Introduction au langage Z

Cours LOG2000 - Éléments du génie logiciel

S.Fourmanoit E.Merlo R.Roy

Résumé

Ce document présente les concepts de base de **Z**, un langage de spécifications formelles. Il présuppose de la part du lecteur une connaissance intuitive de la théorie des ensembles et de la logique des propositions.

Table des matières

Introduction

Méthodes formelles Techniques, basées sur les mathématiques, pouvant être utilisées à tous les stages du développement logiciel. Par « formelles », on désigne des méthodes dont le but principal est de rendre précices des idées de développement auparavant vagues ou simplement intuitives [SP98].

Langage de spécifications formelles Sert à la représentation abstraite de certains aspects d'un produit logiciel, principalement quant à la manipulation des données (*data manipulation design*). Par « abstraite », on désigne une représentation parfaitement découplée de toute implémentation, bien que suffisement précise et contraignante pour l'orienter [WW96].

La notation Z

Au point de vue sémantique, le **Z** peut être subdivisé en trois parties, dont seulement les deux premières seront couvertes dans cette introduction :

1. Un langage mathématique utilisants propositions, logique prédicative, ensembles et relations.
2. Un langage schématique représentant des manipulations algorithmiques de données ou des objets.
3. Une théorie du raffinement entre des types abstraits de données.

Ainsi, on parle souvent de **Z** comme d'un langage de description d'états (*states description language*) : en utilisant un dialecte mathématique particulier enchâssé dans une notation schématique, il décrit les différents états de données par lequel peut passer un produit logiciel, et sous quelles conditions [DW96].

Considérations importantes

La notation mathématique de Z n'est pas une notation mathématique conventionnelle. La notation mathématique de **Z** mêle en une seule entité différents concepts et écritures provenant aussi bien de l'algèbre, de la théorie des ensembles ou de la logique des prédicats que des besoins ou de l'imagination de ses différents auteurs : gardez donc toujours à l'esprit que le sens d'un énoncé **Z** n'est pas toujours celui que vous lui donneriez intuitivement par votre culture mathématique, et ne vous étonnez pas de rencontrer à l'occasion des notations qui ne vous disent rien.

La notation schématique de Z n'est pas mathématique. Bien qu'elle fasse appel aux mathématiques, la notation schématique de **Z**, qui permet de faire le lien entre les mathématiques pures et une abstraction informatique, n'est basée sur aucune formalisation. Elle n'est qu'une commodité qu'il est important que vous distinguiez du reste [SP98].

Rappel - logique propositionnelle

La base

- Une proposition est une expression qui est soit vraie, soit fausse.
- Des propositions peuvent être combinées entre elles par des connecteurs logiques.
- La signification d'une telle combinaison est déterminée par la signification des propositions impliquées.

Connecteurs logiques

$\neg$	négation	(non)
$\wedge$	conjonction	(et)
$\vee$	disjonction	(ou inclusif)
$\Rightarrow$	implication	(si, alors)
$\Leftrightarrow$	équivalence	(si et seulement si)

Rappel - logique propositionnelle (suite)

Priorité des opérations En **Z**, les connecteurs logiques ont une priorité d'opération qui diminue selon l'ordre du tableau précédent : la négation a la plus haute priorité, et l'équivalence la plus basse. Ainsi, la proposition :

$$\neg\, p \land q \lor r \Leftrightarrow q \Rightarrow p \land r$$

est équivalente à :

$$(((\neg\, p) \land q) \lor r) \Leftrightarrow (q \Rightarrow (p \land r))$$

Rappel - logique propositionnelle (suite et fin)

Propositions atomiques Une proposition atomique est une proposition qui n'utilise pas de connecteurs logiques. En **Z**, une proposition atomique ne pourra affirmer (ou infirmer) que deux choses :

1. que deux éléments sont congrus ou que deux ensembles sont identiques. Il s'agit d'une extension **Z** de la notion d'égalité à la théorie des ensembles.
2. qu'un élément ou un ensemble maintient une certaine relation (via des opérateurs relationnels) avec d'autres éléments ou ensemble.

Ainsi, les propositions atomiques sont à la base de toutes les constructions mathématiques ultérieures en **Z** : même si elles en sont les éléments constitutifs, elle échappent au langage en ce sens qu'elles prennent leur sens dans des cadres formels extérieurs. Par exemple, la proposition atomique fausse « $3+3=8$ » a un sens précis qui provient de l'arithmétique : elle n'en sera pas moins acceptée en **Z**.

Introduction - logique des prédicats

Définition intuitive du prédicat Un prédicat est une expression contenant des inconnues qui se transforme en proposition quand celles-ci sont identifiées. Par exemple, l'expression $x > 3$ est un prédicat arithmétique pour tout x appartenant aux réels.

