Université de Nice-Sophia Antipolis

Examen de **EPU-SI-STREAM 07**

Date : Février 2007

Note :

Nom :

Prénom :

Documents personnels autorisés.
Durée : 2h

1.1	1.2.1	1.2.2	1.2.3	2.1	2.2	23	2.4	2.5

1 Modélisation

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

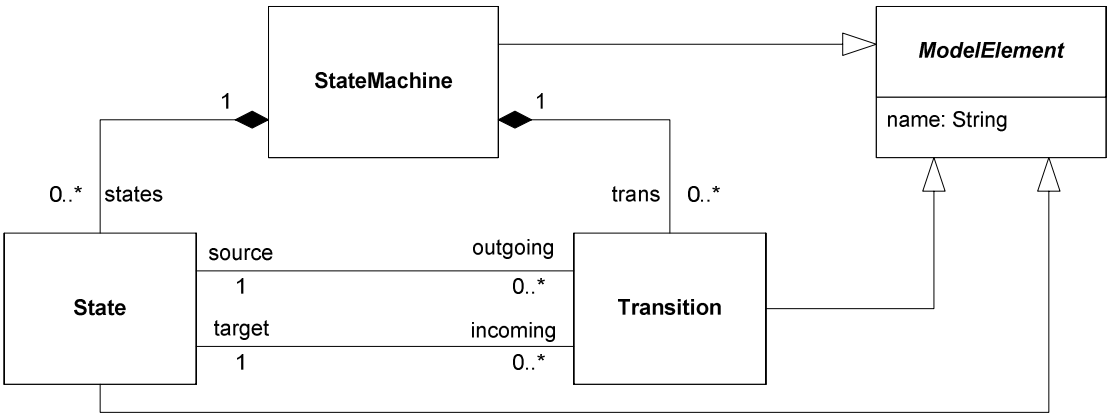

Figure 1 : StateMachine – Méta-modèle.

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

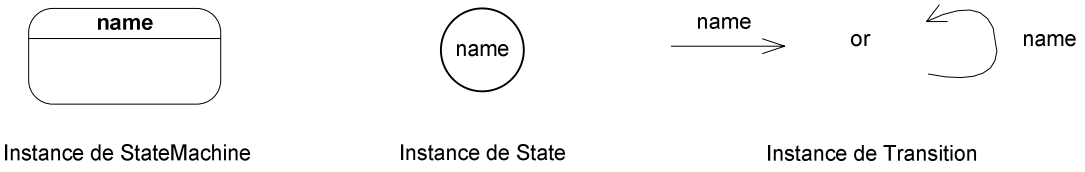

Figure 2 : Représentation des instances.

1.1 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**).

