บทที่ 1 ความรู้เบื้องต้นในการวิเคราะห์และออกแบบระบบ

1. ระบบคืออะไร และ นักวิเคราะห์ระบบคือใคร

ระบบ (System) คือ สิ่งต่างๆ ที่มีการติดต่อสัมพันธ์กันเพื่อบรรลุเป้าหมายอย่างหนึ่งอย่างใดร่วมกัน กลุ่มขององค์ประกอบต่างๆ ที่ทำงานร่วมกันเพื่อจุดประสงค์อันเดียวกัน ระบบอาจจะประกอบด้วยบุคลากร เครื่องมือ เครื่องใช้ พัสดุ วิธีการ ซึ่งทั้งหมดนี้จะต้องมีระบบจัดการอันหนึ่งเพื่อให้บรรลุจุดประสงค์อันเดียวกัน

เมื่อเราศึกษาระบบใดระบบหนึ่ง เราควรจะต้องเข้าใจการทำงานของระบบนั้นให้ดีโดยการถามตัวเองตลอดเวลาด้วยคำถามเหล่านี้

1. What คือ ระบบทำอะไร , วัตถุประสงค์ของระบบคืออะไร มีแผนงานขั้นตอนอย่างไรเพื่อนำไปสู่ความสำเร็จ (Goal)
2. Who คือ ทำโดยใคร ,บุคคลหรือใครที่รับผิดชอบ
3. When คือ ทำเมื่อไร , การเริ่มดำเนินงานและผลสำเร็จของงานจะสำเร็จลุล่วงได้เมื่อไร ควรมีการจัดตารางการทำงานอย่างมีระบบ การทำงานโดยไม่มีการจัดตารางการทำงานที่แน่นอนส่งผลให้ระบบงานยืดเยื่อ ไม่สามารถปิดงานได้ ก่อให้เกิดค่าใช้จ่ายเพิ่มขึ้น
4. How คือ ทำอย่างไร มีวิธีการทำงานอย่างไร ต้องใช้เครื่องมือใดเพื่อให้งานสำเร็จได้รวดเร็ว

ระบบที่เราควรทราบเพื่อประโยชน์ในการวิเคราะห์และออกแบบได้แก่ ระบบธุรกิจ และ ระบบสารสนเทศ (MIS)

ระบบธุรกิจ (Business System) ได้แก่ระบบที่ทำงานเพื่อจุดประสงค์ด้านธุรกิจ โรงงานอุตสาหกรรมเป็นระบบธุรกิจเพื่อจุดประสงค์ด้านการผลิต นอกจากนี้ระบบขนส่ง ระบบโรงแรม ระบบการพิมพ์ ระบบธนาคาร และอื่นๆ อีกมาก ล้วนแล้วแต่เป็นระบบธุรกิจทั้งนั้น ซึ่งมีจุดประสงค์แตกต่างๆกันออกไป ระบบธุรกิจอาจจะแบ่งย่อยๆลงไปได้อีก เช่น ในโรงงานเราจัดแบ่งเป็นฝ่ายต่างๆ เช่น ฝ่ายผลิต ฝ่ายซ่อมบำรุง ฝ่ายสินค้าคงคลัง หรืออาจจะรวมฝ่ายขายเข้ามาอยู่ด้วยก็ได้ในระบบย่อยของฝ่ายขายจะต้องทำหน้าที่หลายอย่าง เช่น รับใบสั่งสินค้าจากลูกค้า ส่งใบเก็บเงินไปให้ลูกค้า สำหรับฝ่ายบัญชีทำหน้าที่จ่ายเงินเดือนพนักงานเป็นต้น นักวิเคราะห์ต้องทราบขั้นตอนการทำงานในระบบที่ได้รับมอบหมายและทำความเข้าใจให้ดี

ระบบสารสนเทศ (Management Information System) ระบบนี้ช่วยจัดการข้อมูลที่ต้องการใช้ในระบบธุรกิจ ช่วยเก็บตัวเลขและข่าวสารเพื่อช่วยในการดำเนินธุรกิจและการตัดสินใจ เช่น ระบบการเก็บเงินลูกค้า เราต้องการที่จะทราบว่าลูกค้าแต่ละคนชื่ออะไร อยู่ที่ไหน สินค้าและจำนวนที่ขายให้แก่ลูกค้าแต่ละคนเป็นอย่างไร การจ่ายเงินของลูกค้าเป็นอย่างไร ติดค้างนานหรือไม่ หรือหนี้สูญ รวมทั้งจำนวนเงินที่ลูกค้าจะต้องชำระ

นักวิเคราะห์ระบบ (System Analysts) คือ ผู้ที่เป็นตัวกลางในการติดต่อระหว่างระบบสารสนเทศกับกลุ่มผู้เกี่ยวข้อง ได้แก่ เจ้าของระบบ (System Owners) ผู้ใช้ระบบ (System Users) และผู้สร้างระบบ (System Builders) เพื่อพัฒนาระบบสารสนเทศขององค์กรขึ้นมา ทั้งนี้หน้าที่หลักของนักวิเคราห์ระบบจะแบ่งเป็น 2 ส่วน

ส่วนที่ 1 วิเคราะห์ระบบ (System Analysis) เป็นการศึกษา วิเคราะห์ และแยกแยะถึงปัญหาที่เกิดขึ้นในระบบ พร้อมทั้งเสนอแนวทางแก้ไขตามความต้องการของผู้ใช้งานและความเหมาะสมต่อสถานะทางการเงินขององค์กร

ส่วนที่ 2 ออกแบบระบบ (System Design) เป็นวิธีการออกแบบ และกำหนดคุณสมบัติทางเทคนิคโดยนำระบบคอมพิวเตอร์มาประยุกต์ใช้ เพื่อแก้ปัญหาที่ได้ทำการวิเคราะห์มาแล้ว

2. หน้าที่ของนักวิเคราะห์ระบบ

1.รวบรวมข้อมูล

เป็นการรวบรวมข้อมูลของระบบเดิมเพื่อให้ทราบถึงปัญหาที่เกิดขึ้น และนำไปใช้เป็นข้อมูลส่วนหนึ่งในการพัฒนาระบบใหม่ทั้งนี้อาจจะทำแบบสอบถามหรือการสัมภาษณ์เพื่อเก็บรายละเอียดต่างๆ จากผู้ใช้ระบบ จากผู้ใช้ระบบ เพราะผู้ใช้ระบบเป็นผู้ที่เข้าใจถึงปัญหาที่เกิดขึ้นได้ดีที่สุด

2. จัดทำเอกสาร

ในระหว่างการทำพัฒนาระบบนั้น นักวิเคราะห์ระบบจะต้องจัดทำเอกสารประกอบในแต่ละขั้นตอนของการวิเคราะห์ระบบโดยละเอียด และปรับปรุงให้เป็นปัจจุบันอยู่เสมอ เพื่อความคล่องตัวหากมีการเปลี่ยนทีมงานในระหว่างการพัฒนาระบบ

3. จัดทำพจนานุกรมข้อมูล (Data Dictionary)

เป็นการรวบรวมเอกสารทั้งหมด และอธิบายถึงเอกสารต่างๆ ที่ต้องมีการใช้งานในระบบ พจนานุกรมข้อมูลจัดเป็นสิ่งหนึ่งที่นักวิเคราะห์ระบบจำเป็นต้องใช้ในการติดต่อประสานงานกับโปรแกรมเมอร์และเจ้าของระบบ

4.ออกแบบระบบ

นักวิเคราะห์ระบบจะต้องทำการออกแบบการทำงานของระบบใหม่ให้ตรงตามความต้องการของผู้ใช้ระบบ และมีความเหมาะสมมากที่สุด รวมทั้งออกแบบลักษณะการติดต่อของโปรแกรมกับผู้ใช้งาน ฮาร์ดแวร์และเทคโนโลยีสารสนเทศที่จะนำมาใช้ในระบบ กำหนดลักษณะของเครือข่ายที่ใช้ในการเชื่อมต่อคอมพิวเตอร์ กำหนดมาตรการรักษาความปลอดภัยของระบบ รวมไปถึงการประมาณการค่าใช้จ่ายต่างๆ ในส่วนที่จะเกิดขึ้น

5.สร้างแบบจำลอง

ทำการสร้างแบบจำลองของโปรแกรมที่พัฒนาขึ้นเพื่อนำเสนอแก่เจ้าของระบบและผู้ใช้งาน ในบางองค์กรหน้าที่การสร้างแบบจำลองจะเป็นของโปรแกรมเมอร์

6. ทดสอบโปรแกรมที่พัฒนาขึ้น

ในบางครั้งนักวิเคราะห์ระบบและโปรแกรมเมอร์จะเป็นผู้ทดสอบโปรแกรมเอง แต่หากมอบหมายให้ผู้ใช้ระบบเป็นผู้ทดสอบจะมีผลการทดสอบที่มีประสิทธิภาพมากกว่า เนื่องจากผู้ใช้ระบบเป็นผู้ที่รู้และเข้าใจระบบงานอย่างแท้จริง จึงสามารถบอกได้ว่าโปรแกรมที่พัฒนาขึ้นมานั้นทำงานได้สอดคล้องกับการทำงานจริงมากน้อยเพียงใด

7. ติดตั้งและทำการปรับเปลี่ยนระบบ

ทำการติดตั้งและปรับเปลี่ยนระบบเดิมเป็นระบบใหม่ ซึ่งสามารถทำได้หลายลักษณะ เช่น ติดตั้งทั้งหมดทันที ติดตั้งเป็นบางส่วนก่อน หรือติดตั้งระบบใหม่ควบคู่ไปกับการทำงานของระบบเก่า เป็นต้น

8. จัดทำคู่มือ

จัดทำคู่มือและจัดเตรียมหลักสูตรฝึกอบรมให้แก่ผู้ใช้ระบบ เนื่องจากเป็นการเปลี่ยนแปลงของระบบซึ่งหมายรวมถึงการเปลี่ยนแปลงวิธีการดำเนินงาน ลักษณะของโปรแกรมที่ใช้งานก็เปลี่ยนแปลงไป การที่ผู้ใช้ระบบจะสามารถเข้าใจและรับรู้การเปลี่ยนแปลงที่เกิดขึ้นได้อย่างรวดเร็ว คือ การได้รับการฝึกอบรมอย่างถูกต้อง

9. จัดทำแบบสอบถาม

จัดทำแบบสอบถามถึงผลการดำเนินงานของระบบใหม่ที่ได้ติดตั้งไปแล้วในรูปแบบของรายงานผลการใช้งาน (Feedback) เพราะจะทำให้นักวิเคราะห์ระบบทราบว่าผลของการติดตั้งระบบใหม่เป็นอย่างไร และมีปัญหาอะไรเกิดขึ้นตามมาบ้างเพื่อจะได้นำปัญหาเหล่านั้นมาทำการปรับปรุงแก้ไขเพื่อให้ได้เป็นระบบที่ตรงตามความต้องการของผู้ใช้ได้ในที่สุด

10. บำรุงรักษาและประเมินผลการปฏิบัติงานของระบบ

เป็นการดูแลระบบเมื่อมีข้อผิดพลาดเกิดขึ้น รวมทั้งเป็นการปรับปรุง ดัดแปลง หรือแก้ไขทั้งโปรแกรมและขั้นตอนการทำงานของระบบ เพื่อให้ระบบมีการทำงานที่ถูกต้องมากที่สุด นอกจากนั้นยังทำให้สามารถประเมินผลการปฏิบัติงานของระบบใหม่ได้อีกด้วย

11. เป็นผู้ให้คำปรึกษา

คอยให้คำปรึกษาแก่ผู้ใช้ระบบและทุกคนในระบบ (Consulting) ภายหลังจากการติดตั้งระบบแล้ว การใช้งานอาจเกิดข้อสงสัยหรือข้อผิดพลาดขึ้นได้ตลอดเวลา ดังนั้นนักวิเคราะห์ระบบจะต้องคอยให้คำปรึกษา ไม่ว่าจะเป็นทางด้านการใช้โปรแกรมหรือทางด้านเทคนิคก็ตาม

12. เป็นผู้ประสานงาน

ทำหน้าที่ประสานงานระหว่างทุกฝ่ายที่เกี่ยวข้องกับการพัฒนาระบบ (Coordinator) เพื่อให้เข้าใจในเหตุการณ์หรือข้อมูลที่เกิดขึ้นภายในองค์กรได้ถูกต้องตรงกันที่สุด

13. เป็นผู้แก้ไขปัญหา

ในที่นี้จะเป็นผู้ที่นำแนวคิดของคำว่า “ระบบ” มาใช้ในการแก้ปัญหาทั้งการดำเนินงานทางธุรกิจขององค์กรและแก้ปัญหาด้านระบบสารสนเทศด้วย โดยการเปรียบเทียบในลักษณะของงานทางธุรกิจคือระบบ ซึ่งจะต้องกำหนดขอบเขตของระบบผู้ที่เกี่ยวข้องกับการทำงานของระบบพิจารณาว่าข้อมูลที่เข้าและออกจากระบบนั้นเกิดจากบุคคลฝ่ายใดหรือเกิดจากขั้นตอนการทำงานขั้นตอนใด เพื่อให้เการแก้ไขปัญหานั้นสามารถดำเนินการได้อย่างชัดเจนภายในขอบเขตของระบบนั้น

14. เป็นตัวแทนการเปลี่ยนแปลง

นักวิเคราะห์ระบบเป็นผู้ที่สามารถแสดงให้ทุกคนเล็งเห็นถึงผลประโยชน์ที่จะเกิดขึ้นหลังจากการเปลี่ยน

แปลงจากระบบเก่าเป็นระบบใหม่ได้

15. เป็นผู้เตรียมข้อมูลให้กับองค์กร
    เมื่อมีการเปลี่ยนแปลงระบบแล้ว นักวิเคราะห์ระบบจะเป็นผู้ที่ทราบรายละเอียดของการเปลี่ยนแปลงดีที่สุด ซึ่งสามารถเตรียมข้อมูลเพื่อนำไปใช้ในการแข่งขัน หรือการหาตลาดใหม่ขององค์กรได้อย่างรวดเร็วและทันเหตุการณ์

3. คุณสมบัติของนักวิเคราะห์ระบบ

1. มีความชำนาญหลากหลายในศาสตร์คอมพิวเตอร์ เช่น โปรแกรมคอมพิวเตอร์ โปรแกรมภาษา ฮาร์ดแวร์ เทคโนโลยีสารสนเทศ เป็นต้น
2. มีความเข้าใจในระบบธุรกิจ ระบบการเงิน และระบบการตลาด เป็นอย่างดี
3. มีความเข้าใจในความต้องการของผู้ใช้ระบบเป็นอย่างดี
4. ต้องเป็นนักสำรวจ ที่ช่างสังเกตในรายละเอียดต่างๆ ของระบบ รวมไปถึงองค์ประกอบภายนอกที่เกี่ยวข้องกับระบบ เพื่อนำมาเป็นข้อมูลประกอบการพัฒนาระบบ
5. มีจรรยาบรรณต่อองค์กรที่พัฒนาระบบให้ ไม่นำข้อมูลที่ได้ซึ่งเป็นความลับขององค์กรไปเผยแพร่ภายนอก อันอาจจะก่อให้เกิดผลเสียแก่องค์กรนั้นได้
6. ต้องทำงานเป็นทีมได้เป็นอย่างดี เช่น ทีมพัฒนาระบบ ทีมนักวิเคราะห์ระบบ เป็นต้น
7. มีมนุษย์สัมพันธ์ที่ดี เนื่องจากนักวิเคราะห์ระบบจะต้องมีการติดต่อประสานงานระหว่างบุคคลหลายกลุ่ม เพื่อคอยอำนวยความสะดวกและเก็บรวบรวมข้อมูลต่างๆ เพื่อการพัฒนาระบบ
8. สามารถเรียนรู้สิ่งใหม่ๆ ได้ด้วยตนเอง
9. มีความสามารถสูงในการนำเสนอข้อมูลให้ทั้งผู้บริหารระดับสูงรวมไปถึงผู้ใช้ระบบ ให้สามารถเข้าใจได้โดยง่าย และตรงกัน
10. มีความสามารถในการติดต่อสื่อสารเป็นภาษาอังกฤษได้ดี หากองค์กรนั้นสื่อสารภายในเป็นภาษาอังกฤษ
11. สามารถทำงานภายใต้ภาวะกดดันได้ เนื่องจากต้องทำงานกับบุคคลหลายฝ่าย ซึ่งแน่นอนว่าจะต้องมีปัญหาเกิดขึ้นจากบุคคลต่างๆ มากมาย
12. เป็นนักจิตวิทยา ในการที่จะพูดคุยหรือติดต่อกับกลุ่มบุคคลหลายกลุ่มเพื่อให้ได้ข้อมูลมาอย่างละเอียดถูกต้องและสามารถโน้มน้าวจิตใจผู้ใช้ระบบได้

4. นักวิเคราะห์ระบบพัฒนาการระบบสารสนเทศอย่างไร

การที่มีนักวิเคราะห์ระบบในองค์กรนั้นเป็นการได้เปรียบเพราะจะรู้โดยละเอียดว่า การทำงานในระบบนั้นๆ เป็นอย่างไร และอะไรคือความต้องการของระบบ ในกรณีที่นักวิเคราะห์ระบบไม่ได้อยู่ในองค์กรนั้น ก็สามารถวิเคราะห์ระบบได้เช่นกันโดยการศึกษาสอบถามผู้ใช้และวิธีการอื่นๆ ซึ่งจะกล่าวในภายหลัง ผู้ใช้ในที่นี้ก็คือ เจ้าของ และผู้ที่เกี่ยวข้องในระบบสารสนเทศนั่นเอง ผู้ใช้อาจจะมีคนเดียว หรือหลายคนก็ได้ เพื่อให้

นักวิเคราะห์ระบบทำงานได้อย่างคล่องตัวมีลำดับขั้น และเป้าหมายที่แน่นอน นักวิเคราะห์ระบบควรจะทราบถึงว่า ระบบสารสนเทศนั้นพัฒนาขึ้นมาอย่างไร มีขั้นตอนอย่างไร

วงจรการพัฒนาระบบ (System Development Life Cycle)

คือ กระบวนการทางความคิด (Logical Process) ในการพัฒนาระบบสารสนเทศเพื่อแก้ปัญหาทางธุรกิจและตอบสนองความต้องการของผู้ใช้ได้ โดยระบบที่จะพัฒนานั้น อาจเริ่มด้วยการพัฒนาระบบใหม่เลย หรือนำระบบเดิมที่มีอยู่แล้วมาปรับเปลี่ยนให้ดียิ่งขึ้น ขั้นตอนในวงจรการพัฒนาระบบ ช่วยให้นักวิเคราะห์ระบบสามารถดำเนินการได้อย่างมีแนวทางและเป็นขั้นตอน ทำให้สามารถควบคุมระยะเวลาและงบประมาณในการปฏิบัติงานของโครงการพัฒนาระบบได้

ระบบสารสนเทศทั้งหลายมีวงจรชีวิตที่เหมือนกันตั้งแต่เกิดจนตาย วงจรนี้จะเป็นขั้นตอนที่เป็นลำดับตั้งแต่ต้นจนเสร็จเรียบร้อย เป็นระบบที่ใช้งานได้ ซึ่งนักวิเคราะห์ระบบต้องทำความเข้าใจให้ดีว่าในแต่ละขั้นตอนจะต้องทำอะไร และทำอย่างไร ขั้นตอนการพัฒนาระบบมีอยู่ด้วยกัน 7 ขั้นตอนคือ

1. ค้นหาและเลือกสรรโครงการ (Project Identification and Selection) เนื่องจากในสภาพเศรษฐกิจปัจจุบันมีสภาวะแข่งขันของธุรกิจค่อนข้างสูง จึงทำให้องค์กรจำเป็นต้องหากลยุทธ์ทางการแข่งขันเพื่อเพิ่มความได้เปรียบต่อคู่แข่งขัน และแย่งส่วนแบ่งในตลาดให้ได้มากขึ้นอันจะนำไปสู่ผลกำไรที่มากขึ้น ซึ่งกลยุทธ์การแข่งขันดังกล่าวอาจจะเป็นการพัฒนาระบบงานที่ดำเนินการอยู่ในปัจจุบันหรือพัฒนาระบบใหม่ แต่จะมีระบบงานใดบ้างนั้น จะต้องค้นหาจากผู้ที่ปฏิบัติงานกับระบบงานจริง โครงการที่รวบรวมมาได้อาจมีหลายโครงการ แต่อาจดำเนินการพร้อมกันหมดไม่ได้ เนื่องจากมีข้อจำกัดเรื่องของต้นทุนและเวลาที่ใช้ในการดำเนินการ ดังนั้นจำเป็นต้องมีการเลือกสรรโครงการที่เหมาะสมและให้ผลประโยชน์แก่องค์กรมากที่สุดในสภาวะการณ์ปัจจุบัน โดยที่บุคคลากรในองค์กร อาจต้องการพัฒนาระบบภายในองค์กรขึ้นมาหลากหลายโครงการที่ล้วนแต่เป็นการพัฒนาประสิทธิภาพในการดำเนินงานขององค์กร แต่การดำเนินการพัฒนาระบบในทุกๆ โครงการพร้อมกันอาจเป็นไปไม่ได้เนื่องจากมีข้อจำกัดเรื่องของต้นทุนที่ใช้ในการพัฒนา การพัฒนาระบบงานสารสนเทศในขั้นตอนแรกของวงจรการพัฒนาระบบ (SDLC) เป็นขั้นตอนที่อธิบายถึงการค้นหาโครงการของระบบงานที่ต้องการพัฒนา และพิจารณาเลือกโครงการที่จะทำให้องค์กรได้รับผลตอบแทนมากที่สุด

   เริ่มจากการที่ผู้บริหารขององค์กรหรือบุคลากรมีความต้องการที่จะพัฒนาระบบงาน จึงได้มีการแต่งตั้งกลุ่มบุคคลเพื่อค้นหาโครงการที่เห็นสมควรว่าควรได้รับการพัฒนา จากกิจกรรมการค้นหาโครงการนี้ ส่งผลให้เกิดโครงการพัฒนาขึ้นมาหลายโครงการ ผู้บริหารและนักวิเคราะห์ระบบจะต้องทำการจำแนกกลุ่มของโครงการให้เป็นหมวดหมู่อย่างมีหลักเกณฑ์ เช่น จำแนกตามความสำคัญ หรือจำแนกตามผลตอบแทนที่จะได้รับ กิจกรรมสุดท้ายของขั้นตอนนี้จะทำการเลือกโครงการที่เหมาะสมที่สุด และตรงกับวัตถุประสงค์ (Objective) ขององค์กรในสถานการณ์ปัจจุบันมากที่สุด

สรุป การทำงานในขั้นตอนการค้นหาและการเลือกสรรโครงการ (Project Identification / Selection)	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคที่ใช้
1. ค้นหาโครงการพัฒนาระบบที่เห็นสมควรได้รับการพัฒนา	ตารางเมตริกซ์ (Matrix Table)
2. จำแนกและจัดลำดับโครงการ	
3. เลือกโครงการที่เหมาะสมที่สุด	

2. เริ่มต้นและวางแผนโครงการ (Project Initiating and Planning) รวบรวมข้อมูลเพิ่มเติมเพื่อเริ่มต้นจัดทำโครงการที่ได้รับอนุมัติ โดยเริ่มจากการจัดตั้งทีมงาน เพื่อเตรียมการดำเนินงานจากนั้นทีมงานดังกล่าวร่วมกันค้นหา สร้างแนวทาง และเลือกทางที่ดีที่สุดในการนำระบบใหม่มาใช้งาน เมื่อได้ทางเลือกที่ดีและเหมาะสมที่สุดแล้ว ทีมงานจึงเริ่มวางแผนดำเนินงานโครงการ โดยศึกษาความเป็นไปได้ กำหนดระยะเวลาดำเนินงานแต่ละขั้นตอนและกิจกรรม เพื่อนำเสนอต่อผู้บริหารพิจารณาอนุมัติให้ดำเนินการในขั้นตอนต่อไป

สรุป การทำงานในขั้นตอนการเริ่มต้นและการวางแผนโครงการ (Project Initiating and Planning	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคที่ใช้
1. เริ่มต้นโครงการ 2. เสนอแนวทางเลือกในการนำระบบใหม่มาใช้งาน 3. วางแผนโครงการ	- เทคนิคการรวบรวมสารสนเทศและข้อเท็จจริง (Fact-Finding and Information Gathering) - เทคนิคการวิเคราะห์ต้นทุนและผลกำไร (Cost-Benefit Analysis) - PERT Chart - GANTT Chart

3. วิเคราะห์ระบบ (System Analysis) ศึกษาขั้นตอนการดำเนินการของระบบเดิมเพื่อหาปัญหาที่เกิดขึ้น รวบรวมความต้องการในระบบใหม่จากผู้ใช้ระบบแล้วนำความต้องการเหล่านั้นมาศึกษาและวิเคราะห์เพื่อแก้ปัญหาดังกล่าว ด้วยการใช้แบบจำลองต่างๆ ช่วยในการวิเคราะห์

เริ่มจากทำการศึกษาถึงขั้นตอนการดำเนินงานของระบบเดิมหรือระบบปัจจุบันว่าเป็นไปอย่างไรบ้าง ปัญหาที่เกิดขึ้นคืออะไร หลังจากนั้นจึงรวบรวมความต้องการในระบบใหม่จากผู้ใช้ระบบ โดยอาจจะมีการใช้เทคนิคในการเก็บรวบรวมข้อมูลเช่น การออกแบบสอบถาม การสัมภาษณ์ จากนั้นนำข้อมูลที่รวบรวมได้มาทำการวิเคราะห์ด้วยการจำลองแบบข้อมูลเหล่านั้น ได้แก่ แบบจำลองขั้นตอนการทำงานของระบบ (Process Model) แบบจำลองข้อมูล (Data Model) โดยมีการใช้เครื่องมือในการจำลองแบบชนิดต่างๆ เช่น แผนภาพกระแสข้อมูล (Data Flow Diagram) แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (Entity

Relationship Diagram) เป็นต้น

สรุป การทำงานในขั้นตอนการวิเคราะห์ระบบ (System Analysis)	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคใช้
1. ศึกษาขั้นตอนการทำงานของระบบเดิม 2. กำหนดความต้องการในระบบใหม่จากผู้ใช้ระบบ 3. จำลองแบบขั้นตอนการทำงาน 4. อธิบายขั้นตอนการทำงานของระบบ 5. จำลองแบบข้อมูล	1.เทคนิคการรวบรวมสารสนเทศและข้อเท็จจริง (Fact-Finding and Information Gathering) 2. แผนภาพกระแสข้อมูล (Data Flow Diagram) 4. แผนภาพแสดงความสัมพันธ์ของข้อมูล (E-R Diagram) 5. พจนานุกรมข้อมูล (Data Dictionary) 6. ตัวต้นแบบ (Prototyping) 7. ผังงานระบบ (System Flowcharts) 8. เครื่องมือช่วยในการวิเคราะห์และออกแบบระบบ (CASE Tools)

4. ออกแบบเชิงตรรกะ (Logical Design) เป็นขั้นตอนในการออกแบบลักษณะการทำงานของระบบตามทางเลือกที่ได้ทำการเลือกไว้จากขั้นตอนการวิเคราะห์ระบบ โดยการออกแบบในเชิงตรรกะนี้ยังไม่ได้มีการระบุถึงคุณลักษณะของอุปกรณ์ที่จะนำมาใช้ เพียงแต่กำหนดถึงลักษณะของรูปแบบรายงานที่เกิดจากการทำงานของระบบ ลักษณะของการนำข้อมูลเข้าสู่ระบบและผลลัพธ์ที่ได้จากระบบ

ขั้นตอนการออกแบบเชิงตรรกะจะสัมพันธ์และเชื่อมโยงกับขั้นตอนการวิเคราะห์ระบบเป็นอย่างมาก เนื่องจากอาจจะมีการนำแผนภาพที่แสดงถึงความต้องการของผู้ใช้ระบบที่ได้จากขั้นตอนการวิเคราะห์ระบบมาทำการแปลงเพื่อให้ได้ข้อมูลเฉพาะของการออกแบบ (System Desing Specification) ที่สามารถนำไปเขียนโปรแกรมได้สะดวกขึ้น เช่น การออกแบบส่วนนำเข้าข้อมูลและผลลัพธ์นั้นต้องอาศัยข้อมูลที่เป็น Data Flow ที่ปรากฏอยู่บนแผนภาพกระแสข้อมูลในขั้นตอนการวิเคราะห์ระบบ

สรุป การทำงานในขั้นตอนการออกแบบเชิงตรรกะ (Logical Design)	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคที่ใช้
1. ออกแบบแบบฟอร์มข้อมูลและรายงาน (Form/Report) 2. ออกแบบ User Interface 3. ออกแบบฐานข้อมูลในระดับตรรกะ	1. แผนภาพกระแสข้อมูล (Data Flow Diagram) 2. แผนภาพแสดงความสัมพันธ์ของข้อมูล (E-R Diagram) 3. พจนานุกรมข้อมูล (Data Dictionary) 4. ตัวต้นแบบ (Prototyping) 5. เครื่องมือช่วยในการวิเคราะห์และออกแบบระบบ (CASE Tools)

5. ออกแบบเชิงกายภาพ (Physical Design) ระบุถึงลักษณะการทำงานของระบบทางกายภาพหรือทางเทคนิค โดยระบุถึงคุณลักษณะของอุปกรณ์ที่จะนำมาใช้เทคโนโลยี โปรแกรมภาษาที่จะนำมาใช้เขียนโปรแกรม ฐานข้อมูล ระบบปฏิบัติการ และระบบเครือข่ายที่เหมาะสม สิ่งที่ได้จากขั้นตอนการออกแบบทางกายภาพนี้จะเป็นข้อมูลเฉพาะของการออกแบบ (System Design Specification) เพื่อส่งมอบให้กับโปรแกรมเมอร์เพื่อใช้เขียนโปรแกรมตามลักษณะการทำงานของระบบที่ได้ออกแบบและกำหนดไว้

ทั้งนี้ในการออกแบบที่นอกเหนือจากที่กล่าวมานี้ ขึ้นอยู่กับระบบขององค์กรว่าจะต้องมีการเพิ่มเติมรายละเอียดส่วนใดบ้างแต่ควรจะมีการออกแบบระบบความปลอดภัยในการใช้ระบบด้วย โดยการกำหนดสิทธิในการใช้งานข้อมูลที่อยู่ในระบบของผู้ใช้ตามลำดับความสำคัญ เพื่อป้องกันการนำข้อมูลไปใช้ในทางที่ไม่ถูกต้อง นอกจากนี้นักวิเคราะห์ระบบอาจจะมีการตรวจสอบความพึงพอใจในรูปแบบและลักษณะการทำงานที่ออกแบบไว้ โดยอาจจะมีการสร้างตัวต้นแบบ (Prototype) เพื่อให้ผู้ใช้ได้ทดลองใช้งาน

สรุป การทำงานในขั้นตอนการออกแบบทางกายภาพ (Physical Design)	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคที่ใช้
1. ออกแบบฐานข้อมูลในระดับกายภาพ 2. ออกแบบ Application	1. แผนภาพกระแสข้อมูล (Data Flow Diagram) 2. แผนภาพแสดงความสัมพันธ์ของข้อมูล (E-R Diagram) 3. พจนานุกรมข้อมูล (Data Dictionary) 4. ตัวต้นแบบ (Prototyping) 5. เครื่องมือช่วยในการวิเคราะห์และออกแบบระบบ (CASE Tools)

6. พัฒนาและติดตั้งระบบ (System Implementation) เป็นการนำระบบที่ออกแบบแล้วมาทำการเขียนโปรแกรมเพื่อให้เป็นไปตามคุณลักษณะและรูปแบบต่างๆ ที่ได้กำหนดไว้หลังจากเขียนโปรแกรมเรียบร้อยแล้ว นักวิเคราะห์จะต้องทำการทดสอบโปรแกรม ตรวจสอบหาข้อผิดพลาดของโปรแกรมที่พัฒนาขึ้นมา และสุดท้ายคือการติดตั้งระบบไม่ว่าจะเป็นระบบใหม่หรือเป็นการพัฒนาระบบเดิมที่มีอยู่แล้ว โดยทำการติดตั้งตัวโปรแกรม ติดตั้งอุปกรณ์ พร้อมทั้งจัดทำคู่มือและจัดเตรียมหลักสูตร อบรมให้แก่ผู้ใช้งานที่เกี่ยวข้อง

เริ่มจากการเขียนโปรแกรมซึ่งโปรแกรมเมอร์จะได้รับชุดเอกสารที่เกิดขึ้นตั้งแต่ขั้นตอนการออกแบบโดยเฉพาะข้อมูลส่วนของการออกแบบที่จะช่วยให้การเขียนโปรแกรมง่ายขึ้น หลังจากนั้นจะต้องมีการทดสอบโปรแกรมเพื่อหาข้อผิดพลาดที่เกิดขึ้นและทำการแก้ไขในเบื้องต้น เมื่อโปรแกรมผ่านการทดสอบแล้ว กิจกรรมต่อไปคือการติดตั้งระบบใหม่ พร้อมทั้งจัดทำคู่มือประกอบการใช้โปรแกรม จัดหลักสูตรฝึกอบรมผู้ใช้ระบบและคอยช่วยเหลือในระหว่างการทำงาน

สรุป การทำงานในขั้นตอนพัฒนาและติดตั้งระบบ (System Implementation)	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคที่ใช้
1. เขียนโปรแกรม (Coding) 2. ทดสอบโปรแกรม (Testing) 3. ติดตั้งระบบ (Installation) 4. จัดทำเอกสาร (Documentation) 5. ฝึกอบรม (Training) 6. บริการให้ความช่วยเหลือหลังการติดตั้ง (Support)	1. โปรแกรมช่วยสอน (Computer Aid Instruction :CAI) 2. ระบบคอมพิวเตอร์เพื่อการฝึกอบรม (Computer-Based Training :CBT) 3. ระบบการฝึกอบรมผ่านเว็บ (Web-Base Training: WBT) 4. โปรแกรมแก้ไขข้อผิดพลาด (Debugging Program)

7. ซ่อมบำรุงระบบ (System Maintenance) เป็นขั้นตอนสุดท้ายของวงจรพัฒนาระบบ (SDLC) หลังจากระบบใหม่ได้เริ่มดำเนินการ ผู้ใช้ระบบอาจจะพบกับปัญหาที่เกิดขึ้นเนื่องจากความไม่คุ้นเคยกับระบบใหม่ และอาจค้นพบวิธีการแก้ไขปัญหานั้นเพื่อให้ตรงกับความต้องการของผู้ใช้เอง ดังนั้นนักวิเคราะห์ระบบและโปรแกรมเมอร์จะต้องคอยแก้ไขและเปลี่ยนแปลงระบบที่พัฒนาขึ้นมาจนกว่าจะเป็นที่พอใจของผู้ใช้ระบบมากที่สุด ปัญหาที่ผู้ใช้ระบบค้นพบระหว่างการดำเนินการนั้นเป็นผลดีในการทำให้ระบบใหม่มีประสิทธิภาพมากยิ่งขึ้น เนื่องจากผู้ใช้ระบบเป็นผู้ที่เข้าใจในการทำงานทางธุรกิจเป็นอย่างดี ซึ่งสามารถให้คำตอบได้ว่าระบบที่พัฒนามานั้นตรงต่อความต้องการหรือไม่

เริ่มจากการมีการใช้งานระบบใหม่ที่ได้ติดตั้งแล้วในระยะแรก ผู้ใช้จะพบกับปัญหาที่เกิดขึ้น ซึ่งอาจจะมีการทำการบันทึกปัญหาเหล่านั้นไว้เพื่อส่งให้นักวิเคราะห์ระบบและโปรแกรมเมอร์ทำการแก้ไขต่อไป ซึ่งเป็นเรื่องปกติที่จะมีการปรับปรุงเปลี่ยนแปลง และแก้ไขระบบที่เพิ่งมีการติดตั้งใช้งานในระยะเริ่มต้น โดยนักวิเคราะห์ระบบจะทำการพิจารณาถึงปัญหาเหล่านั้นเพื่อหาแนวทางแก้ไขต่อไป

สรุป การทำงานในขั้นตอนการบำรุงรักษาระบบ (System Maintenance)	
กิจกรรม	ตัวอย่างแผนภาพ เครื่องมือและเทคนิคที่ใช้
1. เก็บรวบรวมคำร้องขอให้ปรับปรุงระบบ 2. วิเคราะห์ข้อมูลคำร้องขอเพื่อการปรับปรุง 3. ออกแบบการทำงานที่ต้องการปรับปรุง 4. ปรับปรุงระบบ	แบบฟอร์มแจ้งข้อผิดพลาดของระบบ

5. เครื่องมือสนับสนุนการพัฒนาระบบ (Computer-Aided Systems Engineering Tools: CASE Tools)

แม้ว่าในแต่ละขั้นตอนของการพัฒนาระบบ จะมีการนำเทคนิค แบบจำลอง และแผนภาพ ชนิดต่างๆ อธิบายแทนข้อมูลจากเอกสารที่เป็นข้อความอธิบายลักษณะการทำงานของระบบ และวิธีแก้ไขปัญหาที่เกิดขึ้น

