



Validas AG

Building Validation Suites with Eclipse for Model-based Generation Tools

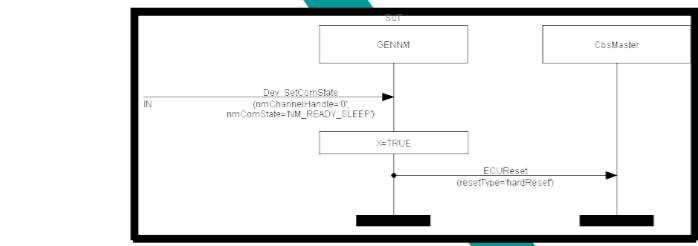
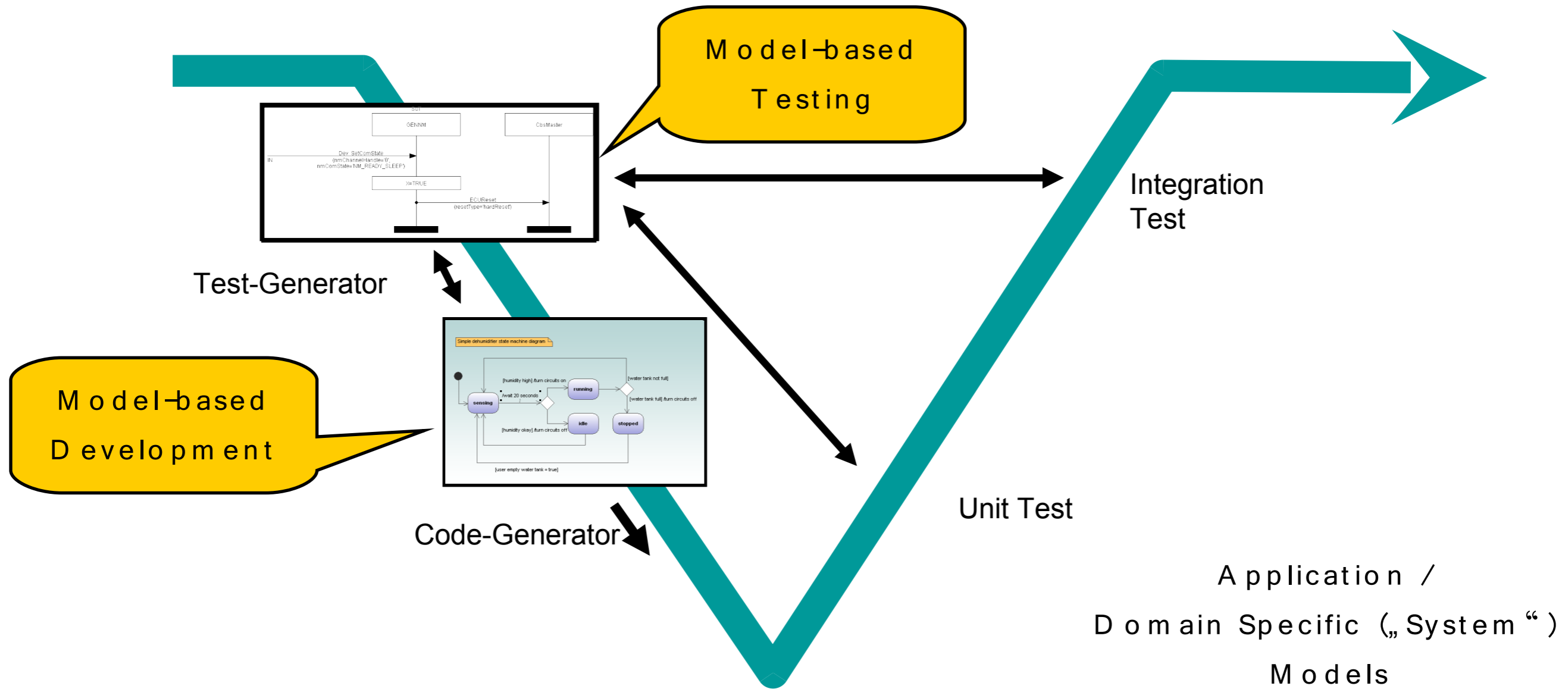
24.6.2010
Dr. Oscar Slotosch
Validas AG

- ▶ Model-based Development (of Embedded Systems)
- ▶ ISO 26262
- ▶ Validation Suite
 - Architecture
 - Used Eclipse Technologies for building
 - Experiences
- ▶ Conclusion

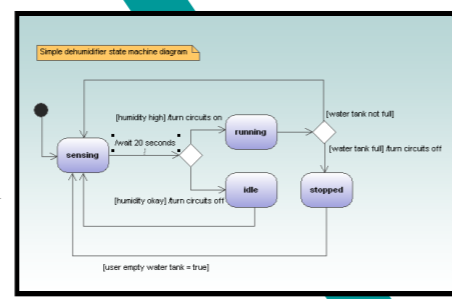
Validas A G

- ▶ Founded 2000
- ▶ 20 Employes
- ▶ Motto: Validated Quality
 - ▶ Model-based Development
 - ▶ Model-based Testing
 - ▶ Tool Qualifications
- ▶ We solve Your Challanges!
- ▶ Clients
 - ▶ BMW
 - ▶ EADS
 - ▶ ESG
 - ▶ Giesecke & Devrient
 - ▶ MAN
 - ▶ Eurocopter

