Harmonizing System Development and Test Development with MDA

Zhen Ru Dai

Fraunhofer FOKUS · Germany

OMG Software Assurance Workshop, March 2007

Motivation

- Testing improves software quality
- Manual test development is resource consuming
- System development independent from test development

➢ Integration and automation of system development and test development

Outline

I. MDA and Testing

II. An Approach to Model-Driven Testing (MDT) with MDA

III. MDT in Action

IV. Summary

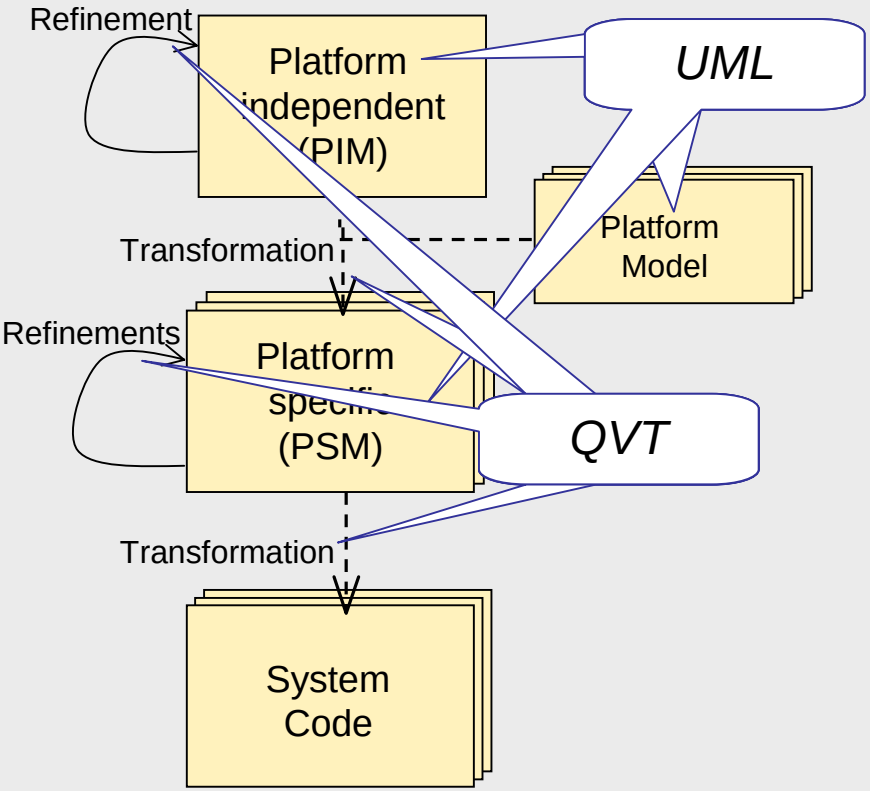
I. Testing Using Models
- Related Work

- Model-based testing / model-driven testing
- Implementation vs. specification
- Automatic test generation from specification
- Existing approaches with UML, SDL, LTS etc.

