1 La méthode B

La *méthode B* définit, comme Z, un format de spécification : la *machine abstraite*. Elle reprend le concept de raffinement en l'adaptant aux machines abstraites. Mais contrairement à ce que nous avons présenté, B ne fournit pas un jeu prédéfini de règles de raffinemen *a priori* correctes, mais un mécanisme de vérification *a posteriori* engendrant des *obligations de preuves*. Enfin, le langage de spécification, s'il est basé sur un langage ensembliste, enrichi celui-ci d'une construction dédiée à la spécification de programmes : les *substitutions généralisées*.

Substitutions généralisées

Les substitutions généralisées sont, en fait un langage d'instrutions (au sens usuel des langages impératifs) plus ou moins abstraites. En tant que substitutions, on peut les appliquer à un terme, une formule, voire une substitution, pour obtenir un nouveau terme, une nouvelle formule, voire substitution...

On note $[S]\Psi$ l'application de la substitution S à l'expression, la formule ou la substitution Ψ. L'application des substitution est définie par induction sur S. Lorsque S est une *substitution simple* (voir ci-dessous), son application est définie par induction sur Ψ.

Substitution simple La substitution de base est la substitution usuelle notée :

$$x \;\texttt{:=}\; E$$

L'application de la substitution simple $x \;\texttt{:=}\; E$ est définie de façon usuelle sur les expressions et les formules :

$$\begin{array}{lcll}
[x \;\texttt{:=}\; E]x & \widehat{=} & E & \\
[x \;\texttt{:=}\; E]y & \widehat{=} & y & \text{si } x \neq y \\
[x \;\texttt{:=}\; E]F(E_1, \ldots, E_n) & \widehat{=} & F([x \;\texttt{:=}\; E]E_1, \ldots, [x \;\texttt{:=}\; E]E_n) & \\
[x \;\texttt{:=}\; E]\{x \in X \mid \varphi\} & \widehat{=} & \{x \in X \mid \varphi\} &
\end{array}$$

Substitution parallèle Si S_1 et S_2 sont deux substitutions *simples*, on note

$$S_1 \| S_2$$

la substitution appliquant *en même temps* et de façon indépendante, S_1 et S_2.

On généralise la substitution parallèle en *substitution multiple*

$$x_1, \ldots, x_n \;\texttt{:=}\; E_1, \ldots, E_n$$

Soient $z_2, \ldots, z_n$ des variables toutes différentes entre elles, différentes des $x_1, \ldots, x_n$ et n'apparaissant dans aucunes des $E_1, \ldots, E_n$, F on pose :

$$[x_1, \ldots, x_n \;\texttt{:=}\; E_1, \ldots, E_n]F \quad \widehat{=} \quad [z_n \;\texttt{:=}\; E_n] \ldots [z_2 \;\texttt{:=}\; E_2][x_1 \;\texttt{:=}\; E_1][x_2 \;\texttt{:=}\; z_2] \ldots [x_n \;\texttt{:=}\; z_n]F$$

Pour faire court, on dira que les z_i sont des *nouvelles variables*. On a recours au *renommage* pour éviter que les x_i pouvant apparaître dans les E_2, ..., E_n ne soient affectées par la mise en séquence.

On s'autorisera à combiner des substitutions multiples avec l'opérateur $\|$. Par exemple :

$$x_1 \;\texttt{:=}\; E_1 \| x_2, x_3 \;\texttt{:=}\; E_2, E_3 \;\widehat{=}\; x_1, x_2, x_3 \;\texttt{:=}\; E_1, E_2, E_3$$

Substitution préconditionnée Si φ est une formule et S une substitution, on note

$$\textsc{pre}\ \varphi\ \textsc{then}\ S$$

la substitution imposant que φ soit satisfaite pour appliquer S :

$$[\textsc{pre}\ \varphi\ \textsc{then}\ S]F \quad \widehat{=} \quad \varphi \wedge [S]F$$

Substitution indéterminée Si $x_1, \ldots, x_n$ sont des variables, φ un formule et S une substitution, on note

$$\text{ANY } x_1, \ldots, x_n \text{ WHERE } \varphi \text{ THEN } S$$

la substitution qui consiste à choisir n'importe quels $x_1, \ldots, x_n$ qui satisfont φ pour appliquer S :

$$[\text{ANY } x_1, \ldots, x_n \text{ WHERE } \varphi \text{ THEN } S]F \quad \widehat{=} \quad \forall z_1 \ldots z_n.(\varphi' \Rightarrow [S']F)$$

avec $z_1, \ldots, z_n$ nouvelles variables, $\varphi' = [x_1, \ldots, x_n \texttt{ := } z_1, \ldots, z_n]\varphi$ et $S' = [x_1, \ldots, x_n \texttt{ := } z_1, \ldots, z_n]S$. Le renommage est rendu nécessaire à cause de l'introduction du quantificateur universel.

Machines abstraites

Intuitivement, une machine abstraite peut être comparée à un un *objet* comprenant un état interne (les VARIABLES) et des moyens d'actions sur cet état (les OPERATIONS). Comme une machine est un élément de spécification, elle contiendra également des *commentaires logiques* exprimant ses propriétés (l'INVARIANT).

Une machine est constituée de plusieurs rubriques jouant chacune un rôle dans la description des spécifications. Chacune de ces rubriques est identifiée par un mot clef. Voici quelles sont les rubriques essentielles d'une machine.

MACHINE cette rubrique ne contient qu'un seul élément : le nom de la machine.

SETS cette rubrique contient la déclaration des *ensembles* dont se servira la machine. On peut déclarer un ensemble soit comme un simple identificateur (auquel cas son contenu reste abstrait) soit par extension (comme un ensemble énuméré) Tous ces ensembles sont finis (explicitement ou implicitement)

VARIABLES cette rubrique contient la déclaration des variables qu'utilise la machine. L'ensemble de ces variables constitue l'*état* interne de la machine.

INVARIANT cette rubrique contient une formule. C'est un élément trés important de la spécification. L'invariant contient la propriété globale que doit satisfaire la machine. Nous reviendrons ultérieurement sur ce point.

INITIALIZATION cette rubrique contient une susbtitution (généralisée) qui définit la valeur initiale des variables.

OPERATIONS cette rubrique contient la définition d'une liste d'objets appelés *opérations*. Une opération peut modifier l'état de la machine, prendre des arguments ou (inclusif) renvoyer une valeur. Les opérations sont définies en termes de substitutions généralisées.

On peut rajouter encore deux rubriques permetant d'introduire des constantes :

CONSTANTS cette rubrique contient les déclarations des noms des constantes.

CONSTRAINTS cette rubrique contient les axiomes caratérisant les constantes.

L'exemple

Nous allons utiliser tout au long de notre étude de la méthode B un exemple tiré d'une petite étude de cas que J.-Y. Chauvet a publiée dans *1st Conference on the B method, Proceedings, ed. Henri Habrias, Nantes 1996* : le *CARREFOUR*.

Spécification informelle

> La circulation d'un carrefour est réglée par deux feux tricolores dont les couleurs sont vert, orange ou rouge. Sur chacun des feux une seule des couleurs est active à la fois. Le système de feux du carrefour peut être en service ou hors service. Lorsque le système est hors service, les deux feux sont oranges. Lorsque le système est en service, la couleur de chacun des feux change suivant le cycle : orange puis rouge puis vert puis orange, etc ...

