SIEMENS

SIEMENS CORPORATE RESEARCH

UML-based Test Generation and Execution

Axel Ruder
Email:axel.ruder@siemens.com
+1 (609) 734 3632

Software Engineering Department
Siemens Corporate Research
Princeton, NJ

1

SIEMENS

SIEMENS CORPORATE RESEARCH

Outline

- Research at SCR
- Model based testing
  - Category/Partition Method
  - TDE/UML for Unit/Integration Testing
  - TDE/UML for System Testing
- Benefits
- Summary

2

SIEMENS

SIEMENS CORPORATE RESEARCH

Corporate Technology:-
About 1,750 Researchers and Developers Worldwide

3

SIEMENS

SIEMENS CORPORATE RESEARCH

Motivation

- Siemens is one of the largest software development houses in the world (> 30.000 developers worldwide)
- Siemens software ranges from embedded to enterprise level software systems
- These complex, distributed systems need to be modeled using some standardized notation => UML
- For embedded systems, for example telecom systems and reactive components we need to model their behavior based on statecharts and sequence diagrams
- For enterprise level applications, for example hospital IT systems and web-based applications we need to model their behavior based on Use Cases and Activity diagrams
- With this presentation, I'll be providing you an overview of our techniques, tools and their applications

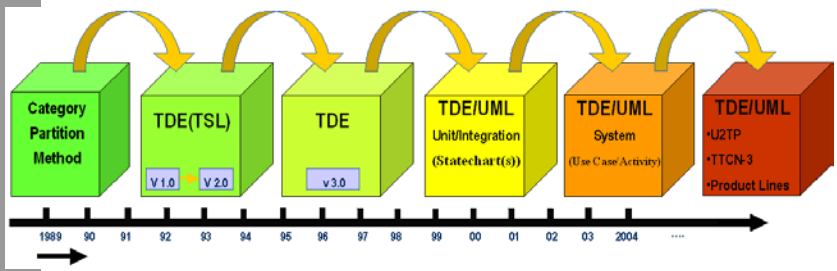
4

SIEMENS

SIEMENS CORPORATE RESEARCH

Testing Research at SCR

- Focus on developing efficient techniques and tools for test case generation
- Coupling them with suitable execution tools
- Specification-based black-box tests
- Testing based on UML diagrams
- Support of all test levels:
  - Unit testing TDE/UML 1999
  - Integration testing TDE/UML 2000
  - System testing TDE/UML 2003



5

SIEMENS

SIEMENS CORPORATE RESEARCH

The Category-Partition Method

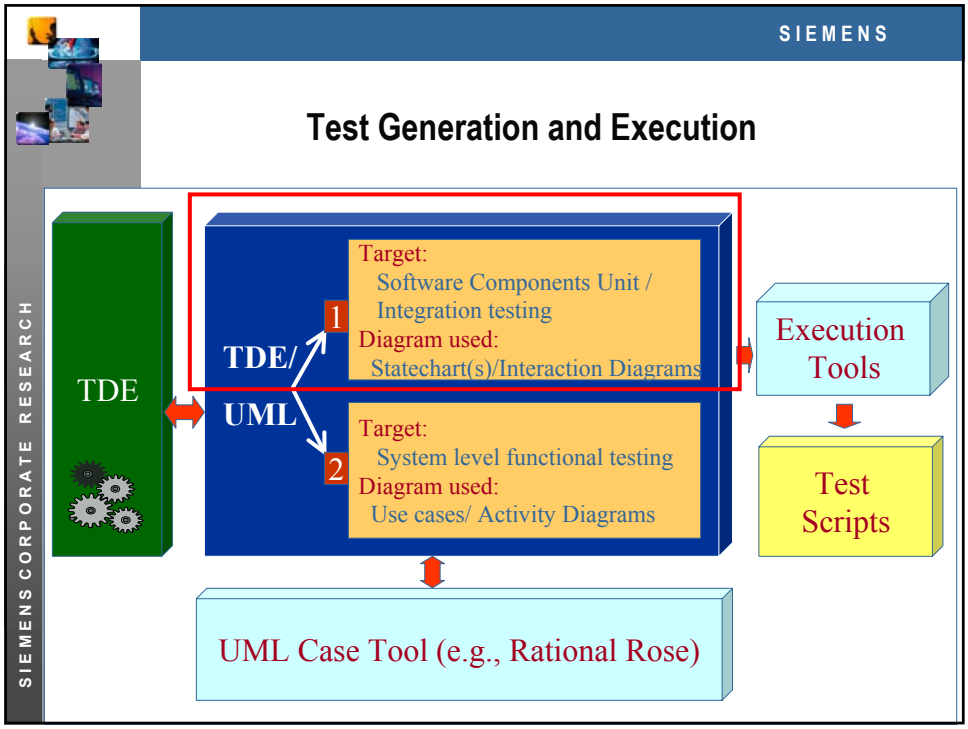
- The method identifies behavioral equivalence classes within the structure of a SUT
- A category or partition is defined by specifying all possible data choices that it can represent.
  - Choices: data values, references to other categories or partitions, or a combination of both.
- A test design is created using TSL - Test Specification Language

