```
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 ≜ ⟨replicas⟩

VARIABLE messages

messageVars ≜ ⟨messages⟩

VARIABLE globalTime

VARIABLE time

VARIABLE requestID
```

$\textsc{variable}\ \mathit{responses}$

$\textsc{variable}\ \mathit{writes}$

$\textsc{variable}\ \mathit{reads}$

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

$\textsc{variable}\ \mathit{status}$

$\textsc{variable}\ \mathit{log}$

$\textsc{variable}\ \mathit{viewID}$

$\textsc{variable}\ \mathit{lastNormalView}$

$\textsc{variable}\ \mathit{viewChanges}$

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

$\textsc{variable}\ \mathit{transitions}$

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

Helpers

$$\begin{array}{l}
\textsc{recursive}\ \mathit{SeqFromSet}(\_) \\
\mathit{SeqFromSet}(S) \triangleq \\
\quad \textsc{if}\ S = \{\}\ \textsc{then}\ \langle\rangle \\
\quad\ \ \textsc{else}\ \ \textsc{let}\ x \triangleq \textsc{choose}\ x \in S : \textsc{true} \\
\qquad\qquad\ \ \textsc{in}\ \ \ \langle x \rangle \circ \mathit{SeqFromSet}(S \setminus \{x\})
\end{array}$$

$\mathit{Max}(s) \triangleq \textsc{choose}\ x \in s : \forall\, y \in s : x \geq y$

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

$\mathit{Quorums} \triangleq \{r \in \textsc{subset}\ \mathit{Replicas} : \mathit{IsQuorum}(r)\}$

$\mathit{Primary}(v) \triangleq \mathit{replicas}[(v \% \mathit{Len}(\mathit{replicas})) + (\textsc{if}\ v \geq \mathit{Len}(\mathit{replicas})\ \textsc{then}\ 1\ \textsc{else}\ 0)]$

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

Messaging helpers

$\mathit{Sends}(\mathit{ms}) \triangleq \mathit{messages}' = \mathit{messages} \cup \mathit{ms}$

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

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

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

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

$$\begin{array}{l}
AdvanceTime(c) \triangleq \\
\quad \land\ globalTime' = globalTime + 1 \\
\quad \land\ \textsc{if}\ time[c] < globalTime \land globalTime - time[c] > MaxClockInterval\ \textsc{then} \\
\qquad\quad time' = [time\ \textsc{except}\ ![c] = globalTime' - MaxClockInterval] \\
\qquad \textsc{else} \\
\qquad\quad time' = [time\ \textsc{except}\ ![c] = time[c] + 1]
\end{array}$$

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

$$\begin{array}{l}
Write(c) \triangleq \\
\quad \land\ AdvanceTime(c) \\
\quad \land\ requestID' = [requestID\ \textsc{except}\ ![c] = requestID[c] + 1] \\
\quad \land\ Sends(\{[src \mapsto c, \\
\qquad\qquad\quad dest \mapsto r, \\
\qquad\qquad\quad type \mapsto WriteRequest, \\
\qquad\qquad\quad requestID \mapsto requestID'[c], \\
\qquad\qquad\quad timestamp \mapsto CurrentTime(c)] : r \in Replicas\}) \\
\quad \land\ \textsc{unchanged}\ \langle globalVars,\ replicaVars,\ responses,\ writes,\ reads \rangle
\end{array}$$

$$\begin{array}{l}
Read(c) \triangleq \\
\quad \land\ requestID' = [requestID\ \textsc{except}\ ![c] = requestID[c] + 1] \\
\quad \land\ Sends(\{[src \mapsto c, \\
\qquad\qquad\quad dest \mapsto r, \\
\qquad\qquad\quad type \mapsto ReadRequest, \\
\qquad\qquad\quad requestID \mapsto requestID'[c]] : r \in Replicas\}) \\
\quad \land\ \textsc{unchanged}\ \langle globalVars,\ replicaVars,\ globalTime,\ time,\ responses,\ writes,\ reads \rangle
\end{array}$$

$$\begin{array}{l}
ChecksumsMatch(c1,\ c2) \triangleq \\
\quad \land\ Len(c1) = Len(c2) \\
\quad \land\ \neg \exists\, i \in \textsc{domain}\ c1 : c1[i] \neq c2[i]
\end{array}$$

$$\begin{array}{l}
IsCommitted(acks) \triangleq \\
\quad \exists\, msgs \in \textsc{subset}\ acks : \\
\qquad \land\ \{m.src : m \in msgs\} \in Quorums \\
\qquad \land\ \exists\, m1 \in msgs : \forall\, m2 \in msgs : m1.viewID = m2.viewID \land ChecksumsMatch(m1.checksum,\ m2.checksu \\
\qquad \land\ \exists\, m \in msgs : m.primary
\end{array}$$