➢ Test development decoupled from system development

The MDA Framework

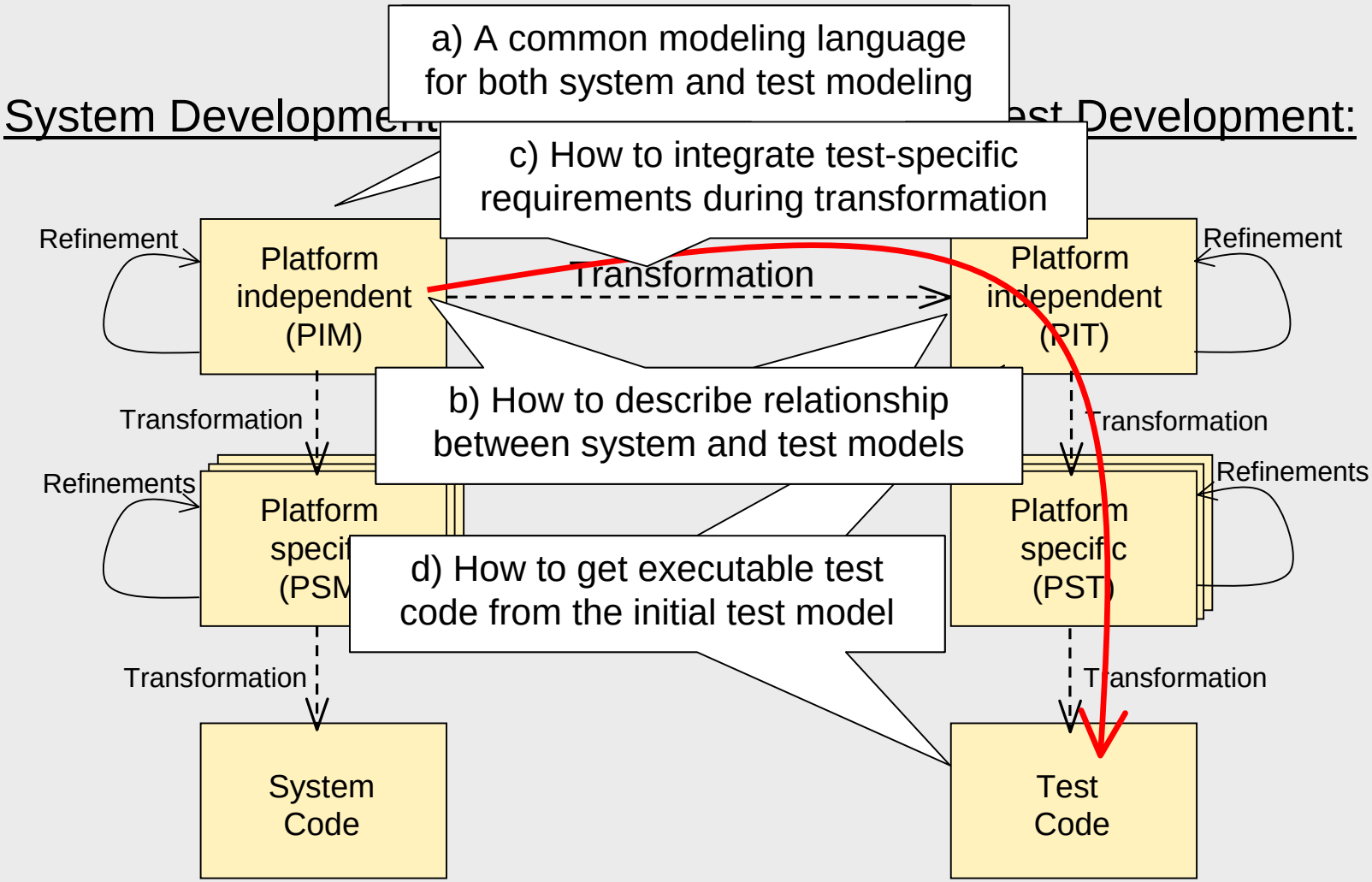
- Addresses interoperability problem
- Generates system codes running on different platforms
- Models and transformation as key technologies
- Standardized technologies and languages, e.g. UML, QVT and MOF

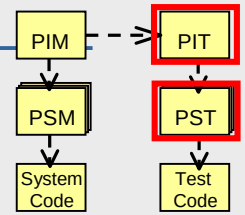


Can we use a model-driven approach for test development, similar to that of MDA to improve software quality?

BUT: No test aspects defined in the framework!

