```
──────────────────── MODULE AJupiter ────────────────────
Specification of the Jupiter protocol presented by Attiya et al.
EXTENDS JupiterInterface, OT, BufferStateSpace
─────────────────────────────────────────────────────────
VARIABLES
    cbuf,     cbuf[c]: buffer for locally generated operations at client c ∈ Client
    crec,     crec[c]: number of remote operations received by client c ∈ Client
                       since the last time a local operation was generated
    sbuf,     sbuf[c]: buffer for transformed remote operations w.r.t client c ∈ Client
    srec      srec[c]: number of locally generated operations by client c ∈ Client
                       since the last time a remote operation was transformed at the Server

vars ≜ ⟨intVars, cbuf, crec, sbuf, srec⟩

AJMsg ≜
    [c : Client, ack : Nat, op : Op ∪ {Nop}] ∪   messages sent to the Server from client c ∈ Client
    [ack : Nat, op : Op ∪ {Nop}]   messages broadcast to Clients from the Server
─────────────────────────────────────────────────────────
TypeOK ≜
    ∧  TypeOKInt
    ∧  cbuf ∈ [Client → Seq(Op ∪ {Nop})]
    ∧  crec ∈ [Client → Nat]
    ∧  sbuf ∈ [Client → Seq(Op ∪ {Nop})]
    ∧  srec ∈ [Client → Nat]
─────────────────────────────────────────────────────────
Init ≜
    ∧ InitInt
    ∧ cbuf = [c ∈ Client ↦ ⟨⟩]
    ∧ crec = [c ∈ Client ↦ 0]
    ∧ sbuf = [c ∈ Client ↦ ⟨⟩]
    ∧ srec = [c ∈ Client ↦ 0]
─────────────────────────────────────────────────────────
ClientPerform(c, m) ≜
    LET xform ≜ xFormShift(OT, m.op, cbuf[c], m.ack)   [xop, xops]
    IN   ∧ cbuf′ = [cbuf EXCEPT ![c] = xform.xops]
         ∧ crec′ = [crec EXCEPT ![c] = @ + 1]
         ∧ SetNewAop(c, xform.xop)

ServerPerform(m) ≜
    LET      c  ≜ m.c
         xform  ≜ xFormShift(OT, m.op, sbuf[c], m.ack)  [xop, xops]
           xop  ≜ xform.xop
    IN    ∧ srec′ = [cl ∈ Client ↦ IF cl = c THEN srec[cl] + 1 ELSE 0]
          ∧ sbuf′ = [cl ∈ Client ↦ IF cl = c THEN xform.xops
                                              ELSE Append(sbuf[cl], xop)]
```

$\quad\quad \land\ SetNewAop(Server,\ xop)$
$\quad\quad \land\ Comm!SSend(c,\ [cl \in Client \mapsto [ack \mapsto srec[cl],\ op \mapsto xop]])$

$DoOp(c,\ op)\ \triangleq$
$\quad \land\ SetNewAop(c,\ op)$
$\quad \land\ cbuf' = [cbuf\ \text{EXCEPT}\ ![c] = Append(@,\ op)]$
$\quad \land\ crec' = [crec\ \text{EXCEPT}\ ![c] = 0]$
$\quad \land\ Comm!CSend([c \mapsto c,\ ack \mapsto crec[c],\ op \mapsto op])$

$Do(c)\ \triangleq$
$\quad \land\ DoInt(DoOp,\ c)$
$\quad \land\ \text{UNCHANGED}\ \langle sbuf,\ srec\rangle$

$Rev(c)\ \triangleq$
$\quad \land\ RevInt(ClientPerform,\ c)$
$\quad \land\ \text{UNCHANGED}\ \langle sbuf,\ srec\rangle$

$SRev\ \triangleq$
$\quad \land\ SRevInt(ServerPerform)$
$\quad \land\ \text{UNCHANGED}\ \langle cbuf,\ crec\rangle$

$Next\ \triangleq$
$\quad \lor\ \exists\, c \in Client : Do(c) \lor Rev(c)$
$\quad \lor\ SRev$

$Fairness\ \triangleq$
$\quad \mathrm{WF}_{vars}(SRev \lor \exists\, c \in Client : Rev(c))$

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

$QC\ \triangleq$ Quiescent Consistency
$\quad Comm!EmptyChannel \Rightarrow Cardinality(Range(state)) = 1$

THEOREM $Spec \Rightarrow \Box QC$

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