Michael Wahler, E. Ferranti, R. Steiger, R. Jain, K. Nagy. ABB Corporate Research, Industrial Software Systems

CAST: Automating Software Tests for Embedded Systems

Motivating Example
Manual System Tests

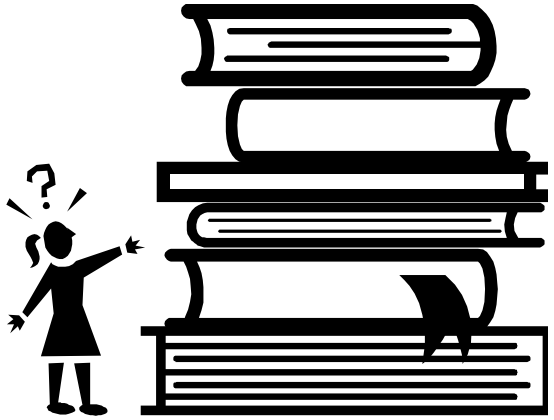
Action 1	Initialize system according to Table 17.3.
Expected result	 and 
Action 2	Set ac.on to true.
Expected result	 and 
Action 3	Set Room.temperature to a random value between 25 and 30.
Expected result	 and 
Action 4	Within 5 seconds, the following result may **not** occur.
Expected result	 and 

