DESS de Génie Informatique
Faculté des sciences et des techniques

Module D2 - Notes de cours

Christian Attiogbé

Introduction au formalisme de spécification Z

Christian Attiogbé

attiogbe@irin.univ-nantes.fr

Octobre 1996

Module D2
DESS Génie Informatique
Université de Nantes

Table des matières

1 Démarche de construction de spécifications en Z

Z est une notation et pas une méthode. Il y a cependant des démarches, des conventions et des pratiques établies par l'utilisation de la notation. Une des démarches communément admises et largement utilisée est celle présentée dans le livre de Barden, Stepney et Cooper [BSC94]. Nous reprenons ici cette démarche avec quelques commentaires.

1.1 Principaux constituants d'une spécification Z

La specification type contient les éléments suivants :

- la **déclaration des ensembles de base** ou types de base (given sets) et des **constantes** pour la spécification. Un texte informel doit accompagner chaque déclaration pour "préciser" sa signification.
- un **schéma** qui décrit l'**état abstrait** du système spécifié. Ce schéma peut être la combinaison de plusieurs autres schémas élaborés séparément. La combinaison des schémas est faite avec le *calcul de schémas.*
- un schéma qui décrit l'**état initial** du système. La preuve de l'existence d'un tel état initial doit être également donné.
- des schémas décrivant les **opérations abstraites** sur l'état. Ces schémas peuvent être donnés dans un premier temps sans tenir compte du traitement des cas d'erreur (qui seront pris en compte plus tard). Ces opérations utilisent des préconditions.
- les descriptions des préconditions apparaissant dans les opérations partielles ci-dessus.
- les schémas qui traitent les cas d'erreur.
- un résumé des opérations introduites par les schémas décrivant les opérations abstraites et les traitements d'erreur
- une fiche d'information à l'intention du lecteur de la spécification. Un sorte de carte ou graphe des schémas.

1.2 Construction des spécifications

Analyse préliminaire

Le cahier de charges du système à spécifier est soigneusement analysé afin d'identifier les principales données du système. Ces données sont vues comme des ensembles et des constantes.

Introduction des ensembles de base et des constantes

Il s'agit de la mise en forme (modélisation ou formalisation) du résultat de la phase d'analyse qui précède. Ce travail constitue les premiers éléments apparaissant dans la spécification Z. Il y a une déclaration des ensembles de base (ils seront utilisés comme types de base), une déclaration des variables et constantes communes aux différentes parties de la spécification et les textes explicatifs dans chaque cas.

Théorie orientée application

On a souvent besoin d'une théorie spécifique lorsqu'on écrit des spécifications. On l'introduit après la partie des déclarations.

Description de l'état abstrait et invariant

On élabore le schéma de l'état abstrait du système. Ce schéma doit être concis et ne doit inclure que les déclarations nécessaires à la construction du modèle de la spécification.
Le schéma abstrait a la forme suivante :

$$\begin{array}{|l}
\hline \mathit{EtatAbstrait} \\
\mathit{declarations\ pour\ l'etatabstrait} \\
\hline
\mathit{predicat\ pour\ l'etat\ abstrait} \\
\hline
\end{array}$$

La partie prédicat de l'état abstrait représente l'**invariant** de l'état du système. C'est un prédicat dont l'interprétation doit être vraie dans tous les états du système.

Un **texte informel** doit accompagner le schéma pour le commenter. Il en est de même pour tous les schémas ou autres composantes de la spécification. On parle aussi de la partie informelle de la spécification.
La partie formelle est constituée de l'ensemble des **textes formels**, ce sont les composantes de la spécification formelle (déclarations et schémas).

Description de l'état initial

On construit un schéma qui décrit l'état initial du système puis on fournit la preuve de l'existence de cet état. Cette construction peut être vue comme une opération dont le résultat est l'état initial.

$$\begin{array}{|l}
\hline \mathit{EtatAbstraitInitial} \\
\mathit{EtatAbstrait}' \\
\hline
\mathit{predicat\ de\ l'etatinitial} \\
\hline
\end{array}$$

La preuve à faire (**obligation de preuve**) est celle du théorème suivant :

$$\vdash \exists \mathit{EtatAbstrait}' \bullet \mathit{EtatAbstraitInitial}$$

Typiquement, le prédicat de l'état abstrait donne les affectations de valeurs aux variables. Dans ce cas la preuve se réduit à démontrer que ces affectations satisfont les propriétés de l'état abstrait.

Spécification des opérations

Chaque opération est spécifiée sans traitement des cas d'erreurs. Ce sont des opérations partielles qui utilisent des schémas de la forme suivante :

OperationAbstraite
$\Delta EtatAbstrait$ *Entrees* *Sorties*
$predicat\ pour\ l'operation\ abstraite$

Le prédicat relie les *Entrees* et les *Sorties*, l'état initial et l'état final.
Le symbole Δ indique le schéma utilisé par l'opération. Nous le verrons en détail plus loin.

