MODULE *UpdateCluster*

EXTENDS *Integers*, *FiniteSets*

CONSTANTS
- $\_Requests$, the requests sent by the user
- $\_Workers$, the pool of workers
- $NULL$

VARIABLES
- $lastVOK$, last successfully applied version
- $toApply$, the version to apply (last requets that passed the initial tests)
- $cluster$, cluster state
- $requests$, the state of all requests
- $workers$, the state of all workers
- $lock$ damn, I used a lock...

VARIABLES

these variables are $tla+$ details
- $confOK$, are we able to get a valid conf ?
- $reqCounter$ just to keep track of the order of submissions

$vars \triangleq \langle confOK, reqCounter, lastVOK, toApply, cluster, requests, workers, lock \rangle$

$TypeInvariants \triangleq$
- $\land\ confOK \in$ BOOLEAN won't change for a specific behavior
- $\land\ lock \in$ BOOLEAN
- $\land\ cluster.st \in \{$
  - "idle",
  - "starting",
  - "partial",
  - "failed"
  - $\}$
- $\land \forall\, r \in \_Requests : requests[r].st \in \{$
  - "waiting", the request (req) hasn't been submitted yet
  - "submitted", req has been submitted
  - "rejected", req has been rejected (auth problem)
  - "valid" auth etc passed
  - $\}$
- $\land \forall\, w \in \_Workers : workers[w].st \in \{$
  - "waiting",
  - "starting",
  - "working"
  - $\}$

Initial State

$Init \triangleq$
$\land\ requests = [r \in \_Requests \mapsto [st \mapsto \text{“waiting”},\ v \mapsto NULL]]$
$\land\ workers\ = [w \in \_Workers \mapsto [st \mapsto \text{“waiting”},\ v \mapsto NULL]]$
$\land\ cluster\ = [v \mapsto 0,\ st \mapsto \text{“idle”}]$
$\land\ lastVOK = 0$
$\land\ reqCounter = 0$
$\land\ toApply = 0$
$\land\ confOK \in \text{BOOLEAN}$
$\land\ lock = \text{FALSE}$

Actions

$Submit(r) \triangleq$ update request received from the user
LET $newV \triangleq reqCounter + 1$ IN
$\land\ requests[r].st = \text{“waiting”}$
$\land\ reqCounter' = newV$
$\land\ requests' = [requests\ \text{EXCEPT}\ ![r].st = \text{“submitted”},\ ![r].v = newV]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ lastVOK,\ toApply,\ cluster,\ workers,\ lock\rangle$

$Initialcheck(r) \triangleq$ request validation (auth, quotas . . .)
$\land\ requests[r].st = \text{“submitted”}$
$\land\ \exists\, ok \in \text{BOOLEAN}\ :$
IF ok
THEN
$requests' = [requests\ \text{EXCEPT}\ ![r].st = \text{“valid”}]$
ELSE
$requests' = [requests\ \text{EXCEPT}\ ![r].st = \text{“rejected”}]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ cluster,\ workers,\ lock\rangle$

$PushToPending(r) \triangleq$ the request is pushed to queue
$\land\ requests[r].st = \text{“valid”}$
$\land$ IF $toApply < requests[r].v$
THEN $\land\ toApply' = requests[r].v$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ cluster,\ requests,\ workers,\ lock\rangle$
ELSE $\land\ requests' = [requests\ \text{EXCEPT}\ ![r].st = \text{“rejected”}]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ cluster,\ workers,\ lock\rangle$

$SpawnWorker(w) \triangleq$ spawns a new worker
$\land\ workers[w].st = \text{“waiting”}$
$\land\ toApply \neq lastVOK$
$\land\ lock = \text{FALSE}$
$\land\ \ \lor\ cluster.st = \text{“idle”}$
$\lor\ cluster.st = \text{“failed”}$
$\land\ workers' = [workers\ \text{EXCEPT}\ ![w].v = toApply,\ ![w].st = \text{“starting”}]$

$\land\ lock' = \text{TRUE}$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ requests,\ cluster\rangle$