ก็ตาม หากขั้นตอนในการทำงานเหล่านี้สามารถลดระยะเวลาลงได้ จะทำให้สามารถเพิ่มเวลาในขั้นตอนอื่น ที่เห็นว่าควรใส่ใจในรายละเอียดเพิ่มขึ้นได้ ส่งผลให้การพัฒนาระบบมีความถูกต้องมากขึ้นและผิดพลาดน้อยลงได้

ปัจจุบันมีซอฟต์แวร์ที่ช่วยสร้างแผนภาพ รายงาน โค้ดโปรแกรม ในระหว่างการวิเคราะห์และออกแบบระบบให้เป็นไปโดยอัตโนมัติ นั่นคือ Computer-Aided Systems Engineering(CASE) ซึ่งเป็นโปรแกรมประยุกต์หรือซอฟต์แวร์ชนิดหนึ่งของเทคโนโลยี ที่ช่วยในการพัฒนาระบบ คอยสนับสนุนการทำงานในแต่ละขั้นตอนของการพัฒนา ด้วยการเตรียมฟังก์ชันการทำงานต่างๆ ที่ทำให้การทำงานแต่ละขั้นตอนมีความรวดเร็วและมีคุณภาพมากขึ้น

CASE จะช่วยแบ่งเบาภาระของนักวิเคราะห์ระบบได้มาก ตั้งแต่การช่วยสร้าง Context Diagram, Flowchart, E-R Diagram สร้างรายงานและแบบฟอร์ม ตลอดจนการสร้างโค้ดโปรแกรม (Source Code) ให้อัตโนมัติอีกด้วย

5.1. ขอบข่ายของเครื่องมือสนับสนุนการพัฒนาระบบ (CASE Tool Framework)

CASE ที่ใช้ในการพัฒนาระบบถูกแบ่งขอบข่ายการทำงานออกเป็น 2 ช่วง โดยการแบ่งนั้นอ้างอิงจากขั้นตอนการพัฒนาระบบในวงจร SDLC ซึ่งมีดังต่อไปนี้

Upper-CASE : เป็นเครื่องมือที่ช่วยสนับสนุนการทำงานในขั้นตอนต้นๆ ของการพัฒนาระบบ ได้แก่ ขั้นตอนการวางแผน ขั้นตอนการวิเคราะห์ และขั้นตอนการออกแบบระบบ

Lower-CASE : เป็นเครื่องมือที่ช่วยสนับสนุนการทำงานในขั้นตอนสุดท้ายในการพัฒนาระบบ ได้แก่ ขั้นตอนการออกแบบ ขั้นตอนการพัฒนาและทดสอบระบบ และขั้นตอนการให้บริการหลังการติดตั้งระบบ จะเห็นว่า CASE ทั้งสองระดับนี้ มีการทำงานที่ซ้ำซ้อนกันอยู่ บางครั้งองค์กรอาจเลือกใช้งาน CASE Tools ทั้งสองระดับร่วมกันได้

5.2. คุณสมบัติและความสามารถของ CASE (Facilities and Functions)

ในการทำงานของ CASE จะมีการเรียกใช้ข้อมูลจาก Repository ซึ่งจะทำให้ CASE มีความสามารถและจัดเตรียมสิ่งอำนวยความสะดวกให้กับนักวิเคราะห์ระบบในการพัฒนาระบบได้ ดังนี้

1. เครื่องมือช่วยสร้างแผนภาพ (Diagram Tools) ใช้ในการเขียนแผนภาพเพื่อจำลองสิ่งต่างๆ ของระบบ ซึ่งสามารถเชื่อมโยงกับแบบจำลองส่วนอื่นได้
2. เครื่องมือช่วยเก็บรายละเอียดต่างๆ ของระบบ (Description Tools) ใช้ในการบันทึก ลบ และแก้ไขรายละเอียดต่างๆ ของระบบได้ รวมทั้งยังสามารถแสดงผลลัพธ์ในรูปแบบเอกสารแสดงรายละเอียดได้
3. เครื่องมือช่วยสร้างตัวต้นแบบ (Prototyping Tools) ใช้ในการสร้างโปรแกรมต้นแบบเพื่อจำลองระบบออกมาทดลองใช้งานได้ในระดับที่สามารถบอกถึงความพอใจของผู้ใช้ได้
4. เครื่องมือช่วยสร้างรายงานแสดงรายละเอียดของแบบจำลอง (Inquiry and Reporting) ใช้ในการสร้างรายงานรายละเอียดต่างๆ ของแบบจำลองซึ่งถูกเก็บไว้ใน Repository ได้

5. เครื่องมือเพื่อคุณภาพของแบบจำลอง (Quality Management Tools) ช่วยในการสร้างแบบจำลอง เอกสาร และตัวต้นแบบต่างๆ ที่ถูกสร้างขึ้นมีคุณภาพ โดยมีการตรวจสอบความถูกต้องและความสอดคล้องกันได้ อีกทั้งหากเกิดข้อผิดพลาดขึ้นเครื่องมือชนิดนี้สามารถบ่งบอกถึงข้อผิดพลาดนั้นได้
6. เครื่องมือสนับสนุนการตัดสินใจ (Decision Support Tools) จัดเตรียมสารสนเทศเพื่อการตัดสินใจที่จะเกิดขึ้นระหว่างการพัฒนาระบบ เช่น ช่วยนักวิเคราะห์ระบบประมาณการและวิเคราะห์ถึงความเป็นไปได้ของแนวทางแก้ไขปัญหา เป็นต้น
7. เครื่องมือช่วยจัดการเอกสาร (Documentation Oraanization tools) ใช้ในการสร้าง จัดการ และแสดงรายงานสารสนเทศต่างๆ ซึ่งถูกเก็บไว้ใน Repository เพื่อนำเสนอต่อผู้บริหารและผู้ใช้ระบบได้
8. เครื่องมือช่วยออกแบบ (Design Generation Tools) ใช้ในการออกแบบระบบคร่าวๆ ในเบื้องต้นได้ภายใต้ความต้องการที่รวบรวมมาแล้ว เช่น CASE สามารถออกแบบฐานข้อมูลที่ได้สร้างแบบจำลองข้อมูลมาแล้ว
9. เครื่องมือช่วยสร้างโค้ดโปรแกรม (Code Generator Tools) ใช้ในการสร้างโค้ดของโปรแกรมทั้งหมดหรือสามารถสร้างเพียงบางส่วนได้
10. เครื่องมือช่วยทดสอบ (Testing Tools) ช่วยให้นักวิเคราะห์และโปรแกรมเมอร์สามารถทดสอบโปรแกรมได้รวดเร็วยิ่งขึ้น
11. เครื่องมือช่วยให้สามารถใช้ข้อมูลร่วมกัน (Data Sharing Tools) เตรียมการนำเข้า (Import) และนำออก (Export) ของสารสนเทศระหว่าง CASE Tools ที่ต่างกันได้

คุณสมบัติและความสามารถของ CASE เป็นสิ่งที่คอยอำนวยความสะดวกให้กับนักวิเคราะห์ระบบในการพัฒนาระบบ ซึ่งจะช่วยให้การทำงานมีความสะดวก รวดเร็วและถูกต้องมากยิ่งขึ้น

5.4. **ประโยชน์ที่ได้จากการใช้** CASE

การเลือกใช้ CASE ช่วยในการพัฒนาระบบนั้นสามารถแบ่งเบาการทำงานของนักวิเคราะห์ระบบ ช่วยให้เอกสารหรือแผนภาพต่างๆ ที่จัดทำขึ้น ดูเป็นระเบียบเรียบร้อยและมีคุณภาพ ที่สำคัญคือช่วยลดเวลาในการทำงานได้มาก นอกจากนี้แล้วยังส่งผลให้เกิดประโยชน์ต่างๆ ดังนี้

1. มีการพัฒนาคุณภาพในการทำงาน เนื่องจาก CASE สามารถตรวจสอบความถูกต้อง สมบูรณ์ของแผนภาพและโปรแกรมได้
2. มีการสร้างเอกสารที่ดี
3. ประหยัดเวลาในการบำรุงรักษาให้ข้อมูลนั้นเป็นปัจจุบันมากที่สุด เพียงเข้าไปทำการแก้ไขในฐานข้อมูล Repository เท่านั้นก็สามารถสร้างเอกสารให้เป็นปัจจุบันได้ โดยไม่ต้องตามไปแก้ไขเอกสารที่เกี่ยวข้องทั้งหมดเอง

6. ปัจจัยที่มีผลกระทบต่อนักวิเคราะห์ระบบ

6.1. แหล่งปัจจัยที่สามจากภายนอก (External Third Party)

1. แหล่งข้อมูลภายนอกองค์กร (Outsourcing) แบ่งลักษณะได้ดังนี้

1.1. ว่าจ้างบุคคลภายนอกองค์กรมาทำการพัฒนาระบบ โดยลักษณะการว่าจ้างนั้น อาจมีได้หลายรูปแบบ ตามความเหมาะสมขององค์กร เช่น

- พัฒนาทั้งโครงการ โดยการให้นักวิเคราะห์ระบบจากภายนอกเข้ามาดำเนินการพัฒนาระบบขององค์กรทั้งหมดตั้งแต่ต้นจนเสร็จสิ้นโครงการ ซึ่งจะมีคนในองค์กรคอยให้ข้อมูลแก่นักเคราะห์ระบบทีมนั้น ลักษณะการจ้างงานแบบนี้ จะเป็นโครงการ (Project) ที่ไม่ได้เป็นลูกจ้างประจำ ดังนั้นเมื่อทำการพัฒนาเสร็จสิ้นตามขั้นตอนที่ทีมพัฒนาระบบจากภายนอกได้วางแผนไว้แล้ว ก็หมดสัญญาการว่าจ้าง
- พัฒนาบางขั้นตอนของโครงการ เช่น ว่าจ้างเพื่อให้นักวิเคราะห์ระบบดำเนินการเพียงการวิเคราะห์ปัญหาที่ได้กำหนดไว้แล้วโดยทีมงานในองค์กรเอง หรือ ว่าจ้างเพื่อดำเนินการเพียงขั้นตอนของการออกแบบระบบ เป็นต้น

1.2. การซื้อโปรแกรมประยุกต์สำเร็จรูป (Application Software Package) มาใช้ในระบบ

ข้อดีของการเลือก Outsourcing

1. ลดต้นทุน (Cost Reduction) เนื่องจากกลุ่มบุคคลผู้รับพัฒนาระบบภายนอกองค์กรบางรายมีการเตรียมวิธีการแก้ปัญหา (Solution) ไว้บ้างแล้ว หากองค์กรเลือกใช้ เพียงแต่แก้ไขเล็กน้อยเท่านั้น กลุ่มธุรกิจเหล่านี้มีการแข่งขันสูงจึงมักเสนอเทคโนโลยีสมัยใหม่ในราคาถูกเพื่อเรียกร้องความสนใจจากลูกค้า
2. สามารถเลือก Outsourcing ได้ เพราะกลุ่มบุคคลผู้รับพัฒนาระบบภายนอกองค์กรแต่ละรายนั้นต่างก็แข่งขันเพื่อการครอบครองลูกค้าให้ได้มากที่สุด ดังนั้นกลุ่มที่ได้เปรียบคือกลุ่มองค์กรที่ต้องการพัฒนาระบบ เพราะมีโอกาสเลือกบริษัทผู้รับพัฒนาระบบภายนอกที่เหมาะสมและดีที่สุดสำหรับองค์กร
3. สามารถควบคุมงบประมาณได้ เนื่องจากก่อนที่องค์กรจะตกลงกับ Outsourcing จะมีการสอบราคาตามงบประมาณที่ตั้งไว้เท่านั้น

ข้อเสียของการเลือก Outsourcing

1. มีความเสี่ยงที่ข้อมูลขององค์กรอาจถูกเปิดเผย เนื่องจากทีมพัฒนาระบบมาจากบุคคลภายนอก
2. การบำรุงรักษาระบบไม่เต็มที่ เนื่องจากทีมพัฒนาระบบไม่ได้ประจำอยู่ในองค์กร หากเลือกกลุ่ม Outsourcing ที่มีบริการหลังการขายไม่ดี จะทำให้การทำงานหลังจากติดตั้งระบบแล้วเป็นไปค่อนข้างลำบาก

6.2. การพัฒนาการขององค์กร

การพัฒนาการขององค์กรเป็นอีกปัจจัยหนึ่งที่มีผลกระทบต่อนักวิเคราะห์ระบบ เนื่องจากเป็นการเปลี่ยนแปลงเพื่อก้าวไปสู่คุณภาพในระดับสากลขององค์กร โดยอาจเป็นการเปลี่ยนแปลงในด้านขั้นตอนการดำเนินงาน การนำเทคโนโลยีต่างๆ เข้ามาใช้เพื่อเพิ่มประสิทธิภาพของการทำงาน หรืออาจเป็นการนำกลยุทธ์

การพัฒนาองค์กรในด้านต่างๆ เข้ามาใช้ เป็นต้น ทำให้นักวิเคราะห์จะต้องปรับตัวตามการเปลี่ยนแปลงดังนี้ เนื่องจากนักวิเคราะห์ระบบสมัยใหม่ถูกกำหนดให้เป็นผู้ช่วยแก้ปัญหาต่างๆ ให้กับองค์กรนั่นเอง ตัวอย่างการพัฒนาการขององค์กรได้แก่ TQM, BPR, CPI เป็นต้น

TQM (Total Quality Management : TQM) การบริหารคุณภาพรวม คือ กลยุทธที่ใช้ในการพัฒนาองค์กรไปสู่การจัดการที่มีคุณภาพ ทั้งด้านกระบวนการและการดำเนินงาน โดยจะต้องสร้างความเชื่อมั่นว่างานที่ได้จะต้องมีคุณภาพ

คำว่า “คุณภาพ” ได้กลายมาเป็นปัจจัยสำคัญของชัยชนะในการแข่งขันทางธุรกิจ ดังนั้นองค์กรจึงหันมาสนใจในการพัฒนาสินค้าให้มีคุณภาพกันมากขึ้น แต่การที่จะทำให้สินค้ามีคุณภาพนั้น องค์กรควรหาวิธีที่จะทำให้ “บุคลากร” ขององค์กรเองตระหนักเสียก่อนว่า “คุณภาพของตนเอง” นั้นเป็นสิ่งแรกที่จะทำให้องค์กรและสินค้ามีคุณภาพได้

TQM ส่งผลกระทบต่อนักวิเคราะห์ระบบอย่างน้อย 2 ประการ

1. ทำให้นักวิเคราะห์ระบบต้องทำการกำหนดถึงปัญหาซึ่งเป็นเหตุให้เกิดการพัฒนาคุณภาพของการจัดการ
2. ทำให้นักวิเคราะห์ระบบจะต้องวิเคราะห์และออกแบบระบบได้อย่างสมบูรณ์แบบที่สุด เพราะหากนักวิเคราะห์ระบบทำงานผิดพลาดในส่วนของซอฟต์แวร์ จะส่งผลให้ขาดคุณภาพในการจัดการทันที

จากอิทธิพลของ TQM ส่งผลให้ธุรกิจต่างๆ เริ่มพยายามคิดค้นวิธีการสร้างคุณภาพขององค์กรให้เกิดขึ้น คือ การคิดใหม่และออกแบบใหม่ (Rethink/Redesign) ในเรื่องของกระบวนการดำเนินงานพื้นฐานทางธุรกิจ วิธีการนี้เรียกว่า “BPR”

BPR (Business Process Redesign) คือ การศึกษา วิเคราะห์ ถึงพื้นฐานของกระบวนการดำเนินธุรกิจ เพื่อทำการออกแบบใหม่ ซึ่งจะช่วยลดต้นทุนและปรับปรุงการดำเนินงานของธุรกิจให้ดียิ่งขึ้น โดยอาศัยเทคโนโลยีสารสนเทศ

นักวิเคราะห์ระบบมีบทบาทอย่างยิ่งในการร่วมโครงการ BPR นี้ กล่าวคือ นักวิเคราะห์ระบบถูกมองว่าเป็นผู้ที่มีทักษะในการวิเคราะห์และออกแบบระบบ ซึ่งความสำคัญของ BPR คือ “ระบบ” นั่นเอง อย่างไรก็ตาม กระบวนการดำเนินธุรกิจจะสัมฤทธิ์ผลได้ ต่อเมื่อมีการควบคุมกระบวนการทางธุรกิจอย่างต่อเนื่อง ที่เรียกว่า “CPI”

CPI (Continuous Process Improvement) คือ การควบคุมกระบวนการทางธุรกิจอย่างต่อเนื่อง เพื่อลดต้นทุน ปรับปรุงการดำเนินงาน ประสิทธิภาพ และ เพิ่มผลกำไร

กล่าวคือ CPI เป็นการเปลี่ยนแปลงบางส่วนของกระบวนการทางธุรกิจ แต่ BPR เป็นการออกแบบใหม่ทั้งหมด นักวิเคราะห์ระบบมีส่วนช่วยในการปรับปรุงกระบวนการนี้รวมไปถึงการออกแบบและพัฒนาโปรแกรมเพื่อการปรับปรุงนั้นด้วย

7. โอกาสในอาชีพนักวิเคราะห์ระบบสมัยใหม่

การก้าวเข้ามาสู่อาชีพนักวิเคราะห์ระบบถือได้ว่าเข้ามาสู่วงการของการพัฒนาระบบสารสนเทศ ซึ่งการทำงานอย่างเต็มความสามารถ และการเก็บเกี่ยวประสบการณ์ในการทำงาน พัฒนาความสามารถในการ

ทำงานอย่างสม่ำเสมอ จะทำให้สามารถเลื่อนตำแหน่งจาก นักวิเคราะห์ระบบ หรือ โปรแกรมเมอร์ ไปยังลำดับที่สูงขึ้นต่อไปได้ในอนาคต ดังเช่นตัวอย่างของตำแหน่งที่พัฒนาจากนักวิเคราะห์ระบบ ได้แก่

1. นักวิเคราะห์ระบบ (Junior Systems Analyst/Programmer,Junior Applications Programmer)
2. นักวิเคราะห์ระบบอาวุโส (Senior Systems Analyst)
3. ผู้บริหารจัดการฐานข้อมูล (Database Administrator)
4. ผู้ชำนาญการด้านความปลอดภัยบนระบบคอมพิวเตอร์ (Computer Security Specitalist)
5. วิทยากรฝึกอบรม (Training Specialist)
6. ผู้บริหารโครงการ (Project Manager)
7. ผู้จัดการระบบสารสนเทศ (Information Systems Manager)
8. ผู้ชำนาญการด้านเทคนิค (Technical Specialists for Database, Telecommunications, Microcomputers)
9. นักวิเคราะห์ระบบอาวุโสด้านสนับสนุนเทคนิค (Senior Technical Support Analyst)
10. ผู้บริหารสารสนเทศระดับสูง (Vice-President of MIS หรือ บางครั้งอาจเรียกว่า Chief Information Officer: CIO)

บทที่ 2 ความรู้เบื้องต้นเกี่ยวกับระบบสารสนเทศ (Introduction to Information System)

ในปัจจุบันได้เข้ามามีบทบาทต่อการดำเนินธุรกิจมากขึ้น ทำให้หน่วยงานธุรกิจทั้งหลายจำเป็นต้องจัดสรรงบประมาณส่วนหนึ่งไว้ เพื่อการจัดการกับข้อมูลสารสนเทศโดยเฉพาะ องค์การต่างๆ มีการใช้เทคโนโลยีสารสนเทศเพื่อหาความได้เปรียบในเชิงคู่แข่งขันกับองค์การอื่นๆ โดยระบบสารสนเทศจะมีอิธิพลมากต่อวิธีจัดองค์การและกระบวนการดำเนินการในหน้าที่ต่างๆ ทางธุรกิจ ความสัมพันธ์ระหว่างองค์การและการวางแผนระบบสารสนเทศ เพื่อการจัดการกลายเป็นกิจกรรมที่มีความสำคัญในลำดับสูง และค่อยๆ กลายเป็นกิจกรรมที่มีความสำคัญยิ่งในปัจจุบันนี้เพราะว่า

- องค์การต่างๆ ได้พบว่าสามารถใช้ระบบสารสนเทศเพื่อการจัดการ เพื่อความได้เปรียบในเชิงแข่งขัน
- องค์การต่างๆ สามารถใช้ระบบสารสนเทศในการจัดการเพื่อเพิ่มผลผลิต
- ผู้บริหารองค์การได้ตระหนักถึงความสำคัญเชิงกลยุทธ์ของการบูรณาการฐานข้อมูลที่เป็นประโยชน์ และทำการเผยแพร่สารสนเทศขององค์การมากขึ้น

เนื่องจากเทคโนโลยีเหล่านี้สามารถทำให้ผู้ประกอบการได้รับข้อมูลต่างๆ เพื่อประกอบการตัดสินใจได้รวดเร็ว ทันเหตุการณ์ ประกอบกับคอมพิวเตอร์ในปัจจุบันมีราคาที่ต่ำลง ดังนั้นจึงสังเกตได้ว่า ในปัจจุบันไม่ว่าจะเป็นองค์การหรือหน่วยงานเล็กๆ ต่างก็นำคอมพิวเตอร์มาเป็น

เครื่องมือในการใช้งานอยู่ทั่วไป ประกอบกับเทคโนโลยีคอมพิวเตอร์มีความล้ำหน้าทุกขณะ และไม่ได้ถูกจำกัดการใช้งานเฉพาะงานด้านวิทยาศาสตร์เช่นแต่ก่อนอีกต่อไป

2.1. ความแตกต่างระหว่างข้อมูลกับสารสนเทศ

1. ข้อมูล (Data) คือ เหตุการณ์จริงที่เกิดขึ้นประจำวันในการดำเนินธุรกิจขององค์กร เช่น รายการสั่งซื้อสินค้าจากลูกค้า รายการส่งสินค้า ชื่อที่อยู่ลูกค้า ยอดขายในแต่ละวัน เป็นต้น ข้อมูลอาจเป็นได้หลายชนิด เช่น ตัวเลข ตัวอักษร รูปภาพ รูปถ่าย หรือแม้กระทั่งเสียง
2. สารสนเทศ (Information) คือ ข้อมูลที่ผ่านกระบวนการเก็บรวบรวมและเรียบเรียง เพื่อเป็นแหล่งข้อมูลที่เป็นประโยชน์ต่อผู้ใช้ เช่นการนำเสนอยอดขายรายเดือนต่อผู้บริหาร ซึ่งยอดขายรายเดือนนั้นได้มาจากการรวบรวมยอดขายของตัวแทนขายในแต่ละวัน สารสนเทศที่ดี จะช่วยให้ผู้บริหารสามารถตัดสินใจได้ถูกต้องแม่นยำขึ้น และช่วยให้การประมาณการในด้านต่างๆ ไม่ว่าจะเป็นการลงทุนหรือยอดขาย ใกล้เคียงกับความเป็นจริงที่จะเกิดขึ้นได้มากที่สุด

2.2.ชนิดของระบบสารสนเทศ

ปัจจุบันระบบสารสนเทศได้รับการพัฒนาขึ้นให้เป็นเครื่องมือที่ช่วยในการทำงานทางด้านต่างๆ มากมาย ไม่ว่าจะเป็นด้านการตัดสินใจเพื่อแก้ปัญหาทางธุรกิจ ช่วยในการทำรายงานต่างๆ เพื่อนำเสนอข้อมูล ช่วยประมวลผลข้อมูลที่เกิดขึ้นประจำวันในธุรกิจ ช่วยวิเคราะห์หาทางออกของปัญหา เป็นต้น ระบบสารสนเทศแต่ละชนิด มีดังนี้

1. **ระบบการประมวลผลข้อมูล** (Transaction Processing Systems: TPS)

เป็นระบบที่เกี่ยวข้องกับการดำเนินงานประจำที่ต้องทำในองค์การ เช่น การบันทึกยอดขายแต่ละวัน การบันทึกการทำงานของพนักงานประจำที่ต้องทำในองค์การ เช่น การบันทึกยอดขายแต่ละวัน การบันทึกการทำงานของพนักงานในแต่ละวัน การบันทึกการสั่งสินค้าในแต่ละวัน ซึ่งรายงานต่างๆ ที่บันทึกในแต่ละวันนั้น จะเป็นการปฏิบัติงานที่ซ้ำๆ กันทุกวัน โดยข้อมูลประจำวันเหล่านี้ จะทำการรวบรวมเพื่อนำไปจัดทำรายงานที่ต้องการต่อไป

คุณลักษณะของระบบการประมวลผลข้อมูล

1. สามารถจัดเก็บข้อมูลที่เกิดขึ้นประจำวันของการดำเนินธุรกิจได้ เช่น ประวัติลูกค้า รายการสั่งซื้อสินค้าจากลูกค้า
2. สามารถสร้างข้อมูลเพื่อดำเนินธุรกิจได้ เช่น ออกใบกำกับภาษี ออกใบแจ้งหนี้ ออกใบรายการสินค้า
3. บำรุงรักษาข้อมูล (Data Maintenance) โดยการปรับปรุงข้อมูล (เพิ่ม ลบ แก้ไข) ให้เป็นปัจจุบันมากที่สุดไม่ว่าจะเป็นการเปลี่ยนแปลงของราคาสินค้า ชื่อที่อยู่ของลูกค้า รหัสสินค้า เป็นต้น

สำหรับนักวิเคราะห์ระบบที่ทำการวิเคราะห์และออกแบบการประมวลผลข้อมูลนี้ สิ่งที่ต้องคำนึงถึง ได้แก่

1. เวลาที่ใช้ในการตอบสนองการทำงาน (Response time) ต้องมีความรวดเร็ว
2. ความสามารถในการประมวลผลข้อมูลจำนวนมาก
3. ความถูกต้อง (Accuracy)
4. ความสอดคล้องของข้อมูล (Consistency) กรณีที่มีการประมวลผลพร้อมกันจากผู้ใช้หลายคน

2. **ระบบสารสนเทศเพื่อการจัดการ** (Management Information Systems: MIS)

เป็นระบบสารสนเทศที่ตอบสนองความต้องการของผู้ใช้งานด้วยการจัดทำรายงานที่ช่วยในการตัดสินใจที่เกี่ยวข้องกับการบริหาร ซึ่งข้อมูลในรายงานจะเป็นในลักษณะของการสรุปผลที่ได้จากข้อมูลต่างๆ ที่ถูกจัดเก็บใน TPS รายงานประเภทนี้อาจจะแสดงผลทั้งในรูปแบบของรายงานผลสรุป หรือรายงานรายละเอียดเพื่อไว้พิจารณาประกอบได้ โดยส่วนใหญ่แล้ว MIS มักมีการจัดทำเพื่อส่งไปยังแผนกต่างๆ ตามระยะเวลา เช่น ทุกวัน ทุกสัปดาห์ หรือทุกๆ เดือน เป็นต้น

สำหรับรายงานที่ระบบสารสนเทศเพื่อการจัดการสามารถจัดเตรียมไว้ได้นั้นแบ่งออกได้ดังนี้

1. รายงานตามกำหนดการ (Scheduled Reports) เป็นรายงานที่มีการกำหนดไว้แล้วตามแผนการดำเนินงานของธุรกิจว่าจะต้องมีการนำเสนอเป็นในช่วงเวลาใดเวลาหนึ่ง เช่น รายงานรายสัปดาห์ (Weekly Report) รายงานรายเดือน (Monthly Report) รายงานรายปี (Annual Report)
2. รายงานตามความต้องการ (Demand Reports) เป็นรายงานที่ถูกสร้างขึ้นเมื่อต้องการใช้งาน เช่น การจัดเตรียมสารสนเทศที่เป็นยอดคงเหลือของวัตถุดิบคงคลัง เพื่อนำมาจัดทำรายงานวัตถุดิบคงคลัง สำหรับใช้ในการสั่งซื้อวัตถุดิบในการผลิตครั้งต่อไป
3. รายงานกรณีเฉพาะ (Exception Report) เป็นรายงานที่จัดทำขึ้นในกรณีพิเศษ ที่ไม่มีปรากฏในแผนงาน เช่น ในกรณีมีการหยุดงานของพนักงานมากผิดปกติจนทำให้กำลังการผลิตลดลง ผู้บริหารอาจจะต้องการดูรายงานการลาหยุดเฉพาะพนักงานที่มีจำนวนวันลาหยุดมากเกินไป และสามารถดูรายงานกำลังการผลิตที่ลดลงด้วย จะเห็นว่ารายงานประเภทนี้มักจะมีเงื่อนไขในการจัดทำรายงานที่นอกเหนือจากที่มีอยู่แล้ว
4. รายงานพยากรณ์ (Prediction Report) เป็นรายงานที่เกิดจากการประมาณ คาดคะเน หรือพยากรณ์เหตุการณ์ล่วงหน้า เช่น รายงานการประมาณยอดขายที่เพิ่มขึ้นในปีถัดไป รายงานการประมาณกำลังการผลิต เป็นต้น

คุณลักษณะของระบบสารสนเทศเพื่อการจัดการ

1. สามารถสร้างสารสนเทศที่อ้างอิงได้ตามหลักการด้านการจัดการ ด้านคณิตศาสตร์ หรือสถิติ ที่เป็นที่ยอมรับได้
2. โดยปกติแล้วสารสนเทศเพื่อการจัดการนี้ได้มาจากฐานข้อมูลที่มีการเก็บข้อมูลจากแหล่งข้อมูลมากมาย ซึ่งแหล่งข้อมูลนั้นหมายรวมถึงระบบการประมวลผลข้อมูลด้วย
3. มีการเตรียมสารสนเทศในรูปแบบต่างๆ ได้ 4 ประการดังนี้
   3.1. สารสนเทศส่วนที่เป็นรายละเอียด (Detailed Information) สารสนเทศลักษณะนี้ใช้เพื่อการจัดการการปฏิบัติงานและเพื่อความต้องการการควบคุมการปฏิบัติงาน
   3.2. สารสนเทศส่วนที่เป็นผลสรุป (Summary Information) เป็นสารสนเทศที่เกิดจากการรวบรวมข้อมูลดิบ เพื่อนำไปใช้ในการวิเคราะห์แนวโน้มและความเป็นไปได้ที่จะเกิดปัญหาในด้านต่างๆ

3.3. สารสนเทศกรณีเฉพาะ (Exception Information) เป็นสารสนเทศที่เกิดจากการกรองข้อมูลตามเงื่อนไขที่ผู้ใช้ต้องการแล้ว เพื่อนำไปสร้างเป็นรายงานกรณีเฉพาะ (Exception Report) ต่อไป

3.4. สารสนเทศเพื่อการพยากรณ์ (Prediction Information) เป็นสารสนเทศที่มีการคำนวณเพื่อนำไปใช้ในการสร้างรายงานในการคาดคะเนผลประกอบการขององค์กรหรือการคาดคะเนปริมาณการผลิตที่แท้จริงของปีถัดไป

บทที่ 2 ความรู้เบื้องต้นเกี่ยวกับระบบสารสนเทศ (Introduction to Information System)

ในปัจจุบันได้เข้ามามีบทบาทต่อการดำเนินธุรกิจมากขึ้น ทำให้หน่วยงานธุรกิจทั้งหลายจำเป็นต้องจัดสรรงบประมาณส่วนหนึ่งไว้ เพื่อการจัดการกับข้อมูลสารสนเทศโดยเฉพาะ องค์การต่างๆ มีการใช้เทคโนโลยีสารสนเทศเพื่อหาความได้เปรียบในเชิงคู่แข่งขันกับองค์การอื่นๆ โดยระบบสารสนเทศจะมีอิธิพลมากต่อวิธีจัดองค์การและกระบวนการดำเนินการในหน้าที่ต่างๆ ทางธุรกิจ ความสัมพันธ์ระหว่างองค์การและการวางแผนระบบสารสนเทศ เพื่อการจัดการกลายเป็นกิจกรรมที่มีความสำคัญในลำดับสูง และค่อยๆ กลายเป็นกิจกรรมที่มีความสำคัญยิ่งในปัจจุบันนี้เพราะว่า

- องค์การต่างๆ ได้พบว่าสามารถใช้ระบบสารสนเทศเพื่อการจัดการ เพื่อความได้เปรียบในเชิงแข่งขัน
- องค์การต่างๆ สามารถใช้ระบบสารสนเทศในการจัดการเพื่อเพิ่มผลผลิต
- ผู้บริหารองค์การได้ตระหนักถึงความสำคัญเชิงกลยุทธ์ของการบูรณาการฐานข้อมูลที่เป็นประโยชน์ และทำการเผยแพร่สารสนเทศขององค์การมากขึ้น

เนื่องจากเทคโนโลยีเหล่านี้สามารถทำให้ผู้ประกอบการได้รับข้อมูลต่างๆ เพื่อประกอบการตัดสินใจได้รวดเร็ว ทันเหตุการณ์ ประกอบกับคอมพิวเตอร์ในปัจจุบันมีราคาที่ต่ำลง ดังนั้นจึงสังเกตได้ว่า ในปัจจุบันไม่ว่าจะเป็นองค์การหรือหน่วยงานเล็กๆ ต่างก็นำคอมพิวเตอร์มาเป็นเครื่องมือในการใช้งานอยู่ทั่วไป ประกอบกับเทคโนโลยีคอมพิวเตอร์มีความล้ำหน้าทุกขณะและไม่ได้ถูกจำกัดการใช้งานเฉพาะงานด้านวิทยาศาสตร์เช่นแต่ก่อนอีกต่อไป

2.1. ความแตกต่างระหว่างข้อมูลกับสารสนเทศ

1. ข้อมูล (Data) คือ เหตุการณ์จริงที่เกิดขึ้นประจำวันในการดำเนินธุรกิจขององค์กร เช่น รายการสั่งซื้อสินค้าจากลูกค้า รายการส่งสินค้า ชื่อที่อยู่ลูกค้า ยอดขายในแต่ละวัน เป็นต้น ข้อมูลอาจเป็นได้หลายชนิด เช่น ตัวเลข ตัวอักษร รูปภาพ รูปถ่าย หรือแม้กระทั่งเสียง
2. สารสนเทศ (Information) คือ ข้อมูลที่ผ่านกระบวนการเก็บรวบรวมและเรียบเรียง เพื่อเป็นแหล่งข้อมูลที่เป็นประโยชน์ต่อผู้ใช้ เช่นการนำเสนอยอดขายรายเดือนต่อผู้บริหาร ซึ่งยอดขายรายเดือนนั้นได้มาจากการรวบรวมยอดขายของตัวแทนขายในแต่ละวัน สารสนเทศที่ดี จะช่วยให้ผู้บริหารสามารถตัดสินใจได้ถูกต้องแม่นยำขึ้น และช่วยให้การประมาณการในด้านต่างๆ ไม่ว่าจะเป็นการลงทุนหรือยอดขาย ใกล้เคียงกับความเป็นจริงที่จะเกิดขึ้นได้มากที่สุด

2.2.ชนิดของระบบสารสนเทศ

ปัจจุบันระบบสารสนเทศได้รับการพัฒนาขึ้นให้เป็นเครื่องมือที่ช่วยในการทำงานทางด้านต่างๆ มากมาย ไม่ว่าจะเป็นด้านการตัดสินใจเพื่อแก้ปัญหาทางธุรกิจ ช่วยในการทำรายงานต่างๆ เพื่อนำเสนอข้อมูล ช่วยประมวลผลข้อมูลที่เกิดขึ้นประจำวันในธุรกิจ ช่วยวิเคราะห์หาทางออกของปัญหา เป็นต้น ระบบสารสนเทศแต่ละชนิด มีดังนี้

1. **ระบบการประมวลผลข้อมูล** (Transaction Processing Systems: TPS)

เป็นระบบที่เกี่ยวข้องกับการดำเนินงานประจำที่ต้องทำในองค์การ เช่น การบันทึกยอดขายแต่ละวัน การบันทึกการทำงานของพนักงานประจำที่ต้องทำในองค์การ เช่น การบันทึกยอดขายแต่ละวัน การบันทึกการทำงานของพนักงานในแต่ละวัน การบันทึกการสั่งสินค้าในแต่ละวัน ซึ่งรายงานต่างๆ ที่บันทึกในแต่ละวันนั้น จะเป็นการปฏิบัติงานที่ซ้ำๆ กันทุกวัน โดยข้อมูลประจำวันเหล่านี้ จะทำการรวบรวมเพื่อนำไปจัดทำรายงานที่ต้องการต่อไป

คุณลักษณะของระบบการประมวลผลข้อมูล

1. สามารถจัดเก็บข้อมูลที่เกิดขึ้นประจำวันของการดำเนินธุรกิจได้ เช่น ประวัติลูกค้า รายการสั่งซื้อสินค้าจากลูกค้า
2. สามารถสร้างข้อมูลเพื่อดำเนินธุรกิจได้ เช่น ออกใบกำกับภาษี ออกใบแจ้งหนี้ ออกใบรายการสินค้า
3. บำรุงรักษาข้อมูล (Data Maintenance) โดยการปรับปรุงข้อมูล (เพิ่ม ลบ แก้ไข) ให้เป็นปัจจุบันมากที่สุดไม่ว่าจะเป็นการเปลี่ยนแปลงของราคาสินค้า ชื่อที่อยู่ของลูกค้า รหัสสินค้า เป็นต้น

