MODULE $E2T$

LOCAL INSTANCE $Naturals$

LOCAL INSTANCE $Sequences$

LOCAL INSTANCE $FiniteSets$

LOCAL INSTANCE TLC

An empty value
CONSTANT Nil

Node states
CONSTANT $Stopped$, $Started$

A set of $E2T$ node identifiers
CONSTANT $E2Term$

ASSUME $\land IsFiniteSet(E2Term)$
$\land Cardinality(E2Term) > 0$
$\land \forall\, n \in E2Term : n \in \text{STRING}$

A mapping of node states
VARIABLE $state$

$gRPC$ connection states
VARIABLE $grpc$

$SCTP$ connection states
VARIABLE $sctp$

A global store of mastership for each $E2$ node
VARIABLE $masterships$

A global store of configuration for each $E2$ node
VARIABLE $nodes$

A global store of connections for each $E2$ node
VARIABLE $conns$

A node local store of outstanding transactions
VARIABLE $txID$, txs

A node local store of outstanding requests
VARIABLE $reqID$, $reqs$

A store of streams for each node
VARIABLE $streams$

A global store of channel states
VARIABLE *chans*

A global store of subscription states
VARIABLE *subs*

$vars \triangleq \langle state, masterships, grpc, sctp, streams, chans, subs \rangle$

LOCAL $API \triangleq$ INSTANCE $E2TService$ WITH $conns \leftarrow grpc$

LOCAL $E2AP \triangleq$ INSTANCE $E2AP$ WITH $conns \leftarrow sctp$

$StartNode(e2TermID) \triangleq$
$\land\ state[e2TermID] = Stopped$
$\land\ state' = [state \text{ EXCEPT } ![e2TermID] = Started]$
$\land\ E2AP!Server(e2TermID)!Start$
$\land$ UNCHANGED $\langle masterships, conns, streams, chans, subs \rangle$

$StopNode(e2TermID) \triangleq$
$\land\ state[e2TermID] = Started$
$\land\ state' = [state \text{ EXCEPT } ![e2TermID] = Stopped]$
$\land\ E2AP!Server(e2TermID)!Start$
$\land\ streams' = [streams \text{ EXCEPT } ![e2TermID] = [id \in \{\} \mapsto [id \mapsto Nil]]]$
$\land\ txs' = [txs \text{ EXCEPT } ![e2TermID] = [id \in \{\} \mapsto [txID \mapsto id]]]$
$\land\ txID' = [txID \text{ EXCEPT } ![e2TermID] = 0]$
$\land\ reqs' = [reqs \text{ EXCEPT } ![e2TermID] = [id \in \{\} \mapsto [reqID \mapsto id]]]$
$\land\ reqID' = [reqID \text{ EXCEPT } ![e2TermID] = 0]$
$\land$ UNCHANGED $\langle masterships, conns, chans, subs \rangle$

$HandleSubscribeRequest(e2TermID, apiConn, apiReq) \triangleq$
$\land\ \lor\ \land\ apiReq.sub.id \notin streams[e2TermID]$
$\quad\land\ streams' = [streams \text{ EXCEPT } ![e2TermID] = streams[e2TermID] \mathbin{@@} (apiReq.sub.id :> [id \mapsto apiReq$
$\quad\lor\ \land\ apiReq.sub.id \in streams[e2TermID]$
$\quad\land$ UNCHANGED $\langle streams \rangle$
$\land$ UNCHANGED $\langle chans, subs \rangle$

$SendSubscribeResponse(e2TermID, apiConn, s) \triangleq$
$\land\ Len(streams[e2TermID][s]) > 0$
$\land\ API!Server!Send!SubscribeResponse(apiConn, [indication \mapsto streams[e2TermID][s][1]])$
$\land\ streams' = [streams \text{ EXCEPT } ![e2TermID] = [streams[e2TermID] \text{ EXCEPT } ![s] = SubSeq(streams[e2Tern$
$\land$ UNCHANGED $\langle chans, subs \rangle$

$HandleUnsubscribeRequest(e2TermID, apiConn, apiReq) \triangleq$
$\land\ \lor\ \land\ apiReq.sub.id \notin streams[e2TermID]$

