```
┌──────────────────── MODULE Config ────────────────────┐
INSTANCE Naturals

INSTANCE FiniteSets

INSTANCE Sequences

INSTANCE TLC

├───────────────────────────────────────────────────────┤

GenerateTestCases ≜ TRUE

Nil ≜ "<nil>"

Change ≜ "Change"
Rollback ≜ "Rollback"

Commit ≜ "Commit"
Apply ≜ "Apply"

Pending ≜ "Pending"
InProgress ≜ "InProgress"
Complete ≜ "Complete"
Aborted ≜ "Aborted"
Canceled ≜ "Canceled"
Failed ≜ "Failed"

Node ≜ {"node1"}

NumTransactions ≜ 3
NumTerms ≜ 1
NumConns ≜ 1
NumStarts ≜ 1

Path ≜ {"path1"}
Value ≜ {"value1", "value2"}

├───────────────────────────────────────────────────────┤

  A transaction log.
VARIABLE transactions

  A record of per-target configurations
VARIABLE configuration

  A record of target masterships
VARIABLE mastership

  A record of node connections to the target
```

VARIABLE $conn$

The target state
VARIABLE $target$

A sequence of state changes used for model checking.
VARIABLE $history$

$vars \triangleq \langle transactions, configuration, mastership, conn, target, history \rangle$

LOCAL $Transaction \triangleq$ INSTANCE $Transaction$

LOCAL $Configuration \triangleq$ INSTANCE $Configuration$

LOCAL $Mastership \triangleq$ INSTANCE $Mastership$

LOCAL $Target \triangleq$ INSTANCE $Target$

```
$AppendChange(i) \triangleq$
    $\land Transaction!AppendChange(i)$

$RollbackChange(i) \triangleq$
    $\land Transaction!RollbackChange(i)$

$ReconcileTransaction(n, i) \triangleq$
    $\land i \in$ DOMAIN $transactions$
    $\land \lor \land Transaction!ReconcileTransaction(n, i)$
            $\land GenerateTestCases \Rightarrow Transaction!Test!Log([node \mapsto n, index \mapsto i])$
       $\lor \land GenerateTestCases$
            $\land \neg$ENABLED $Transaction!ReconcileTransaction(n, i)$
            $\land$ UNCHANGED $vars$
            $\land Transaction!Test!Log([node \mapsto n, index \mapsto i])$

$ReconcileConfiguration(n) \triangleq$
    $\lor \land Configuration!ReconcileConfiguration(n)$
       $\land$ UNCHANGED $\langle transactions, history \rangle$
       $\land GenerateTestCases \Rightarrow Configuration!Test!Log([node \mapsto n])$
    $\lor \land GenerateTestCases$
       $\land \neg$ENABLED $Configuration!ReconcileConfiguration(n)$
       $\land$ UNCHANGED $vars$
       $\land Configuration!Test!Log([node \mapsto n])$

$ReconcileMastership(n) \triangleq$
    $\lor \land Mastership!ReconcileMastership(n)$
       $\land$ UNCHANGED $\langle transactions, configuration, target, history \rangle$
```

```
      $\land GenerateTestCases \Rightarrow Mastership!Test!Log([node \mapsto n])$
   $\lor$ $\land GenerateTestCases$
      $\land \neg \text{ENABLED } Mastership!ReconcileMastership(n)$
      $\land \text{UNCHANGED } vars$
      $\land Mastership!Test!Log([node \mapsto n])$

$ConnectNode(n) \triangleq$
   $\land Target!Connect(n)$
   $\land \text{UNCHANGED } \langle transactions, configuration, mastership, history \rangle$

$DisconnectNode(n) \triangleq$
   $\land Target!Disconnect(n)$
   $\land \text{UNCHANGED } \langle transactions, configuration, mastership, history \rangle$

$StartTarget \triangleq$
   $\land Target!Start$
   $\land \text{UNCHANGED } \langle transactions, configuration, mastership, history \rangle$

$StopTarget \triangleq$
   $\land Target!Stop$
   $\land \text{UNCHANGED } \langle transactions, configuration, mastership, history \rangle$
```

