

CANoe - the premier solution for professional ECU test

Siegfried Beeh
Vector Informatik GmbH



Siegfried Beeh studierte von 1989 bis 1995 Elektrotechnik an der Universität Stuttgart. Nach dem Diplomabschluß war er vorrangig tätig in der Entwicklung eingebetteter Systeme mit und ohne Sicherheitsverantwortung. Weiterhin beschäftigte er sich mit der Definition und Umsetzung von Prozessen. Seit 2002 ist er bei Vector als Senior Software Development Engineer tätig. Herr Beeh ist u.a. verantwortlich für die Entwicklung des "Test Feature Set" in CANoe/DENoe.

Between 1989 and 1995 Dipl.-Ing. Siegfried Beeh studied Electrical Engineering at the University of Stuttgart. Afterwards he was mainly involved in the development of Embedded Systems with and without secure responsibility and the definition and implementation of processes. Since 2002 he has been working for Vector as Senior Software Development Engineer and inter alia he is responsible for the development of the "Test Feature Set" in CANoe/DENoe.

CANoe – The premier solution for professional ECU test

Vector Informatik GmbH



The Vector Group

Vector's "Mission Statement":

Vector provides OEMs and suppliers of automotive and related industries with a professional and open development platform of tools, software components and services for creating embedded systems.



Vector Group
480 employees
Date: February 2005

Subsidiaries:

