

### Test Executor

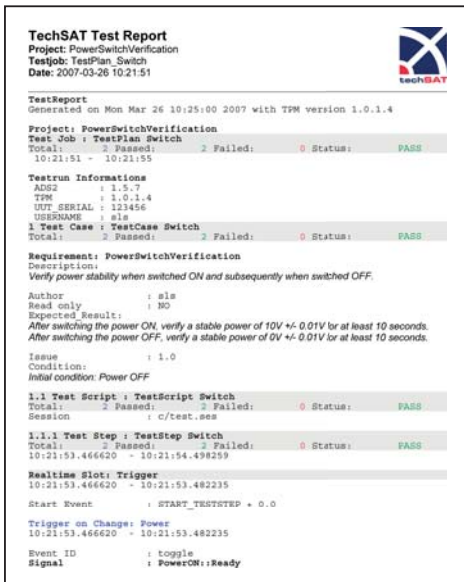
The test executor runs the test executable using the underlying execution engine. For debugging purposes, the executor also provides the ability to pause and step through the test execution at specified time interval steps. Subsequently realtime execution can be resumed or cancelled.

Upon successful execution the test executor generates the test result which is automatically fed into the test reporter for report generation. Additionally during execution the test executor logs all relevant signals including their absolute timestamps.

### Test Reporter

The test reporter automatically generates test reports that are displayed in the integrated viewer. In addition, the test reports can be exported to either HTML or PDF file format. Furthermore, the test reporter allows configuring various properties of the report generation such as the layout and level of detail.

Figure 3. Test result report excerpt



### Software and Hardware Support

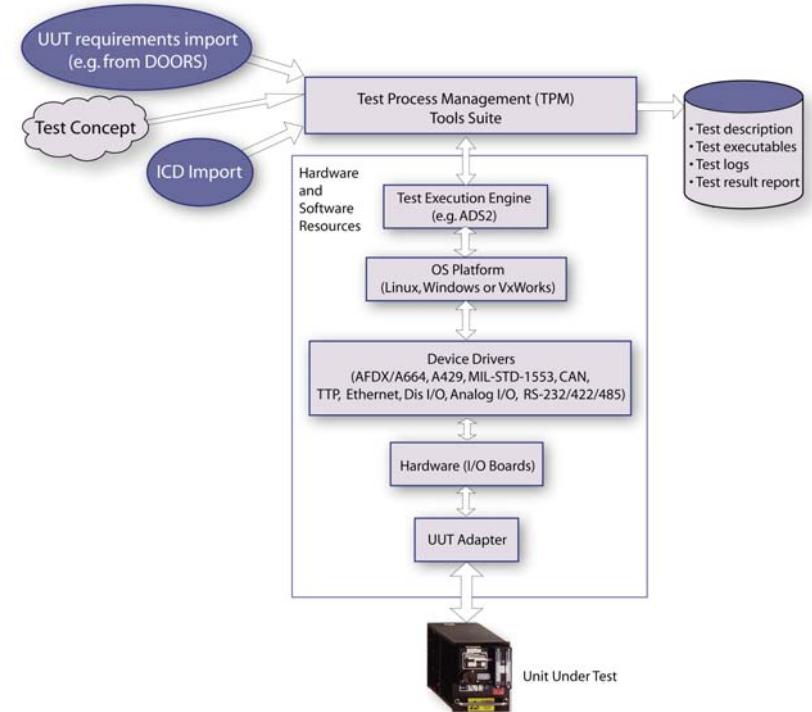
Currently, TPM uses ADS2 as test execution engine, thereby making use of the ADS2 platform's powerful features such as realtime execution, data logging etc. Using ADS2 as execution engine also means that TPM runs on all OS platforms supported by ADS2, i.e. Linux, Windows and VxWorks.

TPM is capable of testing validating and certifying many types of electronic equipment. Since TPM itself is designed to be entirely independent of the UUT's I/O type(s), any limitation in terms of I/O support is defined by the underlying execution engine. Using ADS2 as execution engine will provide a wide range of supported I/O types which are listed in the following:

- > AFDX®
- > ARINC 664/787 AFDX (Boeing)
- > ARINC 429
- > Analog I/O
- > CAN
- > Discrete I/O
- > GPIB
- > LVDT / RVDT I/O
- > MIL-STD-1553
- > RS-232 / 422 / 423 / 485
- > RSS (Resistor Sensor Simulation)
- > Synchro / Resolver
- > TTP (Time Triggered Protocol)

### Distributor

## Test Process Management (TPM) Software Tools Suite



- Designed for testing, validating and certifying electronic equipment
- Ideally suited for integration, production and maintenance testing
- Complete management of the entire test life cycle
- Simplified programming with intuitive Visual Programming Language (VPL)
- Realtime test execution engine
- Automatic test execution and test report generation

# Test Process Management (TPM) Software Tools Suite

## Overview

Next generation aircraft are required to be fuel efficient and feature advanced technology. This requires an increased amount of avionics equipment with more and better functionality while reducing weight and size. Inevitably this leads to increased avionics complexity in general; a complexity the test engineer has to be able to deal with when testing, validating and certifying avionics equipment. At the same time, the test engineer has to perform his work in a shorter time frame due to ever shorter aircraft development cycles.