Formal specification, constraints, and theorems.

```
$Init \triangleq$
   $\land transactions = [$
         $i \in \{\} \mapsto [$
           $phase \quad \mapsto Nil,$
           $values \mapsto [$
               $p \in \{\} \mapsto Nil],$
           $change \quad \mapsto [$
               $commit \mapsto Nil,$
               $apply \quad \mapsto Nil],$
           $rollback \mapsto [$
               $commit \mapsto Nil,$
               $apply \quad \mapsto Nil]]]$
   $\land configuration = [$
         $state \;\; \mapsto Pending,$
         $term \;\; \mapsto 0,$
         $committed \mapsto [$
             $index \quad \mapsto 0,$
             $change \;\; \mapsto 0,$
             $target \quad \mapsto 0,$
             $ordinal \;\; \mapsto 0,$
             $revision \mapsto 0,$
             $values \quad \mapsto [$
```

```
                p ∈ {} ↦ Nil]],
            applied ↦ [
                index    ↦ 0,
                target   ↦ 0,
                ordinal  ↦ 0,
                revision ↦ 0,
                values   ↦ [
                    p ∈ {} ↦ Nil]]]
    ∧ target = [
            id      ↦ 1,
            running ↦ TRUE,
            values  ↦ [
                p ∈ {}  ↦ [
                    index ↦ 0,
                    value ↦ Nil]]]
    ∧ mastership = [
            master ↦ CHOOSE n ∈ Node : TRUE,
            term   ↦ 1,
            conn   ↦ 1]
    ∧ conn = [
            n ∈ Node ↦ [
                id        ↦ 1,
                connected ↦ TRUE]]
    ∧ history = ⟨⟩

Next ≜
    ∨ ∃ i ∈ 1 . . NumTransactions :
        ∨ AppendChange(i)
        ∨ RollbackChange(i)
    ∨ ∃ n ∈ Node, i ∈ 1 . . NumTransactions :
        ReconcileTransaction(n, i)
    ∨ ∃ n ∈ Node :
        ReconcileConfiguration(n)
    ∨ ∃ n ∈ Node :
        ReconcileMastership(n)
    ∨ ∃ n ∈ Node :
        ∨ ConnectNode(n)
        ∨ DisconnectNode(n)
    ∨ StartTarget
    ∨ StopTarget

Spec ≜
    ∧ Init
    ∧ □[Next]_vars
    ∧ ∀ i ∈ 1 . . NumTransactions :
```

$$\begin{array}{l}
\qquad \mathrm{WF}_{\langle \mathit{transactions} \rangle}(\mathit{Transaction}\,!\,\mathit{RollbackChange}(i))\\
\quad \land\ \forall\, n \in \mathit{Node},\ i \in 1 \mathrel{..} \mathit{NumTransactions} :\\
\qquad \mathrm{WF}_{\langle \mathit{transactions},\, \mathit{configuration},\, \mathit{mastership},\, \mathit{conn},\, \mathit{target},\, \mathit{history} \rangle}(\mathit{Transaction}\,!\,\mathit{ReconcileTransaction}(n,\ i))\\
\quad \land\ \forall\, n \in \mathit{Node} :\\
\qquad \mathrm{WF}_{\langle \mathit{configuration},\, \mathit{mastership},\, \mathit{conn},\, \mathit{target} \rangle}(\mathit{Configuration}\,!\,\mathit{ReconcileConfiguration}(n))\\
\quad \land\ \forall\, n \in \mathit{Node} :\\
\qquad \mathrm{WF}_{\langle \mathit{mastership},\, \mathit{conn} \rangle}(\mathit{Mastership}\,!\,\mathit{ReconcileMastership}(n))\\
\quad \land\ \forall\, n \in \mathit{Node} :\\
\qquad \mathrm{WF}_{\langle \mathit{conn},\, \mathit{target} \rangle}(\mathit{Target}\,!\,\mathit{Connect}(n) \lor \mathit{Target}\,!\,\mathit{Disconnect}(n))\\
\quad \land\ \mathrm{WF}_{\langle \mathit{conn},\, \mathit{target} \rangle}(\mathit{Target}\,!\,\mathit{Start} \lor \mathit{Target}\,!\,\mathit{Stop})
\end{array}$$

