**1. ไม่เห็นด้วย**

Agile เน้นไปยังการปรับปรุงซอฟต์แวร์อย่างรวดเร็ว หรือเน้นการพัฒนาโดยรองรับการเปลี่ยนแปลงในอนาคต ทั้งนี้ไม่ได้หมายความว่าการพัฒนาด้วยวิธีนี้ไม่ได้มีการวางแผนใดๆ   
โดย Agile ได้ทำการแบ่งงานให้เป็นงานเล็กๆ และมีการวางแผนในช่วงสั้นๆ โดยได้แบ่งการทำงานออกเป็นช่วงสั้นๆ ในแต่ละช่วงการทำงานจะใช้รูปแบบวงจรการพัฒนาซอฟต์แวร์เต็มขั้น ซึ่งรวมถึงการวางแผน การเก็บข้อมูลความต้องการ การออกแบบซอฟต์แวร์ การพัฒนา และการทดสอบซอฟต์แวร์ และในเมื่อหมดช่วงการทำงานนั้นๆ ทีมผู้พัฒนาจะแสดงผลงานที่พัฒนาในช่วงสั้นๆ ให้ลูกค้าได้รับทราบ

ส่วน Waterfall คือการที่เราต้องไปจมอยู่กับแต่ละกล่องเป็นเวลานานๆ วันๆ ก็นั่งเขียนแต่ Requirement ไปเรื่อยๆ คุยกันจนได้ครบ แล้วก็เข้าไปขั้น Design เสียเวลาอีกนานๆ แล้วก็ลงไปทำขั้นถัดไปๆ เรื่อยๆ จนมาวันหนึ่ง test จบ เอามันไปให้ลูกค้าได้ดู ก็จะได้รายการบ่นจากลูกค้ามาชุดใหญ่ๆ ไปให้แก้ (ก็เพราะเราให้เค้าดูทีละเยอะๆ เองนิ ช่วยไม่ได้) Waterfall จริงๆ แล้วก็คือแค่การล้มเหลวในการสร้าง “ความพึงพอใจของลูกค้าที่มีต่อการส่งมอบซอฟท์แวร์ที่มีคุณค่าต่อลูกค้า ตั้งแต่ต้นอย่างต่อเนื่อง” เท่านั้นเอง

**2. เห็นด้วย**

CVS หรือ SVN ถือเป็น versioning control system ยุคแรก ที่มีการทำงานเป็นแบบทุกอย่างรวมอยู่ที่ศูนย์กลาง คือ ต้องมี server แม่ข่ายอยู่ 1 แห่ง และทุกคนที่ทำงานร่วมกันใน project เดียวกัน ก็สามารถแบ่งปันข้อมูล อัพเดทไฟล์งานของตนเอง และเห็นการทำงานของเพื่อนๆได้ โดยมีเครื่อง server แม่ข่ายนี้เป็นศูนย์กลาง ข้อเสียก็คือ หากวันใดที่ server นี้ มีปัญหา ไม่สามารถใช้งานได้ตามปกติ ก็จะทำให้การทำงานของ project ต้องมีเหตุให้หยุดชะงักลง

Git และ Github ซื่งปฏิวัติการงานแบบเดิมๆ โดยเปลี่ยนมาเป็นการทำงานแบบกระจาย (distributed) คือ ทุกคนเสมือนมีเครื่อง server อยู่ที่เครื่องของตนเอง ทำให้ปัญหาที่เครื่องศูนย์กลางการทำงานเป็นตัวปัญหาหมดไป ทุกคนสามารถทำงานบนเครื่องของตัวเองได้เสมอ (เว้นเสียแต่ว่าเครื่องตัวเองพังเสียเอง) เมื่อทำงานได้ปริมาณจนเป็นที่พอใจแล้ว จึงค่อยส่งไฟล์ของตนเองไปให้คนอื่นๆ ที่ทำงานร่วมกัน

**3.**

**4.** การรวม branch จะมีเหตุการณ์หนึ่งซึ่งอาจจะเกิดขึ้นได้ คือการชนกันของโค้ด เรียกด้วยภาษาทางเทคนิคเขาเรียก conflict เหตุการนี้จะเกิดขึ้นกับโปรเจ็กที่ร่วมกันทำหลายคนเท่านั้น ทำอยู่คนเดียวมันไม่เกิด เมื่อเกิด conflict โปรแกรมเมอร์จะต้องเป็นคนแก้ด้วยมือ