สำหรับนักวิเคราะห์ระบบที่ทำการวิเคราะห์และออกแบบการประมวลผลข้อมูลนี้ สิ่งที่ต้องคำนึงถึง ได้แก่

1. เวลาที่ใช้ในการตอบสนองการทำงาน (Response time) ต้องมีความรวดเร็ว
2. ความสามารถในการประมวลผลข้อมูลจำนวนมาก
3. ความถูกต้อง (Accuracy)
4. ความสอดคล้องของข้อมูล (Consistency) กรณีที่มีการประมวลผลพร้อมกันจากผู้ใช้หลายคน

2. **ระบบสารสนเทศเพื่อการจัดการ** (Management Information Systems: MIS)

เป็นระบบสารสนเทศที่ตอบสนองความต้องการของผู้ใช้งานด้วยการจัดทำรายงานที่ช่วยในการตัดสินใจที่เกี่ยวข้องกับการบริหาร ซึ่งข้อมูลในรายงานจะเป็นในลักษณะของการสรุปผลที่ได้จากข้อมูลต่างๆ ที่ถูกจัดเก็บใน TPS รายงานประเภทนี้อาจจะแสดงผลทั้งในรูปแบบของรายงานผลสรุป หรือรายงานรายละเอียดเพื่อไว้พิจารณาประกอบได้ โดยส่วนใหญ่แล้ว MIS มักมีการจัดทำเพื่อส่งไปยังแผนกต่างๆ ตามระยะเวลา เช่น ทุกวัน ทุกสัปดาห์ หรือทุกๆ เดือน เป็นต้น

สำหรับรายงานที่ระบบสารสนเทศเพื่อการจัดการสามารถจัดเตรียมไว้ได้นั้นแบ่งออกได้ดังนี้

1. รายงานตามกำหนดการ (Scheduled Reports) เป็นรายงานที่มีการกำหนดไว้แล้วตามแผนการดำเนินงานของธุรกิจว่าจะต้องมีการนำเสนอเป็นในช่วงเวลาใดเวลาหนึ่ง เช่น รายงานรายสัปดาห์ (Weekly Report) รายงานรายเดือน (Monthly Report) รายงานรายปี (Annual Report)
2. รายงานตามความต้องการ (Demand Reports) เป็นรายงานที่ถูกสร้างขึ้นเมื่อต้องการใช้งาน เช่น การจัดเตรียมสารสนเทศที่เป็นยอดคงเหลือของวัตถุดิบคงคลัง เพื่อนำมาจัดทำรายงานวัตถุดิบคงคลัง สำหรับใช้ในการสั่งซื้อวัตถุดิบในการผลิตครั้งต่อไป
3. รายงานกรณีเฉพาะ (Exception Report) เป็นรายงานที่จัดทำขึ้นในกรณีพิเศษ ที่ไม่มีปรากฏในแผนงาน เช่น ในกรณีมีการหยุดงานของพนักงานมากผิดปกติจนทำให้กำลังการผลิตลดลง ผู้บริหารอาจจะต้องการดูรายงานการลาหยุดเฉพาะพนักงานที่มีจำนวนวันลาหยุดมากเกินไป และ

สามารถดูรายงานกำลังการผลิตที่ลดลงด้วย จะเห็นว่ารายงานประเภทนี้มักจะมีเงื่อนไขในการจัดทำรายงานที่นอกเหนือจากที่มีอยู่แล้ว

4. รายงานพยากรณ์ (Prediction Report) เป็นรายงานที่เกิดจากการประมาณ คาดคะเน หรือพยากรณ์เหตุการณ์ล่วงหน้า เช่น รายงานการประมาณยอดขายที่เพิ่มขึ้นในปีถัดไป รายงานการประมาณกำลังการผลิต เป็นต้น

คุณลักษณะของระบบสารสนเทศเพื่อการจัดการ

1. สามารถสร้างสารสนเทศที่อ้างอิงได้ตามหลักการด้านการจัดการ ด้านคณิตศาสตร์ หรือสถิติ ที่เป็นที่ยอมรับได้
2. โดยปกติแล้วสารสนเทศเพื่อการจัดการนี้ได้มาจากฐานข้อมูลที่มีการเก็บข้อมูลจากแหล่งข้อมูลมากมาย ซึ่งแหล่งข้อมูลนั้นหมายรวมถึงระบบการประมวลผลข้อมูลด้วย
3. มีการเตรียมสารสนเทศในรูปแบบต่างๆ ได้ 4 ประการดังนี้
   3.1. สารสนเทศส่วนที่เป็นรายละเอียด (Detailed Information) สารสนเทศลักษณะนี้ใช้เพื่อการจัดการการปฏิบัติงานและเพื่อความต้องการการควบคุมการปฏิบัติงาน
   3.2. สารสนเทศส่วนที่เป็นผลสรุป (Summary Information) เป็นสารสนเทศที่เกิดจากการรวบรวมข้อมูลดิบ เพื่อนำไปใช้ในการวิเคราะห์แนวโน้มและความเป็นไปได้ที่จะเกิดปัญหาในด้านต่างๆ
   3.3. สารสนเทศกรณีเฉพาะ (Exception Information) เป็นสารสนเทศที่เกิดจากการกรองข้อมูลตามเงื่อนไขที่ผู้ใช้ต้องการแล้ว เพื่อนำไปสร้างเป็นรายงานกรณีเฉพาะ (Exception Report) ต่อไป
   3.4. สารสนเทศเพื่อการพยากรณ์ (Prediction Information) เป็นสารสนเทศที่มีการคำนวณเพื่อนำไปใช้ในการสร้างรายงานในการคาดคะเนผลประกอบการขององค์กรหรือการคาดคะเนปริมาณการผลิตที่แท้จริงของปีถัดไป

3. **ระบบสนับสนุนการตัดสินใจ** (Decision Support Systems :DSS)

เป็นระบบที่ช่วยสนับสนุนการตัดสินใจที่ตอบสนองความต้องการของผู้บริหาร ด้วยการจัดทำรายงานเพื่อใช้ประโยชน์ต่อการตัดสินใจของผู้บริหารในระดับต่างๆ ด้วยการสามารถนำข้อมูลที่ได้มาวิเคราะห์ผล เพื่อช่วยในการตัดสินใจแก้ปัญหา ระบบสนับสนุนการตัดสินใจนี้ยังสามารถทำการปรับเปลี่ยนตัวแปรต่างๆ และวิเคราะห์ผลใหม่ เพื่อนำมาประกอบเป็นทางเลือกการตัดสินใจได้หลายๆ ทาง จุดนี้ทำให้ DSS มีความแตกต่างจาก TPS และ MIS คือ TPS และ MIS จะแสดงรายละเอียดหรือผลสรุปของข้อมูลที่เกิดขึ้นจริงจากเหตุการณ์ในประจำวัน

คุณลักษณะของระบบสนับสนุนการตัดสินใจ

1. จัดเตรียมสารสนเทศซึ่งได้ทำการประมวลผลแล้วจากระบบประมวลผลข้อมูล (TPS) เพื่อช่วยในการตัดสินใจ

2. สนับสนุนการตัดสินใจแบบไม่มีโครงสร้าง (Unstructured Decisions) หรือแบบกึ่งโครงสร้าง (Semi-Structured Decisions)
3. สนับสนุนการตัดสินใจของผู้ใช้ในด้านต่างๆ ดังนี้

   3.1. ระบุถึงปัญหาหรือโอกาสในการทำการตัดสินใจ

   3.2. ระบุถึงความเป็นไปได้ในการแก้ปัญหาหรือการตัดสินใจนั้นๆ

   3.3. เตรียมสารสนเทศที่จำเป็นต่อการแก้ปัญหานั้นหรือที่จำเป็นต่อการกระทำการตัดสินใจ

   3.4. ทำการวิเคราะห์ทางเลือกในการตัดสินใจที่เป็นไปได้

   3.5. เลียนแบบทางเลือกและผลลัพธ์ของการตัดสินในที่เป็นไปได้

   นอกจากคุณลักษณะดังกล่าวข้างต้นแล้ว จะเห็นว่าการทำงานของ DSS นั้นต้องอาศัยสารสนเทศจากฐานข้อมูล (Database) ซึ่งมีการเตรียมสารสนเทศที่เหมาะสำหรับการตัดสินใจไว้โดยเฉพาะเพื่อเพิ่มความรวดเร็วในการดึงสารสนเทศนั้นมาใช้งาน จึงได้มีการแยกฐานข้อมูลของสารสนเทศที่เตรียมไว้สำหรับ DSS และระบบอื่นที่เกี่ยวข้องโดยเฉพาะ เรียกฐานข้อมูลนั้นว่า “คลังข้อมูล (Data Warehouse)”

   คลังข้อมูล (Data Warehouse) คือ ฐานข้อมูลที่ได้มีการเตรียมสานสนเทศเพื่อระบบสนับสนุนการตัดสินใจไว้โดยเฉพาะมีลักษณะดังนี้

   - DSS สามารถอ่านสารสนเทศจากคลังข้อมูลได้อย่างเดียว (Read-only) ไม่สามารถแก้ไขสารสนเทศภายในคลังข้อมูลได้
   - เก็บสารสนเทศไว้ 3 ประเภท ได้แก่ สารสนเทศที่เป็นส่วนรายละเอียด (Details) ส่วนที่เป็นผลสรุป (Summary) และส่วนที่เป็นสารสนเทศกรณีเฉพาะ (Exception)
   - สามารถเข้าถึงข้อมูลได้ทั้งผู้ใช้ลำดับสุดท้าย (End-Users) และผู้บริหาร (Managers)
   - เตรียมเครื่องมือ (Tools) ที่สนับสนุนการตัดสินใจ เช่น โปรแกรมทางด้าน Spreadsheet (Excel) โปรแกรมทางด้านการจัดการฐานข้อมูล (Access) เครื่องมือในการสร้างรายงาน (Focus) และโปรแกรมวิเคราะห์งานทางสถิติ (SAS,SPSS)

4. ระบบผู้เชี่ยวชาญ (Expert System :ES)

เป็นระบบที่รวบรวมความรู้ความเชี่ยวชาญเฉพาะด้านต่างๆ เข้าด้วยกัน รวมทั้งปัจจัยต่างๆ ที่เกี่ยวข้อง สามารถนำเหตุการณ์ต่างๆ มาประมวลเป็นภาพรวมเพื่อให้คำตอบแก่ผู้ใช้ ซึ่งจะแตกต่างกับระบบสนับสนุนการตัดสินใจ ที่เสนอเพียงทางเลือกที่ดีและให้ผู้ใช้ตัดสินใจเอง

คุณสมบัติของระบบผู้เชี่ยวชาญ

1. ES จะทำการเลียนแบบวิธีการคิดและเหตุผลของผู้เชี่ยวชาญจากประสบการณ์ในการแก้ปัญหาจริงในด้านต่างๆ
2. อาจนำ ES มาใช้ร่วมงานกับเทคโนโลยีปัญญาประดิษฐ์ (Artificial Intelligence :AI) เรียกการทำงานร่วมกันว่า “Expert System Shells”

3. มีการดึงสารสนเทศจากคลังข้อมูลเช่นเดียวกับระบบสนับสนุนการตัดสินใจ (DSS)

จะเห็นได้ว่าระบบสนับสนุนการตัดสินใจ (DSS) กับระบบผู้เชี่ยวชาญ (ES) จะมีลักษณะคล้ายกัน แตกต่างกันตรงที่ DSS เป็นระบบที่เสนอทางเลือกในการแก้ปัญหาหรือตัดสินใจเพื่อให้ผู้ใช้ได้ตัดสินใจเอง ส่วน ES นั้นเป็นระบบที่ตัดสินใจแทนผู้ใช้โดยอาศัยสารสนเทศที่รวบรวมมาจากเหตุผลและประสบการณ์จริง

5. ระบบสารสนเทศเพื่อสำนักงาน (Office Information Systems: OIS)

เป็นระบบสารสนเทศในสำนักงานที่ใช้เทคโนโลยีคอมพิวเตอร์ เทคโนโลยีด้านการสื่อสารและเครือข่าย รวมถึงการใช้ซอฟต์แวร์ต่างๆ ที่เกี่ยวกับงานออฟฟิคมาใช้ในสำนักงาน เช่น MS-Office ที่ประกอบด้วยโปรแกรมประมวลผลคำ (Word Processing) โปรแกรมตารางงาน (Spread Sheet) และโปรแกรมเครือข่ายสำนักงาน เช่น MS-Outlook เป็นต้น

คุณลักษณะของระบบสารสนเทศเพื่อสำนักงาน

1. มีการเก็บรวบรวมสารสนเทศต่างๆ ที่เกี่ยวข้องกับกลุ่มบุคคลทุกกลุ่มไว้เพื่อการใช้งาน
2. ช่วยการทำงานอัตโนมัติด้านต่างๆ ได้แก่
    - การประมวลผลคำ (Word Processing)
    - ส่งข้อความอิเล็กทรอนิกส์ หรือ จดหมายอิเล็กทรอนิกส์
    - การทำงานร่วมกันเป็นเครือข่ายคอมพิวเตอร์ (Work Group Computing)
    - การกำหนดการทำงานร่วมกัน (Work Group Scheduling)
    - เอกสารอิเล็กทรอนิกส์หรือรูปภาพ
    - การจัดการกระแสการทำงาน (Work Flow Management)
3. มีการประยุกต์ใช้เทคโนโลยีร่วมกันระหว่าง OIS กับ TPS อันได้แก่
    - เทคโนโลยีการสร้างแบบฟอร์มอิเล็คทรอนิกส์ (Electronic Form Technology) เป็นการช่วยสร้างแบบฟอร์มอิเล็คทรอนิกส์เพื่อให้สามารถใช้งานร่วมกับฐานข้อมูลของระบบประมวลผลข้อมูล
    - เทคโนโลยีการทำงานร่วมกันเป็นกลุ่ม (Work Group Technology) เช่น โปรแกรม Lotus Notes เพื่อเตรียมวิธีการสำหรับการทำงานที่มีการใช้งานของผู้ใช้หลายคนเพื่อเข้าถึง และการปรับปรุงข้อมูลร่วมกันจากการทำงานที่เกิดขึ้นประจำวัน
    - เทคโนโลยีข้อความอิเล็คทรอนิกส์ (Electronic Messing Technology) พนักงานสามารถติดต่อสื่อสารกันได้ด้วยการส่งข้อความอิเล็คทรอนิกส์
    - เทคโนโลยีชุดโปรแกรมสำนักงานอัตโนมัติ (Office Automation Suite Technology) นำโปรแกรมที่ใช้ในสำนักงานมาประยุกต์ใช้ร่วมกัน
    - เทคโนโลยีรูปภาพ (Imaging Technology) เป็นการผสมผสานกันระหว่างรูปภาพและแบบฟอร์มอิเล็คทรอนิกส์เพื่อใช้ในการทำงาน หรือเป็นการสแกนรูปภาพนั่นเอง

6. **ระบบสารสนเทศส่วนบุคคลและสารสนเทศเพื่อการทำงานเป็นกลุ่ม** (Personal and Work Group Information Systems)
    - ระบบสารสนเทศส่วนบุคคล (Personal Information System : PIS) เป็นระบบที่ออกแบบมาเพื่อตอบสนองความต้องการเฉพาะบุคคลเพื่อเพิ่มผลผลิตในการทำงาน
    - ระบบสารสนเทศเพื่อการทำงานเป็นกลุ่ม (Work Group Information System :WIS) เป็นระบบที่ออกแบบมาเพื่อตอบสนองการทำงานที่เป็นกลุ่ม เพื่อเพิ่มผลผลิตในการทำงาน

คุณลักษณะของ PIS และ WIS

1. ทั้งสองระบบนี้สร้างขึ้นมาเพื่อรองรับการใช้งานของเทคโนโลยีคอมพิวเตอร์และซอฟต์แวร์
2. WIS เป็นระบบที่เชื่อมการทำงานระหว่างเครื่องคอมพิวเตอร์ส่วนบุคคลกับเครือข่ายคอมพิวเตอร์แบบ LAN

บทที่ 3 การบริหารโครงการ

3.1. พื้นฐานของโครงการ

โครงการ (Project) คือ กลุ่มของงานที่เกี่ยวข้องกัน และต้องปฏิบัติงานเหล่านั้นตามลำดับก่อนหลัง เพื่อให้บรรลุตามวัตถุประสงค์ที่กำหนดไว้ โดยมีจุดเริ่มต้นและจุดเสร็จสิ้นเพียงจุดเดียวซึ่งคำว่า งาน (Activity) หมายถึง งานที่เป็นส่วนหนึ่งของโครงการ ซึ่งต้องใช้เวลาและทรัพยากรในการทำงานนั้นๆ งานต้องมีจุดเริ่มต้นและจุดสิ้นสุดที่บ่งชี้ได้ ดังนั้นความสมบูรณ์ของโครงการดังกล่าวจะบรรลุได้ตามวัตถุประสงค์ จึงจำเป็นต้องมีการบริหารโครงการ (Project Management) การบริหารโครงการหมายถึง การรู้จักวางแผน และควบคุมโครงการ ให้โครงการนั้นเป็นไปตามวัตถุประสงค์ของโครงการตามระยะเวลาที่กำหนดอย่างมีประสิทธิภาพและประสิทธิผล

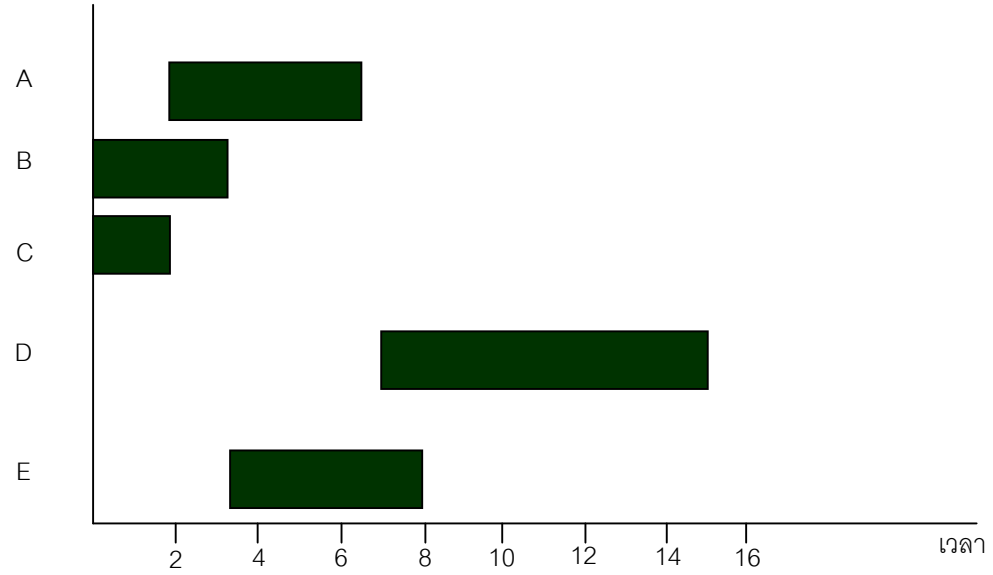
การเริ่มต้นโครงการนั้นสิ่งที่ต้องพิจารณาคือ พิจารณาถึงความเป็นไปได้ของโครงการ การจัดตารางเวลาทำงาน การบริหารกิจกรรม และสมาชิกในทีมที่ทำโครงการ ซึ่งทั้งหมดนี้ คือ ส่วนสำคัญในการทำให้โครงการนั้นเป็นไปได้อย่างราบรื่น และประสบความสำเร็จ ผู้บริหารโครงการต้องมีการควบคุมงานโดยการกำหนดตารางเวลาในการทำกิจกรรมต่างๆ ภายในโครงการ และบริหารให้กิจกรรมต่างๆนั้นได้ออกมาอย่างมีประสิทธิผล ณ เวลาที่กำหนด ซึ่งหลักการบริหารงานในลักษณะนี้ ผู้บริหารสามารถใช้วิธี ผังแกนท์(GANTT Charts) และแผนภาพเพิร์ธ (PERT Diagram) มาช่วยบริหารโครงการ

3.2. การใช้ผังแกนท์ (Gantt Chart) และแผนภาพเพิร์ธ (Pert Diagram)

ผังแกนท์ (Gantt Chart) คือ ผังที่ใช้เขียนแทนงานต่างๆ ของโครงการในรูปกราฟแท่งโดยแกน Y แทนงานต่างๆ ที่มีในโครงการนั้นและแกน X แทนเวลาในการทำงานของแต่ละงาน โดยการเขียนรูปแท่งแทนงานความยาวของแท่งเป็นสัดส่วนโดยตรงกับระยะเวลาการทำงาน และการเขียนแท่งกราฟต้องเรียงลำดับตามการวางแผนการทำงาน ผังแกนท์เป็นเครื่องมือที่ใช้ในการวางแผนและกำหนดเวลาในการทำงานของโครงการ ซึ่งเป็นผังที่ใช้งานง่ายไม่ซับซ้อน และในปัจจุบันก็ยังนิยมผังแกนท์นี้มาเป็นเครื่องมือการวางแผนโครงการ แต่อย่างไรก็ตาม ผังแกนท์นี้จะไม่ได้แสดงความสัมพันธ์ระหว่างงานให้เห็นได้อย่างชัดเจน และไม่สามารถบอกได้ว่างานที่ปฏิบัติการล่าช้าจะมีผลต่อโครงการด้วย ดังนั้นโครงการขนาดใหญ่ที่มีระบบงานที่กระจายเป็นระบบย่อยๆ และมีจำนวนมาก มีขั้นตอนการดำเนินงานที่ซับซ้อน จึงมักนำเทคนิคของเพิร์ต (PERT) และซีพีเอ็ม (CPM) มาประยุกต์การใช้งานมากกว่า

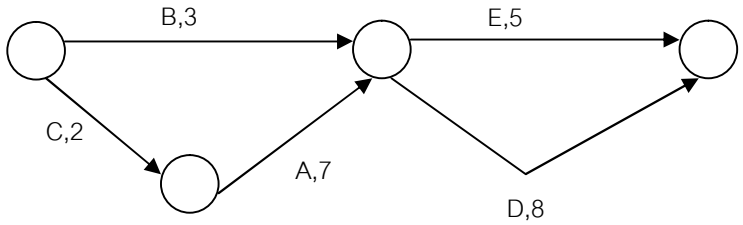
ตัวอย่าง

งาน	ระยะเวลาการทำงาน (วัน)	เริ่มหลังจากงาน (วัน)
A	5	C
B	3	-
C	2	-
D	8	A,B
E	5	B



แผนภาพเพิร์ธ (Pert Diagram) คือ แผนภาพที่เขียนแทนงาน โดยแต่ละงานต้องเขียนกำกับด้วยโหนด c(Node) เริ่มต้นงาน ตามด้วยเส้นที่แสดงชื่อกำกับงานคั่นด้วย “,” ตามด้วยระยะเวลาในการดำเนินงานและสิ้นสุดที่โหนดเขียนแทนด้วยวงกลมและมีตัวเลขกำกับ ส่วนตัวเลขกำกับจะเริ่มจากค่าน้อยแล้วเพิ่มค่ามากขึ้นในการเขียนกำกับแต่ละโหนด การบริหารโครงการด้วยการวางแผน ควบคุม โดยใช้เทคนิค PERT (Program Evaluation and Review Technique) และ CPM (Critical Path Method) เป็นการวิเคราะห์ข่ายงานที่มักนำมาใช้ในการบริหารโครงการ ที่มีจุดเริ่มต้นของโครงการจนถึงการปิดโครงการที่แน่นอน มีส่วนงานย่อยต่างๆ ที่มีการกระจายโดยมีความสัมพันธ์กันซึ่งกันและกัน วัตถุประสงค์และหลักการของ PERT และ CPM มีพื้นฐานที่คล้ายคลึงกัน โดย PERT จะเน้นด้านเวลาในการดำเนินโครงการ ส่วน CPM จะเน้นด้านค่าใช้จ่ายของโครงการ แต่ปัจจุบันได้มีการนำมาใช้ร่วมกัน โดยคำว่า PERT เพียงคำเดียว อาจหมายถึงการนำเทคนิคของ CPM มาใช้ร่วมด้วย

<u>ตัวอย่าง</u> แผนภาพเพิร์ธ (Pert Diagram) ข้อมูลจากแผนการดำเนินงานตารางด้านบน



สายงานที่ 1 B – E = 3+5 = 8

สายงานที่ 2 C – A – E = 2+7+5 = 14

สายงานที่ 3 C – A – D = 2+7+8 =17

วัตถุประสงค์ของ PERT

PERT เป็นแผนงานที่สามารถแสดงภาพรวมของโครงการด้วยข่ายงาน (Network) โดยแสดงกิจกรรมต่างๆในโครงการ ลำดับการทำงาน และความสัมพันธ์ระหว่างกิจกรรมต่างๆ ทั้งนี้มีวัตถุประสงค์เพื่อ

1. วางแผนโครงการ (Project Planning) โดยจะทำการคำนวณระยะเวลาการทำงาน และแสดงถึงกิจกรรมแต่ละกิจกรรมว่าควรเริ่มเมื่อใด แล้วเสร็จเมื่อใด และสามารถกำหนดได้ว่ากิจกรรมใดเป็นกิจกรรมสำคัญ ทำงานล่าช้าไม่ได้ หรือล่าช้าได้ไม่เกินเท่าใด
2. ควบคุมโครงการ (Project Control) สามารถควบคุมการทำงานตามแผนที่ได้วางไว้ และควบคุมการทำงานไม่ให้ล่าช้ากว่ากำหนด
3. บริหารทรัพยากร (Resource) กล่าวคือ สามารถใช้ทรัพยากรต่างๆ เช่น เงินลงทุน บุคลกร เครื่องมือ อุปกรณ์ และอื่นๆ ได้อย่างมีประสิทธิภาพและประโยชน์เต็มที่
4. บริหารโครงการ (Project Management) งานที่ดำเนินการอยู่อาจจำเป็นต้องเร่งการดำเนินการเพื่อแล้วเสร็จกว่ากำหนด ก็สามารถทำได้ด้วยการเร่งทำกิจกรรมใดบ้าง เพื่อให้งานเสร็จในระยะเวลาที่เร็วขึ้น

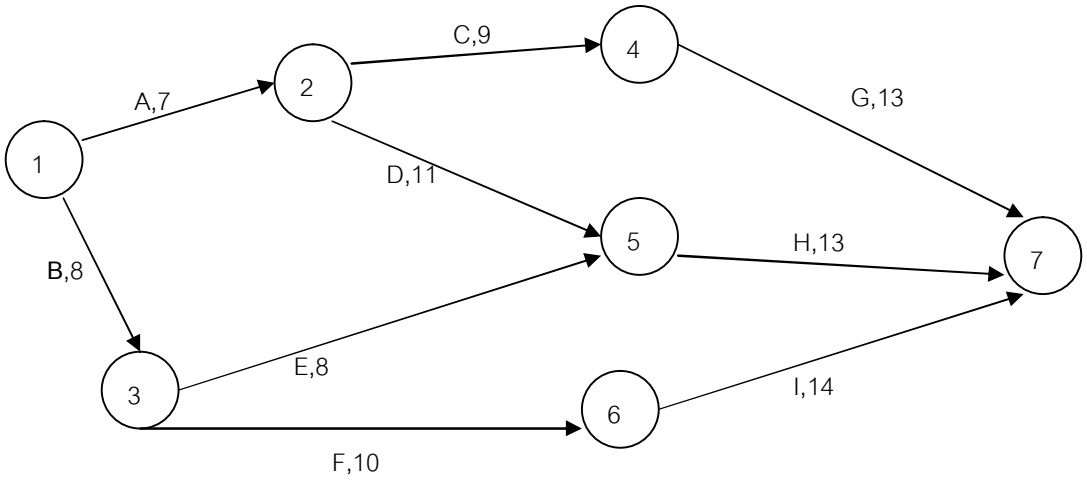
สายงานวิกฤต (Critical Paths) จะพิจารณาจากสายงานที่มีเวลารวมยาวนานที่สุด ซึ่งในที่นี้ คือสายงานที่ 3 คือ C – A – D รวมเวลาทั้งสิ้น 17 วัน นั่นหมายถึงการดำเนินงานทุกอย่างในแต่ละขั้นตอนจะแล้วเสร็จโดยใช้เวลา 17 วัน โดยในบางโครงการอาจมีสายงานวิกฤตมากกว่า 1 สายงาน

การเร่งโครงการ

สายงานวิกฤต คือ สายงานที่มีระยะยาวนานที่สุดซึ่งถือเป็นสายงานที่มีความสำคัญ หากงานหรือกิจกรรมภายในสายงานวิกฤตช้ากว่าที่กำหนดไว้ในโครงการ นั่นหมายถึงว่าโครงการก็จะเสร็จช้าไปด้วย ดังนั้นการควบคุมโครงการให้สำเร็จตามเวลาที่ได้กำหนดไว้ จำเป็นต้องมีการควบคุมกิจกรรมในสายงานวิกฤตให้เป็นไปตามที่ได้วางแผนไว้ ดังนั้นหากต้องการเร่งโครงการให้เสร็จเร็วขึ้น ก็สามารถทำได้ด้วยการเร่งกิจกรรมภายในสายงานวิกฤตนั่นเอง

งาน	งานที่ต้องเสร็จก่อน	ระยะเวลา (วัน)		ค่าใช้จ่ายในการเร่งงาน 1 วัน (บาท)
		ปกติ	เร่ง	
A	-	7	6	150
B	-	8	6	75
C	A	9	7	200
D	A	11	9	125
E	B	8	5	115
F	B	10	7	100
G	C	13	11	200
H	D,E	13	12	100

I	F	14	10	125



สายงานที่ 1 A – C – G = 7+9+13 = 29

สายงานที่ 2 A – D – H = 7+11+13 = 31

สายงานที่ 3 B – E – H = 8+8+13 =29

สายงานที่ 4 B – F – I = 8+10+14 =32 ---→ สายงานวิกฤต

จากการคำนวณระยะเวลา สายงานวิกฤต คือ B – F – I ซึ่งใช้เวลารวม 32 วัน หากต้องการเร่งโครงการให้เสร็จเร็วขึ้น จึงต้องจัดการกับสายงานนี้ โดยสมมติว่าถ้าต้องการให้โครงการเสร็จภายใน 28 วัน และเสียค่าใช้จ่ายน้อยที่สุด

สายงาน B – F – I ปรากฏว่ากิจกรรม B มีค่าใช้จ่ายต่อวันต่ำที่สุดดังนั้นจึงทำการเร่งกิจกรรม B จาก 8 วัน เหลือ 6 วัน

สายงานที่ 1 A – C – G = 7+9+13 = 29

สายงานที่ 2 A – D – H = 7+11+13 = 31 ---→ สายงานวิกฤต

สายงานที่ 3 B – E – H = 6+8+13 = 27

สายงานที่ 4 B – F – I = 6+10+14 = 30

หลังจากที่ได้ทำการเร่งกิจกรรม B ยังไม่ได้ทำให้โครงการเสร็จลงได้ตามเวลาที่กำหนดไว้ ดังนั้นจึงต้องเร่งกิจกรรมอื่น โดยสายงานวิกฤตในที่นี้คือ A – D – H โดยค่าใช้จ่ายของกิจกรรม H มีต่ำสุดคือวันละ 100 บาท ดังนั้นจึงเลือกกิจกรรม H ด้วยการเร่งเวลาจาก 13 วัน เป็น 12 วัน

สายงานที่ 1 A – C – G = 7+9+13 = 29

สายงานที่ 2 A – D – H = 7+11+12 = 30 ---→ สายงานวิกฤต

สายงานที่ 3 B – E – H = 6+8+12 = 26

สายงานที่ 4 B – F – I = 6+10+14 = 30 ---→ สายงานวิกฤต

หลังจากที่ได้ทำการเร่งกิจกรรม H ยังไม่ได้ทำให้โครงการเสร็จลงได้ตามเวลาที่กำหนดไว้ ดังนั้นจึงต้องเร่งกิจกรรมอื่น โดยสายงานวิกฤตในที่นี้มีสองสายงาน คือ A – D – H และ B – F – I โดยเส้นทางที่หนึ่งกิจกรรม D จะมีค่าใช้จ่ายต่ำสุด คือ 125 บาทต่อวัน และเร่งได้เร็วขึ้น 2 วัน ส่วนเส้นทางที่สอง กิจกรรม F จะมีค่าใช้จ่ายต่ำสุด คือ 100 บาทต่อวัน และเร่งได้เร็วขึ้น 3 วัน โดยจะทำการเร่งกิจกรรม D และ F ลง 2 วัน ซึ่งกิจกรรม F นั้นสามารถเร่งได้เร็วขึ้น 3 วัน แต่ถ้าพิจารณาแล้วจะเห็นได้ว่าถึงแม้เร่งกิจกรรม F เป็น 3 วันไม่ได้ทำให้โครงการสามารถเสร็จเร็วขึ้น ซึ่งหากเร่งกิจกรรม F เป็น 3 วัน จะมีผลทำให้เสียค่าใช้จ่ายเพิ่มขึ้นโดยใช่เหตุ

สายงานที่ 1 A – C – G = 7+9+13 = 29 ---→ สายงานวิกฤต

สายงานที่ 2 A – D – H = 7+9+12 = 28

สายงานที่ 3 B – E – H = 6+8+12 = 26

สายงานที่ 4 B – F – I = 6+8+14 = 28

หลังจากที่ได้ทำการเร่งกิจกรรม D และ F แล้ว ยังไม่ได้ทำให้โครงการเสร็จลงได้ตามเวลาที่กำหนดไว้ ดังนั้นจึงต้องเร่งกิจกรรมอื่น โดยสายงานวิกฤตในที่นี้ คือ A – C – G โดยค่าใช้จ่ายของกิจกรรม A มีต่ำสุด คือ วันละ 150 บาท ดังนั้นจึงเลือกกิจกรรม A ด้วยการเร่งเวลาจาก 7 วัน เป็น 6 วัน

สายงานที่ 1 A – C – G = 6+9+13 = 28 ---→ สายงานวิกฤต

สายงานที่ 2 A – D – H = 6+9+12 = 27

สายงานที่ 3 B – E – H = 6+8+12 = 26

สายงานที่ 4 B – F – I = 6+8+14 = 28 ---→ สายงานวิกฤต

จากการลดกิจกรรม A,B,D,F และ H จึงสามารถจัดทำโครงการได้แล้วเสร็จภายใน 28 วันตามแผนที่ได้วางไว้ โดยจะได้สายงานวิกฤตอยู่สองสายงาน และมีจำนวนวันยาวนานที่สุด คือ 28 วัน และค่าใช้จ่ายที่ต้องเพิ่มขึ้นจากการเร่งงานให้เสร็จเร็วขึ้น สามารถแสดงรายละเอียดได้ดังนี้

กิจกรรมที่เร่ง	จำนวนวัน	ค่าใช้จ่ายต่อวัน	รวม(บาท)
A	1	150	150
B	2	75	150
D	2	125	250
F	2	100	200
H	1	100	100
รวมค่าใช้จ่ายที่เพิ่มขึ้นจากการเร่งโครงการ			850

บทที่ 4 การค้นหาและเลือกสรรโครงการ

(Project Identification and Selection)

4.1. การค้นหาโครงการที่ต้องการการพัฒนา

สามารถค้นหาโครงการพัฒนาระบบภายในองค์กรได้จากกลุ่มบุคคลที่อยู่ภายในองค์กรดังนี้

1. ผู้บริหารสูงสุดขององค์กรเอง
2 ผู้จัดการของแผนกต่างๆ ที่มีความสนใจในโครงการที่ต้องการพัฒนา
3. ผู้ใช้ระบบหรือหัวหน้าแผนก ซึ่งเป็นผู้ประสงค์ที่จะพัฒนาระบบให้ดีกว่าเดิม
4. ผู้บริหารอาวุโสของทีพัฒนาระบบสารสนเทศ

สามารถจำแนกลักษณะของโครงการตามมุมมองของกลุ่มบุคคลในองค์กรต่างๆ ดังนี้

บุคคลที่เกี่ยวข้อง	ลักษณะของโครงการ
ผู้บริหารระดับสูง (Top Management)	1. มุ่งเน้นเชิงกลยุทธ์มากที่สุด 2. โครงการมีขนาดใหญ่ที่สุด 3. มีระยะเวลาดำเนินโครงการนานที่สุด
ผู้บริหารระดับล่าง (Steering Committee)	1. มุ่งเน้นการปฏิบัติหน้าที่ร่วมกัน 2. มีการเปลี่ยนแปลงโครงสร้างมากกว่า 3. มีการวิเคราะห์ต้นทุนและกำไรอย่างเป็นแบบแผน 4. โครงการมีความเสี่ยงและมีขนาดใหญ่
ผู้ใช้ระบบ (User Department)	1. ไม่มุ่งเชิงกลยุทธ์ 2. มีการพัฒนาที่รวดเร็ว 3. มีผู้เกี่ยวข้องในการดำเนินโครงการน้อย
ทีมนักพัฒนา (Development Group)	1. คำนึงถึงการนำระบบใหม่ไปประยุกต์ใช้งานร่วมกับระบบเดิมที่มีอยู่แล้ว 2. สามารถพัฒนาได้อย่างรวดเร็ว 3. มักไม่คำนึงถึงต้นทุนและกำไร

