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*

The set of possible paths and values
CONSTANT *Path*, *Value*

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

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 sequence of configuration changes used for model checking.
VARIABLE *history*

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

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

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

$$
\begin{array}{l}
\mathit{Test} \triangleq \text{INSTANCE } \mathit{Test} \text{ WITH} \\
\quad \mathit{File} \leftarrow \text{“Transaction.log”}
\end{array}
$$

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

```
LOCAL Transaction(i) ≜
  IF i ∈ DOMAIN transaction THEN
    transaction[i]
  ELSE [
    index    ↦ i,
    revision ↦ 0,
    change   ↦ [
      index    ↦ 0,
      revision ↦ 0],
    rollback ↦ [
      index    ↦ 0,
      revision ↦ 0],
    commit   ↦ Nil,
    apply    ↦ Nil]

LOCAL LastTransaction ≜ Transaction(Len(transaction))

CHANGE [index = 1, revision = 1, change = (index = 1, revision = 1), rollback = (index =
0, revision = 0)] ← − Change revision 1
CHANGE [index = 2, revision = 2, change = (index = 2, revision = 2), rollback = (index =
1, revision = 1)]
CHANGE [index = 3, revision = 3, change = (index = 3, revision = 3), rollback = (index =
2, revision = 2)]
ROLLBACK [index = 4, revision = 3, change = (index = 2, revision = 2), rollback = (index =
3, revision = 3)] ← − Roll back revision 3 at index 3, leading to revision 2
ROLLBACK [index = 5, revision = 3, change = (index = 1, revision = 1), rollback = (index =
2, revision = 2)]
CHANGE [index = 6, revision = 4, change = (index = 6, revision = 4), rollback = (index =
1, revision = 1)]
CHANGE [index = 7, revision = 5, change = (index = 7, revision = 5), rollback = (index =
6, revision = 4)]
ROLLBACK [index = 8, revision = 5, change = (index = 6, revision = 4), rollback = (index =
7, revision = 5)] ← − Roll back revision 5 at index 7, leading to revision 4
ROLLBACK [index = 9, revision = 5, change = (index = 1, revision = 1), rollback = (index =
6, revision = 4)] ← − Roll back revision 4 at index 6, leading to revision 1 CHANGE [index =
10, revision = 6, change = (index = 10, revision = 6), rollback = (index = 1, revision = 1)]

Add a set of changes 'c' to the transaction log
AppendChange(i) ≜
  ∧ LastTransaction.revision = i − 1
  ∧ Len(transaction) > 0 ⇒ transaction[Len(transaction)].commit = Complete
  ∧ ∃ p ∈ Path, v ∈ Value :
      ∧ transaction′ = Append(transaction, [
                         type ↦ Change,
                         index ↦ Len(transaction) + 1,
                         revision ↦ i,
                         change ↦ [
                           index ↦ Len(transaction) + 1,
```

```
                                    revision  ↦ i,
                                    values    ↦ (p :> v)],
                                rollback ↦ [
                                    index     ↦ LastTransaction.change.index,
                                    revision  ↦ LastTransaction.change.revision,
                                    values    ↦ IF p ∈ DOMAIN configuration.committed.values THEN
                                                    (p :> configuration.committed.values[p])
                                                ELSE
                                                    (p :> Nil)],
                                    commit    ↦ Pending,
                                    apply     ↦ Pending])
    ∧ UNCHANGED ⟨configuration, mastership, conn, target, history⟩

  Add a rollback of transaction 't' to the transaction log
RollbackChange(i) ≜
    ∧ LastTransaction.change.revision = i
    ∧ Len(transaction) > 0 ⇒ transaction[Len(transaction)].commit = Complete
    ∧ transaction′ = Append(transaction, [
                            type      ↦ Rollback,
                            index     ↦ Len(transaction) + 1,
                            revision  ↦ LastTransaction.revision,
                            change    ↦ [
                                index     ↦ transaction[LastTransaction.change.index].rollback.index,
                                revision  ↦ transaction[LastTransaction.change.index].rollback.revision,
                                values    ↦ transaction[LastTransaction.change.index].rollback.values],
                            rollback  ↦ [
                                index     ↦ LastTransaction.change.index,
                                revision  ↦ i,
                                values    ↦ Empty],
                            commit    ↦ Pending,
                            apply     ↦ Pending])
    ∧ UNCHANGED ⟨configuration, mastership, conn, target, history⟩

──────────────────────────────────────────────────────────────────────────────

CommitChange(n, i) ≜
    ∧ transaction[i].commit = Pending
    ∧ i − 1 ∈ DOMAIN transaction ⇒
            transaction[i − 1].commit ∈ Done
    ∧ configuration′ = [configuration EXCEPT !.committed.index    = transaction[i].change.index,
                                             !.committed.revision = transaction[i].change.revision,
                                             !.committed.values   = transaction[i].change.values @@
                                                                    configuration.committed.values]
    ∧ transaction′ = [transaction EXCEPT ![i].commit = Complete]
    ∧ history′ = Append(history, [
                        type      ↦ Change,
```

$$
\begin{array}{l}
\qquad\qquad\qquad\quad phase \quad \mapsto Commit, \\
\qquad\qquad\qquad\quad revision \mapsto transaction[i].change.revision]) \\
\quad \land \text{UNCHANGED } \langle target \rangle
\end{array}
$$

