



SOFTWARE ENGINEERING

Software Testing

Dr. Rathachai Chawuthai

Department of Computer Engineering

Faculty of Engineering

King Mongkut's Institute of Technology Ladkrabang

Agenda

- 
- Introduction
 - Software Testing
 - Test Documentation

Introduction



SOFTWARE ENGINEERING

Software Testing

Tester → ❶ Bug, ❷ Dev coding ❸ ยังมีคุณภาพ

Dr. Rathachai Chawuthai

Department of Computer Engineering

Faculty of Engineering

King Mongkut's Institute of Technology Ladkrabang

Agenda

- 
- Introduction
 - Software Testing
 - Test Documentation

Introduction



Discussion

ลูกค้า - จ้างทำระบบ

ผู้ใช้ - คนใช้ระบบ

ทำไมต้องมี Software Testing ?

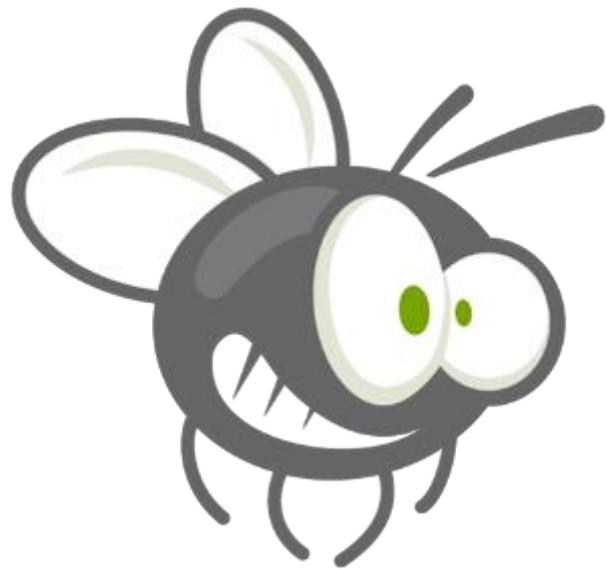
Ans ให้ SW มีคุณภาพ

[ตามลูกค้า → functional
ตามที่ควรจะเป็น]
non-functional

Qualitied Software

- Reliability ความน่าเชื่อถือ
- Efficiency ประสิทธิภาพ (การใช้ทรัพยากรคุ้มค่า)
- Maintainability ความสามารถในการดูแลรักษา^{เวลา change จะแก้ไขน้อยที่สุด} Ex จดที่ไฟล์ URL for service คราวใดๆ ใน config file > ปุ๊บ hard code
- Compatibility ความสามารถเข้ากันได้กับสิ่งอื่นหรือเวอร์ชันอื่น
- Usability สามารถใช้ได้ง่ายเข้าใจได้ง่าย
- Performance ความสามารถในตอบสนองผู้ใช้ (เร็ว, เยอะ)
ยกสุด ↴ เพิ่ม sever, load balance ร่วม แต่ < coding ให้ดี

Bug ?



Bug ?

(033) PRO 2 2.130476415

control 2.130676415

Relays 6-2 in 033 failed special speed test
in relay " 10.00 test .

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Multi Adder Test.

1545



Relay #70 Panel F
(moth) in relay.

1630 Argument started.

First actual case of bug being found.

1700 closed down .

US NAVAL HISTORICAL CENTER

Terms

นำไปใช้!!

- Defect ปัญหาที่พบในการทดสอบระบบ (เกิดจาก tester)
- Error กิจกรรมของมนุษย์ที่ทำให้เกิดผลลัพธ์ที่ผิด (เกิดจาก user)
- Bug การประมวลผลตัวของ error นั้นๆ ของซอฟต์แวร์ทำงาน สิ่งที่มีใน code แล้วท้าวไป
- Fault สถานะของซอฟต์แวร์ที่เกิดจาก error ตัวนั้น เกิด error
- Failure ความผิดพลาดจากการทำงานของซอฟต์แวร์



A person makes an **error**

that creates a **fault** in the software

that can cause a **failure** in operation.

Defect Types เจ้อได้ทุกอย่าง

- Requirement → ผิดกันเอง Ex รัน req. จากหลายคน
 - เป็น Defect ที่เกิดจากการแก้ไข Requirement ของทาง Business โดยไม่แจ้งทีมที่เกี่ยวข้อง
- Coding
 - เป็น Defect ที่เกิดจากการ Coding ของทาง Developer ที่ไม่ตรวจสอบในส่วนนั้นๆ
- Graphic Design → ถูก Responsive Ex. กดแล้วมือะไรหายไปหมด
 - เป็น Defect ที่เกิดจากการ Design ที่ไม่ลองรับกับ Browser ต่างๆ หรือ เมื่อ developer นำ Design มาประกอบกับ Code แล้วทำให้การแสดงผลไม่ถูกต้อง
- Data Test Ex ส่งข้อมูลที่ไม่ภาษาไทย ระบบที่ส่งไม่รองรับ
 - เป็น Defect ที่เกิดจาก Test data อาจไม่มีใน Environment หรือระบบอาจไม่รองรับ data นี้
- Other
 - ข้อจำกัดของระบบ, ข้อจำกัดของ Environment

Defect Severity ในชีวิต tester ไม่ตายตัว → ดูผลกระทบกับ business

- Critical (show stopper)
 - Defect ที่ไม่สามารถทดสอบโปรแกรมในส่วนของ Function นั้นต่อได้เลย
- High → ผลกระทบกับ business Ex คำนวณเงินคนร. ผิด คล้าย Critical | แต่ยังฟื้นฟูได้ Ex ลูกบ.วิชา ไม่ได้ ยังส่งซ่อมให้ได้
 - Defect ที่เกิดจากการใส่ข้อมูลถูกต้อง แต่ระบบแสดงผลผิดพลาด เช่น Error ต่างๆ
- Medium Ex เงินนาทีไม่ใช้ล็อกต่างๆ
 - ระบบจะแสดงผลถูกต้องเมื่อใส่ข้อมูลถูกต้อง แต่เมื่อใส่ข้อมูลไม่ถูกต้อง ระบบจะแสดงผลผิดพลาด เช่น Filed ที่มีการ Validate ผล เมื่อใส่ ค่าว่าง, อักขระพิเศษ (‘ , ” , % , &) และ Script ที่มีผลต่อการแสดงผลของระบบ ๆ จะแสดงผลผิดพลาด
- Low ผิดที่ VI
 - Defect ที่เกิดจากการแสดงผลของข้อความ หรือ เรื่องของการ Design ซึ่ง Defect เหล่านี้จะไม่มีผลกระทบกับการทำงานของระบบ

Software Testing

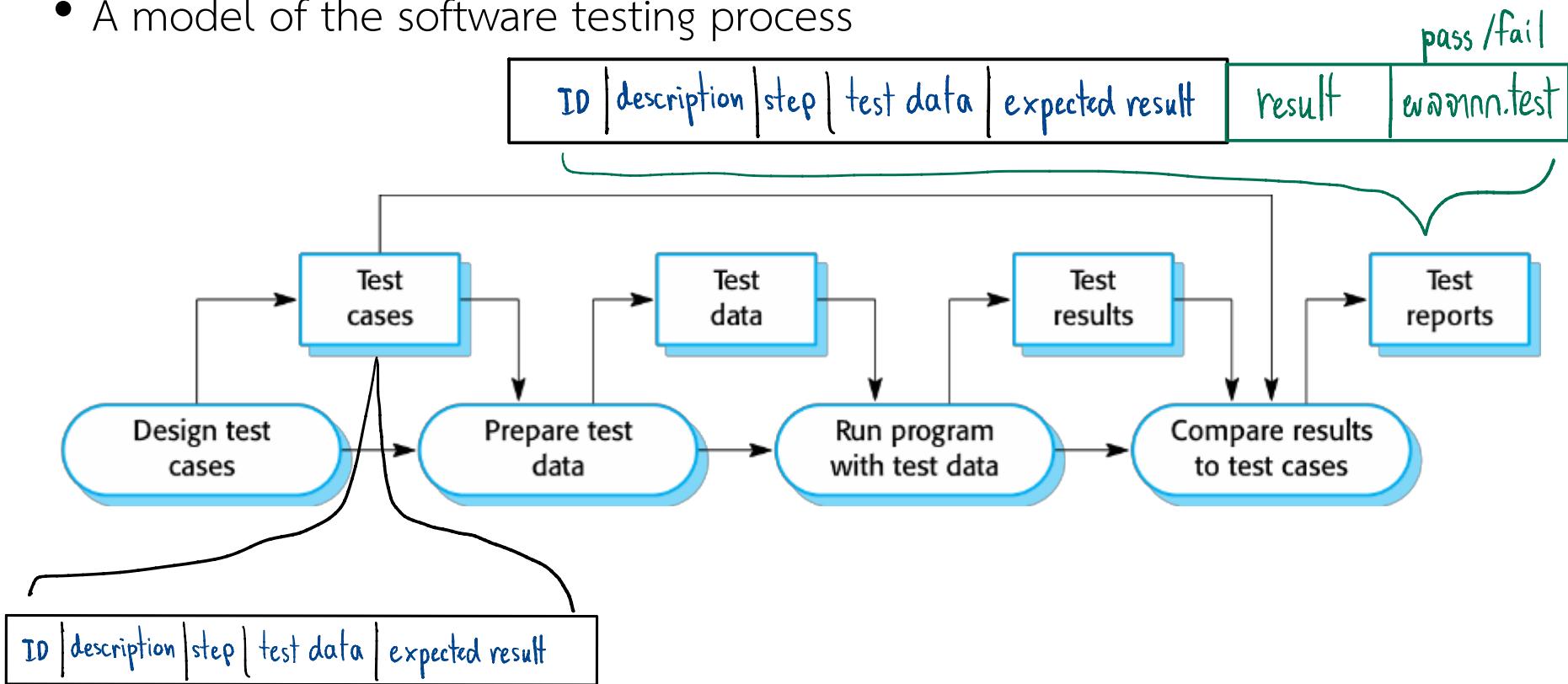
SW tester ที่ดี < ช่างจินตนาการ - ว่า P. ควรเน้นย้ำไป
ช่างลงสัญ - ถ้าทำอย่างนี้จะเป็นยังไง

- เป็นการทดสอบความสมบูรณ์ของโปรแกรม รวมทั้งความน่าเชื่อถือและความถูกต้องของผลลัพธ์จากโปรแกรมที่พัฒนาขึ้น
- ตรวจสอบประสิทธิภาพการทำงานของซอฟต์แวร์ไปด้วย
- เป็นผลให้สัมพันธ์กับคุณภาพของซอฟต์แวร์ตามไปด้วย
- เป็นกิจกรรมที่จัดทำขึ้นเพื่อประเมินและปรับปรุงคุณภาพของซอฟต์แวร์ โดยการหาข้อผิดพลาดและปัญหาที่เกิดขึ้นและทำการแก้ไขปัญหา
- ส่วนใหญ่ทำโดย Testers → นั้นคันนี่ Dev ทำ unit test
→ unit test - test ทุก function
- มีทำโดย Developers และ Users
- มีแบบแผนการปฏิบัติ

Testing Process

ຈະ. Man day ໃນກ.ທ່າງນີ້ຄັດ

- A model of the software testing process



Software Testing

Test?



