МОДИФИЦИРАН КРЪГОВ АЛГОРИТЪМ ЗА РАЗПРЕДЕЛЕНО ВЗАИМНО ИЗКЛЮЧВАНЕ

Modified Distributed Ring-based Mutual Exclusion Algorithm *Mx1ME*

(Generalized Formal Specification[[1]](#footnote-1))

**Pi::Mx1ME**

**{SYNOPSIS}**

**The basic distributed ring-based mutual exclusion algorithm described in the literature is modified to integrate it with the Token Management/Recovery Algorithm MrkME[[2]](#footnote-2). This overcomes the main drawback of the basic algorithm - the assumption of inadmissibility of failures.**

**{ASSUMPTIONS}**

**The algorithm is valid if the following conditions are met:**

* **The type of process failures is strongly “*fail-stop*”.**

**Physical failure/recovery of any of the system processes is allowed at any time.**

**After recovery of the failed process, all other processes in the ring continue their normal execution from the operating state they were in at the time of the failure, and only one instance of ME marker is restored.**

**{SYSTEM CONSTANTS}**

String MRK\_ME\_CLR // Init ME token message type (Pass 1)

String MRK\_ME // ME token message type (Pass 2)

**{MESSAGES}**

<mrk\_me\_clr>

<mrk\_me, *Ti*>

**{SET OF STATES}**

<State> := {ME\_INIT, ME\_RELEASED, ME\_WANTED, ME\_HELD}

**{INTERNAL STATE SPACE}**

State meState // current process ME state

E::pidCoordinator // Coordinator id

MrkME::boolMrkME // current process MrkME state

MrkME::timeMrkME // characteristic token time*Ti*

MrkME::strClrPending // hold message <mrk\_me\_clr>

**{EVENTS}**

**OnInit:**

meStatus := ME\_INIT

**OnRingUp:**

{Start of Distributed Election}

Е::pidCoordinator := Election()

**OnAfterElection:**

**If** meStatus = ME\_INIT

meStatus := ME\_RELEASED

**EndIf**

**OnEnter:**

**If** meStatus = ME\_RELEASED

**If** timeMrkME ≠ NULL

meStatus := ME\_WANTED

**EndIf**

**EndIf**

**OnReceiptOf <mrk\_me>:**

**If** meStatus = ME\_WANTED

**If** timeMrkME ≠ NULL

meStatus := ME\_HELD

**EndIf**

**Else If** meStatus = ME\_RELEASED

Send <mrk\_me>

**EndIf**

**OnRelease:**

**If** meStatus = ME\_HELD

meStatus := ME\_RELEASED

Send <mrk\_me>

**If** MrkME::strClrPending ≠ NULL

MrkME::OnClear()

MrkME::strClrPending := NULL

**EndIf**

**EndIf**

1. Implementation <https://github.com/milphaser/XME.Ring> [↑](#footnote-ref-1)
2. Spec.Recovery.RingBased.MrkME.docx [↑](#footnote-ref-2)