Déclaration L'exemple précédent met clairement en relief la nécessité de déclaration, d'introduction des variables impliquées dans une expression pour connaitre sa valeur prédicative. En **Z**, on notera ainsi $x : a$ le fait que la variable x sera forcément élément de l'ensemble a.

Quantification Il s'agit d'une des opérations les plus importantes en **Z**. Soit p, un prédicat impliquant la variable x, élément de a. On pourra retransformer ce prédicat en proposition en appliquant un quantificateur $\mathcal{Q}$ à x. En **Z**, on notera :

$$\mathcal{Q}\, x : a \bullet p$$

Introduction - logique des prédicats (suite)

Quantificateurs Trois quantificateurs sont principalement utilisés en **Z** :

- Quantificateur d'universalité $\forall$
  $\forall\, x : a \bullet p$
  Pour tout x élément de a, p est une proposition vraie.

- Quantificateurs d'existence $\exists$
  $\exists\, x : a \bullet p$
  Pour certains x éléments de a, p est une proposition vraie.

- Quantificateur d'unicité $\exists_1$
  $\exists_1\, x : a \bullet p$
  Pour un et un seul x élément de a, p est une proposition vraie.

Introduction - logique des prédicats (suite)

Contraintes Il est possible d'ajouter un prédicat r à la déclaration d'une proposition quantifiée pour restreindre le domaine d'une variable.

$$\mathcal{Q}\, x : a \mid r \bullet p$$

Ici, la variable x est restreinte aux éléments de a pour lesquels r est vrai. Par exemple, il nous est possible d'écrire les tautologies :

$$(\forall\, x : a \mid r \bullet p) \quad \Leftrightarrow \quad (\forall\, x : a \bullet r \Rightarrow p)$$
$$(\exists\, x : a \mid r \bullet p) \quad \Leftrightarrow \quad (\exists\, x : a \bullet r \wedge p)$$

Introduction - logique des prédicats (suite)

Propositions quantifiées : autres notions

- Dans l'expression $\mathcal{Q}\, x : a \mid r \bullet p$, on dit que la variable x est liée au quantificateur $\mathcal{Q}$.
- Si une variable x apparaît dans un prédicat sans être préalablement liée à un quantificateur, on dit que x est libre dans ce prédicat.

Substitution Il s'agit d'une facilité de **Z** permettant parfois d'alléger les notations. Retenez-la comme une curiosité utile. On écrira :

$$p[t/x]$$

pour représenter le prédicat qui résulte de la substitution par t de chaque occurence libre de x dans le prédicat p. Cette opération a une priorité plus haute que toutes les opérations logiques vues précédemment.

Introduction - logique des prédicats (suite et fin)

Description définie Désigner « un objet d'un certain genre ayant telle caractéristique » peut être accompli dans une syntaxe très proche de celle utilisée précédemment avec des quantificateurs sur des prédicats. Cependant, le résultat de la désignation n'est pas ici une proposition logique, mais plutôt un terme qui représente l'objet décrit : c'est une description définie.

La notation utilisée, très proche de celle des quantificateurs, se nomme notation mu. Soit une déclaration D, un prédicat P exprimant une contrainte sur les variables déclarées en D et une expression E, donnant facultativement la valeur que l'on désire obtenir. On écrira la description définie : $\mu\, D \mid P \bullet E$. Par exemple, le cube de l'entier supérieur à zéro dont le carré est 100 s'écrira :

$$\mu\, x : \mathbb{Z} \mid x > 0 \land x^2 = 100 \bullet x^3$$

Rappel - Théorie des ensembles

Définition d'un ensemble Un ensemble est une collection bien définie d'éléments.

Construction d'ensembles en Z On peut y bâtir des ensembles principalement de quatre façons :

- **Par extension**. On se contente d'en faire la liste des éléments : $\{0, 1, 4, 9, 16\}$

- **Par compréhension**. De façon très similaire à la quantification de prédicats, on pourra définir un ensemble sous forme $\{D \mid P \bullet E\}$, pour laquelle D sera une déclaration, P un prédicat dépendant de la déclaration qui définira une contrainte sur celle-ci et E une expression définissant chacun des termes de l'ensemble. Ainsi, l'ensemble précédent devient : $\{x : \mathbb{N} \mid x \leq 4 \bullet x^2\}$

Rappel - Théorie des ensembles (suite)

Construction d'ensemble en Z (suite et fin)

– **Par l'opérateur d'ensemble de puissance** $\mathbb{P}$. La notation $\mathbb{P}\, a$ désigne l'ensemble des sous-ensembles de a (incluant $\varnothing$, l'ensemble vide). Par exemple :
$\mathbb{P}\{1, 2\} = \{\varnothing, \{1\}, \{2\}, \{1, 2\}\}$

