Test Case Design II

2013

Control structure

- Condition testing
- Loop testing

Condition testing

- Method to exercise the logical condition
- E1 <relational-operator> E2
- E1, E2 = arithmetic expressions
- <relational-operator> = < <= > >= |=
- Compound condition | & ~

Errors

- Boolean operator error (incorrect/missing/extra Boolean operators).
- Boolean variable error.
- Boolean parenthesis error.
- Relational operator error.
- Arithmetic expression error.

Loop testing

- Simple loops
  - n is the maximum number of allowable passes through the loop.

1. Skip the loop entirely.
2. Only one pass through the loop.
3. Two passes through the loop.
4. m passes through the loop where m< n.
5. n -1, n, n + 1 passes through the loop.

Loop testing

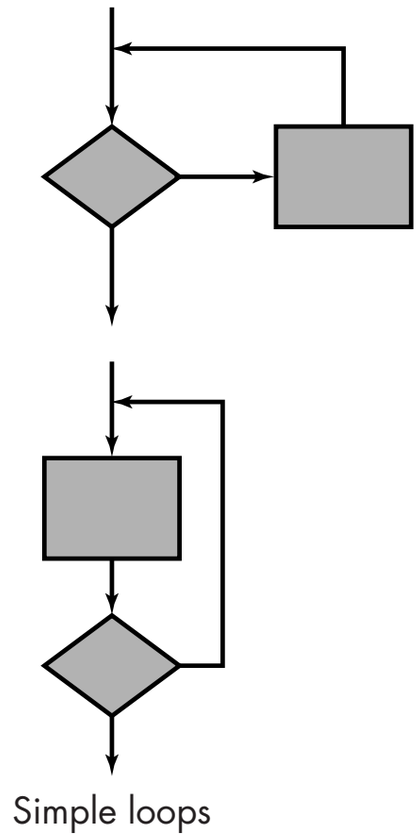
- Nested loops

1. Start at the innermost loop. Set all other loops to minimum values.

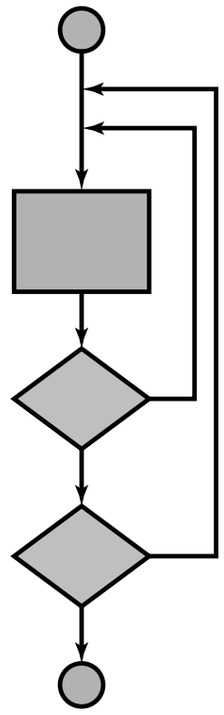
2. Conduct simple loop tests for the innermost loop while holding the outer loops at their minimum iteration parameter (e.g., loop counter) values. Add other tests for out-of-range or excluded values

3. Work outward, conducting tests for the next loop, but keeping all other outer loops at minimum values and other nested loops to "typical" values.

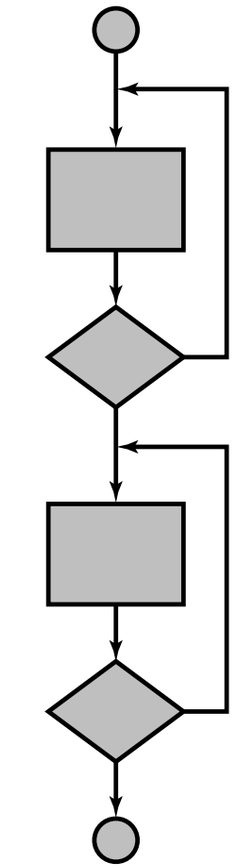
4. Continue until all loops have been tested.



