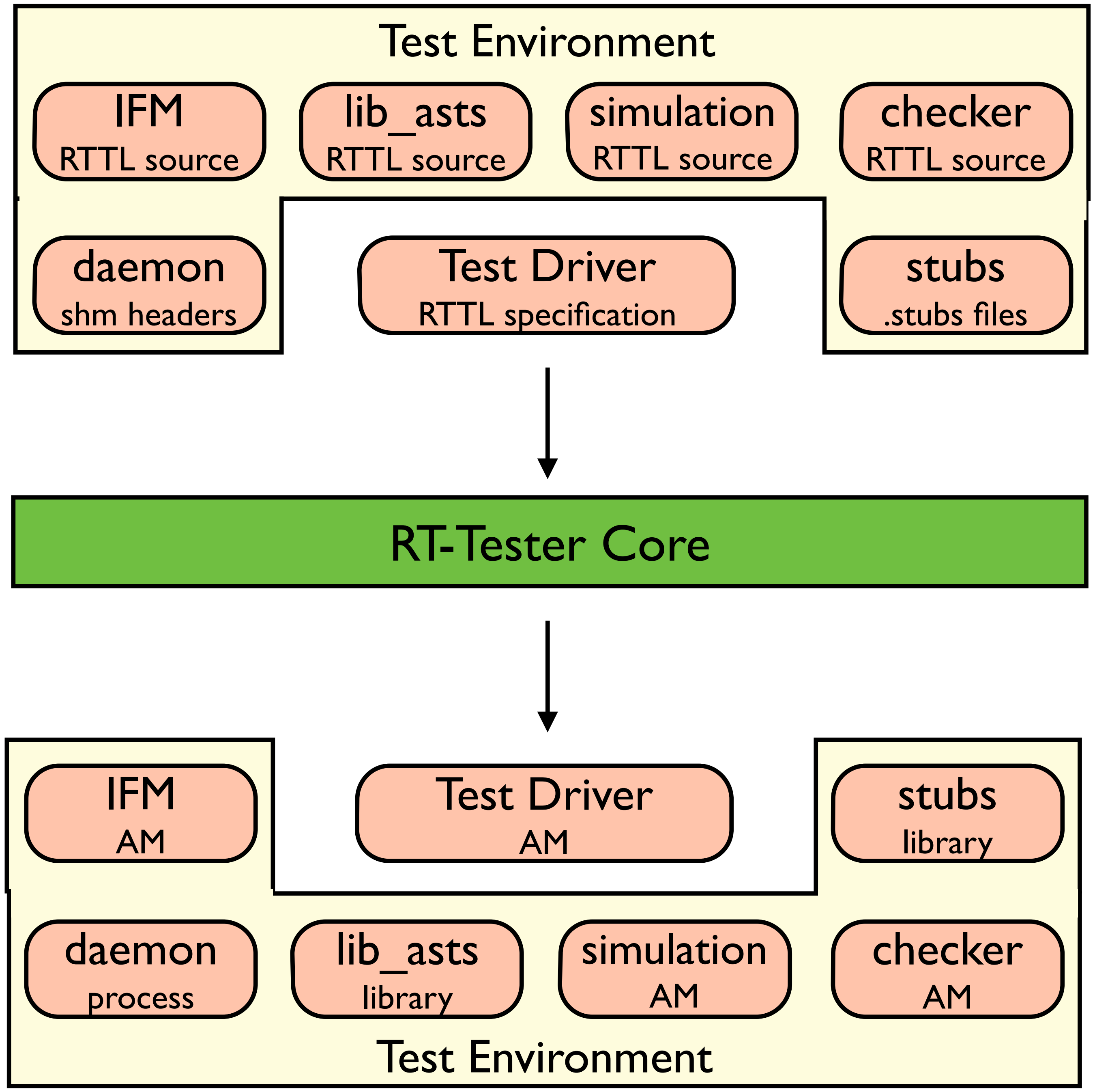
Content

- Concepts, Terms, Architecture
  - RT-Tester Core
  - RT-Tester TMS
  - RTT-MBT
- Creating Test Models
  - Template Models
  - Supported UML/SysML Elements
  - Modeling Styles and Stereotypes

Content

- Test/Checker Interface to Test Environment
- RTT-Plugin Perspective
- Preferences and Properties

