MODULE *Config*

```
INSTANCE Naturals

INSTANCE FiniteSets

INSTANCE Sequences

INSTANCE TLC
```

```
GenerateTestCases ≜ FALSE

Nil ≜ “<nil>”

Change ≜ “Change”
Rollback ≜ “Rollback”

Commit ≜ “Commit”
Apply ≜ “Apply”

Pending ≜ “Pending”
Complete ≜ “Complete”
Canceled ≜ “Canceled”
Aborted ≜ “Aborted”
Failed ≜ “Failed”
Done ≜ {Complete, Canceled, Aborted, Failed}

Node ≜ {“node1”}

NumTransactions ≜ 3
NumTerms ≜ 2
NumConns ≜ 2
NumStarts ≜ 2

Path ≜ {“path1”}
Value ≜ {“value1”, “value2”}
```

```
  A transaction log of changes and rollbacks.
VARIABLE transaction

  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 ≜ ⟨transaction, configuration, mastership, conn, target, history⟩
```

```
LOCAL Transaction ≜ INSTANCE Transaction

LOCAL Configuration ≜ INSTANCE Configuration

LOCAL Mastership ≜ INSTANCE Mastership

LOCAL Target ≜ INSTANCE Target
```

```
AppendChange(i) ≜
    ∧ Transaction!AppendChange(i)

RollbackChange(i) ≜
    ∧ Transaction!RollbackChange(i)

ReconcileTransaction(n, i) ≜
    ∧ Transaction!ReconcileTransaction(n, i)
    ∧ GenerateTestCases ⇒ Transaction!Test!Log([node ↦ n, index ↦ i])

ReconcileConfiguration(n) ≜
    ∧ Configuration!ReconcileConfiguration(n)
    ∧ UNCHANGED ⟨transaction, history⟩
    ∧ GenerateTestCases ⇒ Configuration!Test!Log([node ↦ n])

ReconcileMastership(n) ≜
    ∧ Mastership!ReconcileMastership(n)
    ∧ UNCHANGED ⟨transaction, configuration, target, history⟩
    ∧ GenerateTestCases ⇒ Mastership!Test!Log([node ↦ n])

ConnectNode(n) ≜
    ∧ Target!Connect(n)
    ∧ UNCHANGED ⟨transaction, configuration, mastership, history⟩

DisconnectNode(n) ≜
    ∧ Target!Disconnect(n)
    ∧ UNCHANGED ⟨transaction, configuration, mastership, history⟩
```

```
StartTarget ≜
    ∧ Target!Start
    ∧ UNCHANGED ⟨transaction, configuration, mastership, history⟩

StopTarget ≜
    ∧ Target!Stop
    ∧ UNCHANGED ⟨transaction, configuration, mastership, history⟩
```

Formal specification, constraints, and theorems.

```
Init ≜
    ∧ transaction = [
          i ∈ {} ↦ [
            type     ↦ Nil,
            index    ↦ 0,
            revision ↦ 0,
            commit   ↦ Nil,
            apply    ↦ Nil,
            change   ↦ [
                index    ↦ 0,
                revision ↦ 0,
                values   ↦ [
                    p ∈ {}  ↦ [
                       index ↦ 0,
                       value ↦ Nil]]],
            rollback ↦ [
                index    ↦ 0,
                revision ↦ 0,
                values   ↦ [
                    p   ∈ {}  ↦ [
                         index ↦ 0,
                         value ↦ Nil]]]]]
    ∧ configuration = [
          state  ↦ Pending,
          term   ↦ 0,
          committed ↦ [
              index    ↦ 0,
              revision ↦ 0,
              values   ↦ [
                  p ∈ {}  ↦ [
                     index ↦ 0,
                     value ↦ Nil]]],
          applied ↦ [
              target   ↦ 0,
              index    ↦ 0,
            $\mathit{revision} \mapsto 0,$
            $\mathit{values} \;\; \mapsto [$
                $p \in \{\} \;\; \mapsto [$
                    $\mathit{index} \mapsto 0,$
                    $\mathit{value} \;\mapsto \mathit{Nil}]]]]$
    $\land\ \mathit{target} = [$
        $\mathit{id} \qquad\;\; \mapsto 1,$
        $\mathit{running} \mapsto \textsc{true},$
        $\mathit{values} \;\; \mapsto [$
            $p \in \{\} \;\; \mapsto [$
                $\mathit{index} \mapsto 0,$
                $\mathit{value} \;\mapsto \mathit{Nil}]]]$
    $\land\ \mathit{mastership} = [$
        $\mathit{master} \mapsto \textsc{choose}\ n \in \mathit{Node} : \textsc{true},$