$\land\ streams' = [streams \text{ EXCEPT } ![e2TermID] = [i \in \{subId \in \text{DOMAIN } streams[e2TermID] : subId \neq a$
$\lor\ \land\ apiReq.sub.id \in streams[e2TermID]$
$\land\ \text{UNCHANGED } \langle streams \rangle$
$\land\ API!Server!Reply!UnsubscribeResponse(apiConn, [id \mapsto apiReq.id])$
$\land\ \text{UNCHANGED } \langle chans, subs \rangle$

$HandleControlRequest(e2TermID, apiConn, apiReq) \triangleq$
$\land\ API!Server!Reply!ControlResponse(apiConn, [foo \mapsto \text{“bar”}, bar \mapsto \text{“baz”}])$
$\land\ \text{UNCHANGED } \langle chans, subs \rangle$

$HandleE2TRequest(e2TermID, apiConn) \triangleq$
$\land\ \lor API!Server!Handle!SubscribeRequest(apiConn, \text{LAMBDA } m : HandleSubscribeRequest(e2TermID, ap$
$\lor API!Server!Handle!UnsubscribeRequest(apiConn, \text{LAMBDA } m : HandleUnsubscribeRequest(e2TermI$
$\lor API!Server!Handle!ControlRequest(apiConn, \text{LAMBDA } m : HandleControlRequest(e2TermID, apiCo$
$\land\ \text{UNCHANGED } \langle state \rangle$

$ReconcileMastership(e2TermID, e2NodeID) \triangleq$
$\land\ mastership[e2NodeID].master \notin \text{DOMAIN } conns[e2NodeID]$
$\land\ \exists\, c \in \text{DOMAIN } conns[e2NodeID] : c \neq mastership[e2NodeID].master$
$\land\ mastership' = [mastership \text{ EXCEPT } ![e2NodeID] = [$
$term \mapsto mastership[e2NodeID].term + 1,$
$conn \mapsto \text{CHOOSE } c \in \text{DOMAIN } conns[e2NodeID] : c \neq mastership[e2NodeID].master]$
$\land\ \text{UNCHANGED } \langle state, subs \rangle$

$ReconcileStream(e2TermID, streamID) \triangleq$
$\land\ \text{UNCHANGED } \langle state, subs \rangle$

ReconcileChannel reconciles a channel's state
$ReconcileChannel(e2TermID, chanID) \triangleq$
$\land\ \text{UNCHANGED } \langle state, streams \rangle$

ReconcileSubscription reconciles a subscription's state
$ReconcileSubscription(e2TermID, subID) \triangleq$
$\land\ \text{UNCHANGED } \langle state, streams, chans \rangle$

ReconcileConfiguration reconciles an $E2$ node configuration
$ReconcileConfiguration(e2TermID, e2NodeID) \triangleq$
$\land\ \text{UNCHANGED } \langle state, streams, chans \rangle$

$HandleE2SetupRequest(e2TermID, e2apConn, e2apReq) \triangleq$
$\land\ E2AP!Server(e2TermID)!Receive!E2SetupRequest(e2apConn, e2apReq)$
$\land\ E2AP!Server(e2TermID)!Reply!E2SetupResponse(e2apConn, [foo \mapsto \text{“bar”}, bar \mapsto \text{“baz”}])$
$\land\ \text{UNCHANGED } \langle chans, subs \rangle$

$HandleRICControlResponse(e2TermID,\ e2apConn,\ e2apRes) \triangleq$
$\land\ E2AP!Server(e2TermID)!Receive!RICControlResponse(e2apConn,\ e2apRes)$
$\land$ UNCHANGED $\langle chans,\ subs \rangle$

$HandleRICSubscriptionResponse(e2TermID,\ e2apConn,\ e2apRes) \triangleq$
$\land\ E2AP!Server(e2TermID)!Receive!RICSubscriptionResponse(e2apConn,\ e2apRes)$
$\land$ UNCHANGED $\langle chans,\ subs \rangle$

$HandleRICSubscriptionDeleteResponse(e2TermID,\ e2apConn,\ e2apRes) \triangleq$
$\land\ E2AP!Server(e2TermID)!Receive!RICSubscriptionDeleteResponse(e2apConn,\ e2apRes)$
$\land$ UNCHANGED $\langle chans,\ subs \rangle$

$HandleRICIndication(e2TermID,\ e2apConn,\ e2apReq) \triangleq$
$\land\ E2AP!Server(e2TermID)!Receive!RICIndication(e2apConn,\ e2apReq)$
$\land$ UNCHANGED $\langle chans,\ subs \rangle$