Tester / Date

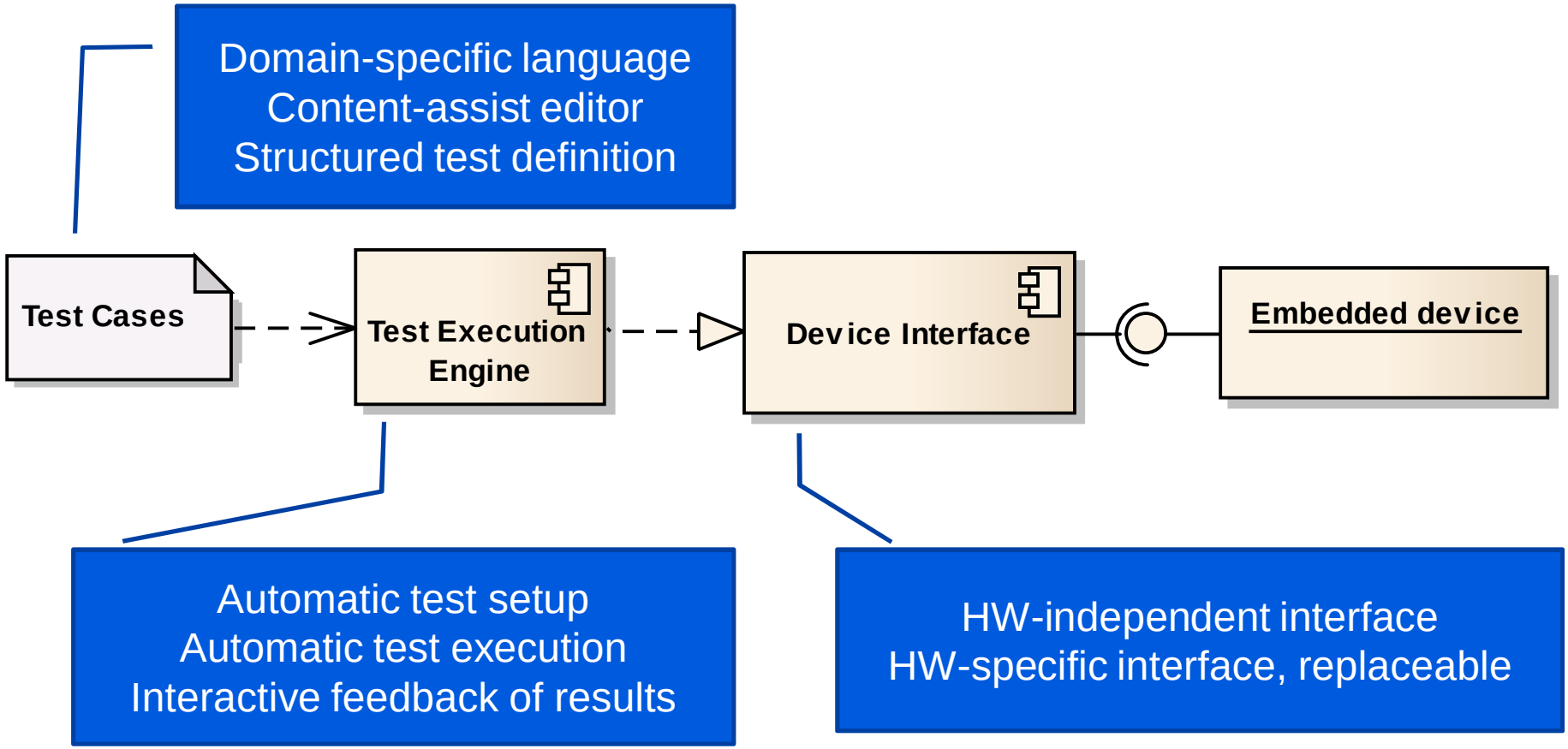
Motivation

- Bloated test case specification
- Time consuming
- Error prone
- Not accurate enough
- Repetitive work

To what extent can we automate test execution?



Outline of the Solution
CAST: Computer-Aided Specification and Testing

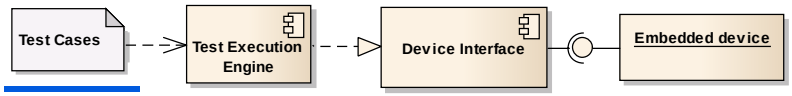


Test Specification and Management
The TESLA Domain-Specific Language

- Basic blocks
  - Actions
  - Checks
- Critical and noncritical checks
- Time intervals for checks
  - Example:

$$\forall t \in [t_{start}, t_{end}] \quad \neg P(t)$$

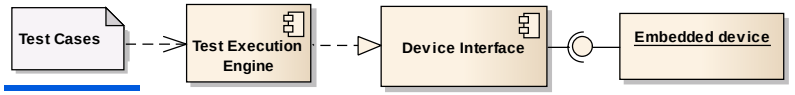
```
critical check never in [0, 5] sec {
  Room.overload occurred
  ac.power > MAX_POWER
}
}
```



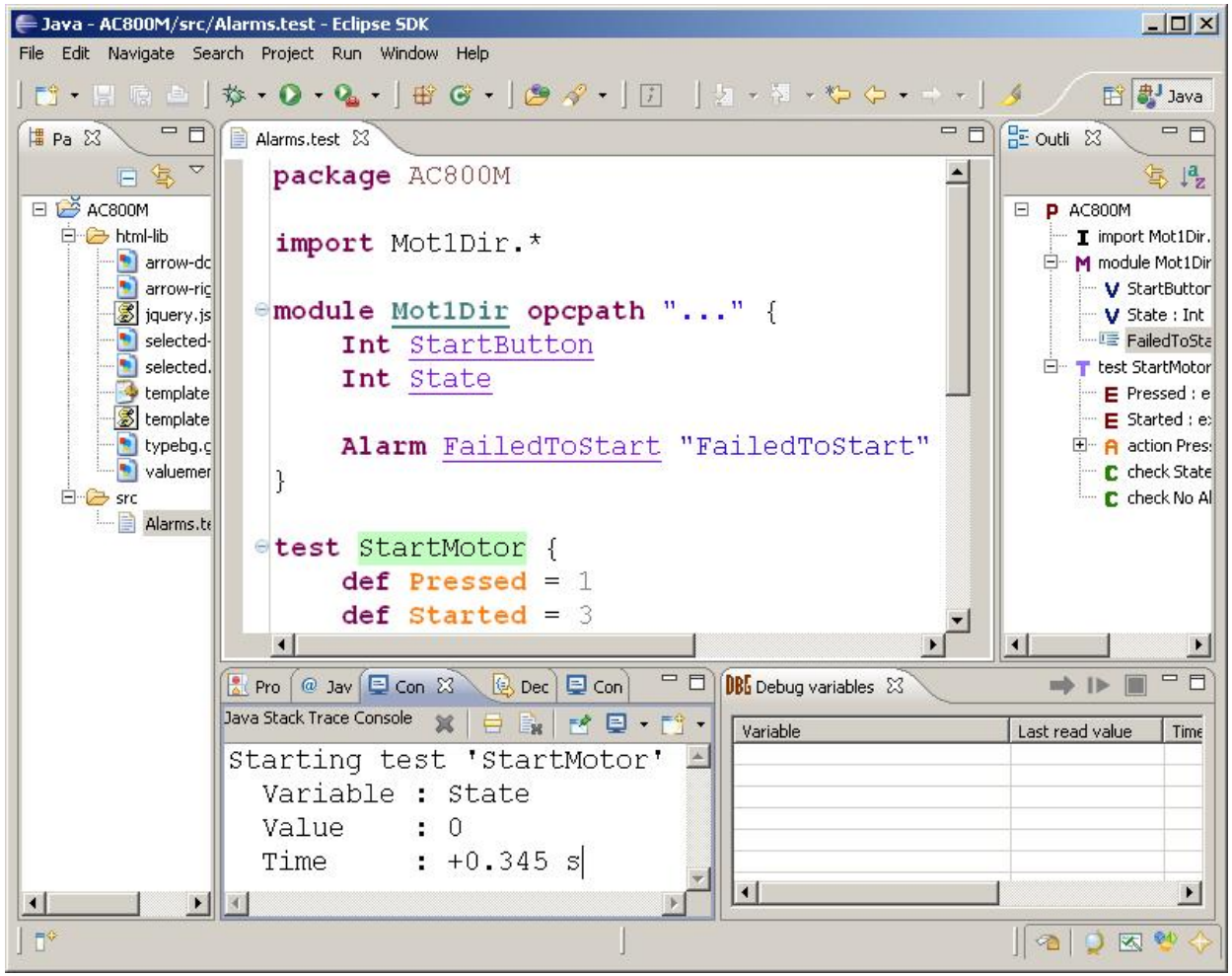
Test Specification and Management
TESLA Example

