Introducción al Modelado de Sistemas
Capítulo 4

Camilo Rueda

1 de agosto de 2013

1. Operadores sobre expresiones y predicados en Event B

Como se ha mencionado en los capítulos anteriores, el lenguaje de *event B* define dos tipos de construcciones fundamentalmente diferentes: *expresiones* y *predicados*. Cada expresión representa un *valor*. Este valor puede ser un número o también puede ser una entidad de otro tipo, como se verá más adelante. Cuando las expresiones representan un valor numérico se llaman *expresiones aritméticas*. Por ejemplo, $(x - y + 2) * 3 - 1$ es una expresión aritmética cuyo valor es igual a 11 cuando $x = 3, y = 1$. Las expresiones aritméticas se construyen usando algunos de los operadores siguientes:

Operador	Símbolo en Rodin	Símbolo en matemáticas
suma	$+$	$+$
resta	$-$	$-$
multiplicación	$*$	$\times$
división entera	$/$	$\div$
exponenciación	$\hat{}$	$\hat{}$
módulo	mod	mod

Note que la división es siempre *entera*. Es decir, el valor de la expresión $a \div b$ es un número entero, sin decimales. Por ejemplo, el valor de $7 \div 2$ es igual a 3. El símbolo *mod* representa la operación que obtiene el residuo de la división entre sus dos operandos. A esta operaciñ se denomina *módulo*. Por ejemplo,

Expresión	Valor
$7\ mod\ 2$	1
$16\ mod\ 4$	0
$17\ mod\ 3$	2

La operación *mod* tiene una gran importancia en computación. Un uso típico de esta operación es para la obtención de los dígitos de un número. Por ejemplo, dado el número

$n = 421$, tenemos:

Expresión	Valor
$n \ mod \ 10$	1
$(n \div 10) \ mod \ 10$	2
$(n \div 100) \ mod \ 10$	4

En efecto, el residuo de dividir 421 por 10 es 1; si se divide (con división entera) 421 por 10 da 42 y si a ese resultado se saca el residuo de dividir por 10, da 2, etc.

A diferencia de las expresiones, los predicados son afirmaciones, falsas o verdaderas, que no representan ningún valor. Por ejemplo, "el color de la flor del árbol guayacán es rosado". Los predicados pueden combinarse con alguno de los siguientes operadores:

Operador	Símbolo en Rodin	Símbolo en matemáticas
Conjunción	$\&$	$\wedge$
Disyunción	or	$\vee$
Negación	not	$\neg$
Implicación	$=>$	$\Rightarrow$

Hay operadores que sirven para construir predicados a partir de dos expresiones. Por ejemplo, los operadores relacionales, $>, <, \geq, \leq, =$ construyen predicados a partir de dos expresiones aritméticas. La construcción $(x + y) * 5 + 1 < z - 3$ es un predicado. Ese predicado es verdadero si, por ejemplo, $x = 1, y = 2, z = 20$. Es falso, por ejemplo, cuando $x = 1, y = 1, z = 12$.

