

# Test Design Strategies

Louise Tamres, CSQE

ASQ Software Division Webinar  
18 July 2008

# Objectives

- Translate requirements into test cases
- Improve communication by producing models
- Identify incomplete requirements
- Provide schedule estimates

# Content

- Test Design Techniques
  - Use Case
  - Classification Tree
  - Decision Table
  - State Transition Diagram
- Wrap Up
  - Schedule estimation

# Techniques Presented Here

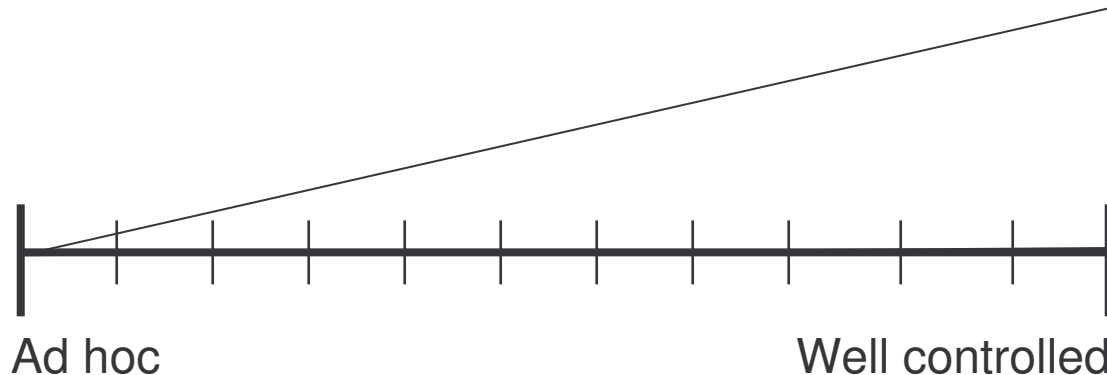
- Selected just a few techniques, based on:
  - Easy to focus on content
  - Useful in most cases
  - Learning something new for most people

# Basic Terminology

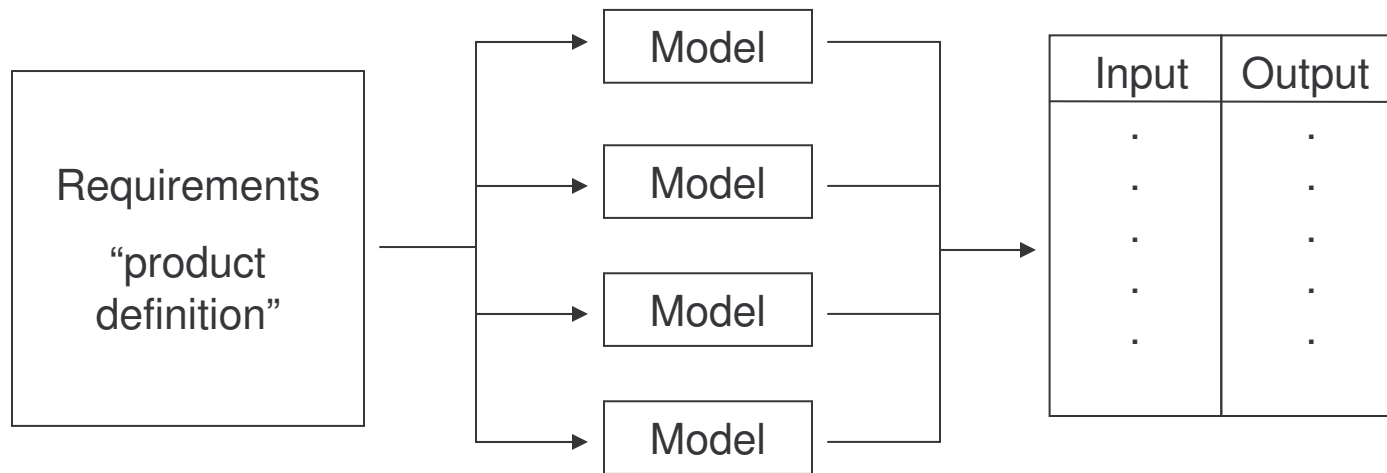
- “Test”
  - Generic term used in this class
  - *One scenario* that will be exercised
    - Some initial state
    - Some input stimulus
    - Something that’s expected to happen
- I won’t quibble between *test case* and *test procedure*

