บทที่ 3
การพัฒนาแบบอาไจล์
(Agile Development)

การพัฒนาแบบอาไจล์ (Agile Development) เป็นคำเรียกกลุ่มของกระบวนการพัฒนาซอฟต์แวร์ที่มีลักษณะร่วมบางอย่างที่แสดงให้เห็นถึง “ความว่องไว” (Agility) เกี่ยวกับการเปลี่ยนแปลง การวางแผน การสื่อสาร และการเรียนรู้ โดยรูปแบบของกระบวนการพัฒนาในกลุ่มนี้มักมีจุดเด่นตามแบบจำลองเชิงวนรอบและแบบจำลองเชิงเพิ่ม

3.1 คำแถลงการณ์อาไจล์

ในปี ค.ศ. 2001 ผู้คิดค้นกระบวนการพัฒนาซอฟต์แวร์จำนวน 17 คน ได้พูดคุยเพื่อร่วมกันหาจุดร่วมจากแต่ละกระบวนการพัฒนาของตนเอง โดยสรุปออกมาเป็นคำแถลงการณ์ของการพัฒนาซอฟต์แวร์แบบอาไจล์ (Manifesto for Agile Software Development) ดังนี้ (AgileManifesto.org, 2001)

“เรากำลังเผยวิธีการพัฒนาซอฟต์แวร์ที่ดีขึ้นจากการได้ลงมือทำเองและช่วยผู้อื่นทำ ที่ได้ทำมานั้นเราให้คุณค่าแก่:

บุคคลและการติดต่อ	*มากกว่า กระบวนการและเครื่องมือ*
ซอฟต์แวร์ที่ทำงานได้	*มากกว่า เอกสารที่ละเอียด*
ความร่วมมือของลูกค้า	*มากกว่า การเจรจาด้วยสัญญา*
การตอบสนองต่อการเปลี่ยนแปลง	*มากกว่า การทำตามแผนงาน*

นั่นคือ ในขณะที่สิ่งทางขวามือมีคุณค่า เราให้คุณค่าสิ่งทางซ้ายมือมากกว่า”

3.2 หลักการของอาไจล์

กลุ่มผู้คิดค้นการพัฒนาซอฟต์แวร์แบบอาไจล์ได้ตั้งหลักการของอาไจล์ (Agile Principles) ไว้จำนวน 12 ข้อ เพื่ออธิบายคุณค่าที่กล่าวไว้ในคำแถลงการณ์ให้ละเอียดและเป็นรูปธรรมมากขึ้น ดังนี้ (AgileManifesto.org, 2001)

1) อาไจล์ให้ความสำคัญสูงสุดในการทำให้ลูกค้าพึงพอใจผ่านทางการนำส่งซอฟต์แวร์ที่ใช้งานได้อย่างต่อเนื่องตั้งแต่ช่วงแรกของการพัฒนา โดยหลังจากมีการเตรียมร่างของโครงการเรียบร้อยแล้ว การพัฒนาแบบอาไจล์จะหาวิธีนำส่งซอฟต์แวร์ให้ได้เร็วที่สุดเท่าที่จะทำได้ โดยวิธีการเช่นนี้สามารถเข้าไปแทนที่การจัดการความต้องการและระยะการออกแบบที่พบได้ในกระบวนการเดิมๆ วิธีการทางอาไจล์จะ

เหมือนการทำต้นแบบในกระบวนการอื่นๆ แต่จะแตกต่างตรงที่เป็นการสร้างซอฟต์แวร์ที่ใช้จริงได้แทนการสร้างต้นแบบแล้วทิ้ง การพัฒนาซอฟต์แวร์แบบอาไจล์จะใช้กลยุทธ์การพัฒนาเชิงเพิ่ม (Incremental) และลดความเสี่ยงของโครงการด้วยการส่งมอบอย่างรวดเร็วและต่อเนื่อง เพื่อให้ลูกค้าสามารถตรวจทาน ทวนสอบการตัดสินใจและสมมติฐานของโครงการว่าเป็นไปตามที่ต้องการหรือไม่ รวมทั้งสามารถช่วยตรวจสอบประโยชน์การใช้งานจากซอฟต์แวร์ที่ส่งมอบในแต่ละครั้งได้อย่างต่อเนื่อง

