Date: 2018-02-12

Author: Jasvir S. Dhillon, CarMaker Service Team Germany

Release No.: CarMaker 5.x - 6.x

## Background on StartProc and EndProc in Test Automation

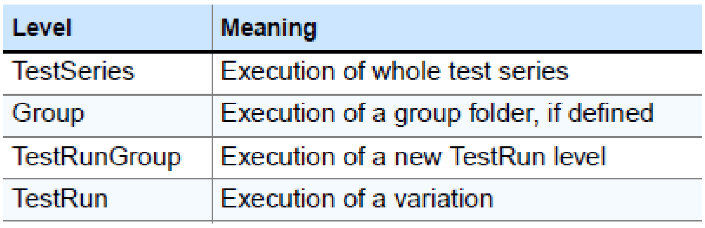
*StartProc and EndProc are special pre-existing procedures available for use inside the Test Manager. Once defined in a Script File, they are called during the execution of a given Test Series to perform their specific functions.*

*StartProc: This procedure is to be used when it is desired to perform any initialization steps before the intended simulation begins (in the pre-processing phase), for example declaring new variables, activation of data storage process, defining user-specific vehicle start velocities etc.*

*EndProc: When the user wishes to perform calculations of a characteristic value after the simulation gets finished and not during the simulation, the procedure EndProc is to be made use of. Hence, this provides a means of carrying out post-processing analysis of the stored data and hence is not limited to calculation of characteristic values that only depend on previous time-steps unlike when using RealTime Expressions.*

### 1. How can we define StartProc and EndProc using a ScriptFile?

These procedures need to be defined within a ScriptFile written in the Tcl/Tk programming language which can be loaded into the Test Space. The Start and End procedures can be called at all levels of a Test Series as seen in the figure below:



| Level | Meaning |
|---|---|
| TestSeries | Execution of whole test series |
| Group | Execution of a group folder, if defined |
| TestRunGroup | Execution of a new TestRun level |
| TestRun | Execution of a variation |

*StartProc*
The following code below shows a sample content of the ScriptFile when a StartProc procedure needs to be used. It is advisable to use this template as a reference or baseline for further modifications.

| | |
|---|---|
| Lines 1-3: | Log the content that the user wishes to read in the ScriptControl Console before the simulation begins. |
| Line 5: | Define the procedure name using which it would be called in Test Manager. |
| Lines 7-10: | These lines provide us with options to enter the user commands at different levels of the Test Series. |
| Line 12: | Run Driver Adaption for each variation of the TestRun (here a simple Braking test) |
| Line 14: | Save the TestRun |

```
1:  Log ""
2:  Log "New Run..."
3:  Log "Reading file [info script]"
4:
5:  proc MyStartProc {key name args} {
6:      switch $key {
7:          TestSeries      {Log "MyStartProc TestSeries:   $name"}
8:          Group           {Log "MyStartProc Group:        $name"}
9:          TestRunGroup    {Log "MyStartProc TestRunGroup: $name"}
10:         TestRun         {
11:                   Log "Initializing Driver Adaption..."
12:                   Driver adaptbasic 1 1
13:                   Log "Driver Adaption Ends..."
14:                   SaveTestRun "Braking Test"
15:                   }
16:                   default {}
17:     }
18: }
```

*EndProc*

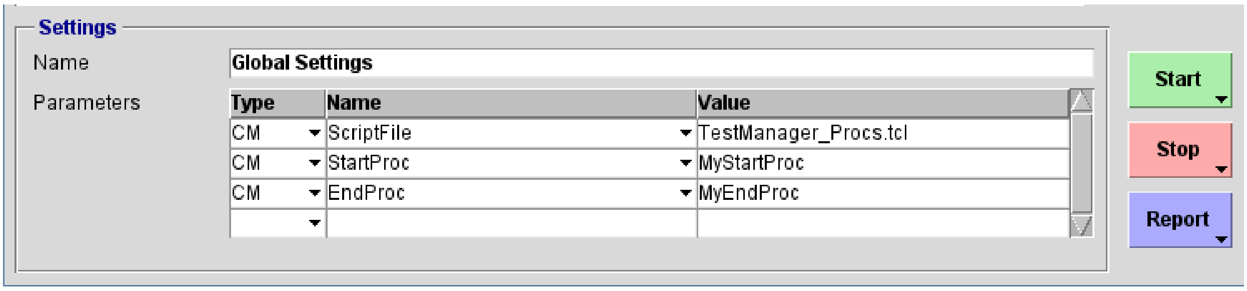
The following code shows an example of the ScriptFile when an EndProc procedure is required to be used. Using the same code template, this example shows us how a post-processing analysis can be performed using the result file from the simulation.

