บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ระบบการรับสมัครนักศึกษา มหาวิทยาลัยราชภัฏเชียงใหม่ ผ่านเครือข่ายอินเตอร์เน็ต มีทฤษฎีและงานวิจัยที่เกี่ยวข้องและจำเป็นต่อการพัฒนาโปรแกรมคอมพิวเตอร์ ทั้งทางด้านแนวคิดและทางด้านเทคนิคในการวางระบบฐานข้อมูล โดยมีรายละเอียดของทฤษฎีและงานวิจัยที่เกี่ยวข้องดังต่อไปนี้

2.1 การบริหารโครงการ (Project Management)

2.2 การวางแผนและจัดตารางงานโครงการ (Project Planning and Scheduling)

2.3 แผนการพัฒนาซอฟต์แวร์ (Software Development Plan: SDP)

2.4 เทคนิคการวางแผน (Planning Techniques)

2.5 สมการในการวิเคราะห์และประเมินเวลาที่ต้องใช้ในแต่ละกิจกรรมของโครงการ

2.6 กระบวนการผลิตซอฟต์แวร์แบบเอ็กซ์ทรีมโปรแกรมมิ่ง (Extreme Programming: XP)

2.7 ไอเอสโอ 12207 (ISO 12207) มาตรฐานสำหรับกระบวนการผลิตและพัฒนาซอฟต์แวร์

2.1 การบริหารโครงการ (Project Management)

กิตติ และพนิดา (2550) การบริหารโครงการ (Project Management) หมายถึง การประยุกต์ใช้องค์ความรู้ ทักษะ เครื่องมือ และเทคนิค เพื่อดำเนินกิจกรรมตามความต้องการของโครงการให้บรรลุวัตถุประสงค์ที่กำหนดไว้ โครงการที่กล่าวถึงในที่นี้คือ โครงการผลิตซอฟต์แวร์ จำเป็นต้องอาศัยการบริหารโครงการที่มีประสิทธิภาพ เนื่องจากโครงการเป็นงานที่ต้องดำเนินการภายใต้ข้อจำกัดหลายอย่าง ไม่ว่าจะเป็นแรงงาน ต้นทุน และเวลา หากการบริหารโครงการบกพร่องจะส่งผลเสียต่อโครงการอย่างมาก กล่าวคือ อาจทำให้ส่งมอบซอฟต์แวร์ไม่ทันเวลา ใช้ต้นทุนเกินที่คาดการณ์ไว้ และซอฟต์แวร์ไม่มีคุณภาพ ไม่ตรงตามข้อกำหนดความต้องการ

2.2 การวางแผนและจัดตารางงานโครงการ (Project Planning and Scheduling)

การวางแผนโครงการ เป็นการกำหนดกิจกรรมหลัก กิจกรรมย่อย เป้าหมายของแต่ละกิจกรรม (Milestone) การส่งมอบงาน และจัดตารางงานโดยการกำหนดเวลาเริ่มต้นและส่งมอบงาน แผนงานดังกล่าวจะใช้เป็นแนวทางการดำเนินงานโครงการให้ได้ตามเป้าหมายของโครงการ

เมสินี นาคมณี (2545) การวางแผนโครงการถือเป็นส่วนสำคัญในการบริหารจัดการโครงการเมื่อมีการริเริ่มและคัดเลือกโครงการที่ต้องการพัฒนาเสร็จสิ้นแล้ว ขั้นตอนต่อไปคือ การวางแผนโครงการ เพื่อให้สามารถวิเคราะห์งานที่จะเกิดขึ้น และสามารถดำเนินการพัฒนา ติดตาม และสามารถทำให้โครงการดำเนินไปตามที่คาดหวังไว้ได้

