# final\_part\_1

1. Agile น่าจะเหมาะกับงานที่ต้องติดต่อกับลูกค้าตัวจริงมากที่สุด Waterfall เป็นความผิดพลาดส่วนหนึ่งที่ช่วงแรกๆของการมี Software Development Project

นั้นเค้ามองว่าการพัฒนา Software จะเหมือนกับการสร้างทั่วไปๆ แต่คิดว่ามันก็เป็นวิวัฒนาการอย่างหนึ่งของเรา เมื่อมีปัญหากับ Waterfall เราก็สร้างสิ่งใหม่ๆขึ้นมาใช้แทน

สุดท้ายก็มาเป็น Agile Development ซึ่งก็ไม่แน่ว่าในอนาคตเมื่ออะไรๆเปลี่ยนไป Agile Development ก็อาจจะถูกแทนด้วยวิธีแบบอื่นๆอีก ทุกอย่างมีข้อดีข้อเสียของมัน

Waterfall เองก็ยังมีประโยชน์อยู่ในสถานการณ์บางอย่าง ซึ่งแน่นอนอาจจะถูกมองข้ามมากขึ้นเมื่อมีของใหม่อย่าง Iterative, Agile Development, หรือ Lean มาให้เลือก

Agile Model ไม่เหมาะกับ โครงการใหญ่ และไม่เหมาะกับ complex application อย่างยิ่ง

2. github เป็น web ที่ให้บริการจัดเก็บ source code (version control) ที่ใช้ version control system ที่ชื่อ git นอกจากนั้นก็มีพวกเสริม ส่วน svn,

cvs เป็น version control system ถ้าเทียบ git กับ svn, cvs ตัว git เป็น distributed แต่ svn, cvs เป็น centralizegit เราสามารถ clone

repository (ที่เก็บ source code) มาไว้ที่เครื่องเรา แล้ว commit งานในเครื่องได้ แล้วค่อย sync กับตัวที่เรา clone มาแต่ svn, cvs ต้องมี repository

กลางที่เราจะ commit source code ที่เราแก้ไขเข้าไป (repository กลางนี้ อาจจะเป็น server อื่น หรือ set เป็น local file ในเครื่องก็ได้)

3. git add README.md

git commit -m "first commit"

git remote add origin https://github.com/nattarika/final\_part\_1.git

git push -u origin master

4. Git ไม่ได้สร้าง merge commit อันใหม่ให้อัตโนมัติ มันกด pause เพื่อหยุดให้คุณ resolve merge conflict ถ้าคุณอยากดูว่า file ไหนบ้างที่ยังไม่ถูก

merge ณ เวลาใดๆหลังจากเกิด merge conflict ก็สามารถดูได้ด้วยคำสั่ง git status อะไรก็ตามที่มี merge conflict และยังไม่ถูก resolve จะถูก

list ออกมาว่า unmerged Git จะเติม conflict-resolution markers ลงไปใน files ที่มี conflicts เพื่อคุณจะได้เปิดมันและ resolve conflicts

เหล่านั้นได้

5. e"e".."e"

6. application มีเพื่อสามารถใช้งานได้หลายมากกว่า มันมีส่วนที่ติดต่อกับผู้ใช้ด้วย มันถึงเรียกว่าประยุกต์มาให้ใช้งานกับผู้ใช้ตรงๆ

Software ก็ใหญ่กว่าโปรแกรม เพราะมันรวมไปถึงข้อมูลต่างๆ จริงๆใช้งานส่วนไหนก็สามารถใช้งานได้ดีทั้งคู่

7. 1. เริ่มจาก Client ส่ง Request ไปที่ Web App ซึ่งจะถูกส่งต่อให้ Controller

ทำการตรวจสอบข้อมูลที่มาให้ (Request Method, Request Parameters)

2. แล้ว Controller จะเรียก Method ให้ทำงานเพื่อจัดการ Request นั้น

3. Model จะทำการคำนวณและอาจติดต่อกับ Database เพื่อจัดการกับ Request

นั้น แล้วส่งผลลัพธ์กลับไปที่ Controller

4. เมื่อ Controller ได้ผลลัพธ์จาก Model แล้วก็ใช้ผลลัพธ์นั้นส่งต่อให้ View ทำงาน

5. View จะสร้าง Page สำหรับแสดงผลลัพธ์นั้น แล้วส่ง page กลับไปที่ Controller

6. Controller ส่ง Page นั้น (เป็น Response) กลับไปยัง Client

8. เคยทำของรายวิชา Human and Computer Interaction เป็นการสร้างระบบใบลาต่างๆ

ข้อดีของ Rails

- Code เขียนง่าย ไม่ซับซ้อน เมื่อเทียบกับ java

- ถ้า Code สั้น ดังนั้นโอกาสในการเกิด Bug จึงน้อย

- เป็น Open source

- มี Library ให้นำไปใช้เป็นจำนวนมาก

- ง่ายต่อการนำไปขยายและพัฒนาต่อ (extend)

- เป็น pure Object Oriented้

ข้อเสียของ Rails

- ผู้พัฒนาต้องการเรียนรู้ตามโครงสร้างของ Rails

9. Heroku จะจัดการเรื่องเครื่องและโปรแกรมให้เราทั้งหมด หน้าที่เรามีแค่เอาเว็บแอพพลิเคชั่นไปใส่เพียงเท่านั้นและแน่นอนว่ามีแบบ free ให้เลือกใช้งานด้วย Heroku ช่วยในการพัฒนาApplicatin

ได้ค่อนข้างดีเลย ซึ่งเราใช้คําสั่งนี้ มันจะสร้าง app บน heroku ให้ และ add remote ลง config ของ git ให้เราเลย หลังจากนั้น เราก็ push ขึ้น heroku ได้เลยแล้วก็รอมัน setting

application ให้เราเท่านั้นแหละ พอมันเสร็จก็เปิดดูได้เลยน่ะ ด้วยคําสั่งนี้จบอย่างสวยงาม ก็จะเห็น application เราทํางาน

10.เพราะต้องการให้นิสิตเข้าใจถึงการทำงานของ Railframework ได้เข้าถึงมากขึ้น รู้จักการใช้งานของgithub ซึ่งไม่เคยใช้มาก่อนและสามารถใช้งานได้เข้าใจ