– **Par le produit cartésien**. Le produit cartésien $a \times b$ désigne l'ensemble des paires ordonnées ayant comme premier membre un élément de a et comme second membre un élément de b. Ainsi :
$\{x, y\} \times \{1, 2\} = \{(x, 1), (x, 2), (y, 1), (y, 2)\}$

Rappel - Théorie des ensembles (suite et fin)

Appartenance à un ensemble On écrira respectivement $x \in a$ ou $x \notin a$ en **Z** pour affirmer que x est élément ou non à l'ensemble a.

Inclusion d'ensembles On écrira respectivement $a \subseteq b$ ou $a \not\subseteq b$ en **Z** pour affirmer que a est sous-ensemble de b ou non.

Les opérations sur les ensembles en Z

$\#$	cardinal	$\#\{3,4\} = 2$
$\cup$	union	$\{1,2\} \cup \{2,3\} = \{1,2,3\}$
$\cap$	intersection	$\{1,2\} \cap \{2,3\} = \{2\}$
$\setminus$	différence	$\{1,2\} \setminus \{2,3\} = \{1\}$
$\bigcup$	union généralisée	$\bigcup\{\{1,2\},\{2,3\}\} = \{1,2,3,4\}$
$\bigcap$	intersection généralisée	$\bigcap\{\{1,2\},\{2,3\}\} = \{2\}$

Extension Z de la théorie des ensembles

Les types Mathématiquement, on dit que les ensembles sont homogènes : ils représentent des valeurs ayant des caractéristiques communes, souvent rattachées entre elles par une règle d'inférence. Cette caractéristique qu'auront des éléments « d'être de la même sorte » est très importante du point de vue de la formalisation de produits logiciels, puisqu'elle cristallise l'idée intuitive des créateurs selon laquelle certains membres d'un système auront des caractéristiques communes [WW96].

Ainsi apparaît naturellement la notion de type. En **Z**, un type est un « ensemble d'extension maximale, tout au moins dans les limites d'une spécification donnée [DW96] » . Ainsi, chaque élément, chaque objet x présent dans une spécification sera associé à un et un seul type : le plus grand ensemble a existant pour lequel $x \in a$.

Extension Z de la théorie des ensembles (suite)

Déclaration de types En **Z**, la déclaration d'ensembles « types » se fait habituellement de façon axiomatique, selon une notation particulière, soit l'énumeration entre crochets carrés de ceux-ci au début d'une spécification. Ainsi, la ligne suivante défini deux types, Fruits et Légumes, utilisable par la suite dans une spécification quelconque.

$[Fruits, Légumes]$

Notons sans explication que le caractère particulier des ensembles types prévient leur construction plus avant par les méthodes déja vues. Par une notation spéciale, on pourra cependant écrire, si on le désire :

$Fruits ::= pomme \mid poire \mid banane$

Extension Z de la théorie des ensembles (suite)

Abbréviation Un « objet » e (élément, ensemble, etc) déjà défini dans une spécification formelle peut se voir attribuer un nom x par l'opération $x == e$. Comme nous le verrons avec les relations, cette notation est particulièrement utile pour donner des définitions génériques. Ainsi, on nommera a l'ensemble des carrés des nombres naturels inférieurs à cinq en écrivant :

$$a == \{x \in \mathbb{N} \mid x < 5 \bullet x^2\}$$

Types numériques L'ensemble des naturels $\mathbb{N}$ et des entiers naturels $\mathbb{Z}$ sont des types prédéfinis en **Z**. Tel que mentionné précédemment, l'ensemble des propositions atomiques de nature arithmétique que ces nombres permettent sont ainsi utilisables en **Z** : addition, soustraction, multiplication, division entière, reste modulo, puissance, plus petit que, plus grand que, égalité, etc.

Extension Z de la théorie des ensembles (suite et fin)

Notations particulières impliquant les types numériques $\mathbb{N}$ et $\mathbb{Z}$

- La notation a..b permet de désigner un sous-ensemble de $\mathbb{N}$ ou de $\mathbb{Z}$: $5 \mathinner{..} 8 = \{5, 6, 7, 8\}$, et $2 \mathinner{..} 1 = \varnothing$

- La notation succ x permet de connaitre l'entier égal à $x + 1$: $\mathrm{succ}\ 3 = 4$

- Les notations min a et max a donnent respectivement la plus petite et la plus grande des valeurs de a, sous-ensemble correctement borné non vide de $\mathbb{N}$ ou de $\mathbb{Z}$. Ainsi : $\max\{x : \mathbb{N} \mid x \leq 4 \bullet x^2\} = 16$

Les relations

Intérêt des relations Nous avons introduit précédemment l'utilité du typage, pour regrouper un ensemble d'éléments ayant des caractéristiques communes. Cependant, il est évident qu'un produit logiciel fait interagir une séries de composantes (données, objets) qui sont de types différents. C'est là toute l'utilité des relations : modéliser la structure des intéractions entre des composantes de différentes natures [WW96].