เมสินี นาคมณี (2547) ในการวางแผนงานนั้น เมื่อได้เรียนรู้ในส่วนของกระบวนการ และประมาณการซอฟต์แวร์แล้ว สิ่งที่ต้องทำการเรียนรู้ต่อไป คือ การจะนำกรรมวิธีและขั้นตอนที่ได้กล่าวถึงนั้นมาผนวกรวมกับค่าที่ได้ประมาณการ และนำมาใช้ในการวางแผนงาน โดยแผนงานที่นำมาใช้ในการพัฒนาซอฟต์แวร์นั้น โดยหลักแล้วจะให้ความสนใจในแผนงานที่เป็นแผนโครงการ แต่ทั้งนี้ยังมีแผนงานอื่นที่เป็นแผนงานย่อยๆ เข้ามาเกี่ยวข้องอีก โดยมีประเภทของแผนงานต่างๆ ดังนี้

แผนงานโครงการ (Project Plan) จะมองภาพรวมในการพัฒนาซอฟต์แวร์ทั้งหมด โดยสามารถแบ่งย่อยออกได้เป็น 2 ลักษณะนั้น คือ

1) แผนงานในการพัฒนาซอฟต์แวร์ในเบื้องต้น (Preliminary Project Plan) โดยเป็นการจัดทำซอฟต์แวร์เบื้องต้นที่รู้จักกันในชื่อของพรีลิมินารีนั้นเอง

2) แผนงานการพัฒนาซอฟต์แวร์ (Project Development Plan) ซึ่งจะมองภาพรวมในการพัฒนาซอฟต์แวร์ทั้งหมด

แผนงานตรวจรับซอฟต์แวร์ (Project Acceptance Plan) เป็นแผนงานเพื่อตรวจสอบรับซอฟต์แวร์ที่ได้รับการพัฒนาโดยเป็นแผนสำหรับการจัดหาแนวทางและกรรมวิธีในการตรวจสอบและเงื่อนไขในการยอมรับซอฟต์แวร์ที่ได้ทำการพัฒนาว่า ตรงตามความต้องการหรือไม่

แผนบริหารความเสี่ยง (Risk Management Plan) ซึ่งจะจัดทำสำหรับโครงการเพื่อที่จะทำการประเมินความเสี่ยง และหาทางบริหารความเสี่ยงที่อาจจะเกิดในระหว่างการพัฒนาโครงการ

แผนการจัดการคุณภาพ (Quality Management Plan) แผนในการบริหาร และจัดการคุณภาพซอฟต์แวร์ โดยนอกเหนือจากการวางแผนเพื่อควบคุมให้การพัฒนาซอฟต์แวร์บรรลุเป้าหมายแล้ว ยังต้องการวางแผนเพื่อบริหารจัดการให้ซอฟต์แวร์ที่ทำการผลิตนั้น มีคุณภาพตามที่ต้องการอีกด้วย

2.3 แผนการพัฒนาซอฟต์แวร์ (Software Development Plan: SDP)

แผนการพัฒนาซอฟต์แวร์ ทำหน้าที่เสมือนเป็นผลลัพธ์จากการวางแผนงานเป็นเอกสารที่ช่วยในการสื่อสารระหว่างทุกฝ่ายที่เกี่ยวข้อง และช่วยเป็นเสมือนข้อตกลงในการดำเนินงาน และนอกจากนั้นยังนำไปใช้ในการติดตามและตรวจสอบอีกด้วยเช่นกัน โดยองค์ประกอบที่จัดไว้ในแผนการพัฒนาประกอบด้วย

กรรมวิธีในการพัฒนา (Process Life Cycle Model) ในแผนการพัฒนาซอฟต์แวร์ควรมีการระบุกรรมวิธีในการพัฒนาที่จะนำมาใช้ในการพัฒนาระบบงาน

การประมาณการ (Estimation) การประมาณการต่างๆ ของซอฟต์แวร์ และอ้างอิงไปยังข้อมูลที่นำมาใช้ในการประมาณหรือวิธีที่ใช้ประมาณ หรือผู้ที่ทำการประมาณ