$ApplyStart(w)\ \triangleq$ the cluster starts to be modified
$\land\ workers[w].st = \text{“starting”}$
$\land\ \text{IF}\ workers[w].v \geq toApply$
THEN
IF $confOK$
THEN
$\land\ cluster' = [v \mapsto workers[w].v,\ st \mapsto \text{“partial”}]$
$\land\ workers' = [workers\ \text{EXCEPT}\ ![w].st = \text{“working”}]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ requests,\ lock\rangle$
ELSE
$\land\ lock' = \text{FALSE}$
$\land\ workers' = [workers\ \text{EXCEPT}\ ![w].st = \text{“waiting”},\ ![w].v = NULL]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ cluster,\ requests\rangle$
ELSE a new version has been submitted, no need to apply this one
$\land\ workers' = [workers\ \text{EXCEPT}\ ![w].st = \text{“waiting”},\ ![w].v = NULL]$
$\land\ lock' = \text{FALSE}$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ cluster,\ requests\rangle$

$RollbackVersion\ \triangleq$
to differenciate it from the original last *VOK*
$lastVOK + 10$

$ApplyFinish(w)\ \triangleq$ the cluster update finishes
$\land\ workers[w].st = \text{“working”}$
$\land\ lock' = \text{FALSE}$
$\land\ \exists\, ok\ \in \text{BOOLEAN}\ :$
IF $ok \lor workers[w].v = RollbackVersion$ rollback always works
THEN
$\land\ cluster' = [cluster\ \text{EXCEPT}\ !.st = \text{“idle”}]$
$\land\ lastVOK' = workers[w].v$
$\land\ workers' = [workers\ \text{EXCEPT}\ ![w].st = \text{“waiting”},\ ![w].v = NULL]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ toApply,\ requests\rangle$
ELSE
$\land\ cluster' = [cluster\ \text{EXCEPT}\ !.st = \text{“failed”}]$
$\land\ workers' = [workers\ \text{EXCEPT}\ ![w].st = \text{“waiting”},\ ![w].v = NULL]$
$\land\ \text{IF}\ workers[w].v < toApply$
THEN a newer version has been submitted
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ toApply,\ requests\rangle$
ELSE let's trigger a rollback
$\land\ toApply' = RollbackVersion$
$\land\ \text{UNCHANGED}\ \langle confOK,\ reqCounter,\ lastVOK,\ requests\rangle$

Requirements

$NoConcurrentUpdate \triangleq$
$\quad \Box(Cardinality(\{r \in \text{DOMAIN } requests : requests[r].st = \text{“working”}\}) < 2)$

$NoPartialUpdateTermination \triangleq$ we don't want the cluster to end up in a partially update st
$\quad \Diamond\Box(cluster.st = \text{“idle”})$

$EveryReqIsProcessed \triangleq$
$\quad \Diamond\Box(\neg\exists\, r \in \_Requests : requests[r].st = \text{“waiting”})$

Spec

$Next \triangleq$
$\quad \lor \exists\, r \in \_Requests :$
$\qquad \lor Submit(r)$
$\qquad \lor Initialcheck(r)$
$\qquad \lor PushToPending(r)$
$\quad \lor \exists\, w \in \_Workers :$
$\qquad \lor SpawnWorker(w)$
$\qquad \lor ApplyStart(w)$
$\qquad \lor ApplyFinish(w)$

$Fairness \triangleq \forall\, r \in \_Requests,\ w \in \_Workers :$
$\qquad \land \text{WF}_{vars}(Submit(r))$
$\qquad \land \text{WF}_{vars}(Initialcheck(r))$
$\qquad \land \text{WF}_{vars}(PushToPending(r))$
$\qquad \land \text{WF}_{vars}(SpawnWorker(w))$
$\qquad \land \text{WF}_{vars}(ApplyStart(w))$
$\qquad \land \text{WF}_{vars}(ApplyFinish(w))$

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

THEOREM $Spec \Rightarrow \Box(TypeInvariants)$
THEOREM $Spec \Rightarrow NoPartialUpdateTermination$