Examen, 1h45, documents autorisés

NOM : **Prénom :** **Signature :**

Notez le nom de vos voisins :

Gauche :		Devant :	
Droite :		Derrière :	

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

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

1.1 Rappeler ce qu'est l'introspection (ou la réflexivité) et indiquer un exemple pertinent de son utilisation.

1.2 Expliquer comment représenter en Ecore les associations qui sont présentes en UML.

1.3 Expliquer l'objectif de l'outil Sirius.

Exercice 2 : Porte de garage motorisée

On considère une porte de garage motorisée à enroulement. L'utilisateur dispose d'une télécommande comportant un bouton unique permettant d'actionner cette porte. Une pression sur le bouton a pour effet :

- d'ouvrir la porte si celle-ci est fermée,
- de la fermer si elle est ouverte,
- de demander son ouverture si elle est en cours de fermeture,
- de demander sa fermeture si elle est en cours d'ouverture.

La porte dispose d'un capteur de butée qui indique que la porte a atteint sa butée haute (porte ouverte) ou basse (porte fermée). Lors de la mise en service, la porte est fermée.

2.1 Indiquer les états possibles d'une porte de garage.

2.2 Indiquer les événements possibles.

2.3 Dessiner le diagramme UML de machine à états qui décrit le comportement d'une porte de garage.

Exercice 3 : Langage flot de donnée (DataFlow)

On considère un diagramme de bloc de type *flot de données*. Son métamodèle est donné figure 1.

3.1 Expliquer en quoi le modèle de la figure 2 est conforme au métamodèle dataflow (figure 1). En particulier, on explicitera la correspondance des notations utilisées au niveau du modèle avec les éléments du métamodèle.

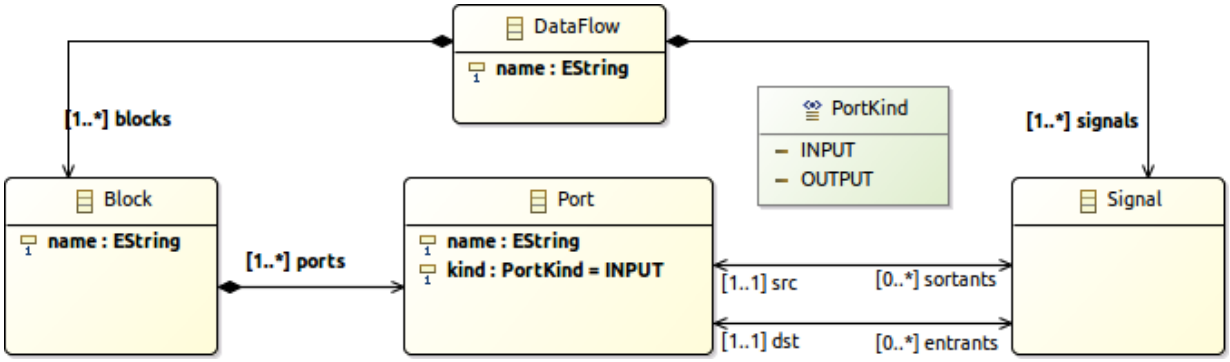

FIGURE 1 – Le métamodèle d'un diagramme flot de données (dataflow)

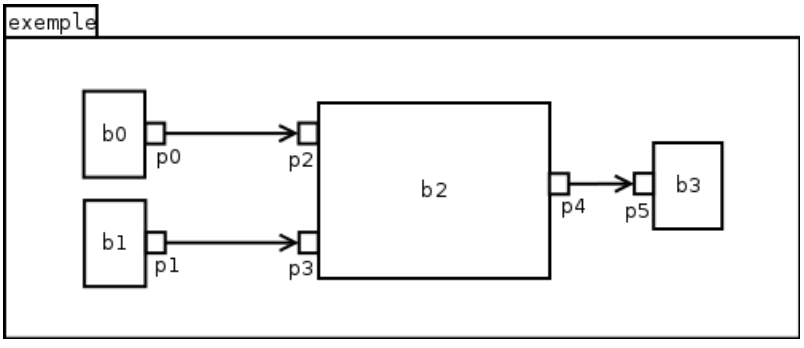

FIGURE 2 – Un modèle de diagramme flot de données (dataflow)

3.2 Exprimer en OCL les contraintes suivantes :

1. Les noms des blocs doivent être uniques.
2. Les noms des ports doivent être uniques au niveau d'un modèle (et pas seulement du bloc qui les contient).
3. Un signal a pour source un port de sortie et pour destination un port d'entrée
4. Il y a au plus un signal qui connecte deux ports donnés.

Donner l'expression OCL qui permet de connaître le nombre de port de type INPUT sur un modèle.

3.3 Donner le fichier Xtext qui permet d'engendrer un éditeur textuel pour saisir un modèle flot de données respectant la syntaxe du listing 1. On veut que le metamodèle engendré soit le même que celui de la figure 1.