โครงสร้างงานแยกย่อย (Work Breakdown Structure: WBS) ที่ได้จัดทำไว้

ระยะเวลา (Schedule) ระยะเวลาหรือแผนงานในรูปแบบแกนต์ชาร์ต หรือเพิร์ธชาร์ต

บุคลากร และลักษณะองค์กร (Staff and Organization) สามารถระบุแผนผังองค์กรสำหรับโครงการ (Project Organization Chart) พร้อมหน้าที่รับผิดชอบ

สิ่งแวดล้อมในการพัฒนา (Development Environment) ส่วนของสิ่งแวดล้อม หรือองค์ประกอบที่ต้องจัดไว้เพื่อรองรับการพัฒนา เช่น สถานที่และอุปกรณ์

สิ่งที่ต้องจัดส่ง (Deliverables) แนวทางในการจัดส่ง กรรมวิธี และลักษณะของซอฟต์แวร์ที่พัฒนาเสร็จแล้วพร้อมจะนำส่ง

การวางแผน และการบริหารความเสี่ยง (Risk Management and Planning) แผนการบริหารและจัดการความเสี่ยงของโครงการ หรือแผนงานอื่นๆ เช่น แผนในการจัดอบรม แผนในการ

ควบคุมคุณภาพ ซึ่งทั้งนี้อาจจะไม่รวมอยู่ในเอกสารฉบับเดียวแต่มีการอ้างอิงถึงในเอกสารแผนการพัฒนาหลัก

2.4 เทคนิคการวางแผน (Planning Techniques)

เทคนิคในการวางแผน คือ การนำกรรมวิธีในการพัฒนาซอฟต์แวร์ผนวกกับการประมาณการซอฟต์แวร์ที่ได้ประเมินไว้มารวมกันกับโครงสร้างองค์กร และหน้าที่รับผิดชอบที่ได้กระจายไว้ตามในโครงสร้างงานแยกย่อย และนำแผนงานนั้นมาจัดทำในส่วนของแผนงาน โดยเทคนิคที่ใช้ในการจัดสร้างแผนงานนั้นคือ เรื่องของแกนต์ชาร์ต และเพิร์ธชาร์ต ซึ่งเป็นเทคนิคที่ใช้ช่วยในการวางแผนให้เห็นการกระจายงานและระยะเวลาที่ต้องใช้ได้ชัดเจนขึ้น

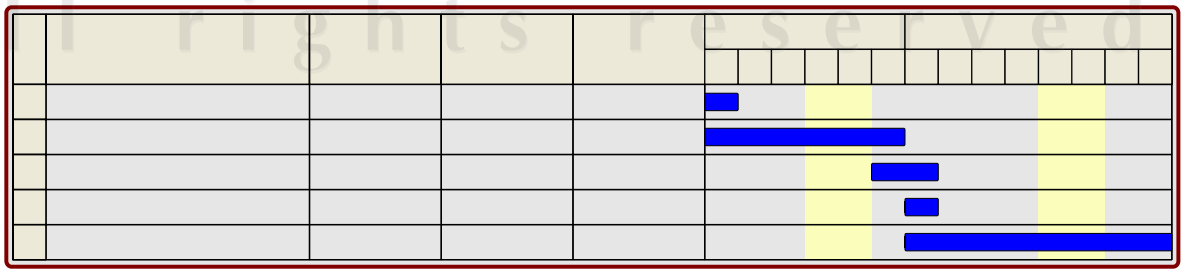
แกนต์ชาร์ต (Gantt Chart) ลักษณะสำคัญของแกนต์ชาร์ต คือ การแสดงงานในลักษณะของแท่งแนวนอน ที่มีการแสดงให้เห็นในส่วนของเวลาที่เริ่มต้น (Start Date) และวันสิ้นสุดในการทำงานแต่ละงาน (Finish Date) ทั้งนี้แกนต์ชาร์ตจะช่วยให้สามารถวางแผนได้ง่าย และนอกจากนั้นยังสามารถมองเห็นการกำหนดหน้าที่รับผิดชอบ จึงช่วยแสดงการกำหนดงานในขั้นตอนที่สมบูรณ์แล้ว ดังตาราง 2.1 จะเห็นได้ว่างานที่เสร็จสิ้นแล้วจะมีสีทึบ