Vector Informatik, Stuttgart, Germany

Vector Japan, Tokyo and Nagoya

Vector CANtech, Michigan, USA

Vector France, Paris

Vector Scandinavia, Gothenburg, Sweden

Vector Consulting, Stuttgart, Germany

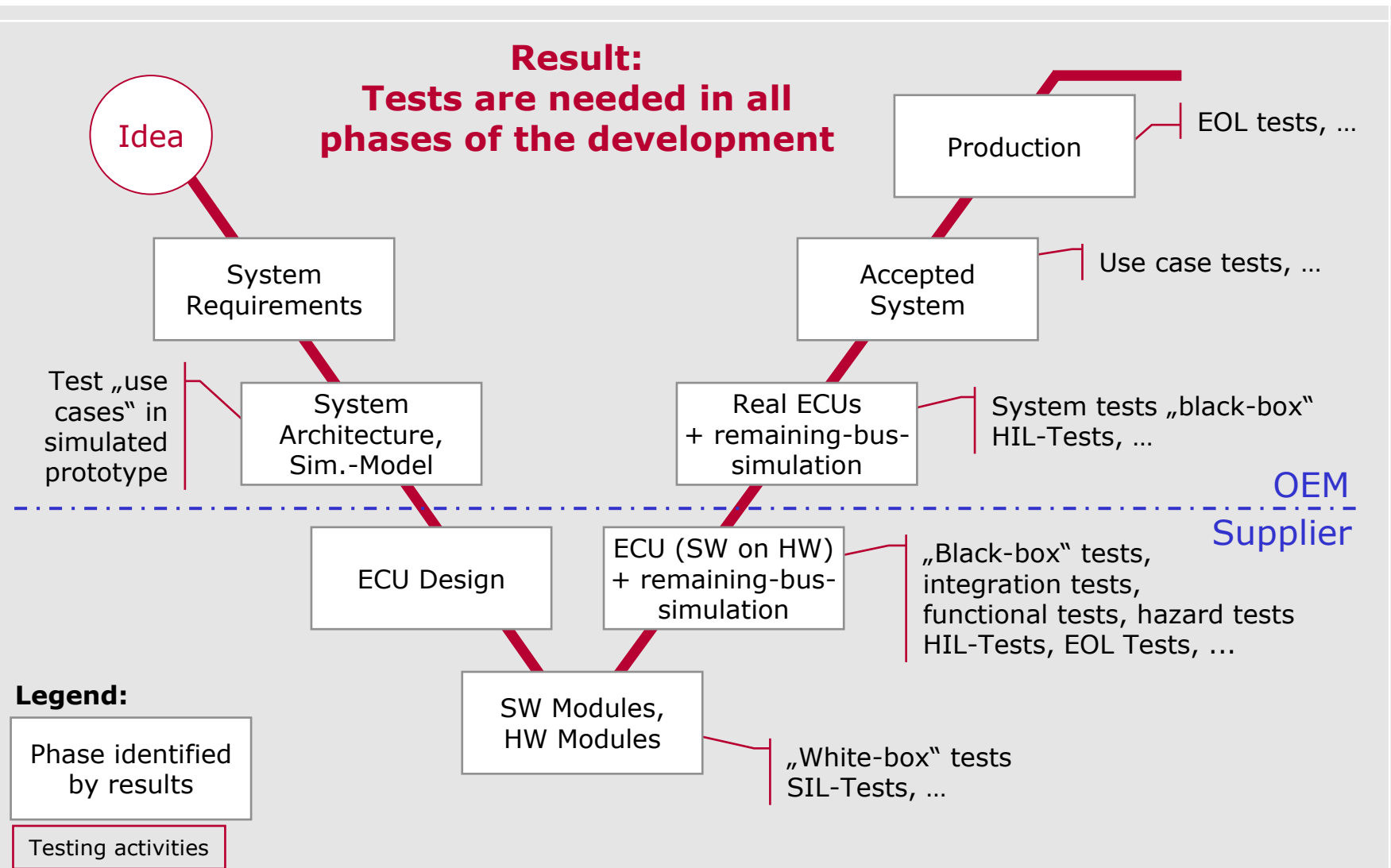


Testing with CANoe: Agenda

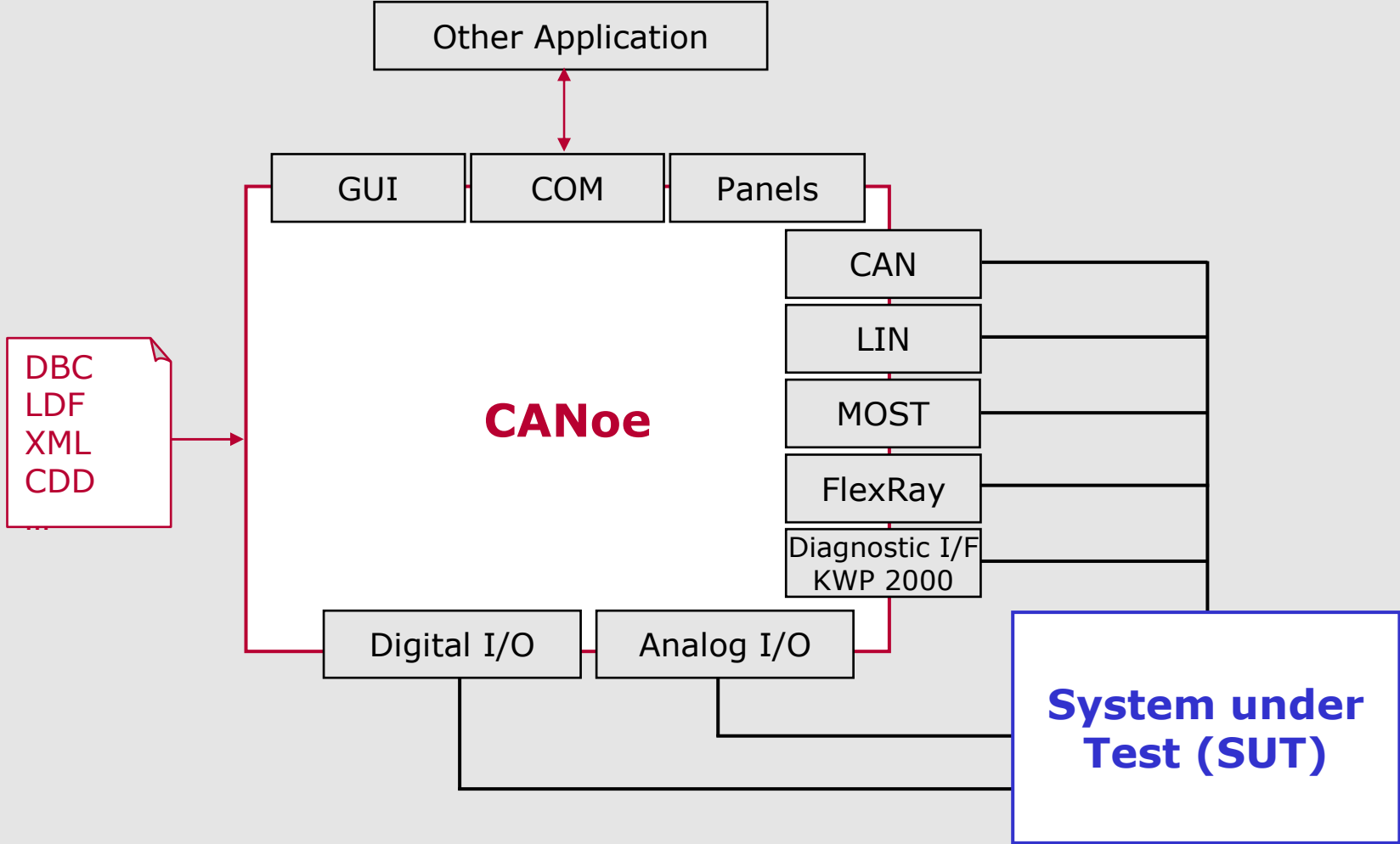
- ❑ Motivation: Product life cycle with testing activities
- ❑ White box test, integration test-, black box test capabilities
- ❑ "Specification" instead of "programming" - concepts
- ❑ Generation of tests
- ❑ Typical use cases within testing projects – test management

- ❑ Conclusion
- ❑ Future perspective

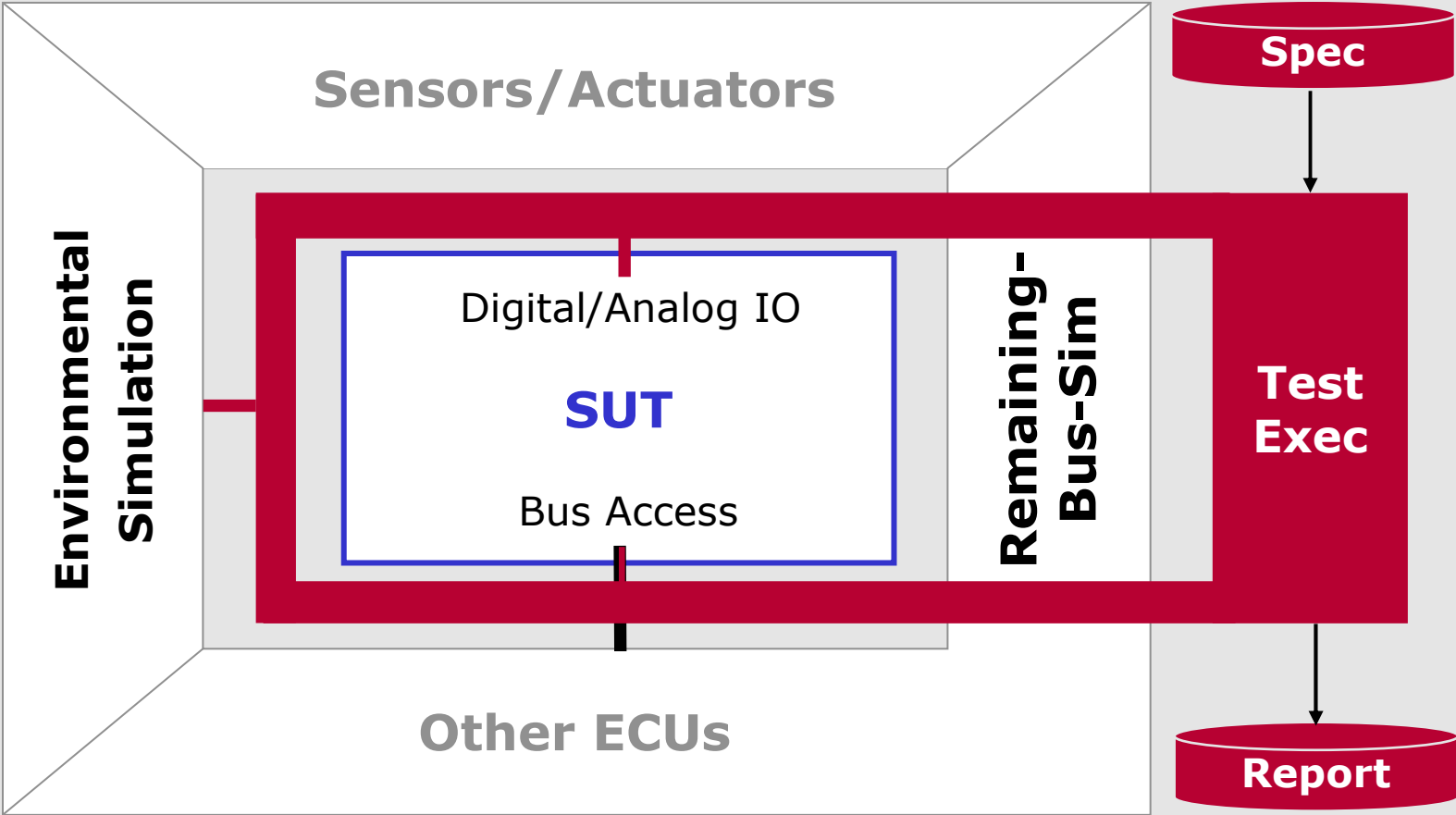
Motivation: Testing Activities in the Product's Life Cycle (simplified)



