Module OMGL - UE Modélisation des données

Abstraction - Modélisation

J. Christian Attiogbé

Université de Nantes - Dpt Info IUT

Septembre 2008, maj octobre 2009, août 2010

Objectifs (selon le PPN[1])

Objectifs du cours (MoDon)

Acquisition de concepts, méthodes et outils permettant une mise en œuvre rigoureuse et maîtrisée des systèmes informatiques

Cette mise en œuvre va :
de l'énoncé des besoins (cahier de charges) de l'utilisateur

à une spécification opérationnelle,

en vue d'aboutir à l'installation d'un logiciel

conforme à la spécification extraite des besoins.

[1] Programme pédagogique national

Modélisation : où se situe-t-on ?

$$Informatique = \begin{cases} \text{Software (Génie logiciel) : construire des logiciels} \\ \\ \text{Hardware : construire des matériels (ordinateurs)} \end{cases}$$

On modélise ce qu'on va construire, les données et des traitements

Modélisation : une étape d'abstraction/conceptualisation en amont de la programmation.

Quel peut être le modèle d'un logiciel ?

Contenu prévisonnel du module (MoDon)

- Modélisation logique et ensembliste
- Raisonnement logique
- Analyse des besoins, propriétés fonctionnelles, non-fonctionnelles, correction
- Systèmes d'information : paradigmes d'analyse et de conception
- Modèle relationnel de Codd
- Familles de langages de modélisation
- Du cahier de charges au projet, à l'atelier de développement
- Etudes de cas

Quels intérêts

Pour construire, on a besoin de :

- Outils
- Méthodes
- Techniques
- et plus spécifiquement de langages, d'astuces/recettes/techniques.

Plus directement, quotidiennement,

- travail en amont de l'algorithmique et de la programmation ;
- structuration des systèmes d'information, des bases de données ;
- analyse des systèmes informatisés
- ...

Organisation et volumes horaires

- Cours (7 séances de 1h20) : C. Attiogbé
- TD/TP 18,40 h (7*2 séances de 1h20)
  C. Attiogbé, N. Hadj-Rabia, J-F Hue
- Travail personnel : passionnément
- Contrôle : continu + surprise + à mi-parcours : 1-2 h
- Suivi des TDs, compte-rendus, ...
- Devoir surveillé final

Plan de ce chapitre

Modélisation formelle

En Sciences et Techniques,

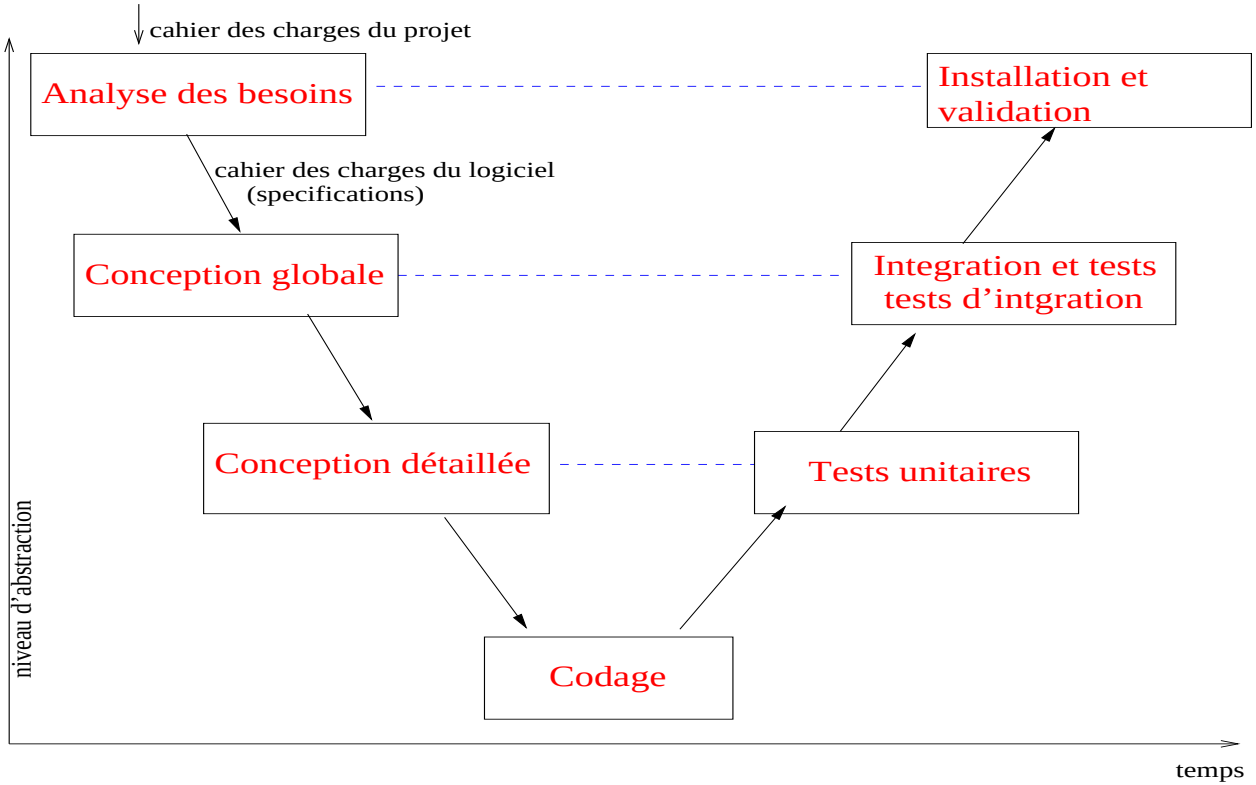
on construit à partir de modèles (abstraits, formels,...) :
Les modèles sont *imaginés*/*conçus* par les techniciens/ingénieurs pour représenter le réel ; mais cela n'en est qu'une abstraction.

- bâtiment (plans de l'architecte, ...)
- génie civil (plans, spécifications techniques ...)
- génie électrique (plans ...)
- génie logiciel (modèles formels)
- ...

Modélisation : où se situe-t-on ?

Informatique > Génie logiciel > construction des logiciels

Cycle de vie du logiciel



Modélisation : suite de la phase d'analyse.

Modélisation : où se situe-t-on ?