        $\mathit{term} \quad\; \mapsto 1,$
        $\mathit{conn} \quad\; \mapsto 1]$
    $\land\ \mathit{conn} = [$
        $n \;\in \mathit{Node} \mapsto [$
            $\mathit{id} \qquad\quad\;\; \mapsto 1,$
            $\mathit{connected} \mapsto \textsc{true}]]$
    $\land\ \mathit{history} = \langle\rangle$

$\mathit{Next} \;\triangleq$
    $\lor\ \exists\, i \in 1 \,..\, \mathit{NumTransactions} :$
        $\lor\ \mathit{AppendChange}(i)$
        $\lor\ \mathit{RollbackChange}(i)$
    $\lor\ \exists\, n \in \mathit{Node} :$
        $\exists\, i \in \textsc{domain}\ \mathit{transaction} :$
          $\mathit{ReconcileTransaction}(n,\ i)$
    $\lor\ \exists\, n \in \mathit{Node} :$
        $\mathit{ReconcileConfiguration}(n)$
    $\lor\ \exists\, n \in \mathit{Node} :$
        $\mathit{ReconcileMastership}(n)$
    $\lor\ \exists\, n \in \mathit{Node} :$
        $\lor\ \mathit{ConnectNode}(n)$
        $\lor\ \mathit{DisconnectNode}(n)$
    $\lor\ \mathit{StartTarget}$
    $\lor\ \mathit{StopTarget}$

$\mathit{Spec} \;\triangleq$
    $\land\ \mathit{Init}$
    $\land\ \Box[\mathit{Next}]_{\mathit{vars}}$
    $\land\ \forall\, i \in 1 \,..\, \mathit{NumTransactions} :$
        $\mathrm{WF}_{\langle \mathit{transaction} \rangle}(\mathit{Transaction}!\mathit{RollbackChange}(i))$
    $\land\ \forall\, n \in \mathit{Node} :$
```

```
        $\mathrm{WF}_{vars}(\exists\, i \in \textsc{domain}\ transaction : Transaction!ReconcileTransaction(n,\, i))$
    $\land\ \forall\, n \in Node :$
        $\mathrm{WF}_{\langle configuration,\, mastership,\, conn,\, target \rangle}(Configuration!ReconcileConfiguration(n))$
    $\land\ \forall\, n \in Node :$
        $\mathrm{WF}_{\langle mastership,\, conn \rangle}(Mastership!ReconcileMastership(n))$
    $\land\ \forall\, n \in Node :$
        $\mathrm{WF}_{\langle conn,\, target \rangle}(Target!Connect(n) \lor Target!Disconnect(n))$
    $\land\ \mathrm{WF}_{\langle conn,\, target \rangle}(Target!Start \lor Target!Stop)$
```

```
$LimitTerms \triangleq$
    $\lor\ mastership.term < NumTerms$
    $\lor\ \land\ mastership.term = NumTerms$
       $\land\ mastership.master \neq Nil$

$LimitConns \triangleq$
    $\forall\, n \in \textsc{domain}\ conn :$
       $\lor\ conn[n].id < NumConns$
       $\lor\ \land\ conn[n].id = NumConns$
          $\land\ conn[n].connected$

$LimitStarts \triangleq$
    $\lor\ target.id < 2$
    $\lor\ \land\ target.id = 2$
       $\land\ target.running$
```

```
$TypeOK \triangleq$
    $\land\ Transaction!TypeOK$
    $\land\ Configuration!TypeOK$
    $\land\ Mastership!TypeOK$

$\textsc{local}\ IsOrderedChange(p,\, i) \triangleq$
    $\land\ \ history[i].type = Change$
    $\land\ \ history[i].phase = p$
    $\land\ \ \neg\exists\, j \in \textsc{domain}\ history :$
              $\land\ j < i$
              $\land\ history[j].type = Change$
              $\land\ history[j].phase = p$
              $\land\ history[j].revision \geq history[i].revision$

$\textsc{local}\ IsOrderedRollback(p,\, i) \triangleq$
    $\land\ \ history[i].type = Rollback$
    $\land\ \ history[i].phase = p$
    $\land\ \ \exists\, j \in \textsc{domain}\ history :$
```

```
            $\land\ j < i$
            $\land\ history[j].type = Change$
            $\land\ history[j].revision = history[i].revision$
    $\land\quad \neg \exists\, j \in \textsc{domain}\ history :$
              $\land\ j < i$
              $\land\ history[j].type = Change$
              $\land\ history[j].phase = p$
              $\land\ history[j].revision > history[i].revision$
              $\land\ \neg \exists\, k \in \textsc{domain}\ history :$
