Université Hassan II de Casablanca
Ecole Normale Supérieure (ENS)

Chapitre 5

Langages de programmation

Module 6 : Introduction à l'informatique

1ère ANNEE LICENCE D'EDUCATION D'ENSEIGNEMENT SECONDAIRE : MATHEMATIQUES (LEESM)

Ghizlane MOUKHLISS

Année universitaire 2023/2024

Plan du Chapitre

1. Qu'est-ce qu'un algorithme ?
2. Notions de base sur la programmation
3. Langage machine
4. Langage de programmation
5. Niveaux de langage
   - Niveau d'abstraction
   - Le type de traducteur
   - Paradigmes de programmation
6. Exemples des langages de programmation

Qu'est-ce qu'un algorithme?

Définition d'un algorithme

Un algorithme est une suite d'instructions, d'étapes, d'opérations permettant la résolution d'un problème donné.

Quelques algorithmes

- Une recette de cuisine.
- Un guide d'installation d'un logiciel.
- Le calcul à la main d'une addition, d'une soustraction, d'une multiplication,
- d'une division.
- Algorithme d'Euclide.
- Résolution d'une équation du premier degré $ax + b = 0$

Notions de base sur la programmation

Paradigme

"Modèle théorique de pensée qui oriente la recherche et la réflexion scientifiques" (Larousse).

Programme

Un programme est l'expression (traduction) d'un algorithme dans un langage de programmation.

Programmation

La programmation est un ensemble des activités orientées vers la conception, la réalisation, le test et la maintenance de programmes.

Langage

"Une abstraction des opérations réalisée par un ordinateur. Un langage est habituellement dominé par un paradigme à partir duquel sa structure a été conçue. Souvent, des éléments issus de différents paradigmes y sont présents."

Langage machine

Qu'est-ce que le langage machine ?

- Le langage machine, est un code binaire (une suite de 0 et de 1) permettant de communiquer directement avec le processeur.
- mais :
  - En raison de sa difficulté, ce langage est assez peu accessible (même impossible) au commun des utilisateurs, seul un expert pourrait faire de la programmation.
  - Le langage machine est spécifique d'un microprocesseur donné d'où l'absence de portabilité.

⇒ **Exemple :**

```
0x32 00000000 00000000 10110000 01100001
0x33 00011010 01011010 11101010 11101010
0x34 01011110 01011010 11000111 11111000
0x35 01011010 01011010 10101010 10101010
0x36 11011110 11101010 10101010 10101010
0x37 11101010 11000111 11000111 11111111
```

==> l'instruction 10110000 01100001 écrit la valeur 97 dans le registre AX.

Langage de programmation

Qu'est-ce qu'un langage de programmation ?

- Un langage de programmation est un langage compréhensible par l'humain lambda, permettant, dans une syntaxe stricte, de décrire un algorithme par la donnée d'une succession d'instructions (des ordres).
- Cette succession d'instructions sera ensuite traduite en langage machine par un programme specifique (le compilateur ou l'interpréteur).
- Ainsi, un langage de programmation est à voir comme un langage intermédiaire permettant de communiquer entre le programmeur et le microprocesseur).

Niveau de langage

Comme vous le savez, il existe un grand nombre de langages de programmation. Pourquoi une telle variété ? Y a-t-il réellement des différences fondamentales entre les différents langages ?

La réponse est OUI, de plusieurs points de vue. Nous abordons 3 de ces points à savoir :

1. Le niveau d'abstraction du langage ;
2. Le type de traducteur (interpréteur ou compilateur) pour transformer le code source en binaire ;
3. Le paradigme de programmation (style de programmation).

Le choix du langage

En raison de la variété des langages de programmation, le choix d'un langage de programmation n'est pas facile, chacun à ses spécificités et correspond mieux à certains types d'utilisations. Pour faire un bon choix, nous étudions les 3 points de vue qui font la différence entre les langages de programmation.

