Test Design Techniques Assignment

This assignment covers fundamental test design techniques, including the four-step test design technique, boundary value analysis, equivalence partitioning, and their application in real-world scenarios. Both team members contributed equally to all exercises, with one focusing on theoretical explanations and the other handling practical applications and examples.

Exercise 1: Test Design Technique

The Four-Step Test Design Technique

The test design technique is a structured approach that helps testers create comprehensive and effective test cases. Following are the four steps of this technique:

1. **Identify Test Conditions**: In this initial step, testers analyze the requirements, specifications, and user stories to determine what needs to be tested. Test conditions represent specific features, functions, or attributes of the system that should be verified. This step requires a thorough understanding of business requirements and technical specifications to ensure complete test coverage.
2. **Design Test Cases**: Based on the identified test conditions, testers design specific test cases that will verify each condition. A test case includes inputs, preconditions, and steps needed to test a particular aspect of the software. Various techniques like equivalence partitioning, boundary value analysis, or decision tables might be applied at this stage to derive effective test cases.
3. **Determine Expected Results**: For each test case, testers must specify the expected outcome or behavior. This step is crucial as it provides the benchmark against which actual results will be compared during test execution. Expected results should be derived directly from requirements and specifications to ensure validity.
4. **Create Test Procedures**: The final step involves creating detailed, step-by-step instructions for executing the test cases. Test procedures include preconditions, specific test data, concrete actions to perform, and verification points. These procedures ensure tests can be executed consistently, regardless of who performs them.

How the Test Design Technique Helps in Creating Structured Test Cases

The four-step test design technique brings several benefits to the testing process:

First, it ensures systematic coverage of requirements by forcing testers to methodically analyze specifications and derive test conditions before jumping into test execution. This structured approach prevents important aspects from being overlooked.

Second, it improves test efficiency by organizing the testing effort around clearly defined conditions and cases, which minimizes redundant testing while maximizing coverage. The technique allows testers to apply various test design methods strategically based on the context.

Third, it enhances test documentation and traceability by creating clear links between requirements, test conditions, test cases, and results. This traceability is valuable for demonstrating test coverage and for impact analysis when requirements change.

Finally, it promotes reusability of test assets by organizing tests in a modular fashion based on conditions, which makes it easier to update tests when requirements change and to reuse test components across different testing cycles.

Exercise 2: Boundary Value Analysis and Equivalence Partitioning

Explanation of Boundary Value Analysis and Equivalence Partitioning

Equivalence Partitioning (EP)

Equivalence Partitioning is a test design technique that divides input data into equivalent classes or partitions. The fundamental principle behind EP is that if one test case in an equivalence class detects a defect, other test cases in the same class would likely find the same defect. Similarly, if one test case doesn't detect a defect, others in the same class probably won't either.

This technique aims to reduce the number of test cases while maintaining comprehensive coverage. It works by identifying ranges of inputs that should be handled similarly by the system and testing representative values from each range.

For example, in a form that accepts ages between 18 and 60, EP would divide the input domain into three partitions: 1-17 (invalid), 18-60 (valid), and >60 (invalid). Testing one value from each partition would be sufficient to verify the handling of all values in that partition.

Boundary Value Analysis (BVA)

Boundary Value Analysis focuses on testing values at the edges or boundaries of equivalence partitions. The technique is based on the observation that errors often occur at the boundaries of input domains rather than in the center.

BVA involves testing values that are directly on, just above, and just below the edges of each equivalence partition. These boundary values are more likely to reveal defects because boundary conditions are often handled incorrectly in programming logic.

For example, if a system accepts values between 1 and 100, the boundary values would be 0, 1, 100, and 101, where 1 and 100 are on the boundaries, 0 is just below the lower boundary, and 101 is just above the upper boundary.

Comparison of Usability

Both techniques have distinct advantages and are often used together:

Equivalence Partitioning is particularly useful for:

- Reducing the number of test cases while maintaining good coverage
- Handling large input domains efficiently
- Providing a systematic approach to test case selection
- Identifying missing logic in the application

Boundary Value Analysis excels at:

- Finding defects related to boundary conditions
- Detecting off-by-one errors in programming
- Covering edge cases that might be overlooked
- Testing system behavior at transition points between different equivalence classes

While EP helps achieve broad coverage with fewer test cases, BVA focuses on high-risk areas where defects are more likely to occur. EP is more effective for initial testing to ensure basic functionality, while BVA adds depth by targeting specific areas prone to errors.