- *Hardware* (matériel)
- *Software* (logiciel)
- Système d'exploitation (*Operating system*)
- Ordinateur = matériel + logiciel de base [+ logiciel d'application]
- Système informatique
- Système d'information

ACSI : Analyse et Conception des Systèmes d'Information
Analyse des besoins et modélisation : *Requirement Analysis*
Modèle des données, modèles des processus, interaction

Modélisation formelle(suite)

On raisonne à partir de modèles (abstraits, formels,...) :
soit d'objets existants ou de phénomènes naturels observés,
soit d'objets futurs qu'on veut construire ;

- bâtiment (plans de l'architecte, ...)
- génie civil (plans, spécifications techniques ...)
- génie électrique (plans ...)
- génie logiciel (modèles formels)
- ...

☛ Besoin de **concepts** pour modéliser rigoureusement

Modélisation

Modélisation :

HOARE : *A scientific theory is formalised as a mathematical model* ***of reality****, from which can be deduced or calculated the observable properties* ***and of a well-defined class of processes*** *in the physical world.*

Il y a deux principales notions de modèles (en informatique).

1. Modèle = une approximation de la réalité par une structure mathématique.
2. Un objet O est modèle d'une réalité R, si O permet de répondre aux questions que l'on se pose sur R.

En Mathématique, Physique, ...: systèmes d'équations portant sur des grandeurs (masses, énergie, ...) ou des lois hypothétiques.

Modélisation

$$f(x) = \begin{cases} x < 0 & y = -x \\ \\ x >= 0 & y = x + 4 \end{cases}$$

$$f : Entier \rightarrow Entier$$
$$x \mapsto y$$

Entier + - * < <= > . . . Entier

Figure: Idée d'un modèle

Modélisation : exemple

Etat d'occupation d'une salle

Construire un logiciel qui contrôle l'affichage d'un panneau en fonction de l'utilisation des places dans une salle.
Sur le panneau, toutes les places apparaissent avec :
un point lumineux vert quand la place est libre,
un point lumineux rouge quand la place est occupée.

Comment faire pour construire ce logiciel ?

Quelles sont les données, les traitements ? les propriétés ?
Analyse du besoin > Abstraction > Conception > Codage > ...

Architecture

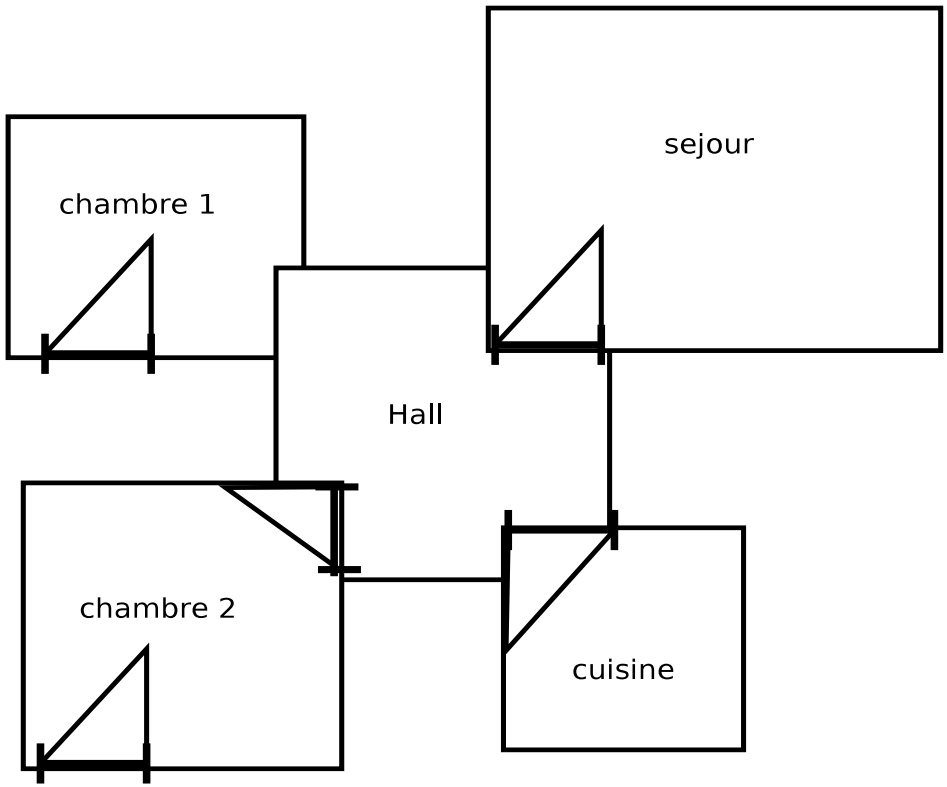


Figure: Plan d'une maison

Raisonnement

Peut-on aller dans toutes les pièces en passant par le Hall ?

