutefois, on peut les nommer.
6. Nommer les stockages de données.

Le nom d'un stockage de données informatisé doit être précédé par un 'D'

D base de données

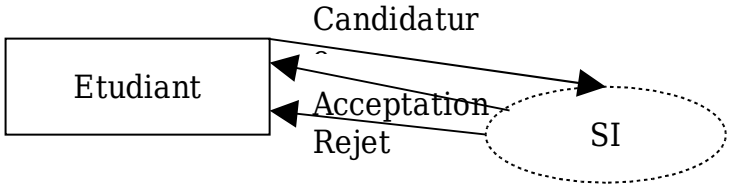
Le nom d'un stockage de données manuelle (non informatisé) doit être précédé de la lettre M.
Le nom d'un stockage de données temporaire et informatisé doit être précédé des lettres T(D).

Le nom d'un stockage de données temporaire non informatisé doit être précédé des lettres T(M).
Quelques erreurs à éviter lors de la construction d'un DFD

Faux	Correct	Description
Acteur externe 1 → Flux → Acteur externe 2	Acteur Externe 1 → FD1 → Traitement 1 → FD2 → Acteur externe 2	On ne peut pas modéliser les flux échangés par 2 acteurs externes si ces flux ne transitent pas par un traitement
Acteur externe → FD → stockage de données	Acteur externe → traitement 1 → stockage de données	Un acteur externe ne peut pas échanger des informations avec un stockage de données si ces informations n'ont pas été transformées par un traitement.
Stockage de données → FD → Stockage de données	Stockage de données 1 → FD1 → Traitement 1 → FD2 → Stockage de données 2	Deux stockages de données ne peuvent pas échanger des informations qui n'ont pas été préalablement transformées par un traitement.
→ FD → traitement 1 → FD →	→ FD → traitement 1 → FD1 →	Un flux de données output d'un traitement doit être différent du flux de données input. => une transformation est nécessaire pour que le traitement ait un sens.
3 autres erreurs		Qu'on ne détaille pas

Exercice : un étudiant envoie sa candidature pour s'inscrire à un cours dans une université. Les services d'inscription de l'université vérifient dans la liste des cours si le cours demandé est disponible actuellement et si le cours est disponible alors ces services inscrivent l'étudiant et lui envoient une lettre d'acceptation. Sinon, les services d'inscription envoient une lettre de rejet.
Construire le Diagramme de Contexte et le DFD1.

Acteur externe : étudiant
Flux de données : candidature (étudiant vers Système, flux entrant), lettre d'acceptation (système vers étudiant, flux sortant), lettre de rejet (système vers étudiant, flux sortant)
Stockage de données : Liste des cours permanente informatisée (→ D BDD liste des cours)
Processus