**5. a b c d e**

**6. ไม่เห็นด้วย** Web Application เหมาะกับองค์กรขนาดเล็กเพราะมีค่าใช้จ่ายต่ำกว่า (หลายตัวฟรี) และคิดค่าใช้จ่ายตามจำนวนการใช้งานจริง การใช้งานในองค์กรทำได้ง่าย เพียงแค่มีเว็บบราวเซอร์ซึ่งเป็นสิ่งพื้นฐานในคอมพิวเตอร์ปัจจุบันแทบทุกเครื่องก็ใช้งานได้ ข้อมูลจัดเก็บที่เดียว ง่ายต่อการจัดการ และไม่เกิดความซ้ำซ้อน ไม่ต้องการเครื่องคอมพิวเตอร์ประสิทธิภาพสูงซึ่งมีราคาแพง อยู่ที่ไหนก็ทำงานได้เพราะสามารถล๊อกอินเข้าใช้งานได้เลยไม่ต้องติดตั้งโปรแกรม ไม่ต้องมีบุคคลากรด้านเทคนิคเป็นของตัวเอง เพราะผู้ให้บริการดูแลเซิฟเวอร์และการบำรุงรักษาเองทั้งหมด ส่วนมากใช้ได้หลากหลายแพลทฟอร์มทั้ง Windows, Linux และ Mac ทำให้องค์กรสามารถเลือกใช้บางเครื่องเป็น Linux ได้เพื่อลดค่าใช้จ่ายด้านลิขสิทธ์ เชื่อมต่อกับเว็บแอพหรือบริการออนไลน์อื่นๆได้ง่าย

**7.**

**8. MVC (Model-View-Controller) คือ** สถาปัตยกรรมซอฟต์แวร์ชนิดหนึ่ง ซึ่งปัจจุบันมี Frameworks สำหรับสร้าง Web Apps จำนวนมาก แต่เกือบทั้งหมดมีโครงสร้างแบบ MVC ซึ่งแต่ละ Web Apps จะถูกแบ่งออกเป็นสามส่วน

1. Model คือ ส่วนที่ใช้ในการติดต่อกับฐานข้อมูล ซึ่งทำหน้าที่ในการดึงข้อมูลขึ้นมาจาก  
ฐานข้อมูลขึ้นมา และทำการจัดการข้อมูลต่างๆไว้ในรูปแบบที่เหมาะสม

2. View คือ ส่วนที่จะนำข้อมูลจาก Model ไปใช้แสดงผลให้ผู้ใช้เห็นผลลัพธ์ออกมาใน User Interface

3. Controller คือ ส่วนที่จะคอยรับ Input จาก Client เข้ามาแล้วนำคำสั่งไประมวลผล เพื่อสั่งงาน View และ Model ให้ประมวลผลออกมาเป็นอย่างไร

เมื่อเทียบกับ Rails แล้ว

**Rails** เป็น Web Framework มีลักษณะเป็น MVC (Model-View-Controller) พัฒนาโดย 37signals

Rails ถูกออกแบบมาให้มีการใช้งานที่ง่ายและรวดเร็ว ลดปัญหางานทีี่ต้องทำซ้ำๆ ทำให้ได้ productivity

ที่สูงขึ้น สอดคล้องกับ Methodology แบบ Ajile และมีแนวความคิดพื้นฐานคือ

1. Don't Repeat Yourself (DRY) อะไรที่อยู่แล้ว ก็ไม่มีความจำเป็นต้องทำสิ่งนั้นซ้ำ Reuseable นั้นเอง
2. Convention over configuration คือข้อตกลงที่ Rails สร้างขึ้นเป็นแบบแผน ที่กำหนดมา

ขอเพียงเราทำตามรูปแบบเหล่านั้นผลลัพธ์จะออกมาตามที่เราต้องการ โดยที่ไม่จำเป็นต้อง

ใช้การ configuration สำหรับคนที่ใช้ Convention ได้อย่างคล่องแคล่วและถูกต้องแล้วจะทำให้

การพัฒนาระบบได้เร็วขึ้นเพราะไม่ต้องมาทำการ configuration ให้ยืดยาว แต่มีข้อดีก็ต้องมีข้อเสียครับ เพราะ Convention ถ้าเป็นมือใหม่หัดเขียนคงต้องศึกษาและทำความเข้าใจกันมากซักหน่อย และมันอาจสร้างความประหลาดใจให้คุณได้

**9.** **Heroku** เป็น Platform as a Service (Paas) ที่ให้เราใช้งานได้ฟรี (มีแบบเสียเงินด้วย) โดยรองรับภาษาโปรแกรมที่หลากหลาย เช่น Ruby, PHP, Node.js, Python, Java, Clojure, Scala และยังสามารถสร้าง buildpack สำหรับภาษาอื่นๆได้ เช่น Lua ที่รันอยู่บน OpenResty ได้อีกด้วยนอกจากรองรับภาษาโปรแกรมที่หลากหลายแล้ว ตัว Heroku มี App Store ของมันด้วยเรียกว่า [add-ons](https://addons.heroku.com/) สำหรับเพิ่มเติมบริการอื่นๆเข้าไปในแอปของเรา เช่น PostgreSQL, MongoDB, Redis เป็นต้น ซึ่งก็มีทั้งฟรี และไม่ฟรีให้เลือกใช้งาน

**10.** เพราะวิชานี้จะเป็นการสอนให้นิสิตรู้จักกับการแบ่งงานทำที่เป็นระบบระเบียบและง่ายในการจัดการทำงานเป็นทีมแต่มีการแบ่งงานให้หลายๆคนทำโดยใช้ Github