MODULE *Banking*

EXTENDS *Integers*

```
--algorithm Banking
variables
    people = { "alice", "bob" },
    acc = [p ∈ people ↦ 5],

define
    NoOverdrafts ≜ ∀ p ∈ people : acc[p] ≥ 0
    EventuallyConsistent ≜ ◇□(acc["alice"] + acc["bob"] = 10)
end define ;

fair process Wire ∈ 1 .. 2
    variables
        sender = "alice",
        receiver = "bob",
        amount ∈ 1 .. acc[sender] ;

begin
    CheckAndWithdraw:
        if amount ≤ acc[sender] then
            acc[sender] := acc[sender] − amount ;
          Deposit:
            acc[receiver] := acc[receiver] + amount ;
        end if ;
end process ;

end algorithm ;
```

BEGIN TRANSLATION

VARIABLES *people*, *acc*, *pc*

define statement

$NoOverdrafts \triangleq \forall\, p \in people : acc[p] \geq 0$

$EventuallyConsistent \triangleq \Diamond\Box(acc[\text{“alice”}] + acc[\text{“bob”}] = 10)$

VARIABLES *sender*, *receiver*, *amount*

$vars \triangleq \langle people,\ acc,\ pc,\ sender,\ receiver,\ amount \rangle$

$ProcSet \triangleq (1 \,..\, 2)$

$Init \triangleq$ Global variables

$\wedge\ people = \{\text{“alice”},\ \text{“bob”}\}$

$\wedge\ acc = [p \in people \mapsto 5]$

Process *Wire*

$\wedge\ sender = [self \in 1 \,..\, 2 \mapsto \text{“alice”}]$

$\wedge\ receiver = [self \in 1 \,..\, 2 \mapsto \text{“bob”}]$

$\land\ amount \in [1\,..\,2 \to 1\,..\,acc[sender[\text{CHOOSE } self \in 1\,..\,2 : \text{TRUE}]]]$
$\land\ pc = [self \in ProcSet \mapsto \text{“CheckAndWithdraw”}]$

$$
\begin{aligned}
CheckAndWithdraw(self) \triangleq\ & \land\ pc[self] = \text{“CheckAndWithdraw”} \\
& \land\ \text{IF } amount[self] \leq acc[sender[self]] \\
& \quad \text{THEN } \land\ acc' = [acc \text{ EXCEPT } ![sender[self]] = acc[sender[self]] - amount[\\
& \qquad\ \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Deposit”}] \\
& \quad \text{ELSE } \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Done”}] \\
& \qquad\ \land\ acc' = acc \\
& \land\ \text{UNCHANGED } \langle people, sender, receiver, amount \rangle
\end{aligned}
$$

$$
\begin{aligned}
Deposit(self) \triangleq\ & \land\ pc[self] = \text{“Deposit”} \\
& \land\ acc' = [acc \text{ EXCEPT } ![receiver[self]] = acc[receiver[self]] + amount[self]] \\
& \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“Done”}] \\
& \land\ \text{UNCHANGED } \langle people, sender, receiver, amount \rangle
\end{aligned}
$$

$$Wire(self) \triangleq CheckAndWithdraw(self) \lor Deposit(self)$$

$$
\begin{aligned}
Next \triangleq\ & (\exists\, self \in 1\,..\,2 : Wire(self)) \\
& \lor \text{ Disjunct to prevent deadlock on termination} \\
& \quad ((\forall\, self \in ProcSet : pc[self] = \text{“Done”}) \land \text{UNCHANGED } vars)
\end{aligned}
$$

$$
\begin{aligned}
Spec \triangleq\ & \land\ Init \land \Box[Next]_{vars} \\
& \land\ \forall\, self \in 1\,..\,2 : \mathrm{WF}_{vars}(Wire(self))
\end{aligned}
$$

$$Termination \triangleq \Diamond(\forall\, self \in ProcSet : pc[self] = \text{“Done”})$$

END TRANSLATION

* Modification History
* Last modified Sun May 05 09:26:55 *PDT* 2019 by *jasondebolt*
* Created Sun May 05 08:53:55 *PDT* 2019 by *jasondebolt*