
MODULE *XJupiter*

Specification of the *Jupiter* protocol described in *CSCW'2014* by *Xu*, *Sun*, and *Li*.

EXTENDS *GraphStateSpace*

VARIABLES

$c2ss$, $c2ss[c]$: the 2D state space (2ss, for short) at client $c \in Client$

$s2ss$ $s2ss[c]$: the 2D state space maintained by the *Server* for client $c \in Client$

$vars \triangleq \langle intVars, ctxVars, c2ss, s2ss \rangle$

$TypeOK \triangleq$

$\wedge TypeOKInt$

$\wedge TypeOKCtx$

$\wedge \forall c \in Client : IsSS(c2ss[c]) \wedge IsSS(s2ss[c])$

$Init \triangleq$

$\wedge InitInt$

$\wedge InitCtx$

$\wedge c2ss = [c \in Client \mapsto EmptySS]$

$\wedge s2ss = [c \in Client \mapsto EmptySS]$

$NextEdge(r, u, ss) \triangleq$ Return the unique outgoing edge from u in 2D state space ss

CHOOSE $e \in ss.edge : e.from = u$ before a transformation at u (r is not used).

$ClientPerform(c, cop) \triangleq$

LET $xform \triangleq xForm(NextEdge, c, cop, c2ss[c])$ $xform: [xcop, xss, lss]$

IN $\wedge c2ss' = [c2ss \text{ EXCEPT } ![c] = @ \oplus xform.xss]$

$\wedge SetNewAop(c, xform.xcop.op)$

$ServerPerform(cop) \triangleq$

LET $c \triangleq ClientOf(cop)$

$xform \triangleq xForm(NextEdge, Server, cop, s2ss[c])$ $xform: [xcop, xss, lss]$

$xcop \triangleq xform.xcop$

IN $\wedge s2ss' = [cl \in Client \mapsto \text{IF } cl = c$

THEN $s2ss[cl] \oplus xform.xss$

ELSE $s2ss[cl] \oplus xform.lss]$

$\wedge SetNewAop(Server, xcop.op)$

$\wedge Comm!SSendSame(c, xcop)$ broadcast the transformed $xcop$

$DoOp(c, op) \triangleq$

LET $cop \triangleq [op \mapsto op, oid \mapsto [c \mapsto c, seq \mapsto cseq[c]], ctx \mapsto ds[c]]$

IN $\wedge ClientPerform(c, cop)$

$\wedge Comm!CSend(cop)$

$Do(c) \triangleq$

$\wedge DoInt(DoOp, c)$

$$\begin{aligned}
& \wedge DoCtx(c) \\
& \wedge \text{UNCHANGED } s2ss \\
\\
Rev(c) & \triangleq \\
& \wedge RevInt(ClientPerform, c) \\
& \wedge RevCtx(c) \\
& \wedge \text{UNCHANGED } s2ss \\
\\
SRev & \triangleq \\
& \wedge SRevInt(ServerPerform) \\
& \wedge SRevCtx \\
& \wedge \text{UNCHANGED } c2ss \\
\\
\hline
Next & \triangleq \\
& \vee \exists c \in Client : Do(c) \vee Rev(c) \\
& \vee SRev \\
\\
Fairness & \triangleq \\
& WF_{vars}(SRev \vee \exists c \in Client : Rev(c)) \\
\\
Spec & \triangleq Init \wedge \Box[Next]_{vars} \wedge Fairness \\
\\
\hline
CSSync & \triangleq \text{Each client } c \in Client \text{ is synchronized with the Server.} \\
& \forall c \in Client : (ds[c] = ds[Server]) \Rightarrow c2ss[c] = s2ss[c] \\
\\
\text{THEOREM } Spec & \Rightarrow \Box CSSync \\
\\
\hline
\end{aligned}$$

\ * Modification History
\ * Last modified Sat Jan 12 15:57:05 CST 2019 by hengxin
\ * Created Tue Oct 09 16:33:18 CST 2018 by hengxin