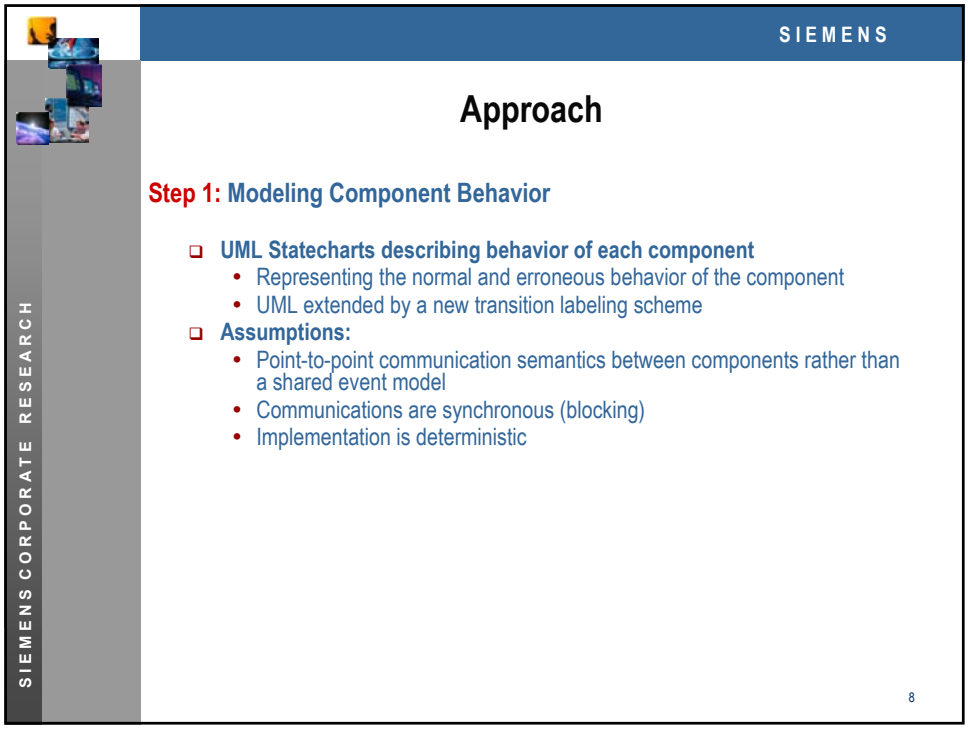
For detailed explanation, see Boris Beizer, "**Black-Box Testing : Techniques for Functional Testing of Software and Systems", John Wiley 1995.**

Example:

```
Test Bank_Account {
  frame value expr
    "$operation $account_number $amount
     $operation $account_number $amount
     $operation $account_number $amount";
  partition amount: {
     * zero value 0;
     * small value select (1..100);
     * high value select (101..10000);  }
  category account_number:
    * individual value "01-123456";
    * joint value "02-222567";
    * corporate value "11-987654"; }
   partition operation: {
    * create_account
    * deposit {pre (operation ~ create_account);}
    * withdraw {pre (operation ~ create_account);}}
  }
...
```

