**Software Development Lifecycles**

1. Planning
2. Analysis
3. Design
4. Implementation
5. Testing & Integration
6. Maintenance

แบบจำลองวงจรการพัฒนา software อธิบายกิจกรรมการดำเนินการในแต่ละขั้นตอนในโครงการพัฒนา software และกิจกรรมที่เกี่ยวข้อง logic และลำดับเวลา

**Sequential development**

**Originator have idea** -> requirement -> design -> build -> test and fix -> release -> **Customer(S)**

**SDLC**

* **Sequential**
* **Iterative & Incremental**

**Sequential Software Development**

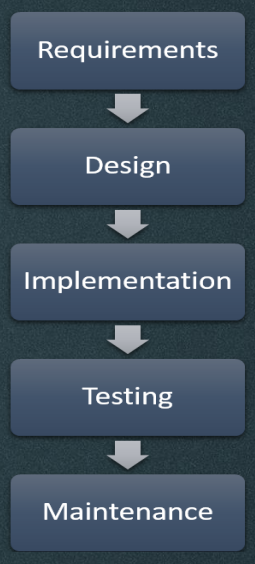
รูปแบบการพัฒนาแบบ sequential development model จะอธิบายการพัฒนา software ที่มีกระบวนการแบบเส้นตรงไหลตามลำดับกิจกรรม

หมายความว่ากระบวนการเฟสใดๆจะทำการก็ต่อเมื่อกระบวนการเฟสก่อนหน้าดำเนินการเสร็จสมบูรณ์

ในทางทฤษฎีจะไม่มีการทับซ้อนกันของเฟส แต่ในทางปฏิบัติมันจะมีประโยชน์ในส่วนที่จะได้รับ ข้อเสนอแนะจากเฟสก่อนหน้า

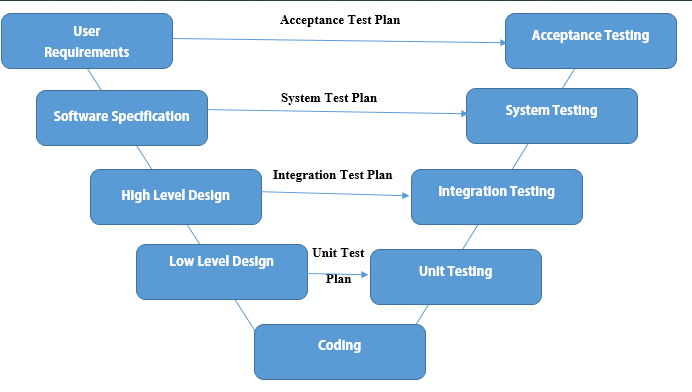
**Waterfall Model**

ในส่วนของ waterfall model กิจกรรมการ development จะสำเร็จก่อนกิจกรรมอื่น และกิจกรรมการ test จะเริ่มขึ้นเมื่อกิจกรรมการ development ดำเนินการเสร็จสมบูรณ์



**V-Model**

V-Model จะไม่เหมือน Waterfall Model เพราะจะมีกระบวนการทดสอบร่วมกับกระบวนการ development โดยใช้หลักการ early testing



