MODULE *callback*

EXTENDS *TLC*, *Sequences*, *SequencesExt*, *FiniteSets*, *Integers*

CONSTANTS *Timers*, *DeltaRange*, *Servers*, *Clients*, *Subscribers*

$Tasks \triangleq Subscribers \cup Servers \cup Clients$

--algorithm *callback*
variables

list for timer

example: $\langle [delta \mapsto 3,\ name \mapsto \text{“}timer1\text{”}],\ [delta \mapsto 2,\ name \mapsto \text{“}timer2\text{”}] \rangle$

$delta\_list = SetToSeq(\{[delta \mapsto random\_num(0,\ DeltaRange),\ name \mapsto x] : x \in Timers\})$;

events

$runnable = \{\}$;

tasks

$running = \{\}$;
$waiting\ = Tasks$;

define

$random\_num(min,\ max) \triangleq$ CHOOSE $i \in min\ ..\ max$: TRUE
$starvation\_free \triangleq \forall\, x\ \in (Timers \cup Tasks) :$
$(((x \in \{y.name : y \in ToSet(delta\_list)\}) \vee (x \in runnable)) \rightsquigarrow \Diamond(x \in running))$
$pick\_task(set) \triangleq$ CHOOSE $x \in set$: TRUE
end define

To emulate incrementing clock, decrement the delta of the head of the *delta_list*.

macro *increment_clock*()
begin
if $delta\_list \neq \langle\rangle \wedge delta\_list[1].delta > 0$ **then**
$delta\_list[1].delta := delta\_list[1].delta - 1$;
end if ;
end macro ;

execute a *callback* function

procedure *callback*(*name*)
begin
start_callback:
increment_clock() ;
$running := running \cup \{name\}$;

end_callback:
$running := running \setminus \{name\}$;
return ;
end procedure ;

reenable timer with at random delay

```
procedure reload_timer(name)
variables
    idx ;
    delta ;
begin
    start_reload_timer:
        increment_clock() ;

         choose insertion point
        idx := random_num(1, Len(delta_list) + 1) ;
        if idx ≤ Len(delta_list) then
             insert to middle
            delta := random_num(0, delta_list[idx].delta) ;

            reload_insert1:
                 update delta and insert
                delta_list[idx].delta := delta_list[idx].delta − delta ;

            reload_insert2:
                delta_list := InsertAt(delta_list, idx, [delta ↦ delta, name ↦ name]) ;
         else
             insert to the end
            delta := random_num(0, DeltaRange) ;

            reload_insert_end:
                delta_list := Append(delta_list, [delta ↦ delta, name ↦ name]) ;
            skip ;
        end if ;

    end_reload_timer:
        return ;
end procedure ;

 execute a task
procedure execute_task(wait_set)
variables
    task ;
begin
    start_task:
        while wait_set ≠ {} do
            task := pick_task(wait_set) ;
            wait_set := wait_set \ {task} ;
            call callback(task) ;

            finish_a_task:
                waiting := waiting ∪ {task} ;
        end while ;
```

```
        return ;
end procedure ;

fair process trigger_event ∈ Tasks
begin
    fire_event:
        while TRUE do
            if self ∈ waiting then
                runnable := runnable ∪ {self} ;
                waiting := waiting \ {self} ;
            end if ;
        end while ;
end process ;

fair + process executor = "executor"
variables
    head ;
    to_be_reloaded = ⟨⟩ ;
begin
    start_executor:
        while TRUE do
            increment_clock() ;

            execute:
                while delta_list ≠ ⟨⟩ ∧ delta_list[1].delta = 0 do
                    pop front
                    head := Head(delta_list) ;
                    delta_list := Tail(delta_list) ;

                    call the callback function
                    call callback(head.name) ;

                    reenable timer later
                    save_timer:
                        to_be_reloaded := Append(to_be_reloaded, head.name) ;
                end while ;

            reload:
                reenable timer
                while to_be_reloaded ≠ ⟨⟩ do
                    call reload_timer(to_be_reloaded[1]) ;

                    reload2:
                        to_be_reloaded := Tail(to_be_reloaded) ;
                end while ;

            execute_tasks:
```

```
                pick runnable tasks up
              with tmp_runnable = runnable do
                  runnable := {} ;
                  call execute_task(tmp_runnable) ;
              end with ;

        end while ;
end process ;
end algorithm ;
```

