SOFTWARE ENGINEERING
CO3001

CHAPTER 9 – SOFTWARE QUALITY & QUALITY ASSURANCE

Anh Nguyen-Duc
Quan Thanh Tho

WEEK 9

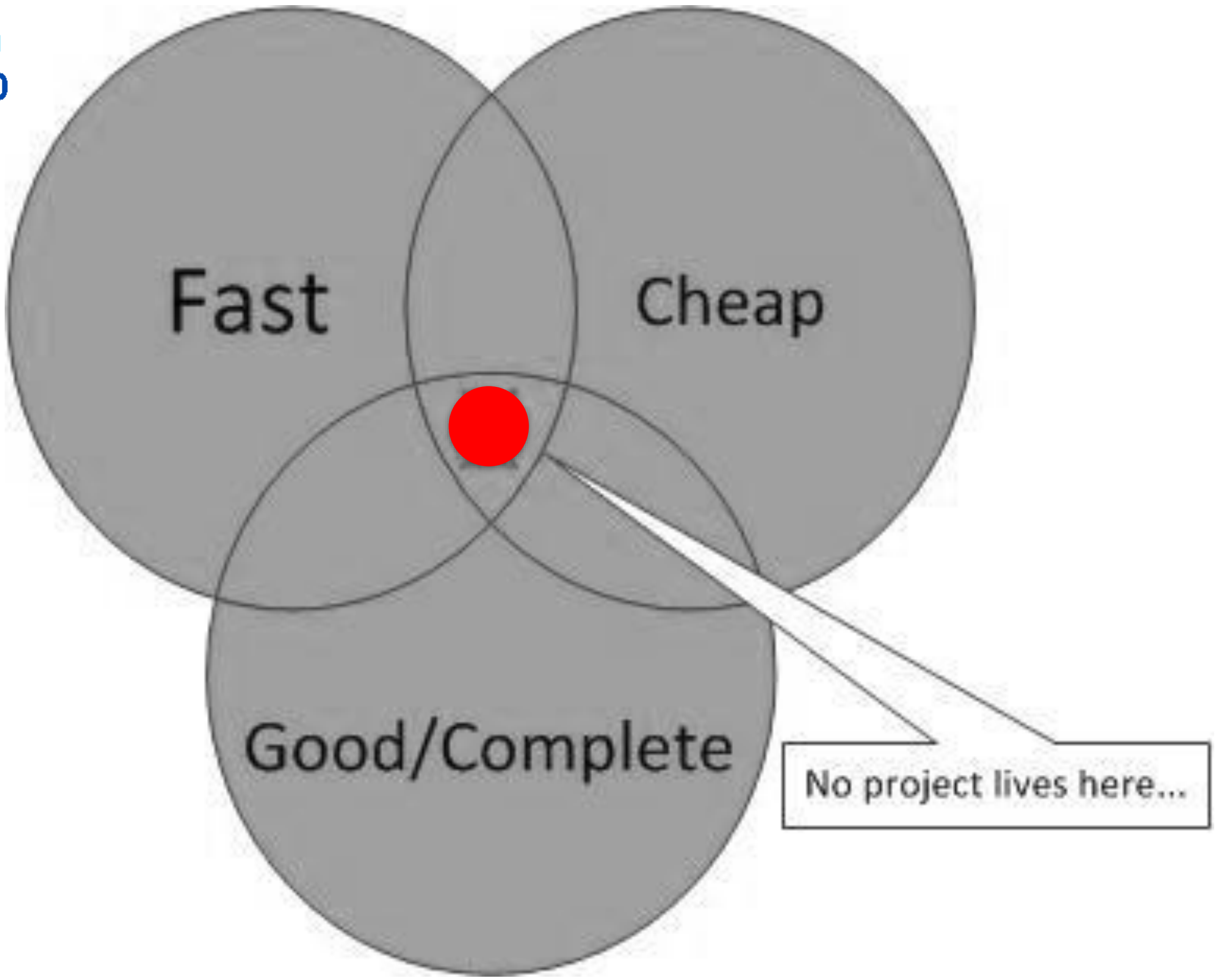
TOPICS COVERED

- ✓ Software Quality & its importance
- ✓ Software testing
- ✓ Test planning
- ✓ Test driven development

THE IMPORTANCE OF SOFTWARE QUALITY

WHY IS SOFTWARE QUALITY IMPORTANT?

DIMECC Program
NEED FOR SPEED



DEFINITION OF QUALITY

- ✓ (ISO) defines **quality** as the totality of characteristics of an entity that bear on its ability to satisfy stated or implied needs (ISO8042:1994) or the degree to which a set of inherent characteristics fulfils requirements. (ISO9000:2000).
- ✓ **Conformance to requirements** means the project s processes and products meet written specifications.
- ✓ **Fitness for use** means a product can be used as it was intended.
- ✓ ***Quality aspects:***
  - *product*: delivered to the customer
  - *process:* produces the software product
  - *resources:* (both the product and the process require

PROCESS QUALITY VS. PRODUCT QUALITY

- ✓ Quality can mean the difference between excellence and disaster
  - Airbus A400M Atlas crash in 2015, 4 killed

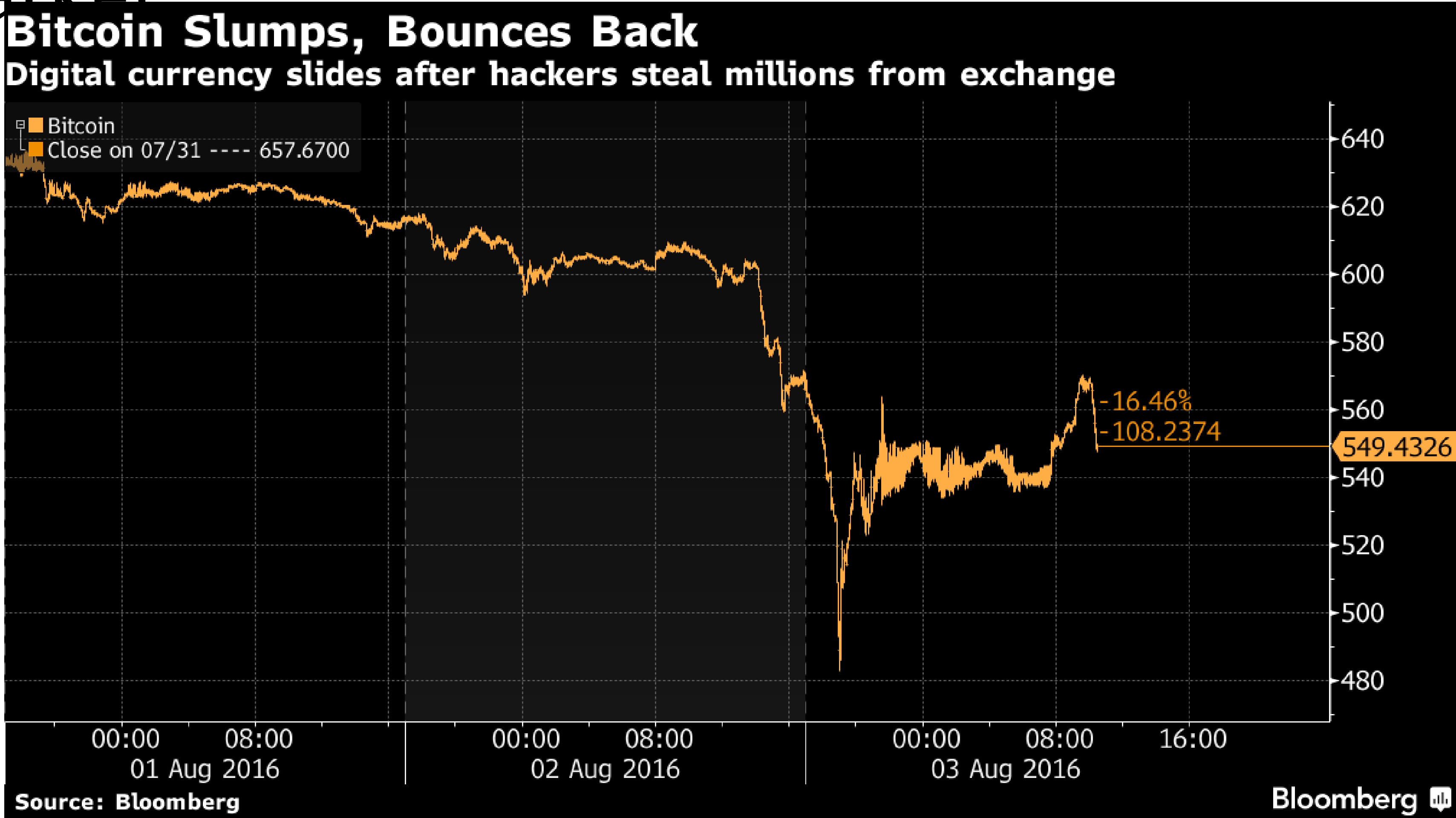
PROCESS QUALITY VS. PRODUCT QUALITY

“*The black boxes attest to that there are no structural defects [with the aircraft], but we have a serious **quality** problem in the final assembly.*”

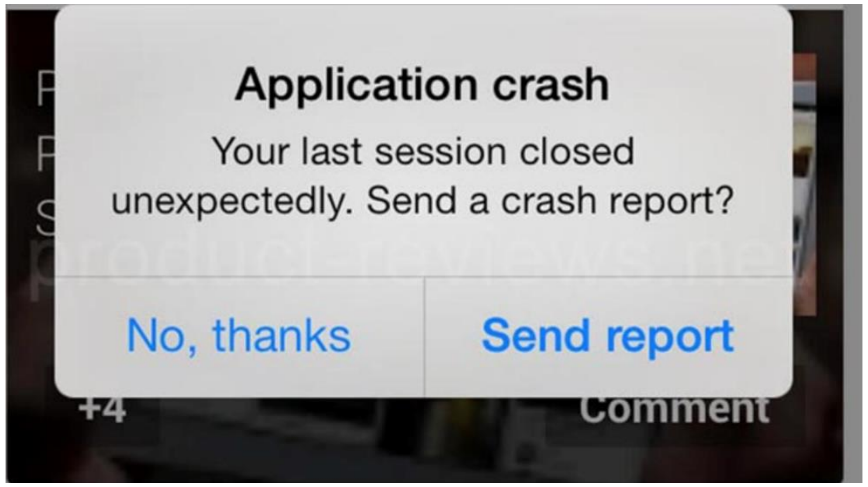
“*…either a weakness in the **test procedure** of planes before they fly, or a problem that results from the implementation of these procedures.*”

PRODUCT OR PROCESS ISSUE?

