Feuille de TP 3 – Conception et Programmation de Composants MAY

L'objectif de cette troisième feuille de TP est de mettre en pratique les compétences acquises en conception et programmation de composants MAY pour modéliser un patron de conception architecturale (MVC).

Exercice 1 – Implémentation du patron MVC en MAY

En vous aidant du diagramme C&C figurant dans l'article Wikipédia ci-dessous, définissez des interfaces (dans le paquetage « interfaces ») **IRead**, **IWrite**, **IUpdate** et **IEvent** comportant chacune une méthode et permettant de typer les interactions entres les composants **Model**, **Controller** et **View**.

Définissez l'architecture du patron MVC dans un fichier « mvc.speadl » et proposez une implémentation des composants Model, View, Controller, et de l'assemblage global **Root** (dans le paquetage « implem »).

Le composant View comportera un état (par exemple un entier **int**) et fournira un port typé par l'interface **IUser**, comportant une méthode « void sendValue(int val) » (permettant d'envoyer une valeur au contrôleur, qui la transmettra ensuite au modèle et signalera le changement d'état à la vue) et « void printValue() » (permettant d'afficher l'état de la vue).
https://fr.wikipedia.org/wiki/Modèle-vue-contrôleur

Définissez un petit programme de test dans une fonction main qui affiche l'état initial de la vue, envoie une nouvelle valeur et affiche le nouvel état obtenu.

Exercice 2 – Implémentation générique

En vous aidant du lien suivant vers la documentation de MAY, pouvez-vous généraliser votre implémentation de sorte que chaque interface mentionnée dans le fichier .speadl et dans les implémentations de composants soit générique ?
https://wwwsecu.irit.fr/redmine/projects/may/wiki/SpeADL_Minus_Reference#Type-Parameters

Indication : l'instanciation de l'assemblage global devrait ressembler au code suivant :

```
Root.Component<IRead, IWrite, IUpdate, IEvent, IUser>
      component = new RootImpl().newComponent();
```