

# **Typologie des modèles, outils et des méthodes de spécification en Génie Logiciel**

Bernard ESPINASSE

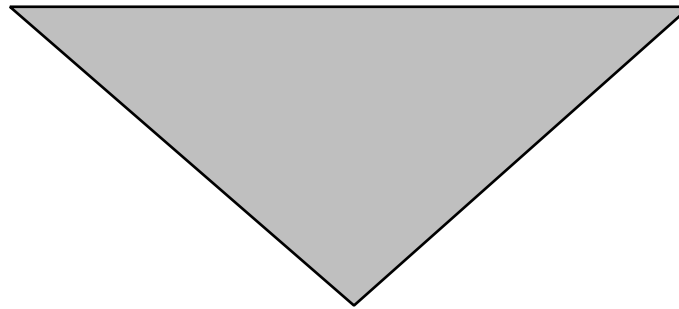
Professeur à l'Université d'Aix-Marseille

## **Plan**

- **Typologie des modèles : analytiques, conceptuels, structurels, comportementaux**
- **Typologie des outils informels ou semi-formels :**
  - **Dictionnaire de données, tables de décision, d'états-transitions**
  - **Diagrammes de flot de données, de structures, d'états-transitions,**
  - **Réseaux de Pétri et le Grafcet,**
  - **Modèle Entité-Association de base et étendu**
- **Typologie des méthodes : fonctionnelles, systémiques, orientées objet**

**Modèles**

**Outils**



**Méthodes**

- **Modèles** = **représentation abstraite** de tout ou partie du réel
- **Outils** = **formalisme**, langue permettant d'exprimer un modèle
- **Méthode** = {**modèle, outils**} + **démarche** de mise en oeuvre

# Modèles pour la spécification et la conception

- modèles pour la **spécification** du logiciel :

- exprimer les **caractéristiques** de l'objet à développer
- selon une **vue externe** (comportement, propriété, contraintes)

- modèles pour la **conception** du logiciel :

- donner une **description interne** de l'objet à développer
- la **plus explicite possible** (structure, comportement des composants)

- **Qualité générale d'un modèle**

- **abstraction** : permet de décrire le système sans faire référence aux détails de toutes ses parties
- **refinement** : un sous-ensemble du modèle doit pouvoir être décrit à l'aide d'un autre modèle :
  - du même type (description progressive)
  - d'un autre type pour compléter la description ou exprimer un point de vue différent
- **lisibilité** : le modèle doit être simple à interpréter (intérêt des représentations graphiques)

- **Classification des modèles**

- **iconique** : reproduction en miniature d'un objet réel (voiture, avion, maquette bâtiment, ... pour soufflerie)
- **analogique** : exploitent une apparence physique différente du phénomène ou objet du réel (réseau électrique ... pour une suspension de voiture)
- **analytique** : relations mathématiques et logiques pour représenter les lois physiques de l'objet
- **conceptuels** : emploi de symboles pour la représentation des aspects qualitatifs.

# Modèles analytiques

---

- très **répandus** et très **variés**
- utilisés pour **prédire** ou **estimer** (partiellement) le **comportement** de l'objet
- utilisés comme **moyen de validation**
- **classification** des modèles analytiques (Wilson 86) :

	statique	dynamique
déterministe	<i>relations algébriques</i>	<i>relations différentielles</i>
non déterministe	<i>relations statistiques et probabilistes</i>	<i>relations stochastiques</i>
	<i>modèles indépendants du temps</i>	<i>modèles dépendants du temps</i>

# Modèles conceptuels

---

permettent de :

- **clarifier une situation** (organigramme d'une société)
- **illustrer un concept** (boucle de rétroaction)
- **définir des relations entre entités d'une structure** (circulation de flux d'information)
- **définir une méthode**
- **classification** des modèles conceptuels (Wilson 86) :

	structurel		comportemental	
activités				
données				
	vertical	horizontal	continu	discret

