

Cours No 1 : Introduction.

Notes de cours
Christophe Dony

1 Histoire et Définitions

Programmation Applicative :

- un texte de programme est un ensemble de définition de fonctions
- l'exécution d'un programme est une succession, d'**application** de fonctions à des arguments au sens algébrique du terme (calcul utilisant des opérations, des lettres et des nombres) et selon la sémantique opératoire de l'"application" du *Lambda-Calcul*;
- utilisant potentiellement (mais sans obligation) des variables mutables (par opposition à la programmation fonctionnelle),
- où toute instruction est une expression (dont le calcul a une valeur).

Programmation récursive : programmation utilisant des fonctions récursives i.e. se définissant en s'utilisant elles-mêmes (exemple : *factorielle*)

Une fonction récursive est appropriée pour implanter un algorithme pour tout problème auquel peut s'appliquer une résolution par récurrence, ou pour parcourir et traiter des structures de données elles-mêmes récursives telles que les listes et les arbres.

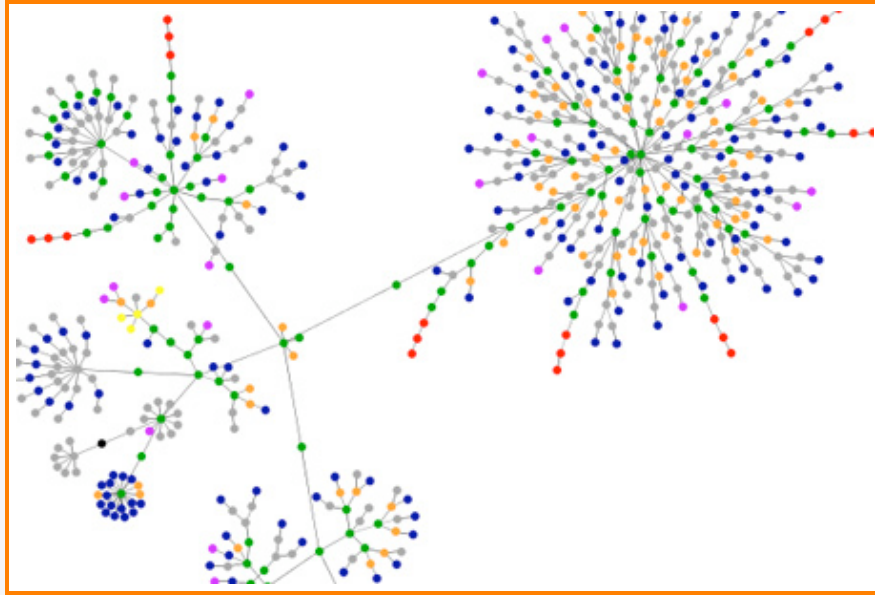


Figure (1) – “Web site tree - une représentation graphique du code HTML des pages WEB” - journal “Libération”, 23 janvier 2008

- **Lisp**, 1960 John Mc Carthy : “*lists processing*”
 - syntaxe simple et non ambiguë,
 - programmation numérique ou symbolique (programmer un joueur d’échec),
 - programmation récursive (arbres, IA, Web),
 - programmes = données (traiter les programmes comme des données)
 - extensible - macros (on peut implanter d’autres langages en Lisp)
- **Scheme**, 1970, Gerald J. Sussman et Guy L. Steele : Héritier de LISP à liaison strictement lexicale : enseignement de la programmation.

“Structure and Interpretation of Computer Programs - Abelson, Fano, Sussman - MIT Press - 1984”.

Caractéristiques de *Scheme* et des langages applicatifs :

- Toute phrase syntaxiquement correcte du langage (y compris une instruction) est une expression algébrique ayant une valeur calculable^{1 2}.
- La mémoire est allouée et récupérée automatiquement
- si on le souhaite, programmation sans **Effets de bords**, dite “fonctionnelle pure” - Voir langage Haskell).
- De nombreuses abstractions simplifient la vie du programmeur. Exemple : les grands nombres, les itérateurs, etc.

```

1 (fact 30)
2 = 265252859812191058636308480000000
3 (map fact '(2 3 4 30))
4 = (2 6 24 265252859812191058636308480000000)

```

1. Même une expression réalisant un effet de bord (voir cours no 9).
 2. Ceci n’est pas vrai dans les langages dits “impératifs”.