- ✓ 8/2016: Security breach with Bitcoin cost 72 mil. Usd lost in market

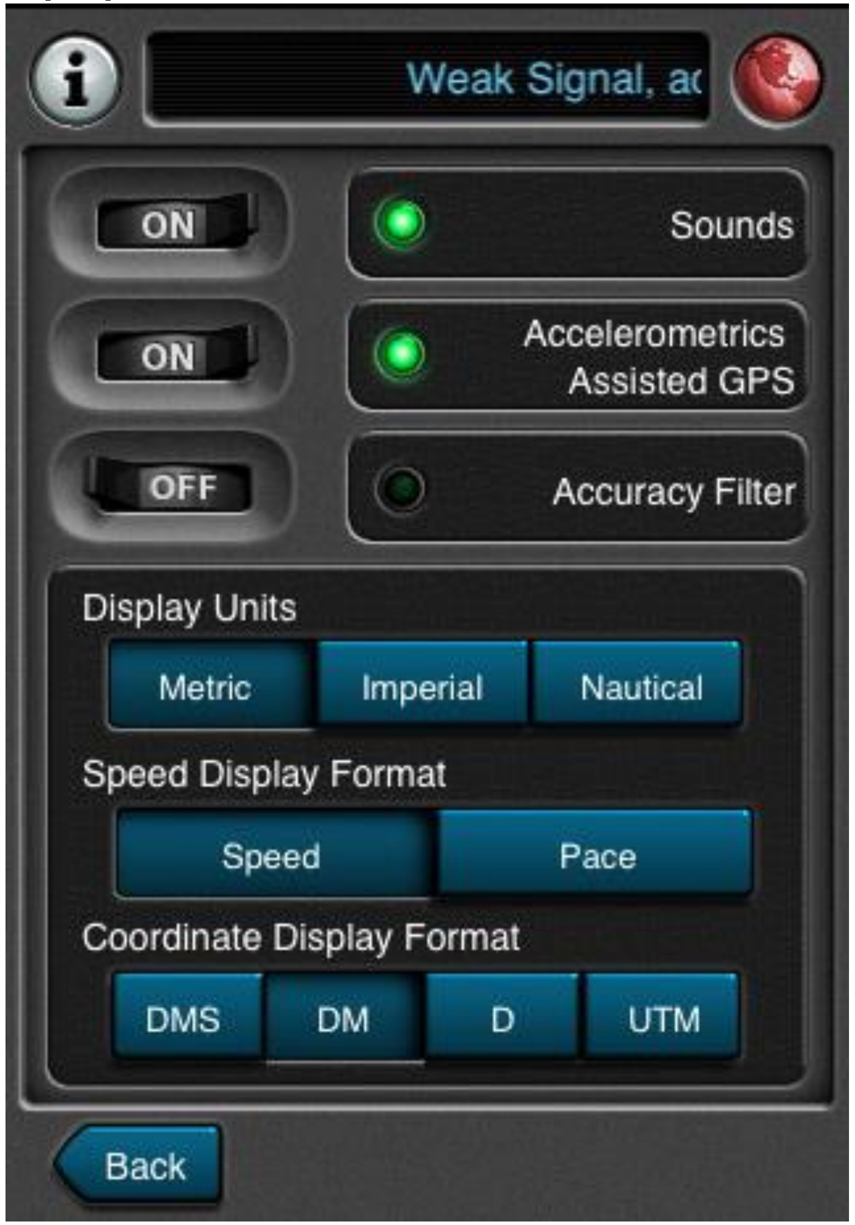


SOFTWARE QUALITY ATTRIBUTES (1)

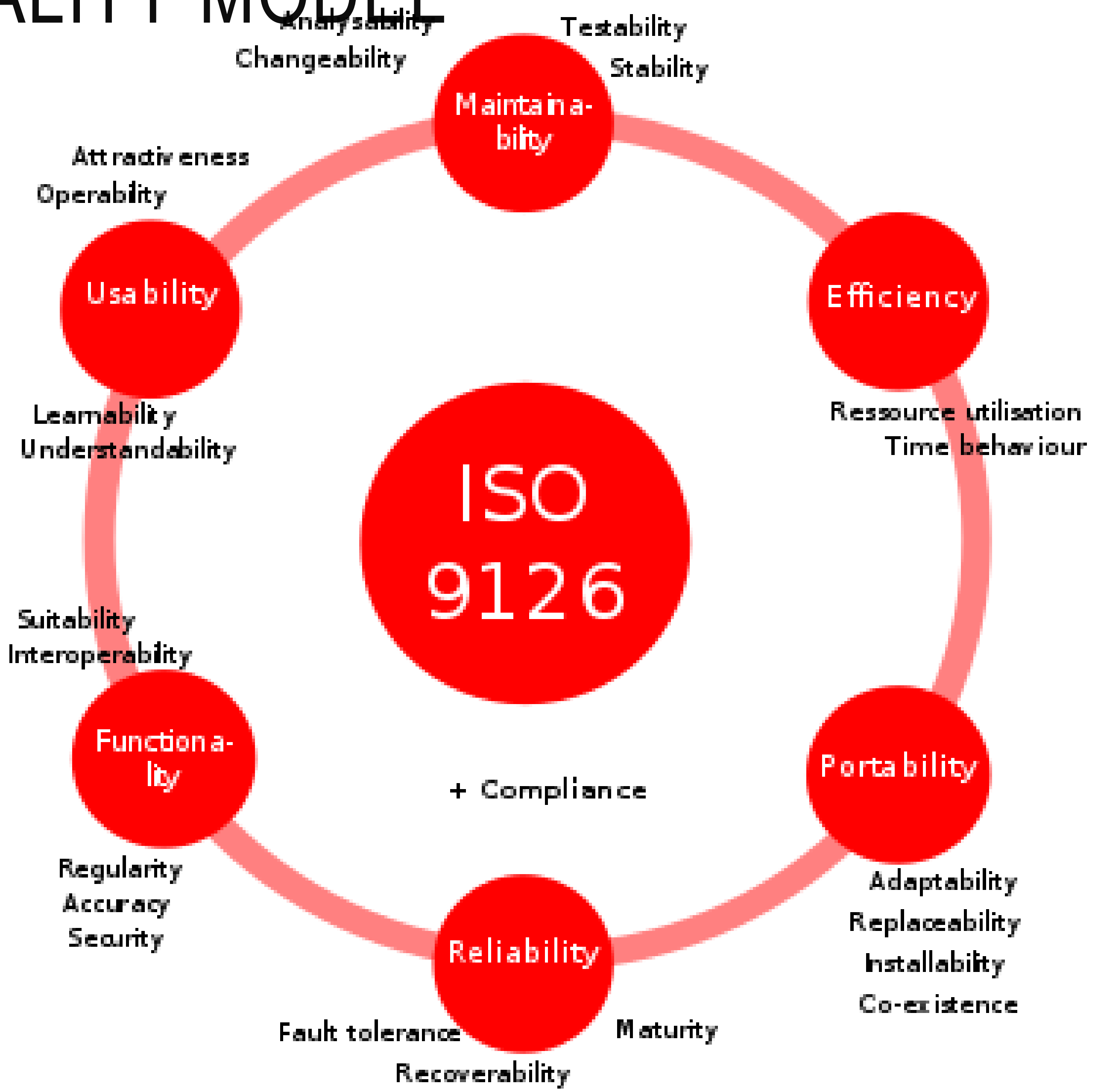


Application crash

Your last session closed unexpectedly. Send a crash report?



SOFTWARE QUALITY MODEL



SOFTWARE TESTING

PROGRAM TESTING

- ✓ Testing is intended to show that a program does what it is intended to do and to discover program defects before it is put into use.
- ✓ **Can reveal the presence of errors NOT their absence.**
- ✓ Testing is part of a more general verification and validation process, which also includes static validation techniques.

PROGRAM TESTING GOALS

- ✓ To demonstrate to the developer and the customer that the software meets its requirements.
  - validation testing
- ✓ To discover situations in which the behavior of the software is incorrect, undesirable or does not conform to its specification.
  - defect testing

QUALITY ASSURANCE

The Product
of Testing
is
CONFIDENCE

PSYCHOLOGY OF TESTING (1)

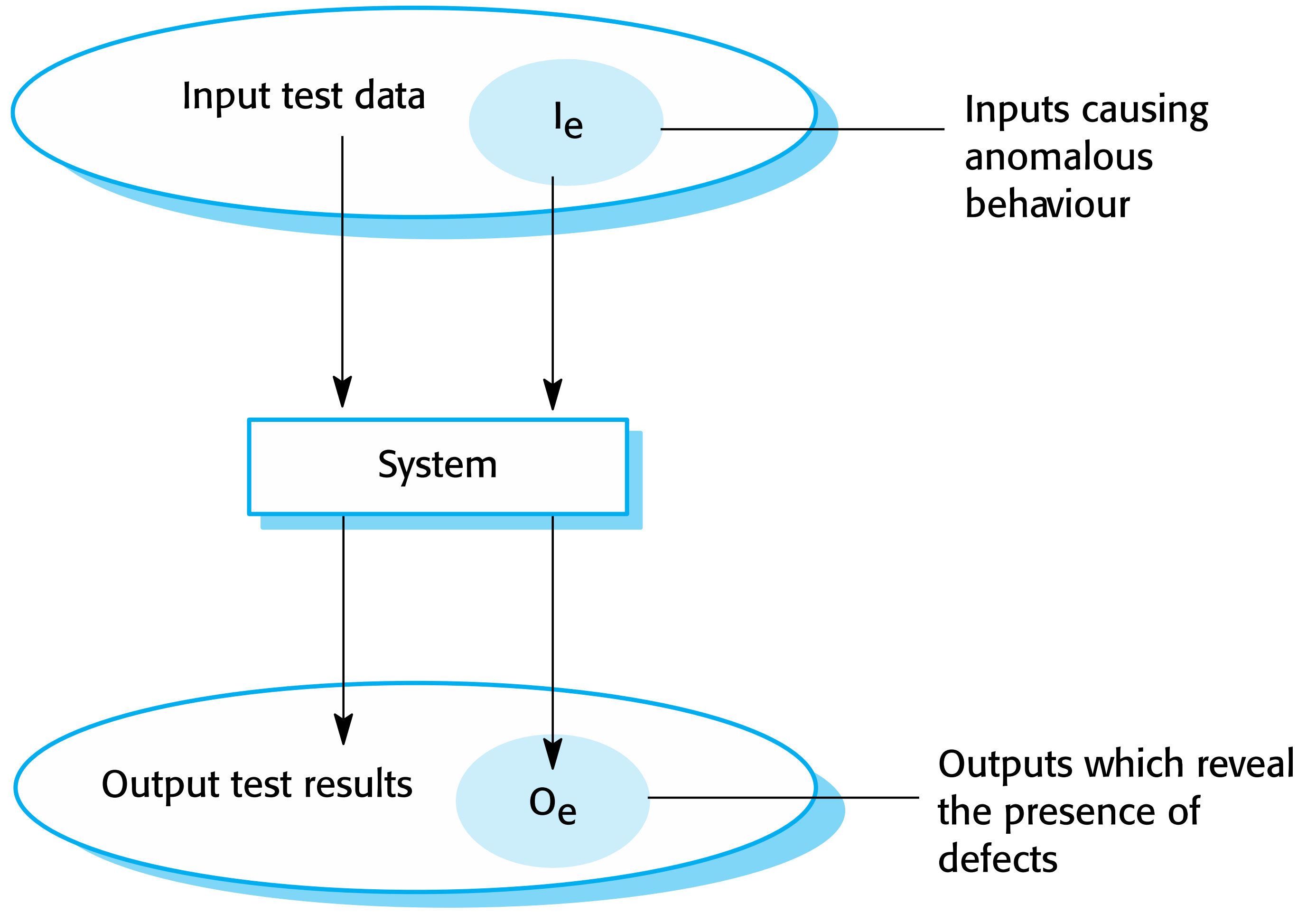
- ✓ A program is its programmer’s baby!
  - Trying to find errors in one’s own program is like trying to find defects in one’s own baby.
  - It is best to have someone other than the programmer doing the testing.
- ✓ Tester must be highly skilled, experienced professional.
- ✓ It helps if he or she possesses a diabolical mind.