ตาราง 2.1 แกนต์ชาร์ต (Gantt Chart)

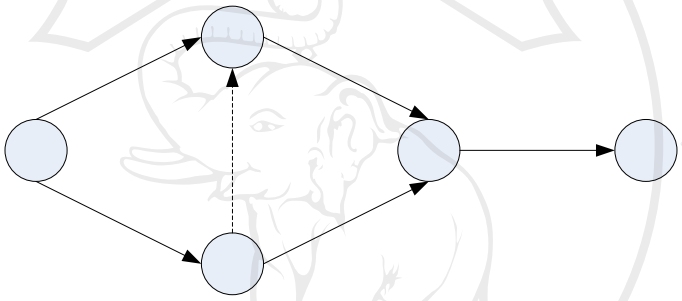
Activity	Time			
A				
B				
C				

อภิรักษ์ จิรายุสกุล (2547) แกนต์ชาร์ต พัฒนาโดย Henry L.Gantt มีรูปแบบคล้ายกราฟแท่งตามแนวนอน แนวตั้งของกราฟแท่ง คือ กิจกรรมทั้งหมดที่เป็นไปได้ในโครงการ และแนวนอนแสดงระยะเวลาที่ใช้ในการพัฒนา ดังตาราง 2.2

ตาราง 2.2 แกนต์ชาร์ต (Gantt Chart)



เพิร์ธชาร์ต (Project Evaluation Review Technique: Pert Chart) เมสินี นาคมณี (2547) มีลักษณะแตกต่างจากแกนต์ชาร์ตที่เห็นได้ชัดเจนคือ มีการแสดงงานในลักษณะของโหนด และแสดงความเกี่ยวเนื่อง (Dependency) ของงานแต่ละอันที่เกิดขึ้นอย่างชัดเจน ช่วยให้เห็นลักษณะการวางแผนในแต่ละงานได้อย่างชัดเจน เนื่องจากลักษณะของเพิร์ธชาร์ตนั้นเน้นในส่วนที่เรียกว่าการจัดงานอยู่บนเส้นทางวิกฤต (Critical Path) โดยเส้นทางวิกฤต คือ เส้นทางที่สั้นที่สุดที่ต้องใช้ตลอดทั้งโครงการหรือเวลาน้อยที่สุดที่จำเป็นต้องใช้ในการทำงานให้เสร็จ ดังนั้นหากงานที่อยู่บนเส้นทางวิกฤต คือ งานที่ไม่สามารถล่าช้าได้ เนื่องจากหากมีการจัดทำล่าช้าจะส่งผลกระทบทำให้โครงการล่าช้าไปด้วย ดังแสดงไว้ในรูป 2.1



รูป 2.1 เพิร์ธชาร์ต (Project Evaluation Review Technique: Pert Chart)

อภิรักษ์ จิรายุสกุล (2547) เพิร์ธชาร์ต พัฒนาในปี 1958 โดย US Navy และ Booze Allen และ Hanilton นิยมใช้โครงการขนาดใหญ่ที่มีกิจกรรมต่อเนื่องกันหลายๆ อย่าง เพิร์ธชาร์ต มีลักษณะคล้ายกราฟ ประกอบวงกลมแทนเหตุการณ์ที่เกิดขึ้นในระบบ ลูกศรแทนกิจกรรมและระยะเวลาดำเนินการ เส้นปะ แสดงการเชื่อมระหว่างกิจกรรมที่ระบุว่า ต้องรอให้กิจกรรมก่อนหน้าดำเนินเสร็จก่อน

