## 23.6 The Timeout operation

The **timeout** operation allows to check the expiration of timers.

***Syntactical Structure***

( ( ( *TimerIdentifier* | *TimerParIdentifier* ) { "[" *SingleExpression* "]" } ) |

**any** **timer** |

**any from** TimerArrayRef )

"." **timeout**

**[**"->" **@index value** VariableRef **]**

***Semantic Description***

The **timeout** operation allows to check the expiration of a specific timer in the scope unit of a test component or module control in which the timeout operation has been called or of any timer that has been started on a test component or module control before entering the scope in which the **timeout** operation has been called.

When a **timeout** operation is processed, if a timer name is indicated, the timeout-list is searched according to the TTCN‑3 scope rules. If there is a timeout event matching the timer name, that event is removed from the timeout-list, and the **timeout** operation succeeds.

The **timeout** can be used to determine an alternative in an **alt** statement or as stand-alone statement in a behaviour description. In the latter case a **timeout** operation is considered to be shorthand for an **alt** statement with the **timeout** operation as the only alternative.

The **any** keyword used with the **timeout** operation succeeds if the timeout-list is not empty. In this case a randomly chosen timeout event is removed from the timeout-list.

When the **any from** TimerArrayRefnotation is used, where TimerArrayRef shall be a timer array identifier, the timers from the referenced array are iterated over and individually checked for timeout from innermost to outermost dimension from lowest to highest index for each dimension. The first timer to be found in the timeout-list causes that timer to be removed from the list and the timeout operation succeeds. The index of the matched timer can be optionally stored in an integer variable for single-dimensional arrays or to an integer array or record of integer variable for multi-dimensional timer arrays.

***Restrictions***

In addition to the general static rules of TTCN‑3 given in clause 5 and shown in table 15, the following restrictions apply:

1. The **timeout** operation does not return any value and therefore shall not be used in a expressions.
2. *TimerArrayRef* shall be a reference to a completely initialized timer array.
3. The index redirection shall only be used for **any from** timer array timeout operations.
4. If the index redirection is used for single-dimensional timer arrays, the type of the integer variable shall allow storing the highest index of the respective timer array.
5. If the index redirection is used for multi-dimensional timer arrays, the size of the integer array or record of integer type shall exactly be the same as the dimension of the respective timer array, and its type shall allow storing the highest index (from all dimensions) of the timer array.

***Examples***

EXAMPLE 1: Timeout of a specific timer

t\_myTimer1.**timeout**; // checks for the timeout of the previously started timer MyTimer1

EXAMPLE 2: Timeout of an arbitrary timer

**any timer.timeout**; // checks for the timeout of any previously started timer

EXAMPLE 3: Timeout of a timer from a timer array

**timer** t\_myTimerArray[2][2];

**var integer** v\_i[2];

**any from** t\_myTimerArray**.timeout -> @index value** v\_i;

// checks for the timeout of any timer from array

// assigns index of matched timer to v\_i