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 *time*

VARIABLE *requestID*

VARIABLE *responses*

VARIABLE $writes$

VARIABLE $reads$

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

VARIABLE $status$

VARIABLE log

VARIABLE $viewID$

VARIABLE $lastNormalView$

VARIABLE $viewChanges$

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

VARIABLE $transitions$

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

Helpers

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

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

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

$Quorums \triangleq \{r \in$ 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

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

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

$Replies(req, resps) \triangleq messages' = (messages \cup resps) \setminus \{req\}$

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

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

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

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

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

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

$HandleWriteResponse(c, r, m) \triangleq$
$\land \neg\exists\, w \in writes[c] : w.requestID = m.requestID$
$\land \lor \land m.requestID \notin \text{DOMAIN } responses[c][r]$
$\land responses' = [responses \text{ EXCEPT } ![c] = [responses[c] \text{ EXCEPT } ![r] = responses[c][r] @@ (m.requestI$
$\land \text{UNCHANGED } \langle writes\rangle$
$\lor \land m.requestID \in \text{DOMAIN } responses[c][r]$
Do not overwrite a response from a newer view
$\land responses[c][r][m.requestID].viewID \leq m.viewID$
$\land responses' = [responses \text{ EXCEPT } ![c] = [responses[c] \text{ EXCEPT } ![r] = [responses[c][r] \text{ EXCEPT } ![m.$
$\land \text{LET } committed \triangleq IsCommitted(\{responses'[c][x][m.requestID] : x \in \{x \in Replicas : m.requestID$
IN
$\lor \land committed$
$\land writes' = [writes \text{ EXCEPT } ![c] = writes[c] \cup \{m\}]$
$\lor \land \neg committed$
$\land \text{UNCHANGED } \langle writes\rangle$

$\land\ Discard(m)$
$\land$ UNCHANGED $\langle globalVars,\ replicaVars,\ time,\ requestID,\ reads\rangle$

$HandleReadResponse(c,\ r,\ m)\ \triangleq$
$\land\ \lor\ \land\ m.primary$
$\land\ m \notin reads[c]$
$\land\ reads' = [reads$ EXCEPT $![c] = reads[c] \cup \{m\}]$
$\lor\ \land\ \neg m.primary$
$\land$ UNCHANGED $\langle reads\rangle$
$\land\ Discard(m)$
$\land$ UNCHANGED $\langle globalVars,\ replicaVars,\ time,\ requestID,\ responses,\ writes\rangle$

Server request/response handling

$HandleWriteRequest(r,\ c,\ m)\ \triangleq$
$\land\ status[r] = NormalStatus$
$\land\ \lor\ \land\ \lor\ Len(log[r]) = 0$
$\lor\ \land\ Len(log[r]) \neq 0$
$\land\ m.timestamp > log[r][Len(log[r])].timestamp$
$\land$ LET $checksum \triangleq Append([i \in$ DOMAIN $log[r] \mapsto log[r][i].timestamp],\ m.timestamp)$
$entry \triangleq [client \mapsto c,$
$requestID \mapsto m.requestID,$
$timestamp \mapsto m.timestamp,$
$checksum \mapsto checksum]$
IN
$\land\ log' = [log$ EXCEPT $![r] = Append(log[r],\ entry)]$
$\land\ Reply(m,\ [src \mapsto r,$
$dest \mapsto c,$
$type \mapsto WriteResponse,$
$requestID \mapsto m.requestID,$
$viewID \mapsto viewID[r],$
$primary \mapsto IsPrimary(r),$
$index \mapsto Len(log'[r]),$
$checksum \mapsto log'[r][Len(log'[r])].checksum,$
$succeeded \mapsto$ TRUE$])$
$\lor\ \land\ Len(log[r]) \neq 0$
$\land\ m.timestamp \leq log[r][Len(log[r])].timestamp$
$\land\ Reply(m,\ [src \mapsto r,$
$dest \mapsto c,$
$type \mapsto WriteResponse,$
$requestID \mapsto m.requestID,$
$viewID \mapsto viewID[r],$
$primary \mapsto IsPrimary(r),$
$index \mapsto Len(log[r]),$

$$
\begin{array}{l}
\qquad\qquad\qquad\quad checksum \mapsto log[r][Len(log[r])].checksum, \\
\qquad\qquad\qquad\quad succeeded \mapsto \text{FALSE}]) \\
\qquad\quad \wedge \text{UNCHANGED } \langle log \rangle \\
\quad \wedge \text{UNCHANGED } \langle globalVars,\ clientVars,\ status,\ viewID,\ lastNormalView,\ viewChanges \rangle
\end{array}
$$

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

$$
\begin{array}{l}
ChangeView(r) \triangleq \\
\quad \wedge Sends(\{[src \mapsto r, \\
\qquad\qquad dest \mapsto d, \\
\qquad\qquad type \mapsto ViewChangeRequest, \\
\qquad\qquad viewID \mapsto viewID[r] + 1] : d \in Replicas\}) \\
\quad \wedge \text{UNCHANGED } \langle globalVars,\ clientVars,\ replicaVars \rangle
\end{array}
$$

$$
\begin{array}{l}
HandleViewChangeRequest(r,\ s,\ m) \triangleq \\
\quad \wedge viewID[r] < m.viewID \\
\quad \wedge viewID' = [viewID \text{ EXCEPT } ![r] = m.viewID] \\
\quad \wedge status' = [status \text{ EXCEPT } ![r] = ViewChangeStatus] \\
\quad \wedge viewChanges' = [viewChanges \text{ EXCEPT } ![r] = \{\}] \\
\quad \wedge Reply(m, [src \mapsto r, \\
\qquad\qquad dest \mapsto Primary(m.viewID), \\
\qquad\qquad type \mapsto ViewChangeResponse, \\
\qquad\qquad viewID \mapsto m.viewID, \\
\qquad\qquad lastNormal \mapsto lastNormalView[r], \\
\qquad\qquad log \mapsto log[r]]) \\
\quad \wedge \text{UNCHANGED } \langle globalVars,\ clientVars,\ log,\ lastNormalView \rangle
\end{array}
$$

