MODULE *ViewstampedReplication*

EXTENDS *Integers*, *Sequences*, *FiniteSets*

CONSTANTS *Replica*, *Quorum*

Replica *Status*

CONSTANTS *Normal*, *ViewChange*, *Recovering*

$Statuses \triangleq \{Normal, ViewChange, Recovering\}$

Client operation

CONSTANT *Operation*

Result of executing operation

$Result \triangleq Operation$

types of *log* blocks

CONSTANTS *RequestBlock*, *ViewBlock*

$RequestNumber \triangleq Nat$

Special value

CONSTANT *None*

Message types for processing client request

CONSTANTS *Request*, *Prepare*, *PrepareOk*, *Reply*, *Commit*

Message types for view changing

CONSTANTS *StartViewChange*, *DoViewChange*, *StartView*

Message types for replica recovery

CONSTANTS *Recovery*, *RecoveryResponse*

Sequence with all replicas (for view selection)

CONSTANT *ReplicaSequence*

State on each replica

VARIABLE *replicaState*

VARIABLE *msgs*

$vars \triangleq \langle replicaState, msgs \rangle$

$RequestMessage \triangleq [type : \{Request\}, op : Operation]$

$LogEntry \triangleq [type : \{RequestBlock\}, opNumber : Nat, m : RequestMessage]$
$\quad \cup [type : \{ViewBlock\}, view : Nat]$

All possible messages

$$
\begin{aligned}
&Message \triangleq [type : \{Prepare\},\ v : Nat,\ m : RequestMessage,\ n : Nat,\ k : Nat] \\
&\quad \cup [type : \{PrepareOk\},\ v : Nat,\ n : Nat,\ i : Replica] \\
&\quad \cup [type : \{Commit\},\ v : Nat,\ k : Nat] \\
&\quad \cup [type : \{StartViewChange\},\ v : Nat,\ i : Replica] \\
&\quad \cup [type : \{DoViewChange\},\ v : Nat,\ vv : Nat, \\
&\qquad n : Nat,\ k : Nat,\ i : Replica] \\
&\quad \cup [type : \{StartView\},\ v : Nat,\ n : Nat,\ k : Nat]
\end{aligned}
$$

$$Send(m) \triangleq msgs' = msgs \cup \{m\}$$

$$Drop(m) \triangleq \land msgs' = msgs \setminus \{m\}$$

$$
\begin{aligned}
&ReplyMessage(request,\ response) \triangleq \\
&\quad \land request \in msgs \\
&\quad \land msgs' = (msgs \setminus \{request\}) \cup \{response\}
\end{aligned}
$$

$$
\begin{aligned}
&TypeOK \triangleq \land replicaState \in [\\
&\qquad Replica \to [\\
&\qquad\quad viewNumber : Nat, \\
&\qquad\quad status : Statuses, \\
&\qquad\quad log : Seq(LogEntry), \\
&\qquad\quad downloadReplica : Replica \cup \{None\}, \\
&\qquad\quad commitNumber : Nat, \\
&\qquad\quad executedOperations : Seq(LogEntry) \\
&\qquad] \\
&\qquad] \\
&\qquad \land msgs \in \text{SUBSET } Message
\end{aligned}
$$

$$
\begin{aligned}
&\text{ASSUME } QuorumAssumption \triangleq \land \forall\, Q \in Quorum : Q \subseteq Replica \\
&\qquad\qquad\qquad\qquad\qquad\qquad\quad \land \forall\, Q1,\ Q2 \in Quorum : Q1 \cap Q2 \neq \{\}
\end{aligned}
$$

$$\text{ASSUME } IsFiniteSet(Replica)$$

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

$$Min(S) \triangleq \text{CHOOSE } x \in S : \forall\, y \in S : x \leq y$$

$$lastOpNumber(l) \triangleq \text{IF } l = \langle\rangle \text{ THEN } 0 \text{ ELSE } l[Len(l)].opNumber$$