BEGIN TRANSLATION ($chksum(pcal) =$ "3e628361" $\wedge\ chksum(tla) =$ "5beb5484")

Parameter name of procedure *callback* at line 38 col 20 changed to *name_*

CONSTANT $defaultInitValue$

VARIABLES $delta\_list$, $runnable$, $running$, $waiting$, pc, $stack$

define statement

$random\_num(min,\ max) \triangleq$ CHOOSE $i \in min\ ..\ max$: TRUE

$starvation\_free \triangleq \forall\, x \in (Timers \cup Tasks) :$
$\quad (((x \in \{y.name : y \in ToSet(delta\_list)\}) \vee (x \in runnable)) \rightsquigarrow \Diamond(x \in running))$

$pick\_task(set) \triangleq$ CHOOSE $x \in set$: TRUE

VARIABLES $name\_$, $name$, idx, $delta$, $wait\_set$, $task$, $head$, $to\_be\_reloaded$

$vars \triangleq \langle delta\_list,\ runnable,\ running,\ waiting,\ pc,\ stack,\ name\_,\ name,$
$\quad idx,\ delta,\ wait\_set,\ task,\ head,\ to\_be\_reloaded\rangle$

$ProcSet \triangleq (Tasks) \cup \{$"executor"$\}$

$Init \triangleq$ Global variables
$\wedge\ delta\_list = SetToSeq(\{[delta \mapsto random\_num(0,\ DeltaRange),\ name \mapsto x] : x \in Timers\})$
$\wedge\ runnable = \{\}$
$\wedge\ running = \{\}$
$\wedge\ waiting\ = Tasks$
Procedure *callback*
$\wedge\ name\_ = [self \in ProcSet \mapsto defaultInitValue]$
Procedure *reload_timer*
$\wedge\ name = [self \in ProcSet \mapsto defaultInitValue]$
$\wedge\ idx = [self \in ProcSet \mapsto defaultInitValue]$
$\wedge\ delta = [self \in ProcSet \mapsto defaultInitValue]$
Procedure *execute_task*
$\wedge\ wait\_set = [self \in ProcSet \mapsto defaultInitValue]$
$\wedge\ task = [self \in ProcSet \mapsto defaultInitValue]$
Process executor
$\wedge\ head = defaultInitValue$
$\wedge\ to\_be\_reloaded = \langle\rangle$
$\wedge\ stack = [self \in ProcSet \mapsto \langle\rangle]$
$\wedge\ pc = [self \in ProcSet \mapsto$ CASE $self \in Tasks \rightarrow$ "fire_event"
$\qquad\qquad\qquad\qquad\qquad\qquad \square\quad self =$ "executor" $\rightarrow$ "start_executor"$]$

$start\_callback(self) \triangleq$ $\land\ pc[self] =$ "start_callback"
$\land$ IF $delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$
THEN $\land\ delta\_list' = [delta\_list$ EXCEPT $![1].delta = delta\_list[1].delta - 1]$
ELSE $\land$ TRUE
$\land$ UNCHANGED $delta\_list$
$\land\ running' = (running \cup \{name\_[self]\})$
$\land\ pc' = [pc$ EXCEPT $![self] =$ "end_callback"$]$
$\land$ UNCHANGED $\langle runnable,\ waiting,\ stack,\ name\_,\ name,$
$idx,\ delta,\ wait\_set,\ task,\ head,$
$to\_be\_reloaded\rangle$

