ALL4TEC
MODEL BASED TESTING SOLUTIONS

ALL4TEC
MODEL BASED TESTING SOLUTIONS

Model-based the gap from the **functional requirements** to TestStand **test sequences**!

Abstract:

Validation still need optimization and tools support to efficiently deal with today complex systems. TestStand helps by productively automating the test execution. Nevertheless, SUT quality remains highly dependent of the test cases that are performed.
We will show how **MaTeLo can help test designers to optimise** their test campaigns by using a **Model-based approach with TestStand**:

- MaTeLo offers a **graphical way** to formalize the requirements test specification based on a usage model that makes the **links between Doors, TestStand**.
- MaTeLo generates automatically different kind of TestStand test sequences and systematically **improve the SUT reliability, reduce the testing cost and duration**

Model-based testing

Model-Based Testing is the application of Model-Based Design for software testing; this concept has been adopted by many companies and integrated into both hardware and software systems testing. The process typically consumes functional requirements as inputs, and produces test cases. The test cases generation is done according to a model describing the required behavior of the system. The testing effort is made by test engineers during the design of the usage model. The effectiveness of Model-Based Testing is highly increased with automation, i.e. the capacity to automatically translate test cases into executable test material.

MaTeLo model

MaTeLo – Markov Test Logic – is implementing the Model-Based Testing concept using the usage statistical testing approach. Testers design a usage model, from their knowledge of the system under test and the system requirements specification documentation. This testing model summarizes system test cases according to its final use.

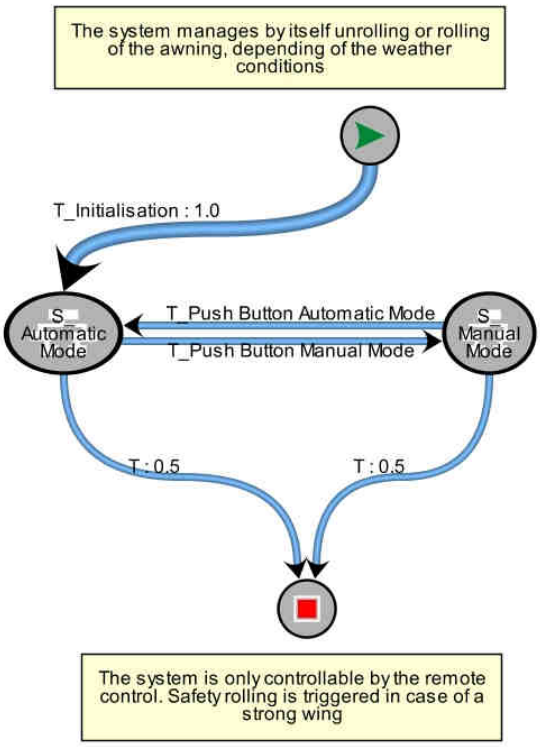

Figure 1: MaTeLo Model

Importing requirements

The very first model design step is to import system requirements from the requirements managing tool into the MaTeLo requirement library. This action can be performed in several ways depending on the tool used. We offer a Doors plugin for both import and update of the requirement library. Once the requirements are imported, they can be associated with models items.

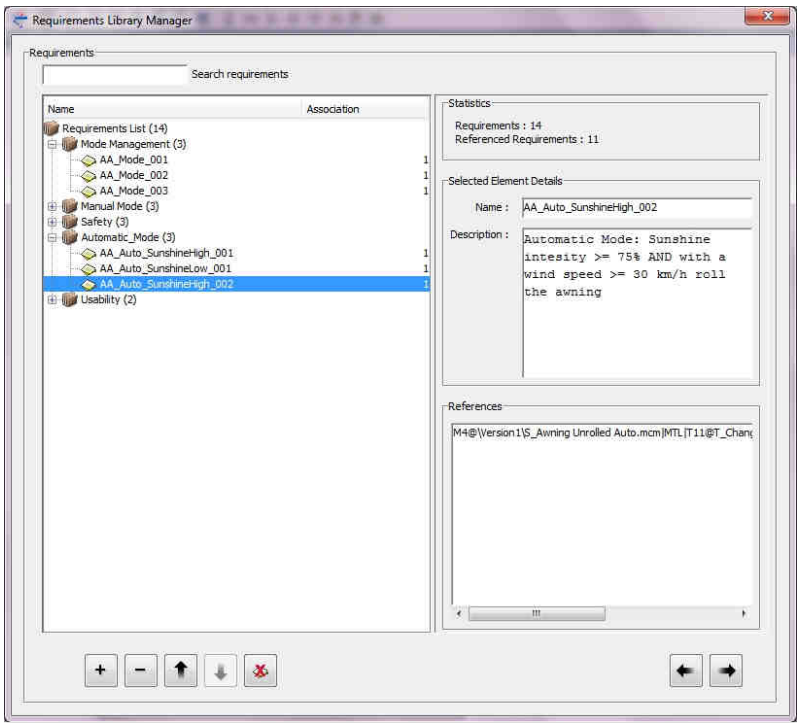

Figure 2: Requirement library

MaTeLo model items

A MaTeLo usage model is a state diagram, composed of extended Markov chains. It thus contains transitions and states, and probability layers – aka profiles.
States can be seen as stable testing states. I.e. if the system is in a state and no action is applied, the system stays in this state. Transitions are actions that allow the system to change state. Each transition may wear one “stimulation input” and several “expected results”. “Stimulation input” describe system inputs classes of equivalence using statistical repartition laws, whereas “expected results”, used to check the system behavior
Profiles are probabilities sets that allow the designer to statistically characterize the system use. They have two axis of probability; inputs law of repartition and transitions weight.