Un véhicule ne peut s'engager sur une voie que si le feu n'est pas rouge. Lorsque le système est en service, les feux doivent être réglés de façon à ce que deux véhicules venant de voies différentes ne se trouvent pas en même temps sur le carrefour.
On désire obtenir un système assurant la mise en route du carrefour et la gestion du changement de couleur des feux.

De cette spécification informelle, il faut extraire les composantes essentielles et les traduire en objets formels.

Pré-spécification formelle Il est utile pour pouvoir valider la pertinance de l'analyse de la spécification formelle, de garder trace de l'origine des éléments formels proposés.

1. « les couleurs sont vert, orange ou rouge » :
$$Couleurs = \{vert, orange, rouge\}$$
2. « deux feux tricolores » :
$$feuA \in Couleurs,\ \ feuB \in Couleurs$$
3. « cycle : orange puis rouge puis vert » :
$$Succ \in Couleurs \to Couleurs$$
$$orange \mapsto rouge$$
$$rouge \mapsto vert$$
$$vert \mapsto orange$$
4. « en service ou hors service » :
$$Etats = \{hs, es\}$$
$$etat \in Etats$$
5. « Lorsque le système est hors service, les deux feux sont oranges » :
$$(etat = hs) \Rightarrow (feuA = orange \wedge feuB = orange)$$
6. « Lorsque le système est en service, les feux doivent être réglés de façon à ce que deux véhicules venant de voies différentes ne se trouvent pas en même temps sur le carrefour » :
$$(etat = es) \Rightarrow ((feuA = rouge \vee feuB = rouge) \wedge feuA \neq feuB)$$
Notez que notre formule dit plus que ce que réclamait la spécification informelle. Nous avons ajouté à la propriété réclamée de *sécurité*, une propriété de *disponibilité* garantissant que l'un des feux permet le passage des véhicules.

Spécification formelle Il faut maintenant réunir les éléments formels retenus dans le cadre des machines abstraites.

MACHINE
  $CARREFOUR$

SETS
  $Etats = \{hs, es\}$
  $Couleurs = \{vert, orange, rouge\}$

CONSTANTS
  $Succ$

CONSTRAINTS

$Succ \in Couleurs \rightarrow Couleurs \wedge$
$Succ(orange) = rouge \wedge$
$Succ(rouge) = vert \wedge$
$Succ(vert) = orange$

VARIABLES

$etat, feuA, feuB$

INVARIANT

$etat \in Etats \wedge$
$feuA \in Couleurs \wedge$
$feuB \in Couleurs \wedge$

$(etat = hs) \Rightarrow (feuA = orange \wedge feuB = orange) \wedge$
$(etat = es) \Rightarrow ((feuA = rouge \vee feuB = rouge) \wedge feuA \neq feuB)$

INITIALIZATION

$etat, feuA, feuB$:= $hs, orange, orange$

OPERATIONS

MiseEnService $\widehat{=}$
  PRE $etat = hs$ THEN
    ANY f_1, f_2 WHERE
      $f_1 \in Couleurs \wedge f_2 \in Couleurs \wedge$
      $(f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2$
    THEN
      $etat, feuA, feuB$:= es, f_1, f_2

Changement $\widehat{=}$
  PRE $etat = es$ THEN
    ANY f_1, f_2 WHERE
      $f_1 \in Couleurs \wedge f_2 \in Couleurs \wedge$
      $(f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \wedge$
      $(f_1 = Succ(feuA)) \vee (f_2 = Succ(feuB))$
    THEN
      $feuA, feuB$:= f_1, f_2

Remarquez que dans l'opération de changement de couleur, on a rajouté une propriété de *vivacité* : la formule $f_1 = Succ(feuA)) \vee (f_2 = Succ(feuB)$ énonce que la couleur d'un des deux feux au moins change effectivement.

Correction des machines abstraites

La communauté B a l'habitude de décrire une machine abstraite comme un systéme possédant un état initial et offrant à son utilisateur une série de *boutons* (les opérations) permettant d'agir sur cet état. La correcion d'une machine est établit vis-à-vis de l'invariant :

1. L'initialisation doit *établir* l'invariant.
2. Chaque opération doit *conserver* l'invariant.

Soit le schéma de machine suivant :

MACHINE

M

SETS

X

VARIABLES

x

INVARIANT

I

INITIALIZATION

S_0

OPERATIONS

$\mathrm{O}_1 \;\widehat{=}$

PRE φ THEN S_1

Puisque l'initialisation et les opérations sont définies comme des substitution, la correction de la machine M est exprimée par les formules obtenues n appliqunt ces substitutions à la formule de l'inivariant :

1. $[S_0]I$
2. $I \wedge \varphi \Rightarrow [S_1]I$

Correction de $CARREFOUR$

Décomposons la formule de l'invariant de $CARREFOUR$ en trois conjonctions : $I_1 \wedge I_2 \wedge I_3$ avec

$I_1 \;\widehat{=}\; etat \in Etats \wedge feuA \in Couleurs \wedge feuB \in Couleurs$
$I_2 \;\widehat{=}\; (etat = hs) \Rightarrow (feuA = orange \wedge feuB = orange)$
$I_3 \;\widehat{=}\; (etat = es) \Rightarrow ((feuA = rouge \vee feuB = rouge) \wedge feuA \neq feuB)$

Initialisation Posons

$$S_0 \;\widehat{=}\; etat, feuA, feuB \;:=\; hs, orange, orange$$

L'obligation de preuve de correction de l'initialisation est la formule obtenue en développant l'application

$$[S_0](I_1 \wedge I_2 \wedge I_3)$$

Comme dans la substitution multiple S_0, aucune des variables apparaissant à gauche du signe := n'apparaît à sa droite, on peut traiter cette substitution multiple comme une substitution simple et affirmer que notre obligation de preuve est logiquement équivalente à la formule obtenue en développant

$$[S_0]I_1 \wedge [S_0]I_2 \wedge [S_0]I_3$$

Nous utiliserons par la suite systématiquement cette équivalence pour simplifier le développement des applications de substitutions multiples.

Il faut donc montrer la validité des trois formules $[S_0]I_1$, $[S_0]I_2$ et $[S_0]I_3$.

1. $[S_0]I_1 \;\widehat{=}\; es \in Etats \wedge orange \in Couleurs \wedge orange \in Couleurs$
   qui est trivialement vraie par définition des ensembles $Etats$ et $Couleurs$.
2. $[S_0]I_2 \;\widehat{=}\; (hs = hs) \Rightarrow (orange = orange \wedge orange = orange)$
   qui est on ne peut plus trivialement vraie puisque le vrai entraîne la vrai.
3. $[S_0]I_2 \;\widehat{=}\; (es = hs) \Rightarrow ((orange = rouge \vee orange = rouge) \wedge orange \neq orange)$
   qui est tout aussi trivialement vraie puisque le faux $(es = hs)$ entraîne n'importe quoi.