$end\_callback(self) \triangleq$ $\land\ pc[self]\ =$ "end_callback"
$\land\ running' = running \setminus \{name\_[self]\}$
$\land\ pc' = [pc$ EXCEPT $![self] = Head(stack[self]).pc]$
$\land\ name\_' = [name\_$ EXCEPT $![self] = Head(stack[self]).name\_]$
$\land\ stack'\ = [stack$ EXCEPT $![self]\ = Tail(stack[self])]$
$\land$ UNCHANGED $\langle delta\_list,\ runnable,\ waiting,\ name,\ idx,$
$delta,\ wait\_set,\ task,\ head,$
$to\_be\_reloaded\rangle$

$callback(self) \triangleq start\_callback(self) \lor end\_callback(self)$

$start\_reload\_timer(self) \triangleq$ $\land\ pc[self] =$ "start_reload_timer"
$\land$ IF $delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$
THEN $\land\ delta\_list' = [delta\_list$ EXCEPT $![1].delta = delta\_list[1].delta$ -
ELSE $\land$ TRUE
$\land$ UNCHANGED $delta\_list$
$\land\ idx' = [idx$ EXCEPT $![self] = random\_num(1,\ Len(delta\_list') + 1)]$
$\land$ IF $idx'[self] \leq Len(delta\_list')$
THEN $\land\ delta' = [delta$ EXCEPT $![self] = random\_num(0,\ delta\_list'[idx$
$\land\ pc' = [pc$ EXCEPT $![self] =$ "reload_insert1"$]$
ELSE $\land\ delta' = [delta$ EXCEPT $![self] = random\_num(0,\ DeltaRange)]$
$\land\ pc' = [pc$ EXCEPT $![self] =$ "reload_insert_end"$]$
$\land$ UNCHANGED $\langle runnable,\ running,\ waiting,\ stack,$
$name\_,\ name,\ wait\_set,\ task,\ head,$
$to\_be\_reloaded\rangle$

$reload\_insert1(self) \triangleq$ $\land\ pc[self] =$ "reload_insert1"
$\land\ delta\_list' = [delta\_list$ EXCEPT $![idx[self]].delta = delta\_list[idx[self]].delta - d$
$\land\ pc' = [pc$ EXCEPT $![self] =$ "reload_insert2"$]$
$\land$ UNCHANGED $\langle runnable,\ running,\ waiting,\ stack,$
$name\_,\ name,\ idx,\ delta,\ wait\_set,$
$task,\ head,\ to\_be\_reloaded\rangle$

$reload\_insert2(self) \triangleq$ $\land\ pc[self] =$ "reload_insert2"
$\land\ delta\_list' = InsertAt(delta\_list,\ idx[self],\ [delta \mapsto delta[self],\ name \mapsto name[s$

$$
\begin{array}{l}
\quad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“end\_reload\_timer”}] \\
\quad \land \text{ UNCHANGED } \langle runnable,\ running,\ waiting,\ stack, \\
\qquad\qquad\qquad name\_,\ name,\ idx,\ delta,\ wait\_set, \\
\qquad\qquad\qquad task,\ head,\ to\_be\_reloaded\rangle
\end{array}
$$

$$
\begin{array}{l}
reload\_insert\_end(self) \triangleq \land\ pc[self] = \text{“reload\_insert\_end”} \\
\qquad\qquad\qquad \land\ delta\_list' = Append(delta\_list,\ [delta \mapsto delta[self],\ name \mapsto name[self]]) \\
\qquad\qquad\qquad \land \text{ TRUE} \\
\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“end\_reload\_timer”}] \\
\qquad\qquad\qquad \land \text{ UNCHANGED } \langle runnable,\ running,\ waiting,\ stack, \\
\qquad\qquad\qquad\qquad\qquad name\_,\ name,\ idx,\ delta,\ wait\_set, \\
\qquad\qquad\qquad\qquad\qquad task,\ head,\ to\_be\_reloaded\rangle
\end{array}
$$