2) กระบวนการแบบอาไจล์ ยินดีที่จะเปลี่ยนแปลงความต้องการแม้ในช่วงหลังการพัฒนาซอฟต์แวร์เพื่อให้ได้ประโยชน์สูงสุดแก่ลูกค้า หลักการนี้เป็นหัวใจสำคัญของอาไจล์ที่สะท้อนจากชื่อของกระบวนการซึ่งแสดงถึงความคล่องตัวโดยยอมรับว่าการเปลี่ยนแปลงเป็นสิ่งที่ต้องเกิดขึ้นและหลีกเลี่ยงไม่ได้ อาไจล์จึงตั้งปรัชญาที่ยอมให้เกิดการเปลี่ยนแปลงตลอดกระบวนการพัฒนาแทนความพยายามในการปิดกั้นหรือควบคุมไม่ให้เกิดการเปลี่ยนแปลงเพื่อให้เกิดประโยชน์ต่อลูกค้ามากที่สุด

3) ส่งมอบซอฟต์แวร์ที่ใช้งานได้อย่างสม่ำเสมอภายในช่วงเวลาไม่กี่สัปดาห์หรือไม่กี่เดือน โดยให้ช่วงเวลาสั้นที่สุดเท่าที่จะเป็นไปได้ หลักการข้อนี้เป็นการขยายความการส่งมอบซอฟต์แวร์อย่างต่อเนื่องจากหลักการแรก โดยเน้นว่าแต่ละรอบของโครงการที่ใช้อาไจล์ควรจะสั้นที่สุดเท่าที่จะสั้นได้ และเน้นต่อไปอีกว่าระยะเวลาหนึ่งหรือสองสัปดาห์นั้นไม่ได้สั้นจนเกินไปสำหรับรอบการพัฒนา หลักการนี้ยังตั้งเงื่อนไขเวลามากสุดไว้ที่ 2-3 เดือน ซึ่งเมื่อเปรียบเทียบกับระยะในการพัฒนาซอฟต์แวร์โดยทั่วไปแล้วรอบของการพัฒนามักจะไม่ต่ำกว่า 3 เดือน

4) บุคลากรทางธุรกิจและนักพัฒนาต้องทำงานด้วยกันทุกวันตลอดโครงการ เพราะวิธีการประเภทอาไจล์ต้องการการติดต่อสื่อสารระหว่างทีมพัฒนาและลูกค้าตลอดเวลา โดยบทบาทที่สำคัญที่สุดของโครงการอยู่ที่ลูกค้า รวมทั้งผู้มีส่วนในโครงการซึ่งมีบทบาทอื่นก็มีความจำเป็นที่จะต้องพูดคุยกับทีมพัฒนาเป็นประจำ

5) ตั้งโครงการรอบๆ บุคคลที่มีความตั้งใจ รวมทั้งเตรียมสภาพแวดล้อมที่สนับสนุนบุคคลในทีม และมีความเชื่อมั่นว่าทีมจะทำงานได้สำเร็จ หลักการนี้ของวิธีการแบบอาไจล์สร้างอยู่บนสมมติฐานที่ว่าบุคลากรในทีมพัฒนามีความทุ่มเท ตั้งใจ และได้รับการสนับสนุนจากองค์กรเป็นอย่างดีเพื่อส่งเสริมให้ทำงานได้อย่างมีประสิทธิภาพสูงสุด ในขณะเดียวกันกระบวนการแบบอาไจล์จะเน้นการสร้างสภาพแวดล้อมการทำงานที่กระตุ้นให้ทีมพัฒนาแต่ละคนเรียนรู้และจัดการด้วยตัวเองอย่างเป็นธรรมชาติที่สุดเท่าที่จะทำได้

