EventB Exercises 3 (extended)
Supermarket

The objective of this set of tutorial exercises is to develop a specification of a simple supermarket.

1 The machines

1.1 The Supermarket_ctx machine

This context machine models the “things” that you find in a supermarket.

CONTEXT Supermarket_ctx

SETS

TROLLEY
PRODUCT

CONSTANTS

SHELF
PRICE
Milk
Cheese
Cereal
BASKET
ProdInTrolleys
maxtrolley

AXIOMS

\[ \text{axm}7: \quad \text{PRICE} = \mathbb{N} \]
\[ \text{axm}1: \quad \text{SHELF} = \text{PRODUCT} \rightarrow \mathbb{N} \]
\[ \text{axm}2: \quad \text{PRODUCT} = \{\text{Milk, Cheese, Cereal}\} \]
\[ \text{axm}3: \quad \text{Milk} \neq \text{Cheese} \]
\[ \text{axm}4: \quad \text{Milk} \neq \text{Cereal} \]
\[ \text{axm}5: \quad \text{Cheese} \neq \text{Cereal} \]
\[ \text{axm}6: \quad \text{BASKET} = \text{PRODUCT} \rightarrow \mathbb{N}_1 \]
\[ \text{axm}8: \quad \text{ProdInTrolleys} \in \text{PRODUCT} \times (\text{TROLLEY} \rightarrow \text{BASKET}) \rightarrow \mathbb{N} \]
\[ \text{axm}9: \quad \forall p, ts, t \cdot p \in \text{PRODUCT} \land ts \in \text{TROLLEY} \rightarrow \text{BASKET} \land t \in \text{dom}(ts) \rightarrow \text{ProdInTrolleys}(p \mapsto ts) = 0 \]
\[ \text{axm}10: \quad \text{maxtrolley} \in \mathbb{N}_1 \]
1.2 The Supermarket machine

For the supermarket we want to model the products in the supermarket, the shelf containing the
products, the trolleys available for customers, the customers with trolleys and products in those
trolleys.

Important: all products on the shelves of the supermarket and in the trolleys must have a
price.

Here is part of the Supermarket machine.

MACHINE Supermarket
SEES Supermarket_ctx

VARIABLES

shelf The supermarket shelf containing products
trolleys Trolley identities
products products with known prices
price total function from products to a price
customers customers with trolleys
topay amount a customer —represented by a trolley— needs to pay
stock all product in the supermarket: on shelf and in trolleys
reorderlevel level at which a product needs to be re-ordered
reorder a set of all products requiring re-ordering
onorder record reordering of product

INVARINTS

inv1: shelf ∈ SHELF
1.2 The Supermarket machine

\textbf{inv2:} trolleys \subseteq TROLLEY
\textbf{inv3:} finite(trolleys)
\textbf{inv4:} products \subseteq PRODUCT
\textbf{inv5:} finite(products)
\textbf{inv6:} price \in products \rightarrow PRICE
\textbf{inv7:} dom(shelf) = products
\textbf{inv8:} customers \in trolleys \rightarrow BASKET
\textbf{inv9:} \forall t \cdot t \in trolleys \Rightarrow dom(customers(t)) \subseteq products
\textbf{inv10:} topay \in trolleys \rightarrow \mathbb{N}
\textbf{inv11:} stock \in products \rightarrow \mathbb{N}
\textbf{inv12:} \forall p \cdot p \in products
\Rightarrow stock(p) = shelf(p) + ProdInTrolleys(p \mapsto customers)
\textbf{inv13:} reorderlevel \in products \rightarrow \mathbb{N}_1
\textbf{inv14:} reorder \subseteq products
\textbf{inv15:} onorder \subseteq products

\textbf{EVENTS}

\textit{Initialisation}

\textbf{begin act1:} shelf := \emptyset
\textbf{act2:} trolleys := \emptyset
\textbf{act3:} products := \emptyset
\textbf{act4:} price := \emptyset
\textbf{act5:} customers := \emptyset
\textbf{act6:} reorderlevel := \emptyset
\textbf{act7:} reorder := \emptyset
\textbf{act8:} topay := \emptyset
\textbf{act9:} stock := \emptyset
\textbf{act10:} onorder := \emptyset
\textbf{end}

The above machine is intended to model:

- products on the shelf of the supermarket
- products in customer trolleys
- total stock of products: note that stock includes all products that are still in the supermarket, either on the shelf, in customers' trolleys or perhaps in reserve somewhere else in the supermarket.
- checkout
• stock alert when stock level drops below some minimum requirement

Complete the Initialisation and add the following events:

**Setprice** set the *price of a product*;

**AddStock** add some *amount of product* to the supermarket *stock*;

**AddProductShelf** add some *amount of product* to the shelf of the supermarket;

**GetTrolley** get a vacant *trolley*;

**AddProductTrolley** take some *amount of product* on shelf and add to *trolley*;

**RemProductTrolley** take some *amount of product* from *trolley* and return to shelf.

**SetMinStock** set the minimum *amount of product* to have in stock;

**CheckOut** checkout *product* from *trolley*;

**Pay** pay for products in *trolley*;

**ReturnTrolley** return empty *trolley*;

**ReStock** indicate that stock of *product* has fallen below minimum stock level.

### 1.3 Refinement of Supermarket Machine

Refine the Supermarket machine, especially showing two methods of implementing CheckOut: one allowing multiple product items to be processed and the other processing one product items at a time.

Events that don’t change can be simply inherited using the mysterious first menu on the event line in Rodin.