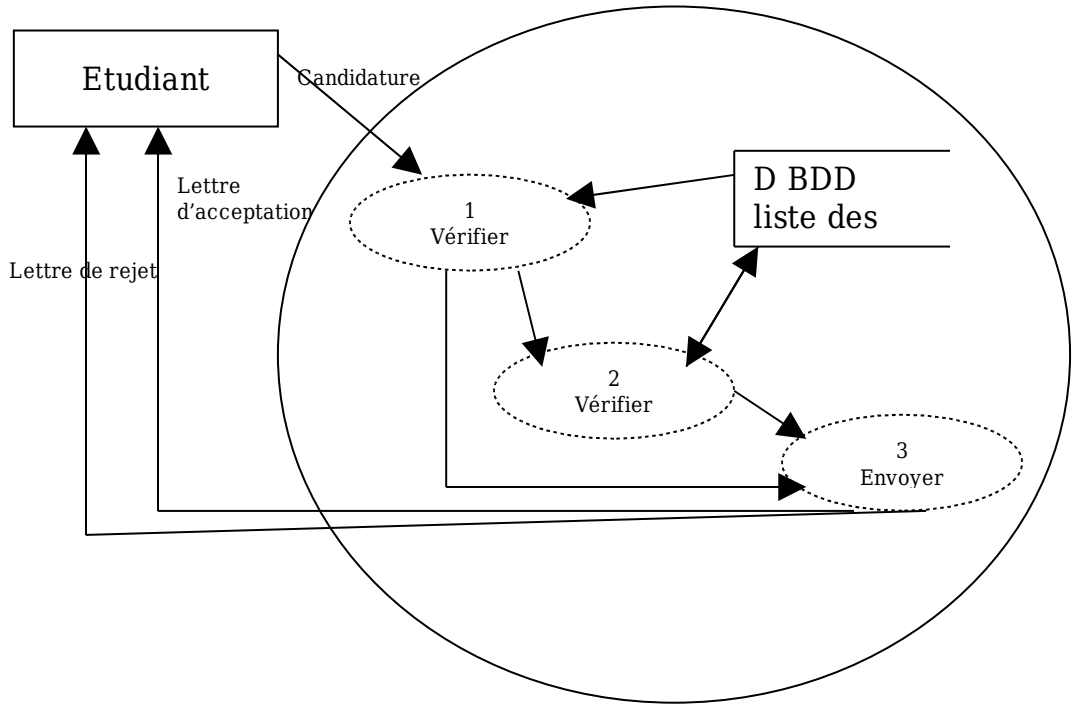
DFD1 :

→ Acteurs externes : identiques à ceux du DC

→ Flux externes : identiques à ceux du DC

→ Traitements

- Vérifier, T1
- Inscrire, T2
- Envoyer, T3



Le paradigme Systémique

Théorie des systèmes.
Système : Boîte noire avec des entrées (input) et des sorties (output).
Un SI est un système :

- C'est une organisation
- Transforme des inputs (données + informations) en output (autres données + informations)

La complexité d'un système est une complexité structurelle et une complexité interactionnelle (aussi appelée systémique).

Frederick Brook Jr. A dit « il ne faut pas ajouter des individus à un projet en retard »

La complexité structurelle résulte de la structure du projet, c'est-à-dire de son algorithmique. On a le DFD pour traiter ça (méthode structurée SA/SD).
La complexité interactionnelle a 3 étages :

- Une interaction entre les composants du système.
- Une interaction entre les acteurs organisationnels et le SI.
- Une interaction entre les acteurs organisationnels et les acteurs organisationnels.

Merise gère la complexité interactionnelle à l'aide des niveaux d'abstraction.
Il y a 3 niveaux d'abstraction :

- Conceptuel → QUOI ? invariants de l'organisation = métier (stable)
- Organisationnel (logique pour les données) → QUOI + QUI + QUAND + OU + COMBIEN ? Moins stable que le niveau conceptuel. Peut devenir stable et passer au niveau conceptuel.
- Physique → QUOI + QUI + QUAND + OU + COMBIEN + COMMENT ? niveau le moins stable, la technique change tout le temps.

« Hier, à l'époque des dinosaures, enfin, vers 1200 et quelques, quoi … » Dahkli.

Merise/2 :

- Conceptuel (pareil)
- Organisationnel (sans la logique pour les données)
- Logique → AVEC QUOI ? outils : architecture des traitements et des données
- Physique (pareil)

De plus, la version 2 a rajouté aux niveaux conceptuels et organisationnels des modèles de flux afin de mettre l'accent sur le lien entre l'acteur externe et le SI. Dans « MEGA méthode », ces diagrammes de flux s'appellent modèle conceptuel des communications (MCC) et modèle organisationnel des communications (MOC).

Merise gère aussi la complexité interactionnelle en séparant les données et les traitements, mais cela crée un autre type de complexité lié à validation conjointe des modèles de données et de traitements.

Dans le cadre de ce cours, on ne s'intéressera qu'au niveau conceptuel. A ce niveau, on distinguera le modèle conceptuel de données (MCD) et le modèle conceptuel de traitement (MCT).

