MODULE *RaHeartbeat*

EXTENDS *Naturals*, *FiniteSets*, *Sequences*, *TLC*

Is leader *ALIVE* or *CRASHED*
VARIABLE *leaderState*

Helper variable tracking the number of times the leader node has failed
VARIABLE *nodedownIndex*

A collection of 'nodedown' messages the leader has sent.
In *Erlang*, when a monitored node crashes, the 'nodedown' message is sent to the follower.
The value of a message is not a part of the implementation and
represents the number of times the node has failed.
VARIABLE *nodedownMessages*

$nodedownInfo \triangleq \langle nodedownMessages, nodedownIndex \rangle$

Heartbeat messages the leader has sent
The value of a message is not a part of the implementation and
represents the number of times a heartbeat message was sent.
VARIABLE *heartbeatMessages*

Helper variable tracking the number of sent heartbeat messages
VARIABLE *heartbeatIndex*

$heartbeatInfo \triangleq \langle heartbeatMessages, heartbeatIndex \rangle$

Whether the follower timed out
VARIABLE *isTimeout*

$vars \triangleq \langle leaderState, nodedownInfo, heartbeatInfo, isTimeout \rangle$

The leader crashes and doesn't recover
$CrashLeader \triangleq$
$\quad \land leaderState = \text{“ALIVE”}$
$\quad \land leaderState' = \text{“CRASHED”}$
Erlang sends the *NODEDOWN* message if the leader was monitored
$\quad \land nodedownMessages' = Append(nodedownMessages, nodedownIndex)$
$\quad \land nodedownIndex' = nodedownIndex + 1$
$\quad \land$ UNCHANGED $\langle heartbeatInfo, isTimeout \rangle$

Helper function to remove a message from a sequence of messages
$RemoveMessage(i, seq) \triangleq$
$\quad [j \in 1 \,..\, Len(seq) - 1 \mapsto$ IF $j < i$ THEN $seq[j]$ ELSE $seq[j + 1]]$

The network drops a heartbeat message
$DropHeartbeat \triangleq$

$\land\ Len(heartbeatMessages) \geq 1$
$\land\ \exists\, i \in 1\,..\,Len(heartbeatMessages) :$
$\quad heartbeatMessages' = RemoveMessage(i,\ heartbeatMessages)$
$\land$ UNCHANGED $\langle leaderState,\ heartbeatIndex,\ nodedownInfo,\ isTimeout\rangle$

The follower receives a heartbeat message from the leader.

$ReceiveHeartbeat \triangleq$
$\land\ Len(heartbeatMessages) \geq 1$
$\land\ \exists\, i \in 1\,..\,Len(heartbeatMessages) :$
$\quad heartbeatMessages' = RemoveMessage(i,\ heartbeatMessages)$
$\land$ UNCHANGED $\langle leaderState,\ heartbeatIndex,\ nodedownInfo,\ isTimeout\rangle$

The leader sends a heartbeat message to the follower.

$SendHeartbeat \triangleq$
$\land\ leaderState =$ "ALIVE"
$\land\ heartbeatMessages' = Append(heartbeatMessages,\ heartbeatIndex)$
$\land\ heartbeatIndex' = heartbeatIndex + 1$
$\land$ UNCHANGED $\langle leaderState,\ nodedownInfo,\ isTimeout\rangle$

The network drops a nodedown message.

$DropNodedown \triangleq$
$\land\ Len(nodedownMessages) \geq 1$
$\land\ \exists\, i \in 1\,..\,Len(nodedownMessages) :$
$\quad nodedownMessages' = RemoveMessage(i,\ nodedownMessages)$
$\land$ UNCHANGED $\langle leaderState,\ nodedownIndex,\ heartbeatInfo,\ isTimeout\rangle$

The follower receives a nodedown message from the leader which causes it time out immediately.

$ReceiveNodedown \triangleq$
$\land\ Len(nodedownMessages) \geq 1$
$\land\ \exists\, i \in 1\,..\,Len(nodedownMessages) :$
$\quad nodedownMessages' = RemoveMessage(i,\ nodedownMessages)$
$\land\ isTimeout' =$ TRUE
$\land$ UNCHANGED $\langle leaderState,\ nodedownIndex,\ heartbeatInfo\rangle$

$Timeout \triangleq$
$\land\ isTimeout' =$ TRUE
$\land$ UNCHANGED $\langle leaderState,\ heartbeatInfo,\ nodedownInfo\rangle$

Initial state of model

$Init \triangleq$ $\land\ leaderState =$ "ALIVE"
$\land\ nodedownIndex = 0$
$\land\ nodedownMessages = \langle\rangle$
$\land\ heartbeatIndex = 0$
$\land\ heartbeatMessages = \langle\rangle$
$\land\ isTimeout =$ FALSE

Next state function

$$
\begin{aligned}
Next \triangleq\ & \land \neg isTimeout \\
& \land \ \lor DropHeartbeat \\
& \quad \lor SendHeartbeat \\
& \quad \lor ReceiveHeartbeat \\
& \quad \lor DropNodedown \\
& \quad \lor ReceiveNodedown \\
& \quad \lor CrashLeader
\end{aligned}
$$

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