การเปรียบเทียบแบบจำลองกระบวนการพัฒนาซอฟต์แวร์

A Comparison of Software Development Process Model

วรรณฤดี พึ่งเฮง[1]
วุฒิพงษ์ ชินศรี[2]
[1]นักศึกษาปริญญาตรี วิทยาลัยเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยรังสิต
[2]ผู้ช่วยศาสตราจารย์ ดร. วิทยาลัยเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยรังสิต
Email: wanrudee.ph@gmail.com

บทคัดย่อ

บทความฉบับนี้มีวัตถุประสงค์เพื่อ 1) ศึกษาแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ และ 2) เพื่อเปรียบเทียบแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ ได้แก่ Waterfall model V-shaped model Iterative model Agile model Spiral model และ Scrum โดยจากการศึกษาและเปรียบเทียบได้ข้อสรุปดังนี้ โมเดลที่ไม่มีความซับซ้อนในการทำงาน แต่ไม่มีความยืดหยุ่น คือ Waterfall model และ V-shaped model ส่วนตัวโมเดลที่มีการทำงานที่ซับซ้อนปานกลาง และสามารถยืดหยุ่นได้ คือ Iterative model และ Agile model ในส่วนของ Spiral model นั้นมีการทำงานที่ค่อนข้างซับซ้อน แต่มีความยืดหยุ่นสูง สามารถทำให้งานมีโอกาสสำเร็จได้มากขึ้น สำหรับ Scrum นั้นมีข้อดี คือสามารถส่งมอบงานให้ลูกค้าได้เร็ว และลูกค้าสามารถมีส่วนร่วมได้ แต่ก็ต้องใช้บุคลากรที่มีความสามารถในการทำงาน ทั้งนี้การเลือกใช้แบบจำลองการพัฒนาซอฟต์แวร์ จึงต้องพิจารณาให้เหมาะสม จะช่วยให้กระบวนการพัฒนาซอฟต์แวร์มีประสิทธิภาพ

คำสำคัญ: วิศวกรรมซอฟต์แวร์ แบบจำลองกระบวนการพัฒนาซอฟต์แวร์ การพัฒนาซอฟต์แวร์

ABSTRACT

The objectives of this article are 1) to study of software development process model, and 2) comparison of software development process model (Waterfall model, V-shaped model, Iterative model, Agile model, Spiral model and Scrum). The study and comparison are summarized as follows; Waterfall model and V-shaped model are not complicated to work but not flexible. Iterative model and Agile model is complicated but flexible. Spiral model is very complicated but very flexible, it can make project more successful. Scrum has the advantage of the ability to deliver to customers faster and customers can participate but it requires skilled personnel to work. However, selecting model must be considered appropriate. It will help the software development process to be effective.

KEYWORDS: Software Engineering, Software Development Process Model, Software Development

บทนำ

ปัจจุบันเทคโนโลยีมีความสำคัญต่อชีวิตประจำวันของทุกคน และได้กลายเป็นส่วนหนึ่งของการดำรงชีวิต อีกทั้งช่วยสนับสนุนการทำงานในด้านต่างๆ ไม่ว่าจะเป็น ด้านอุตสาหกรรมการแพทย์ การบริการต่างๆ ตลอดจนแผนพัฒนาดิจิทัลเพื่อเศรษฐกิจและสังคมของประเทศได้ (Rastogi, 2015) ได้มีการกล่าวถึงบทบาทและความสำคัญของเทคโนโลยีใน

ยุทธศาสตร์ต่างๆ ทำให้ทุกวันนี้องค์กรต่างๆ ไม่ว่าจะเป็นองค์กรขนาดเล็กหรือใหญ่ต่างจำเป็นที่ต้องปรับตัวโดยการนำเทคโนโลยีและซอฟต์แวร์คอมพิวเตอร์มาช่วยให้การทำงานของแต่ละองค์กรมีประสิทธิภาพมากขึ้น (Chandrasekar et al., 2014)

วิศวกรรมซอฟต์แวร์เป็นความรู้หนึ่งในด้านเทคโนโลยีสารสนเทศ ที่มีความสำคัญในการพัฒนาซอฟต์แวร์ โดยวิศวกรรมซอฟต์แวร์หมายถึงการประยุกต์ความรู้ทางวิทยาศาสตร์ในการออกแบบและสร้างโปรแกรมคอมพิวเตอร์รวมถึงการทำเอกสารที่เกี่ยวข้องเพื่อพัฒนาดำเนินการบำรุงรักษาโปรแกรมนั้นๆ (Laplante, 2007) โดยมีกระบวนการพัฒนาซอฟต์แวร์ที่สำคัญ คือ 1) Specification คือ กระบวนการ กำหนดคุณสมบัติของซอฟต์แวร์ให้ตรงกับความต้องการ 2) Development คือ กระบวนการพัฒนาซอฟต์แวร์ 3) Validation คือ กระบวนการตรวจสอบความถูกต้องของซอฟต์แวร์ เพื่อให้ตรงกับความต้องการ 4) Evolution คือ กระบวนการทำให้ซอฟต์แวร์มีวิวัฒนาการโดยปรับเปลี่ยนหรือเพิ่มสิ่งดีและเหมาะสมเข้ามาและเอาสิ่งที่ไม่ดีหรือไม่จำเป็นออกไป ซึ่งกระบวนการนี้จะเปลี่ยนแปลงไปตามเทคโนโลยีและความต้องการของตัวผู้ใช้เอง (Bhuvaneswari & Prabaharan, 2013)