Les préconditions

Les préconditions des opérations partielles sont calculées. Le schéma de précondition d'une opération est formée en cachant toute les variables de la sortie et de l'état final. Pour ce faire on utilise le quantificateur existenciel ($\exists$). Le schéma de précondition d'une opération abstraite a la forme suivante :

PreOperationAbstraite
EtatAbstrait *Entrees*
$\exists\ EtatAbstrait'; Sorties \bullet$ $\quad predicat\ pour\ l'operation\ abstraite$

La partie prédicat est simplifiée. On teste la description de l'opération abstraite pour s'assurer que sa précondition est explicite dans le prédicat de l'opération. Si ce n'est pas le cas la définition de l'opération est modifiée. Dans le but de préparer le traitement des cas d'erreur, il est important de résumer les préconditions des opérations partielles ainsi testées.

Construction des schémas pour le traitement des erreurs

Pour la robustesse des opérations du système spécifié, on souhaite que ces opérations puissent gérer n'importe quelle entrée et donner en conséquence une sortie. Les préconditions indiquent où commencer la description des erreurs. Pour s'assurer qu'elles sont bien traitées on sollicite l'utilisateur (le commanditaire) pour la validation.

Par exemple, si la précondition d'une opération est une conjonction de deux prédicats

$$p\ \wedge\ q$$

il y a trois conditions d'erreur possibles :

$$\neg\, p\ \wedge\ q$$

$$p\ \wedge\ \neg\, q$$

$$\neg\, p\ \wedge\ \neg\, q$$

Il est préférable de décomposer en trois cas plutôt que de donner le cas global $\neg(p\ \wedge\ q)$. En effet les traitements ou messages d'erreur sont plus clairs et le comportement du système en présence d'erreur est ainsi mieux défini.

Totalisation des opérations

Les opérations partielles qui décrivent les cas à succès et les cas d'erreurs sont combinées pour avoir une description totale. On utilise pour ce faire la composition (souvent des disjonctions) de schémas. L'opération totale est alors de la forme :

$$\begin{aligned} OperationAbstraiteTotale \;\widehat{=}\; & OperationAbstraite \\ & \lor\ Erreur_1 \\ & \lor\ Erreur_2 \\ & \lor\ \ldots \end{aligned}$$

L'utilisation de la disjonction peut entrainer le non-déterminisme. Si plusieurs erreurs se produisent simultanément, il y a plusieurs conditions d'erreurs vérifiées, l'opération a alors le choix entre plusieurs conditions.
Cependant, ce non-déterminisme peut être exploité au moment de l'implantation car on a le choix entre plusieurs options. Il est néanmoins nécessaire de documenter les options prises.

On fournit enfin une table (c'est une forme de mode d'emploi ou de contrat) donnant pour chaque opération partielle ses entrées et sa précondition.

Opération	**Entrées**	**Preconditions**
OpérationAbstraite1	Entrées1	Prédicat de PreOpérationAbstraite1
OpérationAbstraite2	Entrées2	Prédicat de PreOpérationAbstraite2
OpérationAbstraite3	Entrées3	Predicat de PreOpérationAbstraite3
...	...	...

Aide aux lecteurs de la spécification

A la fin de la construction de la spécification d'un système, on peut ajouter des informations complémentaires dans le but d'assister le lecteur des spécifications obtenues. On peut commenter les (noms de) schémas et les variables et montrer la dépendance entre les différents schémas.
Pour les spécifications très longues on peut fournir un résumé. Tout ceci rend la lecture de la spécification plus simple aux lecteurs.

1.3 Discussion

Une phase de relecture de la spécification par un "expert" de Z (n'ayant pas participé à son élaboration) est nécessaire pour "valider" la spécification.
Cet expert a pour rôle d'indiquer les défauts de la spécification ou au contraire de dire qu'elle est "correcte".

De manière générale, il s'agit de vérifier que la spécification répond au cahier de charges. Pour cela l'expert doit avant tout bien comprendre le cahier de charges.

Pour améliorer la spécification, on se pose les questions suivantes :

Sur le texte informel

- est-ce que le vocabulaire technique utilisé dans le commentaire des spécifications correspond à celui qui est utilisé dans le domaine du système spécifié ?
- est-ce que les figures,les listages et les diagrammes sont utilisés à bon escient ?

- est-ce que le texte informel est facile à lire?

Sur la relation entre texte informel et texte formel

- est-ce qu'il y a suffisament (ou trop) de textes informels pour les spécifications construites?
- est-ce que le vocabulaire utilisé dans le texte informel correspond aux noms ou identificateurs utilisés dans le texte formel?
- est-ce que le texte informel donne de bons exemples d'utilisation de la spécification (les interfaces des opérations).

La cohérence doit aller dans l'autre sens aussi c'est à dire entre le texte formel et le texte informel.

Sur le texte formel

On doit utiliser des outils logiciels pour s'assurer que la spécification ne contient ni d'erreurs syntaxiques ni d'erreurs de typage.

