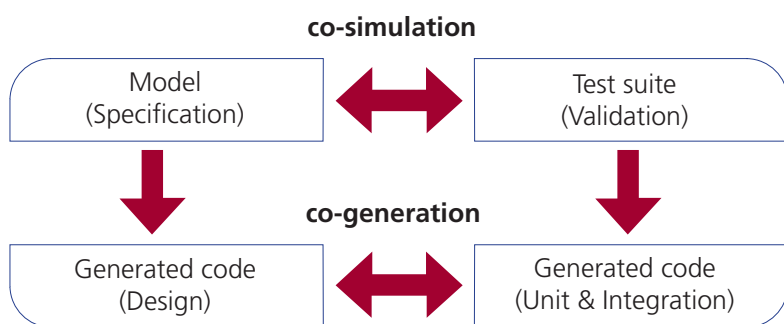


# Testing the specification as well as the implementation



- MODEL DRIVEN TESTING
- CONTINUOUS INTEGRATION
- IMPLEMENTATION INDEPENDENT TEST SUITES
- LEGIBLE TEST CASES
- GRAPHICAL EXECUTION TRACES

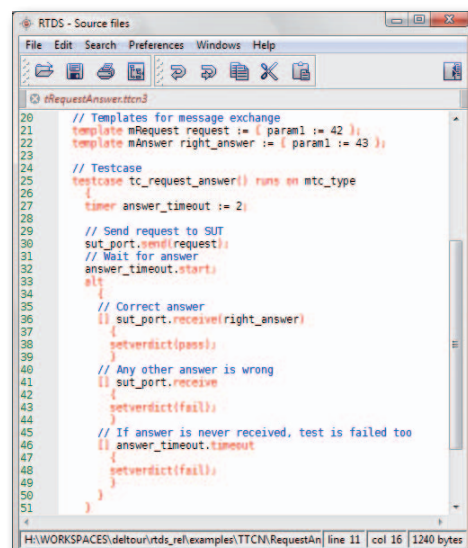
## STANDARD

PragmaDev Real Time Developer Studio implements ITU-T standard testing language **TTCN-3** (Testing and Test Control Notation). TTCN-3 has been designed to test all types of systems whether they are event based (asynchronous) or function based (synchronous). **It has been successfully used the last 15 years** in the industry for testing and certification in various domains such as Telecommunication, Automotive, Railway, and Finance.

## DEDICATED

TTCN-3 is basically a programming language with dedicated test features:

- It has its own data types or it can use external data types defined in ASN.1.
- It embeds a number of basic services such as timers, messages, parallel testing.
- The concept of templates allows to match very complex information in a single line.
- It embeds the concept of verdict and defines rules to aggregate verdicts coming from different test cases.

```

20 // Templates for message exchange
21 template mRequest request := ( param1 := 42 );
22 template mAnswer right_answer := ( param1 := 43 );
23
24 // Testcase
25 testcase tc_request_answer() runs on mtc_type
26 {
27     timer answer_timeout := 2;
28
29     // Send request to SUT
30     sut_port.send(request);
31     // Wait for answer
32     answer_timeout.start;
33     alt {
34         // Correct answer
35         || sut_port.receive(right_answer)
36         {
37             setverdict(pass);
38         }
39         // Any other answer is wrong
40         || sut_port.receive
41         {
42             setverdict(fail);
43         }
44         // If answer is never received, test is failed too
45         || answer_timeout.timeout
46         {
47             setverdict(fail);
48         }
49     }
50 }
51
  
```

A simple test case

Some test equipments support TTCN-3 natively including a TTCN-3 compiler and the adapters to the testing equipment interfaces.

## ABSTRACT

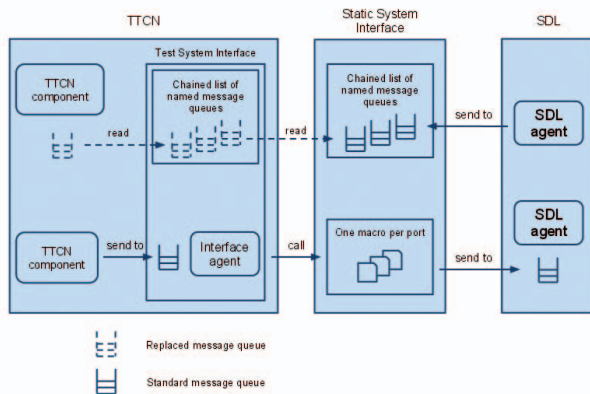
TTCN-3 is a high level testing language that can be **used to test a high level model of your system**. As PragmaDev Real Time Developer Studio supports different modeling technologies such as UML, SDL-RT, and SDL, **the implementation of TTCN-3 in our tool allows to test the model very early in the development process**. In that case the tool will generate the interface between the System Under Test and the Test Suites.