$HandleE2NodeConfigurationUpdate(e2TermID,\ e2apConn,\ e2apReq) \triangleq$
$\land\ E2AP!Server(e2TermID)!Receive!E2NodeConfigurationUpdate(e2apConn,\ e2apReq)$
$\land$ UNCHANGED $\langle chans,\ subs \rangle$

$HandleE2APRequest(e2TermID,\ e2apConn) \triangleq$
$\land\ \lor E2AP!Server(e2TermID)!Handle!E2SetupRequest(e2apConn,$ LAMBDA $c,\ m : HandleE2SetupReque$
$\lor E2AP!Server(e2TermID)!Handle!RICControlResponse(e2apConn,$ LAMBDA $c,\ m : HandleRICContr$
$\lor E2AP!Server(e2TermID)!Handle!RICSubscriptionResponse(e2apConn,$ LAMBDA $c,\ m : HandleRICS$
$\lor E2AP!Server(e2TermID)!Handle!RICSubscriptionDeleteResponse(e2apConn,$ LAMBDA $c,\ m : Handl$
$\lor E2AP!Server(e2TermID)!Handle!RICIndication(e2apConn,$ LAMBDA $c,\ m : HandleRICIndication(e$
$\lor E2AP!Server(e2TermID)!Handle!RICIndication(e2apConn,$ LAMBDA $c,\ m : HandleE2NodeConfigur$
$\land$ UNCHANGED $\langle state \rangle$

$Init \triangleq$
$\land\ state = [e2TermID \in E2Term \mapsto Stopped]$
$\land\ masterships = [e2TermID \in E2Term \mapsto [e \in \{\} \mapsto [master \mapsto Nil,\ term \mapsto 0]]]$
$\land\ nodes = [e \in \{\} \mapsto [version \mapsto 0,\ conns \mapsto \{\}]]$
$\land\ conns = [e \in \{\} \mapsto [mgmt \mapsto Nil,\ data \mapsto \{\}]]$
$\land\ txs = [e2TermID \in E2Term \mapsto [id \in \{\} \mapsto [txID \mapsto id]]]$
$\land\ txID = [e2TermID \in E2Term \mapsto 0]$
$\land\ reqs = [e2TermID \in E2Term \mapsto [id \in \{\} \mapsto [reqID \mapsto id]]]$
$\land\ reqID = [e2TermID \in E2Term \mapsto 0]$
$\land\ streams = [n \in E2Term \mapsto [x \in \{\} \mapsto [id \mapsto x]]]$
$\land\ chans = [x \in \{\} \mapsto [id \mapsto x]]$
$\land\ subs = [x \in \{\} \mapsto [id \mapsto x]]$

$Next \triangleq$
$\lor \exists\, n \in E2Term :$
$StartNode(n)$

$\lor \exists\, n \in E2Term :$
$\quad StopNode(n)$
$\lor \exists\, n \in E2Term,\ c \in API\,!\,Connections :$
$\quad HandleE2TRequest(n,\ c)$
$\lor \exists\, n \in E2Term,\ c \in API\,!\,Connections :$
$\quad \exists\, s \in \text{DOMAIN}\ streams[n] :$
$\qquad SendSubscribeResponse(n,\ c,\ s)$
$\lor \exists\, n \in E2Term :$
$\quad \exists\, c \in E2AP\,!\,Server(n)\,!\,Connections :$
$\qquad HandleE2APRequest(n,\ c)$
$\lor \exists\, n \in E2Term :$
$\quad \exists\, e \in \text{DOMAIN}\ nodes[n] :$
$\qquad ReconcileMastership(n,\ e)$
$\lor \exists\, n \in E2Term :$
$\quad \exists\, s \in \text{DOMAIN}\ streams[n] :$
$\qquad ReconcileStream(n,\ s)$
$\lor \exists\, n \in E2Term,\ c \in chans :$
$\quad ReconcileChannel(n,\ c)$
$\lor \exists\, n \in E2Term,\ s \in subs :$
$\quad ReconcileSubscription(n,\ s)$

* Modification History
* Last modified *Tue Sep* 21 18:51:04 *PDT* 2021 by *jordanhalterman*
* Created *Mon Sep* 13 03:23:39 *PDT* 2021 by *jordanhalterman*