MODULE *JustInTimePaxos*

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

The set of *Paxos* replicas
CONSTANT *Replicas*

The set of *Paxos* clients
CONSTANT *Clients*

An empty value
CONSTANT *Nil*

Client request/response types+
CONSTANTS
MWriteRequest,
MWriteResponse,
MReadRequest,
MReadResponse

Server request/response types
CONSTANTS
MRepairRequest,
MRepairResponse,
MAbortRequest,
MAbortResponse,
MViewChangeRequest,
MViewChangeResponse,
MStartViewRequest

Replica roles
CONSTANTS
SNormal,
SAborting,
SViewChange

Entry types
CONSTANTS
TValue,
TNoOp

VARIABLE *replicas*

$globalVars \triangleq \langle replicas \rangle$

VARIABLE *messages*

$messageVars \triangleq \langle messages \rangle$

VARIABLE $cTime$

VARIABLE $cViewID$

VARIABLE $cSeqNum$

VARIABLE $cResps$

VARIABLE $cWrites$

VARIABLE $cReads$

$clientVars \triangleq \langle cTime,\ cViewID,\ cSeqNum,\ cResps,\ cWrites,\ cReads \rangle$

VARIABLE $rStatus$

VARIABLE $rLog$

VARIABLE $rViewID$

VARIABLE $rSeqNum$

VARIABLE $rLastView$

VARIABLE $rViewChanges$

VARIABLE $rAbortSeqNum$

VARIABLE $rAbortResps$

$replicaVars \triangleq \langle rStatus,\ rLog,\ rViewID,\ rSeqNum,\ rLastView,\ rViewChanges,\ rAbortSeqNum,\ rAbortResps \rangle$

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 $Replicas : IsQuorum(r)\}$

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

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

$Replace(l,\, i,\, x) \triangleq [j \in 1 \,..\, Max(\{Len(l),\, i\}) \mapsto \text{IF } j = i \text{ THEN } x \text{ ELSE } l[j]]$

Messaging helpers

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

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

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

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

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

$Write(c) \triangleq$
- $\wedge\ cTime' = cTime + 1$
- $\wedge\ cSeqNum' = [cSeqNum \text{ EXCEPT } ![c] = cSeqNum[c] + 1]$
- $\wedge\ Sends(\{[src \mapsto c,\ dest \mapsto r,\ type \mapsto MWriteRequest,\ viewID \mapsto cViewID[c],\ seqNum \mapsto cSeqNum'[c],\ timestamp \mapsto cTime'] : r \in Replicas\})$
- $\wedge$ UNCHANGED $\langle globalVars,\, replicaVars,\, cViewID,\, cResps,\, cWrites,\, cReads\rangle$

$Read(c) \triangleq$
- $\wedge\ Sends(\{[src \mapsto c,\ dest \mapsto r,\ type \mapsto MReadRequest,\ viewID \mapsto cViewID[c]] : r \in Replicas\})$
- $\wedge$ UNCHANGED $\langle globalVars,\, replicaVars,\, cTime,\, cSeqNum,\, cResps,\, cWrites,\, cReads\rangle$

