Fiche méthodologique de projet

P.1

Définir l'analyse et la conception orientées objet (A/COO)

Objectif

L'analyse et la conception sont résumées par la formule : construire le bon système (l'analyse) et bien construire le système (la conception).

Préambule

« Posséder un marteau ne fait pas de vous un architecte »

Le fait de connaître un langage orienté objet (C++ ou Java par exemple) est nécessaire mais n'est pas suffisant pour créer des systèmes objets.

Le type de notation comme UML est utile mais d'autres aspects existent qui s'avèrent plus importants à assimiler, notamment la façon de penser en termes d'objets. UML n'est qu'une notation graphique standard.

L'analyse « Quoi faire ? »

L'analyse met l'accent sur une **investigation du problème et des besoins** plutôt que sur la recherche d'une solution. C'est une abstraction précise et concise du but de l'application.

On distinguera :
- l'analyse des besoins : l'investigation des besoins
- l'analyse orientée objet : l'investigation des objets du domaine

L'analyse orientée objet est davantage tournée vers la recherche et la description des objets (ou concepts) du domaine du problème.

La conception « Comment faire ? »

La conception sous-entend l'**élaboration d'une solution** répondant aux besoins plutôt que la mise en oeuvre de cette solution.

La conception orientée objet fait apparaître des objets « informatique ».

Quand la conception sera implémentée, c'est cette implémentation (le code par exemple) qui exprimera l'analyse complète et réalisée.

L'implémentation « Le faire ! »

Les classes d'objets et leurs relations de la phase de conception sont traduites en une implémentation dans un langage de programmation (, une base de données, un matériel spécifique, ...). On peut aussi parler de fabrication ou de construction concrète du système.

Les décisions difficiles devront avoir été prises pendant la phase de conception.

Approche « Orientée Objet »

Le terme « orienté objet » signifie que l'on organise le logiciel comme une **collection d'objets dissociés**.

Ces objets comprennent à la fois : une **structure de données** (attributs) et un **comportement** (opérations).

Les quatre aspects à retenir qui caractérisent une approche orientée objet sont :

- L'identité : les **objets**

- La classification : les objets qui ont les mêmes structure de données (attributs) et le même comportement (opérations) sont regroupées en une **classe**. Une classe est une abstraction qui décrit des propriétés pertinentes dans le contexte d'une application et ignore les autres.

- Le polymorphisme : la même opération peut se comporter différemment suivant les classes.

- L'héritage : le partage des attributs et des opérations entre classes. La possibilité de factoriser des propriétés communes à plusieurs classes est l'un des principaux avantages d'un système orienté objet.

Développement « Orienté Objet »

C'est penser le logiciel en s'appuyant sur les abstractions existant dans le monde réel : on identifie et organise des concepts des domaines de l'application en mettant en évidence les propriétés importantes.

<u>Quelques règles essentielles :</u>

- Se concentrer sur les aspects essentiels sur ce qu'est un objet et sur ce qu'il fait avant de décider comment il doit être implémenté.

- Spécifier ce qu'est un objet plutôt que sur la façon dont il est utilisé.

- L'utilisation d'un objet est souvent modifiée pendant le développement.

- Les autres concepts (fonctions, relations, évènements) seront organisés autour des objets.

Réflexion

« Il semble que l'on n'ait jamais assez de temps pour faire correctement le travail dès la première fois, mais que l'on trouve toujours le temps de le refaire »
« Mesurez deux fois, ne coupez qu'une fois » (maxime de charpentier)