

```

1  |----- MODULE AJupiter -----|
   | Specification of the Jupiter protocol presented by Hagit Attiya and others. |
5  | EXTENDS JupiterInterface |
6  |-----|
7  VARIABLES
8      cbuf,      cbuf[c]: buffer for locally generated operations at client c ∈ Client
9      crec,      crec[c]: number of remote operations received by client c ∈ 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 ∈ Client
12      srec      srec[c]: number of locally generated operations by client c ∈ Client
13                     since the last time a remote operation was transformed at the Server
15  vars ≜ ⟨intVars, cbuf, crec, sbuf, srec⟩
17  Msg ≜ [c : Client, ack : Int, op : Op ∪ {Nop}] ∪ messages sent to the Server from a client c ∈ Client
18          [ack : Int, op : Op ∪ {Nop}] messages broadcast to Clients from the Server
19  |-----|
20  TypeOK ≜
21      ∧ TypeOKInt
22      ∧ Comm(Msg)!TypeOK
23      ∧ cbuf ∈ [Client → Seq(Op ∪ {Nop})]
24      ∧ crec ∈ [Client → Int]
25      ∧ sbuf ∈ [Client → Seq(Op ∪ {Nop})]
26      ∧ srec ∈ [Client → Int]
27  |-----|
28  Init ≜
29      ∧ InitInt
30      ∧ Comm(Msg)!Init
31      ∧ cbuf = [c ∈ Client ↦ ⟨⟩]
32      ∧ crec = [c ∈ Client ↦ 0]
33      ∧ sbuf = [c ∈ Client ↦ ⟨⟩]
34      ∧ srec = [c ∈ Client ↦ 0]
35  |-----|
   | Client c ∈ Client issues an operation op. |
39  DoOp(c, op) ≜
40      ∧ state' = [state EXCEPT ![c] = Apply(op, @)]
41      ∧ cbuf' = [cbuf EXCEPT ![c] = Append(@, op)]
42      ∧ crec' = [crec EXCEPT ![c] = 0]
43      ∧ Comm(Msg)!CSend([c ↦ c, ack ↦ crec[c], op ↦ op])
45  DoIns(c) ≜
46      ∃ ins ∈ {op ∈ Ins : op.pos ∈ 1 .. (Len(state[c] + 1) ∧ op.ch ∈ chins ∧ op.pr = Priority[c]} :
47          ∧ DoOp(c, ins)
48          ∧ chins' = chins \ {ins.ch}
50  DoDel(c) ≜

```

```

51    $\exists del \in \{op \in Del : op.pos \in 1 \dots Len(state[c])\} :$ 
52      $\wedge DoOp(c, del)$ 
53      $\wedge UNCHANGED \ chins$ 
54
55    $Do(c) \triangleq$ 
56      $\wedge \vee DoIns(c)$ 
57      $\vee DoDel(c)$ 
58      $\wedge UNCHANGED \langle sbuf, srec \rangle$ 
59 |-----|
    Client  $c \in Client$  receives a message from the Server.
60
61    $Rev(c) \triangleq$ 
62      $\wedge Comm(Msg)!CRev(c)$ 
63      $\wedge crec' = [crec \text{ EXCEPT } ![c] = @ + 1]$ 
64      $\wedge LET \ m \triangleq Head(cincoming[c])$ 
65        $cBuf \triangleq cbuf[c]$ 
66        $cShiftedBuf \triangleq SubSeq(cBuf, m.ack + 1, Len(cBuf))$ 
67        $xop \triangleq XformOpOps(Xform, m.op, cShiftedBuf)$ 
68        $xcBuf \triangleq XformOpsOp(Xform, cShiftedBuf, m.op)$ 
69       IN  $\wedge cbuf' = [cbuf \text{ EXCEPT } ![c] = xcBuf]$ 
70          $\wedge state' = [state \text{ EXCEPT } ![c] = Apply(xop, @)]$ 
71          $\wedge UNCHANGED \langle chins, sbuf, srec \rangle$ 
72
73 |-----|
    The Server receives a message.
74
75    $SRev \triangleq$ 
76      $\wedge Comm(Msg)!SRev$ 
77      $\wedge LET \ m \triangleq Head(sincoming)$ 
78        $c \triangleq m.c$ 
79        $cBuf \triangleq sbuf[c]$ 
80        $cShiftedBuf \triangleq SubSeq(cBuf, m.ack + 1, Len(cBuf))$ 
81        $xop \triangleq XformOpOps(Xform, m.op, cShiftedBuf)$ 
82        $xcBuf \triangleq XformOpsOp(Xform, cShiftedBuf, m.op)$ 
83       IN  $\wedge srec' = [cl \in Client \mapsto$ 
84         IF  $cl = c$ 
85         THEN  $srec[cl] + 1$ 
86         ELSE  $0]$ 
87        $\wedge sbuf' = [cl \in Client \mapsto$ 
88         IF  $cl = c$ 
89         THEN  $xcBuf$ 
90         ELSE  $Append(sbuf[cl], xop)]$ 
91        $\wedge state' = [state \text{ EXCEPT } ![Server] = Apply(xop, @)]$ 
92        $\wedge Comm(Msg)!SSend(c, [cl \in Client \mapsto [ack \mapsto srec[cl], op \mapsto xop]])$ 
93        $\wedge UNCHANGED \langle chins, cbuf, crec \rangle$ 
94
95 |-----|
    Next  $\triangleq$ 
96    $\vee \exists c \in Client : Do(c) \vee Rev(c)$ 

```

100       $\vee SRev$

102     $Fairness \triangleq$     There is no requirement that the clients ever generate operations.

103       $WF_{vars}(SRev \vee \exists c \in Client : Rev(c))$

105     $Spec \triangleq Init \wedge \Box[Next]_{vars} \wedge Fairness$

106 |-----|

107     $QC \triangleq$     Quiescent Consistency

108       $Comm(Msg)!EmptyChannel \Rightarrow Cardinality(Range(state)) = 1$

110 THEOREM  $Spec \Rightarrow \Box QC$

111 |-----|

\ \* Modification History

\ \* Last modified Sat Dec 29 17:25:40 CST 2018 by hengxin

\ \* Created Sat Jun 23 17:14:18 CST 2018 by hengxin