$HandleWriteResponse(c,\, r,\, m) \triangleq$
- $\wedge\ \vee\ \wedge\ m.viewID = cViewID[c]$
  - $\wedge$ IF $m.seqNum \notin$ DOMAIN $cResps[c][r]$ THEN
    - $cResps' = [cResps \text{ EXCEPT } ![c] = [cResps[c] \text{ EXCEPT } ![r] = cResps[c][r] \,@@\, (m.seqNum :> m)]]$
    - ELSE
    - $cResps' = [cResps \text{ EXCEPT } ![c] = [cResps[c] \text{ EXCEPT } ![r] = [cResps[c][r] \text{ EXCEPT } ![m.seqNum]$
  - $\wedge$ LET
    - $allResps \triangleq \{cResps[c][r][r1] : r1 \in \{r2 \in Replicas : r2 \in \text{DOMAIN } cResps[c][r]\}\}$
    - $isCommitted \triangleq \{r1.src : r1 \in \{r2 \in allResps : r2.succeeded\}\} \in Quorums$

```
        IN
            ∧ ∨ ∧ isCommitted
                  ∧ cWrites' = [cWrites EXCEPT ![c] = cWrites[c] ∪ {m}]
              ∨ ∧ ¬isCommitted
                  ∧ UNCHANGED ⟨cWrites⟩
              ∧ UNCHANGED ⟨cViewID, cSeqNum⟩
          ∨ ∧ m.viewID > cViewID[c]
            ∧ cViewID'  = [cViewID EXCEPT ![c] = m.viewID]
            ∧ cSeqNum'  = [cSeqNum EXCEPT ![c] = 0]
            ∧ cResps'   = [cResps  EXCEPT ![c]  = [i ∈ Replicas ↦ {}]]
            ∧ UNCHANGED ⟨cWrites⟩
          ∨ ∧ m.viewID < cViewID[c]
            ∧ UNCHANGED ⟨cWrites⟩
      ∧ Discard(m)
      ∧ UNCHANGED ⟨globalVars, replicaVars, cTime, cSeqNum, cReads⟩

HandleReadResponse(c, r, m) ≜
      ∧ ∨ ∧ m.viewID = cViewID[c]
            ∧ cReads' = [cReads EXCEPT ![c] = cReads[c] ∪ {m}]
            ∧ UNCHANGED ⟨cViewID, cSeqNum⟩
          ∨ ∧ m.viewID > cViewID[c]
            ∧ cViewID'  = [cViewID EXCEPT ![c] = m.viewID]
            ∧ cSeqNum'  = [cSeqNum EXCEPT ![c] = 0]
            ∧ UNCHANGED ⟨cReads⟩
          ∨ ∧ m.viewID < cViewID[c]
            ∧ UNCHANGED ⟨cViewID, cSeqNum, cReads⟩
      ∧ Discard(m)
      ∧ UNCHANGED ⟨globalVars, replicaVars, cTime, cSeqNum, cResps, cWrites⟩
```

Server request/response handling

```
Repair(r, c, m) ≜
      ∧ Replies(m, {[src      ↦ r,
                     dest     ↦ d,
                     type     ↦ MRepairRequest,
                     viewID   ↦ rViewID[r],
                     client   ↦ c,
                     seqNum ↦ rSeqNum[r][c] + 1] : d ∈ Replicas})

Abort(r, c, m) ≜
      ∧ IsPrimary(r)
      ∧ rStatus[r]      = SNormal
      ∧ rStatus'        = [rStatus      EXCEPT ![r]    = SAborting]
      ∧ rAbortResps'    = [rAbortResps  EXCEPT ![r]    = [rAbortResps[r] EXCEPT ![c] = {}]]
```

$\land\ rAbortSeqNum' = [rAbortSeqNum \text{ EXCEPT } ![r] = [rAbortSeqNum[r] \text{ EXCEPT } ![c] = m.seqNum]]$
$\land\ Replies(m, \{[src \mapsto r,$
$dest \mapsto d,$
$type \mapsto MAbortRequest,$
$viewID \mapsto rViewID[r],$
$client \mapsto c,$
$seqNum \mapsto m.seqNum] : d \in Replicas\})$

$HandleWriteRequest(r, c, m) \triangleq$
$\land\ rStatus[r] = SNormal$
$\land\ \lor\ \land\ m.viewID = rViewID[r]$
$\land\ m.seqNum = rSeqNum[r][c] + 1$
$\land\ \text{LET } entry \triangleq [type \mapsto TValue, value \mapsto m.value, timestamp \mapsto m.timestamp]$
$\text{IN } rLog' = [rLog \text{ EXCEPT } ![r] = Append(rLog[r], entry)]$
$\land\ rSeqNum' = [rSeqNum \text{ EXCEPT } ![r] = m.seqNum]$
$\land\ Reply(m, [src \mapsto r,$
$dest \mapsto c,$
$type \mapsto MWriteResponse,$
$seqNum \mapsto m.seqNum,$
$viewID \mapsto rViewID[r],$
$succeeded \mapsto \text{TRUE}])$
$\lor\ \land\ m.viewID = rViewID[r]$
$\land\ m.seqNum > rSeqNum[r][c] + 1$
$\land\ \lor\ \land\ IsPrimary(r)$
$\land\ Abort(r, c, m)$
$\lor\ \land\ \neg IsPrimary(r)$
$\land\ Repair(r, c, m)$
$\land\ \text{UNCHANGED } \langle rLog \rangle$
$\lor\ \land\ m.viewID < rViewID[r]$
$\land\ Reply(m, [src \mapsto r,$
$dest \mapsto c,$
$type \mapsto MWriteResponse,$
$seqNum \mapsto m.seqNum,$
$viewID \mapsto rViewID[r],$
$succeeded \mapsto \text{FALSE}])$
$\land\ \text{UNCHANGED } \langle rLog \rangle$
$\land\ \text{UNCHANGED } \langle globalVars, clientVars, rStatus, rViewID, rLastView, rViewChanges \rangle$

