PROMPT Software Testing

Types of Test Design Techniques

SWEDISH ICT SICS

>PROMPT

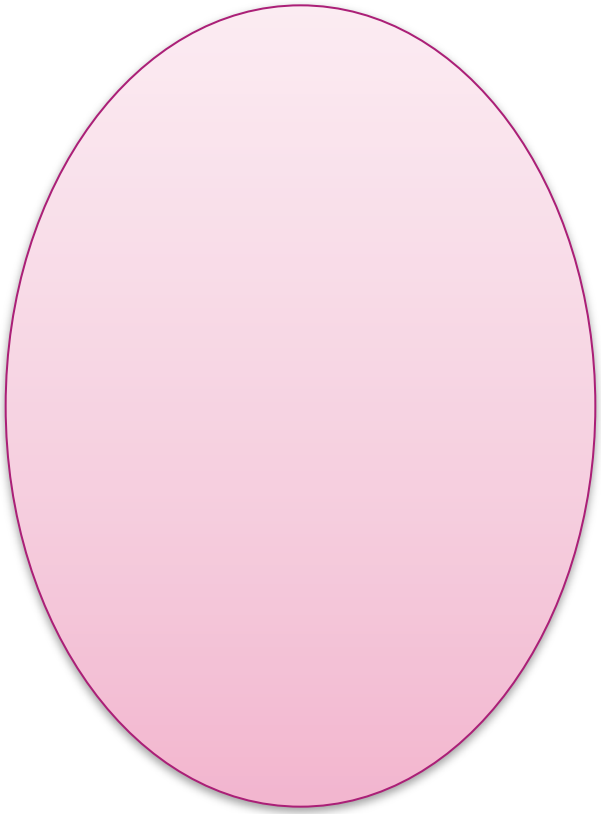
Test Design Techniques

...can be classified based on several different aspects

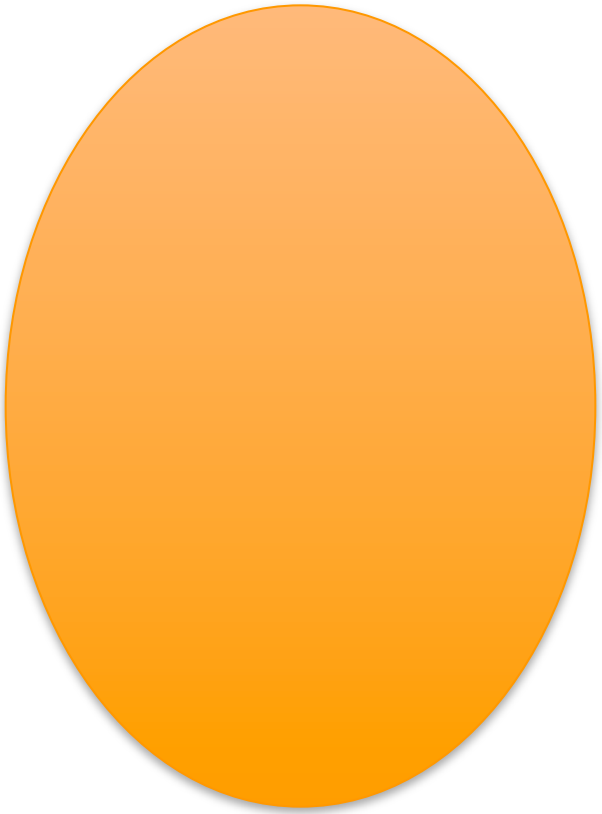
- Type of abstraction
  - Graph-based, logic-based, syntax-based, etc.
- Software engineering artifact
  - Specification-based versus implementation-based
- Level of knowledge of the inner workings of the software
  - Black-box, Gray-box, or White-box

However, we will start from the perspective of software behavior

Software Behavior?

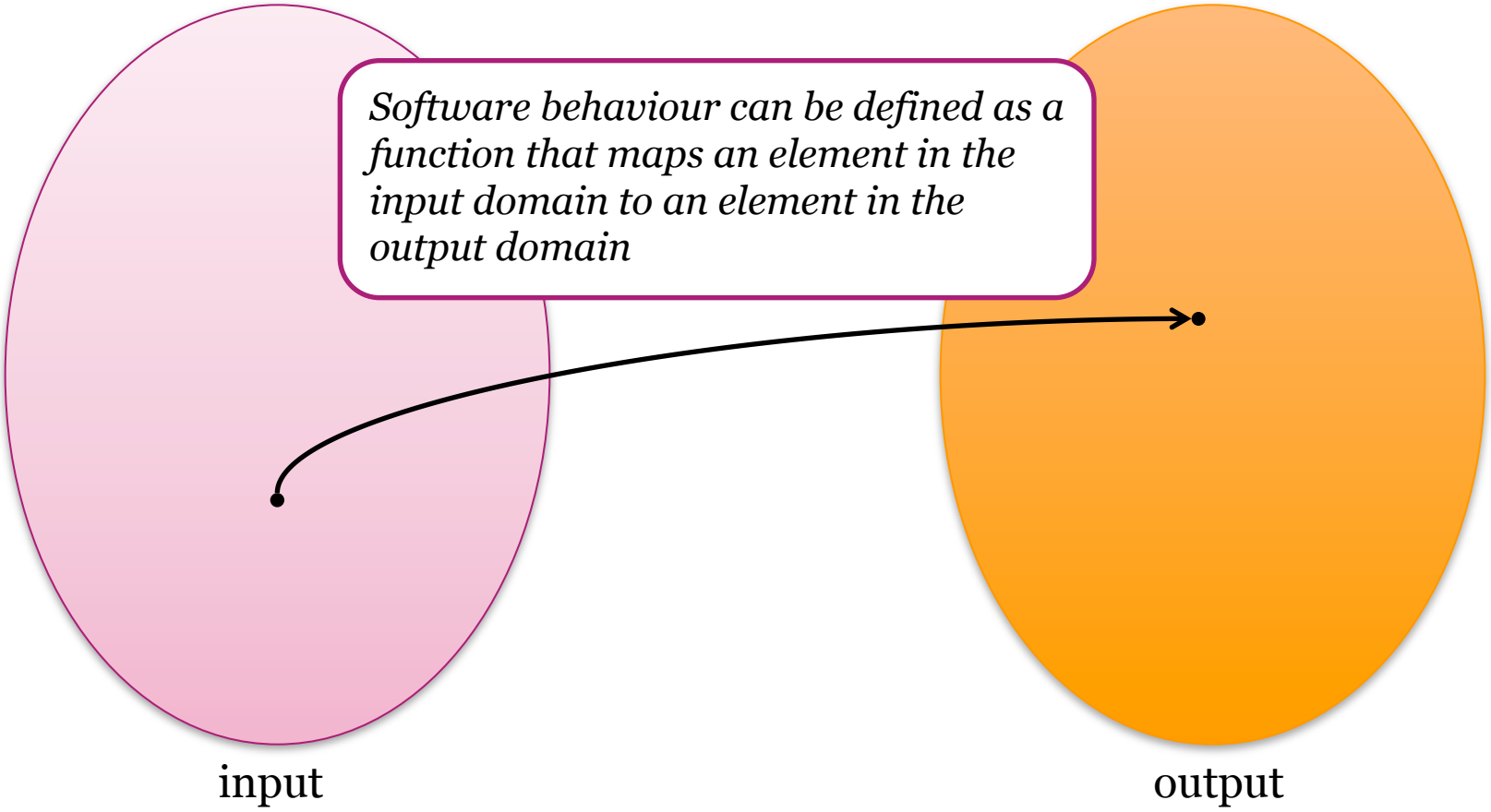


input

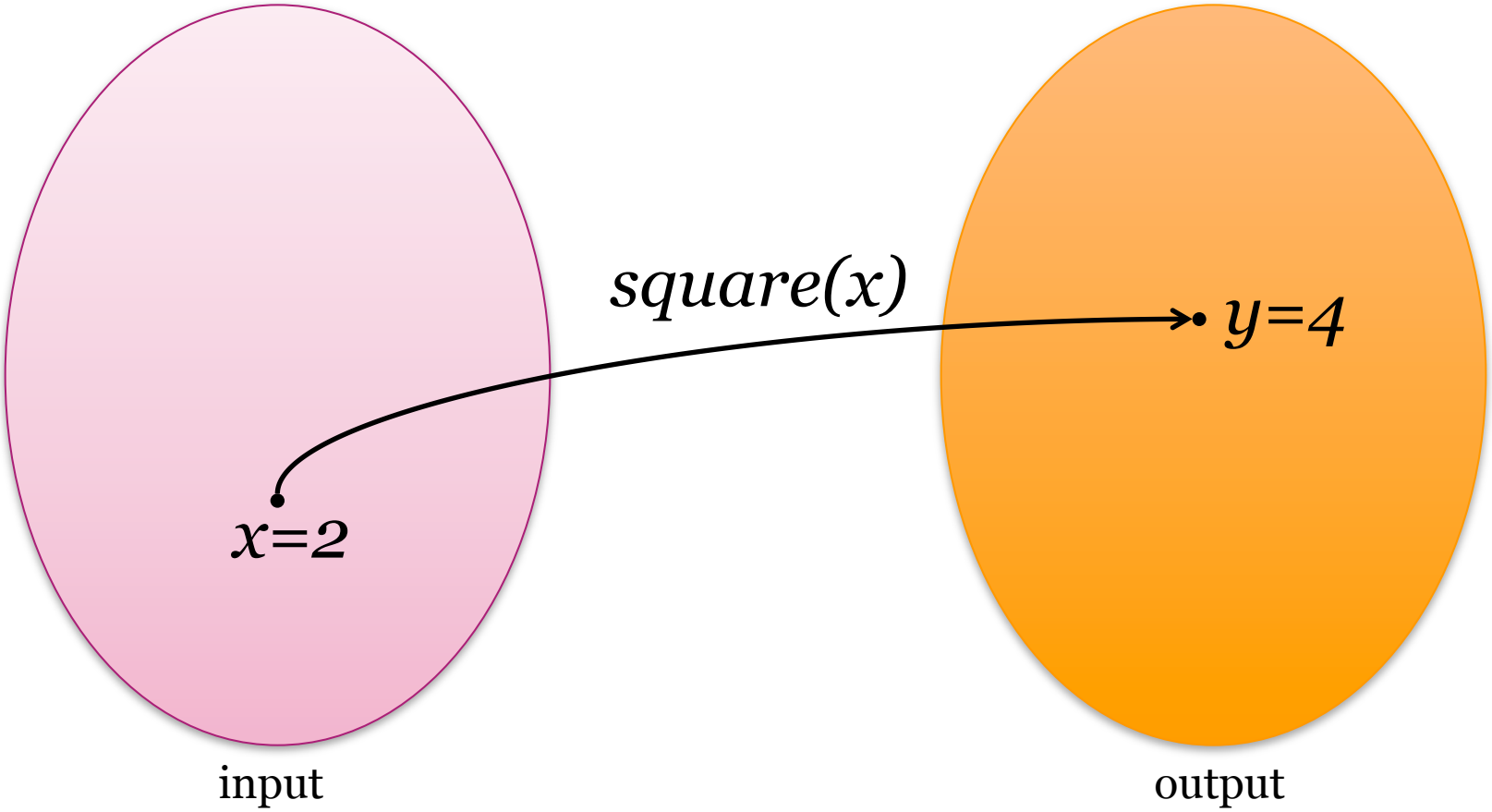


output

Software Behavior?

