MODULE *NJupiterExtended*

NJupiter extended with *JupiterCtx*. Used to show *NJupiter* implements *XJupiter*.

EXTENDS *JupiterCtx*, *BufferStateSpace*

VARIABLES *cbuf*, *cack*, *sbuf*, *sack*, *cincomingXJ*, *sincomingXJ*

variable name cseq has been used in *JupiterCtx*, we use *cack* instead.

$$varsEx \triangleq \langle intVars,\ ctxVars,\ cbuf,\ cack,\ sbuf,\ sack,\ cincomingXJ,\ sincomingXJ\rangle$$

$$NMsgEx \triangleq [ack : Nat,\ cop : Cop,\ oid : Oid]$$

$$commXJ \triangleq \text{INSTANCE } CSComm \text{ WITH } Msg \leftarrow Cop,\ cincoming \leftarrow cincomingXJ,\ sincoming \leftarrow sincomingXJ$$

$$TypeOKEx \triangleq \land TypeOKInt \land TypeOKCtx \land commXJ!TypeOK \land cack \in [Client \to [l : Nat,\ g : Nat]] \land sack \in [Client \to [l : Nat,\ g : Nat]] \land cbuf \in [Client \to Seq(Cop)] \land sbuf \in [Client \to Seq(Cop)]$$

$$InitEx \triangleq \land InitInt \land InitCtx \land commXJ!Init \land cack = [c \in Client \mapsto [l \mapsto 0,\ g \mapsto 0]] \land sack = [c \in Client \mapsto [l \mapsto 0,\ g \mapsto 0]] \land cbuf = [c \in Client \mapsto \langle\rangle] \land sbuf = [c \in Client \mapsto \langle\rangle]$$

$$DoOpEx(c,\ op) \triangleq \text{LET } cop \triangleq [op \mapsto op,\ oid \mapsto [c \mapsto c,\ seq \mapsto cack[c].l + 1],\ ctx \mapsto ds[c]] \text{ IN } \land cack' = [cack \text{ EXCEPT } ![c] = [l \mapsto @.l + 1,\ g \mapsto @.g]] \land cbuf' = [cbuf \text{ EXCEPT } ![c] = Append(@,\ cop)] \land SetNewAop(c,\ op) \land Comm!CSend([ack \mapsto cack[c].g,\ cop \mapsto cop,\ oid \mapsto cop.oid]) \land commXJ!CSend(cop)$$

$$RemoveAckedOps(s,\ ack) \triangleq \text{LET } F[i \in 0\,..\,Len(s)] \triangleq \text{IF } i = 0 \text{ THEN } \langle\rangle \text{ ELSE IF } s[i].oid.seq > ack$$

```
                THEN Append(F[i − 1], s[i])
                ELSE F[i − 1]
    IN  F[Len(s)]

ClientPerformEx(c, m) ≜
    LET xform ≜ xFormFull(COT, m.cop, RemoveAckedOps(cbuf[c], m.ack))
    IN  ∧ cbuf′ = [cbuf EXCEPT ![c] = xform.xops]
        ∧ cack′ = [cack EXCEPT ![c] = [l ↦ @.l, g ↦ @.g + 1]]
        ∧ SetNewAop(c, xform.xop.op)

ServerPerformEx(m) ≜
    LET c ≜ ClientOf(m.cop)
        xform ≜ xFormFull(COT, m.cop, RemoveAckedOps(sbuf[c], m.ack))
        xcop ≜ xform.xop
    IN  ∧ sack′ = [cl ∈ Client ↦ IF cl = c
                                 THEN [l ↦ sack[cl].l + 1, g ↦ sack[cl].g]
                                 ELSE [l ↦ sack[cl].l, g ↦ sack[cl].g + 1]]
        ∧ sbuf′ = [cl ∈ Client ↦ IF cl = c
                                 THEN xform.xops
                                 ELSE Append(sbuf[cl], xcop)]
        ∧ SetNewAop(Server, xcop.op)
        ∧ Comm!SSend(c, [cl ∈ Client ↦ [ack ↦ sack[cl].l,
                                        cop ↦ xcop,
                                        oid ↦ xcop.oid]])
        ∧ commXJ!SSendSame(c, xcop)
```

```
DoEx(c) ≜
    ∧ DoInt(DoOpEx, c)
    ∧ DoCtx(c)
    ∧ UNCHANGED ⟨sbuf, sack⟩

RevEx(c) ≜
    ∧ RevInt(ClientPerformEx, c)
    ∧ RevCtx(c)
    ∧ commXJ!CRev(c)
    ∧ UNCHANGED ⟨sbuf, sack⟩

SRevEx ≜
    ∧  SRevInt(ServerPerformEx)
    ∧  SRevCtx
    ∧  commXJ!SRev
    ∧  UNCHANGED ⟨cbuf, cack⟩
```

```
NextEx ≜
    ∨  ∃ c ∈ Client : DoEx(c) ∨ RevEx(c)
    ∨  SRevEx
```

$FairnessEx \triangleq$
$\quad \mathrm{WF}_{varsEx}(SRevEx \lor \exists\, c \in Client : RevEx(c))$

$SpecEx \triangleq InitEx \land \Box[NextEx]_{varsEx}$ $\land FairnessEx$

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

THEOREM $SpecEx \Rightarrow \Box QC$

---

* Modification History
* Last modified Sun *Apr* 21 15:44:38 *CST* 2019 by *tangruize*
* Created *Wed Mar* 19 04:49:31 *CST* 2019 by *tangruize*