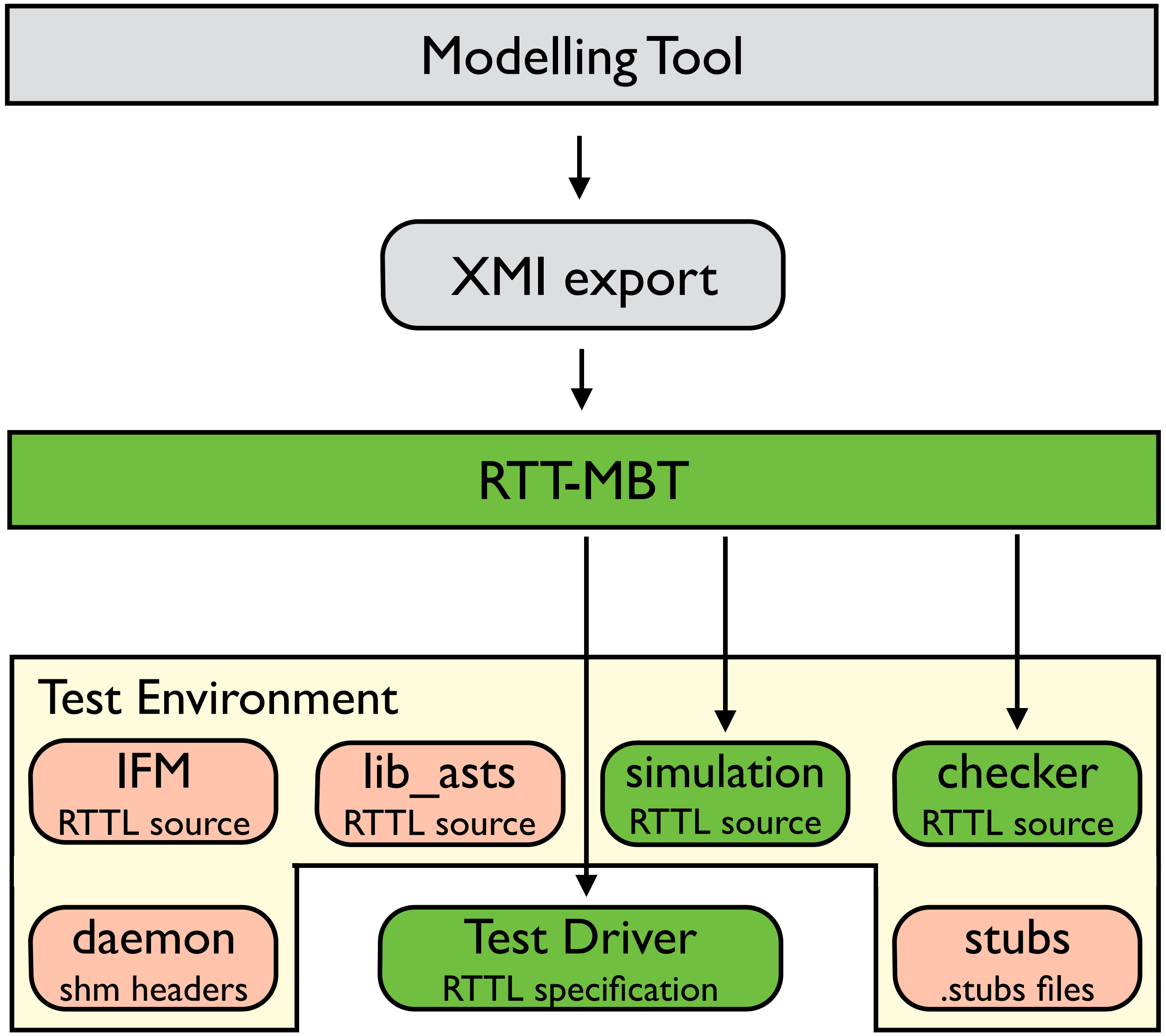
RT-Tester


Test Environment
IFM
RTTL source
lib_asts
RTTL source
simulation
RTTL source
checker
RTTL source
daemon
shm headers
Test Driver
RTTL specification
stubs
.stubs files
RT-Tester Core
IFM
AM
Test Driver
AM
stubs
library
daemon
process
lib_asts
library
simulation
AM
checker
AM
Test Environment

RT-Tester

- Manually crafted parts
    - Test Environment
    - Simulations
    - Checkers / Test Driver
- Benefits
    - Very flexible
    - Functionality directly visible
- Drawbacks
    - Hi cost for developing test procedures
    - Errors in Simulation and Checkers / Test Drivers in RTTL code