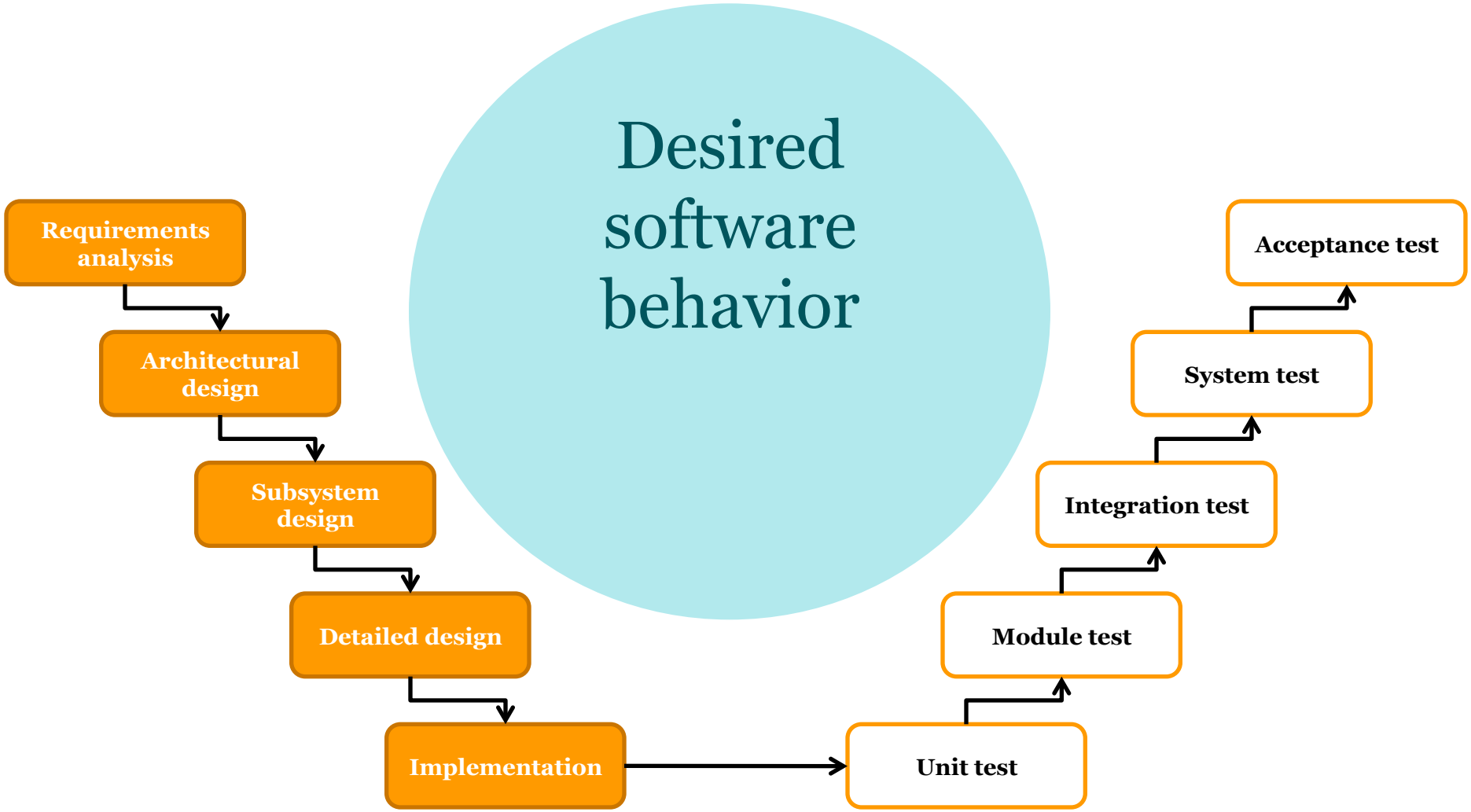


Software Behavior: example



Function: y=square(x)

Software Behavior



Software Behavior

Needs or ideas

Desired software behavior

Requirements analysis

Architectural design

Subsystem design

Detailed design

Implementation

Unit test

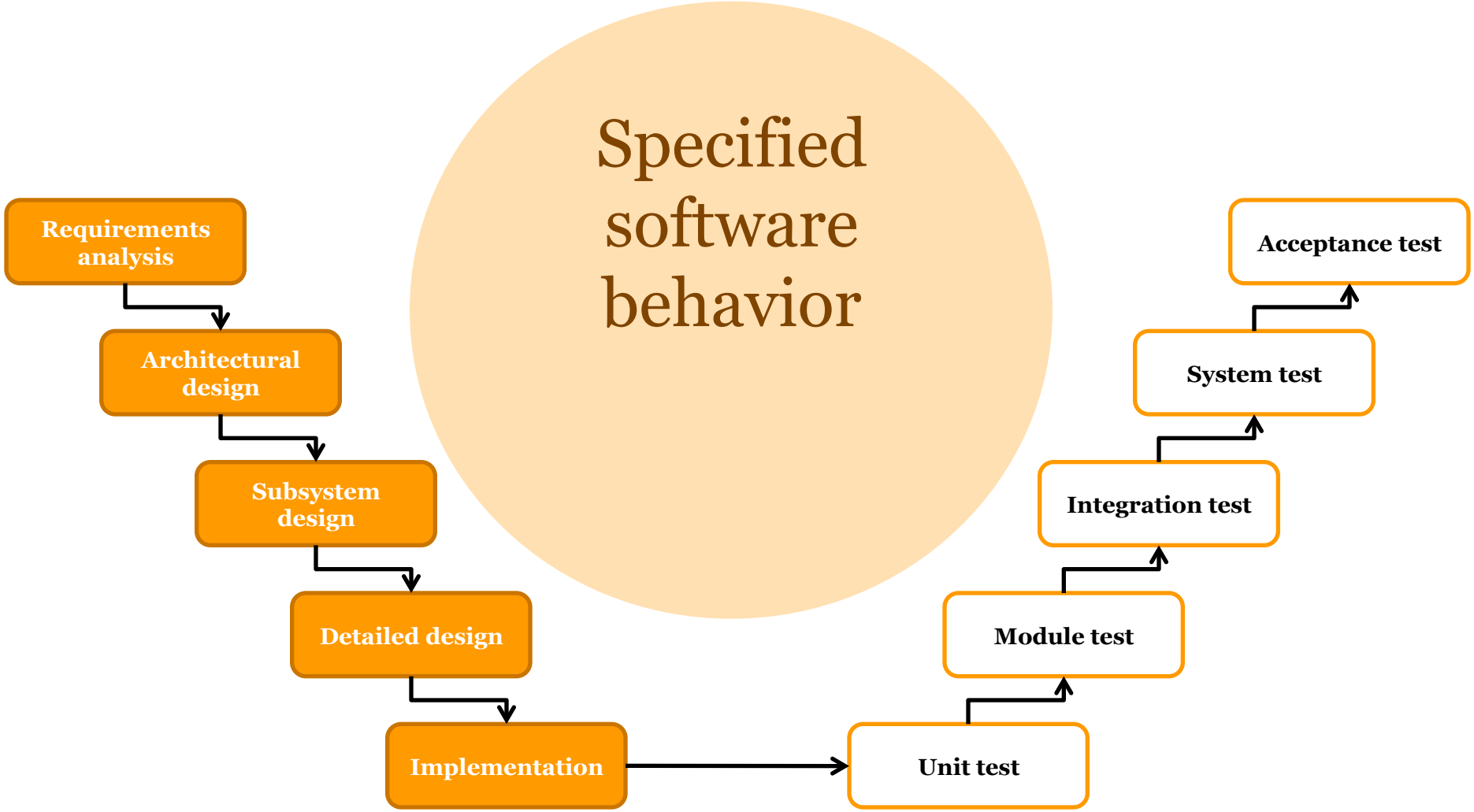
Module test

Integration test

System test

Acceptance test

Software Behavior



Software Behavior

Specified software behavior

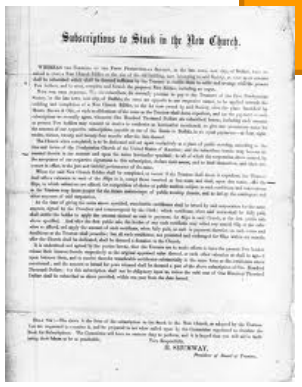
Requirements analysis

Architectural design

Subsystem design

Detailed design

Implementation


Subscriptions to Stock in the New Church.