Line 11: Import Simulation Result File (Needs Main GUI>Storage of Results>Mode>Save All)
Line 12: Select quantities that need to be used for analysis
Lines 15-19: Perform Calculation (here, calculation of maximum longitudinal slip for front left wheel)
Line 24: Assign calculated value to Characteristic value

```
1:
2:  proc MyEndProc {key name args} {
3:      switch $key {
4:          TestSeries      {Log "MyEndProc TestSeries:   $name"}
5:          Group           {Log "MyEndProc Group:        $name"}
6:          TestRunGroup    {Log "MyEndProc TestRunGroup: $name"}
7:          TestRun         {
8:                      Log "Initializing Offline Analysis..."
9:
10:                     # Import Simulation Result File.
11:                     set fname [GetLastResultFName]
12:                     ImportResFile $fname {Car.LongSlipFL} Var
13:
14:                     # Perform Your Calculation
15:                     set MaxFL 0
16:
17:                     foreach {value} $Var(Car.LongSlipFL) {
18:                     set MaxFL [expr {max($value, $MaxFL)}]
19:                     }
20:
21:                     # Log calculated value Characteristic value
    in Script Control Console
22:                     Log [format "Maximum Longitudinal Slip FL:
    %.4f" $MaxFL]
23:                     # Assign calculated value of Max Slip to
    Characteristic Value
24:                     TestMgr::SetCharacteristicValue $MaxFL
25:         }
26:                      default {}
27:     }
28: }
```

## 2. How to define settings inside Test Manager to call the procedures?

- Once the code has been written, save the ScriptFile file with a '.tcl' extension, here 'TestManager_Procs.tcl'
- Open Test Manager > Load Desired Test Run > Add Variation(s)
- Go to Global Settings > Call the ScriptFile (.tcl), StartProc and EndProc procedures as shown in the figure below. By defining this in the Global Settings, the Start and End procedures get applied to each and every variation



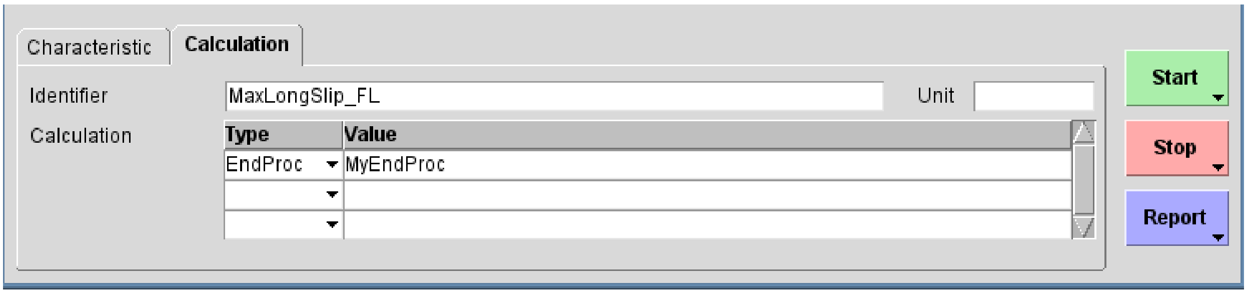
Settings

Name: Global Settings

Parameters

| Type | Name | Value |
|---|---|---|
| CM | ScriptFile | TestManager_Procs.tcl |
| CM | StartProc | MyStartProc |
| CM | EndProc | MyEndProc |
| | | |

- Add a Characteristic value and assign the EndProc calculation value to it as shown in the figure below. This value gets calculated for each variation and can be used to define the Criterion, if any used.



Identifier: MaxLongSlip_FL

Unit:

Calculation

| Type | Value |
|---|---|
| EndProc | MyEndProc |
| | |
| | |

The Test Manager has now been set-up to perform the simulation with the designated Start and End procedures.

Enjoy!