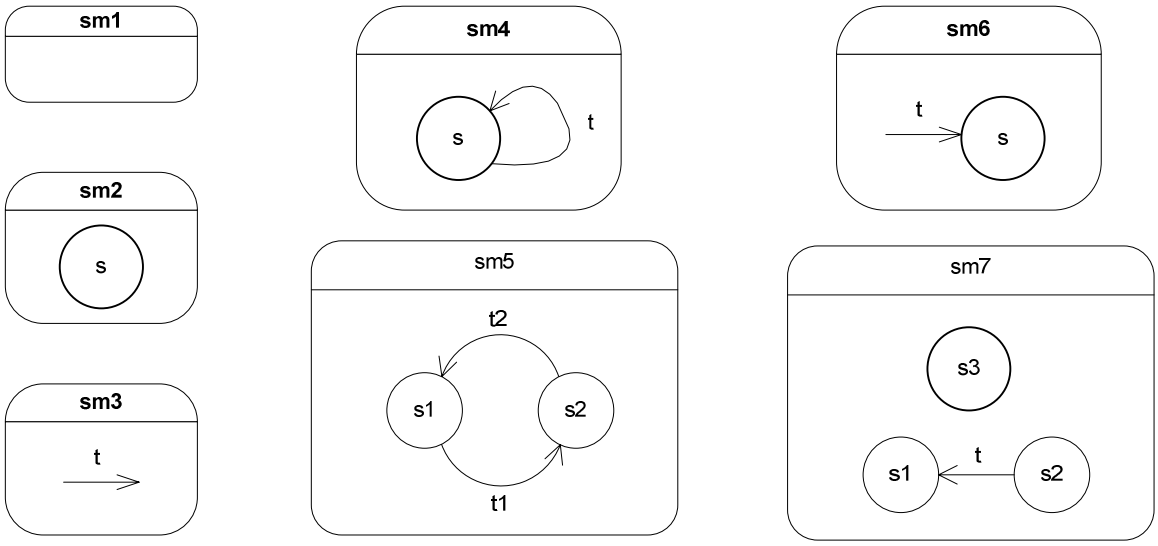

Figure 3 : Instances de StateMachines

StateMachine	sm1	sm2	sm3	sm4	sm5	sm6	sm7
Conformité							

Table 1 : Conformité

1.2 OCL

1.2.1 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).

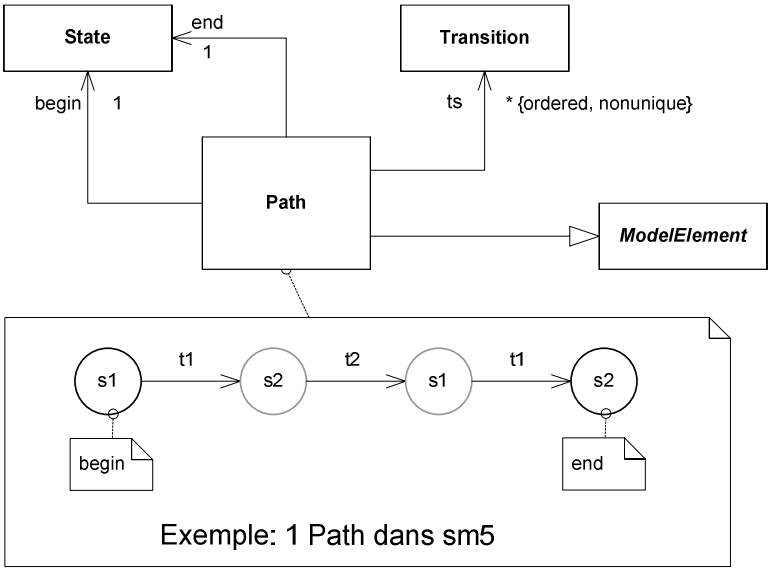

Figure 4 : Enrichissement du métamodèle

1.2.2 Path : contraintes

Q 1.2.2.a Que signifie la contrainte **{ordered, nonunique}** sur la propriété **Path::ts** ?

Exprimez les contraintes suivantes :

Q 1.2.2.b La séquence des transitions du chemin commence par une transition partant de l'état **begin**.

Q 1.2.2.c La séquence des transitions du chemin termine par une transition arrivant à l'état **end**.

