Examen, 1h45, documents autorisés

NOM : **Prénom :** **Signature :**

Noter le nom de vos voisins :

Gauche :		Devant :	
Droite :		Derrière :	

Barème indicatif : Exercice 1 : 4 points, Exercice 2 : 2 + 2 + 3 + 3 + 3 + 3

Exercice 1 Répondre de manière concise et précise aux questions suivantes.

1.1 Indiquer quels sont les diagrammes principaux proposés par UML ainsi que leurs objectifs.

1.2 Qu'appelle-t-on des transformations M2M et M2T dans l'IDM ?

1.3 Rappeler en quelques lignes le principe général d'ATL et indiquer l'intérêt de ce langage par rapport à un langage généraliste tel que Java par exemple.

Exercice 2 : IHM

Cet exercice définit un langage dédié (DSL) pour décrire des interfaces homme machine (IHM). Le métamodèle de la figure 1 présente les concepts et relations entre concepts de ce DSL.

Une IHM est composée d'un conteneur (Container) qui contient des composants (Component). Ces composants sont soit élémentaires (TextArea, Label, Button, TextField) ; soit des conteneurs. À chaque conteneur est associé un gestionnaire de placement (Layout).

Les classes Component et Layout sont abstraites.

2.1 *Le métamodèle IHM.* Indiquer le(s) patron(s) de conception utilisé(s) dans le DSL IHM parmi ceux présentés dans les transparents de cours. Préciser les éléments du métamodèle (concepts et relations) concernés par le(s) patron(s) utilisé(s).

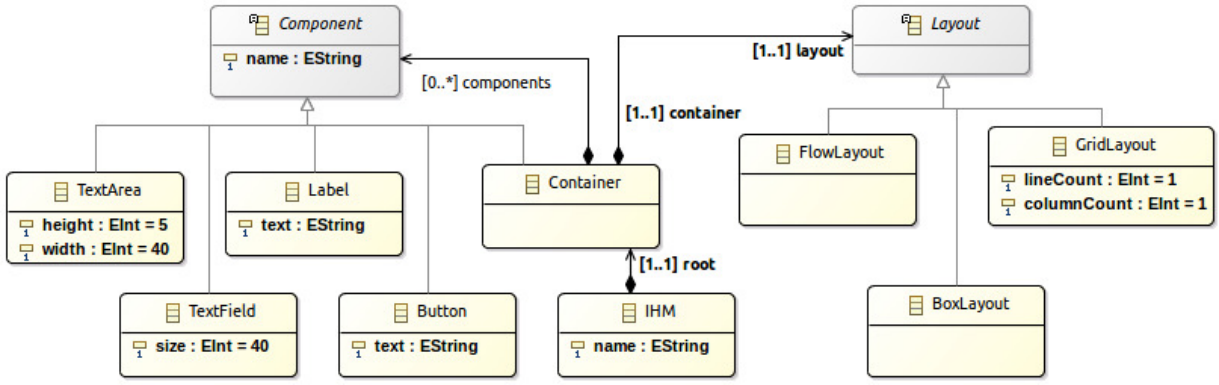

FIGURE 1 – Le métamodèle du DSL IHM

2.2 *Modèle d'IHM.* Soit un modèle d'interface qui comporte deux vues : la première présente l'organisation hiérarchique des composants (figure 2) et la seconde donne une disposition géographique de ces composants ainsi que l'affichage que l'on pourrait obtenir en Swing (figure 3). Ce modèle d'IHM, appelé *example1*, a un composant racine (*root*) équipé d'un gestionnaire de placement *BoxLayout*. Il contient un premier élément qui est un conteneur *p1* équipé d'un gestionnaire *FlowLayout* et contenant une zone de saisie *text* et un bouton *go* ; et un deuxième élément, une zone de saisie *messages*.

2.2.1 Indiquer pourquoi il s'agit d'un modèle conforme au métamodèle IHM en explicitant les conventions de la notation graphique de la figure 2 vis à vis du métamodèle IHM.

2.2.2 Quel(s) outil(s) étudié(s) lors des TPs pourrai(en)t être utilisé(s) pour produire la première vue (organisation hiérarchique des composants) sur un modèle IHM ?

2.3 *Contraintes OCL.* Formaliser en utilisant OCL les contraintes suivantes :

1. Un GridLayout doit avoir des nombres de lignes et de colonnes strictement positifs.
2. Les composants d'un conteneur doivent avoir des noms différents.
3. Un conteneur équipé d'un GridLayout ne peut pas avoir plus de composants que de cases dans la grille.

2.4 *Génération de code.* Soit la transformation acceleo du listing 1.

2.4.1 Donner le résultat que produit cette transformation lorsqu'elle est appliquée sur le modèle des figures 2 et 3.

2.4.2 Est-ce que cette génération de code fonctionnera dans tous les cas ? Dans la négative expliquer les cas dans lesquels elle échouera et indiquer des solutions à apporter (sans les mettre en œuvre).

2.5 *Syntaxe concrète.* Nous souhaitons proposer une syntaxe concrète textuelle pour représenter un modèle IHM. Le listing 2 donne un exemple de cette syntaxe sur l'exemple des figures 2 et 3.

2.5.1 En annotant le sujet, indiquer ce que représentent les éléments de la syntaxe concrète en utilisant les concepts et les relations du métamodèle IHM.

2.5.2 Donner le fichier Xtext qui permet d'engendrer un éditeur textuel pour un modèle IHM respectant la syntaxe précédente.