Définitions

- **Couple ordonné**. Soit deux éléments distincts, x et y. On appelera couple ordonné x vers y la paire ayant comme premier élément x (élément de départ) et comme dernier élément y (élément d'arrivée). On la notera : $x \mapsto y$ ou encore (x, y).
- **Relation**. Un ensemble de paires ordonnées telles que leurs éléments de départ et d'arrivée font respectivement partie d'un même ensemble.

Les relations (suite)

Notation Z des relations Soit deux ensembles, X et Y. Toute relation R ayant X comme ensemble de départ et Y comme ensemble d'arrivée sera telle que :

$$R : \mathbb{P}(X \times Y)$$

Par volonté de simplification, on utilise habituellement une notation différente qui aura exactement le même sens :

$$X \leftrightarrow Y == \mathbb{P}(X \times Y)$$

Les relations (suite)

Domaine, image, relation inverse Soit deux ensembles, X et Y, et la relation $R : X \leftrightarrow Y$. En **Z**, les ensembles suivants sont définis :

$$\begin{aligned} \operatorname{dom} R &== \{x : X, y : Y \mid x \mapsto y \in R \bullet x\} \\ \operatorname{ran} R &== \{x : X, y : Y \mid x \mapsto y \in R \bullet y\} \\ R^{\sim} &== \{x : X, y : Y \mid x \mapsto y \in R \bullet y \mapsto x\} \end{aligned}$$

Restrictions de domaine et d'image Soit deux ensembles A et B, et les ensembles X, Y et R tels que définis précédemment. En **Z**, on définira :

$$\begin{aligned} A \lhd R &== \{x : X, y : Y \mid x \mapsto y \in R \land x \in A \bullet x \mapsto y\} \\ R \rhd B &== \{x : X, y : Y \mid x \mapsto y \in R \land y \in B \bullet x \mapsto y\} \end{aligned}$$

Les relations (suite)

Soustraction de domaine et d'image Soit les ensembles X, Y, R, A et B définis précédemment. En **Z**, les ensembles suivant sont également définis :

$$A \lhd R == \{x : X, y : Y \mid x \mapsto y \in R \land x \notin A \bullet x \mapsto y\}$$
$$R \rhd B == \{x : X, y : Y \mid x \mapsto y \in R \land y \notin B \bullet x \mapsto y\}$$

Image relationelle Avec, encore une fois, les ensembles X, Y, R et A, on définira :

$$R(\!|A|\!) == \mathrm{ran}(A \lhd R)$$

Les relations (suite)

Composition Soit trois ensembles, X, Y et Z, et deux relations R et S telles que $R : X \leftrightarrow Y$ et $S : Y \leftrightarrow Z$. On définira en **Z** la composition par :

$$R \fatsemi S == \{x : X, z : Z \mid (\exists\, y : Y \bullet (x \mapsto y \in R \land y \mapsto z \in S)) \bullet x \mapsto z\}$$

Itération Soit, encore une fois, la relation $R : X \leftrightarrow Y$. On définira récursivement :

$$R^k == R \fatsemi R^{k-1}$$

avec $R^1 == R$.

Les relations (suite)

Fonctions : quelques définitions

- **Une fonction** est une relation telle que tous les éléments de l'ensemble de départ sont impliqués dans au plus un couple ordonné de celle-ci.
- **Une fonction totale** est une fonction pour laquelle le domaine correspond à l'ensemble de départ.
- **Une fonction partielle** est une fonction pour laquelle le domaine est strictement inclu dans l'ensemble de départ.
- **Une fonction injective** est une fonction pour laquelle à chaque élément du domaine correspond un élément différent de l'image.
- **Une fonction surjective** est une fonction pour laquelle l'image correspond à l'ensemble d'arrivée.
- **Une fonction bijective** est une fonction à la fois injective et surjective.
- **Une fonction finie** est une fonction dont le cardinal existe.