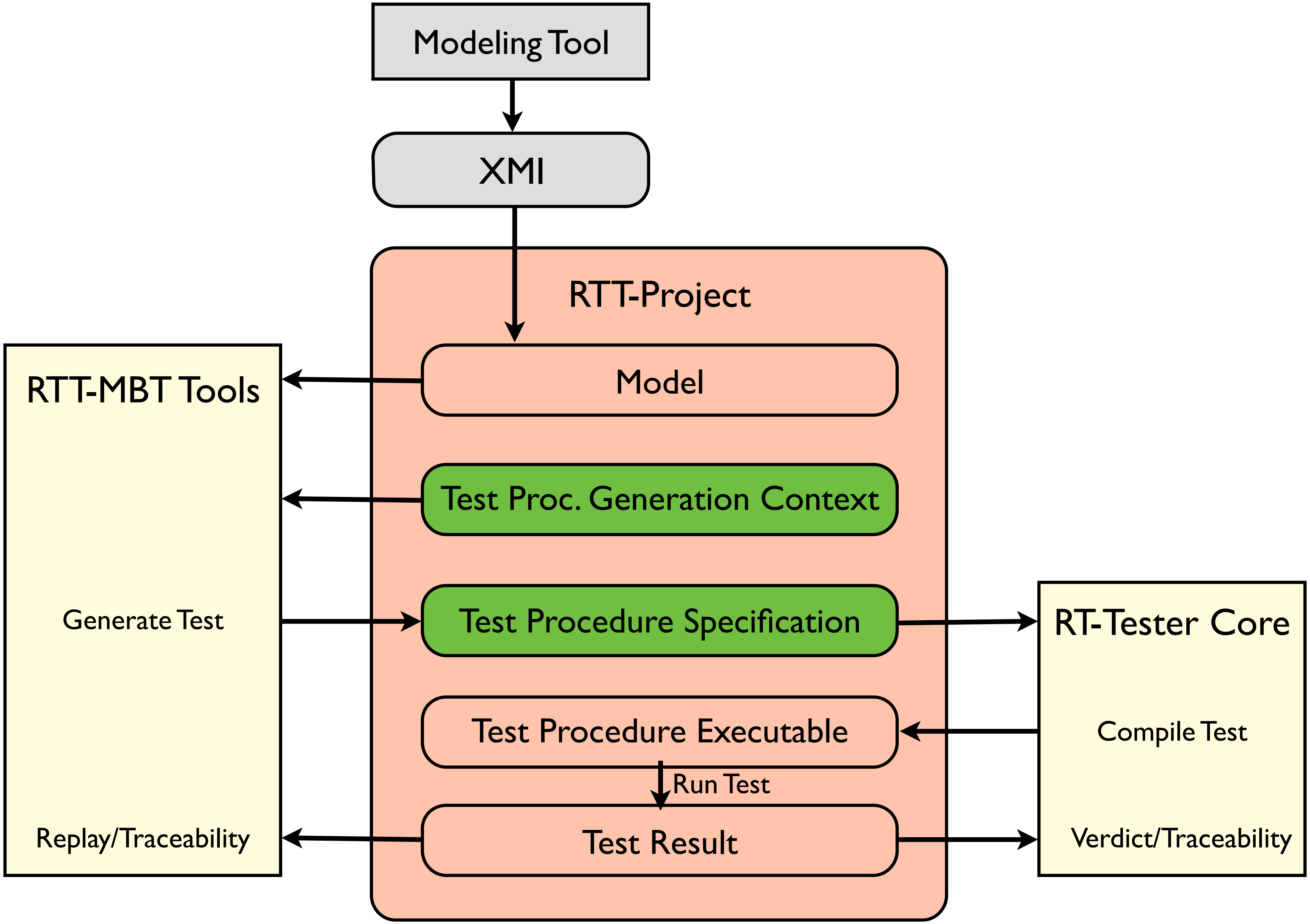
RTT-MBT


Modelling Tool
XMI export
RTT-MBT
Test Environment
IFM
RTTL source
lib_asts
RTTL source
simulation
RTTL source
checker
RTTL source
daemon
shm headers
Test Driver
RTTL specification
stubs
.stubs files

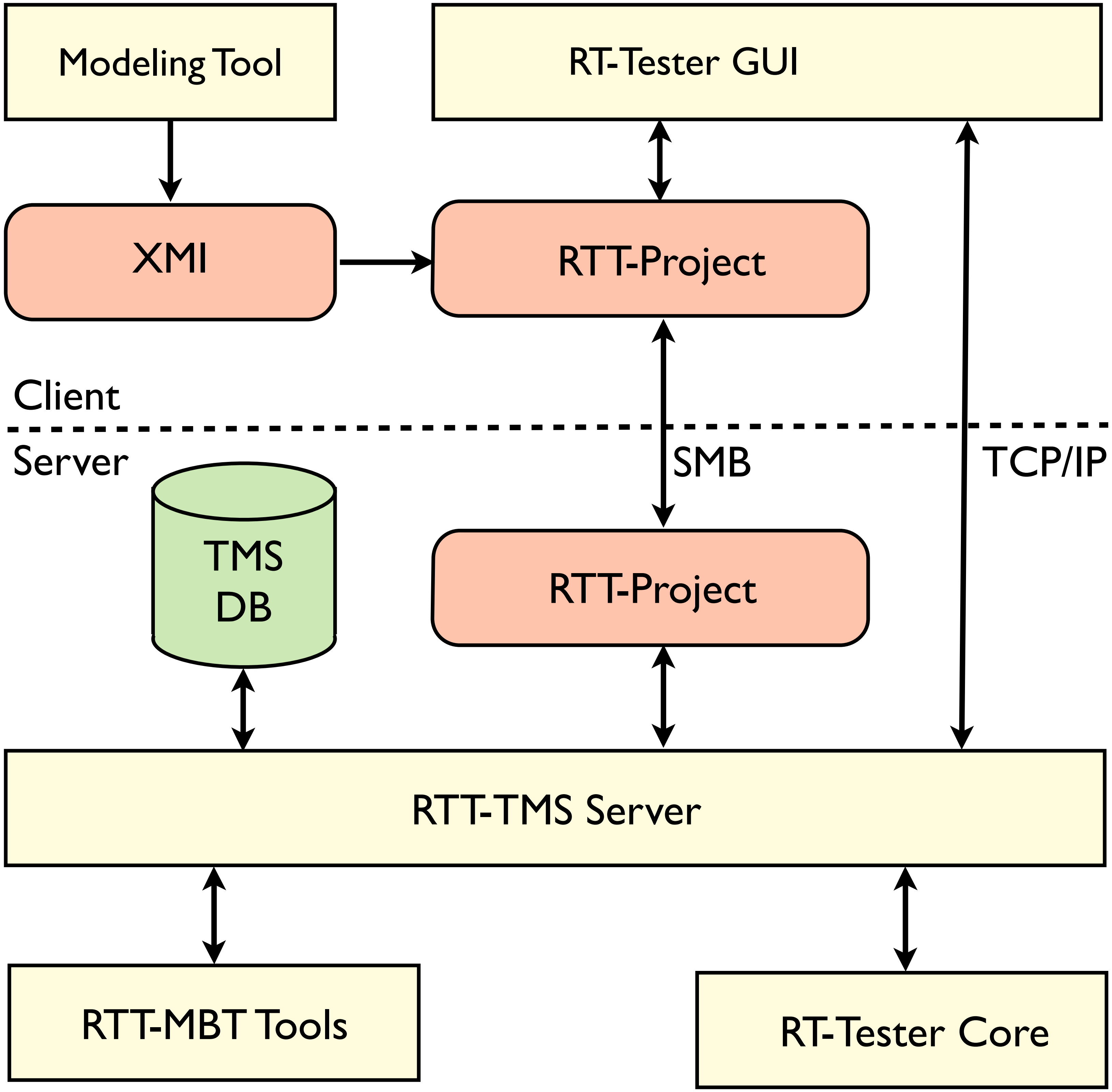
RT-Tester

- Manually crafted parts
  - Test Environment
- Generated parts
  - Simulations, Checkers, Test Drivers
- Benefits
  - Behaviour defined on model level
  - Correct RTTL code
  - Requirements tracing
  - Test coverage, test depth (MC/DC, Robustness)
- Drawbacks
  - Additional modelling training required
  - Model deployment, refinement costs