แบบจำลองกระบวนการพัฒนาซอฟต์แวร์ (Software Development Process Model) เป็นส่วนหนึ่งของความรู้ด้านวิศวกรรมซอฟต์แวร์ ซึ่งช่วยให้ผู้พัฒนาซอฟต์แวร์สามารถสื่อสารและทำงานร่วมกันเพื่อพัฒนาซอฟต์แวร์ได้อย่างมีประสิทธิภาพ โดยสามารถเขียนในรูปของแผนภูมิหรือรูปภาพที่สามารถเข้าใจได้ง่าย โดยต้องมีส่วนสำคัญต่างๆ ดังนี้ 1) work flow คือ การให้ความสำคัญกับการลำดับงานเป็นหลัก 2) data flow คือ การให้ความสนใจกับข้อมูลเป็นหลัก 3) role action คือ การให้ความสนใจในบทบาทและหน้าที่ เช่น ใครทำหน้าที่อะไรบ้าง การปฏิบัติงานก่อนหลังหรือทำไปพร้อมกัน (Adolph & Kruchten, 2013)

เนื่องจากแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ มีหลายแบบจำลองให้เลือกใช้ ทำให้ผู้ใช้บางส่วนไม่สามารถเลือกแบบจำลองได้ตรงกับความต้องการ จึงส่งผลให้เกิดปัญหาระหว่างการพัฒนาซอฟต์แวร์ หรือทำให้การพัฒนาซอฟต์แวร์ไม่เป็นไปตามที่วางแผนเอาไว้ ผู้เขียนจึงมีแนวคิดที่จะทำการเปรียบเทียบแบบจำลองกระบวนการพัฒนาซอฟต์แวร์เพื่อให้ทราบถึงจุดเด่น จุดด้อย และแนวทางในการเลือกใช้แบบจำลองกระบวนการพัฒนาซอฟต์แวร์ให้เหมาะกับความต้องการของผู้ใช้งาน ซึ่งจะช่วยให้กระบวนการพัฒนาซอฟต์แวร์มีประสิทธิภาพมากยิ่งขึ้น

วัตถุประสงค์

1. เพื่อศึกษาแบบจำลองกระบวนการพัฒนาซอฟต์แวร์

2. เพื่อเปรียบเทียบแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ ได้แก่ Waterfall model V-shaped model Iterative model Agile model Spiral model และ Scrum

สรุปผลการศึกษา

1. การศึกษาแบบจำลองกระบวนการพัฒนาซอฟต์แวร์

ในวัตถุประสงค์นี้ผู้เขียนได้ทำการศึกษาแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ทั้งหมด 6 รูปแบบ ดังนี้

1.1 Waterfall model

เป็นโมเดลการพัฒนาซอฟต์แวร์ซึ่งเหมาะสำหรับงานที่เน้นคุณภาพมากกว่าต้นทุนหรือเวลางาน ที่มีข้อกำหนดชัดเจน งานที่ต่อยอดจากงานซอฟต์แวร์ตัวเก่า และงานที่ต้องการย้ายระบบซอฟต์แวร์เก่าเข้าสู่แพลตฟอร์มตัวใหม่ โดยมีจำนวนขั้นตอน 7 ขั้นตอน หรือ 5 ขั้นตอน แต่โดยส่วนใหญ่จะใช้อยู่ 5 ขั้นตอนหลักๆ (Royce, 1987) ซึ่งขั้นตอนการทำงานจะมีความคล้ายสายงานการผลิต มีการแบ่งขั้นตอนการทำงานต่างๆ อย่างชัดเจน ซึ่งจากบทความ Software engineering methodologies: a review of the waterfall model and object oriented approach (Adenowo & Adenowo, 2013) รูปแบบการพัฒนานี้ประกอบไปด้วยขั้นตอนต่างๆ ดังนี้

1) การกำหนดความต้องการ (Requirement definition)

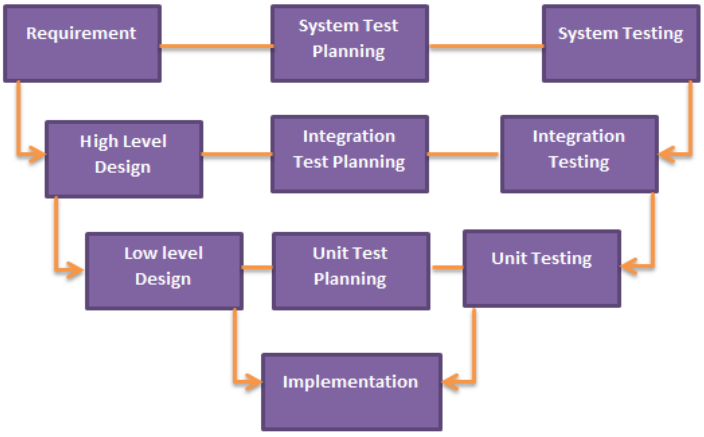
2) การออกแบบซอฟต์แวร์และระบบ (System and software design)

3) การลงมือทำและทดสอบในระดับหน่วย (Implementation and unit testing)

4) การประสานและทดสอบระบบ (Integration and system testing)

5) การนำไปใช้และการบำรุงรักษา (Operation and maintenance)