TTCN-3 is also a **very detailed testing language** that can test a software or even a hardware implementation of a system. TTCN-3 defines the concept of Test System Interface in order to adapt the test suite to any kind of system.

## INTEGRATED

PragmaDev Real Time Developer Studio offers **three levels of modeling** from informal, semi-formal, to formal. Formal modeling allows to verify the dynamic behavior of the model with the built-in simulator without generating any code. On the test side, the same simulation feature is available so that the simulated model can be tested against a simulated test suite.

Once a high level model is verified, code can be generated for implementation on a real target. Once again, it is possible to generate code out of the test suite to be executed on a real tester and test the implementation of the system. PragmaDev code generator provides integration for the **most popular real time operating systems** and the model debugger has interface with several debuggers and cross-debuggers in order to graphically debug in the model. Integration features and debugger interfaces are available for both the code generated out of the model as well as from the test suite. Last but not least test execution can be **graphically traced with PragmaDev MSC Tracer**.

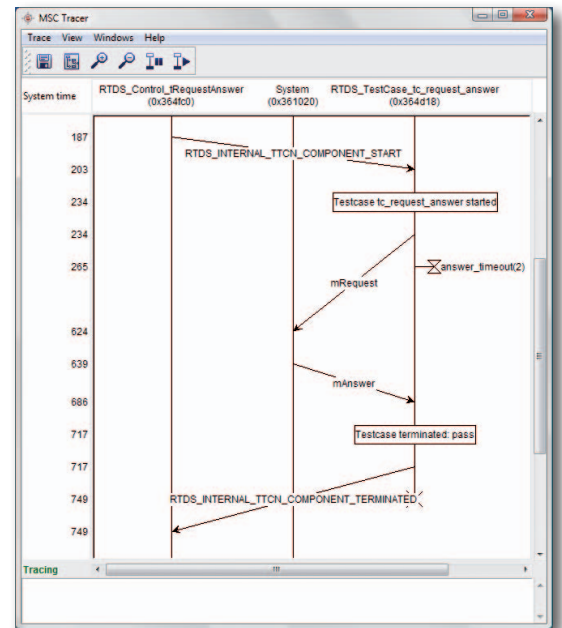
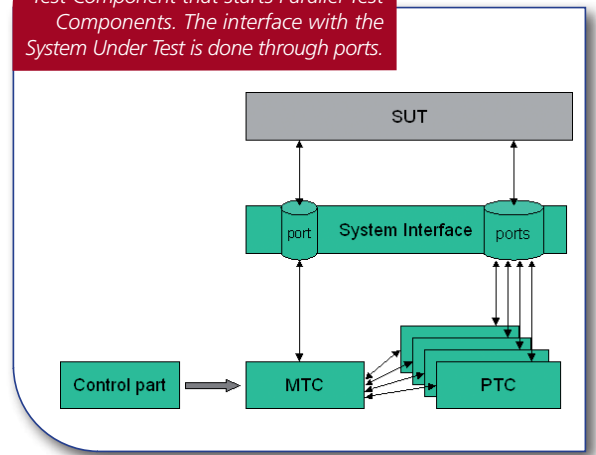


Easy integration of the generated code with the System Under Test

## MODEL DRIVEN TESTING

PragmaDev has setup a technical partnership with Verimag laboratory in Grenoble that has a well recognized expertise in model checking technologies. Integration of that model checking technology allows to automatically generate a set of test cases based on objectives defined by rules graphically described in observers. Once the objective has been reached, the generated test case includes: the data types, the template definitions, the scenario.

TTCN-3 testing architecture: the control part launches the Master Test Component that starts Parallel Test Components. The interface with the System Under Test is done through ports.



Execution trace of a test case