MiseEnRoute Soit S_1 la substitution de mise en route, posons

$$S_1 \mathrel{\widehat{=}} \text{PRE } \varphi_1 \text{ THEN } S_1'$$

avec

$$\begin{array}{l} \varphi_1 \mathrel{\widehat{=}} etat = hs \\ S_1' \mathrel{\widehat{=}} \text{ANY } f_1, f_2 \text{ WHERE } \psi_1^1 \wedge \psi_1^2 \text{ THEN } S_1'' \end{array}$$

et

$$\begin{array}{l} \psi_1^1 \mathrel{\widehat{=}} f_1 \in Couleurs \wedge f_2 \in Couleurs \\ \psi_1^2 \mathrel{\widehat{=}} (f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \\ S_1'' \mathrel{\widehat{=}} etat, feuA, feuB \ := \ es, f_1, f_2 \end{array}$$

Il faut montrer que $I \wedge \varphi_1 \Rightarrow [S_1']I$, c'est-à-dire, $I \wedge \varphi_1 \Rightarrow \forall f_1, f_2.(\psi_1^1 \wedge \psi_1^2 \Rightarrow [S_1'']I)$. Pour ce, on suppose I et φ_1 et on se donne f_1 et f_2 telles que ψ_1^1 (ie $f_1 \in Couleurs \wedge f_2 \in Couleurs$) et ψ_1^2 (ie $(f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2$) ; reste à montrer $[S_1'']I$, c'est-à-dire $[S_1'']I_1$, $[S_1'']I_2$ et $[S_1'']I_3$.

1. $[S_1'']I_1 \mathrel{\widehat{=}} es \in Etats \wedge f_1 \in Couleurs \wedge f_2 \in Couleurs.$
   On a $es \in Etats$ par définition de l'ensemble $Etats$; et $f_1 \in Couleurs$ et $f_2 \in Couleurs$ par hypothèse (ψ_1^1).
2. $[S_1'']I_2 \mathrel{\widehat{=}} (es = hs) \Rightarrow (f_1 = orange \wedge f_2 = orange)$
   Ce qui est trivialement vrai puisque $es \neq hs.$
3. $[S_1'']I_3 \mathrel{\widehat{=}} (es = es) \Rightarrow ((f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2).$
   Ce qui est rendu vrai par l'hypothèse ψ_1^2.

Changement Soit S_2 la substitution définissant le changement de couleur des feux, posons

$$S_2 \mathrel{\widehat{=}} \text{PRE } \varphi_2 \text{ THEN } S_2'$$

avec

$$\begin{array}{l} \varphi_2 \mathrel{\widehat{=}} etat = hs \\ S_2' \mathrel{\widehat{=}} \text{ANY } f_1, f_2 \text{ WHERE } \psi_2^1 \wedge \psi_2^2 \wedge \psi_2^3 \text{ THEN } S_2'' \end{array}$$

et

$$\begin{array}{l} \psi_2^1 \mathrel{\widehat{=}} f_1 \in Couleurs \wedge f_2 \in Couleurs \\ \psi_2^2 \mathrel{\widehat{=}} (f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \\ \psi_2^3 \mathrel{\widehat{=}} (f_1 = Succ(feuA)) \vee (f_2 = Succ(feuB)) \\ S_2'' \mathrel{\widehat{=}} feuA, feuB \ := \ f_1, f_2 \end{array}$$

Il faut montrer que $I \wedge \varphi_2 \Rightarrow [S_2']I$, c'est-à-dire, $I \wedge \varphi_2 \Rightarrow \forall f_1, f_2.(\psi_2^1 \wedge \psi_2^2 \wedge \psi_2^3 \Rightarrow [S_2'']I)$. Pour ce, on suppose I et φ_2 et on se donne f_1 et f_2 telles que ψ_2^1 (ie $f_1 \in Couleurs \wedge f_2 \in Couleurs$), ψ_2^2 (ie $(f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2$) et ψ_2^3 (ie $(f_1 = Succ(feuA)) \vee (f_2 = Succ(feuB))$) ; reste à montrer $[S_2'']I$, c'est-à-dire $[S_2'']I_1$, $[S_2'']I_2$ et $[S_2'']I_3$.

1. $[S_2'']I_1 \mathrel{\widehat{=}} etat \in Etats \wedge f_1 \in Couleurs \wedge f_2 \in Couleurs.$
   On a $etat \in Etats$ par l'hypothèse I et $f_1 \in Couleurs$ et $f_2 \in Couleurs$ par ψ_2^1.
2. $[S_2'']I_2 \mathrel{\widehat{=}} (etat = hs) \Rightarrow (f_1 = orange \wedge f_2 = orange).$
   Par l'hypothèse φ_2 (la précondition), on a que $etat = es$. Ce qui rend trivialement vraie l'implication recherchée.
3. $[S_2'']I_3 \mathrel{\widehat{=}} (etat = es) \Rightarrow ((f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2).$
   Ce qui nous est donné par I et ψ_2^2

Commentaire sur cette première machine abstraite Les deux opérations de la machine $CARREFOUR$ peuvent se paraphraser par : « *n'importe quelles valeurs qui satisfasse au moins l'invariant* ». Dés lors, la correction est facilement acquise.

Néanmoins, dans les preuves de correction, il faut noter comment la précondition intervient, rendant caduque l'examen de certains cas de l'invariant (ici, $etat = hs$, par exemple).

Raffinement

On peut raffiner une machine sur deux points : les données ou(inclusif) les opérations (c'est à dire, les substitutions). Le raffinement des substitutions porte à son tour sur deux points : affaiblissement des préconditions ou levée de l'indéterminisme. Il va de soit que dans la plus part des cas, le raffinement de données imlique un raffinement d'opération. Une machine et son raffinement définissent les *mêmes opérérations*, seuls peuvent changer les variables, les ensembles et les substitutions définissant les opérations.

Lorsque l'on passe d'une machine à son raffinement, on change d'espace de variables. Pour pouvoir contrôler la légitimité du raffinement, il faut établir le lien entre l'état interne (ie les variables) de la machine originale et celui de son raffinement. On exprime ce lien comme une sous-formule de l'invariant du raffinement que l'on appelle *invariant de liaison*. Cette sous-formule utilise aussi bien les variables de la machine originale que celles du raffinement.

Un raffinement n'est un raffinement correct que s'il conserve la validité des propriétés de la machine d'origine. La méthode B permet de s'assurer de la légitimité d'un raffinement en engendrant des *obligations de preuves* énonçant cette propriété de conservation.

Obligations de preuve

Supposons donc une machine M dont l'invariant est I, l'initialisation est $S0$ et n'ayant qu'une seule opération définie par la substitution conditionnelle PRE φ THEN $S1$. Supposons également que cette machine est correcte. C'est à dire que l'on a effectivement démontré que les substitution $S0$ et PRE φ THEN $S1$ conservent l'invariant I qui exprime les propriétés statiques du système que l'on a en tête.

Soit alors, une machine R dont l'invariant (de liaison) est J, l'initialisation est $T0$ et définissant la même opération que M, mais par la substitution PRE ψ THEN $T1$. Pour établir que R est un raffinement de M, il faut s'assurer que celle-là vérifie encore les propriétés que vérifie celle-ci. En particulier, R doit satisfaire, en quelque sorte, I. Mais puisque M et R sont deux machines distinctes, elles agissent sur des variables différentes. La machine R ne peut donc directement établir I. Pour, depuis la machine R pouvoir parler des variables de M, on utilise le bien nommé invariant de liaison J dont le rôle est justement de *mettre en relation* les variables de M et de R.