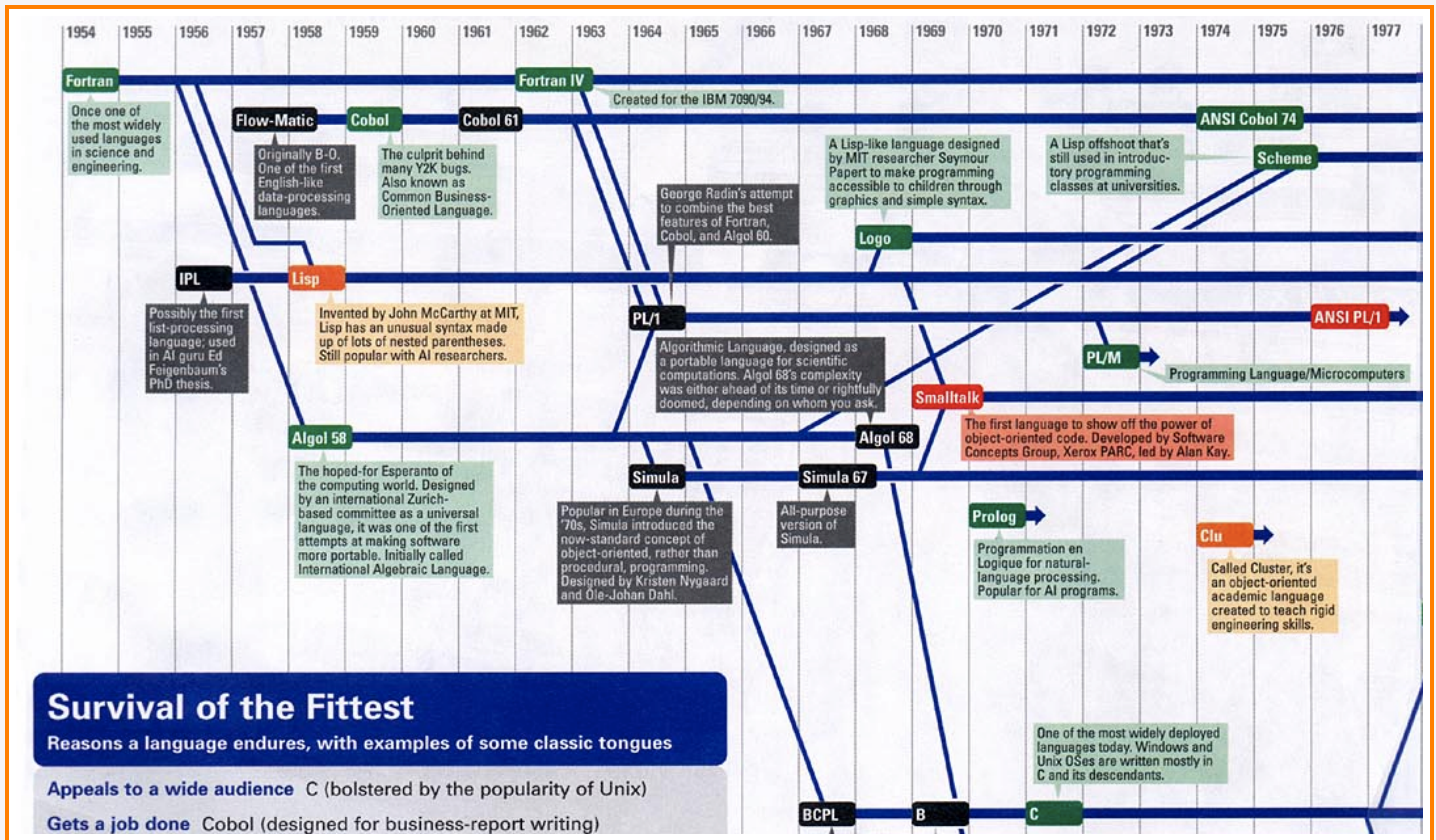


Figure (2) – Les langages de programmation - Vision Historique - extrait1 - <http://www.digibarn.com/collections/posters/tongues/tongues.jpg>