การค้นหาโครงการพัฒนาระบบจากกลุ่มบุคคลที่กล่าวไว้ข้างต้น จะทำให้ได้โครงการต่างๆ ที่มีลักษณะและผลการอนุมัติที่แตกต่างกัน ดังนั้นจึงจำเป็นต้องมีการจำแนกโครงการพัฒนาระบบออกเป็นกลุ่ม เพื่อพิจารณาถึงผลตอบแทนที่จะได้รับจากโครงการ และความเหมาะสมกับสภาวะการณ์ปัจจุบันขององค์กร เนื่องจากบางโครงการที่ค้นหามานั้นอาจไม่ตรงตามวัตถุประสงค์ขององค์กรในขณะนั้น

4.2. การจำแนกและจัดกลุ่มโครงการที่ค้นหามา

การจำแนกและจัดกลุ่มของโครงการเป็นผลเนื่องมาจาก แต่ละโครงการที่ค้นหามาอาจไม่สอดคล้องกับวัตถุประสงค์ขององค์กร ในการจำแนกโครงการจะใช้วิธีที่เรียกว่า “การประเมินค่าความเป็นไปได้ของโครงการ” (หมายถึง โครงการที่สามารถสร้างผลตอบแทนสูงสุดให้องค์กร) โดยจะมีการกำหนดกฎเกณฑ์ขึ้นมาเป็นหลักในการประเมิน ซึ่งกฎเกณฑ์ของแต่ละองค์กรอาจแตกต่างกันไป แต่โดยทั่วไปแล้วจะมีหลักเกณฑ์ในการประเมินโครงการดังแสดงในตารางต่อไปนี้

เกณฑ์ในการประเมินเพื่อใช้ในการจัดกลุ่ม	รายละเอียด
1. การวิเคราะห์ Value Chain (Value Chain Analysis)	พิจารณาถึงกิจกรรมในโครงการที่จะสามารถเพิ่มคุณค่าให้แก่สินค้าหรือบริการ รวมทั้งพิจารณาถึงต้นทุนที่จะเกิดขึ้นด้วย
2. สอดคล้องกับกลยุทธ์ขององค์กร	พิจารณาถึงผลของโครงการที่จะสามารถช่วยให้องค์กรบรรลุเป้าหมายตามกลยุทธ์ที่วางไว้ได้
3. ผลตอบแทนที่สามารถเป็นไปได้	พิจารณาโครงการที่จะสามารถเพิ่มผลกำไร เพิ่มลูกค้าและบริการในช่วงระยะเวลาหนึ่งได้
4. แหล่งทรัพยากรที่สามารถนำมาดำเนินการได้	พิจารณาถึงแหล่งทรัพยากรที่แต่ละโครงการต้องการว่าสามารถนำมาดำเนินการได้หรือไม่
5. ขนาดของโครงการและระยะเวลาในการดำเนินโครงการ	พิจารณาถึงระยะเวลาทั้งหมดที่ใช้ในการดำเนินโครงการเมื่อเสร็จสมบูรณ์แล้ว รวมทั้งพิจารณาขนาดของโครงการอันจะส่งผลต่อต้นทุนที่เกิดขึ้น
6. ความยากในด้านเทคนิคและความเสี่ยง	พิจารณาถึงระดับความยากทางด้านเทคนิคเพื่อจะทำให้โครงการประสบผลสำเร็จภายในเวลาที่กำหนดและภายใต้ข้อจำกัดของแหล่งทรัพยากร

Value Chain Analysis คือ การวิเคราะห์กิจกรรมในวงจรการผลิตสินค้าหรือบริการ เพื่อพิจารณาถึงสิ่งที่ควรกระทำเพื่อเพิ่มคุณค่าของสินค้าหรือบริการนั้นในมุมมองของผู้บริโภค โดยคุณค่าของสินค้าหรือบริการนั้นหมายถึง ราคาที่ไม่สูงจนเกินไปหรือการมีบริการที่ดี

เมื่อทำการประเมินและกำหนดรายละเอียดของแต่ละโครงการเสร็จสิ้นแล้ว งานต่อไปคือการเลือกโครงการที่เหมาะสมกับวัตถุประสงค์ขององค์กร ดังรายละเอียดในหัวข้อถัดไป

4.3. การเลือกโครงการที่เหมาะสม

การเลือกโครงการ คือ กระบวนการพิจารณาโครงการทั้งระยะสั้นและระยะยาว แล้วทำการเลือกโครงการที่มีความเป็นไปได้มากที่สุดที่จะทำให้องค์กรนั้นบรรลุสู่เป้าหมายที่ได้กำหนดไว้

เนื่องจากในการดำเนินธุรกิจนั้นมีการเปลี่ยนแปลงเงื่อนไขทางธุรกิจอยู่ตลอดเวลา และส่งผลกระทบต่อโครงการให้มีการเปลี่ยนแปลงไปด้วย ดังนั้นในการค้นหาและเลือกโครงการพัฒนาระบบนั้น ควรให้ความสำคัญและเพิ่มความระมัดระวังเป็นอย่างยิ่งไม่ว่าจะเป็นขั้นตอนใดก็ตาม

ปัจจัยที่ใช้ในการตัดสินใจเลือกโครงการ ได้แก่

- ความจำเป็นขององค์กร ต่อโครงการพัฒนาระบบนั้นๆ ว่ามีความจำเป็นมากน้อยเพียงใด
- รายชื่อของโครงการพัฒนาระบบที่มีความเป็นไปได้
- แหล่งทรัพยากรที่มีอยู่แล้วและสามารถใช้ประโยชน์ได้

- เกณฑ์การประเมิน
- ปัจจัยแวดล้อมที่เกี่ยวข้องกับองค์กรหรือเงื่อนไขทางธุรกิจ

ผลลัพธ์ที่อาจเป็นไปได้ในการตัดสินใจเลือกโครงการ

- ยอมรับโครงการ (Accept Project) หมายถึง การเริ่มต้นจัดทำโครงการและดำเนินการในขั้นตอนการพัฒนาระบบต่อไป
- ปฏิเสธโครงการ (Reject Project) หมายถึง ไม่มีการพิจารณาโครงการนั้นเพื่อการพัฒนาต่อไป หรืออาจเป็นการปฏิเสธในกรณีที่ให้ระงับโครงการนั้น แต่ยังต้องการให้ส่วนงานนั้นเสนอโครงการมาโดยให้หาโครงการใหม่มาเสนออีครั้ง
- ชะลอโครงการ (Delay Project) หมายถึง ให้ชะลอโครงการที่เสนอไว้ก่อน เนื่องจากองค์กรยังไม่พร้อมเมื่อพร้อมแล้วจะนำมาพิจารณาอีกครั้ง
- ให้ผู้ใช้พัฒนาระบบเอง (End-User Development) โดยการให้ผู้ใช้ระบบทำการพัฒนาระบบกันเองภายในองค์กรอาจจะเริ่มจากการส่งผู้ใช้ไปฝึกอบรมการพัฒนาระบบก่อน แล้วจึงนำความรู้ที่ได้กลับมาพัฒนากันเอง
- ให้ทบทวนโครงการ (Proof of Concept) โดยให้ผู้จัดทำโครงการกลับไปเรียบเรียง หรือแก้ไขหลักการและเหตุผลที่จัดทำโครงการนั้นขึ้นมาร แล้วจึงนำมาเสนออีกครั้งในภายหลัง

ในขั้นตอนการค้นหาและการเลือกโครงการ นักวิเคราะห์ระบบควรมีความละเอียดรอบคอบในการพิจารณาเพื่อให้ตรงกับวัตถุประสงค์ขององค์กรและสามารถแก้ปัญหาที่เกิดขึ้นให้ได้มากที่สุด สิ่งที่จะช่วยเพิ่มความถูกต้องและตรงประเด็นในการเลือกโครงการของนักวิเคราะห์ระบบคือ การนำแผนกลยุทธ์ขององค์กร (Corporate Strategic Planning) และแผนงานระบบสารสนเทศ (Information System Planning) มาใช้กำหนดทิศทางในการเลือกโครงการพัฒนาระบบที่ก่อให้เกิดผลประโยชน์สูงสุดแก่องค์กร โดยมีรายละเอียดดังต่อไปนี้

4.3.1. การวางแผนกลยุทธ์ขององค์กร (Corporate Strategic Planning)

ก่อนการพิจารณาเลือกโครงการ ควรมีความเข้าใจในสถานะปัจจุบันและเป้าหมายที่แท้จริงขององค์กร นอกจากนั้นควรพิจารณาถึงวิธีการที่จะทำให้โครงการนั้นสามารถบรรลุถึงเป้าหมายได้ นั่นคือ การนำแผนกลยุทธ์ขององค์กร (Corporate Strategic Planning) มาใช้ในกำหนดทิศทางการเลือกโครงการ การวางแผนกลยุทธ์ขององค์กรมีขั้นตอนดังนี้

1. เข้าใจในสถานะปัจจุบันขององค์กร (Current Enterprise) เป็นการสร้างความเข้าใจในสถานะปัจจุบันขององค์กรในสถาวะการณ์เศรษฐกิจในขณะนั้น เพื่อหาแนวทางในการพัฒนาองค์กรในอนาคต เสมือนเป็นการกำหนดถึงภารกิจ (Mission) ขององค์กรนั่นเอง
2. พิจารณาถึงสถานะในอนาคตขององค์กร (Future Enterprise) เป็นการกำหนดจุดยืนที่แท้จริงขององค์กรในอนาคตเสมือนเป็นการกำหนดวัตถุประสงค์ (Objective) ขององค์กรนั่นเอง

3. วางแผนกลยุทธ์ (Strategic Plan) เมื่อมีความเข้าใจต่อสถานะปัจจุบันและกำหนดสิ่งที่ต้องการในอนาคตแล้ว ขั้นตอนนี้เป็นการดำเนินการเพื่อให้บรรลุวัตถุประสงค์ขององค์กรที่วางไว้โดยใช้กลยุทธ์ในการแข่งขัน (Competitive Strategy)

การเลือกใช้กลยุทธ์ในการแข่งขันทางธุรกิจ ย่อมต้องนำสารสนเทศที่แตกต่างกันมาประกอบการตัดสินใจในการเลือกใช้กลยุทธ์

ดังนั้น การเข้าใจถึงภารกิจหลัก วัตถุประสงค์ และกลยุทธ์ในการดำเนินธุรกิจที่ชัดเจน สามารถนำมาเป็นปัจจัยเพื่อกำหนดทิศทางในการค้นหาและการเลือกโครงการที่เหมาะสมและสอดคล้องกับวัตถุประสงค์เหล่านั้น เพื่อให้เกิดผลประโยชน์สูงสุดแก่องค์กร กระบวนการวางแผนจะมีส่วนช่วยในการกำหนดและเลือกโครงการที่จะทำการพัฒนาระบบนอกจากการใช้กระบวนการวางแผนกลยุทธ์ขององค์กร (Corporate Strategic Planning) เพื่อกำหนดและเลือกโครงการที่จะพัฒนาดังที่กล่าวมาข้างต้นแล้ว ยังสามารถใช้อีกกระบวนการหนึ่งได้คือ การวางแผนระบบสารสนเทศ

4.3.2. การวางแผนระบบสารสนเทศ (Information System Planning : ISP)

การวางแผนระบบสารสนเทศ หมายถึง การประเมินความต้องการสารสนเทศขององค์กร และระบุถึงระบบสารสนเทศฐานข้อมูลและเทคโนโลยี เพื่อให้ได้มาซึ่งสารสนเทศที่พึงพอใจมากที่สุด

ผู้ที่มีหน้าที่ในการวางแผนงานระบบสารสนเทศโดยทั่วไปแล้วจะเป็นผู้บริหารระบบสารสนเทศอาวุโส โดยเริ่มจากการกำหนดความต้องการสารสนเทศขององค์กรทั้งในปัจจุบันและในอนาคต หลังจากนั้นทำการพัฒนากลยุทธ์และวางแผนการเพื่อรองรับการเปลี่ยนแปลงของระบบสารสนเทศและเทคโนโลยี อันอาจจะเกิดขึ้นในอนาคต โดยมุ่งเน้นไปที่ระบบสารสนเทศและเทคโนโลยีในเชิงวิธีการที่จะช่วยให้ธุรกิจนั้นบรรลุสู่วัตถุประสงค์ (Objectives) ที่ได้วางไว้ในแผนกลยุทธ์ขององค์กร (Corporate Strategic Planning) เช่นเดียวกับการวางแผนกลยุทธ์ขององค์กร คือมีการแบ่งกระบวนการการวางแผน เป็น 3 ขั้นตอน มีรายละเอียดดังนี้

1. ประเมินความต้องการสารสนเทศในปัจจุบันขององค์กร

   โดยพิจารณาความต้องการที่มีส่วนเกี่ยวข้องกับระบบงานบัญชี ทรัพยากรบุคคล ข้อมูล กระบวนการทำงานและเทคโนโลยี เพื่อนำไปสู่การพัฒนาระบบงานที่จะทำให้ได้มาซึ่งสารสนเทศตามความต้องการเหล่านั้นได้อย่างมีประสิทธิภาพ

   ขั้นตอนการประเมินความต้องการสารสนเทศในปัจจุบัน เริ่มจากการแต่งตั้งทีมงานเพื่อทำการเก็บรวบรวมข้อมูลของความต้องการสารสนเทศ โดยการสัมภาษณ์ผู้บริหารและลูกค้า ซึ่งเป็นความต้องการสารสนเทศในด้านของคู่แข่งขันทางการค้าสารสนเทศด้านการตลาด ผลิตภัณฑ์และการเงิน การประเมินความต้องการในสารสนเทศที่รวบรวมมานั้นจะต้องครอบคลุมทั้งองค์กรและทุกส่วนที่เกี่ยวข้องกับองค์กร ไม่ว่าจะมีสาขาใดก็ตาม รวมไปถึงระบบสารสนเทศที่ใช้งานอยู่ในปัจจุบัน

   ผลที่ได้จากขั้นตอนนี้คือ

   1. รายการของการทำงานทั้งหมดในองค์กรและหน้าที่ (Functions)
   2. ตารางเมตริกซ์แสดงความสัมพันธ์ของส่วนต่างๆ ที่เกี่ยวข้องกับองค์กร

2.กำหนดจุดมุ่งหมาย แนวโน้มและเงื่อนไข

หลังจากที่ได้ทราบถึงความต้องการสารสนเทศในปัจจุบันแล้ว ขั้นตอนต่อไปคือการกำหนดจุดมุ่งหมายขององค์กร เช่นต้องการที่จะขยายสาขาใหม่ หรือทำการผลิตสินค้าชนิดใหม่ ส่งผลให้ข้อมูลในส่วนที่เป็นสถานที่ หน่วยงาน หน้าที่การทำงาน กระบวนการทำงาน ข้อมูล และ สารสนเทศต่างๆ ย่อมมีการเปลี่ยนแปลงไป ดังนั้นตารางเมตริกซ์ต่างๆ ที่ได้จัดทำในขั้นตอนแรกนั้นต้องมีการเปลี่ยนแปลงไปด้วย ทำให้ทราบถึงความต้องการสารสนเทศในอนาคต

3. การพัฒนากลยุทธ์และการวางแผน

เมื่อสามารถประเมินความต้องการสารสนเทศในปัจจุบันขององค์กร แล้วนำไปกำหนดจุดมุ่งหมายและแนวโน้มและเงื่อนไขในอนาคตได้แล้ว ในขั้นตอนนี้องค์กรจะต้องพัฒนากลยุทธ์ โดยการวางแผนการที่จะพัฒนาระบบงานต่างๆ เพื่อทำให้บรรลุถึงจุดมุ่งหมายนั้น

ที่กล่างมาข้างต้นเป็นแผนกลยุทธ์ที่ช่วยให้การค้นหา และการเลือกโครงการพัฒนาระบบขององค์กรมีประสิทธิภาพมากขึ้นเนื่องจากมีการกำหนดวัตถุประสงค์ และมีแผนกลยุทธ์ต่างๆ เป็นหลักเกณฑ์ในการพิจารณาตัดสินใจเลือกโครงการ กล่าวคือ ในการเลือกโครงการควรจะมีการคำนึงถึงรายละเอียดของทั้งแผนงานทางกลยุทธ์ขององค์กรและแผนงานระบบสารสนเทศ เพื่อเปรียบเทียบกับรายละเอียดโครงการพัฒนาระบบต่างๆ ว่าโครงการใดมีความจำเป็นมากที่สุดภายใต้เงื่อนไขที่สอดคล้องกับวัตถุประสงค์และเป้าหมายขององค์กรนั่นเอง

บทที่ 5 การเริ่มต้นและวางแผนโครงการ

(Project Initiating and Planning)

5.1. การเริ่มต้นโครงการ (Project Initiation)

การเริ่มต้นจัดทำโครงการเป็นกิจกรรมแรกที่ต้องทำ โดยในระหว่างการเริ่มต้นหรือการจัดทำโครงการนี้ ผู้จัดการหรือผู้บริหารโครงการจะต้องกำหนดทิศทางของโครงการ โดยขึ้นอยู่กับขนาด ขอบเขต และความซับซ้อนของโครงการนั้น ในบางกรณีกิจกรรมนี้อาจจะไม่มีความจำเป็นสำหรับบริษัทเล็กๆ แต่หากเป็นองค์กรใหญ่ๆ ที่มีขั้นตอนในการดำเนินการต่างๆ มากมาย กิจกรรมนี้ย่อมมีความสำคัญต่อการดำเนินงานในขั้นตอนต่อไป โดยที่กิจกรรมต่างๆ ในการเริ่มต้นจัดทำโครงการมีดังนี้

1. จัดตั้งทีมงานจัดทำโครงการ เป็นกิจกรรมจัดตั้งทีมงานเพื่อดำเนินการโครงการพัฒนาระบบ และกำหนดตำแหน่งหน้าที่อย่างชัดเจน ให้กับสมาชิกของทีม
2. จัดทำแผนการในการเริ่มต้นโครงการ กำหนดกิจกรรมที่จะต้องทำ และระยะเวลาที่ใช้ในแต่ละกิจกรรมที่เกิดขึ้นในระหว่างการเริ่มต้นจัดทำโครงการนั้น
3. จัดทำกระบวนการบริหารโครงการ เป็นกิจกรรมกำหนดกระบวนการในการบริหารโครงการ และมาตรฐานที่ใช้ในการควบคุมการดำเนินการของโครงการนั้น โดยมาตรฐานนั้นจะขึ้นอยู่กับผู้บริหารของแต่ละองค์กรที่ได้รับการแต่งตั้งให้เป็นผู้จัดการโครงการ
4. จัดทำสมุดงานของโครงการ (Project Workbook) เป็นกิจกรรมรวบรวมข้อมูลที่ได้จากขั้นตอนแรกของการเริ่มต้นโครงการ ซึ่งสมุดงานของโครงการ (Project Workbook) หมายถึง แหล่งเก็บรวบรวมข้อมูลทั้งหมดที่เกี่ยวกับการจัดทำโครงการ เช่น ภาพรวมของโครงการ ความรับผิดชอบของโครงการ ภาพรวมของระบบ ข้อมูลที่เข้าสู่ระบบ ข้อมูลที่ออกจากระบบ แบบจำลองต่างๆ พจนานุกรมข้อมูล สิ่งที่ได้เมื่อข้อมูลผ่านขั้นตอนการประมวลผล มาตรฐานในการตรวจสอบการทำงานตำแหน่งหน้าที่รับผิดชอบของทีมงาน ขอบเขตของโครงการ รายละเอียดการบริหารโครงการ เป็นต้น ซึ่งหากองค์กรนั้นมีการใช้ CASE Tools ในการพัฒนาระบบก็จะจัดเก็บข้อมูลเหล่านี้ไว้ใน Repository ทั้งนี้เพื่อประโยชน์ในการเรียนใช้ข้อมูลเหล่านั้น

หมายเหตุ CASE Tools คือซอฟต์แวร์ที่เป็นเครื่องสนับสนุนการพัฒนาระบบ ช่วยสร้างแผนภาพ รายงานและแบบฟอร์ม สร้าง Source Code ในระหว่างการวิเคราะห์และออกแบบระบบได้โดยอัตโนมัติ ทั้งยังตรวจสอบความถูกต้องของแผนภาพได้อีกด้วย

Repository คือ ฐานข้อมูลของ CASE ที่ใช้จัดเก็บรายละเอียด ข้อมูลของระบบ แผนภาพ รายงานและแบบฟอร์ม ของระบบที่กำลังพัฒนา

รายละเอียดในการดำเนินโครงการ เช่น

1. ภาพรวมของโครงการ
2. การเริ่มต้นโครงการ
3. ขอบเขตของโครงการ

4. การบริหารโครงการ
5. แบบจำลองขั้นตอนการทำงาน (DFD) คำอธิบาย และพจนานุกรมข้อมูล
6. แบบจำลองข้อมูลและพจนานุกรมข้อมูล
7. หน้าที่รับผิดชอบของทีมงาน
8. รายงานสถานะการทำงาน
9. ตารางแสดงระยะเวลาดำเนินโครงการ

ทั้งนี้การกำหนดรายละเอียดต่างๆ ในสมุดงานของโครงการนั้นไม่ได้มีกฎเกณฑ์กำหนดไว้แน่นอน ขึ้นอยู่กับนโยบายขององค์กร

5.2. การเสนอแนวทางเลือกในการนำระบบใหม่มาใช้งาน

หลังจากที่ได้จัดตั้งทีมงานเพื่อรับผิดชอบโครงการพัฒนาระบบแล้ว ทีมงานดังกล่าวจะต้องศึกษาข้อมูลเพิ่มเติม เพื่อนำมาใช้ประโยชน์ในกิจกรรมนำเสนอทางเลือกในการนำระบบใหม่มาใช้งาน โดยมีวัตถุประสงค์เพื่อศึกษาและวิเคราะห์ถึงแนวทางเลือกที่เหมาะสม ไม่ว่าจะเป็นแนวทางเลือกประยุกต์ใช้เทคโนโลยีการสื่อสาร ฮาร์ดแวร์ ซอฟต์แวร์ต่าง ๆ เป็นต้น รวมทั้งประมาณการต้นทุนของแต่ละแนวทางเลือก เพื่อใช้ประกอบการพิจารณาสนับสนุนการตัดสินใจของผู้บริหารหรือเจ้าของระบบในการนำระบบใหม่มาใช้ในองค์กรได้อย่างมีประสิทธิผลสูงสุด

แนวทางที่ใช้ในการตัดสินใจเลือกนั้น นอกจากจะต้องสอดคล้องและเหมาะสมกับสถานการณ์ปัจจุบันขององค์กรมากที่สุดไม่ว่าจะเป็นสถานะทางการเงินและความพร้อมในด้านต่างๆ แล้ว ยังจะต้องสัมฤทธิ์ผลถึงวัตถุประสงค์ที่วางไว้ทุกประการสำหรับวิธีการเสนอแนวทางเลือกต่างๆ เหล่านี้ ล้วนเป็นหน้าที่ของนักวิเคราะห์ระบบ ที่จะต้องค้นหาและสร้างแนวทางเลือกพร้อมทั้งข้อเสนอแนะในแต่ละแนวทางเลือกเพื่อใช้เป็นข้อมูลสนับสนุนการตัดสินใจของผู้บริหารต่อไป

วิธีการเสนอแนวทางเลือกในการนำระบบใหม่มาใช้งานจะประกอบด้วย 2 กิจกรรม คือ

1. ค้นหาและสร้างแนวทางเลือกในการนำระบบใหม่มาใช้งาน
2. เลือกทางเลือกที่ดีที่สุด

5.2.1. ค้นหาและสร้างแนวทางเลือกในการนำระบบใหม่มาใช้งาน

การค้นหาและสร้างแนวทางเลือกในการนำระบบใหม่มาใช้งาน เป็นความรับผิดชอบโดยตรงของนักวิเคราะห์ระบบในองค์กรแต่ละแห่งจะมีนักวิเคราะห์ระบบที่มีประสบการณ์ในการทำงานแตกต่างกัน หากเป็นผู้ที่มีประสบการณ์ในการทำงานสูงจะสามารถเสนอแนวทางเลือกจะสามารถได้ดีกว่าผู้ที่มีประสบการณ์น้อย เนื่องจากปัญหาของระบบงานใหม่นี้อาจจะเป็นปัญหาที่นักวิเคราะห์ระบบเคยผ่านงานมาแล้ว ทำให้เข้าใจและแก้ปัญหาได้ตรงจุดกว่า

สำหรับการสร้างแนวทางเลือกนั้นจะเกี่ยวข้องกับการพิจารณาถึงข้อกำหนดคุณสมบัติการทำงานของระบบ (System's Functionality) ไม่ว่าจะเป็นระบบฮาร์ดแวร์ (Hardware System) และระบบซอฟต์แวร์ (Software System) ต่างๆ ยังรวมถึงต้นทุน (Costs) ผลตอบแทน (Benefits) และความเสี่ยง (Risks) จากเรื่อง

การค้นหาและเลือกสรรโครงการพัฒนาระบบ นำมาเป็นข้อกำหนดหรือเงื่อนไขของแต่ละแนวทางเลือก โดยกำหนดแนวทางเลือกไว้อย่างน้อยที่สุด 3 แนวทางเลือกที่แตกต่างกัน ซึ่งสามารถจำแนกเป็น 3 ระดับได้แก่

1. ทางเลือกที่ 1 คือทางเลือกระดับล่าง (Low-end Solutions) หมายถึง แนวทางเลือกที่มีความแตกต่างจากระบบเดิมไม่มากนัก เนื่องจากเป็นแนวทางเลือกที่ต้องคำนึงถึงต้นทุนเป็นหลักหรือกล่าวอีกนัยหนึ่งคือเน้นความประหยัด ดังนั้นการแก้ปัญหาจึงอาจจะเป็นเพียงแค่การลดการทำงานที่ซ้ำซ้อน หรือการทำงานในระบบเดิมที่ทำให้มีประสิทธิภาพมากยิ่งขึ้นหรือหากมีการเลือกเทคโนโลยีสมัยใหม่เข้ามาใช้อาจจะเลือกใช้กับงานบางส่วนของระบบเท่านั้น
2. ทางเลือกที่ 2 คือทางเลือกระดับสูง (High-end Solution) หมายถึง แนวทางเลือกที่นำเสนอการใช้เทคโนโลยีที่ล้ำยุค ซึ่งสามารถช่วยในการทำงานของระบบเดิมดีขึ้น มีลักษณะการทำงานที่อาจเป็นที่สนใจของผู้ใช้ระบบมากที่สุดและสามารถรองรับความต้องการของระบบใหม่ที่อาจจะมีเพิ่มมากขึ้นได้ในอนาคต แต่ทางเลือกระดับนี้จะมีค่าใช้จ่ายสูงที่ค่อนข้างสูง
3. ทางเลือกที่ 3 คือทางเลือกระดับกลาง (Midrange Solution) หมายถึง แนวทางเลือกที่มีการนำเสนอการใช้เทคโนโลยีที่เป็นกลางระหว่าง Low-end กับ High-end Solution คือมีการผสมผสานระหว่างการคำนึงถึงความประหยัดและการเพิ่มประสิทธิภาพในการทำงานด้วยเทคโนโลยีระดับสูง ด้วยลักษณะของ Midrange Solution นี้เอง ที่มีความเป็นไปได้มากที่สุดในการถูกเลือกเป็นแนวทางเลือกที่ดีที่สุด

5.2.2. เลือกทางเลือกที่ดีที่สุด

หลังจากที่ได้เสนอแนวทางเลือกในการนำระบบมาใช้งานต่อผู้บริหารแล้ว ขั้นตอนต่อไปคือการพิจารณาตัดสินใจเลือกแนวทางที่เหมาะสมที่สุด วิธีการหรือกฎเกณฑ์ในการเลือกนั้นขึ้นอยู่กับนโยบายของแต่ละองค์กรเอง ทั้งนี้ในการเลือกทางเลือกควรคำนึงถึงความต้องการของผู้ใช้ระบบ และประสิทธิภาพของงานที่จะได้รับกลับมามากที่สุด

5.2.3. ปัจจัยเพื่อใช้สร้างแนวทางเลือกในการนำระบบใหม่มาใช้งาน

การสร้างแนวทางเลือกในการนำระบบใหม่มาใช้งาน นักวิเคราะห์ระบบสามารถพิจารณาได้จากปัจจัยหลายประการ ได้แก่

1. แหล่งทรัพยากรภายนอกองค์กร (Outsourcing)
2. แหล่งซอฟต์แวร์ (Sources of Software)
3. การพัฒนาระบบด้วยองค์กรเอง (In-House Development)
4. การตรวจสอบข้อมูลของซอฟต์แวร์ที่ต้องการ
5. ฮาร์ดแวร์และซอฟต์แวร์ระบบ
6. การติดตั้งระบบ (Implementation)

แหล่งทรัพยากรภายนอกองค์กร (Outsourcing)

Outsourcing หมายถึง บริษัทหรือองค์กรอื่นจากภายนอกที่ถูกว่าจ้างให้มาดำเนินการพัฒนาระบบสารสนเทศให้กับองค์กร ตัวอย่างของการเลือกพัฒนาระบบสารสนเทศด้วย Outsourcing ได้แก่

ตัวอย่างที่ 1 องค์กรว่าจ้างให้บริษัทรับพัฒนาระบบจากภายนอก ทำการพัฒนาโปรแกรมสำเร็จรูปเพื่อจัดทำบัญชีเงินเดือนให้กับองค์กร โดยที่องค์กรเป็นผู้ส่งข้อมูลของลูกจ้างให้แก่บริษัทรับพัฒนาระบบบัญชีเงินเดือนเพื่อทำการประมวลผลจากบริษัทนั้น

ตัวอย่างที่ 2 องค์กรว่าจ้างให้บริษัทรับพัฒนาระบบจากภายนอก ทำการพัฒนาโปรแกรมสำเร็จรูปเพื่อจัดทำบัญชีเงินเดือนให้กับองค์กร แต่นำโปรแกรมนั้นมาใช้ในองค์กรเอง

ตัวอย่างที่ 3 องค์กรว่าจ้างให้บริษัทรับพัฒนาระบบจากภายนอก ทำการพัฒนาระบบสารสนเทศให้กับองค์กรเพียงบางส่วนหรือบางระบบเท่านั้น

การว่าจ้างให้บริษัทรับพัฒนาระบบจากภายนอกองค์กรมาทำการพัฒนาระบบสารสนเทศให้นั้น ขึ้นอยู่กับกลยุทธ์ของแต่ละองค์กรเองในการพิจารณาเลือกวิธีการในการว่าจ้าง ในบางองค์กรอาจจะว่าจ้างให้พัฒนาเพียงบางส่วนของระบบเท่านั้นเพื่อประหยัดค่าใช้จ่าย เนื่องจากองค์กรไม่ทีทรัพยากรที่มีความสามารถในการพัฒนาระบบในส่วนนั้น หากจะจ้างบุคคลากรเพิ่มเติมจะทำให้เกิดต้นทุนสูงกว่าว่าจ้างบริษัทภายนอก

จากปัจจัย Outsourcing หากนักวิเคราะห์ระบบพิจารณา Outsourcing เป็นหนึ่งในแนวทางเลือก 3 ระดับ นักวิเคราะห์ระบบควรทำความเข้าใจกับความต้องการของผู้ใช้งานในระบบใหม่เป็นอย่างดี พร้อมกับจัดทำข้อกำหนดคุณสมบัติทางเทคนิคและความต้องการของระบบใหม่ให้ถูกต้อง ครบถ้วน และสมบูรณ์ ก่อนที่จะส่งมอบให้บริษัทพัฒนาระบบทำการพัฒนาระบบใหม่ต่อไป

แหล่งซอฟต์แวร์

นอกจากบริษัทภายนอกเองค์กรที่รับพัฒนาระบบแล้ว ปัจจัยที่ประกอบการพิจารณาของนักวิเคราะห์ระบบยังมีแหล่งที่รับพัฒนาระบบสารสนเทศและพัฒนาซอฟต์แวร์หรือโปรแกรมสำเร็จรูปเฉพาะด้านเพื่อการค้า ซึ่งสามารถจำแนกได้ 4 กลุ่ม ดังนี้

ผู้ผลิตฮาร์ดแวร์ (Hardware Manufacturers)

ผู้ผลิตชุดซอฟต์แวร์สำเร็จรูป (Packaged Software Producers)

ผู้ผลิตซอฟต์แวร์ตามสั่ง (custom Software Producers)

ซอฟต์แวร์ระดับองค์กร (Enterprise Solution Software)

ผู้ผลิตฮาร์ดแวร์ (Hardware Manufacturers)

จัดเป็นกลุ่มผู้ผลิตฮาร์ดแวร์และดำเนินธุรกิจผลิตซอฟต์แวร์ควบคู่ไปด้วย เช่น บริษัท IBM ซึ่งเป็นผู้นำทางด้านการพัฒนาซอฟต์แวร์ในปัจจุบัน อย่างไรก็ตามกลุ่มผู้ผลิตเหล่านี้จะมีซอฟต์แวร์หลังที่ทำกำไรให้กับตน เช่น กำไรส่วนใหญ่ที่ได้อาจมาจากซอฟต์แวร์ระบบปฏิบัติการ (Operating System Software) ในขณะที่ซอฟต์แวร์ประยุกต์อื่นๆ ทำกำไรให้ในส่วนน้อยก็ได้ เนื่องจากมีคู่แข่งขันจำนวนมากนั่นเอง

ผู้ผลิตชุดซอฟต์แวร์สำเร็จรูป (Packaged Software Producers)

อัตราการเจริญเติบโตของอุตสากรรมการผลิตซอฟต์แวร์เริ่มสูงขึ้นเป็นอย่างมากในช่วงกลางปี ค.ศ.1960 จากอัตราการแข่งขันที่เพิ่มขึ้นทำให้ธุรกิจด้านนี้เริ่มมีทิศทางเปลี่ยนไป จากการผลิตเพียงซอฟต์แวร์

เฉพาะด้านกลายมาเป็น ชุดซอฟต์แวร์สำเร็จรูป(Prepackaged or Off-the-Shelf) ซึ่งประกอบไปด้วยซอฟต์แวร์หลายโปรแกรมในหนึ่งชุดสำหรับงานที่มีความสัมพันธ์กันในการดำเนินธุรกิจหรือในสำนักงาน ยกตัวอย่างเช่น Microsoft Project และ Quicken ซึ่งเป็นชุดซอฟต์แวร์ที่ได้รับความนิยมสำหรับงานทางด้านธุรกิจการเงิน การวางแผนการดำเนินงาน

อุตสาหกรรมการผลิตซอฟต์แวร์เริ่มมีการขยายส่วนแบ่งทางการตลาดสู่กลุ่มเป้าหมายหลายกลุ่ม โดยการผลิตซอฟต์แวร์ที่สนับสนุนการทำงานทั่วไปในสำงาน ชุดซอฟต์แวร์สำหรับงานบัญชี เช่น บัญชีแยกประเภท เป็นต้น กลุ่มเป้าหมายต่อไปคือผลิตซอฟต์แวร์เฉพาะงาน หรือซอฟต์แวร์เพื่อสนับสนุนการดำเนินธุรกิจที่มีลักษณะเฉพาะ เช่น ธุรกิจสถานรับเลี้ยงเด็ก ธุรกิจการบิน เป็นต้น การขยายกลุ่มเป้าหมายและเพิ่มส่วนแบ่งในตลาดการค้าในโลกของคอมพิวเตอร์ กลุ่มผู้ผลิตเหล่านี้ได้เพิ่มความสามารถของซอฟต์แวร์ด้วยการพัฒนาซอฟต์แวร์ที่สามารถทำงานระบบปฏิบัติการใดๆ ได้ ไม่ว่าจะเป็นระบบปฏิบัติการวินโดวส์ (Windows) แมคอินทอช (Macintosh) หรือแม้กระทั่งยูนิกส์ (UNIX) เป็นต้น ก่อนที่จะมีการผลิตซอฟต์แวร์ใดๆ เพื่อวางขาย ผู้ผลิตจะมีการสอบถามความต้องการจากกลุ่มผู้ใช้งานเพื่อนำแนวคิดเหล่านั้นมาทำการพัฒนาเป็นซอฟต์แวร์ หลังจากนั้นจะมีการทดสอบโปรแกรมที่พัฒนาขึ้นเพื่อค้นหาข้อผิดพลาดและแก้ไขให้สมบูรณ์ในระดับหนึ่ง หลังจากวางขายจะพบว่าผู้ผลิตจะได้รับทราบข้อมูลจากการใช้ซอฟต์แวร์ชุดนั้นว่าพบข้อผิดพลาดใดเพิ่มเติมบ้าง จึงได้นำมาปรับปรุงแก้ไขเป็น Version ที่มีประสิทธิภาพมากกว่าเดิม

