

TD3: Modélisation (Compléments)

On considère une version très simplifiée des machines à états, dont le méta-modèle est donné dans la Figure 1.

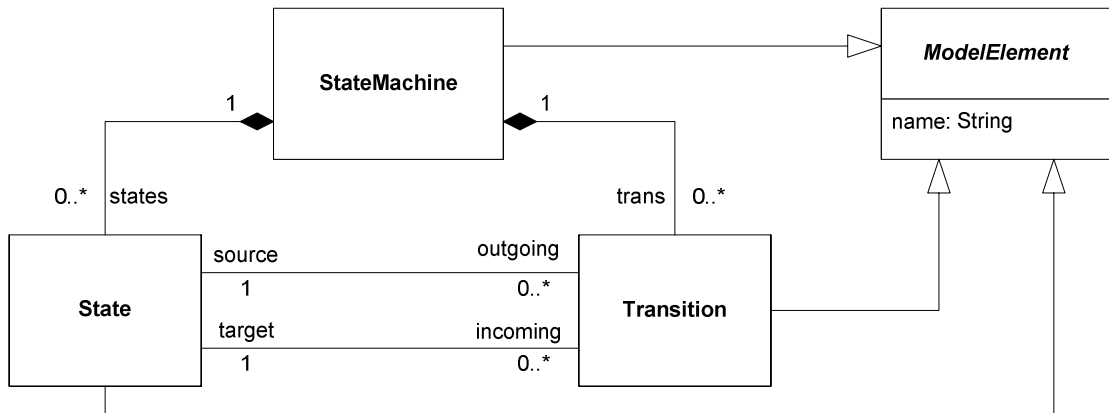


Figure 1 : StateMachine – Méta-modèle.

Les modèles (utilisateur) sont exprimés avec la syntaxe concrète spécifiée dans la Figure 2.

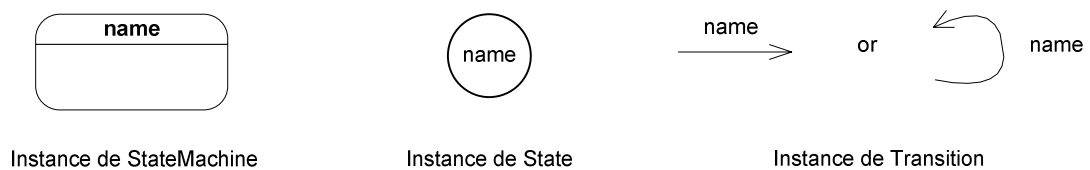


Figure 2 : Représentation des instances.

1 Modèles conformes

Pour chacun des 6 modèles de la Figure 3, précisez dans la Table 1, s'il est conforme au méta-modèle de la Figure 1. En cas de conformité, donnez la représentation en utilisant la syntaxe abstraite (On pourra utiliser les abréviations SM pour **StateMachine**).