Génie Civil

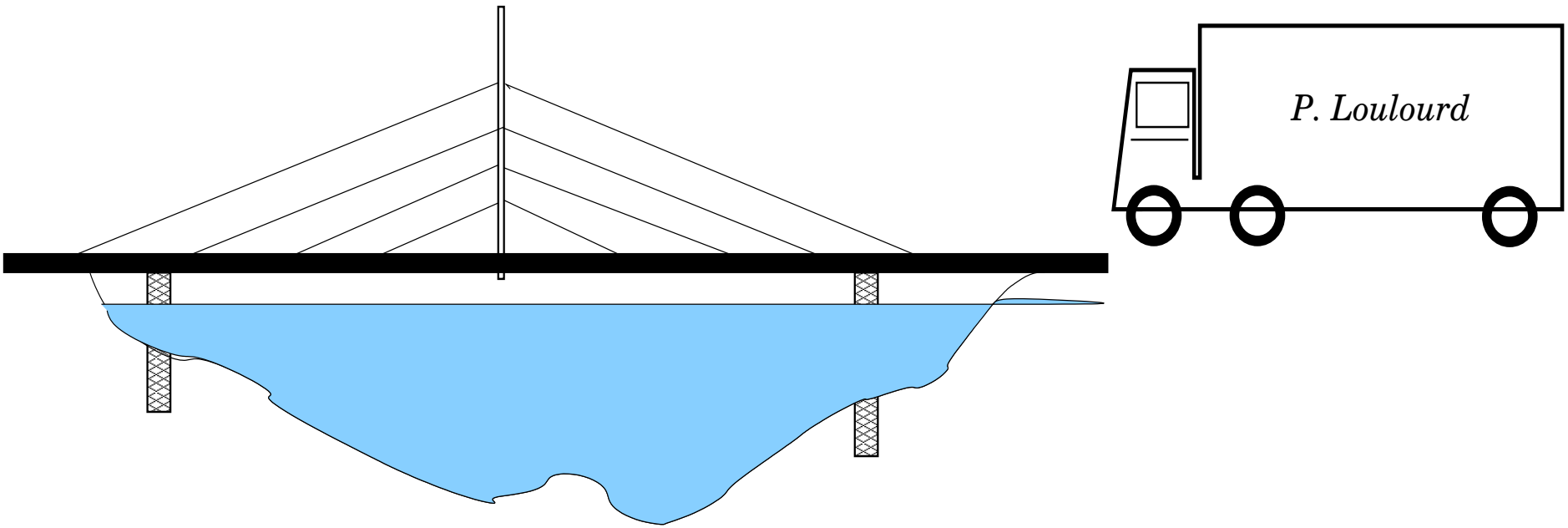


Figure: Schéma/modèle d'un pont

Est-ce que le pont peut supporter le poids des véhicules de plus de 40 tonnes ?

Modélisation à l'aide de graphe

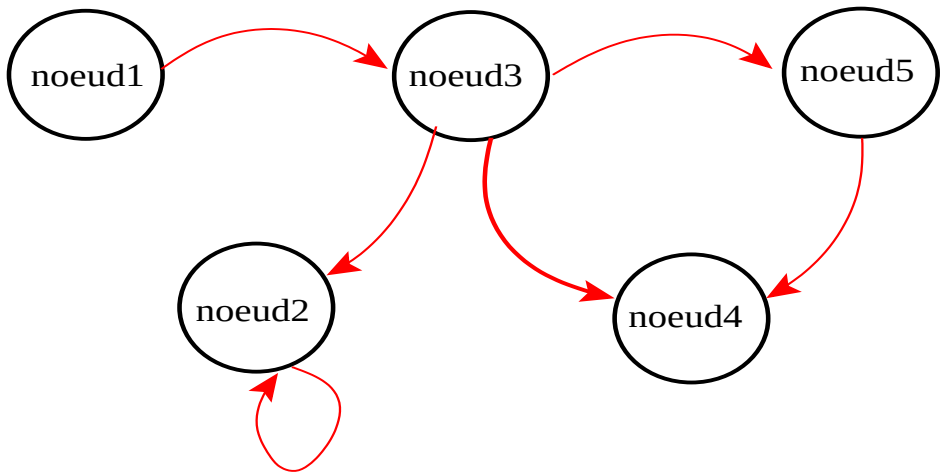


Figure: Modèle sous forme de graphe

Est-ce que le graphe est connexe ?

Modélisation à l'aide de graphe

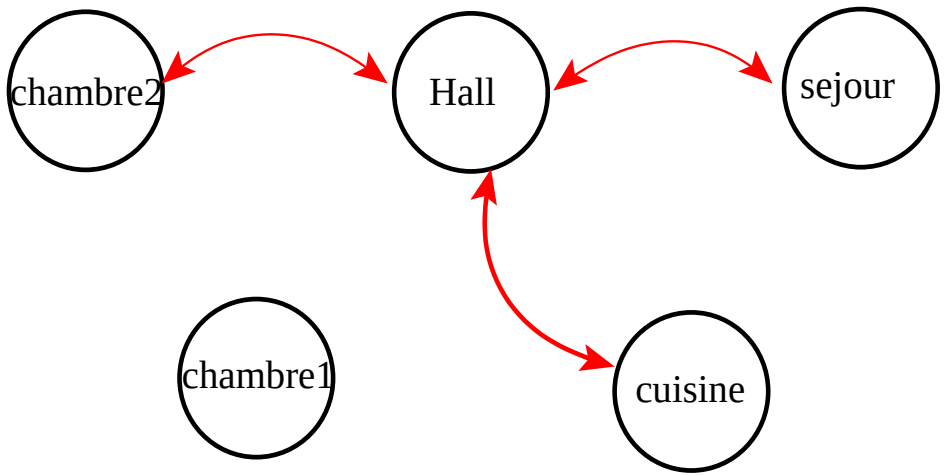


Figure: Modèle d'une maison à l'aide de graphe

Est-ce qu'on peut aller du Hall vers toutes les chambres ?

Construction du logiciel (programmes)

Place de la modélisation (spécification) dans la démarche de construction de programmes/logiciels.

Puis

- les autres étapes de la construction,
- les tâches (ou activités) et les rôles (des acteurs).

Construction du logiciel (programmes)