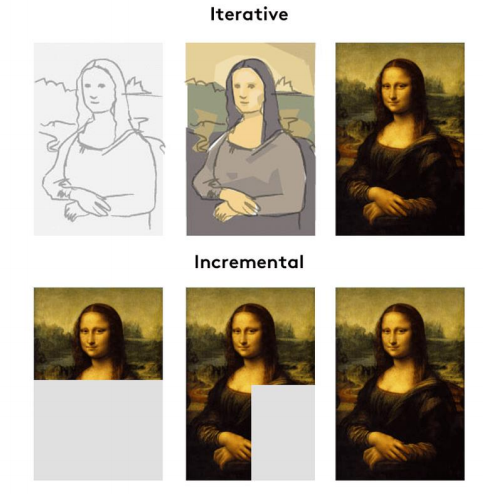
**Iterative & Incremental Model**

* Iterative

ตัวอย่าง คือ ส่วนของ โครงสร้าง สร้าง และเพิ่มรายละเอียด

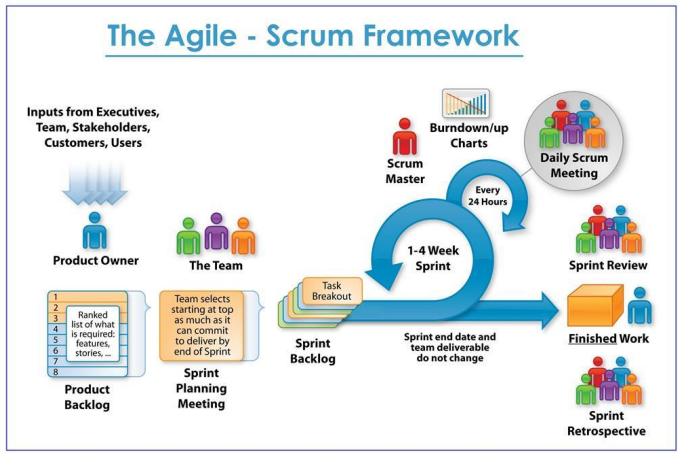
* Incremental Model

ตัวอย่าง คือ ส่วนของการสร้างทั้งหมดแล้วแบ่งเป็นส่วนๆ



**Scrum**

**The Agile Scrum Framework**



การวนซ้ำการทำงานจะสั้น เช่น ชั่วโมง วัน หรือ สองสามสัปดาห์ และ feature ที่พัฒนาจะเป็นส่วนที่เพิ่มขึ้นที่ละน้อย

**What is Software Testing?**

คนส่วนใหญ่มีประสบการณ์เกี่ยวกับ Software ที่ไม่ทำงานตามที่คาดการณ์ไว้ software ที่ทำงานไม่ถูกต้องสามารถนำไปสู่ปัญหามากมาย ได้แก่ สูญเสียเงิน เวลา หรือชื่อเสียงทางธุรกิจ บาดเจ็บ หรือเสียชีวิต

Software testing เป็นทางที่จะ ประเมินคุณภาพของ software และลดความเสี่ยงในการใช้งาน

การทดสอบ software(Software Testing) ไม่ใช่การดำเนินการทดสอบ(Test Execution)

การทดสอบ software(Software Testing) เป็นกระบวนการที่รวมกิจกรรมที่ต่างกันหลายๆแบบ

ส่วนการดำเนินการทดสอบ(Test Execution) เป็นแค่หนึ่งในกระบวนการเหล่านี้

**Software Testing**

* Dynamic Testing เป็นการ Test โดยใช้ User เอง เช่น การใส่ค่าต่างในหน้าเว็บ
* Static Testing ส่วนนี้จะเป็นส่วนของการอ่าน code โดยจะใช้ Tool ในการทำงาน จะทำตัวเหมือนเป็น Compiler ส่วนนี้จะเป็นการทำ Automate Test

**Software Testing**

* Validation กระบวนการตรวจสอบ Software ว่าทำงานตรง Requirement ที่ User ต้องการ

***Are we building the right product?***

* Verification กระบวนการตรวจสอบและควบคุมให้ Software ถูกออกแบบและพัฒนาอย่างถูกต้องตรงตาม Requirement Specification ของ User

***Are we building the product right?***

**Objectives of Testing**

1. Work-product Evaluation ประเมินงาน
2. Requirements Fulfillment ปฎิบัติตาม Requirement
3. Building Confidence สร้างความไว้ลางใจ
4. Finding Defects ค้นหาข้อบกพร่อง
5. Preventing Defect ป้องกันข้อบกพร่อง
6. Providing Information to Stakeholders ให้ข้อมูลกับผู้เกี่ยวข้อง
7. Reduce Risk ลดความเสี่ยง
8. Compliance with Raw ปฎิบัติตามกฏ
9. Objective may vary แจ้งวัตถุประสงค์

**Testing & Debugging**

1. Testing ทดสอบข้อผิดพลาดและส่งให้ developer
2. Debugging developer ทำการแก้ไขและทดสอบอีกครั้ง
3. Confirm Testing ยืนยันผลการทดสอบ

**Test Process**

ไม่มีกระบวนการทดสอบซอฟต์แวร์ที่เป็นสากล แต่มีชุดกิจกรรมทดสอบทั่วไปมากมาย แต่ที่มีมาตราฐานคือ ISTQB in the 2018 syllabus

1. Plan | Analysis, Strategy, Plan, Tool
2. Design |Test cases, Script, Scenarios, Environment
3. Execution |Test Report, Defect Tracking, Defect Analysis, Report