# Outils pour développer des Modèles de structures et de comportements

---

- Outils pour des modèles de **structures**
  - **pour les activités** : diagramme de flots de données (DFD)
  - **pour les données** : diagramme de Jackson, Entité-Association, modèles objets,...
  - **pour les fonctions** : diagrammes hiérarchiques de fonctions, Process Structure Graph (DARTS), diagramme de Booch,...
- Outils pour des modèles de **comportement**
  - **mathématiques** : décrit le domaine des variables d'entrée et de sorties et la transformation des entrée vers les sorties
  - **formels** : langage Z, modèle explicite dans VDM (Vienna Development Method -IBM Vienne),...
  - **pseudo-code**
  - **automates à états finis**
  - **state charts (Harel 87)**
  - **réseaux de Pétri**

# Présentations formatées

---

- spécifications écrites uniquement en **langage naturel** (même respectent plans types normalisés : STD 830, DoD 2167-A) posent des problèmes de non cohérence, d'ambiguïté, de non complétude

=> **présentations formatées** les plus connues :

- **le dictionnaire de données**
- **les tables de décision, les tables d'état-transition**

## Dictionnaire des données ou glossaire

---

- spécifications des données utilisées aux différents niveaux d'analyse et de conception
- contient en général les définitions des termes utilisés classées par ordre alphabétique
- présente des sigles, codes ou symboles employés dans les documents, précise les synonymes, alias.
- permet de définir la structure d'une donnée (notation syntaxique stricte - Naur-Backus)
- peut intégrer des informations sur les fichiers contenant les données et les processus qui les utilisent
- peut indiquer le nombre des versions stockées (pour chaque information) avec dates de création ou de modification



## Table de décision

---

- représentation tabulaire de tous les cas des **valeurs d'entrée d'un processus** et des **valeurs de sortie** correspondant à chacune de ces combinaisons
- adaptée à la spécification de **systèmes dont les sorties sont, à tout moment, uniquement définies par les entrées** :

condition 1	O	O	O	O	N	N	N	N
condition 2	O	O	N	N	O	O	N	N
condition 3	O	N	O	N	O	N	O	N
action 1					X	X	X	X
action 2		X						
action 3	X							
action 4			X					

## Table états-transitions

---

- composée de **colonnes** représentant les **différents états du système**
- pour chaque **état**, les **événements** qui provoquent des **transitions** d'un état à un autre, les **actions** à effectuer et **l'état suivant** pour chaque **transition**
- adaptée à la spécification de **systèmes** dont **les sorties** sont **déterminées par les entrées** et **l'historique** des **états antérieurs**
- représentation similaire : **matrice états-transitions**

# Outils graphiques ou semi-formelles

---

- **favorisent communication** entre développeurs du système et futurs utilisateurs de celui-ci
- introduisent un **aspect formel**
- en général accompagné de "**textes**" **informels**

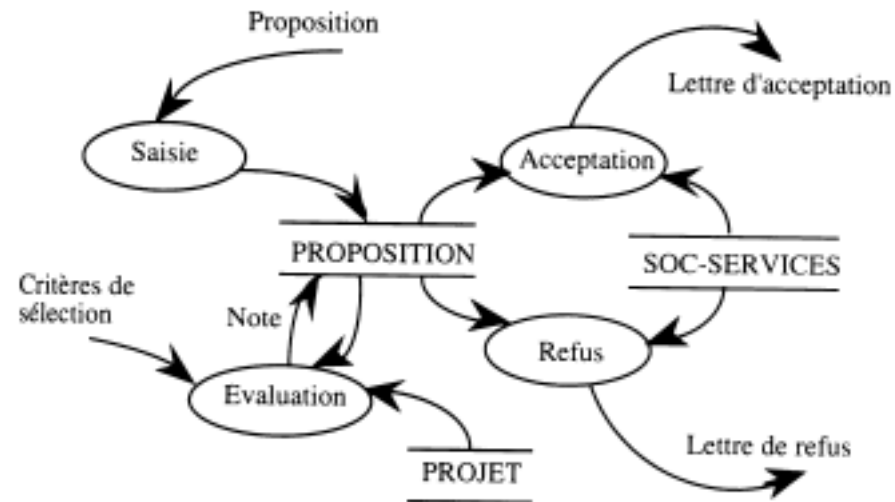
=> techniques "**semi-formelles**"