PSYCHOLOGY OF TESTING (2)

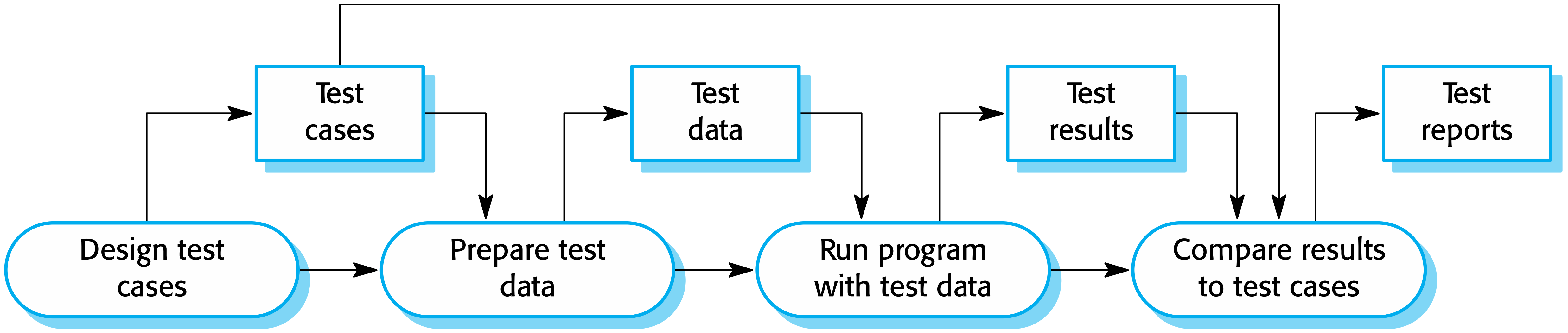
- ✓ Testing achievements depend a lot on what are the goals.
- ✓ Myers says (79):
  - If your goal is to **show absence of errors**, you will not discover many.
  - If you are trying to **show the program correct**, your subconscious will manufacture safe test cases.
  - If your goal is to **show presence of errors**, you will discover large percentage of them.

Testing is the process of executing a program with the intention of finding errors (G. Myers)

AN INPUT-OUTPUT MODEL OF PROGRAM TESTING


Input test data
I_e
Inputs causing anomalous behaviour
System
Output test results
O_e
Outputs which reveal the presence of defects

A MODEL OF THE SOFTWARE TESTING PROCESS



STAGES OF TESTING

- ✓ Development testing
  - the system is tested during development to discover bugs and defects.
- ✓ Release testing
  - a separate testing team test a complete version of the system before it is released to users.
- ✓ User testing
  - users or potential users of a system test the system in their own environment.

DEVELOPMENT TESTING

carried out by the team developing the system.

- ✓ Unit testing:
  - for individual program units or object classes
  - focus on testing the functionality of objects or methods.
- ✓ Component testing:
  - several individual units are integrated to create composite components
  - focus on testing component interfaces.
- ✓ System testing:
  - some or all of the components in a system are integrated and the system is tested as a whole
  - focus on testing component interactions.

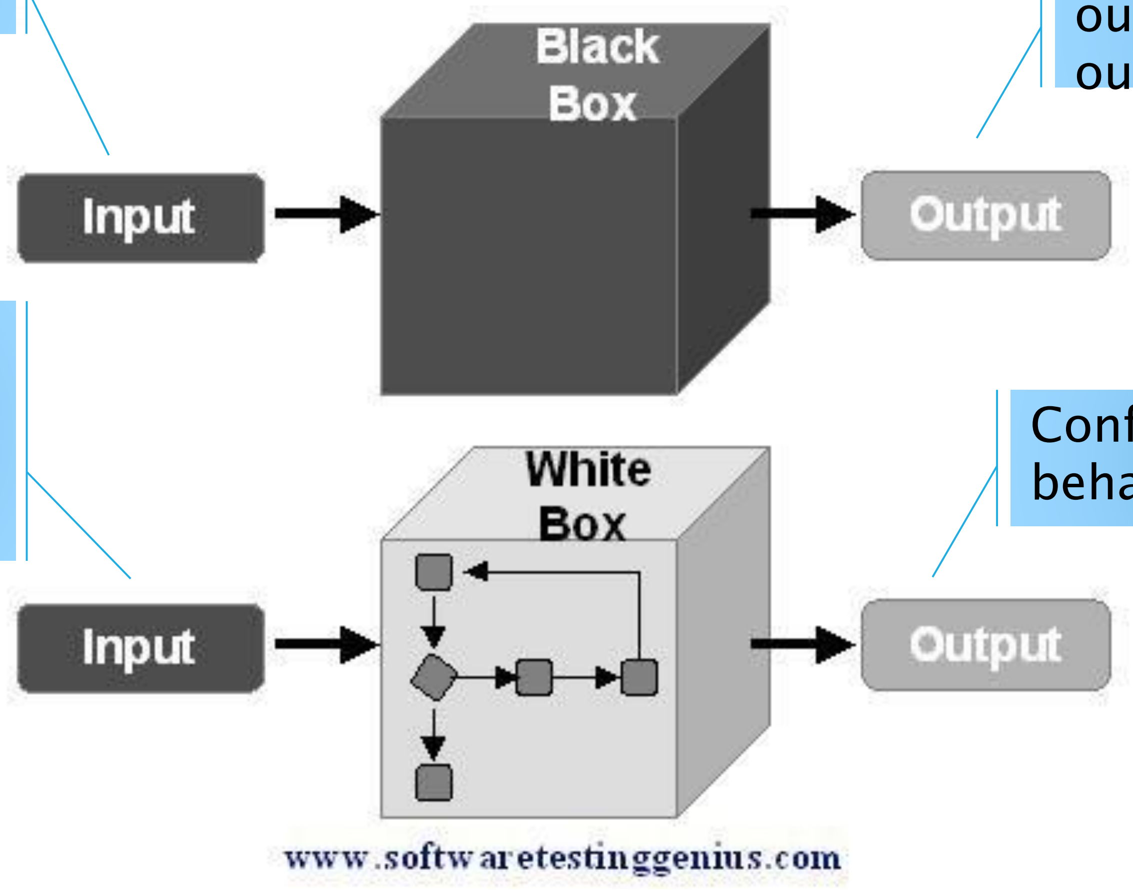
UNIT TESTING

- ✓ Unit testing is the process of testing individual components in isolation.
- ✓ It is a defect testing process.
- ✓ Units may be:
  - Individual functions or methods within an object
  - Object classes with several attributes and methods
  - Composite components with defined interfaces used to access their functionality.

UNIT TESTING: BLACK-/WHITE-BOX TEST

from requirement s

Compare actual output with require output



from requirement s & key design elements

Confirm expected behaviour

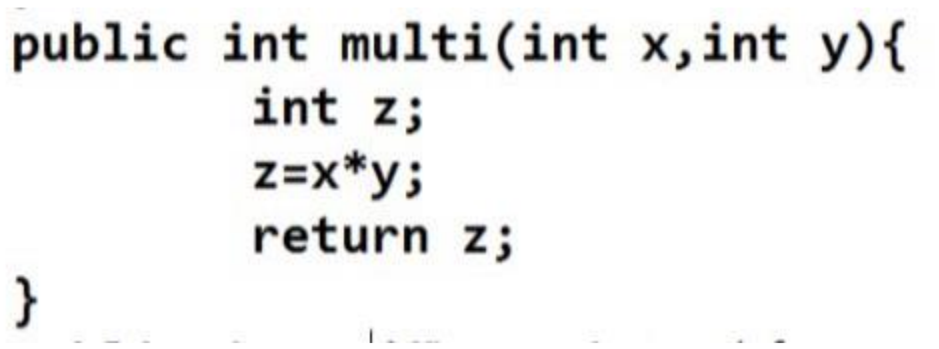
Gray-box: mix of black- and white-box testing

```
for (i=0; i<numrows; i++)
  for (j=0; j<numcols; j++);
    pixels++;
```