$$
\begin{aligned}
&Init \triangleq \land replicaState = [r \in Replica \mapsto [\\
&\qquad\qquad viewNumber \mapsto 0, \\
&\qquad\qquad status \mapsto Normal, \\
&\qquad\qquad log \mapsto \langle [type \mapsto ViewBlock,\ view \mapsto 0] \rangle, \\
&\qquad\qquad downloadReplica \mapsto None, \\
&\qquad\qquad commitNumber \mapsto 0, \\
&\qquad\qquad executedOperations \mapsto \langle\rangle
\end{aligned}
$$

$]$

$]$

$\land \mathit{msgs} = \{\}$

Getters

$\mathit{ViewNumber}(r) \triangleq \mathit{replicaState}[r].\mathit{viewNumber}$

$\mathit{Status}(r) \triangleq \mathit{replicaState}[r].\mathit{status}$

$\mathit{Log}(r) \triangleq \mathit{replicaState}[r].\mathit{log}$

$\mathit{LogLen}(r) \triangleq \mathit{Len}(\mathit{Log}(r))$

$\mathit{LastNormalView}(r) \triangleq \mathit{Max}(\{v.\mathit{view} : v \in \{i \in 1 \,..\, \mathit{LogLen}(r) : \mathit{Log}(r)[i].\mathit{type} = \mathit{ViewBlock}\}\})$

$\mathit{OpNumber}(r) \triangleq \mathit{LogLen}(r)$

$\mathit{NewOpNumber}(r) \triangleq \mathit{Len}(\mathit{Log}(r)')$

$\mathit{DownloadReplica}(r) \triangleq \mathit{replicaState}[r].\mathit{downloadReplica}$

$\mathit{CommitNumber}(r) \triangleq \mathit{replicaState}[r].\mathit{commitNumber}$

$\mathit{ExecutedOperations}(r) \triangleq \mathit{replicaState}[r].\mathit{executedOperations}$

$\mathit{RecievedPrepareOkOpNumber}(r) \triangleq \mathit{replicaState}[r].\mathit{recievedPrepareOkOpNumber}$

Helpful functions

$\mathit{ExecuteOperation}(op) \triangleq op$

$\mathit{ReplicaIndex}(r) \triangleq \textsc{choose}\ i \in 1 \,..\, \mathit{Cardinality}(\mathit{Replica}) : \mathit{ReplicaSequence}[i] = r$

$\mathit{PrimaryReplicaInView}(v) \triangleq \mathit{ReplicaSequence}[(v \% \mathit{Len}(\mathit{ReplicaSequence})) + 1]$

$\mathit{IsPrimaryInView}(r, v) \triangleq \mathit{PrimaryReplicaInView}(v) = r$

$\mathit{IsPrimary}(r) \triangleq \mathit{IsPrimaryInView}(r, \mathit{ViewNumber}(r))$

$\mathit{IsDownloadingBeforeView}(r) \triangleq$

$\quad \land \mathit{replicaState}[r].\mathit{downloadReplica} \neq \mathit{None}$

$\mathit{AddClientRequest}(r, m) \triangleq$

$\quad \land \mathit{replicaState}' = [\mathit{replicaState}\ \textsc{except}\ ![r].\mathit{log} = \mathit{Append}(@, [$

$\qquad \mathit{type} \mapsto \mathit{RequestBlock},$

$\qquad \mathit{opNumber} \mapsto \mathit{OpNumber}(r) + 1,$

$\qquad m \mapsto m$

$\quad])]$

Implemented as Primary "generates" it by itself

$RecieveClientRequest(p,\ op) \triangleq$
$\land\ IsPrimary(p)$
$\land\ Status(p) = Normal$
$\land\ \neg IsDownloadingBeforeView(p)$
$\land\ AddClientRequest(p,\ [type \mapsto Request,\ op \mapsto op])$
$\land\ Send([type \mapsto Prepare,$
$v \mapsto ViewNumber(p),\ m \mapsto Log(p)'[OpNumber(p)+1].m,$
$n \mapsto OpNumber(p)+1,\ k \mapsto CommitNumber(p)])$