Ejemplo 1 *El predicado que afirma "la suma de a y b es impar", se escribe*

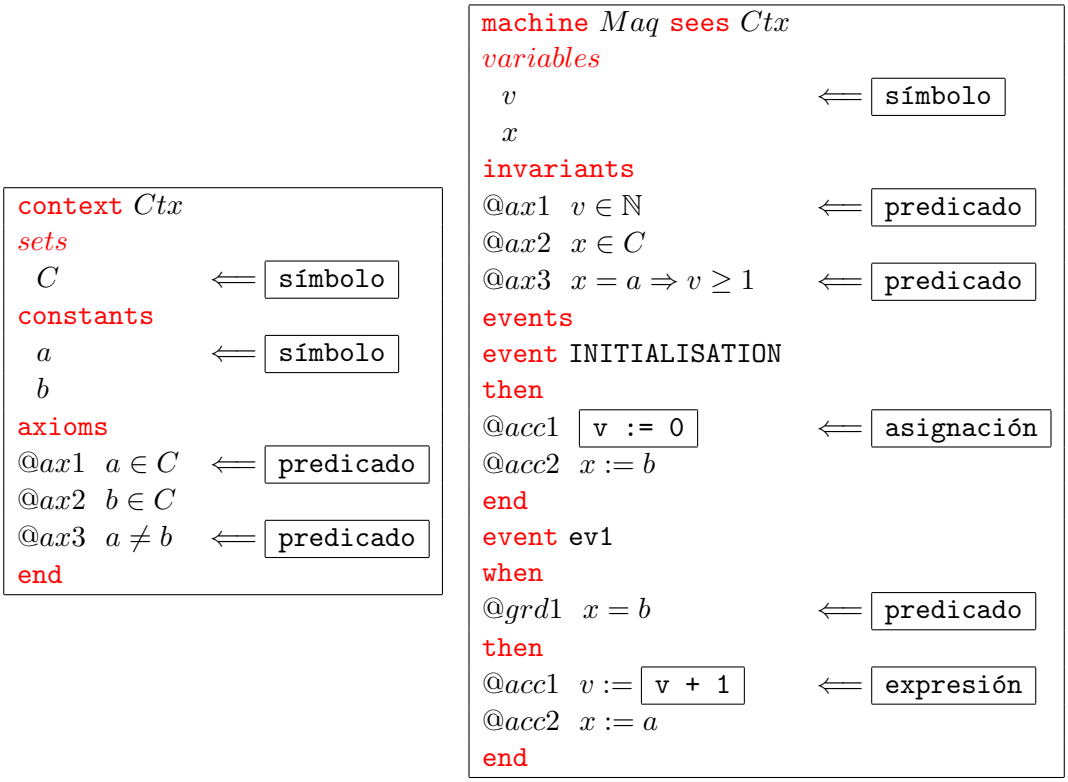
$$(a + b) \ mod \ 2 = 1$$

Ejemplo 2 *El predicado que afirma " la resta de c y d divide a r cuando z es impar", se escribe*

$$(z \ mod \ 2 = 1) \Rightarrow (\ (c - d) \ mod \ r = 0\ )$$

En el lenguaje *event B* , las expresiones se utilizan para calcular algún valor con el que se quiere modificar una observación. Por lo tanto, se pueden usar únicamente al lado derecho de las asignaciones en las acciones de los eventos. Por otro lado, los axiomas e invariantes definen propiedades que deben obedecer todas las observaciones de un sistema. Estas propiedades son predicados. En consecuencia, los predicados se usan en los axiomas del contexto, los invariantes de la máquina y las condiciones (guardas) de los eventos, como

se muestra a continuación:



2. Modelar colecciones

Los modelos que se han construido hasta ahora observan únicamente valores simples, generalmente números. Por ejemplo, la cantidad de pedalazos en el sistema de la bicicleta. Cuando se consideran sistemas más complejos, lo que se desea observar de ellos puede requerir otro tipo de valores. Por ejemplo, supongamos el sistema de transporte del MIO. Visto de manera global, lo que se observa en cada momento es que algún bus llega a alguna estación o que sale de ella. Puede interesar observar entonces exactamente cuáles buses son los que están detenidos y en cuáles estaciones. Es decir, nos interesa observar *colecciones* de objetos, sean buses o estaciones. En matemáticas existe una noción para representar colecciones. esta noción es la de *conjunto*.

Un conjunto se define simplemente como una colección de elementos. Los elementos pueden ser de naturaleza muy variada. Por ejemplo, pueden ser buses, estaciones, personas, puertas, etc., como se muestra en la figura 1.

El conjunto de buses puede representar los que están detenidos en alguna estación, el de estaciones las que tienen algún bus detenido en ellas y el de personas las que están en alguna estación, por ejemplo. Aunque los elementos de un conjunto pueden ser de cualquier

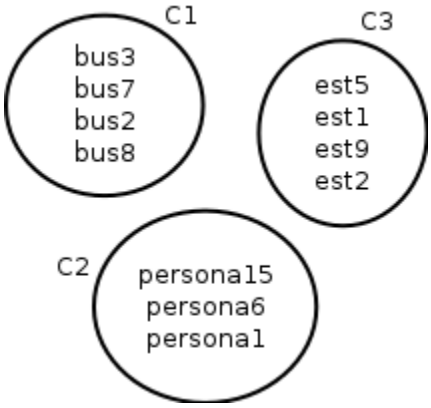


Figura 1: Conjuntos

naturaleza, es necesario que en un conjunto dado todos los elementos sean del mismo tipo. Es decir, no es válido tener un conjunto cuyos elementos sean personas y también buses. En la notación de *event B*, los conjuntos de la figura 1 se escriben:

$$C1 = \{bus3, bus7, bus2, bus8\}$$
$$C2 = \{persona15, persona6, persona1\}$$
$$C3 = \{est5, est1, est9, est2\}$$

El conjunto vacío $\varnothing$, que no tiene elementos, se escribe { } en $eventB$.