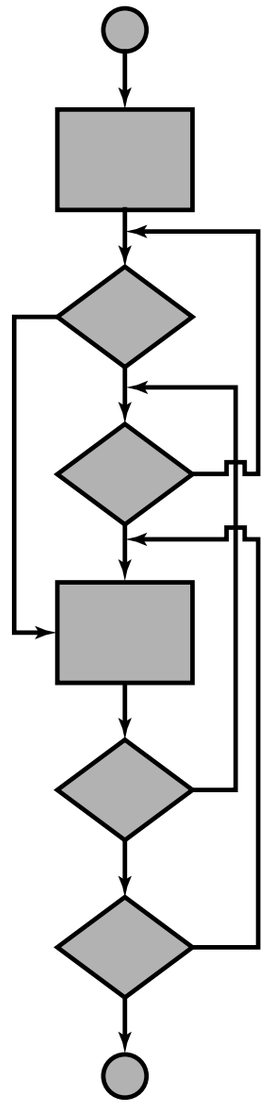
Simple loops



Nested loops



Concatenated loops



Unstructured loops

Classes of Loops

Black-Box testing

- Also called *behavior testing*
- Performs in later stages of testing
- Focuses on functional requirements of software
- Derive sets of input conditions to test
- Not an alternative to white-box
- likely to uncover a different class of error than white-box

Black-Box errors

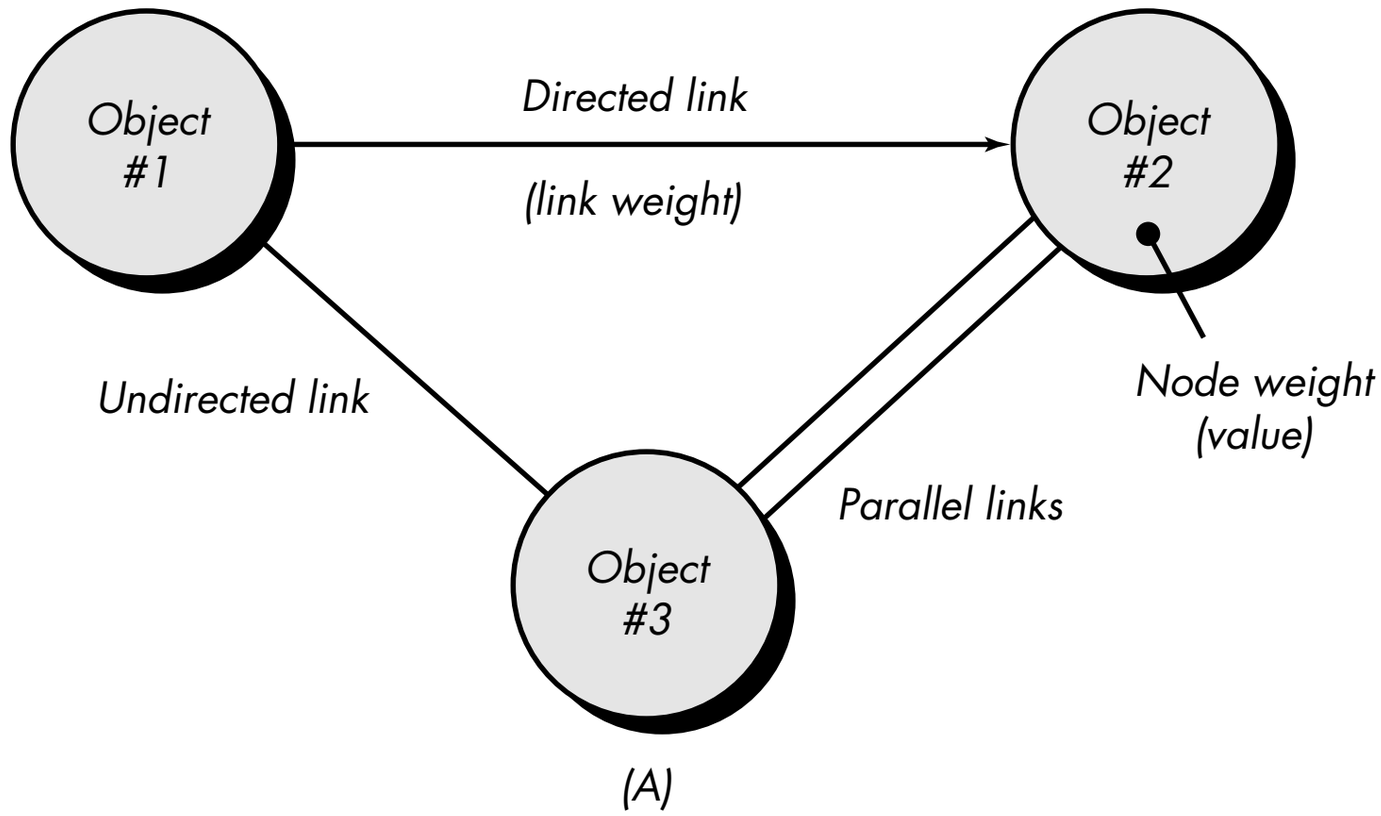
- Incorrect or missing functions
- Interface errors
- Errors in data structures or external data base access
- Behavior or performance errors
- Initialization and termination errors.