Tool Interaction



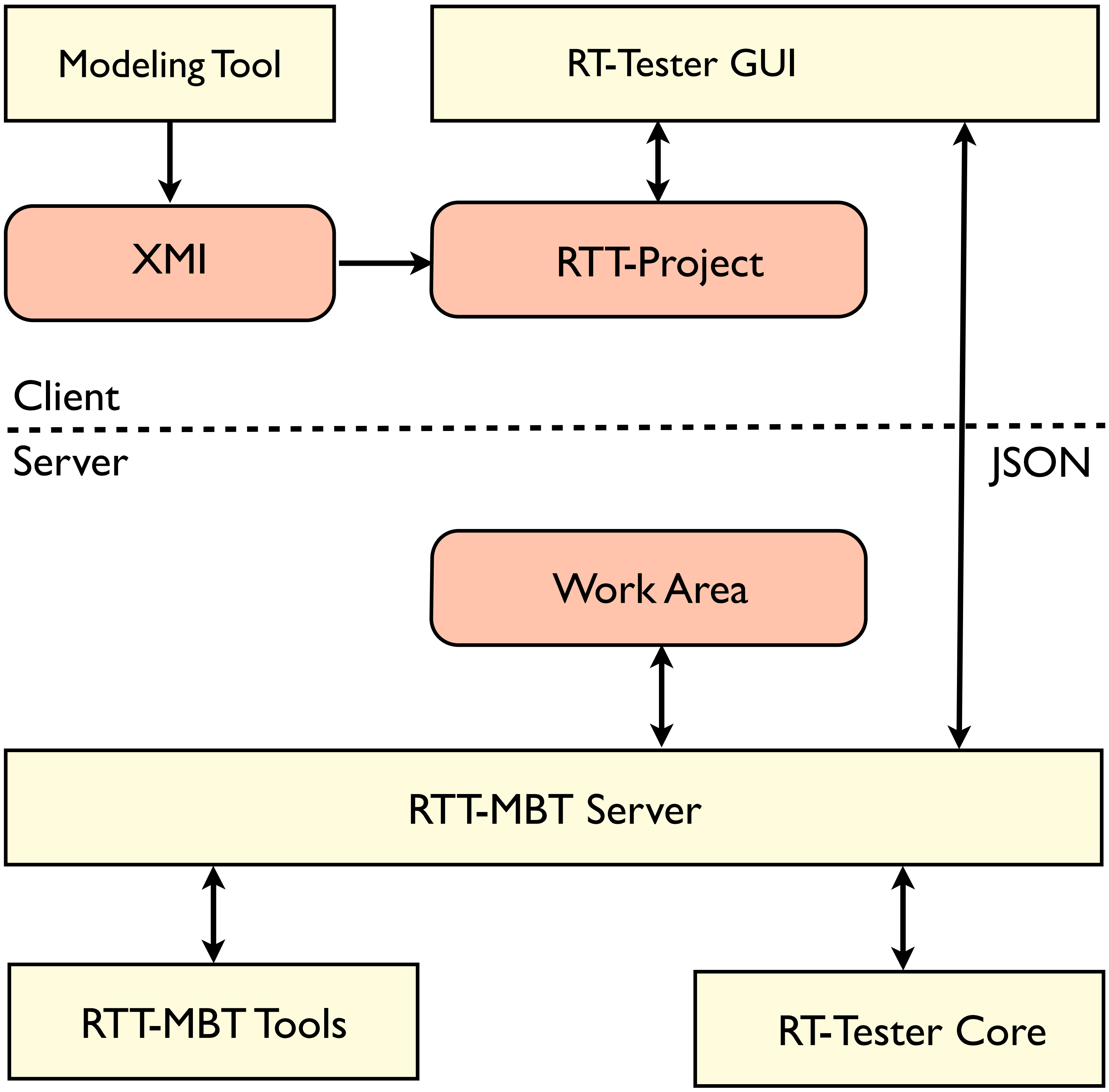
Managed Project



TMS Project

- Benefits
  - Easy installation of the client
  - Projects can be shared between clients (SMB network drive)
  - Good for teams of testers on local area networks
- Drawbacks
  - Complex installation of the server
- Requirements
  - Client: RT-Tester GUI available for Windows and Linux
  - Client: Network share must be available
  - Server: RTT-MBT Tools and RT-Tester Core available/installed
  - Server: SMB File server and database server
  - Server: Verified TMS available for Linux

