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

	SlotLookup
	SlotLookupRep
	GapCommit
	GapCommitRep
ViewChangeRequest,	*ViewChangeReq*
ViewChangeResponse,	*ViewChange*
StartViewRequest,	*StartView*
SyncPrepareRequest,	*SyncPrepare*
SyncPrepareResponse,	*SyncPrepareRep*
SyncCommitRequest	*SyncCommit*

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 $viewID$

VARIABLE log

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

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 $\langle x \rangle \circ SeqFromSet(S \setminus \{x\})$

$Quorums \triangleq \{r \in \text{SUBSET } Replicas : Cardinality(r) * 2 > Cardinality(Replicas)\}$

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

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

Messaging helpers

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

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

$Reply(req,\ res) \triangleq messages' = messages \setminus \{req,\ res\}$

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

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

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

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

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

```
            IN
                ∨ ∧ committed
                  ∧ writes' = [writes EXCEPT ![c] = writes[c] ∪ {m}]
                ∨ ∧ ¬committed
                  ∧ UNCHANGED ⟨writes⟩
    ∧ Discard(m)
    ∧ UNCHANGED ⟨globalVars, messageVars, 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, 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
        ∧ log' = [log EXCEPT ![r] = Append(log[r], m)]
        ∧ Reply(m, [source    ↦ r,
                    target    ↦ c,
                    type      ↦ WriteResponse,
                    requestID ↦ m.requestID,
                    viewID    ↦ viewID[r],
                    primary   ↦ IsPrimary(r),
                    index     ↦ Len(log'[r]),
                    checksum  ↦ {log'[r][i].timestamp : i ∈ DOMAIN log'[r]},
                    succeeded ↦ TRUE])
      ∨ ∧ Len(log[r])    ≠ 0
        ∧ m.timestamp ≤ log[r][Len(log[r])].timestamp
        ∧ Reply(m, [source    ↦ r,
                    target    ↦ c,
                    type      ↦ WriteResponse,
                    requestID ↦ m.requestID,
                    viewID    ↦ viewID[r],
                    primary   ↦ IsPrimary(r),
                    index     ↦ Len(log[r]),
                    checksum  ↦ {log[r][i].timestamp : i ∈ DOMAIN log[r]},
```

$succeeded \mapsto \text{FALSE}])$
$\land$ UNCHANGED $\langle log \rangle$
$\land$ UNCHANGED $\langle globalVars,\ clientVars,\ status,\ viewID \rangle$

$HandleReadRequest(r,\ c,\ m) \triangleq$
$\land\ status[r] = NormalStatus$
$\land\ Len(log[r]) > 0$
$\land\ Reply(m, [source \mapsto r,$
$target \mapsto c,$
$type \mapsto WriteResponse,$
$requestID \mapsto m.requestID,$
$viewID \mapsto viewID[r],$
$primary \mapsto IsPrimary(r),$
$index \mapsto Len(log[r]),$
$checksum \mapsto \{log[r][i].timestamp : i \in \text{DOMAIN } log[r]\},$
$succeeded \mapsto \text{TRUE}])$
$\land$ UNCHANGED $\langle globalVars,\ clientVars,\ status,\ viewID,\ log \rangle$

$HandleViewChangeRequest(r,\ s,\ m) \triangleq$
$\land$ FALSE
$\land$ UNCHANGED $\langle globalVars,\ messageVars,\ clientVars,\ replicaVars \rangle$

$HandleViewChangeResponse(r,\ s,\ m) \triangleq$
$\land$ FALSE
$\land$ UNCHANGED $\langle globalVars,\ messageVars,\ clientVars,\ replicaVars \rangle$

$HandleStartViewRequest(r,\ s,\ m) \triangleq$
$\land$ FALSE
$\land$ UNCHANGED $\langle globalVars,\ messageVars,\ clientVars,\ replicaVars \rangle$

$HandleSyncPrepareRequest(r,\ s,\ m) \triangleq$
$\land$ FALSE
$\land$ UNCHANGED $\langle globalVars,\ messageVars,\ clientVars,\ replicaVars \rangle$

