MODULE *JupiterInterface*

Interface of a family of *Jupiter* protocols.

EXTENDS *SequenceUtils*, *OT*

CONSTANTS

Client, the set of client replicas
Server, the (unique) server replica
Msg, the set of messages
Char, the set of characters
InitState the initial state of each replica

ASSUME We assume that all inserted elements are unique.

$\land \quad Range(InitState) \cap Char = \{\}$ due to the uniqueness requirement

VARIABLES

aop, *op*[*r*]: the actual operation applied at replica $r \in Replica$
state, *state*[*r*]: state (the list content) of replica $r \in Replica$
cincoming, *cincoming*[*c*]: incoming channel at the client $c \in Client$
sincoming, incoming channel at the *Server*
chins a set of chars allowed to insert; this is for model checking

$Comm \triangleq$ INSTANCE *CSComm*

$intVars \triangleq \langle aop, state, cincoming, sincoming, chins \rangle$

$Replica \triangleq Client \cup \{Server\}$

$List \triangleq Seq(Char \cup Range(InitState))$ all possible lists

$MaxLen \triangleq Cardinality(Char) + Len(InitState)$ the max length of lists in any state

$ClientNum \triangleq Cardinality(Client)$

$Priority \triangleq$ CHOOSE $f \in [Client \to 1 \,..\, ClientNum] : Injective(f)$

$Rd \triangleq [type : \{\text{“Rd”}\}]$

$Del \triangleq [type : \{\text{“Del”}\}, pos : 1 \,..\, MaxLen]$ The positions (*pos*) are indexed from 1.

$Ins \triangleq [type : \{\text{“Ins”}\}, pos : 1 \,..\, (MaxLen + 1), ch : Char, pr : 1 \,..\, ClientNum]$ *pr*: priority

$Op \triangleq Ins \cup Del$ The set of all operations (now we don't consider *Rd* operations).

$SetNewAop(r, aopr) \triangleq$
$\quad aop' = [aop \text{ EXCEPT } ![r] = aopr]$

$ApplyNewAop(r) \triangleq$
$\quad state' = [state \text{ EXCEPT } ![r] = Apply(aop'[r], @)]$

$TypeOKInt \triangleq$
$\quad \land\, aop \in [Replica \to Op \cup \{Nop\}]$
$\quad \land\, state \in [Replica \to List]$

```
    ∧ Comm!TypeOK
    ∧ chins ⊆ Char

InitInt ≜
    ∧ aop = [r ∈ Replica ↦ Nop]
    ∧ state = [r ∈ Replica ↦ InitState]
    ∧ Comm!Init
    ∧ chins = Char

DoIns(DoOp(_, _), c) ≜  Client c ∈ Client generates and processes an "Ins" operation.
  ∃ ins ∈ Ins :
    ∧ ins.pos ∈ 1 .. (Len(state[c]) + 1)
    ∧ ins.ch ∈ chins
    ∧ ins.pr = Priority[c]
    ∧ DoOp(c, ins)
    ∧ chins' = chins \ {ins.ch}  We assume that all inserted elements are unique.

DoDel(DoOp(_, _), c) ≜  Client c ∈ Client generates and processes a "Del" operation.
  ∃ del ∈ Del :
    ∧ del.pos ∈ 1 .. Len(state[c])
    ∧ DoOp(c, del)
    ∧ UNCHANGED chins

DoInt(DoOp(_, _), c) ≜  Client c ∈ Client issues an operation.
  ∧ ∨ DoIns(DoOp, c)  DoOp(c ∈ Client, op ∈ Op)
    ∨ DoDel(DoOp, c)
  ∧ ApplyNewAop(c)

RevInt(ClientPerformInt(_, _), c) ≜  Client c ∈ Client receives and processes a message.
    ∧ Comm!CRev(c)
    ∧ ClientPerformInt(c, Head(cincoming[c]))  ClientPerformInt(c ∈ Client, m ∈ Msg)
    ∧ ApplyNewAop(c)
    ∧ UNCHANGED chins

SRevInt(ServerPerformInt(_)) ≜  The Server receives and processes a message.
    ∧  Comm!SRev
    ∧  ServerPerformInt(Head(sincoming))  ServerPerformInt(m ∈ Msg)
    ∧  ApplyNewAop(Server)
    ∧  UNCHANGED chins

\* Modification History
\* Last modified Thu Jan 03 16:39:10 CST 2019 by hengxin
\* Created Tue Dec 04 19:01:01 CST 2018 by hengxin
```