MODULE *Agreement*

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

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

LOCAL *Processes* $\triangleq \{i : i \in 1 \,..\, NPROCESSES\}$
LOCAL *Groups* $\triangleq 1 \,..\, NGROUPS$
LOCAL *ProcessesInGroup* $\triangleq [g \in Groups \mapsto Processes]$

LOCAL *AllMessages* $\triangleq$ *CreateMessages*(*NMESSAGES*, *Groups*, *Processes*)
LOCAL *MessagesCombinations* $\triangleq$ *CreatePossibleMessages*(*AllMessages*)

VARIABLES *K*, *PreviousMsgs*, *Delivered*, *Votes*, *MemoryBuffer*,
QuasiReliableChannel, *AtomicBroadcastBuffer*

Initialize the instance for the Generic Multicast 1. The *INITIAL_MESSAGES* is a sequence, totally ordered within a group, wherein the elements are tuples with the message, state, and timestamp.

Algorithm $\triangleq$ INSTANCE *GenericMulticast*1 WITH
INITIAL_MESSAGES $\leftarrow$ [
$g \in Groups \mapsto$
TotallyOrdered(*MessagesCombinations*[(*g*%*NMESSAGES*) + 1])]

Weak fairness is necessary.

Spec $\triangleq$ *Algorithm*!*SpecFair*

If a correct process deliver a message m , then all correct processes in $m.d$ eventually delivers m .

We verify that all messages in *AllMessages*, for all the processes that delivered a message, eventually, all the correct members in the destination will deliver.

Agreement $\triangleq$
$\forall\, m \in AllMessages :$
$\forall\, g\_i \in Groups :$
$\exists\, p\_i \in ProcessesInGroup[g\_i] :$
Algorithm!*WasDelivered*(*g_i*, *p_i*, *m*)
$\rightsquigarrow \forall\, g\_j \in m.d :$
$\exists\, p\_j \in ProcessesInGroup[g\_j] :$
$p\_j \in Processes \wedge$ *Algorithm*!*WasDelivered*(*g_j*, *p_j*, *m*)