Les relations (suite)

Fonctions : les notations spécifiques de déclaration

- $\rightarrow$ fonction totale
- $\nrightarrow$ foncton partielle
- $\rightarrowtail$ fonction totale injective
- $\nrightarrow\!\!\!\!\!\!\rightarrowtail$ fonction partielle injective
- $\twoheadrightarrow$ fonction totale surjective
- $\nrightarrow\!\!\!\!\!\!\twoheadrightarrow$ fonction partielle surjective
- $\rightarrowtail\!\!\!\!\!\!\twoheadrightarrow$ fonction bijective
- $\rightarrow\!\!\!\!\!\!\rightarrow\!\!\!\!\!\!\rightarrow$ fonction finie
- $\rightarrowtail\!\!\!\!\!\!\rightarrow\!\!\!\!\!\!\rightarrow$ fonction finie injective

Les relations (suite)

La déclaration de fonctions En tant que relations (et donc en tant qu'ensembles), les fonctions profitent de certaines méthodes de construction précédemment énumérées (extension, compréhension), ainsi que d'une autre qui s'ajoute.

- **Extension**
  $\{0 \mapsto 0, 1 \mapsto 1, 2 \mapsto 4, 3 \mapsto 9, 4 \mapsto 16\}$
- **Compréhension**
  $\{x : \mathbb{N} \mid x \leq 4 \bullet x \mapsto x^2\}$
- **Abstraction lambda** Il s'agit essentiellement d'un raccourci d'écriture de la déclaration en compréhension :
  $\lambda\, x : \mathbb{N} \mid x \leq 4 \bullet x^2$

Les relations (suite et fin)

L'application de fonction Soit une fonction f et x, un élément tel que $x \in \mathrm{dom}\, f$. On définira $f(x)$ en **Z** comme l'élément y d'arrivée du couple ordonné tel que $(x, y) \in f$. Ainsi :

$$\{x : \mathbb{N} \bullet x^2 \mapsto x\}(9) = 3$$

La combinaison de fonctions Soit deux ensembles X et Y ainsi que deux fonctions f et g telles que $f : X \pfun Y$ et $g : X \pfun Y$. On définira :

$$f \oplus g == g \cup ((\mathrm{dom}\, g) \ndres f)$$

Conclusion sur le langage mathématique de Z

Nous avons maintenant couvert ce que nous désirions dire du **Z** mathématique. En résumé, nous avons introduit :

- La notion de proposition logique
- Les prédicats, et leur notation en **Z**
- La théorie des ensembles et leur construction en **Z**
- Ceraines extensions **Z** de la théorie des ensembles, principalement concernant la notion de type
- Les relations, notamment les fonctions, et leur utilisation en **Z**

Bien entendu, le **Z** mathématique est nettement plus riche que ce qui vous en fut présenté dans cette introduction. Principalement, nous pensons que les notions de séquence et de typage libre sont importantes pour la rédaction de spécifications réellement élaborées, bien qu'elles dépassent le niveau d'un cours d'introduction. Nous encouragons l'étudiant intéressé à lire les chapitres correspondants de [DW96].

Introduction - langage schématique Z

La base

- Le langage schématique de **Z** est utilisé pour composer et structurer un série de descriptions mathématiques : il permet de grouper ensemble des informations (données ou opérations), de les encapsuler et de les nommer pour réemploi.
- Il s'agit de la seconde composante de la notation **Z**, la première étant le langage mathématique vu précédemment.
- Une spécification formelle est composées de deux choses :
  - Des déclarations globales : noms de types, d'ensembles, abbréviations (équivalences syntaxiques et définitions génériques)
  - Un ou plusieurs (habituellent plusieurs) schémas.

Taxonomie des schémas

Type de schémas Les schémas courants sont habituellement de deux sortes :

- **Orientés données** : ils regroupent ou décrivent alors un ensemble de données. Ils donnent possiblement leur état initial.
- **Orientés opérations** : ils donnent alors des manipulations abstraites des données définies ailleurs, possiblement dans des schémas orientés données.

> Le [langage **Z**] permet donc de décrire, à un niveau très abstrait (c'est-à-dire fortement découplé de toute implémentation logicielle), un produit logiciel sous une perspective formelle alignée sur l'information que celui-ci traite. [...] [En **Z**], tout s'articule en effet autours des données : le langage décrit formellement celles-ci, et la façon dont elles sont manipulées [SP98].

Morphologie des schémas

Les schémas viennent toujours en trois parties :

- **Un nom** qui identifie le schéma en portée globale sur la spécification.
- **Une partie déclarative** (peut être vide) qui définit les différentes données membres du schéma. Les données définies pour la première fois sont de portée locale. Son contenu s'appelle **signature**.
- **Une partie prédicative** (peut être vide) qui énumère les contraintes que doivent respecter les données accessibles dans la portée courante, localement ou globalement, pour que le schéma soit suivit.