6) วิธีการที่มีประสิทธิภาพและประสิทธิผลมากที่สุดในการได้มาซึ่งข้อมูลสำหรับการพัฒนา คือ การพูดคุยกันตัวต่อตัว หลักการนี้ของอาไจล์แสดงถึงเหตุผลในการลดขั้นตอนการทำงานด้านเอกสารลง ในกระบวนการแบบอาไจล์จะใช้การสื่อสารแบบตัวต่อตัวเป็นกลไกหลักในการแลกเปลี่ยนข้อมูลกัน โดยมีเครื่องมือ เช่น กระดานไวท์บอร์ดช่วยในการบันทึก จากนั้นข้อมูลที่ได้อาจจะบันทึกเป็นเอกสารในรูปแบบที่เรียกว่า “ตัวแผ่ข้อมูล” (Information Radiation) ซึ่งผู้เกี่ยวข้องกับข้อมูลนั้นๆ ต้องสามารถเข้าถึงได้ทันทีเมื่อต้องการ ตัวอย่างของตัวแผ่ข้อมูลที่ใช้กัน เช่น การบันทึกข้อมูลลงในกระดาษโน้ตที่แยกเป็นสีๆ แล้วแปะไว้บนกระดานไวท์บอร์ด เป็นต้น

7) ใช้ซอฟต์แวร์ที่ทำงานได้เป็นตัวหลักในการวัดความก้าวหน้าของโครงการ หลักการนี้เป็นอีกหลักการหนึ่งของอาไจล์ที่ระบุไว้เพื่อเน้นว่าซอฟต์แวร์สำคัญกว่าเอกสารประกอบ เนื่องจากโครงการส่วนใหญ่เน้นการทำงานด้านเอกสารข้อกำหนดความต้องการซอฟต์แวร์และระบุให้เป็นไมล์สโตน (Milestone) เพื่อแสดงถึงความก้าวหน้าของโครงการ แต่ในกระบวนการแบบอาไจล์นั้นจะให้คุณค่ากับซอฟต์แวร์ที่เป็นผลลัพธ์มากกว่าเอกสารประกอบ และเนื่องจากมีการส่งมอบซอฟต์แวร์ที่ทำงานอย่างต่อเนื่องตั้งแต่ช่วงต้นของโครงการ ซอฟต์แวร์จึงเป็นตัววัดความก้าวหน้าที่สมเหตุสมผล

8) กระบวนการแบบอาไจล์สนับสนุนการพัฒนาซอฟต์แวร์ที่ยั่งยืน โดยผู้สนับสนุนโครงการ ผู้พัฒนา และลูกค้าควรจะรักษาอัตราการทำงานให้คงที่ต่อเนื่อง ในหลายองค์กรมีการทำงานนอกเวลาจนเป็นเรื่องธรรมดา และอีกหลายองค์กรต้องการการทำงานล่วงเวลาอยู่บ่อยครั้ง แต่สำหรับกระบวนการแบบอาไจล์จะเน้นให้ตารางเวลาคงที่และยั่งยืน นั่นคือ การวางแผนงานไว้ที่ 40 ชั่วโมงต่อสัปดาห์ ก็จะต้องใช้เวลา 40 ชั่วโมง ให้ได้ประสิทธิภาพที่สุดและไม่ทำเกินแผนที่วางไว้ ตามหลักการนี้กระบวนการแบบอาไจล์จะเน้นให้รักษาการทำงานให้คงที่สม่ำเสมออย่างเป็นธรรมชาติ

9) สนใจพัฒนาความเป็นเลิศทั้งทางเทคนิคและการออกแบบอย่างต่อเนื่องเพื่อเพิ่มความคล่องตัวในการทำงานให้ดียิ่งขึ้น หลักการนี้เน้นให้สมาชิกในทีมพัฒนาแบบอาไจล์ปรับปรุงคุณภาพและทักษะให้สูงขึ้นเรื่อยๆ เพื่อให้แน่ใจได้ว่างานที่พัฒนาออกมาจะมีคุณภาพอยู่ในระดับสูง โดยงานของนักพัฒนาไม่ใช่เพียงแค่การเขียนโปรแกรมแล้วมีผู้อื่นเป็นคนตรวจสอบเพียงอย่างเดียว แต่นักพัฒนาจะต้องเป็นกลุ่มคนที่รับผิดชอบหลักในการส่งมอบสิ่งที่ถูกต้องและทดสอบแล้วว่าตรงตามความต้องการของลูกค้า อย่างไรก็ตามการตรวจสอบโดยบุคคลอื่นก็ยังเป็นสิ่งจำเป็นเช่นกัน