# Process Maturity

- Rigor depends upon your company's maturity level
  - Requirements range between inadequate to suitable
  - Thoroughness of test details will vary accordingly
  - Deficient requirements  $\Rightarrow$  more time uncovering details for tests



# Test Design Progression



... is a form of modeling requirements

# Advantages of Models

- Get “big picture” of what to test
- Development team focuses on contents
- Prioritize tests before writing any detailed test cases
- Provide data for schedules

# Value of Modeling

- Any method that helps the tester understand the application and ask smart questions is valuable
- The actual act of building a model is often more important because of the learning acquired during the process

# Test Design Techniques

- Use Case
- Decision Table
- Classification Tree Method
- State Machine

# Use Case

- Accomplish a goal within the system
  - End-to-end transaction
  - Often from the end user perspective
  - Test individual transactions

|  |   |
|--|---|
| <b>Use Case Example #1</b>   | User wants to order a book              |
| <b>Success Criteria:</b>   |   |
| User receives confirmation on book order.<br>Book order information sent to shipping department's queue. |   |
| <b>Main Success Scenario:</b>  |   |
| 1. Login to application  | 4. Go to checkout                       |
| 2. Search for book   | 5. Enter shipping address               |
| 3. Add book to shopping cart   | 6. Enter credit card information        |
|  | 7. Submit purchase request              |
| <b>Extensions:</b>   |   |
| 1a. User ID does not exist   | [Note: No extension for step #4]        |
| 1b. Invalid password   | 5a. Address left blank                  |
| 1c. Password is case sensitive (switch upper/lowercase)  | 5b. Address invalid format              |
| 1d. User cancels out   | 6a. User wants to pay by check          |
| 2a. Title (as entered) does not exist  | 6b. User wants to pay by credit card    |
| 2b. Author name misspelled   | 6c. User wants to pay by gift card      |
| 2c. Author name left blank   | 6d. Credit card information rejected    |
| 2d. User exits search  | 6e. User cancels out of application     |
| 2e. User cancels out of application  | 7a. User cancels purchase order         |
| 3a. User wants to select a different book  | 7b. User cancels out of application     |
| 3b. User cancels out of application  |   |
| 3c. User proceeds to checkout (without adding book to cart)  | [Note: Partial list to convey strategy] |

# Tests From Use Case #1

- Test A (from main scenario)
  - Path through Use Case: Steps 1, 2, 3, 4, 5, 6, 7
  - Input: data that satisfy
    - All steps listed in Main Success Scenario
  - Outcome:
    - All items listed in Success Criteria
- Test B (from Extension 3a)
  - Path through Use Case: Steps 1, 2, 3, 3a
  - Input: data that satisfy
    - Complete steps 1, 2, 3 as listed in Main Success Scenario
    - Select a new book (as specified in Extension 3a)
  - Outcome:
    - Understand resulting action (should be in requirements)

**Test Log for Use Case Example #1**

| Test ID       | Priority | Input Values | Results | Pass / Fail | Bugs/ Observations |
|---------------|----------|--------------|---------|-------------|--------------------|
| main scenario |          |              |         |             |                    |
| 1a            |          |              |         |             |                    |
| 1b            |          |              |         |             |                    |
| 1c            |          |              |         |             |                    |
| 1d            |          |              |         |             |                    |
| 2a            |          |              |         |             |                    |
| 2b            |          |              |         |             |                    |
| 2c            |          |              |         |             |                    |
| 2d            |          |              |         |             |                    |
| 2e            |          |              |         |             |                    |
| 3a            |          |              |         |             |                    |
| 3b            |          |              |         |             |                    |
| 3c            |          |              |         |             |                    |
| 5a            |          |              |         |             |                    |
| 5b            |          |              |         |             |                    |
| 6a            |          |              |         |             |                    |
| 6b            |          |              |         |             |                    |
| 6c            |          |              |         |             |                    |
| 6d            |          |              |         |             |                    |
| 6e            |          |              |         |             |                    |
| 7a            |          |              |         |             |                    |
| 7b            |          |              |         |             |                    |

**Use Case Example #2**

Send data from local to remote machine

**Success Criteria:**

- Remote machine receives all converted data.
- Remote machine sends an acknowledgement to local machine.
- On local machine, data marked as successfully sent.
- Local machine receives acknowledgement from remote machine.