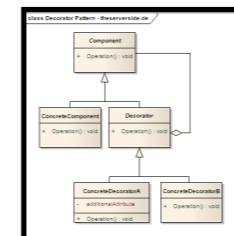




Test-Generator



Code-Generator



Model-based Tools

Supporting Tools

„Unified“ Models

- ▶ Development

- ▶ Model Creation =>

- „Graphical Tool“

- ▶ Code Generation => „Generator“

- ▶ Code Compilation => Compiler

- ▶ Important Verification Methods for Models

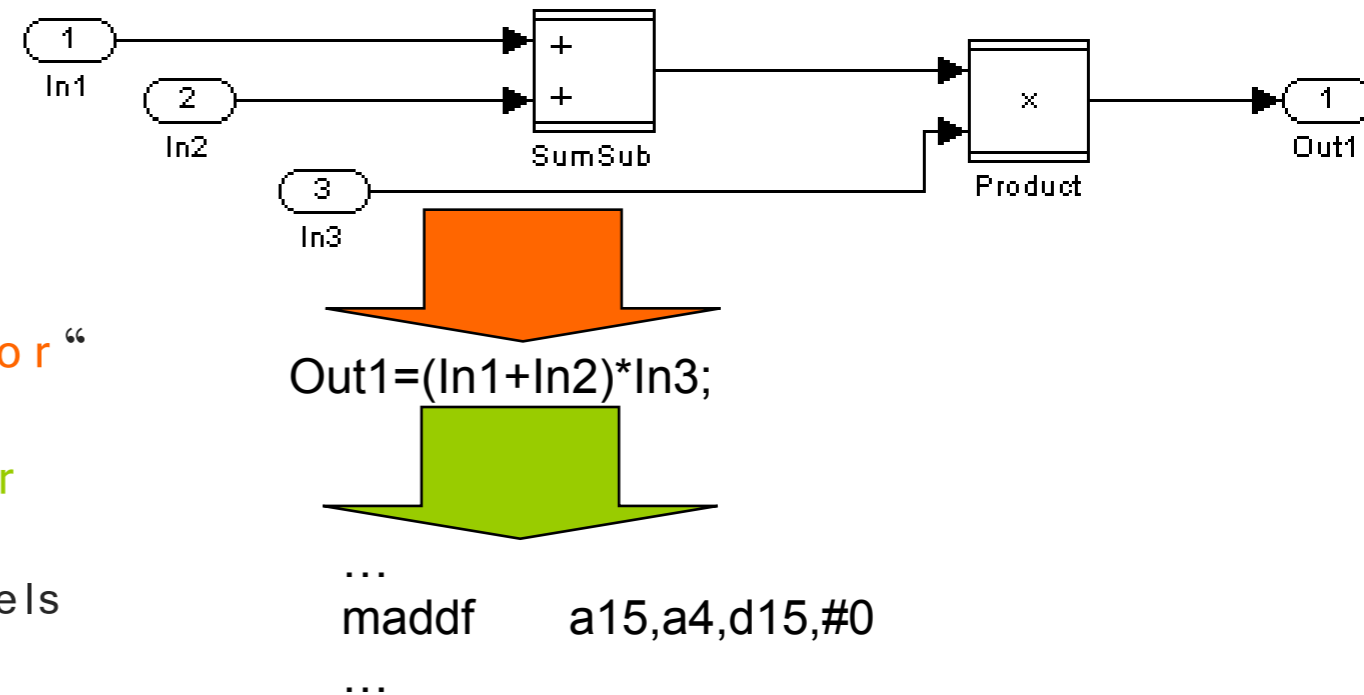
- ▶ Simulation (MIL, SIL, PIL)

- ▶ Rule Checking („Subset-Checker“)

- ▶ Property Verification (Formal Methods)

- ▶ Model Coverage

- ▶ Back-To-Back-Testing: MIL = SIL = PIL

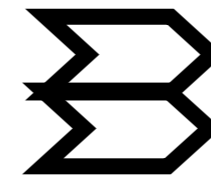


ISO 26262-8, Chapter 11: Qualification of software tools

► Classification in „Tool Confidence Level (TCL)“

► Tool Impact (TI)

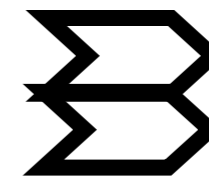
– TI0: no impact => Tool is TCL1



– TI1: some impact

• Tool Error Detection/prevention probability (TD)

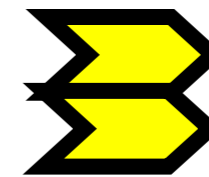
❖ TD1: high confidence => Tool is TCL1



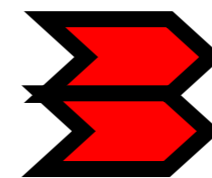
❖ TD2: medium confidence => Tool is TCL2



