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*

The set of all targets
CONSTANT *Target*

The set of available paths
CONSTANT *Path*

The set of available values
CONSTANT *Value*

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 $Target :$
$\land\, IsFiniteSet(Target[t])$

$\land\ t \notin Node$
$\land\ t \in \text{STRING}$

TYPE $Change \triangleq$ [
$target ::= target \in \text{STRING}$,
$path ::= path \in \text{STRING}$,
$value ::= value \in \text{STRING}$,
$delete ::= delete \in \text{BOOLEAN}$
]

TYPE State $\triangleq$ $state \in \{Pending, Validating, Applying, Complete, Failed\}$

TYPE Transaction $\triangleq$ [
$id ::= id \in \text{STRING}$,
$index ::= index \in Nat$,
revision ::= revision $\in Nat$,
$atomic ::= atomic \in \text{BOOLEAN}$,
$sync ::= sync \in \text{BOOLEAN}$,
$changes ::= [i \in 1 \,..\, Nat \mapsto changes[i] \in Change]$,
$status ::= [state ::= state \in$ State]]

TYPE Element $\triangleq$ [
$path ::= path \in \text{STRING}$,
$value ::= value \in \text{STRING}$,
$index ::= index \in Nat$,
$deleted ::= deleted \in \text{BOOLEAN}$]

TYPE Configuration $\triangleq$ [
$id ::= id \in \text{STRING}$,
revision ::= revision $\in Nat$,
$target ::= target \in \text{STRING}$,
$elements ::= [i \in 1 \,..\, Nat \mapsto elements[i] \in$ Element],
$status ::= [$
$transactionIndex ::= transactionIndex \in Nat$,
$targetIndex ::= targetIndex \in Nat$,
$mastershipTerm ::= mastershipTerm \in Nat$]]

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*

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

$paths \triangleq Seq(Path)$

$values \triangleq Seq(Value)$

This section models the northbound *API* for the configuration service.

Changes a set of paths/values on a set of targets

$$
\begin{array}{l}
Change(n,\ ts,\ d) \triangleq \\
\quad \land \quad \text{LET } tss \triangleq Seq(ts) \\
\qquad \text{IN} \\
\qquad\quad \land\ transactions' = Append(transactions, [index \mapsto Len(transactions) + 1, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad atomic \mapsto \text{FALSE}, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad sync \mapsto \text{FALSE}, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad changes \mapsto [i \in 1\ ..\ Len(tss) \mapsto [\\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad target \mapsto tss[i], \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad path \mapsto paths[(i\%Len(paths)) + 1], \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad value \mapsto values[(i\%Len(values)) + 1], \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad delete \mapsto d]], \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad status \mapsto [state \mapsto Pending]]) \\
\quad \land \quad \text{UNCHANGED } \langle configurations,\ targets \rangle
\end{array}
$$

This section models the Transaction *log* reconciler.

$RemoveElement(elements,\ path) \triangleq [i \in \{e \in \text{DOMAIN } elements : elements[e].path \neq path\} \mapsto elements[i]]$

$AddElement(elements,\ element) \triangleq Append(elements,\ element)$

$UpdateElement(elements,\ element) \triangleq AddElement(RemoveElement(elements,\ element.path),\ element)$

$Paths(elements,\ changes) \triangleq \{e.path : e \in elements\} \cup \{c.path : c \in elements\}$

$$
\begin{array}{l}
UpdateElements(elements,\ changes) \triangleq \\
\quad \text{LET } configPaths \triangleq \{e.path : e \in elements\} \\
\qquad\quad configMap \triangleq [path \in configPaths \mapsto \text{CHOOSE } e \in elements : e.path = path] \\
\qquad\quad changePaths \triangleq \{c.path : c \in changes\} \\
\qquad\quad changeMap \triangleq [path \in changePaths \mapsto \text{CHOOSE } c \in changes : c.path = path] \\
\qquad\quad allPaths \triangleq configPaths \cup changePaths \\
\quad \text{IN} \\
\qquad Seq(\{\text{IF } path \in \text{DOMAIN } changeMap \text{ THEN } changeMap[path] \text{ ELSE } configMap[path] : path \in allPaths\})
\end{array}
$$

