MTE Software Developer Guide - Config

By Edward Kaetz, Ver 12-22-2017

# Test Executive Config

The TE will read/setup configuration in the sequence of the sections listed below.

## Working Directory

When Test Exec launches it reads the Local Config file to determine where the working directory is located. This file is in the Config folder that is in the same folder where the Test Executive.exe it located:

C:\Program Files (x86)\Valve\Test Executive\Test Executive.exe

C:\Program Files (x86)\Valve\Test Executive\Config\Local Config.ini

The contents of this file are:

[Dir]

WorkingDir = "C:\MTE"

## Station Identity

The TE will assume the identity of the test station it is deployed on. It gets this info from the file:

C:\MTE\Test Executive\Test Executive Config.ini

An example of the contents of this file are:

[Station]

StationName = "Circle Test"

StationIPPort = 8500

StationExePath = "C:\Program Files (x86)\Valve\Circle Test\Circle Test.exe"

The Test Executive will put the StationName as the title on its front panel.

The Test Executive will use the StationName to locate the working directory for the local test station:

C:\MTE\StationName\

The Test Executive will launch the test station application at the path of StationExePath.

The Test Executive will connect to the station application using TCP-IP and the port StationIPPort.

## Working Folders

The top-level module (in this case Test Exec) will define the station working folders. If these folders do not exist, they will be created.

These folders will be provided to any child modules launched by the top-level module.

The working folders are:

* Configuration file location
* This is where any configuration files are stored.
  + The test station code may create subfolders in this folder to organize config info.
  + The TE will read from station config and test limit config files in this folder.

C:\MTE\StationName\Config

* Log file location
  + This is where the test station code may put local logs. The test station code may create subfolders in this folder to organize info.
  + The built in logger will place log files in this folcer.

C:\MTE\StationName\Logs

* Report file location
  + This is where the test station code may put local reports. The test station code may create subfolders in this folder to organize info.

C:\MTE\StationName\Reports

* Application Data location
  + This is where the test station code may put data that the operator should not tamper with.

C:\ProgramData\StationName

## Local Station Config

The local station config file is located at:

C:\MTE\StationName\Config\Local Station.ini

The following information is obtained from this config file:

* Identification (Section [ID])
  + The Station Name Part Number, Serial Number, and Revision
  + The info from this section is displayed in the top left corner of the TE UI.
* Factory Info (Section [Factory])
  + The Factory Location Identifier
  + The info from this section is displayed in the top left corner of the TE UI.
* Identification (Section [Facility])
  + Facility Config
    - This is the path to a file that contains facility name, address, and other info.
    - See section 1.7 for details.
    - The file may be located locally (same computer) or at a network location.
    - The info from this section is displayed in the top left corner of the TE UI.
  + Facility Modes
    - This is the path to a file that contains the test station operating mode names and options.
    - The file may be located locally (same computer) or at a network location.
* Station Config Info (Section [Station])
  + Station Config File
    - This is the path to a file that contains station configuration info.
    - See section 1.8 for details.
    - The file may be located locally (same computer) or at a network location.

Example contents of the Local Station.ini:

;This file provides configuration for the station.

;If this file does not exist the first time the application is launched it will be created.

; Parameters that need to be setup by the local terst engineer will be named UNDEFINED by default.

[ID]

Name = "Circle Test"

PN = "TBD"

SN= "001"

Rev = "-"

[Factory]

Location = "Solve1"

[Facility]

;The 2 items in this section point to the Facility identification and the Modes config files.

; Facility config - this is an ini file that dentifies the name and location of the facility

; Mode config - this is a JSON file that dentifies the modes and the options for each mode that are used at the factory.

FacilityConfig = "c:\MTE\Facility\Facility Config.ini"

FacilityModes = "c:\MTE\Facility\Modes.json"

[Station]

StationConfigFile = "c:\MTE\Facility\Station Config, Circle Test Solve1.ini"

[Sequence]

Default = "Full Test"

AutoLaunchEn = False

AutoLaunchDelay = 3

## Temp Data Folder

The TE will create the following folder if it does not exist.

C:\MTE\StationName\Data\Temp

This will be used for scratchpad data.

This path will be forwarded to the station module to use also.

## Facility Config

The Facility Config is an INI file.

It will contain:

* Section [ID]
  + Items:
    - Name - string
    - Location - string

Example:

[ID]

Name = "ATS\_TL"

Location = "Kirkland WA"

## Station Config

The Station Config is an INI file.