6

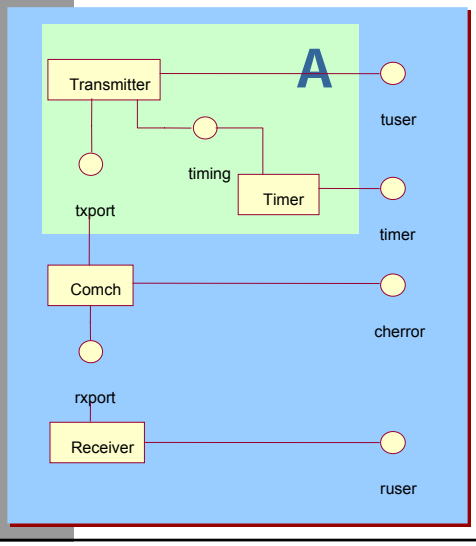

SIEMENS
Test Generation and Execution
SIEMENS CORPORATE RESEARCH
TDE
TDE/
UML
1
2
Target:
Software Components Unit /
Integration testing
Diagram used:
Statechart(s)/Interaction Diagrams
Target:
System level functional testing
Diagram used:
Use cases/ Activity Diagrams
Execution
Tools
Test
Scripts
UML Case Tool (e.g., Rational Rose)


SIEMENS
Approach
SIEMENS CORPORATE RESEARCH
Step 1: Modeling Component Behavior
UML Statecharts describing behavior of each component
Representing the normal and erroneous behavior of the component
UML extended by a new transition labeling scheme
Assumptions:
Point-to-point communication semantics between components rather than a shared event model
Communications are synchronous (blocking)
Implementation is deterministic
8

SIEMENS

SIEMENS CORPORATE RESEARCH

TDE/UML
An Example: A Communication Protocol



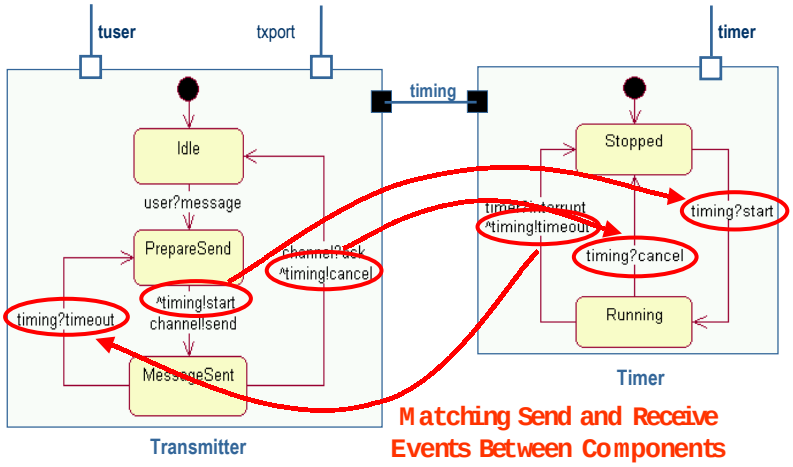
- Focus on integration testing
- Generate test cases to validate component interaction
- Consider subsystem A
- External interfaces:
  - tuser
  - timer
  - txport
- Internal Interfaces:
  - timing

9

SIEMENS

SIEMENS CORPORATE RESEARCH

Describing Component Behavior and Interaction



Matching Send and Receive Events Between Components

10

SIEMENS

SIEMENS CORPORATE RESEARCH

TDE/UML- Approach

Step 2: Generating Test Cases

(1) Normalizing the UML-based Models

- Resolving transitions with multiple events

(2) Computation of a global behavioral model based on matching send and receive events

- Composing the Global Behavioral Model (*complexity is better than exponential*)
- Determining a composition order (based on subsystem definition)

(3) Generating test cases from the global behavioral model

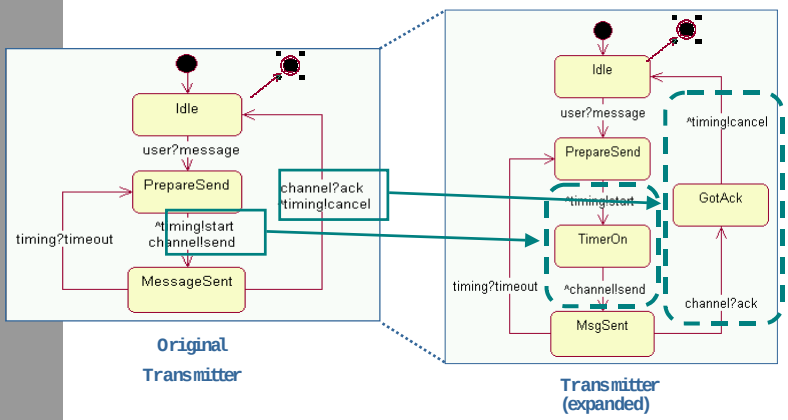
- Default coverage criterion: all transitions (within/between components)
- Using the data variations specified during the initial modeling phase