**Main Success Scenario:**

1. Programmatically Login to local database
2. Pull unsent data from the database
3. Convert that data to a pre-determined XML format and validate against a given schema definition
4. Connect to remote message queue
5. Send converted XML data to the remote queue
6. Receive acknowledgement from remote machine
7. Mark converted data as “successfully sent to remote queue”
8. Look for more unsent data to send to remote queue (Back to step 2)

**Extensions:**

- 1a. Unable to connect to database for whatever reason
- 1b. Invalid username/password
- 1c. Unable to connect to database port
- 2a. Expected data not found
- 2b. Inconsistency in database
- 2c. Connection fails during database pull
- 3a. Schema validation fails
- 4a. Connection to remote queue fails
- 5a. Connection fails during sending of data.
- 6a. No acknowledgement received by local machine
- 7a. Failure to mark data as sent in local database

# Test Design Techniques

- Use Case
- Decision Table
- Classification Tree Method
- State Machine

# Decision Table

- Record complex business rules
- Useful when specification has
  - Complicated decision logic
  - Many If-Then-Else statements
- Merge (simplify) rules whose resulting actions are identical

# Decision Table Format

|                    | <b>Rule 1</b> | <b>Rule 2</b> | <b>...</b> | <b>Rule n</b> |
|--------------------|---------------|---------------|------------|---------------|
| <b>Condition 1</b> |               |               |            |               |
| <b>Condition 2</b> |               |               |            |               |
| <b>Condition 3</b> |               |               |            |               |
| <b>...</b>         |               |               |            |               |
| <b>Action 1</b>    |               |               |            |               |
| <b>Action 2</b>    |               |               |            |               |
| <b>Action 3</b>    |               |               |            |               |
| <b>...</b>         |               |               |            |               |

# Decision Table Components

## 1. Set of conditions

- List of input types

## 2. Rules

- All combinations of input values/states

## 3. Actions

- Based on applying rules
- Depends on values of conditions; not order in which conditions are evaluated

# Testing Decision Tables

- Create one test case for each rule
  - One test for each column in table
- Conditions are the input
- Actions are the expected results

# Decision Table Example

Example #1 - All combinations of input data values

|            |                                    | Rule 1 | Rule 2 | Rule 3   | Rule 4   | Rule 5 | Rule 6 | Rule 7   | Rule 8   |
|------------|------------------------------------|--------|--------|----------|----------|--------|--------|----------|----------|
| Conditions | Deductible met?                    | yes    | yes    | yes      | yes      | no     | no     | no       | no       |
|            | Type of visit (doctor or hospital) | doctor | doctor | hospital | hospital | doctor | doctor | hospital | hospital |
|            | In- or out-of-network provider?    | in     | out    | in       | out      | in     | out    | in       | out      |
| Actions    | Reimburse at 60%                   |        | X      |          |          |        |        |          |          |
|            | Reimburse at 70%                   |        |        |          | X        |        |        |          |          |
|            | Reimburse at 80%                   | X      |        | X        |          |        |        |          |          |
|            | No reimbursement                   |        |        |          |          | X      | X      | X        | X        |

# Decision Table Example

Example #2 – Identify rules that have same resulting actions

|            |                                    | Rule 1, Rule 3 |          | Rule 2 | Rule 4   | Rule 5, Rule 6, Rule 7, Rule 8 |        |          |          |
|------------|------------------------------------|----------------|----------|--------|----------|--------------------------------|--------|----------|----------|
| Conditions | Deductible met?                    | yes            | yes      | yes    | yes      | no                             | no     | no       | no       |
|            | Type of visit (doctor or hospital) | doctor         | hospital | doctor | hospital | doctor                         | doctor | hospital | hospital |
|            | In- or out-of-network provider?    | in             | in       | out    | out      | in                             | out    | in       | out      |
| Actions    | Reimburse at 60%                   |                |          | X      |          |                                |        |          |          |
|            | Reimburse at 70%                   |                |          |        | X        |                                |        |          |          |
|            | Reimburse at 80%                   | X              | X        |        |          |                                |        |          |          |
|            | No reimbursement                   |                |          |        |          | X                              | X      | X        | X        |