- **outils graphiques ou semi-formels les plus utilisés :**
  - les diagrammes de flot de données (DFD)
  - les diagrammes de structures
  - les diagrammes d'états-transitions
  - les réseaux de Pétri et le Grafcet
  - l'Entité-Association de base et étendu

# Diagrammes de flots de données (DFD) : Data Flow Diagrams)

- **notationS** : Myers (1975), Yourdon (1975), Constantine&Yourdon (1979)
- **intégrés** dans **diverses méthodes**
- utilisés pour la **modélisation des traitements**
- permettent de montrer **comment chaque processus transforme ses entrées** (flots de données entrants) **en sorties correspondantes** (flots de données sortants)
- concepts majeurs :
  - **noeud processus**
  - **arc orienté : flot de données**
  - **dépôt de donnée** : stockage de données = regroupements de données utilisables par tout processus
- souvent accompagnés de **diagrammes de contexte** présentant les échanges de flots de données avec les acteurs extérieurs au système à modéliser
- bien **adaptés à la description de systèmes réactifs** (systèmes toujours prêts à réagir à l'arrivée de données)

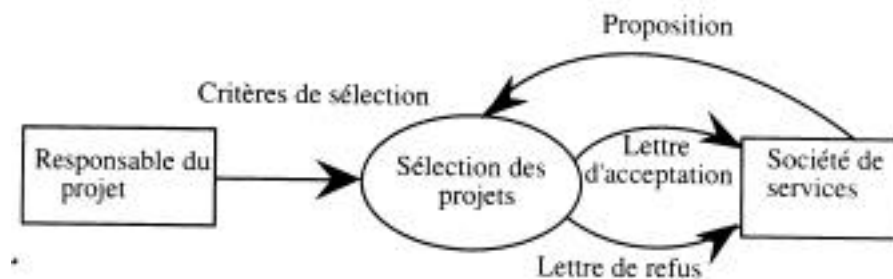
# Exemple de Diagramme de flots de données



- le processus "Évaluation" prend en compte des critères obtenus à partir du flux entrant des "Critères de sélection" pour évaluer une proposition rangée dans la zone de stockage "proposition" par rapport à un projet rangé dans la zone de stockage "projet".
- la note attribuée (flux sortant) est rangée dans la zone de stockage "proposition".

# Exemple de Diagramme de contexte associé à un DFD



Le diagramme de contexte permet de présenter les échanges de flots de données avec les acteurs extérieurs au système à modéliser :



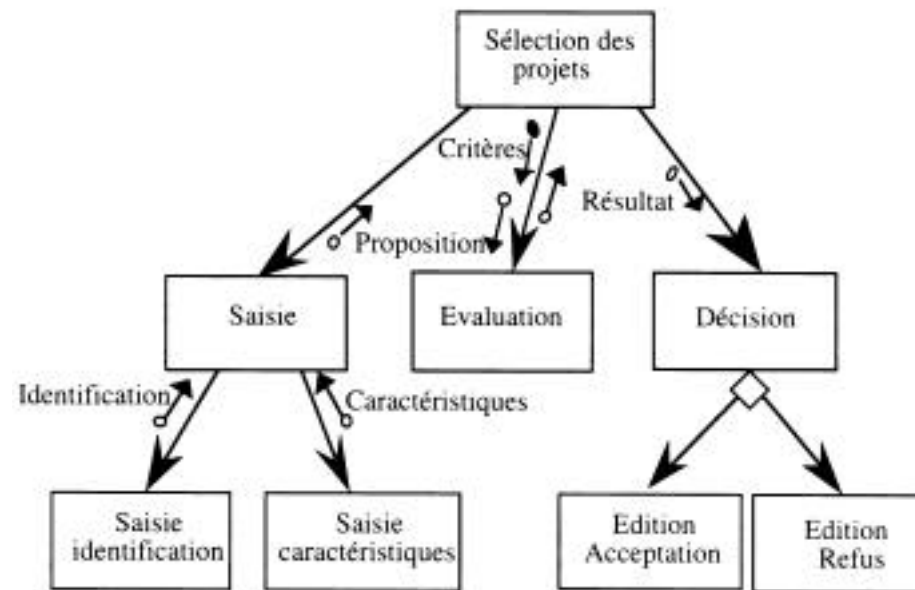
- le flux entrant correspondant aux "critères de sélection" provient d'un acteur extérieur "responsable du projet"
- une "société de service" est aussi un acteur extérieur qui fournit une proposition.

