Travaux dirigés - OCL - partie 1 (sans collections)

1 Contraintes sur un modèle : les Tortues

Nous proposons d'étudier le modèle de classes pour un système d'information dédié à une association de sauvegarde des tortues terrestres. Les tortues sont décrites par leur nom, âge, taille, sexe, dates auxquelles elles pondent, si elles vivent en captivité, le nom latin de leur espèce, leur taille maximale à l'âge adulte, leur alimentation actuelle et leur alimentation possible. Elles peuvent changer de taille (en vieillissant) et manger des aliments. On peut déterminer par un prédicat si un type d'aliment leur est autorisé. Une espèce de tortue a une répartition géographique constituée de différents lieux. Une tortue habite en un lieu donné. Ce lieu se caractérise par un type de milieu (garrigue, montagne, maquis, désert, etc.). Une espèce de tortue admet un mode d'élevage qui précise la température de jour et la température de nuit, si ce peut être dans un terrarium, si ce peut être en plein air. Une espèce de tortue se caractérise par des particularités biologiques incluant un comportement, si l'espèce hiberne, et un régime alimentaire général composé de types d'aliments admis. Le mode d'élevage précise aussi les types d'aliments utilisé en captivité (régime alimentaire de captivité).

Question 1.1 *Nous complétons la modélisation présentée figure 1 en y ajoutant des contraintes (les contraintes 1 à 3 ne nécessitent pas de connaître les collections) :*

1. *La taille d'une tortue est comprise entre 0 et la taille maximale admise pour son espèce ;*
2. *la température de jour est supérieure à la température de nuit*
3. *on ne peut changer la taille d'une tortue que pour l'augmenter ; Précisez la pré-condition et post-condition de l'opération ;*

2 Contraintes sur un métamodèle : UML2.5

Définissez en OCL les contraintes et les opérations suivantes. Elles font référence aux diagrammes du méta-modèle UML 2.5 dont les noms vous sont rappelés. Notez que l'absence de valeur pour une propriété `p` se contrôle par `p->isEmpty()`.

Multiplicities Diagram
La requête `lowerBound()` retourne la borne inférieure de multiplicité sous forme d'un entier si elle est spécifiée, 1 sinon.

```
context MultiplicityElement
  def : lowerBound() : Integer =
        if (lowerValue=null or lowerValue.integerValue()=null) then 1
        else lowerValue.integerValue() endif
```

La requête `upperBound()` retourne la borne supérieure de multiplicité sous forme d'un entier (éventuellement * pour un nombre indéfini) si elle est spécifiée, 1 sinon.

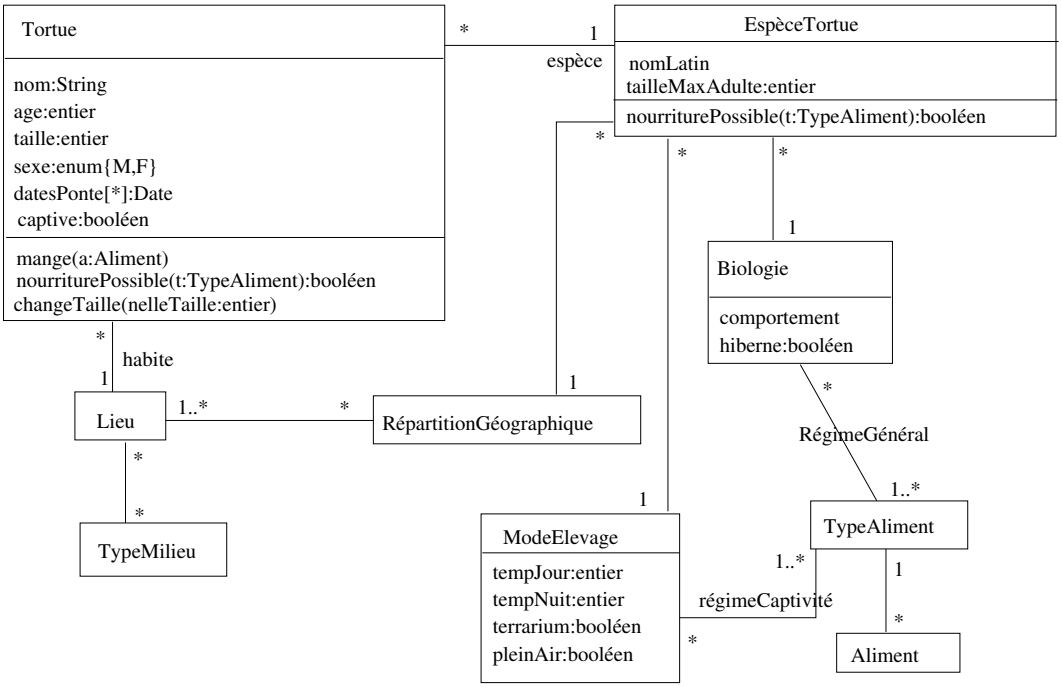

FIGURE 1 – Des tortues

```
context MultiplicityElement
  def : upperBound() : UnlimitedNatural =
        if (upperValue=null or upperValue.unlimitedValue()=null) then 1
        else upperValue.unlimitedValue() endif
```

Question 2.1 *(Multiplicities Diagram)*
Ecrivez en OCL les contraintes suivantes :

- *La borne inférieure doit être positive ou nulle.*
- *La borne supérieure doit être supérieure à la borne inférieure.*
- *La valeur dérivée de* `/lower` *doit être égale à la borne inférieure.*
- *La valeur dérivée de* `/upper` *doit être égale à la borne supérieure.*
- *La requête* `isMultivalued()` *retourne vrai si la propriété peut prendre plus d'une valeur ; elle ne s'applique que lorsqu'une borne supérieure a été spécifiée.*
- *La requête* `includesMultiplicity(M: MultiplicityElement)` *retourne vrai si la multiplicité de l'élément inclut M. Vous devez déterminer les conditions d'application.*

Question 2.2 *(Operation Diagram) Ecrivez en OCL la contrainte suivante :*

- *Une* `bodyCondition` *ne peut être spécifiée que pour une opération de type requête.*

Question 2.3 *(Classes Diagram)*
Ecrivez en OCL les contraintes suivantes :

- *Dans une composition, la multiplicité n'est pas supérieure à 1.*
- *Une union dérivée est une propriété dérivée (particulière).*
- *Une union dérivée est accessible seulement en consultation.*
- *Définir la valeur dérivée de l'attribut* `isComposite`*.*