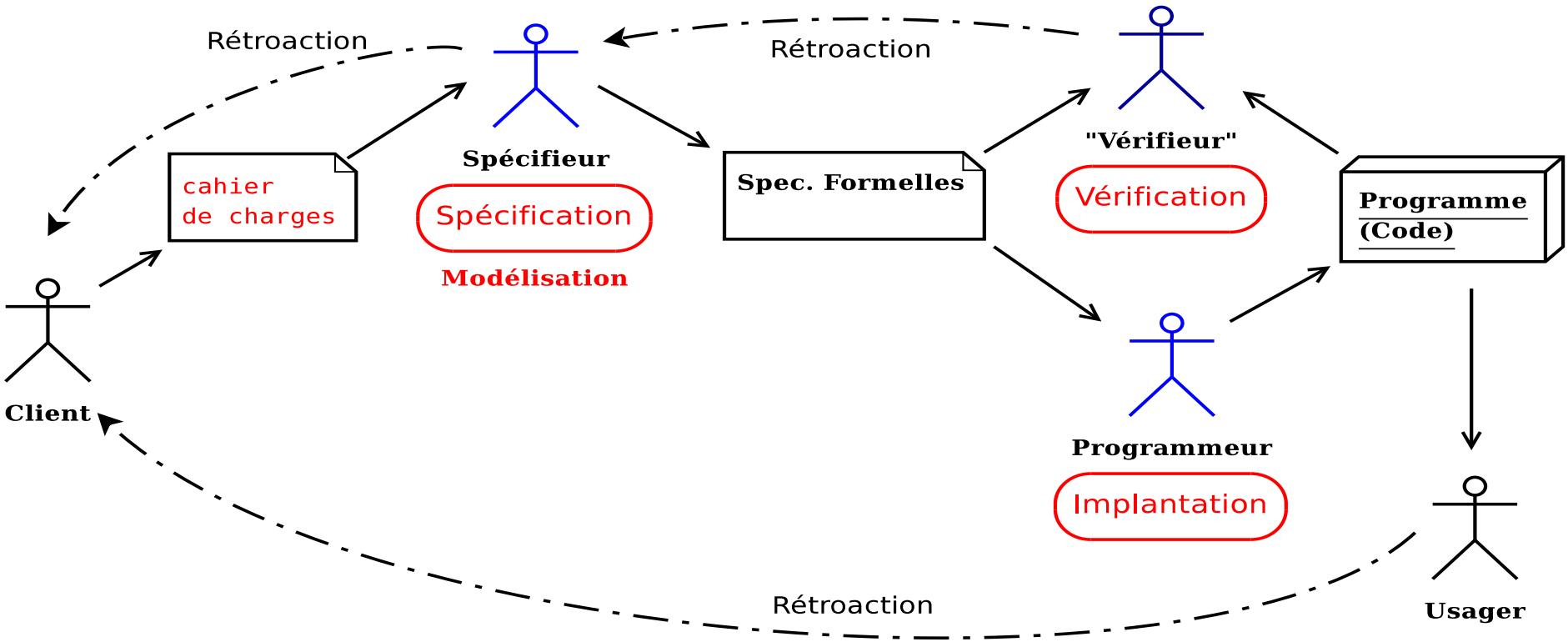


Figure: Aperçu du cycle de construction du logiciel

Abstraire, c'est quoi ?

Abstraire : Oter les détails pour ne retenir que la principale substance/caractéristique/propriété qui donne du sens à ...

Abstraction : la principale façon de construire des modèles
= structurer des données et les contraindre par des propriétés.

Abstraction

- Soit à manipuler une chaise dans un programme (logiciel).
- Comment la modéliser (la représenter) ?
- Qu'est-ce qu'une chaise ?

Abstraction

Qu’est-ce que c’est ?

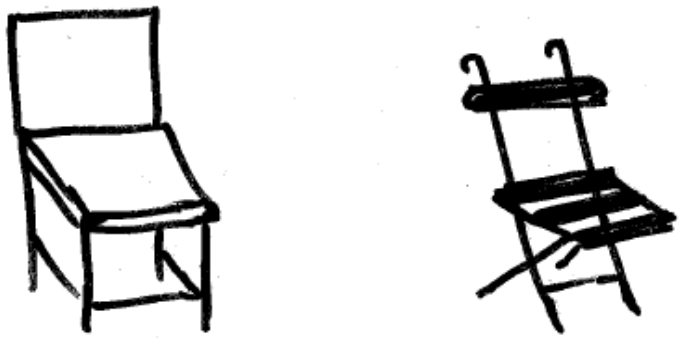


Figure: Dessins (abstraits) de chaises

Abstraction

Qu’est-ce que c’est ?

Figure: Dessins (abstraits) de chaises

Abstraction

Qu'est-ce que c'est ?

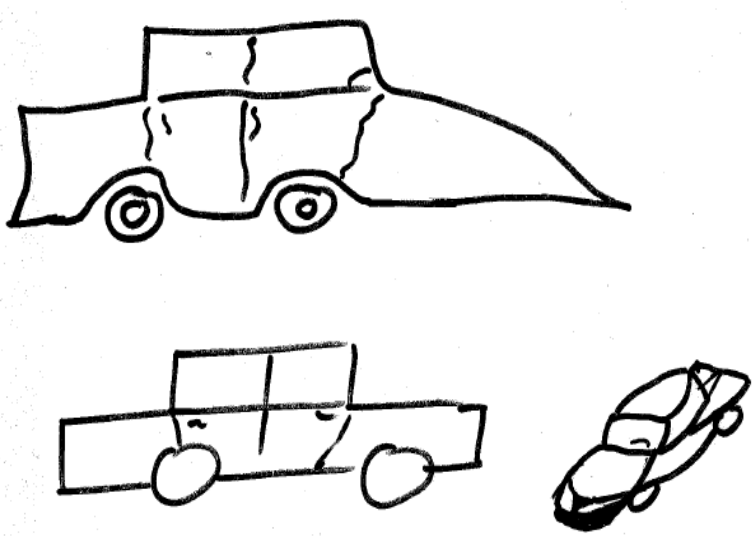


Figure: Dessins (abstraits) de voitures

Abstraction

Qu’est-ce que c’est ?

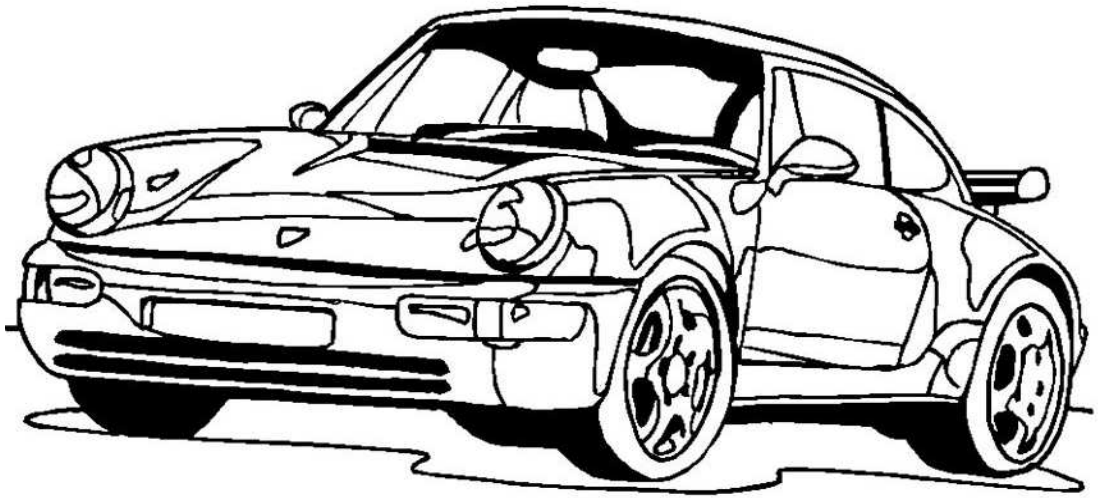


Figure: Dessin de voiture

Abstraction

Attention, il faut néanmoins avoir suffisamment d'information !