Black Box Testing

~
writing code



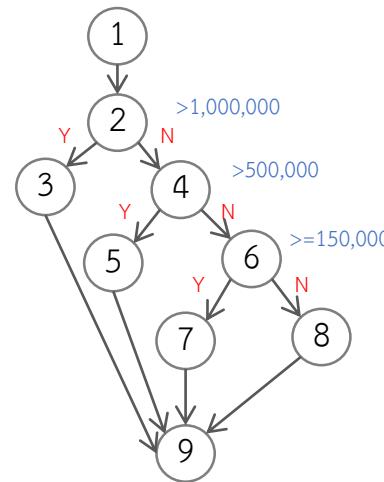
Test case	Precondition	Event	Expected Result	Note
TC1	Your cart is empty	Click btn Add item	1 item in your cart	S1=>S2
TC2	n>=1 items in your cart	Click btn Add item	n+1 items in your cart	S2=>S2
TC3	1 item in your cart	Click btn Remove item	Your cart is empty	S2=>S1
TC4	n>=2 items in your cart	Click btn Remove item	n-1 items in your cart	S2=>S2
TC5	n>=1 items in your cart	Click btn Check out	Display screen Check out	S2=>S3
TC6	Direct to screen Check out	Click btn Back	Display screen Shopping	S3=>S2
TC7	Direct to screen Check out	Click btn Payment	Display screen Payment	S3=>S4

White Box Testing នៃកូដ

ត្រូវការបន្ទាត់ទាំងអស់ execute

```
float taxCal (float salary) {  
    1   float tax = 0.0;  
    2   if(salary>1,000,000) {  
    3       tax = salary*0.25 ;  
    4   }else if(salary>500,000) {  
    5       tax = salary*0.15 ;  
    6   }else if(salary>=150,000) {  
    7       tax = salary*0.05 ;  
    8   }else{  
    9       tax = 0.0  
    }  
    return tax;  
}
```

Flowchart



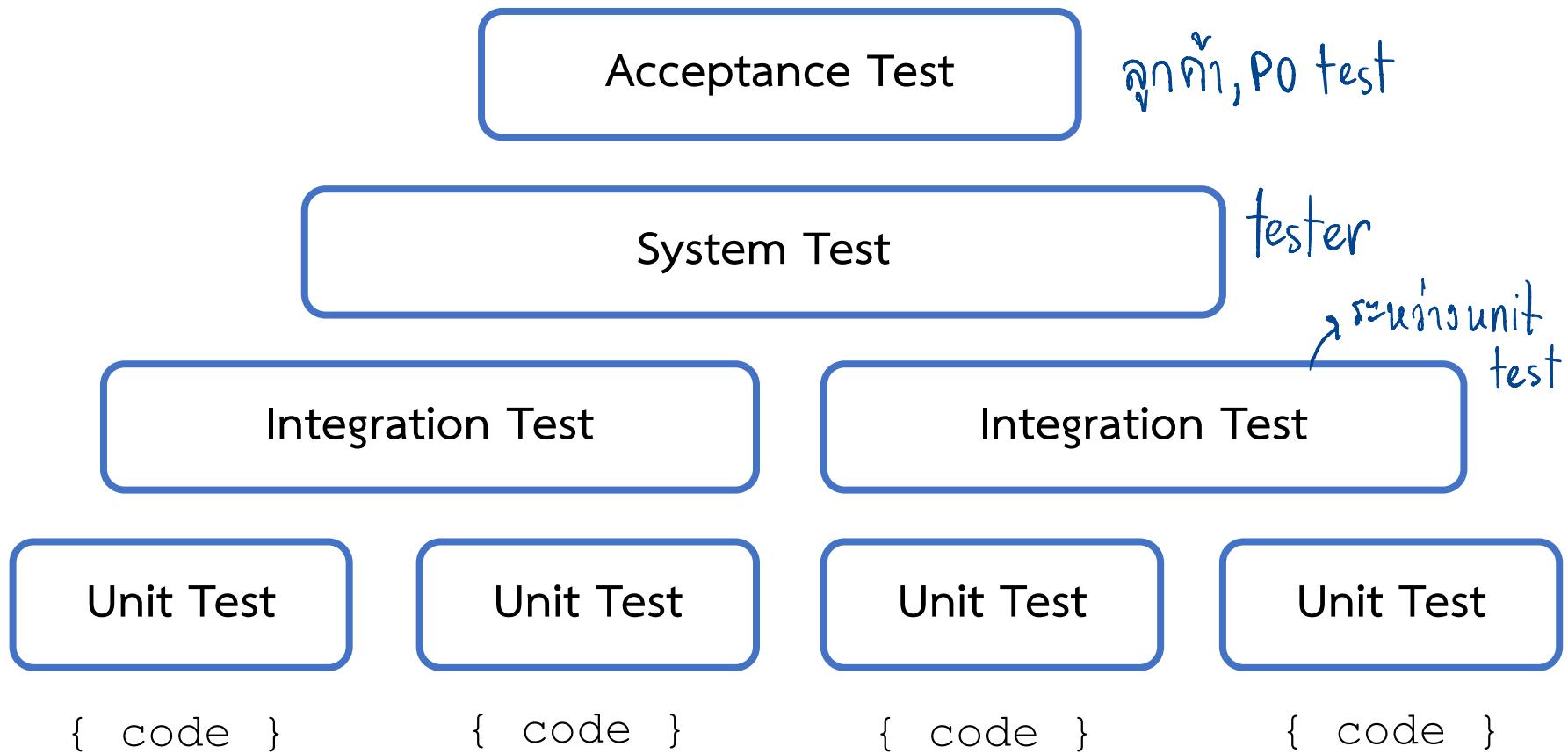
Path Coverage

- 1,2,3,9
- 1,2,4,5,9
- 1,2,4,6,7,8
- 1,2,4,6,8,9

Test Cases

- | | |
|----------------------|--------------------|
| • salary = 0 | • salary = 499,999 |
| • salary = 999,999 | • salary = 500,000 |
| • salary = 1,000,000 | • salary = 500,001 |
| • salary = 1,000,001 | • ខ្លួន |

Software Testing



Unit Test

Function ใน class ชื่อ “Tax”

```
float taxCal (float salary){  
    float tax = 0.0;  
    if(salary>1,000,000){  
        tax = salary*0.25 ;  
    }else if(salary>500,000){  
        tax = salary*0.15 ;  
    }else if(salary>=150,000){  
        tax = salary*0.05 ;  
    }else{  
        tax = 0.0  
    }  
    return tax;  
}
```

Unit Test Script (เป็น White Box Testing)

```
public class MyTest{  
    @Test  
    public void TestTax25(){  
        Tax tester = new Tax();  
        assertEquals("salary > 1,000,000",  
                    25000.25, tester.taxCal(1000001));  
  
        assertEquals("salary = 1,000,000",  
                    25000.00, tester.taxCal(1000000));  
    }  
    // more cases  
}
```

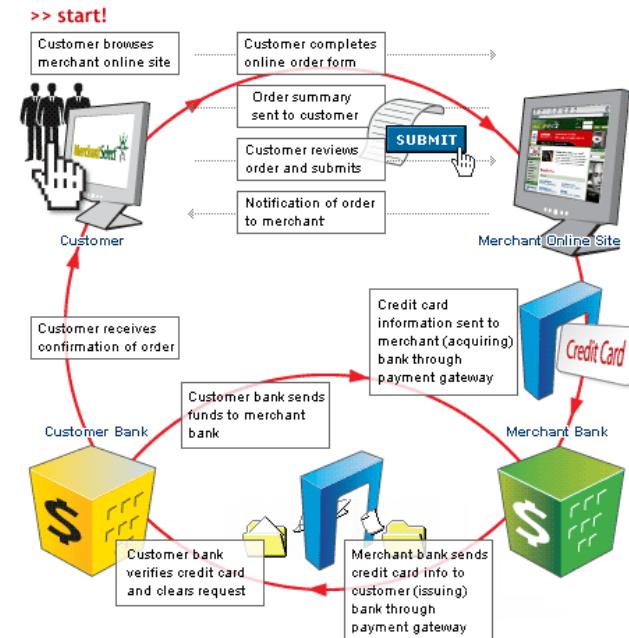
Integration Test

- เพื่อให้เห็นว่าระบบย่อยทั้งหมดทำงานร่วมกันได้
- ทดสอบการทำงานของ classes, modules, หรือ subsystems ต่างๆ เมื่อมาประกอบร่วมกันทำงานแล้ว
- ถ้ามี API ก็ต้องทดสอบร่วมกับ API ด้วย
- ถ้ามี Database ก็ต้องทดสอบกับ Database ด้วย



System Test

- เป็นการทดสอบการเชื่อมต่อระหว่างระบบของซอฟต์แวร์ที่พัฒนาขึ้น หรือทดสอบกับระบบอื่นๆ
- ทดสอบระบบการโอนเงิน กับระบบธนาคาร
- ทดสอบระบบการโอนเงิน กับระบบบัญชีผู้ใช้
- Alpha Testing คือ จำลองการทำงานระบบให้เหมือนจริงในฝั่งนักพัฒนา
- Beta Testing หรือ Pilot Testing ทดสอบกับระบบจริงๆ ด้วยสิ่งแวดล้อมจริงก่อนส่งมอบ



Acceptance Test บ.รัฐ - อบรมก.ป.งงาน

- ทดสอบระบบจาก Requirement หรือ User Story ของลูกค้า
- ระบบต้องสามารถใช้งานได้จริงและสมบูรณ์ตรงตาม Business Logic ที่ตกลงกันไว้
- ลูกค้า และ/หรือ คนที่ให้ requirement มีส่วนร่วมในการเขียน Test Case และทดสอบ
- ทดสอบทุก Roles ของผู้ใช้
- สภาพแวดล้อม (Hardware, Software, และ Infrastructure) ต้องเหมือนจริงมากที่สุด.
- ตัวอย่าง
 - ผู้ดูแลระบบสามารถนำข้อมูลสินค้าเข้าในระบบ และเมื่อนำสินค้าเข้าสู่ระบบแล้ว ผู้ใช้จะสามารถเห็นข้อมูลของสินค้านั้นได้ และสามารถค้นหาได้