Testing with CANoe: System Under Test

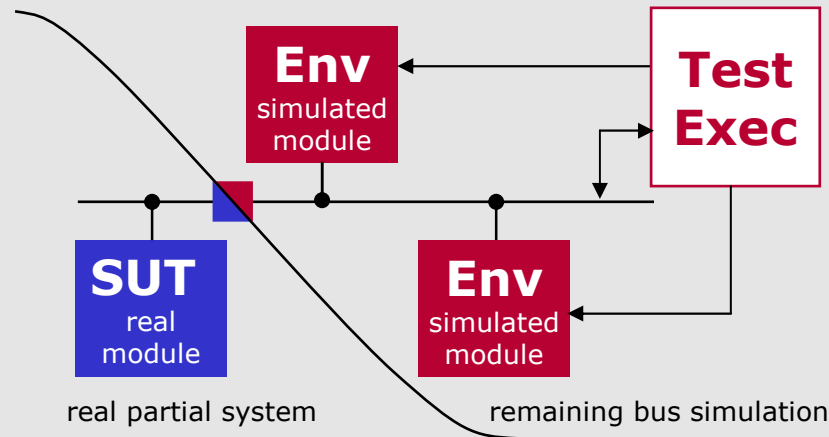


Testing with CANoe: Simple Test Arrangement – Proceeding for Test

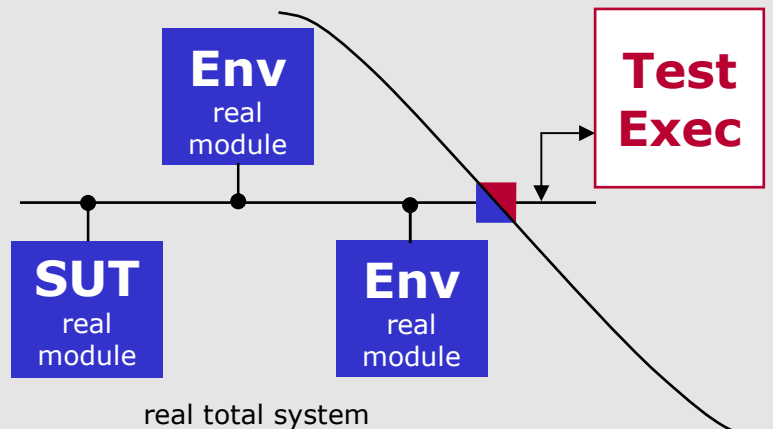


Testing with CANoe: Integration Testing Capabilities

Test of SUT in simulated environment

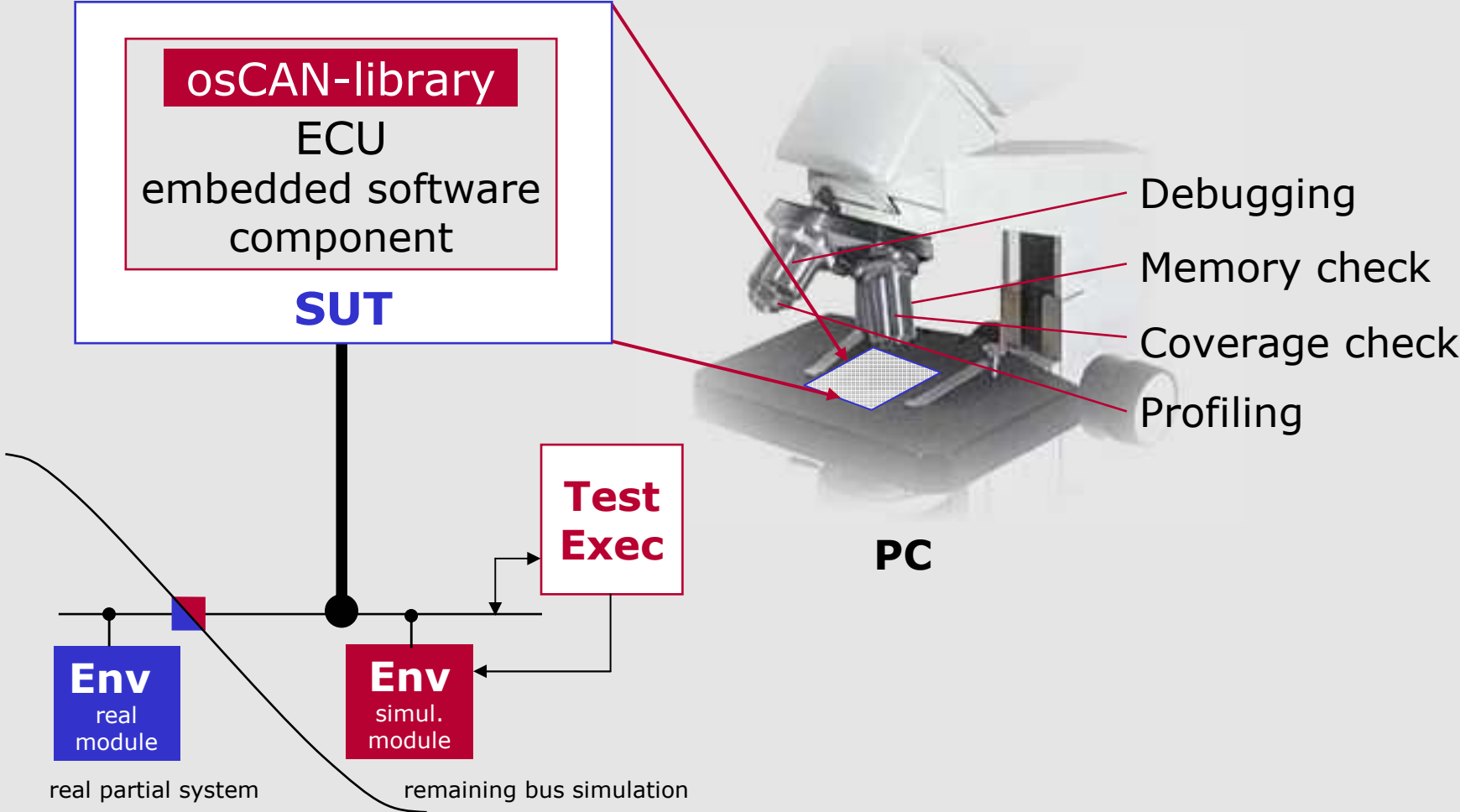


Test of SUT in real environment



- ❑ Black box tests of SUT
 - ❑ Environment either completely simulated, or
 - ❑ Some modules simulated and others available as real modules, or
 - ❑ Environment available as real modules
- ❑ Test execution tool
 - ❑ is running on PC
 - ❑ In all phases, the analysis features and windows (graph, trace, data, ...) are available.