Niveau d'abstraction du langage

Le niveau d'abstraction de langage consiste à savoir à quel point le langage de programmation s‘éloigne du langage machine et des contraintes organisationnelles de la mémoire et des périphériques qui lui sont liées pour s'approcher du langage humain.

On distingue deux types d'abstraction :

- **Un langage de bas niveau** : est un langage restant proche des contraintes de la machine ; le but est davantage la recherche de l'efficacité par une gestion pensée et raisonnée des ressources (mémoire, périphérique...), au détriment du confort de programmation. Parmi les langages de bas niveau, on peut citer langage assembleur ou langage C.

- **Un langage de haut niveau** : est un langage s'approchant davantage du langage humain, et dégageant la programmation de toutes les contraintes matérielles qui sont gérées automatiquement (mémoire, périphériques...). L’intérêt principal est un confort de programmation et la possibilité de se concentrer sur l'aspect algorithmique. Généralement, un langage de haut niveau permet une plus grande portabilité (indépendant du matériel). Parmi les langages de haut niveau, on peut citer : C, C++, Java, Python,..

Compilation et Interprétation

Il existe deux techniques pour la traduction d'un programme en langage machine :

- **La compilation** : elle consiste à traduire entièrement un programme écrit dans un langage de haut niveau en langage machine. Le programme chargé de faire cette traduction s'appelle le **compilateur**, et est fourni avec le langage de programmation. Le compilateur prend en argument le code initial, et retourne en sortie un nouveau fichier (le programme compilé), directement exploitable (ou presque) par l'ordinateur. Son avantage est que une fois la compilation effectuée, le traitement par l'ordinateur est plus rapide. Par contre la compilation peut être lente, et nécessite que le programme soit syntaxiquement complet.
- **L'interprétation** : contrairement à la compilation, il ne s'agit pas de la conversion en un autre programme, mais d'une exécution dynamique. L'exécution est effectuée directement à partir du fichier source : l'interpréteur lit les instructions les unes après les autres, et envoie au fur et à mesure sa traduction au processeur.

Remarque

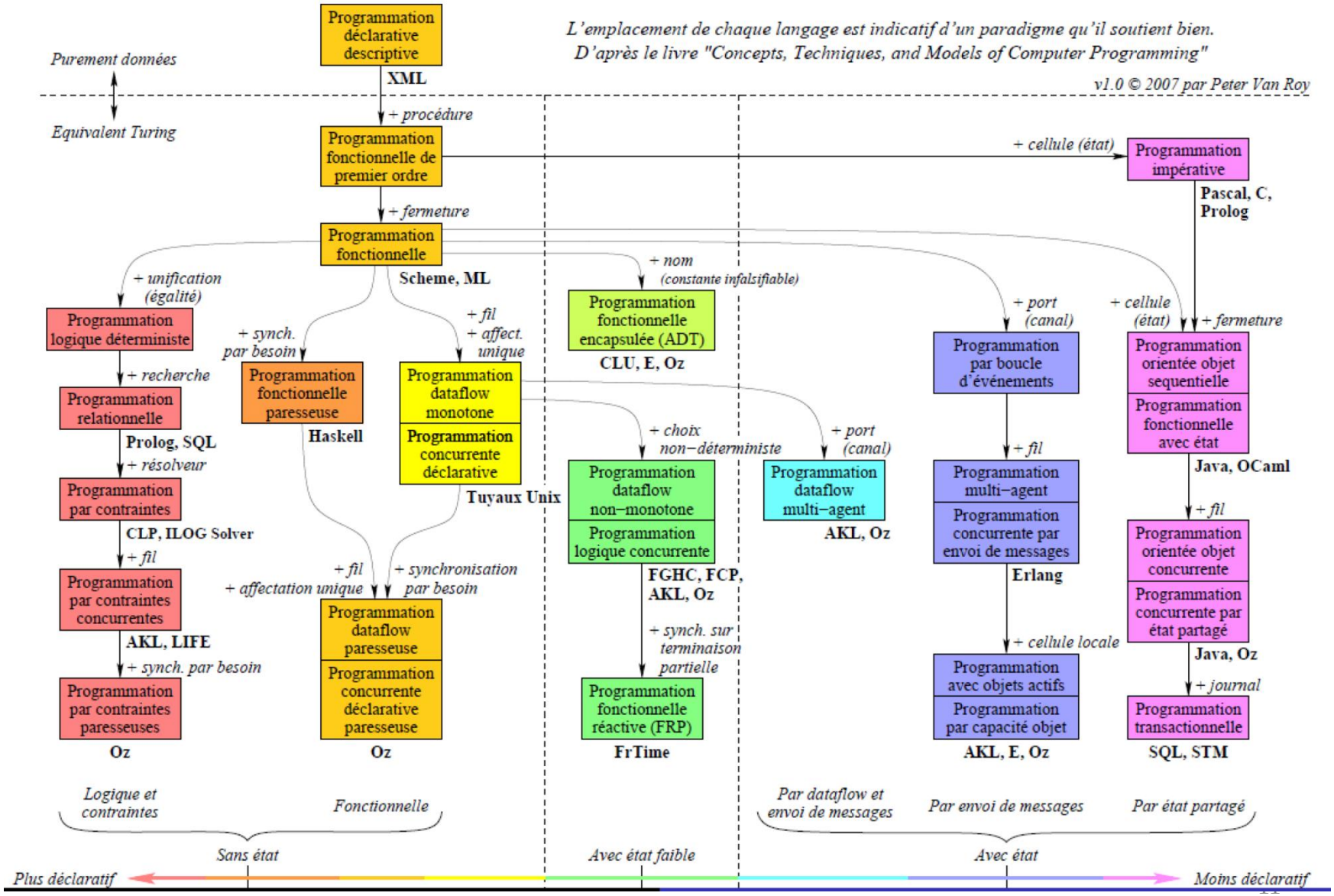
Ces deux techniques peuvent coexister pour un même langage, à titre d'exemple, nous citons **Python**.