II. MDA & MDT


a) A common modeling language for both system and test modeling
System Development:
est Development:
c) How to integrate test-specific requirements during transformation
Refinement
Platform independent (PIM)
Transformation
Platform independent (PIT)
Refinement
Transformation
b) How to describe relationship between system and test models
Transformation
Refinements
Platform specif (PSM
Platform specific (PST)
Refinements
d) How to get executable test code from the initial test model
Transformation
Transformation
System Code
Test Code



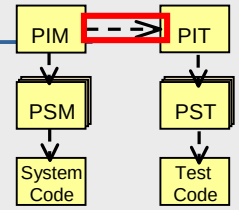
Test Modeling Language

a) Common modeling language for system and test modeling

☞ ***UML 2.0 & UML 2.0 Testing Profile*** *(U2TP)*

- Black-box testing
- Four Concept-Groups:
  - Test Architecture
  - Test Behavior
  - Test Data
  - Time
- Mapping to existing test infrastructures of JUnit and TTCN-3

Architecture concepts	Behavior concepts	Data concepts	Time concepts
SUT	Test control	Wildcards	Timer
Test component	Test objective	Data partition	Time zone
Test context	Test case	Coding rules	
Test configuration	Defaults	Data pool	
Arbiter	Verdicts	Data selector	
Scheduler			



Deriving Test Model

b) Integrate test-specific requirements

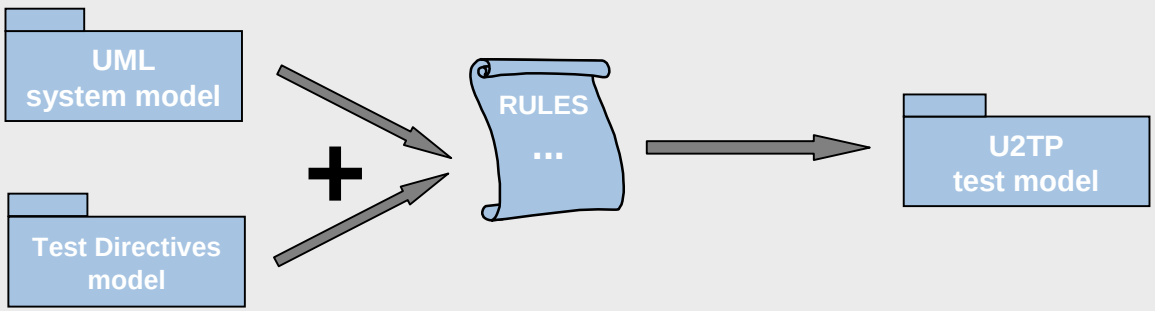
☞ ***Test directives***

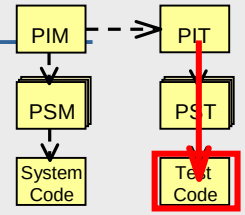
- Set test configuration
- Select test scenarios
- Add timers, default behavior and test results

c) Describe relationship between system and test models

☞ ***Test model transformation***

- Re-use system model
- Integrate test directives
- Generate test model





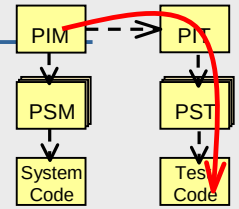
Test Code Generation

d) Get executable functional test code from test model

- Testing and Test Control Notation, version 3 (TTCN-3)
  - Testing language standardized at the European Telecommunication Standardization Institute (ETSI)
  - Popular in the telecommunication domain, but also suitable for other domains
  - Executable test code with test environment

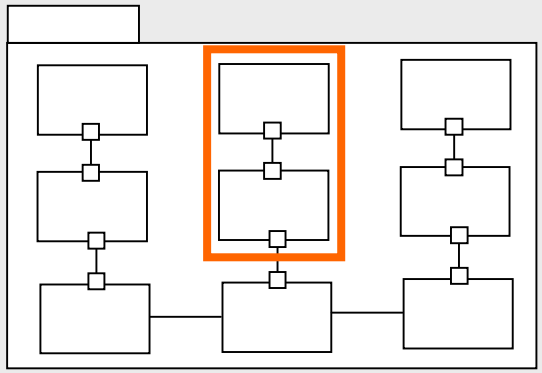
☞ ***Mapping rules***

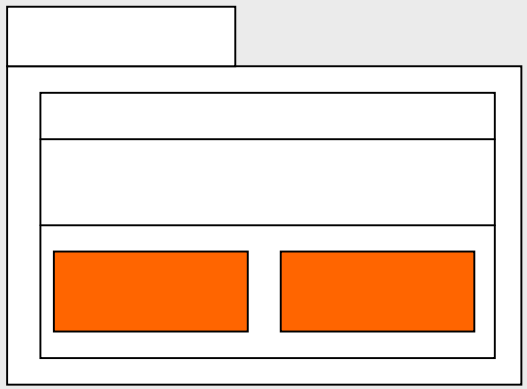
- Reuse of existing TTCN-3 infrastructure
- U2TP → TTCN-3