11

SIEMENS

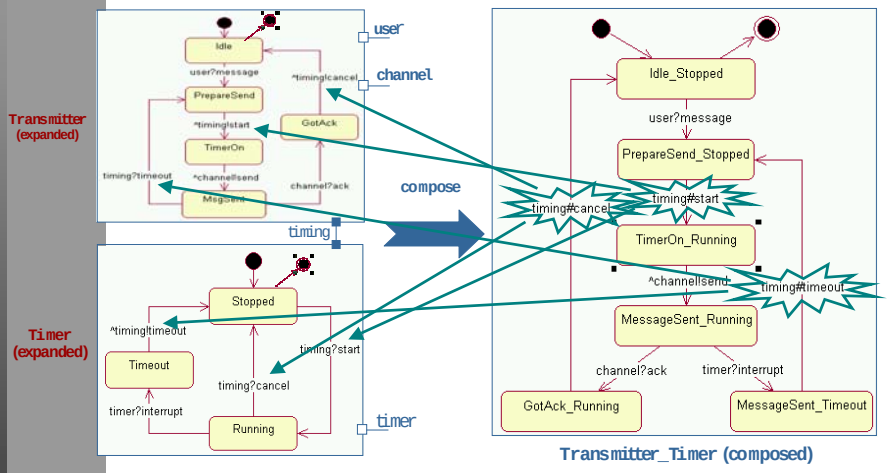
SIEMENS CORPORATE RESEARCH

Normalizing the UML-based Models...

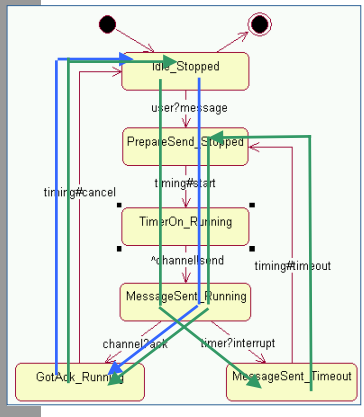


12

Composing a Global Behavioral Model...