Le MCD

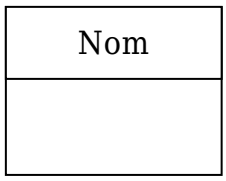
Basé sur le modèle entité-association de Peter Chen.
Décrit les informations invariantes de l'organisation. Ces informations caractérisent le métier de l'organisation et sont donc stables.
Concepts de base :

- Entité
- Association
- Attribut ou propriété
- Cardinalité
- Identifiant

Définition : Entité = objet concret ou abstrait sur lequel on veut garder des informations.

Exemple : étudiant, professeur, contrat d'assurance.
Les entités d'un MCD dépendent du contexte organisationnel → un MCD est situé.

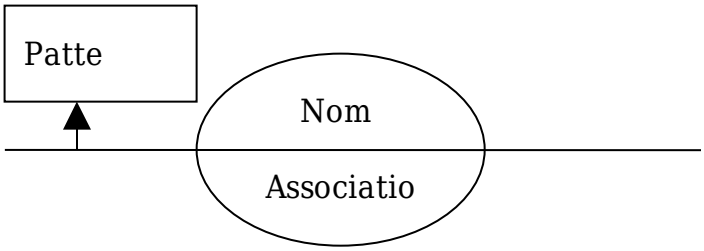
Représentation graphique :



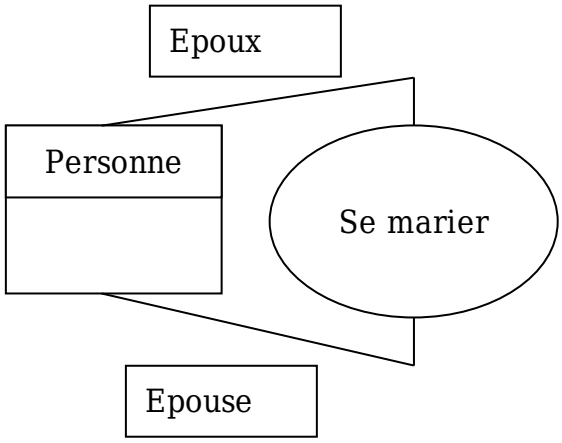
Une association est un lien entre plusieurs entités.
Exemple :
L'association « passer commande » lie les entités « client » et « commande » (association binaire)
L'association « enseigner » lie les entités « professeur », « salle » et « matière » (association ternaire)

Une association peut lier une entité à elle-même (association réflexive)

Représentation graphique :



Les pattes d'une association réflexive doivent être nommées.



Attention, une entité peut (doit ?) avoir plusieurs occurrences.

Exemple :
Personnel EFREI → Jean Klein
→ Patrice Lukowski
→ Mme Deneu
→ Pascal

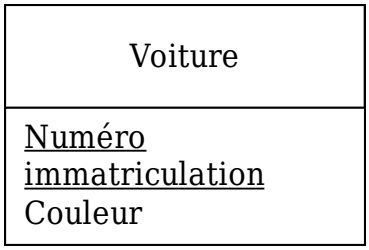
Propriété : donnée élémentaire utilisée pour décrire une entité ou une association.
La propriété est une variable qui prend une valeur particulière pour chaque occurrence.

Exemple : Voiture de Dahkli, couleur = rouge. Voiture de Lukowski, couleur = blanche.

Identifiant :
Ensemble de propriétés permettant de désigner de manière unique une entité.
Exemple : numéro d'immatriculation = identifiant d'une voiture. Numéro national d'étudiant = identifiant d'un étudiant.

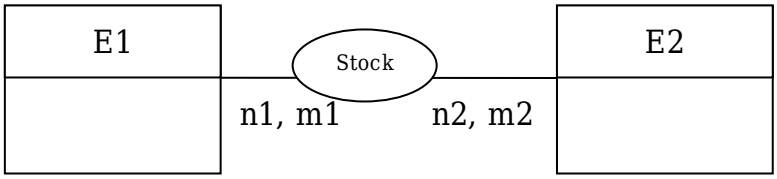
Attention, les propriétés composant un identifiant doivent être soulignées.