III. MDT in Action

UML 2.0 Model

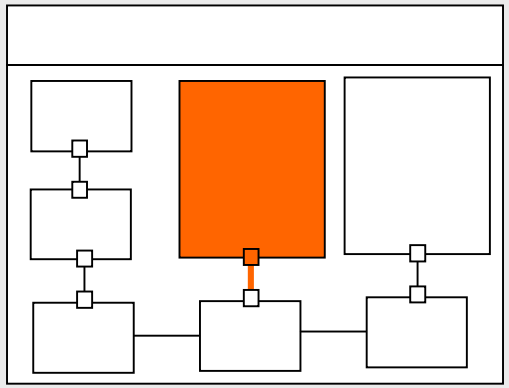




Test Directives Model

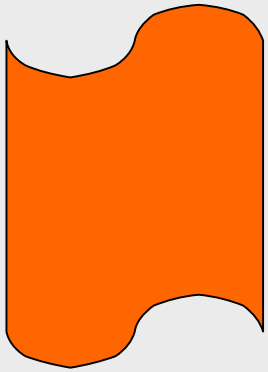
T

+

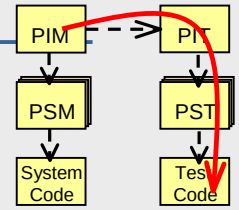


U2TP Model

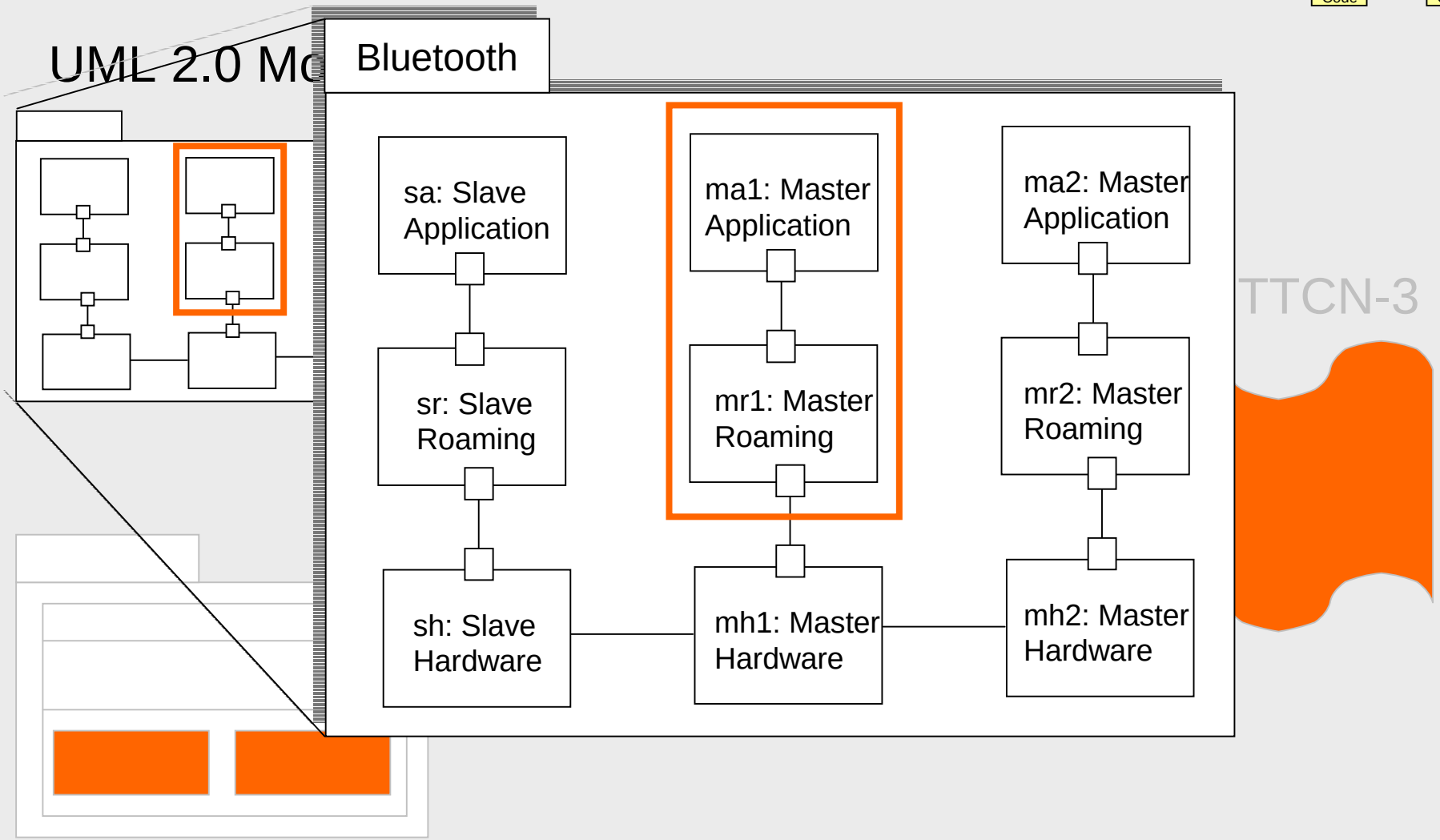
M

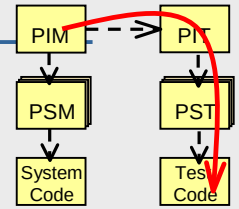


TTCN-3

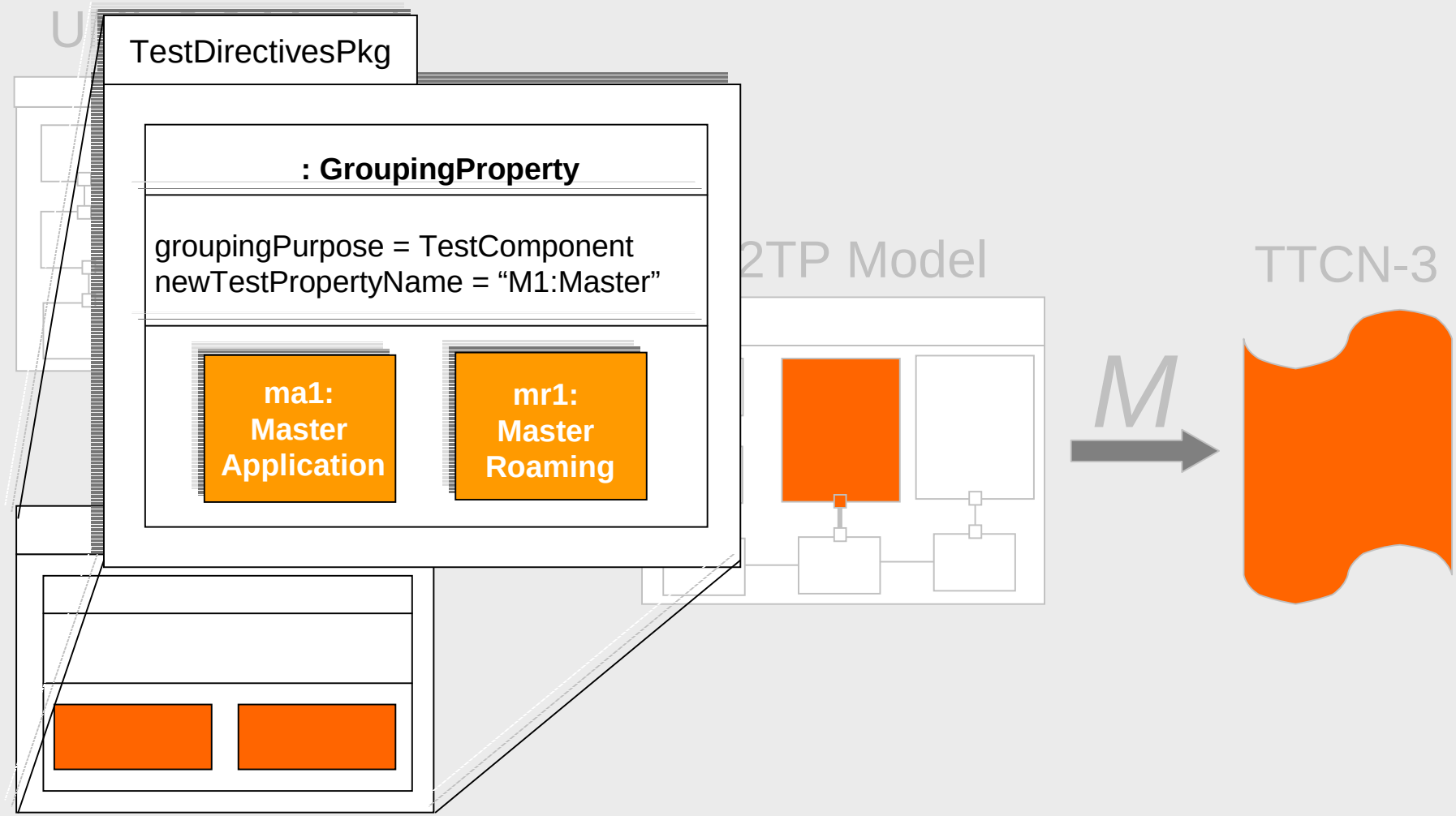


III. MDT in Action

