TP 2 Génie Logiciel : Design Patterns Gestion de l'Emploi du Temps

Cheref Walid
Spécialité : RSD

2. Principes non respectés

Mon code enfreint deux principes du modèle **SOLID** :

2.1 DIP – Dependency Inversion Principle

Le *GestionnaireEmploiDuTemps* dépend directement de classes concrètes telles que *Etudiant* ou *Responsable*. Toutes ces classes n'implémentent pas systématiquement l'interface *Observer*, ce qui crée une dépendance aux implémentations plutôt qu'à une abstraction.

Solution : garantir que l'ensemble des observateurs implémentent bien l'interface *Observer*, afin que le gestionnaire ne dépende que d'une abstraction.

2.2 OCP – Open/Closed Principle

L'ajout d'un nouvel observateur (par exemple un enseignant ou une application mobile) nécessite de modifier le code du gestionnaire. Celui-ci n'est donc pas réellement ouvert à l'extension sans modification de son code.

Solution : utiliser exclusivement l'interface *Observer* afin de rendre le gestionnaire extensible sans modification interne.

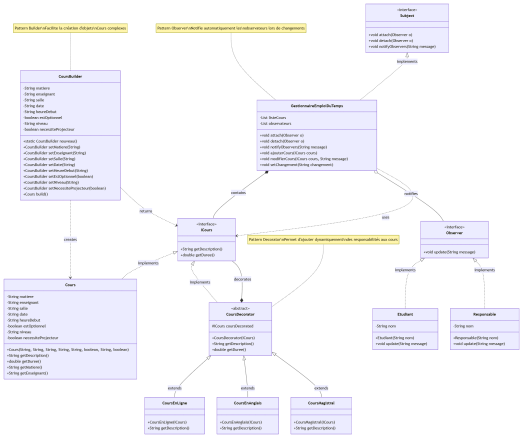

FIGURE 1 – Enter Caption

Conclusion

Le code viole les principes **DIP** et **OCP** : les observateurs ne reposent pas tous sur une abstraction commune, et le gestionnaire n'est pas suffisamment extensible sans modifications, ce qui contrevient aux bonnes pratiques du modèle SOLID.