MODULE *UpdateCluster*

EXTENDS *Integers*, *FiniteSets*

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

VARIABLES
- *confOK*, are we able to get a valid conf ?
- *lastVSubmitted*, just to keep track of the order of submissions
- *lastVOK*, last v where the cluster was fully applied (used by rollback)
- *toApply*, the version to apply (*lastVsubmitted* that passed the initial tests)
- *cluster*, last cluster fully deployed
- *requests*, the *st* of all requests (*requests*[*req*])
- *workers*,
- *lock*

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

$TypeInvariants \triangleq$
- $\land\ confOK \in$ 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
  - "processing", the processing of the req has started
  - "partial", req is partially applied (the infra is partially modified)
  - "partialFailure", req failed in the middle of an application
  - "success", req has been sucessfully applied
  - "failure", the req failed before modifying the cluster
  - "rolledback" the req has been *rolledback*
  - $\}$
- $\land \forall\, w \in \_Workers : workers[w].st \in \{$
  - "waiting",
  - "starting",
  - "working"

}

Initial State

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

Actions

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

$Initialcheck(r) \triangleq$ 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
$requests' = [requests\ \text{EXCEPT}\ ![r].st = \text{“rejected”}]$
$\land\ \text{UNCHANGED}\ \langle confOK,\ lastVSubmitted,\ lastVOK,\ toApply,\ cluster,\ workers,\ lock\rangle$

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

$SpawnWorker(w) \triangleq$ spawns a new worker
$\land\ workers[w].st = \text{“waiting”}$

```
    ∧ lock = FALSE
    ∧  ∨ cluster.st = "idle"
       ∨ cluster.st = "failed"
    ∧ IF cluster.st = "idle"
      THEN
          ∧ workers' = [workers EXCEPT ![w].v = toApply, ![w].st = "starting"]
      ELSE
          ∧ workers' = [workers EXCEPT ![w].v = lastVOK, ![w].st = "starting"]
    ∧ lock' = TRUE
    ∧ UNCHANGED ⟨confOK, lastVSubmitted, lastVOK, toApply, requests, cluster⟩


ApplyStart(w) ≜  the cluster starts to be modified
    ∧ workers[w].st = "starting"
    ∧ IF  ∨ workers[w].v = lastVSubmitted
          ∨ workers[w].v = lastVOK  rollingback
      THEN
        IF confOK
          THEN
              ∧ cluster'  = [v ↦ workers[w].v, st ↦ "partial"]
              ∧ workers' = [workers EXCEPT ![w].st = "working"]
              ∧ UNCHANGED ⟨confOK, lastVSubmitted, lastVOK, toApply, requests, lock⟩
          ELSE
              ∧ lock' = FALSE
              ∧ workers' = [workers EXCEPT ![w].st = "waiting", ![w].v = NULL]
              ∧ UNCHANGED ⟨confOK, lastVSubmitted, lastVOK, toApply, cluster, requests⟩
      ELSE
          ∧ workers' = [workers EXCEPT ![w].st = "waiting", ![w].v = NULL]
          ∧ UNCHANGED ⟨confOK, lastVSubmitted, lastVOK, toApply, cluster, requests, lock⟩

ApplyFinish(w) ≜  the cluster update finishes
    ∧ workers[w].st = "working"
    ∧ lock' = FALSE
    ∧ ∃ ok ∈ BOOLEAN :
      IF ok ∨ workers[w].v = lastVOK  rollback always works
          THEN
              ∧ cluster'  =  [cluster EXCEPT !.st = "idle"]
              ∧ lastVOK' = workers[w].v
              ∧ workers'  = [workers EXCEPT ![w].st = "waiting", ![w].v = NULL]
              ∧ UNCHANGED ⟨confOK, lastVSubmitted, toApply, requests⟩
          ELSE
              ∧ cluster'  =  [cluster EXCEPT !.st = "failed"]
              ∧ workers' = [workers EXCEPT ![w].st = "waiting", ![w].v = NULL]
              ∧ UNCHANGED ⟨confOK, lastVSubmitted, lastVOK, toApply, requests⟩
```

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 $Spec \Rightarrow NoPartialUpdateTermination$
THEOREM $Spec \Rightarrow NoApplicationOfOutdatedReq$
THEOREM $Spec \Rightarrow EveryReqInQueueIsProcessed$