$RecievePrepare(r,\ m) \triangleq$
$\land\ \neg IsPrimary(r)$ Need this?
$\land\ Status(r) = Normal$
$\land\ \neg IsDownloadingBeforeView(r)$
$\land\ m.type = Prepare$
$\land\ m.v = ViewNumber(r)$
$\land\ m.n = OpNumber(r)+1$
$\land\ AddClientRequest(r,\ m.m)$
$\land\ Send([\ src \mapsto r,\ dst \mapsto PrimaryReplicaInView(viewNumber[r]), type \mapsto PrepareOk,$
$v \mapsto ViewNumber(r),\ n \mapsto m.n,\ i \mapsto r])$

$PrepareOperation(r) \triangleq$
$\land\ \neg IsPrimary(r)$
$\land\ Status(r) = Normal$
$\land\ \neg IsDownloadingBeforeView(r)$
$\land$ LET $maxPreparedOpNum \triangleq Max(\{0\} \cup \{m.n : m \in \{m \in msgs : m.type = PrepareOk \land m.i = r \land m.v$
IN $\land\ LogLen(r) > maxPreparedOpNum$
$\land\ Send([type \mapsto PrepareOk,\ v \mapsto ViewNumber(r),$
$n \mapsto maxPreparedOpNum+1,\ i \mapsto r])$
$\land$ UNCHANGED $\langle replicaState \rangle$

$ExecuteRequest(r,\ entry) \triangleq$
$\land\ replicaState' = [replicaState$ EXCEPT $![r].executedOperations = Append(@,\ entry)]$

$ExecuteClientRequest(r) \triangleq$
$\land\ Status(r) = Normal$
$\land\ \neg IsDownloadingBeforeView(r)$
$\land\ Len(ExecutedOperations(r)) < CommitNumber(r)$
$\land\ Len(ExecutedOperations(r)) < LogLen(r)$
$\land\ ExecuteRequest(r,\ Log(r)[Len(ExecutedOperations(r))+1])$
$\land$ UNCHANGED $\langle msgs \rangle$

$AchievePrepareOkFromQuorum(p) \triangleq$
$\land\ IsPrimary(p)$
$\land\ Status(p) = Normal$
$\land\ \neg IsDownloadingBeforeView(p)$

$\land$ $Len(ExecutedOperations(p)) = CommitNumber(p)$
$\land$ LET $newCommit \triangleq CommitNumber(p) + 1$
IN $\land \exists\, Q \in Quorum :$
$\forall\, r \in Q :$
$\lor\ Q \subseteq \{r\} \cup \{m.i : m \in \{m \in msgs : m.type = PrepareOk \land m.v = ViewNumber(p) \land m.n$
$\lor\ r = p$
$\land\ replicaState' = [replicaState$ EXCEPT $![p].commitNumber = newCommit,$
$![p].executedOperations = Append(@,\ Log(p)[newCommit])$
$\land\ Send([type \mapsto Commit,\ v \mapsto ViewNumber(p),\ k \mapsto replicaState[p].commitNumber'])$

$RecieveCommit(r,\ m) \triangleq$
$\land \neg IsPrimary(r)$ Need this?
$\land\ Status(r) = Normal$
$\land \neg IsDownloadingBeforeView(r)$
$\land\ m.type = Commit$
$\land\ m.k > CommitNumber(r)$
$\land\ replicaState' = [replicaState$ EXCEPT $![r].commitNumber = m.k]$
$\land$ UNCHANGED $\langle msgs \rangle$