It will contain:

* Section [DataDirs]
  + Items:
    - ResultsLocalDir - path to folder
    - ResultsExternalDir – path to folder
    - Other Directory names – path to folder
      * There may be none, one or more other directories defined.
      * These are not used by TE.
      * They ae forwarded to the test station module.

Example:

;DataDirs section contains directories used by tests:

; LocalResultsDir - for TE Factory result reporting

; ExternalResultsDir - for TE Data Warehosue result reporting

; Other directories aer used by the test station module

[DataDirs]

ResultsLocalDir = "c:\MTE\Data\FactoryLogix\Circle Test\Solve1"

ResultsExternalDir = "c:\MTE\Data\Data Warehouse\Circle Test\Solve1"

CaptureOutDir\_1 = "c:\MTE\Data\Circle Test\Solve1"

CaptureOutDir\_2 = "c:\MTE\Data\Circle Test\Solve2"

CaptureOutDir\_3 = "c:\MTE\Data\Circle Test\Solve3"

SolveInputDir = "c:\MTE\Data\Circle Test\Solve1"

DataArchiveDir = "c:\MTE\Data\Circle Test\Archive"

DataSumArchiveDir = "c:\MTE\Data\Circle Test\Archive Summary"

CalibrationDir = "c:\MTE\Data\Circle Test\Calibration"

[Released\_SW]

ReleasedSWCheckEnable = TRUE

ReleasedSWCheckInterval\_s = 3600

ConfigFolder = "C:\MTE\Facility\Circle Test\Config"

SequenceFolder = "C:\MTE\Facility\Circle Test\Sequences"

ReleasedSWFile = "C:\MTE\Facility\Circle Test\SW\Releases SW Config.ini"

## Sequences

A sequence list is generated from the subfolder names of the sequences folder:

C:\MTE\StationName\Sequences

The sequence combo box on the lower left area of the UI will be populated with the sequence list.

The default sequence and the Auto run setup is read form the [Sequence] section of Local Station.ini

[Sequence]

Default = "Full Test"

AutoLaunchEn = False

AutoLaunchDelay = 3

The TE will automatically select the default sequence in the Sequence combo box on the UI. If the default sequence name is not in the list, then the top item will be selected.

If AutoLaunchEn is set to True or 1, the TE will start a timer for AutoLaunchDelay seconds and when that timer has elapsed, the default sequence will be launched.

## Released SW

When the TE launched or is reset it will read the Released\_SW section of the station config file.

Example Released\_SW section contents:

[Released\_SW]

ReleasedSWCheckEnable = TRUE

ReleasedSWCheckInterval\_s = 3600

ConfigFolder = "C:\MTE\Facility\Circle Test\Config"

SequenceFolder = "C:\MTE\Facility\Circle Test\Sequences"

ReleasedSWFile = "C:\MTE\Facility\Circle Test\SW\Releases SW Config.ini"

If ReleasedSWCheckEnable = TRUE, the TE will perform the SW checks and actions below. If it is set to FALSE, these checks will be skipped.

The TE maintains the time the last SW check was performed. When the TE is idle, it performs a check once a minute to see if it is time for another SW check.

* It subtracts the last SW check time from the current time and compares it to the ReleasedSWCheckInterval\_s interval.

When the ReleasedSWCheckInterval\_s interval has elapsed, the TE will:

1. Copy the Config folder specified by ConfigFolder to the current station working directory:

C:\MTE\StationName\

Any files will be over written (with the latest released config and test limit files).

Any files put in the config folder when the station was setup and are not part of the released files will not be disturbed.

1. Copy the Sequences folder specified by SequenceFolder to the current station working directory:

C:\MTE\StationName\

Any files will be over written (with the latest released sequence files).

Any files put in the sequence folder when the station was setup and are not part of the released files will not be disturbed.

1. Read the list of SW names and versions from the file specified by ReleasedSWFile.

The TE collects the name and version of each SW item used by the TE and the station application.

The ReleasedSWFile will contain a section named Released\_SW\_Versions. In this section will be a list of names and versions. The SW names contained in the INI file will have any spaces replaced with underscores. The TE will do the same to the SW names to get a correct match.

Example ReleasedSWFile content:

'List SW name and version info.

'Substitue an underscore ('\_') for any space in the SW name.

[Released\_SW\_Versions]

Test\_Executive = V1.0.0.1

Circle\_Test = V1.0.0.31

All SW names will be compared to the list of names in the ReleasedSWFile. If there is a match, the TE will compare the versions. If they are not the same, the TE will indicate a warning.