Association of requirements and test cases coverage

Requirements may be associated directly to a transition. They also can be associated with any input or any expected result of your model. They can also be associated with a specific class of equivalence of data.
A specific requirement can be associated with several model items. Each requirement association has thus a type that can be whether “Necessary”, “Sufficient” or “Necessary and sufficient”. This type is used to determine the requirement coverage in case they have been associated with several items of the model.

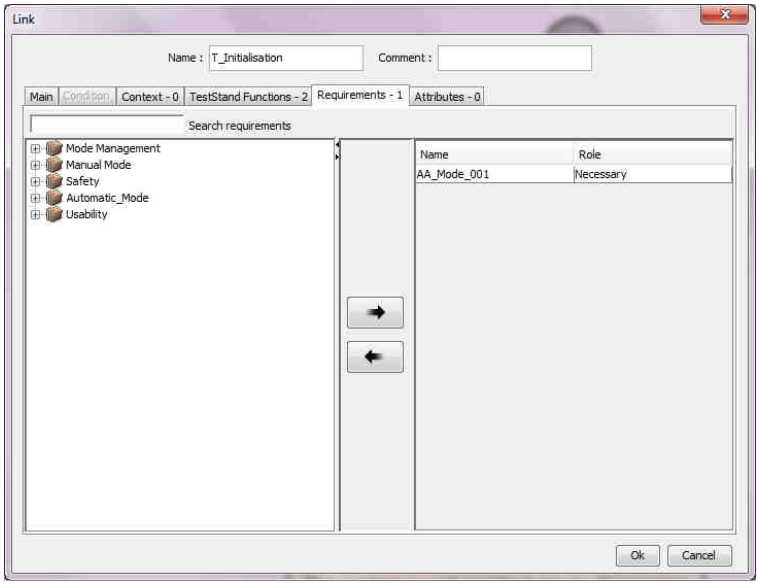

Figure 3: Association of a requirement with a transition

Once the requirements have been associated with a model item, references between requirements and items are automatically created. Such references allow the designer to quickly check which part of the model by double clicking them. The update functionality of the Doors plugin tags requirement modifications and deletion. Using references on such tagged requirements allows knowing which parts of the model are impacted by requirement modifications. Modification of the model is thus made easier.

MaTeLo and automation with TestStand

In addition to requirements associations, test robot API can be associated with model transitions and fed with model input or expected results values. MaTeLo natively interfaces TestStand solution from National Instruments. The first step is to declare test API functions in MaTeLo.

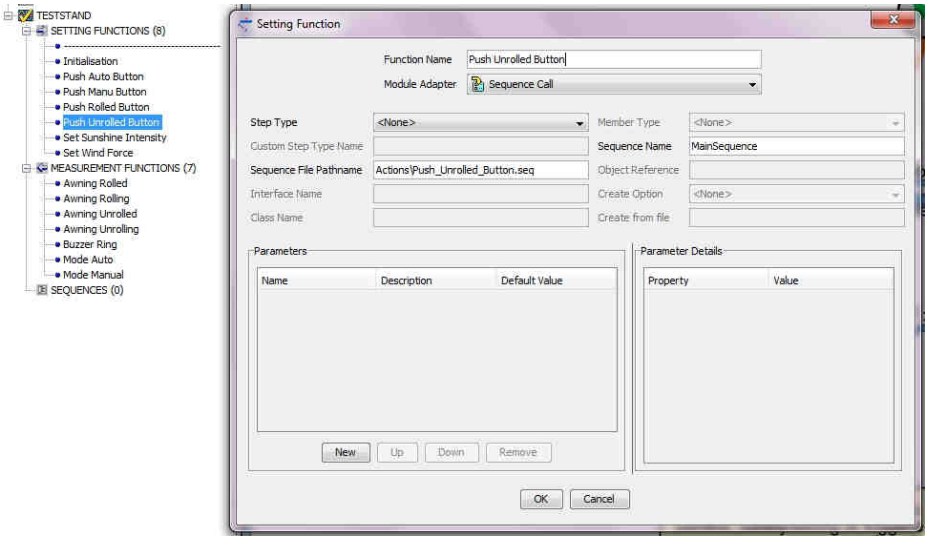

Figure 4: TestStand API function declaration

Declaring a test API function is easy and does not require any knowledge of the test function implementation. The tester fills in information that is typically given by test APIs documentation, like function adapter and code module name. He also indicates the test function prototype. This API functions declaration is done once, and can then be re-used from one project to another.