View Changing

$TimeoutStartViewChanging(r) \triangleq$
$\land\ Status(r) = Normal$
$\land\ replicaState' = [replicaState$ EXCEPT $![r].downloadReplica = None,$
$![r].viewNumber = @ + 1,$
$![r].status = ViewChange]$
$\land\ Send([type \mapsto StartViewChange,\ v \mapsto ViewNumber(r)',\ i \mapsto r])$

$CheckAchieveStartViewChangeFromQuorum(r,\ v) \triangleq$
$\land$ IF $\exists\, Q \in Quorum : \land\ r \in Q$
$\land\ Q = \{r\} \cup \{m.i : m \in \{m \in msgs : m.type = StartViewChange \land m.v = replicaSt$
THEN $Send([$ $src \mapsto r,\ dst \mapsto PrimaryReplicaInView(m.v),$
$type \mapsto DoViewChange,\ v \mapsto v,\ vv \mapsto LastNormalView(r),$
$n \mapsto OpNumber(r),\ k \mapsto CommitNumber(r),\ i \mapsto r])$
ELSE UNCHANGED $\langle msgs \rangle$

$RecieveStartViewChange(r,\ m) \triangleq$
$\land\ m.type = StartViewChange$
$\land\ \lor$ Start View Changing
$\land\ ViewNumber(r) < m.v$
$\land\ replicaState' = [replicaState$ EXCEPT $![r].downloadReplica = None,$
$![r].viewNumber = m.v,$
$![r].status = ViewChange]$
$\land\ CheckAchieveStartViewChangeFromQuorum(r,\ m.v)$
$\lor$ Our view number

$\land$ $ViewNumber(r) = m.v$
$\land$ $Status(r) = ViewChange$
$\land$ UNCHANGED $\langle replicaState \rangle$
$\land$ $CheckAchieveStartViewChangeFromQuorum(r,\ m.v)$
$\lor$ Stale view
$\land$ $\lor$ $ViewNumber(r) > m.v$
$\lor$ $\land$ $ViewNumber(r) = m.v$
$\land$ $Status(r) = Normal$
$\land$ UNCHANGED $\langle replicaState,\ msgs \rangle$

$RecieveDoViewChange(p,\ m) \triangleq$
$\land$ $m.type = DoViewChange$
$\land$ $IsPrimaryInView(p,\ m.v)$
$\land$ $\lor$ Update view number
$\land$ $ViewNumber(p) < m.v$
$\land$ $replicaState' = [replicaState$ EXCEPT $![p].downloadReplica = None,$
$![p].viewNumber = m.v,$
$![p].status = ViewChange]$
$\lor$ Our view number or Stale message
$\land$ UNCHANGED $\langle replicaState \rangle$
$\land$ $Drop(m)$ Better no *Drop*?

Become Primary
$AchieveDoViewChangeFromQuorum(p) \triangleq$
$\land$ $IsPrimary(p)$
$\land$ $Status(p) = ViewChange$
$\land$ LET $recieved \triangleq \{m \in msgs : m.type = DoViewChange \land m.v = ViewNumber(p)\} \cup$
$\{[type \mapsto DoViewChange,\ v \mapsto ViewNumber(p),\ vv \mapsto LastNormalView(p),$
$n \mapsto OpNumber(p),\ k \mapsto CommitNumber(p),\ i \mapsto p]\}$
IN $\land \exists\, Q \in Quorum : \land p \in Q$
$\land Q \subseteq \{m.i : m \in recieved\}$

$\land$ LET $maxVV \triangleq Max(\{m.vv : m \in recieved\})$
$maxN \triangleq Max(\{m.n : m \in \{m \in recieved : m.vv = maxVV\}\})$
$maxReplicaIndex \triangleq Max(\{ReplicaIndex(m.i) : m \in \{m \in recieved : m.vv = maxVV \land m.n$
$maxReplica \triangleq$ (CHOOSE $m \in recieved : ReplicaIndex(m.i) = maxReplicaIndex).i$
IN $\land replicaState' = [replicaState$ EXCEPT $![p].downloadReplica = maxReplica,$
$![p].commitNumber = Max(\{m.k : m \in recieved\}),$
$![p].status = Normal]$