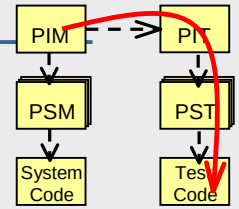




III. MDT in Action



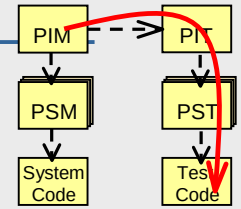
Test Directives Model



III. MDT in Action

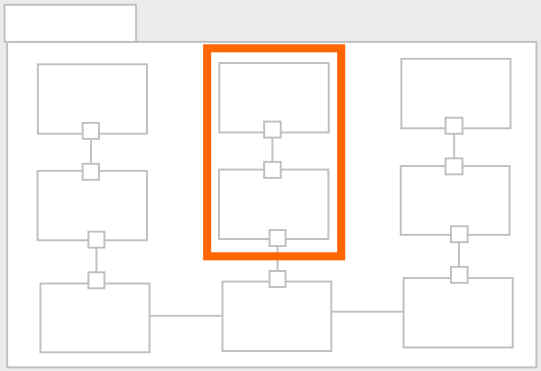
```
RULE SetExternalInterface(uml_encl1,uml_encl2,u2tp_encl)

    FORALL ::uml2::EncapsulatedClassifier@UMLsrc uml_encl1,
           ::uml2::EncapsulatedClassifier@UMLsrc uml_encl2

    WHERE  CheckExternalInterface(uml_encl1, uml_encl2)

    MAKE   ::u2tp::EncapsulatedClassifier@U2TPtgt u2tp_encl

    SET    u2tp_encl.ownedPort.required = uml_encl1.ownedPort.required,
           u2tp_encl.ownedPort.provided = uml_encl1.ownedPort.provided,
           u2tp_encl.ownedPort.required = uml_encl2.ownedPort.required,
           u2tp_encl.ownedPort.provided = uml_encl2.ownedPort.provided
;
```