The main difference is that EP focuses on reducing test cases by selecting representative values, while BVA focuses on finding errors at the boundaries where they commonly occur. Together, they provide a comprehensive approach to testing.

Application to the Age Validation Scenario

For the age validation method that classifies values as impossible (< 0 or > 120), underage (< 18), or valid (≥ 18), here's how we apply BVA and EP:

Equivalence Partitions:

1. Invalid (Impossible) - Lower Range: All values < 0
2. Invalid (Underage): Values between 0 and 17
3. Valid: Values between 18 and 120
4. Invalid (Impossible) - Upper Range: All values > 120

Boundary Values:

1. Lower Impossible/Underage Boundary: -1, 0
2. Underage/Valid Boundary: 17, 18
3. Valid/Upper Impossible Boundary: 120, 121

To thoroughly test this validation method using both techniques, we would create test cases using:

- One representative value from each equivalence partition: e.g., -5, 10, 50, 130
- All boundary values: -1, 0, 17, 18, 120, 121

This approach ensures we test both the general behavior within each range and the critical transition points between ranges, maximizing our chances of detecting defects with minimal test cases.

Exercise 3: Designing Test Cases

Analysis of the Company Building Door Scenario

For the scenario where a door can be opened from outside either by holding a valid company card to a sensor for at least two seconds or by the porter unlocking it, and can always be opened from the inside, we'll apply the test design technique.

Identification of Conditions and Actions

Conditions:

1. User location (inside/outside)
2. Company card status (valid/invalid/no card)
3. Duration of card hold (< 2 seconds, ≥ 2 seconds)
4. Porter action (unlocks door/doesn't unlock)

Actions:

1. User attempts to open the door

Valid Combinations and Expected Outcomes

1. **Combination 1:**
    - Conditions: User outside, valid company card, hold ≥ 2 seconds, porter doesn't unlock
    - Expected outcome: Door opens
2. **Combination 2:**
    - Conditions: User outside, valid company card, hold < 2 seconds, porter doesn't unlock
    - Expected outcome: Door remains closed
3. **Combination 3:**
    - Conditions: User outside, invalid company card, hold ≥ 2 seconds, porter doesn't unlock
    - Expected outcome: Door remains closed
4. **Combination 4:**
    - Conditions: User outside, invalid company card, hold < 2 seconds, porter doesn't unlock
    - Expected outcome: Door remains closed
5. **Combination 5:**

- Conditions: User outside, no company card, porter doesn't unlock
- Expected outcome: Door remains closed

6. **Combination 6:**
   - Conditions: User outside, valid company card, hold ≥ 2 seconds, porter unlocks
   - Expected outcome: Door opens
7. **Combination 7:**
   - Conditions: User outside, valid company card, hold < 2 seconds, porter unlocks
   - Expected outcome: Door opens
8. **Combination 8:**
   - Conditions: User outside, invalid company card, hold ≥ 2 seconds, porter unlocks
   - Expected outcome: Door opens
9. **Combination 9:**
   - Conditions: User outside, invalid company card, hold < 2 seconds, porter unlocks
   - Expected outcome: Door opens
10. **Combination 10:**
    - Conditions: User outside, no company card, porter unlocks
    - Expected outcome: Door opens
11. **Combination 11:**
    - Conditions: User inside, with any combination of other conditions
    - Expected outcome: Door opens

The test design technique helps us systematically identify all these combinations, ensuring we have complete test coverage for this scenario. We've organized the conditions and their possible values, combined them in all relevant ways, and specified the expected outcomes based on the stated requirements.

Conclusion

The test design techniques presented in this assignment provide a structured approach to creating effective and comprehensive test cases. The four-step test design technique offers a methodical framework for developing tests, while boundary value analysis and equivalence partitioning provide specific strategies for input selection.

By applying these techniques to real-world scenarios like age validation and door access control, we can see how they enable testers to achieve good coverage while minimizing the number of test cases needed. These techniques form the foundation of systematic testing and will be valuable tools for all future testing activities in this course.