MODULE *Collision*

EXTENDS *Naturals*, *FiniteSets*, *Commons*

CONSTANT *NPROCESSES*
CONSTANT *NMESSAGES*
CONSTANT *CONFLICTR*(_, _)

Since this algorithm is for failure-free environments, the set of all processes is the same as the correct ones.

LOCAL $Processes \triangleq \{i : i \in 1 \,..\, NPROCESSES\}$
LOCAL $ChooseProcess \triangleq$ CHOOSE $x \in Processes$: TRUE
LOCAL $Create(id) \triangleq [id \mapsto id,\ d \mapsto Processes,\ o \mapsto ChooseProcess]$
LOCAL $AllMessages \triangleq \{Create(id) : id \in 1 \,..\, NMESSAGES\}$

VARIABLES
K,
$Pending$,
$Delivering$,
$Delivered$,
$PreviousMsgs$,
$Votes$,
$QuasiReliableChannel$

Initialize the instance for the Generic Multicast 0. The *INITIAL_MESSAGES* is a set with *NMESSAGES*, unordered, a tuple with the starting state *S*0 and the message.

$Algorithm \triangleq$ INSTANCE $GenericMulticast0$ WITH
$INITIAL\_MESSAGES \leftarrow \{\langle \text{“S0”},\ m\rangle : m \in AllMessages\}$

$Spec \triangleq Algorithm!Spec$

If a correct process p delivers messages m and n, p is in the destination of both messages, m and n do not commute. Then, p delivers either m and then n or n and then m.

$$
\begin{array}{l}
Collision \triangleq \\
\quad \Box\forall\, p \in Processes : \\
\qquad \forall\, m,\ n \in AllMessages : \ \wedge\, m.id \neq n.id \\
\qquad\qquad \wedge\, Algorithm!WasDelivered(p,\ m) \\
\qquad\qquad \wedge\, Algorithm!WasDelivered(p,\ n) \\
\qquad\qquad \wedge\, CONFLICTR(m,\ n) \\
\qquad\qquad\quad \Rightarrow Algorithm!DeliveredInstant(p,\ m) \neq \\
\qquad\qquad\qquad Algorithm!DeliveredInstant(p,\ n)
\end{array}
$$