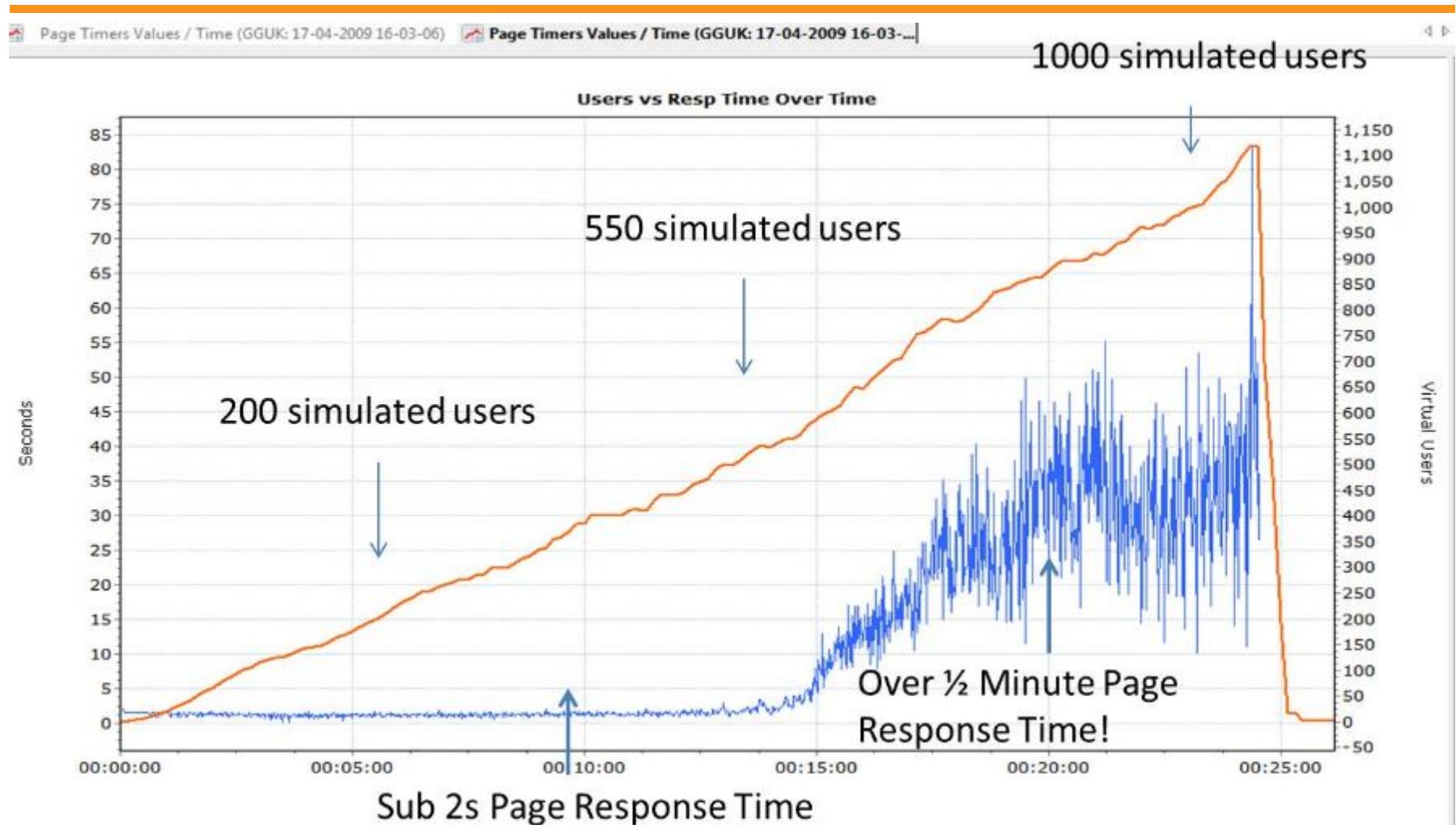
Performance Testing

- **Load testing**
 - การทดสอบซอฟต์แวร์หรือระบบว่า ระบบจะมีความเร็วมากน้อยแค่ไหน ภายใต้สภาวะและขนาดของภาระที่คาดว่าจะเกิดขึ้นจริง เช่น หากมีผู้ใช้งานเข้ามาใช้ระบบพร้อมกัน 100 คน ระบบจะตอบสนองเร็วหรือช้าแค่ไหน (concurrent users)
- **Stress testing**
 - การทดสอบระบบที่นอกเหนือจากการทำงานปกติ เพื่อทดสอบความเสถียร ในความพร้อมและการจัดการข้อผิดพลาด เมื่อระบบมีการทำงานหนัก
- **Spike testing**
 - การทดสอบเมื่อมีการเพิ่มจำนวนผู้ใช้งานอย่างรวดเร็ว
- **Soak testing หรือ Endurance testing**
 - การทดสอบระบบว่า ระบบยังสามารถทำงานได้ดีหรือไม่ เมื่อมีการใช้งานในเวลานาน throughput and/or response times ยังดีเหมือนกับตอนเริ่มต้นหรือไม่
- **Capacity testing**
 - การทดสอบเพื่อกำหนดว่า จะมีผู้ใช้กี่คนที่ระบบสามารถรองรับได้ โดยที่ระบบสามารถยังทำงานได้

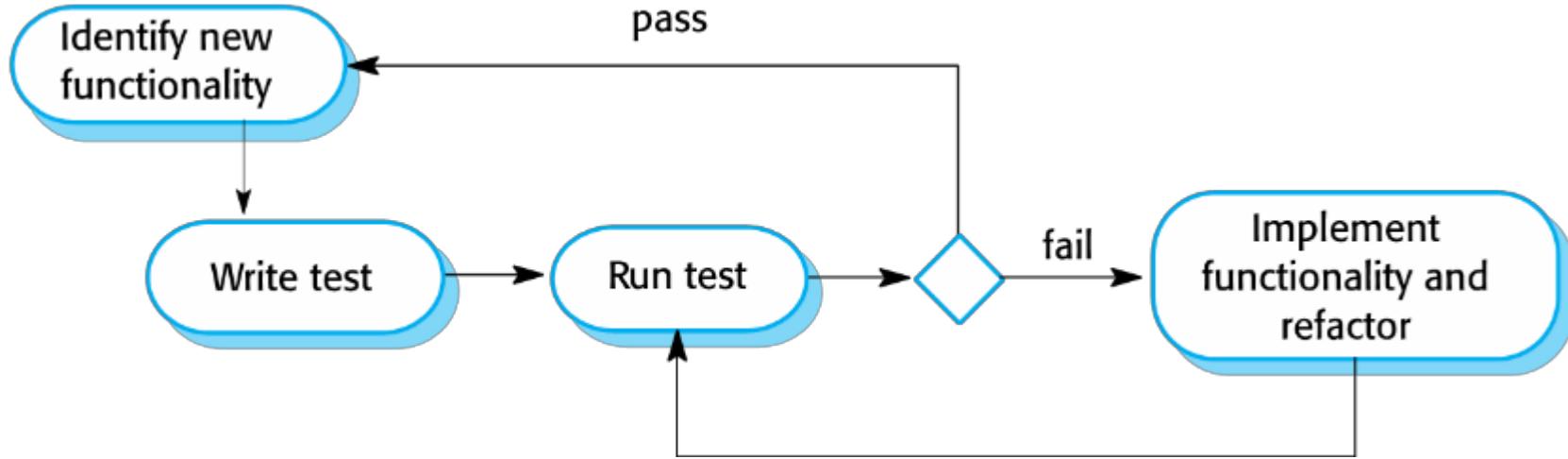
Performance Testing

- Recovery testing
 - การทดสอบระบบว่า ระบบสามารถฟื้นตัวจากการล่มได้เร็วหรือดีแค่ไหน
- Smoke testing **ใช้เวลาเพียง environment ที่รู้จ้ารพิชิต**
 - การเริ่มต้นทดสอบระบบในการทดสอบประสิทธิภาพ เพื่อดูว่า ระบบสามารถทำงานได้ปกติในสภาพปกติ
- Volume testing
 - การทดสอบโดยการใช้จำนวนข้อมูล เพื่อแสดงให้เห็นว่า จำนวนข้อมูลเท่าไหร่ที่ระบบไม่สามารถรองรับได้
- Network Sensitivity testing
 - การทดสอบที่จำกัดของ WAN และ การทำงานของ network สามารถที่จะคาดการณ์ผลกระทบในส่วนของ WAN และ การสื่อสารบน bandwidth
- Scalability testing
 - การทดสอบเพื่อวัดความสามารถในการประยุกต์ใช้เมื่อนำไปใช้กับระบบที่ใหญ่ขึ้น หรือ ระบบอื่นๆที่จะทำไปใช้

