Introduction: Langages de Programmation

Une description complète d'un langage de programmation inclut:

- le modèle de calcul,
- la syntaxe et la sémantique des programmes,
- les considérations pragmatiques qui forment le langage.

Mots-clès :

Modèle de calcul, calcul, programme, langage de programmation, syntaxe, sémantique, pragmatique, liaison, libre, portée, environnement.

Exemple Introductif

- Soit l'expression :

2 ×3,14 ×5

Dont le résultat est la valeur:

31,4

- **Abstraction** : remplacer 3,14 par pi

2 ×pi ×5 où pi = 3,14

Nous dirons que

- **pi** est **lié** à 3,14 et est une constante.
- "Où" représente un **environnement**
- La **portée** des définitions est seulement l'expression.

- **Généralisation** : Si 5 est destiné pour être la valeur d'un rayon, introduire une variable pour le rayon:

2 × pi × rayon où pi = 3,14

- **Abstraire** d'avantage : assigner un nom à l'expression:
  Circonférence = rayon ×2 ×pi où pi = 3,14
  - Circonférence est lié à 2 ×pi × rayon où pi=3.14.
  - rayon est dite **libre** dans le côté droit de l'équation.

- Autre **généralisation** : paramètriser la circonférence avec la variable rayon.

Circonférence (rayon) = rayon ×2 ×pi où pi = 3,14 (1)

- rayon qui apparaît dans le côté droit n'est plus libre, elle est liée au paramètre rayon.
- Circonférence(5) = 31,4,
- le paramètre rayon est lié à la valeur 5

→Circonférence(5) est lié à 31,4.

(1) est un programme pour calculer la circonférence d'un cercle a partir du rayon du cercle

Exemple Introductif (suite)

- La notation mathématique (syntaxe) fournit le langage de programmation,
- L'arithmétique fournit le modèle du calcul pour effectuer le calcul,
- Le passage de la syntaxe au modèle de calcul fournit la signification (sémantique) pour le programme.
- La notation utilisée dans cet exemple est basée sur des considérations très pragmatiques de facilité d'utilisation et de compréhension.

CCL : Cet exemple sert à illustrer plusieurs idées principales dans l'étude des langages de programmation qui sont récapitulées dans la définition suivante.

Définitions

1. Le modèle de calcul est une collection de valeurs et d'exécutions.

2. Le calcul est l'application d’une séquence d’opérations à une valeur pour rapporter une autre valeur.

3. Le programme est une spécification d'un calcul.

4. Le langage de programmation est une notation pour écrire des programmes.

5. La syntaxe d'un langage de programmation se rapporte à la structure ou à la forme des programmes.

6. La sémantique d'un langage de programmation décrit la relation entre un programme et le modèle de calcul.

7. La pragmatique d'un langage de programmation décrit le degré de succès avec lequel le langage de programmation est conforme à ses buts d’une part dans sa fidélité au modèle de calcul et d’autre part dans son utilité pour les programmeurs humains.

Données

Un programme peut être visualisé comme une fonction, les valeurs de données de sortie sont une fonction des valeurs de données d'entrée.

Sortie = Programme(entrée)

- **Deux catégories de valeurs**:
  - Primitives : nombres, booléens, caractères
  - Composées : tableaux, enregistrements, valeurs définies de manières récursives (piles, files, arbres).

- **Opérations** :
  - Habituelles (arithmétiques) pour les valeurs primitives
  - Associées à chaque valeur composée : pour construire les valeurs de ce type et les opérations pour accéder à ce type.

- **Type de données**: Une collection de valeurs qui partagent un ensemble commun d'opérations.
  - Type primitif : implémenté par le matériel, parfois par un logiciel.

Modèles de calcul

- Trois modèles de calcul de base : fonctionnel, logique et impératif.

- En plus de l'ensemble des valeurs et d'opérations, chacun a un ensemble d'opérations pour effectuer le calcul :

  1. Fonctionnel : application de fonctions

  2. Logique : inférence logique,

  3. Impératif : ordre de changement d'états.

. Le Modèle Fonctionnel

se compose d'un ensemble de valeurs, de fonctions, et des opérations d'applications et de composition de fonctions

{}valeurs, fonctions définition de fonctions application de fonctions composition de fonctions	Programme =ensemble de définitions de fonctions calcul = application de fonctions (évaluation d'une expression).

- On y trouve :
  - Langages universels : expriment tout ce qui est calculable
  - Langages simples, rigoureux, systématiques : conforme à la théorie du Lambda-calcul
  - Fonctions évoluées**:** prendre d'autres fonctions comme arguments et des fonctions de retour comme résultats

Exemples:
Calcul de la circonférence;
Autre exemple: calcul de l'écart type utilisant les fonctions évoluées *map* et *fold* :

ET(xs) = sqrt(v) où (xs est une liste)
n = length(xs)
v = fold (plus, map(sqr, xn))/n
– sqr(fold(plus,xn)/n)

Souci de la programmation fonctionnelle: définir des relations fonctionnelles.

Le Modèle Logique

se compose d'un ensemble de valeurs, de définitions de relations et d'inférence logique

{}valeurs relations inférence logique	programme = ensemble de définitions de relations Calcul = preuve constructive (séquence d'inférence a partir des définitions)

Inférences
programme = ensemble d'axiomes –
la formalisation du calcul = preuve constructive d'un but à partir du programme

- **Exemples**:

1. Calcul de la circonférence:

*cercle(R, C) si pi = 3,14 et C = 2 * pi * R*.

La fonction est représentée comme une relation entre R et C.