ประโยชน์ของเพิร์ธ คือ สามารถคำนวณระยะเวลาน้อยสุดที่กิจกรรมทั้งหมดในโครงการสามารถดำเนินการเสร็จสมบูรณ์ ซึ่งเรียกว่า เส้นทางวิกฤต (Critical path) ค่าที่ได้จากการคำนวณนี้จะแสดงให้เห็นว่า มีกิจกรรมใดบ้างที่มีความเสี่ยงในการดำเนินงาน ซึ่งผู้บริหารโครงการจะต้องให้ความสนใจเป็นพิเศษ เพราะถ้าทำงานเกินกว่าที่กำหนดไว้บนเส้นทางจะส่งผลให้โครงงานเสร็จช้ากว่าที่วางแผน

เส้นปะ แทนกิจกรรมหุ่น (Dummy activity) ที่มีระยะเวลาการดำเนินกิจกรรมเป็นศูนย์ ความหมายของเส้นนี้ คือ เป็นสิ่งที่ระบุว่า กิจกรรมที่เชื่อมด้วยเส้นปะต้องรอให้กิจกรรมก่อนหน้าดำเนินการเสร็จก่อนจึงจะสามารถเริ่มดำเนินกิจกรรมนั้นได้

2.5 สมการในการวิเคราะห์และประเมินเวลาที่ต้องใช้ในแต่ละกิจกรรมของโครงการ

เวลาเร็วสุด (Earliest Time: ET) คือ ระยะเวลาเร็วที่สุดของกิจกรรมที่สามารถดำเนินงานสำเร็จ คำนวณจากผลรวมของเวลาสะสมที่ประมาณไว้จากเหตุการณ์แรก ไปเรื่อยจนถึงเหตุการณ์สุดท้าย แล้วเลือกเส้นทางที่มีค่าสะสมของเวลามากที่สุด

เวลาช้าที่สุด (Latest Time: LT) คือ ระยะเวลาช้าที่สุดของกิจกรรมที่สามารถดำเนินงานสำเร็จ คำนวณได้จากผลต่างของเวลาสะสมที่ประมาณไว้จากเหตุการณ์ที่สุดท้าย ไปเรื่อยจนถึงเหตุการณ์แรก แล้วเลือกเส้นทางที่มีค่าสะสมของเวลามากที่สุด

เวลายืดหยุ่น (Slack time) คือ ผลต่างระหว่าง ET และ LT ถ้ามีค่าเท่ากับศูนย์ แสดงว่านั้นควรได้รับการดูแลและติดตามการทำงานอย่างใกล้ชิด เพื่อป้องกันไม่ให้โครงการเสร็จล่าช้า โดยการคำนวณเพื่อประมาณการระยะเวลาของแต่ละกิจกรรม มีดังนี้ นภัสถ์ หาญพรชัย (2550)

เวลาเร็วสุด (Earliest Time: ET)

$$tj = Maxi\ [ti = Lij] \quad where\ t1 = 0$$

เวลาช้าที่สุด (Latest Time: LT)

$$Ti = Minj\ [Tj = Lij]$$

เมื่อเวลาเร็วสุด ของแต่ละโหนด เท่ากับ เวลาช้าที่สุด ของแต่ละที่โหนด จะไม่เกิดเวลายืดหยุ่น

$$ti = Ti \quad , \quad tj = Tj \quad , \quad tj - ti = Tj - Ti = Lij$$

การคำนวณเพื่อวิเคราะห์เวลายืดหยุ่น ของแต่ละกิจกรรม มีดังนี้

ES ij= Early start time : เวลาที่เร็วที่สุดที่เริ่มทำกิจกรรมได้

EF ij = Early finish time : เวลาที่เร็วที่สุดที่จะทำกิจกรรมนั้นเสร็จได้

LS ij = Late start time : เวลาที่ช้าที่สุดที่จะเริ่มทำกิจกรรมให้เสร็จ

LF ij = Late finish time: เวลาที่ช้าที่สุดที่ต้องทำกิจกรรมให้เสร็จ

$$ESij = ti$$

$$EFij = ESij + Lij$$

$$LFij = Tj$$

$$LSij = LFij - Lij$$

TS ij = Total Slack:เวลาที่เหลือในแต่ละกิจกรรมที่ไม่ทำให้กระทบกับเวลาของโครงการ

$$TSij = LSij - ESij = LFij - EFij$$

FS ij = Free Slack: เวลาที่เหลือในแต่ละกิจกรรมที่ไม่ทำให้กระทบกับเวลาของกิจกรรมต่อไป