โดยการทำงานของ Waterfall model จะแสดงตามรูปที่ 1



รูปที่ 1 Waterfall model

1.2 V-shaped Model

เป็นโมเดลที่ถูกพัฒนาต่อมาจาก Waterfall model โดยใช้ในการตรวจสอบคุณภาพของระบบ ทดสอบและใช้ทดสอบส่วนต่างๆ ของการพัฒนา เนื่องจาก Waterfall model นั้นไม่ได้ให้ความสำคัญด้านการทดสอบ จึงได้มีการพัฒนาตัว V-shaped ขึ้นมาทำหน้าที่ในส่วนนี้ โดยที่ส่วนการพัฒนาจะอยู่ด้านซ้ายและส่วนของการทดสอบจะอยู่ด้านขวามือ โดยได้มีการอ้างอิงจากบทความในเอกสาร Advancements in the V-model (Mathur & Malik, 2010) ซึ่งมีขั้นตอนการทำงาน ดังนี้

1) User requirement → acceptance test: การทดสอบความสามารถในการตอบสนองความต้องการของผู้ใช้

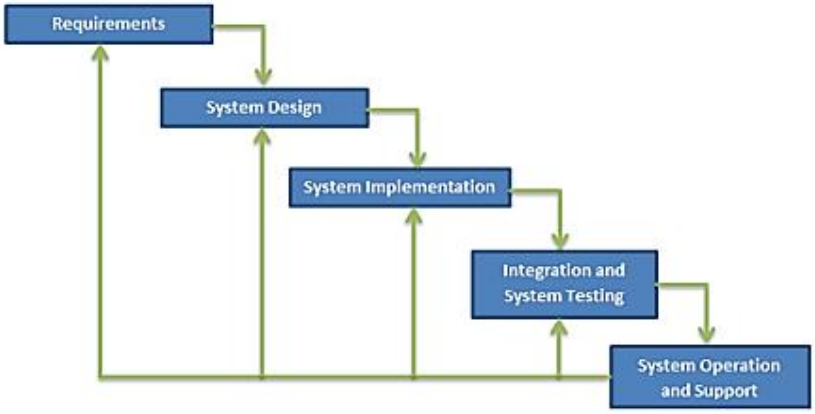
2) System requirement → system test: การทดสอบความต้องการของระบบ

3) Global design → integration test: การทดสอบแบบบูรณาการในการออกแบบ

4) Detailed design → unit test: การทดสอบการออกแบบในเชิงรายละเอียด

5) Coding: การพัฒนาระบบ

โดยการทำงานของ V-shaped model จะแสดงตามรูปที่ 2



รูปที่ 2 V-shaped model

1.3 Iterative model

เป็นโมเดลที่มีลักษณะเด่น คือ การเก็บข้อมูลความต้องการของผู้ใช้งานในตัวโปรแกรมครั้งแรก แล้วนำข้อมูลนั้นมาวิเคราะห์เพื่อพัฒนาต่อ เหมาะสำหรับโครงการที่มีความเสี่ยงปานกลางถึงต่ำ โครงการที่กำหนดระยะเวลาการทำงานที่ยาวนาน โครงการที่ทำงานกับเทคโนโลยีใหม่ๆ และใช้เทคโนโลยีใหม่ๆ เข้าช่วยในการพัฒนา จากบทความเรื่อง Iterative and incremental development: a brief (Larman & Basili, 2003) โดยมีขั้นตอนการทำงาน ดังนี้

1) Initial requirement and analysis: รวบรวมความต้องการของผู้ใช้โปรแกรมนำมาวิเคราะห์ พร้อมทั้งหาเครื่องมือที่ใช้ในการพัฒนาต่อ

2) Design: ทำการออกแบบโปรแกรมตามที่วิเคราะห์

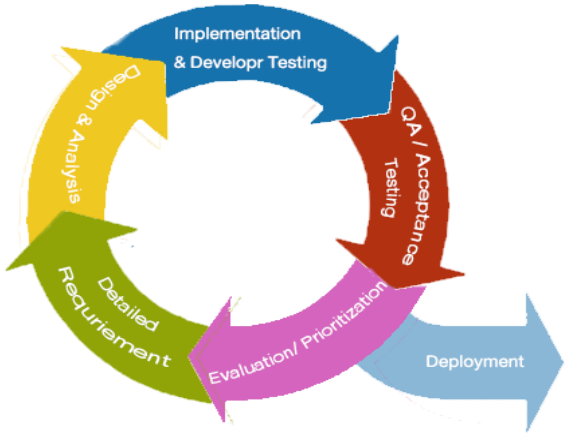
3) Testing: ทำการหาข้อผิดพลาดของโปรแกรมและตรวจสอบทั้งในด้านผู้พัฒนา และผู้ใช้งาน

4) More requirement and analysis: หากทำการทดสอบแล้วได้ผลที่ไม่น่าพอใจ จะนำการวิเคราะห์นั้นมาทำการออกแบบใหม่

5) Product: ถ้าผลการทดสอบเป็นที่น่าพอใจจะทำการใช้จริงในระบบ

6) Maintenance: พัฒนาโปรแกรมที่เคยใช้งานมาแล้ว หากพบปัญหาจะนำกลับเข้าสู่ขั้นตอนการพัฒนาอีกครั้ง ในขั้นตอนของ More requirement and analysis