Dans le cas de schémas « orientés opérations » , la partie prédicative se termine habituellement par des expressions qui ne sont pas des prédicats, mais plutôt des affectations de certaines données membres (utilisant le symbole $=$). Par le cadre formel, il est important de retenir que chaque donnée ne pourra subir qu'une seule affectation dans un même schéma ; ces affectations permettent de décrire les différentes opérations abstraites effectuée par ces schémas.

Notation des schémas

Notation horizontale

$$Nom\_du\_schema \mathrel{\widehat=} [\text{Declarations} \mid \text{Contraintes et affectations}]$$

Cette notation n'est utilisée que pour des schémas de faible envergure, ou les contraintes seront séparées entre elles par le ET ($\wedge$) logique. On peut sans crainte toujours adopter l'autre.

Notation verticale

Nom_du_schema

Declarations

Contraintes et affectations (une par ligne)

Vision ensembliste des schémas

Liaison [1] Une liaison est une association entre des noms et des valeurs. Les noms sont les identificateurs des membres de la partie déclarative d'un schéma, tandis que les valeurs sont les expressions associées qui respectent les contraintes implicites (typages) et explicites (prédicats) d'un schéma. Imaginons un schéma ayant pour membres deux identificateurs de type entier, x et y, qu'on décide de lier à deux valeurs 1 et 2. Cette liaison l sera : $\langle x \rightsquigarrow 1, y \rightsquigarrow 2 \rangle$. En **Z**, On pourra avoir accès à chacun des membres d'une liaison via un opérateur de sélection, qui nous permettra par exemple d'écrire : $l.x \neq 3$. La construction explicite d'une liaison (voir ci-haut) y est cependant impossible.

Définition ensembliste Un schéma correspond à un ensemble de liaisons. Comme précédemment, si cet ensemble est maximal en extension à l'intérieur d'une spécification, alors on le qualifiera de **type schématique**.

[1] Selon [SP98], c'est cette notion qui présente le plus gros obstacle à la formalisation du **Z** schématique.

Vision ensembliste des schémas (suite et fin)

Liaison caractéristique Soit le schéma $schema$ préalablement défini. On notera $\theta schema$ la liaison caractéristique de ce schéma, c'est-à-dire la liaison telle que chaque identificateur membre de $schema$ est associé avec sa valeur présente dans la portée actuelle. Soit le schéma suivant, servant à décrire la réussite ou l'échec d'une classe d' $[Etudiants]$:

$$
\begin{array}{|l}
\hline
Classe \\
couleurs, passeurs : \mathbb{P}\, Etudiants \\
\hline
couleurs \cap passeurs = \varnothing \\
couleurs \cup passeurs = Etudiants \\
\hline
\end{array}
$$

On pourra désigner la classe dans laquelle tous les étudiants ont réussis par :

$$\mu\, Classe \mid couleurs = \varnothing \bullet \theta Classe$$

Intéraction des schémas

Réutilisation des schémas Tout l'intérêt des schémas est qu'ils peuvent être extensivement réutilisés à l'intérieur d'autres schémas d'une même spécification. Sans cette caractéristique, le langage schématique serait tout bonnement inutile. La réutilisation de schémas par d'autres pourra se faire :

- **Dans une déclaration** : seuls ou après des quantificateurs, dans une construction d'ensemble en compréhension ou dans des expressions mu et lambda.
- **Dans un prédicat** : dans ce cas, seule la contrainte à un sens.
- **Dans une affectation** : sous réserve de sens mathématique.

Intéraction des schémas (suite et fin)

Modes principaux de réutilisation Certaines de ces réutilisations peuvent être relativement subtiles, et nous nous contenterons de donner ici les deux grandes techniques de réutilisation dans les déclarations :

- **Par inclusion schématique** Quand le nom d'un schéma est inclu dans la partie déclarative d'un autre, celui-ci est tout simplement ajouté au schéma courant partie par partie : il faut utiliser ce procédé prudemment car il peut facilement entraîner des incohérences.
- **Par utilisation typée** Rien n'empêche de déclarer des membres d'un schéma comme éléments d'un type schématique préalablement défini dans la même spécification : le membre représente alors une liaison particulière appartenant à ce type schématique.

Aide à la présentation

Décorations schématiques Il est facile, quand on rédige des spécifications formelles, d'atteindre rapidement l'illisibilité. Un mécanisme fournit par le langage schématique de **Z** pour alléger les écritures s'apelle la décoration des membres. La présence de décorations dans un schéma indique la nature « orientée opération » de celui-ci.