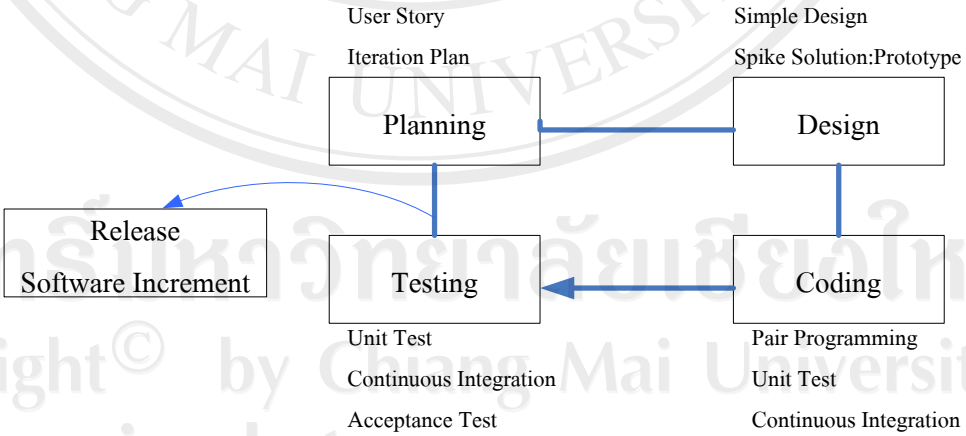
$$FSij = tj - (ti + Lij)$$

ในการจัดการงานในแผนงานนั้น จะทำอย่างไรให้งานที่เป็นงานที่สำคัญไม่อยู่บนเส้นทางวิกฤติ คือ มีเวลายืดหยุ่นในการทำงาน ดังนั้นการจัดการเรื่องความสัมพันธ์ของงานถือเป็นส่วนสำคัญ เนื่องจากเป็นการกำหนดเวลาตามระยะทางในแผนงาน และความยืดหยุ่นที่จะเกิดขึ้นในการทำงาน

ดังนั้นในการวางแผน คือ การนำข้อมูลที่ได้ในส่วนของการประมาณการมารวมกันกับโครงสร้างงานแยกย่อย เพื่อช่วยในการจัดแยกงานความรับผิดชอบ โดยนำเอากระบวนการ และขั้นตอนที่ได้กำหนดไว้เป็นแนวทางในการเชื่อมโยงความสัมพันธ์ระหว่างงาน จากนั้นนำเอาแกนต์ชาร์ต และเพิร์ธชาร์ตมาใช้ในการวางแผน ให้เหมาะสมเพื่อนำไปใช้ในการกำหนดแผนงาน

2.6 กระบวนการผลิตซอฟต์แวร์แบบเอ็กซ์ทรีมโปรแกรมมิ่ง (Extreme Programming: XP)

กิตติ และพนิดา (2550) แบบจำลองกระบวนการผลิตซอฟต์แวร์แบบเอ็กซ์ทรีม โปรแกรมมิ่ง (Extreme Programming: XP) เป็นวิธีปฏิบัติในการพัฒนาระบบวิธีใหม่ที่น่าสนใจตามแนวทางการพัฒนาแบบอิเทอเรชันและอินครีเมนทอล (Iteration and Incremental Development) เป็นแบบจำลองกระบวนการผลิตซอฟต์แวร์ที่ใช้แนวทางเชิงวัตถุเป็นหลัก รองรับความต้องการของผู้ใช้ที่มีความซับซ้อนมากขึ้นและเปลี่ยนแปลงตลอดเวลา แบ่งการทำงานออกเป็น 4 ขั้นตอน ได้แก่ การวางแผน ออกแบบ เขียนโปรแกรม และทดสอบ ดังแสดงไว้ในรูป 2.2



