Introducción al Modelado de Sistemas
Capítulo 3

Camilo Rueda

21 de enero de 2016

1. Construir modelos numéricos de un sistema

Como se dijo antes, para ciertos sistemas puede ser adecuado construir modelos cuyas observaciones son numéricas. En este capítulo estudiamos varios ejemplos de sistemas y analizamos modelos numéricos para ellos.

1.1. Máquina distribuidora de gaseosas y comida

El primer sistema que vamos a considerar es el de una máquina que vende gaseosas y alimentos, como la que se muestra en la figura 1.

Figura 1: Máquina distribuidora de alimentos

La primera decisión que debe tomar el ingeniero es la de cuáles aspectos del sistema considerar relevantes. Esta decisión depende de los intereses particulares de quien diseña. La máquina de la figura 1 tiene muchos detalles que se relacionan con su funcionamiento.

Están, por ejemplo, los productos de que dispone y la manera en que se ubican, los botones para seleccionar los productos, la ranura en que se introducen las monedas, el sitio en que se entrega el cambio o se devuelven las monedas introducidas, el mecanismo que activa el brazo que empuja el producto seleccionado, el lugar en donde cae el producto, la pantalla que indica si la máquina está en servicio, la puerta que permite recargar los productos, la seguridad que esa puerta tiene, etc. Ocurre entonces queun sistema que parece sencillo, como el de esta máquina, puede requerir un modelo bastante complejo, dependiendo de los detalles que decidamos observar.

Es muy importante que, inicialmente, el ingeniero construya su modelo centrándose en unos pocos aspectos que le parecen fundamentales, dejando la observación de otros detalles para etapas posteriores. En este ejemplo, nos concentraremos únicamente en las *cantidades* de productos que hay en la máquina en cada momento, junto con las acciones que hacen que esas cantidades cambien.

Suponemos, para simplificar, que la máquina contiene dos clases de productos, que llamaremos *alimentos* y *gaseosas*. La capacidad que tiene la máquina de cada producto es fija: hay un cierto máximo número de *alimentos* y un cierto máximo número de *gaseosas* que caben en la máquina. Cada producto tiene un cierto precio, que suponemos que no cambia.

El funcionamiento de la máquina es simple. La persona introduce el precio del producto que quiere y la máquina lo entrega. Esto solamente ocurre si hay al menos una unidad del producto que quiere la persona. Si no hay disponibilidad de ese producto, la máquina simplemente devuelve las monedas. Cuando la máquina no tiene ya ninguno de los productos, un operario llega y rellena la máquina con la capacidad de

Lo que elegimos observar son tres cosas: la cantidad disponible de cada producto, y el acumulado de ganancias por ventas. Cuando el operario recarga la máquina, se lleva también la plata de ganancias..

1.1.1. El contexto de la máquina distribuidora

En la descripción que se dió del sistema se pueden identificar cuatro valores que van a permanecer fijos a lo largo de la operación de la máquina:

1. El número máximo de alimentos que cabe en la máquina
2. El número máximo de gaseosas que cabe en la máquina
3. El precio de una gaseosa
4. El precio de un alimento

El *contexto* del modelo define estas constantes, como se muestra en la figura 2.

El tipo de cada constante es $\mathbb{N}_1$, que corresponde al conjunto de números $\{1, 2, 3, 4, 5, ...\}$, que no incluye el cero. El valor de esas constantes no puede ser igual a cero porque si lo

```
CONTEXT
  distribCtx
CONSTANTS
  maxAlim      //   Número máximo de alimentos
  maxGas       //   Número máximo de gaseosas
  precioAlim   //   Precio de un alimento
  precioGas    //   Precio de una gaseosa
AXIOMS
  axm1   :   maxAlim ∈ N1
  axm2   :   maxGas ∈ N1
  axm3   :   precioAlim ∈ N1
  axm4   :   precioGas ∈ N1
END
```

Figura 2: Modelo de distribuidora de alimentos: el contexto

fuera el sistema no podría funcionar correctamente. Por ejemplo, si el número máximo de gaseosas es igual a cero, entonces a la máquina no se le pueden agregar gaseosas nunca, lo que contradice la descripción del sistema.

1.1.2. El modelo de la máquina distribuidora

La parte dinámica del sistema de la máquina distribuidora tiene tres tipos de observaciones:

1. La cantidad disponible de alimentos
2. La cantidad disponible de gaseosas
3. Las ganancias

Estas observaciones se identifican mediante tres variables, $cantAlim, cantGas, ganancias$. En un momento dado, por ejemplo, podríamos tener la observación (el estado):

$$\boxed{\begin{array}{l} cantAlim = 12 \\ cantGas = 7 \\ ganancias = 5800 \end{array}}$$

que indica que restan en la máquina 12 alimentos y 7 gaseosas. Además, que hasta ese momento las ganancias son de 5800 pesos.

En la figura 3 se muestra la definición de las variables del modelo.