Qu'est-ce que c'est ?

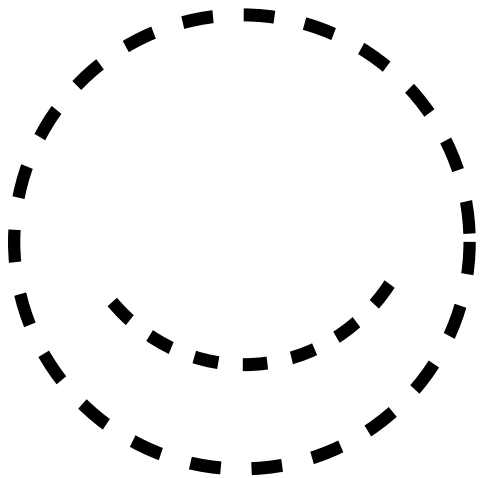


Figure: quoi ?

Abstraction

Attention, il faut néanmoins avoir suffisamment d'information !
Qu'est-ce que c'est ?

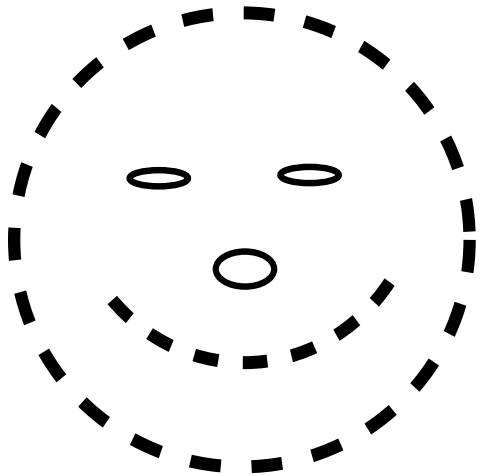


Figure: quoi ?

Différentes techniques d'abstraction

Abstractions orientées Propriétés

Abstraction orientée propriétés : une spécification qui met l'accent sur des propriétés (logiques, algébriques) qui représentent l'objet d'étude.

Abstractions orientées Modèles

Abstraction orientée modèles : une spécification qui met l'accent sur des structures mathématiques (telles que ensembles, relations, produits cartésiens, fonctions, etc) qui représentent l'objet d'étude.

Et aussi : catégorisation, instanciation, composition

Modélisation logique

- Soient les propositions :
  - P_1 x n'a pas le forfait F
  - P_2 x a plus de 2 ans d'ancienneté
  - P_3 x a 10 des services de la liste L

  Que représente :

  $$(P_1 \wedge P_2) \vee P_3 \ ?$$

  Reécrivez sous forme de prédicat (en introduisant des variables).

- Soit l'énoncé suivant : tous les abonnés à 5 ans d'acienneté sont concernés

  $\forall x.(abonne(x) \wedge (anciennete(x) \geq 5 \Rightarrow concerne(x)))$

Modélisation ensembliste

Soient les ensembles A, F, S
A : un ensemble d'abonnés (téléphonie)
S : un ensemble de services
F : un ensemble de forfaits
soient les relations r_s et r_f suivantes :
les abonnés à des services

$$r_s : A \leftrightarrow S$$

les abonnés à des forfaits

$$r_f : A \leftrightarrow F$$

On peut rechercher des abonnés à un service particulier
rechercher des abonnés à un forfait particulier
calculer le nombre d'abonnés au service ss

Modélisation ensembliste (suite)

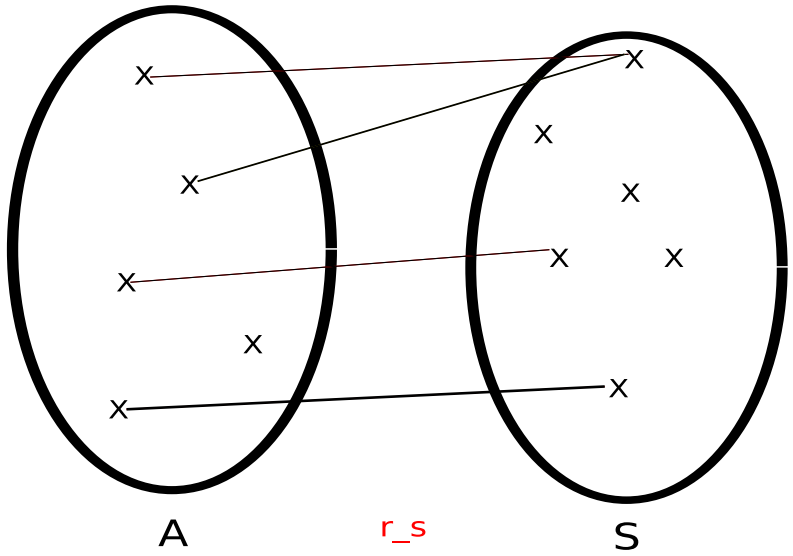


Figure: Exemple de relation (Diagramme de Euler-Venn)

Exercices : Dessiner à l'aide de diagramme de Venn, des fonctions, injections, surjections, bijections

Modélisation ensembliste (suite)

Affichage de l'état des places dans une salle

$PLACE$ / * un ensemble abstrait de places * /
$ETAT = \{occupe, libre\}$
$COULEUR = \{rouge, vert\}$
$etatp : PLACE \rightarrow ETAT$ / * fonction totale * /
$couleurp : ETAT \rightarrow COULEUR$ / * fonction injective * /
...
Affichage(*desPlaces*) :
Pour chaque place p_i *de l'ensemble,*
Afficher(*couleurp*(*etatp*(p_i))

Modélisation ensembliste (suite)

Soit un ensemble d'étudiants. Les étudiants sont inscrits à des modules et passent des épreuves dans ces modules.
Les notes sont utilisées pour calculer des moyennes en vue de la préparation des jury de fin d'année.
On cherche tous les étudiants qui ont moins de 9 en moyenne générale et qui ont moins de 7 dans au moins un module.

