Chapitre 8
Programmation orienté objet.

1- Les Caractéristique d'un bon logiciel

✓ ***Exactitude (précision / qualité):***

Aptitude du logiciel à fournir les résultats voulus dans des conditions normales de l'utilisation.

✓ ***La robustesse :***

Aptitude à bien réagir si on s'éloigne des conditions normales D'utilisation.

✓ ***L'extensibilité :***

- **Le logiciel doit être modifiable pour satisfaire l'évolutions des spécifications.**

✓ ***La réutilisabilité :***

Possibilité d'utiliser des parties du logiciel pour résoudre d'autres problèmes.

✓ ***L'efficacité :***

Le code du logiciel doit être suffisamment rapide.

✓ ***La portabilité :***

Le logiciel doit être utilisé sur d'autres processeur et/ou système d'exploitation

2- Programmation structurée

✓**En programmation structurée, un programme est constitué par différentes procédures et différentes structures de données, généralement indépendantes de ces procédures.**

✓**La programmation structurée a fait progresser la qualité de la production des logiciels.**

✓**Notamment, elle a permis de structurer les programmes, et par suite améliorer l'exactitude et la robustesse. On avait espéré qu'elle permettrait également d'en améliorer l'extensibilité et La réutilisabilité.**

✓**Or, en pratique, on s'est aperçu que l'adaptation ou la réutlisabilté d'un logiciel conduisait souvent à casser le module intéressant.**

✓***Equation de Wirth :***

Programme= Algorithme + Structures de données

3- Programmation orientée objet

✓ **La programmation orientée objet (POO) est un moyen de conceptualiser un programme informatique sous forme d'un ensemble d'objets interagissant.**

✓ **La P.O.O est basée sur les paradigmes suivants :**

Encapsulation → protection
Héritage → Réutilisation
Polymorphisme → générique.

L'objet :

C'est une association des données et des procédures (qu'on appelle des méthodes) agissant sur ces données. Par analogie avec l'équation de Wirth, on pourrait dire que l'équation de la P.O.O est :

Méthodes + Données=Objet

Classe:

✓C'est une généralisation de la notion de type que l'on rencontre dans les langages classiques.

✓En effet, une classe n'est rien d'autre que la description d'un ensemble d'objets ayant une structure de données commune et en disposant des mêmes méthodes.

✓Les objets apparaissent alors comme des variables d'un tel type classe, on dit aussi qu'un objet est une instance de sa classe.

Encapsulation:

✓Cela signifie qu'il n'est pas possible d'agir directement sur les données d'un objet ; il est nécessaire de passer par l'intermédiaire de ses méthodes, qui jouent aussi le rôle d'interface obligatoire.

✓On traduit parfois cela en disant que l'appel d'une méthode est en fait l'envoi d'un message à l'objet.

✓**Le grand mérite de l'encapsulation est que, vu de l'extérieur, un objet se caractérise uniquement par les spécifications de ses méthodes, la manière dont sont réellement implantées les données étant sans importance.**

✓**On dit aussi que cette situation réalise une *abstraction des données.***

✓**L'encapsulation facilite la maintenance des logiciels car toute modification de la structure des données n'a pas d'incidence que sur l'objet lui-même, les utilisateurs de l'objet ne seront pas concernés par cette modification.**

Remarque :

En C++ l'encapsulation n'est pas obligatoire.

Héritage:

✓**Permet de définir une nouvelle classe à partir d'une classe existante, à laquelle on ajoute de nouvelles données et de nouvelles méthodes.**

✓**L'héritage concerne à la réutilisation du logiciel (n'écrivez jamais deux fois le même code), il fournit un moyen efficace pour étendre toutes les fonctionnalités d'une classe à une autre.**

✓**En conséquence, nous n'avons pas à répéter les caractéristiques communes aux deux classes.**

Polymorphisme:

✓**En P.O.O, une classe dérivée peut redéfinir certaines des méthodes héritées de sa classe de base.**

✓**Cette possibilité est la clé de ce qu'on appelle le polymorphisme, c'est-à-dire la possibilité d'ajouter des traitements spécifiques à une méthode en continuant d'utiliser les traitements déjà définis.**

✓**Le polymorphisme ne modifie pas les squelettes des méthodes mais le traitement de ces méthodes.**

✓**Le polymorphisme améliore l'extensibilité des programmes en permettant d'ajouter de nouveaux traitements.**

Remarques :

- ✓ **Certains langages peuvent être conçus pour appliquer à la lettres les principes de la P.O.O et réaliser ce que l'on nomme la P.O.O pure (par exemple Simula, Java ….).**

- ✓ **Le langage C++ à été obtenu en ajoutant au langage procédural C les outils permettant de mettre en œuvre tous les principes de la P.O.O.**