Specification Based Testing – Part 1

Scenario Based Testing

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Contrast requirements based vs scenario based testing

Functional Testing

- **Testing the system as a black box**
- **Testing the system the way a customer will be using the system**
- **Develop test cases from:**
  - User level documentation
  - Requirement specification documentation

Strategies for Developing Tests

Static (requirement driven)

- Traditional approach
- Primarily a verification activity

Behavioral (scenario testing)

- Supports both verification and validation activities

Requirements Driven Testing

- Carefully examine the requirements and develop test cases to ensure that each requirement is verified by one or more tests
- Normally a single test will verify several requirements
- Demonstrate traceability of requirements to test cases

Requirements Traceability Tools

- Numerous tools exist enabling software developers and engineers to “share” evolving requirements
- These tools facilitate the linking of tests to requirements supporting the necessary traceability

Requirements Traceability Example

Consider the following partial set of requirements for an ATM banking system:

1.0 System shall identify customers via a bank card and PIN

1.1 System shall prompt for card insertion

1.2 System shall validate card information

1.2.1 Stolen card shall be confiscated by system.

1.2.2 Unrecognized cards shall be returned with an error message

Requirements Traceability Example (cont'd)

Consider the following partial set of requirements for an ATM banking system:

1.3 System shall prompt for PIN entry

1.4 System shall validate PIN entry

1.4.1 System shall prompt for re-entry of invalid PINS up to 3 times.

1.4.2 System shall confiscate cards for more than 3 invalid entries.

Requirements Traceability Example (cont'd)

Consider the following partial set of requirements for an ATM banking system:

2.0 System shall present transaction options to the customer

3.0 System shall process transaction requests

3.1 System shall support a withdraw transaction

3.1.1 System shall prompt for amount and source of funds

Requirements Traceability Example (cont'd)

Consider the following partial set of requirements for an ATM banking system:

3.1.2 System shall validate sufficient funds exist

3.1.2.1 If insufficient funds, the system shall prompt for re-entry

3.1.3 System shall update the account and dispense cash

4.0 Upon end of transaction requests, the system shall generate a receipt and return the customer card

4.1 If card is not taken, system shall confiscate card

Requirements Traceability Example

(Partial test cases showing only inputs) Test #1

Customer inserts valid card, enters correct PIN, requests a withdrawal amount that is OK and takes his money and card.

Requirements covered: 1.0, 1.1, 1.2, 1.3, 1.4, 2.0, 3.0, 3.1, 3.1.1, 3.1.2, 3.1.3, 4.0

Requirements Traceability Example

Test #2

Customer inserts stolen card.