# Diagrammes de structure : Structured Charts

---

- **introduits** par Constantine, Yourdon & Myers (1979)
- permettent de **décrire l'architecture d'un système**, comme une **hiérarchie de fonctions**
- par un **arbre, à lire de gauche à droite**
- une connexion entre 2 fonctions est représentée par une **flèche orientée de la fonction appelante vers la fonction appelée**
- les **paramètres d'entrée et de sortie** sont identifiés par de petites flèches orientées :
  - données des paramètres **destinées à être traitées** :  

  - données des paramètres servant au **contrôle** :  

- il est possible de traduire par des symboles particuliers les **structures itératives** (boucles) et **alternatives** (choix parmi différents cas)

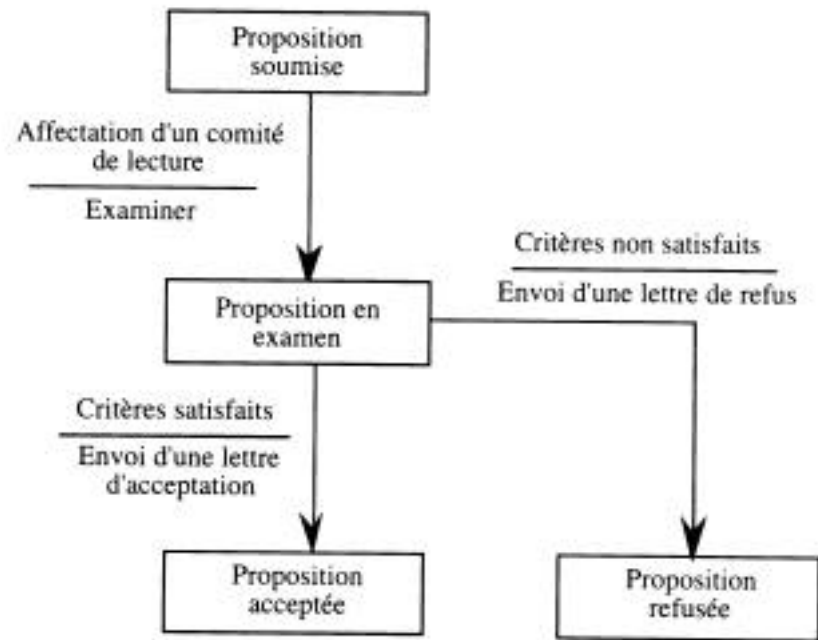
# Exemple de Diagramme de structure



- la hiérarchie du système "Sélection des projets"
- la fonction "Évaluation" opère sur les données "Proposition" (sortie de la fonction "Saisie") à partir des informations de contrôle "Critères" et retourne un "Résultat"
- la fonction "Décision" donne lieu à une alternative.

# Diagrammes états-transitions :

- permettent de spécifier l'**incidence des événements** sur les différents états du **système** en indiquant les actions à effectuer
- utile pour la réalisation des **tables ou des matrices états-transitions**.
- adaptés pour modéliser le **cycle de vie d'un objet** (cf. méthodes d'analyse orientée objet)



- une proposition peut passer par différents états : soumis, en examen, accepté, refusé
- les changements d'états sont dus aux événements : "Arrivée d'une proposition", "Affectation d'un comité de lecture", "Critères satisfaits" ou "Critères non satisfaits".
- préciser ainsi les valeurs prises par un attribut "état" (défini par ailleurs dans un modèle de données)