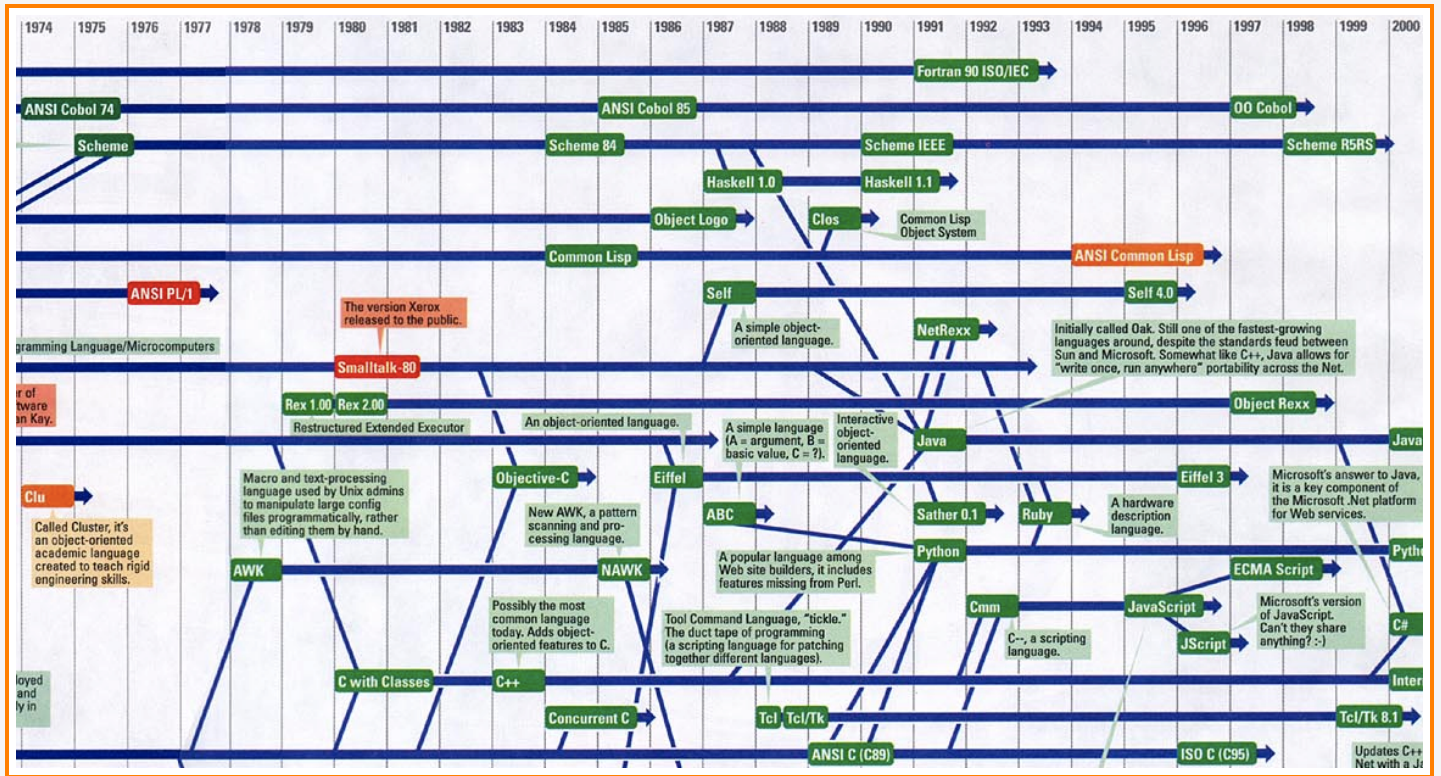


Figure (3) – Les langages de programmation - Vision Historique - extrait 2-
<http://www.digibarn.com/collections/posters/tongues/tongues.jpg>

