MODULE *Transaction*

INSTANCE *Naturals*

INSTANCE *FiniteSets*

INSTANCE *Sequences*

INSTANCE *TLC*

An empty constant
CONSTANT *Nil*

Transaction phase constants
CONSTANTS
Change,
Rollback

Proposal phase constants
CONSTANTS
Commit,
Apply

Status constants
CONSTANTS
Pending,
Complete,
Canceled,
Aborted,
Failed

$Status \triangleq \{Pending, Complete, Canceled, Aborted, Failed\}$

$Done \triangleq \{Complete, Canceled, Aborted, Failed\}$

The set of all nodes
CONSTANT *Node*

$Empty \triangleq [p \in \{\} \mapsto Nil]$

Variables defined by other modules.
VARIABLES
configuration,
mastership,
conn,
target

A transaction log. Transactions may either request a set
of changes to a set of targets or rollback a prior change.

VARIABLE *transaction*

A proposal log.

VARIABLE *proposal*

A sequence of configuration changes used for model checking.

VARIABLE *history*

$$
\begin{array}{l}
\mathit{TransactionOK} \triangleq \\
\quad \forall\, i \in \text{DOMAIN } \mathit{transaction} : \\
\qquad \land\ \mathit{transaction}[i].\mathit{phase} \in \{\mathit{Change}, \mathit{Rollback}\} \\
\qquad \land\ \mathit{transaction}[i].\mathit{change}.\mathit{proposal} \in \mathit{Nat} \\
\qquad \land\ \mathit{transaction}[i].\mathit{change}.\mathit{revision} \in \mathit{Nat} \\
\qquad \land\ \forall\, p \in \text{DOMAIN } \mathit{transaction}[i].\mathit{change}.\mathit{values} : \\
\qquad\quad \mathit{transaction}[i].\mathit{change}.\mathit{values}[p] \neq \mathit{Nil} \Rightarrow \\
\qquad\qquad \mathit{transaction}[i].\mathit{change}.\mathit{values}[p] \in \text{STRING} \\
\qquad \land\ \mathit{transaction}[i].\mathit{rollback}.\mathit{proposal} \in \mathit{Nat} \\
\qquad \land\ \mathit{transaction}[i].\mathit{rollback}.\mathit{revision} \in \mathit{Nat} \\
\qquad \land\ \forall\, p \in \text{DOMAIN } \mathit{transaction}[i].\mathit{rollback}.\mathit{values} : \\
\qquad\quad \mathit{transaction}[i].\mathit{rollback}.\mathit{values}[p] \neq \mathit{Nil} \Rightarrow \\
\qquad\qquad \mathit{transaction}[i].\mathit{rollback}.\mathit{values}[p] \in \text{STRING}
\end{array}
$$

$$
\begin{array}{l}
\mathit{ProposalOK} \triangleq \\
\quad \forall\, i \in \text{DOMAIN } \mathit{proposal} : \\
\qquad \land\ \mathit{proposal}[i].\mathit{transaction} \in \mathit{Nat} \\
\qquad \land\ \mathit{proposal}[i].\mathit{commit} \in \mathit{Status} \\
\qquad \land\ \mathit{proposal}[i].\mathit{apply} \in \mathit{Status}
\end{array}
$$

$$\mathit{TypeOK} \triangleq \mathit{TransactionOK} \land \mathit{ProposalOK}$$

$$
\begin{array}{l}
\text{LOCAL } \mathit{State} \triangleq [\\
\quad \mathit{transactions} \mapsto [i \in \text{DOMAIN } \mathit{transaction} \mapsto \mathit{transaction}[i] \mathbin{@@} [\mathit{index} \mapsto i]], \\
\quad \mathit{proposals} \mapsto [i \in \text{DOMAIN } \mathit{proposal} \mapsto \mathit{proposal}[i] \mathbin{@@} [\mathit{index} \mapsto i]], \\
\quad \mathit{configuration} \mapsto \mathit{configuration}]
\end{array}
$$

$$
\begin{array}{l}
\text{LOCAL } \mathit{Transitions} \triangleq \\
\quad \text{LET} \\
\qquad \mathit{transactions} \triangleq \{i \in \text{DOMAIN } \mathit{transaction}' : \\
\qquad\qquad\qquad i \in \text{DOMAIN } \mathit{transaction} \Rightarrow \mathit{transaction}'[i] \neq \mathit{transaction}[i]\} \\
\qquad \mathit{proposals} \triangleq \{i \in \text{DOMAIN } \mathit{proposal}' : \\
\qquad\qquad\qquad i \in \text{DOMAIN } \mathit{proposal} \Rightarrow \mathit{proposal}'[i] \neq \mathit{proposal}[i]\} \\
\quad \text{IN} \\
\qquad [\mathit{transactions} \mapsto [i \in \mathit{transactions} \mapsto \mathit{transaction}'[i] \mathbin{@@} [\mathit{index} \mapsto i]], \\
\qquad\ \mathit{proposals} \mapsto [i \in \mathit{proposals} \mapsto \mathit{proposal}'[i] \mathbin{@@} [\mathit{index} \mapsto i]]]
\end{array}
$$

