MODULE *JustInTimePaxos*

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

The set of *Paxos* replicas
CONSTANT *Replicas*

The set of *Paxos* clients
CONSTANT *Clients*

The maximum clock interval
CONSTANT *MaxClockInterval*

An empty value
CONSTANT *Nil*

Client request/response types+
CONSTANTS
WriteRequest,
WriteResponse,
ReadRequest,
ReadResponse

Server request/response types
CONSTANTS
ViewChangeRequest,
ViewChangeResponse,
StartViewRequest

Replica roles
CONSTANTS
NormalStatus,
ViewChangeStatus,
RecoveringStatus

VARIABLE *replicas*

$globalVars \triangleq \langle replicas \rangle$

VARIABLE *messages*

$messageVars \triangleq \langle messages \rangle$

VARIABLE *globalTime*

VARIABLE *time*

VARIABLE *requestID*

VARIABLE $responses$

VARIABLE $writes$

VARIABLE $reads$

$clientVars \triangleq \langle globalTime, time, requestID, responses, writes, reads \rangle$

VARIABLE $status$

VARIABLE log

VARIABLE $viewID$

VARIABLE $lastNormalView$

VARIABLE $viewChangeResponses$

$replicaVars \triangleq \langle status, log, viewID, lastNormalView, viewChangeResponses \rangle$

VARIABLE $transitions$

$vars \triangleq \langle globalVars, messageVars, clientVars, replicaVars, transitions \rangle$

Helpers

RECURSIVE $SeqFromSet(\_)$

$$SeqFromSet(S) \triangleq$$
$$\text{IF } S = \{\} \text{ THEN } \langle\rangle$$
$$\text{ELSE LET } x \triangleq \text{CHOOSE } x \in S : \text{TRUE}$$
$$\text{IN } \langle x \rangle \circ SeqFromSet(S \setminus \{x\})$$

$Max(s) \triangleq \text{CHOOSE } x \in s : \forall\, y \in s : x \geq y$

$IsQuorum(s) \triangleq Cardinality(s) * 2 \geq Cardinality(Replicas)$

$Quorums \triangleq \{r \in \text{SUBSET } Replicas : IsQuorum(r)\}$

$Primary(v) \triangleq replicas[(v\%Len(replicas)) + (\text{IF } v \geq Len(replicas) \text{ THEN } 1 \text{ ELSE } 0)]$

$IsPrimary(r) \triangleq Primary(viewID[r]) = r$

Messaging helpers

$Sends(ms) \triangleq messages' = messages \cup ms$

$Send(m) \triangleq Sends(\{m\})$

$Replies(reqs, resps) \triangleq messages' = (messages \cup resps) \setminus reqs$

$Reply(req,\ resp) \triangleq Replies(\{req\},\ \{resp\})$

$Discard(m) \triangleq messages' = messages \setminus \{m\}$

$AdvanceTime(c) \triangleq$
$\land\ globalTime' = globalTime + 1$
$\land$ IF $time[c] < globalTime \land globalTime - time[c] > MaxClockInterval$ THEN
$time' = [time$ EXCEPT $![c] = globalTime' - MaxClockInterval]$
ELSE
$time' = [time$ EXCEPT $![c] = time[c] + 1]$

$CurrentTime(c) \triangleq time'[c]$

$Write(c) \triangleq$
$\land\ AdvanceTime(c)$
$\land\ requestID' = [requestID$ EXCEPT $![c] = requestID[c] + 1]$
$\land\ Sends(\{[src \mapsto c,$
$dest \mapsto r,$
$type \mapsto WriteRequest,$
$requestID \mapsto requestID'[c],$
$timestamp \mapsto CurrentTime(c)] : r \in Replicas\})$
$\land$ UNCHANGED $\langle globalVars,\ replicaVars,\ responses,\ writes,\ reads\rangle$