$ETUDIANT \quad /* \text{un ensemble abstrait} */$
$MODULE$
$Etudiants \subseteq ETUDIANT \quad /* \text{un sous} - \text{ensemble donne} */$
$Modules \subseteq MODULE$
$moyenneEtud : Etudiants \rightarrow N \quad /* \text{fonction totale} */$
$moyenneModule : Etudiant * Modules \nrightarrow Moyenne$
$\ldots$

Après la modélisation, le raisonnement

Du modèle au programme (mais correct !)

Supposons le programme écrit (à partir de la phase de modélisation...)

Est-ce qu'il est correct ?

Comment le montrer ?

☛ Logique de Hoare (les fondamentaux sur la correction de programme)

Introduction à la Logique

Les propositions

Appelons P une assertion élémentaire.
C'est un énoncé VRAI ou FAUX dans un contexte ou une théorie donnée (par exemple la théorie des ensembles).
Une proposition est une expression du langage dont la grammaire (règle de formation) est la suivante :

$$\begin{array}{lll} prop & ::= & P \\ & | & prop \Rightarrow prop \\ & | & prop \wedge prop \\ & | & \neg prop \end{array}$$

Des parenthèses peuvent être utilisées si nécéssaire, la grammaire est alors modifiée en conséquence.

Sémantique des opérateurs

La sémantique des opérateurs ou connecteurs logiques $\Rightarrow$, $\wedge$, $\neg$ est donnée par des tables de vérité

Exercice : donner les tables de vérité des opérateurs de base

Principe du **tiers exclu** : toute formule est soit VRAI soit FAUX.

Extension du langage à la disjonction et l'équivalence

Le symbole $\equiv$ désigne l'équivalence syntaxique.

Disjonction ($\vee$) :

$$P \vee Q \equiv \neg P \Rightarrow Q$$

Equivalence ($\Leftrightarrow$) :

$$p \Leftrightarrow Q \equiv (P \Rightarrow Q) \wedge (Q \Rightarrow P)$$

On peut compléter la grammaire donnée plus haut avec ces nouveaux opérateurs ; on a ainsi $\Rightarrow, \wedge, \neg, \vee, \equiv$

Les prédicats : introduction

Le calcul des propositions traite de vérité absolue.
Le calcul des prédicats traite de vérité relative.
On fait une extension du calcul propositionnel.
Quelle est la valeur de vérité de l'énoncé suivant (i et j sont des entiers) ?

$$i < j$$

Cet énoncé décrit une propriété de deux entiers i et j.
Il est $vrai$ ou $faux$ selon les valeurs affectées à i et j.

Les prédicats : introduction

i	j	prédicat $i < j$
0	1	Vrai
2	1	Faux
3	3	Faux

i et j sont des variables.

$i < j$ est un prédicat (on parle aussi de *formule* ou de *relation*).

Les prédicats : introduction

Dans les prédicats on utilise deux sortes de variables :

- les variables libres et
- les variables liées.

Exemples :
Que dire de

$$x + 3 > y$$

Formation des prédicats

Les prédicats sont construits avec :

- des constantes (0, 2, . . .),
- des symboles de fonction (f, g, +, *, . . .),
- d'autres prédicats (P, Q, R, . . .)

et avec des symboles communs à tous les langages du premier ordre :

- les connecteurs logiques $\neg$, $\Rightarrow$, $\Leftrightarrow$, $\wedge$, $\vee$ de la logique des propositions,
- les quantificateurs :
  le quantificateur universel noté $\forall$ et le quantificateur existentiel noté $\exists$
- les variables (x, y, z, . . .) qui forment un ensemble dénombrable,
- les parenthèses ouvrantes et fermantes, parfois les crochets [et] et la virgule.

Les quantificateurs

- Le prédicat $\forall x.P$ dit que tout prédicat obtenu en substituant dans P les occurrences libres de x par une expression est un théorème.
- Le prédicat $\exists x.P$ est équivalent à $\neg\forall x.\neg P$.

  Il exprime qu'**il existe au moins une expression F telle que le prédicat obtenu en substituant dans P les occurrences libres de x par F est un théorème**.

Exemples de prédicats

1. $$\neg(x^2 < 0) \vee \forall y.y + 2 = x$$

2. $$\forall n.\exists m \mid m > n$$

   Tout entier n a un supérieur

3. ...

Substitutions en logique

Une substitution est un remplacement des occurrences libres d'une variable dans une expression par une autre expression donnée.
Il y a diverses notations ; on trouve souvent :

soit $P[v\backslash x]$: la substitution de v à toutes les occurrences libres de x dans P

soit $[v\backslash x]P$: la substitution de v à toutes les occurrences libres de x dans P.

Substitutions en logique

Des exemples :

- Soit le prédicat $P_1 : x > 2$
  la substitution $P_1[3\backslash x]$ donne $3 > 2$

- Soit le prédicat $P_2 : \forall x.(x > 2) \vee (x = 0)$
  Que donne la substitution $P_2[3\backslash x]$?

  Attention aux variables libres, liées !
  On ne remplace (substitue) que les variables libres.

Introduction à la théorie des ensembles

Introduction à la théorie des ensembles

Cantor :

- Un ensemble : regroupement d'éléments de même nature ou propriété.
- Exemples : entiers naturels, relatifs, réels R , ...
  mais aussi HOMME, FEMME, ETUDIANT, NUMINSEE

On peut définir les ensembles

- en extension
- en compréhension
- en composant d'autres ensembles (avec des opérateurs spécifiques)

Introduction à la théorie des ensembles

- Eléments dans un ensemble E ;
  opérateur d'appartenance : $x \in E$
- Opérateurs sur les ensembles :
  union $\cup$,
  intersection $\cap$,
  différence $\setminus$,
  produit cartésien $\times$,
  inclusion $\subseteq$, ...