Load Test Report



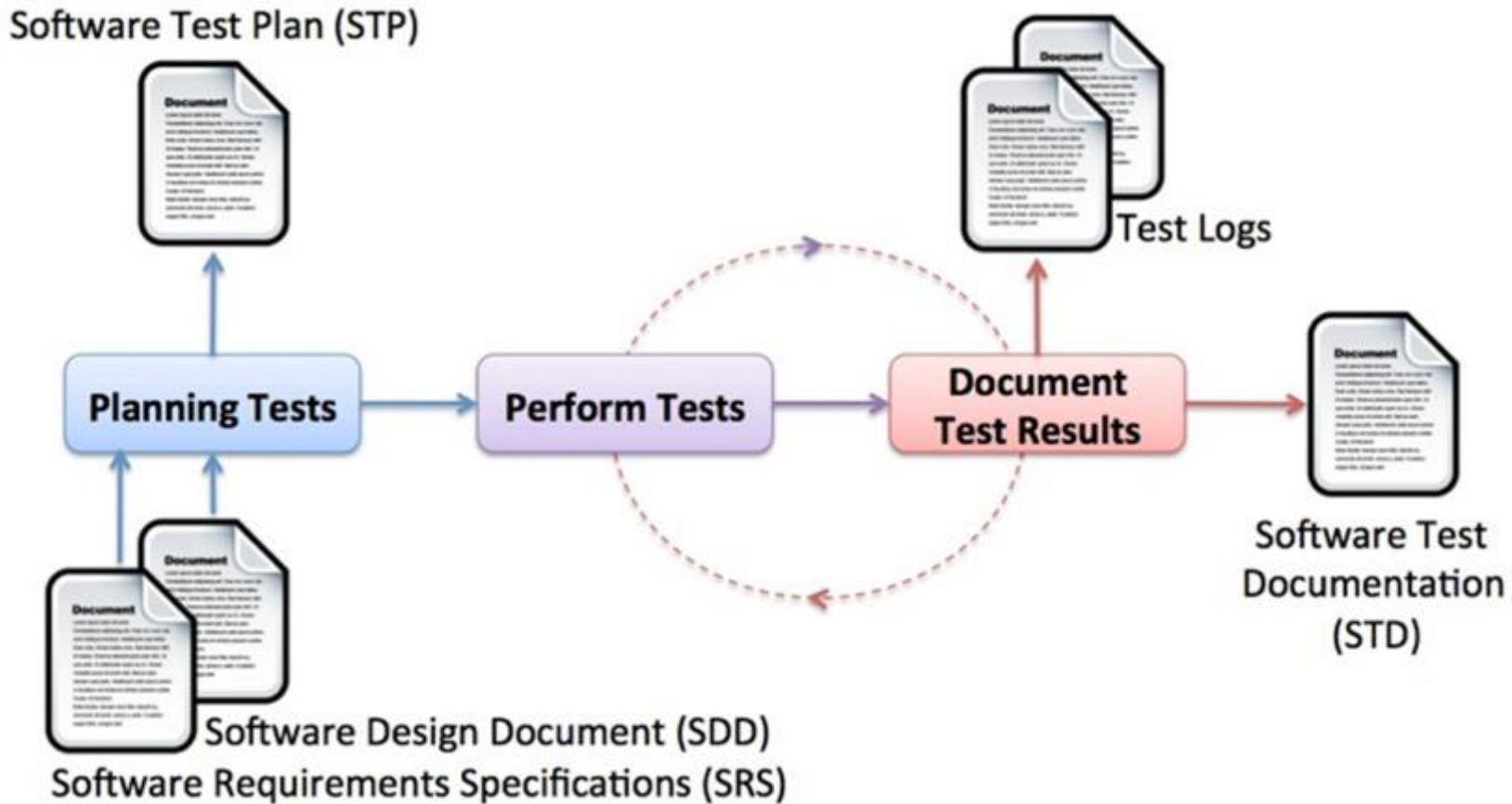
Test-Driven Development



- เป็นการเขียน Test Script ขึ้นมาก่อนแล้วจึงเขียน Code เพื่อทำให้แต่ละ Test Case ผ่าน
- ประโยชน์ของ Test-Driven Development
 - เขียน Code ได้อย่างมีทิศทาง
 - ครอบคลุมทุก Requirements
 - ไม่ได้ถือว่าเสียเวลา เพราะเมื่อนอกจากการทำ Unit Test ก่อนเริ่มเขียน Code

Testing Documentation

Test Documentation



Software Test Plan

- Introduction
- Test items
- Features to be tested
- Testing approach
- Item pass/fail criteria
- Suspension and resumption
- Deliverables
- Tasks
- Environmental needs
- Responsibilities
- Staffing and training needs
- Costs and schedule
- Risks and contingencies.

Software Testing Plan

- ตัวอย่าง: Testing Mobile Business Applications

Approach	Type of Testing	Manual Testing		Automated Testing on Device
		Using Device	Using Emulators	
Standard Testing	Unit Testing	No	Yes	No
	Integration Testing	No	Yes	No
	System Testing	Yes	No	No
	Regression testing	Yes	No	Yes
	Acceptance testing	Yes	No	No
Special type of testing to address specific challenges	Compatibility Testing	Yes	No	Yes
	GUI Testing	Yes	No	No
Type of testing more relevant for enterprise mobile business application	Performance Testing	Yes	No	Yes
	Security Testing	Yes	No	Yes
	Synchronization Testing	Yes	No	No

Test Case Description

- Test items
- Input specifications
- Output specifications
- Environmental needs
- Special procedural requirements/rules
- Intercase dependencies.

Test Case Description

- Verify the login of Gmail

Project Name:	Google Email								
Module Name:	Login								
Reference Document:	If any								
Created by:	Rajkumar								
Date of creation:	DD-MMM-YY								
Date of review:	DD-MMM-YY								
TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/FAIL)
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Valid Password>	Successful login	Gmail inbox is shown		
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown			
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Valid Password>	A message "The email and password you entered don't match" is shown			
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown			

Test Case Description

- Verify the login of Gmail

TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT
Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Valid Password>	Successful login
Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown
Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Valid Password>	A message "The email and password you entered don't match" is shown
Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown

Test Log /Defect Log

- **Description**
 - Test item identification
 - Test environment description
- **Activity and event entries**
 - Date and time
 - Author
 - Test procedure identifier
 - Staff present
 - Pass/fail
 - Error messages generated
 - Environmental information
 - Anomalous events

Defect Tracking

Jira

Marker / MAR-131
[Pricing] - Update the price to \$29

Edit Comment Assign To Do In Progress Workflow Admin

Details

Type:	Bug	Status:	TO DO (View workflow)
Priority:	High	Resolution:	Unresolved
Labels:	None		
Environment:	Browser Chrome 54.0.2840.71 Screen Size 1920 x 1200 Viewport Size 1607 x 920 Zoom L...		

Description

* Summary: → test arls
The price mentioned on the pricing page is not correct

Steps to Reproduce: → step wa Bug
Go to the pricing page

Expected Results
The price for the basic plan should be \$29

Actual Results:
The price for the basic plan is currently \$25

—

Source URL: <https://www.shopify.com/pricing>

People

Assignee: gary Assign to me

Reporter: Christophe Han

Votes: 0

Watchers: 1 Stop watching th

Dates

Created: 1 minute ago
Updated: 1 minute ago

Agile

[View on Board](#)

HipChat discussions

Do you want to discuss this issue? Connect

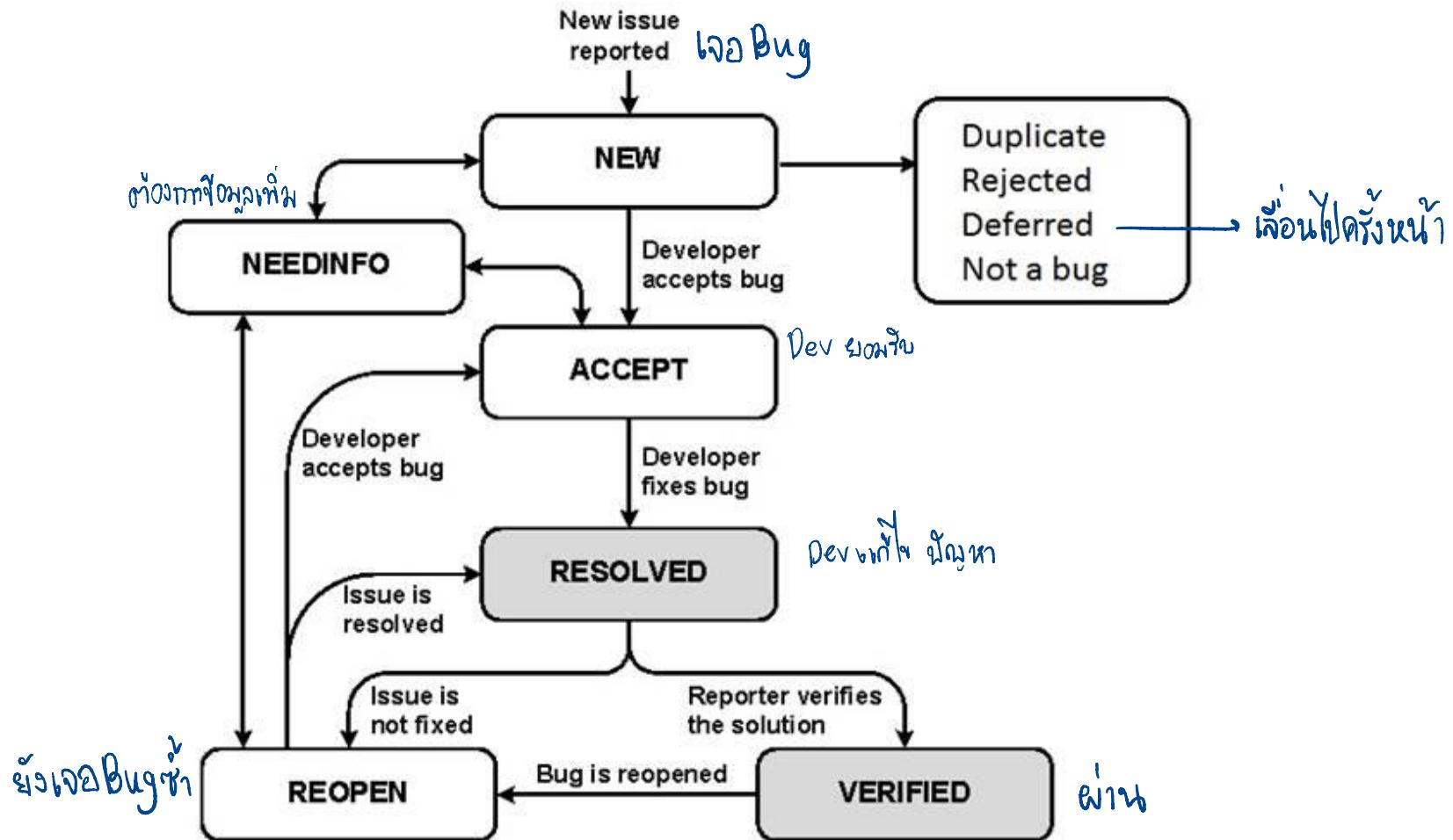
Connect Dismiss

Attachments

Drop files to attach, or browse.



Defect Status Flow



Summary

Summary



- Test Techniques
 - Black Box Testing
 - White Box Testing
- Test Types
 - Unit Testing
 - Integration Testing
 - System Testing
 - Acceptance Testing
 - Performance Testing *
 - Test-First Development

“

Quality is never an accident;
it is always the result of
intelligent effort.

”

John Ruskin

つづく

Discussion

សម្រាប់ software ពីអគ្គនភាព បង្កើតក្នុងក្រោម

ทำไมต้องมี Software Testing ?