10) ความเรียบง่ายเป็นสิ่งจำเป็นยิ่งยวด กระบวนการแบบอาไจล์เน้นการใช้กระบวนการพัฒนาซอฟต์แวร์ที่เบาที่สุดเท่าที่จะเป็นไปได้ อย่างไรก็ตามความเรียบง่ายไม่ได้หมายถึงความมักง่าย

11) ทั้งสถาปัตยกรรม ข้อกำหนดความต้องการ และการออกแบบ จะปรากฏออกมาจากทีมพัฒนาเอง การจัดการด้วยตนเองในทีมเป็นจุดเด่นของกระบวนการแบบอาไจล์ซึ่งตรงข้ามกับการควบคุมจากภายนอกที่พบเห็นในกระบวนการอื่นๆ อย่างไรก็ตามการสร้างทีมที่สามารถจัดการตัวเองได้นั้นจำเป็นต้องมีการเปลี่ยนแปลงจากระดับองค์กร การเปลี่ยนแปลงในลักษณะนี้ทำได้ยากและจำเป็นต้องให้บุคลากรเรียนรู้ทักษะและพฤติกรรมใหม่ แต่หากทำได้แล้วจะได้ผลลัพธ์ในเชิงบวกอย่างมีนัยสำคัญ

12) ในแต่ละช่วงเวลาทีมพัฒนาต้องสามารถประเมินได้ว่าจะเพิ่มประสิทธิภาพในการทำงานได้อย่างไร จากนั้นจึงปรับพฤติกรรมให้เป็นไปตามการประเมิน กระบวนการแบบอาไจล์มีการปรับใช้กิจกรรมตามหลักการนี้เพื่อกระตุ้นให้เกิดการปรับปรุงกระบวนการพัฒนา กระบวนการแบบอาไจล์บางแบบมีการบังคับให้ทำการประเมินทุกๆ ระยะของการพัฒนา ส่งผลให้เกิดการเรียนรู้และปรับวิธีการทำงานหลายครั้งภายในระยะเวลา 1 ปี

3.3 แบบจำลองกระบวนการแบบอาไจล์ (Agile Process Model)

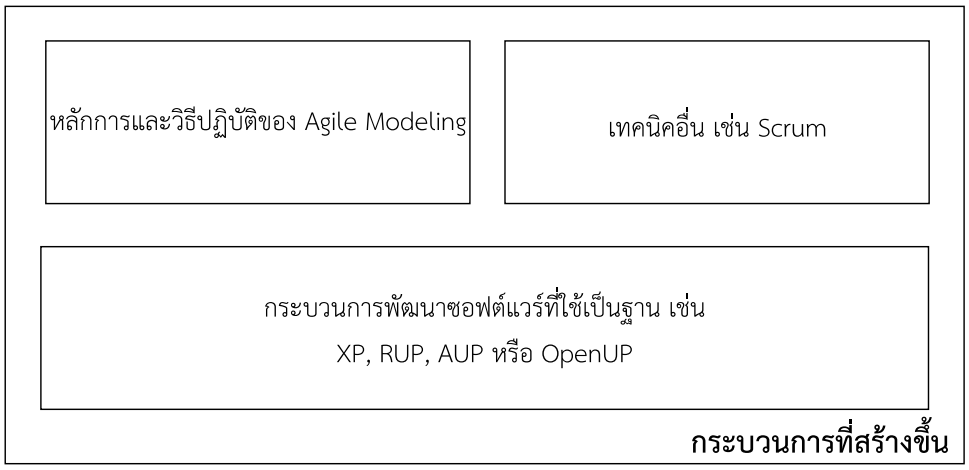
กระบวนการพัฒนาแบบอาไจล์ที่ได้รับความนิยมในปัจจุบันมีด้วยกันหลายกระบวนการดังนี้

3.3.1 การพัฒนาแบบสกรัม (Scrum Development)