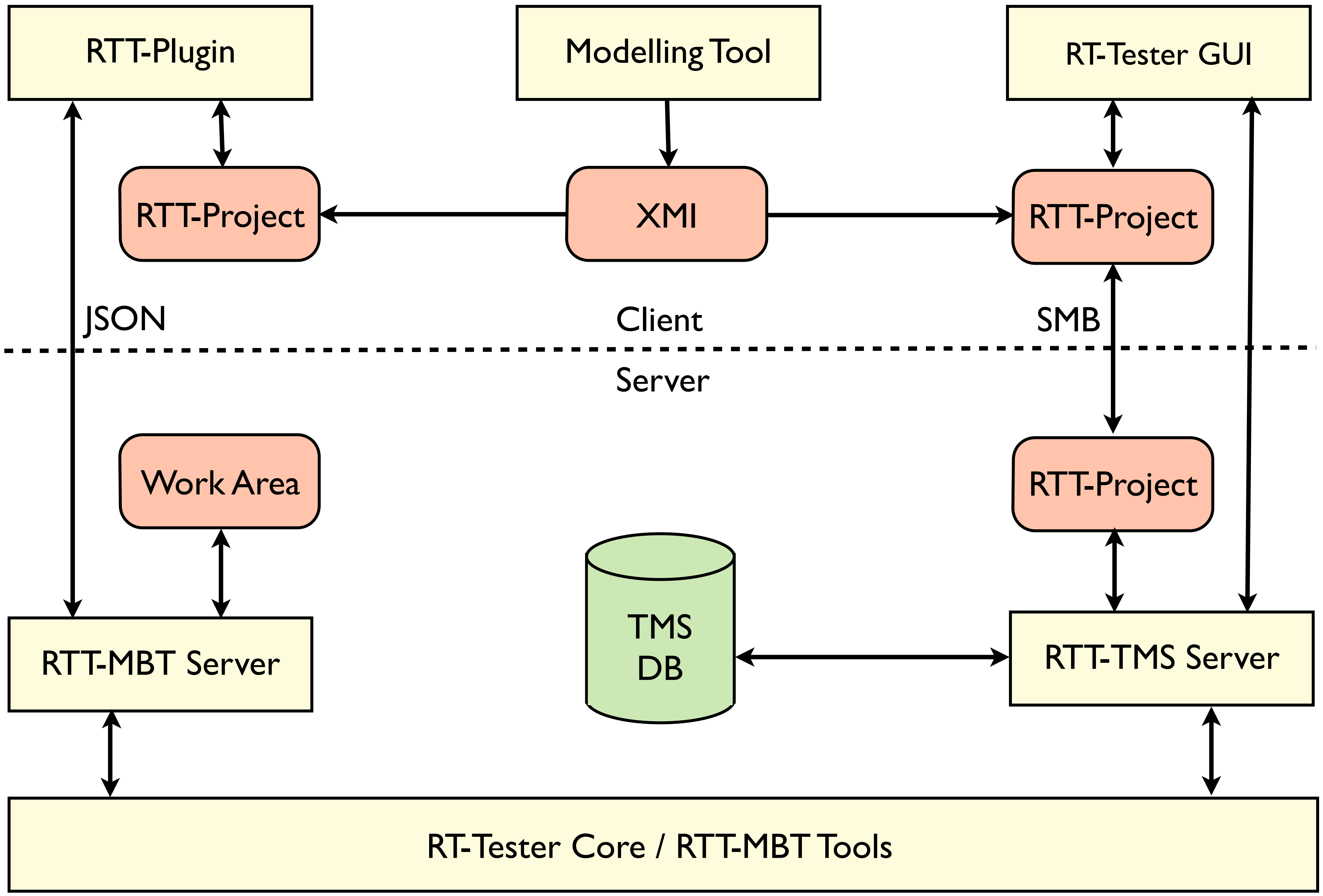
RTT-Plugin Project



RTT-Plugin Project

- Benefits
  - Easy installation of the client (free Eclipse plugin)
  - Local project under version control of your choice
  - Can be used with WAN network connection to RTTMBT server
- Drawbacks
  - No database connection (yet)
  - Resources have to be synchronised through version control
- Requirements
  - Client: Eclipse (Kepler) available for Windows, Linux and Mac OS X
  - Client: RTT-MBT server must be reachable
  - Server: RTT-MBT Tools and RT-Tester Core available/installed (Linux)