$$
\begin{array}{l}
end\_reload\_timer(self) \triangleq \land\ pc[self] = \text{“end\_reload\_timer”} \\
\qquad\qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc] \\
\qquad\qquad\qquad \land\ idx' = [idx \text{ EXCEPT } ![self] = Head(stack[self]).idx] \\
\qquad\qquad\qquad \land\ delta' = [delta \text{ EXCEPT } ![self] = Head(stack[self]).delta] \\
\qquad\qquad\qquad \land\ name' = [name \text{ EXCEPT } ![self] = Head(stack[self]).name] \\
\qquad\qquad\qquad \land\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])] \\
\qquad\qquad\qquad \land \text{ UNCHANGED } \langle delta\_list,\ runnable,\ running, \\
\qquad\qquad\qquad\qquad\qquad waiting,\ name\_,\ wait\_set,\ task,\ head, \\
\qquad\qquad\qquad\qquad\qquad to\_be\_reloaded\rangle
\end{array}
$$

$$
\begin{array}{l}
reload\_timer(self) \triangleq\ start\_reload\_timer(self) \lor reload\_insert1(self) \\
\qquad\qquad\qquad\qquad \lor\ reload\_insert2(self) \lor reload\_insert\_end(self) \\
\qquad\qquad\qquad\qquad \lor\ end\_reload\_timer(self)
\end{array}
$$

$$
\begin{array}{l}
start\_task(self) \triangleq \land\ pc[self] = \text{“start\_task”} \\
\qquad\qquad\qquad \land \text{ IF } wait\_set[self] \neq \{\} \\
\qquad\qquad\qquad\qquad \text{THEN } \land\ task' = [task \text{ EXCEPT } ![self] = pick\_task(wait\_set[self])] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ wait\_set' = [wait\_set \text{ EXCEPT } ![self] = wait\_set[self] \setminus \{task'[self]\}] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ \land\ name\_' = [name\_ \text{ EXCEPT } ![self] = task'[self]] \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad \land\ stack' = [stack \text{ EXCEPT } ![self] = \langle [procedure \mapsto \text{“callback”}, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad pc \mapsto \text{“finish\_a\_task”}, \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad name\_ \mapsto name\_[self]]\rangle \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \circ\ stack[self]] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“start\_callback”}] \\
\qquad\qquad\qquad\qquad \text{ELSE } \land\ pc' = [pc \text{ EXCEPT } ![self] = Head(stack[self]).pc] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ task' = [task \text{ EXCEPT } ![self] = Head(stack[self]).task] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ wait\_set' = [wait\_set \text{ EXCEPT } ![self] = Head(stack[self]).wait\_set] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ stack' = [stack \text{ EXCEPT } ![self] = Tail(stack[self])] \\
\qquad\qquad\qquad\qquad\qquad\quad \land\ name\_' = name\_ \\
\qquad\qquad\qquad \land \text{ UNCHANGED } \langle delta\_list,\ runnable,\ running,\ waiting, \\
\qquad\qquad\qquad\qquad\qquad name,\ idx,\ delta,\ head,\ to\_be\_reloaded\rangle
\end{array}
$$

$$
finish\_a\_task(self) \triangleq \land\ pc[self] = \text{“finish\_a\_task”}
$$

$\land\ waiting' = (waiting \cup \{task[self]\})$
$\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“start\_task”}]$
$\land \text{ UNCHANGED } \langle delta\_list,\ runnable,\ running,\ stack,$
$name\_,\ name,\ idx,\ delta,\ wait\_set,\ task,$
$head,\ to\_be\_reloaded\rangle$