- est-ce que le modèle est approprié?
  - est-ce que le nombre d'états est réduit?
  - est-ce que le calcul de schémas est bien utilisé pour présenter le modèle?
  - est-ce que les schémas ont des prédicats courts, clairements exprimés et en nombre réduit?
  - est-ce que les opérations sont facilement comprises?
  - est-ce que tous les invariants des états sont bien documentés?
  - est-ce que les opérations sont totales, sinon est-ce que les préconditions sont bien documentées?
  - est-ce que les preuves sont simples? est-ce qu'elles procurent de la confiance?
- est-ce que le texte formel prend en compte tous les aspects des opérations?
- est-ce que le texte formel surspécifie les besoins exprimés dans le cahier de charge en prenant des décisions non justifiées?

1.4 Conclusion

La démarche présentée est une façon de construire des spécifications Z bien structurées en suivant différentes étapes. Elle donne l'ordre utilisé pour présenter la spécification finale et l'ordre dans lequel la spécification est élaborée. On peut la voir comme un processus de spécification. Des itérations ou des retours sur certaines étapes sont nécessaires lorsqu'on en éprouve le besoin.

2 La notation Z

2.1 Les déclarations : types, variables et constantes

Les types de base (ou ensembles de base) d'une spécification sont déclarés par simple listage entre crochets. Par convention, on écrit les noms de ces types en majuscule.

Exemple :

$$[PERSONNE, ETUDIANT, MACHINE, DATE]$$

déclare que $PERSONNE, ETUDIANT, MACHINE, DATE$ sont des types de base (issus de l'analyse d'un problème donné).

Une déclaration est de la forme :

$x : T$ où T est un type ou
$x : S$ où S est un sous-ensemble d'un type.

On peut ajouter des contraintes en faisant suivre la(les) déclaration(s) par une partie $\mid$ *Predicat* qui exprime des relations entre les variables déclarées.
Exemples :

$$m, n : \mathbb{N}$$
$$i, j : 1..8 \mid i \neq j$$
$$e1, e2 : S; s : \mathbb{P}S \mid e1 \in s \wedge e2 \notin s$$

On peut définir des types de base en énumérant leurs éléments.
Exemples :

$$COULEUR ::= bleu \mid rouge \mid jaune \mid vert$$
$$POSITION ::= on \mid off$$
$$REPONSE ::= Ok \mid PasOk$$

2.2 Construction d'ensembles et de types

On peut construire de nouveaux ensembles en utilisant :

l'ensemble des parties noté $\mathbb{P}$

ou le produit cartésien noté $\times$.
Lorsqu'on applique $\mathbb{P}$ et $\times$ aux types, on obtient de nouveaux types.
Toute variable ou expression de Z a un unique type. Cependant il peut appartenir à différents ensembles.

2.3 Les définitions axiomatiques ou schémas ouverts

Elles servent à introduire des déclarations globales dans la spécification. Elles ont la forme :

$$\begin{array}{|l} Declaration \\ \hline Predicat \end{array}$$

Exemples

$$\begin{array}{|l} maxinscrits : \mathbb{N} \end{array}$$

$$\begin{array}{|l} age : \mathbb{N} \\ etudiants : \mathbb{P} ETUDIANT \\ \hline \# etudiants \leq maxinscrits \\ 0 \leq age \leq 120 \end{array}$$

$$\begin{array}{|l} stockinitial : Stock \\ \hline \# stockinitial \leq seuil \\ \qquad \forall p : PRODUIT, n : \mathbb{N} \mid (p, n) \in stockinitial \bullet n \leq 1000 \end{array}$$

2.4 Abbréviations et constantes

Une abbréviation permet d'introduire une constante globale dans la spécification. Elle a la forme :

$$Nom == Expression$$

Exemples

$$\begin{array}{l} Stock == \{ s : \mathbb{P}(PRODUIT \times \mathbb{N}) \mid \\ \qquad \forall p : PRODUIT; m, n : \mathbb{N} \bullet (i, m) \in s \wedge (i, n) \in s \Rightarrow m = n \} \end{array}$$

$$\mathbb{P}_1 X == \mathbb{P} X \setminus \emptyset$$

2.5 Les schémas

Le schéma est un des principaux éléments de structuration des spécifications Z.
Les schémas permettent de décrir des états du système spécifié ou des opérations. Lorsqu'il est utilisé pour les états, un schéma définit un espace d'états.
Les schémas ont

la forme textuelle (horizontale) :

$$NomSchema \mathrel{\widehat{=}} [Declaration \mid Predicat]$$

ou la forme bidimensionnelle (verticale) :

$$
\begin{array}{|l}
\hline
\textit{NomSchema} \\
\textit{Declaration} \\
\hline
\textit{Predicat} \\
\hline
\end{array}
$$

Chaque schéma a un nom par lequel on le désigne et l'utilise.
Les schémas permettent de structurer convenablement les spécifications en faisant référence à un ou plusieurs autres schémas, déjà définis, dans un schéma en cours d'élaboration.
Il existe plusieurs opérateurs permettant de référencer des schémas ou de composer des schémas, on les verra plus loin dans les sections consacrées à l'utilisation des schémas 4.6 et au calcul de schémas 4.7.