Décorations post-fixées Soit n, un membre d'un schéma donné. On notera :

n	La variable dans son état initial
n'	La variable dans son état final (modifiée par affectation)
$n?$	La variable, paramètre d'entrée du schéma, dans son état initial
$n!$	La variable, paramètre de sortie du schéma, dans son état final

Aide à la présentation (suite et fin)

Décorations pré-fixées Soit n, un membre d'un schéma donné OU un schéma en inclusion dans un autre. On notera, dans sa partie déclarative seulement :

Δn **Notation Delta**
Pour les membres, il s'agit d'un raccourci pour déclarer n et n'.
Pour les schéma en inclusion, cela signale que le schéma actuel modifie l'ancien : tous les membres n et n' de cet ancien schéma sont inclus.

Ξn **Notation Xi**
Cela signale que le membre ou le schéma en inclusion n
ne sera pas modifié.

Introduction aux schémas génériques

Redénomination À l'instart des prédicats, un schéma **Z** peut subir la substitution de variables. Soit un schéma $schema$ ayant x pour membre. Il nous sera possible de construire un nouveau schéma strictement identique pour lequel y remplacera x en écrivant : $schema[y/x]$. On nomme cette opération redénomination car le nom du nouveau schéma sera justement $schema[x/y]$.

Schémas génériques Une façon plus transparente et élégante de gérer la redénomination est la rédaction de schémas génériques. De tels schémas sont identiques à des schémas standards, à ceci près que leur nom est complété (ou remplacé) par une séquence d'identificateurs entre crochets carrés, et qu'ils débutent sur un trait double plutôt que simple.

Introduction aux schémas génériques (suite et fin)

Définitions schématiques génériques Les schémas génériques non nommés sont spécialement intéressants car ils permettent de définir de nouveaux opérateurs en **Z**, et ils offrent ainsi une alternative puissante au processus d'abbréviation déjà présenté. Ils utilisent, en plus des caractéristiques déjà énumérées des schémas génériques, une notation déclarative particulière utilisant la barre de soulignement que nous nous contenterons d'illustrer [2] :

$$[X, Y]$$

$$\_ \lhd \_ : \mathbb{P}\, X \times (X \leftrightarrow Y) \longrightarrow (X \leftrightarrow Y)$$

$$\forall S : \mathbb{P}\, X; R : X \leftrightarrow Y \bullet$$
$$S \lhd R = \{x : X; y : Y \mid ((x \in S) \land (x \mapsto y \in R)) \bullet x \mapsto y\}$$

[2] Tous les opérateurs spécifiques au **Z** mathématique sont d'ailleurs définis ainsi. Voir [SP98], chap. 5.

Conclusion sur le langage schématique de Z

Les notions importantes du **Z** schématique ont maintenant été couvertes. En résumé, nous avons introduit :

- Le rôle des schémas dans une spécification formelle
- La classification fondamentale des schémas en schémas orientés données ou opérations.
- La structure et la notation des schémas en **Z**.
- La signification intuitive des schémas en terme de théorie des ensembles
- Les méthodes d'intéraction des schémas entre eux
- Les décorations
- Les schémas génériques

Une couverture plus étendue du langage schématique devrait tout d'abord s'attacher à traiter des opérateurs de schémas, qui permettent de combiner entre eux des schémas de façon très puissante ; l'étudiant intéressé est encouragé de lire à ce sujet le chapitre 12 de [DW96].

Exemple de spécification formelle

Bibliothèque de Polytechnique Comme base de notre spécification, définissons le schéma $Bibliotheque$, basé sur les types $[Livre, Etudiant]$:

$$
\begin{array}{|l}
\hline
Bibliotheque \\
\hline
livres : \mathbb{P}\, Livre \\
emprunteurs : \mathbb{P}\, Etudiant \\
livres\_empruntes, livres\_disponibles : \mathbb{P}\, Livre \\
emprunts : Livre \pfun Etudiant \\
\hline
livres\_empruntes = \mathrm{dom}\, emprunts \\
\mathrm{ran}\, emprunts \subseteq emprunteurs \\
livres\_empruntes \cup livres\_disponibles = livres \\
livres\_empruntes \cap livres\_disponibles = \varnothing \\
\hline
\end{array}
$$

Exemple de spécification formelle (suite)

Initialisation Lors d'une première utilisation du système, il est important de définir un état initial abstrait :

$$
\begin{array}{|l}
\hline
Initialiser\_bibliotheque \\
Bibliotheque' \\
\hline
livres' = \varnothing \\
emprunteurs' = \varnothing \\
\hline
\end{array}
$$

Exemple de spécification formelle (suite)

Ajouter des livres

$$
\begin{array}{|l}
\hline
Ajouter\_livre \\
\Delta Bibliotheque \\
nouveau\_livre? : Livre \\
\hline
nouveau\_livre? \notin livres \\
livres\_empruntes' = livres\_empruntes \\
livres\_disponibles' = livres\_disponibles \cup \{nouveau\_livre?\} \\
emprunteurs' = emprunteurs \\
emprunts' = emprunts \\
\hline
\end{array}
$$