Relations

- Relation = sous-ensemble du produit cartésien
- Exemple
  $r : ETUDIANT \leftrightarrow GROUPE$
  r est un sous-ensemble de $ETUDIANT \times GROUPE$

Abstraction, analyse de besoins

Soit à construire un logiciel de gestion d'une base de documents musicaux de divers types.
Ici, un document musical est l'œuvre d'un auteur ou d'un compositeur ; il a un titre, un genre, une durée, une occupation sur le support de stockage et est enregistré sous un format donné (mp3, mpeg, ...).
Une application logicielle qui fonctionnera sur un lecteur mp3 devra permettre à son utilisateur de trier les documents selon leurs auteurs, leurs genres, leur occupation ou leur durée ; d'écouter un document, d'arrêter une écoute, effacer un document, envoyer à..., etc
Un document ne peut pas être rangé dans deux genres différents.
Tout document doit avoir un titre, un genre et un format.
Un document ne peut être en même temps en écoute, ...

Analyse des besoins

Données identifiés et caractéristiques :

- document
  - auteur
  - titre
  - genre
  - format de stockage
  - durée
  - occupation

Fonctionnalités voulues :

Pour des documents :

- Trier un **ensemble de documents**

Pour un document :

- écouter
- arrêter l'écoute
- envoyer à
- supprimer

☛ Structuration, spécification fonctionnelle

Analyse, abstraction, modélisation

Prenons des ensembles abstraits : DOCUMENT, TITRE, GENRE, FORMAT, AUTEUR, ETATDOC, ...

$$
\begin{aligned}
&ETATDOC = \{enEcoute, ...\}\\
&titre \in documents \rightarrow TITRE \qquad /* \text{ fonction totale } */\\
&genre \in documents \rightarrow GENRE\\
&format \in documents \rightarrow FORMAT\\
&etatDocument \in documents \nrightarrow ETATDOC \qquad /* \text{ fonction partielle } */\\
&duree \in documents \nrightarrow ETATDOC\\
&auteur \in documents \nrightarrow AUTEUR\\
&occupation \in documents \nrightarrow POIDS\\
&\ldots
\end{aligned}
$$

Comment définir les fonctionnalités ?

Abstractions - propriétés - Illustration (suite)

Exemple : Une figure géométrique polygone a plusieurs côtés qui partagent deux à deux des coordonnées identiques (extrémités avec le suivant et le précédent).
Du fait du nombre de côtés, des dimensions et de leurs coordonnées, certaines figures (polygones) sont dites carré, rectangle, triangle.

Propriétés :
triangle : 3 côtés
carré : quatre côtés égaux
rectangle : quatre côtés égaux 2 à 2.

Abstractions - propriétés - Illustration (suite)

Soit $FIGURE$ un ensemble abstrait donné ;

$$carres \subseteq FIGURE$$
$$rectangles \subseteq FIGURE$$
$$triangles \subseteq FIGURE$$
$$figures \subseteq FIGURE$$
$$figures = carres \cup rectangles \cup triangles$$
$$X1 == ...; X2 == ...$$
$$Y1 == ...; Y2 == ...$$
$$dim \in FIGURE \times 1..nbCote \nrightarrow N \quad /* \text{ dimension d'un cote } */$$
$$nbcote \in figures \rightarrow N \quad /* \text{ nombre de cote d'une figure } */$$
$$coord \in 1..nbCote \rightarrow X1 * Y1 * X2 * Y2$$

Abstractions - propriétés - Illustration (suite)

$$
\begin{array}{l}
\ldots \\
x \in triangles \Rightarrow nbcote(x) = 3 \\
x \in carres \Rightarrow nbcote(x) = 4 \\
\qquad \wedge \forall i, j \in 1..nbcote(x)\ . \\
\qquad\qquad dim(x, i) = dim(x, j) \\
x \in rectangles \Rightarrow nbcote(x) = 4 \\
\qquad \wedge \forall i \in 1..nbcote(x)\ .\ \forall\ j,\ k\ \in 1..nbcote(x)\ | \\
\qquad\qquad j = (i + 1)\ mod\ 4 \wedge k = (i + 2)\ mod\ 4. \\
\qquad\qquad dim(x, i) \neq dim(x, j) \wedge dim(x, i) = dim(x, k) \\
\qquad\qquad \ldots
\end{array}
$$

On peut ainsi raisonner sur des figures..., programmer ...

Abstractions - propriétés - Illustration (suite)

Autre exemple : On veut savoir si un segment donné en coupe un autre.

On considère d'abord des modélisations :

Modélisation1

Segment1 : ensemble de points (S_1)
Segment2 : ensemble de points (S_2)

Modélisation2

moins abstraite
Segment1 [AB]: (xA,yA) ; (xB,yB)
Segment2 [CD]: (xC,yC) ; (xD,yD)

Lequel des deux modèles est plus satisfaisant ?

Abstractions - Propriétés - Illustration (suite)

La question initiale savoir si un segment coupe un autre :
Pour la modélisation1 : il suffit de voir

$$S_1 \cap S_2 \neq \{\}$$

Pour la modélisation2 : il faut

- Expliciter les équations des droites qui supportent les segments.
- Déterminer le point de croisement des droites.
- Vérifier que le point de croisement appartient aux deux segments.

Abstractions - Propriétés - Illustration (suite)

1. Equation des deux droites.

   Rappel : équation de la droite support d'un segment : y=ax+b

   Le premier segment qui passe par les points A (xA,yA) et B (xB, yB) est tel que

$$yA = a1.xA + b1 \quad et \quad yB = a1.xB + b1$$

   pour trouver les valeurs de a1 et b1 il suffit de résoudre le système de 2 équations à deux inconnues (a1 et b1):
   yA=a1.xA + b1 et yB= a1.xB + b1

   Pour le deuxième segment, on fait le même travail
   Equation de la droite support du segment 2.
   Elle passe par les points C (xC, yC) et D (xD, yD)
   Il faut résoudre les 2 équations à deux inconnues a2, b2:
   yC =a2.xC + b2 et yD= a2.xD + b2
   pour trouver la valeur de a2 et b2.