โดยการทำงานของ Iterative model จะแสดงตามรูปที่ 3



รูปที่ 3 Iterative model

1.4 Agile model

แนวคิดของ Agile นั้นเกิดจากการพัฒนาเพิ่มเติมจากรูปแบบดั้งเดิม traditional method คือ ลักษณะการทำงานตามลำดับขั้นตอนตามข้อกำหนดที่ลูกค้าได้ทำการร้องขอมาที่ยังพบข้อเสียอยู่พอสมควร โดยเฉพาะในด้านความพึงพอใจของผู้ใช้ เวลาในการพัฒนา และงบประมาณ การนำ Agile model มาใช้เพื่อตอบสนองต่อความต้องการที่เปลี่ยนแปลงไปของผู้ใช้ อ้างอิงตามบทความในเอกสาร Waterfall vs. agile methodology (McCormick, 2012) โดยมีแนวคิดที่สำคัญหลักๆดังนี้ ไม่เน้นกระบวนการและเอกสาร ยอมรับความเปลี่ยนแปลง ทำงานทีละนิดแต่ทำอย่างสม่ำเสมอ ไม่กลัวที่จะผิดพลาด นำข้อผิดพลาดมาแก้ไข และสุดท้ายคือเน้นการทำงานเป็นทีมอ้างอิงจากบทความมาทำความรู้จักกับ “Agile and scrum” แนวคิดการทำงานของบริษัทยุคใหม่ (Tantisuttivet, 2016) ในส่วนของกระบวนการของ agile นั้นมีทั้งหมด 4 ขั้นตอน ดังนี้

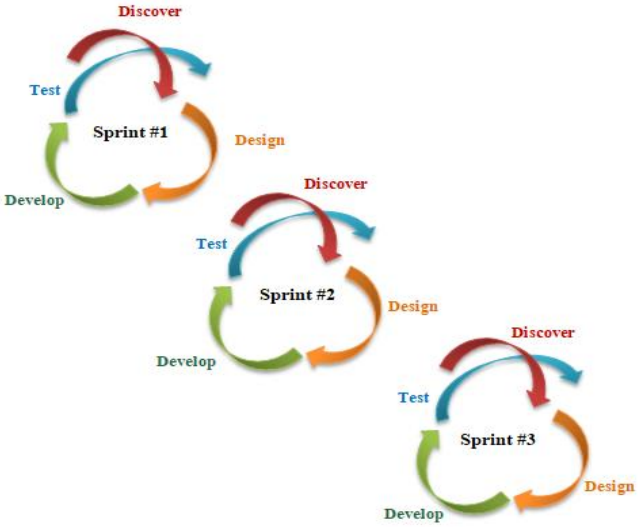
1) Individuals interactions over process and tool คือ การให้ความสำคัญกับบุคคลมากกว่ากระบวนการและเครื่องมือ

2) Working software over comprehensive documentation คือ การส่งมอบซอฟต์แวร์ที่นำไปใช้งานได้จริง มากกว่าการใช้เอกสารอธิบายรายละเอียดต่างๆ

3) Customer collaboration over contract negotiation คือ การให้ความร่วมมือกับลูกค้าในการพัฒนาตลอดระยะเวลา มากกว่าการต่อรองสัญญา

4) Responding to change over following a plan คือ การยอมรับการเปลี่ยนแปลง มากกว่าการทำตามแผนที่วางไว้

โดยการทำงานของ Agile model จะแสดงตามรูปที่ 4



รูปที่ 4 Agile model

1.5 The spiral model

เป็นโมเดลรูปแบบเกลียว มุ่งเน้นด้านประเมินความเสี่ยงและลดความเสี่ยงของโครงการ โดยแบ่งโครงสร้างออกเป็นโครงการกลุ่มย่อย ซึ่งจะช่วยปรับเปลี่ยนความต้องการได้ง่ายขึ้นในระหว่างขั้นตอนการพัฒนาตลอดจนประเมินความเสี่ยงและความต่อเนื่องของโครงการ เมื่อโครงการมีเรื่องค่าใช้จ่ายและความเสี่ยงเข้ามาเป็นปัจจัยสำคัญ อีกทั้งยังต้องคำนึงถึงการเปลี่ยนแปลงของเศรษฐกิจในระยะยาวให้เหมาะกับงานที่ผู้ใช้งานยังไม่แน่ใจในความต้องการของตนเองและมีข้อกำหนดที่ซับซ้อนหรืองานในสายการผลิตใหม่ โดย Spiral model จากหนังสือ Spiral development: experience, principles, and refinements (Boehm & Hansen, 2000) มีขั้นตอนการทำงาน ดังนี้

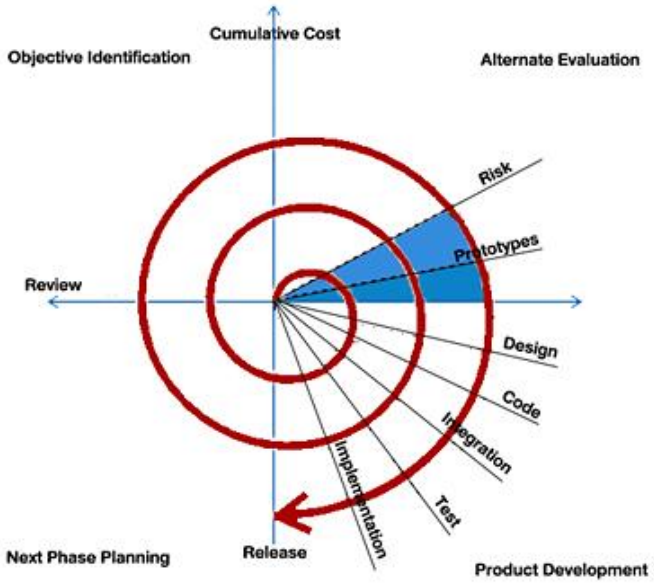
1) Planning: วางแผน และทำความเข้าใจในตัวโครงการระหว่างผู้ใช้งานและนักวิเคราะห์ระบบ