ปัญหาในการพิจารณาเพื่อเลือกชุดซอฟต์แวร์สำเร็จรูป คือ ชุดซอฟต์แวร์สำเร็จรูปเหล่านี้ในบางครั้งเป็นซอฟต์แวร์ที่ไม่สามารถปรับปรุง เปลี่ยนแปลง หรือทำการแก้ไขใดๆ เพื่อให้สอดคล้องกับความต้องการในการทำงานของผู้ใช้ระบบได้เลยแต่ผู้ผลิตบางบริษัททำการผลิต ชุดซอฟต์แวร์สำเร็จรูป ที่สามารถทำการแก้ไขได้โดยที่ลูกค้าจะต้องแจ้งไปยังผู้ผลิตเพื่อทำการแก้ไข หรือผู้ผลิตอนุญาตให้ลูกค้าสามารถแก้ไขเองได้ด้วยคำแนะนำต่างๆ ดังนั้นในการเลือกปัจจัยข้อนี้นักวิเคราะห์ระบบจะต้องพิจารณาถึงความสามารถในการแก้ไขซอฟต์แวร์ว่า โปรแกรมเมอร์ขององค์กรเองต้องทำการแก้ไขมากน้อยเพียงใด เพื่อให้เหมาะกับการทำงานของระบบใหม่มากที่สุด

ผู้ผลิตซอฟต์แวร์ตามสั่ง (Custom Software Producers)

หากองค์กรมีความจำเป็นต้องพัฒนาระบบสารสนเทศขึ้นมาเพื่อเพิ่มประสิทธิภาพในการทำงาน แต่มีข้อจำกัดอยู่ว่าทางองค์กรไม่มีผู้เชี่ยวชาญหรือไม่มีทีมนักพัฒนาระบบสารสนเทศเอง หรือเมื่อพิจารณาชุดซอฟต์แวร์สำเร็จรูป แล้วยังไม่สามารถแก้ปัญหานี้ได้ ทางเลือกหนึ่งขององค์กรคือ ว่าจ้างบริษัทผู้ผลิตซอฟต์แวร์ตามสั่ง (Custom Software Producers) โดยกลุ่มผู้ผลิตซอฟต์แวร์ตามสั่งเหล่านี้จะมีทีมนักพัฒนาระบบของตัวเองที่มีความเชี่ยวชาญทางด้านต่างๆ เพื่อรองรับองค์กรลูกค้าที่มีคำสั่งให้พัฒนาระบบเพื่อใช้ภายในองค์กร กลุ่มผู้ผลิตเหล่านี้จะทำการจัดสรรนักพัฒนาระบบที่เหมาะกับงานขององค์กรนั้นๆ ในการพัฒนาระบบ

ชุดซอฟต์แวร์เพื่อแก้ปัญหาระดับองค์กร (Enterprise Solution Software)

แหล่งซอฟต์แวร์สุดท้ายที่จะกล่าวถึง คือ ชุดซอฟต์แวร์เพื่อแก้ปัญหาระดับองค์กร (Enterprise Solution Software) หรือเรียกอีกอย่างหนึ่งว่า ERP (Enterprise Resource Planning System) ซึ่งเป็นที่รู้จักกันดีในปัจจุบัน

ERP (Enterprise Resource Planning System) เป็นระบบการวางแผนการใช้ทรัพยากรสำหรับองค์กร ซึ่งพัฒนาขึ้นเพื่อสร้างประสิทธิภาพการทำงานและการใช้ทรัพยากร เช่น ข้อมูล เวลา บุคคล และวัตถุดิบ เป็นต้น แนวคิดหลัก คือการสร้างระบบบริหารทรัพยากรทางธุรกิจแบบรวมส่วนงานทั้งหมดขององค์กร ทั้งการใช้ทรัพยากรและการใช้ข้อมูลร่วมกัน

หากองค์กรใดที่สนใจจะเลือกซอฟต์แวร์เพื่อแก้ปัญหาระดับองค์กร เพียงแค่ทราบความต้องการขององค์กรเองว่าต้องการซอฟต์แวร์ของการทำงานส่วนใดเท่านั้น หรือหากต้องการซอฟต์แวร์ของการทำงานหลายส่วนแล้วองค์กรนำมาประกอบการใช้งานให้เป็นระบบเองก็สามารถทำได้ การนำซอฟต์แวร์ของงานส่วนต่างๆ มาใช้งานร่วมกันได้นี้เรียกได้ว่าเป็นข้อดีของ Enterprise Solution Software ที่น่าสนใจสำหรับองค์กร เช่น การนำซอฟต์แวร์เพื่อการป้อนข้อมูลสั่งซื้อ ทำงานร่วมกับซอฟต์แวร์เพื่อการผลิต และซอฟต์แวร์เพื่อการออกใบเสร็จ เป็นต้น แต่หากมองในแง่ของการแก้ไขซอฟต์แวร์ของแต่ละส่วนการทำงานเพื่อให้สามารถทำงานร่วมกันได้นั้น ค่อนข้างมีความซับซ้อนเมื่อพบกับระบบที่มีความซับซ้อนอยู่ในตัว ดังนั้นนักวิเคราะห์ระบบควรวิเคราะห์ถึงความสามารถในการแก้ไขซอฟต์แวร์ขององค์กรว่ามีมากพอหรือไม่เพียงใด

จะเห็นว่าแหล่งที่มาของซอฟต์แวร์ในปัจจุบันนั้นมีมากมายหลายแห่ง หากองค์กรตัดสินใจที่จะเลือกซื้อซอฟต์แวร์เพื่อนำมาใช้ในระบบใหม่ ไม่ว่าจะเป็นการซื้อทั้งระบบหรือเพียงบางส่วนก็ตาม การพิจารณาเลือกซื้อนั้นจะต้องมีข้อกำหนดในการเลือกซื้อ เพื่อเป็นการเปรียบเทียบต้นทุนหรือค่าใช้จ่าย และผลตอบแทนที่จะได้รับระหว่างการซื้อชุดซอฟต์แวร์จากภายนอกองค์กร และค่าใช้จ่ายที่เกิดขึ้นในแต่ละขั้นตอนของการพัฒนาหากทำการพัฒนาระบบด้วยบุคลากรขององค์กรเอง โดยมีหลักเกณฑ์ในการพิจารณาชุดซอฟต์แวร์ ดังนี้

1. ต้นทุน (Cost)

   เมื่อต้องการเลือกระหว่างการซื้อชุดซอฟต์แวร์จากภายนอกองค์กรกับการพัฒนาเองภายในองค์กร สิ่งสำคัญในการพิจารณาคือต้นทุน นักวิเคราะห์ระบบต้องเปรียบเทียบต้นทุนที่เกิดขึ้นจากทั้งสองทางเลือกนี้ โดยต้องแสดงรายละเอียดของต้นทุนในการซื้อชุดซอฟต์แวร์ การบำรุงรักษาและการบริการหลังการขายของผู้ขาย หรือแม้กระทั่งค่าลิขสิทธิ์ให้จัดรวมในการพิจารณาเรื่องต้นทุนด้วย อย่างไรก็ตามการพิจารณาเลือกระหว่างการซื้อชุดซอฟต์แวร์กับการพัฒนาระบบเองภายในองค์กรควรมีการเปรียบเทียบความเป็นไปได้ด้านเศรษฐศาสตร์ (Economic Feasibility) จะช่วยให้สามารถพิจารณาได้ง่ายยิ่งขึ้น

2. หน้าที่การทำงาน (Functionality)

   นอกจากการเปรียบเทียบเรื่องต้นทุนแล้ว นักวิเคราะห์ระบบหรือองค์กรควรมีการเปรียบเทียบเรื่องของการทำหน้าที่หรือส่วนการทำงานธุรกิจของระบบที่ต้องการนำซอฟต์แวร์นั้นเข้ามาใช้งานเพื่อเพิ่มประสิทธิภาพในการทำงาน การพิจารณาเพื่อเปรียบเทียบว่าซอฟต์แวร์ที่จะเลือกซื้อนั้นสามารถตอบสนองความต้องการ

ในการทำงานของผู้ใช้ระบบนั้นได้มากน้อยเพียงใด เทียบเท่ากับซอฟต์แวร์ขององค์กรที่พัฒนาขึ้นเองได้หรือไม่

3. การบริการหลังการขายชุดซอฟต์แวร์ของผู้ขาย (Vendor Support)
   พิจารณาการเลือกซื้อชุดซอฟต์แวร์จากผู้ขายหลายแห่ง ว่ามีการบริการหลังการขายดีมากน้อยเพียงใด เช่น มีการติดตั้งซอฟต์แวร์ การอบรมพนักงาน หรือหากโปรแกรมมีปัญหาสามารถแก้ปัญหาได้ทันท่วงทีหรือไม่
4. ความยืดหยุ่น (Flexibility)
   ซอฟต์แวร์ที่จะเลือกมานั้นมีความยืดหยุ่นในการทำงานมากน้อยเพียงใด เมื่อต้องการปรับเปลี่ยนซอฟต์แวร์ให้เหมาะกับลักษณะการทำงานที่ผู้ใช้งานระบบในองค์กรคุ้นเคย หากผู้ขายไม่สามารถแก้ไขให้ตามความต้องการขององค์กรจัดว่าซอฟต์แวร์นั้นไม่มีความยืดหยุ่นพอ เนื่องจากผู้ใช้ระบบจะต้องมีการเปลี่ยนแปลงลักษณะการทำงานตามซอฟต์แวร์ซึ่งจะทำให้เกิดการปรับตัวพอสมควร
5. คู่มือประกอบการใช้งาน (Documentation)
   ซอฟต์แวร์ที่จะเลือกมานั้นมีคู่มือประกอบการใช้งานหรือไม่ หากมีคู่มือนั้นมีความละเอียดมากน้อยเพียงใด เนื่องจากในระหว่างการทำงานถ้ามีข้อสงสัยใดจะสามารถศึกษาได้จากคู่มือประกอบการใช้งานได้เลย โดยไม่ต้องเสียค่าใช้จ่ายในการติดต่อสื่อสารกับผู้ขาย เช่น สอบถามทางโทรศัพท์ เป็นต้น

การพัฒนาระบบด้วยองค์กรเอง (In-House Development)

การพัฒนาระบบด้วยบุคลากรขององค์กรเองยังเป็นทางเลือกหนึ่งที่น่าสนใจ หากบุคลากรขององค์กรมีความสามารถและความชำนาญมากพอ การพัฒนาระบบสามารถดำเนินการได้โดยไม่ต้องว่าจ้างบุคคลภายนอกเนื่องจากองค์กรมีทีมพัฒนา ซึ่งเป็นบุคลากรขององค์กรอยู่แล้ว หรือหากองค์กรมีบุคลากรไม่เพียงพออาจจะมีการผสมผสานการพัฒนาระบบโดยเลือกซอฟต์แวร์สำหรับงานบางอย่างจากภายนอกองค์กรเพื่อประยุกต์ใช้กับระบบที่มีอยู่แล้วภายในองค์กรก็สามารถทำได้ (ทั้งนี้การเลือกวิธีการต่างๆ ต้องมีการพิจารณาจากผู้บริหารและผู้สนับสนุนโครงการที่จะทำการตัดสินใจ)

การตรวจสอบข้อมูลของซอฟต์แวร์ที่ต้องการ

กรณีที่องค์กรต้องการซอฟต์แวร์จากภายนอก จำเป็นต้องมีการตรวจสอบข้อมูลการทำงานของซอฟต์แวร์ที่ต้องการจากผู้ขายหลายรายเพื่อเปรียบเทียบข้อมูลการทำงานของซอฟต์แวร์เหล่านั้นให้ตรงกับความต้องการของระบบมากที่สุด โดยอาจรวบรวมข้อมูลจากเอกสารหรือคู่มือประกอบการใช้งานของซอฟต์แวร์ หรือการติดต่อกับผู้ขายโดยตรงเพื่อสอบถามข้อมูลนี้โดยเฉพาะ หรืออาจศึกษาจากนิตยสารซอฟต์แวร์ตามท้องตลาดที่เป็นที่นิยมกันก็ได้

เพื่อที่จะให้ได้ซอฟต์แวร์ที่มีคุณภาพและตรงตามความต้องการของผู้ใช้ระบบมากที่สุด องค์กรควรมีการจัดลำดับให้กับซอฟต์แวร์ของผู้ขายแต่ละราย โดยอาจทำการทดสอบโปรแกรมและให้คะแนนแต่ละโปรแกรมโดยการถามความคิดเห็นจากผู้ใช้ระบบ

ฮาร์ดแวร์และซอฟต์แวร์ของระบบ

ไม่ว่านักวิเคราะห์ระบบจะเลือกซอฟต์แวร์จากแหล่งใดก็ตาม สิ่งแรกที่นักวิเคราะห์ระบบควรคำนึงถึงคือ ฮาร์ดแวร์และซอฟต์แวร์ระบบที่มีอยู่แล้วในองค์กรเพื่อประกอบการตัดสินใจเลือกซอฟต์แวร์ใหม่มาใช้ในองค์กร เนื่องจากการนำซอฟต์แวร์ใหม่มาใช้ในองค์กร หมายถึงซอฟต์แวร์นั้นจะต้องถูกใช้งานร่วมกับฮาร์ดแวร์และซอฟต์แวร์ระบบหรือระบบปฏิบัติการที่มีอยู่แล้ว และการที่จะประหยัดต้นทุนที่เกิดจากการซื้อซอฟต์แวร์จากแหล่งผู้ผลิตภายนอกองค์กร ซอฟต์แวร์นั้นจะต้องสามารถทำงานร่วมกับฮาร์ดแวร์และซอฟต์แวร์ระบบปฏิบัติการขององค์กรที่มีอยู่แล้วได้ โดยที่ไม่ต้องมีการแก้ไขเปลี่ยนแปลงอะไรมากมายนัก เหนือสิ่งอื่นใดซอฟต์แวร์ที่จะเลือกซื้อนั้นจะต้องถูกต้องตรงตามความต้องการของผู้ใช้ระบบมากที่สุด

ข้อดีของการเลือกซอฟต์แวร์ที่สามารถทำงานร่วมกับฮาร์ดแวร์และซอฟต์แวร์ระบบเดิมที่มีอยู่แล้วได้ มีดังนี้

1. สามารถลดต้นทุนในการซื้อฮาร์ดแวร์หรือซอฟต์แวร์ระบบใหม่ได้
2. บุคลากรในองค์กรจะมีความคุ้นเคยกับฮาร์ดแวร์ และระบบปฏิบัติการเดิมที่เคยใช้ ทำให้ไม่เกิดปัญหาอันสืบเนื่องมาจากฮาร์ดแวร์หรือระบบปฏิบัติการใหม่
3. การติดตั้งซอฟต์แวร์ใหม่ที่สามารถทำงานร่วมกับระบบปฏิบัติการเดิมและฮาร์ดแวร์เดิมได้นั้นมีความสะดวกและรวดเร็ว

ในทางกลับกันยังมีเหตุผลของการซื้อฮาร์ดแวร์และซอฟต์แวร์ระบบใหม่ที่น่าสนใจ ดังนี้

1. ซอฟต์แวร์บางชนิดสามารถทำงานบนระบบปฏิบัติการเฉพาะกับระบบใดระบบหนึ่งซึ่งเหมาะกับฮาร์ดแวร์บางชนิดเท่านั้น ดังนั้นองค์กรจึงไม่ต้องเสียเวลากับการเลือกระบบปฏิบัติการและฮาร์ดแวร์เพื่อนำมาทำงานร่วมกัน
2. การพัฒนาระบบที่ต้องซื้อระบบปฏิบัติการใหม่จะทำให้องค์กรสามารถขยายขีดความสามารถในการทำงานเนื่องจากใช้ระบบปฏิบัติการใหม่ และสรรหาเทคโนโลยีใหม่เพื่อช่วยให้องค์กรมีข้อได้เปรียบคู่แข่งขันทางด้านธุรกิจได้

ดังนั้น ในการเสนอทางเลือกในการนำระบบมาใช้งาน นอกจากจะคำนึงถึงฮาร์ดแวร์และซอฟต์แวร์ของระบบเดิมที่มีอยู่แล้วนักวิเคราะห์ระบบยังจะต้องสำรวจประสิทธิภาพในการใช้งานของฮาร์ดแวร์และซอฟต์แวร์ที่สามารถนำมาเป็นทางเลือกดังกล่าวด้วย แต่การที่จะได้มาซึ่งข้อมูลเหล่านั้นขึ้นอยู่กับวิธีการของนักวิเคราะห์ระบบของแต่ละองค์กรเอง

การติดตั้งระบบ

การติดตั้งระบบเป็นอีกปัจจัยหนึ่งที่มีผลต่อการกำหนดทางเลือก ไม่ว่าองค์กรเลือกที่จะพัฒนาระบบเอง หรือเลือกที่จะซื้อซอฟต์แวร์ของระบบงานจากแหล่งภายนอกองค์กรก็ตาม เมื่อระบบใหม่แล้วเสร็จจะต้องมีการติดตั้งระบบด้วย (Implementation) ดังนั้นสิ่งที่ผู้บริหารขององค์กรควรได้รับทราบในการกำหนดทางเลือกเพื่อแก้ปัญหาระบบงาน คือรายละเอียดในการติดตั้งระบบงานใหม่นั้น ซึ่งได้แก่

1. ระยะเวลาที่ใช้ในการติดตั้ง
2. ขั้นตอนในการติดตั้ง
3. ระยะเวลาในการฝึกอบรมผู้ใช้ระบบ

รายละเอียดดังกล่าว เป็นข้อมูลเพื่อประกอบการตัดสินใจของผู้บริหารในการเลือกแนวทางเพื่อการนำระบบใหม่มาใช้งาน หากทางเลือกใด ที่ต้องใช้ระยะเวลาในการติดตั้งยาวนานและมีขั้นตอนในการติดตั้งที่ซับซ้อนมากเกินไป อาจทำให้ผู้บริหารปฏิเสธทางเลือกนั้นได้เมื่อพิจารณาถึงต้นทุนที่ตามมาเนื่องจากระยะเวลานั่นเอง

5.3. การวางแผนโครงการ (Project Planning)

หลังจากการเริ่มต้นโครงการด้วยการจัดตั้งทีมงาน วางแผนการดำเนินโครงการ และเริ่มจัดทำสมุดงานโครงการแล้วทีมงานดังกล่าวจะต้องสร้างแนวทางเลือกในการนำระบบใหม่มาใช้งานซึ่งพิจารณาจากปัจจัยหลายประการดังที่ได้กล่าวไว้แล้ว จากนั้นทีมงานต้องเลือกแนวทางเลือกที่ดีที่สุดเพียง 1 แนวทางเพื่อนำมาวางแผนดำเนินโครงการต่อไป

การวางแผนโครงการเป็นกิจกรรมที่ดำเนินการต่อจากการเลือกแนวทางที่ต้องการได้แล้ว โดยในการวางแผนโครงการมีกิจกรรมดังนี้

1. แสดงรายละเอียดขอบเขตของโครงการ

เป็นการระบุถึงขอบเขตของโครงการพัฒนาระบบนั้น และสรุปรายละเอียดของโครงการ ได้แก่ ปัญหาของระบบที่จะพัฒนาข้อจำกัด และสถานะปัจจุบันของระบบ เป็นต้น

2. รายงานการศึกษาความเป็นไปได้และการประมาณการใช้งบประมาณ

แสดงรายละเอียดการประมาณค่าใช้จ่ายที่จะต้องใช้ไปและรายได้ที่จะได้รับของโครงการพัฒนาระบบในเบื้องต้น รวมทั้งแสดงรายงานการศึกษาความเป็นไปได้ไม่ว่าจะเป็นด้านเศรษฐศาสตร์ ด้านเทคนิค ด้านการปฏิบัติงาน หรือด้านระยะเวลาดำเนินงาน ทั้งนี้ขึ้นอยู่กับองค์กรว่ามีความประสงค์จะแสดงรายงานการศึกษาความเป็นไปได้ด้านใดบ้าง สำหรับรายละเอียดการศึกษาความเป็นไปได้

3. ประมาณการใช้แหล่งทรัพยากรและวางแผนการใช้ทรัพยากรนั้น

แสดงรายละเอียดความต้องการใช้ทรัพยากรของแต่ละกิจกรรม และวางแผนการใช้ทรัพยากรนั้น เช่น การว่าจ้างโปรแกรมเมอร์เพิ่มเพื่อช่วยงานด้านการเขียนโปรแกรมนั้นจำเป็นหรือไม่ เมื่อเปรียบเทียบกับการใช้โปรแกรมเมอร์ขององค์กรเอง

4. แบ่งแยกกิจกรรมในการดำเนินการพัฒนาระบบ

ทำการแบ่งแยกกิจกรรมทั้งหมดที่ต้องดำเนินการในโครงการพัฒนาระบบ เรียบลำดับกิจกรรมและแสดงรายละเอียดของแต่ละกิจกรรมตามความเหมาะสม แต่หากแสดงรายละเอียดมากเกินไปจะทำให้การบริหารโครงการ (Project Management) มีความซับซ้อนตามมา

5. จัดตารางระยะเวลาดำเนินการในเบื้องต้น

จากกิจกรรมที่ได้แบ่งแยกและจัดเรียงไว้แล้ว ให้นำมาใช้กำหนดระยะเวลาในการดำเนินการแต่ละกิจกรรม

โดยการระบุจะเริ่มที่วันเริ่มดำเนินการกิจกรรมและวันสิ้นสุดกิจกรรม ซึ่งในการแสดงระยะเวลาดำเนินการนี้อาจนำเสนอในรูปของ Gantt Chart หรือ PERT Chart (ศึกษารายละเอียดได้จาก ภาคผนวก ก “การบริหารโครงการ”)

6. วางแผนการติดต่อสื่อสารกับผู้ที่เกี่ยวข้องในระหว่างการพัฒนาระบบ

ทำการวางแผนการดำเนินการติดต่อประสานงานระหว่างทีมพัฒนาระบบ กับผู้ใช้ระบบหรือผู้ที่เกี่ยวข้องอื่นๆ โดยรวมถึงการระบุถึงวันที่ที่จะเขียนรายงานเพื่อเสนอแก่ผู้บริหาร ทีมพัฒนาระบบจะประสานงานในระหว่างการทำงานอย่างไร และมีข้อมูลใดบ้างที่ผู้เกี่ยวข้องสามารถรับทราบได้

7. จัดทำมาตรฐานในการดำเนินงาน

ทำการระบุผลที่ได้จากการดำเนินงานในแต่ละขั้นตอนเพื่อประโยชน์ในการตรวจสอบ โดยอาจจะกำหนดรูปแบบของผลลัพธ์และรูปแบบของรายงานที่ใช้แสดงความคืบหน้าในการดำเนินงาน เพื่อให้เป็นมาตรฐานเดียวกัน และสามารถตรวจสอบได้ว่าควรได้รับการแก้ไขหรือไม่

8. ระบุและประเมินความเสี่ยง

ระบุถึงแหล่งที่มาที่อาจทำให้เกิดความเสี่ยงในการลงทุนดำเนินการโครงการ พร้อมทั้งประเมินระดับความเสี่ยงที่จะเกิดขึ้นโดยความเสี่ยงนั้นอาจเกิดจากการนำเทคโนโลยีใหม่เข้ามาใช้ การต่อต้านต่อการเปลี่ยนแปลงระบบของผู้ใช้งาน สภาพการแข่งขันทางธุรกิจ หรือแม้กระทั่งความไม่มีประสบการณ์ในเรื่องของเทคโนโลยีใหม่ๆ หรือในเรื่องของการดำเนินงานด้านธุรกิจของทีมงานพัฒนาระบบ

9. จัดทำรายงานแสดงสถานะของงาน (Developing a Statement Of Work : SOW)

เป็นการจัดทำเอกสารเพื่อผู้บริหารขององค์กรหรือลูกค้า โดยแสดงรายละเอียดของงานที่จะต้องทำทั้งหมดและผลที่จะได้รับอย่างชัดเจน เอกสารชุดนี้จะเป็นประโยชน์ในการสร้างความเข้าใจที่ตรงกันของทีมพัฒนาระบบ ลูกค้า และผู้บริหาร

10. จัดทำแผนงาน/โครงการ (Baseline Project Plan)

Baseline Project Plan (BPP) เป็นเอกสารที่แสดงรายละเอียดขอบเขตของโครงการ ต้นทุน กำไร ความเสี่ยง และความต้องการใช้ทรัพยากร โดยในชุดเอกสาร BPP นี้ประกอบไปด้วย 4 ส่วน ดังนี้

- ส่วนแนะนำโครงการ แสดงขอบเขตของโครงการและแหล่งทรัพยากรที่จะต้องใช้ เป็นต้น
- ส่วนรายละเอียดของระบบ แสดงรายละเอียดการทำงานของระบบอย่างคร่าวๆ ข้อมูลนำเข้าและออกจากระบบ
- ส่วนรายละเอียดการศึกษาความเป็นไปได้ (Feasibility Study) แสดงการศึกษาความเป็นไปได้ทั้ง 4 ด้าน
- ส่วนรายละเอียดการบริหารโครงการ แสดงรายละเอียดของทีมงานพัฒนาระบบ แผนงาน และมาตรฐานในการทำงาน

สิ่งสำคัญที่ได้จากขั้นตอนการเริ่มต้น การนำเสนอแนวทางเลือกในการนำระบบมาใช้งาน และการวางแผนโครงการ คือ สมุดงานโครงการ (Project Workbook) ที่ประกอบไปด้วยเอกสาร State of

Work และ Baseline Project Plan โดยที่ State of Work จะแสดงให้เห็นถึงงานหรือกิจกรรมที่จะต้องทำทั้งหมดของโครงการและที่เกิดจากกิจกรรมแต่ละขั้นตอน ส่วน Baseline Project Plan จะแสดงถึงขอบเขตของโครงการที่แท้จริง ต้นทุน กำไร ความเสี่ยง และความต้องการใช้ทรัพยากร โดยเอกสารทั้งสองชุดนี้จะช่วยให้สามารถเข้าใจถึงสถานะปัจจุบันของระบบที่จะต้องพัฒนาได้ และช่วยในการบริหารโครงการตามรายละเอียดของกิจกรรมที่อาจนำเสนอไว้ด้วย Gantt Chart และ PERT Chart

5.3.1. การศึกษาความเป็นไปได้ของโครงการ (Project Feasibility Study)

ในกิจกรรมการวางแผนดำเนินการโครงการพัฒนาระบบนี้ ถึงแม้หน้าที่หลักของนักวิเคราะห์ระบบคือ การวิเคราะห์และออกแบบระบบตามความต้องการของผู้ใช้งานและเจ้าของระบบ แต่ด้วยความรับผิดชอบของนักวิเคราะห์ระบบที่เป็นตัวแทนการเปลี่ยนแปลงที่จะเกิดขึ้นในระบบ จึงต้องมีการศึกษาความเป็นไปได้และวิเคราะห์ต้นทุนและกำไรของโครงการ เพื่อเป็นการเพิ่มความมั่นใจให้กับผู้ใช้และเจ้าของระบบก่อนที่การเปลี่ยนแปลงจะเกิดขึ้น

หัวข้อหลักการพัฒนาระบบนั้นเจ้าของระบบจะมองการพัฒนาระบบเป็นการลงทุน ดังนั้นสิ่งที่เป็นปัจจัยประกอบการตัดสินใจอนุมัติโครงการพัฒนาระบบให้สามารถดำเนินการต่อไปได้ คือ เรื่องของต้นทุนและผลกำไร หรือผลตอบแทนที่จะได้รับ ดังนั้นการเสนอโครงการที่มีการวิเคราะห์ความเป็นไปได้ในเรื่องของผลตอบแทน ต้นทุนและผลกำไรจึงเป็นหัวข้อหลักในการเสนอโครงการ

ความเป็นไปได้ (Feasibility) หมายถึง การพิจารณาถึงความเหมาะสมและการประเมินผลประโยชน์เปรียบเทียบกับค่าใช้จ่ายที่ใช้ไปในการพัฒนาระบบขององค์กร

ในการศึกษาความเป็นไปได้ของโครงการพัฒนาระบบ มีปัจจัยที่ใช้เป็นหลักเกณฑ์ในการพิจารณา 4 ประการดังนี้

1. ความเป็นไปได้ด้านเศรษฐศาสตร์ (Economic Feasibility)
2. ความเป็นไปได้ด้านเทคนิค (Technical Feasibility)
3. ความเป็นไปได้ด้านการปฏิบัติงาน (Operational Feasibility)
4. ความเป็นไปได้ด้านเวลาการดำเนินงาน (Schedule Feasibility)

ความเป็นไปได้ด้านเศรษศาสตร์ (Economic Feasibility)

การศึกษาความเป็นไปได้ทางเศรษฐศาสตร์หรือเรียกอีกอย่างหนึ่งว่า “การวิเคราะห์ต้นทุนและผลตอบแทน (Cost – Benefits Analysis) “ เป็นการศึกษาถึงผลตอบแทนทางการเงินและต้นทุนที่เกิดขึ้นจากโครงการพัฒนาระบบ

วัตถุประสงค์ที่สำคัญของการศึกษาความเป็นไปได้ทางด้านเศรษฐศาสตร์คือ การจำแนกผลตอบแทนต้นทุนที่จะใช้ในโครงการพัฒนาระบบ ในการวิเคราะห์ต้นทุนและผลตอบแทนจะใช้ฟังก์ชันทางการเงินเพื่อคำนวณหาต้นทุนและกำไรตลอดจนผลตอบแทนที่คาดว่าจะได้รับ โดยมีวิธีการดังต่อไปนี้

- การพิจารณาผลตอบแทนที่จะได้รับจากโครงการ

- พิจารณาต้นทุนของโครงการ
- คำนวณผลตอบแทนสุทธิที่จะได้รับจากโครงการ

1. การพิจารณาผลตอบแทนที่จะได้รับจากโครงการ

ผลตอบแทนของโครงการเป็นสิ่งสำคัญที่ผู้บริหารให้ความสนใจเทียบเท่ากับต้นทุนที่ต้องใช้ การที่โครงการพัฒนาระบบจะสามารถเพิ่มผลประโยชน์ที่อยู่ในรูปของกำไรให้กับองค์กรได้ นั่นหมายถึงใช้ต้นทุนน้อยนั่นเอง ซึ่งการพิจารณาถึงผลตอบแทนของโครงการสามารถจำแนกลักษณะได้ 2 ประเภทดังนี้

1.1. ผลตอบแทนที่จับต้องได้ (Tangible Benefits) หมายถึง ผลตอบแทนที่สามารถประเมินค่าเป็นตัวเงินได้ เช่น กำไรการลดต้นทุนต่อหน่วย การลดความผิดพลาดของการนำเข้าข้อมูล การเพิ่มความเร็วในการประมวลผลข้อมูลที่นำเข้าการเพิ่มยอดขาย เป็นต้น

1.2. ผลตอบแทนที่จับต้องไม่ได้ (Intangible Benefits) หรือผลตอบแทนที่ไม่ใช่ตัวเงิน หมายถึง ผลตอบแทนที่ไม่สามารถวัดค่าเป็นตัวเงินได้ หรือยากแก่การประเมินค่า เช่น การเพิ่มภาพลักษณ์ที่ดีให้แก่องค์กร การสร้างขวัญและกำลังใจให้แก่พนักงาน การคืนผลประโยชน์สู่สังคม และการเพิ่มประสิทธิภาพในการตัดสินใจของผู้บริหาร เป็นต้น แสดงตัวอย่างผลตอบแทนที่จับต้องได้และจับต้องไม่ได้ ดังตารางต่อไปนี้

ผลตอบแทนที่จับต้องได้ (Tangible Benefits)	ผลตอบแทนที่จับต้องไม่ได้ (Intangible Benefits)
1. ความผิดพลาดในการประมวลผลลดน้อยลง 2. การเพิ่มความเร็วในการประมวลผล 3. ลดขั้นตอนในการทำงาน 4. ลดค่าใช้จ่าย 5. เพิ่มยอดขาย 6. ลดจำนวนลูกหนี้	1. ภาพลักษณ์ที่ดีขึ้นขององค์กร 2. ความเต็มใจในการทำงานของลูกจ้าง 3. การบริการขององค์กรที่มีต่อสังคม 4. การตัดสินใจที่ดีขึ้น 5. การมีเครดิตดีขึ้น

เนื่องจากในกรณีการประเมินผลตอบแทนที่จับต้องไม่ได้ที่จะได้จากโครงการค่อนข้างมีความลำบาก และหากไม่สามารถประเมินผลตอบแทนที่จะได้รับจากโครงการได้ นั่นหมายถึงไม่สามารถวิเคราะห์ต้นทุนและผลตอบนั้นได้ส่งผลให้โครงการนั้นไม่เป็นที่ยอมรับในที่สุด

ดังนั้นจึงจำเป็นต้องหาค่าความเป็นไปได้จากผลประโยชน์ที่ไม่ใช่ตัวเงิน เช่น กรณีภาพลักษณ์ขององค์กรที่มีต่อลูกค้าหลังจากมีการติดตั้งระบบใหม่ มีรายละเอียดดังนี้

- จากคำถามที่ว่า “จะเกิดอะไรขึ้นหากลูกค้ามองภาพพจน์ขององค์กรไม่ดี” และหากคำตอบที่ได้คือลูกค้าจะสั่งซื้อสินค้าในปริมาณที่ลดลงหรือไม่สั่งเลย
- “ปริมาณการสั่งซื้อสินค้าลดลงในระดับใด” จากคำถามดังกล่าวการวิเคราะห์ถึงระดับการลดลงของปริมาณการสั่งซื้อสินค้า อาจใช้การวิเคราะห์ค่าความเป็นไปได้ของโอกาสที่จะเกิดในการสั่งซื้อในแต่ละกรณี

2. การพิจารณาต้นทุนของโครงการ

ต้นทุนสามารถแบ่งได้ 2 ลักษณะ คือต้นทุนที่จับต้องได้ (Tangible Costs) และต้นทุนที่จับต้องไม่ได้ (Intangible Costs)

1. ต้นทุนที่จับต้องได้ (Tangible Costs) คือต้นทุนในส่วนของการพัฒนาระบบที่สามารถประเมินค่าเป็นตัวเงินได้ เช่น ต้นทุนในการซื้อเครื่องคอมพิวเตอร์ เงินเดือน และต้นทุนที่ใช้ในการดำเนินงานเมื่อทำการติดตั้งระบบ (ค่าใช้จ่ายในการฝึกอบรมพนักงานและค่าใช้จ่ายในการปรับปรุงระบบ)
2. ต้นทุนที่จับต้องไม่ได้ (Intangible Costs) คือต้นทุนในส่วนของการพัฒราระบบที่ไม่สามารถประเมินค่าเป็นตัวเงินได้ ได้แก่ ความไม่เต็มใจในการทำงานของพนักงาน และการทำงานที่ไม่มีประสิทธิภาพ

จากลักษณะของต้นทุนทั้งที่เป็นต้นทุนที่จับต้องได้และจับต้องไม่ได้ นักวิเคราะห์ระบบยังสามารถจำแนกต้นทุนในส่วนของการพัฒนาระบบออกได้อีก 2 ประเภท คือ ต้นทุนที่เกิดขึ้นครั้งเดียว (One-time Costs) และต้นทุนที่เกิดขึ้นซ้ำอีก (Recurring Costs)

1. ต้นทุนที่เกิดขึ้นครั้งเดียว (One – time Costs) คือต้นทุนที่เกิดขึ้นในการเริ่มต้นโครงการ และเกิดขึ้นเมื่อมีการเริ่มใช้งานระบบ เช่น ค่าใช้จ่ายในการซื้อเครื่องคอมพิวเตอร์ใหม่ ค่าใช้จ่ายในการซื้อซอฟต์แวร์ ค่าใช้จ่ายในการฝึกอบรม
2. ต้นทุนที่เกิดขึ้นซ้ำอีก (Recurring Costs) คือ ต้นทุนที่เกิดในระหว่างการดำเนินงานของระบบใหม่ เช่น ค่าใช้จ่ายในการบำรุงรักษาโปรแกรม การซื้อสื่อเก็บข้อมูลเพิ่มเติม ค่าใช้จ่ายที่เกิดจากการติดต่อสื่อสาร ค่าใช้จ่ายเกี่ยวกับอุปกรณ์สำนักงาน

นอกจาก One-time Costs และ Recurring Costs แล้ว ในส่วนของการพัฒนาระบบ ต้นทุนยังสามารถจำแนกได้อีก 2 ประเภท คือ ต้นทุนคงที่ (Fixed Costs) และต้นทุนผันแปร (Variable Costs)

1. ต้นทุนคงที่ (Fixed Costs) คือ ต้นทุนที่ไม่เปลี่ยนแปลงไปตามการใช้งานหรือการผลิตอื่นๆ เช่น ค่าบำรุงไฟฟ้า น้ำประปา เงินเดือนพนักงาน
2. ต้นทุนผันแปร (Variable Costs) คือ ต้นทุนที่แปรผันไปตามการใช้งานหรือการผลิตอื่นๆ เช่น ค่าใช้โทรศัพท์ที่ไม่รวมค่าบริการรายเดือนที่ต้องจ่ายเท่ากันในทุกๆ เดือน

ความเป็นไปได้ทางด้านเทคนิค (Technical Feasibility)

การศึกษาความเป็นไปได้ทางด้านเทคนิคมีวัตถุประสงค์ เพื่อทำให้เข้าใจถึงความสามารถในการพัฒนาระบบใหม่ขององค์กร และเป็นการประเมินเทคนิคของระบบใหม่ที่ใช้ในการแก้ปัญหา โดยอาจจะอาศัยคำถามเพื่อเป็นแนวทางในการประเมิน ดังนี้