# Réseau de Petri (RdP)

---

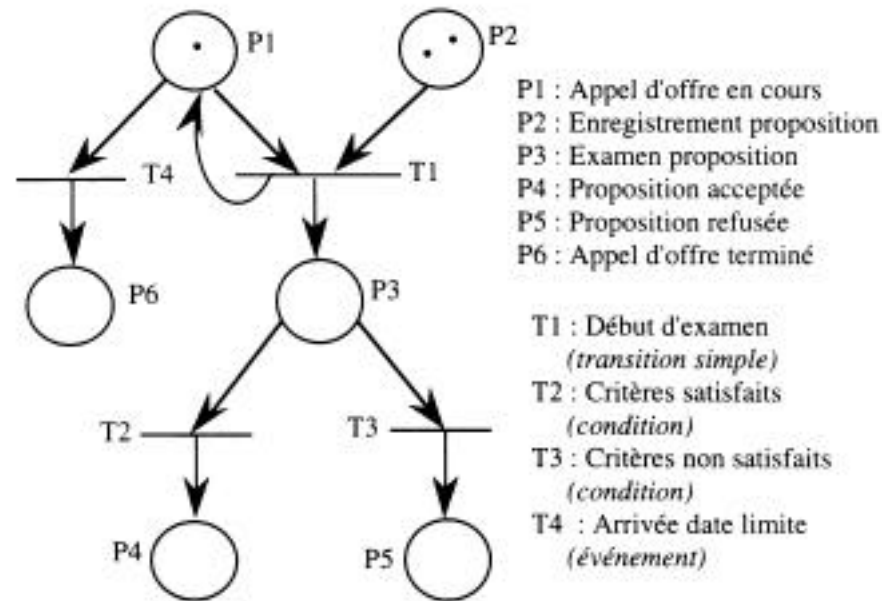
- outil mathématique pour la modélisation du **comportement** d'un système dynamique à **événements discrets**
- utilise :
  - des **places** (cercles), correspondant aux différents états du système,
  - des **transitions** (traits) associées aux changements d'états et
  - des **arcs** reliant places et transitions.
- des techniques de **marquages** (jetons) permettent de définir l'état du système à un moment donné
- aux transitions peuvent être associés des **conditions** ou des **événements** externes. On parlera alors de réseaux de **Petri interprétés**
- **divers RdP** : colorées, synchronisés, ....
- **a inspiré d'autres modèles de comportement : formalisme de traitement MERISE**

## Grafcet

---

- résultat d'un groupe de travail AFCET = outil de spécification des automates logiques, inspiré des RdP
- **norme internationale** (1987), très utilisé par la communauté française des **automaticiens**