$HandleReadRequest(r, c, m) \triangleq$
$\land\ rStatus[r] = SNormal$
$\land\ Len(rLog[r]) > 0$
$\land\ Reply(m, [src \mapsto r,$
$dest \mapsto c,$
$type \mapsto MReadResponse,$
$viewID \mapsto rViewID[r],$

```
                    primary   ↦ IsPrimary(r),
                    index     ↦ Len(rLog[r]),
                    checksum  ↦ rLog[r][Len(rLog[r])].checksum,
                    succeeded ↦ TRUE])
    ∧ UNCHANGED ⟨globalVars, clientVars, rStatus, rLog, rViewID, rLastView, rViewChanges⟩

HandleRepairRequest(r, s, m) ≜
    ∧ m.viewID = rViewID[r]
    ∧ IsPrimary(r)
    ∧ rStatus[r] = SNormal
    ∧ ∨ ∧ m.seqNum ≤ Len(rLog[r][m.client])
        ∧ Reply(m, [src       ↦ r,
                    dest      ↦ s,
                    type      ↦ MRepairResponse,
                    viewID    ↦ rViewID[r],
                    client    ↦ m.client,
                    seqNum    ↦ m.seqNum])
        ∧ UNCHANGED ⟨rStatus, rAbortResps, rAbortSeqNum⟩
      ∨ ∧ m.seqNum = Len(rLog[r][m.client]) + 1
        ∧ Abort(r, m.client, m)
    ∧ UNCHANGED ⟨globalVars, clientVars⟩

HandleRepairResponse(r, s, m) ≜
    ∧ HandleWriteRequest(r, m.client, [m EXCEPT !.src = m.client])

HandleAbortRequest(r, s, m) ≜
    ∧ m.viewID  = rViewID[r]
    ∧ m.seqNum ≤ Len(rLog[r][m.client]) + 1
    ∧ rStatus[r] ∈ {SNormal, SAborting}
    ∧ LET entry ≜ [type ↦ TNoOp]
      IN  rLog' = [rLog EXCEPT ![r] = [rLog[r] EXCEPT ![m.client] = Replace(rLog[r][m.client], m.seqNum,
    ∧ ∨ ∧ m.seqNum > rSeqNum[r][m.client]
        ∧ rSeqNum'  = [rSeqNum EXCEPT ![r] = [rSeqNum[r] EXCEPT ![m.client] = m.seqNum]]
      ∨ ∧ m.seqNum ≤ rSeqNum[r][m.client]
        ∧ UNCHANGED ⟨rSeqNum⟩
    ∧ Replies(m, {[src       ↦ r,
                   dest      ↦ Primary(rViewID[r]),
                   type      ↦ MAbortResponse,
                   viewID    ↦ rViewID[r],
                   seqNum    ↦ m.seqNum],
                  [src       ↦ r,
                   dest      ↦ Primary(rViewID[r]),
                   type      ↦ MWriteResponse,
                   viewID    ↦ rViewID[r],
                   seqNum    ↦ m.seqNum,
                   succeeded ↦ FALSE]})
```

$\land$ UNCHANGED $\langle globalVars,\ clientVars,\ rStatus,\ rViewID,\ rLastView,\ rViewChanges\rangle$