Reconcile the transaction *log*

$ReconcileTransaction(n,\ tx) \triangleq$

```
        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.state = Pending
        ∧ ∨ ∧ tx.index > 1
              ∧ transactions[tx.index − 1].status.state ∈ {Complete, Failed}
          ∨ tx.index = 1
        ∧ transactions′ = [transactions EXCEPT ![tx.index].status.state = Validating]
        ∧ UNCHANGED ⟨configurations⟩
        If the transaction is in the Validating state, compute and validate the
        Configuration for each target. Mark the transaction Failed if validation
        fails any target.
        TODO: Model validation failures here
      ∨ ∧ tx.status.state = Validating
          Validate the target configurations here
          Then change the transaction state to Applying
        ∧ transactions′ = [transactions EXCEPT ![tx.index].status.state = Applying]
        ∧ UNCHANGED ⟨configurations⟩
        If the transaction is in the Applying state, update the Configuration for each
        target and Complete the transaction.
      ∨ ∧ tx.status.state = Applying
        ∧ ∨ ∧ tx.atomic
              TODO: Apply atomic transactions here
            ∧ transactions′ = [transactions EXCEPT ![tx.index].status.state = Complete]
            ∧ UNCHANGED ⟨configurations⟩
        ∧ ∨ ∧ ¬tx.atomic
            ∧ configurations′ = [t ∈ Target ↦ [
                                    configurations[t] EXCEPT !.elements = UpdateElements(configurations[t].
                                                             !.status.transactionIndex = tx.index]]
            ∧ transactions′ = [transactions EXCEPT ![tx.index].status.state = Complete]
    ∧ UNCHANGED ⟨targets⟩

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

This section models the Configuration reconciler.
ReconcileConfiguration(n, c) ≜
    Only the master should reconcile the configuration
    ∧ 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.
    ∧ ∨ ∧ masters[c.target].term > c.status.mastershipTerm
          TODO: Reconcile the target state here
        ∧ configurations′ = [configurations EXCEPT ![c.id].status.mastershipTerm = masters[c.target].term,
                                                   ![c.id].status.targetIndex = c.status.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 target index to match the reconciled transaction index.

$\land \lor \land masters[c.target].term = c.status.mastershipTerm$
$\quad \land c.status.transactionIndex > c.status.targetIndex$
$\quad$ *TODO*: Reconcile the target state here
$\quad \land configurations' = [configurations \text{ EXCEPT } ![c.id].status.targetIndex = c.status.transactionIndex]$
$\land \text{UNCHANGED } \langle transactions \rangle$

Init and next state predicates

$$
\begin{aligned}
&Init \triangleq \\
&\quad \land transactions = \langle\rangle \\
&\quad \land configurations = [t \in Target \mapsto [\\
&\qquad\qquad id \quad\quad \mapsto t, \\
&\qquad\qquad config \mapsto [path \in \{\} \mapsto [\\
&\qquad\qquad\qquad\qquad path \quad \mapsto path, \\
&\qquad\qquad\qquad\qquad value \quad \mapsto Nil, \\
&\qquad\qquad\qquad\qquad index \quad \mapsto 0, \\
&\qquad\qquad\qquad\qquad deleted \mapsto \text{FALSE}]]]] \\
&\quad \land targets \;\; = [t \in Target \mapsto [\\
&\qquad\qquad id \quad\quad \mapsto t, \\
&\qquad\qquad config \mapsto [path \in \{\} \mapsto [\\
&\qquad\qquad\qquad\qquad path \mapsto path, \\
&\qquad\qquad\qquad\qquad value \mapsto Nil]]]] \\
&\quad \land masters = [t \in Target \mapsto [master \mapsto Nil,\ term \mapsto 0]]
\end{aligned}
$$

$$
\begin{aligned}
&Next \triangleq \\
&\quad \lor \exists\, n \in Node : \\
&\qquad \exists\, ts \in \text{SUBSET } Target : \\
&\qquad\quad \exists\, b \in \text{BOOLEAN} : \\
&\qquad\qquad Change(n,\ ts,\ b) \\
&\quad \lor \exists\, n \in Node : \\
&\qquad \exists\, t \in \text{DOMAIN } transactions : \\
&\qquad\quad ReconcileTransaction(n,\ t) \\
&\quad \lor \exists\, n \in Node : \\
&\qquad \exists\, c \in configurations : \\
&\qquad\quad ReconcileConfiguration(n,\ c)
\end{aligned}
$$

$$Spec \triangleq Init \land \Box[Next]_{vars}$$

---

* Modification History
* Last modified *Thu Jan* 13 04:08:17 *PST* 2022 by *jordanhalterman*
* Created *Wed Sep* 22 13:22:32 *PDT* 2021 by *jordanhalterman*