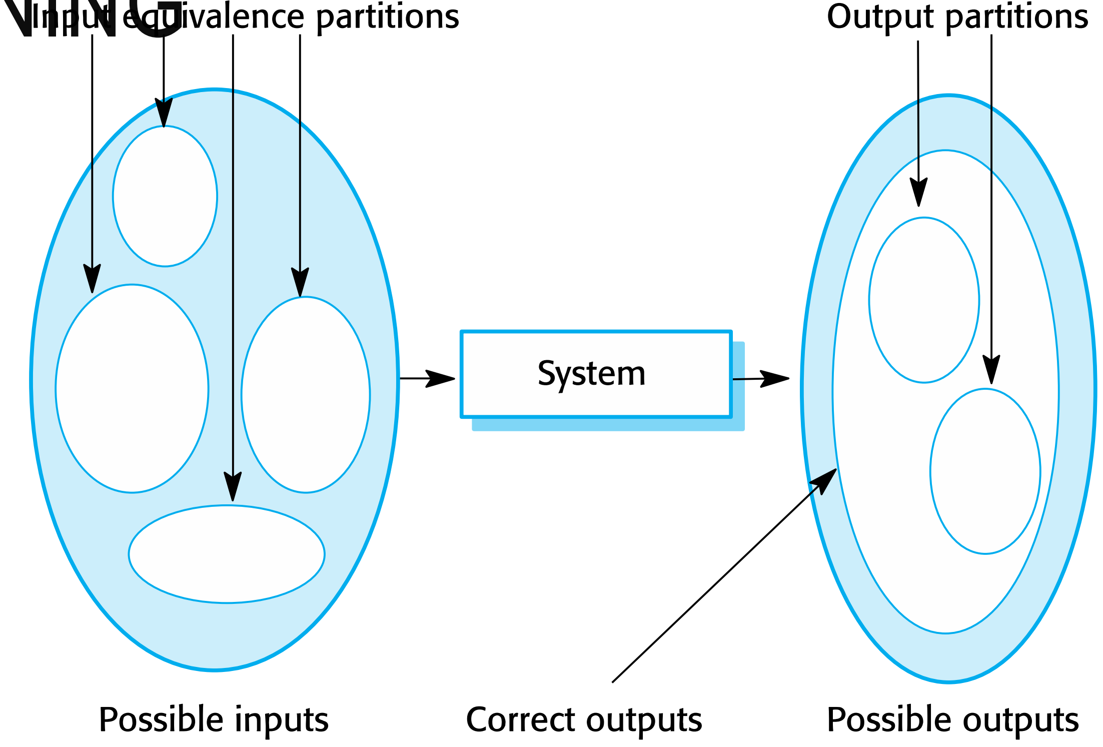
```
int minval(int *A, int n) {
  int currmin;

  for (int i=0; i<n; i++)
    if (A[i] < currmin)
      currmin = A[i];
  return currmin;
}
```

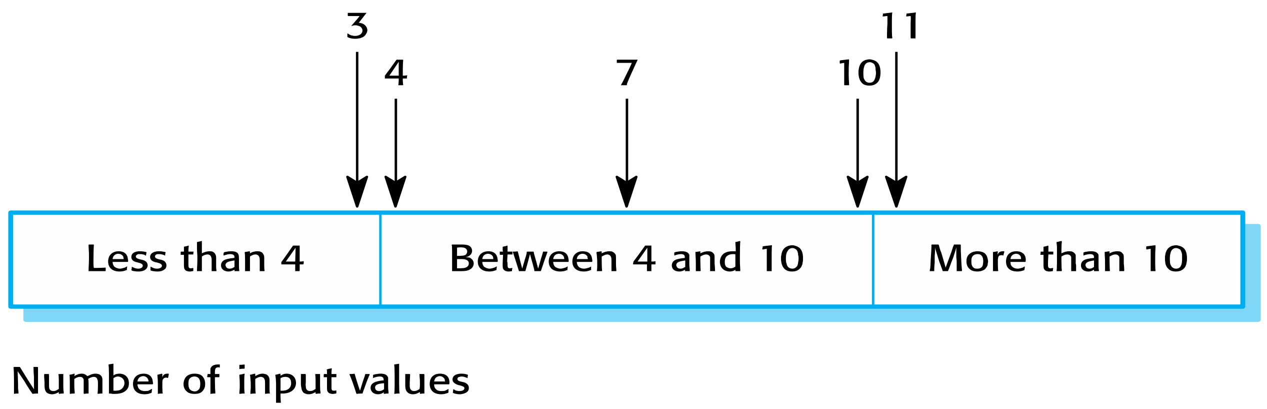
```
switch (i) {
  case 1:
    do_something(1); break;
  case 2:
    do_something(2); break;
  case 3:
    do_something(1); break;
  case 4:
    do_something(4); break;
  default:
    break;
}
```

EQUIVALENCE PARTITIONING

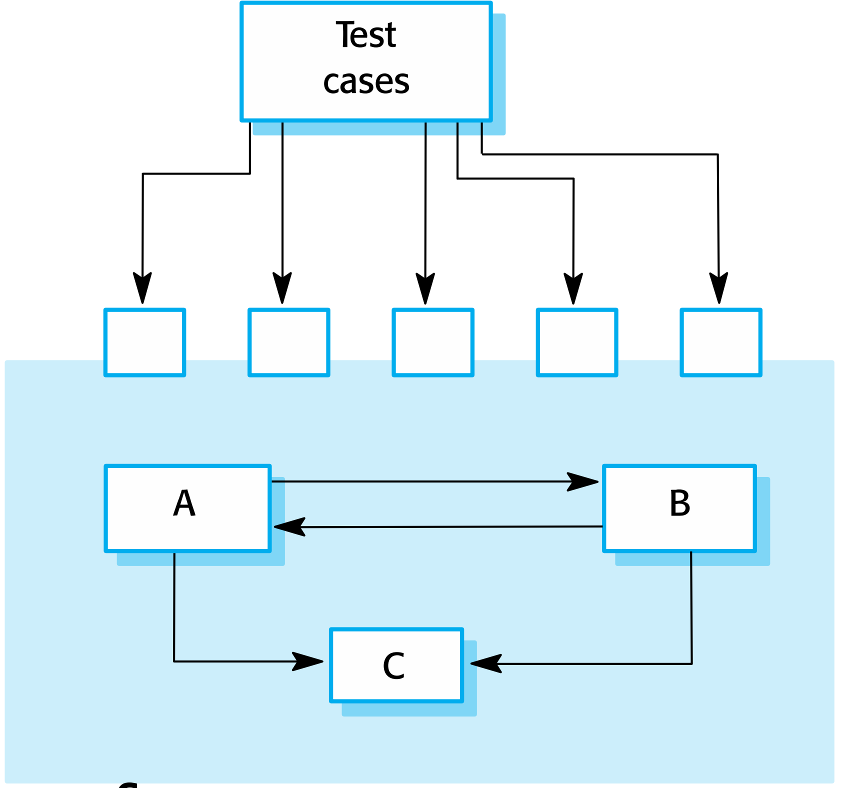


```
public int multi(int x,int y){
        int z;
        z=x*y;
        return z;
}
```





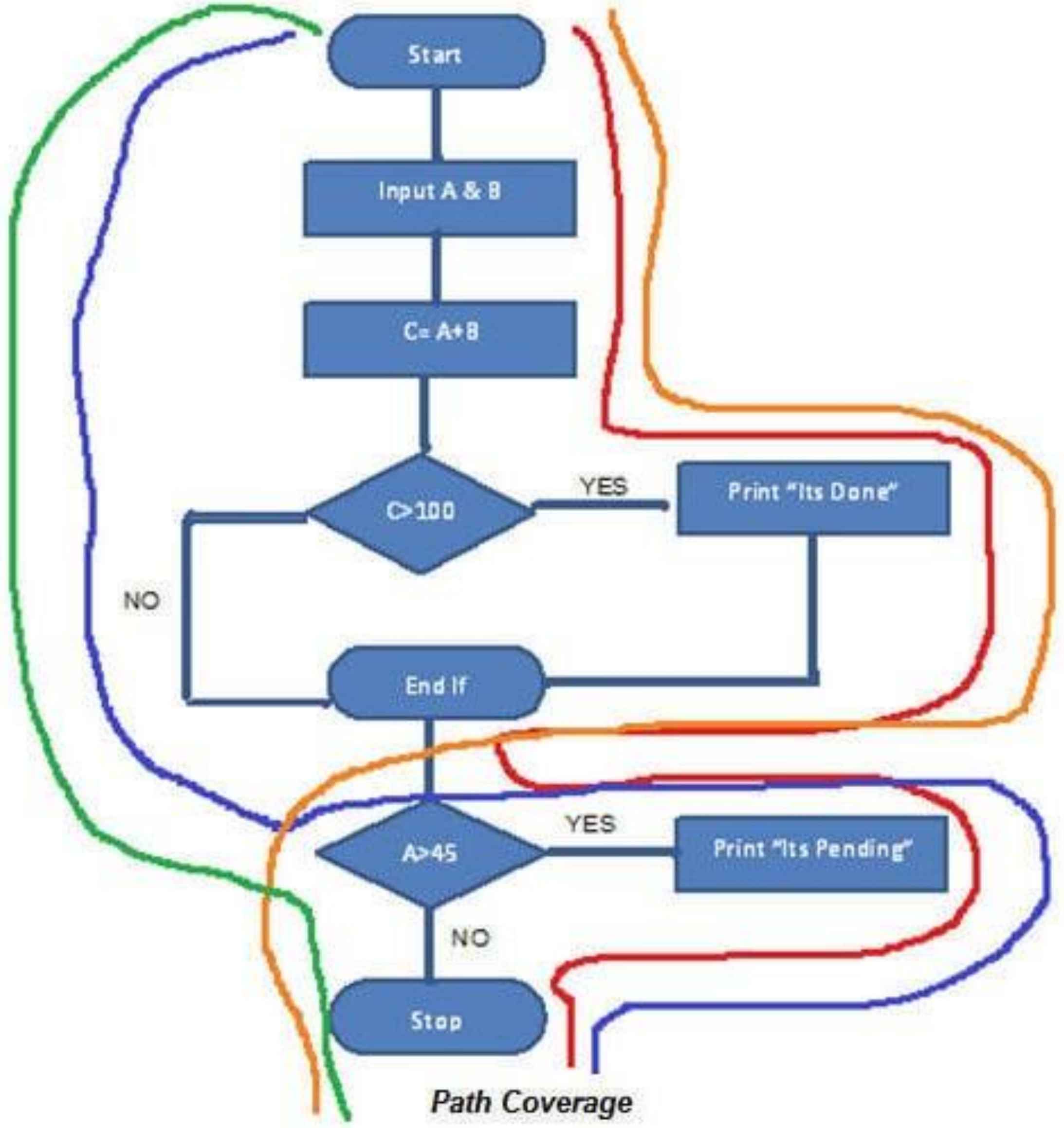
INTERFACE TESTING



- ✓ Detect faults due to
  - interface errors
  - or invalid assumptions about interfaces.
- ✓ Interface types
  - Parameter interfaces
  - Shared memory interfaces
  - Procedural interfaces
  - Message passing interfaces