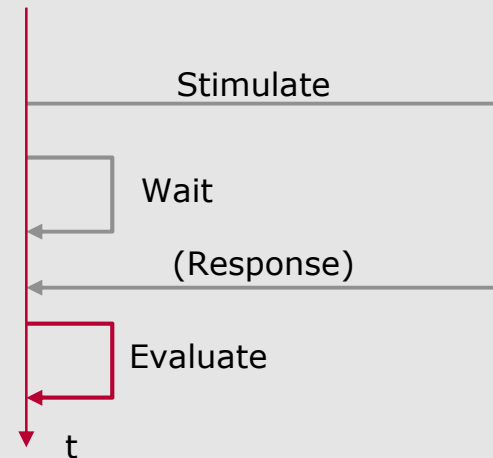
Testing with CANoe: White Box Testing Capabilities



Testing with CANoe: Black-Box-Test Programming in CAPL

- ❑ Stimulation
 - ❑ Already available constructs to output messages
 - ❑ Constructs to set single signals
- ❑ Wait for event conditions
 - ❑ Wait for message, user confirmation, environment variable, ...
 - ❑ Wait for combinations
- ❑ Evaluation + report
 - ❑ Retrieve message- and signal-data
 - ❑ Use any CAPL function

Test Framework SUT+Env



```
// Stimulate
output(Req2MsgBuf)
SetSignal(Req1Msg::SigX, 5)

// Wait
TestWaitForMessage(RespMsg, 2000)

// Evaluate + report
TestValidateSignalInRange(RespMsg::SigA, 0, 1)

TestGetWaitEventMsgData(RespMsgBuf)
If(RespMsgBuf.Time > lMaxTime) {
    TestStepFail(„Time“, „Too late“);
}
```

Testing with CANoe: Black-Box-Test Specification in XML (Example)

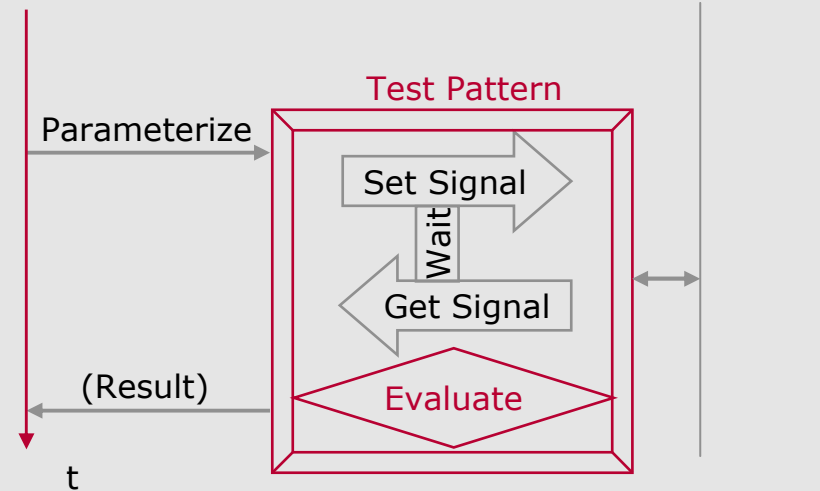
- ❑ Stimulation parameters
 - ❑ Input vector: Values, e.g. signal, environment variable

- ❑ Evaluation parameters
 - ❑ Output vector: Values, e.g. signal, environment variable
 - ❑ Timing
 - ➔ Evaluation is done automatically

- ❑ Control flow
 - ❑ Is built-in to predefined 'test patterns', several patterns are available
 - ➔ Not needed on specification level ➔ Maintainability increased

- ❑ Report
 - ❑ Is written automatically
 - ❑ Formatting can be customized by changing the style sheet

Test Framework



```

<statechange wait="2000">
  <input>
    <cansignal name="GlobState::Ignition">1</cansignal>
    <cansignal name="Wipers::Level">2</cansignal>
  </input>

  <expected>
    <envvar name="WiperMotor"><gt;0</gt></envvar>
  </expected>
</statechange>
    
```

Testing with CANoe: Example Test Specification in Altova "XMLSpy"

WiperSystemTest.xml

- engineer
 - info (3)
 - testgroup title=Initialization
 - testgroup
 - title Core tests of the wiping and washing system
 - constraints
 - conditions
 - cycletime_abs min=0 max=1100
 - testcase
 - ident TC1
 - title Wiping Level 1
 - description Wiping level 1, fall back to off
 - miscinfo title=
 - statecheck
 - wait 800
 - title Select Wiping level 1, check the wiper position being not 0.
 - in
 - cansignal (2)

	name	fibc Text
1	WipingLevel	1
2	WasherRequest	0
 - expected
 - envvar
 - name EnvWiperPosition
 - ne 0



Testing with CANoe: Example Test Specification in Altova "AuthenticDesk"

WiperSystemTest.xml *

Testcase

Identifier

Title

[add version](#)

Description:

Miscellaneous Information:

Title

Name	Content
Additional information	This is an additional information block for the current test case

[add extendedinfo](#) [add constraints](#) [add conditions](#)

Statechange Pattern

Title

Wait

Input

CANSignals:

Name	Message	Value
WipingLevel	add msg	1
WasherRequest	add msg	0

[add envvar](#)

Expected

Environment Variables:

Name	=	<>	<	>	<=	>=	Range
EnvWiperPosition	add eq	0	add lt	add gt	add le	add ge	add range

[add cansignal](#)