$$
\begin{array}{l}
ApplyChange(n,\ i) \triangleq \\
\quad \land transaction[i].apply = Pending \\
\quad \land transaction[i].commit = Complete \\
\quad \land i - 1 \in \text{DOMAIN } transaction \Rightarrow \\
\qquad\quad transaction[i-1].apply \in Done \\
\quad \land \lor \land i - 1 \in \text{DOMAIN } transaction \Rightarrow \\
\qquad\qquad\quad transaction[i-1].apply = Complete \\
\qquad\quad \land configuration.state = Complete \\
\qquad\quad \land configuration.term = mastership.term \\
\qquad\quad \land conn[n].id = mastership.conn \\
\qquad\quad \land conn[n].connected \\
\qquad\quad \land target.running \\
\qquad\qquad \text{Apply to the target successfully.} \\
\qquad\quad \land \lor \land target' = [target \text{ EXCEPT } !.values = transaction[i].change.values \mathbin{@@} target.values] \\
\qquad\qquad\quad \land configuration' = [configuration \text{ EXCEPT } !.applied.index \quad = transaction[i].change.index, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad !.applied.revision = transaction[i].change.revision, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad !.applied.values \quad = transaction[i].change.values \mathbin{@@} \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad configuration.applied.values] \\
\qquad\qquad\quad \land transaction' = [transaction \text{ EXCEPT } ![i].apply = Complete] \\
\qquad\qquad\quad \land history' = Append(history, [\\
\qquad\qquad\qquad\qquad\qquad type \quad \mapsto Change, \\
\qquad\qquad\qquad\qquad\qquad phase \quad \mapsto Apply, \\
\qquad\qquad\qquad\qquad\qquad revision \mapsto transaction[i].change.revision]) \\
\qquad\qquad \text{Apply to the target failed.} \\
\qquad\quad \lor \land transaction' = [transaction \text{ EXCEPT } ![i].apply = Failed] \\
\qquad\qquad\quad \land \text{UNCHANGED } \langle configuration,\ target,\ history \rangle \\
\qquad \lor \land i - 1 \in \text{DOMAIN } transaction \\
\qquad\quad \land transaction[i-1].apply \in \{Aborted,\ Failed\} \\
\qquad\quad \land transaction' = [transaction \text{ EXCEPT } ![i].apply = Aborted] \\
\qquad\quad \land \text{UNCHANGED } \langle configuration,\ target,\ history \rangle
\end{array}
$$

$$
\begin{array}{l}
ReconcileChange(n,\ i) \triangleq \\
\quad \land transaction[i].type = Change \\
\quad \land \lor CommitChange(n,\ i) \\
\qquad \lor ApplyChange(n,\ i)
\end{array}
$$

$$
\begin{array}{l}
CommitRollback(n,\ i) \triangleq \\
\quad \land transaction[i].commit = Pending \\
\quad \land i - 1 \in \text{DOMAIN } transaction \Rightarrow \\
\qquad\quad transaction[i-1].commit \in Done \\
\quad \land configuration' = [configuration \text{ EXCEPT } !.committed.index \quad = transaction[i].change.index, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad !.committed.revision = transaction[i].change.revision,
\end{array}
$$

```
                                                !.committed.values    = transaction[i].change.values @@
                                                                        configuration.committed.values]
    ∧ transaction' = [transaction EXCEPT ![i].commit = Complete]
    ∧ history' = Append(history, [
                        type      ↦ Rollback,
                        phase     ↦ Commit,
                        revision  ↦ transaction[i].rollback.revision])
    ∧ UNCHANGED ⟨target⟩

ApplyRollback(n, i) ≜
    ∧ transaction[i].apply = Pending
    ∧ transaction[i].commit = Complete
    ∧ i − 1 ∈ DOMAIN transaction ⇒
          transaction[i − 1].apply ∈ Done
    ∧ ∨ ∧ transaction[transaction[i].rollback.index].apply ∈ {Complete, Failed}
        ∧ configuration.state = Complete
        ∧ configuration.term = mastership.term
        ∧ conn[n].id = mastership.conn
        ∧ conn[n].connected
        ∧ target.running
        ∧ target' = [target EXCEPT !.values = transaction[i].change.values @@ target.values]
        ∧ configuration' = [configuration EXCEPT !.applied.index    = transaction[i].change.index,
                                                 !.applied.revision = transaction[i].change.revision,
                                                 !.applied.values   = transaction[i].change.values @@
                                                                      configuration.applied.values]
        ∧ transaction' = [transaction EXCEPT ![i].apply = Complete]
        ∧ history' = Append(history, [
                            type      ↦ Rollback,
                            phase     ↦ Apply,
                            revision  ↦ transaction[i].rollback.revision])
      ∨ ∧ transaction[transaction[i].rollback.index].apply = Aborted
        ∧ transaction' = [transaction EXCEPT ![i].apply = Aborted]
        ∧ UNCHANGED ⟨configuration, target, history⟩

ReconcileRollback(n, i) ≜
    ∧ transaction[i].type = Rollback
    ∧ ∨ CommitRollback(n, i)
      ∨ ApplyRollback(n, i)

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