Université Pierre & Marie Curie DESS – GLA

Programmation – Spécifications formelles
II. Spécification ensemblistes – Z

P. Manoury

Nov. - Déc. *2001*

Le langage des ensembles
Rapide rappels

Appartenance (abstraite) : $x \in y$

Égalité (axiome) : $x = y \Leftrightarrow (\forall z. z \in x \Leftrightarrow z \in y)$

Inclusion (définie) : $x \subseteq y \mathrel{\widehat{=}} (\forall z. z \in x \Rightarrow z \in y)$

Constructions de base et axiomes :

	notation	axiome
Ensemble vide :	$\emptyset$	$\forall x. x \notin \emptyset$
Couples	(x, y)	$(x, y) = (x', y') \Leftrightarrow x = x' \wedge y = y'$
Produit	$X \times Y$	$z \in X \times Y \Leftrightarrow \exists x \in X. \exists y \in Y. z = (x, y)$
Union	$X \cup Y$	$z \in X \cup Y \Leftrightarrow z \in X \vee z \in Y$
Ensemble des parties	$\mathcal{P}(X)$	$z \in \mathcal{P}(X) \Leftrightarrow z \subseteq X$
Schéma de compréhension	$\{x \in X \mid \varphi\}$	$y \in \{x \in X \mid \varphi\} \Leftrightarrow y \in X \wedge \varphi[y/x]$

Construtions définies :

	notation	définition
Intersection	$X \cap Y$	$\{z \in X \mid z \in Y\}$
Différence	$X \backslash Y$	$\{z \in X \mid z \notin Y\}$

etc.

Notions fondamentales
Relations – fonctions

Ensemble des relations binaires entre X et Y (définition) :

$$\boxed{X \leftrightarrow Y \mathrel{\widehat{=}} \mathcal{P}(X \times Y)}$$

Relations entre éléments (notation) :

$$\boxed{\text{si } R \in X \leftrightarrow Y,\ x \,\underline{R}\, y \mathrel{\widehat{=}} (x, y) \in R}$$

Domaine et codomaine de $R \in X \leftrightarrow Y$ (définitions) :

$$\boxed{\begin{array}{c} \mathrm{dom}(R) \mathrel{\widehat{=}} \{x \in X \mid \exists y \in Y . x \,\underline{R}\, y\} \\ \mathrm{ran}(R) \mathrel{\widehat{=}} \{y \in Y \mid \exists x \in X . x \,\underline{R}\, y\} \end{array}}$$

Ensemble des fonctions partielles de X vers Y (définition) :

$$\boxed{X \nrightarrow Y \mathrel{\widehat{=}} \{f \in X \leftrightarrow Y \mid \forall x \in \mathrm{dom}(f) . \exists ! y \in Y . x \,\underline{f}\, y\}}$$

Ensemble des fonctions totales de X vers Y (définition) :

$$\boxed{X \rightarrow Y \mathrel{\widehat{=}} \{f \in X \nrightarrow Y \mid \mathrm{dom} f = X\}}$$

Application d'une fonction $f \in X \nrightarrow Y$ (axiome) :

$$\boxed{f(x) = y \Leftrightarrow (x, y) \in f}$$

Opération de «mise à jour» :

$$\boxed{\begin{array}{ll} R \oplus S \mathrel{\widehat{=}} \{\ (x, y) \in R \cup S \mid & (x \in \mathrm{dom}(R) \backslash \mathrm{dom}(S) \Rightarrow (x, y) \in R) \wedge \\ & (x \in \mathrm{dom}(S) \Rightarrow (x, y) \in S) \quad \} \end{array}}$$

Exemple : en notant $x \mapsto y$ le couple (x, y)

$$\left\{ \begin{array}{lcl} f \oplus \{x \mapsto y\}(x) & = & y \\ f \oplus \{x \mapsto y\}(z) & = & f(z) \quad \text{si } z \neq x \end{array} \right.$$

Notions fondamentales (suite)
Arithmétique

L'ensemble des entiers naturels (définition) :

$$\boxed{\mathit{IN} \mathrel{\widehat{=}} \textit{ un truc un peu compliqué ...}}$$

Constantes, opérations, relations :

$$\boxed{0\ 1\ \ldots\ +\ \times\ \leq\ \ldots}$$

Entiers non nuls (définition) :

$$\boxed{\mathit{IN}_1 \mathrel{\widehat{=}} \{n \in \mathit{IN} \mid n \neq 0\}}$$

Intervales d'entiers (définition) :

$$\boxed{n..m \mathrel{\widehat{=}} \{k \in \mathit{IN} \mid n \leq k \wedge k \leq m\}}$$

etc.