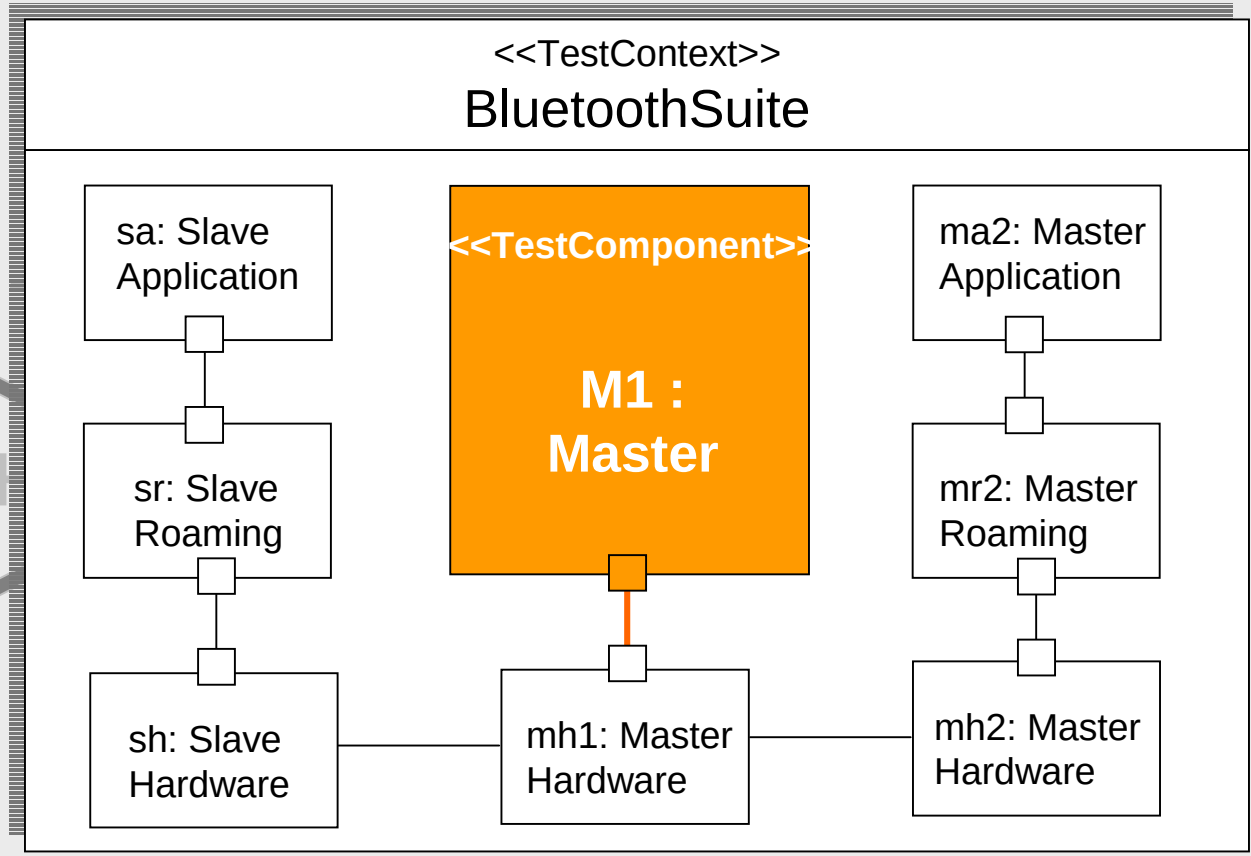
Test Directives Model

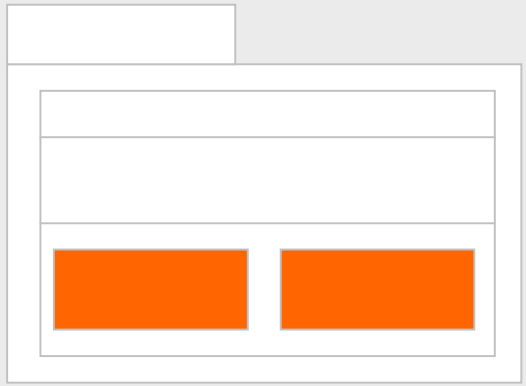


III. MDT in Action

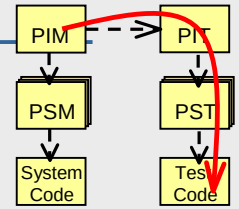
UML 2.0 Model







Test Directives Model

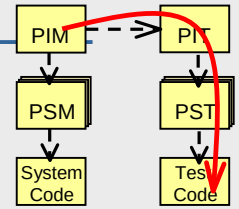


III. MDT in Action

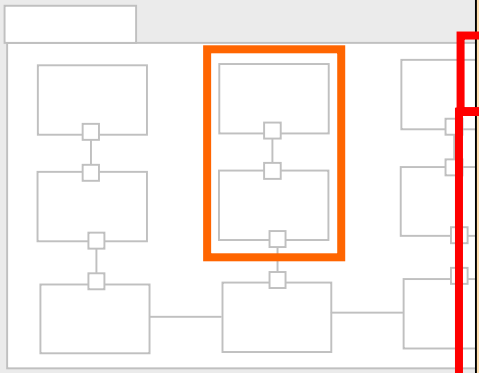
U2TP	TTCN-3
System under test (SUT)	Abstract test system interfaces.
Test component	TTCN-3 test component type.
Test configuration	Test configuration function with *create*, *start*, *connect/disconnect* and *map/unmap* operations.
...	...

Test Directives Model

III. MDT in Action

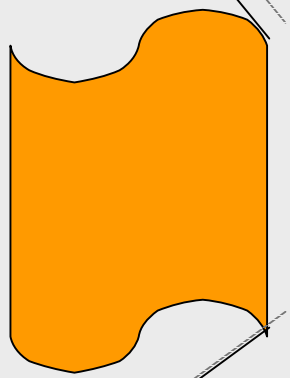