2.1. Operaciones sobre conjuntos

Se definen las siguientes tres operaciones sobre los conjuntos;

Operación	Símbolo en EventB	Símbolo matemático
Intersección	/\	$\cap$
Unión	\/	$\cup$
Diferencia	\	\

Cada una de estas operaciones toma como argumentos dos conjuntos y devuleve como resultado otro conjunto. Dados dos conjuntos A, B, la intersección $A \cap B$ da como valor resultante un conjunto con los elementos comunes a los conjuntos A y B. Por ejemplo, si $C1 = \{bus3, bus7, bus2, bus8\}$ y $C4 = \{bus2, bus13, bus7, bus1\}$, entonces $C1 \cap C4 = \{bus2, bus7\}$. La unión de dos conjuntos produce un conjunto con los elementos de los

dos. Por ejemplo, $C1 \cup C4 = \{bus3, bus7, bus2, bus8, bus13, bus1\}$. Note que un conjunto no puede tener elementos repetidos. La diferencia entre un conjunto A y un conjunto B, escrita $A \backslash B$, produce un conjunto con los elementos de A que no están en el conjunto B. Por ejemplo, $C1 \backslash C4 = \{bus3, bus8\}$. Hay que tener en cuenta que estas operaciones se deben hacer entre conjuntos *del mismo tipo*. Por ejemplo, para los conjuntos de la figura 1 sería un error escribir $C1 \cap C2$ porque sus elementos son de tipos diferentes. El conjunto $C1$ es de buses mientras que el conjunto $C2$ es de personas.

Un concepto importante con respecto a los conjuntos es su *cardinalidad*. Intuitivamente, la cardinalidad de un conjunto representa el número de elementos que contiene el conjunto. Por ejemplo, la cardinalidad del conjunto $C1$, que se escribe $card(C1)$, es igual a 6. Obviamente, para que esta noción de "número de elementos" tenga sentido, es necesario que el conjunto al que se le aplica sea *finito*. No es claro, por ejemplo, cuál sería el "número de elementos" del conjunto de todos los números naturales.

Las operaciones descritas permiten construir *expresiones* sobre conjuntos. Cada una de esas expresiones solamente puede realizarse entre dos conjuntos. Un ejemplo de expresión es $(C1 \cap C4) \cup \{bus12, bus4\}$, que da como resultado el conjunto $\{bus2, bus7, bus12, bus4\}$. Como son expresiones, solamente pueden aparecer en la parte derecha de asignaciones en los eventos de las máquinas en $eventB$.

2.2. Predicados sobre conjuntos

Hay también operadores que permiten construir predicados a partir de conjuntos. Ya habíamos utilizado el predicado de la forma $x \in C$, que afirma que el valor de la variable x es un elemento del conjunto C. También pueden formarse predicados con los operadores $\subset$ y $\subseteq$. Dados dos conjuntos A y B, el predicado $A \subset B$ afirma que el conjunto A es *subconjunto* del conjunto B. Es decir, que todos los elementos de A están también en B y que, además, hay algunos elementos de B que no están en A. Más concretamente, el predicado $A \subset B$ afirma que el conjunto A está contenido en B y además que esos dos conjuntos no son iguales. El predicado $A \subseteq B$, por otra parte, también afirma que A es subconjunto de B, pero permite que A sea igual a B. Por ejemplo, si $A = \{e1, e7, e5, e3\}$, $B = \{e1, e2, e3, e4, e5, e6, e7\}$ y $C = \{e1, e7, e3, e5\}$, entonces,

$A \subset B$ `es verdadero`
$A \subseteq B$ `es verdadero`
$A \subset C$ `es falso`
$A \subseteq C$ `es verdadero`

Existe también el operador $finite$ que afirma que un conjunto es finito. Por ejemplo, si $A = \{e1, e7, e5, e3\}$, el predicado $finite(A)$ afirma que el conjunto A tiene un número finito de elementos, lo que es verdadero. Por el contrario, el predicado $finite(\mathbb{N})$ es falso.

Un operador que se usa frecuentemente es el de cálculo del *conjunto potencia* o *conjunto de partes* de un conjunto dado. Para un conjunto C, la expresión $\mathbb{P}(C)$ construye

el conjunto de todos los subconjuntos de C. Por ejemplo, si $C = \{e1, e7, e3, \}$, entonces $\mathbb{P}(C) = \{\ \varnothing, \{e1\}, \{e2\}, \{e3\}, \{e1, e2\}, \{e1, e3\}, \{e2, e3\}, \{e1, e2, e3\}\ \}$. Note que cada elemento del conjunto potencia es a su vez un conjunto. La expresión $\mathbb{P}(C)$ se escribe `POW`(C) en $eventB$.

3. Ejemplo de modelo usando conjuntos

En esta sección construimos un modelo simplificado del sistema de transporte del MIO. En este sistema hay buses y estaciones. Los buses entran y salen de las estaciones. Hay un cierto número de buses y de estaciones. Suponemos que en cada estación solamente puede detenerse un bus en cada momento. Es decir, consideramos cada lado del paradero como una estación diferente. Nos interesa observar en cada momento cuál bus está detenido en cuál estación.