Test Case Generation

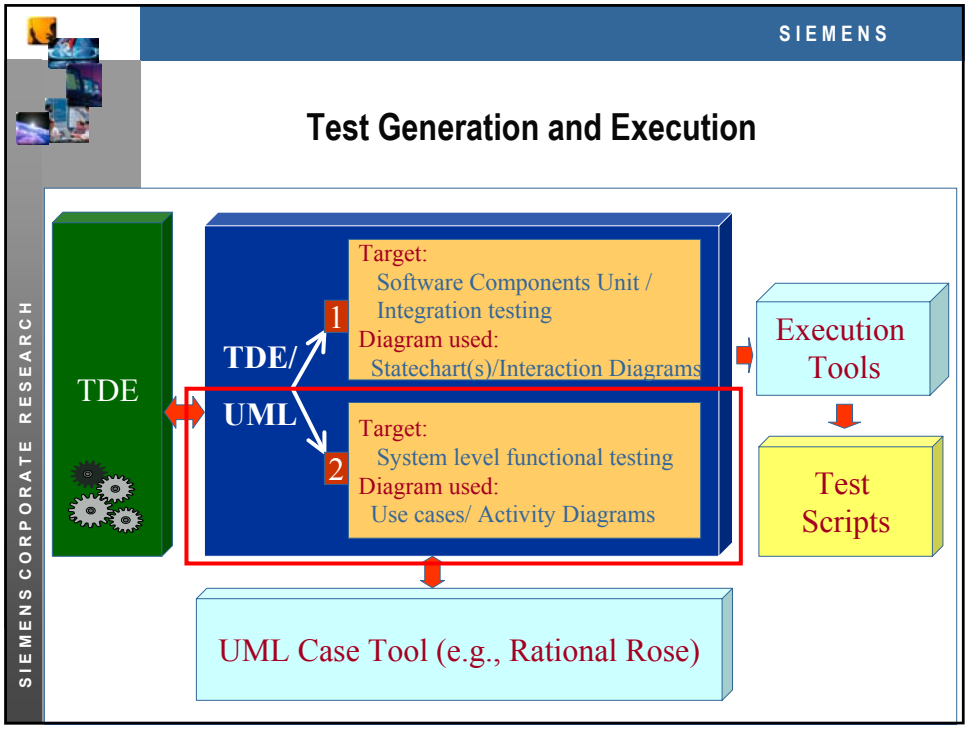


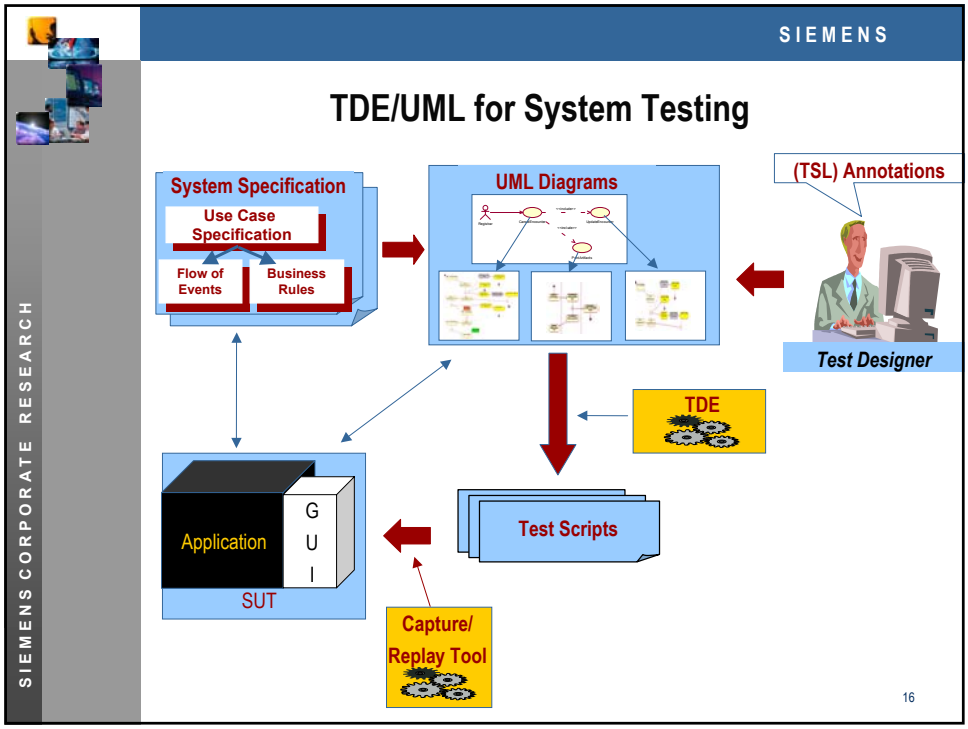
Transmitter_Timer (composed)

```
TEST CASE #1
 * IN  user.message();
 * OUT channel.send
 * IN  channel.ack();
 * IN  user.message();
 * OUT channel.send
 * IN  channel.ack();
TEST CASE #2
 * IN  user.message();
 * OUT channel.send
 * IN  timer.interrupt();
 * OUT channel.send
 * IN  channel.ack();
 * IN  user.message();
 * OUT channel.send
 * IN  channel.ack();
```

Test Cases for Subcomponent

For integration testing: all transitions between components are covered


SIEMENS
Test Generation and Execution
SIEMENS CORPORATE RESEARCH
TDE
TDE/
UML
1
2
Target:
Software Components Unit /
Integration testing
Diagram used:
Statechart(s)/Interaction Diagrams
Target:
System level functional testing
Diagram used:
Use cases/ Activity Diagrams
Execution
Tools
Test
Scripts
UML Case Tool (e.g., Rational Rose)


SIEMENS
TDE/UML for System Testing
SIEMENS CORPORATE RESEARCH
System Specification
Use Case
Specification
Flow of
Events
Business
Rules
UML Diagrams
(TSL) Annotations
Test Designer
TDE
Application
G
U
I
SUT
Test Scripts
Capture/
Replay Tool
16