$Read(c) \triangleq$
$\land\ requestID' = [requestID$ EXCEPT $![c] = requestID[c] + 1]$
$\land\ Sends(\{[src \mapsto c,$
$dest \mapsto r,$
$type \mapsto ReadRequest,$
$requestID \mapsto requestID'[c]] : r \in Replicas\})$
$\land$ UNCHANGED $\langle globalVars,\ replicaVars,\ globalTime,\ time,\ responses,\ writes,\ reads\rangle$

$ChecksumsMatch(c1,\ c2) \triangleq$
$\land\ Len(c1) = Len(c2)$
$\land\ \neg\exists\, i \in$ DOMAIN $c1 : c1[i] \neq c2[i]$

$IsCommitted(acks) \triangleq$
$\exists\, msgs \in$ SUBSET $acks :$
$\land\ \{m.src : m \in msgs\} \in Quorums$
$\land\ \exists\, m1 \in msgs : \forall\, m2 \in msgs : m1.viewID = m2.viewID \land ChecksumsMatch(m1.checksum,\ m2.checksu$
$\land\ \exists\, m \in msgs : m.primary$

$HandleWriteResponse(c,\ r,\ m) \triangleq$
$\land\ \neg\exists\, w \in writes[c] : w.requestID = m.requestID$
$\land\ \lor\ \land\ m.requestID \notin$ DOMAIN $responses[c][r]$
$\land\ responses' = [responses$ EXCEPT $![c] = [responses[c]$ EXCEPT $![r] = responses[c][r]$ @@ $(m.requestI$
$\land$ UNCHANGED $\langle writes\rangle$

```
        ∨ ∧ m.requestID ∈ DOMAIN responses[c][r]
          Do not overwrite a response from a newer view
          ∧ responses[c][r][m.requestID].viewID ≤ m.viewID
          ∧ responses' = [responses EXCEPT ![c] = [responses[c] EXCEPT ![r]  = [responses[c][r] EXCEPT ![m.
          ∧ LET committed ≜ IsCommitted({responses'[c][x][m.requestID] : x ∈ {x ∈ Replicas : m.requestID
            IN
              ∨ ∧ committed
                ∧ writes' = [writes EXCEPT ![c] = writes[c] ∪ {m}]
              ∨ ∧ ¬committed
                ∧ UNCHANGED ⟨writes⟩
    ∧ Discard(m)
    ∧ UNCHANGED ⟨globalVars, replicaVars, globalTime, time, requestID, reads⟩

HandleReadResponse(c, r, m) ≜
    ∧ ∨ ∧ m.primary
        ∧ m ∉ reads[c]
        ∧ reads' = [reads EXCEPT ![c] = reads[c] ∪ {m}]
      ∨ ∧ ¬m.primary
        ∧ UNCHANGED ⟨reads⟩
    ∧ Discard(m)
    ∧ UNCHANGED ⟨globalVars, replicaVars, globalTime, time, requestID, responses, writes⟩
```

Server request/response handling

```
HandleWriteRequest(r, c, m) ≜
    ∧ status[r] = NormalStatus
    ∧ ∨ ∧ ∨ Len(log[r]) = 0
          ∨ ∧ Len(log[r])    ≠ 0
            ∧ m.timestamp > log[r][Len(log[r])].timestamp
        ∧ LET checksum ≜ Append([i ∈ DOMAIN log[r] ↦ log[r][i].timestamp], m.timestamp)
              entry    ≜ [client    ↦ c,
                          requestID ↦ m.requestID,
                          timestamp ↦ m.timestamp,
                          checksum  ↦ checksum]
          IN
            ∧ log' = [log EXCEPT ![r] = Append(log[r], entry)]
            ∧ Reply(m, [src       ↦ r,
                        dest      ↦ c,
                        type      ↦ WriteResponse,
                        requestID ↦ m.requestID,
                        viewID    ↦ viewID[r],
                        primary   ↦ IsPrimary(r),
                        index     ↦ Len(log'[r]),
                        checksum  ↦ log'[r][Len(log'[r])].checksum,
```