# Decision Table Example

Example #3 – Merge rules that have same actions

|            |                                    | Rule 1, 3 | Rule 2 | Rule 4   | Rule 5,6,7,8 |
|------------|------------------------------------|-----------|--------|----------|--------------|
| Conditions | Deductible met?                    | yes       | yes    | yes      | no           |
|            | Type of visit (doctor or hospital) | --        | doctor | hospital | --           |
|            | In- or out-of-network provider?    | in        | out    | out      | --           |
| Actions    | Reimburse at 60%                   |           | X      |          |              |
|            | Reimburse at 70%                   |           |        | X        |              |
|            | Reimburse at 80%                   | X         |        |          |              |
|            | No reimbursement                   |           |        |          | X            |

-- Signifies "don't care"

# Tests From Decision Table

- Test A (from Rule 1,3)
  - Input: data that satisfy
    - Deductible met
    - Provider is in-network
  - Outcome:
    - Calculate reimbursement at 80%
- Test B (from Rule 2)
  - Input: data that satisfy
    - Deductible met
    - Service provided in doctor's office
    - Provider is out-of-network
  - Outcome:
    - Calculate reimbursement at 60%

# Test Design Techniques

- Use Case
- Decision Table
- Classification Tree Method
- State Machine

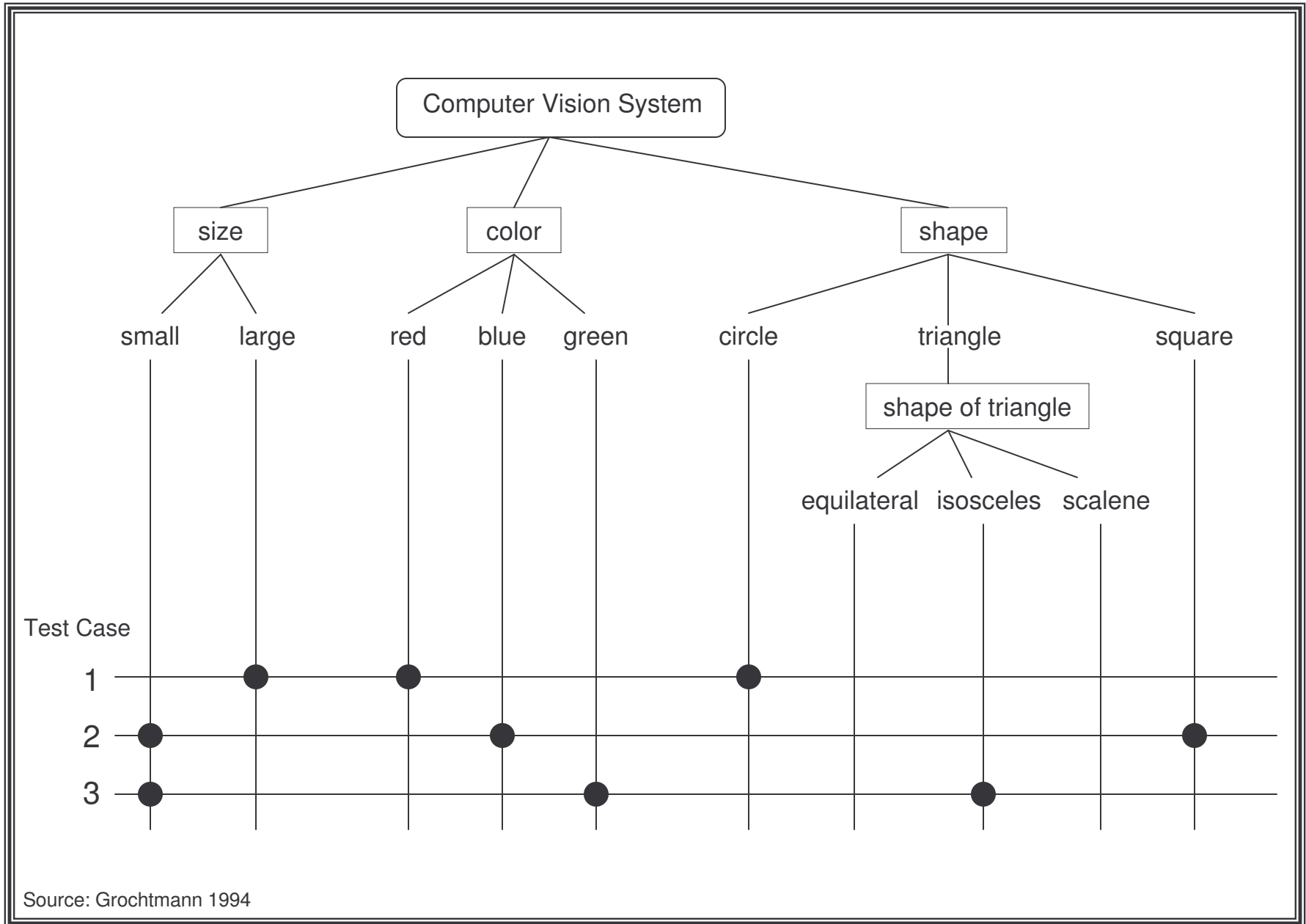
# Classification Tree Method

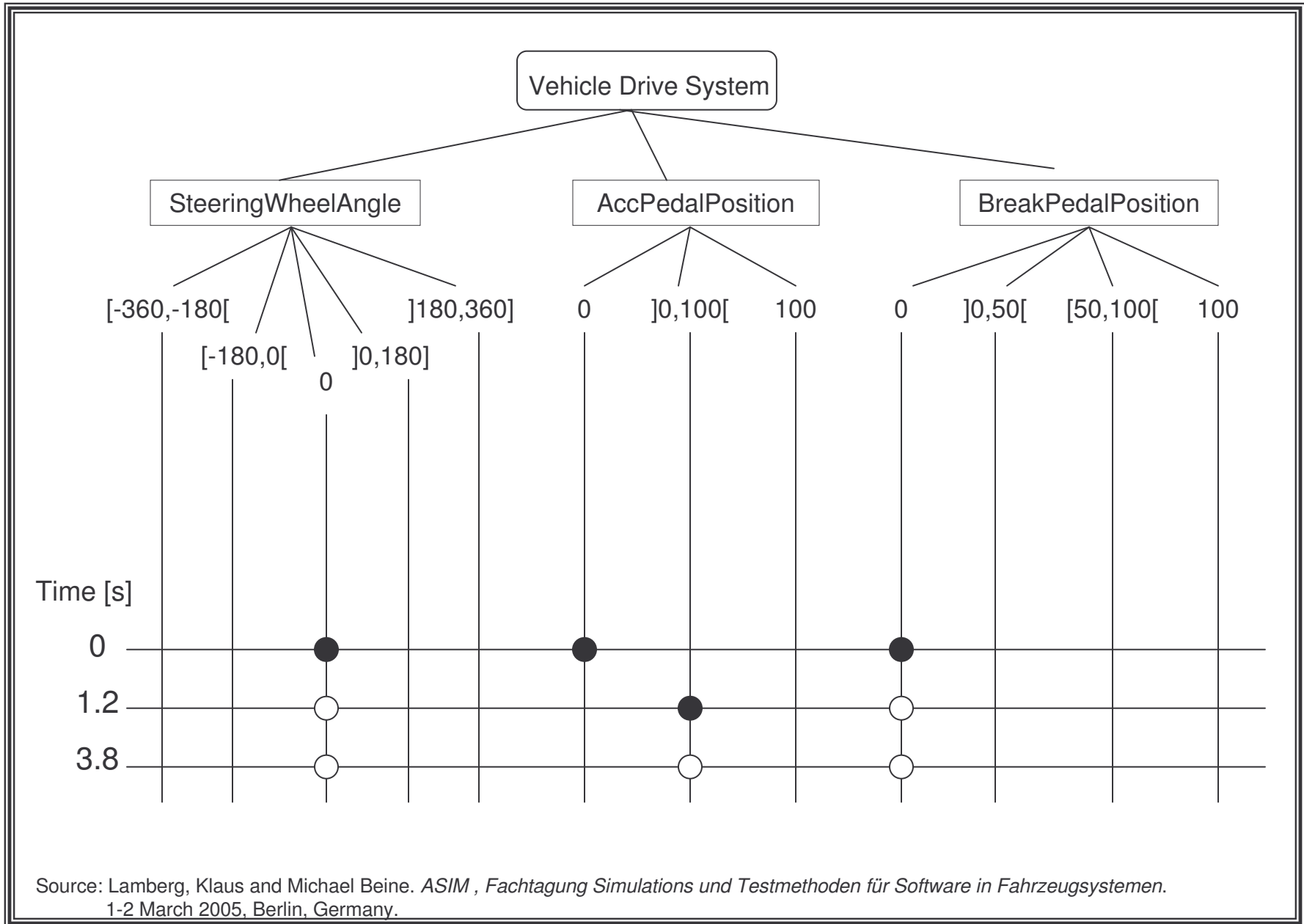
- Developed by Daimler-Benz Research
- Used in Embedded Systems
- Test case design tool using black-box approach (based on functional spec)
- Based on Category Partitioning
- Compact graphical representation of test scenarios