Test questions

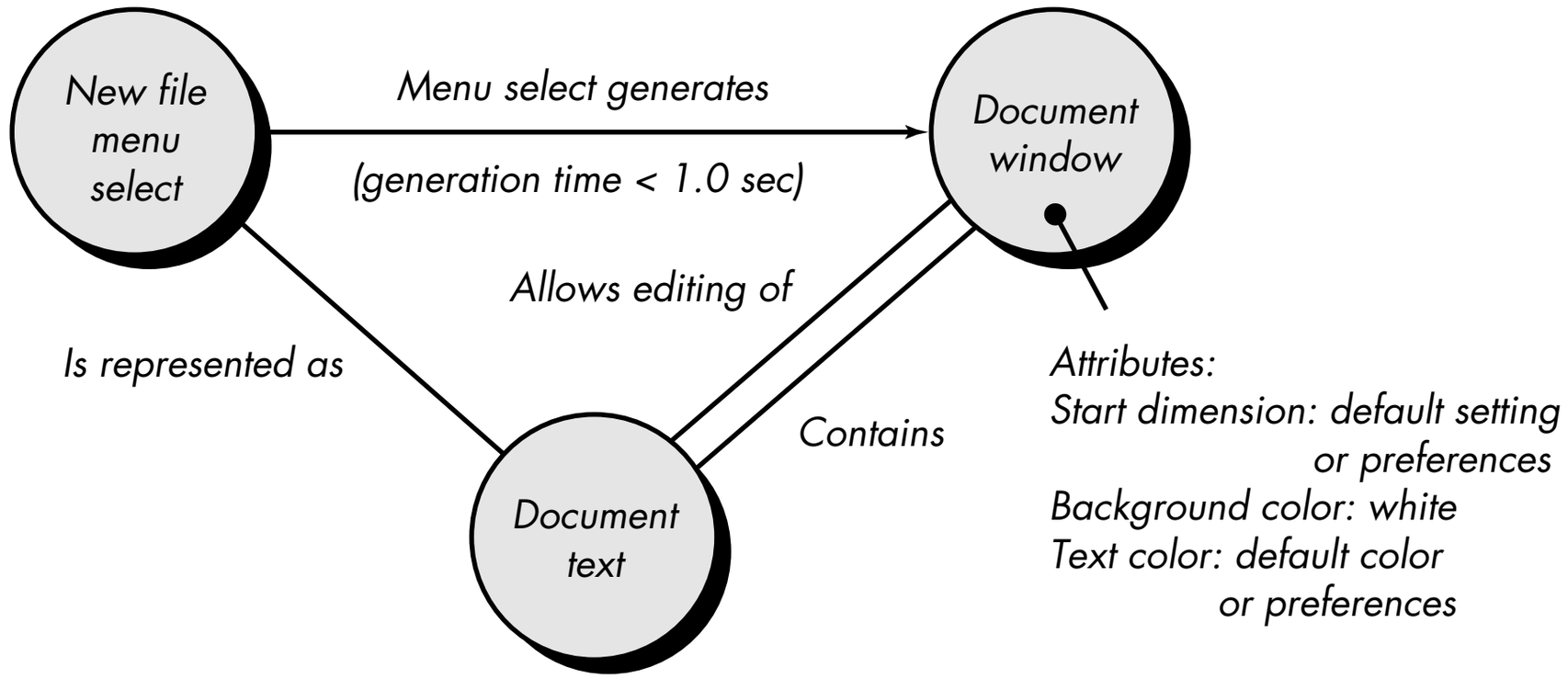
- How is functional validity tested?
- How is system behavior and performance tested?
- What classes of input will make good test cases?
- Is the system particularly sensitive to certain input values?
- How are the boundaries of a data class isolated?
- What data rates and data volume can the system tolerate?
- What effect will specify combinations of data have on system operation?