การพัฒนาแบบสกรัมเป็นกระบวนการที่คิดค้นโดย Sutherland และทีมในช่วงต้นของทศวรรษ 1990 ในยุคหลังสกรัมได้รับการพัฒนาต่อโดย Schwaber และ Beedle หลักการของสกรัมเป็นไปตามคำแถลงการณ์ของอาไจล์และหลักการเหล่านี้ถูกใช้เพื่อเป็นแนวทางให้กับกิจกรรมต่างๆ ในการพัฒนา โดยอยู่ในรูปแบบของกระบวนการเป็นรอบ เรียกว่าสปรินต์ (Sprint) โดยงานที่นำเข้าไปทำในแต่สปรินต์นั้นจะนำมาจากกลุ่มของความต้องการเชิงซอฟต์แวร์ที่เรียกว่าโปรดักแบ็คล็อก (Product Backlog) และระหว่างกระบวนการจะมีการประชุมกันทุกวัน วันละประมาณ 15 นาที เรียกว่าการประชุมสกรัมรายวัน (Daily Scrum Meeting) (Schwaber, 2004)

3.3.2 การจำลองแบบเชิงอาไจล์ (Agile Modeling)

การจำลองแบบเชิงอาไจล์ เป็นวิธีการสำหรับสร้างแบบจำลองและเอกสารของระบบซอฟต์แวร์โดยใช้แนวทางของอาไจล์ ซึ่งสามารถใช้เป็นตัวเสริมให้กับกระบวนการอื่นๆ เช่น เอ็กซ์ตรีมโปรแกรมมิง (Extreme Programming) กระบวนการยูนิฟายด์เชิงอาไจล์ (Agile Unified Process) หรือสกรัม โดยการจำลองแบบเชิงอาไจล์เน้นกลุ่มของคุณค่า หลักการ และวิธีปฏิบัติสำหรับการสร้างแบบจำลองของซอฟต์แวร์ ซึ่งสามารถประยุกต์ใช้กับโครงการได้อย่างหยืดหยุ่นและมีประสิทธิภาพ รูปที่ 3.1 อธิบายการใช้การจำลองแบบเชิงอาไจล์ร่วมกับวิธีการอื่น เช่น เอ็กซ์ตรีมโปรแกรมมิงหรือกระบวนการยูนิฟายด์ เพื่อสร้างกระบวนการพัฒนาซอฟต์แวร์ขึ้นเองให้สอดคล้องกับความต้องการ (Ambler, 2002)



รูปที่ 3.1 *การนำการจำลองแบบเชิงอาไจล์ไปใช้เพื่อสร้างกระบวนการอื่น (Ambler, 2002)*

3.3.3 กระบวนการยูนิฟายด์เชิงอาไจล์ (Agile Unified Process)

กระบวนการยูนิฟายด์เชิงอาไจล์เป็นการประยุกต์ใช้ระยะการพัฒนาที่พบในกระบวนการยูนิฟายด์แบบเดิม ได้แก่ อินเซ็พชัน เอลาโบเรชัน คอนสตรักชัน และทรานสิชัน เข้ากับปรัชญาแบบอาไจล์โดยแต่ละรอบจะมีกิจกรรมดังต่อไปนี้ (Ambler, 2005)

1) การจำลองแบบ (Modeling) เป็นการสร้างแบบจำลองธุรกิจและโดเมนปัญหาด้วย UML โดยแบบจำลองเหล่านี้สร้างเพียงแค่ให้ “พอใช้ได้” เพื่อให้ทีมสามารถทำส่วนถัดไปได้อย่างต่อเนื่อง
2) การอิมพลีเมนต์ (Implementation) เป็นการแปลงแบบจำลองเป็นต้นรหัส (Source Code) ของโปรแกรม
3) การทดสอบ (Testing) เป็นการทดสอบเพื่อให้แน่ใจว่าต้นรหัสเป็นไปตามข้อกำหนด
4) การส่งมอบ (Deployment) เป็นการส่งมอบ “ส่วนเพิ่ม” ของซอฟต์แวร์ที่พัฒนาเสร็จแล้วเพื่อรวบรวมผลตอบรับจากผู้ใช้
5) การปรับแต่ง (Configuration) และการบริหารโครงการ (Project Management) ในบริบทของการยูนิฟายด์เชิงอาไจล์เป็นการจัดการการเปลี่ยนแปลงและความเสี่ยง รวมทั้งการติดตามงานและควบคุมดูแลความก้าวหน้าของโครงการ
6) การจัดการสภาวะแวดล้อม (Environment Management) เป็นการจัดการเพื่ออำนวยความสะดวกให้กับกระบวนการรวมถึงการใช้มาตรฐาน เครื่องมือ และเทคโนโลยีสนับสนุนอื่นๆ