Requirements covered: 1.0, 1.1, 1.2, 1.2.1

Scenario Testing

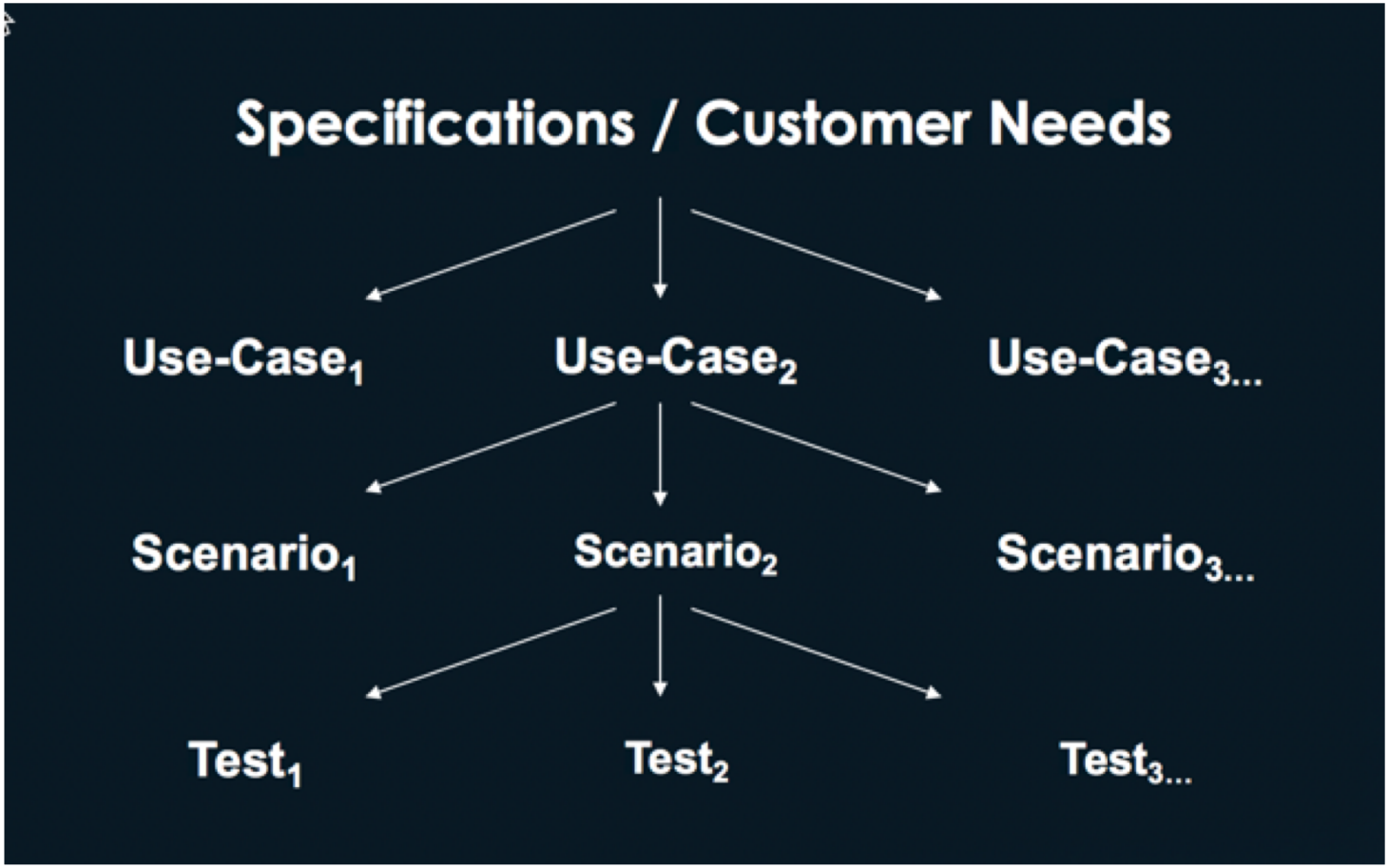
Use Cases

- All functional requirements are captured in use-cases (the requirements addressed by a use-case should be identified)
- Many non-functional requirements can be viewed as attributes of use-cases
- Use-cases provide a way of organizing and abstracting the requirements

Use-Case Testing Approach

- Develop a set of use-cases from the specifications if they do not already exist
- Verify and validate the use-cases
- Develop scenarios for the use-cases
- Test each scenario

Scenario Testing



Use-Case Verification and Validation

- Use-cases must be carefully examined to ensure that they are complete and correct
- Ensure that all of the needs of the customer are addressed
- Ensure all interactions among actors are correct
- Ensure all requirements are covered