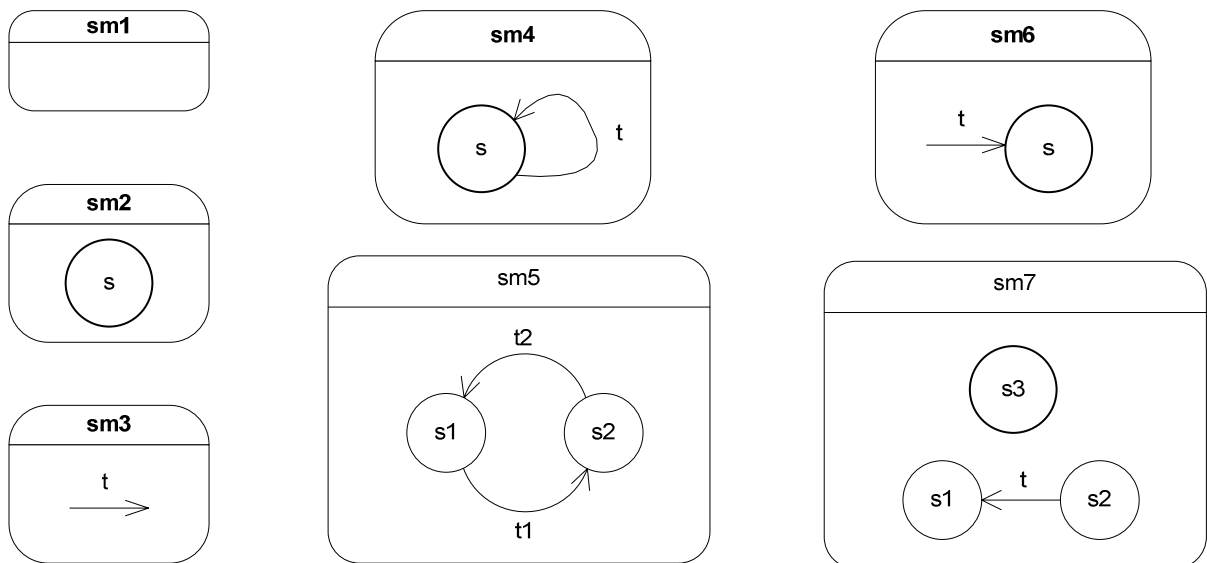


Figure 3 : Instances de StateMachines

StateMachine	sm1	sm2	sm3	sm4	sm5	sm6	sm7
Conformité							

Table 1 : Conformité

2 OCL

2.1 Règles simples

Boucle élémentaire

Une boucle élémentaire est une transition telle que la transition **t** dans la machine **sm4**.
Écrivez une contrainte qui interdit les boucles élémentaires.

Successeurs directs d'un état

Définissez l'opération **State::directSuccessors():Set(State)** qui retourne l'ensemble des successeurs directs d'un état. Un état **s2** est successeur direct de **s1** si et seulement si il existe une transition **t** ayant pour source **s1** et pour cible **s2**.

On rajoute le concept de chemin : **Path** (Figure 4).

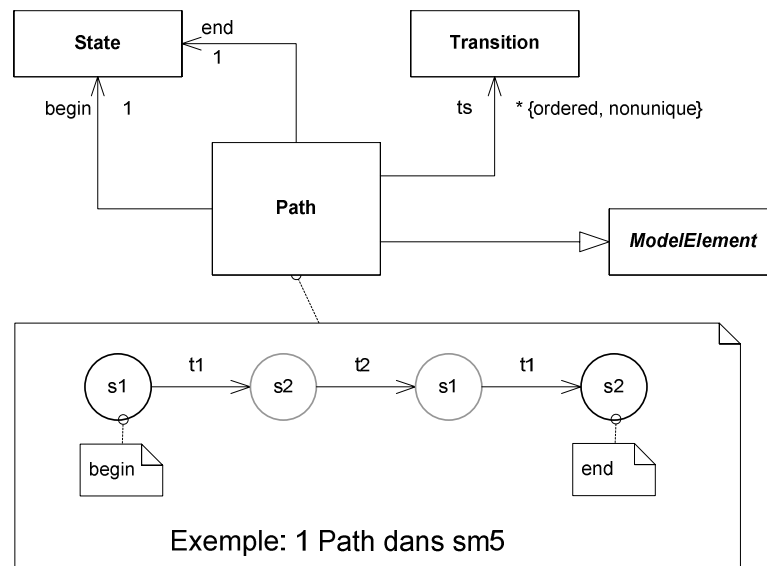


Figure 4 : Enrichissement du métamodèle

2.2 Path : contraintes

Q 2.2.a Que signifie la contrainte **{ordered, nonunique}** sur la propriété **Path::ts** ?

Exprimez les contraintes suivantes :

Q 2.2.b La séquence des transitions du chemin commence par une transition partant de l'état **begin**.

Q 2.2.c La séquence des transitions du chemin termine par une transition arrivant à l'état **end**.

Q 2.2.d Expliquez la contrainte suivante. Quelle propriété exprime-t-elle ?

```
context Path
  Sequence{1..ts->size()-1} ->
    forAll ( i | ts->at(i).target = ts->at(i+1).source )
```

2.3 Path : opérations

Q 2.3.a Que fait l'opération définie par

```
Path::visited( ): Set(State) ;
  visited = begin->union(ts.target) -> asSet( )
```

A quoi sert le **asSet()** ?

Q 2.3.b Spécifiez l'opération **Path::isCircuitFree() : Boolean** qui retourne vrai si et seulement si le chemin est sans circuit.

3 Modélisation avec Papyrus