$\land$ UNCHANGED $\langle msgs \rangle$

TODO: add messages for downloading
$Mc \to Mc$ / $Mc \to M$
$MasterDownloadBeforeView(p) \triangleq$
$\land$ $IsPrimary(p)$
$\land$ $Status(p) = Normal$

```
    ∧ IsDownloadingBeforeView(p)
    ∧ ViewNumber(p) = ViewNumber(DownloadReplica(p))   If replica will increase view, then this Primary could on
    ∧ LogLen(p) ≤ LogLen(DownloadReplica(p))
    ∧ ∨ ∧ LogLen(p) = LogLen(DownloadReplica(p))
        ∧ replicaState' = [replicaState EXCEPT ![p].log = Append(@, [type ↦ ViewBlock, view ↦ ViewNumb
                                                ![p].downloadReplica = None]
        ∧ Send([type ↦ StartView,
                v ↦ ViewNumber(p),
                n ↦ OpNumber(p)',
                k ↦ replicaState[p].commitNumber'])
      ∨ ∧ LogLen(p) < LogLen(DownloadReplica(p))
        ∧ replicaState' = [replicaState EXCEPT ![p].log = Append(@, Log(DownloadReplica(p))[LogLen(p) +
        ∧ UNCHANGED ⟨msgs⟩

RecieveStartView(r, m) ≜
    ∧ m.type = StartView
    ∧ ∨ ViewNumber(r) < m.v
      ∨ ∧ ViewNumber(r) = m.v
        ∧ Status(r) = ViewChange
    ∧ replicaState' = [replicaState EXCEPT ![r].log = SubSeq(Log(r), 1, Min({LogLen(r), CommitNumber(r)}
                                           ![r].downloadReplica = PrimaryReplicaInView(m.v),
                                           ![r].viewNumber = m.v,
                                           ![r].status = Normal]
    ∧ UNCHANGED ⟨msgs⟩

 TODO: add messages for downloading
 Rc → Rc / Rc → R
ReplicaDownloadBeforeView(r) ≜
    ∧ ¬IsPrimary(r)
    ∧ Status(r) = Normal
    ∧ IsDownloadingBeforeView(r)
    ∧ IF LogLen(DownloadReplica(r)) ≤ LogLen(r)
      THEN ∧ replicaState' = [replicaState EXCEPT ![r].downloadReplica = None]
      ELSE LET newEntry ≜ Log(DownloadReplica(r))[LogLen(r) + 1]
           IN   ∧ replicaState' = [replicaState EXCEPT ![r].log = Append(@, newEntry),
                                                       ![r].downloadReplica =
                                                           IF newEntry = [type ↦ ViewBlock, view ↦ Vie
                                                            THEN None
                                                            ELSE @]
    ∧ UNCHANGED ⟨msgs⟩

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

Next ≜ ∨ ∃ r ∈ Replica, op ∈ Operation : RecieveClientRequest(r, op)
       ∨ ∃ r ∈ Replica, m ∈ msgs : RecievePrepare(r, m)
```

$\lor \exists\, r \in \mathit{Replica} : \mathit{PrepareOperation}(r)$
$\lor \exists\, p \in \mathit{Replica} : \mathit{AchievePrepareOkFromQuorum}(p)$
$\lor \exists\, r \in \mathit{Replica},\, m \in \mathit{msgs} : \mathit{RecieveCommit}(r,\, m)$
$\lor \exists\, r \in \mathit{Replica} : \mathit{ExecuteClientRequest}(r)$
$\lor \exists\, r \in \mathit{Replica}$: $\mathit{TimeoutStartViewChanging}(r)$
$\lor \exists\, r \in \mathit{Replica},\, m \in \mathit{msgs}$: $\mathit{RecieveStartViewChange}(r,\, m)$
$\lor \exists\, p \in \mathit{Replica},\, m \in \mathit{msgs}$: $\mathit{RecieveDoViewChange}(p,\, m)$
$\lor \exists\, r \in \mathit{Replica}$: $\mathit{AchieveDoViewChangeFromQuorum}(r)$
$\lor \exists\, p \in \mathit{Replica}$: $\mathit{MasterDownloadBeforeView}(p)$
$\lor \exists\, r \in \mathit{Replica},\, m \in \mathit{msgs}$: $\mathit{RecieveStartView}(r,\, m)$
$\lor \exists\, r \in \mathit{Replica}$: $\mathit{ReplicaDownloadBeforeView}(r)$