UML 2.0 Model



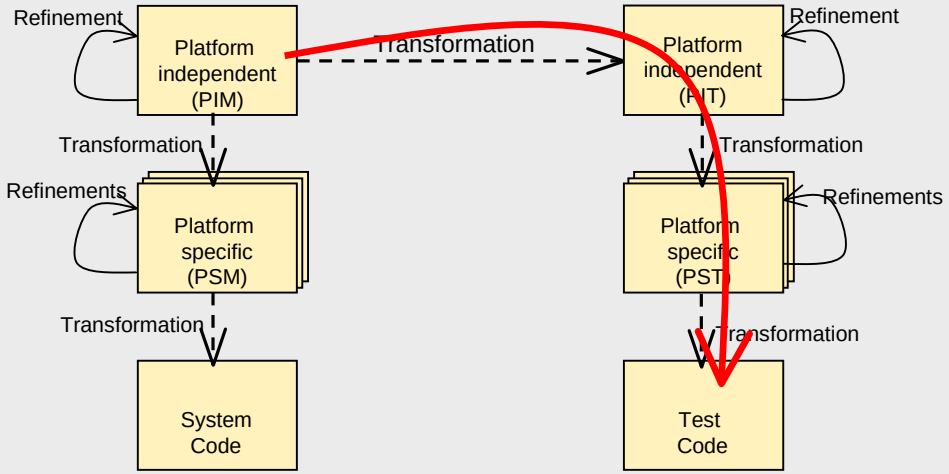
```
module BluetoothSuite {

function SetConfig(...) runs on mtc_type
  { ...
    M1 := Master_CType.create;
    connect(HW:hwiMaster, M1:masterHW);
    ... }

 testcase TestRoaming() runs on... system...
  {  ...
     var Master_CType M1;
     SetConfig (...);
     M1.start(Master1_Behavior());
     ... }

control
  { execute(TestRoaming() }
}
```

TTCN-3



Test Directives Model

IV. Summary



- Enhanced MDA Framework by test aspects for software quality assurance.
- Integrated and automated system and test development.
- System development strongly coupled with test development.

Thank you!