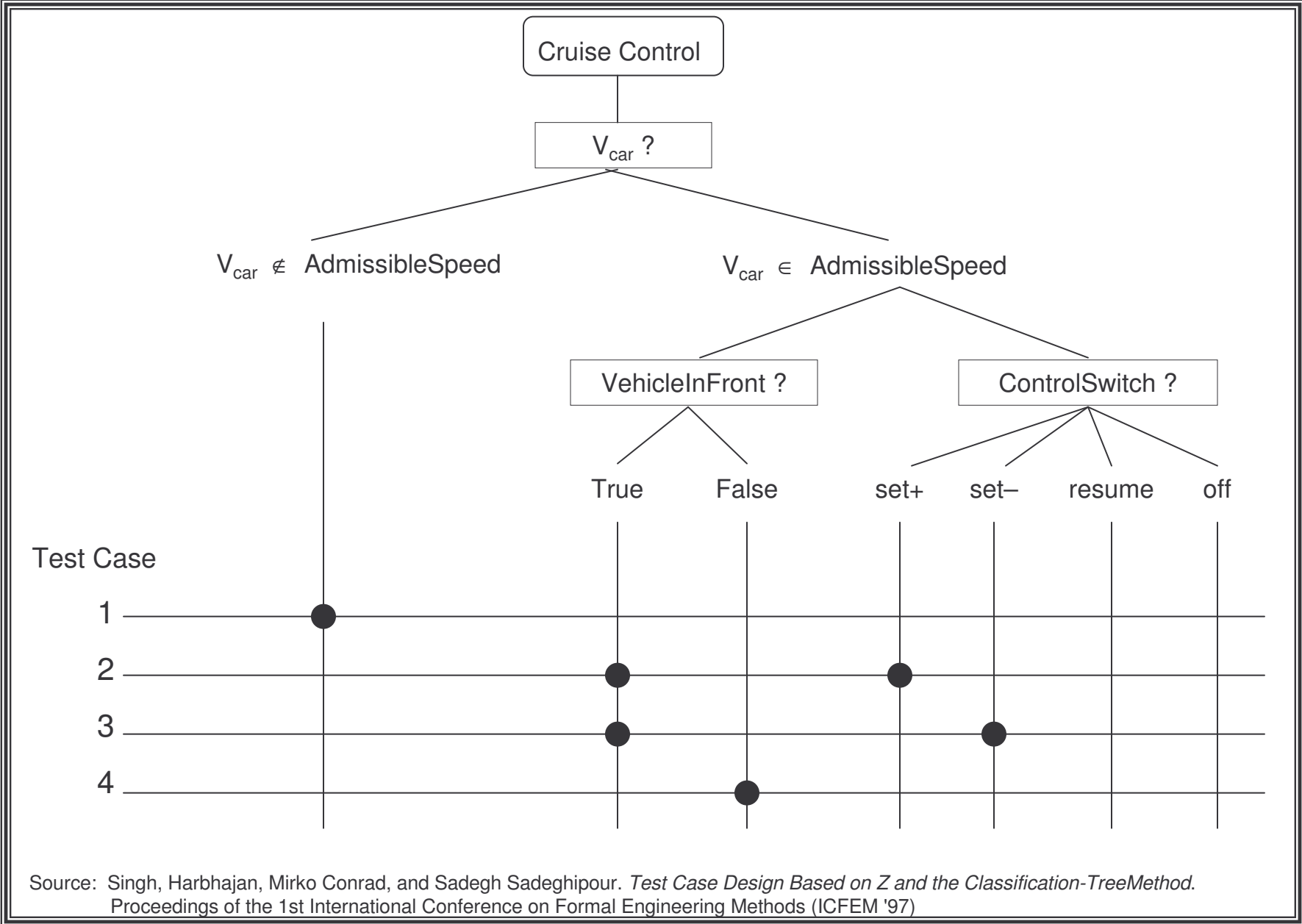
# Classification Tree Example

- Computer Vision System
  - Inputs are various building blocks
  - Aspects are size, color, and shape of a block
    - Size: small, large
    - Color: red, green, blue
    - Shape: circle, triangle, square

Source for example: Grochtmann, Matthias. *Test Case Design Using Classification Trees*.  
Proceedings of STAR '94, 8-12 May 1994, Washington, DC.







# Classification Tree Editor

- Tool that supports
  - Design of classification tree
  - Definition of test cases in the table area
  - Hierarchies and structure of large trees
  - Creation of automated test cases
  - Documentation of test cases
  - Free download available at  
[www.systematic-testing.com](http://www.systematic-testing.com)

# Classification Trees as Spreadsheets

| Test id | Size  |       | Color |      |       | Shape  |        |       |       |      |