Qualitied Software

- WordPress
- คูม่า } front end

มี functional requirement ที่อย่างไร , ตามที่ งาน ความต้องการ
non functional requirement ไม่ต้อง

- Reliability

ความน่าเชื่อถือ

- Efficiency

ประสิทธิภาพ (การใช้ทรัพยากรุ่มค่า)

- Maintainability

ความสามารถในการดูแลรักษา
ก้าวหน้าต่อไป → Software ต้องเก่งหัดอยู่

- Compatibility

ความสามารถเข้ากันได้กับสิ่งอื่นหรือเวอร์ชันอื่น
ไฟล์ config file

- Usability

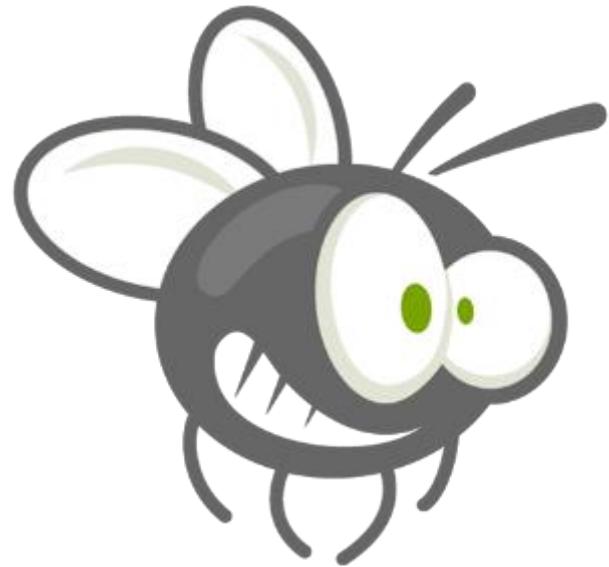
สามารถใช้ได้ง่ายเข้าใจได้ง่าย

- Performance

ความสามารถในตอบสนองผู้ใช้ (เร็ว, เยอะ)
Coding ก่อไป big O

Bug ?

Dev no nnnnnn



Bug ?

(033) PRO 2 2.130476415

control 2.130676415

Relays 6-2 in 033 failed special speed test
in relay " 10.00 test .

Relays changed

1100 Started Cosine Tape (Sine check)

1525 Started Multi Adder Test.

1545



Relay #70 Panel F
(moth) in relay.

1630 Argument started.

First actual case of bug being found.

1700 closed down .

US NAVAL HISTORICAL CENTER

Terms

- Defect ปัญหาที่พบในการทดสอบระบบ เนื่องจากมี requirement ที่ลืมกันลง ไม่ตรงกับ
- Error กิจกรรมของมนุษย์ที่ทำให้เกิดผลลัพธ์ที่ผิด นูก ÷ 0
- Bug การประกูตัวของ error นั้นขณะซอฟต์แวร์ทำงาน ← หน้าที่ Tester ก็อยู่นี่เอง
ร่องรอยใน code ที่เกิดขึ้นโดยบังเอิญ
- Fault สถานะของซอฟต์แวร์ที่เกิดจาก error ตัวนั้น (อาจร้ายแรงถึงระเบิด)
- Failure ความผิดพลาดจากการทำงานของซอฟต์แวร์
↑
or



A person makes an **error**

that creates a **fault** in the software

that can cause a **failure** in operation.

Defect Types

- Requirement
 - เป็น Defect ที่เกิดจากการแก้ไข Requirement ของทาง Business โดยไม่แจ้งทีมที่เกี่ยวข้อง
- Coding
 - เป็น Defect ที่เกิดจากการ Coding ของทาง Developer ที่ไม่ตรวจสอบในส่วนนั้นๆ
- Graphic Design ดีไซน์ ดีไซน์
 - เป็น Defect ที่เกิดจากการ Design ที่ไม่ลองรับกับ Browser ต่างๆ หรือ เมื่อ developer นำ Design มาประกอบกับ Code แล้วทำให้การแสดงผลไม่ถูกต้อง
- Data Test
 - เป็น Defect ที่เกิดจาก Test data อาจไม่มีใน Environment หรือระบบอาจไม่รองรับ data นี้
- Other
 - ข้อจำกัดของระบบ, ข้อจำกัดของ Environment

Defect Severity ความคุณภาพของ Defect

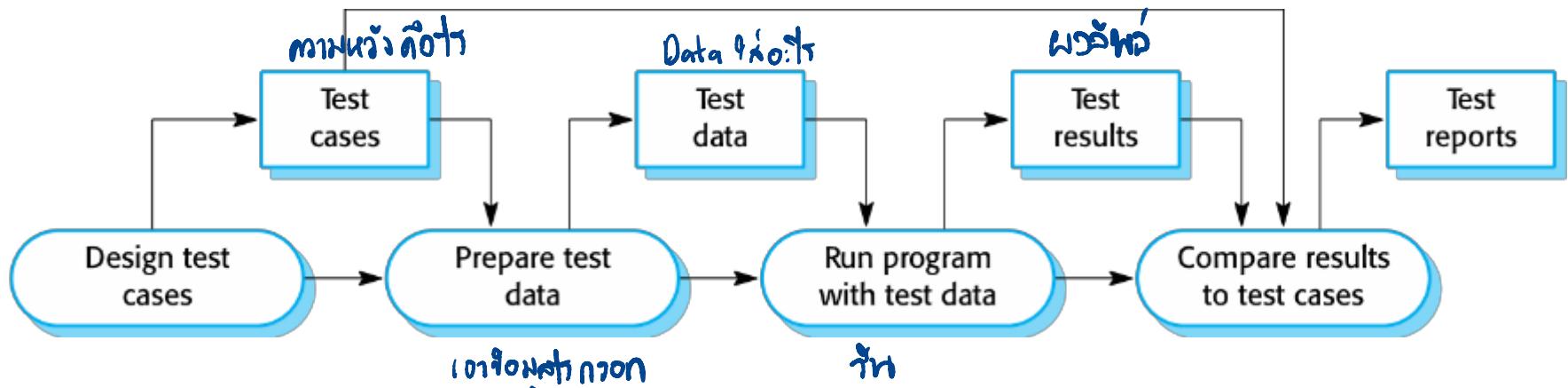
ระดับความสำคัญ	คำอธิบาย	คำแนะนำ
Critical	Software หลุดจากมาตรฐานที่กำหนดไว้ ทางกีตากนแบบส่วนมาก	ห้ามนำเข้าระบบ
High	Defect ที่ไม่สามารถทดสอบโปรแกรมในส่วนของ Function นั้นต่อได้เลย	ห้ามนำเข้าระบบ
Medium	Defect ที่เกิดจากการใส่ข้อมูลถูกต้อง แต่ระบบแสดงผลผิดพลาด เช่น Error ต่างๆ	ห้ามนำเข้าระบบ
Low	Defect ที่เกิดจากการแสดงผลของระบบ หรือเรื่องของการ Design ซึ่ง Defect เหล่านี้จะไม่มีผลกระทบกับการทำงานของระบบ	ห้ามนำเข้าระบบ

Software Testing

- เป็นการทดสอบความสมบูรณ์ของโปรแกรม รวมทั้งความน่าเชื่อถือและความถูกต้องของผลลัพธ์จากโปรแกรมที่พัฒนาขึ้น
- ตรวจสอบประสิทธิภาพการทำงานของซอฟต์แวร์ไปด้วย
- เป็นผลให้สัมพันธ์กับคุณภาพของซอฟต์แวร์ตามไปด้วย
- เป็นกิจกรรมที่จัดทำขึ้นเพื่อประเมินและปรับปรุงคุณภาพของซอฟต์แวร์ โดยการหาข้อผิดพลาดและปัญหาที่เกิดขึ้นและทำการแก้ไขปัญหา
- ส่วนใหญ่ทำโดย Testers Dev ห้องทำ unit test เทคนิค: ทฤษฎี, ทฤษฎี function ทำ TDD
- มีทำโดย Developers และ Users
- มีแบบแผนการปฏิบัติ

Testing Process

- A model of the software testing process

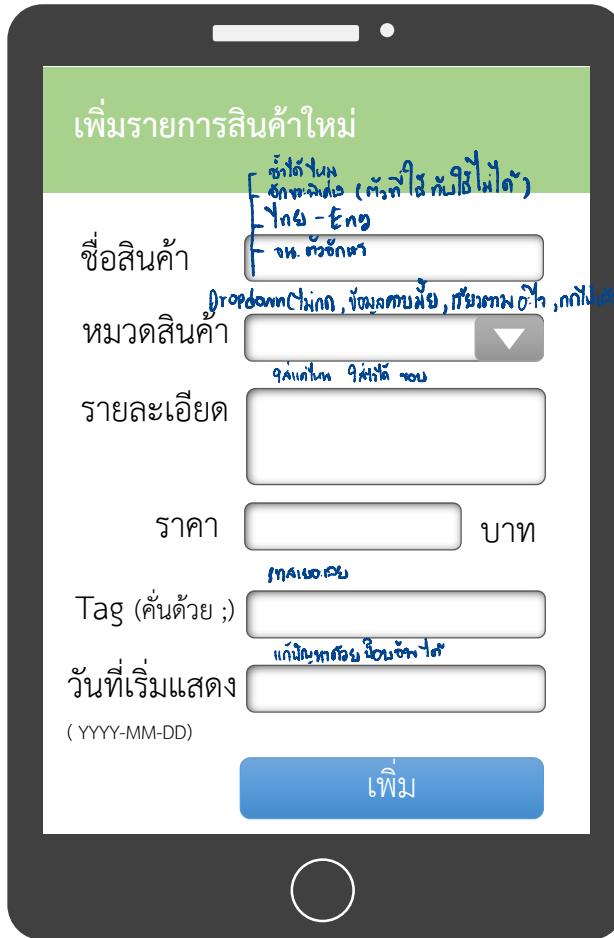


Software Testing

1. แบบ

2.

Test?