WHITE-BOX TESTING

- ✓ Statement coverage
- ✓ Branch coverage
- ✓ Path coverage



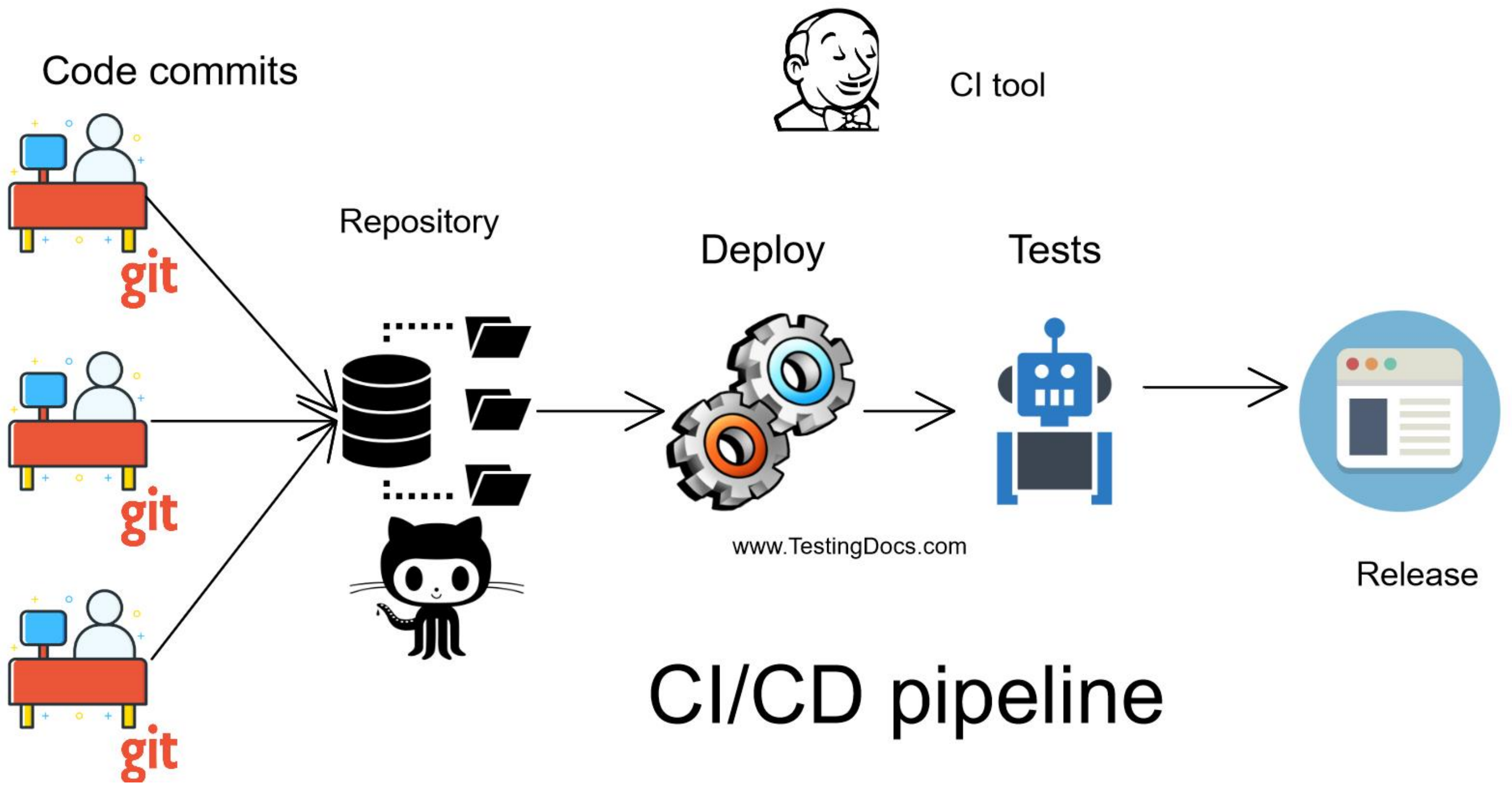
REGRESSION TESTING

Test the system to check that changes have not 'broken' previously working code.

- ✓ Better with automated testing
- ✓ All tests are re-run every time a change is made to the program.
- ✓ Tests must run 'successfully' before the change is committed.

AUTOMATED TESTING

- ✓ Whenever possible, unit testing should be automated
- ✓ Us t)



SYSTEM TESTING

System testing during development = to create a version of the system and then testing the integrated system.

- ✓ Focus on testing the interactions between components.
  - System testing checks that components are compatible, interact correctly and transfer the right data at the right time across their interfaces.

- ✓ And tests the emergent behaviour of a system.

TYPES OF SYSTEM TESTS

- ✓ Volume
  - Subject product to large amounts of input.
- ✓ Usability
  - Measure user reaction (e.g., score 1-10).
- ✓ Performance
  - Measure speed under various circumstances.
- ✓ Configuration
  - Configure to various hardware / software
- ✓ Compatibility
  - with other designated applications
- ✓ Reliability / Availability
  - Measure up-time over extended period.
- ✓ Security
  - Subject to compromise attempts.
- ✓ Resource usage
  - Measure usage of RAM and disk space etc.
- ✓ Install-ability
  - Install under various circumstances.
- ✓ Recoverability
  - Force activities that take the application down.
- ✓ Serviceability
  - Service application under various situations.
- ✓ Load / Stress
  - Subject to extreme data & event traffic

USE-CASE TESTING – TEST CASE

Test case	SUC3 Correct PIN entry on first try
Test description	A customer enters the PIN number correctly on the first attempt.
Related screens	
Pre-conditions	1. The expected PIN is known 2. Screen 2 is displayed
Actions	1. Screen 2 shows ‘– – – – ’ 2. Customer touches 1st digit 3. Screen 2 shows ‘– – – * ’ 4. Customer touches 2nd digit 5. Screen 2 shows ‘– – * * ’ 6. Customer touches 3rd digit 7. Screen 2 shows ‘– * * * ’ 8. Customer touches 4th digit 9. Screen 2 shows ‘* * * * ’ 10. Customer touches Enter 11. Screen 5 is displayed
Inputs	Correct PIN number: 1357
Expected Outputs	Select Transaction screen is active
Testing environment	Software interface run in Windows 10

USE-CASE TESTING

The use-cases developed to identify system interactions can be used as a basis for system testing.

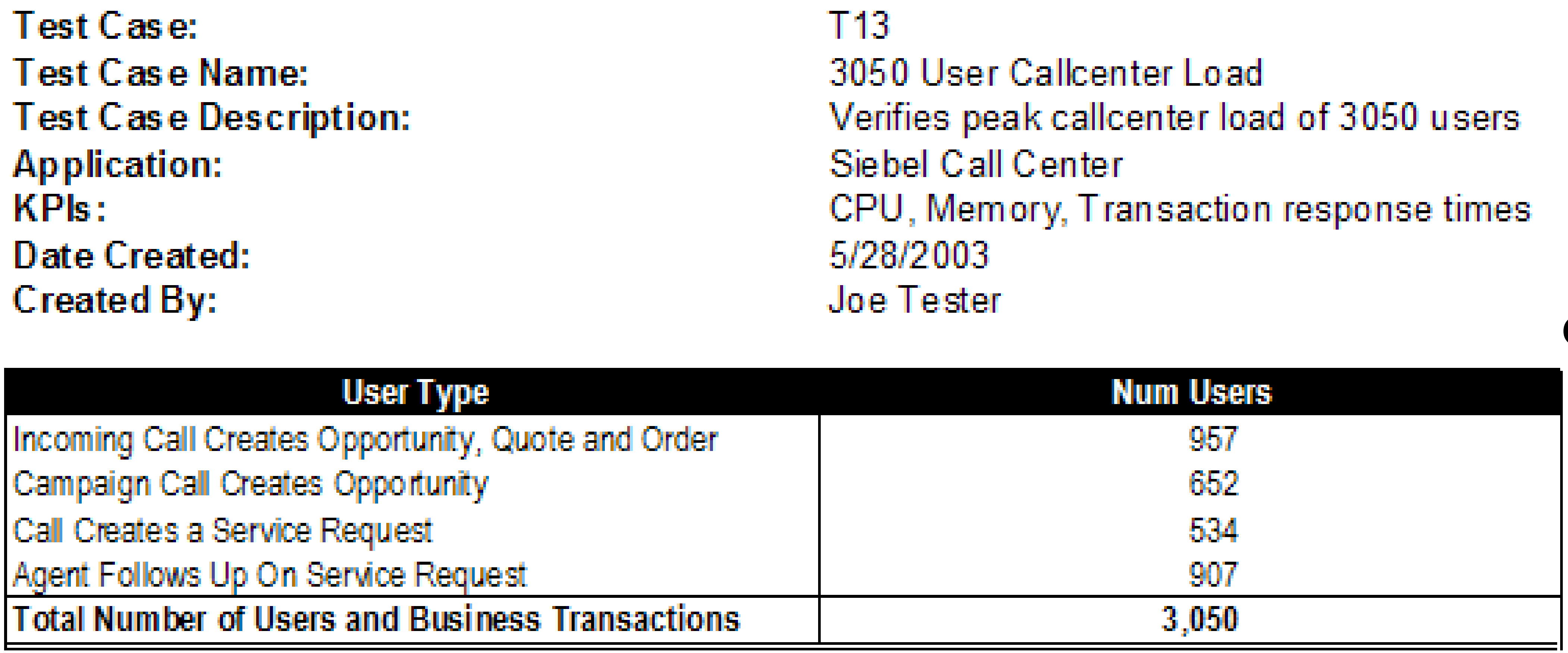
- ✓ Each use case usually involves several system components so testing the use case forces these interactions to occur.
  - The sequence diagrams associated with the use case documents the components and interactions that are being tested.

PERFORMANCE TESTING

Part of rele
the emerg
as perform

- ✓ Tests at the system level
- ✓ Tests should reflect the profile of use of the system.
- ✓ Is usually a series of tests
  - the load is steadily increased until the system performance becomes unacceptable.
- ✓ Stress testing
  - is a form of performance testing where the system is deliberately overloaded to test its failure behaviour.