2.5.1 Les paramètres

Une opération abstraite peut utiliser des paramètres d'entrée et de sortie. Par convention un paramètre d'entrée (ou variable d'entrée) est désigné par un identificateur alphanumérique terminé par le caractère ? et un paramètre de sortie est désigné par un identificateur alphanumérique qui se termine par le caractère !.

2.5.2 Exemple

On veut spécifier une opération qui additionne une valeur donnée en entrée avec une valeur de travail et qui retourne le résultat dans un paramètre donné.

$$
\begin{array}{|l}
valtrav : \mathbb{Z} \\
\hline
valtrav >= 0
\end{array}
$$

Nous avons déclaré une variable commune (globale à la spécification) qui est un entier positif ou nul.

$$
\begin{array}{|l}
\hline
\textit{Additionner} \\
x? : \mathbb{N} \\
res! : \mathbb{N} \\
\hline
res! = valtrav + x? \\
\hline
\end{array}
$$

Ce schéma définit une opération qui additionne le paramètre d'entrée ($x?$) à une variable commune ($valtrav$) et fournit le résultat dans un paramètre de sortie ($res!$).

3 Exemples de spécification Z

3.1 La classique histoire d'annuaire

Cahier de charges

On veut un système de gestion d'un annuaire. Dans l'annuaire nous avons des noms de personnes. Certaines de ces personnes ont un ou plusieurs numéros de télephone. Les numéros des personnes ne sont pas systématiquement notés, on fini par les noter quand on les a. Il y de temps en temps des changements de numéro de téléphone. Il arrive aussi qu'on efface le nom d'une personne de l'annuaire ...

Analyse des besoins

Nous avons des personnes et des numéros. Nous avons aussi des personnes qui ont des numéros et d'autres qui n'en ont pas. Il y donc une relation entre les personnes membres de l'annuaire et les numéros.
Nous devons réaliser les opérations de gestion de l'annuaire : ajout d'une nouvelle personne, ajout d'un numéro de téléphone, consultation d'un numéro de téléphone, modification d'un numéro, etc.

3.1.1 Déclarations des types de base

$$[PERSONNE, NO\_TELEPHONE]$$

$PERSONNE$ est le type de base qui va caractériser les personnes utilisées dans la spécification. $NO\_TELEPHONE$ est le type de base des numeros de teléphone utilisés dans la spécification.

$$\begin{aligned} MESSAGE ::= {} & NONMEMBRE \mid ESTMEMBRE \mid DEJACONNU \\ & \mid OPERATIONOK \mid PASDANS\_ANNUAIRE \mid NUMERO\_INCONNU \end{aligned}$$

$MESSAGE$ est un type libre. Il représente le type de tous les messages utilisés dans la spécification des opérations.

3.1.2 Construction de l'espace d'états du système

$$\begin{array}{|l} \hline \textit{Annuaire} \\ telephones : PERSONNE \leftrightarrow NO\_TELEPHONE \\ membres : \mathbb{P}\, PERSONNE \\ \hline dom\ telephones \subseteq membres \\ \hline \end{array}$$

Notre système d'annuaire est modélisé par une relation entre des personnes et des numéros de téléphone. Dans notre annuaire, il y a un ensemble de personnes, nous l'appelons $membres$.
La partie prédicat du schéma constitue un invariant qui indique que seules les personnes membres de l'annuaire ont un numéro de téléphone.

3.1.3 Construction des opérations abstraites

Ajout d'un nouveau membre

$$\begin{array}{|l} \hline \textit{NouveauMembre} \\ \Delta Annuaire \\ pers? : PERSONNE \\ \hline pers? \notin membres \\ membres' = membres \cup \{pers?\} \\ \hline \end{array}$$

Le symbole Δ permet de faire référence à un schéma que l'on veut modifier.

$NouveauMembre$ est une opération abstraite partielle qui permet d'ajouter une personne dans notre annuaire s'il elle n'y était pas (c'est la condition de l'opération).
Dans la partie prédicat (formellement c'est une conjonction des prédicats) du schéma nous avons deux lignes de prédicat séparées par une ligne vide. C'est une façon de faire en Z : la première ligne représente une condition et l'autre ligne (ou les autres) représente(nt) le prédicat caractérisant l'opération spécifiée.

Enregistrement du numèro de téléphone d'un membre

$$
\begin{array}{|l}
\hline
\textit{EnregNumero} \\
\Delta Annuaire \\
pers? : PERSONNE \\
num? : NO\_TELEPHONE \\
\hline
pers? \in membres \land \\
\qquad \neg(\{pers? \mapsto num?\} \subseteq telephones) \\
telephones' = telephones \cup \{pers? \mapsto num?\} \\
\hline
\end{array}
$$

EnregNumero est une opération abstraite partielle qui permet d'enregistrer un numéro de téléphone donné pour un membre dans le cas où ce dernier n'avait pas déjà ce numéro.