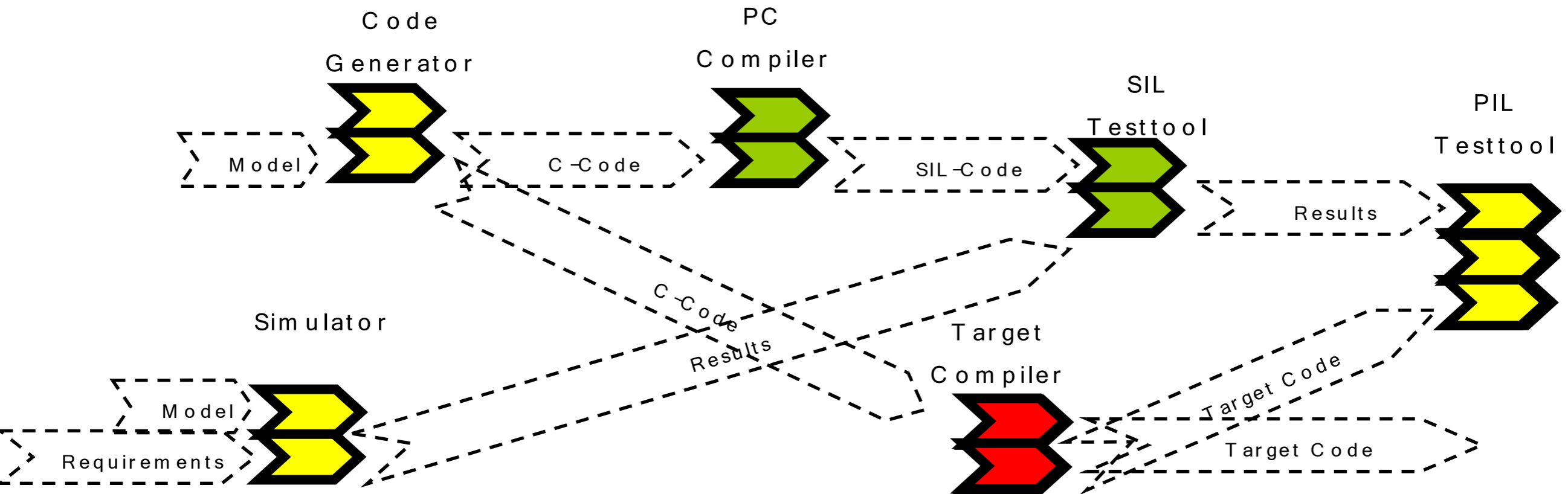
❖ TD3: low confidence => Tool is TCL3



❖ TD4: other => Tool is TCL4



Example: Tool Chains in Model-based Development

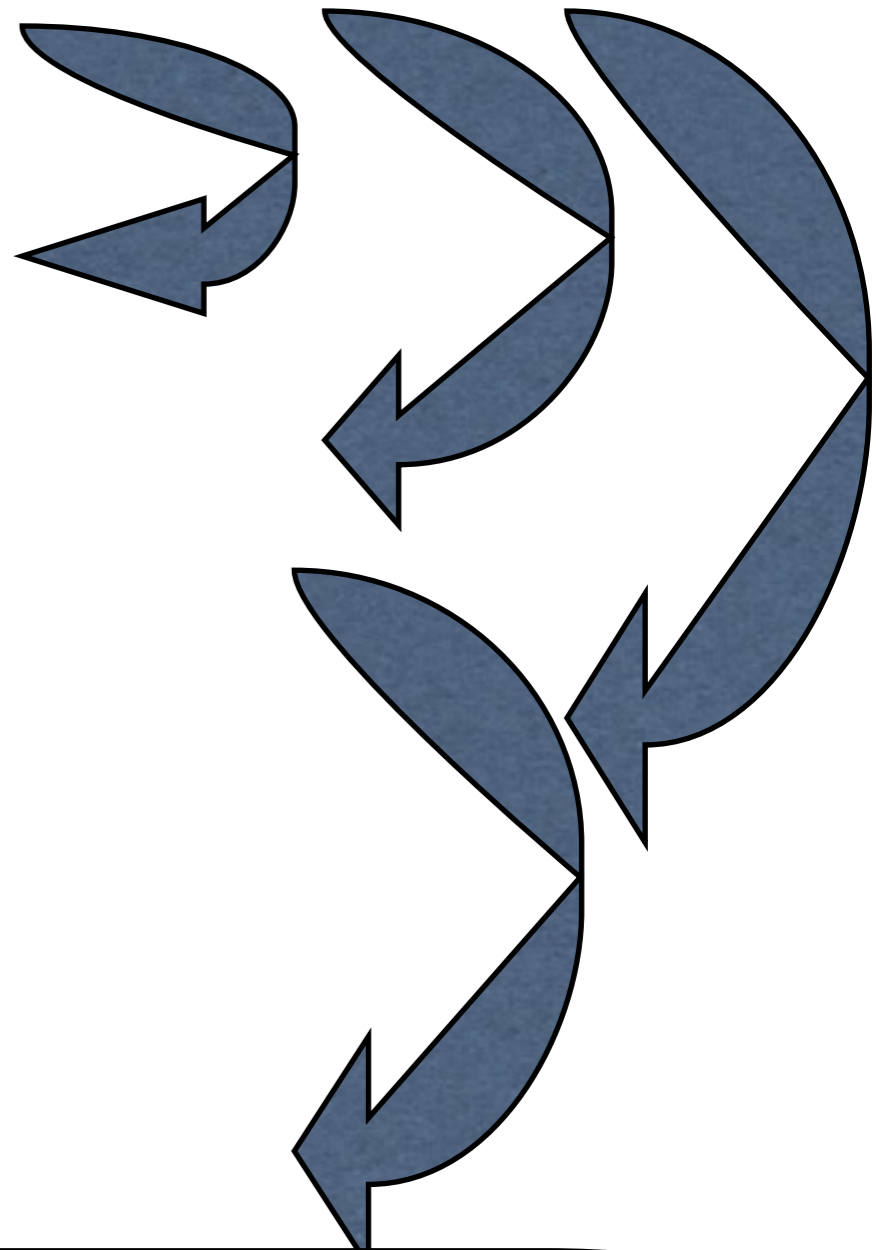
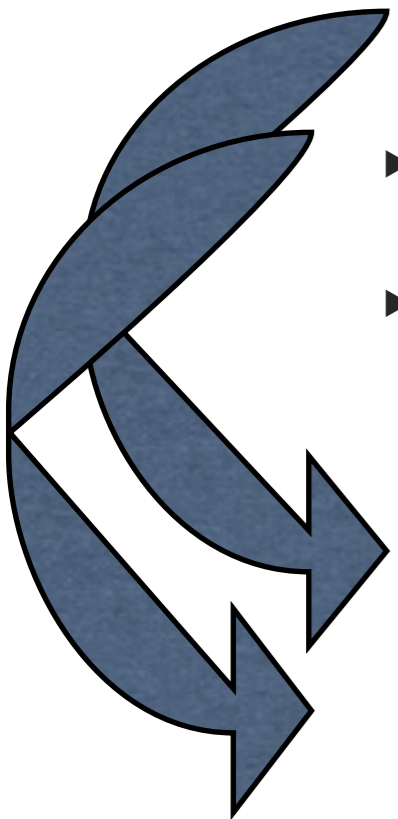


