1 ─────────────────────────────── MODULE *AJupiter* ───────────────────────────────

Specification of the *Jupiter* protocol presented by *Attiya* et al.

5 EXTENDS *JupiterInterface*, *OT*, *BufferStateSpace*

6 ├──

7 VARIABLES

8   *cbuf*,   *cbuf*[*c*]: buffer for locally generated operations at client $c \in Client$

9   *crec*,   *crec*[*c*]: number of remote operations received by client $c \in Client$

10   since the last time a local operation was generated

11   *sbuf*,   *sbuf*[*c*]: buffer for transformed remote operations $w.r.t$ client $c \in Client$

12   *srec*   *srec*[*c*]: number of locally generated operations by client $c \in Client$

13   since the last time a remote operation was transformed at the *Server*

15 $vars \triangleq \langle intVars, cbuf, crec, sbuf, srec \rangle$

17 $AJMsg \triangleq$

18     $[c : Client, ack : Nat, op : Op \cup \{Nop\}] \cup$ messages sent to the *Server* from client $c \in Client$

19     $[ack : Nat, op : Op \cup \{Nop\}]$ messages broadcast to Clients from the *Server*

20 ├──

21 $TypeOK \triangleq$

22     $\land$ *TypeOKInt*

23     $\land$ $cbuf \in [Client \rightarrow Seq(Op \cup \{Nop\})]$

24     $\land$ $crec \in [Client \rightarrow Nat]$

25     $\land$ $sbuf \in [Client \rightarrow Seq(Op \cup \{Nop\})]$

26     $\land$ $srec \in [Client \rightarrow Nat]$

27 ├──

28 $Init \triangleq$

29     $\land InitInt$

30     $\land cbuf = [c \in Client \mapsto \langle \rangle]$

31     $\land crec = [c \in Client \mapsto 0]$

32     $\land sbuf = [c \in Client \mapsto \langle \rangle]$

33     $\land srec = [c \in Client \mapsto 0]$

34 ├──

35 $ClientPerform(c, m) \triangleq$

36   LET $xform \triangleq xFormShift(OT, m.op, cbuf[c], m.ack)$ [*xop, xops*]

37   IN $\land cbuf' = [cbuf \text{ EXCEPT } ![c] = xform.xops]$

38           $\land crec' = [crec \text{ EXCEPT } ![c] = @ + 1]$

39           $\land SetNewAop(c, xform.xop)$

41 $ServerPerform(m) \triangleq$

42   LET $c \triangleq m.c$

43           $xform \triangleq xFormShift(OT, m.op, sbuf[c], m.ack)$ [*xop, xops*]

44            $xop \triangleq xform.xop$

45   IN $\land srec' = [cl \in Client \mapsto \text{IF } cl = c \text{ THEN } srec[cl] + 1 \text{ ELSE } 0]$

46           $\land sbuf' = [cl \in Client \mapsto \text{IF } cl = c \text{ THEN } xform.xops$

47                                $\text{ELSE } Append(sbuf[cl], xop)]$

1

```
48                  ∧ SetNewAop(Server, xop)
49                  ∧ Comm!SSend(c, [cl ∈ Client ↦ [ack ↦ srec[cl], op ↦ xop]])
```

```
51  DoOp(c, op) ≜
52          ∧ SetNewAop(c, op)
53          ∧ cbuf′ = [cbuf EXCEPT ![c] = Append(@, op)]
54          ∧ crec′ = [crec EXCEPT ![c] = 0]
55          ∧ Comm!CSend([c ↦ c, ack ↦ crec[c], op ↦ op])

57  Do(c) ≜
58          ∧ DoInt(DoOp, c)
59          ∧ UNCHANGED ⟨sbuf, srec⟩

61  Rev(c) ≜
62          ∧ RevInt(ClientPerform, c)
63          ∧ UNCHANGED ⟨sbuf, srec⟩

65  SRev ≜
66          ∧ SRevInt(ServerPerform)
67          ∧ UNCHANGED ⟨cbuf, crec⟩
```

```
69  Next ≜
70          ∨ ∃ c ∈ Client : Do(c) ∨ Rev(c)
71          ∨ SRev

73  Fairness ≜
74          WF_vars(SRev ∨ ∃ c ∈ Client : Rev(c))

76  Spec ≜ Init ∧ □[Next]_vars  ∧ Fairness
```

```
78  QC ≜   Quiescent Consistency
79          Comm!EmptyChannel ⇒ Cardinality(Range(state)) = 1

81  THEOREM Spec ⇒ □QC
```
---

* Modification History
* Last modified Thu Jan 17 10:30:39 CST 2019 by hengxin
* Created Satchins, Jun 23 17:14:18 CST 2018 by hengxin