MODULE *Config*

INSTANCE *Naturals*

INSTANCE *FiniteSets*

INSTANCE *Sequences*

INSTANCE *TLC*

An empty constant
CONSTANT *Nil*

Transaction constants
CONSTANTS
Pending,
Validating,
Applying,
Complete,
Failed

The set of all nodes
CONSTANT *Node*

Target is the possible targets, paths, and values

Example: *Target* $\triangleq$ [
*target*1 $\mapsto$ [
*path*1 $\mapsto$ {"*value*1", "*value*2"},
*path*2 $\mapsto$ {"*value*2", "*value*3"}],
*target*2 $\mapsto$ [
*path*2 $\mapsto$ {"*value*3", "*value*4"},
*path*3 $\mapsto$ {"*value*4", "*value*5"}]]

CONSTANT *Target*

ASSUME *Nil* $\in$ STRING

ASSUME *Pending* $\in$ STRING
ASSUME *Validating* $\in$ STRING
ASSUME *Applying* $\in$ STRING
ASSUME *Complete* $\in$ STRING
ASSUME *Failed* $\in$ STRING

ASSUME $\land$ *IsFiniteSet*(*Node*)
$\land \forall\, n \in$ *Node* :
$\land\ n \notin$ DOMAIN *Target*
$\land\ n \in$ STRING

ASSUME $\land \forall\, t \in$ DOMAIN $Target$:
$\land\ IsFiniteSet(Target[t])$
$\land\ t \notin Node$
$\land\ t \in$ STRING

TYPE Status ::= $status \in \{Pending, Validating, Applying, Complete, Failed\}$

TYPE Transaction $\triangleq$ [
id ::= $id \in$ STRING,
$index$ ::= $index \in Nat$,
revision ::= revision $\in Nat$,
$atomic$ ::= $atomic \in$ BOOLEAN ,
$sync$ ::= $sync \in$ BOOLEAN ,
$changes$ ::= [$target \in$ SUBSET (DOMAIN $Target$) $\mapsto$ [
$path \in$ SUBSET (DOMAIN $Target[target]$) $\mapsto$ [
$value$ ::= $value \in$ STRING,
$delete$ ::= $delete \in$ BOOLEAN]]],
$status$ ::= $status \in$ Status]

TYPE Configuration $\triangleq$ [
id ::= $id \in$ STRING,
revision ::= revision $\in Nat$,
$target$ ::= $target \in$ STRING, $paths$ ::= [
$path \in$ SUBSET (DOMAIN $Target[target]$) $\mapsto$ [
$value$ ::= $value \in$ STRING,
$index$ ::= $index \in Nat$,
$deleted$ ::= $delete \in$ BOOLEAN]],
$transactionIndex$ ::= $transactionIndex \in Nat$,
$syncIndex$ ::= $syncIndex \in Nat$,
$mastershipTerm$ ::= $mastershipTerm \in Nat$]

A sequence of transactions
Each transactions contains a record of 'changes' for a set of targets
VARIABLE $transactions$

A record of target configurations
Each configuration represents the desired state of the target
VARIABLE $configurations$

A record of target states
VARIABLE $targets$

A record of target masters
VARIABLE $masters$

$vars \triangleq \langle transactions, configurations, targets \rangle$

This section models the northbound API for the configuration service.

```
This crazy thing returns the set of all possible sets of valid changes
ValidChanges ≜
  LET allPaths ≜ UNION {(DOMAIN Target[t]) : t ∈ DOMAIN Target}
      allValues ≜ UNION {UNION {Target[t][p] : p ∈ DOMAIN Target[t]} : t ∈ DOMAIN Target}
  IN
    {targetPathValues ∈ SUBSET (Target × allPaths × allValues × BOOLEAN ) :
      ∧ ∀ target ∈ DOMAIN Target :
        LET targetIndexes ≜ {i ∈ 1 .. Len(targetPathValues) : ∧ targetPathValues[i][1] = target}
        IN   ∨ Cardinality(targetIndexes) = 0
             ∨ ∧ Cardinality(targetIndexes) = 1
               ∧ LET targetPathValue ≜ targetPathValues[CHOOSE index ∈ targetIndexes : TRUE]
                 IN
                   ∧ targetPathValue[2] \ (DOMAIN Target[target]) = {}
                   ∧ targetPathValue[3] ∈ Target[target][targetPathValue[2]]}

Add a set of changes to the transaction log
Change ≜
  ∧ ∃ changes ∈ ValidChanges :
      ∧ transactions' = Append(transactions, [index   ↦ Len(transactions) + 1,
                                              atomic  ↦ FALSE,
                                              sync    ↦ FALSE,
                                              changes ↦ changes,
                                              status  ↦ Pending])
  ∧ UNCHANGED ⟨configurations, targets⟩
```