Exemple de spécification formelle (suite)

Disponibilité et emprunt d'un livre

$$Resultat ::= livre\_empruntable \mid livre\_non\_empruntable$$

$$\begin{array}{|l}
\hline
Livre_non_disponible \\
\Xi Bibliotheque \\
livre? : Livre \\
resultat! : Resultat \\
\hline
livre? \in livres_empruntes \\
resultat! = livre_non_empruntable \\
\hline
\end{array}$$

Exemple de spécification formelle (suite)

Emprunt

$$\Delta Bibliotheque$$
$$livre? : Livre$$
$$emprunteur? : Etudiant$$
$$resultat! : Resultat$$

$$livre? \in livres\_disponibles$$
$$emprunteur? \in emprunteurs$$
$$livres\_empruntes' = livres\_empruntes \cup \{livre?\}$$
$$livres\_disponibles' = livres\_disponibles \setminus \{livre?\}$$
$$emprunteurs' = emprunteurs$$
$$emprunts' = emprunts \cup \{livre? \mapsto emprunteur\}$$
$$resultat! = livre\_empruntable$$

Exemple de spécification formelle (suite et fin)

« Sécurisation » du schéma d'emprunt

À partir des schémas $Livre\_non\_disponible$ et $Emprunt$ précédemment définis, on peut aisément en écrire un autre, toujours respecté celui-là :

$$Tentative\_emprunt \mathrel{\widehat=} Livre\_non\_disponible \lor Emprunt$$

En conclusion Évidemment, cette spécification formelle est clairement insuffisante pour gérer une bibliothèque. Un certain nombre de schémas importants y ont été omis : dans son état actuel, les seules opérations possibles sont d'y ajouter des livres ou d'en emprunter ; il est impossible de gérer les retours ou même la liste des emprunteurs authorisés. Le but de cet exemple était simplement de vous donner une illustration simple de l'utilisation de **Z** dans un contexte concret : libre à quiconque de le compléter comme bon lui plaira.

Quelques questions et exercices

Retour sur la spécification formelle précédente

- Dans le schéma $Bibliotheque$, quel impact aurait la déclaration de remplacement $emprunts : Livre \rightarrowtail\!\!\!\!\!\!\rightarrow Etudiant$ sur les modalités d'emprunt des étudiants ?
- La déclaration $emprunts : Livre \twoheadrightarrow Etudiant$ rendrait la notion d'emprunteurs inutile. Expliquez pourquoi.
- On désire modifier le schéma $Bibliotheque$ pour qu'il soit impossible d'en emprunter plus de la moitié des livres. Ajoutez le prédicat nécessaire (la solution n'est pas unique).
- En quoi cette dernière modification affecte-t-elle le schéma $Tentative\_emprunt$? Comment remédier à cette situation ?

Quelques questions et exercices (suite)

- Le schéma $Ajouter\_livre$ ne fait aucune affectation de $livres'$. Pourquoi n'est-ce pas une erreur ?
- On désire restreindre le nombre de livres qu'un étudiant peut emprunter à quatre. Par l'ajout d'une seule ligne, apportez la modification nécessaire au schéma $Emprunt$.
- Expliquez l'effet du schéma $Inconnu$ suivant :

$$
\begin{array}{|l}
\hline
Inconnu \\
Bibliotheque \\
emprunteurs' : \mathbb{P}\ Etudiant \\
groupe\_etudiants? : \mathbb{P}\ Etudiant \\
\hline
emprunteurs' = emprunteurs \setminus groupe\_etudiants? \\
\hline
\end{array}
$$

Quelques questions et exercices (suite et fin)

- Bien que cela soit possible, pourquoi n'a-t-on pas utilisé la notation Delta pour la rédaction du schéma précédent ?
- Modifiez le schéma $Inconnu$ pour ne supprimer de l'ensemble des emprunteurs potentiels que les étudiants qui n'ont aucun livre en emprunt (possible sans rajouter de lignes).
- Créez le schéma $Supprimer\_livre\_perdu$ qui permettra d'éliminer de la bibliothèque un livre emprunté qui ne sera jamais rendu. Tous les membres de $Bibliotheque$ devront être transformés de façon cohérente.

■

Références

[SP98] Spivey, John M. *The Z notation : a reference manual*, Prentice Hall, 1998

[WW96] Wordsworth, John B. *Software Development with Z*, Addison-Wesley, 1996

[DW96] Davies, Jim , Jim Woodcock. *Using Z*, Prentice Hall, 1996