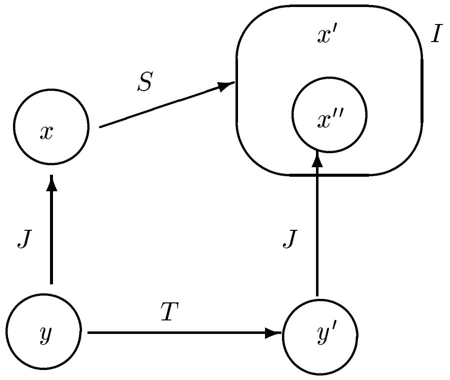
Formellement, les obligations de preuve engendrées par le raffinement de M par R sont

1. $[T0]\neg[S0]\neg J$ et
2. $I \wedge J \wedge \varphi \Rightarrow \psi \wedge [T1]\neg[S1]\neg J$

Pour comprendre intuitivement ce que signifient ces formules nous allons supposer que M travaille sur une variable x appartenant à un ensemble X, et R, sur une variable y appartenant à un ensemble Y. L'invariant de liaison J exprime une relation (au sens mathématique du terme) entre X et Y. Une susbtitution peut également être vue comme une relation : la relation qui à chaque valeur initiale d'une variable associe la valeur obtenue par application de la substitution. Soit alors une substitution S de M et T une substitution de R. Si nous appelons x' l'ensemble des valeurs associées à $x \in X$ par S et y' l'ensemble des valeurs associées à y par T, si nous appelons x'' l'ensemble des valeurs que J associe à y' dans X alors la formule $[T]\neg[S]\neg J$ nous dit que :

il n'est pas possible que x'' ne soit pas une partie de x'

La figure ci-dessous illustre cet énoncé.



Puisque l'on a supposé que M est correcte, ses substitutions établissent I. L'ensemble de valeurs x' satisfait donc I. En démontrant l'obligation de preuve de raffinement $[T]\neg[S]\neg J$ on obtient donc bien que x'' satisfait également I.

Dans l'obligation de preuve concernant les opérations apparaissent explicitement leur préconditions ; celle de l'opération de M en antécédant et celle de l'opération de R en conséquent. Il faudra donc prouver, en particulier, que $I \wedge J \wedge \varphi \Rightarrow \psi$. Ce qui nous donnera que ψ (la précondition du raffinement) est satisfaite chaque fois que φ (la précondition originelle) est satisfaite. On pourra alors utiliser de façon licite, à la fois les substitutions $T1$ et $S1$.

Nous allons à présent illustrer successivement comment utiliser nos deux formes de raffinements en revenant à l'exemple de gestion des feux d'un carrefour.

Raffinement d'opération

Nous allons préciser de faç définitive l'opération de mise en service et de façon encore transitoire l'opération de changement de couleur.

Substitutions généralisées en plus

Pour notre seconde opération, nous allons utiliser deux nouvelles substitutions généralisées.

Substitution gardée Comme la substitution conditionnelle, la substitution gradée est une substitution soumise à la satisfaction d'une certaine condition, ou, plus précisément, un certain *test*. La différence est que dans le cas de la substitution conditionnelle, c'est à l'*utilisateur extérieur* de vérifier la validité de la condition avant d'utiliser la substitution, alors que dans le cas de la substitution gardée, c'est la *substitution elle-même* qui doit vérifier la validité du test. Cette différence est manifeste lorsque l'on regarde comment une substitution gardée s'applique à une formule.

Si φ est une formule et S une substitution, la substitution gardée s'écrit

$$\text{WHEN } \varphi \text{ THEN } S$$

On définit l'application de la substitution gardée par

$$[\text{WHEN } \varphi \text{ THEN } S]F \mathrel{\widehat{=}} \varphi \Rightarrow [S]F$$

Alors que l'application d'une substitution conditionnelle engendrait une conjonction, nous avons ici une implication. Dans le cas de la substitution gardée, la formule de garde ψ décrit un état interne pouvant ne pas apparaître. Dans ce cas, on ne « déclanchera » pas la substitution S. On n'a donc pas d'obligation de toujours satisfaire φ. Il suffit de s'assurer que chaque fois que φ est vraie, S établit bien F. C'est bien ce qu'exprime le connecteur propositionnel $\Rightarrow$.

Substitution à choix borné La substitution à choix borné est une autre sorte de substitution indéterminée. Elle permet de pouvoir utiliser au choix un nombre fini de substitutions. Elle s'écrit

$$\text{CHOICE } S_1 \text{ OR} \ldots \text{OR } S_n$$

Chacune des substitutions proposées par un choix borné doit pouvoir être utilisée de façon indifférente. On définit donc l'application d'une substituiton à choix borné par une conjonction :

$$[\text{CHOICE } S_1 \text{ OR} \ldots \text{OR } S_n]F \mathrel{\widehat{=}} [S_1]F \wedge \ldots \wedge [S_n]F$$

Le raffinement $CARREFOUR1$

Pour raffiner l'opération de mise en service, nous allons choisir arbitrairement de mettre un feu au vert et l'autre au rouge.
L'idée du raffinement de l'opération de changement de couleur est de spécifier la fonction $Succ$ comme substitution décrivant les divers cas de figures déterminés par les équations qui définissent la fonction $Succ$. C'est une première étape vers l'*implémentation* de la fonction mathématique $Succ$.

Une machine en raffinant une autre n'est pas nommée par la rubrique MACHINE, mais par la rubrique REFINEMENT. De plus, elle doit obligatoirement contenir une rubrique additionnelle REFINES indiquant quelle machine elle raffine.

REFINEMENT
  $CARREFOUR1$

REFINES
  $CARREFOUR$

SETS
  $Etats = \{hs, es\}$
  $Couleurs = \{vert, orange, rouge\}$

VARIABLES
  $etat_1, feuA_1, feuB_1$

INVARIANT
  $etat_1 = etat \wedge$
  $feuA_1 = feuA \wedge$
  $feuB_1 = feuB$

INITIALIZATION
  $etat_1, feuA_1, feuB_1 := hs, orange, orange$

OPERATIONS
  MiseEnService $\widehat{=}$
    PRE $etat_1 = hs$ THEN
      $etat_1, feuA_1, feuB_1 := es, rouge, vert$

  Changement $\widehat{=}$
    PRE $etat_1 = es$ THEN
      CHOICE
        WHEN $feuA_1 = vert$ THEN $feuA_1 := orange$
      OR
        WHEN $feuA_1 = orange$ THEN $feuA_1, feuB_1 := rouge, vert$

$$
\begin{array}{l}
\text{OR} \\
\quad \text{WHEN } feuA_1 = rouge \wedge feuB_1 = vert \text{ THEN } feuB_1 \ := \ orange \\
\text{OR} \\
\quad \text{WHEN } feuA_1 = rouge \wedge feuB_1 = orange \text{ THEN } feuA_1, feuB_1 \ := \ vert, rouge
\end{array}
$$

Obligations de preuves

Nous donnons, sans tout le détail de leur calcul, les obligations de preuves engendrées.