This section models the Transaction log reconciler.

```
Reconcile the transaction log
ReconcileTransaction(n, tx) ≜
    If the transaction is Pending, begin validation if the prior transaction
    has already been applied. This simplifies concurrency control in the controller
    and guarantees transactions are applied to the configurations in sequential order.
  ∧  ∨  ∧ tx.status = Pending
        ∧  ∨  ∧ tx.index > 1
              ∧ transactions[tx.index − 1].status ∈ {Complete, Failed}
           ∨ tx.index = 1
        ∧ transactions' = [transactions EXCEPT ![tx.index].status = Validating]
        ∧ UNCHANGED ⟨configurations⟩
    If the transaction is in the Validating state, compute and validate the
    Configuration for each target.
     ∨  ∧ tx.status = Validating
        If validation fails any target, mark the transaction Failed.
        If validation is successful, proceed to Applying.
        ∧ ∃ valid ∈ BOOLEAN :
            ∨ ∧ valid
```

$\land$ $transactions' = [transactions$ EXCEPT $![tx.index].status = Applying]$
$\lor \land \neg valid$
$\land$ $transactions' = [transactions$ EXCEPT $![tx.index].status = Failed]$
$\land$ UNCHANGED $\langle configurations \rangle$

If the transaction is in the *Applying* state, update the Configuration for each target and *Complete* the transaction.

$\lor \land tx.status = Applying$
$\land \lor \land tx.atomic$

TODO: Apply atomic transactions here

$\land$ $transactions' = [transactions$ EXCEPT $![tx.index].status = Complete]$
$\land$ UNCHANGED $\langle configurations \rangle$
$\land \lor \land \neg tx.atomic$

Add the transaction index to each updated path

$\land$ $configurations' = [$
$t \in$ DOMAIN $Target \mapsto [$
$configurations[t]$ EXCEPT
$!.paths = [path \in$ DOMAIN $tx.changes \mapsto$
$tx.changes[path] @@ [index \mapsto tx.index]] @@ configurations[t].paths,$
$!.transactionIndex = tx.index]]$
$\land$ $transactions' = [transactions$ EXCEPT $![tx.index].status = Complete]$
$\land$ UNCHANGED $\langle targets \rangle$

This section models the Configuration reconciler.

$ReconcileConfiguration(n,\ c) \triangleq$

Only the master should reconcile the configuration

$\land\ masters[c.target].master = n$

If the configuration's mastership term is less than the current mastership term, assume the target may have restarted/reconnected and perform a full reconciliation of the target configuration from the root path.

$\land \lor \land masters[c.target].term > c.mastershipTerm$

Merge the configuration paths with the target paths, removing paths that have been marked deleted

$\land$ $targets' = [targets$ EXCEPT $![c.target] =$
$[p \in \{p \in$ DOMAIN $c.paths : \neg c.paths[p].deleted\} \mapsto [value \mapsto c.paths[p]]] @@$
$[p \in \{p \in$ DOMAIN $targets[c.target] : \neg c.paths[p].deleted\} \mapsto targets[c.target][p]]]$

Set the configuration's mastership term and *sync* index

$\land$ $configurations' = [configurations$ EXCEPT $![c.id].mastershipTerm = masters[c.target].term,$
$![c.id].syncIndex = c.transactionIndex]$

If the Configuration's transaction index is greater than the target index, reconcile the configuration with the target. Once the target has been updated, update the *sync* index to match the reconciled transaction index.

$\land \lor \land masters[c.target].term = c.mastershipTerm$
$\land\ c.transactionIndex > c.syncIndex$

```
        Compute the set of updated and deleted paths by comparing
        their indexes to the target's last sync index.
      ∧ LET updatedPaths ≜ {p ∈ DOMAIN c.paths : c.paths[p].index > c.syncIndex}
            deletedPaths ≜ {p ∈ updatedPaths : c.paths[p].deleted}
        IN
          Update the target paths by adding/updating paths that have changed and
          removing paths that have been deleted since the last sync.
          ∧ targets' = [targets EXCEPT ![c.target] =
                [p ∈ updatedPaths \ deletedPaths ↦ c.paths[p]] @@
                [p ∈ DOMAIN targets[c.target] \ deletedPaths ↦ targets[c.target][p]]]
      ∧ configurations' = [configurations EXCEPT ![c.id].syncIndex = c.transactionIndex]
  ∧ UNCHANGED ⟨transactions⟩
```

```
  Init and next state predicates
Init ≜
  ∧ transactions = ⟨⟩
  ∧ configurations = [t ∈ Target ↦ [
                        id      ↦ t,
                        config ↦ [path ∈ {} ↦ [
                                    path    ↦ path,
                                    value   ↦ Nil,
                                    index   ↦ 0,
                                    deleted ↦ FALSE]]]]
  ∧ targets  = [t ∈ Target ↦
                  [path ∈ {} ↦ [
                      value ↦ Nil]]]
  ∧ masters = [t ∈ Target ↦ [master ↦ Nil, term ↦ 0]]

Next ≜
  ∨ Change
  ∨ ∃ n ∈ Node :
      ∃ t ∈ DOMAIN transactions :
        ReconcileTransaction(n, t)
  ∨ ∃ n ∈ Node :
      ∃ c ∈ configurations :
        ReconcileConfiguration(n, c)

Spec ≜ Init ∧ □[Next]_vars
```

```
\* Modification History
\* Last modified Thu Jan 13 15:45:19 PST 2022 by jordanhalterman
\* Created Wed Sep 22 13:22:32 PDT 2021 by jordanhalterman
```