MODULE *DistFinish*

Resilient distributed finish as implemented in *PPoPP*14 See *FinishState.FinishResilientDistributed*

EXTENDS *Integers*, *Sequences*

CONSTANTS *PLACE*, *MXFINISHES*, *PROG_HOME*, *BACKUP*

VARIABLES *fid*, *fstates*, *msgs*, *thrds*, *pstate*, *waitForMsgs*, *killed*, *fbackups*, *seq*

INSTANCE *Commons*

$$Terminated \triangleq \\ \quad \land fstates[fid].status = \text{“forgotten”}$$

$$Running \triangleq \\ \quad \land fstates[fid].status = \text{“waiting”}$$

$$IsRoot \triangleq \\ \quad \land fstates[fid].type = \text{“distroot”}$$

$$LastActivity \triangleq \\ \quad \land fstates[fid].count = 1$$

$$\begin{array}{l} SendMasterAddChild(eroot,\ erootPlace,\ here) \triangleq \\ \quad \land SendMsg([mid \mapsto seq.mseq, \\ \qquad\qquad src \mapsto here, \\ \qquad\qquad dst \mapsto erootPlace, \\ \qquad\qquad eroot \mapsto eroot, \\ \qquad\qquad fid \mapsto fid, \\ \qquad\qquad type \mapsto \text{“addChild”}]) \\ \quad \land IncrMSEQ(1) \\ \quad \land waitForMsgs' = waitForMsgs \cup \{[src \mapsto erootPlace, \\ \qquad\qquad\qquad\qquad\qquad\qquad dst \mapsto here, \\ \qquad\qquad\qquad\qquad\qquad\qquad fid \mapsto fid, \\ \qquad\qquad\qquad\qquad\qquad\qquad eroot \mapsto eroot, \\ \qquad\qquad\qquad\qquad\qquad\quad type \mapsto \text{“addChildDone”}\ ]\} \end{array}$$

$$\begin{array}{l} SendMasterTransit(dst) \triangleq \\ \quad \land dst \neq fstates[fid].here \\ \quad \land \text{LET}\ parentId \triangleq fstates[fid].parent \\ \qquad\quad here \triangleq fstates[fid].here \\ \qquad\quad root \triangleq fstates[fid].root \\ \qquad\quad rootPlace \triangleq GetFinishHome(fstates[fid].root) \\ \qquad \text{IN} \quad \land SendMsg([mid \mapsto seq.mseq, \\ \qquad\qquad\qquad\qquad src \mapsto here, \\ \qquad\qquad\qquad\qquad dst \mapsto rootPlace, \\ \qquad\qquad\qquad\quad target \mapsto dst, \\ \qquad\qquad\qquad\qquad fid \mapsto root, \\ \qquad\qquad\qquad\qquad type \mapsto \text{“masterTransit”}]) \end{array}$$

$$
\begin{array}{l}
\quad \land\ waitForMsgs' = waitForMsgs \cup \{[src \mapsto rootPlace, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad dst \mapsto here, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad target \mapsto dst, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad fid \mapsto root, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad type \mapsto \text{“masterTransitDone”}\ ]\} \\
\quad \land\ IncrMSEQ(1)
\end{array}
$$

$$
\begin{array}{l}
SendMasterTransitToLive(src,\ actId,\ inMsg,\ here,\ root)\ \triangleq \\
\quad \text{LET}\ rootPlace \triangleq GetFinishHome(root) \\
\quad \text{IN}\ \ \land\ ReplaceMsg(inMsg, \\
\qquad\qquad\qquad [mid \mapsto seq.mseq, \\
\qquad\qquad\qquad\ src \mapsto here, \\
\qquad\qquad\qquad source \mapsto src, \\
\qquad\qquad\qquad target \mapsto here, \\
\qquad\qquad\qquad\ dst \mapsto rootPlace, \\
\qquad\qquad\qquad\ fid \mapsto root, \\
\qquad\qquad\qquad\ aid \mapsto actId, \\
\qquad\qquad\qquad type \mapsto \text{“masterLive”}]) \\
\qquad\ \land\ waitForMsgs' = waitForMsgs \cup \{[src \mapsto rootPlace, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad dst \mapsto here, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad fid \mapsto root, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad aid \mapsto actId, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad source \mapsto src, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad target \mapsto here, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad type \mapsto \text{“masterLiveDone”}\ ]\}
\end{array}
$$