Paradigmes de programmation

Un paradigme de programmation est un style de programmation déterminant de quelle manière et sous quelle forme le programmeur manipule des données et donne des instructions. Il s'agit en quelque sorte de la philosophie du langage. Il existe un grand nombre de paradigmes de programmation. Parmi les plus utilisés, citons les suivants :

- **La programmation impérative (structurée)** : Le programme consiste en une succession d'instructions. L'exécution d'une instruction a pour conséquence une modification de l'‘état de l'ordinateur. L'‘état final de l'ordinateur fournit le résultat voulu. Par exemple C et Pascal.
- **La programmation orientée objet (POO)** : On agit sur des objets (des structures de données). Les objets réagissent à des messages extérieurs au moyen de méthodes. Ainsi, le gros de la programmation porte sur la définition de méthodes associées à des classes d'objets. Par exemple C++ et Java.
- **La programmation logique :** C'est un paradigme basé sur les règles de la logique formelle. Il est notamment utilisé pour les démonstrations automatiques, ou pour l'intelligence artificielle. Par exemple Lisp et Prolog.
- **La programmation par scripting/dynamique**
- **La programmation déclarative**
- **La programmation orientée aspect**
- **La Programmation concurrente**
- **et plus. . .**

Les principaux paradigmes de programmation



Programmation impérative ou déclarative

- Deux styles de programmation différents :
  - un programme est déclaratif s'il décrit le **quoi**
  - un programme est impératif s'il décrit le **comment**
- Exemple :
  - déclaratif : le rendez-vous est sur la place **Jemaa El Fna** à Marrakech
  - impératif :

    * en voiture (façon GPS) :
      - montez dans la voiture
      - attachez vos ceinture
      - démarrez le moteur
      - avancez tout droit sur 50 m
      - tournez à droite
      - etc.