Unit test

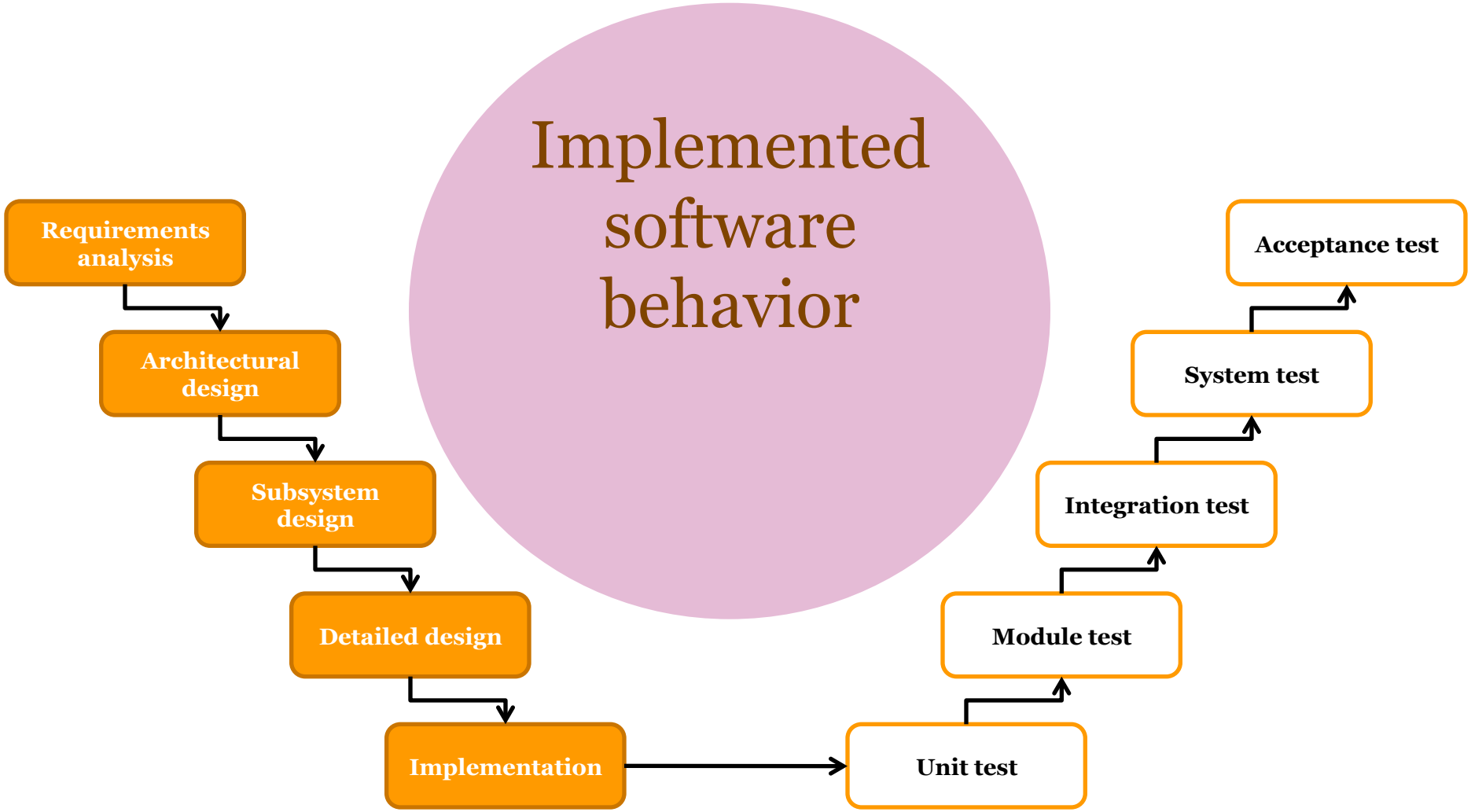
Module test

Integration test

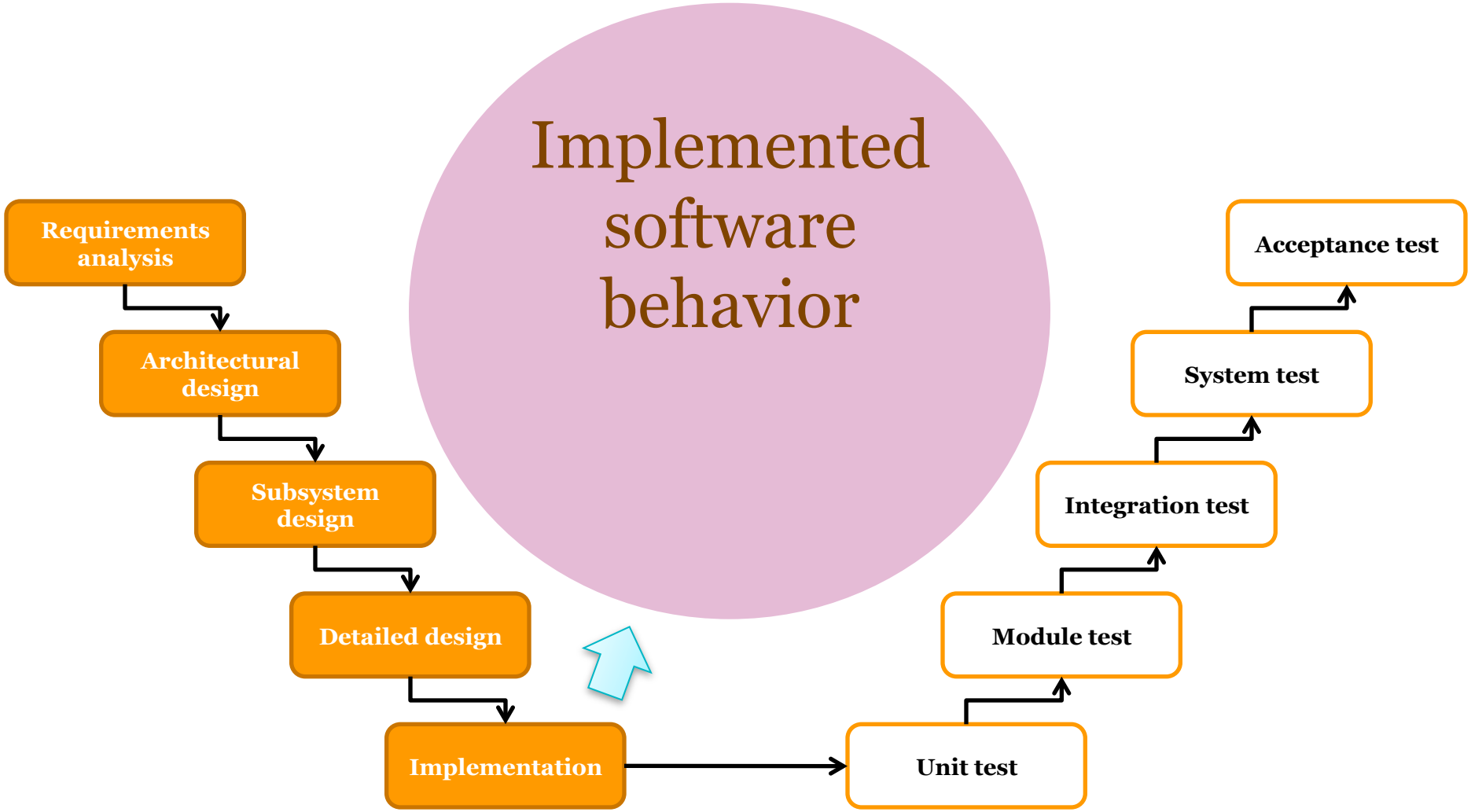
System test

Acceptance test

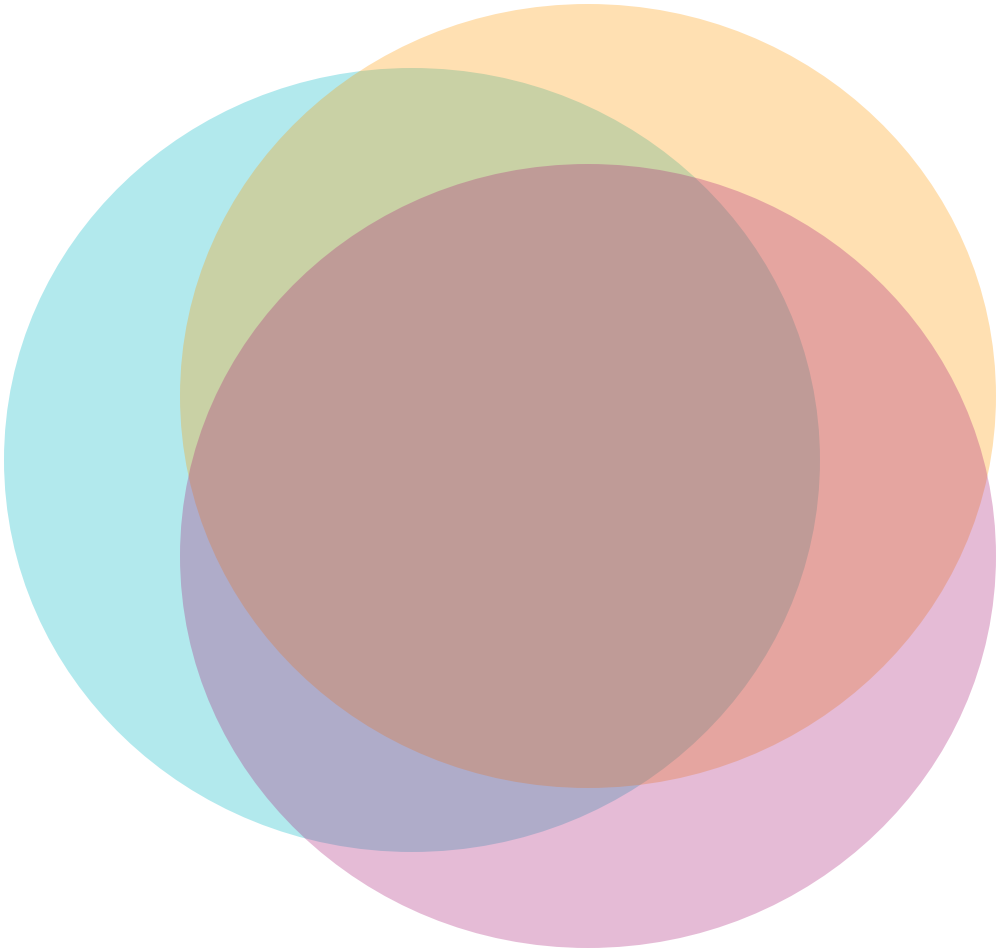
Software Behavior



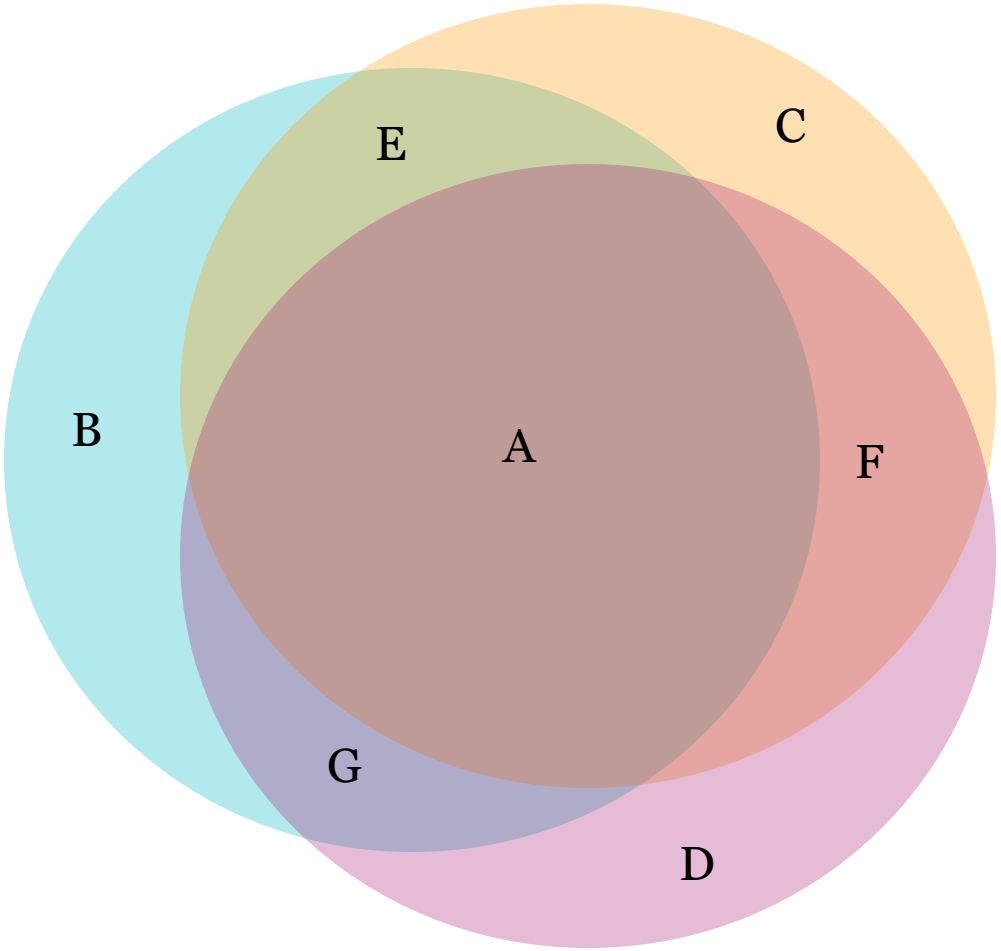
Software Behavior



Testing and Software Behavior

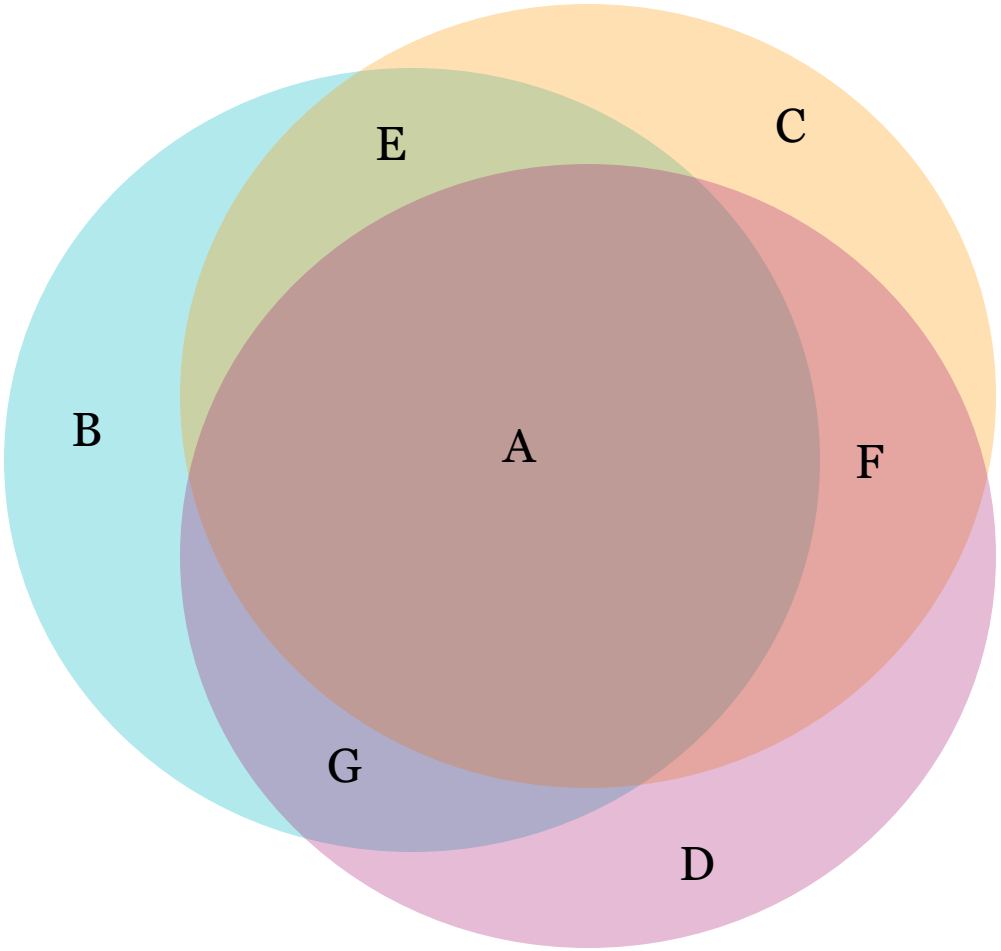


Testing and Software Behavior (cont.)



