```
──────────────────────── MODULE DistFinish ────────────────────────
Resilient distributed finish as implemented in PPoPP14 See
FinishState.FinishResilientDistributed
EXTENDS Integers, Sequences
CONSTANTS PLACE, MXFINISHES, PROG_HOME, BACKUP
VARIABLES fid, fstates, msgs, thrds, pstate, waitForMsgs, killed, fbackups, seq

INSTANCE Commons

Terminated ≜
    ∧ fstates[fid].status = "forgotten"

Running ≜
    ∧ fstates[fid].status = "waiting"

IsRoot ≜
    ∧ fstates[fid].type = "distroot"

LastActivity ≜
    ∧ fstates[fid].count = 1


SendMasterTransit(dst) ≜
    ∧ dst ≠ fstates[fid].here
    ∧ LET parentId ≜ fstates[fid].parent
          here ≜ fstates[fid].here
          root ≜ fstates[fid].root
          rootPlace ≜ GetFinishHome(fstates[fid].root)
      IN  ∧ SendMsg([mid       ↦ seq.mseq,
                         src       ↦ here,
                         dst       ↦ rootPlace,
                         target    ↦ dst,
                         fid       ↦ root,
                         taskFID   ↦ fid,
                         finishSrc ↦ fstates[fid].src,
                         type      ↦ "masterTransit"])
          ∧ waitForMsgs′ = waitForMsgs ∪ {[src       ↦ rootPlace,
                                                dst       ↦ here,
                                                target    ↦ dst,
                                                fid       ↦ root,
                                                taskFID   ↦ fid,
                                                finishSrc ↦ fstates[fid].src,
                                                type      ↦ "masterTransitDone" ]}
          ∧ IncrMSEQ(1)

SendMasterLiveToCompleted(source, finishEnd) ≜
    LET root ≜ fstates[fid].root
        rootPlace ≜ GetFinishHome(fstates[fid].root)
        $here \triangleq fstates[fid].here$
    IN  $\land SendMsg([mid \mapsto seq.mseq,$
                        $src \mapsto here,$
                        $dst \mapsto rootPlace,$
                        $source \mapsto$ IF $finishEnd$ THEN $here$ ELSE $source,$
                        $target \mapsto here,$
                        $fid \mapsto root,$
                        $taskFID \mapsto fid,$
                        $finishEnd \mapsto finishEnd,$
                        $type \mapsto$ "masterCompleted"$])$
        $\land waitForMsgs' = waitForMsgs \cup \{[src \mapsto rootPlace,$
                                                  $dst \mapsto here,$
                                                  $source \mapsto$ IF $finishEnd$ THEN $here$ ELSE $source,$
                                                  $target \mapsto here,$
                                                  $fid \mapsto root,$
                                                  $taskFID \mapsto fid,$
                                                  $type \mapsto$ "masterCompletedDone" $]\}$
        $\land IncrMSEQ(1)$
```

```
$Alloc(type, here, parent, root, finishSrc) \triangleq$
    LET $encRoot \triangleq GetEnclosingRoot(parent, fid)$
        $encRootPlace \triangleq$ IF $fid = FIRST\_ID$ THEN $PROG\_HOME$ ELSE $fstates[encRoot].here$
    IN  $\land fstates[fid].status =$ "unused"
        $\land fstates' = [fstates$ EXCEPT $![fid].id = fid,$
                                        $![fid].count = 1,$
                                        $![fid].status =$ "waiting",
                                        $![fid].type = type,$
                                        $![fid].here = here,$
                                        $![fid].parent = parent,$
                                        $![fid].root = root,$
                                        $![fid].eroot = encRoot,$
                                        $![fid].isGlobal = (type =$ "distremote"$),$
                                        $![fid].src$ = IF $type =$ "distroot"
                                                          THEN $finishSrc$
                                                          ELSE $NotPlace,$
                                        $![fid].received[finishSrc] = @ + 1]$
```

needed for the local path of *Runtime.runAsync*

```
$NotifyLocalActivitySpawnAndCreation(here, act) \triangleq$
    $\land fstates[fid].status =$ "waiting"
    $\land fstates' = [fstates$ EXCEPT $![fid].count = @ + 1]$

$NotifySubActivitySpawn(dst) \triangleq$
    $\land fstates[fid].status =$ "waiting"
    $\land fstates' = [fstates$ EXCEPT $![fid].isGlobal =$ TRUE$]$
    ∧ $SendMasterTransit(dst)$

$AllocRemoteAndNotifyRemoteActivityCreation(src, act, inMsg, type, here, parent, root, finishSrc) \triangleq$
    ∧ $RecvMsg(inMsg)$
    ∧ $here \neq NotPlace$
    ∧ $type =$ "distremote"                create and notify
    ∧ $Alloc(type, here, parent, root, finishSrc)$

$NotifyRemoteActivityCreation(src, act, inMsg, type, here, parent, root, finishSrc) \triangleq$
    ∧ $RecvMsg(inMsg)$
    ∧ $here \neq NotPlace$
    ∧ $type =$ "distremote"
    ∧ $fstates' = [fstates$ EXCEPT $![fid].received[finishSrc] = @ + 1]$

$NotifyActivityTermination(source, finishEnd) \triangleq$
    ∧ $fstates[fid].status =$ "waiting"
    ∧ $fstates[fid].count > 0$
    ∧ IF $LastActivity \land \neg fstates[fid].isGlobal$
       THEN ∧ $fstates' = [fstates$ EXCEPT $![fid].count = @ - 1,$
                                          $![fid].status =$ "forgotten"$]$
            ∧ $msgs' = msgs$
            ∧ $seq' = seq$
            ∧ $waitForMsgs' = waitForMsgs$
       ELSE IF $LastActivity \land fstates[fid].isGlobal$
            THEN ∧ $SendMasterLiveToCompleted(source, finishEnd)$
                 ∧ $fstates' = [fstates$ EXCEPT $![fid].count = @ - 1,$
                                               $![fid].status =$ IF $fstates[fid].type =$ "distremote"
                                                                THEN "forgotten"
                                                                ELSE "pendingRelease"$]$
            ELSE ∧ $fstates' = [fstates$ EXCEPT $![fid].count = @ - 1]$
                 ∧ $msgs' = msgs$
                 ∧ $seq' = seq$
                 ∧ $waitForMsgs' = waitForMsgs$
```

* Modification History
* Last modified *Thu Dec* 14 18:05:06 *AEDT* 2017 by *u*5482878
* Last modified Sun *Dec* 10 12:28:32 *AEDT* 2017 by *shamouda*
* Created *Tue Nov* 07 17:50:59 *AEDT* 2017 by *u*5482878