$$\begin{array}{l}
\mathit{LimitTerms} \triangleq\\
\quad \lor\ \mathit{mastership.term} < \mathit{NumTerms}\\
\quad \lor\ \land\ \mathit{mastership.term} = \mathit{NumTerms}\\
\qquad \land\ \mathit{mastership.master} \neq \mathit{Nil}
\end{array}$$

$$\begin{array}{l}
\mathit{LimitConns} \triangleq\\
\quad \forall\, n \in \textsc{domain}\ \mathit{conn} :\\
\qquad \lor\ \mathit{conn}[n].\mathit{id} < \mathit{NumConns}\\
\qquad \lor\ \land\ \mathit{conn}[n].\mathit{id} = \mathit{NumConns}\\
\qquad\quad \land\ \mathit{conn}[n].\mathit{connected}
\end{array}$$

$$\begin{array}{l}
\mathit{LimitStarts} \triangleq\\
\quad \lor\ \mathit{target.id} < 2\\
\quad \lor\ \land\ \mathit{target.id} = 2\\
\qquad \land\ \mathit{target.running}
\end{array}$$

$$\begin{array}{l}
\mathit{TypeOK} \triangleq\\
\quad \land\ \mathit{Transaction}\,!\,\mathit{TypeOK}\\
\quad \land\ \mathit{Configuration}\,!\,\mathit{TypeOK}\\
\quad \land\ \mathit{Mastership}\,!\,\mathit{TypeOK}
\end{array}$$

$$\begin{array}{l}
\mathit{StatusCommitted}(i) \triangleq\\
\quad \land\ \mathit{Len}(\mathit{history}) = \mathit{Len}(\mathit{history}')\\
\quad \land\ \lor\ \land\ \mathit{transactions}'[i].\mathit{change.commit} \notin \{\mathit{Pending},\ \mathit{Canceled}\}\\
\qquad\qquad \land\ \mathit{transactions}[i].\mathit{change.commit} \neq \mathit{transactions}'[i].\mathit{change.commit}\\
\qquad \lor\ \land\ \mathit{transactions}'[i].\mathit{rollback.commit} \notin \{\mathit{Pending},\ \mathit{Canceled}\}\\
\qquad\qquad \land\ \mathit{transactions}[i].\mathit{rollback.commit} \neq \mathit{transactions}'[i].\mathit{rollback.commit}
\end{array}$$