รูป 2.2 แบบจำลองกระบวนการผลิตซอฟต์แวร์แบบเอ็กซ์ทรีม โปรแกรมมิ่ง (Extreme Programming: XP)

2.6.1 ขั้นตอนที่ 1 การวางแผน (Planning) คือ เก็บรวบรวมข้อมูลที่ต้องการ และวิเคราะห์ความต้องการของผู้ใช้ กำหนดความต้องการ และวางแผนการดำเนินงาน กำหนดรายละเอียดของข้อมูล เพื่อจัดสร้างสารสนเทศ แล้วนำมาพิจารณาว่าต้องใช้ระยะเวลาและต้นทุนเท่าใด

2.6.2 ขั้นตอนที่ 2 การออกแบบ (Design) ออกแบบระบบตามข้อกำหนดความต้องการ โดยยึดหลักทำให้ง่ายที่สุด และจัดทำเป็นต้นแบบ กำหนดรายละเอียดที่เอื้อประโยชน์ต่อการเขียนโปรแกรม และมีการเพิ่มฟังก์ชันที่คาดว่าผู้ใช้ต้องการไว้ให้ด้วย

2.6.3 ขั้นตอนที่ 3 การเขียนโปรแกรม (Coding) ทำการเขียนโปรแกรม ตามที่ได้ออกแบบและวางแผนไว้ เพื่อให้สอดคล้องกับวัตถุประสงค์ โดยทีมงานจะจับคู่โปรแกรมเมอร์ 2 คนให้นั่งเขียนโปรแกรมด้วยกัน เป็นการแก้ปัญหาที่โปรแกรมเมอร์คนใดคนหนึ่งไม่อยู่ และเพื่อเป็นการประกันคุณภาพในการเขียนโปรแกรมด้วย

2.6.4 ขั้นตอนที่ 4 การทดสอบโปรแกรม (Testing) ทดสอบระบบว่ามีความถูกต้องตามที่ได้วิเคราะห์ออกแบบไว้มากน้อยเพียงใด จะทดสอบหน่วยย่อยของระบบ โดยมีการสร้างกรณีทดสอบไว้ก่อน การเขียนโปรแกรมภายใต้กรอบการสร้างงานทดสอบ ทำให้สามารถทดสอบโปรแกรมได้โดยอัตโนมัติ และทำให้ง่ายต่อการทดสอบซ้ำเมื่อต้องแก้ไขโปรแกรม จากนั้นจะนำไปให้ลูกค้าทดสอบ และทำการปรับปรุงแก้ไขข้อบกพร่องของระบบงานให้ถูกต้องและเหมาะสมตรงตามที่ได้วิเคราะห์และออกแบบไว้ และจัดทำคู่มือการใช้งาน

2.7 ไอเอสโอ 12207 (ISO 12207) มาตรฐานสำหรับกระบวนการผลิตและพัฒนาซอฟต์แวร์

มาตรฐานไอเอสโอ 12207 เป็นเกณฑ์คุณภาพของการผลิตซอฟต์แวร์ที่มีลักษณะเป็นโปรเซสโมเดลลิ่ง (Process Modeling) คือ เน้นในส่วนของการกำหนดขั้นตอนทีละขั้นตอน (Process) ในการผลิตซอฟต์แวร์ที่มีคุณภาพ ตั้งแต่เริ่มต้นจนกระทั่งจบขั้นตอนของการผลิตซอฟต์แวร์ เพื่อให้การผลิตซอฟต์แวร์นั้นมีคุณภาพ โดยจะมีการกำหนดว่าจะมีผลลัพธ์จากขั้นตอนของการผลิตซอฟต์แวร์ตามที่กำหนดในมาตรฐานไอเอสโอ 12207