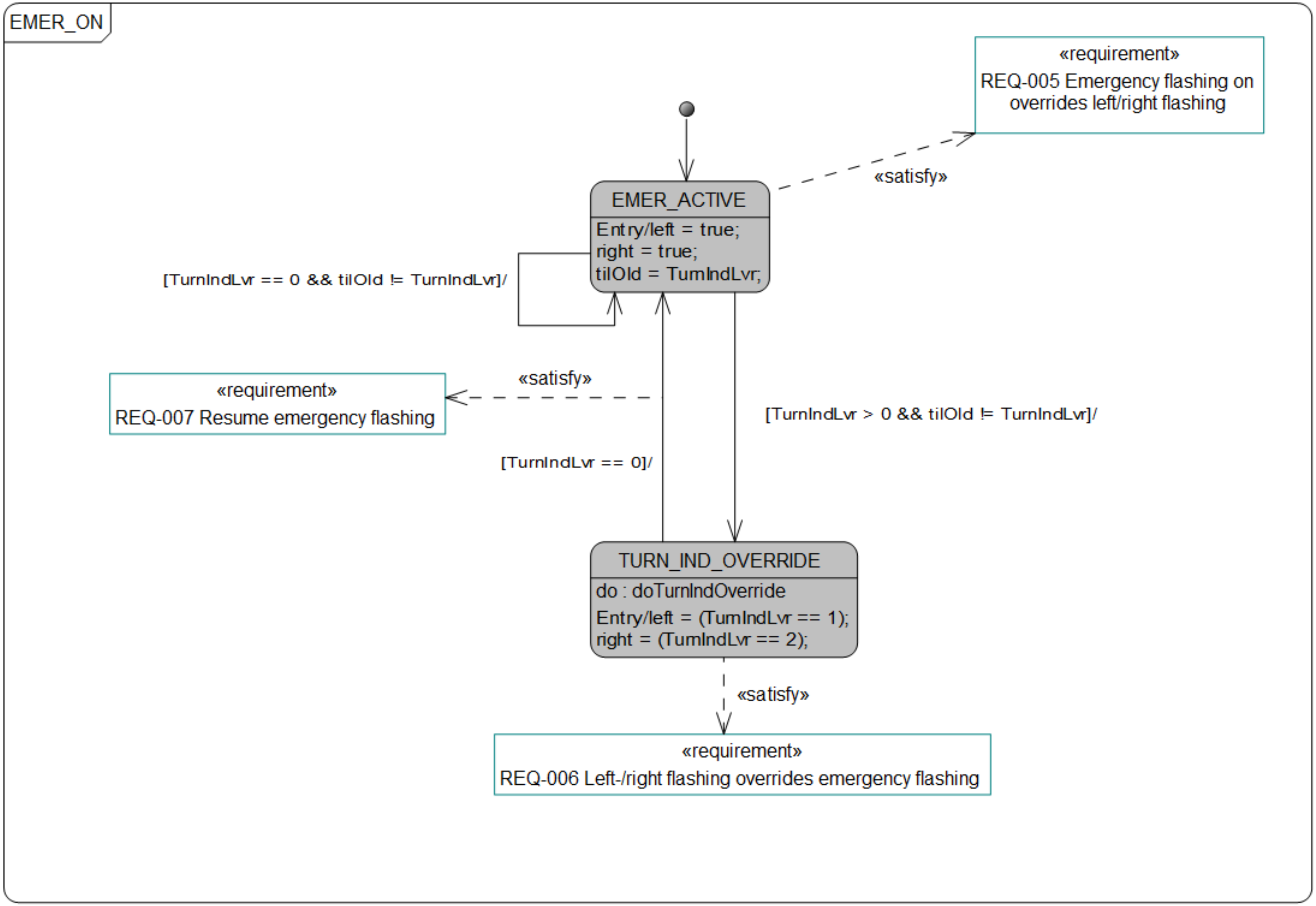
Compare Client Architecture



UML Elements

- Supported UML Elements
  - Composite Structure Diagrams with
    - Classes
    - Interfaces
    - Constraints (associated with classes)
    - Requirements
  - State Diagrams with
    - Initial States
    - Atomic States
    - Submachine States
    - Transitions
    - Requirements (associated with states and transitions)
  - Requirement Diagrams
    - Satisfy relations

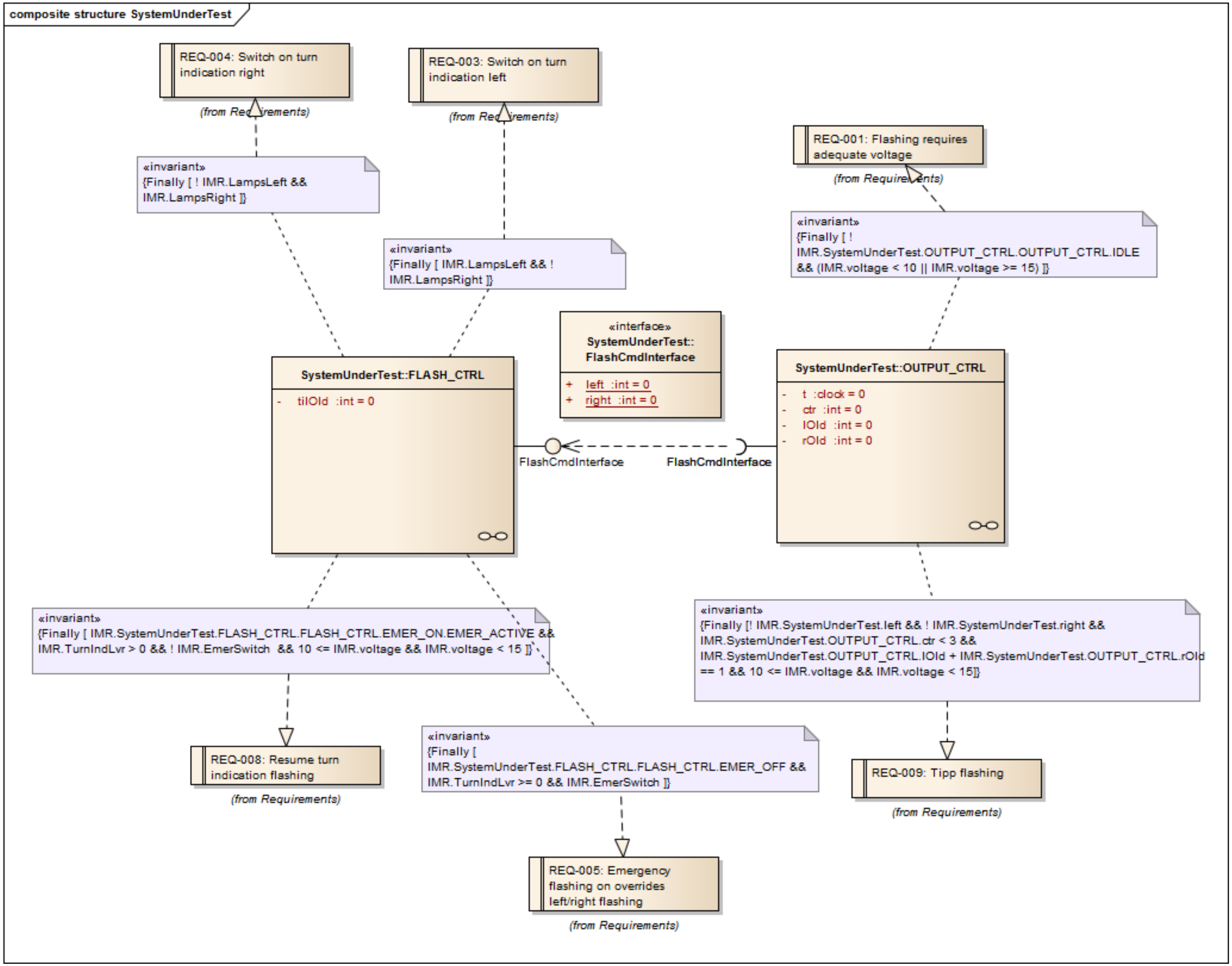
Example


EMER_ON
«requirement»
REQ-005 Emergency flashing on overrides left/right flashing
«satisfy»
EMER_ACTIVE
Entry/left = true;
right = true;
tilOld = TumIndLvr;
[TurnIndLvr == 0 && tilOld != TurnIndLvr]/
«requirement»
REQ-007 Resume emergency flashing
«satisfy»
[TurnIndLvr > 0 && tilOld != TurnIndLvr]/
[TurnIndLvr == 0]/
TURN_IND_OVERRIDE
do : doTurnIndOverride
Entry/left = (TumIndLvr == 1);
right = (TumIndLvr == 2);
«satisfy»
«requirement»
REQ-006 Left-/right flashing overrides emergency flashing