2. Autre exemple : la mortalité de Socrates et de Penelope

faisons l'hypothèse qu'il n'y a aucun mortels c.-à-d.. ¬mortal(Y)

1a. human(Socrates) Fait
1b. human(Penelope) Fait
2. mortal(X) if human(X) Règle
3. ¬mortal(Y) hypothèse
4.a. X = Y de 2 et de 3 par unification
4b. ¬human(Y) et modus tollens
5a. Y = Socrates de 1 et de 4 par unification
5b. Y = Penelope
Contradiction 5a, 4b, et 1a; 5b, 4b et 1b

Souci de la programmation logique: définir des relations

Le Modèle Impératif

se compose d'un ensemble de valeurs comprenant un état et l'opération d'affectation pour modifier l'état

Etat : ensemble de paire (nom-valeur) de constantes et de variables :

S0 - O0 - > S1 -... - > Sn-1 - On-1 - > Sn

cellules mémoire {}valeurs commandes (séquences d'affectations)	Programme = ordre des commandes Calcul = ordre de changement d'état (séquences d'états)

Exemple calcul de la circonférence

constant pi = 3,14
entrée circonférence (rayon): = 2 * pi * rayon *sortie* (circonférence)

– Le calcul exige l'implémentation pour déterminer la valeur du rayon et de pi

constant pi = 3,14
Rayon = _ | _, circonférence = _ | _, pi=3.14
entrée (Rayon)
Rayon X, circonférence = _ | _, pi=3.14
*Circonférence: = 2 * pi * Rayon*
*Rayon X, circonférence =2 *X*pi, pi=3.14*
Sortie (Circonférence)
*Rayon X, circonférence = 2 *X *pi, pi=3.14*

où _ | _ indique une valeur non définie.

Modèle impératif - Modèle procédural

- Regrouper un ensemble d’opérations dans des procédures

- Modèle de calcul le plus proche du matériel sur lequel les programmes sont exécutés.
  - plus facile à mettre en application

- Souci principal dans la programmation impérative est de définir un ordre de changements d'état.

Calculabilité

- Equivalence des modèles du calcul fonctionnel, logique et impératif : n'importe quel problème qui a une solution dans un modèle est soluble (en principe) dans chacun des autres modèles.

- La plupart des langages de programmation utilisent plus d'un modèle de calcul mais un modèle prédomine habituellement.

- LISP, Scheme, et ML sont basés sur le modèle fonctionnel de calcul mais fournissent quelques constructions impératives

- Miranda fournit presque une mise en oeuvre pure.

- Prolog fournit une mise en place partielle du modèle de calcul logique, mais, pour des raisons d'efficacité et pratique, il contient des constructions impératives.

- Pascal, C/C++, Ada et Java contiennent les affectations, des méthodes pour définir diverses séquences de calcul et fournissent des réalisations minimales du modèle de calcul fonctionnel et de logique.

Syntaxe et Sémantique

- syntaxe décrit la structure des programmes

- sémantique définit le rapport entre la syntaxe et le modèle de calcul.

- simplifier la tâche du raisonnement : la syntaxe d'un langage de programmation doit être étroitement liée au modèle de calcul : principe de clarté.

  Principe de clarté
  La structure d'un langage de programmation devrait être bien définie, et le résultat d'une partie du code devrait être facilement prévisible.

- La notation utilisée dans les modèles fonctionnel et logique reflète la pratique mathématique courante et montre la simplicité

- la notation utilisée pour le modèle impératif doit pouvoir indiquer une variété d'ordre d'états et des expressions
  →irrégulière et complexe

- L'utilisation plus large des langages impératifs
  →la majeure partie du travail effectué dans le domaine de la sémantique de langage de programmation concerne des langages de programmation impératifs.

Pragmatique

- Concerne :
  Facilité, succès, pratique, domaine d'application du langage de programmation

- Applicabilité : assurée par l'abstraction, généralisation, modularité.

1. **Abstraction :**
Associer un nom à un objet et utiliser le nom à chaque fois que l'objet est nécessaire ;
→Supprime le détail et fournit les constructions qui permettent l'extension d'un langage de programmation.

2**.Généralisation** : remplacer une constante par une variable
→permet l'application des constructions à d'autres objets et probablement à d'autres classes d'objets.

3**. Modularité** : division d'un programme en des sections
→compilation séparée et dans des bibliothèques de code réutilisable.

➔*L'abstraction, la généralisation et la modularité* soulagent le programmeur en lui permettant d'introduire des niveaux de détail et la division logique d'un programme.

Pragmatique (suite)

Programmation concurrente :

- implique des notations pour exprimer l'exécution parallèle des parties d'un programme, ainsi que des techniques pour résoudre des problèmes résultants de la synchronisation et de transmission.

- peut être mise en application dans n'importe quel modèle de calcul :
  - **Cas des modèles de calcul fonctionnel et logique** :
    →Attractive, puisque l'évaluation d'une sous expression ou bien l'inférence peuvent être exécutés de manière concurrente,

  - **Cas du modèle de calcul impérative :**
    →Nécessite un rajout d'éléments syntaxique

Pragmatique (suite)

Programmation Objets:

- Implique des notations pour structurer un programme en une collection d'objets effectuant des calculs,

- Chaque objet est lié à une valeur et à un ensemble d'opérations qui déterminent quels sont les messages auxquels il faut répondre,

- Les objets sont organisés de manière hiérarchique et peuvent hériter des opérations des objets de plus haut niveau,

- Peut être mise en application dans n'importe quel modèle de calcul.