

```

\ * Copyright (c) 2018, Backyard Innovations Pte. Ltd., Singapore.
\ *
\ * Released under the terms of the Apache License 2.0
\ * See: file LICENSE in root directory for details.
\ *
\ * This file contains Intellectual Property that belongs to
\ * Backyard Innovations Pte Ltd., Singapore.
\ *
\ * Authors: Santhosh Raju < santhosh@byisystems.com >
\ *          Cherry G. Mathew < cherry@byisystems.com >
\ *          Fransisca Andriani < sisca@byisystems.com >
\ *

```

---

MODULE *VoucherLifeCycle*

---

This specification is of a Voucher and it's life cycle. This is based on the definition of Vouchers in *RFC 3506* with the tuple part decoupled.

Note: A new state called “phantom” was introduced to indicate the state of a voucher that is yet to be issued, once a voucher is issued it becomes a “valid” voucher. This is a one way transition and it cannot be reversed.

CONSTANT $V$	The set of vouchers.
VARIABLE $vState$ ,	$vState[v]$ is the state of a voucher $v$ .
$vlcState$	The state of the voucher life cycle machine.
	$vlcState[v]$ is the state of the life cycle machine for the voucher $v$ .

---

$VTypeOK \triangleq$

The type-correctness invariant

$$\wedge vState \in [V \rightarrow \{\text{“phantom”}, \text{“valid”}, \text{“redeemed”}, \text{“cancelled”}\}]$$

$$\wedge vlcState \in [V \rightarrow \{\text{“init”}, \text{“working”}, \text{“done”}\}]$$

$VInit \triangleq$

The initial predicate.

$$\wedge vState = [v \in V \mapsto \text{“phantom”}]$$

$$\wedge vlcState = [v \in V \mapsto \text{“init”}]$$


---

We now define the actions that may be performed on the  $Vs$ , and then define the complete next-state action of the specification to be the disjunction of the possible  $V$  actions.

$Issue(v) \triangleq$

$$\wedge vState[v] = \text{“phantom”}$$

$$\wedge vlcState[v] = \text{“init”}$$

$$\wedge vState' = [vState \text{ EXCEPT } ![v] = \text{“valid”}]$$

$$\wedge vlcState' = [vlcState \text{ EXCEPT } ![v] = \text{“working”}]$$

$Transfer(v) \triangleq$

$$\wedge vState[v] = \text{“valid”}$$

$$\wedge \text{UNCHANGED } \langle vState, vlcState \rangle$$

$Redeem(v) \triangleq$   
 $\wedge vState[v] = \text{"valid"}$   
 $\wedge vlcState[v] = \text{"working"}$   
 $\wedge vState' = [vState \text{ EXCEPT } ![v] = \text{"redeemed"}]$   
 $\wedge vlcState' = [vlcState \text{ EXCEPT } ![v] = \text{"done"}]$

$Cancel(v) \triangleq$   
 $\wedge vState[v] = \text{"valid"}$   
 $\wedge vlcState[v] = \text{"working"}$   
 $\wedge vState' = [vState \text{ EXCEPT } ![v] = \text{"cancelled"}]$   
 $\wedge vlcState' = [vlcState \text{ EXCEPT } ![v] = \text{"done"}]$

$VNext \triangleq \exists v \in V : Issue(v) \vee Redeem(v) \vee Transfer(v) \vee Cancel(v)$

The next-state action.

$VConsistent \triangleq$

A state predicate asserting that a  $V$  started at a valid start state and has reached a valid final state at the end of the life cycle.  $V$  can be "valid" only when the state of the machine is "working". It is an invariant of the specification.

$\wedge \forall v \in V : \vee \wedge vlcState[v] = \text{"done"}$   
 $\wedge vState[v] \in \{ \text{"redeemed"}, \text{"cancelled"} \}$   
 $\vee \wedge vlcState[v] = \text{"init"}$   
 $\wedge vState[v] = \text{"phantom"}$   
 $\vee \wedge vlcState[v] = \text{"working"}$   
 $\wedge vState[v] \in \{ \text{"valid"} \}$

$VSpec \triangleq VInit \wedge \Box [VNext]_{\langle vState, vlcState \rangle}$

The complete specification of the protocol written as a temporal formula.

THEOREM  $VSpec \Rightarrow \Box (VTypeOK \wedge VConsistent)$

This theorem asserts the truth of the temporal formula whose meaning is that the state predicate  $VTypeOK \wedge VConsistent$  is an invariant of the specification  $VSpec$ . Invariance of this conjunction is equivalent to invariance of both of the formulas  $VTypeOK$  and  $VConsistent$ .

$\backslash$  \* Modification History  
 $\backslash$  \* Last modified *Tue Jun 12 13:25:29 IST 2018* by Fox  
 $\backslash$  \* Created *Fri Mar 16 11:56:25 SGT 2018* by Fox