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad\qquad\qquad\quad succeeded \mapsto \text{TRUE}]) \\
\quad \vee \wedge Len(log[r]) \quad \neq 0 \\
\qquad \wedge m.timestamp \leq log[r][Len(log[r])].timestamp \\
\qquad \wedge Reply(m, [src \quad\quad \mapsto r, \\
\qquad\qquad\qquad\quad dest \quad\quad \mapsto c, \\
\qquad\qquad\qquad\quad type \quad\quad \mapsto WriteResponse, \\
\qquad\qquad\qquad\quad requestID \mapsto m.requestID, \\
\qquad\qquad\qquad\quad viewID \quad \mapsto viewID[r], \\
\qquad\qquad\qquad\quad primary \quad \mapsto IsPrimary(r), \\
\qquad\qquad\qquad\quad index \quad\quad \mapsto Len(log[r]), \\
\qquad\qquad\qquad\quad checksum \mapsto log[r][Len(log[r])].checksum, \\
\qquad\qquad\qquad\quad succeeded \mapsto \text{FALSE}]) \\
\qquad \wedge \text{UNCHANGED } \langle log \rangle \\
\quad \wedge \text{UNCHANGED } \langle globalVars,\ clientVars,\ status,\ viewID,\ lastNormalView,\ viewChangeResponses \rangle
\end{array}
$$

$$
\begin{array}{l}
HandleReadRequest(r,\ c,\ m) \ \triangleq \\
\quad \wedge status[r] = NormalStatus \\
\quad \wedge Len(log[r]) > 0 \\
\quad \wedge Reply(m, [src \quad\quad \mapsto r, \\
\qquad\qquad\quad dest \quad\quad \mapsto c, \\
\qquad\qquad\quad type \quad\quad \mapsto ReadResponse, \\
\qquad\qquad\quad requestID \mapsto m.requestID, \\
\qquad\qquad\quad viewID \quad \mapsto viewID[r], \\
\qquad\qquad\quad primary \quad \mapsto IsPrimary(r), \\
\qquad\qquad\quad index \quad\quad \mapsto Len(log[r]), \\
\qquad\qquad\quad checksum \mapsto log[r][Len(log[r])].checksum, \\
\qquad\qquad\quad succeeded \mapsto \text{TRUE}]) \\
\quad \wedge \text{UNCHANGED } \langle globalVars,\ clientVars,\ status,\ log,\ viewID,\ lastNormalView,\ viewChangeResponses \rangle
\end{array}
$$

$$
\begin{array}{l}
ChangeView(r) \ \triangleq \\
\quad \text{LET } nextViewID \ \triangleq \ viewID[r] + 1 \\
\quad \text{IN} \\
\qquad \wedge Primary(nextViewID) = r \\
\qquad \wedge status' \ = [status \text{ EXCEPT } ![r] \ = ViewChangeStatus] \\
\qquad \wedge viewID' = [viewID \text{ EXCEPT } ![r] = nextViewID] \\
\qquad \wedge viewChangeResponses' = [viewChangeResponses \text{ EXCEPT } ![r] = \{\}] \\
\qquad \wedge Sends(\{[src \quad\quad \mapsto r, \\
\qquad\qquad\quad dest \quad\quad \mapsto d, \\
\qquad\qquad\quad type \quad\quad \mapsto ViewChangeRequest, \\
\qquad\qquad\quad viewID \mapsto nextViewID] : d \in Replicas\}) \\
\qquad \wedge \text{UNCHANGED } \langle globalVars,\ clientVars,\ log,\ lastNormalView \rangle
\end{array}
$$

$$
\begin{array}{l}
HandleViewChangeRequest(r,\ s,\ m) \ \triangleq \\
\quad \wedge viewID[r] \neq m.viewID \\
\quad \wedge viewID' = [viewID \text{ EXCEPT } ![r] = m.viewID] \\
\quad \wedge status' \ = [status \text{ EXCEPT } ![r] \ = ViewChangeStatus]
\end{array}
$$