- ▶ Customer: can verify tool results
- ▶ Tool provider
 - Has to require verification of his tools („redundancy“)
 - or
 - Validate his tool (Validation Suite / Proven in use)

- ▶ Requirements / Standard
- ▶ Manual of Tool with
 - Used Functions
- ▶ Tool Developer Guide (Process)
- ▶ Validation and Verification Plan for
 - Requirements
 - **Functions**
 - Compliance with Process
- ▶ Validation and Verification Report

Validation Suite

Quote from Ada-Test Suite (ACATS): "...
the ACATS tests the normal usage of these features,
not unusual corner-cases."



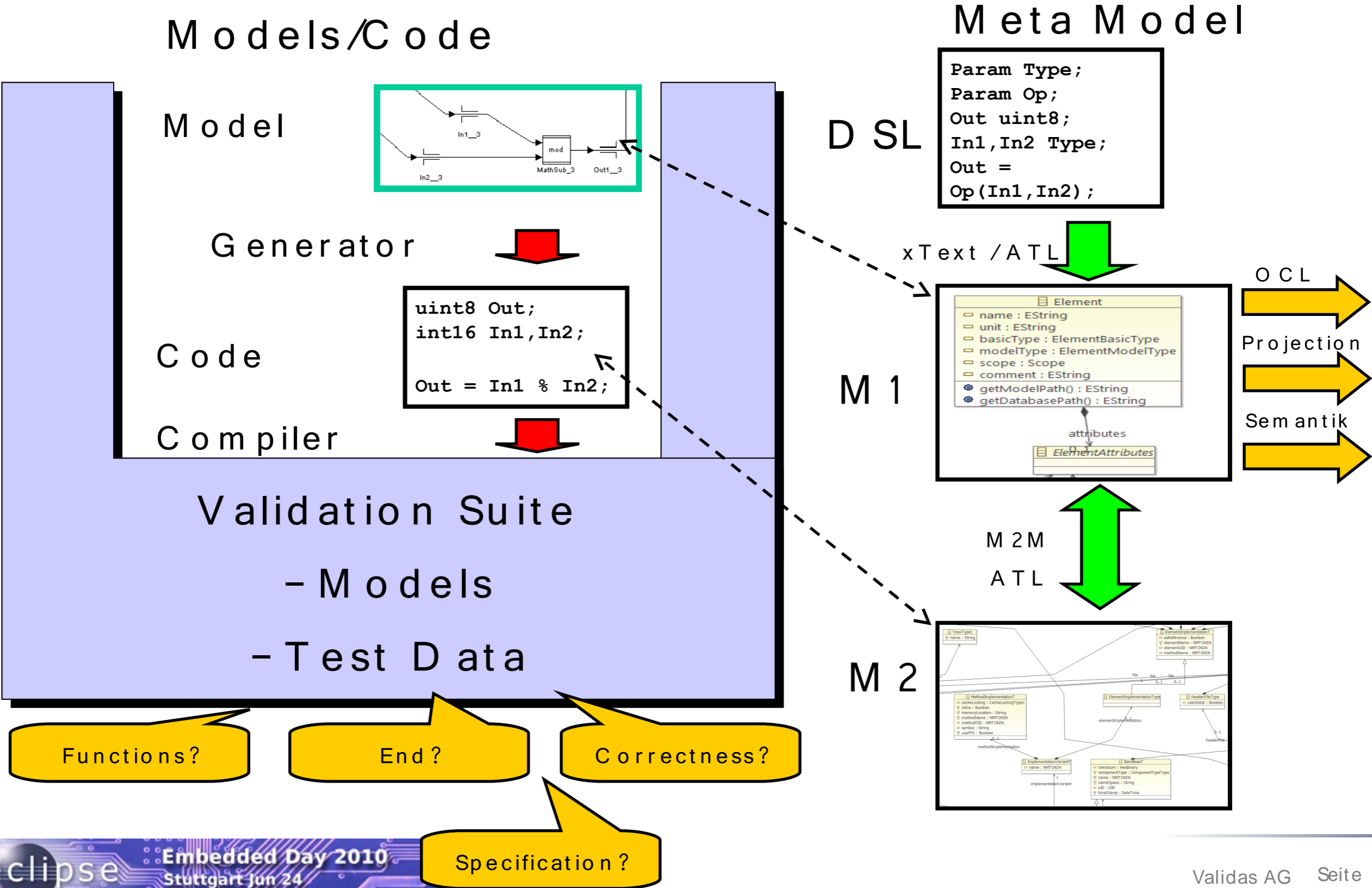
Test Method for Generators

Construction of Validation Suites

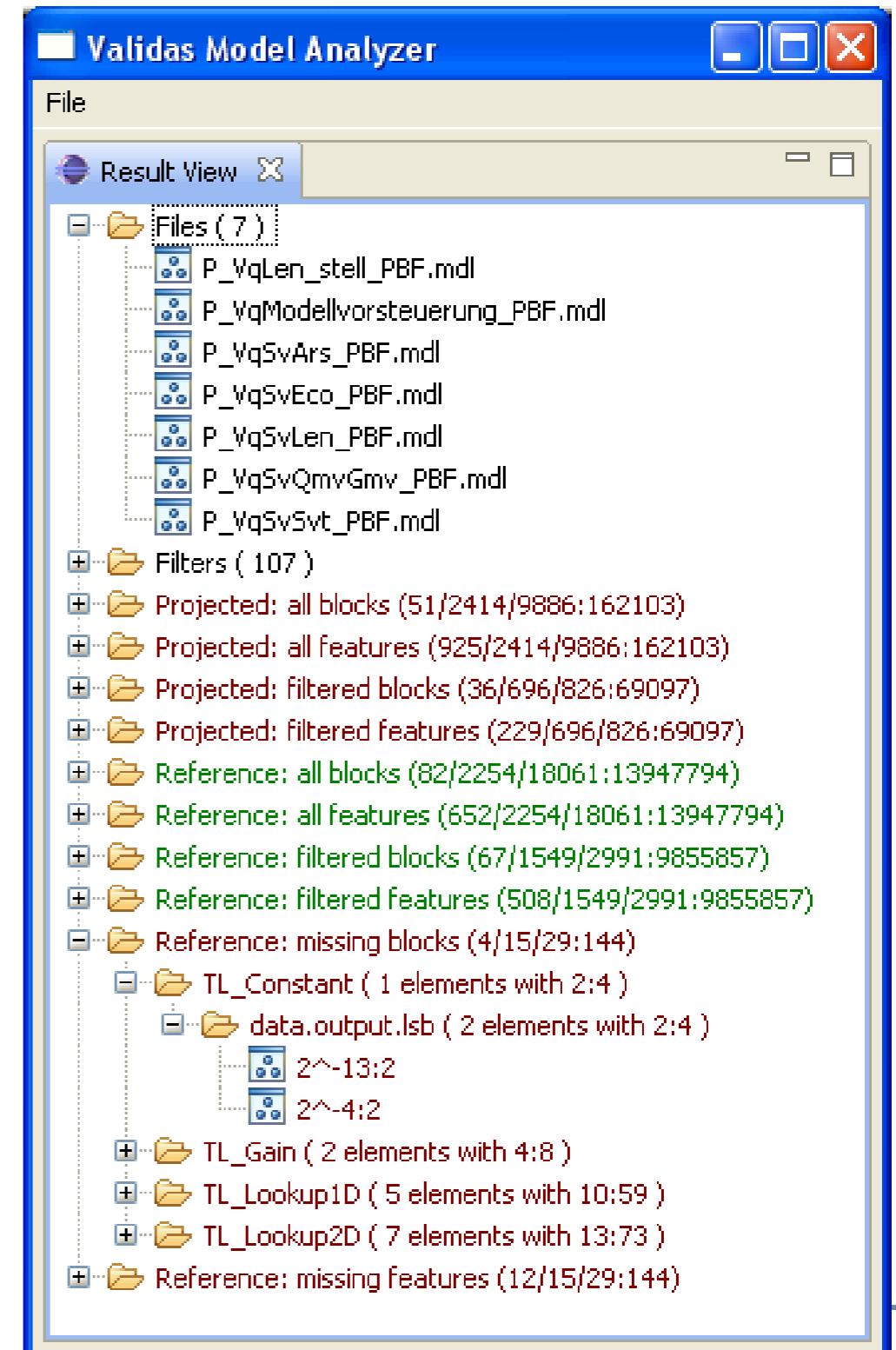
Test Specification: Domain/Tool Specific Language

- ▶ Test Inputs: (Model-)Generator
- ▶ Test Outputs: Tool Interpreter (Reference Tool)
- ▶ Test Automation of
 - Code Generation, Compilation and
 - Execution (on Target)
- ▶ Analysis tools:
 - Report Generation
 - Root Cause Analysis
- ▶ Until:
 - All Model/Features are covered -> Model Coverage
 - All Tests have been Executed
 - All Deviations have been analyzed