$Test \triangleq$ INSTANCE $Test$ WITH
  $File \leftarrow$ "Transaction.log"

This section models configuration changes and rollbacks. Changes are appended to the transaction log and processed asynchronously.

Add a set of changes 'c' to the transaction log

$AppendChange(p,\ v) \triangleq$
  $\land\ transaction' = Append(transaction,\ [$
                $phase \mapsto Change,$
                $change \mapsto [$
                  $proposal \mapsto 0,$
                  $revision \mapsto Len(transaction) + 1,$
                  $values \mapsto (p :> v)],$
                $rollback \mapsto [$
                  $proposal \mapsto 0,$
                  $revision \mapsto 0,$
                  $values \mapsto Empty]])$
  $\land$ UNCHANGED $\langle proposal,\ configuration,\ mastership,\ conn,\ target,\ history\rangle$

Add a rollback of transaction 't' to the transaction log

$RollbackChange(i) \triangleq$
  $\land\ i \in$ DOMAIN $transaction$
  $\land\ transaction[i].phase = Change$
  $\land\ transaction[i].change.proposal \neq 0$
  $\land\ proposal[transaction[i].change.proposal].commit \neq Pending$
  $\land\ transaction' = [transaction$ EXCEPT $![i].phase = Rollback]$
  $\land$ UNCHANGED $\langle proposal,\ configuration,\ mastership,\ conn,\ target,\ history\rangle$

$ReconcileChange(n,\ i) \triangleq$
    The change proposal has not yet been created.
  $\land\ \lor\ \land\ transaction[i].change.proposal \notin$ DOMAIN $proposal$
      The prior transaction must have created a change proposal.
    $\land\ i - 1 \in$ DOMAIN $transaction \Rightarrow transaction[i-1].change.proposal \in$ DOMAIN $proposal$
    $\land\ proposal' = Append(proposal,\ [transaction \mapsto i,\ commit \mapsto Pending,\ apply \mapsto Pending])$
    $\land\ transaction' = [transaction$ EXCEPT $![i].change.proposal = Len(proposal')]$
    $\land$ UNCHANGED $\langle configuration,\ target,\ history\rangle$
    The change proposal has been created.
  $\lor\ \land\ transaction[i].change.proposal \in$ DOMAIN $proposal$
      The change is pending commit. Validate and commit the change once the prior change has been committed.
    $\land\ \lor\ \land\ proposal[transaction[i].change.proposal].commit = Pending$
        The prior proposal has been committed.

$\land$ $transaction[i].change.proposal - 1 \in$ DOMAIN $proposal \Rightarrow$
$proposal[transaction[i].change.proposal - 1].commit \in Done$

The prior change has been committed.

$\land$ $configuration.committed.index = i - 1$

Valid change is committed to the configuration.

$\land$ $\lor$ $\land$ $transaction' = [transaction$ EXCEPT $![i].rollback.revision = configuration.committed.revis$
$![i].rollback.values = [$
$p \in$ DOMAIN $transaction[i].change.values \mapsto$
IF $p \in$ DOMAIN $configuration.committed.values$
$configuration.committed.values[p]$
ELSE
$Nil]]$
$\land$ $configuration' = [configuration$ EXCEPT $!.committed.index = i,$
$!.committed.revision = i,$
$!.committed.values = transaction[i].change.val$
$configuration.committ$
$\land$ $proposal' = [proposal$ EXCEPT $![transaction[i].change.proposal].commit = Complete]$
$\land$ $history' = Append(history, [type \mapsto Change, phase \mapsto Commit, index \mapsto i])$

The change is invalid. Mark the proposal *Failed*.

$\lor$ $\land$ $configuration' = [configuration$ EXCEPT $!.committed.index = i]$
$\land$ $proposal' = [proposal$ EXCEPT $![transaction[i].change.proposal].commit = Failed]$
$\land$ UNCHANGED $\langle transaction, history\rangle$

$\land$ UNCHANGED $\langle target\rangle$

The change was committed and apply is pending.