Black Box Testing

ไม่สนใจใน code



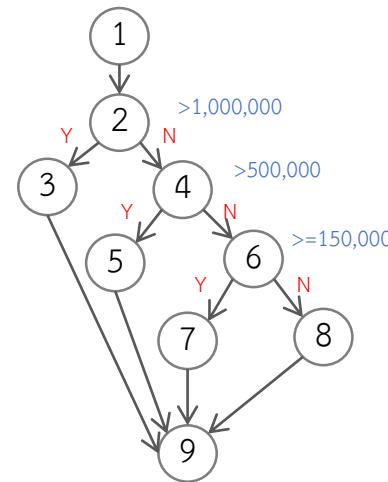
Test case	Precondition	Event	Expected Result	Note
TC1	Your cart is empty	Click btn Add item	1 item in your cart	S1=>S2
TC2	n>=1 items in your cart	Click btn Add item	n+1 items in your cart	S2=>S2
TC3	1 item in your cart	Click btn Remove item	Your cart is empty	S2=>S1
TC4	n>=2 items in your cart	Click btn Remove item	n-1 items in your cart	S2=>S2
TC5	n>=1 items in your cart	Click btn Check out	Display screen Check out	S2=>S3
TC6	Direct to screen Check out	Click btn Back	Display screen Shopping	S3=>S2
TC7	Direct to screen Check out	Click btn Payment	Display screen Payment	S3=>S4

White Box Testing

b&w code [Dev mi]

```
float taxCal (float salary) {  
    1    float tax = 0.0;  
    2    if(salary>1,000,000) {  
    3        tax = salary*0.25 ;  
    4    }else if(salary>500,000) {  
    5        tax = salary*0.15 ;  
    6    }else if(salary>=150,000) {  
    7        tax = salary*0.05 ;  
    8    }else{  
    9        tax = 0.0  
    }  
    return tax;  
}
```

Flowchart



Path Coverage

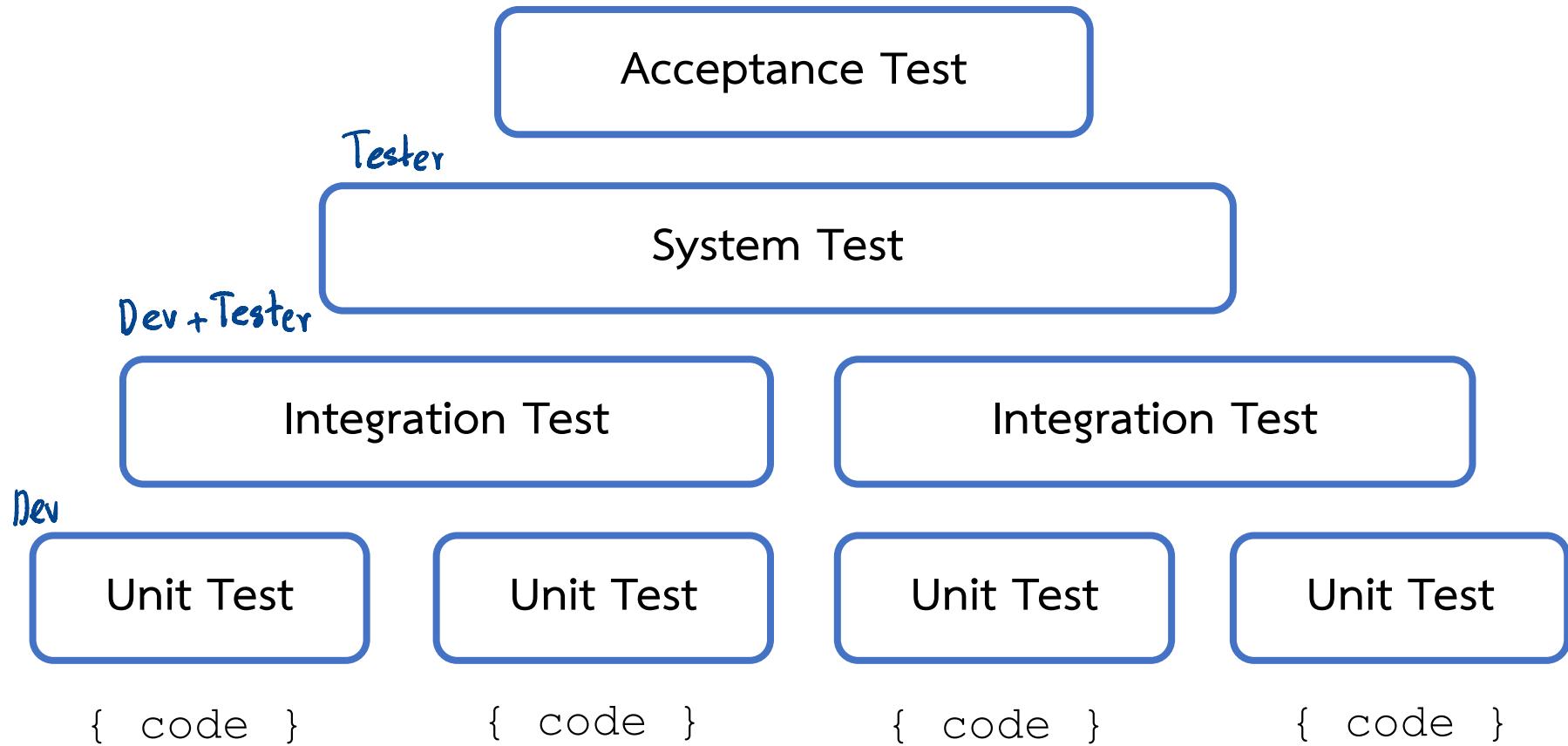
- 1,2,3,9
- 1,2,4,5,9
- 1,2,4,6,7,8
- 1,2,4,6,8,9

Test Cases

- | | |
|---|--|
| <ul style="list-style-type: none">• salary = 0• salary = 999,999• salary = 1,000,000• salary = 1,000,001 | <ul style="list-style-type: none">• salary = 499,999• salary = 500,000• salary = 500,001• อื่นๆ |
|---|--|

Software Testing

POINS (ជិត្យអនុញ្ញាត Test)



Unit Test

Function ใน class ชื่อ “Tax”

```
float taxCal (float salary){  
    float tax = 0.0;  
    if(salary>1,000,000){  
        tax = salary*0.25 ;  
    }else if(salary>500,000){  
        tax = salary*0.15 ;  
    }else if(salary>=150,000){  
        tax = salary*0.05 ;  
    }else{  
        tax = 0.0  
    }  
    return tax;  
}
```

Unit Test Script (เป็น White Box Testing)

```
public class MyTest{  
    @Test  
    public void TestTax25(){  
        Tax tester = new Tax();  
        assertEquals("salary > 1,000,000",  
                    25000.25, tester.taxCal(1000001));  
  
        assertEquals("salary = 1,000,000",  
                    25000.00, tester.taxCal(1000000));  
    }  
    // more cases  
}
```

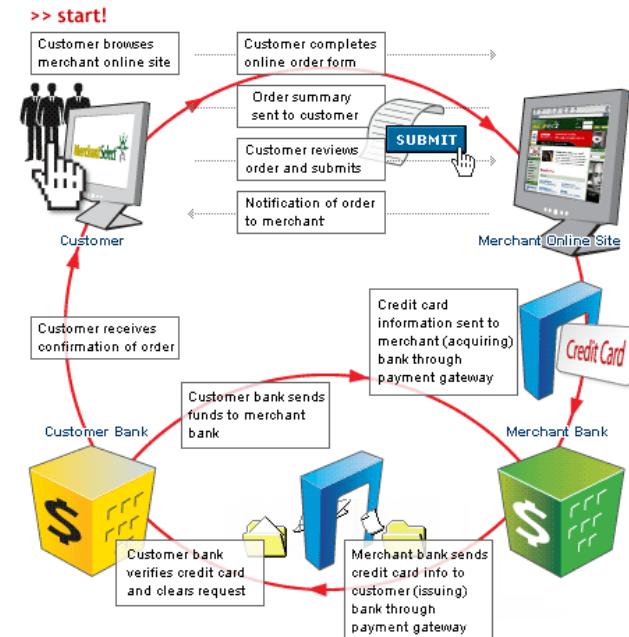
Integration Test

- เพื่อให้เห็นว่าระบบย่อยทั้งหมดทำงานร่วมกันได้
- ทดสอบการทำงานของ **classes**, **modules**, หรือ **subsystems** ต่างๆ เมื่อมาประกอบร่วมกันทำงานแล้ว
- ถ้ามี **API** ก็ต้องทดสอบร่วมกับ **API** ด้วย
- ถ้ามี **Database** ก็ต้องทดสอบกับ **Database** ด้วย



System Test

- เป็นการทดสอบการเชื่อมต่อระหว่างระบบของซอฟต์แวร์ที่พัฒนาขึ้น หรือทดสอบกับระบบอื่นๆ
- ทดสอบระบบการโอนเงิน กับระบบธนาคาร
- ทดสอบระบบการโอนเงิน กับระบบบัญชีผู้ใช้
- Alpha Testing คือ จำลองการทำงานระบบให้เหมือนจริงในฝั่งนักพัฒนา
- Beta Testing หรือ Pilot Testing ทดสอบกับระบบจริงๆ ด้วยสิ่งแวดล้อมจริงก่อนส่งมอบ



Acceptance Test

ການຟັດທະນາການ

- ทดสอบระบบจาก Requirement หรือ User Story ของลูกค้า
- ระบบต้องสามารถใช้งานได้จริงและสมบูรณ์ตรงตาม Business Logic ที่ตกลงกันไว้
- ลูกค้า และ/หรือ คนที่ให้ requirement มีส่วนร่วมในการเขียน Test Case และทดสอบ
- ทดสอบทุก Roles ของผู้ใช้
- สภาพแวดล้อม (Hardware, Software, และ Infrastructure) ต้องเหมือนจริงมากที่สุด.
- ตัวอย่าง
 - ผู้ดูแลระบบสามารถนำข้อมูลสินค้าเข้าในระบบ และเมื่อนำสินค้าเข้าสู่ระบบแล้ว ผู้ใช้จะสามารถเห็นข้อมูลของสินค้านั้นได้ และสามารถค้นหาได้

