Spécifications formelles avec B : TP1

1 Premiers pas avec l'atelier B

L'atelier B est un atelier de génie logiciel basé sur la méthode B. Initialement développé par GEC Alsthom Transport, il est actuellement maintenu et commercialisé par la société ClearSys.

1.1 Lancement et aide en ligne

L'atelier B est disponible sur la machine `marine`. On lance l'interface graphique de l'atelier en tapant: **startAB &**.

Lancez l'interface, examinez les différents boutons et accedez à l'aide en ligne (une fenêtre netscape). En utilisant `acroread`, visualisez les liens suivants :

1. `Le langage B - Manuel de référence` ;
2. `Liste des mots réservés et opérateurs du langage B` ;

Ils vous seront utiles pour la suite.

1.2 Gestion des projets

L'interface graphique s'ouvre sur la fenêtre des *projets*. Un projet développé avec l'atelier B comporte des sources B (machines abstraites, raffinement et implémentations), des obligations de preuve (générées automatiquement), des preuves (réalisées de façon semi-automatique) et, éventuellement, de la documentation et des traductions en `C`,`C++`,`Ada`. Un projet peut aussi faire appel à des librairies standard comme `BASIC_IO` qui gère les entrées-sorties.

Un projet se définit par : son nom, un repertoire *Base de Données Projet* (`bdp`) contenant les fichiers propres à l'atelier B, un repertoire `trad` pour stocker les *traductions* vers des langages classiques, et un repertoire `spec` dans lequel reposent les spécifications des machines B.

2 Spécification de propriétés en B

Pour s'entrainer à spécifier des propriétés, on utilise une machine abstraite un peu particulière. Cette machine ne contient qu'une clause `ASSERTIONS` au sein de laquelle on exprimera certaines propriétés sur les nombres naturels que l'on souhaite prouver.

Commencez par créer un repertoire `Props` avec trois sous repertoires `bdp`, `spec` et `trad`.

Lancez l'atelier B et créez un projet (Bouton `Attach`). Nommez ce projet de façon unique (par exemple `Props_votre_login`) et indiquez respectivement `Props/bdp` et `Props/trad` comme repertoires de Base de Donnée et de Traduction.

Exercice 1 *Dans le repertoire* `Props/spec`*, lancez* `emacs` *et créez un fichier* `ExosDeSpec.mch`*, dans lequel vous spécifierez une machine B qui a la structure suivante:*

```
MACHINE ExosDeSpec
ASSERTIONS
  1=1
END
```

Ajoutez ce composant au projet `Props` *(* `Components/Add` *).*

Exercice 2 *Testez (et corrigez si necessaire) la syntaxe de* `ExosDeSpec`*. (* `Type Check` *).*

Exercice 3 *Générez les obligations de preuves (* `PO Generate` *)*
et prouvez-les (`Prove/Automatic (force 0)` *.*

Pour chacun des exercices suivants,

- ajoutez la formule exprimant la propriété à la machine:

```
MACHINE ExosDeSpec
ASSERTIONS
  F1 & F2 & ...
END
```

- assurez-vous, à l'aide du vérificateur de type de l'atelier B du bon typage des formules écrites.

On ne vous demande pas de les prouver tout de suite. Rappelons qu'en B on note `FIN(E)` l'ensemble des parties finies de `E`.

Exercice 4 *Exprimez : pour tout sous-ensemble fini S_1 des entiers naturels, il existe un ensemble fini S_2 d'entiers naturels contenant strictement S_1.*

Rappelons que la divisibilité sur les entiers est définie par : un entier naturel p divise n s'il existe un entier naturel k tel que $n = pk$ (p et k sont alors des diviseurs de n). Cela peut aussi s'exprimer avec l'opérateur modulo.

Exercice 5 *Exprimez : tout entier naturel admet un diviseur.*

Exercice 6 *Exprimez : tout entier naturel admet un nombre fini de diviseurs.*

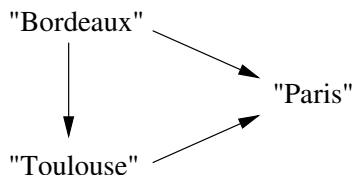
Exercice 7 *Exprimez : pour tout sous-ensemble fini S des entiers naturels, il existe un entier qui n'est divisible par aucun élément de S.*

Exercice 8 *Exprimez (avec et sans quantification) : tout entier naturel est ou bien pair ou bien impair.*

Exercice 9 *Les cinq propriétés précédentes restent-elles vraies avec des entiers de type NAT (entiers naturels codés en machine) ? Expliquez vos réponses.*

3 Spécification ensembliste de fonctions

On considère des graphes finis orientés et sans point isolé dont les sommets sont des chaînes de caractère. On rappelle que le support d'un graphe est l'union de son domaine et de son codomaine: C'est l'ensemble des sommets connectés par des arcs. De même, le degré sortant d'un sommet dans un graphe est le nombre de ses successeurs dans le graphe.



La figure précédente représente le graphe {(Bordeaux $\mapsto$ Paris), (Bordeaux $\mapsto$ Toulouse), (Toulouse $\mapsto$ Paris)}, dont le support est {Bordeaux, Paris, Toulouse}. Le degré sortant du sommet "Bordeaux" est 2.

Exercice 10 *Exprimez formellement (i.e. de manière ensembliste) les objets suivants:*

- *la fonction* `Support` *qui à un graphe associe son support.*
- *la fonction* `Degre` *qui à un sommet (et un graphe) associe son degré sortant.*

```
MACHINE
        Graphs

SETS
        Sommets = {....}
CONSTANTS
        Graphes, ...
PROPERTIES
    Graphes =  FIN(Sommets *  Sommets)
...
END
```