$\lor$ $\land$ $proposal[transaction[i].change.proposal].apply = Pending$
$\land$ $proposal[transaction[i].change.proposal].commit \in Done$

The prior proposal has been applied.

$\land$ $transaction[i].change.proposal - 1 \in$ DOMAIN $proposal \Rightarrow$
$proposal[transaction[i].change.proposal - 1].apply \in Done$

If the prior change completed apply, or if apply failed the prior change has been rolled back, apply this change.

$\land$ $\lor$ $\land$ $configuration.applied.index = i - 1$

The change was committed successfully. Apply the change.

$\land$ $\lor$ $\land$ $proposal[transaction[i].change.proposal].commit = Complete$
$\land$ $configuration.state = Complete$
$\land$ $configuration.term = mastership.term$
$\land$ $conn[n].id = mastership.conn$
$\land$ $conn[n].connected$
$\land$ $target.running$

The change is successfully applied to the target.

$\land$ $\lor$ $\land$ $target' = [target$ EXCEPT $!.values = transaction[i].change.values$ @@ $target.valu$
$\land$ $configuration' = [configuration$ EXCEPT $!.applied.index = i,$
$!.applied.revision = i,$
$!.applied.values = transaction[i].cha$
$configuration.a$

$\land\ proposal' = [proposal \text{ EXCEPT } ![transaction[i].change.proposal].apply = Compl$
$\land\ history' = Append(history, [type \mapsto Change, phase \mapsto Apply, index \mapsto i])$

The change fails being applied to the target.

The configuration's applied index is not incremented here to block applying

subsequent changes until the failed change is rolled back.

$\lor\ \land\ proposal' = [proposal \text{ EXCEPT } ![transaction[i].change.proposal].apply = Failed]$
$\land\ \text{UNCHANGED } \langle configuration, target, history\rangle$

The change failed validation. Increment the applied index and cancel the change.

$\lor\ \land\ proposal[transaction[i].change.proposal].commit = Failed$
$\land\ configuration' = [configuration \text{ EXCEPT } !.applied.index = i]$
$\land\ proposal' = [proposal \text{ EXCEPT } ![transaction[i].change.proposal].apply = Canceled]$
$\land\ \text{UNCHANGED } \langle target, history\rangle$

If the prior change failed apply or was aborted due to an earlier apply failure

and the change has not been rolled back, abort this change.

$\lor\ \land\ i - 1 \in \text{DOMAIN } transaction \Rightarrow$
$\land\ proposal[transaction[i-1].change.proposal].apply \in \{Aborted, Failed\}$
$\land\ transaction[i-1].rollback.proposal \in \text{DOMAIN } proposal \Rightarrow$
$proposal[transaction[i-1].rollback.proposal].apply \neq Complete$
$\land\ proposal' = [proposal \text{ EXCEPT } ![transaction[i].change.proposal].apply = Aborted]$
$\land\ \text{UNCHANGED } \langle configuration, target, history\rangle$
$\land\ \text{UNCHANGED } \langle transaction\rangle$

$ReconcileRollback(n, i) \triangleq$
$\land\ transaction[i].phase = Rollback$
$\land\ transaction[i].change.proposal \in \text{DOMAIN } proposal$

The rollback proposal has not yet been created.

$\land\ \lor\ \land\ transaction[i].rollback.proposal \notin \text{DOMAIN } proposal$

The subsequent transaction, if present, is being rolled back.

$\land\ i + 1 \in \text{DOMAIN } transaction \Rightarrow$
$\land\ transaction[i+1].rollback.proposal \in \text{DOMAIN } proposal$
$\land\ Len(proposal) = transaction[i+1].rollback.proposal$
$\land\ proposal' = Append(proposal, [transaction \mapsto i, commit \mapsto Pending, apply \mapsto Pending])$
$\land\ transaction' = [transaction \text{ EXCEPT } ![i].rollback.proposal = Len(proposal')]$
$\land\ \text{UNCHANGED } \langle configuration, target, history\rangle$

The rollback proposal has been created.

$\lor\ \land\ transaction[i].rollback.proposal \in \text{DOMAIN } proposal$

The rollback commit is pending.

$\land\ \lor\ \land\ proposal[transaction[i].rollback.proposal].commit = Pending$

The prior proposal has been committed.