El invariante define el tipo de cada variable como $\mathbb{N}$, es decir, los números $\{0, 1, 2, 3, 4, ...\}$, que incluyen el cero. El cero se incluye porque en un momento dado puede que se hayan acabado todas las unidades de un producto. Por ejemplo, puede que ya no haya gaseosas, y en ese caso $cantGas = 0$.

```
MACHINE
  distMaq
SEES
  distribCtx
VARIABLES
  cantAlim      //   cantidad disponible de alimentos
  cantGas      //   cantidad disponible de gaseosas
  ganancias      //   ganancias actuales
INVARIANTS
  inv1   :   cantAlim ∈ ℕ
  inv2   :   cantGas ∈ ℕ
  inv3   :   ganancias ∈ ℕ
EVENTS
  INITIALISATION   ≙
  STATUS
   ordinary
  BEGIN
   act1   :   cantAlim ≔ maxAlim
   act2   :   cantGas ≔ maxGas
   act3   :   ganancias ≔ 0
  END
```

Figura 3: Modelo de distribuidora de alimentos: definición de variables

La inicialización del modelo determina que el sistema empieza con una máquina llena: la cantidad de alimentos y de gaseosas es igual al máximo posible. Inicialmente no hay ganancias.

Lo siguiente que debemos considerar son las acciones que ocasionan modificaciones en las observaciones. La cantidad disponible de alimentos puede variar si alguien compra uno. Similarmente para la cantidad de gaseosas. Estas cantidades también pueden variar si el operario vuelve a llenar la máquina con los productos. Cuando alguien compra una gaseosa, la cantidad disponible de gaseosas debe disminuir en uno. También deben incrementarse las ganancias en el precio de una gaseosa. Igual ocurre cuando se compra un alimento.

La figura 4 muestra los eventos del sistema.

Las condiciones de los eventos de compra, tanto de alimentos como de gaseosas, requiere que haya al menos una unidad. Esto se controla en las guardas (o condiciones) del evento. Por ejemplo, la condición $cantAlim > 0$ exige que el número de unidades de alimentos sea *estrictamente* mayor a cero.

El evento *rellenar* corresponde a la acción de un operario que vuelve a cargar de productos la máquina. Las guardas de ese evento, $cantAlim = 0$ y $cantGas = 0$, exigen que esto ocurra solamente cuando no hay disponibilidad de ninguno de los productos. La máquina se rellena entonces con el máximo posible de cada tipo de producto. El operario saca también la plata de la máquina, luego las ganancias vuelven a estar en cero.

```
comprarAlim   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantAlim > 0
THEN
 act1   :   cantAlim ≔ cantAlim − 1
 act2   :   ganancias ≔ ganancias + precioAlim
END

comprarGaseosa   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantGas > 0
THEN
 act1   :   cantGas ≔ cantGas − 1
 act2   :   ganancias ≔ ganancias + precioGas
END

rellenar   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantAlim = 0
 grd2   :   cantGas = 0
THEN
 act1   :   cantAlim ≔ maxAlim
 act2   :   cantGas ≔ maxGas
 act3   :   ganancias ≔ 0
END
```

Figura 4: Modelo de distribuidora de alimentos: los eventos

1.1.3. Agregar Monedas

Una gran simplificación del modelo que hemos construido es que no observa la introducción de las monedas. Es decir, el modelo supone que el pago y la obtención del producto se hacen en un solo paso. Los eventos *comprarGaseosa* y *comprarAlim* distribuyen el producto, cuando hay, sin chequear el pago, porque *se supone* que en cada compra se introduce además el precio del producto. Desde luego que una máquina distribuidora de productos no opera de esta manera. El modelo que hemos hecho opta por *ignorar* este detalle. Otra forma de decirlo es que el modelo desarrollado hasta ahora *hace abstracción* del proceso de introducción de las monedas. Este tipo de modelos los llamamos *abstractos* por esta razón.

Es muy importante construir inicialmente modelos abstractos de un sistema porque eso permite concentrarse en unos pocos aspectos que se consideran fundamentales. Observando ese modelo simple nos aseguramos de entender el comportamiento del sistema en lo que corresponde a los pocos aspectos que decidimos incluir en el modelo. Solamente después de haber garantizado este entendimiento se deben considerar otros detalles del sistema.

Procedemos ahora a considerar la introducción de las monedas. Lo que se hará por ahora es *modificar* el modelo ya construido. Desde luego que hacer esto tiene la desventaja de que podríamos entonces, por algún error, cambiar las observaciones que ya habíamos logrado entender. Es decir, podríamos obtener valores para $maxAlim$, o para las otras variables, que no corresponden a los que observábamos antes.