Listing 1 – Représentation textuelle du modèle de la figure 2

```
diagram exemple {
       block b1 () returns (p0)
       block b2 () returns (p1)
       block b2 (p2,p3) returns (p4)
       from p0 to p2
       from p1 to p3
       block b3 (p5) returns ()
       from p4 to p5
}
```

3.4 On souhaite transformer un modèle flot de données en réseau de Petri. Le principe de cette traduction est le suivant :

— Pour chaque élément Block, on crée une transition qui porte le même nom.
— Pour chaque élément Signal, on crée une place qui symbolise le fait que le signal est en transit entre le port source et le port destination et deux arcs pour relier cette place aux transitions associées aux blocs des ports source et destination de ce signal.

3.4.1 Donner le réseau de Petri qui correspond au modèle flot de données de la figure 2.

3.4.2 Donner les règles ATL qui transforment un modèle flot de données en un réseau de Petri (le métamodèle des réseaux de Petri est donné figure 3).

3.5 Donner le résultat de la transformation M2T Acceleo du listing 2 quand elle est appliquée sur le modèle flot de données de la figure 2.

3.6 On souhaite pouvoir gérer des modèles flots de données hiérarchiques : un bloc peut être décomposé en blocs et en signaux. Notons que dans ce cas, que les ports du bloc englobant doivent être connectés aux ports de même type (input ou output) des blocs qui le composent.

3.6.1 Proposer une évolution du métamodèle pour prendre en compte cette extension.

3.6.2 Indiquer les évolutions à prévoir sur les contraintes OCL du métamodèle (on ne donnera pas le code OCL correspondant).

3.6.3 Indiquer les évolutions à prévoir sur le fichier Xtext (on ne donnera pas le nouveau fichier correspondant).

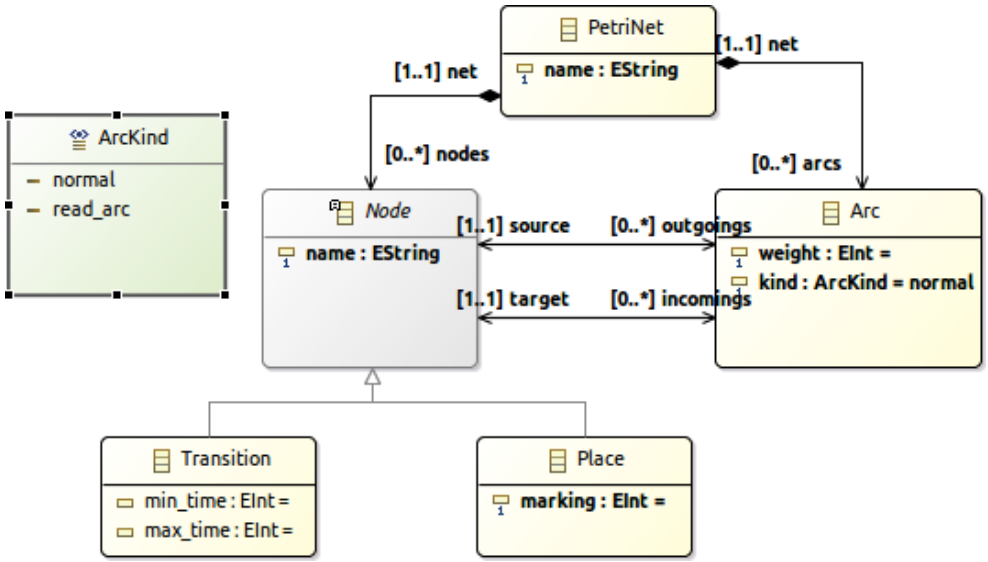

FIGURE 3 – Le métamodèle des réseaux de Petri

Listing 2 – Transformation M2T Acceleo

```
1  [comment encoding = UTF-8 /]
2  [module generate('http://www.example.org/dataflow')]
3
4  [template public generateLaTeX(aDataFlow : DataFlow)]
5  [comment @main/]
6  [file (aDataFlow.name + '.txt', false, 'UTF-8')]
7  DataFlow [aDataFlow.name /] :
8        Blocks:[aDataFlow.blocks.toText() /]
9        Signals:
10 [for (s: Signal | aDataFlow.signals) ]
11             [s.toText() /]
12 [/for]
13 [/file]
14 [/template]
15
16 [query public toText(b: Block): String =
17             '\n\t\t' + b.name + ' /' + b.ports.toText()
18 /]
19
20 [query private toText(p: Port): String =
21       ' ' +
22       if p.kind = PortKind::INPUT then
23             'in'
24       else
25             'out'
26       endif
27       + ' ' + p.name + ' /'
28 /]
29
30 [template private toText(s: Signal) ]
31 [s.src.name /] --> [s.dst.name /];
32 [/template]
```