2) Risk analysis: วิเคราะห์ความเสี่ยง และแนวทางแก้ไขของโครงการ

3) Development/engineering: ดำเนินการพัฒนาและทดสอบ

4) Evaluation phase: ประเมินผลลัพธ์ของโครงการ

โดยการทำงานของ Spiral model จะแสดงตามรูปที่ 5



รูปที่ 5 Spiral model

1.6 Scrum

Scrum จะทำหน้าที่เป็น framework อย่างง่าย เพื่อให้เกิดประสิทธิภาพในการทำงานร่วมกันในทีมในงานที่ซับซ้อน โดย Schwaber & Sutherland ได้เขียน Scrum guide (Schwaber & Sutherland, 2016) เพื่ออธิบายกระบวนการไว้ ดังนี้

1) The scrum framework ใช้สำหรับการจัดการความซับซ้อนของการพัฒนางาน ทว่าไม่ใช่เทคนิคในการสร้างงาน แต่สามารถใช้กระบวนการและเทคนิคที่หลากหลาย เพื่อช่วยให้เกิดประสิทธิภาพในการจัดการและพัฒนามากขึ้น โดยเน้นไปที่ความสำเร็จของการใช้งานเป็นสำคัญ

2) The scrum team ประกอบไปด้วย

2.1) The Product owner

2.2) Development team

2.3) The Scrum master

3) Scrum events

3.1) Sprint คือ การทำงานอย่างมีประสิทธิภาพในระยะเวลา 30 วัน

3.2) Sprint planning ประชุมวางแผนงาน

3.3) Daily scrum พูดคุยว่าในวันก่อนหน้าได้ทำอะไรไปบ้าง

3.4) Sprint review meeting ในทุกๆ ขั้นตอนสุดท้ายของการทำงาน ทั้งทีมจะต้องทำการประชุม เพื่อทบทวนสิ่งเกิดขึ้น

3.5) Sprint retrospective เป็นการประชุมระหว่างทีมพัฒนา และเจ้าของโครงการ

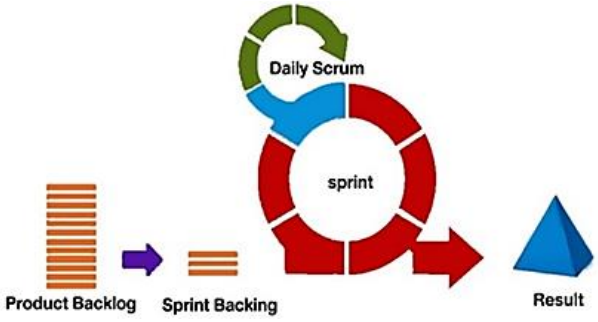
4) Scrum artifacts

4.1) Product backlog การเรียงลำดับความสำคัญของส่วนประกอบต่างๆ ในแต่ละงาน

4.2) Sprint backlog แผนการการพัฒนาที่คิดค้นโดยทีมพัฒนา

4.3) Increment คือ ส่วนผลรวมของรายการที่เพิ่มเข้ามาใน product backlog ระหว่างกระบวนการ Sprint

โดยการทำงานของ Scrum จะแสดงตามรูปที่ 6



รูปที่ 6 Scrum

2. การเปรียบเทียบแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ ได้แก่ Waterfall model V-shaped model Iterative model Agile model Spiral model และ Scrum

จากการศึกษาข้อมูลต่างๆ ตามหัวข้อที่ 1 ผู้เขียนได้ทำการเปรียบเทียบจุดเด่น จุดด้อย ของแต่ละแบบจำลอง ซึ่งแสดงดังตารางที่ 1 ซึ่งจากตารางที่ 1 จะบอกถึงจุดเด่นของในแต่ละโมเดล โดยที่จุดเด่นของ Waterfall model คือ เป็นโมเดลที่ไม่ซับซ้อน ใช้งานง่าย V-shaped model มีความชัดเจนในแต่ละส่วนของงาน และมีการทำการทดลองใช้จริงในแต่ละขั้นตอน Iterative model เหมาะสำหรับงานที่มีความเสี่ยงสูง และต้องการพัฒนางานเร็ว Agile model เหมาะสำหรับงานที่มีความเปลี่ยนแปลงบ่อย เป็นโครงการขนาดเล็ก Spiral model ช่วยในการลดความเสี่ยงของการล้มเหลวในโครงการ และสุดท้าย Scrum model นั้นข้อดีอย่างเด่นชัด คือ มีแผนการทำงานที่ชัดเจน เพื่อลดความซับซ้อนของการทำงาน และช่วยประหยัดเวลา

