Contents

1 Routes **2**
1.1 Narrative 2
1.2 Basic Types and System State 2
1.3 Choose Route Operation 2

2 Stack **3**
2.1 Narrative 3
2.2 Basic Types and System State 3
2.3 Operations 3

1 Routes

1.1 Narrative

This system describes a airport connection system. The system state simply models the airports that are connected. So, dub is connected to cork, cork is connected to paris,etc. The operation we specify is, given a starting point (from?) and ending point (to?) to specify a possible route between these two AIRPORTs . Note the nature of the specification of the operation. The specification of the proposed route (route!) just states what rules this route! obeys, i.e. that each adjacent member of the sequence is connected according to the connected relation.

1.2 Basic Types and System State

$$[AIRPORT]$$

AirSystem

$$connected : AIRPORT \leftrightarrow AIRPORT$$

$$connected = \{dub \mapsto cork, cork \mapsto paris, cork \mapsto dub, paris \mapsto london, london \mapsto rome\}$$
$$route! = \langle dub, cork, paris, london, rome \rangle$$

1.3 Choose Route Operation

ChoseRoute

$$\Xi AirSystem$$
$$to?, from? : AIRPORT$$
$$route! : \text{iseq}\, AIRPORT$$

$$(from?, to?) \in connected^+$$
$$\forall i : 1..(\#route! - 1) \bullet (route!(i), route!(i+1)) \in connected$$

Note the precondition that states that there is an indirect connection between the starting point and destination. Also, note the use of the injective sequence iseq in the declaration of the *route*! sequence. This ensures that no AIRPORT will be visited more than once.

2 Stack

2.1 Narrative

This is the specification of a simple stack system. There is no limit on the size of the stack. We specify the system state and the two operations push and pop.

2.2 Basic Types and System State

$$[X]$$

Stack

$$s : \operatorname{seq} X$$

InitStack

$$Stack'$$

$$s' = \langle\,\rangle$$

2.3 Operations

Push

$$\Delta Stack$$
$$x? : X$$

$$s' = s \frown \langle x? \rangle$$

Pop

$$\Delta Stack$$
$$x! : X$$

$$s = s' \frown \langle x! \rangle$$

Please note that you could chose whether to 'push' the element at the end of the sequence or the start of the sequence as long as you take this into account when 'popping'. Also please note that the specification of a Queue system would be very similar, the main difference being that the queue's version of pushing(enqueue) happens at the other end of the sequence from its version of popping (dequeue).