MODULE *JupiterInterface*

This module declares the parameters and defines the operators that describe the interface of a family of *Jupiter* specs.

EXTENDS $Integers$, $SequenceUtils$, OT

CONSTANTS

- $Char$, the set of characters
- $Client$, the set of client replicas
- $Server$, the (unique) server replica
- $InitState$ 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

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

$Comm(Msg) \triangleq$ INSTANCE $CSComm$

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

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

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

$ClientNum \triangleq Cardinality(Client)$

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

The set of all operations. Note: The positions are indexed from 1.

$Rd \triangleq [type : \{\text{"Rd"}\}]$

$Del \triangleq [type : \{\text{"Del"}\}, pos : 1 \,..\, MaxLen]$

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

$Op \triangleq Ins \cup Del$ Now we don't consider Rd operations

$SetNewOp(r, aopr) \triangleq$

$\quad aop' = [aop$ EXCEPT $![r] = aopr]$

$ApplyNewOp(r) \triangleq$

$\quad state' = [state$ EXCEPT $![r] = Apply(aop'[r], @)]$

$TypeOKInt \triangleq$

```
    ∧ aop ∈ [Replica → Op ∪ {Nop}]
    ∧ state ∈ [Replica → List]
    ∧ chins ⊆ Char

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

DoIns(DoOp(_, _), c) ≜   Client c ∈ Client generates 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 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)
      ∨ DoDel(DoOp, c)
    ∧ ApplyNewOp(c)

RevInt(c) ≜   Client c ∈ Client receives a message from the Server.
    ∧ ApplyNewOp(c)
    ∧ UNCHANGED chins

SRevInt ≜   The Server receives a message.
    ∧ ApplyNewOp(Server)
    ∧ UNCHANGED chins
────────────────────────────────────────────────────────────
\* Modification History
\* Last modified Tue Jan 01 11:32:36 CST 2019 by hengxin
\* Created Tue Dec 04 19:01:01 CST 2018 by hengxin
```