1. เทคโนโลยีที่จะนำมาใช้นั้นสามารถรองรับปริมาณลูกค้าที่อาจเพิ่มจำนวนมากขึ้น และสามารถปรับเข้ากับปัญหาที่จะเกิดขึ้นได้หรือไม่
2. เทคโนโลยีที่มีอยู่เดิมนั้นสามารถปรับใช้กับระบบใหม่ได้หรือไม่ ถ้าไม่ได้ องค์กรสามารถซื้อมาได้โดยมีค่าใช้จ่ายที่ผู้บริหารพึงพอใจหรือไม่
3. บุคลากรขององค์กรมีความเชี่ยวชาญกับเทคโนโลยีที่จะนำมาใช้มากพอหรือไม่

นอกจากจะประเมินความสามารถขององค์กรในการพัฒนาระบบ ของโครงการพัฒนาระบบที่คัดเลือกมาว่ามีความสามารถเพียงพอหรือไม่แล้ว ยังจะต้องทำการประเมินระดับความเสี่ยงของโครงการ เนื่องจากผู้บริหารย่อมมีความคาดหวังผลตอบแทนที่ได้จากโครงการมากกว่าความเสี่ยงในด้านต่างๆ ที่จะเกิดขึ้น ดังนั้นจึงควรมีการประเมินความเสี่ยงของโครงการเพื่อป้องกันผลลัพธ์ที่ไม่พึงประสงค์ที่อาจเกิดขึ้น อันเนื่องจากการนำเทคโนโลยีเข้ามาใช้งานกับระบบใหม่ โดยผลลัพธ์ที่อาจเป็นไปได้หากไม่มีการประเมินความเสี่ยงของโครงการ มีดังนี้

1. ทำให้การคาดหวังที่จะได้รับผลตอบแทนนั้นล้มเหลว
2. ทำให้การประมาณการต้นทุนผิดพลาด
3. ทำให้การประมาณการระยะเวลาในการดำเนินโครงการผิดพลาด
4. ทำให้ประสิทธิภาพในการทำงานของระบบไม่เป็นไปตามที่คาดไว้
5. ทำให้ไม่สามารถติดตั้งระบบใหม่เข้ากับระบบคอมพิวเตอร์ที่มีอยู่แล้วได้

นักวิเคราะห์ระบบหรือผู้บริหารควรมีการป้องกันการเกิดความเสี่ยงในด้านต่างๆ ดังที่กล่าวไว้แล้ว โดยอาจจะมีการแต่งตั้งทีมงานเพื่อคอยควบคุมไม่ให้เกิดผลลัพธ์ดังกล่าวได้ อาจจะใช้เทคนิคการประเมินปัจจัยที่จะทำให้เกิดความเสี่ยงได้ทั้งหมด 4 ประการ ได้แก่

1. ขนาดของโครงการ โครงการที่มีขนาดใหญ่จะมีความเสี่ยงมากกว่าโครงการที่มีขนาดเล็ก เนื่องจากโครงการที่มีขนาดใหญ่จะยากต่อการบริหารโครงการ
2. โครงสร้างของโครงการ โครงการที่มีการดำเนินงานอย่างมีโครงสร้างและมีความต้องการ (Requirement) ที่ไม่ซับซ้อนย่อมมีความเสี่ยงน้อยกว่า โครงการที่มีความต้องการ (Requirement) ที่มีความซับซ้อน
3. เทคโนโลยีที่นำมาใช้ในโครงการ โครงการที่นำเทคโนโลยีที่มีมาตรฐานมาใช้ย่อมมีความเสี่ยงต่อความเข้าใจของกลุ่มผู้ใช้งานน้อยกว่าโครงการที่นำเทคโนโลยีที่ไม่มีมาตรฐานเพียงพอ หรือล้ำยุคเกินไป
4. ความคุ้นเคยของผู้ใช้งานกับการพัฒนาระบบสารสนเทศ ผู้ใช้งานที่มีความคุ้นเคยกับระบบสารสนเทศ จะมีความเข้าใจในขั้นตอนการทำงานได้ดีกว่าผู้ใช้งานที่ไม่มีความคุ้นเคย

ความเป็นไปได้ทางด้านการปฏิบัติงาน (Operational Feasibility)

เป็นการประเมินถึงระบบใหม่เมื่อมีการใช้งาน ว่าจะสามารถแก้ไขปัญหาของระบบเดิมได้มากน้อยเพียงใด รวมถึงความรู้สึกของผู้ใช้ระบบที่มีต่อการทำงานของระบบใหม่ด้วย

การจะประเมินว่าระบบใหม่นั้นจะสามารถแก้ไขปัญหาของระบบเดิมได้มากน้อยเพียงใด มีหลักเกณฑ์ในการพิจารณาดังนี้

1. ประสิทธิภาพ (Performance) ระบบใหม่นั้นมีความเร็วในการทำงานมากน้อยเพียงใด

2. สารสนเทศ (Information) สารสนเทศที่จะได้จากระบบใหม่นั้น มีความถูกต้อง ตรงประเด็น และสามารถใช้ร่วมกันได้หรือไม่
3. เศรษฐศาสตร์ (Economy) ระบบใหม่นั้นสามารถช่วยลดต้นทุนหรือเพิ่มกำไรให้กับองค์กรได้อย่างไร
4. การควบคุม (Control) มีความสามารถในการควบคุมระบบเพื่อป้องกันการโกงและการยักยอก และมีความถูกต้องปลอดภัย ของข้อมูลมากน้อยเพียงใด
5. ประสิทธิผล (Efficiency)d ระบบใหม่จะต้องมีการใช้แหล่งทรัพยากรมากที่สุดเพียงใด เช่น ทรัพยากรบุคคล เวลา ข้อมูล เป็นต้น
6. การบริการ (Services) ระบบใหม่มีการเตรียมการบริการเมื่อเกิดปัญหาแก่ผู้ใช้งาน และมีความยืดหยุ่นหรือไม่

การประเมินว่าผู้ใช้ระบบจะมีความรู้สึกอย่างไรต่อระบบใหม่นั้น อาจอาศัยคำถามในการประเมินดังนี้

1. การบริหารองค์กรกับระบบมีความสอดคล้องกันหรือไม่
2. ผู้ใช้ระบบมีความรู้สึกเช่นไรกับบทบาทของตนที่มีต่อระบบใหม่
3. ผู้ใช้ระบบหรือเจ้าของระบบมีการต่อต้านหรือมีแนวโน้มการต่อต้านระบบใหม่หรือไม่ ถ้ามีจะสามารถแก้ปัญหาได้อย่างไร
4. มีผลกระทบต่อสิ่งแวดล้อมทางกายภาพต่อผู้ใช้หรือไม่ ถ้ามีผู้ใช้ระบบจะสามารถปรับตัวได้อย่างไร

การประเมินการใช้งานระบบ (Usability) อาจมีหลักเกณฑ์ดังนี้

1. ง่ายต่อการเรียนรู้หรือไม่
2. ง่ายต่อการใช้งานหรือไม่
3. ผู้ใช้งานพึงพอใจหรือไม่

ความเป็นไปได้ทางด้านระยะเวลาการดำเนินงาน (Schedule Feasibility)

เป็นการประเมินระยะเวลาการดำเนินงานในโครงการพัฒนาระบบใหม่นั้น ว่ามีความเหมาะสมหรือไม่ตามข้อจำกัดทางด้านเวลาที่องค์กรประมาณไว้ หากพิจารณาแล้วว่าใช้เวลาในการดำเนินการมากเกินไป นักวิเคราะห์ระบบจำเป็นต้องวางแผนการดำเนินงานใหม่ และต้องดำเนินการให้เสร็จสิ้นตามเวลาที่ได้กำหนดไว้ในแผนงาน

การที่จะประมาณการเวลาว่าระบบใหม่นั้นจะสามารถแก้ไขปัญหาของระบบเดิมได้มากน้อยเพียงใด มีหลักเกณฑ์ในการพิจารณาดังนี้

1. ประมาณการระยะเวลาของโครงการทั้งหมด โดยคำนึงถึงขนาดของโครงการ ปริมาณงานที่จะต้องดำเนินการ ตลอดจนความต้องการทรัพยากรด้านต่างๆ อาทิเช่น จำนวนของระบบย่อย ความยากง่ายทางด้านเทคนิค ทักษะของทีมงาน การตอบสนองของระบบโดยรวม เป็นต้น
2. คำนวณเวลาที่ต้องใช้จริง เป็นการกำหนดช่วงระยะเวลาในการดำเนินงานที่จะต้องเกิดขึ้นจริง
3. คำนวณเวลาที่สูญเสียไป เป็นการพิจารณาเวลาที่ต้องสูญเสียไปในหนึ่งวัน

4. คำนวณแรงงานที่ต้องใช้ในแต่ละกิจกรรม เป็นการพิจารณาระยะเวลาที่ต้องใช้ของแต่ละคนของแต่ละกิจกรรมต่อสัปดาห์
5. คำนวณระยะเวลาของโครงการ เป็นการคำนวณระยะเวลาที่สูญเสียไปกับระยะเวลาของแต่ละกิจกรรมและของโครงการทั้งหมด
6. ทบทวนและปรับปรุงค่าของการประมาณการระยะเวลา เพื่อให้ค่าที่ได้ใกล้เคียงกับความจริง เนื่องจากมุมมองของทีมงานมักจะคิดระยะเวลาเผื่อไว้ ในขณะที่มุมมองของผู้บริหารต้องการพิจารณาถึงความจำเป็นและระยะเวลาที่เหมาะสม

เมื่อวิเคราะห์ความเป็นไปได้เสร็จแล้ว นักวิเคราะห์ระบบจะต้องนำเสนอรายงานผลการศึกษาความเป็นไปได้ต่อผู้บริหารเพื่อพิจารณาอนุมัติ จากนั้นจึงเริ่มเข้าสู่ขั้นตอนการวิเคราะห์ระบบ (System Analysis)s ต่อไป โดยผลการอนุมัตินั้นอาจเป็นไปได้สองทาง ได้แก่ อนุมัติ และไม่อนุมัติ

1. กรณีอนุมัติ นักวิเคราะห์ระบบเริ่มดำเนินการวิเคราะห์ระบบในขั้นตอนต่อไปได้
2. กรณีไม่อนุมัติ อาจเป็นไปได้ทั้งไม่อนุมัติในบางหลักการหรือแนวทางการดำเนินงานในบางกิจกรรม หรือไม่อนุมัติทั้งโครงการ ทั้งนี้ขึ้นอยู่กับเหตุผลและสถานการณ์ขององค์กรในขณะนั้น เช่น แนวทางดังกล่าวมีต้นทุนสูงเกินไป หรือมีระยะเวลาดำเนินการไม่เหมาะสมกับสถานการณ์ที่เปลี่ยนแปลงในขณะนั้น เป็นต้น ดังนั้น นักวิเคราะห์ระบบจำเป็นต้องนำกลับมาทบทวนหรือปรับปรุงแผนงานใหม่เพื่อนำเสนอใหม่อีกครั้ง แต่หากว่าโครงการไม่ได้รับการอนุมัติด้วยเหตุผลบางประการ นั้นหมายถึงโครงการต้องถูกยกเลิกไปโดยปริยาย

บทที่ 6
การกำหนดความต้องการของระบบ (System Requirements Determination)

ขั้นตอนการวิเคราะห์ระบบ (System Analysis Phase) ในที่นี้จะเริ่มต้นที่การกำหนดความต้องการของระบบ (System Requirement Determination) ด้วยการรวบรวมข้อมูลในระบบเดิม ซึ่งจะทำให้ทราบขั้นตอนการทำงานและปัญหาที่เกิดขึ้นนอกจากนี้ นักวิเคราะห์ระบบจะต้องเก็บรวบรวมข้อมูลจากผู้ใช้ว่าต้องการสิ่งใดเพิ่มเติมอีกถ้าจะพัฒนาระบบขึ้นมาใหม่ ทั้งนี้เพื่อนำไปวิเคราะห์หาแนวทางในการแก้ไขปัญหาที่เกิดขึ้นจากระบบเดิมและพัฒนาให้เป็นระบบใหม่ที่ตรงต่อความต้องการของผู้ใช้ในที่สุด

ดังนั้น ในบทนี้จะอธิบายวิธีการเก็บรวบรวมข้อมูล ข้อเท็จจริงและสารสนเทศด้วยเทคนิคต่างๆ เช่นการสัมภาษณ์ การออกแบบสอบถาม การสังเกตสภาพแวดล้อมในสถานที่ทำงาน และการรวบรวมข้อมูลจากเอกสารขององค์กร เพื่อให้ได้ข้อมูลที่เป็นจริงและถูกต้องมากที่สุด

6.1. แนะนำการกำหนดความต้องการของระบบ

การกำหนดความต้องการของระบบ คือการวิเคราะห์ถึงการทำงานของระบบเดิมเพื่อหาปัญหาที่เกิดขึ้นจริงๆ แต่ก่อนที่จะวิเคราะห์การทำงานของระบบเดิมนั้น จะต้องมีการเก็บรวบรวมข้อมูลและข้อเท็จจริงของระบบเดิม ซึ่งเป็นหน้าที่อย่างหนึ่งของนักวิเคราะห์ระบบที่จะต้องดำเนินการ โดยนักวิเคราะห์ระบบและทีมงานจะต้องพบและพูดคุยกับผู้ใช้ระบบในองค์กรเพื่อให้ได้ข้อมูลที่ถูกต้อง ทั้งนี้ไม่ว่าจะเป็นวิธีการใดก็ตามนักวิเคราะห์ระบบควรคำนึงถึงระยะเวลาที่ใช้ในการเก็บรวบรวมข้อมูลด้วย

สิ่งที่ได้จากการรวบรวมข้อมูลคือ แบบฟอร์ม รายงาน รายละเอียดการทำงาน และเอกสารอื่นๆ ที่เป็นแหล่งข้อมูลขององค์กรภายในเอกสารดังกล่าวจะทำให้นักวิเคราะห์ระบบทราบถึงรายละเอียดของระบบได้มากมายเช่น

1. เป้าหมายขององค์กรทำให้ทราบว่าองค์กรดำเนินธุรกิจอะไรและอย่างไร
2. สารสนเทศที่ผู้ใช้ระบบต้องการในการดำเนินงาน
3. ประเภทของข้อมูล ขนาด และจำนวนข้อมูลที่เกิดขึ้นในระหว่างการทำงาน
4. ข้อมูลเกิดขึ้นเมื่อใด เกิดขึ้นได้จากขั้นตอนใดของระบบ และข้อมูลจากขั้นตอนหนึ่งไปยังขั้นตอนใดต่อไปและอย่างไร
5. ลำดับขั้นตอนการทำงาน
6. เงื่อนไขต่างๆ ที่เกิดขึ้นในระหว่างการประมวลผลข้อมูลนั้น
7. นโยบายในการปฏิบัติงาน
8. เหตุการณ์สำคัญใดบ้างที่มีผลกระทบต่อข้อมูล และเหตุการณ์เหล่านั้นจะเกิดขึ้นเมื่อใด

ข้อมูลที่รวบรวมมาได้ จะมีรายละเอียดค่อนข้างมากและซับซ้อน ในบางครั้งหากนำข้อมูลเหล่านี้ไปใช้ในการติดต่อสื่อสารกับผู้บริหารหรือเจ้าของระบบอาจทำให้มีความเข้าใจที่ไม่ตรงกัน หรือหากมีความเข้าใจที่ตรงกันก็อาจต้องใช้เวลานาน เนื่องจากข้อมูลเหล่านั้นเป็นข้อความซึ่งทำให้ไม่สามารถมองภาพรวมการทำงานของระบบได้ชัดเจนเท่าที่ควร ดังนั้นจึงต้องมีการจำลองความต้องการเหล่านั้นด้วยแผนภาพชนิดต่างๆ (จะกล่าวในบทถัดไป) เพื่อให้สามารถเข้าใจภาพรวมการทำงานของระบบได้ชัดเจนและรวดเร็วยิ่งขึ้น

แต่ก่อนที่จะก้าวไปสู่การจำลองข้อมูลและการทำงานของระบบด้วยแผนภาพชนิดต่างๆ ในที่นี้ขอแนะนำเทคนิคและวิธีการในการรวบรวมข้อเท็จจริงและสารสนเทศ (Fact-finding and Information Gathering) ที่จะทำให้ได้ข้อมูลมาอย่าง

เป็นลำดับและไม่ซับซ้อน โดยจะเริ่มจากทฤษฎีแบบดั้งเดิมที่ใช้ในการกำหนดความต้องการของระบบจนถึงการใช้ทฤษฎีแนวใหม่ที่จะทำให้ข้อมูลรวดเร็ว ครบถ้วน และถูกต้องยิ่งขึ้น

6.2. ทฤษฎีแบบดั้งเดิมที่ใช้ในการเก็บรวบรวมข้อเท็จจริงและสารสนเทศ

หน้าที่หลักของนักวิเคราะห์ระบบคือ วิเคราะห์ และออกแบบระบบงานตามความต้องการของหัวหน้าโครงการ ผู้ใช้ และเจ้าของระบบ ดังนั้นสิ่งที่นักวิเคราะห์ระบบต้องการคือข้อเท็จจริง(Fact) ทั้งหมดของระบบงานนั้นๆ ข้อเท็จจริงในที่นี้ไม่ได้หมายถึงเฉพาะข้อมูล (Data) และขั้นตอนการทำงาน (Process) เท่านั้น แต่ได้ครอบคลุมถึงทุกสิ่งที่ประกอบกันขึ้นมาเป็นระบบงานนั้นๆ ทั้งที่เกิดขึ้นก่อน และหลังจากการผ่านขั้นตอนการทำงานต่างๆ เงื่อนไขการดำเนินการทางธุรกิจ (Business Rules หรือ Business Conditions) และสภาพแวดล้อมทางกายภาพต่างๆ (Environment) ที่มีหรืออาจมีผลกระทบในการดำเนินการโดยนักวิเคราะห์ระบบนำข้อเท็จจริงเหล่านั้นมาเป็นข้อมูลประกอบการจัดทำโครงการรวมถึงวิเคราะห์และออกแบบระบบด้วยดังนั้นหน้าที่อีกอย่างหนึ่งที่นักวิเคราะห์ระบบจะต้องดำเนินการ เพื่อให้ได้ข้อเท็จจริง (Fact) ดังกล่าวคือ “การเก็บรวบรวมข้อเท็จจริงและสารสนเทศทั้งหมดของระบบ (Fact-Finding and Information Gathering)”

Fact-Finding เป็นกระบวนการหรือกรรมวิธีในการเก็บรวบรวมข้อเท็จจริงทั้งหมดของระบบงานที่ต้องการพัฒนา ได้แก่ความสัมพันธ์ของข้อมูลในระบบงาน ขั้นตอนการทำงานของระบบงาน ความต้องการของเจ้าของระบบงาน รวมทั้งส่วนประกอบต่างๆ ที่มีความสัมพันธ์ และมีผลกระทบกับระบบงานนั้น โดยทำการค้นคว้าวิจัย สัมภาษณ์บุคคล จัดทำแบบสอบถาม ตัวอย่าง เอกสาร เป็นต้น วิธีการต่างๆ เหล่านี้จัดเป็นทฤษฎีแบบดั้งเดิมที่ยังได้รับความนิยมและนำมาใช้ในการเก็บรวบรวมข้อเท็จจริงของระบบอยู่อย่างต่อเนื่อง โดยรายละเอียดของแต่ละวิธีมีดังต่อไปนี้

6.2.1. ตัวอย่างเอกสาร แบบฟอร์ม และฐานข้อมูลที่ใช้งานในปัจจุบัน (Existing Documents/Sampling)

6.2.2. การค้นหาข้อมูล (Research)

6.2.3. การสังเกตการณ์ (Observation)

6.2.4. การจัดทำแบบสอบถาม (Questionnaire)

6.2.5. การสัมภาษณ์ (Interview)

6.2.1. ตัวอย่างเอกสาร แบบฟอร์ม และฐานข้อมูลที่ใช้งานในปัจจุบัน (Existing Documents/Sampling)

โดยทั่วไปแล้วการศึกษาขั้นตอนการดำเนินการของระบบงานใดๆ นักวิเคราะห์ระบบควรจะเริ่มศึกษาจากสิ่งที่มีอยู่หรือปรากฏอยู่ เช่น ตัวอย่างเอกสารต่างๆ ที่ใช้ในแต่ละขั้นตอนการดำเนินงาน ไม่ควรเริ่มต้นจากการสัมภาษณ์บุคคล เนื่องจากการศึกษาจากตัวอย่างเอกสารนั้นทำให้นักวิเคราะห์ระบบสามารถทำความเข้าใจระบบงานเบื้องต้นได้ก่อน และสามารถทราบได้ว่าจุดใดของระบบงานที่นักวิเคราะห์ยังไม่เข้าใจ เพื่อนำข้อมูลเหล่านี้ไปใช้ในการเตรียมข้อมูลและคำถามต่างๆ สำหรับการเข้าสัมภาษณ์บุคคล จัดทำแบบสอบถาม หรือใช้ในการค้นคว้ารายละเอียดจากแหล่งข้อมูลอื่นๆ ได้

ในการรวบรวมข้อเท็จจริงจากเอกสารที่มีอยู่แล้ว อาจทำได้ 2 วิธี ดังนี้

1. การรวบรวมข้อเท็จจริงจากเอกสารที่มีอยู่ (Existing Documents) เอกสารที่นักวิเคราะห์ระบบควรจะทำการศึกษาเป็นอันดับแรกคือ แผนภูมิองค์กร (Organization Chart) เนื่องจากแผนภูมิองค์กรช่วยให้นักวิเคราะห์ระบบเข้าใจถึงลักษณะโครงสร้างขององค์นั้นๆ ว่าบุคคลใดมีความสำคัญระดับใด ลำดับการบังคับบัญชาอยู่ในลักษณะใด ทำให้นักวิเคราะห์ระบบสามารถนำไปใช้เป็นแนวทางในการศึกษาขั้นตอนการดำเนินการต่างๆ ได้โดยคร่าวๆ นอกจากนี้

นักวิเคราะห์ระบบควรศึกษาปัญหา และข้อผิดพลาด ในการดำเนินการที่เกิดขึ้นพร้อมกันไปได้ เอกสารอื่นๆ ที่ควรศึกษา ได้แก่

- เอกสารที่ใช้ในองค์กร เช่น บันทึกต่างๆ คำแนะนำ แบบแสดงความคิดเห็นของลูกค้าหรือผู้รับบริการ รายงานประจำเดือน รายงานที่กล่าวถึงปัญหาที่เกิดขึ้นในองค์กร
- เอกสารทางการบัญชี รายงานผลการดำเนินการ การประเมินผลงาน
- คำร้อง หรือบันทึกต่างๆ ในองค์กร หรือจากโครงการระบบสารสนเทศขององค์กร ทั้งในอดีตและปัจจุบัน

นอกจากนี้ยังมีเอกสารประเภทอื่นๆ ที่สามารถอธิบายหรือกล่าวถึงหน้าที่ การดำเนินงาน และเงื่อนไขทางธุรกิจต่างๆ ที่สามารถนำไปศึกษาและใช้ประกอบการออกแบบได้ เอกสารเหล่านี้ได้แก่

- แผนกลยุทธ์การดำเนินธุรกิจ (Strategic Plan) และเอกสารแสดงภารกิจ (Mission Statement) ขององค์กร
- วัตถุประสงค์ในการดำเนินธุรกิจขององค์กร (Objective) หรือหน่วยงานต่างๆ
- นโยบายขององค์กร (Policy)
- แบบฟอร์มต่างๆ ที่มีการกรอกข้อความเรียบร้อยแล้ว สามารถใช้แสดงเป็นตัวอย่างในการดำเนินการจริงได้ (Filled Form)
- คู่มือการใช้งานจอภาพ และรายงานต่างๆ (Screens and Report)

นอกจากนี้นักวิเคราะห์ระบบควรตรวจสอบเอกสารของระบบสารสนเทศที่เคยดำเนินการมาก่อนหน้านี้ด้วย ได้แก่

- ผังงาน (Flow Charts) และแผนภาพ (Diagrams)
- พจนานุกรม หรือ แหล่งเก็บข้อมูลโครงการ (Dictionary or Repository)
- เอกสารการออกแบบ ได้แก่ ข้อมูลนำเข้า ข้อมูลผลลัพธ์ และฐานข้อมูล
- เอกสารการเขียนโปรแกรม
- คู่มือการใช้งานและการอบรม (Operation and Training Manual)

2. การสุ่มตัวอย่าง (Sampling) เนื่องจากในความเป็นจริงแล้วนักวิเคราะห์ระบบไม่สามารถศึกษาและรวบรวมเอกสารทั้งหมดในองค์กรได้ ดังนั้นนักวิเคราะห์ระบบสามารถใช้เทคนิคในการสุ่มตัวอย่างเอกสาร 1 กลุ่มตัวอย่างจากที่มีทั้งหมดในองค์กร โดยมีขนาด (จำนวน) ของตัวอย่างมากพอที่จะทำให้ทราบถึงขั้นตอนและเงื่อนไขในการดำเนินงานได้

   การสุ่มตัวอย่าง หมายถึง กระบวนการรวบรวมข้อมูลโดยการเลือกตัวอย่างเอกสาร แบบฟอร์ม หรือแหล่งข้อมูลอื่นๆ เพียงบางส่วนจากทั้งหมดที่มีในองค์กร

6.2.2. **การค้นหาข้อมูล** (Research)

นักวิเคราะห์ระบบสามารถค้นคว้าข้อมูลของหน่วยงานหรือองค์กรอื่นที่ประสบปัญหาการดำเนินงานเช่นเดียวกันหรือมีความต้องการตรงกันได้ เพื่อให้ทราบถึงแนวทางการแก้ไขและนำมาวิเคราะห์ เปรียบเทียบ กับปัญหา หรือความต้องการขององค์กรเองว่าสามารถนำมาประยุกต์ใช้ได้หรือไม่ เช่น อาจจะค้นคว้าได้จาก Web Site ของหน่วยงานหรือองค์กรเหล่านั้นผ่านเครือข่ายอินเตอร์เน็ต หรือจากนิตยสาร หนังสือพิมพ์ธุรกิจต่างๆ เป็นต้น

นอกจากนี้นักวิเคราะห์ระบบ ยังสามารถค้นคว้าข้อมูลของเทคโนโลยีคอมพิวเตอร์ ซอฟต์แวร์สำเร็จรูปสำหรับงานธุรกิจต่างๆ ได้จากเครือข่ายอินเตอร์เน็ต เพื่อนำข้อมูลเหล่านั้นมาประยุกต์ใช้ให้เป็นประโยชน์ต่อการพัฒนาระบบต่อไป

6.2.3. **การสังเกตการณ์** (Observation)

การสังเกตการณ์เป็นหนึ่งในเทคนิคของ Fact-Finding ที่ทั้งนักวิเคราะห์ระบบ และเจ้าหน้าที่ผู้ที่มีส่วนเกี่ยวข้องในการดำเนินการหรือกิจกรรมใดๆ ของระบบได้ทำงานร่วมกัน เพื่อที่จะเรียนรู้ขั้นตอนและกระบวนการดำเนินการต่างๆ อย่างใกล้ชิด ซึ่งเทคนิคนี้มักใช้ในกรณีที่ข้อมูลที่นักวิเคราะห์ระบบรวบรวมมา ยังไม่ละเอียดมากนัก ถึงแม้ว่าเทคนิคนี้จะให้ข้อมูลที่ครบถ้วนและถูกต้อง แต่นักวิเคราะห์ระบบควรคำนึงถึงผลดีและผลเสียจากการนำเทคนิคนี้ไปใช้ด้วย ดังนี้

ข้อดี	ข้อเสีย
1. ข้อมูลที่รวบรวมได้มีความน่าเชื่อถือค่อนข้างสูง 2. นักวิเคราะห์ระบบสามารถเห็นขั้นตอนการดำเนินการที่เกิดขึ้นจริง 3. เมื่อเทียบกับเทคนิคอื่นๆ การสังเกตการณ์เป็นเทคนิคที่ใช้เงินทุนต่ำ 4. นักวิเคราะห์ระบบสามารถวัดผลการดำเนินการของเทคนิคนี้ได้	1. ในการสังเกตการดำเนินการของเจ้าหน้าที่นั้น อาจมีผลกระทบต่อการดำเนินการของเจ้าหน้าที่ โดยเจ้าหน้าที่อาจรู้สึกอึดอัด และดำเนินการผิดพลาดได้ 2. การใช้เทคนิคนี้ในระบบงานอาจต้องใช้ระยะเวลาค่อนข้างมาก ไม่เช่นนั้นอาจได้ข้อมูลที่ไม่ครบถ้วนทุกเงื่อนไขการดำเนินการ 3. การดำเนินการบางงานอาจมีลักษณะงานที่ไม่สะดวก หรือช่วงเวลาการดำเนินการไม่ตรงกับการทำงานของนักวิเคราะห์ระบบ 4. การดำเนินการบางขั้นตอนอาจมีเงื่อนไขบางประการที่มีโอกาสเกิดขึ้นน้อยมาก 5. เจ้าหน้าที่อาจปฏิบัติงานไม่เต็มที่เมื่อทราบว่านักวิเคราะห์ระบบจะเข้ามาสังเกตการณ์ โดยปฏิบัติงานเท่าที่ต้องการให้นักวิเคราะห์ระบบทราบเท่านั้น

ข้อแนะนำ

ในการสังเกตการณ์ นักวิเคราะห์ระบบควรศึกษาการปฏิบัติงานโดยทั่วไปเป็นอันดับแรก หลังจากนั้นจึงมุ่งไปที่เงื่อนไขเฉพาะต่างๆ และศึกษาการปฏิบัติงานในช่วงที่มีปริมาณงานสูงสุด เพื่อให้สามารถรวบรวมข้อมูลและวิเคราะห์ผลกระทบต่างๆ ที่เกิดขึ้นได้ นอกจากนี้นักวิเคราะห์ระบบควรศึกษาข้อมูลจากเอกสารและแบบฟอร์มต่างๆ เสียก่อน ทั้งนี้เพื่อให้เข้าใจการทำงานเบื้องต้นและสามารถวางแผนและเตรียมการสังเกตุการณ์ได้ล่วงหน้า โดยใช้เทคนิคที่เรียกว่า “ การสุ่มตัวอย่างการดำเนินงาน (Work Sampling)”

Work Sampling เป็นเทคนิคการหาข้อมูลการดำเนินงาน โดยการสุ่มการดำเนินงานในช่วงเวลาใดๆ เพื่อสังเกตการปฏิบัติงานของเจ้าหน้าที่

การสุ่มตัวอย่างการดำเนินงานนี้ทำให้เจ้าหน้าที่ไม่รู้สึกกดดันขณะทำงาน เนื่องจากไม่ถูกจับตามองตลอดเวลา ในการสุ่มตัวอย่างการดำเนินงานนั้นนักวิเคราะห์ระบบต้องกำหนดว่าขั้นตอนใดบ้างที่ต้องการศึกษา ระยะเวลาเท่าใด ปริมาณเท่าใด เช่นเดียวกับกำหนดการสุ่มเอกสารตัวอย่าง นักวิเคราะห์ระบบอาจทำการสุ่มตัวอย่างการดำเนินงานในช่วงเวลาที่แตกต่างกันไปในแต่ละวันดังนี้

1. กำหนดตัวบุคคล สิ่งที่ต้องการ สถานที่ เวลา เหตุผล และวิธีที่ใช้ในการสังเกตการณ์การดำเนินงาน
2. ควรได้รับการยินยอมจากผู้จัดการ หรือหัวหน้าระบบงานนั้น
3. จดบันทึกข้อมูลระหว่างการศึกษาการดำเนินงาน

4. ทบทวนข้อมูลที่บันทึกกับเจ้าหน้าที่ผู้ดำเนินงาน
5. ไม่ขัดจังการดำเนินงานใดๆ ของเจ้าหน้าที่
6. มุ่งสนใจการดำเนินงานหลัก
7. ไม่สรุปข้อสันนิษฐานใดๆด้วยตนเอง
8. กำหนดวัตถุประสงค์ในการสังเกตการณ์ดำเนินงานแต่ละขั้นตอน

6.2.4. การจัดทำแบบสอบถาม (Questionnaire)

แบบสอบถาม คือ เอกสารที่สร้างขึ้นโดยมีวัตถุประสงค์เพื่อรวบรวมข้อเท็จจริงและสารสนเทศของระบบจากผู้ตอบแบบสอบถาม ซึ่งจะทำให้นักวิเคราะห์ระบบสามารถวิเคราะห์หาความต้องการในระบบใหม่ของผู้ใช้ได้

แบบสอบถามชุดหนึ่งๆ อาจมีปริมาณเอกสารจำนวนมาก เนื่องจากวัตถุประสงค์ในการทำแบบสอบถามนี้เพื่อให้นักวิเคราะห์ระบบสามารถรวบรวมข้อเท็จจริงให้ได้มากที่สุด แบบสอบถามอาจมีความหลากหลายและประกอบด้วยข้อคิดเห็นต่างๆ นักวิเคราะห์ระบบมักจะหลีกเลี่ยงการใช้แบบสอบถาม เนื่องจากเห็นว่าข้อมูลที่ได้รับมีความน่าเชื่อถือน้อยหรือแทบไม่มีเลยและมักได้ข้อมูลที่ไม่ค่อยมีประโยชน์มากนัก

การเลือกกลุ่มผู้ตอบแบบสอบถาม

บางครั้งมีคนจำนวนมากเกินกว่าจำนวนที่นักวิเคราะห์สามารถที่จะจัดการสำรวจได้ ดังนั้นจึงต้องตัดสินใจว่าจะส่งแบบสอบถามใดไปให้กับกลุ่มคนกลุ่มใด กลุ่มใดก็ตามที่เลือกจะต้องเป็นตัวแทนของผู้ใช้ทั้งหมด โดยปกติแล้วนักวิเคราะห์สามารถเลือกกลุ่มตัวอย่างที่เป็นตัวแทนของผู้ใช้ได้โดยวิธีการใดวิธีการหนึ่งหรือโดยการผสมผสานระหว่างวิธีการต่างๆ 4 วิธีดังนี้

1. เลือกตามความสะดวก (Convenient to Sample) ตัวอย่างเหล่านี้อาจได้แก่ คนที่ทำงาน ณ ที่ตั้งสำนักงาน คนที่ยินดีจะให้ข้อมูลเพื่อการสำรวจ หรือคนที่ถูกกระตุ้นให้อยากแสดงความคิดเห็นมากที่สุด
2. เลือกโดยวิธีสุ่ม (Random) ถ้านักวิเคราะห์ได้รายชื่อของผู้ใช้ระบบปัจจุบันทุกๆคน การเลือกโดยการสุ่มทำได้ง่ายๆ โดยการเลือกคนที่ n จากรายชื่อนั้น หรืออาจเลือกโดยการข้ามชื่อคนที่อยู่ในรายชื่อนั้นโดยใช้ตัวเลขจากตารางตัวเลขสุ่มก็ได้
3. เลือกตามวัตถุประสงค์เฉพาะที่กำหนด (Purposeful Sample) โดยวิธีการนี้นักวิเคราะห์อาจเลือกเฉพาะคนที่มีคุณสมบัติตรงตามหลักเกณฑ์ที่กำหนด เช่น เลือกผู้ใช้ที่เคยใช้ระบบปัจจุบันมานานกว่า 2 ปี หรือเลือกผู้ใช้ระบบบ่อยที่สุด เป็นต้น
4. เลือกจากกลุ่มต่างๆ ที่จัดแบ่งไว้ (Stratified Sample) ในกรณีนี้ จะแบ่งคนทั้งหลายที่อยากจะเลือกเป็นตัวอย่างออกเป็นหลายๆ กลุ่ม (เช่น กลุ่มผู้ใช้ ผู้บริหาร และผู้ใช้ในหน่วยงานธุรกิจต่างประเทศเป็นต้น) จากนั้นจึงใช้วิธีการสุ่มเลือกจากแต่ละกลุ่ม

ประเภทแบบสอบถาม

1. Free Format เป็นแบบสอบถามที่ให้อิสระในการตอบ โดยผู้ตอบแบบสอบถามเขียนคำตอบเอง แบบสอบถามประเภทนี้ค่อนข้างจะทำการประมวลผลได้ยาก เนื่องจากผู้ตอบอาจตอบไม่ตรงตามวัตถุประสงค์ ดังนั้นนักวิเคราะห์ระบบควรใช้คำที่เข้าใจง่าย และสามารถตอบโดยใช้คำเพียง 2-3 คำ หรือเป็นประโยคสั้น แสดงตัวอย่างเช่น

1. คุณต้องการเพิ่มเติมรายละเอียดในแบบฟอร์มใบสมัครหรือไม่ หากต้องการ คุณจะเพิ่มเติมส่วนใด

.....

2. ปัญหาที่เกิดขึ้นในการค้นหาข้อมูลพนักงานคืออะไร

……...