Testing and Software Behavior (cont.)

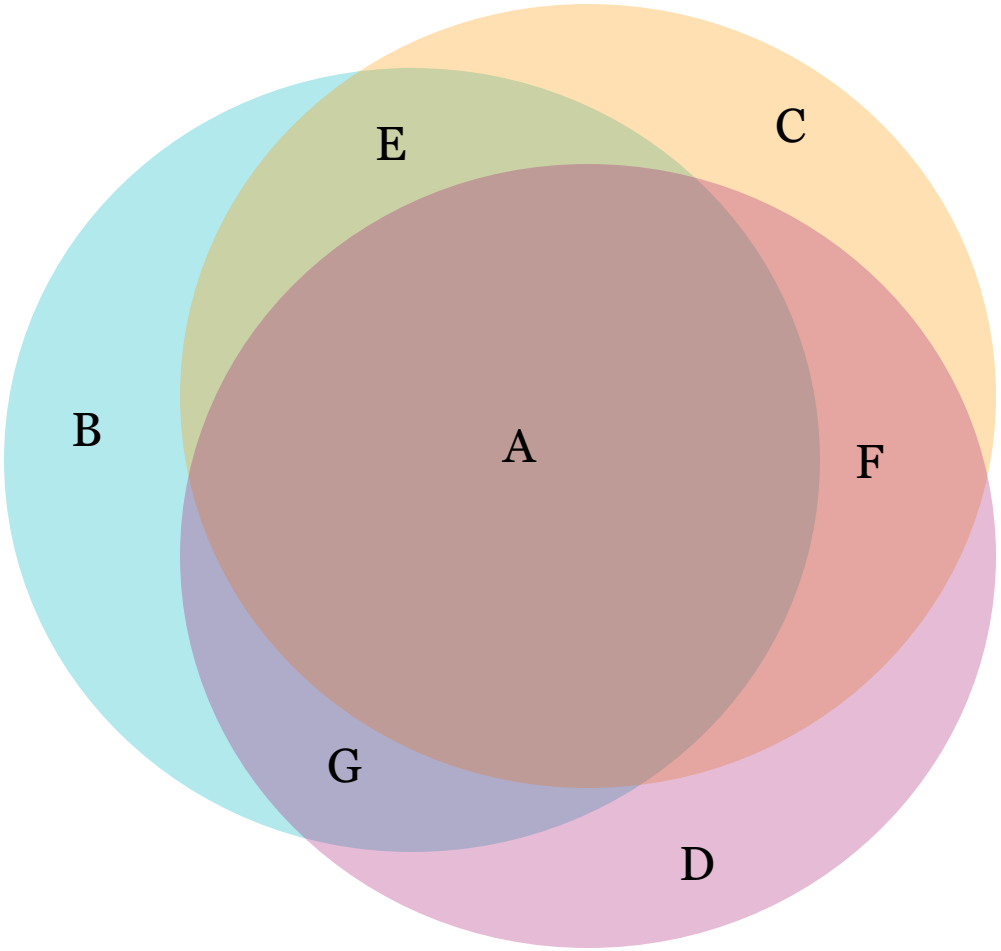
A: Desired, specified and implemented software behavior



Testing and Software Behavior (cont.)

A: Desired, specified and implemented software behavior

B: Desired, but not specified or implemented software behavior

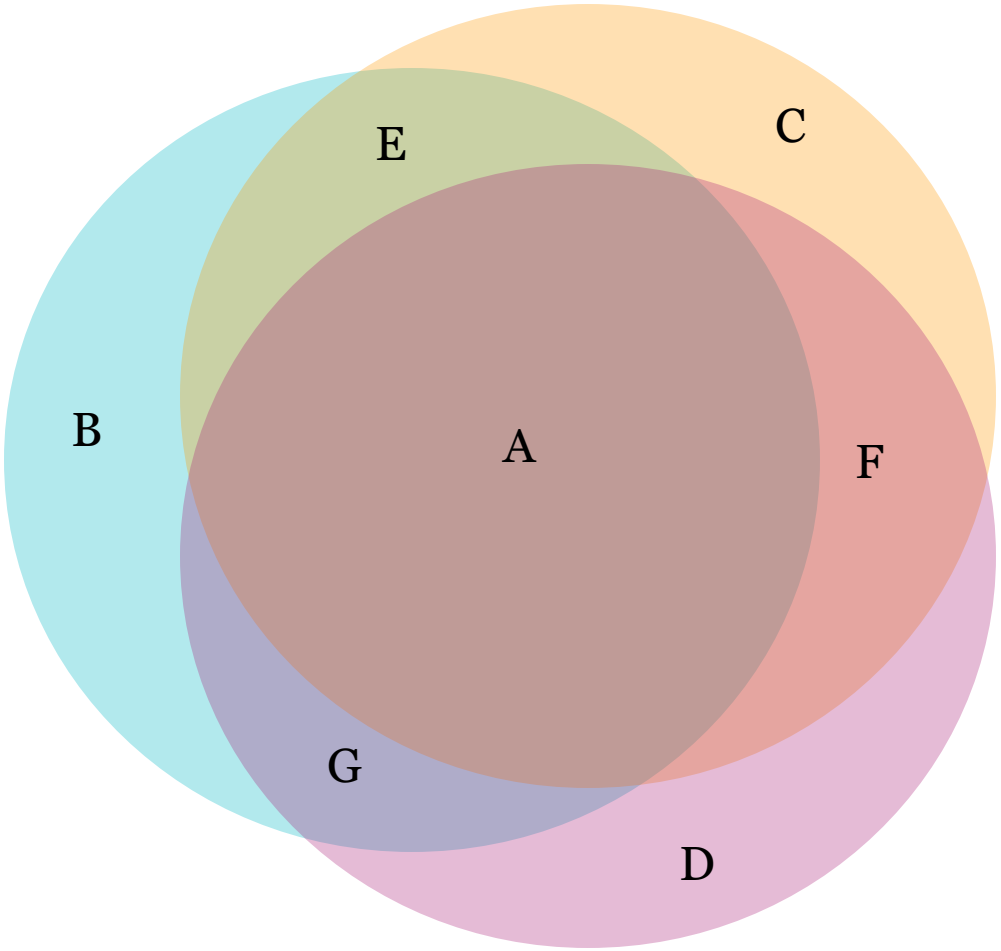


Testing and Software Behavior (cont.)