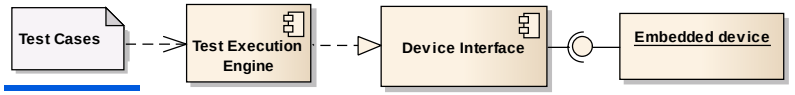
```
package TestGroup1

import Definitions.*

setup InitialSetup {
  action "Clean set-up" {
    Room.ac1.on = false
    Room.ac2.on = false
    download
  }
}
test ACWorking (
  Types.ACStatus ac
  Int room_temperature) {

  def MAX_POWER = 5

  action {
    ac.on = true
    Room.temperature = room_temperature
  }
  critical check never in [0, 5] sec {
    Room.overload occurred
    ac.power > MAX_POWER
  }
}
```

```
test suite MyTests {
  for ( acStatus <- {Room.ac1, Room.ac2} ) {
    run ACWorking using InitialSetup
    (
      ac = acStatus
      room_temperature = random[25, 30]
    )
  }
}
```



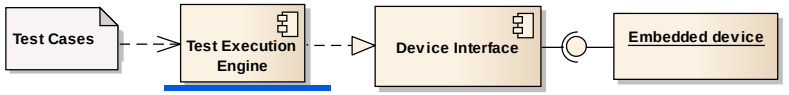
Test Specification and Management
Tool Integration