Initialisation On obtient la formule triviale

$$\neg\neg(hs = hs \wedge orange = orange \wedge orange = orange)$$

Mise en service Appelons J l'invariant de $CARREFOUR1$. On obtient la formule

$$
\begin{array}{l}
I \wedge J \wedge etat = hs \Rightarrow \\
\quad etat_1 = hs \wedge \\
\quad \neg(\forall f_1, f_2.(f_1 \in Couleurs \wedge f_2 \in Couleurs \wedge (f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \Rightarrow \\
\qquad\qquad \neg(es = es \wedge f_1 = rouge \wedge f_2 = vert)))
\end{array}
$$

Ce qui donne, en simplifiant les négations

$$
\begin{array}{l}
I \wedge J \wedge etat = hs \Rightarrow \\
\quad etat_1 = hs \wedge \\
\quad \exists f_1, f_2.(f_1 \in Couleurs \wedge f_2 \in Couleurs \wedge (f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \wedge \\
\qquad\qquad es = es \wedge f_1 = rouge \wedge f_2 = vert)
\end{array}
$$

On obtient $etat_1 = hs$ de $etat_1 = etat$ ($cf\ J$) et l'hypothèse $etat = hs$.
On obtient le second membre de la conjonction en prenant $rouge$ et $vert$ pour valeurs respectives de f_1 et f_2.

Il convient de noter ici comment la quatification universlle de ls substitution déterminée est devenue une quantification existentielle par le jeu de la négation qu'introduit l'obligation de preuve de raffinement. Pour montrer la validité d'une formule existentielle, il nous faut exhiber au moins une valeur satisfaisante (ou, à tout le moins montrer qu'on ne peut pas supposer qu'il n'en existe pas, si l'on raisonne par l'absurde). Ainsi l'obligation de preuve du raffinement rajoute de l'information par rapport à la correction de la machine originelle.

Changement de couleur Soit S_2 la substitution définissant le changement de couleur des feux de la machine initiale $CARREFOUR$, posons

$$S_2 \mathrel{\widehat{=}} \text{PRE } \varphi_2 \text{ THEN ANY } f_1, f_2 \text{ WHERE } \psi_2 \text{ THEN } S'_2$$

Calculons, dans un premier temps $\neg[S2]\neg J$. En simplifiant les négations, on obtient

$$\exists f_1, f_2.(\psi_2 \wedge [S'_2]J)$$

Soit maintenant T_2 la substitution définissant l'opération de changement de couleur dans le raffinement $CARREFOUR1$. Posons

$$T_2 \mathrel{\widehat{=}} \text{PRE } \varphi_2 \text{ THEN CHOICE } T'_1 \text{ OR } T'_2 \text{ OR } T'_3 \text{ OR } T'_4$$

Avec, pour chaque $i \in [1..4]$,

$$T'_i \mathrel{\widehat{=}} \text{WHEN } \gamma_i \text{ THEN } T''_i$$

L'application $[T_2]\exists f_1, f_2.(\gamma_2 \wedge [S'_2]J)$ nous donne la conjonction

$$\begin{array}{l}(\gamma_1 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_1][S'_2]J)) \wedge \\ (\gamma_2 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_2][S'_2]J)) \wedge \\ (\gamma_3 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_3][S'_2]J)) \wedge \\ (\gamma_4 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_4][S'_2]J))\end{array}$$

Au final, l'obligation de preuve concernant l'opération de changement de couleur s'écrit

$$\begin{array}{l}I \wedge J \wedge etat = es \Rightarrow \\ \quad (etat_1 = es) \wedge \\ \quad (\gamma_1 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_1][S'_2]J)) \wedge \\ \quad (\gamma_2 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_2][S'_2]J)) \wedge \\ \quad (\gamma_3 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_3][S'_2]J)) \wedge \\ \quad (\gamma_4 \Rightarrow \exists f_1, f_2.(\psi_2 \wedge [T''_4][S'_2]J))\end{array}$$

On obtient $etat_1 = es$ comme précédemment.
Traitons le premier des quatre autres membres de la conjonction et laissons les derniers en exercice. Rappelons que

$$\gamma_1 \mathrel{\widehat{=}} feuA_1 = vert$$

que

$$\begin{array}{ll}\psi_2 \mathrel{\widehat{=}} & f_1 \in Couleurs \wedge f_2 \in Couleurs \wedge \\ & (f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \wedge \\ & (f_1 = Succ(feuA)) \vee (f_2 = Succ(feuB))\end{array}$$

et que

$$[T''_1][S'_2]J \mathrel{\widehat{=}} etat_1 = es \wedge orange = f_1 \wedge feuB_1 = f_2$$

Il faut donc montrer que sous les hypothèses I, J, $etat = es$ et $feuA_1 = vert$, on peut trouver deux valeurs pour f_1 et f_2 telles que l'on ait la conjonction

$$\begin{array}{l}f_1 \in Couleurs \wedge f_2 \in Couleurs \wedge \\ (f_1 = rouge \vee f_2 = rouge) \wedge f_1 \neq f_2 \wedge \\ (f_1 = Succ(feuA)) \vee (f_2 = Succ(feuB)) \wedge \\ etat_1 = es \wedge orange = f_1 \wedge feuB_1 = f_2\end{array}$$

Le choix de f_1 est simple, puisqu'il faut satisfaire que $orange = f_1$. Pour ce qui est de f_2, on peut faire le choix minimaliste en prenant $f_2 = feuB$. Notre obligation de preuve se résume alors à la vérification des neufs formules suivantes.

1. $orange \in Couleurs$
   Trivial par définition de $Couleurs$.
2. $feuB \in Couleurs$
   On l'a par hypothèse I.
3. $orange = rouge \vee feuB = rouge$
   On ne pourra pas avoir $orange = rouge$, montrons donc $feuB = rouge$: On a l'hypothèse $feuA_1 = vert$ ainsi que (par J) $feuA_1 = feuA$. On a donc que $feuA = vert$. Or I nous dit que $feuA = rouge \vee feuB = rouge$, comme $feuA = vert \neq rouge$, il faut bien que $feuB = rouge$.
4. $orange \neq feuB$
   On l'obtient en montrant comme ci-dessus qu'en fait $feuB = rouge$.
5. $orange = Succ(feuA) \vee feuB = Succ(feuB)$
   Trivial en utilisant l'hypothèse $feuA = vert$.
6. $etat_1 = es$
   On l'a déjà montré...
7. $orange = orange$
   Sans commentaire...
8. $feuB_1 = feuB$
   On l'a par l'hypothèse J.

Raffinement de données

Une spécification manipule en général des données abstraites : des ensembles. Une étape importante du raffinement est donc d'obtenir une repréntation concrètes des données.

Nous illustrons cette catégorie particulière de raffinement sur notre exemple en introduisant l'usage de valeurs booléennes en place des états et couleurs. Ceci implique restructuration de la représentation des feux qui à son tour implique une reformulation des opérations. Cependant, pour minimiser l'impact de la modification de représentation des données, nous conserverons la structure *logique* des substitutions définissant les opérations.

La relation entre les variables de la machine originale et de son raffinement est exprimée dans l'invariant du raffinement.