Structure linéaire générique
Les suites

Ensemble des suites d'éléments de X (définition) :

$$\boxed{\text{seq } X \mathrel{\widehat{=}} \{s \in I\!N_1 \mathrel{+\!\!\!\rightarrow} X \mid \exists n \in I\!N.\text{dom}(s) = 1..n\}}$$

Accés aux éléments de $s \in \text{seq } X$:

$$\boxed{s(i)}$$

avec $i \in \text{dom}(s)$

Suite vide :

$$\boxed{\langle\rangle \mathrel{\widehat{=}} \emptyset}$$

remarque : $\emptyset \in 1..0 \mathrel{+\!\!\!\rightarrow} X \in \text{seq } X$

Longueur d'une suite :

$$\boxed{\#s}$$

cardinal de l'ensemble s

Suites non vides (définition) :

$$\boxed{\text{seq}_1 \ X \mathrel{\widehat{=}} \{s \in \text{seq } X \mid s \neq \langle\rangle\}}$$

Fonctions sur les suites
Définitions axiomatiques

Format des définitions : $\begin{array}{|l} nom : type \\ \hline formule \end{array}$

Concaténation :

$$\begin{array}{|l}
\frown : \text{seq } X \times \text{seq } X \to \text{seq } X \\ \hline
\forall s_1, s_2 \in \text{seq } X . \forall i \in I\!N_1. \\
\quad (i \leq \# s_1 \Rightarrow s_1 \frown s_2(i) = s_1(i)) \wedge \\
\quad (i > \# s_1 \Rightarrow s_1 \frown s_2(i) = s_2(i - \# s_1))
\end{array}$$

Sous suite :

$$\begin{array}{|l}
sub : \text{seq } X \times I\!N \times I\!N \to seqX \\ \hline
\forall s \in \text{seq } X . \forall i, j \in I\!N. \\
\quad \text{dom}(sub(s,i,j)) = 1..j - i + 1 \\
\quad \forall k \in \text{dom}(sub(s,i,j)).sub(s,i,j)(k) = s(k+i-1)
\end{array}$$

Comme des listes :

Constructeurs

$$\begin{array}{|l}
nil : \text{seq } X \\ \hline
\quad nil = \emptyset
\end{array}
\qquad
\begin{array}{|l}
cons : X \times \text{seq } X \to \text{seq}_1 \ X \\ \hline
\forall s \in \text{seq } X. \\
\quad cons(x, s) = \{1 \mapsto x\} \frown s
\end{array}$$

Accesseurs

$$\begin{array}{|l}
car : \text{seq}_1 \ X \to X \\ \hline
\forall s \in \text{seq}_1 \ X. \\
\quad car(s) = s(1)
\end{array}
\qquad
\begin{array}{|l}
cdr : \text{seq}_1 \ X \to \text{seq } X \\ \hline
\forall s \in \text{seq}_1 \ X. \\
\quad cdr(s) = sub(s, 2, \# s)
\end{array}$$

Comme des files d'attentes :

Constructeurs

$$\begin{array}{|l}
new : \text{seq } X \\ \hline
\quad new = \emptyset
\end{array}
\qquad
\begin{array}{|l}
add : \text{seq } X \times X \to \text{seq}_1 \ X \\ \hline
\forall s \in \text{seq } X. \\
\quad add(s, x) = s \frown \{1 \mapsto x\}
\end{array}$$

Accesseurs

$$\begin{array}{|l}
last : \text{seq}_1 \ X \to X \\ \hline
\forall s \in \text{seq}_1 \ X. \\
\quad last(s) = s(\# s)
\end{array}
\qquad
\begin{array}{|l}
front : \text{seq}_1 \ X \to \text{seq } X \\ \hline
\forall s \in \text{seq}_1 \ X. \\
\quad front(s) = sub(s, 1, \# s - 1)
\end{array}$$

Retournement d'une suite
Schéma d'opérations

Relation (prédicat) *avant-aprés* :

$$Rev \in \text{seq } X \leftrightarrow \text{seq } X$$

Intuitivement, procédure *vs* fonction :

$$s \ \underline{Rev} \ s' \mathrel{\widehat{=}} \text{«}s' \textit{ est l'état de } s \textit{ aprés l'opération}\text{»}$$

Format des schémas d'opération :

$$\begin{array}{|l}
\multicolumn{1}{l}{Nom} \\ \hline
\textit{déclarations} \\ \hline
\textit{formule} \\ \hline
\end{array}$$

L'opération de retournement de suite :

$$\begin{array}{|l}
\multicolumn{1}{l}{Rev} \\ \hline
s, s' : \text{seq } X \\ \hline
\text{dom}(s) = \text{dom}(s') \land \\
\forall i \in \text{dom}(s'), s'(i) = s(\#s - i + 1) \\ \hline
\end{array}$$

Schémas d'opérations
Quelque manipulations

Apostrophe :

$$\begin{array}{|l} \multicolumn{1}{l}{S} \\ \hline x : T \\ \hline \varphi \\ \hline \end{array} \quad \rightsquigarrow \quad \begin{array}{|l} \multicolumn{1}{l}{S'} \\ \hline x' : T \\ \hline \varphi[x'/x] \\ \hline \end{array}$$