A: Desired, specified and implemented software behavior

B: Desired, but not specified or implemented software behavior

C: Specified, but not desired or implemented software behavior



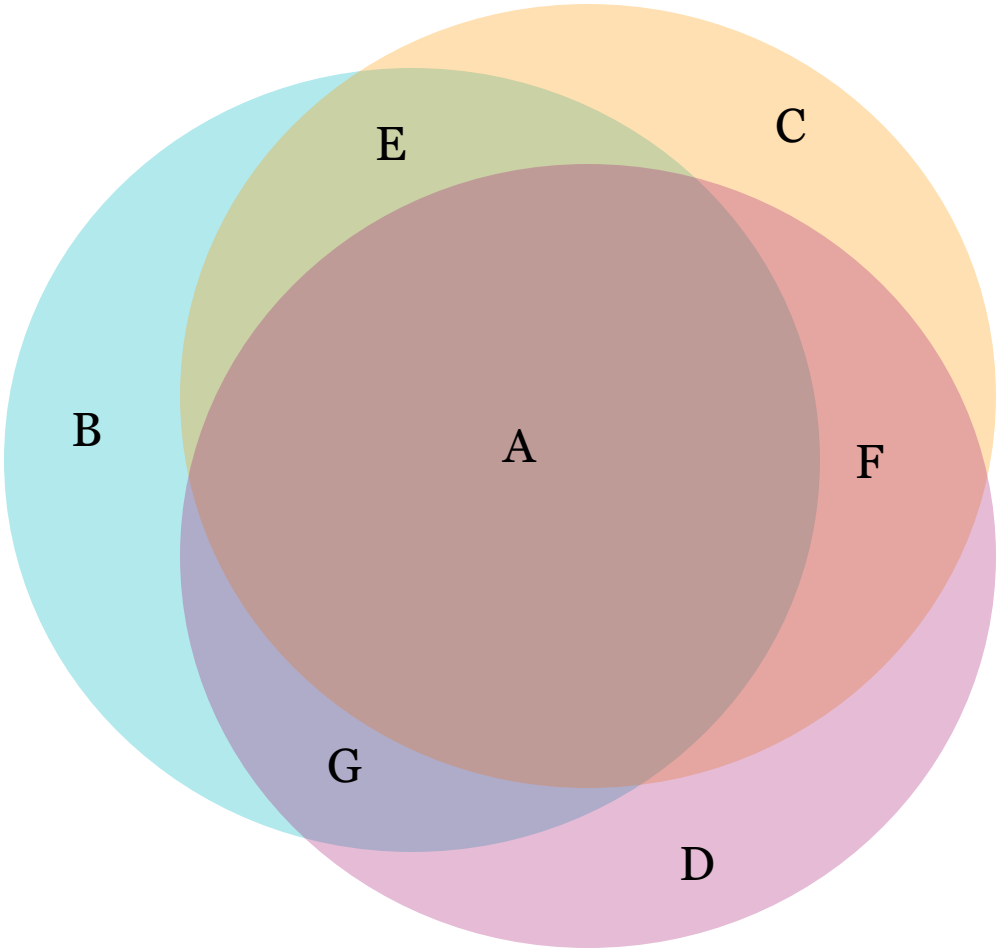
Testing and Software Behavior (cont.)

A: Desired, specified and implemented software behavior

B: Desired, but not specified or implemented software behavior

C: Specified, but not desired or implemented software behavior

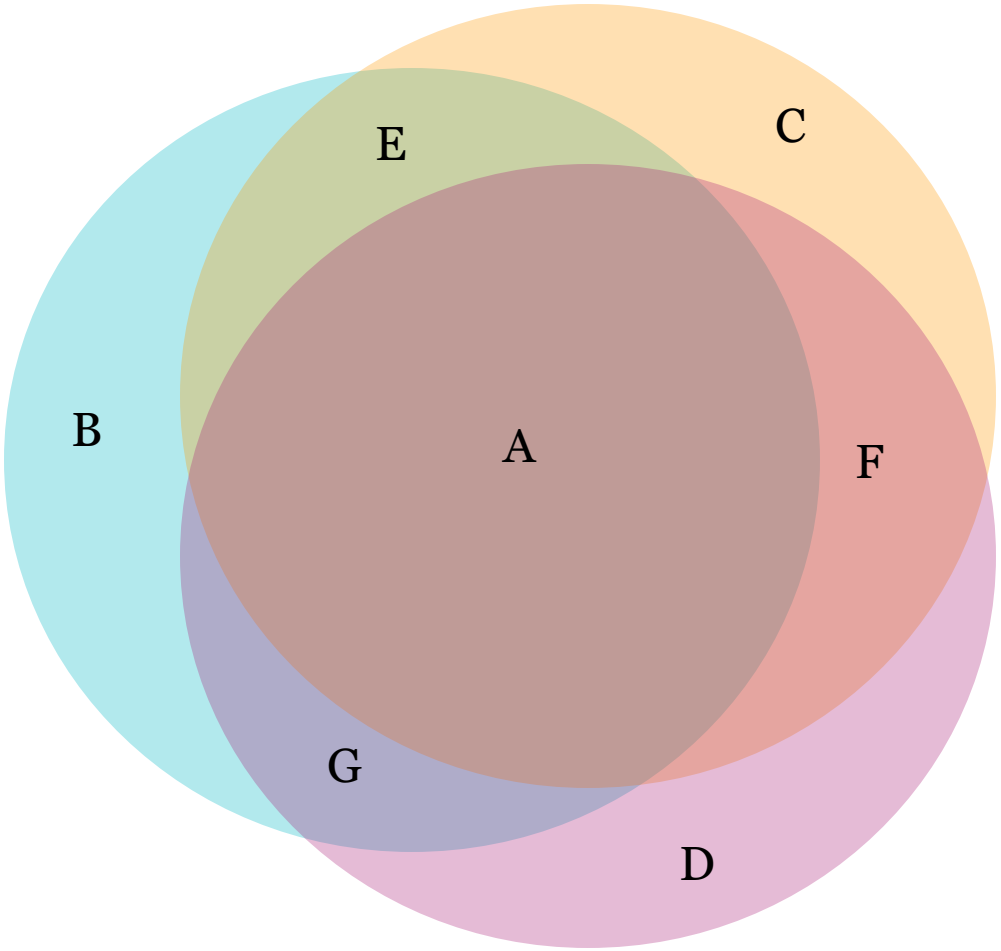
D: Implemented, but not desired or specified software behavior



Testing and Software Behavior (cont.)

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior

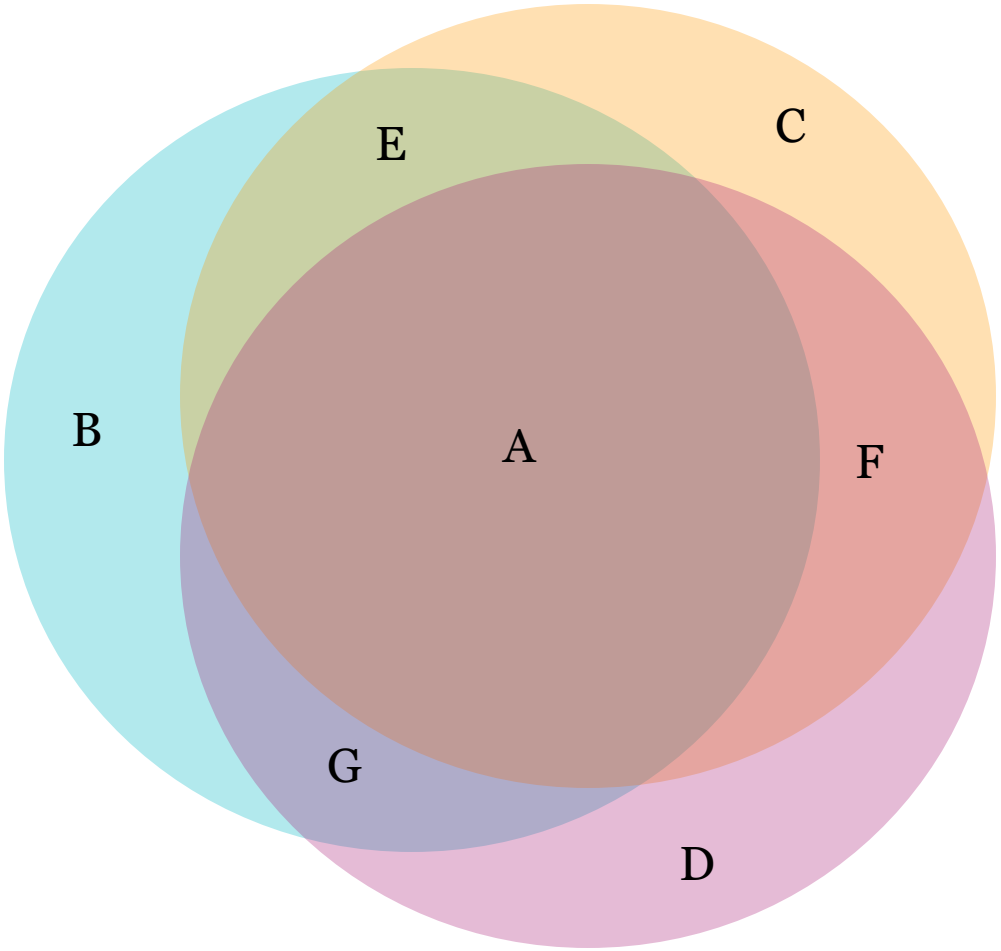
E: Desired and specified, but not implemented software behavior



Testing and Software Behavior (cont.)

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not implemented software behavior

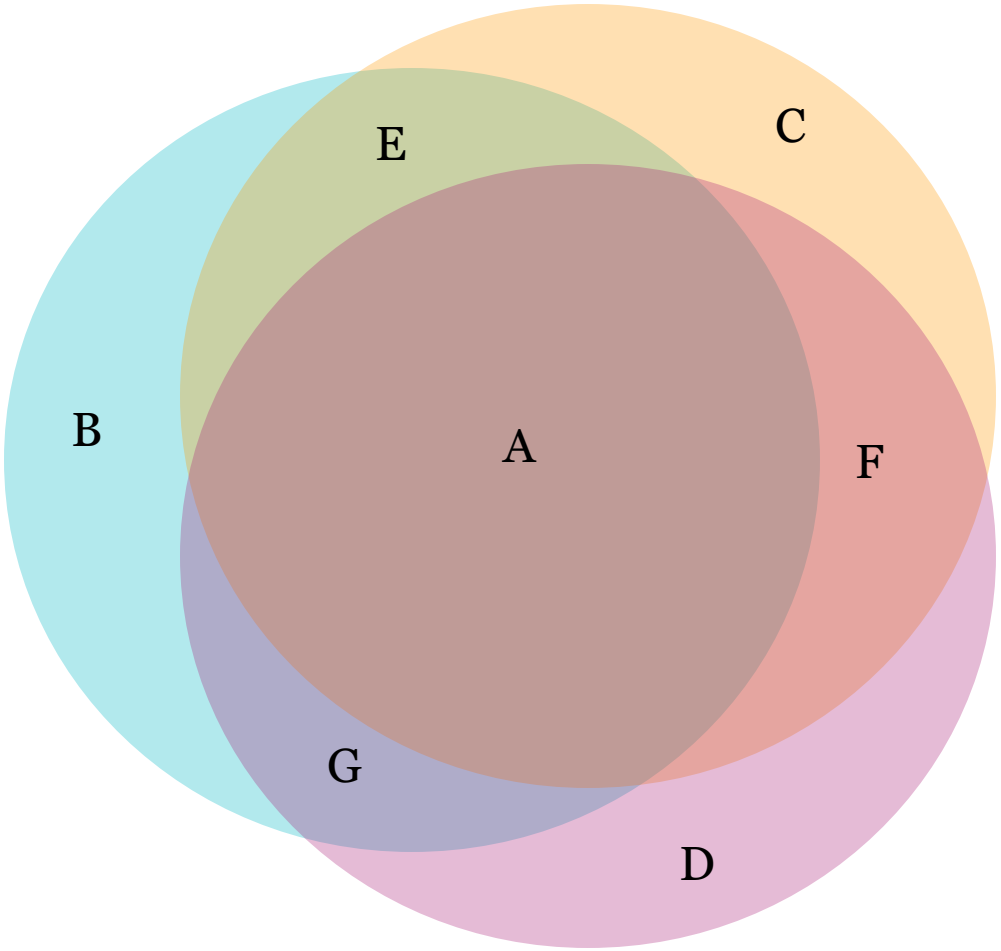
F: Specified and implemented, but not desired software behavior



