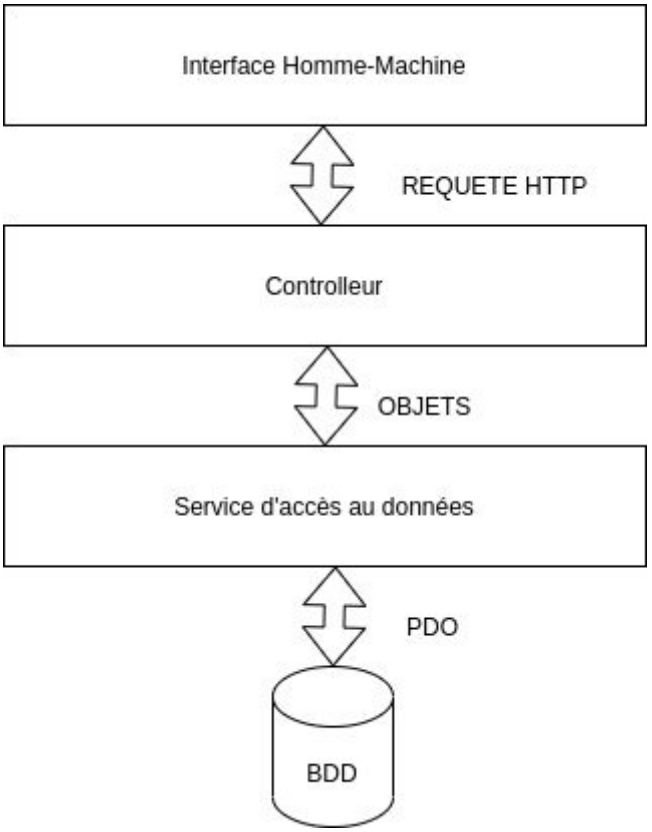
Les modèles et services en PHP

Cet exercice a pour but de vous familiariser avec les services et modèles d'une architecture MVC. Il convient de définir les termes avant d'aller plus loin.

L'architecture orientée services est une forme d'architecture de médiation qui est un modèle d'interaction applicative qui met en œuvre des services (composants logiciels).

Un service est un objet qui va agir comme une couche de notre architecture :



Le modèle est simplement une classe représentative de notre table dans la base de données. Par exemple le modèle Employe sera la classe représentative de notre table `employee`. L'instance de notre modèle, l'objet, servira à stocker les données dans notre programme et permettra de les afficher et de les manipuler.

Durant cet exercice nous allons nous intéresser au service d'accès aux données et son interaction avec la base de données.

Travaux Pratique

1) Ecrire les classes modèles des tables Employee et Department de notre base de données `employee`

Voici un exemple :

```
class Employee {
    private $employee_id;

    private $first_name;

    private $last_name;

    [...]

    public function getEmployeeId()
    {
         return $this->employee_id;
    }

    public function setEmployeeId($employee_id)
    {
         $this->employee_id = employee_id;
    }

    [...]
}
```

2) Créer une classe de service (EmployeeService) pour le modèle Employee et y ajouter les fonctions insert($employee), update($employee) et delete($employee). Chaque fonction prendra en paramètre l'objet à insérer/mettre à jour/supprimer.

3) Coder les fonctions insert, update et delete pour qu'elles effectuent les mise à jour en base de données à l'aide de PDO.

4) Créer une fonction getEmployeeById($employee_id) qui renverra une instance de la classe Employee avec les infos chargée depuis la base de données pour l'identifiant donné.

5) Créer une fonction getAllEmployees() qui renverra un tableau d'Employee contenant tous les employés de la BDD.

6) Répéter les questions 2) à 5) pour la classe Department.

7) Ajouter une propriété $department à notre classe Employee ainsi que les getter et setter associé. Cette propriété servira à stocker une instance de la classe Department.

8) Modifier les fonctions de notre service EmployeeService pour qu'elles mettent à jour les données relatives au Department en fonction des données contenu dans l'objet de la propriété $department.
   Ex : Au lieu d'utiliser la propriété $department_id dans nos requêtes, nous allons désormais utiliser $department->getDepartmentId()

9) Créer un script qui récupère l'employee avec l'identifiant 113, qui récupère le département avec l'identifiant 3 et changer le département de l'employee 113 avec le département 3. Puis sauvegarder l'employé.
   Il faut bien entendu utiliser l'architecture que nous avons développé dans les questions précédentes.
   Vérifier que le département s'est bien mis à jour en base de données.

10)

Lexique

MVC : Modèle - Vue - Controlleur

- Un modèle (Model) contient les données à afficher.
- Une vue (View) contient la présentation de l'interface graphique.
- Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

MODÈLE = OBJET