PERFORMANCE TESTING – TEST CASE



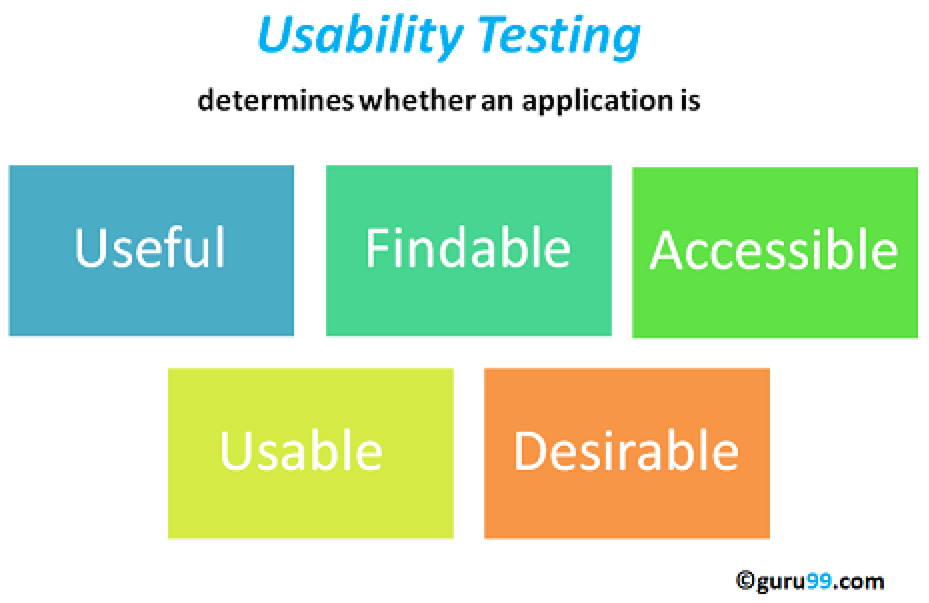
Test Case:	T13
Test Case Name:	3050 User Callcenter Load
Test Case Description:	Verifies peak callcenter load of 3050 users
Application:	Siebel Call Center
KPIs:	CPU, Memory, Transaction response times
Date Created:	5/28/2003
Created By:	Joe Tester

e

User Type	Num Users
Incoming Call Creates Opportunity, Quote and Order	957
Campaign Call Creates Opportunity	652
Call Creates a Service Request	534
Agent Follows Up On Service Request	907
Total Number of Users and Business Transactions	**3,050**

❑ USABILITY TESTING

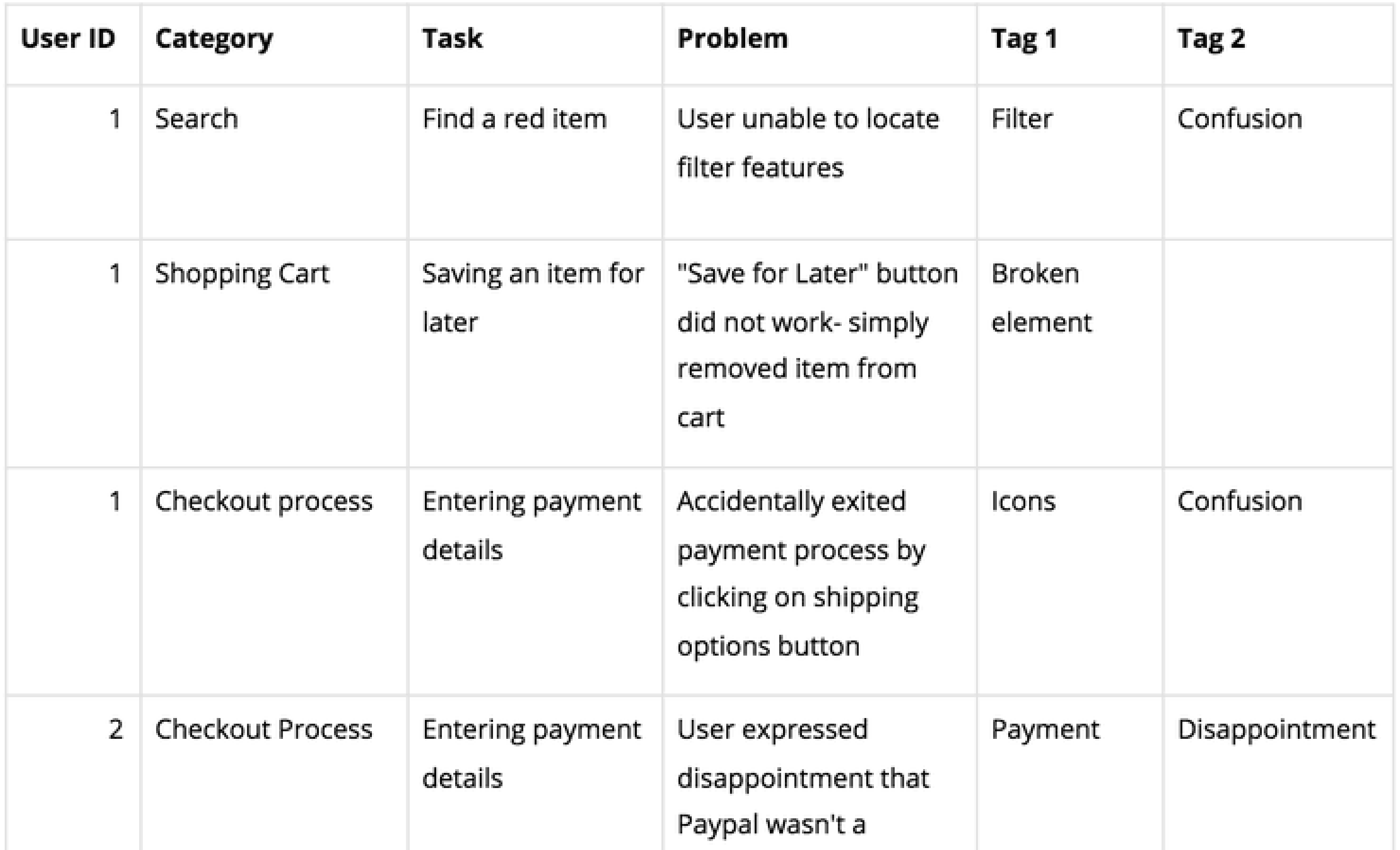
✓ Evaluating the usability aspect of the software by testing it with representative users



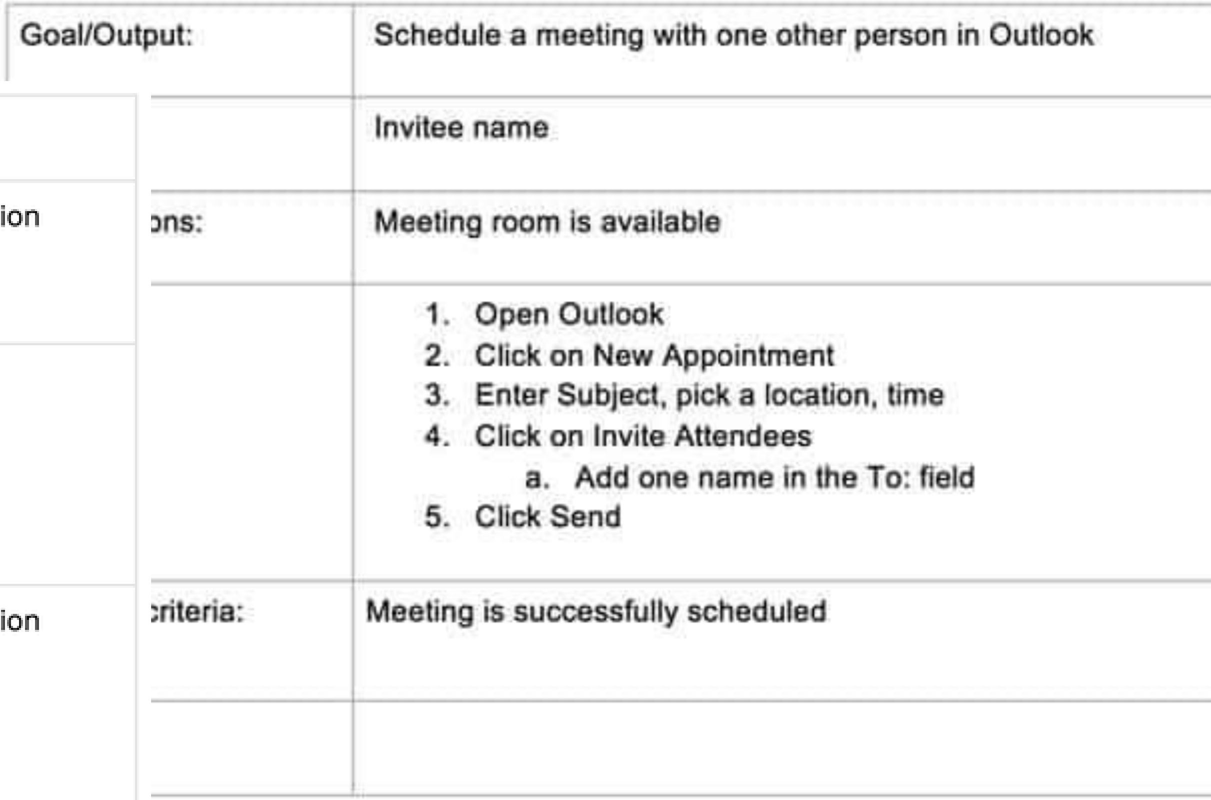
- ❖ Where do I click next?
- ❖ Which page needs to be navigated?
- ❖ Which Icon or Jargon represents what?
- ❖ Error messages are not consistent or effectively displayed
- ❖ Session time not sufficient

❑ USABILITY TESTING – TEST CASE

- ✓ Evaluating the usability aspect of the software by testing it with representat



User ID	Category	Task	Problem	Tag 1	Tag 2
1	Search	Find a red item	User unable to locate filter features	Filter	Confusion
1	Shopping Cart	Saving an item for later	"Save for Later" button did not work- simply removed item from cart	Broken element	
1	Checkout process	Entering payment details	Accidentally exited payment process by clicking on shipping options button	Icons	Confusion
2	Checkout Process	Entering payment details	User expressed disappointment that Paypal wasn't a	Payment	Disappointment



Goal/Output:	Schedule a meeting with one other person in Outlook
	Invitee name
ons:	Meeting room is available
	1. Open Outlook 2. Click on New Appointment 3. Enter Subject, pick a location, time 4. Click on Invite Attendees a. Add one name in the To: field 5. Click Send
criteria:	Meeting is successfully scheduled