Le raffinement $CARREFOUR2$

Puisque l'état du système est à valeur dans un ensemble à deux élément, on peut utiliser des booléens. De plus, on peut représenter chaque feux par un triplet de booléens dont, suivant l'intuition, chaque composante correspond à une lampe, éteinte ou allumée, d'une couleur donnée. Chaque lampe pouvant prendre deux valeurs, on utilise ici encore les booléens.

L'ensemble des booléens est supposé (pré)défini par : $BOOL = \{true, false\}$.

REFINEMENT
  $CARREFOUR2$

REFINES
  $CARREFOUR1$

VARIABLES
  $etat_2, Av, Ao, Ar, Bv, Bo, Br$

INVARIANT
  $etat_2 \in BOOL \wedge$
  $Av, Ao, Ar \in BOOL \times BOOL \times BOOL \wedge$
  $Bv, Bo, Br \in BOOL \times BOOL \times BOOL \wedge$

  $(etat_2 = true \Leftrightarrow etat_1 = es) \wedge$
  $(feuA_1 = vert \Leftrightarrow Av = true \ \wedge \ Ao = false \ \wedge \ Ar = false) \wedge$
  $(feuA_1 = orange \Leftrightarrow Av = false \ \wedge \ Ao = true \ \wedge \ Ar = false) \wedge$
  $(feuA_1 = rouge \Leftrightarrow Av = false \ \wedge \ Ao = false \ \wedge \ Ar = true) \wedge$
  $(feuB_1 = vert \Leftrightarrow Bv = true \ \wedge \ Bo = false \ \wedge \ Br = false) \wedge$
  $(feuB_1 = orange \Leftrightarrow Bv = false \ \wedge \ Bo = true \ \wedge \ Br = false) \wedge$
  $(feuB_1 = rouge \Leftrightarrow Bv = false \ \wedge \ Bo = false \ \wedge \ Br = true)$

INITIALIZATION
  $etat_2$:= $false \parallel$
  Av, Ao, Ar := $false, true, false \parallel$
  Bv, Bo, Br := $false, true, false$

OPERATIONS
  MiseEnService $\widehat{=}$
    PRE $etat_2 = false$ THEN
      $etat_2$:= $true \parallel$
      Ao, Ar := $false, true \parallel$

Bv, Bo := $true, false$

Changement $\widehat{=}$
PRE $etat_2 = true$ THEN
CHOICE
WHEN $Av = true$ THEN Av, Ao := $false, true$
OR
WHEN $Ao = true$ THEN Ao, Ar := $false, true \parallel Bv, Br$:= $true, false$
OR
WHEN $Ar = true \wedge Bv = true$ THEN Bo, Bv := $true, false$
OR
WHEN $Ar = true \wedge Bo = true$ THEN Ar, Av := $false, true \parallel Bo, Br$:= $false, true$

Obligations de preuves

Les obligations de preuves vont ici essentiellement contrôler la cohérence de l'usage de la nouvelle représentation des données dans les opérations vis-à-vis de la façon dont l'invariant définit cette représentation.

Initialisation On obtient, en simplifiant la double négation,

$$
\begin{array}{l}
false \in BOOL \wedge \\
false, true, false \in BOOL \times BOOL \times BOOL \wedge \\
false, true, false \in BOOL \times BOOL \times BOOL \wedge \\
\\
(false = true \Leftrightarrow hs = es) \wedge \\
(orange = vert \Leftrightarrow false = true \ \wedge \ true = false \ \wedge \ false = false) \wedge \\
(orange = orange \Leftrightarrow false = false \ \wedge \ true = true \ \wedge \ false = false) \wedge \\
(orange = rouge \Leftrightarrow false = false \ \wedge \ true = false \ \wedge \ false = true) \wedge \\
(orange = vert \Leftrightarrow false = true \ \wedge \ true = false \ \wedge \ false = false) \wedge \\
(orange = orange \Leftrightarrow false = false \ \wedge \ true = true \ \wedge \ false = false) \wedge \\
(orange = rouge \Leftrightarrow false = false \ \wedge \ true = false \ \wedge \ false = true)
\end{array}
$$

On vérifie facilement la validité de la formule obtenue sachant que deux propositions sont équivalentes lorsqu'elles ont même valeur de vérité ; soit vrai, soit faux.

Mise en service Nous avons présenté les obligations de preuve de correction du raffinement dans le cadre simplifié d'une machine originale et d'un raffinement de celle-ci. Ici, Nous sommes en présence de trois machines dans une relation de raffinement mutuelle : $CARREFOUR \supseteq CARREFOUR1 \supseteq CARREFOUR2$ (pour reprendre la notation de **??**). Les obligation de preuves établissant que $CARREFOUR1 \supseteq CARREFOUR2$ doivent prendre en compte l'invariant de $CARREFOUR$ puisqu'en fin de compte, c'est toujours cette machine originale que nous raffinons. D'ailleurs, une formule ne faisant intervenir que les invariant de $CARREFOUR2$ et $CARREFOUR1$ aurait peu de sens puisque l'invariant (de liaison) de $CARREFOUR1$ fait référence aux variables de $CARREFOUR$.

Si K est l'invariant de $CARREFOUR2$, si on écrit son opération de mise en service sous la forme

$$\text{PRE } etat_2 = false \text{ THEN } U_1$$

si l'on écrit l'opération de mise en service de $CARREFOUR1$ sous la forme

$$\text{PRE } etat_1 = hs \text{ THEN } T_1$$

alors la formule à calculer pour obtenir l'obligation de preuve est

$$I \wedge J \wedge K \wedge etat = hs \wedge etat_1 = hs \Rightarrow etat_2 = false \wedge [U_1]\neg[T_1]\neg K$$

Ce qui donne

$$\begin{array}{l} I \wedge J \wedge K \wedge etat_1 = hs \Rightarrow \\ \quad etat_2 = false \wedge \\ \quad true \in BOOL \wedge \\ \quad Av, false, true \in BOOL \times BOOL \times BOOL \wedge \\ \quad true, false, Br \in BOOL \times BOOL \times BOOL \wedge \\ \quad (true = true \Leftrightarrow es = es) \wedge \\ \quad (rouge = vert \Leftrightarrow Av = true \ \wedge \ false = false \ \wedge \ true = false) \wedge \\ \quad (rouge = orange \Leftrightarrow Av = false \ \wedge \ false = true \ \wedge \ true = false) \wedge \\ \quad (rouge = rouge \Leftrightarrow Av = false \ \wedge \ false = false \ \wedge \ true = true) \wedge \\ \quad (vert = vert \Leftrightarrow true = true \ \wedge \ false = false \ \wedge \ Br = false) \wedge \\ \quad (vert = orange \Leftrightarrow true = false \ \wedge \ false = true \ \wedge \ Br = false) \wedge \\ \quad (vert = rouge \Leftrightarrow true = false \ \wedge \ false = false \ \wedge \ Br = true) \end{array}$$

Seules deux formules méritent ici un peu d'attention

$$rouge = rouge \Leftrightarrow Av = false \ \wedge \ false = false \ \wedge \ true = true$$

et

$$vert = vert \Leftrightarrow true = true \ \wedge \ false = false \ \wedge \ Br = false$$

Pour en montrer la validité, il faut obtenir que $Av = false$ et que $Br = false$ à partir des hypothèses I, J, K, $etat = hs$ et $etat_1 = hs$. Le raisonnement est le suivant : de J et de $etat_1 = hs$, on tire $etat = hs$; de I et $etat = hs$, on tire que $feuA$ et $feuB$ valent $orange$; en utilisant les invariants de liaison J et K, on obtient les deux égalités recherchées.