Abstractions - Propriétés - Illustration (suite)

2. Si les 2 droites se croisent alors il existe un x tel que y = y2 ;
   c'est à dire que l'équation a1.x + b1 = a2.x + b2 a une solution :
   a1.x - a2.x = b2 - b1
   donc on extrait x=(b2 - b1)/(a1 - a2)
3. En conclusion,
   si x est compris entre xA et xB ou égal à xA ou xB et compris entre xC et xD ou égal à xC ou xD, alors les deux segments se croisent a cette valeur de x.

Quel est le modèle le plus satisfaisant ?

Les 7 ponts de Königsberg

La ville de Königsberg (aujourd'hui Kaliningrad): deux îles ; un pont relie les deux îles ; six ponts relient le continent à l'une ou l'autre des deux îles.

Le problème : déterminer s'il existe (ou non) une promenade dans les rues de Königsberg permettant, à partir d'un point de départ au choix, de passer une et une seule fois par chaque pont, et de revenir au point de départ.

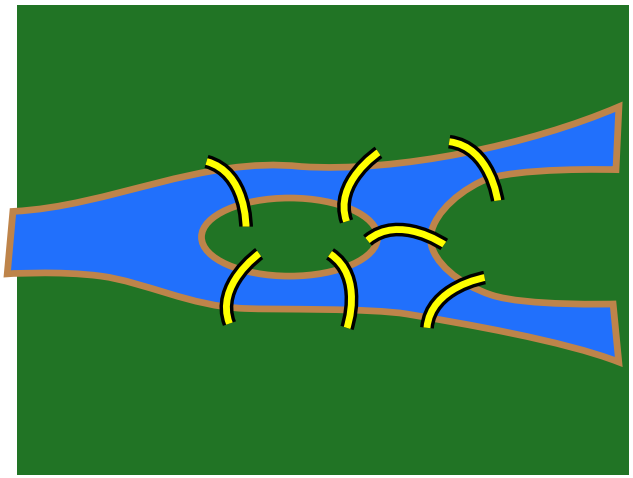


Figure: Ponts de Konisberg

Les ponts de Königsberg

Modélisation des données du problème :

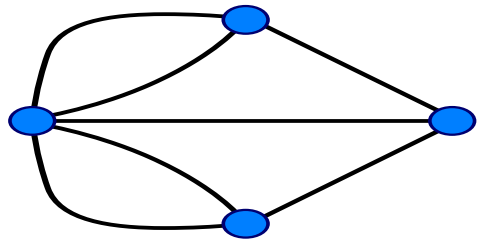


Figure: Graphe des ponts de Konisberg

La promenade demandée n'existe pas ! Solution trouvée par Euler.

Les ponts de Königsberg

EULER a associé un graphe à la ville comme dans la figure (abstraction).
Pour une solution : il faut ordonner les sept arêtes du graphe de façon à ce que deux arêtes consécutives par rapport à notre ordre soient adjacentes dans le graphe (+ première et dernière consecutive).

Il faut pour cela que tout sommet du graphe soit incident à deux (ou un nombre pair) d'arêtes.
Ce qui n'est pas le cas du graphe (il y a des sommets avec 3 arêtes).
DONC, c'est sûr, il n'y a pas de telle promenade ;
Ce n'est pas la peine d'essayer d'écrire un programme qui en cherche !
Mais on peut écrire un programme qui dit s'il y en a ou pas.

Le barbier

Voir Dijskstra

Les fondamentaux sont incontournables

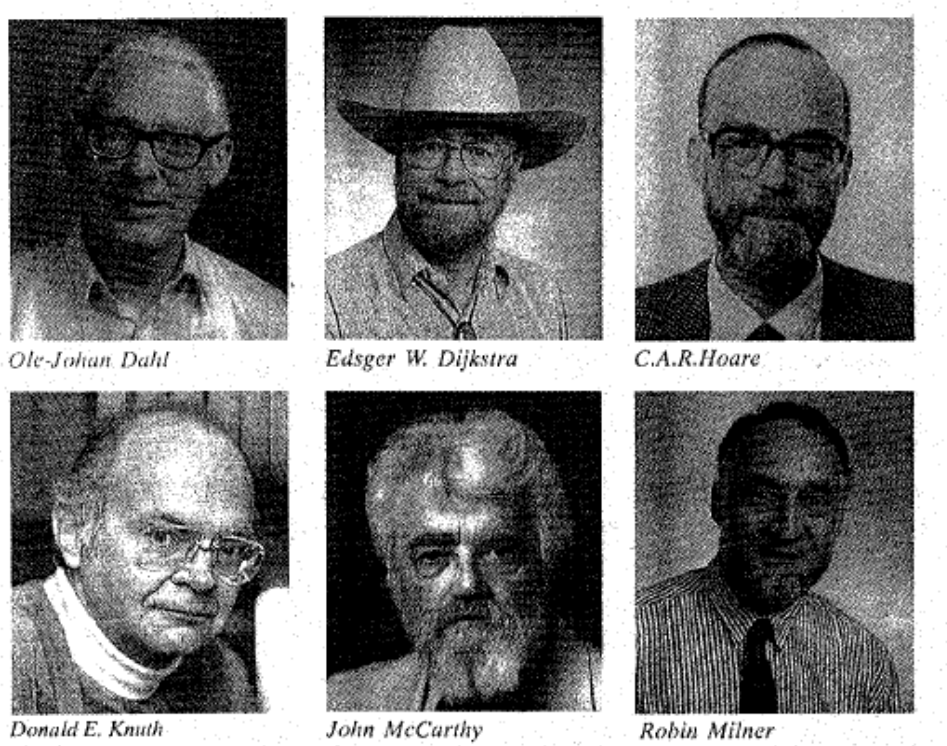


Figure: Des pionniers

Quelques pionniers que vous allez lire, croiser

- A. Turing
- N. Chomsky
- E.F. Codd
- M. Minsky
- K. Gödel
- D. Gries
- R. W. Floyd
- E. W. Dijkstra
- R. Milner
- C. A Hoare
- D. Knuth
- O. Dahl
- J. McCarthy
- D. Parnas
- J. Wing
- ...
- J. Backkus
- P. Naur
- J. Hopcroft
- L. Lamport
- D. Bjorner
- M. Jackson
- J. Sifakis (Français, Prix Turing 2007 !)
- ...