$\land\ transaction[i].rollback.proposal - 1 \in \text{DOMAIN } proposal \Rightarrow$
$proposal[transaction[i].rollback.proposal - 1].commit \in Done$

The prior change has been committed.

$\land\ i - 1 \in \text{DOMAIN } transaction \Rightarrow$
$proposal[transaction[i-1].change.proposal].commit \in Done$

If the change proposal completed, commit the rollback proposal.

$\land \lor \land proposal[transaction[i].change.proposal].commit = Complete$
$\land configuration' = [configuration \text{ EXCEPT } !.committed.revision = transaction[i].rollback.rev$
$!.committed.values = transaction[i].rollback.val$
$configuration.committ$
$\land proposal' = [proposal \text{ EXCEPT } ![transaction[i].rollback.proposal].commit = Complete]$
$\land history' = Append(history, [type \mapsto Rollback, phase \mapsto Commit, index \mapsto i])$
If the change proposal failed, complete the rollback commit.
$\lor \land proposal[transaction[i].change.proposal].commit = Failed$
$\land proposal' = [proposal \text{ EXCEPT } ![transaction[i].rollback.proposal].commit = Complete]$
$\land \text{UNCHANGED } \langle configuration, history \rangle$
$\land \text{UNCHANGED } \langle target \rangle$
The rollback apply is pending.
$\lor \land proposal[transaction[i].rollback.proposal].apply = Pending$
The prior proposal has been applied.
$\land transaction[i].rollback.proposal - 1 \in \text{DOMAIN } proposal \Rightarrow$
$proposal[transaction[i].rollback.proposal - 1].apply \in Done$
The prior change has been applied.
$\land i - 1 \in \text{DOMAIN } transaction \Rightarrow$
$proposal[transaction[i - 1].change.proposal].apply \in Done$
The change has been applied and the rollback has been committed.
Apply the rollback.
$\land \lor \land proposal[transaction[i].change.proposal].apply \in Done$
$\land proposal[transaction[i].rollback.proposal].commit = Complete$
The change was applied or the apply failed. Ensure the rollback
is updated in the target.
$\land \lor \land proposal[transaction[i].change.proposal].apply \in \{Complete, Failed\}$
$\land configuration.state = Complete$
$\land configuration.term = mastership.term$
$\land conn[n].id = mastership.conn$
$\land conn[n].connected$
$\land target.running$
Rollbacks are applied until successful.
$\land target' = [target \text{ EXCEPT } !.values = transaction[i].rollback.values \mathbin{@@} target.values]$
$\land configuration' = [configuration \text{ EXCEPT } !.applied.target = target.id,$
$!.applied.revision = transaction[i].rollback.r$
$!.applied.values = transaction[i].rollback.v$
$configuration.applied$
$\land proposal' = [proposal \text{ EXCEPT } ![transaction[i].rollback.proposal].apply = Complete]$
$\land history' = Append(history, [type \mapsto Rollback, phase \mapsto Apply, index \mapsto i])$
If the change apply was aborted or canceled, no requests were sent to the target.
Complete the rollback apply without changes to the target.
$\lor \land proposal[transaction[i].change.proposal].apply \in \{Aborted, Canceled\}$
$\land proposal' = [proposal \text{ EXCEPT } ![transaction[i].rollback.proposal].apply = Complete]$
$\land \text{UNCHANGED } \langle configuration, target, history \rangle$
The change is pending apply. cancel applying the change.

```
        ∨ ∧ proposal[transaction[i].change.proposal].apply = Pending
          ∧ proposal' = [proposal EXCEPT ![transaction[i].change.proposal].apply = Canceled]
          ∧ UNCHANGED ⟨configuration, target, history⟩
    If the apply is complete and the applied index matches the previous change index,
    increment the applied index to unblock later changes. This ensures that changes
    following a sequence of aborted/failed changes are blocked until the failed/aborted
    changes are rolled back and unblocked once all rollbacks have been applied.
        ∨ ∧ proposal[transaction[i].rollback.proposal].apply = Complete
          ∧ configuration.applied.index = i − 1
          ∧ configuration' = [configuration EXCEPT !.applied.index = i]
          ∧ UNCHANGED ⟨proposal, target, history⟩
    ∧ UNCHANGED ⟨transaction⟩

ReconcileTransaction(n, i) ≜
  ∧ i ∈ DOMAIN transaction
    ∧ ∨ ReconcileChange(n, i)
      ∨ ReconcileRollback(n, i)
    ∧ UNCHANGED ⟨mastership, conn⟩
```