Changement de couleur Bien qu'un peu fastidieuse, il est intéressant d'étudier l'obligation de preuve du raffinement de l'opération de changement de couleur pour voir comment se distribuent les alternatives des substituions à choix borné.

Nous reprenons la décomposition de la substitution définissant l'opération de changement de couleur de $CARREFOUR1$ et introduisons une décomposition analogue pour celle de $CARREFOUR2$:

$$U_2 \mathrel{\hat{=}} \text{PRE } \varphi_3 \text{ THEN CHOICE } U'_1 \text{ OR } U'_2 \text{ OR } U'_3 \text{ OR } U'_4$$

Avec, pour chaque $i \in [1..4]$,

$$U'_i \mathrel{\hat{=}} \text{WHEN } \delta_i \text{ THEN } U''_i$$

Le calcul de $[U_2]\neg[T_2]\neg K$ donne, en simplifiant les négations, la conjonction suivante :

$$\begin{array}{l} (\delta_1 \Rightarrow ((\gamma_1 \wedge [U''_1][T''_1]K) \vee (\gamma_2 \wedge [U''_1][T''_2]K) \vee (\gamma_3 \wedge [U''_1][T''_3]K) \vee (\gamma_4 \wedge [U''_1][T''_4]K))) \wedge \\ (\delta_2 \Rightarrow ((\gamma_1 \wedge [U''_2][T''_1]K) \vee (\gamma_2 \wedge [U''_2][T''_2]K) \vee (\gamma_3 \wedge [U''_2][T''_3]K) \vee (\gamma_4 \wedge [U''_2][T''_4]K))) \wedge \\ (\delta_3 \Rightarrow ((\gamma_1 \wedge [U''_3][T''_1]K) \vee (\gamma_2 \wedge [U''_3][T''_2]K) \vee (\gamma_3 \wedge [U''_3][T''_3]K) \vee (\gamma_4 \wedge [U''_3][T''_4]K))) \wedge \\ (\delta_4 \Rightarrow ((\gamma_1 \wedge [U''_4][T''_1]K) \vee (\gamma_2 \wedge [U''_4][T''_2]K) \vee (\gamma_3 \wedge [U''_4][T''_3]K) \vee (\gamma_4 \wedge [U''_4][T''_4]K))) \wedge \end{array}$$

Notez comment, dans chaque membre de cette conjonction, la négation a introduit une disjonction en place de la conjonction que donnait l'application de T_2.

Que le lecteur se rassure, nous n'allons pas traiter exhaustivement cette obligation de preuve¿Nous nous contenterons de décrire comment, sous les hypothèses I, J, K, $etat = es$ et $etat_1 = es$, on obtient le premier membre de cette conjonction. Les autres se traitent de façon similaire.

Une analyse rapide de la forme obtenue pour $[U_2]\neg[T_2]\neg K$ nous porte à penser que pour chaque δ_i, un seul des membres de la disjonction correspondante doit être pertinent: celui de la forme $\gamma_i \wedge [U''_i][T''_i]K$. C'est ce que nous allons établir pour $i = 1$; les autres cas se traiteraient de façon similaire..

Supposons donc δ_1 (ie $Av = true$) et montrons que γ_1 (ie $feuA1 = vert$) ainsi que $[U''_1][T''_1]K$.
On obtient $feuA1 = vert$ de $Av = true$ et K qui dit que le seul cas où Av a la valeur $true$, c'est quand

$feuA1$ a la valeur $vert$.
L'application $[U_1''][T_1'']K$ se développe en

$$\begin{array}{l}
etat_2 \in BOOL \land \\
false, true, Ar \in BOOL \times BOOL \times BOOL \land \\
Bv, Bo, Br \in BOOL \times BOOL \times BOOL \land \\
(etat_2 = true \Leftrightarrow etat_1 = es) \land \\
(orange = vert \Leftrightarrow false = true \ \land \ true = false \ \land \ Ar = false) \land \\
(orange = orange \Leftrightarrow false = false \ \land \ true = true \ \land \ Ar = false) \land \\
(orange = rouge \Leftrightarrow false = false \ \land \ true = false \ \land \ Ar = true) \land \\
(feuB_1 = vert \Leftrightarrow Bv = true \ \land \ Bo = false \ \land \ Br = false) \land \\
(feuB_1 = orange \Leftrightarrow Bv = false \ \land \ Bo = true \ \land \ Br = false) \land \\
(feuB_1 = rouge \Leftrightarrow Bv = false \ \land \ Bo = false \ \land \ Br = true)
\end{array}$$

La plus par des élément de cette conjonction sont données par hypothèse ou directement en utilisant que faux est équivalent à faux. Le seul cas non trivial est celui de l'équivalence

$$orange = orange \Leftrightarrow false = false \ \land \ true = true \ \land \ Ar = false$$

ou il faut obtenir $Ar = false$. Ce que l'on obtient en utilisant le fait (démontré ci-dessus) que $Ar = false$ implique que $feuA1 = vert$. On utilise alors ce résultat et K pour obtenir $Ar = false$.

L'implantation

L'ultime étape de raffinement reste dans le formalisme des machines abstraites, mais elle est soumise à un certain nombre de restrictions. Nous ne suivrons pas ici complètement *la lettre* de B, mais plutôt son esprit en nous contentant de les principales de ces restrictions :

- les opérations ne peuvent être définies qu'en utilisant un jeu restreint de substitutions correspondant en fait à des instructions d'un langage de programmation impératif ;
- tous les ensembles utilisés doivent correspondre à des représentation concrètes de valeurs.

Substitutions exécutables

Affectation L'affectation est tout simplement la substitution simple x := E en restreignant l'expression E à n'utiliser que des opérateurs connus d'un langage de programmation.

Séquence La séquence des langages de programmations correspond à la composition des substitutions :

$$[S_1 \, ; \;\; S_2]\Psi \;\widehat{=}\; [S_2][S_1]\Psi$$

Conditionnelles La conditionnelle des langages de programmation est définie comme combinaison de deux substitutions gardées et d'une substitution à choix borné :

$$\textsc{if}\ B\ \textsc{then}\ S_1\ \textsc{else}\ S_2 \;\widehat{=}\; \textsc{choice}\ (\textsc{when}\ B\ \textsc{then}\ S_1)\ \textsc{or}\ (\textsc{when}\ \neg B\ \textsc{then}\ S_1)$$

La condition B doit être exprimée en n'utilisant que les connecteurs propositionnels correspondant aux opérateurs booléens usuels.

L'instruction vide On a la toujours utile SKIPque l'on définit comme la substitution identitée qui remplace chaque variable par elle-même.

L'ultime raffinement

En tant que machine particulière, le raffinement correspondant à une implantation est introduit par la clause IMPLEMENTATION en place de REFINEMENT.