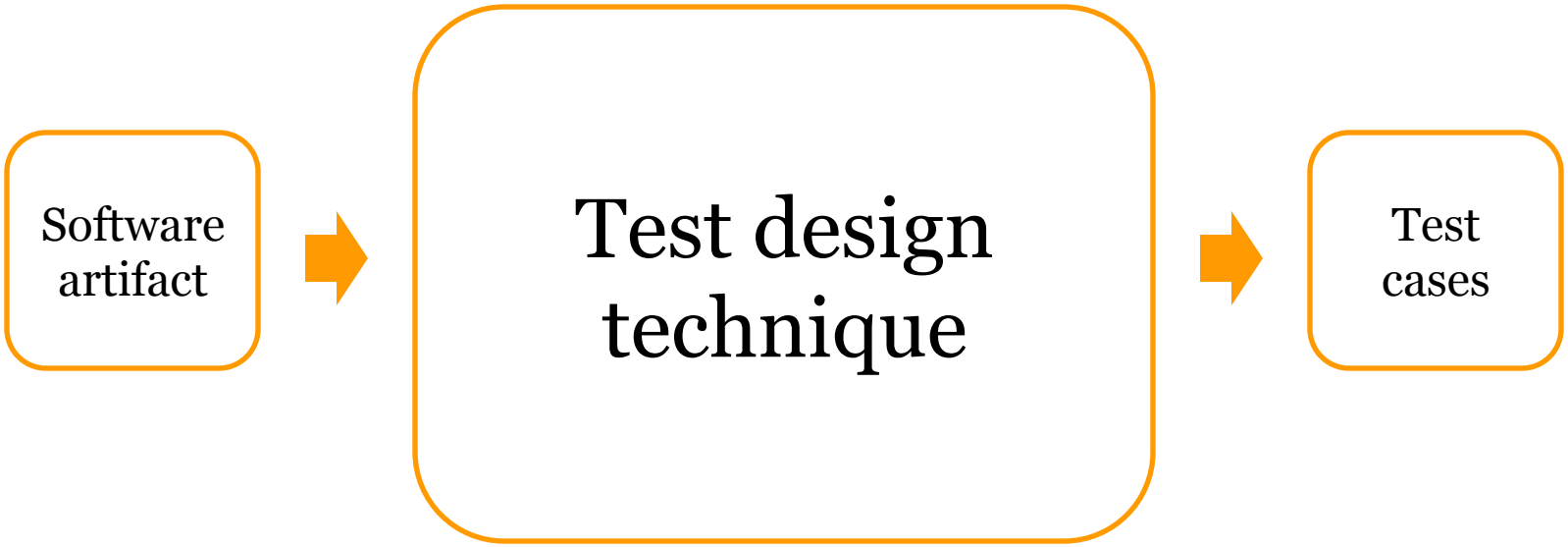
Testing and Software Behavior (cont.)

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not implemented software behavior
F: Specified and implemented, but not desired software behavior

G: Desired and implemented, but not specified software behavior



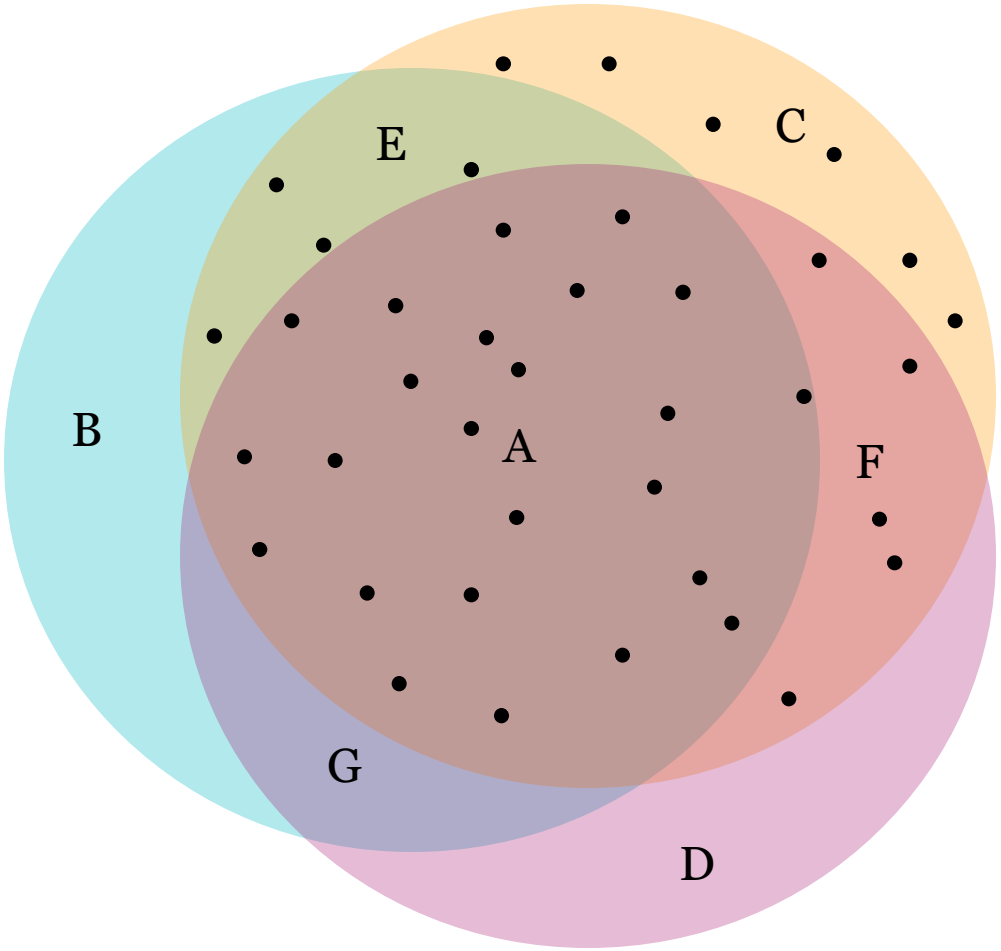
Test Design Techniques



- Based on this, how does the selection of software artifact affect the resulting set of test cases?

Specification-Based (Functional) Test Design

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not implemented software behavior
F: Specified and implemented, but not desired software behavior
G: Desired and implemented, but not specified software behavior



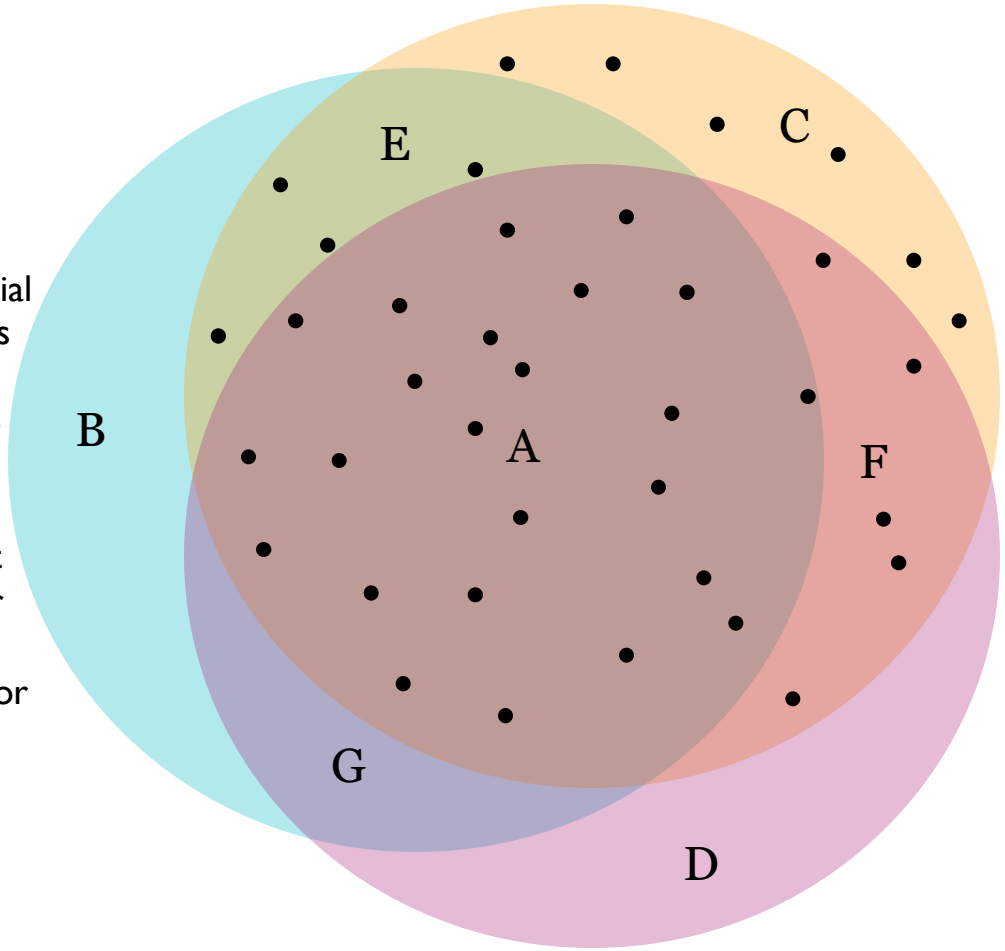
Specification-Based (Functional) Test Design

Example: ARIANE 5

"The specification of the inertial reference system and the tests performed at equipment level did not specifically include the Ariane 5 trajectory data.