$\land\ viewChangeResponses' = [viewChangeResponses \text{ EXCEPT } ![r] = \{\}]$
$\land\ Reply(m, [src \mapsto r,$
$\quad dest \mapsto s,$
$\quad type \mapsto ViewChangeResponse,$
$\quad viewID \mapsto m.viewID,$
$\quad lastNormal \mapsto lastNormalView[r],$
$\quad log \mapsto log[r]])$
$\land\ \text{UNCHANGED } \langle globalVars, clientVars, log, lastNormalView\rangle$

$HandleViewChangeResponse(r, s, m) \triangleq$
$\land\ viewID[r] = m.viewID$
$\land\ status[r] = ViewChangeStatus$
$\land\ IsPrimary(r)$
$\land\ viewChangeResponses' = [viewChangeResponses \text{ EXCEPT } ![r] = viewChangeResponses[r] \cup \{m\}]$
$\land\ \text{LET}$
$\quad isViewQuorum(vs) \triangleq IsQuorum(vs) \land \exists\, v \in vs : v.src = r$
$\quad viewChanges \triangleq \{n \in viewChangeResponses[r] : n.type = ViewChangeResponse \land n.viewID = viewI$
$\quad normalViews \triangleq \{n.lastNormal : n \in viewChanges\}$
$\quad lastNormal \triangleq \{\text{CHOOSE } v \in normalViews : \forall\, v2 \in normalViews : v2 < v\}$
$\quad goodLogs \triangleq \{n.log : n \in \{o \in viewChanges : o.lastNormal = lastNormal\}\}$
$\quad combineLogs(ls) \triangleq$
$\quad\quad \text{LET}$
$\quad\quad\quad logsWith(i) \triangleq \{l \in ls : Len(l) \geq i\}$
$\quad\quad\quad entries(i) \triangleq \{l[i] : l \in logsWith(i)\}$
$\quad\quad\quad quorums(i) \triangleq \{l \in \text{SUBSET } logsWith(i) : IsQuorum(l)\}$
$\quad\quad\quad checksums(l, i) \triangleq \{e.checksum : e \in l[i]\}$
$\quad\quad\quad isCommitted(i) \triangleq \exists\, e \in entries(i) : \forall\, l \in quorums(i) : e.checksum \in checksums(l, i)$
$\quad\quad\quad committed(i) \triangleq \text{CHOOSE } e \in entries(i) : \forall\, l \in quorums(i) : e.checksum \in checksums(l, i)$
$\quad\quad\quad maxIndex \triangleq Max(\{Len(l) : l \in ls\})$
$\quad\quad\quad maxCommitted \triangleq Max(\{i \in 1 \,..\, maxIndex : isCommitted(i)\})$
$\quad\quad \text{IN}$
$\quad\quad\quad [i \in 1 \,..\, maxCommitted \mapsto committed(i)]$
$\quad \text{IN}$
$\quad\quad \lor\ \land\ isViewQuorum(viewChanges)$
$\quad\quad\quad\ \land\ Replies(m, \{[src \mapsto r,$
$\quad\quad\quad\quad\quad\quad dest \mapsto d,$
$\quad\quad\quad\quad\quad\quad type \mapsto StartViewRequest,$
$\quad\quad\quad\quad\quad\quad viewID \mapsto viewID[r],$
$\quad\quad\quad\quad\quad\quad log \mapsto combineLogs(goodLogs)] : d \in Replicas\})$
$\quad\quad \lor\ \land\ \neg isViewQuorum(viewChanges)$
$\quad\quad\quad\ \land\ Discard(m)$
$\land\ \text{UNCHANGED } \langle globalVars, clientVars, status, viewID, log, lastNormalView\rangle$