$execute\_task(self) \triangleq start\_task(self) \lor finish\_a\_task(self)$

$fire\_event(self) \triangleq \land\ pc[self] = \text{“fire\_event”}$
$\land \text{ IF } self \in waiting$
$\text{THEN } \land\ runnable' = (runnable \cup \{self\})$
$\land\ waiting' = waiting \setminus \{self\}$
$\text{ELSE } \land \text{ TRUE}$
$\land \text{ UNCHANGED } \langle runnable,\ waiting\rangle$
$\land\ pc' = [pc \text{ EXCEPT } ![self] = \text{“fire\_event”}]$
$\land \text{ UNCHANGED } \langle delta\_list,\ running,\ stack,\ name\_,\ name,$
$idx,\ delta,\ wait\_set,\ task,\ head,$
$to\_be\_reloaded\rangle$

$trigger\_event(self) \triangleq fire\_event(self)$

$start\_executor \triangleq \land\ pc[\text{“executor”}] = \text{“start\_executor”}$
$\land \text{ IF } delta\_list \neq \langle\rangle \land delta\_list[1].delta > 0$
$\text{THEN } \land\ delta\_list' = [delta\_list \text{ EXCEPT } ![1].delta = delta\_list[1].delta - 1]$
$\text{ELSE } \land \text{ TRUE}$
$\land \text{ UNCHANGED } delta\_list$
$\land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“execute”}]$
$\land \text{ UNCHANGED } \langle runnable,\ running,\ waiting,\ stack,\ name\_,$
$name,\ idx,\ delta,\ wait\_set,\ task,\ head,$
$to\_be\_reloaded\rangle$

$execute \triangleq \land\ pc[\text{“executor”}] = \text{“execute”}$
$\land \text{ IF } delta\_list \neq \langle\rangle \land delta\_list[1].delta = 0$
$\text{THEN } \land\ head' = Head(delta\_list)$
$\land\ delta\_list' = Tail(delta\_list)$
$\land\ \land\ name\_' = [name\_ \text{ EXCEPT } ![\text{“executor”}] = head'.name]$
$\land\ stack' = [stack \text{ EXCEPT } ![\text{“executor”}] = \langle [procedure \mapsto \text{“callback”},$
$pc \mapsto \text{“save\_timer”},$
$name\_ \mapsto name\_[\text{“executor”}]]$
$\circ\ stack[\text{“executor”}]]$
$\land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“start\_callback”}]$
$\text{ELSE } \land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“reload”}]$
$\land \text{ UNCHANGED } \langle delta\_list,\ stack,\ name\_,\ head\rangle$
$\land \text{ UNCHANGED } \langle runnable,\ running,\ waiting,\ name,\ idx,\ delta,$
$wait\_set,\ task,\ to\_be\_reloaded\rangle$

$save\_timer \triangleq \land\ pc[\text{“executor”}] = \text{“save\_timer”}$

$$
\begin{array}{l}
\quad \land\ to\_be\_reloaded' = Append(to\_be\_reloaded,\ head.name) \\
\quad \land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“execute”}] \\
\quad \land \text{ UNCHANGED } \langle delta\_list,\ runnable,\ running,\ waiting,\ stack, \\
\qquad\qquad\qquad name\_,\ name,\ idx,\ delta,\ wait\_set,\ task,\ head\rangle
\end{array}
$$

$$
\begin{array}{lll}
reload \triangleq & \land\ pc[\text{“executor”}] = \text{“reload”} & \\
& \land \text{ IF } to\_be\_reloaded \neq \langle\rangle & \\
& \quad \text{THEN } \land\ \land\ name' = [name \text{ EXCEPT } ![\text{“executor”}] = to\_be\_reloaded[1]] & \\
& \qquad\qquad \land\ stack' = [stack \text{ EXCEPT } ![\text{“executor”}] = \langle [procedure \mapsto \text{“reload\_timer”}, & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad pc \mapsto \text{“reload2”}, & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad idx \mapsto idx[\text{“executor”}], & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad delta \mapsto delta[\text{“executor”}], & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad name \mapsto name[\text{“executor”}]] \rangle & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \circ\ stack[\text{“executor”}]] & \\
& \qquad\qquad \land\ idx' = [idx \text{ EXCEPT } ![\text{“executor”}] = defaultInitValue] & \\
& \qquad\qquad \land\ delta' = [delta \text{ EXCEPT } ![\text{“executor”}] = defaultInitValue] & \\
& \qquad\qquad \land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“start\_reload\_timer”}] & \\
& \quad \text{ELSE } \land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“execute\_tasks”}] & \\
& \qquad\qquad \land \text{ UNCHANGED } \langle stack,\ name,\ idx,\ delta\rangle & \\
& \land \text{ UNCHANGED } \langle delta\_list,\ runnable,\ running,\ waiting,\ name\_, & \\
& \qquad\qquad wait\_set,\ task,\ head,\ to\_be\_reloaded\rangle &
\end{array}
$$