**Test Process Activities**

1. **Test Planning** | Write Test Plan
2. **Test Monitoring Control** | compare our actual progress to expected progress that I write inside the plan if there is a difference between them, we perform control activities or any corrective action that we should use in order to follow our plan
3. **Test Analysis** | Test condition, Test Scenario
4. **Test Design** | Test case
5. **Test Implementation** | Test suites, Test Environment
6. **Test Execution**
7. **Test Completion**

**Test Level**

Test Level เป็นกลุ่มของการทำ test ที่จะบริหารจัดการร่วมกัน

แต่ละ test level เป็นตัวอย่างของกระบวนการทดสอบ

Test Level มีความเกี่ยวเนื่องกับกระบวนการพัฒนา software (software development lifecycle)

* **Acceptance**
* **System |** full system
* **Integration |** all code
* **Unit |** module, class

1. **Component Testing**

หรือเรียกว่า unit หรือ module testing เจาะจงเฉพาะส่วน และสามารถทดสอบแยกกันได้ (ผู้รับผิดชอบ developer)

1. **Integration Testing**

เจาะจงเฉพาะปฎิสัมพันธ์ระหว่าง component หรือ system

* Component Integration (ผู้รับผิดชอบ developer) focuses on integration between components.
* System Integration (ผู้รับผิดชอบ Tester) have a big project ,many system for example web service have a backend database and frontend

1. **System Testing (Important)**

การทดสอบระบบมุ่งเน้นไปที่พฤติกรรมและความสามารถของทั้งระบบหรือผลิตภัณฑ์โดยมักพิจารณาจากงาน end-to-end(ตั้งแต่เริ่มต้นจนจบ) ที่ระบบสามารถทำได้และพฤติกรรมที่ไม่ทำงานได้ ที่จะแสดงในขณะปฏิบัติงานเหล่านั้น

* System testing often produces information that is used by stakeholders to make release decisions.
* The test environment should ideally correspond to the final target or production environment.
* System testing should focus on the overall, end-to-end behavior of the system as a whole.
* Independent testers typically carry out system testing.

1. **Acceptance Testing**

เหมือนกันกับการทดสอง system testing แตกต่างกันที่ ต้องหาข้อผิดพลาดให้เยอะที่สุดเท่าที่จะเป็นไปได้ เป็นหน้าที่ของ Tester เพื่อให้มั่นใจว่า software สร้างได้ถูกต้อง ส่วนมากจะเป็นหน้าที่ของ User หรือผู้ที่เกี่ยวข้อง

* Alpha testing ให้ลูกค้าหรือบริษัททดสอบในสถานที่ของเรา
* Beta testing ให้ลูกค้าหรือบริษัททดสอบในสถานที่ของบริษัทหรือที่บ้าน example : antivirus,window,office

**Testing Type**

1. **Functional Testing** ทดสอบอะไรที่ระบบทำได้ ทดสอบ Main function คำตอบที่ใช้คือ Yes/No เช่น ระบบ facebook มี function login ,function register
2. **Non-functional Testing** ทดสอบการทำงานของระบบ ยากที่จะให้คำตอบ Yes/No คำตอบที่ใช้คือ วัดเป็นช่วง เช่น ระบบจำนวนผู้ใช้งานที่จะไม่สามารถใช้งานได้ ขีดจำกัดของระบบ หรือการใช้งาน function login ใช้ความเร็วเท่าไหร่ในการเข้าใช้งาน
3. **Black-Box Testing** ทดสอบระบบโดยไม่ทราบ โครงสร้างภายในของระบบ เช่น ใส่ข้อมูล ได้ผลลัพธ์โดยไม่รู้เกี่ยวกับ code database หรือ APIs
4. **White-Box Testing** ทดสอบโดยดูตามโครงสร้างภายในของระบบ ต้องเข้าใจ programming language ที่ระบบใช้
5. **Dynamic Testing** ทดสอบโดยการเรียกใช้หรือ run Software
6. **Static Testing** ทดสอบโดยไม่ได้ run Software เช่น review requirement,review user stories
7. **Retesting(Confirmation Testing)** ทดสอบหลักจากที่ debugging เพื่อมั่นใจว่าข้อผิดพลาดได้รับการแก้ไข
8. **Regression Testing** ทดสอบในส่วนของที่ไม่ได้มีการเปลี่ยนแปลงเพื่อมั่นใจว่าไม่กระทบจากการเปลี่ยนแปลง เช่นมีการเปลี่ยนแปลงที่หน้า register และทำConfirmation Testing แล้ว ส่วนงานที่เกี่ยวข้อง คือหน้า login และ edit profile จะต้องทำ Reqression Testing จึงมีการใช้การทดสอบแบบ automate testing เพราะการทดสอบในส่วนนี้ประกอบด้วย test case จำนวนมาก เพื่อทำให้การทดสอบคลอบคลุมมากที่สุด
9. **Smoke Testing** ทดสอบ main functionเพื่อมั่นใจว่าโครงสร้างมีความเสถียรในการทดสอบต่อไป ใช้กับระบบที่มีขนาดใหณ่และหลาย feature เราไม่มีเวลาสำหรับการ regresstion testing จึงต้องการทดสอบฟังก์ชั่นหลักๆก่อนถ้าทดสอบไม่ผ่านจะต้องปรับแก้ก่อนที่จะทดสอบ function testing หรือ non-function testing ต่อไป