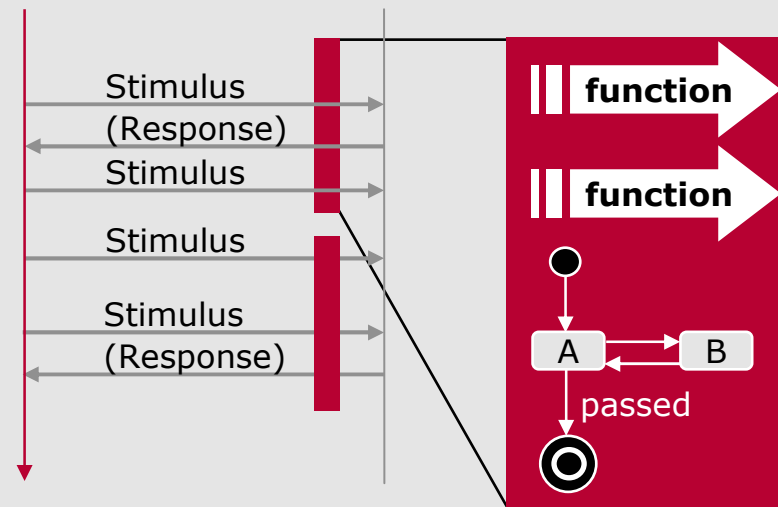
Testing with CANoe: Blackbox Testing Capabilities (Theory)

- ❑ Observing 'invariant' tests
 - ❑ Stimulation is done by e.g. a driving cycle
 - ❑ Mapping-functions are checked dependent to the state

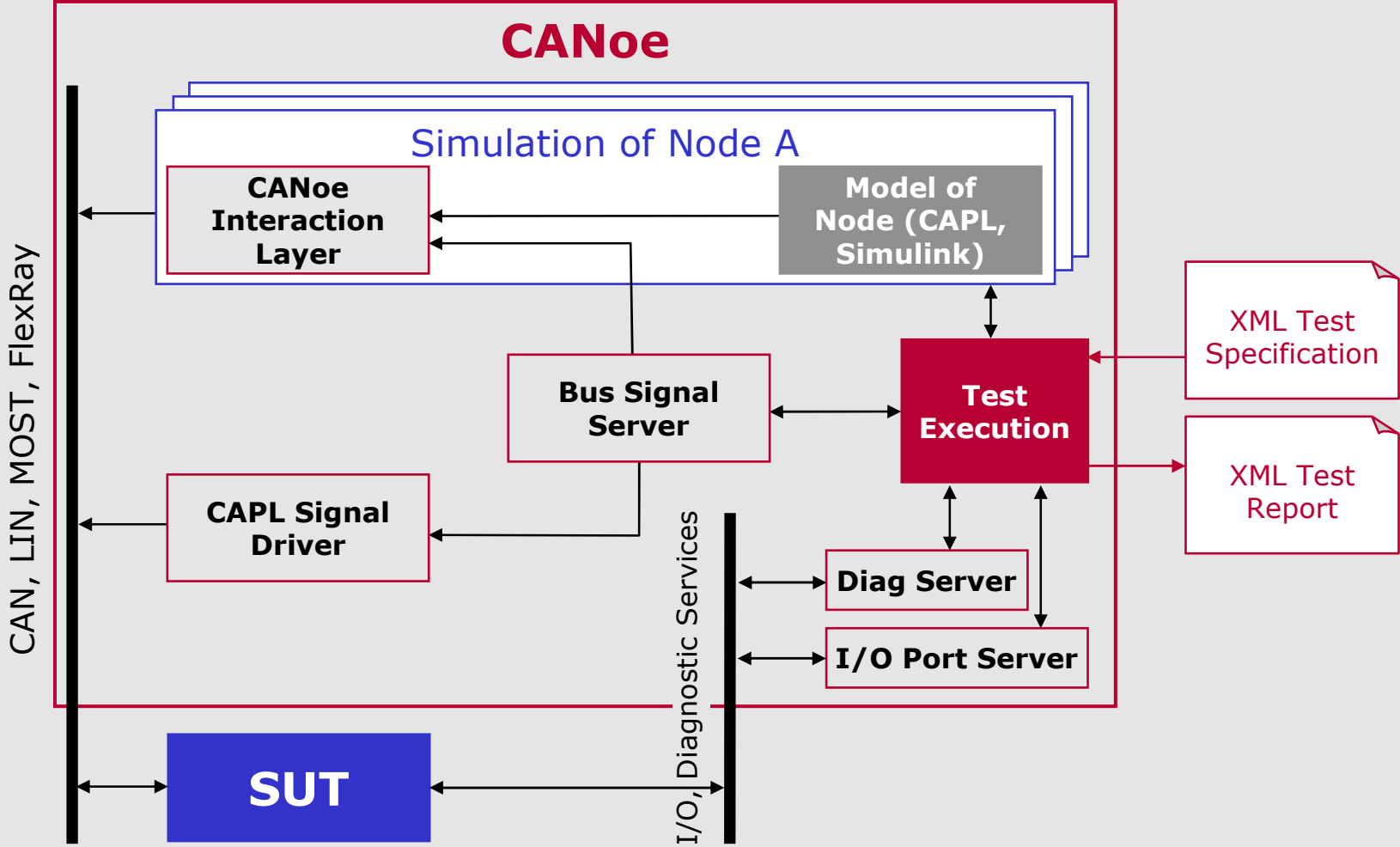


- ❑ Definition of mapping-functions for all states of the SUT is a challenge
- ❑ Additional criteria are needed to define the end of the test