## Mode Config

XXX

## Released SW File Info

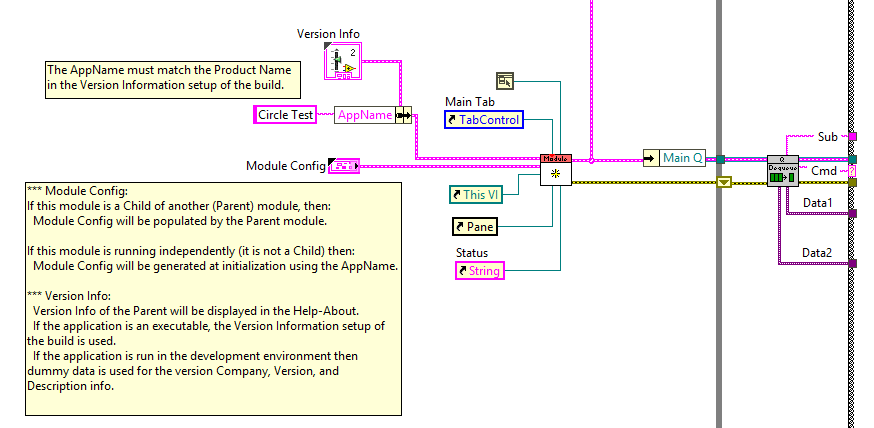
XXX

# Station Module Configuration

The Test Station Module Exec obtains its working folders in a similar way that the TE does. All top level applications will behave the same.

## Station Module Name

The station module name is hard coded in the left side of the block diagram. It is the AppName in the Version Info cluster. It is referred to as the StationName it he is following sections of this document. This name will be used to determine the working folders.



## Working Directory

When Test Station Module Exec launches it reads the Local Config file to determine where the working directory is located. This file is in the Config folder that is in the same folder where the Test executable file is installed.

C:\Program Files (x86)\Valve\StationName\StationName.exe

C:\Program Files (x86)\Valve\StationName\Config\Local Config.ini

The contents of this file are:

[Dir]

WorkingDir = "C:\MTE"

## Working Folders

The top-level module (in this case Station Module) will define the station working folders. If these folders do not exist, they will be created.

These folders will be provided to any child modules launched by the top-level module.

The working folders are:

* Configuration file location
* This is where any configuration files are stored.
  + The test station code may create subfolders in this folder to organize config info.
  + The TE will read from station config and test limit config files in this folder.

C:\MTE\StationName\Config

* Log file location
  + This is where the test station code may put local logs. The test station code may create subfolders in this folder to organize info.
  + The built in logger will place log files in this folder.

C:\MTE\StationName\Logs

* Report file location
  + This is where the test station code may put local reports. The test station code may create subfolders in this folder to organize info.

C:\MTE\StationName\Reports

* Application Data location
  + This is where the test station code may put data that the operator should not tamper with.

C:\ProgramData\StationName

# Station Module Development

## File Naming Convention

LabVIEW projects cannot handle duplicate file names. All LabView files created for your application should be named:

StationName.Function.suffix

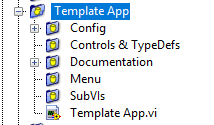
where the suffix (.vi, ctl, rtm…) is the appropriate for the file type.

You can use an abbreviation for the StationName prefix of each file like use TestExec for Test Executive.

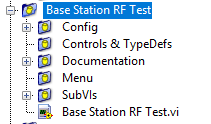
This section assumes the station module is being developed from a template.

Copy the template folder and rename the folder and the top level VI to your station name.

Copy From:



Rename To:



All subfolders will be empty except for the Menu folder. Rename the menu file to ‘StationName.menu.rtm’

The template application VI has all the framework needed to function with the test executive. All you need to do is add functions.

## Top Level Application VI Architecture

### Queued State Machine

All queues use the same common data type. The Queue element is a cluster of:

* Cmd – String, top level state
* Sub – String, next level state
* Data1 – Variant, data to be passed to the case.
* Data2 – Variant, additional data to be passed to the case.

Producer - Event Handler Thread

Handles UI events. Enqueues tasks that is performed by the Main thread.

