

```

1  ┌────────────────── MODULE AJupiter ───────────────────┐
    Specification of the Jupiter protocol presented by Hagit Attiya and others.
5  EXTENDS JupiterInterface
6  ┌──────────────────┐
    Messages between the Server and the Clients.
10 Msg  $\triangleq$   $[c : \textit{Client}, \textit{ack} : \textit{Int}, \textit{op} : \textit{Op} \cup \{\textit{Nop}\}] \cup$  messages sent to the Server from a client  $c \in \textit{Client}$ 
11  $[\textit{ack} : \textit{Int}, \textit{op} : \textit{Op} \cup \{\textit{Nop}\}]$  messages broadcast to Clients from the Server
12 └──────────────────┘
13 VARIABLES
14   cbuf, cbuf[c]: buffer (of operations) at the client  $c \in \textit{Client}$ 
15   crec, crec[c]: the number of new messages have been received by the client  $c \in \textit{Client}$ 
16               since the last time a message was sent
17   sbuf, sbuf[c]: buffer (of operations) at the Server, one per client  $c \in \textit{Client}$ 
18   srec srec[c]: the number of new messages have been ... , one per client  $c \in \textit{Client}$ 
20 vars  $\triangleq$   $\langle \textit{chins}, \textit{cbuf}, \textit{crec}, \textit{sbuf}, \textit{srec}, \textit{cincoming}, \textit{sincoming}, \textit{state} \rangle$ 
21 └──────────────────┘
22 TypeOK  $\triangleq$ 
23    $\wedge$  TypeOKInt
24    $\wedge$  Comm(Msg)! TypeOK
25    $\wedge$  cbuf  $\in [\textit{Client} \rightarrow \textit{Seq}(\textit{Op} \cup \{\textit{Nop}\})]$ 
26    $\wedge$  crec  $\in [\textit{Client} \rightarrow \textit{Int}]$ 
27    $\wedge$  sbuf  $\in [\textit{Client} \rightarrow \textit{Seq}(\textit{Op} \cup \{\textit{Nop}\})]$ 
28    $\wedge$  srec  $\in [\textit{Client} \rightarrow \textit{Int}]$ 
29 └──────────────────┘
30 Init  $\triangleq$ 
31    $\wedge$  InitInt
32    $\wedge$  Comm(Msg)! Init
33    $\wedge$  cbuf  $= [c \in \textit{Client} \mapsto \langle \rangle]$ 
34    $\wedge$  crec  $= [c \in \textit{Client} \mapsto 0]$ 
35    $\wedge$  sbuf  $= [c \in \textit{Client} \mapsto \langle \rangle]$ 
36    $\wedge$  srec  $= [c \in \textit{Client} \mapsto 0]$ 
37 └──────────────────┘
    Client  $c \in \textit{Client}$  issues an operation op.
41 DoOp(c, op)  $\triangleq$ 
42    $\wedge$  state'  $= [\textit{state} \text{ EXCEPT } ![c] = \textit{Apply}(\textit{op}, @)]$ 
43    $\wedge$  cbuf'  $= [\textit{cbuf} \text{ EXCEPT } ![c] = \textit{Append}(@, \textit{op})]$ 
44    $\wedge$  crec'  $= [\textit{crec} \text{ EXCEPT } ![c] = 0]$ 
45    $\wedge$  Comm(Msg)! CSend( $[c \mapsto c, \textit{ack} \mapsto \textit{crec}[c], \textit{op} \mapsto \textit{op}]$ )
47 DoIns(c)  $\triangleq$ 
48    $\exists \textit{ins} \in \{\textit{op} \in \textit{Ins} : \textit{op.pos} \in 1 \dots (\textit{Len}(\textit{state}[c]) + 1) \wedge \textit{op.ch} \in \textit{chins} \wedge \textit{op.pr} = \textit{Priority}[c]\} :$ 
49    $\wedge$  DoOp(c, ins)
50    $\wedge$  chins'  $= \textit{chins} \setminus \{\textit{ins.ch}\}$  We assume that all inserted elements are unique.
51    $\wedge$  UNCHANGED  $\langle \textit{sbuf}, \textit{srec} \rangle$ 