$HandleAbortResponse(r,\ s,\ m)\ \triangleq$
$\land\ rStatus[r] = SAborting$
$\land\ m.viewID = rViewID[r]$
$\land\ IsPrimary(r)$
$\land\ m.seqNum = rAbortSeqNum[r][m.client]$
$\land\ rAbortResps' = [rAbortResps$ EXCEPT $![r] = [rAbortResps[r]$ EXCEPT $![m.client] = rAbortResps[r][m.cl$
$\land$ LET $resps\ \triangleq\ \{res.src : res \in \{resp \in rAbortResps'[r][m.client] :$
$\land\ resp.viewID\ = rViewID[r]$
$\land\ resp.seqNum = rAbortSeqNum[r][m.client]\}\}$
$isQuorum\ \triangleq\ r \in resps \land resps \in Quorums$
IN
$\lor\ \land\ isQuorum$
$\land\ rStatus' = [rStatus$ EXCEPT $![r] = [rStatus[r]$ EXCEPT $![m.client] = SNormal]]$
$\lor\ \land\ \neg isQuorum$
$\land$ UNCHANGED $\langle rStatus\rangle$
$\land$ UNCHANGED $\langle globalVars,\ clientVars\rangle$

$ChangeView(r)\ \triangleq$
$\land\ Sends(\{[src \mapsto r,$
$dest \mapsto d,$
$type \mapsto MViewChangeRequest,$
$viewID \mapsto rViewID[r] + 1] : d \in Replicas\})$
$\land$ UNCHANGED $\langle globalVars,\ clientVars,\ replicaVars\rangle$

$HandleViewChangeRequest(r,\ s,\ m)\ \triangleq$
$\land\ rViewID[r] < m.viewID$
$\land\ rViewID'\ = [rViewID$ EXCEPT $![r] = m.viewID]$
$\land\ rStatus'\ = [rStatus$ EXCEPT $![r]\ = SViewChange]$
$\land\ rViewChanges' = [rViewChanges$ EXCEPT $![r] = \{\}]$
$\land\ Reply(m, [src \mapsto r,$
$dest \mapsto Primary(m.viewID),$
$type \mapsto MViewChangeResponse,$
$viewID \mapsto m.viewID,$
$lastViewID \mapsto rLastView[r],$
$logs \mapsto rLog[r]])$
$\land$ UNCHANGED $\langle globalVars,\ clientVars,\ rLog,\ rSeqNum,\ rAbortSeqNum,\ rAbortResps,\ rLastView\rangle$

$HandleViewChangeResponse(r,\ s,\ m)\ \triangleq$
$\land\ IsPrimary(r)$
$\land\ rViewID[r]\ = m.viewID$
$\land\ rStatus[r]\ = SViewChange$
$\land\ rViewChanges' = [rViewChanges$ EXCEPT $![r] = rViewChanges[r] \cup \{m\}]$
$\land$ LET $viewChanges\ \triangleq\ \{v \in rViewChanges'[r][m.client] :\ \land\ v.viewID = rViewID[r]\}$
$viewSources\ \triangleq\ \{v.src : v \in viewChanges\}$

```
        isQuorum      ≜ r ∈ viewSources ∧ viewSources ∈ Quorums
        lastViews     ≜ {v.lastViewID : v ∈ viewChanges}
        lastView      ≜ (CHOOSE v1 ∈ lastViews : ∀ v2 ∈ lastViews : v2 ≤ v1)
        viewLogs      ≜ [c ∈ Clients ↦ {v1.logs[c] : v1 ∈ {v2 ∈ viewChanges : v2.lastView = lastView}
        mergeEnts(es) ≜
            IF es = {} ∨ ∃ e ∈ es : r.type = TNoOp THEN
                [type ↦ TNoOp]
             ELSE
                CHOOSE e ∈ es : e.type ≠ TNoOp
        range(ls)     ≜ Max({Len(l) : l ∈ ls})
        entries(ls, i) ≜ {l[i] : l ∈ {k ∈ ls : i ≤ Len(k)}}
        mergeLogs(ls) ≜ [i ∈ 1 .. range(ls) ↦ mergeEnts(entries(ls, i))]
    IN
        ∨ ∧ isQuorum
          ∧ Replies(m, {[src  ↦ r,
                         dest ↦ d,
                         type ↦ MStartViewRequest,
                         viewID ↦ rViewID[r],
                         logs ↦ [c ∈ Clients ↦ mergeLogs(viewLogs[c])]] : d ∈ Replicas})
        ∨ ∧ ¬isQuorum
          ∧ Discard(m)
    ∧ UNCHANGED ⟨globalVars, clientVars, rStatus, rViewID, rLog, rSeqNum, rAbortSeqNum, rAbortResps, r

HandleStartViewRequest(r, s, m) ≜
    ∧ ∨ rViewID[r] < m.viewID
      ∨ ∧ rViewID[r] = m.viewID
        ∧ rStatus[r] = SViewChange
    ∧ rLog'      = [rLog      EXCEPT ![r] = m.log]
    ∧ rStatus'   = [rStatus   EXCEPT ![r] = SNormal]
    ∧ rViewID'   = [rViewID   EXCEPT ![r] = m.viewID]
    ∧ rLastView' = [rLastView EXCEPT ![r] = m.viewID]
    ∧ Discard(m)
    ∧ UNCHANGED ⟨globalVars, clientVars, rViewChanges⟩

─────────────────────────────────────────────────────────────────────

InitMessageVars ≜
    ∧ messages = {}

InitClientVars ≜
    ∧ cTime     = 0
    ∧ cViewID   = [c ∈ Clients ↦ 1]
    ∧ cSeqNum   = [c ∈ Clients ↦ 0]
    ∧ cResps    = [c ∈ Clients ↦ [r ∈ Replicas ↦ [s ∈ {} ↦ [index ↦ 0, checksum ↦ Nil]]]]
    ∧ cWrites   = [c ∈ Clients ↦ {}]
```