$$\begin{array}{l}
\mathit{StatusApplied}(i) \triangleq\\
\quad \land\ \mathit{Len}(\mathit{history}) = \mathit{Len}(\mathit{history}')\\
\quad \land\ \lor\ \land\ \mathit{transactions}'[i].\mathit{change.apply} \notin \{\mathit{Pending},\ \mathit{Canceled},\ \mathit{Aborted}\}\\
\qquad\qquad \land\ \mathit{transactions}[i].\mathit{change.apply} \neq \mathit{transactions}'[i].\mathit{change.apply}
\end{array}$$

```
        $\lor \land transactions'[i].rollback.apply \notin \{Pending, Canceled, Aborted\}$
          $\land transactions[i].rollback.apply \neq transactions'[i].rollback.apply$

$ValidStatus(t, i, j) \triangleq$
    $\land j \in \text{DOMAIN } history$
    $\land history[j].index = i$
    $\land \lor \land history[j].phase = Change$
          $\land history[j].event = Commit$
          $\land t[i].change.commit = history[j].status$
       $\lor \land history[j].phase = Change$
          $\land history[j].event = Apply$
          $\land t[i].change.apply = history[j].status$
       $\lor \land history[j].phase = Rollback$
          $\land history[j].event = Commit$
          $\land t[i].rollback.commit = history[j].status$
       $\lor \land history[j].phase = Rollback$
          $\land history[j].event = Apply$
          $\land t[i].rollback.apply = history[j].status$

$ValidCommit(t, i) \triangleq$
    $\text{LET } j \triangleq \text{CHOOSE } j \in \text{DOMAIN } history :$
                    $\land history[j].event = Commit$
                    $\land \neg\exists k \in \text{DOMAIN } history :$
                          $\land history[k].event = Commit$
                          $\land k > j$
    $\text{IN}\quad ValidStatus(t, i, j)$

$ValidApply(t, i) \triangleq$
    $\text{LET } j \triangleq \text{CHOOSE } j \in \text{DOMAIN } history :$
                    $\land history[j].event = Apply$
                    $\land \neg\exists k \in \text{DOMAIN } history :$
                          $\land history[k].event = Apply$
                          $\land k > j$
    $\text{IN}\quad ValidStatus(t, i, j)$

$AtomicStatusChange \triangleq$
    $\forall i \in 1 \,..\, NumTransactions :$
      $\land i \in \text{DOMAIN } transactions \Rightarrow$
           $\land StatusCommitted(i) \Rightarrow ValidCommit(transactions', i)$
           $\land StatusApplied(i) \Rightarrow ValidApply(transactions', i)$

$Transition \triangleq \Box[AtomicStatusChange]_{\langle transactions,\, history \rangle}$

$\text{LOCAL } IsOrderedChange(p, i) \triangleq$
    $\land \quad history[i].phase = Change$
    $\land \quad history[i].event = p$
    $\land \quad history[i].status = Complete$
    $\land \quad \lnot \exists\, j \in \textsc{domain}\ \mathit{history} :$
            $\land\, j < i$
            $\land\, \mathit{history}[j].\mathit{phase} = \mathit{Change}$
            $\land\, \mathit{history}[j].\mathit{event} = p$
            $\land\, \mathit{history}[j].\mathit{status} = \mathit{Complete}$
            $\land\, \mathit{history}[j].\mathit{index} \geq \mathit{history}[i].\mathit{index}$

$\textsc{local}\ \mathit{IsOrderedRollback}(p, i) \triangleq$
    $\land \quad \mathit{history}[i].\mathit{phase} = \mathit{Rollback}$
    $\land \quad \mathit{history}[i].\mathit{event} = p$
    $\land \quad \mathit{history}[i].\mathit{status} = \mathit{Complete}$
    $\land \quad \exists\, j \in \textsc{domain}\ \mathit{history} :$
          $\land\, j < i$
          $\land\, \mathit{history}[j].\mathit{phase} = \mathit{Change}$
          $\land\, \mathit{history}[j].\mathit{status} = \mathit{Complete}$
          $\land\, \mathit{history}[j].\mathit{index} = \mathit{history}[i].\mathit{index}$
    $\land \quad \lnot \exists\, j \in \textsc{domain}\ \mathit{history} :$
            $\land\, j < i$
            $\land\, \mathit{history}[j].\mathit{phase} = \mathit{Change}$
            $\land\, \mathit{history}[j].\mathit{event} = p$
            $\land\, \mathit{history}[j].\mathit{status} = \mathit{Complete}$
