

```

1  ┌────────────────── MODULE UniversalPaxosStore ───────────────────┐
    Specification of the consensus protocol in PaxosStore.
    See [PaxosStore@VLDB2017](https://www.vldb.org/pvldb/vol10/p1730-lin.pdf) by Tencent.
    In this version (adopted from "PaxosStore.tla"):
    - Client-restricted config (Ballot)
      - Message types (i.e., "Prepare", "Accept", "ACK") are deleted. No state flags (such as "Prepare", "Wait-Prepare", "Accept", "Wait-Accept" are needed.
14 EXTENDS Integers, FiniteSets
15 ┌────────────────────────────────────────────────────────────────────────┐
16  $Max(m, n) \triangleq \text{IF } m > n \text{ THEN } m \text{ ELSE } n$ 
17  $Injective(f) \triangleq \forall a, b \in \text{DOMAIN } f : (a \neq b) \Rightarrow (f[a] \neq f[b])$ 
18 ┌────────────────────────────────────────────────────────────────────────┐
19 CONSTANTS
20   Participant, the set of participants
21   Value         the set of possible input values for Participant to propose
22
23   None  $\triangleq \text{CHOOSE } b : b \notin \text{Value}$ 
24    $NP \triangleq \text{Cardinality}(\text{Participant})$  number of  $p \in \text{Participants}$ 
25
26    $Quorum \triangleq \{Q \in \text{SUBSET } Participant : \text{Cardinality}(Q) * 2 = NP + 1\}$ 
27   ASSUME  $QuorumAssumption \triangleq$ 
28      $\wedge \forall Q \in Quorum : Q \subseteq Participant$ 
29      $\wedge \forall Q1, Q2 \in Quorum : Q1 \cap Q2 \neq \{\}$ 
30
31    $Ballot \triangleq Nat$ 
32
33    $PIndex \triangleq \text{CHOOSE } f \in [Participant \rightarrow 1..NP] : Injective(f)$ 
34    $Bals(p) \triangleq \{b \in Ballot : b \% NP = PIndex[p] - 1\}$  allocate ballots for  $p \in Participant$ 
35 ┌────────────────────────────────────────────────────────────────────────┐
36    $State \triangleq [maxBal : Ballot \cup \{-1\},$ 
37      $maxVBal : Ballot \cup \{-1\}, maxVVal : Value \cup \{None\}]$ 
38
39    $InitState \triangleq [maxBal \mapsto -1, maxVBal \mapsto -1, maxVVal \mapsto None]$ 
    For simplicity, in this specification, we choose to send the complete state of a participant each
    time. When receiving such a message, the participant processes only the "partial" state it needs.
45    $Message \triangleq [from : Participant, to : \text{SUBSET } Participant, state : [Participant \rightarrow State]]$ 
46 ┌────────────────────────────────────────────────────────────────────────┐
47   VARIABLES
48     state,  $state[p][q]$ : the state of  $q \in Participant$  from the view of  $p \in Participant$ 
49     msgs   the set of messages that have been sent
50
51    $vars \triangleq \langle state, msgs \rangle$ 
52
53    $TypeOK \triangleq$ 
54      $\wedge state \in [Participant \rightarrow [Participant \rightarrow State]]$ 
55      $\wedge msgs \subseteq Message$ 