UML Elements

- System Components
  - Modelled as class hierarchy
  - Below top level elements SystemUnderTest and TestEnvironment
  - Behaviour of component defined in state machine inside the class
  - Requirements linked to LTL-formula constraints linked to component

Example



composite structure SystemUnderTest

REQ-004: Switch on turn indication right

(from Requirements)

REQ-003: Switch on turn indication left

(from Requirements)

REQ-001: Flashing requires adequate voltage

(from Requirements)

«invariant»
{Finally [! IMR.LampsLeft &&
IMR.LampsRight]}

«invariant»
{Finally [IMR.LampsLeft && !
IMR.LampsRight]}

«invariant»
{Finally [!
IMR.SystemUnderTest.OUTPUT_CTRL.OUTPUT_CTRL.IDLE
&& (IMR.voltage < 10 || IMR.voltage >= 15)]}

«interface»
SystemUnderTest::
FlashCmdInterface

+ left :int = 0
+ right :int = 0

SystemUnderTest::FLASH_CTRL

- tilOld :int = 0

SystemUnderTest::OUTPUT_CTRL

- t :clock = 0
- ctr :int = 0
- lOld :int = 0
- rOld :int = 0

FlashCmdInterface

FlashCmdInterface

«invariant»
{Finally [IMR.SystemUnderTest.FLASH_CTRL.FLASH_CTRL.EMER_ON.EMER_ACTIVE &&
IMR.TurnIndLvr > 0 && ! IMR.EmerSwitch && 10 <= IMR.voltage && IMR.voltage < 15]}

«invariant»
{Finally [! IMR.SystemUnderTest.left && ! IMR.SystemUnderTest.right &&
IMR.SystemUnderTest.OUTPUT_CTRL.ctr < 3 &&
IMR.SystemUnderTest.OUTPUT_CTRL.lOld + IMR.SystemUnderTest.OUTPUT_CTRL.rOld
== 1 && 10 <= IMR.voltage && IMR.voltage < 15]}

«invariant»
{Finally [
IMR.SystemUnderTest.FLASH_CTRL.FLASH_CTRL.EMER_OFF &&
IMR.TurnIndLvr >= 0 && IMR.EmerSwitch]}

REQ-008: Resume turn indication flashing

(from Requirements)

REQ-009: Tipp flashing

(from Requirements)

REQ-005: Emergency flashing on overrides left/right flashing

(from Requirements)

SysML Elements

- Supported UML Elements
- Block Definition Diagrams and Internal Block Definition Diagrams with
  - Blocks
  - Interfaces, standard ports
  - `<<interface>>` blocks, Flow Ports, Item Flows
  - Constraints (associated with classes)
  - Requirements (satisfied by constraints)
- State Diagrams (see UML)
- Requirement Diagrams (see UML)

SysML Elements

- System Components
  - Modelled as block hierarchy
  - Below top level blocks SystemUnderTest and TestEnvironment
- Interfaces, `<<interface>>` blocks
  - Name will be ignored
  - attributes are generated as (global) variables with scoped names
  - attributes can be used in state machine actions and operations
- Class / block attributes
  - are generated as (global) variables with scoped names
  - can be used like interface attributes

Expressions

- Functions/Operations
  - Defined as part of a component (class/block)
  - scoped names
- Data types
  - C basic types
  - special type **`timer`** or **`clock`** to model timing
- Expressions
  - C-like Assignments
  - if-then-else expressions
  - timer expressions (e.g. **`t.reset();`**, **`t.elapsed(340)`**)