# Exemple de Réseau de Petri



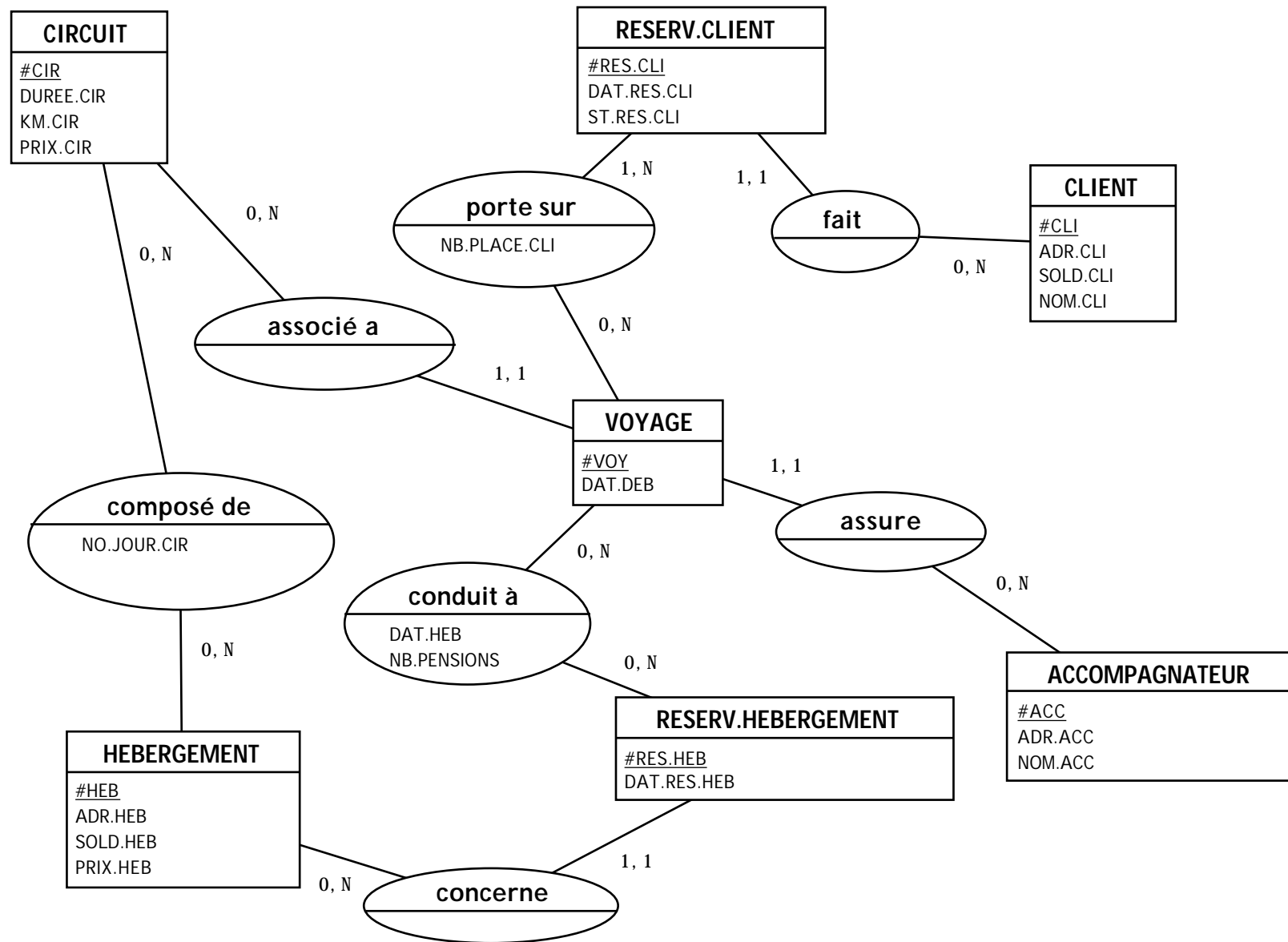
- représente les états du système à partir de l'appel d'offre auprès de plusieurs sociétés jusqu'à la sélection des solutions proposées.
- tant que l'appel d'offre est en cours (1 jeton dans P1), dès qu'une proposition est enregistrée (1 jeton dans P2), on peut passer à l'état "Examen propositions" (1 jeton est consommé dans P1, 1 jeton est consommé dans P2, 1 jeton est produit dans P3 et un jeton est produit dans P1 car on reste dans l'état "Appel d'offre en cours").

# Modèle Entité-Association

---

- modèle des données (P.Chen), sous le nom Entity-Relationship Model
- connu en France sous **différentes appellations** :
  - le modèle Entité-Association, Entité-Liaison, Entité-Relation, Objet-Relation ou Modèle Individuel.
- repose sur différents concepts :
  - **entités**,
  - **associations**,
  - **attributs d'entités ou d'associations**,
  - **cardinalités**.
- permet d'identifier et de **caractériser les objets** du domaine et d'**établir leurs liens**,
- les **cardinalités** donnent des renseignements sur le **minimum** et le **maximum** d'occurrences d'une association liant une entité à une autre
- utilisé dans de **nombreuses méthodes d'analyse** sous sa **forme basique** ou sous une **forme étendue** prenant en compte des concepts introduits par les méthodes orientées objet
- bien adapté à la **conception de bases de données**

# Exemple de modèle Entité-Relation (basique)



# Typologie des méthodes d'analyse et de conception

---

- nombreuses méthodes se sont développées en suivant l'évolution des langages et techniques
- différentes manières de les classer
- **approches descendantes / approches ascendantes :**
  - **méthode descendante** : on décompose le système de base en sous-systèmes, chacun d'eux pouvant être ensuite redécomposé jusqu'à l'obtention de modules programmables "simplement".
  - **méthode ascendante** : on part de modules déjà existants que l'on essaie de composer.
  - certaines méthodes insistent beaucoup sur une conception en vue d'une **réutilisation**
  - d'autres sont un **compromis** entre les 2 approches citées
- **classification communément admise :**
  - les méthodes **dirigées par les données**
  - les méthodes **fonctionnelles** (dirigées par les traitements)
  - les méthodes **systemiques**
  - les méthodes **orientées objet**.