Traitement des cas d'erreur

$$
\begin{array}{|l}
\hline
\textit{PasMembre} \\
\Xi Annuaire \\
pers? : PERSONNE \\
mess! : MESSAGE \\
\hline
pers? \notin membres \\
mess! = NONMEMBRE \\
\hline
\end{array}
$$

Ce schéma permet d'avoir un message approprié lorsqu'une personne donnée n'est pas membre de l'annuaire.
Le symbole Ξ permet d'utiliser un schéma sans le modifier.

$$
\begin{array}{|l}
\hline
\textit{DejaEnregistre} \\
\Xi Annuaire \\
pers? : PERSONNE \\
num? : NO\_TELEPHONE \\
mess! : MESSAGE \\
\hline
\{pers? \mapsto num?\} \subseteq telephones \\
mess! = DEJACONNU \\
\hline
\end{array}
$$

Cette opération permet d'avoir le message $DEJACONNU$ losqu'on lui donne une personne et un numéro qui font déja partie de la relation.

$$
\begin{array}{|l}
\hline
\textit{CestOk} \\
mess! : MESSAGE \\
\hline
mess! = OPERATION\_OK \\
\hline
\end{array}
$$

Cette opération permet d'obtenir le message $CestOK$ lorsqu'une opération s'est bien déroulée.

Totalisation de l'opération d'enregistrement de numéro

$$
\begin{aligned}
AjoutNumeroT \mathrel{\widehat=} & (AjoutNumero \land CestOk) \\
& \lor PasMembre \\
& \lor DejaEnregistre
\end{aligned}
$$

Nous avons utilié là le calcul de schémas. C'est la composition de schéma à l'aide des opérateurs de schémas que sont $\wedge$ et $\vee$.

Par le calcul de schémas nous avons défini l'opération totale $AjoutNumeroT$. C'est un schéma dans lequel les cas des entrées erronées pour l'ajout de numéro sont traités.

Recherche des numéros de télephone d'un membre de l'annuaire

$$
\begin{array}{|l}
\hline
\textit{NumeroTelephoneDe} \\
\Xi Annuaire \\
pers? : PERSONNE \\
num! : \mathbb{F} NO\_TELEPHONE \\
\hline
pers? \in dom\ telephones \\
num! = telephones (\!| \{pers?\} |\!) \\
\hline
\end{array}
$$

Cette opération abstraite partielle donne l'ensemble des numéros de télephone d'un membre lorsqu'il en a (c'est ce qu'exprime la condition).
Nous avons utilisé ici l'image relationnelle notée avec $(\!|\ \ |\!)$.
$\mathbb{F}$ indique un ensemble fini.

Traitement du cas d'erreur où une personne n'a pas de numéro

$$
\begin{array}{|l}
\hline
\textit{PasDansAnnuaire} \\
\Xi Annuaire \\
pers? : PERSONNE \\
mess! : MESSAGE \\
\hline
pers? \notin dom\ telephones \\
mess! = PASDANS\_ANNUAIRE \\
\hline
\end{array}
$$

Ce schéma permet d'avoir le message approprié quand une personne n'a pas de numéro de téléphone (mais elle peut très bien être membre de l'annuaire).

Totalisation de l'opération de recherche de numéro

$$
\begin{aligned}
NumTelephoneDeT \mathrel{\widehat{=}} & NumTelephoneDe \\
& \vee PasDansAnnuaire
\end{aligned}
$$

Ce nouveau schéma obtenu par composition des schémas déjà construits nous permet de traiter toutes les entrées (erronées et non erronées) de l'opération de recherche de numéros de téléphones.

Recherche de la (des) personne(s) ayant un numéro donné

$$
\begin{array}{|l}
\hline
\textit{QuiANumero} \\
\Xi Annuaire \\
num? : NO\_TELEPHONE \\
pers! : \mathbb{F} PERSONNE \\
\hline
num? \in \operatorname{ran} telephones \\
pers! = telephones^{\sim} (\!| \{num?\} |\!) \\
\hline
\end{array}
$$

Cette opération abstraite partielle permet d'obtenir l'ensemble des personnes qui ont le numéro donné comme paramètre.
Nous avons utilisé l'image relationnelle inverse notée $\sim(\!|\ \ |\!)$

Traitement du cas de numéro inconnu

$$
\begin{array}{|l}
\hline
\textit{NumInconnu} \\
\Xi Annuaire \\
num? : NO\_TELEPHONE \\
mess! : MESSAGE \\
\hline
num? \notin \mathrm{ran}\ telephones \\
mess! = NUMERO\_INCONNU \\
\hline
\end{array}
$$

Ce schéma définit une opération qui permet d'avoir le message approprié losqu'un numéro donné n'est pas dans l'annuaire.

Suppression d'une coordonnée d'une personne de l'annuaire

$$
\begin{array}{|l}
\hline
\textit{SupprimerPersNumero} \\
\Delta Annuaire \\
pers? : PERSONNE \\
num? : NO\_TELEPHONE \\
\hline
\{pers? \mapsto num?\} \subseteq telephones \\
telephones' = telephones \setminus \{pers? \mapsto num?\} \\
\hline
\end{array}
$$

Ce schéma définit une opération partielle qui supprime un couple $(personne, numero)$ lorsqu'il est déjà (c'est notre condition) enregistré dans l'annuaire.

