MODULE *ORSet*

EXTENDS *Naturals*, *Sequences*, *Bags*, *TLC*, *SEC*

CONSTANTS
Data the set of data

$Instance \triangleq [d : Data, r : Replica, k : Nat]$

VARIABLES
sSet, *sSet*[*r*]: set of active *Instance*(*s*) maintained by $r \in Replica$
seq, *seq*[*r*]: local sequence number at replica $r \in Replic$
incoming, *incoming*[*r*]: incoming messages at replica $r \in Replica$
msg,
messageSet

$vars \triangleq \langle sSet, seq, incoming, msg, messageSet, SECvars \rangle$

$Msg \triangleq [r : Replica, S : \text{SUBSET } Instance, seq : Nat, update : \text{SUBSET } Update]$

$Network \triangleq \text{INSTANCE } ReliableNetwork$

$TypeOK \triangleq$
$\land \quad sSet \in [Replica \to \text{SUBSET } Instance]$
$\land \quad seq \in [Replica \to Nat]$

$Init \triangleq$
$\land\ sSet = [r \in Replica \mapsto \{\}]$
$\land\ seq = [r \in Replica \mapsto 0]$
$\land\ Network\,!\,RInit$
$\land\ SECInit$

$Add(d, r) \triangleq$
$\land\ seq' = [seq \text{ EXCEPT } ![r] = @ + 1]$
$\land\ sSet' = [sSet \text{ EXCEPT } ![r] = @ \cup \{[d \mapsto d, r \mapsto r, k \mapsto seq'[r]]\}]$
$\land\ SECUpdate(r, seq[r])$
$\land\ \text{UNCHANGED } \langle incoming, msg, messageSet \rangle$

$Remove(d, r) \triangleq$
$\land\ \text{LET } D \triangleq \{ins \in sSet[r] : ins.d = d\}$
$\quad \text{IN} \quad sSet' = [sSet \text{ EXCEPT } ![r] = @ \setminus D]$
$\land\ seq' = [seq \text{ EXCEPT } ![r] = @ + 1]$
$\land\ SECUpdate(r, seq[r])$
$\land\ \text{UNCHANGED } \langle incoming, msg, messageSet \rangle$

$Broadcast(s, m) \triangleq$
$[r \in Replica \mapsto \text{IF } s = r \text{ THEN } incoming[s]$
$\qquad\qquad\qquad\qquad \text{ELSE } incoming[r] \oplus SetToBag(\{m\})]$

$Send(r) \triangleq$
$\quad \land Network!RBroadcast(r, [r \mapsto r, S \mapsto sSet[r], seq \mapsto seq[r],$
$\quad\quad update \mapsto OpUpdate(r)])$
$\quad \land SECSend(r)$
$\quad \land$ UNCHANGED $\langle sSet, seq \rangle$

$Receive(r) \triangleq$
$\quad \land \; Network!RDeliver(r)$
$\quad \land \; SECDeliver(r, msg'[r])$
$\quad \land \; sSet' = [sSet$ EXCEPT $![r] = @ \cup msg'[r].S]$
$\quad \land \;$ UNCHANGED $\langle seq \rangle$

$Next \triangleq$
$\quad \lor \exists\, r \in Replica : \exists\, a \in Data :$
$\quad\quad Add(a, r) \lor Remove(a, r)$
$\quad \lor \exists\, r \in Replica :$
$\quad\quad Send(r) \lor Receive(r)$

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

$Read(r) \triangleq \{ins.d : ins \in sSet[r]\}$

QC: Quiescent Consistency
$Quiescence \triangleq \forall\, r \in Replica : incoming[r] = \langle\rangle$

$Convergence \triangleq \forall\, r, s \in Replica : Read(r) = Read(s)$

$QC \triangleq Quiescence \Rightarrow Convergence$

SEC: Strong Eventual Consistency
$SEC \triangleq \forall\, r1, r2 \in Replica : SameUpdate(r1, r2) \Rightarrow Read(r1) = Read(r2)$

---

* Modification History
* Last modified *Thu* May 16 10:34:44 *CST* 2019 by *zfwang*
* Created *Wed Feb* 27 17:23:32 *CST* 2019 by *zfwang*