With test functions API declared in MaTeLo, one can associate them with transitions. Several function can be associated with one single transition and their call ordered. Parameters are assigned with values from the model, i.e. inputs for settings functions and expected results for measurements functions.

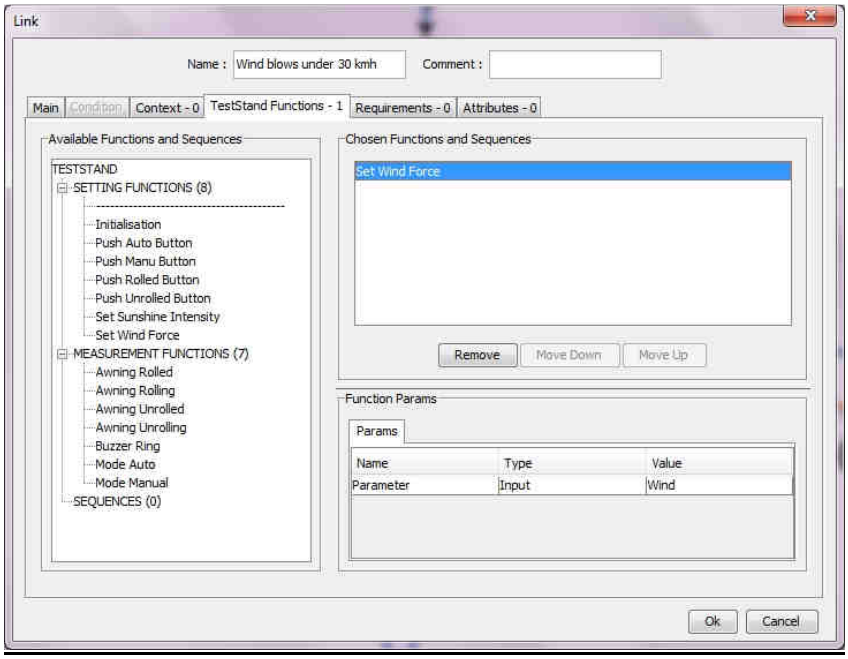

Figure 5:TestStand API function association

Test cases generation and requirement traçability

Once the model is completely associated with requirements and test bench API, the MaTeLo test cases generator is used to run through the model, from the initial state to the end state. Each run thought the model is a test case. Test cases are provided in XML format, and also translated automatically into ready-to-run TestStand sequences.

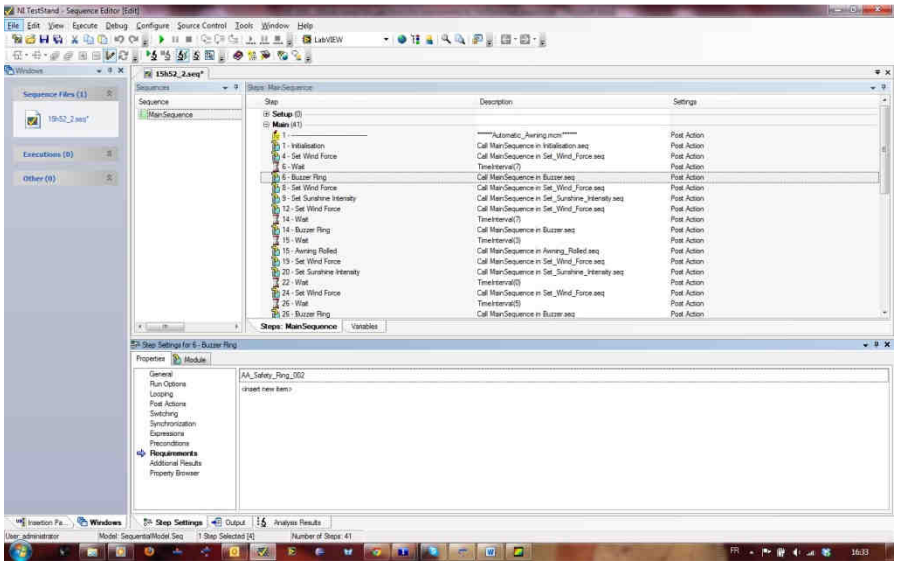

Figure 6: Example of TestStand generated sequence

Testor also generates requirements matrix traceability. The tester knows which requirements are covered by each test case and also which test case covers a specific requirement. Additional indicators are provided as outputs of the test cases generator, like coverage of transitions, states and classes of equivalence.

ALL4TEC
MODEL BASED TESTING SOLUTIONS

Conclusion

MaTeLo tool is dedicated to testers; its easy graphical test cases description actually does not require the knowledge of any development language. The tool can be used to track down requirements during the whole embedded systems testing process, from the requirement managing tool to the execution with TestStand. Traceability matrixes are provided by the tool, and allow knowing which test cases cover which requirements. Execution campaigns can further be managed by the last MaTeLo tool called Test Campaign analysis. This last tool can be used to provide additional indicators, especially cumulated requirements coverage over time and software reliability estimation.

For more information, please contact us: matelo@all4tec.net or 02 43 49 75 30