Conjonction :

$$\begin{array}{|l} \multicolumn{1}{l}{S_1} \\ \hline x_1 : T_1 \\ \hline \varphi_1 \\ \hline \end{array} \quad \begin{array}{|l} \multicolumn{1}{l}{S_2} \\ \hline x_2 : T_2 \\ \hline \varphi_2 \\ \hline \end{array} \quad \rightsquigarrow \quad \begin{array}{|l} \multicolumn{1}{l}{S_1 \wedge S_2} \\ \hline x_1 : T_1 \\ x_2 : T_2 \\ \hline \varphi_1 \wedge \varphi_2 \\ \hline \end{array}$$

Inclusion :

$$\begin{array}{|l} \multicolumn{1}{l}{S_2} \\ \hline S_1 \\ x_2 : T_2 \\ \hline \varphi_2 \\ \hline \end{array} \quad \widehat{=} \quad \begin{array}{|l} \multicolumn{1}{l}{S_2} \\ \hline x_1 : T_1 \\ x_2 : T_2 \\ \hline \varphi_1 \wedge \varphi_2 \\ \hline \end{array}$$

Identificateurs :

$$\begin{array}{|l} \multicolumn{1}{l}{S} \\ \hline x_1 : T_1 \\ \dots \\ x_n : T_n \\ \hline \varphi \\ \hline \end{array} \quad \rightsquigarrow \quad \theta S \ \widehat{=} \ (x_1, \dots, x_n)$$

IBM CICS API
Temporary Storage Queue

Ensembles de base :

$$\begin{array}{rcl} BYTE & \widehat{=} & 0..255 \\ TSElem & \widehat{=} & \mathrm{seq}\ BYTE \end{array}$$

Une suite et un pointeur (sur le dernier objet modifié) :

$$\begin{array}{|l} \multicolumn{1}{l}{TSQ} \\ \hline ar : \mathrm{seq}\ TSElem \\ p : I\!N \\ \hline p \leq \#ar \\ \hline \end{array}$$

Initialisation :

$$\begin{array}{|l} \multicolumn{1}{l}{TSQInit} \\ \hline TSQ \\ \hline ar = \langle\rangle \wedge p = 0 \\ \hline \end{array}$$

Schéma «avant-aprés» :

$$\begin{array}{|l} \multicolumn{1}{l}{\Delta TSQ} \\ \hline TSQ \wedge TSQ' \\ \hline \\ \hline \end{array}$$

Temporary Storage Queue
Opérations fournies

Entrées-sortie, notations conventionnelles :

S

$$\begin{array}{|l} in? : T_1 \\ out! : T_2 \\ \ldots \\ \hline \varphi \end{array}$$

Ajout et retrait :

$Append0$

$$\begin{array}{|l} \Delta TSQ \\ from? : TSElem \\ item! : IN \\ \hline ar' = ar ^\frown \langle from? \rangle \wedge \\ item! = \#ar' \wedge \\ p' = p \end{array}$$

$Remove0$

$$\begin{array}{|l} \Delta TSQ \\ item! : TSElem \\ \hline p < \#ar \wedge \\ p' = p + 1 \wedge \\ into! = ar(p') \wedge \\ ar' = ar \end{array}$$

Lecture et écriture :

$Write0$

$$\begin{array}{|l} \Delta TSQ \\ item? : IN \\ from? : TSElem \\ \hline item? \in 1..\#ar \wedge \\ ar' = ar \oplus \{item? \mapsto from?\} \wedge \\ p' = p \end{array}$$

$Read0$

$$\begin{array}{|l} \Delta TSQ \\ item? : IN \\ into! : TSElem \\ \hline item? \in 1..\#ar \wedge \\ into! = ar(item?) \wedge \\ p' = item? \wedge \\ ar' = ar \end{array}$$

Temporary Storage Queue
Gestion d'erreurs

Statuts d'exécution :

$$OpStatus ::= Success \mid ItemErr \mid NoSpace$$

(macro syntaxique pour $OpStatus = \{Success, ItemErr, NoSpace\}$)

Schéma générique d'erreur :

$ERROR$

$$\begin{array}{|l}
\Delta TSQ \\
report! : OpStatus \\
\hline
\theta TSQ' = \theta TSQ
\end{array}$$

Rapports d'erreurs :

$NoneLeft$

$$\begin{array}{|l}
ERROR \\
\hline
p = \#ar \land \\
report! = ItemErr
\end{array}$$

$OutOfBounds$

$$\begin{array}{|l}
ERROR \\
item? : \mathbb{N} \\
\hline
item? \notin 1..\#ar \land \\
report! = ItemErr
\end{array}$$

$OutOfSpace$

$$\begin{array}{|l}
ERROR \\
\hline
report! = NoSpace
\end{array}$$

Tout va bien :

$Successful$

$$\begin{array}{|l}
report! : OpStatus \\
\hline
report! = Success
\end{array}$$

Temporary Storage Queue
Intégration

$$
\begin{array}{lcl}
Append & \widehat{=} & (Append0 \wedge Successful) \vee OutOfSpace \\
Remove & \widehat{=} & (Remove0 \wedge Successful) \vee NoneLeft \\
Write & \widehat{=} & (Write0 \wedge Successful) \vee OutOfBound \vee OutOfSpace \\
Read & \widehat{=} & (Read0 \wedge Successful) \vee OutOfBound
\end{array}
$$