**Communication Ring Check Up Algorithm RUP.GD.Auto**

(Formalized Specification)

**Pi::RUP.GD.Auto**

**{SYNOPSIS}**

Distributed algorithm for communication ring integrity check up.

It is started just after opening of the output channel of the process which connected it with the immediate neighbor.

The “AutoList” variant of this algorithm automatically fills in the list of processes’ ids. It does not require this list to be known in advance.

**{ASSUMPTIONS}**

The RUP.GD.Auto algorithm is valid if the following conditions are met:

* The distributed system is synchronous.
* The type of process failures is strongly “fail-stop”, but not all failures are allowed if AutoList mode is selected: only failures of other processes (j ≠ i) and only after finishing of the algorithm for Pi are allowed in this mode.

**ALGORITHM 1**: Declarative Part of *Pi::RUP.GD.Auto*

**{SYSTEM CONSTANTS}**

Int MAX\_RUP\_PERIOD // period to next check

String MRK\_RUP // message type „RUP Token“

String MRK\_RUP2 // message type „RUP AutoList Token“

PId i // process Pi identifier

PId j // default neighbor process Pj

CEH::AutoList // <true> if ListPIds is to be filled during RUP

// <false> if ListPIds is known in advance

**{MESSAGES}**

<mrk\_rup, i>

<mrk\_rup2, i, list>

**{SET OF STATES}**

<State> := {INIT, DOWN, UP}

**{INTERNAL STATE SPACE}**

State state // current process RUP state

Timer TimerRUP // timer

CEH::ListPIds // list of process identifiers

CEH::PIdNext // current neighbor process identifier

**ALGORITHM 2**: Event Handlers of *Pi::RUP.GD.Auto*

**OnInit:**

state := INIT

TimerRUP.Interval := MAX\_RUP\_PERIOD

**OnOutputConnect:**

*{Ring Check Up First Attempt}*

state := DOWN

**If** CEH::AutoList = true

Send <mrk\_rup2, i, list.Clear()>

**Else**

Send <mrk\_rup, i>

**End If**

TimerRUP.Start()

**OnOutputDisconnect:**

state := DOWN

**OnOutputError:**

state := DOWN

**OnReceiptOf <mrk\_rup, j> ∪ OnReceiptOf <mrk\_rup2, j, list>:**

**If** j = i

TimerRUP.Stop()

state := UP

ListPIds := list

*{Distributed Election Entry Point}*

E::OnStartElection()

**Else**

**If** CEH::AutoList = true

Send <mrk\_rup2, j, list.Add(i)>

**Else**

Send <mrk\_rup, j>

**EndIf**

**End If**

**OnTimer:**

*{Ring Check Up Next Attempt}*

TimerRUP.Stop()

**If** CEH::AutoList = true

Send <mrk\_rup2, i, list.Clear()>

**Else**

Send <mrk\_rup, i>

**End If**

TimerRUP.Start()