Use-Case Construction

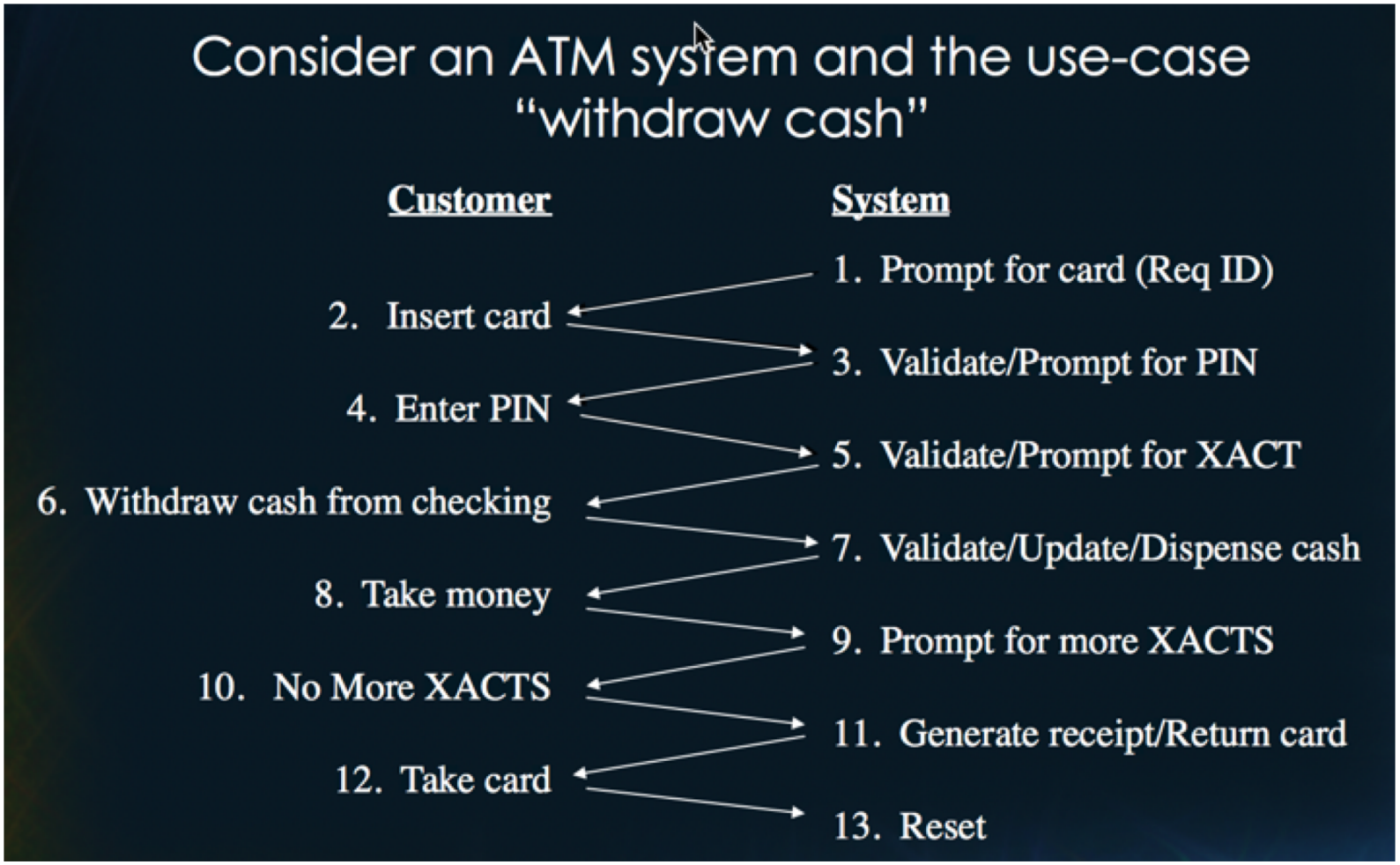
- Identify the actors (determine scope of system)
- For each actor, identify how the actor uses the system to accomplish its functions
- Detail each use-case identifying each of its processing flows as a scenario

Documenting Use-Cases

Use-cases can be documented:

1. Step-by-step description
2. State diagram

Example (Step-by-step Description)



Example Step-by-step Description (cont'd)

- Each path through the user-case represents a scenario which must be tested.

Example Step-by-step Description (cont'd)

Alternate Paths (subset of possibilities):

3a. If stolen card, then keep the card
3b. If account is not recognized, return card and generate message.
5a. If incorrect PIN, prompt for retry.
5b. If 3 invalid attempts, keep card.
7. If insufficient funds, generate message and allow re-entry.
13. If card is not taken, keep card.

Summary