```

```

57  $Send(m) \triangleq msgs' = msgs \cup \{m\}$ 
58 |-----|
59  $Init \triangleq$ 
60    $\wedge state = [p \in Participant \mapsto [q \in Participant \mapsto InitState]]$ 
61    $\wedge msgs = \{\}$ 
62    $p \in Participant$  starts the prepare phase by issuing a ballot  $b \in Ballot$ .
63
64  $Prepare(p, b) \triangleq$ 
65    $\wedge state[p][p].maxBal < b$ 
66    $\wedge b \in Bals(p)$ 
67    $\wedge state' = [state \text{ EXCEPT } ![p][p].maxBal = b]$ 
68    $\wedge Send([from \mapsto p, to \mapsto Participant, state \mapsto state'[p]])$ 
69    $q \in Participant$  updates its own state  $state[q]$  according to the actual state  $pp$  of  $p \in Participant$ 
    extracted from a message  $m \in Message$  it receives. This is called by  $OnMessage(q)$ .
    Note:  $pp$  is  $m.state[p]$ ; it may not be equal to  $state[p][p]$  at the time  $UpdateState$  is called.
70
71  $UpdateState(q, p, pp) \triangleq$ 
72    $state' = [state \text{ EXCEPT}$ 
73      $![q][p].maxBal = Max(@, pp.maxBal),$ 
74      $![q][p].maxVVal = Max(@, pp.maxVVal),$ 
75      $![q][p].maxVVal = \text{IF } state[q][p].maxVVal < pp.maxVVal$ 
76        $\text{THEN } pp.maxVVal \text{ ELSE } @,$ 
77      $![q][q].maxBal = Max(@, pp.maxBal),$ 
78      $![q][q].maxVVal = \text{IF } state[q][q].maxVVal \leq pp.maxVVal$ 
79        $\text{THEN } pp.maxVVal \text{ ELSE } @, \text{ make promise}$ 
80      $![q][q].maxVVal = \text{IF } state[q][q].maxVVal \leq pp.maxVVal$ 
81        $\text{THEN } pp.maxVVal \text{ ELSE } @] \text{ accept}$ 
82    $q \in Participant$  receives and processes a message in  $Message$ .
83
84  $OnMessage(q) \triangleq$ 
85    $\exists m \in msgs :$ 
86      $\wedge q \in m.to$ 
87      $\wedge \text{LET } p \triangleq m.from$ 
88      $\text{IN } UpdateState(q, p, m.state[p])$ 
89      $\wedge \text{IF } \vee m.state[q].maxBal < state'[q][q].maxBal$ 
90        $\vee m.state[q].maxVVal < state'[q][q].maxVVal$ 
91        $\text{THEN } Send([from \mapsto q, to \mapsto \{m.from\}, state \mapsto state'[q]])$ 
92      $\text{ELSE UNCHANGED } msgs$ 
93    $p \in Participant$  starts the accept phase by issuing the ballot  $b \in Ballot$  with value  $v \in Value$ .
94
95  $Accept(p, b, v) \triangleq$ 
96    $\wedge b \in Bals(p)$ 
97    $\wedge \exists Q \in Quorum : \forall q \in Q : state[p][q].maxBal = b$ 
98    $\wedge \forall q \in Participant : state[p][q].maxVVal = -1 \text{ free to pick its own value}$ 
99    $\vee \exists q \in Participant : v \text{ is the value with the highest } maxVVal$ 
100    $\wedge state[p][q].maxVVal = v$ 
101    $\wedge \forall r \in Participant : state[p][q].maxVVal \geq state[p][r].maxVVal$ 

```

```

112       $\wedge state' = [state \text{ EXCEPT } ![p][p].maxVBal = b, ![p][p].maxVVal = v] \text{ accept}$ 
113       $\wedge Send([from \mapsto p, to \mapsto Participant, state \mapsto state'[p]])$ 
114  |-----|
115   $Next \triangleq \exists p \in Participant : \vee OnMessage(p)$ 
116                                $\vee \exists b \in Ballot : \vee Prepare(p, b)$ 
117                                $\vee \exists v \in Value : Accept(p, b, v)$ 
118   $Spec \triangleq Init \wedge \Box [Next]_{vars}$ 
119  |-----|
120  UniversalPaxosStore satisfies the Consistency property.
121  |-----|
122   $ChosenP(p) \triangleq \text{the set of values chosen by } p \in Participant$ 
123   $\{v \in Value : \exists b \in Ballot :$ 
124       $\exists Q \in Quorum : \forall q \in Q : \wedge state[p][q].maxVBal = b$ 
125       $\wedge state[p][q].maxVVal = v\}$ 
126   $chosen \triangleq \text{UNION } \{ChosenP(p) : p \in Participant\}$ 
127
128   $Consistency \triangleq Cardinality(chosen) \leq 1$ 
129  THEOREM  $Spec \Rightarrow \Box Consistency$ 
130
131  |-----|
132  \ * Modification History
133  \ * Last modified Wed Aug 14 16:09:53 CST 2019 by hengxin
134  \ * Last modified Mon Jul 22 13:59:15 CST 2019 by pure_
135  \ * Last modified Mon Jun 03 21:26:09 CST 2019 by stary
136  \ * Last modified Wed May 09 21:39:31 CST 2018 by dell
137  \ * Created Mon Apr 23 15:47:52 GMT +08:00 2018 by pure_

```