$\land\ cReads = [c \in Clients \mapsto \{\}]$

$InitReplicaVars \triangleq$
$\land\ replicas = SeqFromSet(Replicas)$
$\land\ rStatus = [r \in Replicas \mapsto SNormal]$
$\land\ rLog = [r \in Replicas \mapsto [c \in Clients \mapsto \langle\rangle]]$
$\land\ rSeqNum = [r \in Replicas \mapsto [c \in Clients \mapsto 0]]$
$\land\ rAbortSeqNum = [r \in Replicas \mapsto [c \in Clients \mapsto 0]]$
$\land\ rAbortResps = [r \in Replicas \mapsto [c \in Clients \mapsto \{\}]]$
$\land\ rViewID = [r \in Replicas \mapsto 1]$
$\land\ rLastView = [r \in Replicas \mapsto 1]$
$\land\ rViewChanges = [r \in Replicas \mapsto \{\}]$

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

The type invariant checks that no read ever reads a different value than a previous write
$Inv \triangleq$ TRUE *TODO*

$Transition \triangleq transitions' = transitions + 1$

$Next \triangleq$
$\lor \exists\, c \in Clients :$
$\land\ Write(c)$
$\land\ Transition$
$\lor \exists\, c \in Clients :$
$\land\ Read(c)$
$\land\ Transition$
$\lor \exists\, r \in Replicas :$
$\land\ ChangeView(r)$
$\land\ Transition$
$\lor \exists\, m \in messages :$
$\land\ m.type = MWriteRequest$
$\land\ HandleWriteRequest(m.dest,\ m.src,\ m)$
$\land\ Transition$
$\lor \exists\, m \in messages :$
$\land\ m.type = MWriteResponse$
$\land\ HandleWriteResponse(m.dest,\ m.src,\ m)$
$\land\ Transition$
$\lor \exists\, m \in messages :$
$\land\ m.type = MReadRequest$
$\land\ HandleReadRequest(m.dest,\ m.src,\ m)$

$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MReadResponse$
$\land$ $HandleReadResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MRepairRequest$
$\land$ $HandleRepairRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MRepairResponse$
$\land$ $HandleRepairResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MAbortRequest$
$\land$ $HandleAbortRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MAbortResponse$
$\land$ $HandleAbortResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MViewChangeRequest$
$\land$ $HandleViewChangeRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MViewChangeResponse$
$\land$ $HandleViewChangeResponse(m.dest,\ m.src,\ m)$
$\land$ *Transition*
$\lor \exists\, m \in messages$:
$\land$ $m.type = MStartViewRequest$
$\land$ $HandleStartViewRequest(m.dest,\ m.src,\ m)$
$\land$ *Transition*

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

\ * Modification History
\ * Last modified *Tue Sep* 22 04:02:51 *PDT* 2020 by *jordanhalterman*
\ * Created *Fri Sep* 18 22:45:21 *PDT* 2020 by *jordanhalterman*