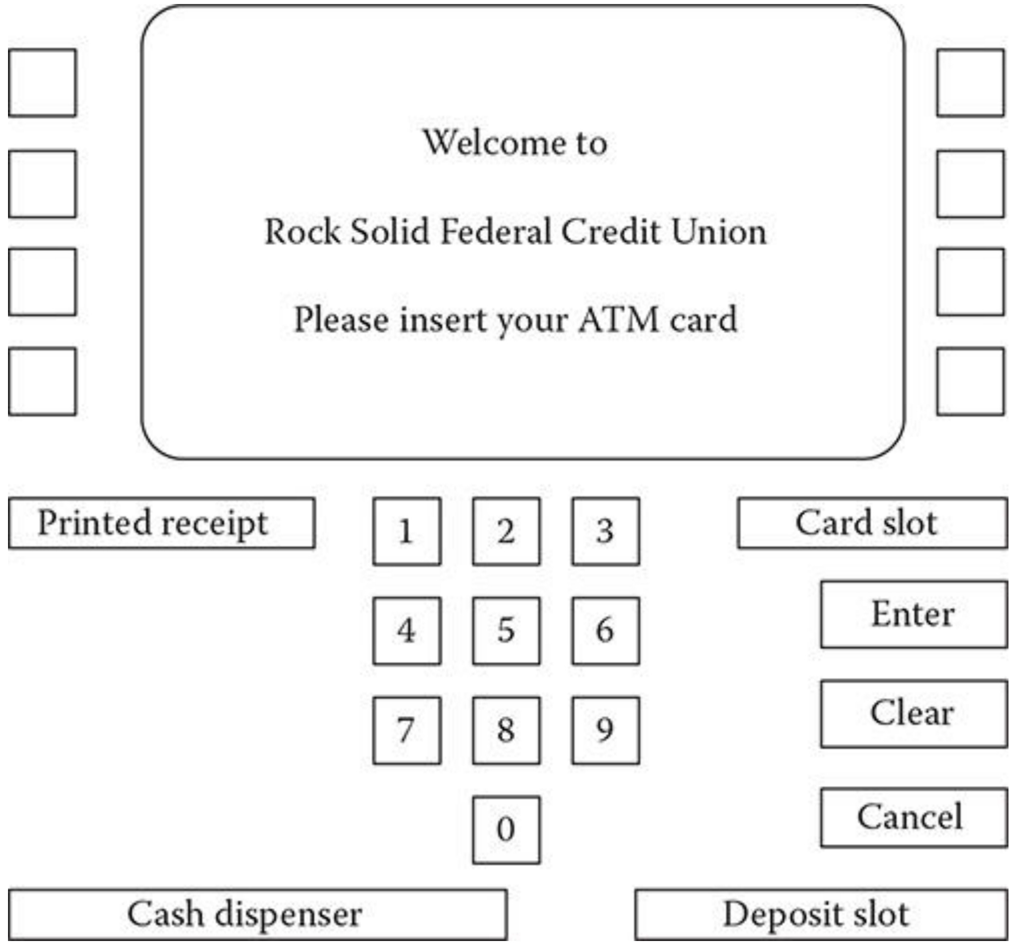
TEST PLANNING

TEST PLAN

- ✓ Testing requires meticulous planning on an operational level and on a strategic level
- ✓ The test manager needs to transpose such generic guidelines to create a concrete testing strategy for the project at hand
- ✓ Test plan should state:
  - Test object: which components, modules, neighboring systems, and interfaces (will) make up the system to be tested
  - test objectives: specific testing objectives and the criteria you are testing against for each test object and the entire system
  - Customize the testing process
  - testing methods and techniques: overall testing approach and the testing techniques
  - required infrastructure
  - Success/ failure criteria: test metrics and threshold

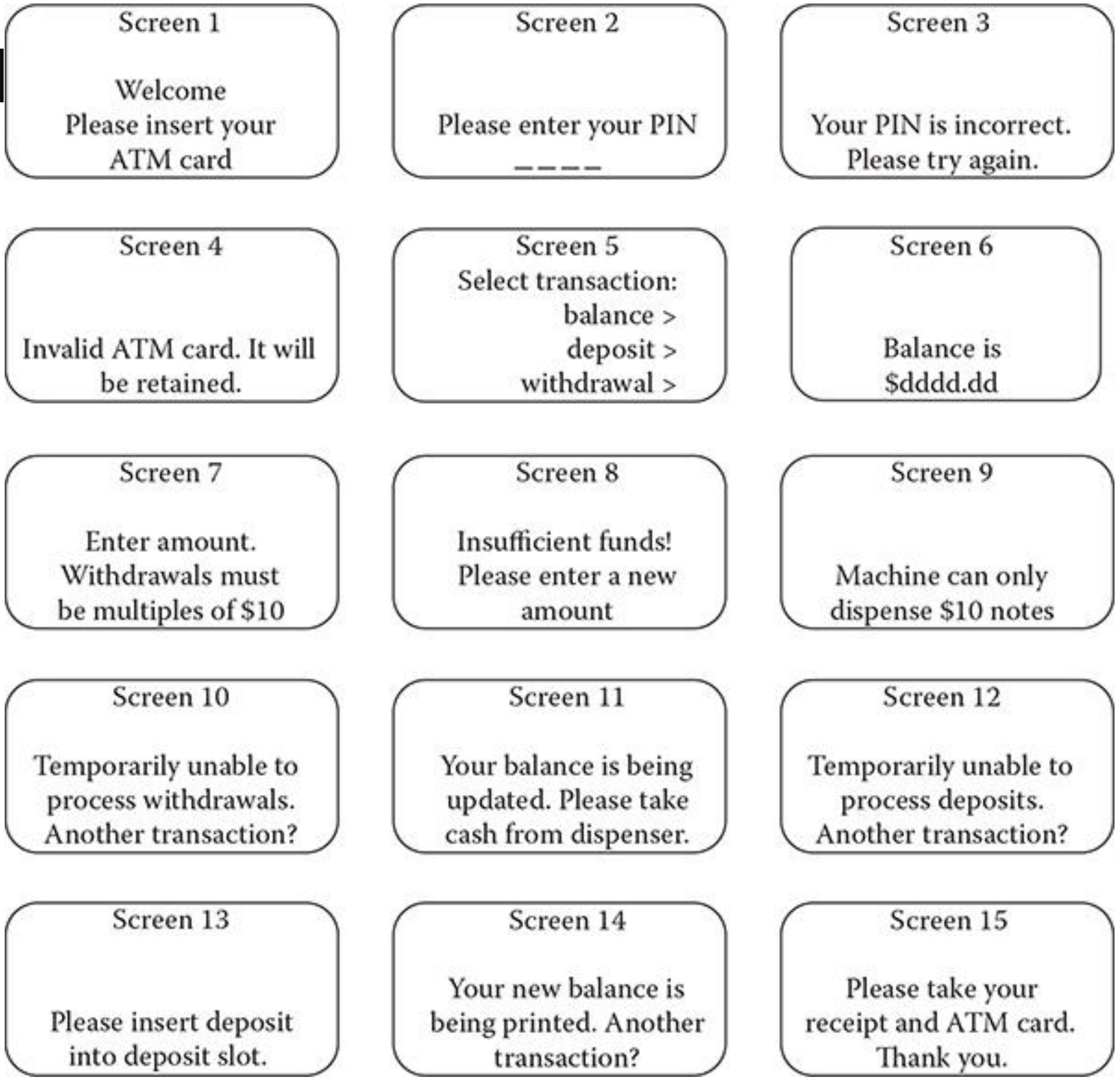
A CASE – ATM TERMINAL SOFTWARE TESTING

- ✓ A simple version of ATM machine
- ✓ Insert your card, type PIN code and withdraw cash from the machine



A CASE – ATM TERMIN

- ✓ List of designed screens



Screen 1

Welcome
Please insert your ATM card

Screen 2

Please enter your PIN
_ _ _ _

Screen 3

Your PIN is incorrect.
Please try again.

Screen 4

Invalid ATM card. It will be retained.

Screen 5

Select transaction:
balance >
deposit >
withdrawal >

Screen 6

Balance is
$dddd.dd

Screen 7

Enter amount.
Withdrawals must be multiples of $10

Screen 8

Insufficient funds!
Please enter a new amount

Screen 9

Machine can only dispense $10 notes

Screen 10

Temporarily unable to process withdrawals.
Another transaction?

Screen 11

Your balance is being updated. Please take cash from dispenser.

Screen 12

Temporarily unable to process deposits.
Another transaction?

Screen 13

Please insert deposit into deposit slot.

Screen 14

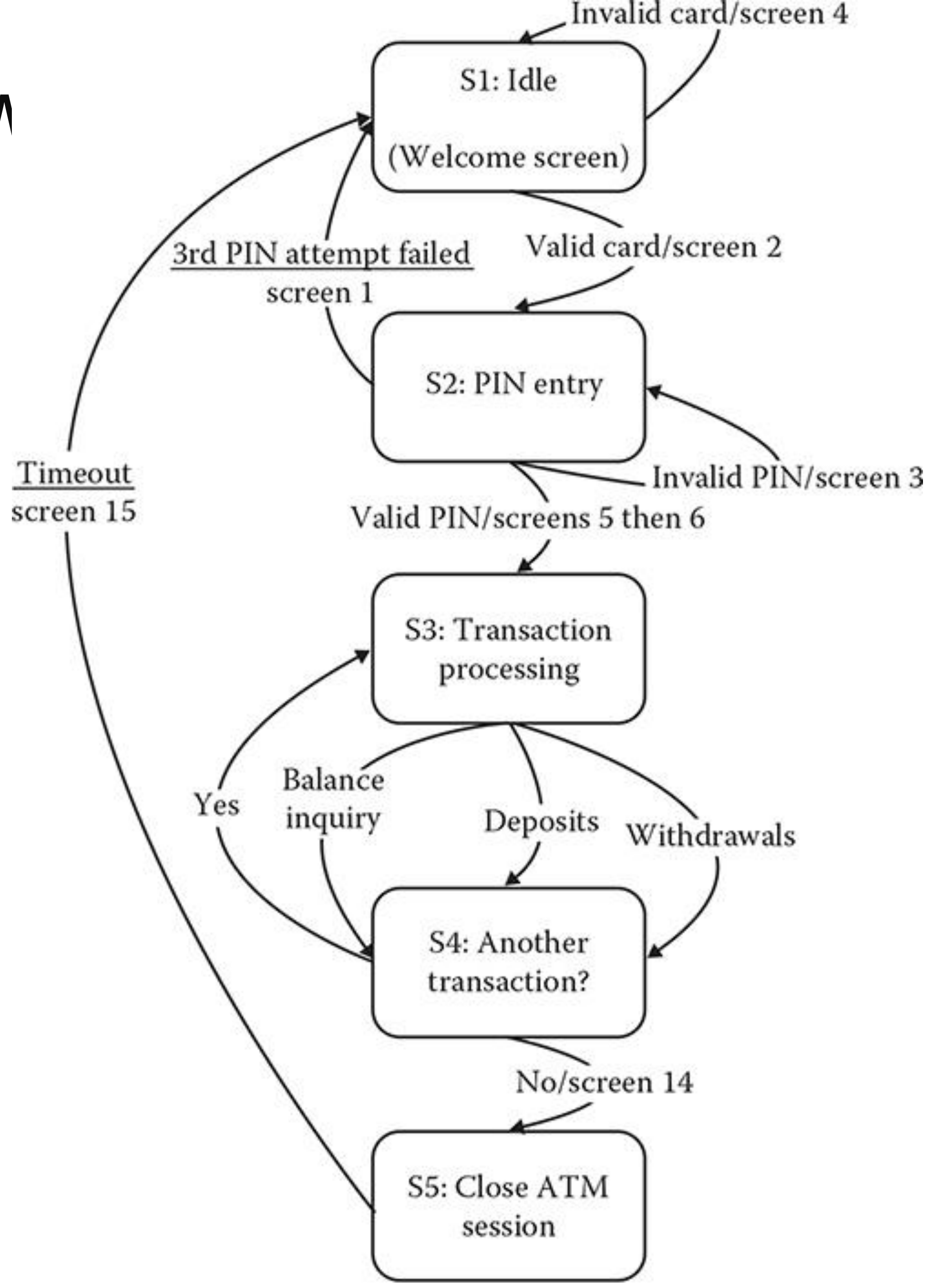
Your new balance is being printed. Another transaction?