3. รายงานที่จัดทำจากข้อมูลพนักงาน ส่งไปยังฝ่ายใดบ้าง

……

3. Fixed Format คำถามในแบบสอบถามประเภทนี้ต้องการคำตอบที่เจาะจงลงไป โดยมีคำตอบให้ผู้ตอบเลือก แบบสอบถามประเภทนี้ประมวลผลได้ง่าย แต่ในทางกลับกันผู้ตอบไม่สามารถเสนอข้อมูลหรือข้อคิดเห็นใดๆ เพิ่มเติม นอกเหนือจากคำตอบที่เตรียมไว้ แบบสอบถามประเภทนี้สามารถจำแนกย่อยได้ 3 ประเภท ได้แก่

3.1. Multiple Choices : มีคำตอบหลายข้อให้เลือกตอบ และผู้ตอบสามารถเลือกคำตอบได้มากกว่า 1 ข้อ หรือมีตัวเลือกให้ผู้ตอบสามารถเติมข้อความได้บ้างเล็กน้อย โดยคำถาม 1 ข้อ ผู้ตอบสามารถเลือกคำตอบได้มากกว่า 1 คำตอบ แสดงดังตัวอย่างเช่น

1. ท่านสังกัดในส่วนงานใด

☐ ฝ่ายบัญชีและการเงิน ☐ ฝ่ายจัดซื้อ ☐ ฝ่ายการตลาด ☐ ฝ่ายบุคคล ☐ อื่นๆ โปรดระบุ…….

2. ในการดำเนินงานของส่วนงานท่านต้องจัดทำรายงานใดบ้าง

☐ รายงานเงินเดือน ☐ รายงานภาษี ☐ รายงานการสั่งซื้อ ☐ รายงานยอดขาย

☐ รายงานการรับสมัครงาน ☐ รายงานเวลาทำงาน ☐ รายงานข้อมูลพนักงาน

☐ อื่นๆ โปรดระบุ……..

3.2. Rating Question : มีคำตอบเป็นตัวเลือกเพื่อให้แสดงความคิดเห็น โดยการกำหนดระดับความคิดเห็นของผู้ตอบในแต่ละข้อว่ามากน้อยเพียงใด แสดงตัวอย่างเช่น

1. คุณเห็นด้วยหรือไม่ กับการนำระบบคอมพิวเตอร์เข้ามาช่วยงานด้านการคำนวณภาษี

☐ ไม่เห็นด้วย ☐ เห็นด้วย ☐ เห็นด้วยอย่างมาก ☐ ไม่มีความเห็น

2. การสืบค้นข้อมูลจากฐานข้อมูลเกิดความล่าช้ามากน้อยเพียงใด

☐ ช้ามากที่สุด ☐ ช้ามาก ☐ ช้า ☐ ไม่ช้า

3. โปรแกรมที่ใช้ในฝ่ายจัดซื้อ Error ในระดับใด

☐ ไม่เคย ☐ มีบ้าง ☐ บ่อย ☐ บ่อยมาก

3.3. Ranking Question: เป็นการจัดลำดับความสำคัญของคำตอบต่างๆ ในแต่ละคำถาม แสดงตัวอย่างเช่น

กรุณาเรียงลำดับความสำคัญจากมากที่สุดไปหาน้อยที่สุด (1-4) ของรายการข้อมูลที่ดำเนินการมากที่สุดต่อวัน

……รายการสั่งซื้อสินค้าจากลูกค้า

……รายการจัดซื้อสินค้า

……รายการรับสมัครพนักงาน

……รายการยกเลิกรายการสั่งซื้อจากลูกค้า

6.2.5. การสัมภาษณ์ (Interview)

การสัมภาษณ์เป็นเทคนิค Fact-Finding ที่นักวิเคราะห์ระบบใช้รวบรวมข้อมูลจากบุคคลต่างๆ ที่เกี่ยวข้องในการดำเนินงานของระบบ แบบตัวตอตัว จากการสัมภาษณ์ นักวิเคราะห์ระบบจได้รับข้อเท็จจริง สามารถตรวสอบ

ข้อเท็จจริงได้ มีความเข้าใจกันมากขึ้น และรับทราบความต้องการที่แท้จริงของผู้ใช้งาน รวมทั้งความคิดเห็นต่างๆ โดยการสัมภาษณ์แต่ละครั้ง นักวิเคราะห์ระบบมีบทบาทเป็นผู้สัมภาษณ์ (Interviewer) มีหน้าที่ดำเนินการสัมภาษณ์ ถามคำถาม และชักจูงผู้ที่มีบทบาทเป็นผู้ให้สัมภาษณ์ (Interviewee) ตอบคำถามนั้นๆ ดังนั้นในการสัมภาษณ์บุคคลใดๆ ผู้สัมภาษณ์ต้องสามารถควบคุมสถานการณ์ต่างๆ ที่อาจเกิดขึ้นระหว่างการสัมภาษณ์ได้ และมีมนุษย์สัมพันธ์ดี สามารถสื่อสารกับบุคคลต่างๆ ได้ทุกประเภทและทุกสถานการณ์

ประเภทของการสัมภาษณ์

1. การสัมภาษณ์แบบไม่มีโครงสร้าง (Unstructured Interview) เป็นลักษณะการสัมภาษณ์หัวข้อทั่วๆ ไปเกี่ยวกับองค์กร ไม่เจาะจงหัวข้อของการสัมภาษณ์ การสัมภาษณ์ประเภทนี้ไม่เหมาะกับการวิเคราะห์และออกแบบระบบสารสนเทศ
2. การสัมภาษณ์แบบมีโครงสร้าง (Structured Interview) ผู้สัมภาษณ์จะต้องเตรียมข้อมูล และคำถามเพื่อสอบถามข้อเท็จจริงต่างๆ จากผู้ให้สัมภาษณ์ โดยสามารถสอบถามข้อสงสัยต่างๆ เพิ่มเติมได้ เพื่อตรวจสอบความเข้าใจของผู้สัมภาษณ์ว่าถูกต้องหรือไม่

เทคนิคการสัมภาษณ์

การสัมภาษณ์จะประสบผลสำเร็จหรือไม่นั้น นอกจากจะขึ้นอยู่กับความสามารถเฉพาะตัวของผู้สัมภาษณ์แล้วยังขึ้นกับองค์ประกอบอื่นๆ ดังนี้

1. การเลือกบุคคลผู้ให้สัมภาษณ์ ผู้ให้สัมภาษณ์ควรเป็นผู้ที่ดำเนินงานในขั้นตอนที่นักวิเคราะห์ระบบต้องการศึกษา นักวิเคราะห์ระบบสามารถเลือกบุคคลนี้โดยการศึกษาจากแผนภูมิโครงสร้างขององค์กร (Organization Chart) เพื่อให้ทราบถึงหน้าที่ความรับผิดชอบของบุคคลต่างๆ ในองค์กร และควรศึกษาทัศนคติต่างๆ ของผู้ให้สัมภาษณ์ล่วงหน้า
2. การเตรียมการสัมภาษณ์ การเตรียมการเป็นกุญแจสำคัญที่จะชี้ว่าการสัมภาษณ์นั้นจะบรรลุวัตถุประสงค์หรือไม่ เนื่องจากผู้ให้สัมภาษณ์สามารถทราบว่า ได้มีการเตรียมตัวมาหรือไม่ ในกรณีที่ไม่มีการเตรียมตัวล่วงหน้านั้น ผู้ให้สัมภาษณ์อาจรู้สึกว่าเสียเวลาที่มีค่าโดยเปล่าประโยชน์ เนื่องจากผู้สัมภาษณ์ขาดความสนใจ และขาดความพร้อมหลายๆ ด้าน ในการสัมภาษณ์ ผู้สัมภาษณ์ควรจัดทำคู่มือการสัมภาษณ์ (Interview Guide) ไว้ด้วย

Interview Guide : เป็นคู่มือประกอบการสัมภาษณ์โดยบันทึกรายการคำถามที่ต้องการสัมภาษณ์ หรืออาจประกอบด้วยคำถามที่ต้องการตรวจสอบ และติดตามข้อมูลเพิ่มเติม แสดงตัวอย่างคู่มือประกอบการสัมภาษณ์

คำถามที่ใช้ในการสัมภาษณ์ควรมีลักษณะดังนี้

- กระชับ และเข้าใจง่าย
- ไม่นำเสนอความคิดเห็นส่วนตัวแฝงในคำถาม
- หลีกเลี่ยงคำถามที่ซับซ้อน หรือยาวเกินไป
- หลีกเลี่ยงการใช้ถ้อยคำในลักษณะคุกคาม หรือข่มขู่

3. การดำเนินการสัมภาษณ์ ในการสัมภาษณ์สามารถจำแนกออกเป็น 3 ขั้นตอนดังนี้
    1. เปิดสัมภาษณ์ (Interview Opening) เป็นการชักจูง โน้มน้าว กระตุ้นผู้ถูกสัมภาษณ์ให้มีความกระตือรือร้นในการให้ความร่วมมือ และครรบอกวัตถุประสงค์ ระยะเวลาการสัมภาษณ์ รวมทั้งอธิบายถึงวิธีการรวบรวมข้อมูลว่าเป็นเช่นไร และข้อมูลที่ได้รับมานั้นจะนำไปใช้อย่างไร

2. สัมภาษณ์ (Interview Body) เป็นช่วงที่ใช้เวลามากที่สุด ในขั้นตอนนี้ผู้สัมภาษณ์จะได้รับคำตอบตามรายการคำถามที่เตรียมไว้ล่วงหน้า และควรบันทึกข้อมูลทั้งที่เป็นภาษาพูด และภาษากายของผู้ให้สัมภาษณ์ โดยสามารถปรับเปลี่ยน หรือข้ามคำถามได้ตามความเหมาะสมของสถานการณ์
3. ปิดสัมภาษณ์ (Interview Conclusion) ผู้สัมภาษณ์ควรแสดงความขอบคุณต่อผู้ที่ให้สัมภาษณ์ ที่ได้เสียสละเวลาอันมีค่า เพื่อรักษาความสัมพันธ์อันดี สร้างความพึงพอใจ และไว้วางใจ

4. ติดตามผลการสัมภาษณ์ เพื่อการรักษาสัมพันธภาพอันดี สร้างความเชื่อมั่น และความไว้วางใน ดังนั้นผู้สัมภาษณ์ควรส่งผลสรุปที่ได้จากการสัมภาษณ์ กลับไปยังผู้ให้สัมภาษณ์ เพื่อเปิดโอกาสให้ผู้สัมภาษณ์ได้ทราบว่าผู้สัมภาษณ์มีความเข้าใจถูกต้องหรือไม่ และผู้ให้สัมภาษณ์สามารถให้ข้อมูลเพิ่มเติมกลับมาได้เช่นกัน

ข้อแนะนำ

สิ่งที่ควรทำ	สิ่งที่ควรหลีกเลี่ยง
1. มีความสุภาและมีมารยาท 2. ตั้งใจฟัง 3. ควบคุมสถานการณ์ 4. ตรวจสอบข้อมูล 5. มีความอดทน 6. สังเกตกิริยา ท่าทาง รวมทั้งภาษากาย(Body Language) 7. ไม่กังวล	8. คำถามที่ไม่จำเป็น หรือไม่ควรถาม 9. การสันนิษฐานเองว่าคำตอบที่ได้รับสมบูรณ์ 10. การชี้นำทั้งด้วยคำพูดและภาษากาย 11. การใช้ศัพท์เฉพาะ หรือคำที่เข้าใจยาก 12. การแสดงความคิดเห็น 13. การพูดมากเกินความจำเป็น แทนที่จะรับฟังคำตอบ 14. การสันนิษฐานเกี่ยวกับหัวข้อเรื่องและผู้ให้สัมภาษณ์ 15. การใช้เทปบันทึกเสียง

6.3. ทฤษฎีแนวใหม่ในการกำหนดความต้องการของระบบ

จากหัวข้อที่ผ่านมา จะเห็นว่าเทคนิคในการเก็บรวบรวมข้อเท็จจริงและสารสนเทศของระบบนั้นมีอยู่หลายวิธีด้วยกัน ทั้งนี้ขึ้นอยู่กับนักวิเคราะห์ระบบว่าจะเลือกวิธีการใดที่จะสามารถเก็บข้อมูลได้อย่างเพียงพอและสมบูรณ์ ทั้งยังเหมาะสมกับขนาดและสถานะการณ์ขององค์กรอีกด้วย ซึ่งนอกจากวิธีการที่เป็นทฤษฎีดั้งเดิมแล้ว ยังมีทฤษฎีใหม่ที่เป็นทางเลือกให้กับนักวิเคราะห์เลือกใช้ เพื่อเพิ่มประสิทธิภาพและความรวดเร็วในการเก็บข้อมูล เทคนิคแนวใหม่นี้คือ

1. Joint Application Design (JAD)
2. Rapid Application Development (RAD)

6.3.1. Joint Application Design (JAD)

JAD คือ กระบวนการในการจัดการ และเพิ่มความสามารถในการปฏิบัติงานร่วมกันของเจ้าของระบบ ผู้ใช้ระบบงาน นักวิเคราะห์ระบบ ผู้ออกแบบระบบ และผู้สร้างระบบ เพื่อร่วมกันกำหนดขอบเขต วิเคราะห์ และออกแบบระบบ กล่าวคือเป็นการประชุมงานร่วมกันของผู้มีส่วนเกี่ยวข้องกับการพัฒนาระบบนั่นเอง

ตัวอย่าง

การเก็บรวบรวมข้อเท็จจริงของนักวิเคราะห์ระบบ โดยเข้าร่วมสังเกตการณ์ในการประชุมของผู้ที่เกี่ยวข้องในการพัฒนาระบบขององค์กร ทั้งนี้เพื่อช่วยลดเวลาและค่าใช้จ่ายในขั้นตอนการเก็บรวบรวมข้อมูล

วัตถุประสงค์ของการนำเทคนิค JAD เข้ามาช่วยในขั้นตอนการกำหนดความต้องการของระยะการวิเคราะห์ระบบ เพื่อช่วยให้การเก็บรวบรวมข้อเท็จจริงและข้อมูลต่างๆ จากผู้ที่เกี่ยวข้องหลายฝ่ายสามารถดำเนินการได้ในคราวเดียวกันอย่างพร้อมเพียง

โดยทั่วไปผู้เข้าร่วมการดำเนินการเก็บรวบรวมข้อมูลด้วยเทคนิค JAD มีดังนี้

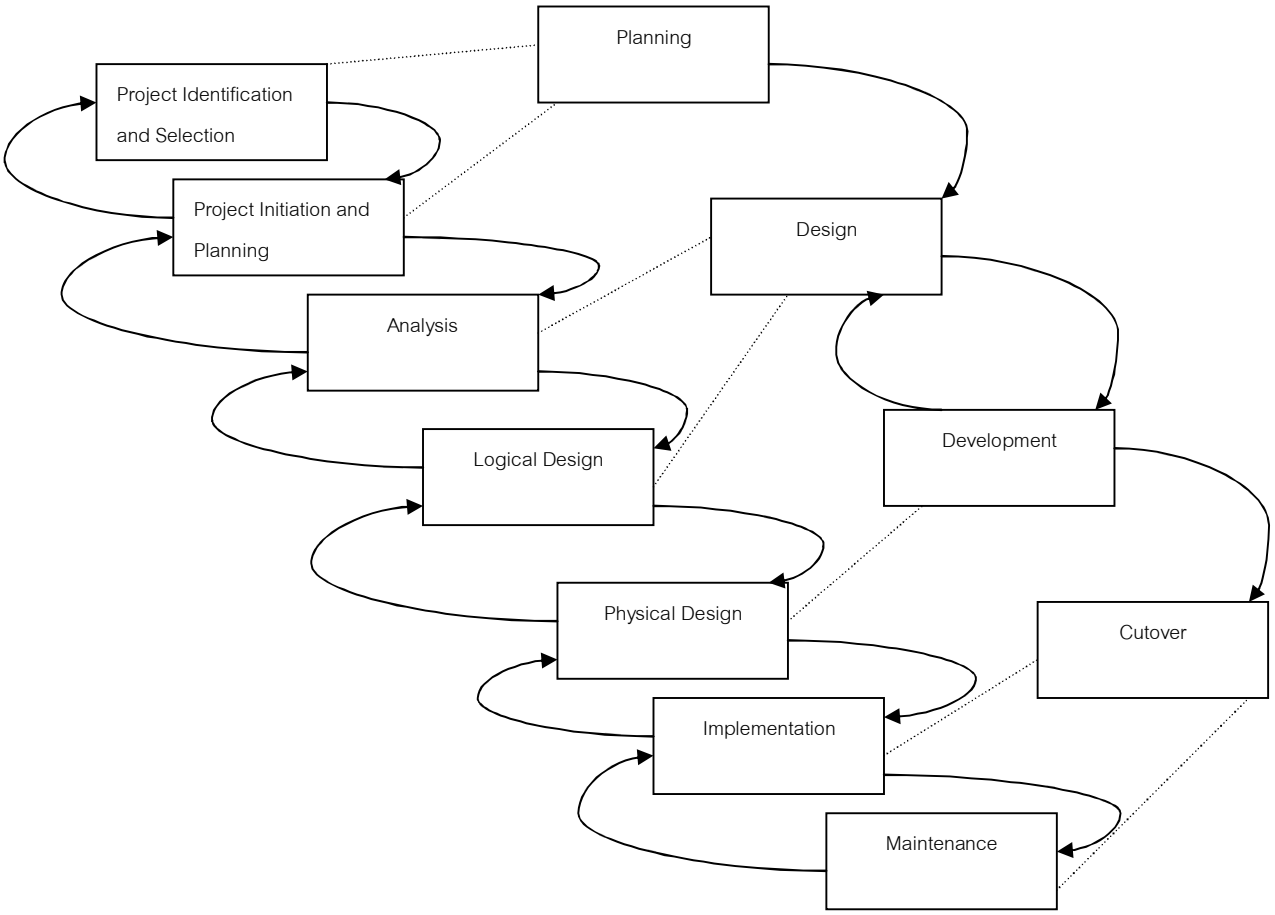
1. JAD Session Leader เป็นผู้ดำเนินการประชุม ต้องเป็นผู้ที่ผ่านการอบรมการทำงานเป็นกลุ่มและเป็นผู้ที่คอยอำนวยความสะดวกระหว่างการประชุม ทั้งยังเป็นผู้จัดตั้งระเบียบวาระการประชุมรวมถึงการควบคุมการประชุมให้อยู่ในวาระ เพื่อให้ได้ข้อมูลตรงจุด
2. Users คือ ผู้ใช้ระบบ เนื่องจากเป็นผู้ที่ใช้ระบบเป็นประจำทุกวัน ดังนั้นจะมีความเข้าใจถึงการทำงานและปัญหาที่เกิดขึ้นเป็นอย่างดี
3. Manager ผู้บริหารขององค์กร ซึ่งเป็นผู้ที่ใช้ระบบเช่นเดียวกับ User ผู้บริหารจะคอยเตรียมคำถามที่มุ่งเน้นไปที่ระบบที่ต้องการพัฒนาขึ้นมาใหม่ คอยจูงใจและคอยช่วยหาข้อสรุปในแต่ละวาระการประชุม
4. Sponsor คือ ผู้ที่รับผิดชอบเรื่องค่าใช้จ่ายในการพัฒนาระบบนั้นๆ ซึ่งอาจจะเป็นผู้บริหารระดับสูงสุดขององค์กรก็ได้
5. System Analyst นักวิเคราะห์ระบบและทีมของนักวิเคราะห์ระบบ ทำหน้าที่เก็บข้อมูลจากการประชุมในแต่ละครั้ง
6. Scribe คือ ผู้ที่ทำหน้าที่จดสรุปรายละเอียดระหว่างการประชุม โดยทั่วไปอาจใช้เครื่องคอมพิวเตอร์แบบพกพาช่วยในการบันทึก
7. IS Staff ทีมของหน่วยบริการสารสนเทศองค์กรเช่น นักวิเคราะห์ระบบ โปรแกรมเมอร์ และผู้เชี่ยวชาญด้านฐานข้อมูล บุคคลเหล่านี้สามารถเสนอความคิดเห็นด้านเทคโนโลยีได้

การเก็บรวบรวมข้อมูลด้วยเทคนิค JAD สามารถดำเนินการได้ด้วยการจัดให้เป็นห้องประชุม ซึ่งลักษณะแตกต่างกันไปขึ้นอยู่แต่ละองค์กร ในระหว่างการประชุม JAD นักวิเคราะห์ระบบสามารถใช้ Case Tools และ Prototype ช่วยสนับสนุนการดำเนินการได้ ซึ่งจะทำให้การประชุมดำเนินการได้รวดเร็วยิ่งขึ้น

6.3.2. Rapid Application Development (RAD)

RAD เป็นวิธีการพัฒนาระบบ (Methodology) วิธีการหนึ่งที่รวบรวมเทคนิค (Techniques) เครื่องมือ (Tools) และเทคโนโลยี (Technologies) เพื่อผสมผสานและประยุกต์ใช้ในการสนับสนุนการพัฒนาระบบให้สำเร็จลุล่วงได้โดยใช้เวลาน้อยที่สุดทั้งนี้ขึ้นอยู่กับความพร้อมขององค์กรในขณะนั้น ไม่ว่าจะเป็นค่าใช้จ่าย บุคลากร รวมทั้งความต้องการที่แน่นอนของผู้ใช้ระบบ จากแนวความคิดของวิธีการแบบ RAD ทำให้มีการผสมผสานและประยุกต์ใช้เทคนิค เครื่องมือ และเทคโนโลยีเข้าด้วยกันเพื่อพัฒนาระบบ โดยอาจจะมีการแบ่งแยกขั้นตอนในวงจรการพัฒนาระบบให้น้อยลง เลือกใช้เทคนิคการเก็บรวบรวมข้อมูลด้วย JAD และเลือกใช้ตัวต้นแบบ (Prototype) ในการออกแบบ เป็นต้น ทำให้การพัฒนาระบบสามารถดำเนินการได้อย่างรวดเร็ว ดังนั้น RAD จึงเหมาะกับองค์กรที่มีความพร้อมในการพัฒนา เช่น ความพร้อมทางด้านการเงิน บุคลากร ประกอบกับผู้ใช้ความต้องการแน่นอนไม่เปลี่ยนแปลง และต้องการระบบใหม่โดยใช้เวลาพัฒนาไม่นาน

การรวมขั้นตอนการทำงานของวงจรการพัฒนาระบบ (SDLC) เพื่อให้เป็นวงจรการพัฒนาแบบ RAD สามารถแสดงได้ดังนี้



จากรูปจะสังเกตว่ามีการรวบรวมขั้นตอน Project Identification and Selection เข้ากับ Project Initiation and Planning ให้เหลือเพียงขั้นตอน Planning เพียงขั้นตอนเดียว ส่วน Analysis และ Logical Design ถูกรวมเข้าเป็นขั้นตอน Design เพียงขั้นตอนเดียว และ Implementation กับ Maintenance ถูกรวมให้เป็น Cutover เพียงขั้นตอนเดียว ดังนั้นการใช้เทคนิค RAD จะช่วยให้การพัฒนาระบบดำเนินการได้อย่างรวดเร็วและมีประสิทธิภาพ

บทที่ 7

แบบจำลองขั้นตอนการทำงานของระบบ (Process Modelling)

ในบทที่แล้ว ผู้อ่านได้รู้จักกับทฤษฎีในการเก็บรวบรวมข้อเท็จจริงและสารสนเทศที่จำเป็นต่อการกำหนดความต้องการของระบบ สิ่งที่ได้คือข้อเท็จจริงและสารสนเทศของระบบเดิมและความต้องการของระบบใหม่ (เพื่อแก้ปัญหาที่เกิดจากระบบเดิม) เช่น ขั้นตอนการทำงานของระบบ ข้อมูลที่นำเข้าสู่ระบบ ข้อมูลและรายงานที่ได้จากการประมวลผลแต่ละขั้นตอน บุคคลที่เกี่ยวข้องกับระบบ แหล่งจัดเก็บข้อมูล เป็นต้น ซึ่งข้อเท็จจริงเหล่านี้มีรายละเอียดจำนวนมาก และซับซ้อน การวิเคราะห์ระบบอาจจะดำเนินไปด้วยความลำบาก ดังนั้นจึงต้อง “จำลองข้อเท็จจริง” เหล่านั้นให้อยู่ในรูปแบบที่สามารถเข้าใจง่าย โดยอาจใช้ แผนภาพ (Diagrams) ชนิดต่างๆ ในการจำลอง ซึ่งจะช่วยให้ผู้ใช้และเจ้าของระบบสามารถทำความเข้าใจได้ง่ายขึ้น

ในการจำลองข้อเท็จจริงที่รวบรวมได้ ในที่นี้จะเริ่มต้นด้วยการจำลองแบบขั้นตอนการทำงานของระบบ (Process Modeling) ซึ่งเป็นเนื้อหาในบทเรียนนี้ โดยจะนำเสนอรายละเอียดของการจำลองขั้นตอนการทำงานของระบบด้วย “แผนภาพกระแสข้อมูล (Data Flow Diagram : DFD)” จากแผนภาพจะแสดงให้เห็นถึงขั้นตอนการทำงานของระบบ ข้อมูลที่เข้าและออกจากระบบ รวมทั้งข้อมูลที่ไหลอยู่ภายในระบบจากขั้นตอนหนึ่งไปยังอีกขั้นตอนหนึ่ง

7.1. แนะนำแบบจำลองขั้นตอนการทำงานของระบบ

ก่อนที่จะเข้าสู่เนื้อหา Process Modeling ในที่นี้ขอแนะนำให้ได้ทราบถึงประเภทของแบบจำลองที่ใช้ในการพัฒนาระบบสารสนเทศ คือ

1. แบบจำลองเชิงตรรกะ (Logical Model) เป็นแบบจำลองที่อธิบายการดำเนินงานในระบบว่ามีการทำงานและความต้องการใดบ้างโดยไม่คำนึงถึงเทคโนโลยี หรือโปรแกรมภาษาใดๆ ที่นำมาติดตั้งใช้งาน
2. แบบจำลองเชิงกายภาพ (Physical Model) เป็นแบบจำลองที่นอกจากจะอธิบายการดำเนินงานของระบบว่าทำงานอะไรแล้ว ยังอธิบายว่ามีการดำเนินงานอย่างไร นอกจากนี้ยังมีการแสดงถึงประสิทธิภาพของเทคโนโลยีที่เลือกมาติดตั้งใช้งานเพื่อสนองความต้องการ และแสดงข้อจำกัดของเทคโนโลยีนั้นๆ ด้วย

ความแตกต่างระหว่าง Physical กับ Logical นั้นอาจทำให้เกิดความสับสน ดังนั้นจึงขอยกตัวอย่างการเปรียบเทียบระหว่าง Physical และ Logical ดังนี้

สมมติว่าไปซื้อสินค้าที่ห้างสรรพสินค้า เมื่อซื้อสินค้าได้ครบตามที่ต้องการแล้ว เราก็จะไป “ชำระเงิน” การชำระเงินนี้ถือเป็น “Logical” แต่การชำระเงินยังสามารถชำระด้วยเงินสดหรือบัตรเครดิตรายละเอียดตรงนี้เรียกว่า “Physical” นั่นหมายถึง Logical จะไม่เน้นรายละเอียด

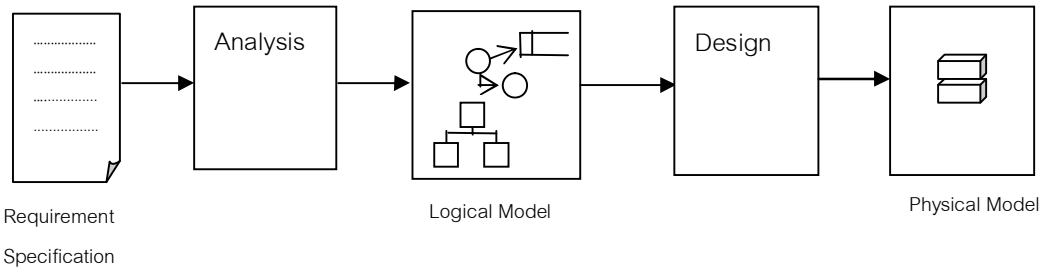
แบบจำลองขั้นตอนการทำงานของระบบ (Process Modeling) คือ เทคนิคที่ใช้ในการรวบรวม บันทึก สร้างโครงสร้างและแสดงทิศทางของข้อมูลในการดำเนินงานขั้นตอนต่างๆ รวมทั้งข้อมูลเชิงตรรกะ (Logical) หลักการ (Policies) และขบวนการ(Procedures) ต่างๆ ของแต่ละขั้นตอน เหตุผลของการจำลองขั้นตอนการทำงานของระบบขึ้น คือต้องการแสดงข้อเท็จจริงในการทำงานและข้อมูลของระบบที่เก็บรวบรวมมาในรูปของข้อความ ให้เป็นแผนภาพเพื่อความสะดวกในการสื่อสารระหว่างนักวิเคราะห์ระบบและโปรแกรมเมอร์หรือผู้ที่เกี่ยวข้องคนอื่นๆ และง่ายต่อความเข้าใจของผู้ใช้และเจ้าของระบบ โดยเครื่องมือที่ใช้ในการจำลองแบบขั้นตอนการทำงานเรียกว่า “แผนภาพกระแสข้อมูล (Data Flow Diagram:DFD)”

แผนภาพกระแสข้อมูล (Data Flow Diagram:DFD) หมายถึง แผนภาพที่แสดงให้เห็นถึงทิศทางการไหลของข้อมูลที่มีอยู่ในระบบ และการดำเนินงานที่เกิดขึ้นในระบบ โดยข้อมูลในแผนภาพทำให้ทราบถึง ข้อมูลมาจากไหน, ข้อมูลไปที่ไหน, ข้อมูลเก็บที่ใด, เกิดเหตุการณ์ใดกับข้อมูลในระหว่างทาง แผนภาพกระแสข้อมูลจะแสดงภาพรวมของระบบ (Overall picture of a system) และรายละเอียดบางอย่าง แต่ในบางครั้งหากต้องการกำหนดรายละเอียดที่สำคัญในระบบ นักวิเคราะห์ระบบอาจจำเป็นต้องใช้เครื่องมืออื่นๆ ช่วย เช่น ข้อความสั้นๆที่เข้าใจ หรืออัลกอริทึม, ตารางการตัดสินใจ (Decision Table), Data Model, Process Description ทั้งนี้ก็ขึ้นอยู่กับความต้องการในรายละเอียด

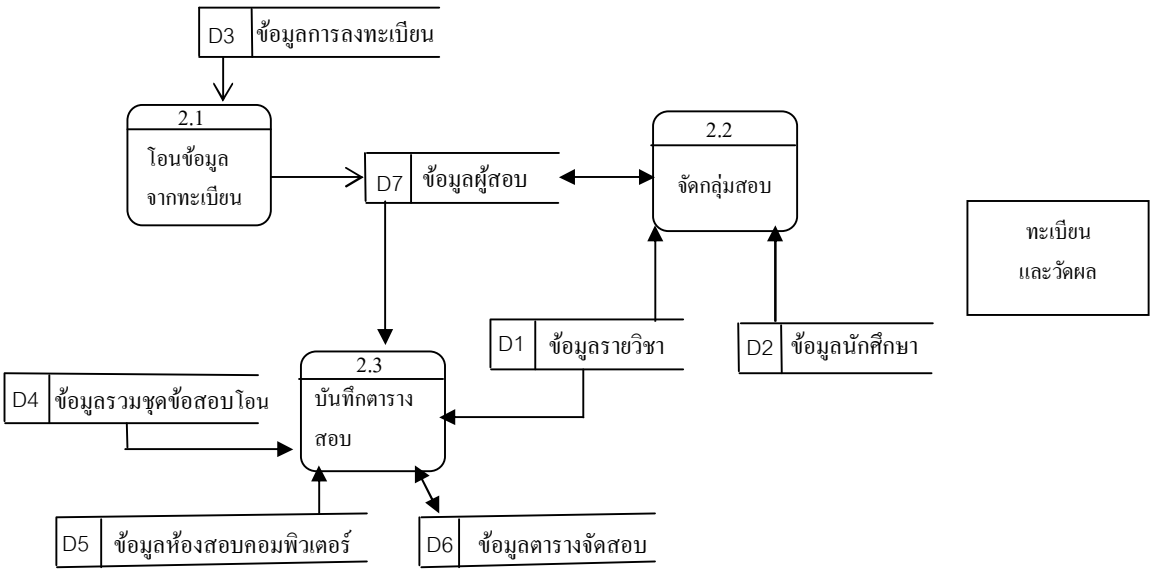
วัตถุประสงค์ของการสร้างแผนภาพกระแสข้อมูลนี้เพื่อ

1. เป็นแผนภาพที่สรุปรวมข้อมูลทั้งหมดที่ได้จากการวิเคราะห์ในลักษณะของรูปแบบที่เป็นโครงสร้าง
2. เป็นข้อตกลงร่วมกันระหว่างนักวิเคราะห์ระบบและผู้ใช้งาน
3. เป็นแผนภาพที่ใช้ในการพัฒนาต่อในขั้นตอนของการออกแบบระบบ
4. เป็นแผนภาพที่ใช้ในการอ้างอิง หรือเพื่อใช้ในการพัฒนาต่อในอนาคต
5. ทราบที่มาที่ไปของข้อมูลที่ไหลไปในกระบวนการต่างๆ (Data and Process)

รูปแสดงขั้นตอนการวิเคราะห์เพื่อไปสู่การออกแบบ

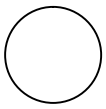
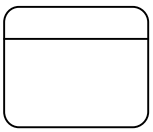
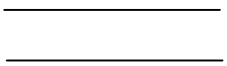
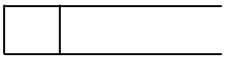
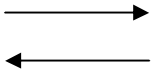
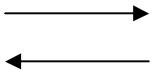


รูปแสดงตังอย่างแผนภาพกระแสข้อมูล



7.2. สัญลักษณ์ที่ใช้ในแผนภาพกระแสข้อมูล

สัญลักษณ์ที่ใช้เป็นมาตรฐานในการแสดงแผนภาพกระแสข้อมูลมีหลายชนิด แต่ในที่นี้จะแสดงให้เห็นเพียง 2 ชนิด ได้แก่ ชุดสัญลักษณ์มาตรฐานที่พัฒนาโดย Gane and Sarson (1979) และชุดสัญลักษณ์มาตรฐานที่พัฒนาโดย DeMarco and Yourdon (DeMarco, 1979); Yourdon and Constantine,1979) โดยมีสัญลักษณ์ดังต่อไปนี้

DeMarco & Yourdon	Gane & Sarson	ความหมาย
		Process : ขั้นตอนการทำงานภายในระบบ
		Data Store : แหล่งข้อมูลสามารถเป็นได้ทั้งไฟล์ข้อมูลและฐานข้อมูล (File or Database)
		External Agent : ปัจจัยหรือสภาพแวดล้อมที่มีผลกระทบต่อระบบ
		Data Store : เส้นทางการไหลของข้อมูล แสดงทิศทางของข้อมูลจากขั้นตอนการทำงานหนึ่งไปยังอีกขั้นตอนหนึ่ง

7.3. แนวคิดของแบบจำลองขั้นตอนการทำงานของระบบ

การสร้างแบบจำลองขั้นตอนการทำงานของระบบโดยใช้แผนภาพกระแสข้อมูล (Data Flow Diagram) มีแนวคิดต่างๆ ดังนี้

7.3.1. ขั้นตอนการทำงานของระบบ (Process)

7.3.2. เส้นทางการไหลของข้อมูล (Data Flow)

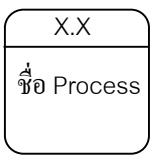
7.3.3. ตัวแทนข้อมูล (External Agent)

7.3.4. แหล่งจัดเก็บข้อมูล (Data Store)

7.3.1. ขั้นตอนการทำงานของระบบ (Process)