Graph-based testing

- Software testing begins by creating a graph of important objects and their relationships.
- Creates a series of tests that will cover the graph so that each object and relationship is exercised and errors are uncovered.


Object #1
Directed link
(link weight)
Object #2
Undirected link
Node weight (value)
Parallel links
Object #3

(A)


New file menu select
Menu select generates
(generation time < 1.0 sec)
Document window
Allows editing of
Is represented as
Contains
Attributes:
Start dimension: default setting or preferences
Background color: white
Text color: default color or preferences
Document text

(B)

Equivalence partitioning

- Black-box methods that divides the input domain of software into classes of data which test cases can be derived.
- Reduce number of test cases

classes guidelines

1. If an input condition specifies a range,one valid and two invalid equivalence classes are defined.
2. If an input condition requires a specific value,one valid and two invalid equivalence classes are defined.
3. If an input condition specifies a member of a set,one valid and one invalid equivalence class are defined.
4. If an input condition is Boolean,one valid and one invalid class are defined.

Example Banking app.

- area code—blank or three-digit number
- prefix—three-digit number not beginning with 0 or 1
- suffix—four-digit number
- password—six digit alphanumeric string
- commands—check, deposit, bill pay, and the like

Example Banking app.

area code: Input condition, Boolean—the area code may or may not be present.

Input condition, range—values defined between 200 and 999, with specific exceptions.

prefix: Input condition, range—specified value >200

Input condition, value—four-digit length

password: Input condition, Boolean—a password may or may not be present.

Input condition, value—six-character string.

command: Input condition, set—containing commands noted previously.

Boundary Value Analysis

- Error tends to occur at the boundaries of input domain rather than center
- Selects test cases at the “edged”

BVA Guideline

1. If an input condition specifies a range bounded by values a and b,test cases should be designed with values aand band just above and just below a and b.

2. If an input condition specifies a number of values, test cases should be developed that exercise the minimum and maximum numbers. Values just above and below minimum and maximum are also tested.

3. Apply guidelines 1 and 2 to output conditions. For example, assume that a temperature vs. pressure table is required as output from an engineering analysis program. Test cases should be designed to create an output report that produces the maximum (and minimum) allowable number of table entries.

4. If internal program data structures have prescribed boundaries (e.g., an array has a defined limit of 100 entries), be certain to design a test case to exercise the data structure at its boundary.

Comparison Testing

- Reliability comes first (military, medical)
- Redundant system software & hardware
- executes in parallel with real-time comparison

Testing Etc.

- Testing GUIs
- Testing Client/Server Architectures
- Testing for real-time systems

Testing Documentation

- Does the documentation accurately describe how to accomplish each mode of use?
- Is the description of each interaction sequence accurate?
- Are examples accurate?
- Are terminology, menu descriptions, and system responses consistent with the actual program?
- Is it relatively easy to locate guidance within the documentation?
- Can troubleshooting be accomplished easily with the documentation?

Testing Documentation

- Are the document table of contents and index accurate and complete?
- Is the design of the document (layout, typefaces, indentation, graphics) conducive to understanding and quick assimilation of information?
- Are all software error messages displayed for the user described in more detail in the document? Are actions to be taken as a consequence of an error message clearly delineated?
- If hypertext links are used, are they accurate and complete?
- If hypertext is used, is the navigation design appropriate for the information required?