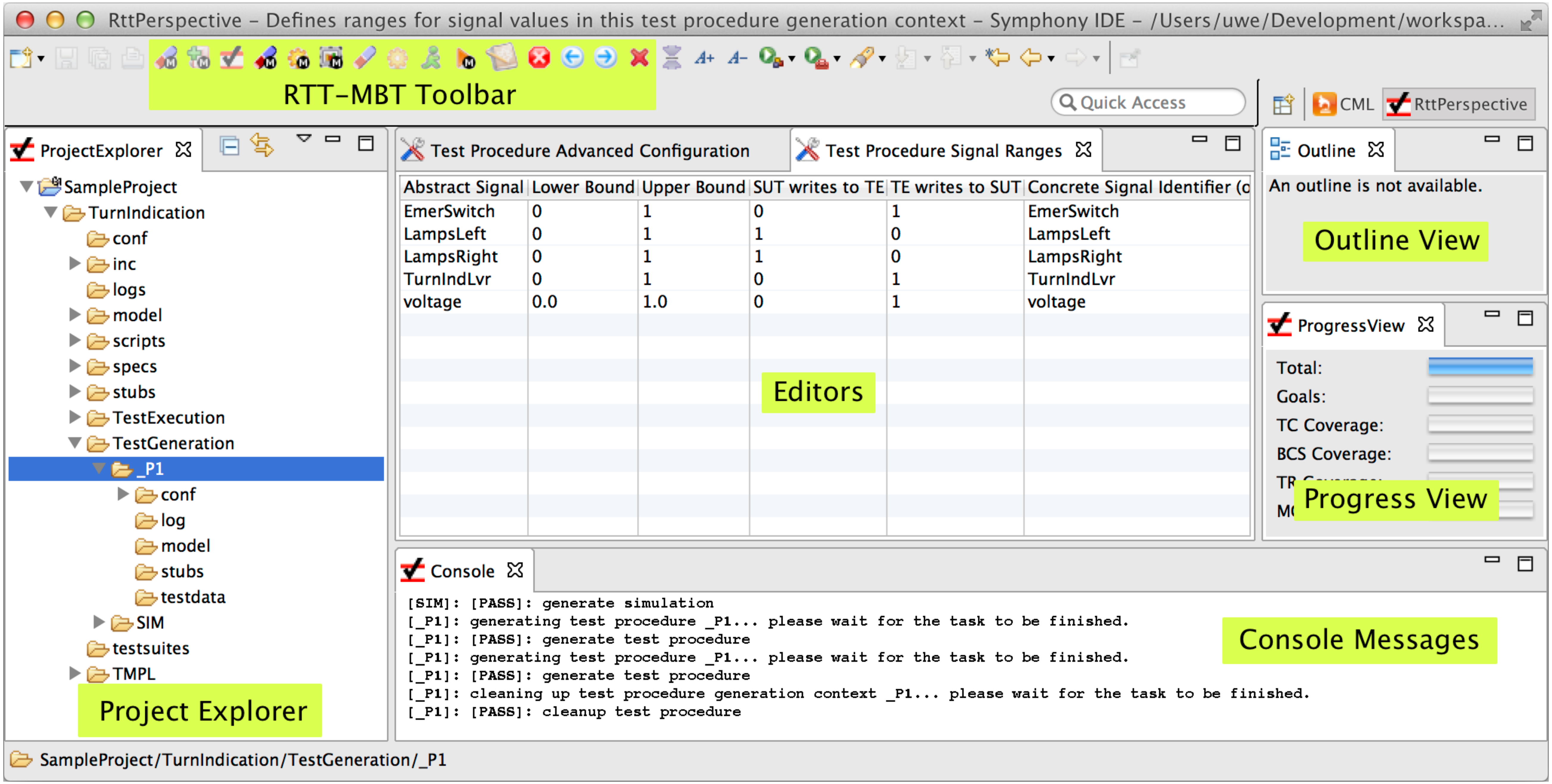
Test Generation

- Test Procedure Generation Context
  - Defines goals for the solver for a single test procedure generation
  - Generated test procedure contains
    - checkers for modelled behaviour
    - Test driver (time triggered stimulations)
    - interface to test environment as defined in the test model
    - RT-Tester configuration files and AMs
- Test Procedure
  - The result of a generation controlled by a test procedure generation context
  - RT-Tester test procedures can must be compiled before they can be executed
  - complete execution consist of the steps (clean), compile, run, doc and replay.

Interface Modules

- generic interface pattern for RTT-MBT generated test procedures:
  - In generated file `rttInterface.h`
  - `typedef struct rttInterface_t`
  - Contains all TE2SUT and SUT2TE interface attributes
  - Pre and post values stored in `rttIOPre` and `rttIOPost`
  - This allows a generic pattern for interface modules to SUT
  - Interface modules allow the re-use of test models on different TI levels

RTT-Perspective



RttPerspective – Defines ranges for signal values in this test procedure generation context – Symphony IDE – /Users/uwe/Development/workspa...

RTT-MBT Toolbar

ProjectExplorer

- SampleProject
  - TurnIndication
    - conf
    - inc
    - logs
    - model
    - scripts
    - specs
    - stubs
    - TestExecution
    - TestGeneration
      - _P1
        - conf
        - log
        - model
        - stubs
        - testdata
      - SIM
    - testsuites
    - TMPL

Project Explorer

Test Procedure Advanced Configuration | Test Procedure Signal Ranges

Abstract Signal	Lower Bound	Upper Bound	SUT writes to TE	TE writes to SUT	Concrete Signal Identifier (c
EmerSwitch	0	1	0	1	EmerSwitch
LampsLeft	0	1	1	0	LampsLeft
LampsRight	0	1	1	0	LampsRight
TurnIndLvr	0	1	0	1	TurnIndLvr
voltage	0.0	1.0	0	1	voltage

Editors

Outline

An outline is not available.

Outline View

ProgressView

Total:
Goals:
TC Coverage:
BCS Coverage:
TR Coverage:
M

Progress View

Console

```
[SIM]: [PASS]: generate simulation
[_P1]: generating test procedure _P1... please wait for the task to be finished.
[_P1]: [PASS]: generate test procedure
[_P1]: generating test procedure _P1... please wait for the task to be finished.
[_P1]: [PASS]: generate test procedure
[_P1]: cleaning up test procedure generation context _P1... please wait for the task to be finished.
[_P1]: [PASS]: cleanup test procedure
```

Console Messages

SampleProject/TurnIndication/TestGeneration/_P1

Feedback an Support

- Feedback on RTT-MBT
  - Problems
  - Feature Requests
- Feedback on RTT-Plugin
  - User Interface
  - Feature / Functionality requests
  - Error reports
- Feedback on Papyrus modelling restrictions
  - with rational
- email: uschulze@informatik.uni-bremen.de
- RTT-MBT Server: algieba.informatik.uni-bremen.de
- VM: rttmbtx64.2013-09-06.rtt-swi-6.0.4.9.7-1.rtt-tms_2.0.4-1.rtt-mbt-9.1-1.0.0-dev.vmdk
- Downloads from https://urmel.informatik.uni-bremen.de/openETCS/
  - user: openETCS
  - pass: atdenHoph9