Test Framework SUT+Env

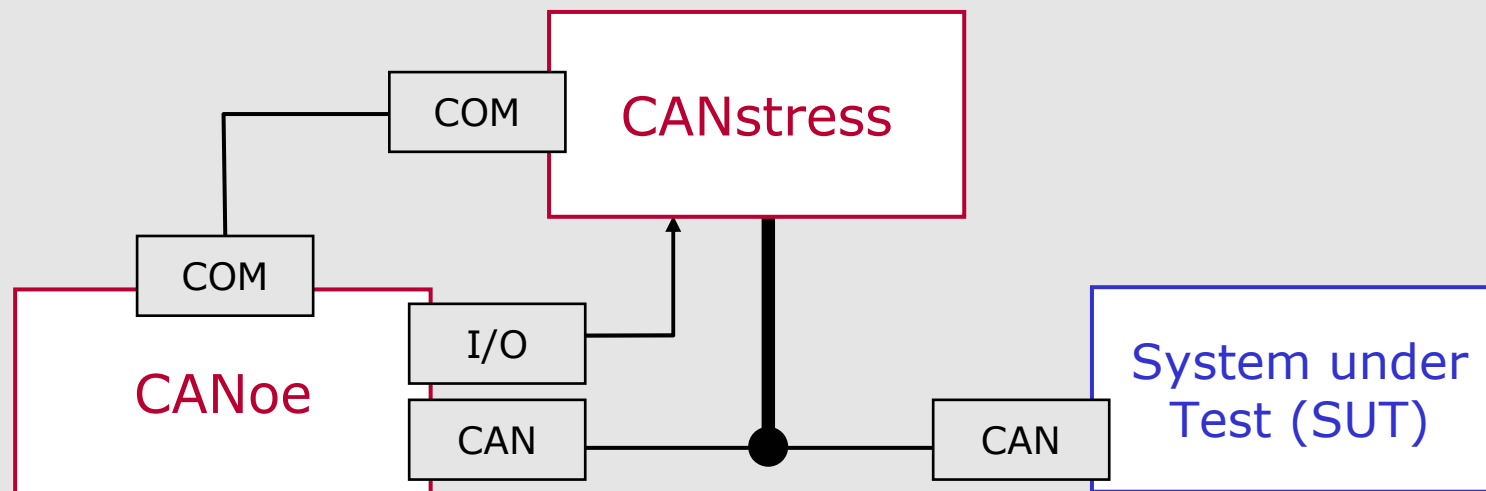


Testing with CANoe: Architecture for Simulation & Test Execution

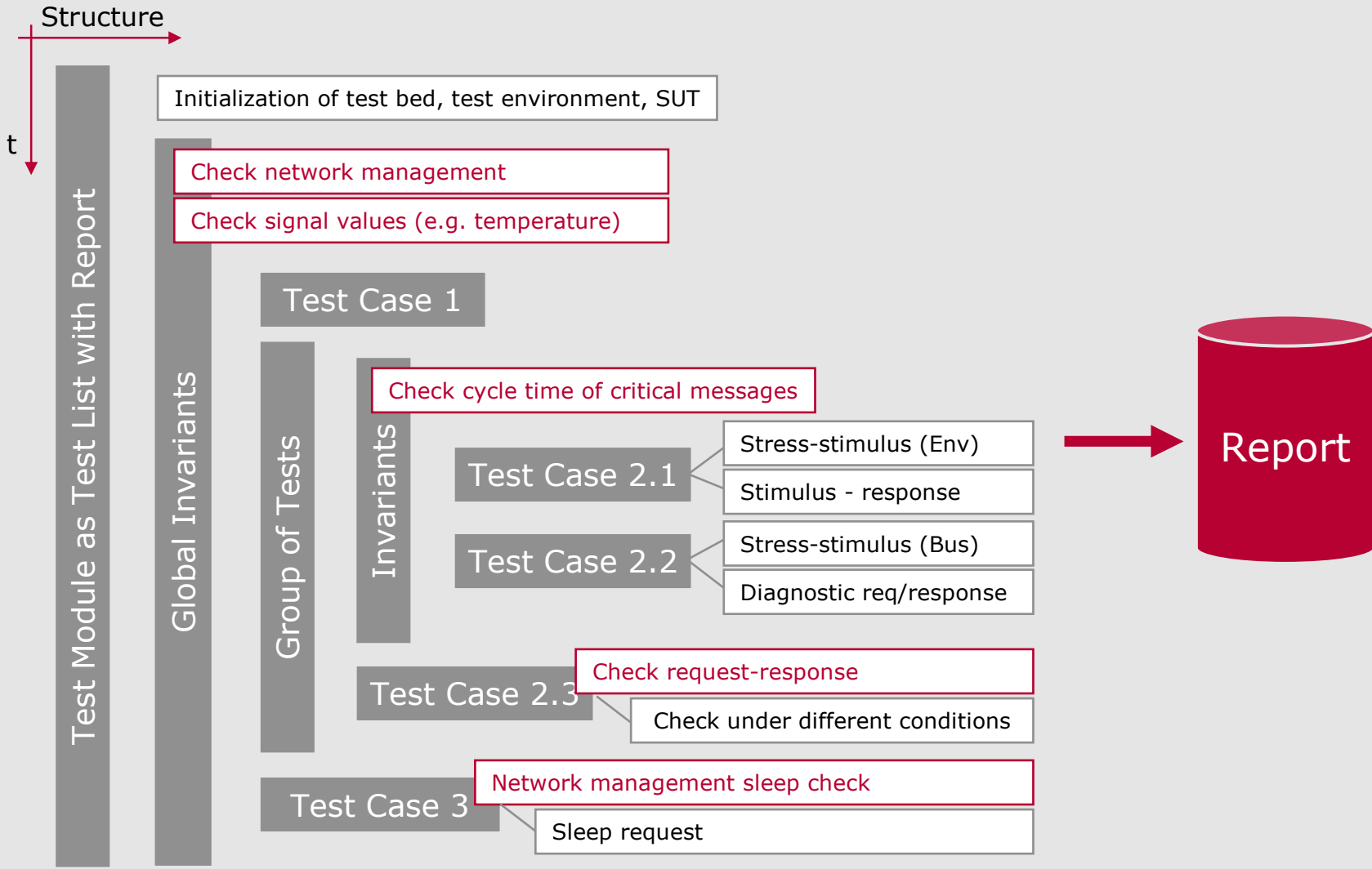


Testing with CANoe: Integration of CANstress

- ❑ CANstress integrated as a test tool in CANoe
 - ❑ CANstress testing functionality is accessible via the COM interface
 - ❑ Additional COM functions to manipulate the important trigger and disturbance parameters in the currently opened CANstress configuration
 - ❑ Trigger/enable trigger via IOCab