ตารางที่ 1 จุดเด่น – จุดด้อยในการทำงานของแต่ละรูปแบบจำลองกระบวนการพัฒนาซอฟต์แวร์

Model	จุดเด่น	จุดด้อย
Waterfall Model	1. มีรูปแบบที่เข้าใจง่าย ทำให้ง่ายต่อการจัดการเนื่องจากในแต่ละส่วนของโมเดลได้มีการกำหนดขั้นตอนอย่างเฉพาะเจาะจงและชัดเจน 2. โมเดลถูกออกแบบให้งานไม่มีการทับซ้อนกัน 3. เหมาะสำหรับงานขนาดเล็ก 4. เป็นรูปแบบที่ใช้งานกันอย่างแพร่หลาย	1. ไม่สามารถเปลี่ยนแปลงรูปแบบได้ 2. ผู้ใช้ระบบมีส่วนในการออกความเห็นได้น้อย อาจจะทำให้ไม่ตรงความต้องการของผู้ใช้งาน 3. ไม่เหมาะกับงานที่มีความซับซ้อน 4. มีความเสี่ยงและความไม่แน่นอน
V-shaped Model	1. ใช้งานง่าย 2. ในแต่ละส่วนงานมีความชัดเจน 3. มีโอกาสในการประสบความสำเร็จมากกว่า waterfall model 4. ทำงานได้มีประสิทธิภาพสำหรับงานขนาดเล็ก	1. มีความยุ่งยากกว่าแบบจำลอง waterfall 2. ใช้งบประมาณในการปรับเปลี่ยนค่อนข้างมาก 3. ซอฟต์แวร์ได้รับการพัฒนาในระหว่างขั้นตอนการใช้งาน ทำให้ไม่มีต้นแบบในการพัฒนาซอฟต์แวร์
Iterative Model	1. พัฒนางานที่มีความเสี่ยงสูงและมีความสำคัญ 2. มุ่งเน้นไปที่การพัฒนาขนาดใหญ่ 3. สามารถนำข้อผิดผลาดจากขั้นตอนก่อนหน้ามาพัฒนาในขั้นตอนต่อไปได้ 4. ผู้ใช้งานสามารถรับรู้ถึงส่วนสำคัญในช่วงต้นและสามารถมีส่วนร่วมในขั้นตอนต่างๆ 5. สามารถส่งมอบงานได้เร็ว 6. ลดความเสี่ยง และความล้มเหลวของงานหากมีการเปลี่ยนข้อกำหนด	1. ต้องมีการวางแผนและออกแบบที่ดี ก่อนเริ่มทำงาน 2. ต้องมีข้อกำจัดความและความต้องการที่ชัดเจนเพื่อที่จะสามารถกำหนดส่วนที่ต้องการเพิ่มเติมได้ 3. ไม่สามารถทำซ้ำในส่วนที่เพิ่มเติมได้
Agile Model	1. ใช้สำหรับงานที่มีความเปลี่ยนแปลงบ่อยและลดงบประมาณลง 2. ประหยัดเวลาในการทำงานของนักพัฒนา 3. เป็นโมเดลที่เหมาะสำหรับการออกแบบความต้องการที่มีการเปลี่ยนแปลงของผู้ใช้งาน 4. ความเปลี่ยนแปลงนั้นๆ กระทบต่องานน้อยที่สุด 5. เหมาะสำหรับโครงการที่มีขนาดเล็ก 6. มีผู้ตัดสินใจเพียงคนเดียวเพื่อให้งานดำเนินได้อย่างรวดเร็ว	1. ไม่เหมาะกับงานที่มีขนาดใหญ่ 2. ไม่เน้นการออกแบบ และการจัดทำเอกสาร 3. ผู้มีอำนาจ หรือนักพัฒนาอาวุโสจะเป็นคนตัดสินใจเพียงคนเดียว อาจจะสร้างความไม่พอใจขึ้นในทีมได้
Spiral Model	1. การวิเคราะห์ความเสี่ยงในระดับสูง 2. มีการเริ่มผลิตตัวซอฟต์แวร์ตั้งแต่เริ่มกระบวนการ 3. มีการอนุมัติที่น่าเชื่อถือและมีเอกสารการทำงานยืนยันที่ชัดเจน 4. สามารถเพิ่มฟังก์ชันการทำงานในภายหลังได้ 5. การตรวจสอบโครงการทำได้ง่ายและมีประสิทธิภาพ 6. สามารทำการตรวจสอบในแต่ละขั้นตอนได้ 7. เหมาะกับการพัฒนางานซอฟต์แวร์ที่มีการเปลี่ยนแปลงสูง	1. มีค่าใช้จ่ายค่อนข้างสูง 2. ต้องใช้ผู้เชี่ยวชาญในการประเมินความเสี่ยง 3. โครงสร้างมีความซับซ้อน เอกสารสำหรับโครงการก็มีความซับซ้อน 4. ใช้เวลาในการประเมินความเสี่ยงนาน 5. ความสำเร็จของโครงการขึ้นอยู่กับการประเมินความเสี่ยง
Scrum Model	1. สามารถวัดผลการทำงานได้ 2. มีการพูดคุย ประชุมกันในทีมสม่ำเสมอ ทำให้รู้ถึงปัญหา และสามารถแก้ไขได้ทัน 3. มีผังการทำงานที่เรียบง่าย	1. ไม่มีความยืดหยุ่น ไม่สามารถเปลี่ยนแปลงการทำงานได้ 2. บุคคลที่จะเข้ามาทำงานให้ตรงประเภทนั้นอาจจะหายาก

ตารางที่ 2 สรุปคุณสมบัติของรูปแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ทั้งหมด

Model/Feature	Waterfall	V-shaped	Iterative	Agile	Spiral	Scrum
ความซับซ้อนของแบบจำลอง	ไม่ซับซ้อน	ไม่ซับซ้อน	ปานกลาง	ปานกลาง	ซับซ้อน	ไม่ซับซ้อน
ขนาดของงาน	เล็ก	เล็ก-ปานกลาง	ใหญ่	เล็ก	ปานกลาง–ใหญ่	ปานกลาง–ใหญ่
ระยะเวลา	ใช้เวลาปานกลาง	ใช้เวลานาน	ใช้เวลาน้อย	ใช้เวลาน้อย	ใช้เวลาปานกลาง	ใช้เวลาน้อย
ผู้ใช้มีส่วนร่วมในกระบวนการ	ไม่มี	มี	มี	มีส่วนร่วมอย่างมาก	ไม่มี	มีส่วนร่วมอย่างมาก
สามารถเปลี่ยนแปลงความต้องการได้	ไม่ได้	ได้	ได้	ได้	ได้	ไม่ได้