            $\land\, \mathit{history}[j].\mathit{index} > \mathit{history}[i].\mathit{index}$
            $\land\, \lnot \exists\, k \in \textsc{domain}\ \mathit{history} :$
                  $\land\, k > j$
                  $\land\, k < i$
                  $\land\, \mathit{history}[k].\mathit{phase} = \mathit{Rollback}$
                  $\land\, \mathit{history}[k].\mathit{event} = p$
                  $\land\, \mathit{history}[j].\mathit{status} = \mathit{Complete}$
                  $\land\, \mathit{history}[k].\mathit{index} = \mathit{history}[j].\mathit{index}$

$\mathit{Order} \triangleq$
    $\land\ \forall\, i \in \textsc{domain}\ \mathit{history} :$
        $\mathit{history}[i].\mathit{status} = \mathit{Complete} \Rightarrow$
            $\lor\, \mathit{IsOrderedChange}(\mathit{Commit}, i)$
            $\lor\, \mathit{IsOrderedChange}(\mathit{Apply}, i)$
            $\lor\, \mathit{IsOrderedRollback}(\mathit{Commit}, i)$
            $\lor\, \mathit{IsOrderedRollback}(\mathit{Apply}, i)$
    $\land\ \forall\, i \in \textsc{domain}\ \mathit{transactions} :$
        $\land\, \mathit{transactions}[i].\mathit{change}.\mathit{apply} = \mathit{Failed}$
        $\land\, \mathit{transactions}[i].\mathit{rollback}.\mathit{apply} \neq \mathit{Complete}$
        $\Rightarrow \lnot \exists\, j \in \textsc{domain}\ \mathit{transactions} :$
              $\land\, j > i$
              $\land\, \mathit{transactions}[i].\mathit{change}.\mathit{apply} \in \{\mathit{InProgress}, \mathit{Complete}\}$

$\textsc{local}\ \mathit{IsChangeCommitted}(i) \triangleq$
    $\land \quad \mathit{configuration}.\mathit{committed}.\mathit{revision} = i$
```

```
LOCAL $IsChangeApplied(i) \triangleq$
    $\land \quad configuration.applied.revision = i$

$Consistency \triangleq$
    $\land \forall i \in$ DOMAIN $transactions :$
        $\land IsChangeCommitted(i)$
        $\land \neg\exists j \in$ DOMAIN $transactions :$
            $\land j > i$
            $\land IsChangeCommitted(j)$
        $\Rightarrow \forall p \in$ DOMAIN $transactions[i].change.values :$
            $\land configuration.committed.values[p] = transactions[i].change.values[p]$
    $\land \forall i \in$ DOMAIN $transactions :$
        $\land IsChangeApplied(i)$
        $\land \neg\exists j \in$ DOMAIN $transactions :$
            $\land j > i$
            $\land IsChangeApplied(j)$
        $\Rightarrow \forall p \in$ DOMAIN $transactions[i].change.values :$
            $\land configuration.applied.values[p] = transactions[i].change.values[p]$
            $\land \land target.running$
              $\land configuration.applied.target = target.id$
              $\land configuration.state = Complete$
              $\Rightarrow target.values[p] = transactions[i].change.values[p]$

$Safety \triangleq \Box(Order \land Consistency)$

THEOREM $Spec \Rightarrow Safety$

LOCAL $IsChanging(i) \triangleq$
    $\land \quad i \in$ DOMAIN $transactions$
    $\land \quad transactions[i].phase = Change$

LOCAL $IsChanged(i) \triangleq$
    $\land \quad i \in$ DOMAIN $transactions$
    $\land \quad transactions[i].change.commit \in \{Complete, Failed\}$
    $\land \quad transactions[i].change.apply \in \{Complete, Aborted, Failed\}$

LOCAL $IsRollingBack(i) \triangleq$
    $\land \quad i \in$ DOMAIN $transactions$
    $\land \quad transactions[i].phase = Rollback$

LOCAL $IsRolledBack(i) \triangleq$
    $\land \quad i \in$ DOMAIN $transactions$
    $\land \quad transactions[i].rollback.commit \in \{Complete, Failed\}$
    $\land \quad transactions[i].rollback.apply \in \{Complete, Aborted, Failed\}$

$Terminates(i) \triangleq$
    $\land IsChanging(i) \leadsto IsChanged(i)$
```

$\quad \land\ \mathit{IsRollingBack}(i) \leadsto \mathit{IsRolledBack}(i)$

$\mathit{Termination}\ \triangleq$
$\quad \forall\, i \in 1 \,..\, \mathit{NumTransactions} : \mathit{Terminates}(i)$

$\mathit{Liveness}\ \triangleq\ \mathit{Termination}$

THEOREM $\mathit{Spec} \Rightarrow \mathit{Liveness}$