Liveness

$\mathit{EventuallyRecieveClientRequest} \triangleq \forall\, r \in \mathit{Replica} : \mathrm{WF}_{\mathit{vars}}(\exists\, \mathit{op} \in \mathit{Operation} : \mathit{RecieveClientRequest}(r,\, \mathit{op}))$

$\mathit{EventuallyRecievePrepare} \triangleq \forall\, r \in \mathit{Replica} : \mathrm{WF}_{\mathit{vars}}(\exists\, m \in \mathit{msgs} : \mathit{RecievePrepare}(r,\, m))$

$\mathit{EventuallyRecieveCommit} \triangleq \forall\, r \in \mathit{Replica} : \mathrm{WF}_{\mathit{vars}}(\exists\, m \in \mathit{msgs} : \mathit{RecieveCommit}(r,\, m))$

$\mathit{LivenessSpec} \triangleq$
$\quad \land\, \mathit{EventuallyRecieveClientRequest}$
$\quad \land\, \mathit{EventuallyRecievePrepare}$
$\quad \land\, \mathit{EventuallyRecieveCommit}$

Full *Spec*

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

$\mathit{VRNoMsgs} \triangleq \text{INSTANCE } \mathit{VR\_without\_message}$

INVARIANTS

$\mathit{EveryViewHasAtLeastOnePrimary} \triangleq \forall\, v \in 0 \,..\, 10 : \exists\, r \in \mathit{Replica} : \mathit{IsPrimaryInView}(r,\, v)$

$\mathit{EveryViewHasAtMostOnePrimary} \triangleq \forall\, v \in 0 \,..\, 10 : \forall\, r1,\, r2 \in \mathit{Replica} :$
$(\mathit{IsPrimaryInView}(r1,\, v) \land \mathit{IsPrimaryInView}(r2,\, v)) =$

$\mathit{PreficiesAreEqual}(s1,\, s2) \triangleq \forall\, i \in \text{DOMAIN } s1 \cap \text{DOMAIN } s2 : s1[i] = s2[i]$

$\mathit{ExecutedOperationsPreficesAreEqual} \triangleq \forall\, r1,\, r2 \in \mathit{Replica} : \mathit{PreficiesAreEqual}(\mathit{ExecutedOperations}(r1),\, \mathit{Exec}$

$\mathit{PreficiesOfLenAreEqual}(s1,\, s2,\, \mathit{prefLen}) \triangleq \forall\, i \ \in \text{DOMAIN } s1 \cap \text{DOMAIN } s2 \cap 1 \,..\, \mathit{prefLen} : s1[i] = s2[i]$

$CommitedLogsPrefecesAreEqual \triangleq \forall\, r1,\, r2 \in Replica : PreficiesOfLenAreEqual(Log(r1),\, Log(r2),\, Min(\{Com$

Properties

$AllClientsWillBeServed \triangleq \forall\, c \in \text{Client}: (pendingRequest[c] \rightsquigarrow \neg pendingRequest[c])$

---

* Modification History
* Last modified *Wed Apr* 05 21:44:23 *MSK* 2023 by *tycoon*
* Created *Mon Nov* 07 20:04:34 *MSK* 2022 by *tycoon*