$$
\begin{array}{l}
reload2 \triangleq\ \land\ pc[\text{“executor”}] = \text{“reload2”} \\
\qquad\quad \land\ to\_be\_reloaded' = Tail(to\_be\_reloaded) \\
\qquad\quad \land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“reload”}] \\
\qquad\quad \land \text{ UNCHANGED } \langle delta\_list,\ runnable,\ running,\ waiting,\ stack, \\
\qquad\qquad\qquad\quad name\_,\ name,\ idx,\ delta,\ wait\_set,\ task,\ head\rangle
\end{array}
$$

$$
\begin{array}{lll}
execute\_tasks \triangleq & \land\ pc[\text{“executor”}] = \text{“execute\_tasks”} & \\
& \land \text{ LET } tmp\_runnable \triangleq runnable \text{ IN} & \\
& \quad \land\ runnable' = \{\} & \\
& \quad \land\ \land\ stack' = [stack \text{ EXCEPT } ![\text{“executor”}] = \langle [procedure \mapsto \text{“execute\_task”}, & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad pc \mapsto \text{“start\_executor”}, & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad task \mapsto task[\text{“executor”}], & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad wait\_set \mapsto wait\_set[\text{“executor”}]] \rangle & \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \circ\ stack[\text{“executor”}]] & \\
& \qquad\ \land\ wait\_set' = [wait\_set \text{ EXCEPT } ![\text{“executor”}] = tmp\_runnable] & \\
& \quad \land\ task' = [task \text{ EXCEPT } ![\text{“executor”}] = defaultInitValue] & \\
& \quad \land\ pc' = [pc \text{ EXCEPT } ![\text{“executor”}] = \text{“start\_task”}] & \\
& \land \text{ UNCHANGED } \langle delta\_list,\ running,\ waiting,\ name\_,\ name, & \\
& \qquad\qquad idx,\ delta,\ head,\ to\_be\_reloaded\rangle &
\end{array}
$$

$$
\begin{array}{l}
executor \triangleq\ start\_executor \lor execute \lor save\_timer \lor reload \lor reload2 \\
\qquad\qquad \lor\ execute\_tasks
\end{array}
$$

$$
Next \triangleq\ executor
$$

$$
\begin{array}{l}
\quad \lor (\exists\, self \in ProcSet : \quad \lor callback(self) \lor reload\_timer(self) \\
\qquad\qquad\qquad\qquad\qquad\quad \lor execute\_task(self)) \\
\quad \lor (\exists\, self \in Tasks : trigger\_event(self)) \\
\\
Spec \triangleq \ \land Init \land \Box[Next]_{vars} \\
\qquad \land \forall\, self \in Tasks : \mathrm{WF}_{vars}(trigger\_event(self)) \\
\qquad \land \ \land \mathrm{SF}_{vars}(executor) \\
\qquad\quad \land \mathrm{SF}_{vars}(callback(\text{“executor”})) \\
\qquad\quad \land \mathrm{SF}_{vars}(reload\_timer(\text{“executor”})) \\
\qquad\quad \land \mathrm{SF}_{vars}(execute\_task(\text{“executor”}))
\end{array}
$$

END TRANSLATION