Consumer - Main Thread

Top Level Cases:

* Init – this is run only once at launch
* Reset – this may be run multiple times to reinitialize the application
* Task – this contains sub states. This handles activities the application must perform.
* Function – these are tasks that can be called by external modules (other high-level modules running in parallel to this module). Functions must return data along with an ACK or NAK.
* Error – This handles errors
* Exit – this performs cleanup and shut down functions

A small application has a modest number of tasks to perform and one Task state is sufficient. However, applications tend to grow in complexity. An application can have other task cases with their sub cases. A standard naming convention for these cases is used:

* App – used for application supervisory tasks
* Seq – used for sequencing activities
* Log – used for logging activities
* Instr – used for instrument activities

## Functions

### Calling a Function

A function is called in the TE script with the command:

Sta.FunctionName(Arguments)

The ‘Sta.’ prefix tells TE to forward the function call to the test station module.

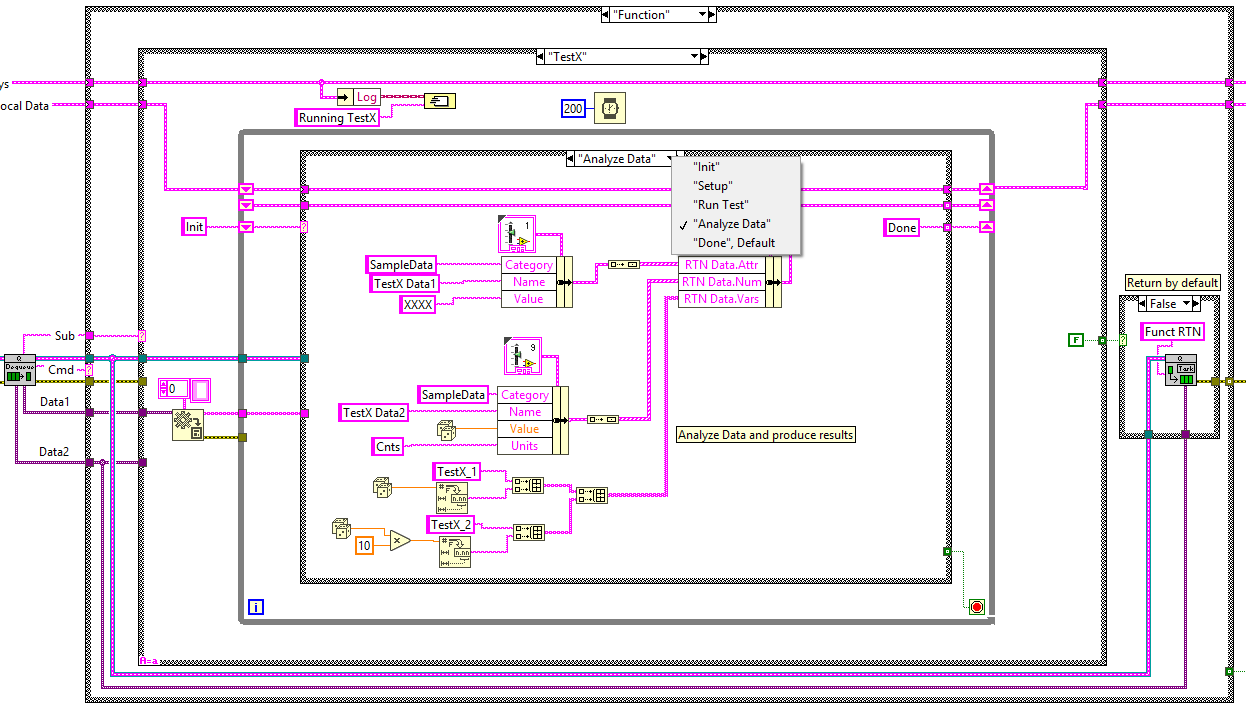
FunctionName is the name of the sub case in the Function case that will be called.

### Function Execution

The Case Function -> FunctionName will be run.

TestX is an example function.

It handles setup, run test, analysis and wrap up steps.



### Returning Data

The Local Data cluster has a ‘RTN Data’ element. This contains arrays of data that will be returned to the Test Executive:

* Result Attributes – these are String type results
* Result Numeric – these are number type results
* SW Items – these are the name and version of SW used. It gets reported to the TE to be put in the test report.
* Variables – these are name/value pairs. All variables are strings. They will be converted to the needed type when they are operated on. They will be available to be processed by the test script.

The function should populate the data in the ‘RTN Data’ item. When the function is done, it will call the ‘Task -> Funct RTN’ which will send this data to the TE.

The Data2 of the function call contains return routing information and must be sent to the Data2 of the Funct RTN case.