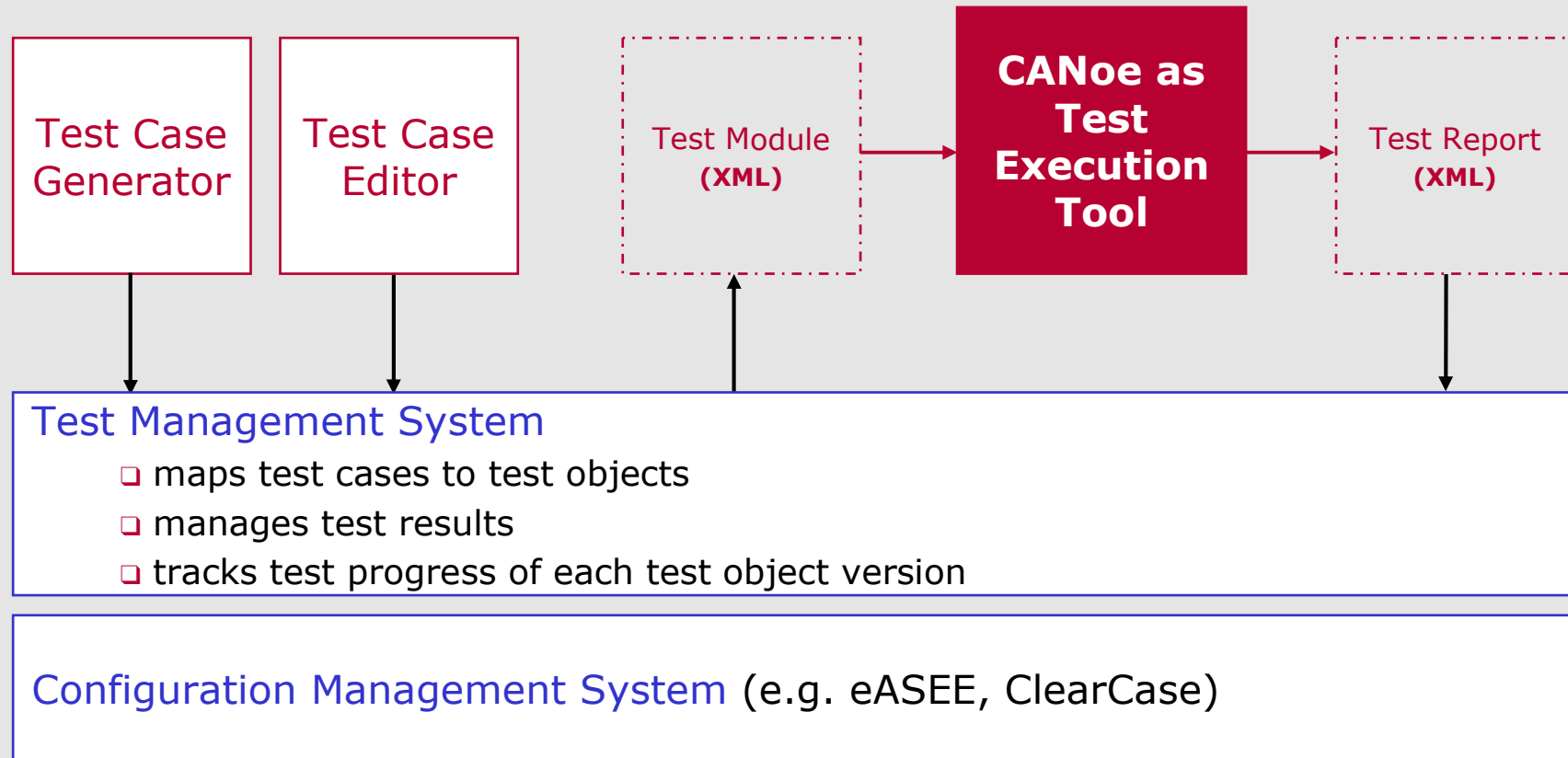
Testing with CANoe: Example of Vector's Approach (CAN)



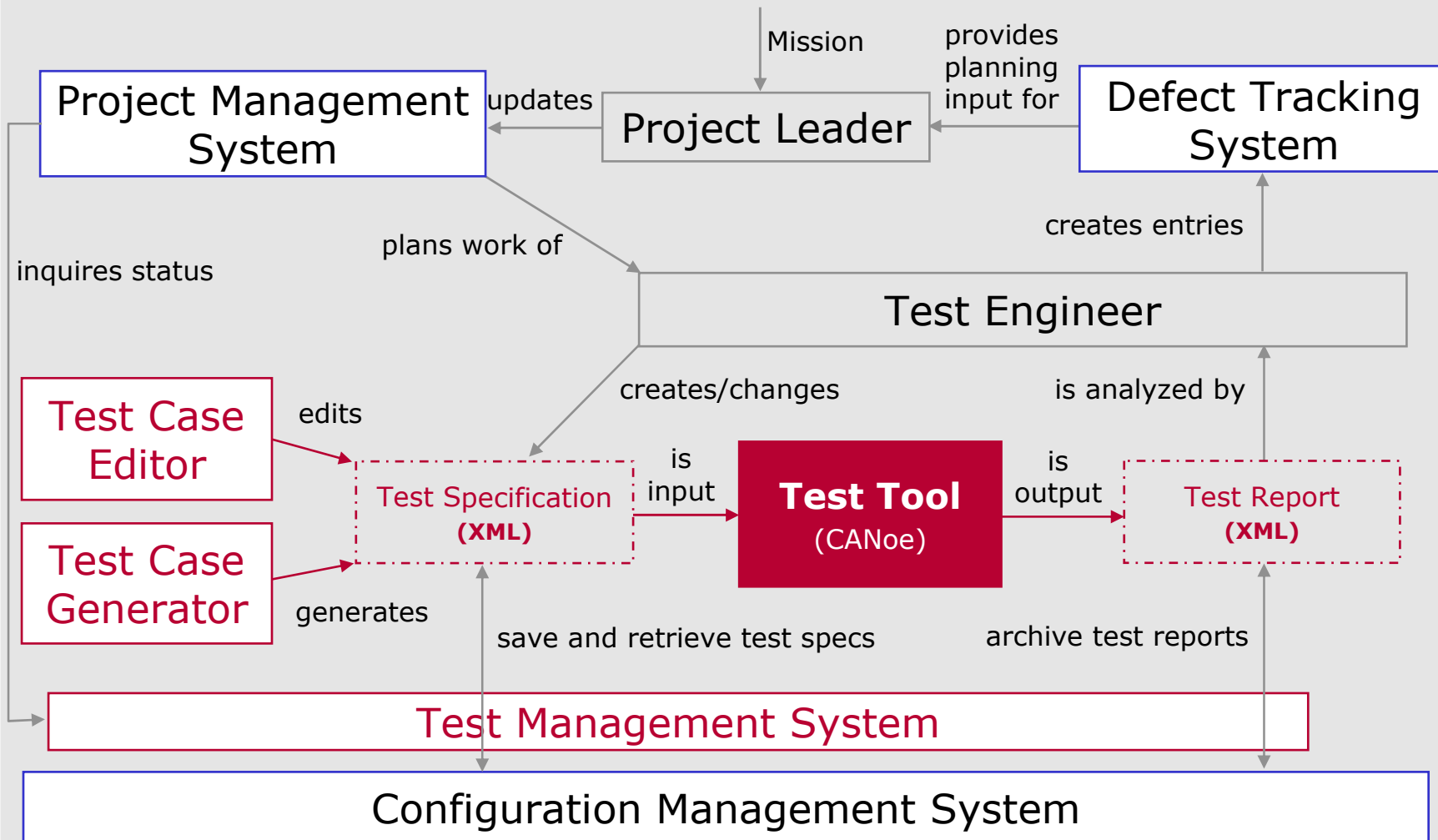
Testing with CANoe: Typical Testing Use Cases