$$\begin{array}{l}
HandleWriteResponse(c,\ r,\ m) \triangleq \\
\quad \land\ \neg \exists\, w \in writes[c] : w.requestID = m.requestID \\
\quad \land\ \lor\ \land\ m.requestID \notin \textsc{domain}\ responses[c][r] \\
\qquad\quad \land\ responses' = [responses\ \textsc{except}\ ![c] = [responses[c]\ \textsc{except}\ ![r] = responses[c][r]\ @@\ (m.requestI \\
\qquad\quad \land\ \textsc{unchanged}\ \langle writes \rangle
\end{array}$$

```
      ∨ ∧ 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,
                          succeeded ↦ TRUE])
        ∨ ∧ Len(log[r])    ≠ 0
          ∧ m.timestamp ≤ log[r][Len(log[r])].timestamp
          ∧ 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,
                      succeeded ↦ FALSE])
          ∧ UNCHANGED ⟨log⟩
    ∧ UNCHANGED ⟨globalVars, clientVars, status, viewID, lastNormalView, viewChanges⟩

HandleReadRequest(r, c, m) ≜
    ∧ status[r] = NormalStatus
    ∧ Len(log[r]) > 0
    ∧ Reply(m, [src       ↦ r,
                dest      ↦ c,
                type      ↦ ReadResponse,
                requestID ↦ m.requestID,
                viewID    ↦ viewID[r],
                primary   ↦ IsPrimary(r),
                index     ↦ Len(log[r]),
                checksum  ↦ log[r][Len(log[r])].checksum,
                succeeded ↦ TRUE])
    ∧ UNCHANGED ⟨globalVars, clientVars, status, log, viewID, lastNormalView, viewChanges⟩

ChangeView(r) ≜
    ∧ Sends({[src    ↦ r,
              dest   ↦ d,
              type   ↦ ViewChangeRequest,
              viewID ↦ viewID[r] + 1] : d ∈ Replicas})
    ∧ UNCHANGED ⟨globalVars, clientVars, replicaVars⟩

HandleViewChangeRequest(r, s, m) ≜
    ∧ viewID[r] < m.viewID
    ∧ viewID'      = [viewID EXCEPT ![r] = m.viewID]
    ∧ status'      = [status EXCEPT ![r]  = ViewChangeStatus]
    ∧ viewChanges' = [viewChanges EXCEPT ![r] = {}]
    ∧ Reply(m, [src        ↦ r,
                dest       ↦ Primary(m.viewID),
                type       ↦ ViewChangeResponse,
                viewID     ↦ m.viewID,
                lastNormal ↦ lastNormalView[r],
                    $log \qquad\qquad \mapsto log[r]])$
        $\land$ UNCHANGED $\langle globalVars, clientVars, log, lastNormalView \rangle$

$HandleViewChangeResponse(r, s, m) \triangleq$
        $\land IsPrimary(r)$
        $\land viewID[r] \quad = m.viewID$
        $\land status[r] \quad = ViewChangeStatus$
        $\land viewChanges' = [viewChanges$ EXCEPT $![r] = viewChanges[r] \cup \{m\}]$
        $\land$ LET
                $isViewQuorum(vs) \triangleq IsQuorum(vs) \land \exists v \in vs : v.src = r$
                $newViewChanges \quad \triangleq \{v \in viewChanges'[r] : v.viewID = viewID[r]\}$
                $normalViews \qquad \triangleq \{v.lastNormal : v \in newViewChanges\}$
                $lastNormal \qquad\; \triangleq$ CHOOSE $v \in normalViews : \forall v2 \in normalViews : v2 \leq v$
                $goodLogs \qquad\quad \triangleq \{n.log : n \in \{v \in newViewChanges : v.lastNormal = lastNormal\}\}$
                $combineLogs(ls) \quad \triangleq$
                    LET
                        $indexLogs(i) \qquad\qquad \triangleq \{l \in ls : Len(l) \geq i\}$
                        $indexEntries(i) \qquad\quad \triangleq \{l[i] : l \in indexLogs(i)\}$
                        $quorumLogs(i) \qquad\quad \triangleq \{L \in$ SUBSET $indexLogs(i) : IsQuorum(L)\}$
                        $isCommittedEntry(i, e) \triangleq \forall L \in quorumLogs(i) :$
                                                        $\exists l \in L :$
                                                            $ChecksumsMatch(e.checksum, l[i].checksum)$
                        $isCommittedIndex(i) \quad \triangleq \exists e \in indexEntries(i) : isCommittedEntry(i, e)$
                        $commit(i) \qquad\qquad\quad \triangleq$ CHOOSE $e \in indexEntries(i) : isCommittedEntry(i, e)$
                        $maxIndex \qquad\qquad\quad \triangleq Max(\{Len(l) : l \in ls\})$
                        $committedIndexes \qquad \triangleq \{i \in 1 \,..\, maxIndex : isCommittedIndex(i)\}$
                        $maxCommit \qquad\qquad \triangleq$ IF $Cardinality(committedIndexes) > 0$ THEN $Max(committedIndexe$
                    IN
                        $[i \in 1 \,..\, maxCommit \mapsto commit(i)]$
            IN
                $\lor \land isViewQuorum(newViewChanges)$
                    $\land Replies(m, \{[src \quad\;\; \mapsto r,$
                                        $dest \quad\; \mapsto d,$
                                        $type \quad\; \mapsto StartViewRequest,$
                                        $viewID \mapsto viewID[r],$
                                        $log \qquad \mapsto combineLogs(goodLogs)] : d \in Replicas\})$
                $\lor \land \neg isViewQuorum(newViewChanges)$
                    $\land Discard(m)$
        $\land$ UNCHANGED $\langle globalVars, clientVars, status, viewID, log, lastNormalView \rangle$

$HandleStartViewRequest(r, s, m) \triangleq$
        $\land \lor viewID[r] < m.viewID$
            $\lor \land viewID[r] \quad = m.viewID$
                $\land status[r] \quad = ViewChangeStatus$
        $\land log' \qquad\qquad\;\; = [log$ EXCEPT $![r] = m.log]$
```

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

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

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

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

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

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

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

$Transition \triangleq transitions' = transitions + 1$

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

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

\ * Modification History
\ * Last modified _Mon Sep_ 21 22:04:34 _PDT_ 2020 by _jordanhalterman_
\ * Created _Fri Sep_ 18 22:45:21 _PDT_ 2020 by _jordanhalterman_