$$
\begin{array}{l}
SendMasterLiveToCompleted(finishEnd)\ \triangleq \\
\quad \text{LET}\ root \triangleq fstates[fid].root \\
\qquad rootPlace \triangleq GetFinishHome(fstates[fid].root) \\
\qquad here \triangleq fstates[fid].here \\
\quad \text{IN}\ \ \land\ SendMsg([mid \mapsto seq.mseq, \\
\qquad\qquad\qquad\qquad\ src \mapsto here, \\
\qquad\qquad\qquad\qquad\ dst \mapsto rootPlace, \\
\qquad\qquad\qquad\qquad target \mapsto here, \\
\qquad\qquad\qquad\qquad\ fid \mapsto root, \\
\qquad\qquad\qquad finishEnd \mapsto finishEnd, \\
\qquad\qquad\qquad\qquad type \mapsto \text{“masterCompleted”}]) \\
\qquad\ \land\ waitForMsgs' = waitForMsgs \cup \{[src \mapsto rootPlace, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad dst \mapsto here, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad target \mapsto here, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad fid \mapsto root, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad isAdopter \mapsto \text{FALSE}, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad type \mapsto \text{“masterCompletedDone”}\ ]\} \\
\qquad\ \land\ IncrMSEQ(1)
\end{array}
$$

```
Alloc(type, here, parent, root) ≜
    LET encRoot ≜  GetEnclosingRoot(parent, fid)
        encRootPlace ≜ IF fid = FIRST_ID THEN PROG_HOME ELSE fstates[encRoot].here
    IN  ∧ fstates[fid].status = "unused"
        ∧ 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 = IF type = "distremote"
                                                       THEN TRUE
                                                       ELSE FALSE]
```

needed for the local path of *Runtime.runAsync*

```
NotifyLocalActivitySpawnAndCreation(here, act) ≜
    ∧ fstates[fid].status = "waiting"
    ∧ fstates' = [fstates EXCEPT ![fid].count = @ + 1]

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

AllocRemoteAndNotifyRemoteActivityCreation(src, act, inMsg, type, here, parent, root) ≜
    ∧ SendMasterTransitToLive(src, act.aid, inMsg, here, root)
    ∧ here ≠ NotPlace
    ∧ type = "distremote"                    create and notify
    ∧ Alloc(type, here, parent, root)

NotifyActivityTermination(finishEnd) ≜
    ∧ fstates[fid].status = "waiting"
    ∧ fstates[fid].count > 0
    ∧ IF LastActivity ∧ ¬fstates[fid].isGlobal
      THEN ∧ fstates' = [fstates EXCEPT ![fid].count = @ − 1,
                                        ![fid].status = "forgotten"]
           ∧ msgs' = msgs
           ∧ seq' = seq
           ∧ waitForMsgs' = waitForMsgs
      ELSE IF LastActivity ∧ fstates[fid].isGlobal
           THEN ∧ SendMasterLiveToCompleted(finishEnd)
                ∧ fstates' = [fstates EXCEPT ![fid].count = @ − 1,
                                             ![fid].status = IF fstates[fid].type = "distremote"
                                                             THEN "forgotten"
                                                             ELSE "pendingRelease"]
```

$$\begin{array}{ll} \text{ELSE} & \land\, fstates' = [fstates \text{ EXCEPT } ![fid].count = @ - 1] \\ & \land\, msgs' = msgs \\ & \land\, seq' = seq \\ & \land\, waitForMsgs' = waitForMsgs \end{array}$$

* Modification History
* Last modified *Mon Dec* 11 16:46:52 *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