Screen 15

Please take your receipt and ATM card.
Thank you.

A CASE – ATM TERMINAL SOFTW

- ✓ State diagram from the Idle state to the Closed ATM session



NOW ABOUT TESTING …

- ✓ We are developing and integrating all of these modules/ screens
- ✓ Which test levels we can do`?
- ✓ What types of tests we can do?

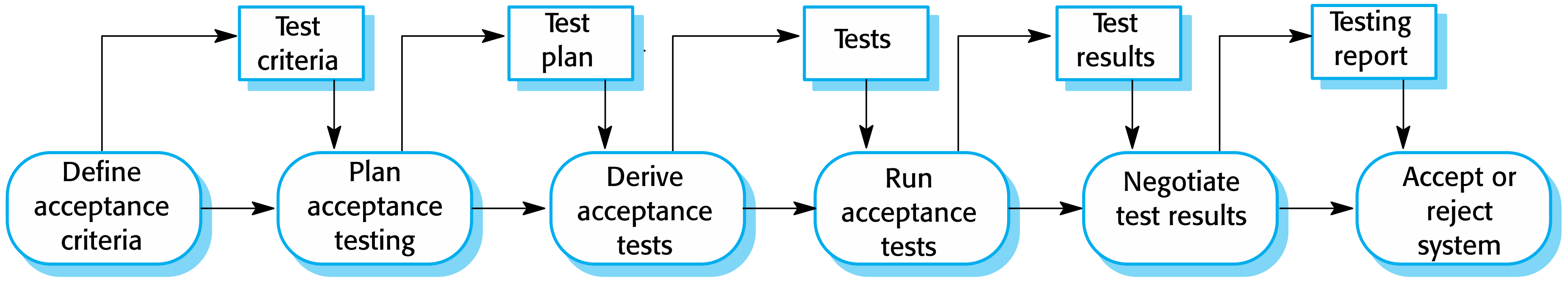
TEST PLAN - EXAMPLE

Type of tests	Description	Number of tests	Test types	Test data	Test metric
Unit test	Testing each screen to make sure each screen displays according to its specification	At least 15 assertation: S1 – S15	Whitebox blackbox	Pin code, Amount to withdraw	Path coverage Line coverage Branch coverage
System test/ Use case test	Testing possible different use case scenarios with successful and unsuccessful ATM transactions	At least 25 test cases as below	Blackbox	Pin code, Amount to withdraw	Requirement coverage
Usability test	Evaluate the user experience with the ATM interfaces with the two scenarios with successful and unsuccessful ATM transactions	At least 2 test cases	Blackbox	Tasks to perform Pin code, Amount to withdraw	Usability score
Performance test	Simulating POS system's transaction processing under extreme conditions Internet-based transactions	1 test with 100 transactions and another test with 1000 transactions	Blackbox	Number of transactions	Transaction time Delayed time

USER ACCEPTANCE TEST (UAT)

- ✓ acceptance testing is a test conducted to determine if the requirements of a specification or contract are met
- ✓ It can look like a system test … with customer involvement
- ✓ It is blackbox testing
- ✓ It may involve performance test, stress test, usability test, etc
- ✓ Testers should be given real-life scenarios such as the three most common or difficult tasks that the users they represent will undertake

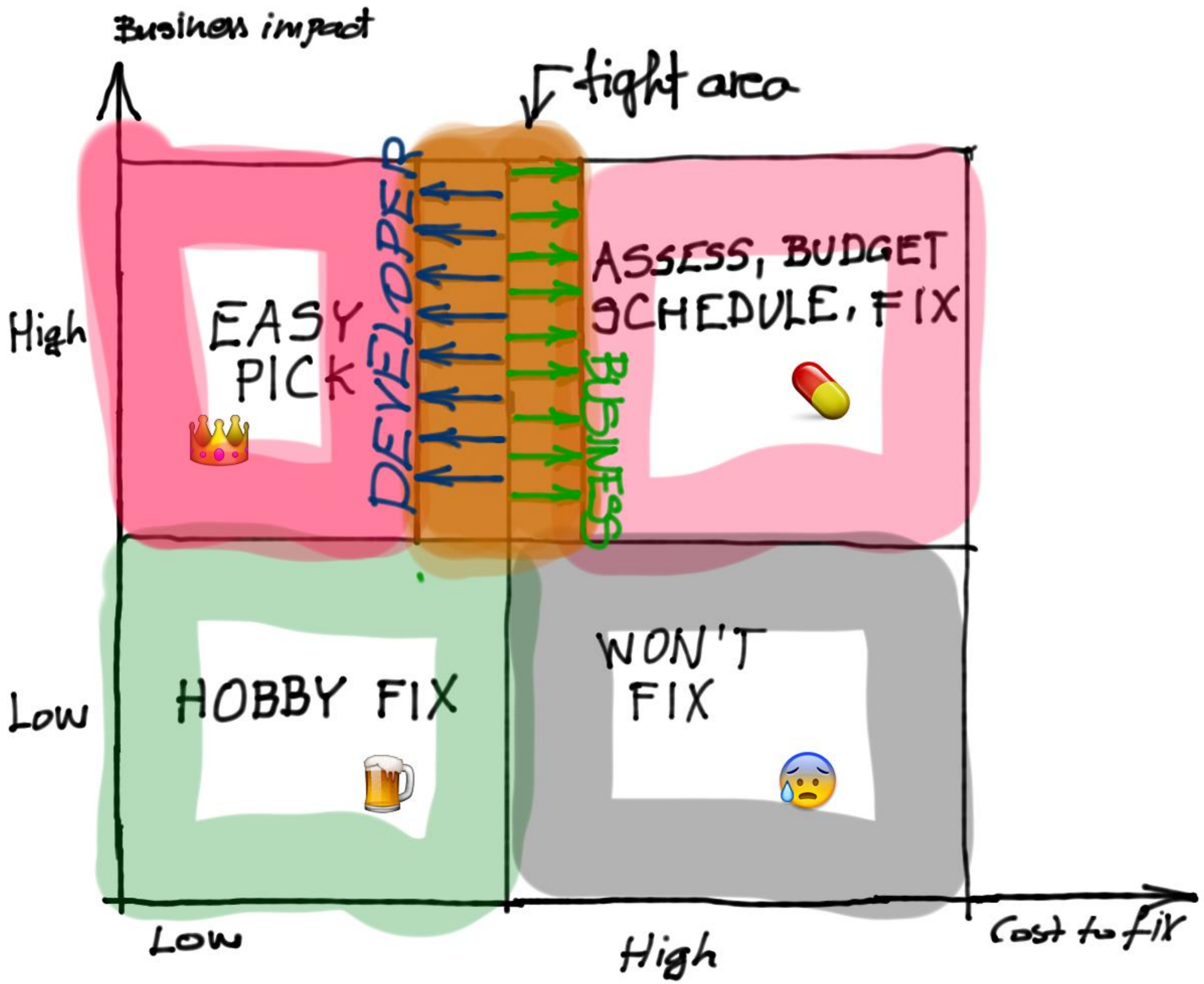
STAGES IN THE ACCEPTANCE TESTING PROCESS



- ✓ Define acceptance criteria
- ✓ Plan acceptance testing
- ✓ Derive acceptance tests
- ✓ Run acceptance tests
- ✓ Negotiate test results
- ✓ Reject/accept system

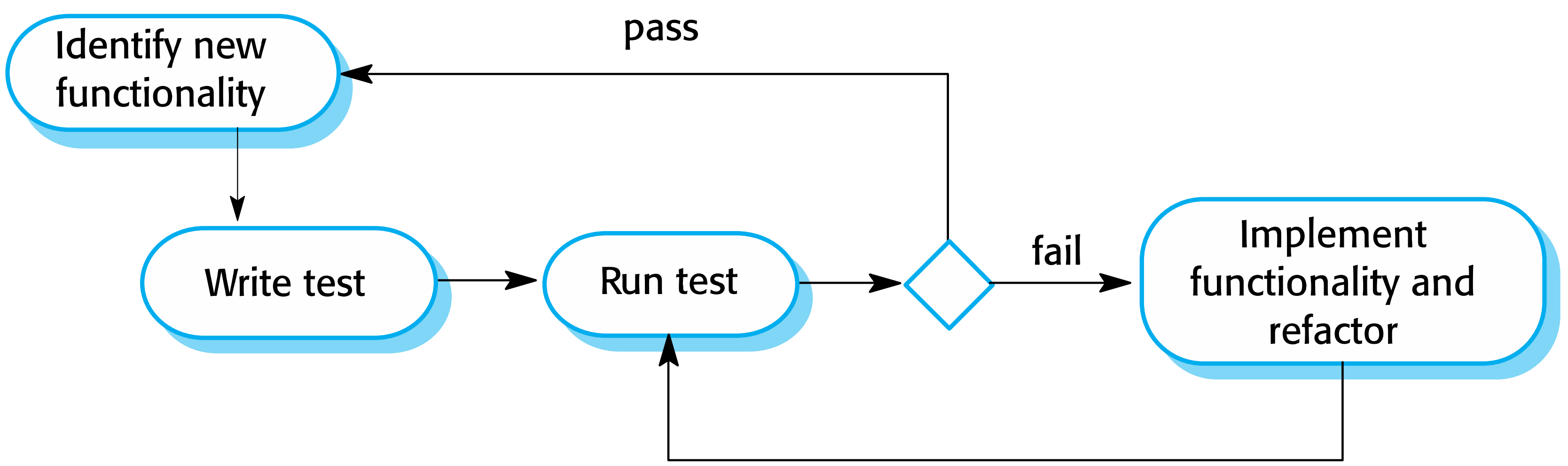
TEST DRIVEN DEVELOPMENT

BUG TRIAGING



TEST-DRIVEN DEVELOPMENT

inter-leave testing and code development


pass
Identify new functionality
Write test
Run test
fail
Implement functionality and refactor

Benefits of test-driven development

- Code coverage
- Regression testing
- Simplified debugging
- System documentation