SIEMENS

SIEMENS CORPORATE RESEARCH

TDE/UML for System Testing

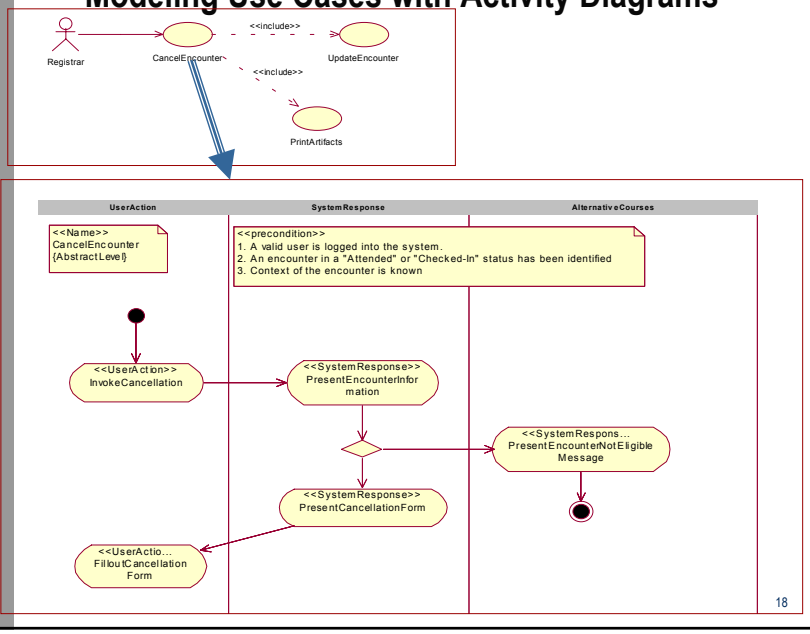
- Why “Use Cases” and “Activity” Diagram ?
  - They offer systematic and intuitive means of capturing functional requirements with a focus on value added to the user
  - They describe system behavior from a user point of view
  - Unfortunately not formal enough for test automation
- Describing Use Case with Activity Diagram
  - Annotations
    - UML stereotypes and notes
  - Activities organized in swim lanes
    - “Happy path” (ActorAction, SystemResponse) and alternative courses
  - Refinements
    - Activity and variables - Strategy: be as abstract or concrete as you like

17

SIEMENS

SIEMENS CORPORATE RESEARCH

Modeling Use Cases with Activity Diagrams



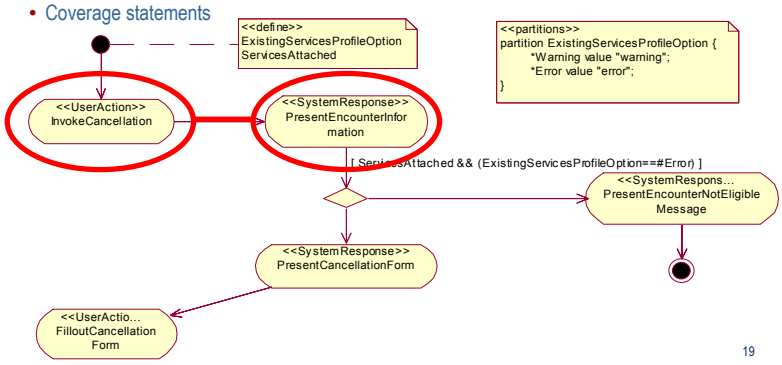
18

SIEMENS

SIEMENS CORPORATE RESEARCH

Annotations to an Activity Diagram

- Structural
  - Activity : <ActorAction>, <SystemResponse> or <Include>
  - Variables: <define>
  - Branches: TSL guard condition (using variables)
- Test Related
  - Partitions
  - Coverage statements



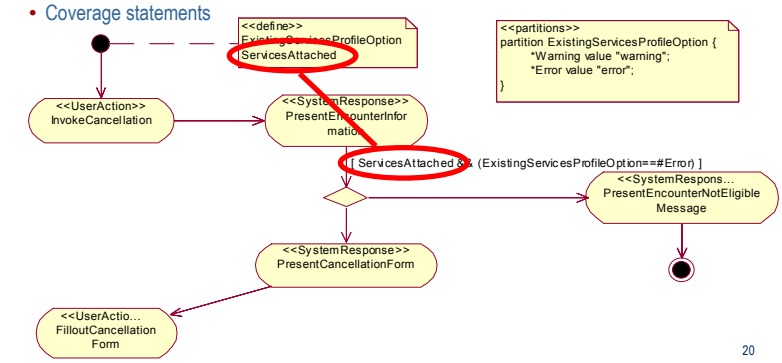
19

SIEMENS

SIEMENS CORPORATE RESEARCH

Annotations to an Activity Diagram

- Structural
  - Activity : <ActorAction>, <SystemResponse> or <Include>
  - Variables: <define>
  - Branches: TSL guard condition (using variables)
- Test Related
  - Partitions
  - Coverage statements