Suppression d'un membre de l'annuaire

$$
\begin{array}{|l}
\hline
\textit{SupprimerMembreAnnuaire} \\
\Delta Annuaire \\
pers? : PERSONNE \\
\hline
pers? \in membres \\
membres' = membres \setminus \{pers?\} \\
telephones' = \{pers?\} \lhd telephones \\
\hline
\end{array}
$$

Nous enlevons un membre de l'annuaire. Si ce membre avait des numéros de téléphone nous les enlevons tous. En effet l'invariant du système veut que seuls les membres aient des numéros de téléphone.

Cette opération abstraite partielle permet d'enlever de l'annuaire toutes les coordonnées de la personne donnée. L'antirestriction de domaine ($\lhd$) est ici l'opération idéale. Comme condition nous voulons que la personne soit membre de l'annuaire.

Suppression du numéro de télephone d'un membre

$$
\begin{array}{|l}
\hline
\textit{SupprimerNumero} \\
\Delta Annuaire \\
num? : NO\_TELEPHONE \\
\hline
num? \in \operatorname{ran} telephones \\
telephones' = telephones \rhd \{num?\} \\
\hline
\end{array}
$$

Cette opération partielle permet d'enlever un numéro de téléphone donné de l'annuaire. Plusieurs personnes peuvent avoir ce numéro, la mise à jour peut concerner plusieurs couples ($pesronne$, $numero$) de l'annuaire. Des personnes peuvent se trouver ainsi sans numéro après l'opération. Dans ce cas elles ne doivent plus être dans la relation *telephones* et il faut les enlever. Cependant elles restent membres de l'annuaire.

L'antirestriction de codomaine ($\rhd$)est l'opération idéale pour cette spécification.

Changement du numéro de téléphone d'un membre

$$
\begin{array}{|l}
\hline
\textit{ChangerNumeroAnnuaire} \\
\Delta Annuaire \\
pers? : PERSONNE \\
num? : NO\_TELEPHONE \\
\hline
pers? \in \operatorname{dom}\ telephones \\
telephones' = telephones \oplus \{pers? \mapsto num?\} \\
\hline
\end{array}
$$

3.1.4 Construction de létat initial

$$
\begin{array}{|l}
\hline
\textit{AnnuaireInit} \\
Annuaire' \\
\hline
telephones = \emptyset \\
membres = \{jean, pierre, paul\} \\
\hline
\end{array}
$$

3.2 Le système de fichier UNIX

(Exercices)

4 Compléments à la notation Z

4.1 Opérations sur les relations et les fonctions

Nous avons déjà vu les opérations d'union ($\cup$), d'intersection ($\cap$), de soustraction ($\setminus$) sur les ensembles dans le chapitre 1. Toutes ces opérations sont utilisées sur les relations (ce sont des ensembles).

4.2 Exemples d'opérations ensemblistes

$\{pers? \mapsto num?\} \subseteq telephones$

$telephones' = telephones \cup \{pers? \mapsto num?\}$

Il y a d'autres opérations qui sont spécifiques aux relations : ce sont les opérations de restriction.

4.3 Les opérations de restriction et d'anti-restriction

La restriction du domaine

Soit une relation binaire *conduit* définie sur un ensemble de personnes et un ensemble de moyens de locomotion. C'est un ensemble de couples de la forme :
$\{(Patrice, velo), (Alice, voiture), (Patrice, patins), (Yves, velo), (Florence, voiture)\}$
Le domaine d'une relation peut être restreint à un sous-ensemble donné. La restriction du domaine de la relation au sous-ensemble $\{Patrice\}$ donne le sous-ensemble de la relation *conduit* qui ne contient que les couples dont le premier élément est $Patrice$. C'est le sous-ensemble :

$\{(Patrice, velo), (Patrice, patins)\}$

L'opération de restriction de domaine est noté par le symbole $\lhd$.

L'anti-restriction du domaine

L'anti-restriction du domaine d'une relation r par un sous-ensemble donné E donne un sous-ensemble de r qui est constitué des couples de r autres que ceux dont le premier élément appartient à E.
$conduit \ntriangleleft \{Patrice\} = \{(Alice, voiture), (Yves, velo), (Florence, voiture)\}$

La restriction de l'image

Elle est notée par le symbole $\rhd$. La restriction de l'image d'une relation r à un sous ensemble E permet d'avoir un sous-ensemble de la relation constitué des seuls couples dont le second élément appartient à E.
$conduit \rhd \{velo\} = \{(Patrice, velo), (Yves, velo)\}$

L'antirestriction de l'image

L'anti-restriction de l'image d'une relation r par un sous-ensemble donné E donne un sous-ensemble de r qui est constitué des couples de r autres que ceux dont le second élément appartient à E.

$conduit \rhd \{velo\} = \{(Alice, voiture), (Patrice, patins), (Florence, voiture)\}$

4.4 Composition et image relationnelle

La composition de relations

Etant donné deux relations :
$f : A \leftrightarrow B$
$g : B \leftrightarrow C$
la composition de f et g notée $f \fatsemi g$ donne une relation entre A et C telle que pour tous ses couples de la forme $(a,\ c)$ il y a un couple $(a,\ b)$ dans la relation f et un couple $(b,\ c)$ dans la relation g.