- ❑ Tests are to be executed
 - ❑ Control test environment
 - ❑ Perform “remaining bus simulation”
 - ❑ Execute test, Collect report information, create report
- ❑ Tests are to be planned, evaluated, repeated, ...
- ❑ Tests and their configurations are to be managed
 - ❑ Domain information databases (dbc, ldf, xml, cdd, ...)
 - ❑ System requirements databases
 - ❑ Manage test configurations and specifications
 - ❑ Manage tests for different SUT-versions
 - ❑ ...
- ❑ Tests are to be automated
 - ❑ Retrieve test configurations, information databases and execute tests
 - ❑ Put test report back into test management

Testing with CANoe: Interface to Test Management System



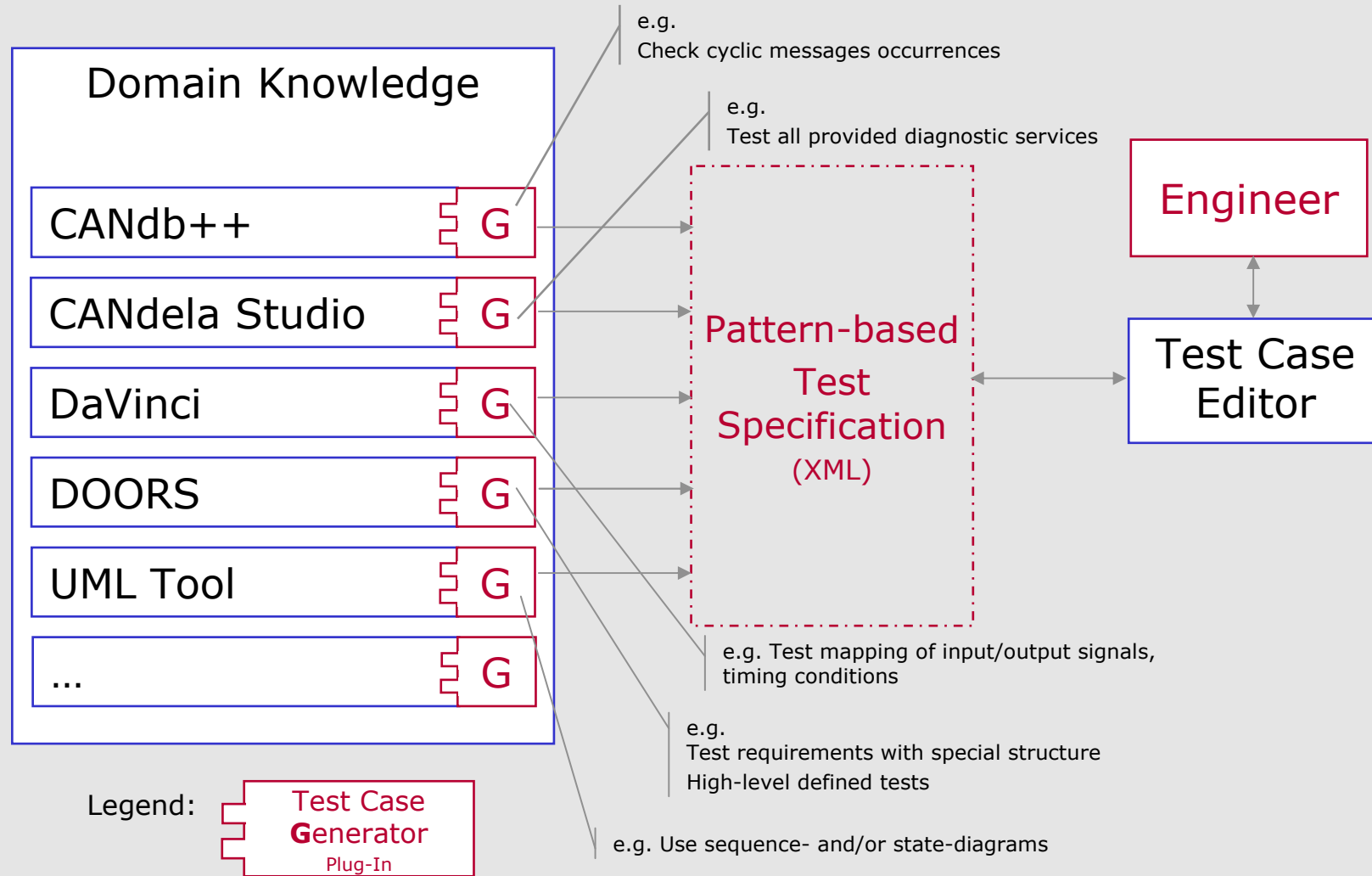
Testing with CANoe: Tool Infrastructure



Testing with CANoe: Process Support

- ❑ Using your already invented process tools
 - ❑ E.g. configuration-, defect-, requirements-, project management
- ❑ Complete/enhance process support by using Vector tools
- ❑ CANoe with test management example - benefits
 - ❑ Configuration management of test specifications and test reports
 - ❑ Share test projects and test specifications between multiple testers
 - ❑ Share core test specifications and combine them specifically to current project
 - ❑ Share result (database) between multiple testers and managers
 - ❑ Managers are able to look onto distilled information (e.g. the test results, time to complete tests, ...)

Testing with CANoe: Creation of Test Specification based on Test Patterns



- ❑ Testing – why Vector?
 - ❑ Vector tools are scalable for optimal usage within supplier and OEM processes
 - ❑ Domain specific patterns are provided
 - ❑ CANoe can be customized – e.g. to provide
 - ❑ particular test patterns
 - ❑ particular test case generators
 - ❑ specialized report formats
 - by project work or by customers
 - ❑ CANoe as easily accessible testing tool makes formalized testing possible:
 - ❑ from the early development phases
 - ❑ for every developer

Thank you for your attention.

For detailed information about Vector
and our products please have a look at:
www.vector-informatik.com

Siegfried Beeh
Vector Informatik GmbH
Ingersheimer Str. 24
D-70499 Stuttgart