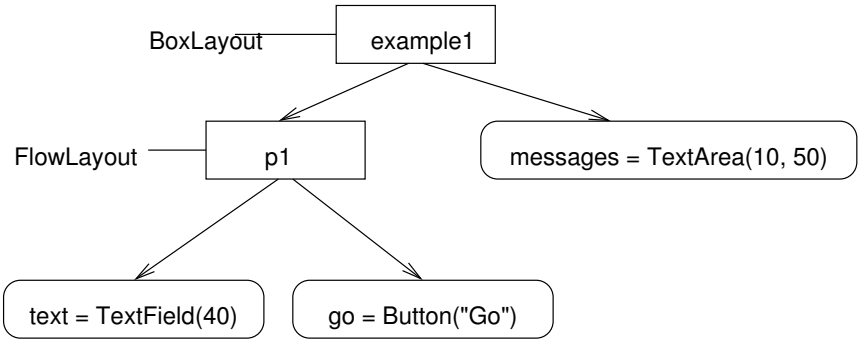

FIGURE 2 – Vue hiérarchique de l'IHM example1

Listing 1 – Transformation acceleo pour IHM

```
[comment encoding = UTF-8 /]
[module toSwing('http://www.example.org/IHM')]

[template public toSwing(anIHM : IHM)]
[comment @main/]
[file (anIHM.name.toUpperFirst() + '.java', false, 'UTF-8')]
import javax.swing.*;
import java.awt.*;

public class [anIHM.name.toUpperFirst() /] extends JFrame {
[for (c : Component | anIHM.root.getAllComponents())]
   private [c.getSwingName()/] [c.name/] = new [c.creation()/];
[/for]
   public [anIHM.name.toUpperFirst() /]() {
[for (c : Container | anIHM.root.getAllContainers())]
      [c.name/].setLayout(new [c.layout.creation()/]);
   [for (elt: Component | c.components) ]
      [c.name/].add([elt.name /]);
   [/for]
[/for]
      this.getContentPane().add([anIHM.root.name/]);
      this.pack();
      this.setVisible(true);
   }
   public static void main(String['[]'/] args) {
      EventQueue.invokeLater(new Runnable() {
         public void run() {
            new [anIHM.name.toUpperFirst()/]();
         }
      });
   }
}
[/file]
[/template]

[query public getSwingName(c: OclAny): String = if c.oclIsKindOf(Layout) then '' else 'J' endif
   + if c.oclIsTypeOf(Container) then 'Panel' else c.eClass().name endif /]
[query public getAllComponents(container : Container) : Collection(Component) =
   let subContainers : Collection(Container) = container.components->selectByKind(Container) in
      if subContainers->isEmpty() then
         container.components->including(container)
      else
         container.components->including(container)->addAll(subContainers->any(true).getAllComponents())
      endif
/]
[query public getAllContainers(container : Container) : Collection(Container) =
   container.getAllComponents()->selectByKind(Container)
/]

[template public creation(o: OclAny)]
[if (o.oclIsTypeOf(BoxLayout)) ]
BoxLayout([o.oclAsType(BoxLayout).container.name/], BoxLayout.Y_AXIS)[elseif (o.oclIsTypeOf(TextField))]
JTextField([o.oclAsType(TextField).size/])[elseif (o.oclIsTypeOf(TextArea))]
JTextArea([o.oclAsType(TextArea).height/], [o.oclAsType(TextArea).width/])[elseif (o.oclIsTypeOf(Label))]
JLabel([o.oclAsType(Label).text/])['']
/][elseif (o.oclIsTypeOf(Button))]
JButton("[o.oclAsType(Button).text/]")[else]
[ o.getSwingName() /]()[/if]
[/template]
```

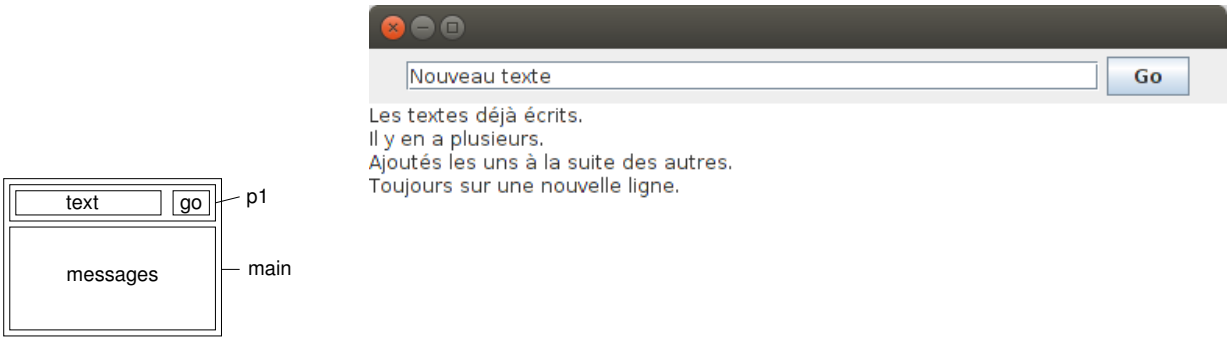

FIGURE 3 – Vues de l'IHM example1 sous forme de boîtes imbriquées et en Swing

Listing 2 – Syntaxe concrète textuelle pour IHM

```
1 ihm example1 root main = layout Box {
2       p1 = layout Flow {
3             text = TextField 40;
4             go = Button "Go";
5       }
6       messages = TextArea 10 50;
7 }
```

2.6 *Visiteur.* Expliquer les modifications à apporter au métamodèle IHM pour l'équiper d'un visiteur (patron de conception Visiteur).

Écrire le visiteur qui donne le nombre de composants élémentaires utilisés pour une IHM.