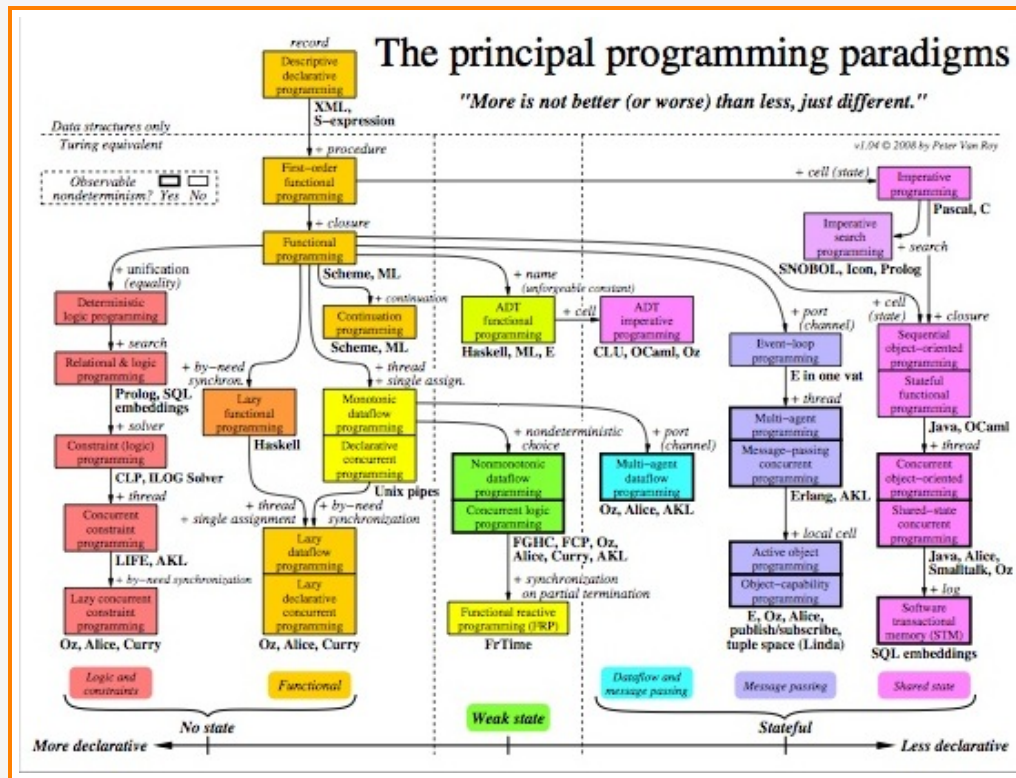


Figure (4) – Vision Historique et Thématique - extrait de : <http://www.ctm.info.ucl.ac.be>,
 “Concepts, Techniques, and Models of Computer Programming”, P. Van Roy et S. Haridi.

Langage Universel. Tous secteurs d’applications ouverts.