|---------|-------|-------|-------|------|-------|--------|--------|-------|-------|------|
|         | small | large | red   | blue | green | circle | square | equil | isosc | scal |
| test 1  |       | ✓     | ✓     |      |       | ✓      |        |       |       |      |
| test 2  | ✓     |       |       | ✓    |       |        | ✓      |       |       |      |
| test 3  | ✓     |       |       |      | ✓     |        |        |       | ✓     |      |

| Test id | Size  | Color | Shape     |
|---------|-------|-------|-----------|
| test 1  | large | red   | circle    |
| test 2  | small | blue  | square    |
| test 2  | small | green | isosceles |

# Test Design Techniques

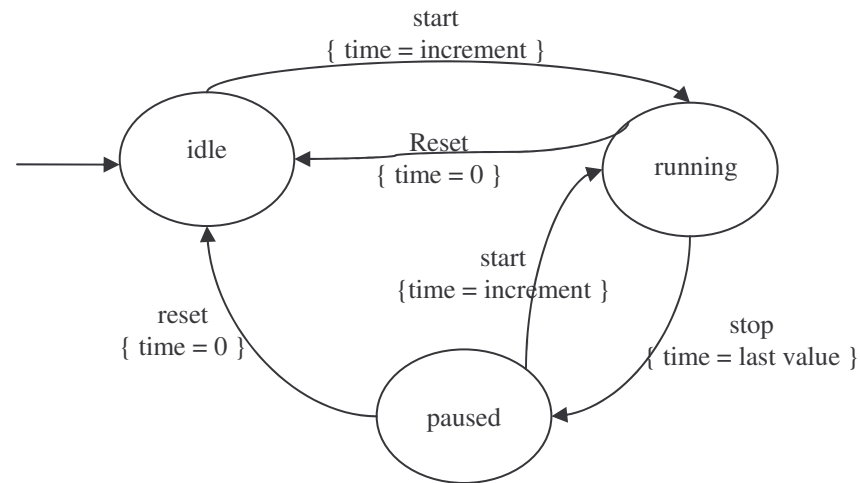
- Use Case
- Decision Table
- Classification Tree Method
- State Machine

