MODULE *design*

Design of a system persisting messages in two distinct databases

CONSTANT $Message$ the set of all possible messages
CONSTANT $Process$ a set of consumer processes

each process loops on the following steps:
r (polling to receive a message)
$w1$ (writes to the first database)
$w2$ (writes to the second database)
$Step \triangleq \{\text{“r”}, \text{“w1”}, \text{“w2”}\}$

one message chosen as default value
$default \triangleq$ CHOOSE $m \in Message$: TRUE

VARIABLE $db1$
VARIABLE $db2$
the next step of each process
VARIABLE $processNextStep$
the current message being processed by each process
VARIABLE $processCurrentMessage$

the tuple of all variables
$vars \triangleq \langle db1, db2, processNextStep, processCurrentMessage\rangle$

$TypeOK \triangleq$
$\land \quad db1 \in Message$
$\land \quad db2 \in Message$
a function mapping processes to steps
$\land \quad processNextStep \in [Process \to Step]$
a function mapping processes to messages
$\land \quad processCurrentMessage \in [Process \to Message]$

$Init \triangleq$
$\land\ db1 = default$
$\land\ db2 = default$
all processes start in the receiving state
$\land\ processNextStep = [p \in Process \mapsto \text{“r”}]$
all processes start with the default message (but it will be discarded on the first receive step)
$\land\ processCurrentMessage = [p \in Process \mapsto default]$

some process receives some (non-default) message
its next step from 'r' becomes 'w1'
its current message becomes the received one
everything else is unchanged

$Receive \triangleq \exists\, m \in Message \setminus \{default\},\ p \in Process :$
$\quad \wedge\ processNextStep[p] = \text{“r”}$

additional condition to ensure eventual consistency:
$\quad \wedge \forall\, x \in Process:\ processNextStep[x] = \text{“r”}$

$\quad \wedge\ processNextStep' = [processNextStep \text{ EXCEPT } ![p] = \text{“w1”}]$
$\quad \wedge\ processCurrentMessage' = [processCurrentMessage \text{ EXCEPT } ![p] = m]$
$\quad \wedge \text{ UNCHANGED } \langle db1,\ db2 \rangle$

some process writes its current message to the first database
and advances its 'program counter' to 'w2'
$Write1 \triangleq \exists\, p \in Process :$
$\quad \wedge\ processNextStep[p] = \text{“w1”}$
$\quad \wedge\ processNextStep' = [processNextStep \text{ EXCEPT } ![p] = \text{“w2”}]$
$\quad \wedge\ db1' = processCurrentMessage[p]$
$\quad \wedge \text{ UNCHANGED } \langle db2,\ processCurrentMessage \rangle$

some process writes its current message to the second database
$Write2 \triangleq \exists\, p \in Process :$
$\quad \wedge\ processNextStep[p] = \text{“w2”}$
$\quad \wedge\ processNextStep' = [processNextStep \text{ EXCEPT } ![p] = \text{“r”}]$
$\quad \wedge\ db2' = processCurrentMessage[p]$
$\quad \wedge \text{ UNCHANGED } \langle db1,\ processCurrentMessage \rangle$

$Next \triangleq$
$\quad \vee\ Receive$
$\quad \vee\ Write1$
$\quad \vee\ Write2$

$Liveness \triangleq \text{WF}_{vars}(Write1 \vee Write2)$

$Spec \triangleq Init \wedge \Box[Next]_{vars} \wedge Liveness$

this property is violated in models with at least two processes
(and two messages besides the default one)
$DbConsistency \triangleq \Box\Diamond(db2 = db1)$