Eclipse-Modeling for the Construction of Suites

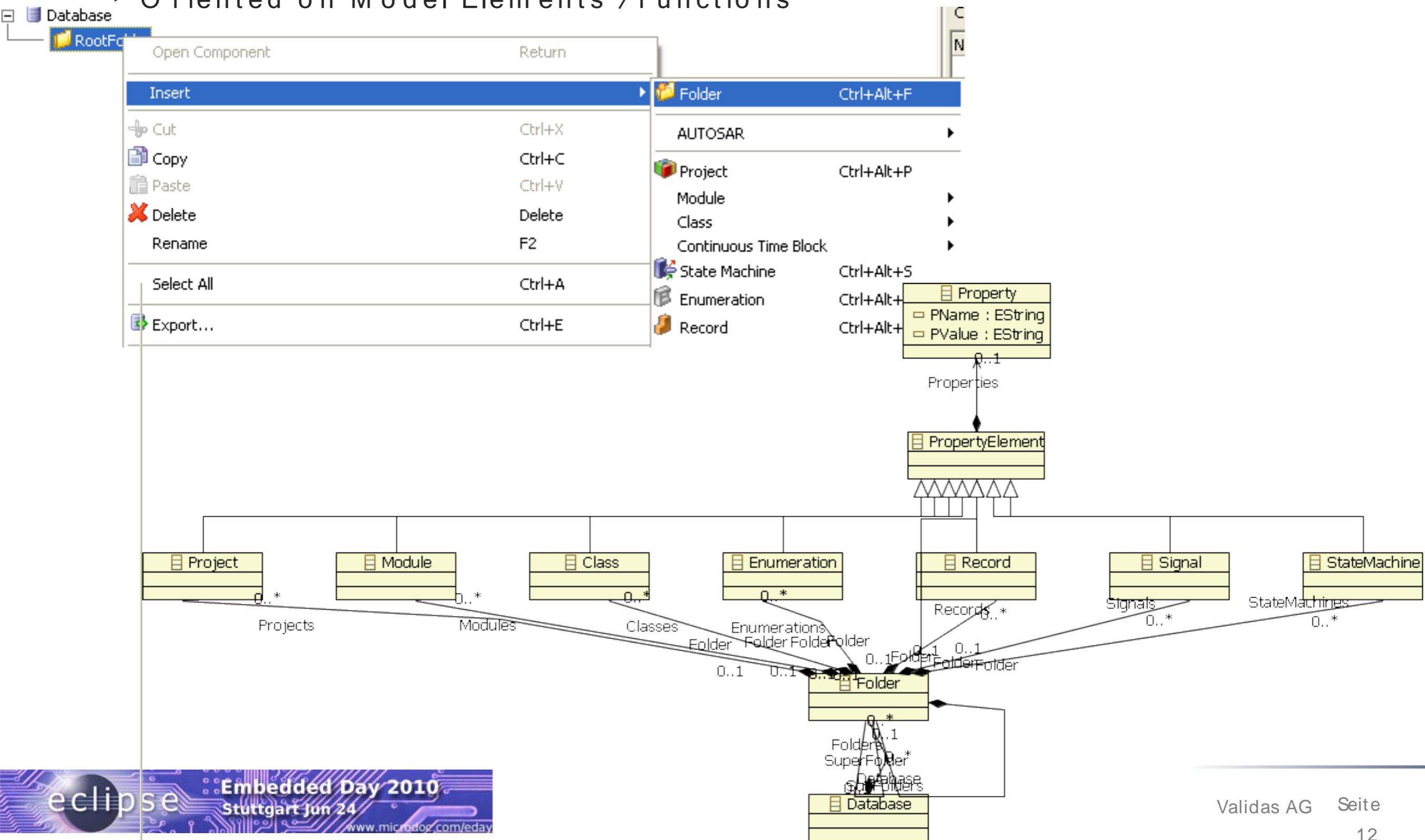


- ▶ Input: Set of Reference
 - Models / Elements of Meta Models
 - XML-Structures
 - Projection Results (hierarchy)
- ▶ Output: List of used
 - Model Elements
 - Model Properties
 - Settings / Configurations
- ▶ Filter Mechanism for not relevant properties (color,..)
- ▶ Comparison of two Sets
 - Validation Suite
 - Application models