Q 1.2.2.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 )
```

1.2.3 Path : opérations

Q 1.2.3.a Que fait l'opération définie par

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

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

Q 1.2.3.b Spécifiez l'opération **Path::isCircuitFree( ): Boolean** qui retourne vrai si et seulement si le chemin est sans circuit.

2 SysML

2.1 Questions générales

Q 2.1.a Que signifie **SysML** ? A quoi sert-il ?
Q 2.1.b Diagrammes **SysML** : Quels sont les types de diagrammes utilisés par **SysML**? En quoi diffèrent-ils des diagrammes **UML**?
Q 2.1.c Que spécifie le paquetage **ModelDomain** (Figure B-2) ? Quel est son intérêt ?

2.2 Cas d'utilisation

Q 2.2.a Expliquez (type et sémantique) tous les éléments du diagramme de la Figure 5 (vous pouvez surcharger la figure).

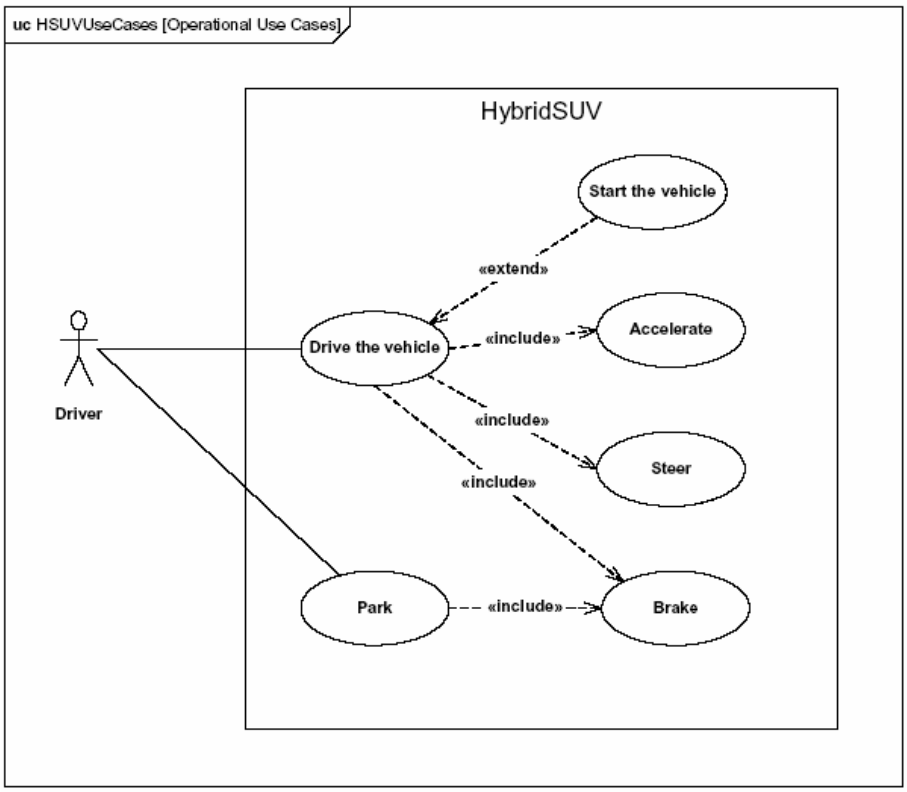