# State Machine

- Useful when specs provided as state machine (state transition diagram)
- Sequence is important
  - Input must be entered in a defined order
- Test transitions between states by creating events that lead to these transitions

# State Transition Diagram Example

Stopwatch example



# State Transition Table #1 initial

| Event →<br>↓ State | START                                    | STOP                            | RESET                |
|--------------------|--|---------------------------------|----------------------|
| Idle               | Running<br>{ Time = increment<br>value } | ?                               | ?                    |
| Running            | ?  | Paused<br>{ Time = last value } | Idle<br>{ Time = 0 } |
| Paused             | Running<br>{ Time = increment<br>value } | ?                               | Idle<br>{ Time = 0 } |

# State Transition Table #1 complete

| Event →<br>↓ State | START                                    | STOP                               | RESET                |
|--------------------|--|------------------------------------|----------------------|
| Idle               | Running<br>{Time = increment<br>value}   | Idle<br>{ Time = 0 }               | Idle<br>{ Time = 0 } |
| Running            | Running<br>{Time = increment<br>value}   | Paused<br>{ Time = last value<br>} | Idle<br>{ Time = 0 } |
| Paused             | Running<br>{ Time = increment<br>value } | Paused<br>{ Time = last value<br>} | Idle<br>{ Time = 0 } |

# State Transition Table #2

State transition diagram information and tests all in one chart

| <b>Current State</b> | <b>Event</b> | <b>Action</b>          | <b>Next State</b> |
|----------------------|--------------|------------------------|-------------------|
| Idle                 | START        | Time = increment value | Running           |
| Idle                 | STOP         | Time = 0               | Idle              |
| Idle                 | RESET        | Time = 0               | Idle              |
| Running              | START        | Time = increment value | Running           |
| Running              | STOP         | Time = last value      | Paused            |
| Running              | RESET        | Time = 0               | Idle              |
| Paused               | START        | Time = increment value | Running           |
| Paused               | STOP         | Time = last value      | Paused            |
| Paused               | RESET        | Time = 0               | Idle              |

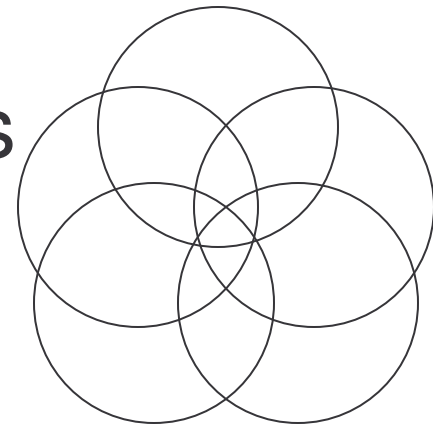
test input

test results

# Wrapping Up

# Test Design Techniques

- Which approach to use?
  - Communicate the information
  - Experience, comfort level
  - Follow requirements format
  - Available tools
- Each design method provides
  - Unique tests
  - Duplicate tests



# Schedule Estimation

- Based on number of tests identified
  - Use Case: main + number of extensions
  - Classification Tree: number of rows
  - Decision Table: number of columns/rules
  - State Machine:
    - number of boxes in state transition table (ex. #1)
    - number of rows in expanded table (example #2)

# Schedule Estimation

Estimated effort for testing =

Number of tests x (time per test)

+ time to set up test environment

+ time to set up and program tools

+ time to create test data

+ time for bug fix and retest cycle

+ . . .

# If We Had More Time...

- Prioritizing tests
- Documenting tests
- Tracking test status
- Mapping requirements to tests
- And yet more test design techniques

# More Test Design Techniques

- Equivalence Class Partitioning
- Boundary Value Analysis
- Pairwise Testing

*Data Driven Techniques*

- Test Outlines
- Classification Tree Method \*
- Decision Tables \*
- State Machine \*
- Tables
- Timing Charts
- Use Cases \*

*Requirements Modeling*

- Control Flow Testing

*Graphical Method*

\* Topic presented in today's session

# References

More examples and strategies

Tamres, Louise. *Introducing Software Testing*.  
Addison-Wesley, 2002.

# Thank You

If you have any questions, contact me at

[l . tamres @ computer . org](mailto:l.tamres@computer.org)