As a consequence of this, the test engineer is faced with challenges that can only be overcome with advanced ATE (Automatic Test Equipment) providing comprehensive tools supporting the test engineer during the entire test phase. These tools must be highly flexible, able to perform realtime testing as well as offering fully automatic test execution and test report generation.

To fulfill these requirements, TechSAT provides its Test Process Management (TPM) software tools suite. TPM has been specifically designed and developed to cover all aspects of testing, validating and certifying electronic equipment. This means that TPM provides all the necessary tools for planning, creating, executing and documenting tests. The application areas of TPM are integration, production and maintenance testing.

Upon importing the equipment ICD (Interface Control Document) as well as the system requirements from e.g. DOORS, (Dynamic Object Oriented Requirement System) the test engineer can implement test cases intuitively using the integrated Visual Programming Language (VPL) of TPM. Based on the test cases, a test compiler will build a test executable as well as a test description. A test executor will execute the test, perform concurrent data logging of all relevant signals and finally generate a test report based on the test description and the test result. The test executor depends on an underlying realtime test execution engine which could be TechSAT's Avionics Development System - 2nd Generation (ADS2). However, since TPM is portable, other test execution engines are possible.

## Test Editor

As depicted in Figure 1, the test editor is the principal tool in the test creation phase. The test editor organizes all tests in test projects which can be easily created and edited.

At the beginning of the test phase, system requirements must be gathered, organized and linked with their corresponding test cases that verify the system requirements. For this purpose the test editor provides an import facility that can import system requirements in either CSV (Comma Separated Value) or XML (Extensible Markup Language) format. The test editor organizes the imported system requirements in a tree structure of test cases each referencing their corresponding system requirement. The import function ensures that the system requirements structure is kept identical to that of the source (e.g. DOORS).

The test editor also facilitates imports of system ICDs to an SQL database. A signal browser serves as a means to manage the database signals including searching for signals which can then be inserted into the test cases via drag and drop with the mouse.

In order to simplify and speed up the process of implementing the test concepts of the test engineer, the test editor provides an integrated and intuitive Visual Programming Language (VPL).

## Visual Programming Language (VPL)

As shown in Figure 2, the principal graphical programming element of the VPL is the **Test Case**. The **Test Case** references the system requirement and can be divided in several **Sub Test Cases**. Each **Sub Test Case** holds one or more **Test Scripts** consisting of a **Prologue** and **Epilogue** as well as one or more **Test Steps**. The **Prologue** and **Epilogue** elements are executed in non-real time and are used to e.g. load and unload resources, respectively. The **Test Step** consists of one or more event controlled **Realtime Slots** executed either in parallel or sequentially. Each **Realtime Slot** contains statements (e.g. Verify statements, value assignments etc.) that are always executed in parallel.

The VPL also features a user code interface allowing the user to integrate customized C code using templates.

## Test Compiler

The test compiler is an integrated part of the test editor and enables the user to compile and build test cases into an executable. In an intermediate step, the test compiler generates C code from the test cases implemented in the integrated VPL and combines it with customized C code, if defined.

The test compiler also generates the test job which is a detailed test description used as the basis for the test result report.

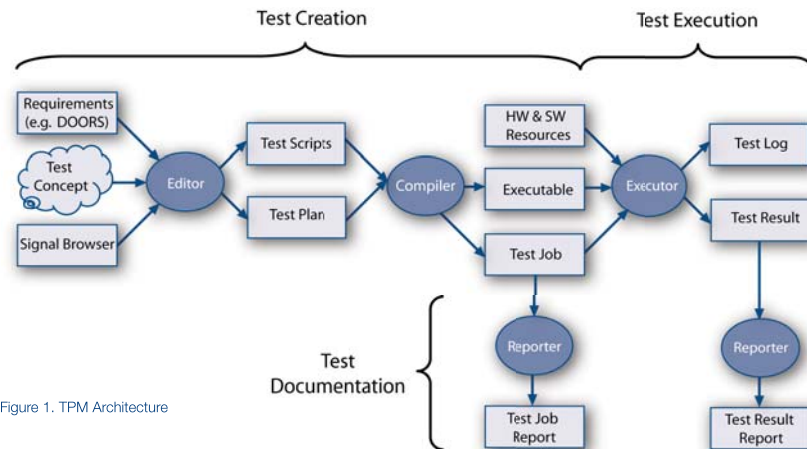


Figure 1. TPM Architecture

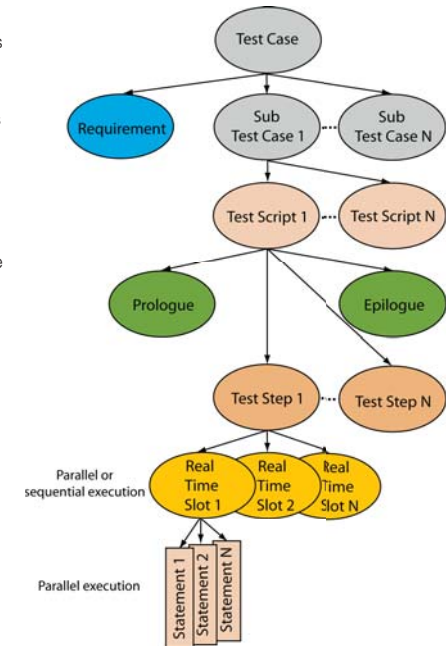


Figure 2. Visual Programming Language structure.