$HandleSyncPrepareResponse(r,\ s,\ m) \triangleq$
$\land$ FALSE
$\land$ UNCHANGED $\langle globalVars,\ messageVars,\ clientVars,\ replicaVars \rangle$

$HandleSyncCommitRequest(r,\ s,\ m) \triangleq$
$\land$ FALSE
$\land$ UNCHANGED $\langle globalVars,\ messageVars,\ clientVars,\ replicaVars \rangle$

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

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

$$
\begin{array}{l}
\mathit{InitReplicaVars} \triangleq \\
\quad \land\ \mathit{replicas} = \mathit{SeqFromSet}(\mathit{Replicas}) \\
\quad \land\ \mathit{status}\ \ = [r \in \mathit{Replicas} \mapsto \mathit{NormalStatus}] \\
\quad \land\ \mathit{viewID} = [r \in \mathit{Replicas} \mapsto 1] \\
\quad \land\ \mathit{log} = [r \in \mathit{Replicas} \mapsto \langle\rangle]
\end{array}
$$

$$
\begin{array}{l}
\mathit{Init} \triangleq \\
\quad \land\ \mathit{InitMessageVars} \\
\quad \land\ \mathit{InitClientVars} \\
\quad \land\ \mathit{InitReplicaVars} \\
\quad \land\ \mathit{transitions} = 0
\end{array}
$$

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

$$
\begin{array}{l}
\mathit{Inv} \triangleq\ \forall\, c1,\ c2 \in \mathit{Clients} : \\
\qquad \neg\exists\, r \in \mathit{reads}[c1] : \\
\qquad\quad \exists\, w \in \mathit{writes}[c2] : \\
\qquad\qquad r.\mathit{index} = w.\mathit{index} \land r.\mathit{requestID} \neq w.\mathit{requestID}
\end{array}
$$

$$
\mathit{Transition} \triangleq\ \mathit{transitions}' = \mathit{transitions} + 1
$$

$$
\begin{array}{l}
\mathit{Next} \triangleq \\
\quad \lor\ \exists\, c \in \mathit{Clients} : \\
\qquad \land\ \mathit{Write}(c) \\
\qquad \land\ \mathit{transitions}' = \mathit{transitions} + 1 \\
\quad \lor\ \exists\, c \in \mathit{Clients} : \\
\qquad \land\ \mathit{Read}(c) \\
\qquad \land\ \mathit{Transition} \\
\quad \lor\ \exists\, m \in \mathit{messages} : \\
\qquad \land\ m.\mathit{type} = \mathit{WriteRequest} \\
\qquad \land\ \mathit{HandleWriteRequest}(m.\mathit{target},\ m.\mathit{source},\ m) \\
\qquad \land\ \mathit{Transition} \\
\quad \lor\ \exists\, m \in \mathit{messages} : \\
\qquad \land\ m.\mathit{type} = \mathit{WriteResponse} \\
\qquad \land\ \mathit{HandleWriteResponse}(m.\mathit{target},\ m.\mathit{source},\ m) \\
\qquad \land\ \mathit{Transition} \\
\quad \lor\ \exists\, m \in \mathit{messages} : \\
\qquad \land\ m.\mathit{type} = \mathit{ReadRequest}
\end{array}
$$

$\land$ $HandleReadRequest(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = ReadResponse$
$\land$ $HandleReadResponse(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = ViewChangeRequest$
$\land$ $HandleViewChangeRequest(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = ViewChangeResponse$
$\land$ $HandleViewChangeResponse(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = StartViewRequest$
$\land$ $HandleStartViewRequest(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = SyncPrepareRequest$
$\land$ $HandleSyncPrepareRequest(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = SyncPrepareResponse$
$\land$ $HandleSyncPrepareResponse(m.target,\ m.source,\ m)$
$\land$ $Transition$
$\lor \exists\, m \in messages :$
$\land$ $m.type = SyncCommitRequest$
$\land$ $HandleSyncCommitRequest(m.target,\ m.source,\ m)$
$\land$ $Transition$

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

\ * Modification History
\ * Last modified Sun *Sep* 20 19:08:16 *PDT* 2020 by *jordanhalterman*
\ * Created *Fri Sep* 18 22:45:21 *PDT* 2020 by *jordanhalterman*