MODULE *Config*

EXTENDS
Northbound,
Proposal,
Configuration,
Mastership,
Southbound

INSTANCE *Naturals*

INSTANCE *FiniteSets*

INSTANCE *Sequences*

LOCAL INSTANCE *TLC*

$vars \triangleq \langle proposal, configuration, mastership, node, target \rangle$

Formal specification, constraints, and theorems.

$Init \triangleq$
$\land\ InitNorthbound$
$\land\ InitProposal$
$\land\ InitConfiguration$
$\land\ InitMastership$
$\land\ InitSouthbound$

$Next \triangleq$
$\lor\ \land\ NextNorthbound$
$\quad\land$ UNCHANGED $\langle\rangle$
$\lor\ \land\ NextProposal$
$\quad\land$ UNCHANGED $\langle\rangle$
$\lor\ \land\ NextConfiguration$
$\quad\land$ UNCHANGED $\langle proposal \rangle$
$\lor\ \land\ NextMastership$
$\quad\land$ UNCHANGED $\langle proposal, configuration \rangle$
$\lor\ \land\ NextSouthbound$
$\quad\land$ UNCHANGED $\langle proposal, configuration, mastership \rangle$

$Spec \triangleq Init \land \Box[Next]_{vars} \land \mathrm{WF}_{vars}(Next)$

$IsCommittedChange(i) \triangleq$
$\land\ proposal[i].state = ProposalChange$
$\land\ \lor\ \land\ proposal[i].change.phase = ProposalCommit$
$\qquad\land\ proposal[i].change.status = ProposalFailed$
$\quad\lor\ proposal[i].change.phase = ProposalApply$

$$
\begin{array}{l}
IsAppliedChange(i) \triangleq \\
\quad \land\ proposal[i].state = ProposalChange \\
\quad \land\ proposal[i].change.phase = ProposalApply \\
\quad \land\ proposal[i].change.status = ProposalComplete
\end{array}
$$

$$
\begin{array}{l}
IsCommittedRollback(i) \triangleq \\
\quad \land\ proposal[i].state = ProposalRollback \\
\quad \land\ \lor\ \land\ proposal[i].change.phase = ProposalCommit \\
\qquad\quad\ \ \land\ proposal[i].change.status = ProposalFailed \\
\qquad \lor\ proposal[i].change.phase = ProposalApply
\end{array}
$$

$$
\begin{array}{l}
IsAppliedRollback(i) \triangleq \\
\quad \land\ proposal[i].state = ProposalRollback \\
\quad \land\ \lor\ proposal[i].rollback.phase = ProposalCommit \\
\qquad \lor\ \land\ proposal[i].rollback.phase = ProposalApply \\
\qquad\quad\ \ \land\ proposal[i].rollback.status \in \{ProposalPending,\ ProposalComplete\}
\end{array}
$$

$$
\begin{array}{l}
Order \triangleq \\
\quad \forall\, i \in \text{DOMAIN}\ proposal : \\
\qquad \land\ IsCommittedChange(i) \Rightarrow \\
\qquad\quad \forall\, j \in \text{DOMAIN}\ proposal : j < i \Rightarrow \\
\qquad\qquad \land\ proposal[j].state = ProposalChange \Rightarrow IsCommittedChange(j) \\
\qquad\qquad \land\ proposal[j].state = ProposalRollback \Rightarrow IsCommittedRollback(j) \\
\qquad \land\ IsAppliedChange(i) \Rightarrow \\
\qquad\quad \forall\, j \in \text{DOMAIN}\ proposal : j < i \Rightarrow \\
\qquad\qquad \land\ proposal[j].state = ProposalChange \Rightarrow IsAppliedChange(j) \\
\qquad\qquad \land\ proposal[j].state = ProposalRollback \Rightarrow IsAppliedRollback(j)
\end{array}
$$

$$
\begin{array}{l}
Consistency \triangleq \\
\quad \land\ target.running \\
\quad \land\ configuration.state = ConfigurationComplete \\
\quad \land\ configuration.apply.incarnation = target.incarnation \\
\quad \Rightarrow \forall\, i \in \text{DOMAIN}\ proposal : \\
\qquad\quad IsAppliedChange(i) \Rightarrow \\
\qquad\qquad\quad \forall\, p \in \text{DOMAIN}\ proposal[i].change.values : \\
\qquad\qquad\qquad \land\ \neg\exists\, j \in \text{DOMAIN}\ proposal : \\
\qquad\qquad\qquad\qquad\quad \land\ j > i \\
\qquad\qquad\qquad\qquad\quad \land\ proposal[j].change.phase = ProposalApply \\
\qquad\qquad\qquad\qquad\quad \land\ proposal[j].change.status = ProposalComplete \\
\qquad\qquad\qquad\qquad\quad \land\ proposal[j].rollback.phase = ProposalApply \\
\qquad\qquad\qquad\qquad\qquad\ \ \Rightarrow proposal[j].rollback.status \neq ProposalComplete \\
\qquad\qquad\qquad\qquad\quad \land\ p \in \text{DOMAIN}\ proposal[j].change.values \\
\qquad\qquad\qquad \Rightarrow\ \land\ p \in \text{DOMAIN}\ target.values \\
\qquad\qquad\qquad\qquad\ \ \land\ target.values[p].value = proposal[i].change.values[p].value \\
\qquad\qquad\qquad\qquad\ \ \land\ target.values[p].index = proposal[i].change.values[p].index
\end{array}
$$

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

THEOREM $Spec \Rightarrow Safety$

$$
\begin{array}{l}
Terminates(i) \triangleq \\
\quad \land\ proposal[i].state = ProposalChange \leadsto \\
\qquad \land\ proposal[i].change.phase = ProposalApply \\
\qquad \land\ proposal[i].change.status = ProposalComplete \\
\quad \land\ proposal[i].state = ProposalRollback \leadsto \\
\qquad \land\ proposal[i].rollback.phase = ProposalApply \\
\qquad \land\ proposal[i].rollback.status = ProposalComplete
\end{array}
$$

$$
\begin{array}{l}
Termination \triangleq \\
\quad \forall\, i \in 1 \,..\, NumProposals : Terminates(i)
\end{array}
$$

$Liveness \triangleq Termination$

THEOREM $Spec \Rightarrow Liveness$

* Modification History
* Last modified *Fri Apr* 21 18:30:03 *PDT* 2023 by *jhalterm*
* Last modified *Mon Feb* 21 01:32:07 *PST* 2022 by *jordanhalterman*
* Created *Wed Sep* 22 13:22:32 *PDT* 2021 by *jordanhalterman*