$$
\begin{array}{l}
HandleViewChangeResponse(r,\ s,\ m) \triangleq \\
\quad \wedge IsPrimary(r) \\
\quad \wedge viewID[r] = m.viewID \\
\quad \wedge status[r] = ViewChangeStatus \\
\quad \wedge viewChanges' = [viewChanges \text{ EXCEPT } ![r] = viewChanges[r] \cup \{m\}] \\
\quad \wedge \text{LET} \\
\qquad isViewQuorum(vs) \triangleq IsQuorum(vs) \wedge \exists\, v \in vs : v.src = r
\end{array}
$$

```
        newViewChanges  ≜ {v ∈ viewChanges'[r] : v.viewID = viewID[r]}
        normalViews     ≜ {v.lastNormal : v ∈ newViewChanges}
        lastNormal      ≜ CHOOSE v ∈ normalViews : ∀ v2 ∈ normalViews : v2 ≤ v
        goodLogs        ≜ {n.log : n ∈ {v ∈ newViewChanges : v.lastNormal = lastNormal}}
        combineLogs(ls) ≜
          LET
            indexLogs(i)              ≜ {l ∈ ls : Len(l) ≥ i}
            indexEntries(i)           ≜ {l[i] : l ∈ indexLogs(i)}
            quorumLogs(i)             ≜ {L ∈ SUBSET indexLogs(i) : IsQuorum(L)}
            isCommittedEntry(i, e)    ≜ ∀ L ∈ quorumLogs(i) :
                                          ∃ l ∈ L :
                                            ChecksumsMatch(e.checksum, l[i].checksum)
            isCommittedIndex(i)       ≜ ∃ e ∈ indexEntries(i) : isCommittedEntry(i, e)
            commit(i)                 ≜ CHOOSE e ∈ indexEntries(i) : isCommittedEntry(i, e)
            maxIndex                  ≜ Max({Len(l) : l ∈ ls})
            committedIndexes          ≜ {i ∈ 1 .. maxIndex : isCommittedIndex(i)}
            maxCommit                 ≜ IF Cardinality(committedIndexes) > 0 THEN Max(committedIndexe
          IN
            [i ∈ 1 .. maxCommit ↦ commit(i)]
      IN
        ∨ ∧ isViewQuorum(newViewChanges)
          ∧ Replies(m, {[src  ↦ r,
                         dest ↦ d,
                         type ↦ StartViewRequest,
                         viewID ↦ viewID[r],
                         log  ↦ combineLogs(goodLogs)] : d ∈ Replicas})
        ∨ ∧ ¬isViewQuorum(newViewChanges)
          ∧ Discard(m)
      ∧ UNCHANGED ⟨globalVars, clientVars, status, viewID, log, lastNormalView⟩

HandleStartViewRequest(r, s, m) ≜
    ∧ ∨ viewID[r] < m.viewID
      ∨ ∧ viewID[r]    = m.viewID
        ∧ status[r]    = ViewChangeStatus
    ∧ log'             = [log EXCEPT ![r] = m.log]
    ∧ status'          = [status EXCEPT ![r]  = NormalStatus]
    ∧ viewID'          = [viewID EXCEPT ![r] = m.viewID]
    ∧ lastNormalView'  = [lastNormalView EXCEPT ![r] = m.viewID]
    ∧ Discard(m)
    ∧ UNCHANGED ⟨globalVars, clientVars, viewChanges⟩
```

```
InitMessageVars ≜
    ∧ messages = {}
```

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

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

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

The type invariant checks that no read ever reads a different value than a previous write

$Inv \triangleq$
$\land \forall\, c1,\ c2 \in Clients :$
$\quad \neg\exists\, r \in reads[c1] :$
$\quad\quad \exists\, w \in writes[c2] :$
$\quad\quad\quad \land\ r.index = w.index$
$\quad\quad\quad \land \neg ChecksumsMatch(r.checksum,\ w.checksum)$
$\land \forall\, c1,\ c2 \in Clients :$
$\quad \neg\exists\, r1 \in reads[c1] :$
$\quad\quad \exists\, r2 \in reads[c2] :$
$\quad\quad\quad \land\ r1.index = r2.index$
$\quad\quad\quad \land \neg ChecksumsMatch(r1.checksum,\ r2.checksum)$

$Transition \triangleq transitions' = transitions + 1$

$Next \triangleq$
$\lor \exists\, c \in Clients :$
$\quad \land\ Write(c)$
$\quad \land\ Transition$
$\lor \exists\, c \in Clients :$
$\quad \land\ Read(c)$
$\quad \land\ Transition$
$\lor \exists\, r \in Replicas :$
$\quad \land\ ChangeView(r)$

$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = WriteRequest$
$\land$ $HandleWriteRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = WriteResponse$
$\land$ $HandleWriteResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = ReadRequest$
$\land$ $HandleReadRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = ReadResponse$
$\land$ $HandleReadResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = ViewChangeRequest$
$\land$ $HandleViewChangeRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = ViewChangeResponse$
$\land$ $HandleViewChangeResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = StartViewRequest$
$\land$ $HandleStartViewRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*

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

\ * Modification History
\ * Last modified *Tue Sep* 22 01:06:09 *PDT* 2020 by *jordanhalterman*
\ * Created *Fri Sep* 18 22:45:21 *PDT* 2020 by *jordanhalterman*