$HandleStartViewRequest(r, s, m) \triangleq$
$\land\ \lor\ viewID[r] < m.viewID$

$\lor \land viewID[r] = m.viewID$
$\land status[r] = ViewChangeStatus$
$\land log' = [log \text{ EXCEPT } ![r] = m.log]$
$\land status' = [status \text{ EXCEPT } ![r] = NormalStatus]$
$\land viewID' = [viewID \text{ EXCEPT } ![r] = m.viewID]$
$\land lastNormalView' = [lastNormalView \text{ EXCEPT } ![r] = m.viewID]$
$\land Discard(m)$
$\land \text{UNCHANGED } \langle globalVars, clientVars, viewChangeResponses \rangle$

$InitMessageVars \triangleq$
$\land messages = \{\}$

$InitClientVars \triangleq$
$\land globalTime = 0$
$\land time = [c \in Clients \mapsto 0]$
$\land requestID = [c \in Clients \mapsto 0]$
$\land responses = [c \in Clients \mapsto [r \in Replicas \mapsto [s \in \{\} \mapsto [index \mapsto 0, checksum \mapsto Nil]]]]$
$\land writes = [c \in Clients \mapsto \{\}]$
$\land reads = [c \in Clients \mapsto \{\}]$

$InitReplicaVars \triangleq$
$\land replicas = SeqFromSet(Replicas)$
$\land status = [r \in Replicas \mapsto NormalStatus]$
$\land log = [r \in Replicas \mapsto \langle\rangle]$
$\land viewID = [r \in Replicas \mapsto 1]$
$\land lastNormalView = [r \in Replicas \mapsto 1]$
$\land viewChangeResponses = [r \in Replicas \mapsto \{\}]$

$Init \triangleq$
$\land InitMessageVars$
$\land InitClientVars$
$\land InitReplicaVars$
$\land transitions = 0$

The type invariant checks that no read ever reads a different value than a previous write
$Inv \triangleq$
$\land \forall c1, c2 \in Clients :$
$\neg\exists r \in reads[c1] :$
$\exists w \in writes[c2] :$
$\land r.index = w.index$
$\land \neg ChecksumsMatch(r.checksum, w.checksum)$
$\land \forall c1, c2 \in Clients :$

$$
\begin{array}{l}
\quad \neg \exists\, r1 \in reads[c1] : \\
\quad\quad \exists\, r2 \in reads[c2] : \\
\quad\quad\quad \land\, r1.index = r2.index \\
\quad\quad\quad \land\, \neg ChecksumsMatch(r1.checksum,\, r2.checksum)
\end{array}
$$

$$Transition \triangleq transitions' = transitions + 1$$

$$
\begin{array}{l}
Next \triangleq \\
\quad \lor \exists\, c \in Clients : \\
\quad\quad \land\, Write(c) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, c \in Clients : \\
\quad\quad \land\, Read(c) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, r \in Replicas : \\
\quad\quad \land\, ChangeView(r) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = WriteRequest \\
\quad\quad \land\, HandleWriteRequest(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = WriteResponse \\
\quad\quad \land\, HandleWriteResponse(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = ReadRequest \\
\quad\quad \land\, HandleReadRequest(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = ReadResponse \\
\quad\quad \land\, HandleReadResponse(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = ViewChangeRequest \\
\quad\quad \land\, HandleViewChangeRequest(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = ViewChangeResponse \\
\quad\quad \land\, HandleViewChangeResponse(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition \\
\quad \lor \exists\, m \in messages : \\
\quad\quad \land\, m.type = StartViewRequest \\
\quad\quad \land\, HandleStartViewRequest(m.dest,\, m.src,\, m) \\
\quad\quad \land\, Transition
\end{array}
$$

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

\ * Modification History
\ * Last modified *Mon Sep* 21 15:12:31 *PDT* 2020 by *jordanhalterman*
\ * Created *Fri Sep* 18 22:45:21 *PDT* 2020 by *jordanhalterman*