MODULE *WhiteBoxMulticast*

The specification of the White-Box *Multicast* protocol for atomic multicast; see Section *IV* of the *DSN*2019 paper "White-Box Atomic *Multicast*" by *Alexey Gotsman*, *Anatole Lefort*, and *Gregory Chockler*.

Note that this version omits the recovery mechanism of leaders. Therefore, this spec. does not involve any "ballots". We leave it to the future work.

EXTENDS *Naturals*, *Sequences*, *FiniteSets*, *TLC*

$Injective(f) \triangleq \forall\, a, b \in \text{DOMAIN } f : (a \neq b) \Rightarrow (f[a] \neq f[b])$

$Max(a, b) \triangleq \text{IF } a > b \text{ THEN } a \text{ ELSE } b$

CONSTANTS

- Msg, the set of messages, ranged over by m
- $Proc$, the set of processes, ranged over by p
- $Group$, the set of groups, ranged over by g
- $Leader$, $Leader[g] \in Proc$: the leader of the group $g \in Group$
- $Member$, $Member[g] \subseteq Proc$: the members of the group $g \in Group$
- $Dest$ $Dest[m] \subseteq Group$: the set of destination groups of $m \in Msg$

ASSUME

$\land\, Leader \in [Group \to Proc]$
$\land\, Member \in [Group \to \text{SUBSET } Proc]$
$\land\, \forall\, g \in Group : Leader[g] \in Member[g]$
$\land\, \forall\, g1, g2 \in Group : Member[g1] \cap Member[g2] = \{\}$
$\land\, Dest \in [Msg \to \text{SUBSET } Group]$

$Priority \triangleq \text{CHOOSE } f \in [Group \to 1 \,..\, Cardinality(Group)] : Injective(f)$

VARIABLES

- $clock$, $clock[p]$: the clock at process $p \in Proc$
- $phase$, $phase[p][m]$: the phase of the message $m \in Msg$ at process $p \in Proc$
- $localTS$, $localTS[p][m]$: the local ts of the message $m \in Msg$ at process $p \in Proc$
- $globalTS$, $globalTS[p][m]$: the global ts of the message $m \in Msg$ at process $p \in Proc$
- $delivered$, $delivered[p][m]$: has $m \in Msg$ been delivered at process $p \in Proc$?
- $incoming$, $incoming[p] \subseteq Message$ (defined below): the incoming channel of process $p \in Proc$
- $sent$ $sent \subseteq Msg$: the set of messages that have been multicast; only for TLC

$pvars \triangleq \langle clock, phase, localTS, globalTS, delivered \rangle$
$vars \triangleq \langle clock, phase, localTS, globalTS, delivered, incoming, sent \rangle$

$MaxCounter \triangleq Cardinality(Msg) * Cardinality(Group)$

$TS \triangleq [c : 0 \,..\, MaxCounter, p : Group]$ c for counter

$GT(u, v) \triangleq$ Is $u > v$?
$\quad \lor\, u.c > v.c$

$$\lor \land u.c = v.c$$
$$\land Priority[u.p] > Priority[v.p]$$

$$MaxV(vs) \triangleq \text{CHOOSE } u \in vs : \forall\, v \in vs : u \neq v \Rightarrow GT(u, v)$$

TODO: type literals as CONSTANTS

$$Message \triangleq [type : \{\text{“MULTICAST”}\},\ m : Msg]$$
$$\cup\, [type : \{\text{“ACCEPT”}\},\ m : Msg,\ g : Group,\ lts : TS]$$
$$\cup\, [type : \{\text{“ACCEPTACK”}\},\ m : Msg,\ g : Proc]$$
$$\cup\, [type : \{\text{“DELIVER”}\},\ m : Msg,\ lts\ \ : TS,\ gts : TS]$$

$$Send(msg) \triangleq$$ Send $msg \in Message$ to the leaders of its destination groups

$$incoming' = [p \in Proc \mapsto$$
$$\text{IF } p \in \{Leader[g] : g \in Dest[msg.m]\} \text{ THEN } incoming[p] \cup \{msg\}$$
$$\text{ELSE } incoming[p]]$$

TODO: to revise it.

Send $smsg \in Message$ to its destination processes and remove $rmsg \in Message$ from $incoming[sender]$

Precondition: $sender \in Dest[msg.m]$

$$SendAndRemove(smsg,\ sender,\ rmsg) \triangleq$$
$$incoming' = [p \in Proc \mapsto$$
$$\text{IF } p = sender \text{ THEN } (incoming[sender] \cup \{smsg\}) \setminus \{rmsg\}$$
$$\text{ELSE IF } p \in Dest[smsg.m] \text{ THEN } incoming[p] \cup \{smsg\}$$
$$\text{ELSE } incoming[p]]$$

$$TypeOK \triangleq$$
$$\land\ clock \in [Proc \to 0\,..\,MaxCounter]$$
$$\land\ phase \in [Proc \to [Msg \to \{\text{“START”},\ \text{“PROPOSED”},\ \text{“ACCEPTED”},\ \text{“COMMITTED”}\}]]$$
$$\land\ localTS \in [Proc \to [Msg \to TS]]$$
$$\land\ globalTS \in [Proc \to [Msg \to TS]]$$
$$\land\ delivered \in [Proc \to [Msg \to \text{BOOLEAN }]]$$
$$\land\ incoming \in [Proc \to \text{SUBSET } Message]$$
$$\land\ sent \subseteq Msg$$

$$Init \triangleq$$
$$\land\ clock = [p \in Proc \mapsto 0]$$
$$\land\ phase = [p \in Proc \mapsto [m \in Msg \mapsto \text{“START”}]]$$
$$\land\ localTS = [p \in Proc \mapsto [m \in Msg \mapsto [c \mapsto 0,\ p \mapsto p]]]$$
$$\land\ globalTS = [p \in Proc \mapsto [m \in Msg \mapsto [c \mapsto 0,\ p \mapsto p]]]$$
$$\land\ delivered = [p \in Proc \mapsto [m \in Msg \mapsto \text{FALSE}]]$$
$$\land\ incoming = [p \in Proc \mapsto \{\}]$$
$$\land\ sent = \{\}$$

$$multicast(m) \triangleq$$ multicast $m \in Msg$

```
102         ∧ m ∈ Msg \ sent
103         ∧ sent' = sent ∪ {m}
104         ∧ Send([type ↦ "MULTICAST", m ↦ m])
105         ∧ UNCHANGED pvars

107   Multicast ≜ FALSE

109   Propose(p) ≜ \* When p ∈ Proc receives a MULTICAST for some m ∈ Msg
110     ∃ msg ∈ incoming[p]:
111       ∧ msg.type = "MULTICAST"
112       ∧ LET m ≜ msg.m
113         IN  ∧ Assert(p ∈ Dest[m], "p should be one of the destination process of m")
114             ∧ clock' = [clock EXCEPT ![p] = @ + 1]
115             ∧ localTS' = [localTS EXCEPT ![p][m] = [c ↦ clock'[p], p ↦ p]]
116             ∧ phase' = [phase EXCEPT ![p][m] = "PROPOSED"]
117             ∧ SendAndRemove([type ↦ "PROPOSE", m ↦ m, p ↦ p,
118                              lts ↦ localTS'[p][m]], p, msg)
119             ∧ UNCHANGED ⟨globalTS, delivered, sent⟩
120 ═══════════════════════════════════════════════════════════════
      \* Modification History
      \* Last modified Fri Jul 30 23:05:23 CST 2021 by hengxin
      \* Created Fri Jul 30 19:56:43 CST 2021 by hengxin
```