UFR 922 Informatique
Master Informatique – Spécialité STL
Implantation des langages de programmation (ILP)

TME2 — Syntaxe abstraite d'ILP1

Jacques Malenfant, Christian Queinnec

1 Les schémas RelaxNG

Les liens :

RelaxNG `http://www.oasis-open.org/committees/relax-ng/`

Documents :

Tutoriel	`http://www.oasis-open.org/committees/relax-ng/tutorial-20011203.html`
Syntaxe compacte :	
spécification	`http://www.oasis-open.org/committees/relax-ng/compact-20021121.html`
tutoriel	`http://www.relaxng.org/compact-tutorial-20030326.html`
Livre	`http://books.xmlschemata.org/relaxng/`

Outils spécifiques :

Jing	`http://www.thaiopensource.com/relaxng/jing.html`
Trang	`http://www.thaiopensource.com/relaxng/trang.html`

1.1 Transformer de schémas RelaxNG et valider les documents

L'un des avantages majeurs des schémas RelaxNG est d'avoir une syntaxe compacte claire et lisible. Cependant, pour ne pas retomber dans les difficultés des DTD qui utilisent une syntaxe mixte XML/non-XML dans des documents qui sont supposés être en XML, RelaxNG introduit une distinction claire entre la syntaxe compacte non-XML et la syntaxe XML.

La syntaxe compacte est utilisée par les développeurs qui profitent ainsi de sa lisibilité, alors que la syntaxe XML est utilisée par les outils, comme le valideur Jing, qui profitent de sa facilité de traitement. Pour faire le pont entre les deux syntaxes, l'outil Trang pour passer de la syntaxe compacte à la syntaxe XML.

Pour utiliser Trang, il faut faire :

```
java -jar ${BASE_WS}/Java/jars/trang.jar grammaire.rnc grammaire.rng
```

où BASE_WS est une variable d'environnement donnant le chemin du workspace Eclipse de votre projet ILP1, grammaire.rnc est le fichier contenant le schéma en syntaxe compacte et grammaire.rng est le fichier dans lequel sera écrit le schéma en syntaxe XML.

Dans le système ILP, les transformations des grammaires du format compact au format XML (rnc vers rng) se fait à l'aide de Ant par le fichier de construction build.xml. Cela se fait dans la cible create.rng.schemas en général et create.rng.schema.for.ilp1 pour ILP1 seulement, en appelant une macro appelée trang.one.file de la manière suivante, en prenant la première grammaire en exemple :

```
<trang.one.file rnc.grammar='${rnc}/grammar1.rnc'
                rng.grammar='${rnc}/grammar1.rng' />
```

Lors de l'ajout d'une nouvelle grammaire dans le système, on peut introduire sa transformation en ajoutant un appel de ce genre dans build.xml.

Valider les documents (programmes)

La validation des programmes se fait en utilisant Jing. Comme pour la transformation des schémas, la validation dans le système ILP se fait dans la cible Ant validate.xml.programs en général du fichier build.xml. Cela se fait en appelant jing sur l'ensemble des fichiers de programmes à valider avec le même schéma de la manière suivante, en prenant par exemple les programmes ILP1 :

```
<jing compactsyntax='true' rngfile='${rnc}/grammar1.rnc'>
  <fileset dir='${rnc}/Samples' includes='*-1.xml'/>
</jing>
```

L'appel à Jing utilise ici un schéma en syntaxe compacte donné par l'attribut rngfile (sic) de l'élément jing. Le paramètre, c'est-à-dire le contenu de l'élément jing, est un ensemble de fichier sélectionnés par l'élément fileset, dont les deux attributs donnent le répertoire dans lequel prendre les fichiers et un patron de noms de fichier dans l'attribut includes.

Lorsqu'on veut faire valider des programmes automatiquement lors de la construction par build.xml, il suffit d'ajouter une ligne similaire avec comme schéma la grammaire du langage et de sélectionner les programmes à valider comme ensemble de fichiers à traiter. Pour les besoins de ce TME, on peut créer une cible particulière à ILP1 nommée validate.xml.programs.for.ilp1 et y inclure la validation des programmes développés avec les différentes grammaires étendues.

1.2 Exercices de modification de la grammaire 1

Pour chacun des traits de langage suivant, introduire successivement le trait dans la grammaire de niveau 1 puis donner un programme (en syntaxe XML) qui respecte la nouvelle grammaire et qui utilise le trait ajouté :

1. affectations (déjà vu en TD)
2. boucles while (déjà vu en TD)
3. blocs locaux n-aires
4. définition et application de fonction

2 Manipulation des programmes ILP1 en XML (DOM) et AST

Objectifs : comprendre la manipulation de programmes ILP1 sous la forme de documents XML en Java lorsqu'ils sont représentés par des objets selon le standard DOM et sous la forme d'objets de l'arbre de syntaxe abstraite.

Buts :

- Comprendre plus finement la structure des programmes ILP1 sous leur forme XML.
- Comprendre plus finement la structure des programmes ILP1 sous leur forme d'arbre de syntaxe abstraite.

Reprenez l'exercice du TD2 pour compter les constantes dans un programme ILP1 et réalisez-le en comptant plutôt séparément :

- le nombre de constantes entières,
- le nombre de constantes réelles, et
- le nombre de variables,

dans un programme ILP1 sous sa forme XML et sous sa forme d'arbre de syntaxe abstraite.