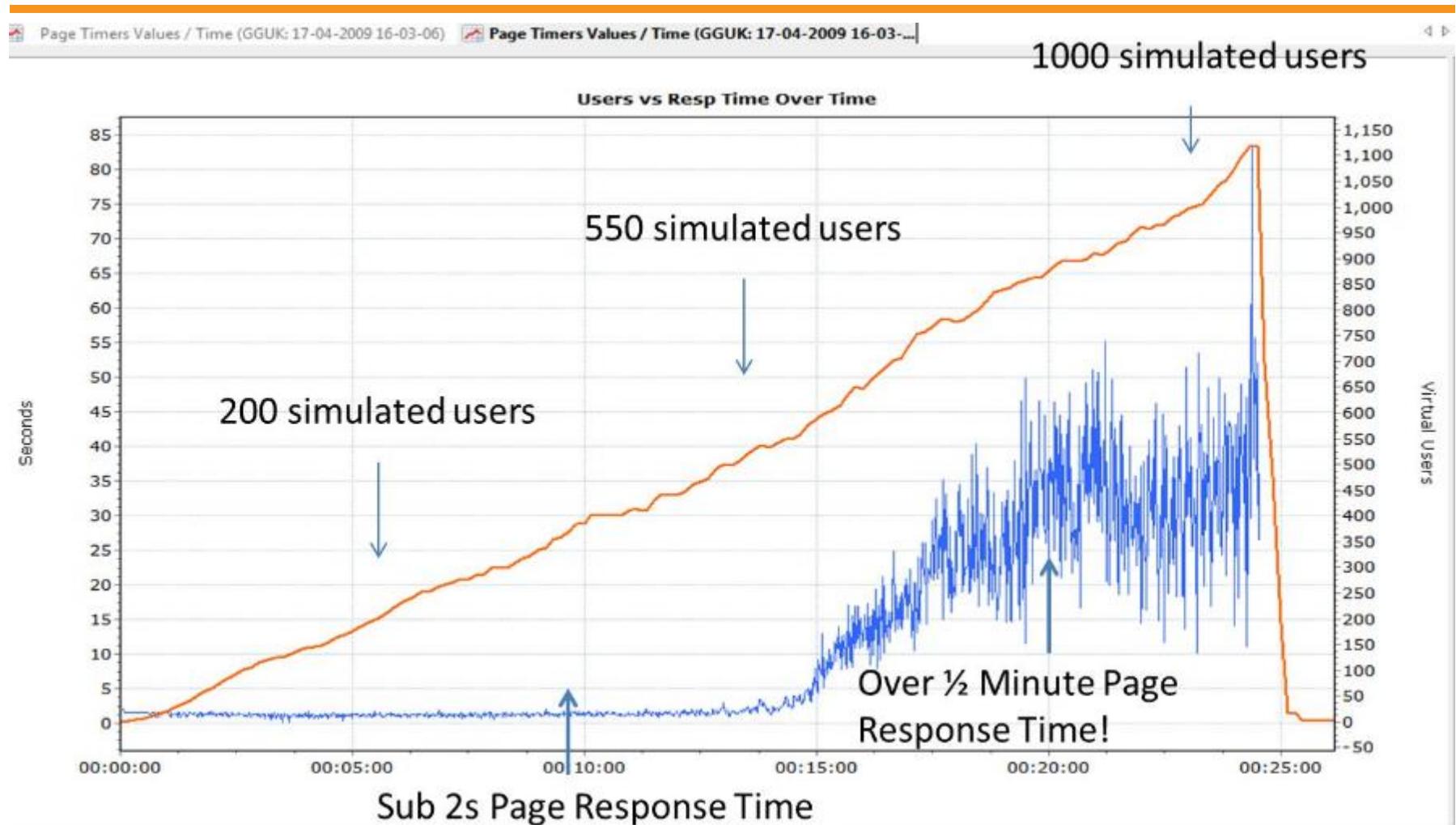
Performance Testing

- **Load testing**
 - การทดสอบซอฟต์แวร์หรือระบบว่า ระบบจะมีความเร็วมากน้อยแค่ไหน ภายใต้สภาวะและขนาดของภาระที่คาดว่าจะเกิดขึ้นจริง เช่น หากมีผู้ใช้งานเข้ามาใช้ระบบพร้อมกัน 100 คน ระบบจะตอบสนองเร็วหรือช้าแค่ไหน (concurrent users)
- **Stress testing**
 - การทดสอบระบบที่นอกเหนือจากการทำงานปกติ เพื่อทดสอบความเสถียร ในความพร้อมและการจัดการข้อผิดพลาด เมื่อระบบมีการทำงานหนัก
- **Spike testing**
 - การทดสอบเมื่อมีการเพิ่มจำนวนผู้ใช้งานอย่างรวดเร็ว
- **Soak testing หรือ Endurance testing**
 - การทดสอบระบบว่า ระบบยังสามารถทำงานได้ดีหรือไม่ เมื่อมีการใช้งานในเวลานาน throughput and/or response times ยังดีเหมือนกับตอนเริ่มต้นหรือไม่
- **Capacity testing**
 - การทดสอบเพื่อกำหนดว่า จะมีผู้ใช้กี่คนที่ระบบสามารถรองรับได้ โดยที่ระบบสามารถยังทำงานได้

Performance Testing

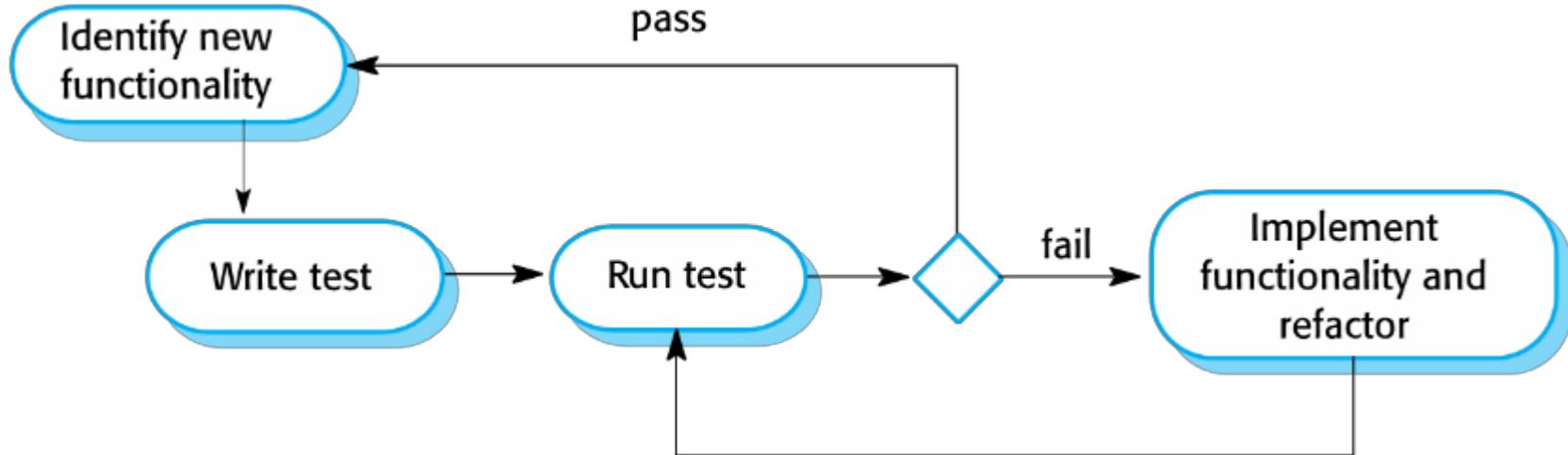
- Recovery testing
 - การทดสอบระบบว่า ระบบสามารถฟื้นตัวจากการล่มได้เร็วหรือดีแค่ไหน
- Smoke testing กgrenผิวเผิน
 - การเริ่มต้นทดสอบระบบในการทดสอบประสิทธิภาพ เพื่อดูว่า ระบบสามารถทำงานได้ปกติในสภาพปกติ
- Volume testing
 - การทดสอบโดยการใช้จำนวนข้อมูล เพื่อแสดงให้เห็นว่า จำนวนข้อมูลเท่าไหร่ที่ระบบไม่สามารถรองรับได้
- Network Sensitivity testing
 - การทดสอบว่าดีจำกัดของ WAN และการทำงานของ network สามารถที่จะคาดการณ์ผลกระทบในส่วนของ WAN และ การสื่อสารบน bandwidth
- Scalability testing
 - การทดสอบเพื่อวัดความสามารถในการประยุกต์ใช้เมื่อนำไปใช้กับระบบที่ใหญ่ขึ้น หรือ ระบบอื่นๆที่จะทำไปใช้

Load Test Report



Test-Driven Development

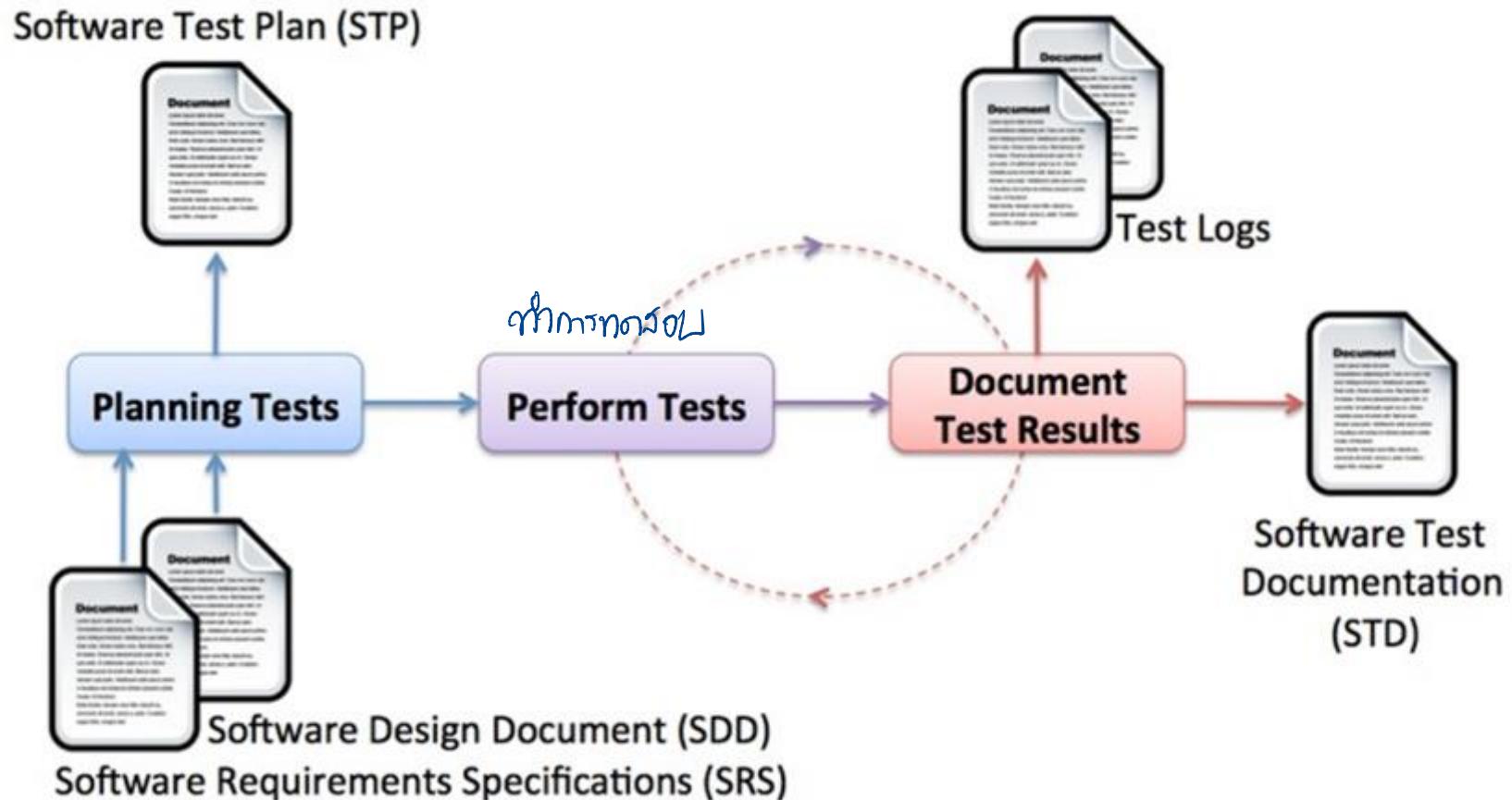
TDD



- เป็นการเขียน Test Script ขึ้นมาก่อนแล้วจึงเขียน Code เพื่อทำให้แต่ละ Test Case ผ่าน
- ประโยชน์ของ Test-Driven Development
 - เขียน Code ได้อย่างมีทิศทาง
 - ครอบคลุมทุก Requirements
 - ไม่ได้ถือว่าเสียเวลา เพราะเมื่อนอกจากการทำ Unit Test ก่อนเริ่มเขียน Code