# Méthodes Dirigées par les Données

---

- assez **anciennes** (1975-1980)
- reposent sur l'idée que **la structure d'un logiciel peut se déduire de la structure des données** apparaissant en entrée et en sortie de ce logiciel.
- **pratiquement plus utilisées** actuellement en France
- **particulièrement bien adaptées** à certains **problèmes simples** d'informatique de gestion
- **traitent mal les problèmes complexes**, en particulier de **synchronisation** => plus vraiment adaptées au développement des logiciels actuels.
- **méthodes dirigées par les données les plus connues :**
  - méthode de **J.D.Warnier** ou **LCS** (Lois de Composition des Systèmes), "Construction logique de programme", LCP Éditions d'organisation, 1976.
  - méthode de **M.Jackson**, "Principles of Program Design", Academic Press, 1975
  - ...

# Les méthodes Fonctionnelles

---

- ont leur origine dans le développement des **langages procéduraux**
- **très utilisées**
- plus **orientées vers les traitements** que vers les données
- mettent en évidence **la ou les fonctions à assurer** et proposent une **approche hiérarchique, descendante et modulaire** en précisant les **liens** entre les différents **modules**.
- utilisent souvent des **notations de type DFD**
- avec l'évolution des langages de programmation et des systèmes, prennent en compte la **modélisation des données** et les problèmes posés par le **temps réel**
- **méthodes fonctionnelle les plus connues :**
  - **SA-SD** (Strutured Analysis -Structured Design - Yourdon, DeMarco, Constantine, Gane & Sarson,...)
  - **SADT** (Structured Analysis and Design Technique -
  - **SA-RT** (Strutured Analysis -Structured Design - Hatley & Pirbhai 1991) spécialisé temps réel
  - ...

# Les méthodes Systémiques

---

- méthodes s'appuyant sur une **approche systémique**
- définissent différents **niveaux de préoccupation ou d'abstraction**
- proposent de **nombreux modèles complémentaires**
- sont souvent **spécialisées** pour la conception d'un **certain type de systèmes**
  
- **méthodes systémiques les plus connues :**
  - **MERISE** (méthode la plus utilisée en informatique de gestion en France et grande partie de l'Europe)
  - **AXIAL** (IBM - systèmes d'information), **MEGA** (Mega - systèmes d'information),...
  - **OSSAD** (systèmes bureautiques)
  - **SAGACE** (CEA - systèmes complexes (centrales atomiques))
  - **GRAI** (Productique)
  - ...



# Les méthodes Orientées Objet

---

- influencées par le développement du langage **Ada** et des langages de programmation basés sur les objets **C++**
- la plupart aborde l'étude d'un problème est réalisée suivant 3 aspects :
  - aspect **statique** : identifie les propriétés des objets et leurs liaisons avec les autres objets,
  - aspect **dynamique** définit le cycle de vie des objets en précisant : le **comportement** des objets, les **différents états** par lesquels ils passent et les **événements** qui déclenchent ces changements d'états.
  - aspect **fonctionnel** : précise les fonctions réalisées par les objets par l'intermédiaire des méthodes
- **méthodes orientées objet les plus connues :**
  - **OMT** (Rumbaugh et al. 1995)
  - **OOA** (Object Oriented Analysis - Coad & Yourdon 1992)
  - **BOOCH** (Booch 1991)
  - **OOA** (Object Oriented Analysis - Shlaer & Mellor 1992)
  - **Objectory-OOSE** (Jacobson & al. 19XX)
  - **HOOD**
  - ...