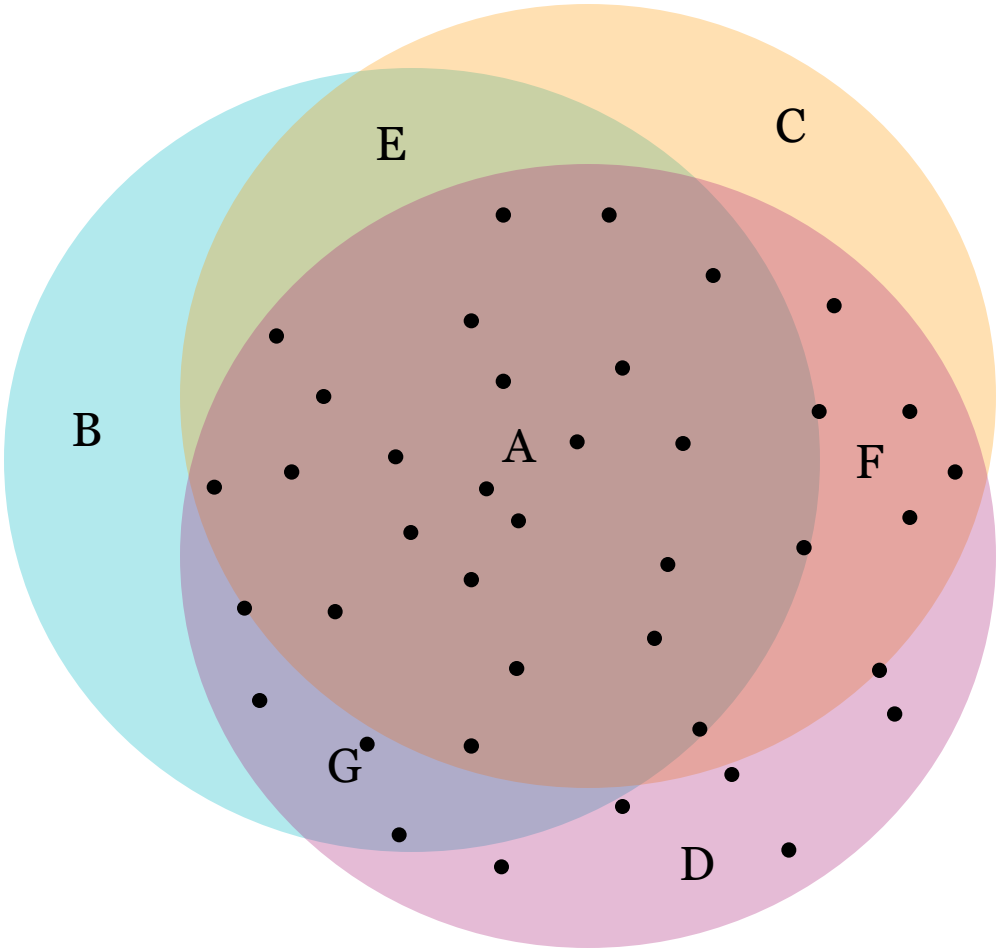
Consequently the realignment function was not tested under simulated Ariane 5 flight conditions, and the design error was not discovered."

- Wikipedia



Implementation-Based (Structural) Test Design

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not implemented software behavior
F: Specified and implemented, but not desired software behavior
G: Desired and implemented, but not specified software behavior

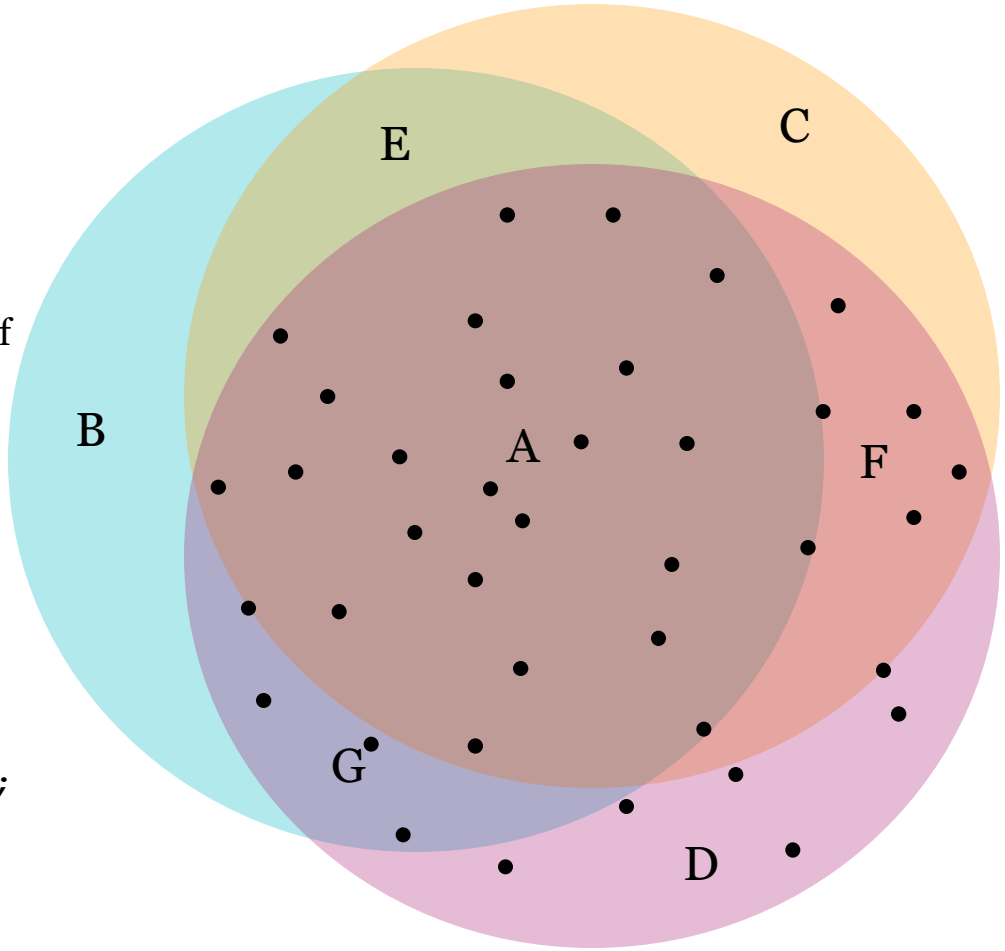


Implementation-Based (Structural) Test Design

Example: Day of the week

Given the number of the day, the "dayOfTheWeek" function prints the name of that day.

```
String dayOfTheWeek(int dayNr)
{
  switch (dayNr)
  {
    case 1: return "Monday";
    case 2: return "Tuesday";
    case 3: return "Wednesday";
    default: return "No such day";
  }
}
```

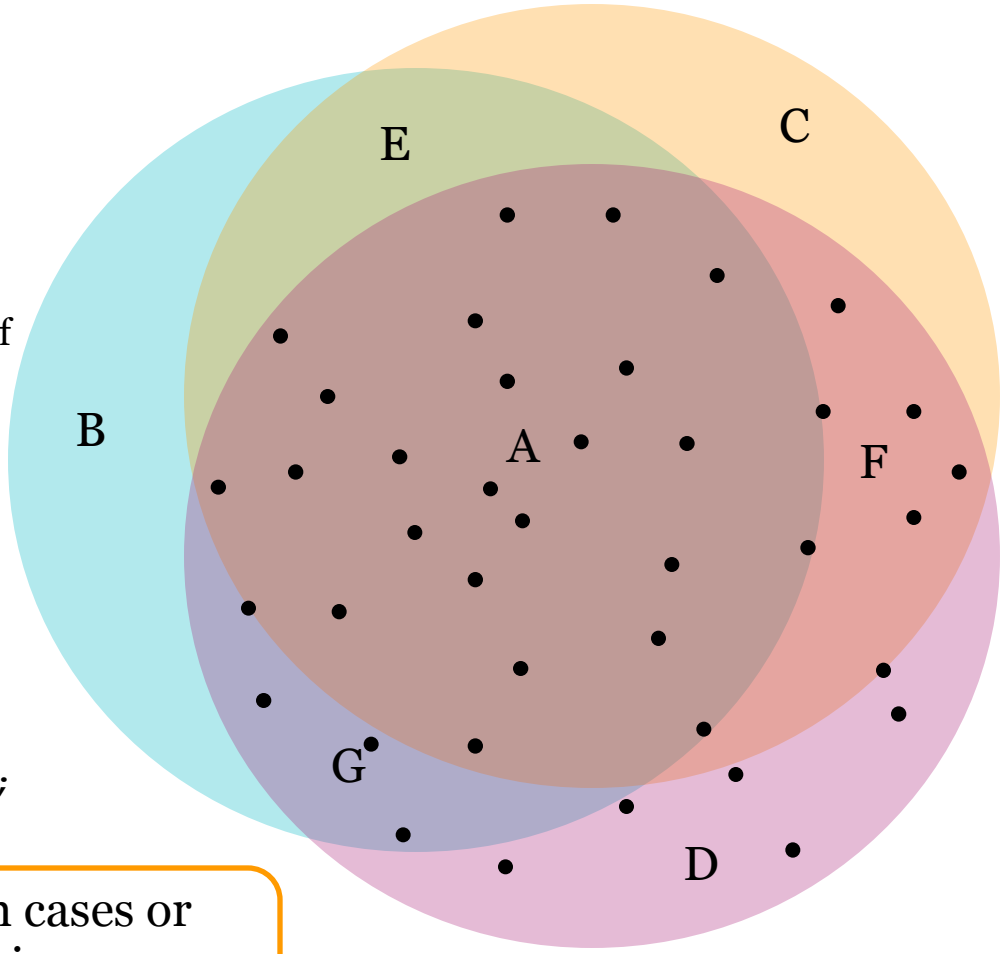


Implementation-Based (Structural) Test Design

Example: Day of the week

Given the number of the day, the "dayOfTheWeek" function prints the name of that day.

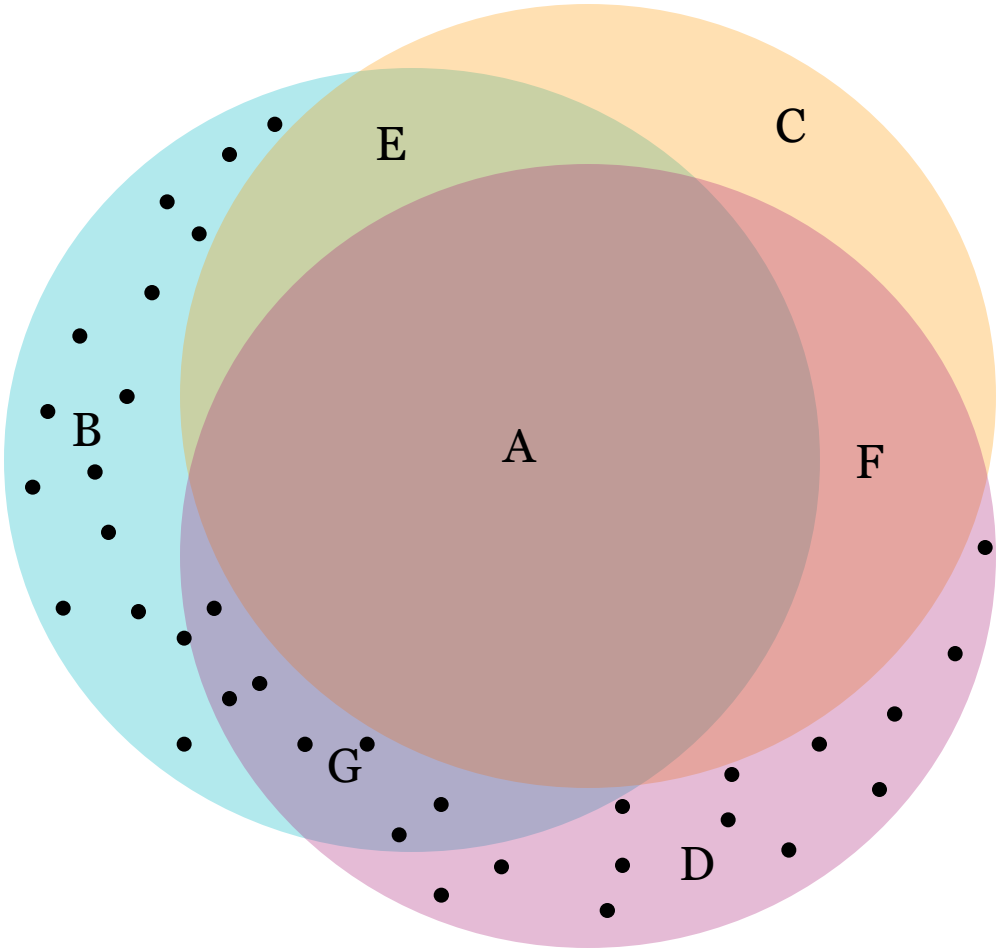
```
String dayOfTheWeek(int dayNr)
{
  switch (dayNr)
  {
    case 1: return "Monday";
    case 2: return "Tuesday";
    case 3: return "Wednesday";
    default: return "No such day";
  }
}
```



We can easily test all switch cases or statements here without coming across any problems.

