Enoncé question 3-2

Critiquez cette proposition d'architecture

Par exemple

- *Le modèle pourrait être la calculette constituée pour ses calculs internes d'une pile,*
- *Pourquoi les "listeners" des boutons sont-ils locaux au contrôleur ?*
- *Ce choix de découpage MVC vous parait-il réaliste ?*
- *...*

Proposer votre architecture MVC , un schéma de type diagramme UML, les interfaces java et votre proposition en quelques lignes sur votre rapport suffiront.

Réponse sur question 3-2

Le concept MVC est consisté de 3 types d'objets indépendants pour augmenter la flexibilité :

1- Model : qui est l'objet de l'application ; il est modifié par le contrôleur suite au choix de l'utilisateur et ses changements sont exposés par la Vue. « Extends Observable » et « update View –NotifyObserver()»
2- View : C'est l'affichage sur l'écran des changements dans le model. « Implements Observer. »
3- Controller : peut être une interface machine ou un conteneur comme Jpanel, avec ses boutons et sa propre logique qui définit le comportement du modèle « Implements ActionListener » et permet à l'utilisateur de choisir le « View ».

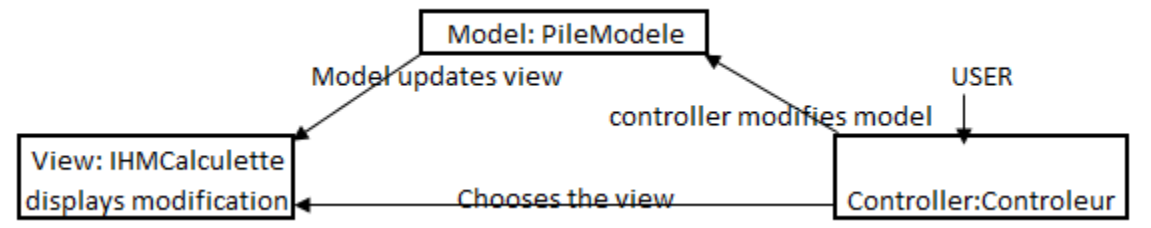
Ce Patron nous donne l'opportunité d'attacher des Vues multiples à un modèle pour lui donner de différentes présentations et de créer de nouvelles Vues pour un même modèle sans le réécrire.

En comparant ce concept à l'architecture de la question3 du TP4 :

1- Model = PileModèle qui vérifie les conditions du Model citées en haut.
2- View : Vue qui affiche les changements du Model ce qui vérifie les conditions de View.
3- Controller = Controleur qui aussi vérifie les conditions du Controller citées en haut.

A l'apparition de la IHMcalculette qui contient la liste réelle que l'utilisateur modifie et qui est un objet du modèle, elle aurait pu être elle-même comme View si on réunit son code à celui de Vue. Ainsi, on y ajoute l'état de la Pile et « addObserver() ».

Donc, je pourrai représenter mon opinion comme suit :



Question 3-3

Vue2