Exemple :



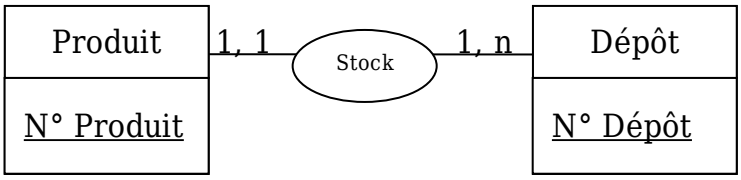
Parfois, une association qui a une propriété est dite porteuse de cette propriété. Cela signifie que la propriété en question est partagée par plusieurs entités.

Cardinalité : Information sur le nombre maximum et minimum d'occurrences de chaque entité participant à une association. Notation : Sur chaque patte de l'association, on indique le nombre min et le nombre max sous la forme n, m.

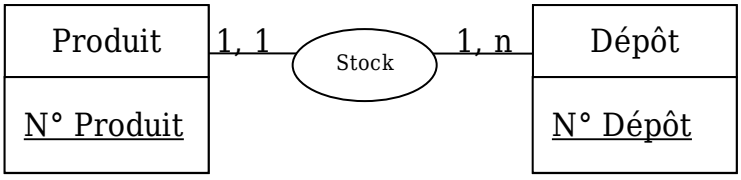


Cardinalités les plus utilisées : 0, 1. 0, n. 1, n. 1, 1.

Exemple 1 :
Un produit doit être stocké dans un et un seul dépôt. Un dépôt peut contenir au moins un produit.

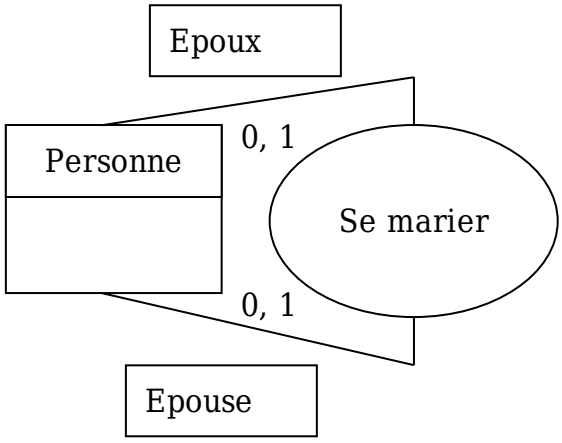


Exemple 2 :
Un produit peut ne pas être stocké dans un dépôt.
Un produit ne peut être stocké que dans un seul dépôt.
Un dépôt peut ou non contenir des produits.



Exemple 3 :

Mariages dans une société monogame.



Dépendances fonctionnelles (DF)

1. DF entre propriétés

Une propriété ou un ensemble de propriétés P2 dépend fonctionnellement d'une propriété ou d'un ensemble de propriété P1 si la connaissance de la valeur de P1 détermine une et une seule valeur de P2.
Notation : P1 → P2.

Exemple : Il existe une dépendance fonctionnelle entre l'identifiant d'une entité et les autres attributs (ou propretés) de cette entité. Identifiant → Propriété.

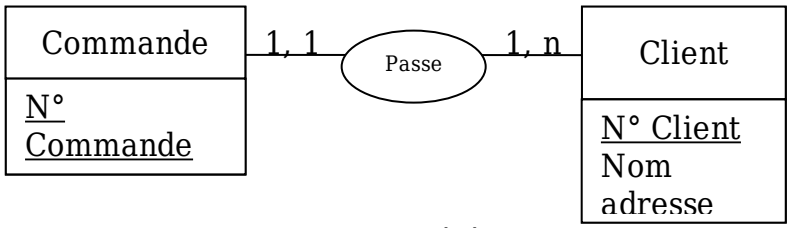
Propriété des DF entre propriétés :
Transitivité P1 → P2 et P2 → P3 => P1 → P3

Définition : une DF P1 → P2 est direct s'il n'existe pas de P3 tel que P1→ P3 et P3 → P2.