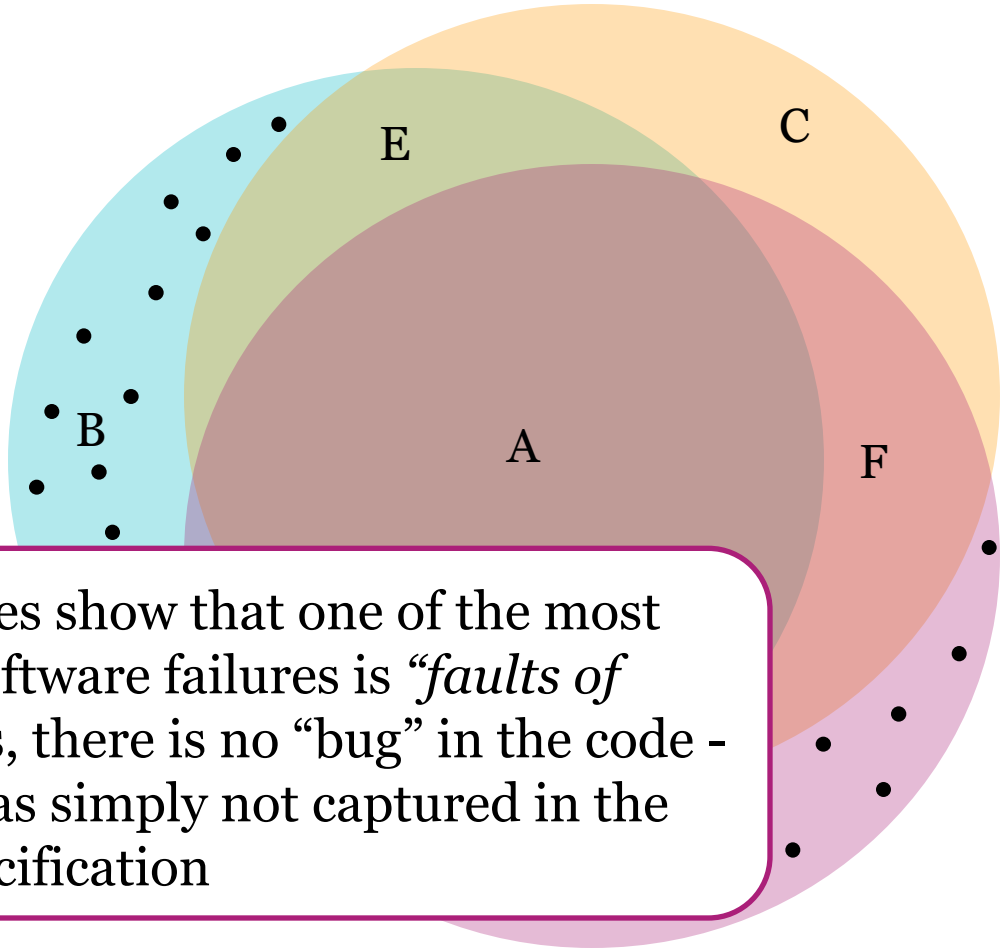
Negative Testing?

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not implemented software behavior
F: Specified and implemented, but not desired software behavior
G: Desired and implemented, but not specified software behavior



Negative Testing?

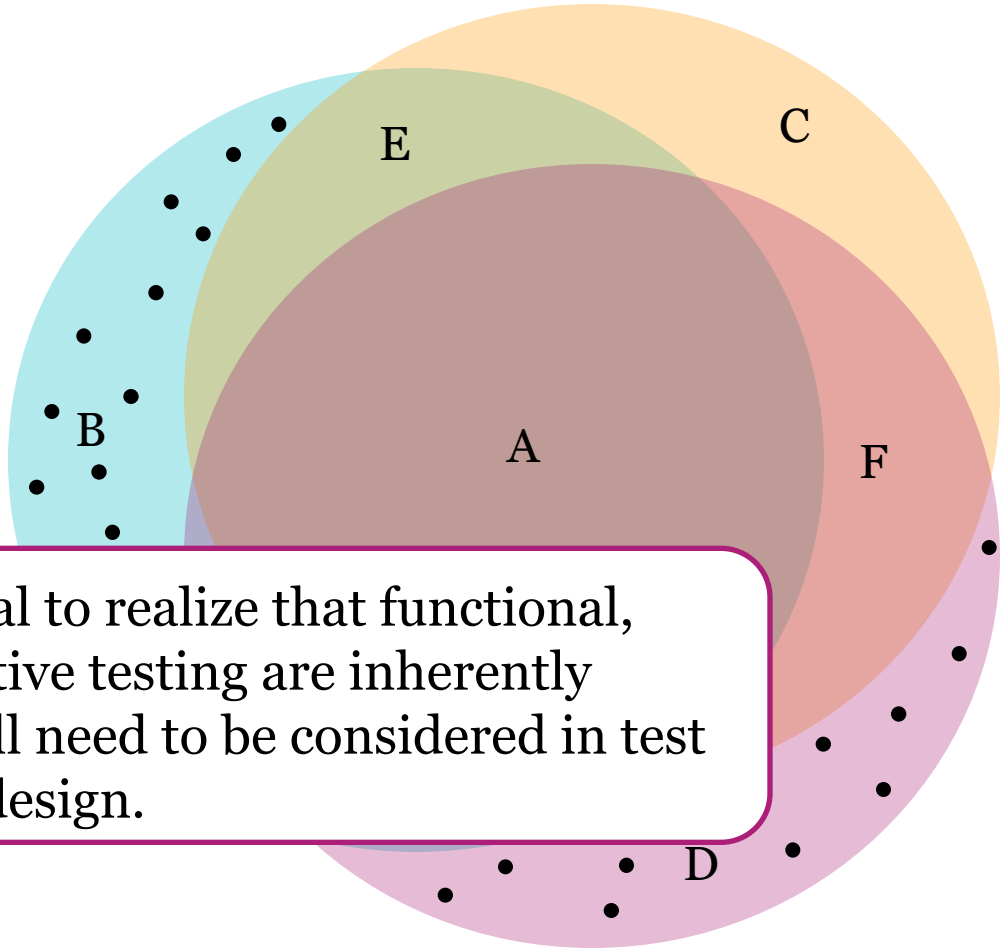
A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not
imp
F: S
but
G: I
not



Note that several studies show that one of the most common causes of software failures is *"faults of omission"*. In other words, there is no "bug" in the code - the desired behavior was simply not captured in the specification

Negative Testing?

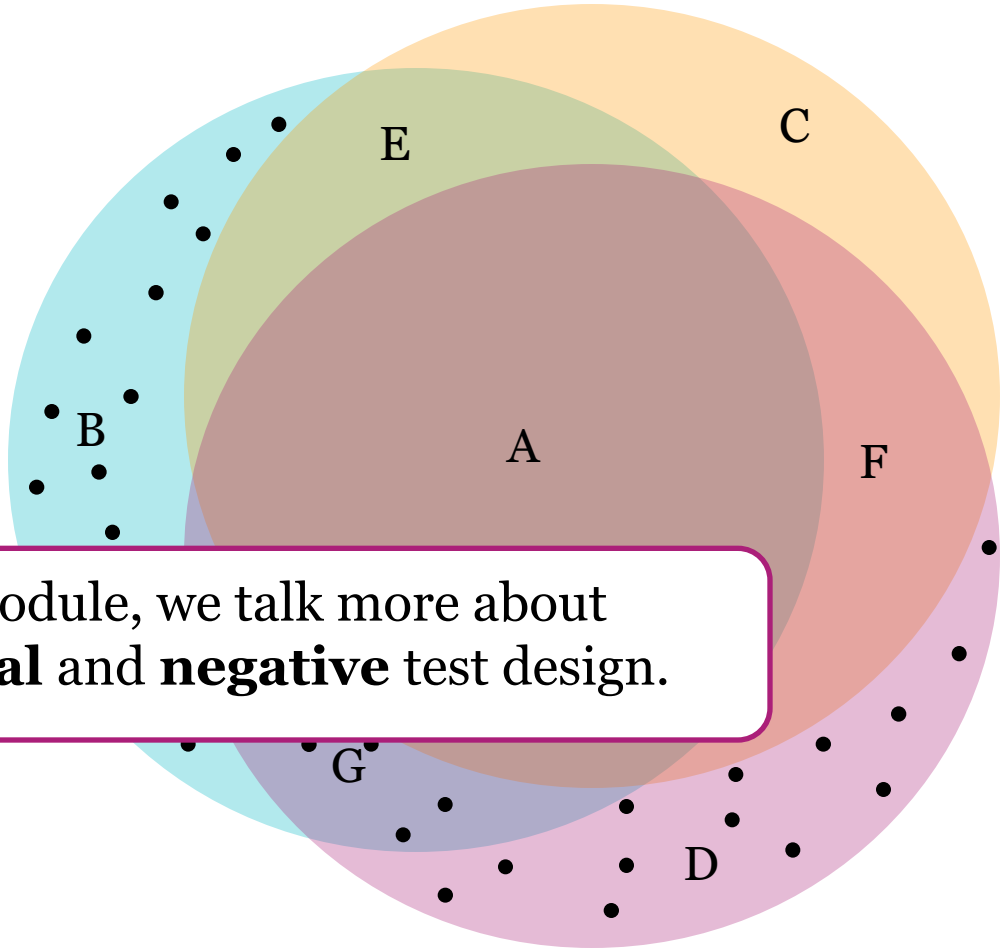
A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not imp
F: S
but
G: I
not s



Note also that it is vital to realize that functional, structural and negative testing are inherently **complementary** and all need to be considered in test design.

Negative Testing?

A: Desired, specified and implemented software behavior
B: Desired, but not specified or implemented software behavior
C: Specified, but not desired or implemented software behavior
D: Implemented, but not desired or specified software behavior
E: Desired and specified, but not imp
F: S
but
G: Desired and implemented, but not specified software behavior



In the unit testing module, we talk more about **functional, structural** and **negative** test design.