On décrit la composition formellement par :

$$f \fatsemi g = \{a : A; b : B; c : C \mid a \mapsto b \in f \wedge b \mapsto c \in g \bullet a \mapsto c\}$$

L'image relationnelle

Soit f une relation. L'image relationnelle d'un sous-ensemble E donné est le sous-ensemble F constitué des images par la relation f de chaque élément de E.
Elle est notée par $f(\!|\ E\ |\!)$.

Exemple
$conduit(\!|\{patrice\}|\!) = \{velo,\ patins\}$

4.5 les opérations spécifiques sur les fonctions

Les fonctions sont des relations dotées de propriétés particulières. Elles sont manipulées à travers plusieurs fonctions :

Symboles	Description
$\pfun$	Fonction partielle
$\rightarrow$	Fonction totale
$\pinj$	Injection partielle
$\rightarrowtail$	Injection totale
$\psurj$	Surjection partielle
$\twoheadrightarrow$	Surjection totale
$\bij$	Bijection

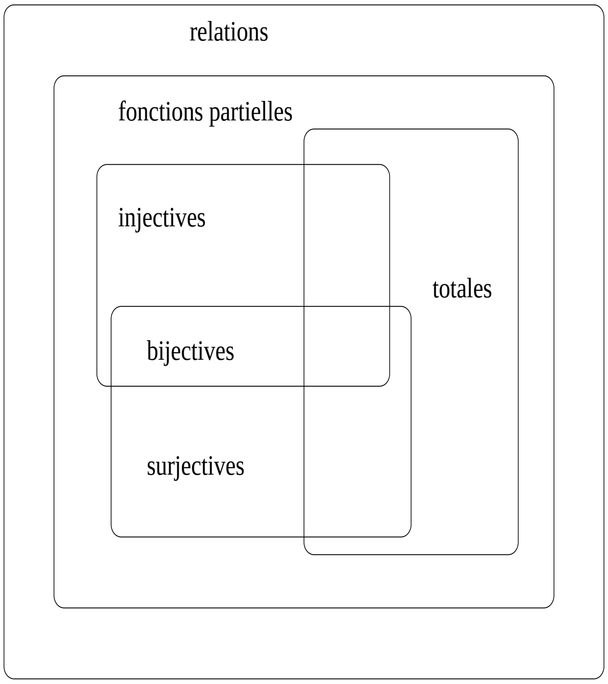
La figure 4.5 donne les imbriactions entre les différentes fonctions et leurs notations.

4.6 Opérations sur les schémas

Utilisation et modification d'un état

Lorsqu'on écrit un schéma d'opération OpS modifiant un état décrit par un schéma S, on utilise le symbole Δ et on écrit ΔS dans la partie déclaration du schéma d'opération. L'effet de ΔS est d'introduire l'état avant et l'état après (où les variables apparaissnt dans l'état sont suffixées de $'$) du schéma S.

Figure 1: Différentes fonctions et notations



Utilisation sans modification d'un état

Lorsqu'on écrit un schéma d'opération OpS consultant (sans modification) un état décrit par un schéma S, on utilise le symbole Ξ et on écrit ΞS dans la partie déclaration de OpS. L'effet de ΞS est d'introduire l'état avant et l'état après su schéma S mais de sorte que l'état après soit identique à l'état avant.

4.7 Le calcul de schéma

Nous donnons ici un rapide aperçu des opérateurs du calcul de schémas. Nous les utiliserons dans les spécifications et pour un approfondissement le lecteur peut consulter [Spi92].

Inclusion de schéma

L'inclusion de schémas consiste à utiliser un ou plusieurs schémas, déjà définis, dans un autre qui est en construction. L'inclusion se fait simplement en listant le nom des schémas à inclure dans la partie déclaration du schéma en construction.
Cette inclusion fait que la partie déclaration du schéma en construction contient toutes les déclarations de tous les schémas inclus.
La partie prédicat du schéma en construction contient tous les prédicats des schémas inclus.
Exemple

$$\begin{array}{|l}
\hline
\textit{S} \\
SO \\
S1 \\
\hline
\textit{predicat de S} \\
\hline
\end{array}$$

Les schémas $S0$ et $S1$ sont ainsi inclus dans S.

Disjonction de schémas

La disjonction de schémas permet de combiner des schémas ayant des déclarations différentes mais sans conflits de type de façon à exprimer la combinaison disjonctive de cas.
Souvent une opération est traitée de différentes façons selon ses entrées par exemple. Le cas normal, les cas d'erreur, les cas particuliers, ... peuvent ainsi être spécifiés distinctement et combinés de manière disjonctive pour avoir un schéma.
Le symbole utilisé pour la disjonction est $\vee$. On écrit alors :

$$NouvSchema \mathrel{\widehat=} SchemaCas1 \vee SchemaCas2 \vee SchemaCas3 \vee \ldots$$

où les $Schema_i$ sont définis distinctement et sans conflit de type dans leurs déclarations. En effet les parties déclarations des schémas sont unies (union ensembliste) et les parties prédicats sont composées avec l'opérateur logique $\vee$.