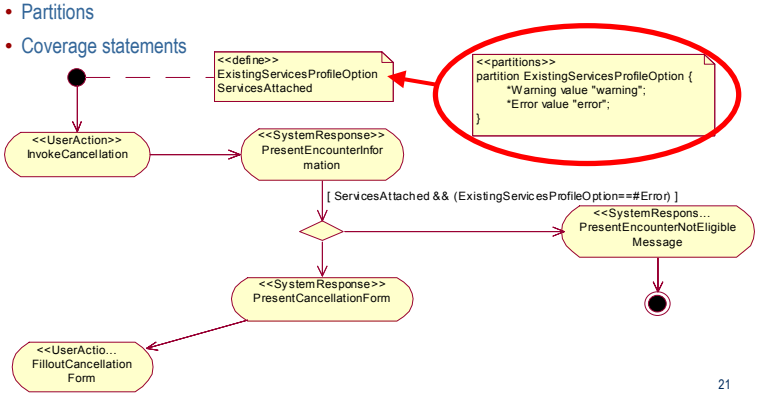
20

SIEMENS

SIEMENS CORPORATE RESEARCH

Annotations to an Activity Diagram

- Structural
  - Activity : <ActorAction>, <SystemResponse> or <Include>
  - Variables: <define>
  - Branches: TSL guard condition (using variables)
- Test Related
  - Partitions
  - Coverage statements



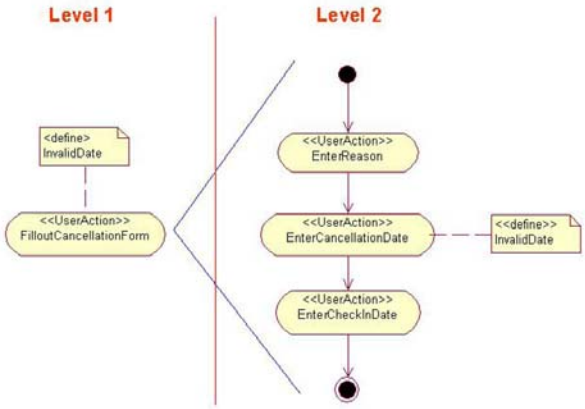
21

SIEMENS

SIEMENS CORPORATE RESEARCH

Annotations to an Activity Diagram

- Activities may be refined using sudiagrams
- Transparent abstraction levels for test generation
- Be as abstract as you want



22

SIEMENS

SIEMENS CORPORATE RESEARCH

Test Case Generation from within Rational Rose

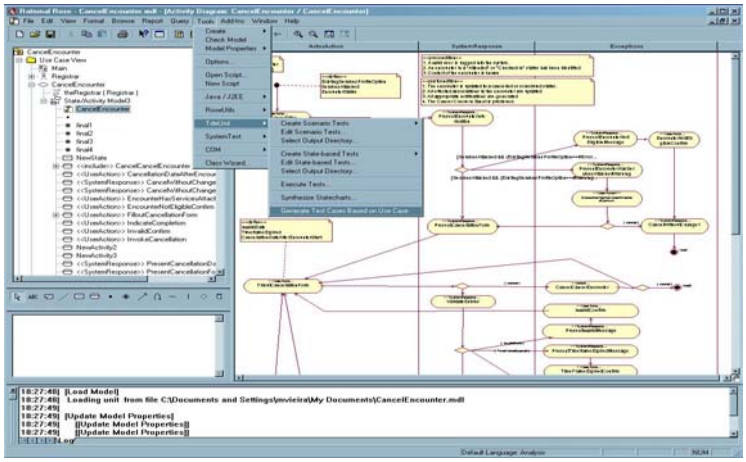


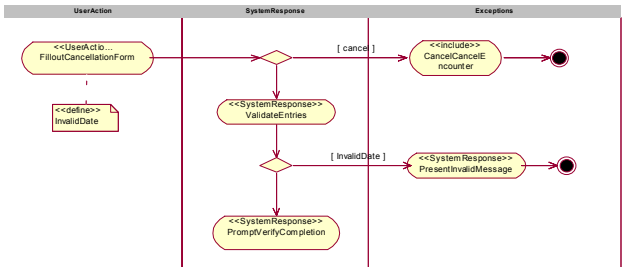
Figure 5: Generating System Tests from within Rational Rose