Example of Meta Model

► Oriented on Model Elements / Functions

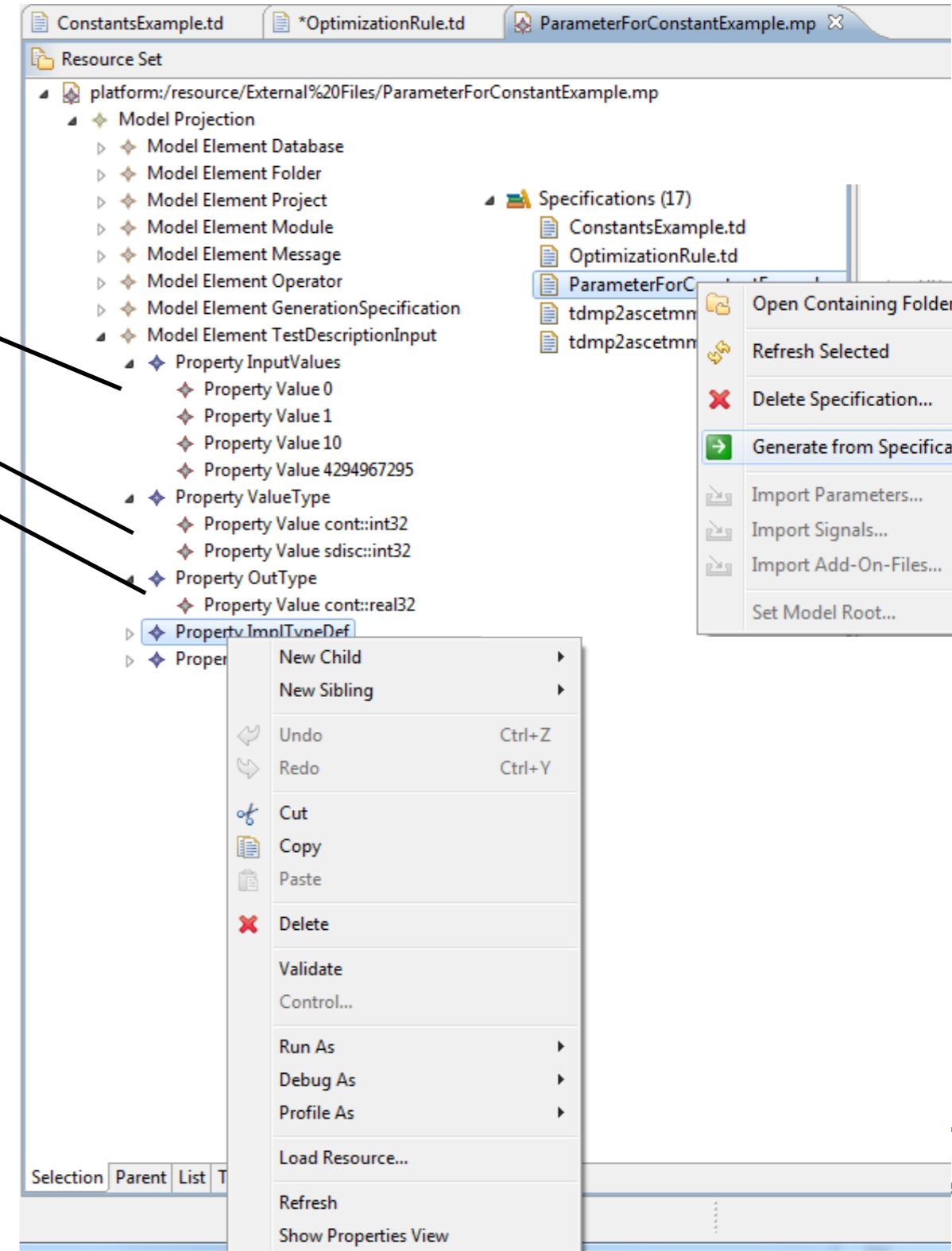


- ▶ DSL (xText) for Specification
 - Parameter: Operators, Types, Constants
- ▶ ATL for Translation xText → M1
- ▶ M2M zur Generation of Models
- ▶ M2M Definition of Semantic
- ▶ OCL for Modeling Rule Checker
- ▶ Model-Projektion on EMF-Models for Coverage/Functions

DSL for Test Specification and Generation

Specification = Model Description + Parameter

```
ConstantsExample.td x *OptimizationRule.td  
LiteralParameter InputValues;  
TypeParameter ValueType;  
TypeParameter OutType;  
  
Class TC {  
  Const myConst: ValueType = Literal(InputValues);  
  
  Method OutType m_const() {  
    return myConst;  
  }  
  
  /** rule 1.2.3 for + with Constant and Expressions: c op e -> op c */  
  Method OutType comLawPlus ( c : ValueType , ex : ValueType ) {  
    return + ( c , ex ) ;  
  }  
}  
  
Module TM {  
  Const globalConst: ValueType = Literal(InputValues);  
  
  ReceiveMsg In1: ValueType;  
  ReceiveMsg In2: ValueType;  
  
  SendMsg outConst: OutType;  
  SendMsg outcomLawPlusIn1In1: OutType;  
  SendMsg outcomLawPlusIn2In1: OutType;  
  SendMsg outcomLawPlusIn1In2: OutType;  
  SendMsg outcomLawPlusIn2In2: OutType;  
  SendMsg outcomLawPlusIn1Const: OutType;  
  SendMsg outcomLawPlusConstIn2: OutType;  
  
  Process p0() {  
    outConst = TC.m_const();  
    outcomLawPlusIn1In1 = TC.m_const(In1, In2);  
    outcomLawPlusIn2In1 = TC.m_const(In2, In1);  
    outcomLawPlusIn1In2 = TC.m_const(In1, In2);  
    outcomLawPlusIn2In2 = TC.m_const(In2, In2);  
    outcomLawPlusIn1Const = TC.m_const(In1, globalConst);  
    outcomLawPlusConstIn2 = TC.m_const(globalConst, In2);  
  }  
}
```



Resource Set

- platform:/resource/External%20Files/ParameterForConstantExample.mp
 - Model Projection
 - Model Element Database
 - Model Element Folder
 - Model Element Project
 - Model Element Module
 - Model Element Message
 - Model Element Operator
 - Model Element GenerationSpecification
 - Model Element TestDescriptionInput
 - Property InputValues
 - Property Value 0
 - Property Value 1
 - Property Value 10
 - Property Value 4294967295
 - Property ValueType
 - Property Value cont::int32
 - Property Value sdisc::int32
 - Property OutType
 - Property Value cont::real32
 - Property ImplTypeDef
 - Property

Specifications (17)

- ConstantsExample.td
- OptimizationRule.td
- ParameterForC
- tdmp2ascetmn
- tdmp2ascetmn

Context Menu:

- Open Containing Folder...
- Refresh Selected
- Delete Specification...
- Generate from Specification
- Import Parameters...
- Import Signals...
- Import Add-On-Files...
- Set Model Root...