จุดด้อยของในแต่ละรูปแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ โดยที่จุดด้อยของ Waterfall model คือ เป็นโมเดลที่ไม่สามารถเปลี่ยนแปลงได้ V-shaped model ใช้เวลาในการทำนาน และไม่สามารถเปลี่ยนแปลงได้ Iterative model ใช้เวลาในการทำงานนาน และต้องการข้อกำหนดงานที่ชัดเจน Agile model ไม่เหมาะสำหรับงานขนาดใหญ่ และมีความเสี่ยงในการล้มเหลวสูง Spiral model มีความซับซ้อนและจำเป็นต้องใช้ผู้เชี่ยวชาญในการออกแบบ และสุดท้าย Scrum model คือการที่มันไม่มีความยืดหยุ่น และบุคลากรที่เหมาะสมกับงาน

ในส่วนของผลการศึกษาและเปรียบเทียบนั้น ผู้เขียนได้ทำงานสรุปผล จุดเด่น – จุดด้อย ของแต่ละรูปแบบจำลองกระบวนการพัฒนาซอฟต์แวร์โดยสรุปจากคุณสมบัติเด่น ที่สามารถส่งผลถึงความสำเร็จของงาน และการตัดสินใจเลือกรูปแบบของรูปแบบจำลองกระบวนการพัฒนาซอฟต์แวร์ เพื่อให้เหมาะสมกับงาน ซึ่งแสดงในตารางที่ 2

จากตารางที่ 2 จะเห็นได้ชัดเจนถึงคุณลักษณะของโมเดลที่แตกต่างกันออกไป โดยสรุป คือ Waterfall model นั้นไม่ซับซ้อน เข้าใจง่าย แต่ไม่สามารถเปลี่ยนแปลงได้ ส่วน V-shaped model นั้นผู้ใช้งานสามารถเข้ามามีส่วนร่วมในการทดสอบระบบได้ตั้งแต่กระบวนการผลิต แต่ส่งผลให้ใช้เวลาในการพัฒนาค่อนข้างนาน Iterative model นั้นเหมาะสำหรับงานขนาดกลาง โดยผู้ใช้งานมีส่วนร่วมได้ และสามารถเปลี่ยนแปลงกระบวนการทำงานได้โดยที่ไม่มีผลกระทบในวงกว้าง Agile model เหมาะกับงานที่ต้องการผลลัพธ์โดยเร็วโดยที่ผู้ใช้งานมีส่วนในการแสดงความเห็นได้อย่างเต็มที่ สามารถเปลี่ยนแปลงตามความต้องการของผู้ใช้งานได้ตลอดเวลา ไม่เน้นเอกสารยืนยัน แต่ก็มีความเสี่ยงสูงที่โครงการจะล้มเหลว Spiral model ที่เน้นไปทางการประเมินความเสี่ยงของโครงการและเข้ามาช่วยแก้ปัญหาในโครงการที่มีความเสี่ยงว่าจะล้มเหลว และสุดท้ายคือ Scrum มีความเรียบง่ายไม่ซับซ้อนของการทำงาน แต่ก็ไม่สามารถยืดหยุ่นได้

การอภิปรายผล

ความเหมาะสมในการใช้งานรูปแบบจำลองกระบวนการพัฒนาซอฟต์แวร์นั้น จากการศึกษา และทำการเปรียบเทียบคุณสมบัติในแต่ละส่วน ผู้เขียนมีความเห็นว่า การเลือกนำมาใช้งานนั้น ขึ้นอยู่กับคุณลักษณะเด่นที่เข้ากับงานที่จะนำไปพัฒนา จากบทความ How to put the right methodology to the right job. (Piyorot, 2014) ได้กล่าวถึงประสบการณ์การใช้งาน Waterfall model Iterative model และ Agile model ว่าเหมาะกับการใช้งานในลักษณะต่าง ๆ ดังนี้

1) มีความต้องการที่ชัดเจนไม่เปลี่ยนแปลง (Requirement stability) เหมาะกับ Waterfall model เนื่องจากรูปแบบโมเดลไม่สามารถยืดหยุ่นได้

2) ความต้องการมีความสัมพันธ์หรือขึ้นอยู่กับความต้องการอื่น (Requirement dependency) เหมาะกับ Waterfall model เพื่อป้องกันการเปลี่ยนแปลงระหว่างการทำงาน

3) ความมีวินัยหรือมีระเบียบของทีมงาน (Staff discipline) เหมาะกับ Agile model เพราะสามารถยืดหยุ่นได้กับคนในทีมที่ต่างกันได้

4) ลูกค้ามีส่วนร่วมในการรีวิวสูง (Customer review) เหมาะกับ Agile model เพราะสามารถแก้ไขงานตามความต้องการของลูกค้าที่มีการเปลี่ยนแปลงบ่อยๆ

ส่วน Iterative model นั้นถือว่าอยู่ในระดับปานกลางมีความยืดหยุ่นแต่ไม่มากเท่า Agile model และอาจจะไม่ได้มีความมั่นคงเท่า Waterfall model แต่เป็นการนำ model ทั้ง 2 แบบ มาปรับเข้าหากันจนเป็น Iterative model ดังนั้น การเลือกใช้ Iterative model จึงไม่มีความชัดเจนเมื่อเทียบกับ Agile model และ Waterfall model