Suponemos en este sistema que los buses y las estaciones no varían. Es decir, observamos el funcionamiento del sistema, por ejemplo, durante un solo da, de manera que no observamos que se compren más buses, ni que se construyan estaciones. Los buses y las estaciones que hay son entonces *constantes* del sistema. Su definición se escribe en el contexto, como se muestra a continuación:

context $MioCtx$
sets
$ESTACIONES$
$BUSES$
constants
nb \\número de buses
ne \\número de estaciones
axioms
@$ax1$ $finite(ESTACIONES)$
@$ax2$ $finite(BUSES)$
@$ax3$ $ne \in \mathbb{N}_1$
@$ax4$ $nb \in \mathbb{N}_1$
@$ax5$ $card(ESTACIONES) = ne$
@$ax6$ $card(BUSES) = nb$
end

El contexto define dos conceptos o tipos, el conjunto de estaciones y el conjunto de buses. El conjunto $BUSES$ corresponde a la colección de todos los buses y el conjunto $ESTACIONES$ a la colección de todas las estaciones del sistema. Naturalmente, estos conjuntos son finitos (axiomas @ax1 y @ax2). El número de buses es igual a nb y el número de estaciones es igual a ne (axiomas @ax4 y @ax5). Los axiomas @ax3 y @ax4 afirman que el tipo de ne y de nb es el conjunto de números naturales diferentes de cero. El cero se

excluye porque si no hay buses o no hay estaciones, el sistema no tendría gran cosa para observar.

La máquina del sistema se muestra a continuación:

```
machine MioMaq sees MioCtx
variables
   busparq
   estparq
invariants
@inv1  busparq ⊆ BUSES
@inv2  estparq ⊆ ESTACIONES

events
event INITIALISATION
then
@acc1  busparq := ∅
@acc2  estparq := ∅
end
```

```
event llegar
any
   bus
   est
when
@grd1  bus ∈ BUSES
@grd2  bus ∉ busparq
@grd3  est ∈ ESTACIONES
@grd4  est ∉ estparq
then
@acc1  busparq := busparq ∪ {bus}
@acc2  estparq := estparq ∪ {est}
end
```

```
event salir
any
   bus
   est
when
@grd1  bus ∈ busparq
@grd2  est ∈ estparq
then
@acc1  busparq := busparq\{bus}
@acc2  estparq := estparq\{est}
end
```

La variable *busparq* permite observar en cada momento los buses que están detenidos en alguna estación. La variable *estparq* contiene las estaciones en las que hay algún bus detenido. El predicado $busparq \subseteq BUSES$ establece el tipo de *busparq*. Afirma que en todo momento la variable *busparq* contiene algún subconjunto de $ESTACIONES$. Similarmente, $estparq \subseteq ESTACIONES$ define que *estparq* contiene siempre algún subconjunto de $ESTACIONES$. El valor de cada variable es entonces una *colección* de objetos. Por esta razón, el valor inicial de cada una debe ser un conjunto. En este caso, se asigna el conjunto vacío. Es decir, inicialmente ningún bus está parqueado en ninguna estación.

El modelo define dos eventos, uno que representa la llegada de un bus a una estación y otro que representa cuando un bus sale de alguna estación. El evento "llegar" considera dos variables, *bus* y *est*. La primera representa un bus cualquiera y la segunda una estación

cualquiera. Las condiciones (o guardas) @grd1 y @grd3 de este evento definen el tipo de estas dos variables, indicando que su valor es algún elemento del conjunto de todos los buses y algún elemento del conjunto de todas las estaciones, respectivamente. La guarda @$grd2$ condiciona que el bus seleccionado no pertenezca a los buses parqueados. Es decir, asegura que el valor de la variable *bus* no es un bus que esté parqueado ya en alguna estación. La guarda @$grd4$ hace un control similar sobre la variable *est*. La acción @$acc1$ del evento agrega el valor de la variable *bus* al conjunto de los buses parqueados, mientras que la acción @$acc2$ agrega el valor de *est* a las estaciones que tienen algún bus parqueado. Por ejemplo, supongamos que antes de ejecutar el evento "llegar" tenemos $busparq = \{b7, b3, b5\}$ y $estparq = \{e1, e4, e7\}$. Las condiciones del evento garantizan que el bus seleccionado no sea ninguno de los de *busparq*. Supongamos entonces que $bus = b8$. Similarmente, supongamos que $est = e9$. Las acciones calcularían,

$\{b7, b3, b5\} \cup \{b8\} = \{b7, b3, b5, b8\}$ y también $\{e1, e4, e7\} \cup \{e9\} = \{e1, e4, e7, e9\}$. El resultado se asignaría como nuevo valor de *busparq* y *estparq*, respectivamente.

Las condiciones del evento "salir" garantizan, por el contrario, que el bus escogido sea uno de los que están parqueados y la estación una de las que tiene un bus parqueado. Las acciones eliminan ese bus de *busparq* y esa estación de *estparq*.