SIEMENS

SIEMENS CORPORATE RESEARCH

Test Generation

- Related to a specific Use Case
- Steps
  - Refinements are expanded
  - Happy paths from included Use Cases are analyzed
  - TSL test design is generated from the activity diagram
    - Mapping activities and transitions to TSL partitions and choices.
  - TDE creates test cases in order to satisfy the coverage requirements
    - Default - all data variations and branches are covered



24

SIEMENS

SIEMENS CORPORATE RESEARCH

Test Script Output

TSL Test Design

TSL Test Generator

```
<Test name="CancelEncounter_SystemTest_1">
    <UserAction name="FilloutCancellationForm">
        <Choice name="InvalidDate" value="T" />
        <Choice name="FilloutCancellationForm_cancel" value="F"/>
    </UserAction>
    <SystemResponse name="ValidateEntries" />
    <SystemResponse name="PresentInvalidMessage" />
</Test>
```

25

SIEMENS

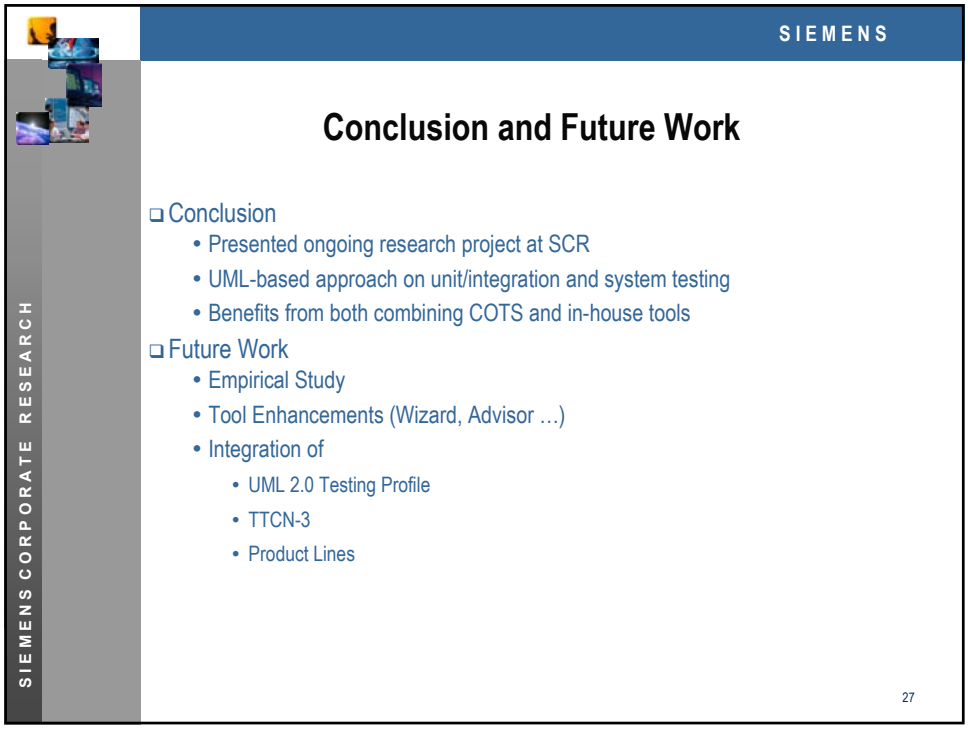
SIEMENS CORPORATE RESEARCH

Benefits

- Modeling system behavior.
  - Results in better, consistent and complete system models
  - Enables test designer to identify and compactly document a greater variety of test scenarios

    → Effectiveness of test design phase can be increased
- Generating test procedures.
  - Supports the automatic and systematic derivation of test cases
  - Notion of test adequacy or coverage with respect to the system functionality is addressed
- Executing test scripts.
  - Support for automated test execution
  - Promotes script reuse and simplifies script maintenance through ‘test snipplets’

26


SIEMENS
Conclusion and Future Work
Conclusion
Presented ongoing research project at SCR
UML-based approach on unit/integration and system testing
Benefits from both combining COTS and in-house tools
Future Work
Empirical Study
Tool Enhancements (Wizard, Advisor …)
Integration of
UML 2.0 Testing Profile
TTCN-3
Product Lines
SIEMENS CORPORATE RESEARCH
27