3.3.4 เอ็กซ์ตรีมโปรแกรมมิง (Extreme Programming)

เอ็กซ์ตรีมโปรแกรมมิงเป็นวิธีการพัฒนาซอฟต์แวร์แบบอาไจล์ที่ใช้กันมากระดับหนึ่ง ในเอ็กซ์ตรีมโปรแกรมมิงมีการนิยามคุณค่า (Value) 5 ประการ ขึ้นมาเป็นรากฐาน ได้แก่

1) การสื่อสาร (Communication)
2) ความเรียบง่าย (Simplicity)
3) การตอบรับ (Feedback)
4) ความมีวินัย (Discipline)
5) การเคารพ (Respect)

โดยคุณค่าทั้ง 5 ประการนี้เป็นตัวขับเคลื่อน “กิจกรรม” “การกระทำ” และ “งาน” ในกระบวนการของเอ็กซ์ตรีมโปรแกรมมิง (Beck & Andres, 2004)

3.3.5 กระบวนการยูนิฟายด์แบบเปิด (OpenUP)

กระบวนการยูนิฟายด์แบบเปิดหรือที่เรียกกันว่า โอเพนยูพี เป็นกระบวนการพัฒนาซอฟต์แวร์ที่สร้างขึ้นโดยสกัดส่วนหลักของกระบวนการยูนิฟายด์ออกมา และปรับใช้ร่วมกับปรัชญาทางอาไจล์ซึ่งเน้นการพัฒนาซอฟต์แวร์เชิงความร่วมมือ อย่างไรก็ตาม โอเพนยูพียังคงลักษณะสำคัญของกระบวนการยูนิฟายด์ซึ่งเป็นการพัฒนาเชิงเพิ่มและยังมีการใช้ยูสเคสในการขับเคลื่อนการพัฒนา นอกจากนี้ยังคงการจัดการความเสี่ยงตามแบบของกระบวนการยูนิฟายด์ไว้ด้วย รวมทั้งยังคงใช้แนวทางที่มีสถาปัตยกรรมเป็นศูนย์กลาง (Kroll & MacIsaac, 2006)

3.4 บทสรุป

บทนี้ได้กล่าวถึงปรัชญาเบื้องหลังแนวคิดของการพัฒนาแบบอาไจล์ซึ่งเน้นการทำงานและความร่วมมือระหว่างบุคคล ซอฟต์แวร์ที่ใช้งานได้ และการตอบสนองต่อการเปลี่ยนแปลง ซึ่งสะท้อนให้เห็นความว่องไวตามชื่อของกระบวนการ นอกจากนี้ยังได้กล่าวถึงหลักการ 12 ข้อของอาไจล์ และกระบวนการพัฒนาแบบอาไจล์ประเภทต่างๆ ได้แก่ การพัฒนาแบบสกรัม การจำลองแบบเชิงอาไจล์ กระบวนการยูนิฟายด์เชิงอาไจล์ เอ็กตรีมโปรแกรมมิง และกระบวนการยูนิฟายด์แบบเปิด โดยบทถัดไปจะกล่าวถึงรายละเอียดของการพัฒนาแบบสกรัม

3.5 คำถามท้ายบทที่ 3

1) การพัฒนาซอฟต์แวร์แบบอาไจล์คืออะไร

2) ใจความหลักในคำแถลงการณ์ของการพัฒนาซอฟต์แวร์แบบอาไจล์เน้นอะไรบ้าง ให้เปรียบเทียบเชิงคุณค่า

3) หลักการของอาไจล์มีอะไรบ้าง

4) หลักการของอาไจล์มีสาระสำคัญเกี่ยวกับเวลาการทำงานอย่างไร

5) หลักการของอาไจล์มีสาระสำคัญเกี่ยวกับการพัฒนาตนเองอย่างไร

6) ผู้อ่านคิดว่าอะไรคือสิ่งสำคัญที่สุดตามหลักการของอาไจล์เพราะเหตุใด

7) แบบจำลองกระบวนการแบบอาไจล์มีอะไรบ้าง

8) ค้นคว้าแบบจำลองกระบวนการแบบอาไจล์ที่พบในปัจจุบันเพิ่มเติมจากอินเตอร์เน็ต