ในส่วนของการทำงานกับองค์กรขนาดใหญ่ ที่ต้องการให้ได้ผลลัพธ์ที่รวดเร็ว มีบุคลากรจำนวนมากในการมีส่วนร่วมกับงานนั้น Scrum จะมีความเหมาะสมด้วยกระบวนการการทำงานและตัวรูปแบบของการทำงานที่ไม่ซับซ้อน (Schwaber & Sutherland, 2016)

ในส่วนของข้อเสนอแนะนั้นผู้เขียนเห็นว่าการนำแบบจำลองไปใช้ในการพัฒนาซอฟต์แวร์นั้นต้องคำนึงถึงองค์ประกอบต่างๆ ไม่ว่าจะเป็น ความซับซ้อนของแบบจำลอง ขนาดของงาน ระยะเวลาที่กำหนด ลูกค้ามีส่วนร่วมในกระบวนด้วยหรือไม่ รวมไปถึงลูกค้าสามารถเปลี่ยนแปลงความต้องการได้หรือไม่ เมื่อพิจารณาประเด็นเหล่านี้แล้ว จะสามารถเลือกแบบจำลองที่เหมาะสม ซึ่งจะช่วยให้เกิดประสิทธิภาพสูงสุดในการพัฒนาซอฟต์แวร์และลดความเสี่ยงต่อการล้มเหลวในการพัฒนาซอฟต์แวร์ได้

สำหรับการเลือกแบบจำลองนั้น โดยปกติทีมพัฒนามักเลือกแบบจำลองเพียงแบบจำลองเดียวและใช้แบบจำลองนั้นจนการพัฒนาซอฟต์แวร์เสร็จตามที่กำหนด แต่อย่างไรก็ตามในความคิดเห็นของผู้เขียน การเลือกใช้งานแบบจำลองนั้นอาจปรับเปลี่ยนแบบจำลองตามสถานการณ์ที่เกิดขึ้นได้ ซึ่งจะช่วยให้เกิดความยืดหยุ่นในการพัฒนาซอฟต์แวร์ได้ ช่วยแก้ปัญหาที่เกิดขึ้นระหว่างการพัฒนาซอฟต์แวร์ และน่าจะเกิดประโยชน์กับทั้งทีมพัฒนาและลูกค้า แต่อย่างไรก็ตามการเปลี่ยนแบบจำลองตามสถานการณ์นั้น จำเป็นต้องทำความเข้าใจกับคนในทีมให้ชัดเจน และคนในทีมต้องมีความรู้ความเข้าใจในแบบจำลองต่างๆ ได้เป็นอย่างดี เพื่อไม่ให้เกิดความสับสน ซึ่งจะนำไปสู่ความผิดพลาดในการพัฒนาซอฟต์แวร์ได้

เอกสารอ้างอิง

Adenowo, A.A.A. and Adenowo, B.A. 2013. Software engineering methodologies: a review of the Waterfall Model and Object Oriented Approach. **International Journal of Scientific and Engineering Research.** 4(7): 427-433.

Adolph, S. and Kruchten, P. 2013. Generating a useful theory of software engineering. **In the 2nd SEMAT workshop on a General Theory of Software Engineering (GTSE)**, 26 May 2013, San Francisco, CA, USA, 47-50.

Bhuvaneswari, T. and Prabaharan, S. 2013. A survey on software development life cycle models**. International Journal of Computer Science and Mobile Computing.** 2(5): 262-267.

Boehm, B. and Hansen, W.J. 2000. **Spiral development: experience, principles, and refinements**. Pittsburgh: SEI Joint Program Office.

Chandrasekar, A., SudhaRajesh and Rajesh, P. 2014. A research study on software quality attributes. **International Journal of Scientific and Research Publications**. 4(1): 1-4.

Laplante, P.A. 2007. **What every engineer should know about software engineering.** Boca Raton: CRC press.

Larman, C. and Basili, V.B. 2003. Iterative and incremental development: a brief history. **Computer**. 36(6): 47–56.

Mathur, S. and Malik, S. 2010. Advancements in the V-model. **International Journal of Computer Applications.** 1(12): 29–34.

McCormick, M. 2012. **Waterfall vs. agile methodology**. MPCS, Inc.

Piyorot. 2014. **Waterfall .. Iterative .. Agile เลือกใช้อะไรดี? How to put the right methodology to the right job.** สืบค้นเมื่อวันที่ 20 พฤษภาคม 2560 จาก Medium: https://medium.com/agile-development - in- thai/ waterfall- iterative- agile- 97106 f0b6934.

Rastogi, V. 2015. Software development life cycle models-comparison, consequences. **International Journal of Computer Science and Information Technologies.** 6(1): 168-172.

Royce, W.W. 1987. Managing the development of large software systems. **In the 9th International Conference on Software Engineering**, 30 March 1987 - 2 April 1987, Monterey, California, USA, 1-9.

Schwaber, K. and Sutherland, J. 2016. **The scrum guide**. USA: Scrum. Org and ScrumInc.

Tantisuttivet, P. 2016. **มาทำความรู้จักกับ "Agile and Scrum" แนวคิดการทำงานของบริษัทยุคใหม่.** สืบค้นเมื่อวันที่ 20 พฤษภาคม 2560 จาก Brandinside: https://brandinside.asia/ agile-and-scrum-for-new-business/.