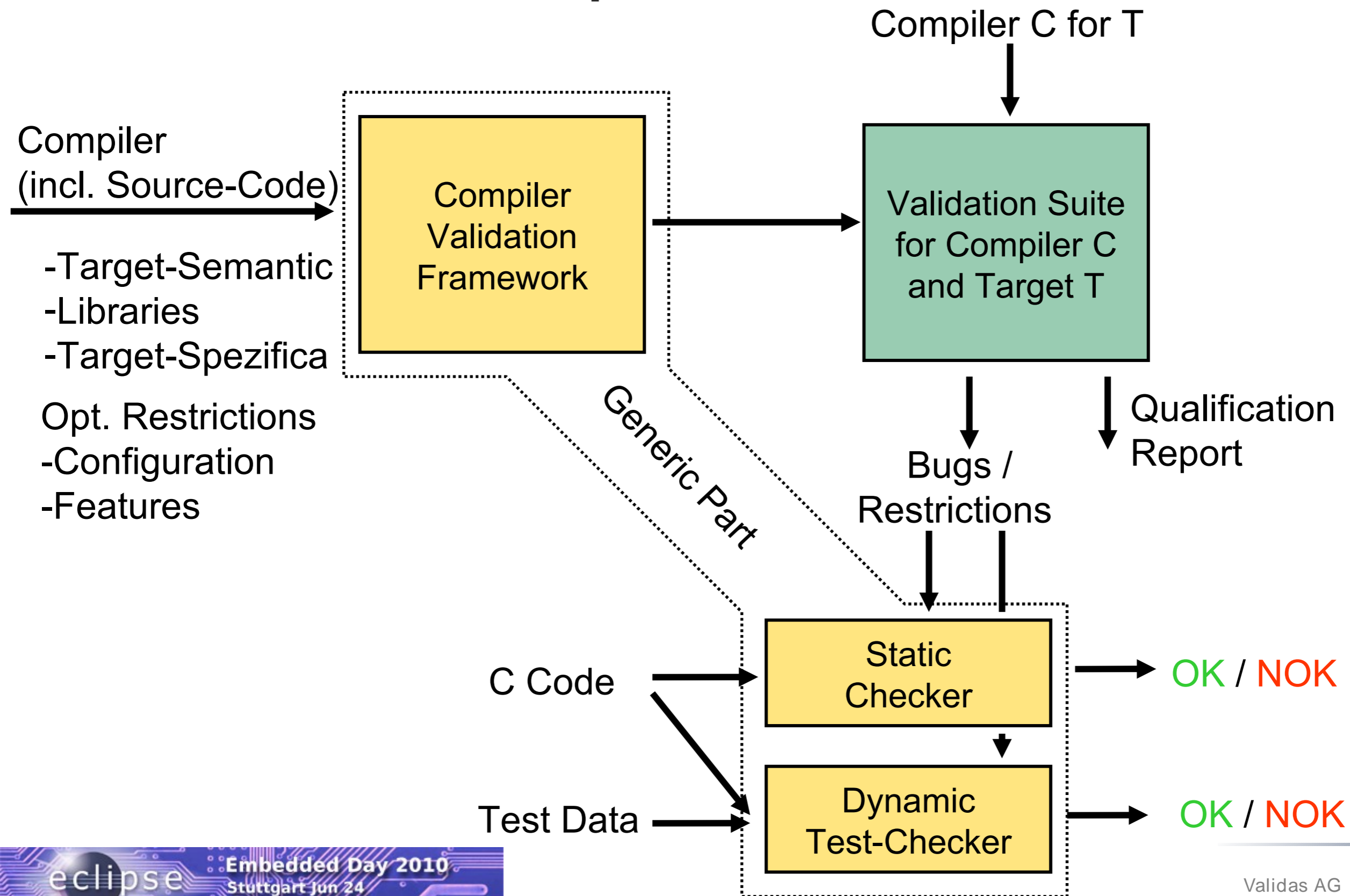
Selection Parent List T

Context Menu (over Property InputValues):

- New Child
- New Sibling
- Undo (Ctrl+Z)
- Redo (Ctrl+Y)
- Cut
- Copy
- Paste
- Delete
- Validate
- Control...
- Run As
- Debug As
- Profile As
- Load Resource...
- Refresh
- Show Properties View

- ▶ M 1 : approx . 1 5 0 C l a s s e s
- ▶ M 2 : approx . 3 5 0 C l a s s e s
- ▶ M 2 M : approx . 1 3 0 0 0 L i n e s o f A T L T r a n s f o r m a t i o n s
- ▶ D S L : approx 4 0 0 0 L i n e s o f T e s t S p e c i f i c a t i o n
- ▶ Results (Testsuite), generated in some hours:
 - 1 0 0 0 0 M o d e l s w i t h
 - 6 0 0 0 0 0 T e s t s e q u e n c e s
- ▶ Stable & performant solution
- ▶ ...
- ▶ W e w o u l d d o i t a g a i n u s i n g t h e s e E c l i p s e t o o l s

Vision: Validation Framework for Different C Compilers



- ▶ Models in Development of Embedded Systems
- ▶ ISO 26262 Tool Classification
- ▶ Validation Suites
- ▶ Eclipse is suitable for building Suites

Thank You !



**Your partner for innovation
in embedded quality**

Arnulfstraße 27
80335 München
www.validas.de
info@validas.de