    * par le train
      - prenez la ligne 1 du train direction Casa voyageur jusqu‘à Marrakech
      - Marchez 10 minutes de la gare Marrakech par la porte bab Doukala

Exemple de programmation déclarative

- **HTML** (Hyper Text Markup Language)
  - utilisé pour la description de pages Web
  - une page Web écrite en HTML décrit le contenu de la page, mais pas explicitement la manière de calculer l'aspect et la position des différents objets de la page.
  - Exemple

```
<!DOCTYPE html>
< html >
< body >
Bonjour tout le monde
< /body >
< /html >
```

- **SQL** (Structured Query Language)
  - 1979-2003
  - utilisé pour manipuler et interroger des bases de données relationnelles.
  - Exemple:

```
SELECT (name, service) FROM employees WHERE (statut='stagiaire')
GROUP BY (name,service) ORDER BY name ;
```

Les langages assembleurs

- **Principe**
  - créés pour faciliter le travail des programmeurs
  - correspondance 1 à 1 entre les instructions du langage machine et le langage d'assemblage correspondant
    * mots du langage machine représentés par des symboles appelés mnémoniques (faciles à retenir)
    * on peut traduire le code dans les deux sens sans perte d'information.

- **Exemple: processeur x86**
  - en langage machine : 10110000 01100001
  - en langage d'assemblage : mov %al,$0x61
  - en langage naturel : charger la valeur hexadécimale 61 dans le registre `AL'

Programmation structurée

- **Objectifs**
  - rendre les programmes plus lisibles
  - plus faciles à modifier
- **Principes** = techniques et méthodologies de structuration des programmes
  - structuration des données
  - structuration des instructions
- **Structures de contrôle de bas niveau**
  - concaténation
  - sélection
  - répétitions

Exemple de programmation structurée en C

⇒**Programme :**

```
#include < stdio.h >
#include < conio.h >
main()
    {
        int a,b,s ;
        a=4 ;
        b=6 ;
        s=a+b ;
        printf("la somme est %d",s) ;
        getch() ;
    }
```

⇒**Résultat sur l'écran :**

La somme est :10

Programmation procédurale

- **Principe**

  - paradigme de programmation basé sur la notion d'appel de procédures
  - une procédure (appelée aussi routine, méthode, fonction)
  - contient une succession d‘étapes de calcul à exécuter
  - une procédure peut être appelée à tout moment pendant l'exécution du programme

- **Intérêts**

  - possibilité de réutiliser le même code à différents endroits d'un programme sans avoir à le recopier en cas de modification de ce code, évite d'avoir à rechercher tous les endroits du programme ou ce code est utilisé
  - améliorer la lisibilité du programme
  - permettre une programmation modulaire et structurée

Exemples de langages procéduraux

- Ada
- ALGOL
- BASIC
- C
- COBOL
- Fortran
- Maple
- Pascal
- etc

La programmation orientée objet (POO)

- **Principes**
  - utilise les « objets » et leurs interactions
  - basés sur les concepts d'encapsulation, d'héritage, etc.
- **Concepts**
  - **classe** : définit les caractéristiques abstraites d'un objet, c-à-d, ses caractéristiques(attributs ou champs) et ses comportements(ses méthodes).
  - **objet** : une instance particulière d'une classe.
  - **encapsulation** : les caractéristiques internes de l'objet sont masquées et ne sont accessibles qu'au travers de méthodes de manipulation de l'objet.
  - **héritage** : une sous-classe est une version spécialisée d'une classe, qui hérite des attributs et comportements de la classe parent et y ajoute les siens.
  - **polymorphisme** : une même méthode sera exécutée différemment en fonction du type de l'objet qui l'exécute.

Exemple de programmation objet en JAVA

⇒**Programme :**

```
class Point {
    float x, y;
    static int NP = 1;
    static void afficherAbscisse(Point p) {
        System.out.println("P : "+p.x);}
    public static void main(String[] args)
    {
        Point pc = new Point(3, 4);
        afficherAbscisse(pc);
        System.out.println("NP="+NP);
    }
}
```

Exemples de langages objets

- Pyhthon
- JAVA
- C++
- C#
- Delphi
- Simula
- Smalltalk
- OCaml
- etc.