IMPLEMENTATION
$CARREFOUR\_SYSTEM$

REFINES
$CARREFOUR2$

VARIABLES
$a_1, a_2, a_3, b_1, b_2, b_3$

INVARIANT
$a1 \in BOOL \wedge a2 \in BOOL \wedge a3 \in BOOL \wedge$
$b1 \in BOOL \wedge b2 \in BOOL \wedge b3 \in BOOL \wedge$

$a1 = Av \wedge a2 = Ao \wedge a3 = Ar \wedge$
$b1 = Bv \wedge b2 = Bo \wedge b3 = Br$

INITIALIZATION
$a1$:= $false$; $a2$:= $true$; $a3$:= $false$;
$b1$:= $false$; $b2$:= $true$; $b3$:= $false$

OPERATIONS
MiseEnService $\widehat{=}$
$a2$:= $false$; $a3$:= $true$;
$b1$:= $true$; $b2$:= $false$

Changement $\widehat{=}$
IF $a1 = true$ THEN
$a1$:= $false$; $a2$:= $true$
ELSE IF $a2 = true$ THEN
$a2$:= $false$; $a3$:= $true$;
$b1$:= $true$; $b3$:= $false$
ELSE IF $b1 = true$ THEN
$b1$:= $false$; $b2$:= $true$
ELSE
$a1$:= $true$; $a3$:= $false$;
$b2$:= $false$; $b3$:= $true$

Notez que les préconditions ont disparues des définitions des opérations. En effet, une précondition n'est pas destinée à être vérifiée par le code du programme à chaque exécution, mais plutôt, *a priori* par l'utilisateur des opérations fournies par une machine. Une précondition est un pur élément de spécification.

Obligations de preuves

L'invariant de l'implantation est trivial. Nous ne traiterons donc pas l'initialisation ni l'opération de mise en route.

L'obligation de preuve de l'opération de changement de couleurs va nous permettre de vérifier la cohérence de l'analyse de cas spécifiée dans l'implantation vis-à-vis de celle définie précédemment en terme de choix entre substitutions gardées. En particulier, nous allons pouvoir nous assurer de l'exhaustivité des choix proposés.

Appelons V_2 la substitution définissant l'opération de changement de couleur de l'implantation et réécrivonssous la sous la forme :

$$\begin{array}{l}\text{IF } B_1 \text{ THEN} V_1' \\ \text{ELSE IF } B_2 \text{ THEN } V_2' \\ \text{ELSE IF } B_3 \text{ THEN } V_3' \\ \text{ELSE } V_4'\end{array}$$

Soit L l'invariant de l'implantation, en reprenant U_2, la substitution définissant le changement de couleur dans $CARREFOUR2$, en développant $\neg[U_2]\neg L$, on obtient :

$$(\delta_1 \wedge [U_1'']L) \vee (\delta_2 \wedge [U_2'']L) \vee (\delta_3 \wedge [U_3'']L) \vee (\delta_4 \wedge [U_4'']L)$$

que nous écrirons plus simplement

$$\bigvee_{i=1}^{4} ((\delta_i \wedge [U_i'']L)$$

Comme nous l'avions remarqué pour les substitutions multiples, on peut vérifier que pour chacune des substitutions séquentielles V_j' avec $j \in [1..4]$ l'application $[V_j'] \bigvee_{i=1}^{4} ((\delta_i \wedge [U_i'']L)$ donne une formule logiquement équivalente à $\bigvee_{i=1}^{4} ((\delta_i \wedge [V_j'][U_i'']L)$. Posons, pour simplifier les écritures

$$\Phi_j \mathrel{\widehat{=}} \bigvee_{i=1}^{4} ((\delta_i \wedge [V_j'][U_i'']L)$$

Tenant compte de cette équivalence, le développement $[V_2]\neg[[U_2]\neg L$ est lui-même équivalent à la formule :

$$(B_1 \Rightarrow \Phi_1) \wedge (\neg B_1 \Rightarrow ((B_2 \Rightarrow \Phi_2) \wedge (\neg B_2 \Rightarrow ((B_3 \Rightarrow \Phi_3) \wedge (\neg B_3 \Rightarrow \Phi_4)))))$$

L'implication se distribuant sur la conjonction, et la proposition $A \Rightarrow (B \Rightarrow C)$ étant équivalente à $(A \wedge B) \Rightarrow C$, notre obligation de preuve est elle-même équivalente à la conjonction

$$\begin{array}{l}(B_1 \Rightarrow \Phi_1) \wedge \\ (\neg B_1 \wedge B_2 \Rightarrow \Phi_2) \wedge \\ (\neg B_1 \wedge \neg B_2 \wedge B_3 \Rightarrow \Phi_3) \wedge \\ (\neg B_1 \wedge \neg B_2 \wedge \neg B_3 \Rightarrow \Phi_4)\end{array}$$

Les différents cas de cette obligation de preuve se traitent selon la technique appliquée lors de la preuve de correction du raffinement $CARREFOUR2$: on vérifie le membre « pertinent » de la disjonction ; celui ou les indices j et i coïncident.
La nouveauté ici est que les tests contiennent plus d'implicite que les gardes. Traitons le dernier cas, qui est celui où l'implicite culmine.

Rappelons nous que nous travaillons sous l'hypothèse que les invariants I, J et K sont satisfaits et que le système est en service. Nous allons, dans un premier déduire de $\neg B_1 \wedge \neg B_2 \wedge \neg B_3$ la couleur des deux feux. Intuitivement : puisque chaque feu doit avoir une couleur, si ni $a1$ ni $a2$ sont à vrai alors nécessairement $a3 = true$. C'est-à-dire que le premier feu ($feuA$) est au rouge. Le second ($feuB$) est donc soit $vert$ soit $orange$. Comme $b1$ (la lampe verte) n'est pas à vrai, c'est que le second feu est à l'orange.
De ce résultat, il est facile de déduire que δ_4 et $[V_4'][U_4'']L$ sont valides. Ce qui nous donne Φ_4.

Quelques mots pour finir

Le travail réalisé sur ce petit exemple peut sembler bien démesuré en regard du résultat atteint : trois petites fonctions (l'initialisation et les deux opérations) dont l'intuition nous était venue dés l'énoncé du problème et que nous aurions donc pu tout aussi bien écrire directement.

À cette objection, il convient de faire deux réponses :

- le problème étudié ici est un *exemple* et comme tout exemple, sa valeur est dans sa simplicité. Mais la vie réelle a rarement cette simplicité et alors le travail d'élaboration de la spécification qui permet une première formulation triviale des opération (substitution indéterminée) puis la décomposition contrôlée par les obligations de preuves de chacune des étapes de raffinement prennent tout leur sens. Même sur ce petit exemple, on peut voir comment la méthode permet d'obtenir une formulation assez concise de l'opération de changement de couleur qui maîtrise un compléxité non négligeable : les 2^6 combinaisons booléennes *a priori* possibles des variables du système.
- le système simple développé ici peut n'être qu'un composant d'un plus vaste système et la défaillance de notre système simple peut entraîner la défaillance du système plus vaste. Aussi est-il nécessaire de s'assurer de la fiabilité du composant. Mais nous avons ici obtenu plus. Non seulement, nous nous sommes assurés de la fiabilité des opérations de manipulations des feux, mais nous sommes également en mesure de fournir un *certificat de garantie* de cette fiabilité sous la forme des étapes successives du raffinement accompagnées chacune de la démonstration de leurs obligations de preuves.