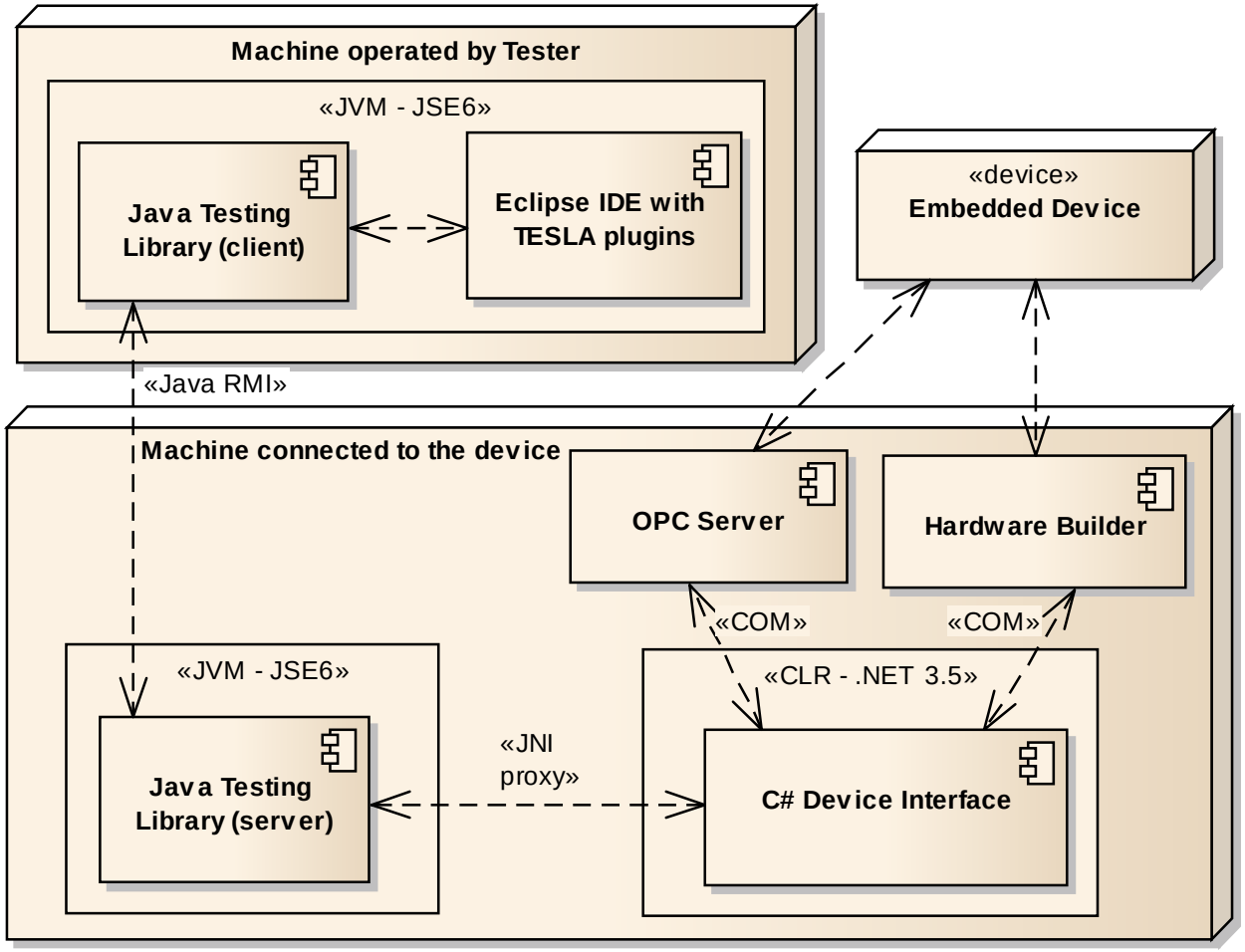


Test Execution Engine

- Interpreter for test case definitions
  - Phase 1: Set up variable subscriptions
  - Phase 2: Execute actions and checks
- Automatic tests
- Interactive tests
- Perfect repeatability
- Test results can be shown in editor



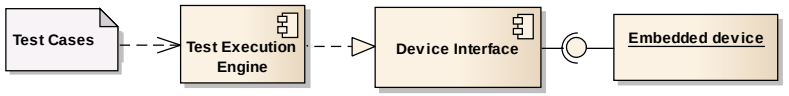
Overall Architecture



Validation
Challenges

- Whole test spec → TESLA ?

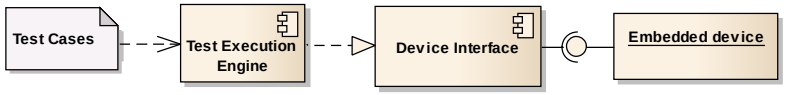
- How did we measure success?
  - Focus on a representative subset
  - Results = feedback from test engineers



Validation
Results

- Shorter execution time
- Conciseness of test specification
- Perfect repeatability
- Better time accuracy
- Less repetitive work

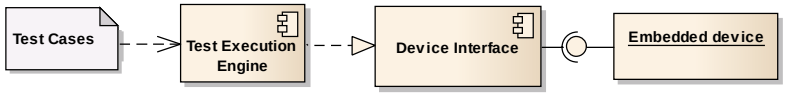
"*Creating my first test case was really straightforward and was even some kind of fun... :-) I really wish we had had this tool before we started all our testing activities and we could have written all our test specs in this environment.*"



Conclusions

- Automation possible for a majority of tests
- Expectations of test engineers exceeded
- Approach is adaptable to other systems
- Future work:
  - Formal semantics for test cases
  - Formalization of requirements

We have made real people real happy!



Power and productivity
for a better world™