                    $\land\ k > j$
                    $\land\ k < i$
                    $\land\ history[k].type = Rollback$
                    $\land\ history[k].phase = p$
                    $\land\ history[k].revision = history[j].revision$

$Order \ \triangleq$
    $\land\ \forall\, i \in \textsc{domain}\ history :$
        $\lor\ IsOrderedChange(Commit,\ i)$
        $\lor\ IsOrderedChange(Apply,\ i)$
        $\lor\ IsOrderedRollback(Commit,\ i)$
        $\lor\ IsOrderedRollback(Apply,\ i)$
    $\land\ \forall\, i \in \textsc{domain}\ transaction :$
         $\land\ transaction[i].type = Change$
         $\land\ transaction[i].apply = Failed$
         $\land\ \neg \exists\, j \in \textsc{domain}\ transaction :$
               $\land\ transaction[j].type = Rollback$
               $\land\ transaction[j].rollback.revision = transaction[i].change.revision$
               $\land\ transaction[j].apply = Complete$
         $\Rightarrow \forall\, j \in \textsc{domain}\ transaction : (j > i \Rightarrow$
              $(transaction[j].type = Change \Rightarrow transaction[j].apply \neq Complete))$

$Consistency \ \triangleq$
    $\land\ \forall\, i \in \textsc{domain}\ transaction :$
        $\land\ transaction[i].commit = Complete$
        $\land\ \neg \exists\, j \in \textsc{domain}\ transaction :$
              $\land\ j > i$
              $\land\ transaction[j].commit = Complete$
        $\Rightarrow \forall\, p \in \textsc{domain}\ transaction[i].change.values :$
             $\land\ configuration.committed.values[p] = transaction[i].change.values[p]$
    $\land\ \forall\, i \in \textsc{domain}\ transaction :$
        $\land\ transaction[i].apply = Complete$
        $\land\ \neg \exists\, j \in \textsc{domain}\ transaction :$
              $\land\ j > i$
              $\land\ transaction[j].apply = Complete$
        $\Rightarrow \forall\, p \in \textsc{domain}\ transaction[i].change.values :$
```

```
                $\land configuration.applied.values[p] = transaction[i].change.values[p]$
                $\land \; \land target.running$
                    $\land configuration.applied.target = target.id$
                    $\land configuration.state = Complete$
                    $\Rightarrow target.values[p] = transaction[i].change.values[p]$

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

$\textsc{theorem}\ Spec \Rightarrow Safety$

$\textsc{local}\ IsChanging(i) \triangleq$
  $\exists j \in \textsc{domain}\ transaction :$
    $\land transaction[j].type = Change$
    $\land transaction[j].change.revision = i$

$\textsc{local}\ IsChanged(i) \triangleq$
  $\exists j \in \textsc{domain}\ transaction :$
    $\land transaction[j].type = Change$
    $\land transaction[j].change.revision = i$
    $\land transaction[j].commit \in Done$
    $\land transaction[j].apply \in Done$

$\textsc{local}\ IsRollingBack(i) \triangleq$
  $\exists j \in \textsc{domain}\ transaction :$
    $\land transaction[j].type = Rollback$
    $\land transaction[j].rollback.revision = i$

$\textsc{local}\ IsRolledBack(i) \triangleq$
  $\exists j \in \textsc{domain}\ transaction :$
    $\land transaction[j].type = Rollback$
    $\land transaction[j].rollback.revision = i$
    $\land transaction[j].commit \in Done$
    $\land transaction[j].apply \in Done$

$Terminates(i) \triangleq$
   $\land IsChanging(i) \leadsto IsChanged(i)$
   $\land IsRollingBack(i) \leadsto IsRolledBack(i)$

$Termination \triangleq$
  $\forall i \in 1 \,..\, NumTransactions : Terminates(i)$

$Liveness \triangleq Termination$

$\textsc{theorem}\ Spec \Rightarrow Liveness$
```