Testing Documentation

Test Documentation



Software Test Plan

ແນວກາທົນ

- Introduction
- Test items ກາຍກາກຄວບ
- Features to be tested ອຸກສອນທີ່ໄດ້ test
- Testing approach ເທົວກາງການ
- Item pass/fail criteria ສັນຍາກິ່າເກົ່າ
ກາກຈິນເລະເສີມຕິຫຼວມ
- Suspension and resumption
- Deliverables ສຳມັນ
- Tasks ຖານ
ຄວາມຕ້ອງການຕ້າເພື່ອກາລົວໜູນ
- Environmental needs
- Responsibilities ຄວາມຮັບຜິດຮັບ
ການຕ້ອງການໃກງາຫ ໂດຍ ການປົກການ
- Staffing and training needs
ຕໍ່ຜົນຕໍ່ຍ ແລະ ດິການຕ້ານ
- Costs and schedule
ຄວາມເລີຍແລະ ການຊູກັດ
- Risks and contingencies.

Software Testing Plan

- ตัวอย่าง: Testing Mobile Business Applications

Approach	Type of Testing	Manual Testing		Automated Testing on Device
		Using Device	Using Emulators	
Standard Testing	Unit Testing	No	Yes	No
	Integration Testing	No	Yes	No
	System Testing	Yes	No	No
	Regression testing	Yes	No	Yes
	Acceptance testing	Yes	No	No
Special type of testing to address specific challenges	Compatibility Testing	Yes	No	Yes
	GUI Testing	Yes	No	No
Type of testing more relevant for enterprise mobile business application	Performance Testing	Yes	No	Yes
	Security Testing	Yes	No	Yes
	Synchronization Testing	Yes	No	No

Test Case Description

- Test items รายการทดสอบ
- Input specifications
- Output specifications
- Environmental needs ความต้องการตัวห้องสีงาชล์และอุณหภูมิ
- Special procedural requirements/rules ข้อกำหนด/เงื่อนไขพิเศษ
- Intercase dependencies. 關係ที่มีอยู่ระหว่างคดี

Test Case Description

- Verify the login of Gmail

Project Name: Module Name: Reference Document: Created by: Date of creation: Date of review:	Google Email Login If any Rajkumar DD-MMM-YY DD-MMM-YY								
TEST CASE ID	TEST SCENARIO	TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT	POST CONDITION	ACTUAL RESULT	STATUS (PASS/FAIL)
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Valid Password>	Successful login	Gmail inbox is shown		
TC_LOGIN_001	Verify the login of Gmail	Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Invalid Password>				
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Valid Password>	A message "The email and password you entered don't match" is shown			
TC_LOGIN_001	Verify the login of Gmail	Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Invalid Password>				

Test Case Description

- Verify the login of Gmail

TEST CASE	PRE-CONDITION	TEST STEPS	TEST DATA	EXPECTED RESULT
Enter valid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Valid Password>	Successful login
Enter valid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Valid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown
Enter invalid User Name and valid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Valid Password>	A message "The email and password you entered don't match" is shown
Enter invalid User Name and invalid Password	1. Need a valid Gmail Account to do login	1. Enter User Name 2. Enter Password 3. Click "Login" button	<Invalid User Name> <Invalid Password>	A message "The email and password you entered don't match" is shown

Defect

Test Log

100

- Description

- Test item identification ^{Item ID} គម្រោង ឈើ
- Test environment description

- Activity and event entries

- Date and time វេលាហេតុ
- Author ផ្ទើសរើស
- Test procedure identifier ពាក្យរាយការណ៍តាមការងារ
- Staff present អង្គភាពក្នុងក្រុង
- Pass/fail ជូន / មិនជូន
- Error messages generated ការបញ្ជាក់ការងារទាំងអស់
- Environmental information ទីតាំងការងារ
- Anomalous events ពេលវេលាប្រាក់ប្រាក់

Defect Tracking

จีร์ / Git หน้าจอ

Marker / MAR-131
[Pricing] - Update the price to \$29

Edit Comment Assign To Do In Progress Workflow Admin

Details

Type:	Bug	Status:	TO DO (View workflow)
Priority:	High	Resolution:	Unresolved
Labels:	None	100% ของผู้รายงาน	
Environment:	Browser Chrome 54.0.2840.71 Screen Size 1920 x 1200 Viewport Size 1607 x 920 Zoom L...		

Description

Summary: 100%
The price mentioned on the pricing page is not correct

Steps to Reproduce: 1. Go to the pricing page

Expected Results: คาดคะเน
The price for the basic plan should be \$29

Actual Results: ผลลัพธ์ที่ได้
The price for the basic plan is currently \$25

Source URL: <https://www.shopify.com/pricing>

Attachments แนกเด็กษาเพื่อนบ้าน

Drop files to attach, or browse.

People

Assignee: gary Assign to me

Reporter: Christophe Han

Votes: 0

Watchers: 1 Stop watching this

Dates

Created: 1 minute ago
Updated: 1 minute ago

Agile

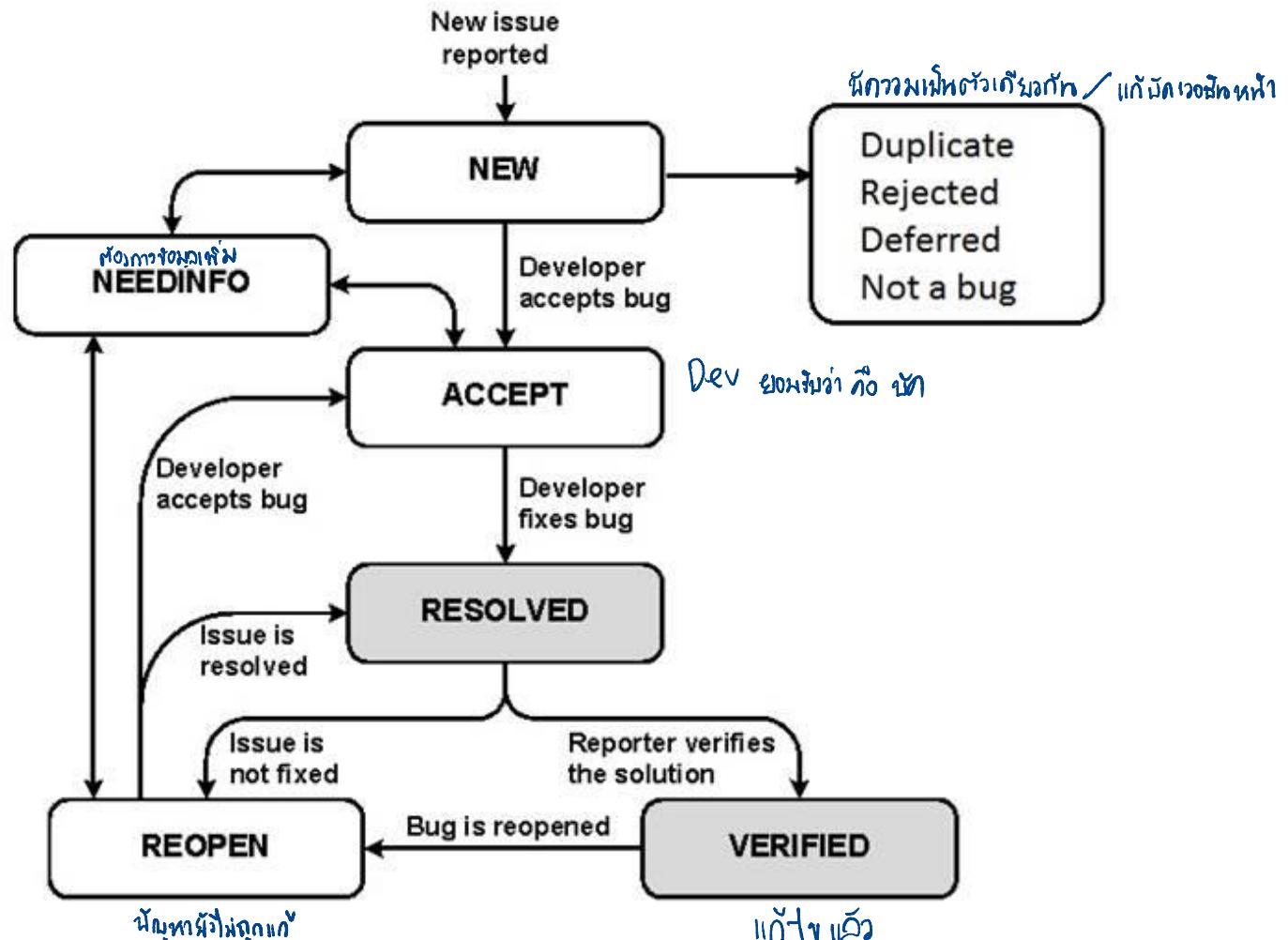
[View on Board](#)

HipChat discussions

Do you want to discuss this issue? Connect

Connect Dismiss

Defect Status Flow



Summary

Summary



- Test Techniques
 - Black Box Testing
 - White Box Testing
- Test Types
 - Unit Testing
 - Integration Testing
 - System Testing
 - Acceptance Testing
 - Performance Testing *
 - Test-First Development

“

Quality is never an accident;
it is always the result of
intelligent effort.

”

John Ruskin

つづく