1

Compléments sur les machines abstraites, raffinements et implantations

Machines paramétrées

2

- Une machine peut prendre un ou plusieurs paramètres qui peuvent être scalaires (l'un des types numériques de B ou tout sous-ensemble) ou de type ensemble.
- Toute contrainte sur ces paramètres (dont obligatoirement les typages des paramètres scalaires) est à mettre dans la clause CONSTRAINTS

```
MACHINE GestTrains3(maxtrains)
CONSTRAINTS maxtrains:NAT1
VARIABLES nbTrains
INVARIANT nbTrains:0..maxtrains
INITIALISATION nbTrains:=0
OPERATIONS AddTrain=
        PRE nbTrains<maxtrains THEN
        nbTrains:=nbTrains+1 END
END
```

Opérations avec paramètres d'entrée

3

- Une opération peut aussi prendre un ou plusieurs paramètres.
- Le typage des paramètres d'entrée d'une opération prend naturellement place dans la précondition.

```
MACHINE GestTrains3(maxtrains)
CONSTRAINTS maxtrains:NAT1
VARIABLES nbTrains
INVARIANT nbTrains:0..maxtrains
INITIALISATION nbTrains:=0
OPERATIONS AddTrain(nTrain)=
        PRE nTrain:NAT1&nbTrains<=maxtrains-nTrain THEN
nbTrains:=nbTrains+nTrain END
END
```

Opérations avec paramètres de sortie

- Une opération peut aussi renvoyer un résultat.
- Pas besoin de typer le résultat qui en revanche doit être affecté en fin d'opération (fonction des variables et paramètres de machine et d'opération)

```
OPERATIONS remPlaces<--AddTrain(nTrain)=
PRE nTrain:NAT1&nbTrains<=maxtrains-nTrain
THEN nbTrains,remPlaces:=
nbTrains+nTrain,maxtrains-nbTrains-nTrain
END
```

Gare au bug ici, se souvenir qu'il s'agit de substitutions parallèles qui doivent se lire « à gauche de := les valeurs après, à droite de := les valeurs avant »

Clause SETS

- La clause SETS permet de déclarer les ensembles abstraits (« deferred ») ou énumérés d'une machine, d'un raffinement ou d'une implantation.
- Exemple :

SETS AIG; SIG; APP={VL,AV,SE,CA}

- Un style (répandu mais non obligatoire dans l'atelier B) consiste à mettre en majuscules les identifiants de SETS
- Les ensembles abstraits ou énumérés déclarés dans SETS servent entre autres à typer les variables et les constantes
- En implantation les valeurs des ensembles abstraits sont précisées par la clauses VALUES

Clauses CONSTANTS

- La clause CONSTANTS (ou ce qui revient au même CONCRETE_CONSTANTS) définit des constantes implantables (qui se conserveront dans les raffinements jusqu'à l'implantation).
- La clause ABSTRACT_CONSTANTS définit quant à elle des constantes (qui seront raffinées lors de raffinements).
- Les constantes abstraites ou concrètes doivent obligatoirement être typées par la clause PROPERTIES qui peut aussi exprimer d'autres contraintes.
- En implantation toutes les constantes doivent être concrètes et il doit leur être attribué des valeurs par la clause VALUES.
- Exemple :

  CONSTANTS Aig, Sig, Maxtrain

Clauses PROPERTIES

- La clause PROPERTIES permet d'exprimer des contraintes sur les constantes concrètes ou abstraites (dont obligatoirement leur type).
- Exemple :

```
CONSTANTS Aig, Sig, Maxtrain
PROPERTIES Aig<:AIG & Sig<:SIG &
Aig/={} & Sig/={} & Maxtrain:NAT1
```

Clauses VALUES

- La clause VALUES réservée aux implantations, permet de donner des valeurs implantables aux ensembles abstraits et aux constantes concrètes.
- Les VALUES doivent bien sûr respecter les PROPERTIES (POs spécifiques générées pour cela…).
- Exemple : Clauses de la machine :

  SETS AIG; SIG; APP={VL,AV,CE,CA}

  CONSTANTS Aig, Sig, Maxtrain

  PROPERTIES Aig<:AIG & Sig<:SIG &

  Aig/={} & Sig/={} & Maxtrain:NAT1
- Exemple : Clauses de l'implantation

  VALUES SIG=1..10; AIG=1..10;

  Maxtrain=10; Sig=1..3; Aig=1..3

Clauses VARIABLES

- La clause VARIABLES (ou ce qui revient au même ABSTRACT_VARIABLES) définit des variables d'une machine ou d'un raffinement qui pourront être raffinées. Elle est interdite dans les implantations.
- La clause CONCRETE_VARIABLES définit quant à elle des variables implantables. Elle est utilisable dans les machines, raffinements et implantations.
- Les variables abstraites ou concrètes doivent obligatoirement être typées par la clause INVARIANT qui peut aussi exprimer d'autres contraintes (par des prédicats).
- Les variables abstraites ou concrètes doivent obligatoirement être initialisées par des substitutions généralisées dans la clause INITIALISATION.

Clause ASSERTIONS

- La clause ASSERTIONS sert à ajouter des prédicats censés être déductibles de l'invariant, et ce afin de faciliter les preuves.
- La présence de la clause ASSERTIONS génère un PO ayant objectif de la prouver à partir de l'invariant.
- Une fois prouvée, les prédicats de ASSERTIONS sont rajoutés comme hypothèse dans toutes les preuves.

Clause DEFINITIONS

- La clause DEFINITIONS permet, comme son nom l'indique, l'introduction de définitions qui sont remplacées lors de l'analyse lexicale de la machine.
- Ces DEFINITIONS peuvent dépendre d'un ou plusieurs paramètres formels (entre parenthèses)
- Exemples :
- DEFINITIONS step==5 ; next(x)==x*x+step ;
  affectdouble(y,x)==y:=2*x
- Appels de définitions possibles (par exemple dans une opération) :

  zz:=next(zz) || affectdouble(uu,vv)

Structure type d'une machine abstraite


```
MACHINE M1(pp1,pp2,PSET)
                    /*paramètres optionnels, scalaires et ENSEMBLES*/
CONSTRAINTS pp1:NAT&pp2:BOOL /*typage paramètres scalaires*/
SETS S1;S2={e1,e2,e3} /*Ensembles abstraits et énumérés*/
CONSTANTS C1,C2
PROPERTIES C1:NAT&C2:S2 /*Typage des constantes*/
VARIABLES v1,v2
INVARIANT v1:S1&v2:PSET /*A minima typage des variables*/
INITIALISATION v1,v2:(v1:S1&v2:PSET)
/*Initialisation des variables obligatoire*/
OPERATIONS
setv1(vv) = PRE vv:S1 THEN v1:=vv END;
          /*Typage des paramètres dans PRE*/
res <-- getv1 = BEGIN res:=v1 END /*Opération avec retour. Corps
d'opération (bloc substitution) entre THEN… END ou BEGIN…END*/
END
```

Des clauses de composition (INCLUDES, PROMOTES, EXTENDS, USES, SEES) peuvent aussi être présentes (voir cours modularité)