MODULE *Integrity*

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

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

LOCAL $Processes \triangleq 1 \,..\, NPROCESSES$
LOCAL $Groups \triangleq 1 \,..\, NGROUPS$
LOCAL $ProcessesInGroup \triangleq [g \in Groups \mapsto Processes]$

This property verifies that we only deliver sent messages. To assert this, we create *NMESSAGES*+ 1 and do not include the additional one in the algorithm execution, then check that the delivered ones are only the sent ones.

LOCAL $AcceptableMessageIds \triangleq \{id : id \in 1 \,..\, NMESSAGES\}$
LOCAL $AllMessages \triangleq CreateMessages(NMESSAGES + 1,\ Groups,\ Processes)$
LOCAL $SentMessage \triangleq \{m \in AllMessages : m.id \in AcceptableMessageIds\}$

LOCAL $MessagesCombinations \triangleq CreatePossibleMessages(AllMessages)$
LOCAL $CombinationsToSend \triangleq [i \in \text{DOMAIN } MessagesCombinations \mapsto$
$SelectSeq(MessagesCombinations[i],\ \text{LAMBDA } m : m \in SentMessage)]$

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
$INITIAL\_MESSAGES \leftarrow [g \in Groups \mapsto$
$TotallyOrdered(CombinationsToSend[(g\%NMESSAGES) + 1])]$

$Spec \triangleq Algorithm!SpecFair$ Weak fairness is necessary.

LOCAL $DeliveredOnlyOnce(g,\ p,\ m) \triangleq$
$Cardinality(Algorithm!FilterDeliveredMessages(g,\ p,\ m)) = 1$

For every message, all the correct processes in the destination deliver it only once, and a process previously sent it.

$Integrity \triangleq$
$\Diamond\Box \forall\, m \in AllMessages :$
$\forall\, g \in m.d :$
$\forall\, p \in ProcessesInGroup[g] :$
$(p \in Processes \land DeliveredOnlyOnce(g,\ p,\ m)) \equiv m \in SentMessage$