**Test Case Writing**

**What is a Test Case?**

ชุดเงื่อนไข ข้อมูลที่รับเข้า(input) การดำเนินการ(action ถ้ามี) และผลลัพธ์ที่ต้องการ เงื่อนไข โดยสร้างตามหลักของ เงื่อนไขการทดสอบ เป็นเอกสารสำคัญที่ต้องทำเมื่อเป็น software tester testcase จะช่วย guides ในระหว่างที่ทดสอบระบบ

เช่น การทดสอบ Login คุณจะได้รับ requirement หรือ UI interface หรือ Wireframes จาก develop หรือ designer และต้องทำ test case จากเอกสารเหล่านี้ และเมื่อทำ test design และเริ่มเขียน test case

1. **Test Case Title** ต้องตั้งให้เข้าใจง่ายและเฉพาะเจาะจง
   1. Verify Login with valid username & password
2. **Precondition** คือสิ่งที่เราจะพบก่อนการทำงานหรือ run testcase
   1. User is already registered using valid credentials
3. **Test Step** ขั้นตอนการดำเนินการจะต้องเข้าใจง่ายและเฉพาะเจาะจง
   1. Enter a valid username
   2. Enter a valid password
   3. Click on sign in
4. **Expected result** ผลลัพธ์ที่คาดหวัง
   1. User is Logged in successfully and redirected to (XYZ) page
5. **Test Scenario (Test Suite)**
   1. Login
6. **Test Environment** พูดถึง 3 อย่างคือ hardware software network
   1. Window 10 - Chrome - Wi-Fi
   2. Samsung Galaxy Note 10 - Android 10 -4G Network
   3. iPhone 11 - iOS 13.3.1 - 5G Network
7. **Actual Result** ผลลัพธ์ที่ได้
   1. Very importance note: Never fill the actual result field until you execute the test case
8. **Status** มี 4 สถานะหลักๆคือ

|  |  |
| --- | --- |
| New(Ready to test) | The test case is not executed |
| Pass | The test case is executed and actual result is the same as the expected result |
| Fail | The test case is executed and actual result is the different as the expected result |
| Blocked/Skipped | The test case can’t be executed |

**Fail vs Blocked/Skipped** คือ Fail คือ ทดสอบไม่ผ่าน ส่วน blocked/skipped คือไม่สามารถใช้งานได้

**Example Test Case**

|  |  |
| --- | --- |
| **Title** | Login with a valid username & password |
| **Precondition** | User is already registered using valid credentials |
| **Test Steps** | 1. Enter a valid username 2. Enter a valid password 3. Click on sign in |
| **Expected Result** | * User is logged in successfully * User is redirected to Home page |
| **Test Suite** | Login |
| **Test Environment** | Samsung Galaxy Note 10 - Android 10 -4G Network |
| **Actual Result** | Same as Expected |
| **Status** | Pass |

**Bug Report Writing**

**What is a defect report?**

เอกสารเกี่ยวกับเหตุการณ์ ลักษณะ และ สถานะของข้อบกพร่อง ความไม่สมบูรณ์หรือความบกพร่องในระบบงานที่ไม่เป็นไปตาม requirement หรือ specification [คำพ้องความหมาย: bug, fault]

1. Bug Report Title
   1. Section -> Description
   2. Register -> No error message appears when user leaves password field empty
2. Step to reproduce
   1. Must have specific
   2. Ex
      1. Open [www.facebook.com](http://www.facebook.com)
      2. Click on the hamburger icon in the upper left corner
      3. Scroll down to the bottom of the screen
      4. Click on Settings
      5. Click on data usage
      6. Change data usage to “minimum limit