Figure 5 : Operational Use Cases for "Drive the Vehicle"

Q 2.2.b Expliquez (en français) le comportement spécifié dans le diagramme de séquence de la Figure B-7.

2.3 Exigences

Q 2.3.a Que signifie le symbole ⊕ dans la figure B-11 ?
Q 2.3.b Expliquez (type et sémantique) tous les éléments du diagramme de la Figure 6 (vous pouvez surcharger la figure).

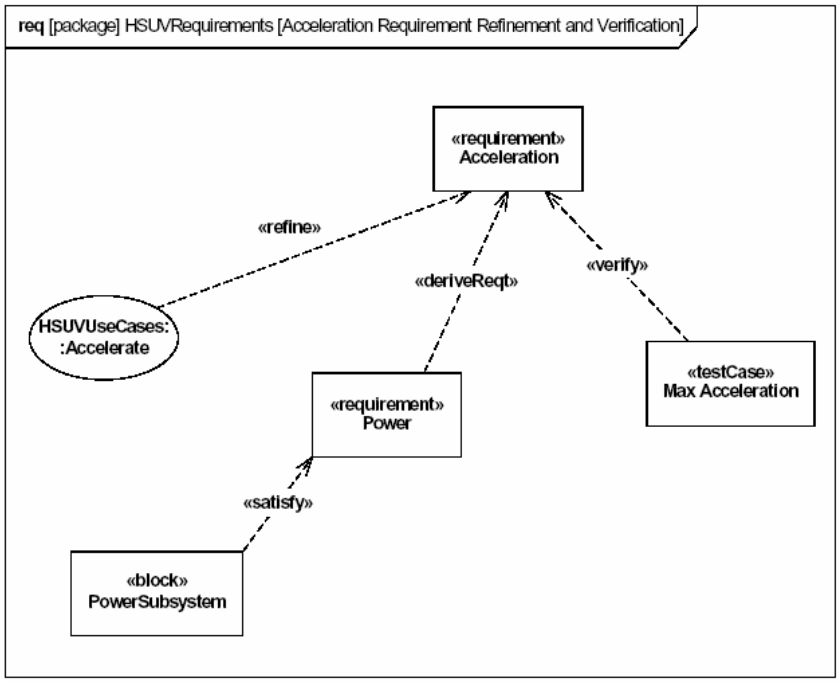

Figure 6 : Accelaration requirement relationship

2.4 Diagrammes de blocs

Q 2.4 Que représentent les diagrammes des figures B-16 et B-17 ? En quoi sont-ils différents ? Quel est le rôle de chacun ?

2.5 Diagramme d'activités

Q 2.5 Expliquez (type et sémantique) les éléments présents dans la Figure 7. Décrivez, en français, le comportement spécifié.

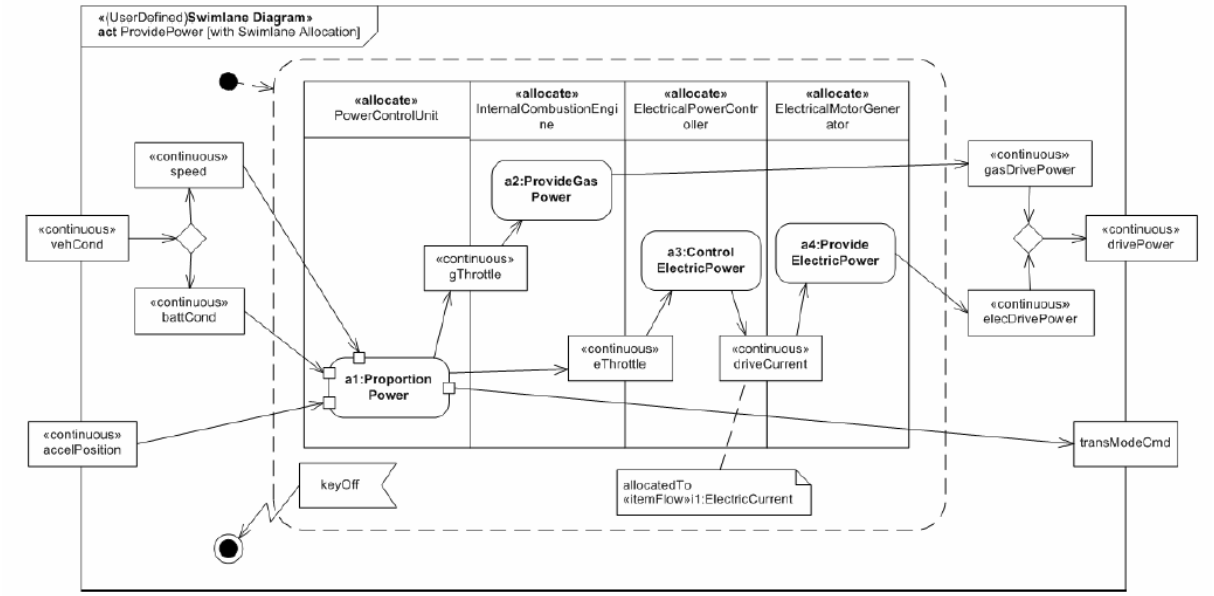

Figure 7 : Activity Diagram for « Provide Power »