2. DF entre entités

Définition : Soient E1 et E2 2 entités liées par une association. On dit qu'il existe une DF entre E1 et E2 si à chaque occurrence de E1 n'est associée qu'une occurrence de E2.

La présence d'une cardinalité (0, 1) ou (1, 1) sur la patte associée à l'entité révèle une DF vers l'entité associée à l'autre patte.
Une DF est dite forte pour une cardinalité (1, 1) et faible pour une (0, 1)



Identifier les DF entre les propriétés de ce MCD.
Réponse :

DF entre propriétés et identifiants.
N° Client → Nom
N° Client → adresse
N° Commande → Date

Cardinalité (1, 1) => DF forte.
Commande → Client
=> DF N° Commande → N° Client
=> Par transitivité :
N° Commande → N° Client + N° Client → Nom => N° Commande → Nom
N° Commande → N° Client + N° Client → adresse => N° Commande → adresse

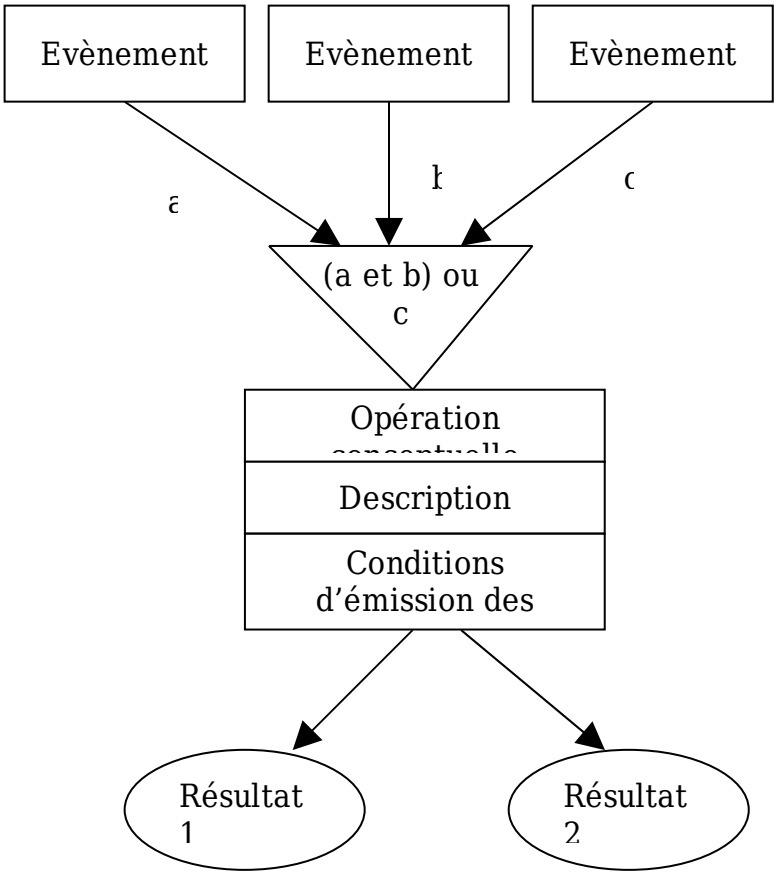
Le MCT

- Basé sur les réseaux de Pétri.
- Décrit l'enchaînement des opérations conceptuelles déclenchées par des évènements internes/externes (provenant d'un acteur externe) et produisant des résultats externes (destinés aux acteurs externes) ou internes (destinés à une opération conceptuelle).

Attention : La première opération conceptuelle est déclenchée uniquement par des évènements externes.

- Concepts de base :
    1. Opération conceptuelle
    2. Événement déclencheur externe
    3. Événement déclencheur interne
    4. Événement résultat externe
    5. Événement résultat externe
    6. Synchronisation (Opération logique du 1er ordre portant sur les évènements déclencheurs)
    7. Condition d'émission des résultats

Représentation graphique :



Une opération conceptuelle ne s’occupe pas de l’organisation du travail (rôle, temporisateurs, …).
Les évènements déclencheurs + évènements résultats.