Conjonction de schémas

L'opérateur de conjonction des schémas est noté : $\wedge$. La conjonction de schémas permet de composer des schémas ayant éventuellement des parties déclarations distinctes ou incomplètes pour avoir un schéma dont la partie déclaration est la "concaténation" des parties déclarations des schémas opérandes et la partie prédicat est la conjonction logique des prédicats des schémas opérandes.
Ici des schémas opérandes pris seuls peuvent ne pas avoir de sens, ils sont donc construits dans le but d'être utilisés dans la conjonction. Chaque schéma opérande dans la conjonction contribue à la définition du schéma final.

Négation de schéma

La négation d'un schéma $S1$ (avec l'opérateur $\neg$) est le schéma obtenu en normalisant le schéma $S1$ puis en prenant comme prédicat la négation logique du prédicat du schéma normalisé. La partie déclaration ne change pas, c'est celle du schéma normalisé.
La normalisation d'un schéma consiste à expanser toutes ses abbréviations et à compléter par conséquent sa partie prédicat. Par exemple une fonction est une forme abbrégée pour un sous-ensemble d'un produit cartésien avec des contraintes exprimées par un prédicat.

Les opérateurs de masquage de schémas

Le masquage, avec l'opérateur \, entre un schéma comme opérande gauche de l'opérateur et un ensemble de variables comme opérande droit de l'opérateur consiste à enlever de la déclaration du schéma, toutes les occurrences des variables de l'ensemble opérande et à masquer ces variables dans la partie prédicat en les quantifiant existentiellement.

Un autre opérateur de masquage est le quantificateur universel $\forall$. On peut écrire $\forall Declaration \mid Predicat \bullet Schema$.

La projection, avec l'opérateur $\upharpoonright$, consiste à masquer toutes les variables du schéma opérande gauche sauf celles apparaisant dans le schéma opérande droit.

Composition des schémas

L'idée de la composition de schéma est de décrire une opération comme compsotion de deux autres. La première opération part d'un état et aboutit à un état qui sert d'état de début de la deuxième opération. Cette dernière décrit un changement d'état qui donne l'état final de l'opération composée. L'opérateur de composition est noté : $\fatsemi$.

Préconditions de schémas

La précondition de schéma noté *pre* est un opérateur qui permet de définir précisément les conditions dans lesquelles une opération est faisable.
Cet opérateur s'applique uniquement aux schémas d'opérations.

Une opération est faisable ou applicable pour les combinaisons d'états avant et de valeurs d'entrée pour lesquelles il existe un état de sortie et des variables de sortie satisfaisant les relations spécifiées.

Soit un schéma d'opération $SchemaOp$:

$pre SchemaOp$ signifie $\exists EtatAbstrait; ValSortie! \bullet SchemaOp$

$EtatAbstrait$ est le schéma d'état du système sur lequel porte l'opération, $ValSortie!$ est l'ensemble des variables de sortie déclarées dans $SchemaOp$.

5 Un autre exemple de spécification Z

(voir exercices)

6 Le raffinement de spécifications Z

(exercices)

7 Notes bibliographiques

Il existe une bonne bibliographie, aussi bien sur le plan qualitatif que sur le plan quantitatif, sur le formalisme Z. Nous citons quelques références ici pour compléter le contenu de cette note de cours.

Le livre de M.Spivey [Spi92] est le manuel de référence pour spécifieur confirmé, il peut tout aussi bien interesser les débutants. Il en existe une traduction française [Spi94]. Le livre de A. Diller [**?**] est une bonne introduction. Un autre livre d'introduction à Z est celi de Lightfoot [Lig91], il est traduit en français [Lig94]. Le livre de Potter, Sinclair et Till [PST91] est aussi une très bonne introduction. Un receuil d'études de cas utilisant les spécifications Z est édité par I. Hayes [Hay93]. Une autre bonne lecture est celle du livre de Barden, Stepney et Cooper [BSC94]. De nombreux articles et rapports se trouvent dans les actes de congrès tels que FME, VDM.

Références

[BSC94] Rosalind Barden, Susan Stepney, and David Cooper. *Z in Practice.* BCS Practitioner Series. Prentice Hall, 94. ISBN 0-13-124934-7.

[Hay93] Ian Hayes. *Specification Case Studies.* in Computer Science. Prentice Hall, 93.

[Lig91] D. Lightfoot. *Formal Specifications using Z.* Macmillan, 91.

[Lig94] D. Lightfoot. *La Specification Formelle avec Z (traduction de [Lig91] par H. Habrias).* Teknea, 94.

[PST91] Ben Potter, Jane Sinclair, and David Till. *An Introduction to Formal Specification and Z.* in Computer Science. Prentice Hall, 91.

[Spi92] Mike Spivey. *The Z Notation: a Reference Manual.* Series in Computer Science. Prentice Hall, 92.

[Spi94] M. Spivey. *La notation Z (traduction de [Spi92] par M. Lemoine).* Masson, 94.