En efecto, un modelo completo de un sistema no se construye modificando un modelo abstracto ya hecho, sino *extendiéndolo* de manera que no se modifique nada de lo que anteriormente se había construido. Más adelante estudiaremos cómo hacer esto. Por ahora, para simplificar la explicación, aceptaremos modificar el modelo que ya teníamos. El pro-

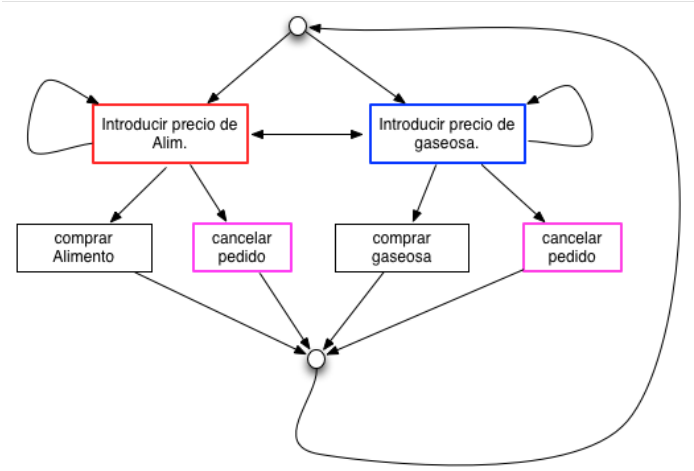

Figura 5: Proceso de pedidos a la máquina

ceso que se quiere modelar debe entonces considerar una nueva acción, *introducir dinero*. Suponemos que cada vez que se introduce dinero en la máquina es una cantidad que es exactamente igual al precio de una gaseosa o al precio de un alimento. Es decir, por ahora no consideramos que puedan introducirse cantidades menores o mayores a esos precios.

El proceso de pedir algo a la máquina sería entonces como se muestra en el ejemplo de la figura 5.

En un momento dado, la persona puede introducir el precio de una gaseosa o de un alimento. Enseguida, puede, sea comprar el producto, sea cancelar. En este último caso las monedas se devuelven. Después puede recomenzar. También puede ingresar varias veces el precio de un producto. Un producto se puede comprar solamente si ya se ha introducido suficiente dinero para pagar su precio. Las acciones nuevas son las que aparecen en colores en la figura 5. Hay también una nueva observación: la cantidad de dinero que se ha introducido. En resumen, lo nuevo es:

`Cantidad de dinero introducida:`	`Nueva variable`
`Introducir precio de Alim:`	`Nuevo evento`
`Introducir precio de gaseosa:`	`Nuevo evento`
`Cancelar:`	`Nuevo evento`

Como no hay nuevas constantes que se deban considerar, el contexto del modelo es el mismo. Las nuevas variables se muestran en la figura 6

Los eventos del modelo se muestran en la figura 7. Los eventos de compra de productos se han modificado con una nueva guarda y una nueva acción. En el evento *comprarAlim*, por ejemplo, se agregó la guarda $plata \geq precioAlim$. Esta condición exige que se haya introducido al menos tanto dinero como el precio de un alimento. La acción $plata := plata - precioAlim$ simplemente le descuenta a la persona, de la plata que haya introducido, el precio de un alimento. El evento *comprarGaseosa* se modificó de manera similar.

La guarda del evento nuevo *plataAlim* deja introducir el dinero del precio de un alimento solamente si hay disponibilidad de ese alimento. Similarmente para el evento *plataGas*. El evento *cancelar* devuelve el dinero a cero, que representa devolverle la plata a la persona.

```
MACHINE
  distrMaq
SEES
  distrCtx
VARIABLES
  cantAlim      //   cantidad disponible de alimentos
  cantGas      //   cantidad disponible de gaseosas
  ganancias      //   ganancias actuales
  plata      //   dinero introducido
INVARIANTS
  inv1   :   cantAlim ∈ N
  inv2   :   cantGas ∈ N
  inv3   :   ganancias ∈ N
  inv4   :   plata ∈ N
EVENTS
  INITIALISATION   ≙
  STATUS
   ordinary
  BEGIN
   act1   :   cantAlim ≔ maxAlim
   act2   :   cantGas ≔ maxGas
   act3   :   ganancias ≔ 0
   act4   :   plata ≔ 0
  END
```

Figura 6: Variables del modelo con introducción de dinero

```
rellenar   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantAlim = 0
 grd2   :   cantGas = 0
THEN
 act1   :   cantGas ≔ maxGas
 act2   :   cantAlim ≔ maxAlim
 act3   :   ganancias ≔ 0
END

plataAlim   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantAlim > 0
THEN
 act1   :   plata ≔ plata + precioAlim
END

plataGas   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantGas > 0
THEN
 act1   :   plata ≔ plata + precioGas
END

cancelar   ≜
STATUS
 ordinary
WHEN
 grd1   :   plata > 0
THEN
 act1   :   plata ≔ 0
END

comprarAlim   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantAlim > 0
 grd2   :   plata ≥ precioAlim
THEN
 act1   :   cantAlim ≔ cantAlim − 1
 act2   :   ganancias ≔ ganancias + precioAlim
 act3   :   plata ≔ plata − precioAlim
END

comprarGaseosa   ≜
STATUS
 ordinary
WHEN
 grd1   :   cantGas > 0
 grd2   :   plata ≥ precioGas
THEN
 act1   :   cantGas ≔ cantGas − 1
 act2   :   ganancias ≔ ganancias + precioGas
 act3   :   plata ≔ plata − precioGas
END
```

Figura 7: Eventos del modelo con introducción de dinero