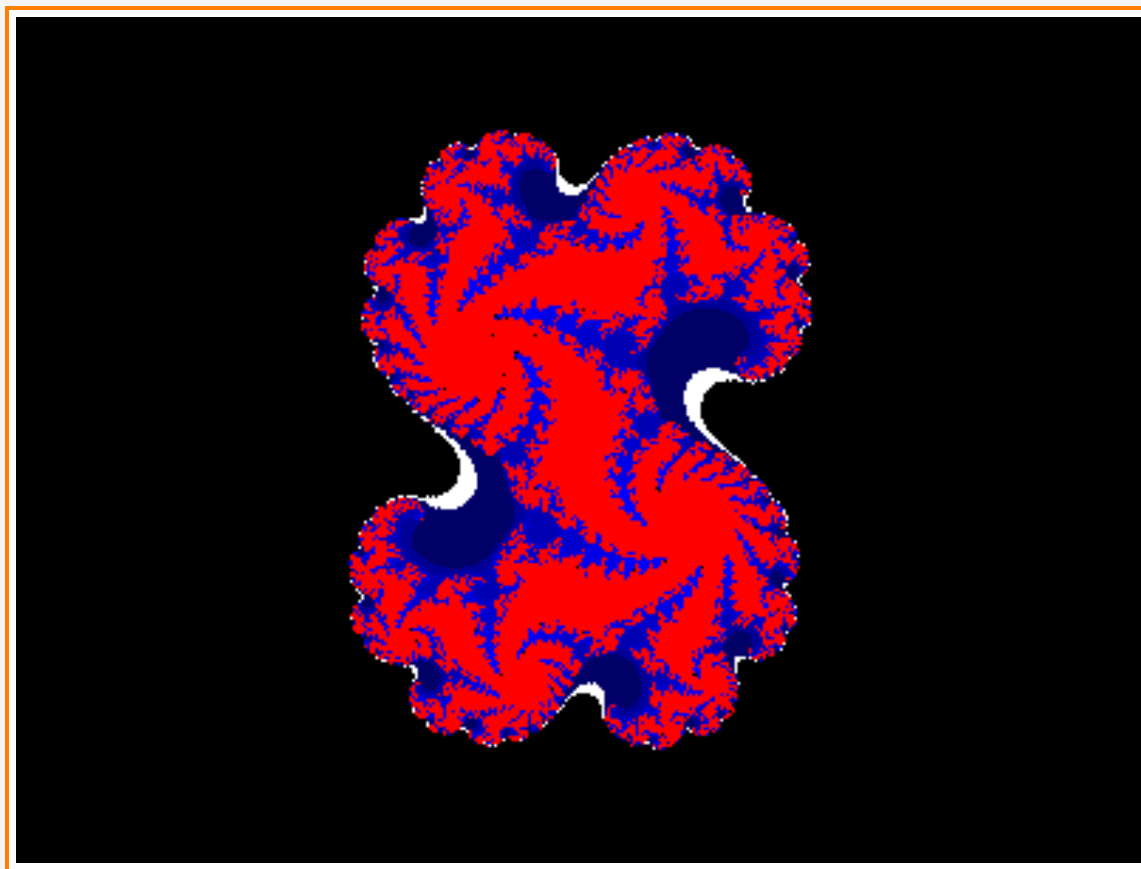


Figure (5) – *Image Fractale dessinée par un programme Scheme - Girault Geoffroy - TER L2 2008*

Contenu du cours :

- Syntaxe.
- Types prédéfinis, Typage, Base de l'interprétation des expressions.
- Identificateurs, Fonctions, Premières Structures de contrôle.
- Blocs, Environnements, Liaisons.
- Fonctions Récursives.
- Types structurés - Listes
- Listes et Symboles : Calcul Symbolique
- Fonctions récursives sur les listes.
- Arbres, Récursions arborescentes.
- Optimisation des fonctions récursives. Récursion terminales, enveloppées.
- Techniques de dérécursivation, transformations, memoization, continuations.
- Abstraction de données. Encapsulation.
- Introduction à l'interprétation des langages, eval comme exemple de fonction récursive.

2 Lectures associées

Livres ou polycopiés conseillés et également utilisés pour construire ce cours :

- “[Structure and Interpretation of Computer Programs](#)” de Abelson, Fano, Sussman.
- “[Passeport pour l’algorithmique et Scheme](#)” de R. Cori et P. Casteran.
- “Programmation récursive (en Scheme), Cours et exercices corrigés”, Anne Brygoo, Titou Durand, Maryse Pelletier, Christian Queinnec, Michèle Soria ; voir page de [C. Queinnec](#).
- “La programmation applicative, de LISP à la machine en passant par le lambda-calcul”, E. Saint-James, 1993, Editions Hermès.