

Les techniques de programmation

Leïla Kloul

**Laboratoire PRiSM
Université de Versailles-St-Quentin**

Sommaire

- Introduction
- Classification des langages de programmation
- Classification du calcul
- Principes de l'ingénierie
- Stratégies du développement du logiciel

Introduction

Crise du logiciel

- ▶ **1968** : Reconnaissance publique de l'existence de la crise du logiciel

Naissance du **Génie Logiciel**

- ▶ Bases théoriques
- ▶ Méthodes et outils validés par la pratique

Le logiciel est un objet manufacturé complexe

Introduction

Le **génie logiciel** est l'art de *concevoir*, *spécifier*, *réaliser*,
et *faire évoluer* des programmes de *qualité*,

- ▶ avec des *moyens raisonnables*,
- ▶ dans des *délais raisonnables*

en vue d'utiliser un ordinateur pour résoudre certains problèmes.

Introduction

Qualité du logiciel

- ▶ **Validité** : respect du cahier de charges et des spécifications.
- ▶ **Fiabilité** : fonctionnement en toutes circonstances.
- ▶ **Extensibilité** : modification et extension aisées des fonctions du logiciel.
- ▶ **Efficacité** : utilisation optimale des ressources matérielles.
- ▶ **Portabilité** : fonctionnement sous différents environnements matériels et logiciels.
- ▶ **Vérifiabilité** : facilité de préparation des procédures de tests.
- ▶ **Intégrité** : protection du code et des données contre des accès non autorisés.

Introduction

- ▶ **1969-73** : Premiers principes de programmation structurée

- ▶ **1974** : Le DoD américain finance le développement d'un nouveau langage de programmation : **Ada**

Conçu pour un *domaine d'application spécifique* : les systèmes embarqués

- ▶ Ordinateurs de guidage de missiles,
- ▶ contrôleurs de processus,
- ▶ réseau de communication d'entreprise,
- ▶ micro-processeur pour contrôler un moteur de voiture, ...

Introduction

ADA (1974-1980)

- ▶ **Lisibilité**
- ▶ **Typage sévère** : beaucoup d'erreurs détectées à la compilation.
- ▶ **Programmation macroscopique** : mécanismes d'abstraction (données et contrôle), compilation séparée, gestion de bibliothèques.
- ▶ **Portabilité** et entretien aisé des programmes.
- ▶ Traitement des **exceptions**.
- ▶ Notion de **tâches** : série d'activités parallèles.
- ▶ **Généricité** : modèle unique (code) instancié avec des types différents.

Classification des langages de programmation

1. Les langages d'assemblage

- ▶ Correspondance univoque entre une instruction machine et une instruction du langage d'assemblage
- ▶ Transcription des actions à réaliser en séquences d'instructions pour la machine
- ▶ **Programmation difficile**, ne permet pas d'éviter les erreurs au niveau du codage
- ▶ **Programme lié à une famille d'ordinateurs** : celle pour lequel il est développé

Classification des langages de programmation

2. Les langages de réalisation de systèmes

- ▶ Développés pour résoudre les problèmes des langages d'assemblage
- ▶ Structures de contrôle évoluées et typage des variables
- ▶ Accès direct aux opérations et à l'espace d'adressage de la machine
- ▶ Disponibles sur un très grand nombre de machines

Exemple : C

Classification des langages de programmation

3. Les langages statiques de haut niveau

- ▶ Structures de contrôle de haut niveau, déclaration de variables
- ▶ **Allocation statique** de la mémoire
- ▶ Espace nécessaire aux variables calculé par le compilateur et réservé en début d'exécution du programme
- ▶ Cela implique certaines contraintes pour le programmeur

Exemples : Cobol, Fortran

Classification des langages de programmation

4. Les langages de haut niveau à structure de bloc

- ▶ Reprennent certaines caractéristiques des langages statiques,
- ▶ **Allocation dynamique** de la mémoire **limitée**, appelée structure de bloc
- ▶ Le compilateur ne peut pas décider à l'avance de l'espace mémoire nécessaire aux variables
- ▶ Allocation d'espace mémoire à l'entrée d'un bloc et sa restitution à la sortie

Exemples : Algol, Pascal, Ada et Simula

Classification des langages de programmation

5. Les langages dynamiques de haut niveau

- ▶ **Allocation dynamique** de l'espace mémoire
- ▶ Définis pour les problèmes d'intelligence artificielle
- ▶ Peuvent être très différents les uns des autres
- ▶ Souvent utilisés pour le prototypage

Exemples : Lisp, Prolog

Classification du calcul

- ▶ **La programmation séquentielle et impérative**
le calcul est exprimé par une succession d'états, mode de raisonnement de Von Neumann
- ▶ **La programmation fonctionnelle**
le calcul est exprimé par une fonction
- ▶ **La programmation logique**
le calcul est exprimé par un ensemble d'assertions et de formules
- ▶ **La programmation orientée objet**
le calcul est exprimé par l'envoi de messages à des objets

Principes d'ingénierie

- ▶ **Abstraction et dissimulation d'informations**
 - ▶ gérer la complexité des applications
 - ▶ supprimer la façon dont un objet ou une opération est implémenté
- ▶ **Modularité et localisation**
 - ▶ créer des modules à faible couplage et à forte cohésion
 - ▶ modules fonctionnels (orientation procédurale) ou déclaratifs (OO)
 - ▶ faciliter un but grâce à la structure d'un objet
- ▶ **Uniformité, intégralité et validation**
 - ▶ notation cohérente, suppression des différences inutiles
 - ▶ garantit que tous les éléments importants sont présents
 - ▶ le système peut être aisément testé

Stratégies du développement du logiciel

- ▶ **Conception fonctionnelle descendante**
 - ▶ Du plus général au plus détaillé
 - ▶ Conception structurée ou par raffinements successifs
- ▶ **Conception orientée objet**
 - ▶ Collection d'objets communiquant entre eux par messages.
 - ▶ A chaque objet est associé un ensemble d'opérations.
- ▶ **Conception dirigée par les données**
 - ▶ La structure d'un logiciel doit refléter la structure des données traitées
 - ▶ Analyse directement dérivée des données en entrée et en sortie