```

```

53  $DoDel(c) \triangleq$ 
54    $\exists del \in \{op \in Del : op.pos \in 1 \dots Len(state[c])\} :$ 
55      $\wedge DoOp(c, del)$ 
56      $\wedge UNCHANGED \langle chins, sbuf, srec \rangle$ 

58  $Do(c) \triangleq$ 
59    $\vee DoIns(c)$ 
60    $\vee DoDel(c)$ 

Client  $c \in Client$  receives a message from the Server.

64  $Rev(c) \triangleq$ 
65    $\wedge Comm(Msg)!CRev(c)$ 
66    $\wedge crec' = [crec \text{ EXCEPT } ![c] = @ + 1]$ 
67    $\wedge LET \ m \triangleq Head(cincoming[c])$ 
68      $cBuf \triangleq cbuf[c]$  the buffer at client  $c \in Client$ 
69      $cShiftedBuf \triangleq SubSeq(cBuf, m.ack + 1, Len(cBuf))$  buffer shifted
70      $xop \triangleq XformOpOps(m.op, cShiftedBuf)$  transform  $op$  vs. shifted buffer
71      $xcBuf \triangleq XformOpsOp(cShiftedBuf, m.op)$  transform shifted buffer vs.  $op$ 
72   IN    $\wedge cbuf' = [cbuf \text{ EXCEPT } ![c] = xcBuf]$ 
73        $\wedge state' = [state \text{ EXCEPT } ![c] = Apply(xop, @)]$  apply the transformed operation  $xop$ 
74    $\wedge UNCHANGED \langle chins, sbuf, srec \rangle$ 

The Server receives a message.

78  $SRev \triangleq$ 
79    $\wedge Comm(Msg)!SRev$ 
80    $\wedge LET \ m \triangleq Head(sincoming)$  the message to handle with
81      $c \triangleq m.c$  the client  $c \in Client$  that sends this message
82      $cBuf \triangleq sbuf[c]$  the buffer at the Server for client  $c \in Client$ 
83      $cShiftedBuf \triangleq SubSeq(cBuf, m.ack + 1, Len(cBuf))$  buffer shifted
84      $xop \triangleq XformOpOps(m.op, cShiftedBuf)$  transform  $op$  vs. shifted buffer
85      $xcBuf \triangleq XformOpsOp(cShiftedBuf, m.op)$  transform shifted buffer vs.  $op$ 
86   IN    $\wedge srec' = [cl \in Client \mapsto$ 
87       IF  $cl = c$ 
88       THEN  $srec[cl] + 1$  receive one more operation from client  $c \in Client$ 
89       ELSE  $0$ ] reset  $srec$  for other clients than  $c \in Client$ 
90    $\wedge sbuf' = [cl \in Client \mapsto$ 
91       IF  $cl = c$ 
92       THEN  $xcBuf$  transformed buffer for client  $c \in Client$ 
93       ELSE  $Append(sbuf[cl], xop)$  store transformed  $xop$  into other clients' bufs
94    $\wedge state' = [state \text{ EXCEPT } ![Server] = Apply(xop, @)]$  apply the transformed operation
95    $\wedge Comm(Msg)!SSend(c, [cl \in Client \mapsto [ack \mapsto srec[cl], op \mapsto xop]])$ 
96    $\wedge UNCHANGED \langle chins, cbuf, crec \rangle$ 

97 |-----|
98  $Next \triangleq$ 
99    $\vee \exists c \in Client : Do(c) \vee Rev(c)$ 
100   $\vee SRev$ 

Fairness: There is no requirement that the clients ever generate operations.

```

```

104 Fairness  $\triangleq$ 
105      $\text{WF}_{vars}(SRev \vee \exists c \in Client : Rev(c))$ 
107 Spec  $\triangleq$  Init  $\wedge \Box[Next]_{vars}$   $\wedge$  Fairness
108 ┌──────────────────────────────────────────────────────────────────────────────────┐
109 │ Quiescent Consistency (QC) ───────────────────────────────────────────────────┘
112 QC  $\triangleq$ 
113      $Comm(Msg)!EmptyChannel \Rightarrow Cardinality(Range(state)) = 1$ 
115 THEOREM Spec  $\Rightarrow \Box QC$ 
116 └──────────────────────────────────────────────────────────────────────────────────┘
    \ * Modification History
    \ * Last modified Tue Dec 04 21:10:51 CST 2018 by hengxin
    \ * Created Sat Jun 23 17:14:18 CST 2018 by hengxin

```