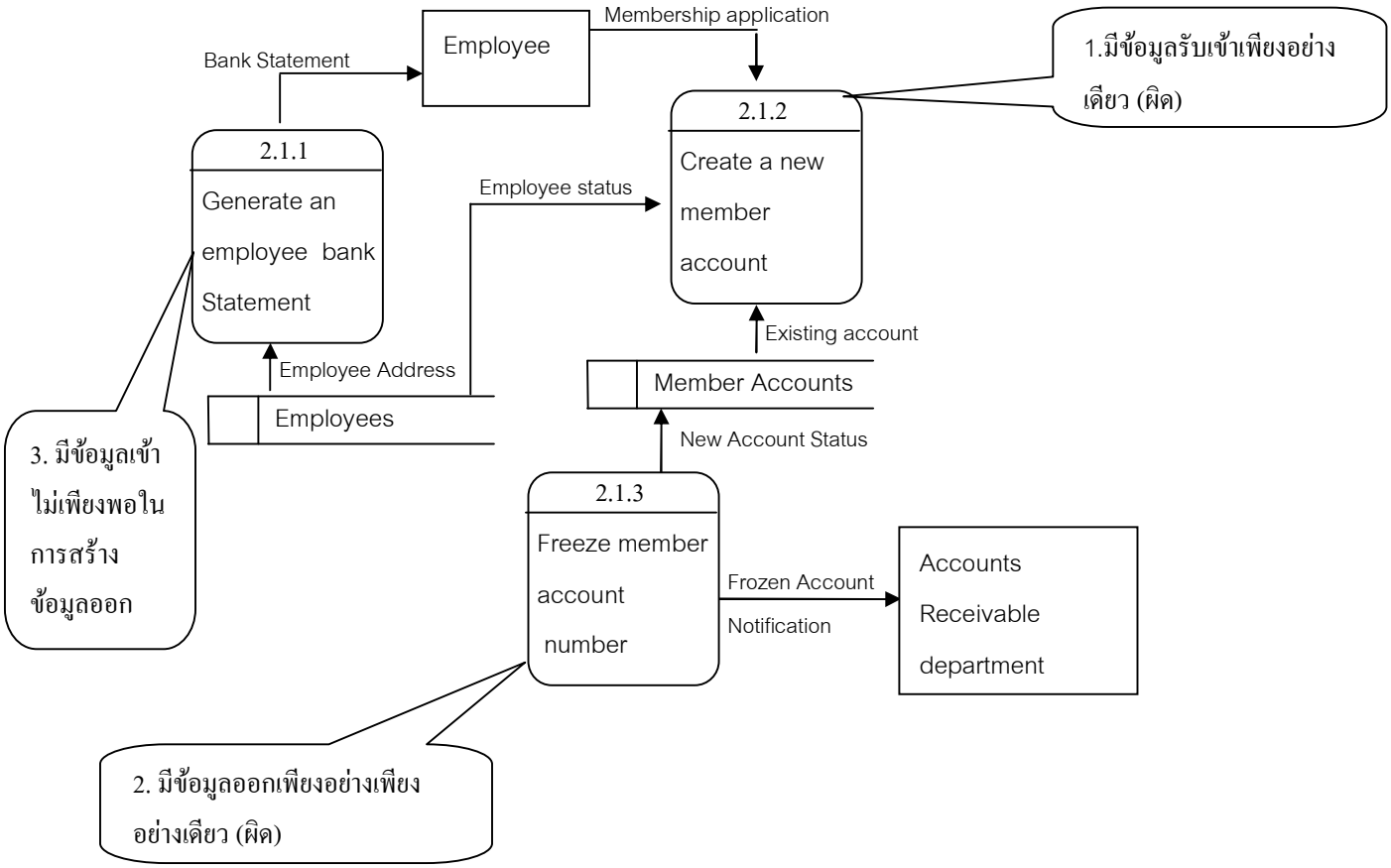
Process หรือ ขั้นตอนการดำเนินงาน คือ งานที่ดำเนินการ/ตอบสนองข้อมูลที่รับเข้าหรือดำเนินการ/ตอบสนองต่อเงื่อนไข/ สภาวะใดๆ ที่เกิดขึ้น ไม่ว่าขั้นตอนการดำเนินงานนั้นจะกระทำโดยบุคคล หน่วยงาน หุ่นยนต์ เครื่องจักร หรือ เครื่องคอมพิวเตอร์ก็ตาม โดยจะเป็นกริยา (Verb) เช่น ลงทะเบียน เพิกถอนวิชา เพิ่มวิชา พิมพ์รายงาน เป็นต้น จำนวนโปรเซสควรมีอยู่ระหว่าง 2-7 โปรเซส หรือในบางตำราได้กำหนดจำนวนโปรเซสควรอยู่ในระหว่าง 7 บวกลบด้วย 2

สัญลักษณ์ที่ใช้แสดงแทน Process



จากรูป แสดงสัญลักษณ์ที่ใช้แสดงแทน Process ด้วยสี่เหลี่ยมมุมมน ประกอบไปด้วย 2 ส่วน คือ ส่วนบนใช้แสดงหมายเลขของ Process เช่น 0, 1.0, 1.1 เป็นต้น ส่วนล่างจะใช้แสดงชื่อของ Process เช่น

<u>กฎของ Process รูปแสดงข้อผิดพลาดของ Process ในแผนภาพกระแสข้อมูล</u>



กฎของ Process

1. ต้องไม่มีข้อมูลรับเข้าเพียงอย่างเดียว โดยไม่มีการส่งข้อมูลออกจากขั้นตอนการทำงาน (Process) เรียกข้อผิดพลาดชนิดนี้ว่า “Black Hole” เนื่องจากข้อมูลที่รับเข้ามาแล้วสูญหายไป จากรูป คือ Process 2.1.2 ที่มีข้อผิดพลาดลักษณะนี้

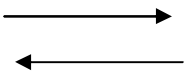
2. ต้องไม่มีข้อมูลออกเพียงอย่างเดียว โดยไม่มีข้อมูลเข้าสู่ Process เลย จากรูป คือ Process 2.1.3 ที่มีข้อผิดพลาดลักษณะนี้
3. ข้อมูลรับเข้าจะต้องเพียงพอในการสร้างข้อมูลส่งออก กรณีที่มีข้อมูลที่รับเข้าไม่เพียงพอในการสร้างข้อมูลส่งออกเรียกว่า “Gray Hole” โดยอาจเกิดจากการรวบรวมข้อเท็จจริงและข้อมูลไม่สมบูรณ์ หรือการใช้ชื่อข้อมูลรับเข้าและข้อมูลส่งออกผิดจากรูปคือ Process 2.1.1 ที่มีข้อผิดพลาดลักษณะเช่นนี้ เนื่องจากข้อมูลที่รับเข้ามามีเพียงที่อยู่ของพนักงาน (Employee Address) แต่ไม่มีข้อมูลกระแสเงินสดในธนาคารของลูกค้าที่เข้าสู่ Process ดังนั้นข้อมูลจึงไม่เพียงพอที่จะสร้างเป็นรายงานสถานะทางการเงินทางธนาคารของพนักงานได้ (Bank Statement)
4. การตั้งชื่อ Process ต้องใช้คำกริยา (Verb) เช่น Prepare Management Report, Calculate Data สำหรับภาษาไทยใช้เป็นคำกริยาเช่นเดียวกัน เช่น บันทึกข้อมูลใบสั่งซื้อ ตรวจสอบข้อมูลลูกค้า คำนวณเงินเดือน เป็นต้น

7.3.2. เส้นทางการไหลของข้อมูล (Data Flow)

เส้นทางการไหลของข้อมูล (Data Flows) เป็นการสื่อสารระหว่างขั้นตอนการทำงาน (Process) ต่างๆ และสภาพแวดล้อมภายนอกหรือภายในระบบ โดยแสดงถึงข้อมูลที่นำเข้าไปในแต่ละ Process และข้อมูลที่ส่งออกจาก Process ใช้ในการแสดงถึงการบันทึกข้อมูล การลบข้อมูล การแก้ไขข้อมูลต่างๆ ในไฟล์หรือในฐานข้อมูล ซึ่งใน Data Flow Diagram เรียกว่า “Data Store”

สัญลักษณ์ของ Data Flow

สัญลักษณ์ที่ใช้อธิบายเส้นทางการไหลของข้อมูลคือ เส้นตรงที่ประกอบด้วยหัวลูกศรตรงปลายเพื่อบอกทิศทางการเดินทางหรือการไหลของข้อมูล ดังรูป



กฎของ Data Flow

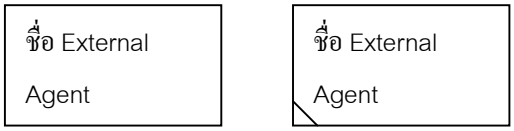
1. ชื่อของ Data Flow ควรเป็นชื่อของข้อมูลที่ส่งโดยไม่ต้องอธิบายว่าส่งอย่างไร ทำงานอย่างไร
2. Data Flow ต้องมีจุดเริ่มต้นหรือสิ้นสุดที่ Process เพราะ Data Flow คือข้อมูลนำเข้า (Inputs) และข้อมูลส่งออก (Outputs) ของ Process
3. Data Flow จะเดินทางระหว่าง External Agent กับ External Agent ไม่ได้
4. Data Flow จะเดินทางจาก External Agent ไป Data Store ไม่ได้
5. Data Flow จะเดินทางจาก Data Store ไป External Agent ไม่ได้
6. Data Flow จะเดินทางระหว่าง Data Store กับ Data Store ไม่ได้

7. การตั้งชื่อ Data Flow จะต้องใช้คำนาม (Noun) เช่น Inventory Data, Goods Sold Data เป็นต้น

7.3.3. ตัวแทนข้อมูล (External Agent)

ตัวแทนข้อมูล (External Agents) หมายถึง บุคคล หน่วยงานในองค์กร องค์กรอื่นๆ หรือระบบงานอื่นๆ ที่อยู่ภายนอกขอบเขตของระบบ แต่มีความสัมพันธ์กับระบบ โดยมีการส่งข้อมูลเข้าสู่ระบบเพื่อดำเนินงาน และรับข้อมูลที่ผ่านการดำเนินงานเรียบร้อยแล้วจากระบบ ในบางครั้งเรียกว่า "External Entity"

สัญลักษณ์ของ External Agents สัญลักษณ์ที่ใช้อธิบาย คือ สี่เหลี่ยมจตุรัส หรือสี่เหลี่ยมผืนผ้า ภายในจะต้องแสดงชื่อของ External Agent โดยสามารถทำการซ้ำ (Duplicate) ได้ด้วยการใช้เครื่องหมาย \ (back slash) ตรงมุมล่างซ้าย



กฎของ External Agents

1. ข้อมูลจาก External Agent จะวิ่งไปสู่อีก External Agent หนึ่งโดยตรงไม่ได้ จะต้องผ่าน Process ก่อนเพื่อประมวลข้อมูลนั้น จึงได้ข้อมูลออกไปสู่อีก External Agent
2. การตั้งชื่อ External Agent ต้องใช้คำนาม (Noun) เช่น Customer, Bank เป็นต้น

7.3.4. แหล่งจัดเก็บข้อมูล (Data Store)

แหล่งจัดเก็บข้อมูล (Data Store) เป็นแหล่งเก็บ/บันทึกข้อมูล เปรียบเสมือนคลังข้อมูล (เทียบเท่ากับไฟล์ข้อมูล และฐานข้อมูล) โดยอธิบายรายละเอียดและคุณสมบัติเฉพาะตัวของสิ่งที่ต้องการเก็บ/บันทึก

สัญลักษณ์ของ Data Store สัญลักษณ์ที่ใช้อธิบายคือสี่เหลี่ยมเปิดหนึ่งข้าง แบ่งออกเป็นสองส่วน ได้แก่ ส่วนที่ 1 ทางด้านซ้ายใช้แสดงรหัสของ Data Store อาจจะเป็นหมายเลขลำดับหรือตัวอักษรได้เช่น D1, D2 เป็นต้น สำหรับส่วนที่ 2 ทางด้านขวา ใช้แสดงชื่อ Data Store หรือชื่อไฟล์ เช่น Employee, Application, Member เป็นต้น ดังรูป

กฎของ Data Store

1. ข้อมูลจาก Data Store หนึ่งจะวิ่งไปสู่อีก Data Store หนึ่งโดยตรงไม่ได้ จะต้องผ่านการประมวลผลจาก Process ก่อน
2. ข้อมูลจาก External Agent จะวิ่งเข้าสู่ External Agent โดยตรงไม่ได้
3. การตั้งชื่อ Data Store จะต้องใช้คำนาม (Noun) เช่น Customer File, Inventory หรือ Employee File เป็นต้น

7.4. วิธีการสร้างแบบจำลองขั้นตอนการทำงานของระบบด้วย DFD

หัวข้อที่ผ่านมาได้รู้จักกับแนวคิด สัญลักษณ์ และกฎเกณฑ์ต่างๆ ของแนวคิดทั้งหมดของแผนภาพกระแสข้อมูล (DFD)ในหัวข้อนี้จะนำเสนอวิธีการสร้าง DFD ตามลำดับดังนี้

7.4.1. สร้างแผนภาพบริบท (Context Diagram)

7.4.2. สร้างแผนภาพระดับ 0 (Level-0 Diagram)

7.4.3. แบ่งย่อยแผนภาพ (Decomposition of DFD)

7.4.4. ตรวจสอบสมดุลของ DFD (Balancing DFD)

7.4.1. สร้างแผนภาพบริบท (Context Diagram)

แผนภาพบริบท (Context Diagram) คือ แผนภาพกระแสข้อมูลระดับบนสุดที่แสดงภาพรวมการทำงานของระบบที่มีความสัมพันธ์กับสภาพแวดล้อมภายนอกระบบ ทั้งยังแสดงให้เห็นขอบเขตและเส้นแบ่งเขตของระบบที่ศึกษาและพัฒนา

อันดับแรกของการสร้างแบบจำลองขั้นตอนการทำงานของระบบ นักวิเคราะห์ระบบควรจะทำการสร้าง Context Diagram ก่อน เนื่องจาก Context Diagram เป็นตัวกำหนดขอบเขต และเส้นแบ่งเขตของระบบที่ศึกษาและพัฒนา แนวทางในการกำหนดขอบเขตมีดังนี้

1. เปรียบระบบเสมือนภาชนะบรรจุ เพื่อแบ่งแยกสิ่งที่อยู่ภายในภาชนะออกจากสิ่งที่อยู่ภายนอกภาชนะ โดยไม่ต้องสนใจสิ่งที่อยู่ภายในภาชนะมีอะไรบ้าง
2. ศึกษาระบบโดยอาจจะการสอบถามผู้ใช้งานถึงเหตุการณ์ (Event) หรือ การดำเนินงานประจำวันที่เกิดขึ้นของระบบว่ามีการติดต่อ จัดการ หรือดำเนินงานอย่างไรบ้าง และระบบมีการตอบสนองต่อเหตุการณ์นั้นๆ อย่างไร อะไรคือข้อมูลที่รับเข้ามา (Input) และส่งมาจากใคร (External Agent)
3. สอบถามผู้ใช้ระบบว่าระบบจะต้องส่งข้อมูลอะไร (Output) ออกไปสู่ External Agent บ้าง ต้องการรูปแบบรายงาน การสอบถามข้อมูล (Query) แบบใด สิ่งเหล่านี้ทำให้นักวิเคราะห์ระบบสามารถพิจารณาการวาด Data Flow ได้
4. จำแนกแหล่งข้อมูลภายนอกระบบ (External data store) ที่ระบบต้องการจากไฟล์หรือฐานข้อมูลจากระบบอื่น ซึ่งอาจเป็นการอ่าน แก้ไข เปลี่ยนแปลง ข้อมูลเหล่านั้น
5. ทำการวาด Context Diagram จากสิ่งที่รวบรวมได้จากข้อ 1-4

หลังจากที่ได้ศึกษาการทำงาน ข้อมูลรับเข้า ข้อมูลส่งออก นักวิเคราะห์ระบบอาจมีเส้นทางการไหลของข้อมูล (Data Flow) มากมาย ซึ่งไม่อาจแสดงได้ทั้งหมดใน Context Diagram นี้ ดังนั้น Data Flow ที่แสดงควรเป็นข้อมูลหลักและมีความสำคัญต่อระบบ ส่วนรายละเอียดของการเคลื่อนไหวของข้อมูลนั้นสามารถนำไปอธิบายใน DFD ระดับต่อไปได้

ใน Context Diagram ประกอบด้วย Process ที่แทน Process ของระบบทั้งหมดเพียงหนึ่ง Process เท่านั้นที่อยู่ภายในขอบเขตของระบบ และให้แสดงหมายเลขศูนย์ (“o”) ตรงส่วนบนของสัญลักษณ์ Process นอกจากนี้ใน Context Diagram ยังแสดงรายละเอียดของ External Agent และ External Data Store รอบๆ ขั้นตอนการดำเนินงาน (ภายนอกขอบเขตของระบบ) และมี Data Flows แสดงการติดต่อระหว่างระบบกับสิ่งที่อยู่ภายนอก และสิ่งสำคัญคือภายใน Context Diagram จะต้องไม่มี Data Store ปรากฏอยู่

7.4.2. สร้างแผนภาพระดับ 0 (Level-0 Diagram)

Level-0 Diagram คือ แผนภาพกระแสข้อมูลในระดับที่แสดงขั้นตอนการทำงานหลักทั้งหมด (Process หลัก) ของระบบแสดงทิศทางการไหลของ Data Flow และแสดงรายละเอียดของแหล่งจัดเก็บข้อมูล (Data Store)

Level-0 Diagram เป็นการแสดงให้เห็นถึงรายละเอียดของ Process การทำงานหลักๆ ที่มีอยู่ภายในภาพรวมของระบบ (Context Diagram) ว่ามีขั้นตอนใดบ้าง โดยแต่ละ Process จะมีหมายเลขกำกับอยู่ด้านบนของสัญลักษณ์ ตั้งแต่ 1 เป็นต้นไป

7.4.3. แบ่งย่อยแผนภาพ (Decomposition of DFD)

ถ้าระบบใดมีการทำงานที่ซับซ้อนมาก นักวิเคราะห์ระบบจะไม่สามารถอธิบายการทำงานทั้งหมดได้ภายในขั้นตอนเดียวใน Context Diagram ดังนั้นในการวิเคราะห์ระบบจึงสามารถจำแนกระบบใหญ่หนึ่งระบบออกเป็นระบบย่อยๆ ได้หลายระบบ โดยแบ่งให้เป็นระบบย่อยที่มีขนาดเล็กลงเรื่อยๆ จนสามารถอธิบายการทำงานได้ทั้งหมด เรียกวิธีนี้ว่า “ การแบ่งย่อย (Decomposition) หรือ Functional Decomposition”

Decomposition คือ การแบ่ง/แยก/ย่อยระบบและขั้นตอนการทำงานออกเป็นส่วนย่อย โดยในแต่ละขั้นตอนที่แยกออกมา (Subsystems) จะแสดงให้เห็นถึงรายละเอียดของการทำงานเพิ่มมากขึ้น

การแบ่งย่อย Process นั้นสามารถแบ่งย่อยลงไปได้เรื่อยๆ จนกระทั่งถึงระดับที่ไม่สามารถแบ่งย่อยได้อีกแล้ว เรียกแผนภาพที่ไม่สามารถแบ่งย่อย Process ได้อีกแล้วว่า Primitive DFD

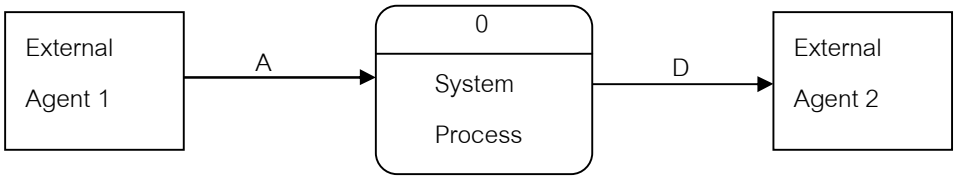
ระดับของแผนภาพที่แบ่งย่อยมาจาก Level-0 เรียกว่า Level-1 ซึ่งแผนภาพที่แบ่งย่อยในระดับถัดมาจาก Level-0 diagram จะต้องมี Process อย่างน้อย 2 Process ขึ้นไป

7.4.4. ตรวจสอบสมดุลของ DFD (Balancing DFD)

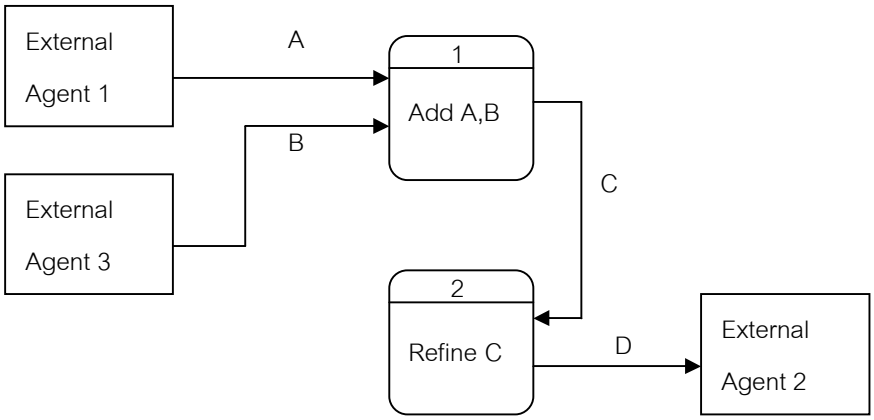
เมื่อมีการแบ่งย่อยแผนภาพจากระดับบนลงไประดับล่าง เช่น จาก Level-0 แบ่งย่อยไปใน Level-1 ของ Process 1 นักวิเคราะห์ระบบ จะต้องการตรวจสอบความสมดุลของแผนภาพ (Balancing DFD) ด้วย

Balancing DFD หมายถึง ความสมดุลของแผนภาพกระแสข้อมูลที่จะต้องมี Input Data Flow ที่เข้าสู่ระบบและ Output Data Flow ที่ออกจากระบบใน DFD ระดับล่างครบทุก Input Data Flow และ Output Data Flow ที่ปรากฏอยู่ใน DFD ระดับบน แต่ในระดับล่างอาจจะมีมากกว่าได้ โดยมีเงื่อนไขว่า Input Data Flow และ Output Data Flow นั้นจะต้องเกิดจาก Process ภายในระดับล่างเท่านั้น และจะนำไปใช้ตรวจสอบความสมดุลของแผนภาพอีกระดับหากมีการแบ่งย่อยแผนภาพในระดับล่างลงไปอีก ดังรูป

(a)



(b)



จากรูป (a) เป็น Context Diagram ที่มี Input Data Flow เข้าสู่ระบบคือ A จาก External Agent 1 เท่านั้น และมี Output Data Flow คือ D วิ่งไปยัง External Agent 2 เมื่อมีการแบ่งย่อยแผนภาพลงที่ Level-0 Diagram ในรูป (b) สังเกตว่ามี Input Data Flow ที่เป็น B จาก External Agent 3 เพิ่มเข้ามา ซึ่งใน Context Diagram ไม่มี External Agent 3 นี้ ดังนั้นถือได้ว่า DFD นี้ไม่สมดุลสำหรับ Input Data Flow C สามารถปรากฏอยู่ใน DFD ระดับล่างได้ เนื่องจากเป็น Input Data Flow ที่เกิดจาก Process ภายในระดับล่างนี้เท่านั้น

7.5. แนวทางในการสร้างแผนภาพกระแสข้อมูลที่สมบูรณ์

เมื่อนักวิเคราะห์ระบบสร้างแผนภาพกระแสข้อมูลของระบบปัจจุบัน และระบบใหม่ที่จะเสนอให้เป็นทางเลือกในการแก้ไขปัญหาเสร็จสิ้นแล้ว นอกจาก Data Flow, Processes, Data Stores และ External Agent จะเป็นไปตามกฎที่ได้กล่าวไว้ในหัวข้อที่ 7.3. และตรวจสอบความสมดุลของแผนภาพแล้ว นักวิเคราะห์ระบบควรมีการตรวจสอบเพิ่มเติมเพื่อให้ได้แผนภาพที่สามารถแสดงให้เห็นถึงรายละเอียดขั้นตอนการทำงาน ข้อมูลที่เกิดจากการประมวลผลแต่ละขั้นตอน และการจัดเก็บข้อมูลได้อย่างถูกต้องและครบถ้วน โดยมีหลักเกณฑ์ดังนี้

7.5.1. มีความสมบูรณ์ (DFD Completeness)

ใจความสำคัญของหลักเกณฑ์นี้คือ หากมีการเพิ่มเติมรายละเอียดใดๆ ที่จำเป็นเข้ามาในระบบ นักวิเคราะห์ระบบจะต้องเพิ่มเติมรายละเอียดเหล่านั้นลงใน DFD ด้วยเสมอ และหาก Data Flow, Data Store, Process และ External Agent บนแผนภาพ DFD ไม่เชื่อมต่ออยู่กับสิ่งใดๆ แสดงว่า DFD นั้นไม่สมบูรณ์

7.5.2. มีความสอดคล้อง (DFD Consistency)

เป็นความสอดคล้องกันของสิ่งที่ปรากฏอยู่บน DFD ในระดับบนและมีการแบ่งย่อยลงมาในระดับล่าง กล่าวคือ สิ่งที่ปรากฏอยู่บน DFD ในระดับบน เมื่อมีการแบ่งย่อย Process หรือแผนภาพลงมาในระดับล่าง จะต้องมีสิ่งที่ปรากฏอยู่ในระดับบนนั้นด้วยเสมอ หลักเกณฑ์นี้จะเกี่ยวข้องกับกฎความสมดุลของแผนภาพ DFD

7.5.3. การทำซ้ำ (Iterative Development)

การสร้าง DFD ในรอบแรกนั้นจะยังไม่เป็นแผนภาพที่มีความถูกต้องและสมบูรณ์ได้ จะต้องมีการตรวจสอบแผนภาพหรือมีการปรับปรุงแผนภาพทุกครั้งที่มีการเปลี่ยนแปลงหรือแก้ไขความต้องการ การปรับปรุงแผนภาพนี้จะทำให้มีความถูกต้องมากขึ้นนั่นเอง หากองค์กรใดเลือกใช้ CASE จะทำให้ประหยัดเวลาในส่วนนี้ไป

7.5.4. DFD ระดับล่างสุด (Primitive DFD)

เมื่อมีการแบ่งย่อยแผนภาพ DFD ลงมาที่ระดับล่าง เพื่ออธิบายรายละเอียดของขั้นตอนการทำงานภายในระบบ ปัญหาที่เกิดขึ้นคือ “ ควรจะสิ้นสุดการแบ่งย่อย Process เมื่อใด” หลักเกณฑ์โดยทั่วไปที่ใช้ในการตัดสินว่า เมื่อใดที่ควรจะหยุดแบ่งย่อย Process คือ “ เมื่อไม่สามารถแบ่งย่อย Process ได้อีกแล้ว” นอกจากหลักเกณฑ์ดังกล่าวแล้ว ในที่นี้ยังมีหลักเกณฑ์ในการตัดสินใจดังนี้

1. เมื่อมีการแบ่งย่อย Process แต่ละ Process ลงมาจนกระทั่งมีการทำงานใน Process นั้นเพียงหน้าที่เดียว เช่นมีการอ่านข้อมูล ปรับปรุง สร้าง และลบข้อมูลในฐานข้อมูล เป็นต้น

2. เมื่อแต่ละ Data Store ที่ใช้ในการจัดเก็บข้อมูล มีการจัดเก็บข้อมูลเพียงไฟล์เดียว เช่น ไฟล์ลูกค้า ไฟล์สินค้า หรือไฟล์สั่งซื้อ เป็นต้น
3. เมื่อผู้ใช้ระบบเห็นว่าไม่มีรายละเอียดใดๆ ที่จะเป็นต่อการทำงานของระบบแล้ว

เหล่านี้เป็นการเพิ่มความถูกต้องและสมบูรณ์ของแผนภาพกระแสข้อมูล ทำให้มั่นใจได้ว่าข้อมูลที่แสดงอยู่ในแผนภาพ รวมทั้งขั้นตอนการทำงานต่างๆ นั้นไม่ผิดพลาดหรือขาดหายไปก่อนที่จะการนำเสนอต่อผู้บริหารและส่งมอบให้กับนักออกแบบระบบต่อไป

บทที่ 8

คำอธิบายขั้นตอนการทำงานของระบบ (Logic of Processes/Logic Modeling)

ในบทที่แล้วได้ทราบถึงการจำลองแบบขั้นตอนการทำงาน ซึ่งทำให้ทราบว่าภายในระบบมีขั้นตอนการทำงานอย่างไรบ้าง และแต่ละขั้นตอนนั้นมีข้อมูลใดบ้างที่ถูกส่งเข้ามาประมวลผลเพื่อให้ได้เป็นข้อมูลหรือสารสนเทศออกจาก Process แต่หากต้องการเพิ่มเติมรายละเอียดในการทำงานภายในและแต่ละ Process เพื่อบอกให้ทราบว่า Process นั้นทำงานอย่างไร (Logic of Process) จะไม่สามารถแสดงไว้ใน DFD ได้ ดังนั้นในบทเรียนนี้จะแนะนำเทคนิคที่ใช้ในการอธิบายการทำงานของ Process ในส่วนของการตัดสินใจประมวลผลข้อมูลที่มีเงื่อนไขต่างๆ ของ process (Process Decision Logic)

8.1. แนะนำ Logic of Processes

แผนภาพกระแสข้อมูล (Data Flow Diagram) ใช้อธิบายขั้นตอนการทำงานทั้งหมดของระบบ แต่อย่างไรก็ตาม แผนภาพกระแสข้อมูลนี้จะใช้งานไม่คล่องตัว หากมีความต้องการรายละเอียดในแต่ละโปรเซส นั่นคือ ยังไม่สามารถอธิบายได้ว่า Process นั้นทำงานอย่างไร มีการรับข้อมูลเข้ามาประมวลผล และตรวจสอบข้อมูลที่รับเข้ามาได้อย่างไร (Logic of Process) ด้วยเหตุนี้ จึงมีเทคนิคในการจำลองวิธีการทำงานและการประมวลผลของ Process ให้ผู้ที่เกี่ยวข้องกับการพัฒนาระบบสามารถทราบได้ว่าแต่ละ Process นั้นมีการทำงานอย่างไร

คำอธิบายขั้นตอนการทำงานของระบบ (Logic of Process) หรืออีกอย่างหนึ่งว่า Logic Modeling เป็นการแสดงให้เห็นถึงโครงสร้าง หน้าที่ และลักษณะการทำงานของ Process ที่ปรากฏอยู่บนแผนภาพกระแสข้อมูล (DFD)

คำอธิบาย Process ช่วยให้นักออกแบบระบบและโปรแกรมเมอร์ สามารถเข้าใจการทำงานภายใน Process ได้โดยง่าย โดยใช้ดูประกอบกับแผนภาพชนิดต่างๆ ที่ได้จากขั้นตอนการวิเคราะห์ระบบ เพื่อนำไปออกแบบ และเขียนโปรแกรมได้สะดวกยิ่งขึ้น สิ่งที่ทำให้การเขียนโปรแกรมสะดวกขึ้นนอกจากจะเป็นแผนภาพต่างๆ แล้ว เทคนิคที่ใช้อธิบาย Process เองยังสามารถช่วยให้การกำหนดตัวแปรที่จะใช้ในโปรแกรมนั้นง่ายขึ้นอีกด้วย

ดังนั้นลักษณะการทำงานของ Process ควรจะมีการอธิบายอย่างละเอียดไว้ในขั้นตอนการวิเคราะห์ระบบก่อนที่จะส่งมอบต่อไปในขั้นตอนการออกแบบระบบ และการอธิบายควรใช้คำที่กะทัดรัด ได้ใจความ และมีลักษณะการตัดสินใจที่คล้ายกับภาษาที่ใช้ในการเขียนโปรแกรม

8.2. **เทคนิคที่ใช้ในการอธิบาย** Logic of Processes

จากหัวข้อที่แล้วผู้อ่านได้ทราบถึงสาเหตุที่ต้องการอธิบาย Process และประโยชน์ของการอธิบายไปแล้ว คำถามที่ตามมาคือ “ เมื่อใดที่ควรจะอธิบาย Process” คำตอบคือ ควรจะมีการอธิบาย Process ที่อยู่บนแผนภาพ DFD ในระดับสุดท้ายหรือล่างสุด (Primitive DFD) หรือควรอธิบายเมื่อนักวิเคราะห์ระบบเห็นควรสำหรับ Process ที่ไม่มีความซับซ้อนและสามารถอ่านจาก DFD แล้วเข้าใจได้เลย กรณีเช่นนี้ไม่จำเป็นต้องเขียนคำอธิบาย ในหัวข้อนี้จะนำเสนอเทคนิคที่ใช้ในการอธิบาย Process ซึ่งมีเทคนิคดังต่อไปนี้

8.2.1. ภาษาอังกฤษแบบโครงสร้าง (Structured English)

8.2.2. ตารางการตัดสินใจ (Decision Table)

8.2.3. การตัดสินใจแบบต้นไม้ (Decision Tree)

8.2.1. ภาษาอังกฤษแบบโครงสร้าง (Structured English)

Structured English คือ การนำภาษาอังกฤษมาเขียนเพื่อบ่งบอกรายละเอียดการทำงานของ Process ที่ปรากฏอยู่บน DFD โดยมีรูปแบบการเขียนใกล้เคียงกับไวยากรณ์ที่ใช้ในการเขียนโปรแกรม

รูปแบบของการเขียน Structured English จะมีลักษณะคล้ายกับรูปแบบของการเขียนโปรแกรมแบบโครงสร้าง (Structured Programming) ที่จำแนกมาจากการทำงานของโปรแกรม ซึ่งมี 3 ลักษณะดังนี้

1. แบบตามลำดับ (Sequence)
2. แบบมีเงื่อนไข (Conditional หรือ Decision Stucture)
3. แบบการทำซ้ำ (Iteration หรือ Repetition)

แบบตามลำดับ (Sequence)

การทำงานแบบตามลำดับ (Sequence) มีลักษณะการทำงานเป็นไปตามลำดับขั้นตอนหรือกิจกรรม ไม่มีการกระโดดข้ามไปทำขั้นตอนหรือกิจกรรมอื่นก่อน ดังนั้นในการเขียนคำอธิบาย Process ด้วยการใช้ Structured English แบบตามลำดับนี้ควรจะมีลักษณะที่เป็นประโยคที่แสดงให้เห็นถึงการทำงานเดียวอย่างชัดเจน ไม่ควรเป็นประโยครวมที่มีลักษณะเป็นการทำงานซ้อนกันหรือประโยคมีความคลุมเครือ
ตัวอย่างการเขียนแบบตามลำดับ เช่น

Read Record

Calculate Gross Pay = HOURS WORKED*HOUR WAGE

Print Gross Pay

กิจกรรมแรกที่ทำคือ Read Record คืออ่านข้อมูลเข้ามา เพื่อได้ข้อมูลแล้วจึงจะสามารถทำการคำนวณ Gross Pay ได้ และสั่งพิมพ์ค่า Gross Pay เป็นลำดับสุดท้าย

แบบมีเงื่อนไข (Conditional /Decision Structure)

เป็นการทำงานที่มีการกำหนดการกระทำหรือกิจกรรมการทำงานแตกต่างกันไปตามแต่ละเงื่อนไข ถ้าข้อมูลที่เข้าสู่ Process นั้นเป็นไปตามเงื่อนไขใด ให้ทำงานภายใต้สิ่งที่เงื่อนไขนั้นกำหนดไว้ โดยรูปแบบการเขียนคำอธิบาย Process โดยใช้ Structured English แบบมีเงื่อนไข แบ่งออกได้ 2 ลักษณะ ดังนี้

1. **If-then-else**

เป็นโครงสร้างการเขียนแบบมีเงื่อนไขโดยใช้ประโยค IF-THEN-ELSE มาช่วยในการอธิบายลักษณะการทำงานที่จะเกิดการกระทำกิจกรรม (Action) ใดๆ ที่กำหนดไว้ได้ ก็ต่อเมื่อเงื่อนไขที่ระบุไว้นั้นเป็นจริง แต่ถ้าเงื่อนไขนั้นเป็นเท็จจะต้องกระทำกิจกรรมอื่นที่กำหนดไว้ ภายใต้เงื่อนไขที่เป็นเท็จนั้น ดังตัวอย่างต่อไปนี้

```
If Accept_Applicant  then
        Print Accepted Letters
        Record Applicant_Data in Applicant_File
Else
        Print Reject Letters
End If
```

หมายเหตุ โครงสร้างของการทำงานแบบมีเงื่อนไขด้วย IF-Then-Else นี้จะมีการกระทำกิจกรรมที่เป็นไปได้เพียงสองทางเท่านั้น คือ กระทำกิจกรรมหากเงื่อนไขนั้นเป็นจริง และกระทำกิจกรรมหากเงื่อนไขนั้นเป็นเท็จ

2. CASE

เป็นโครงสร้างการเขียนแบบมีเงื่อนไข ซึ่งจะใช้ในกรณีที่มีการกระทำกิจกรรมที่เป็นไปได้มากกว่าสองทาง โดยใช้คำว่า CASE เพื่อตรวจสอบแต่ละเงื่อนไขที่เป็นไปได้เหล่านั้น ด้วยรูปแบบที่ดูง่ายกว่าการใช้ IF-Then-Else If-Then-Else If-Then-Else ....หลายๆครั้ง ดังตัวอย่างต่อไปนี้

```
Select Case Item
        Case 1 : if Grade <= 2.00 then
                 Reject Applicant
        Case 2 : if Grade > 2.00 and Grade <= 3.50 then
                 Print Interview Letters
        Case 3 : If Grade > 3.50 then
                 Print Interview Letters
                 Record Application_Data
End Select
```

แบบการทำซ้ำ (Iteation/Repetition)

เป็นโครงสร้างของการเขียนที่มีลักษณะการกระทำกิจกรรมซ้ำไปเรื่อยๆ ภายใต้เงื่อนไขที่กำหนด ลักษณะการทำซ้ำสามารถเขียนคำอธิบาย Process ด้วย Structured English ได้ดังนี้

1. Do-While

เป็นการกระทำกิจกรรภายใต้เงื่อนไขที่เป็นจริงเท่านั้น จึงทำกิจกรรมเหล่านั้นซ้ำ จนกระทั่งเงื่อนไขเป็นเท็จ จึงหยุดประมวลผล ดังตัวอย่าง

```
Read Employee Record
Do No End-of-File  while
    Print Employee Recorde
End Do
```

จากตัวอย่างจะเห็นว่ามีการตรวจสอบเงื่อนไขก่อนว่าเป็นจริงหรือไม่ ถ้าเป็นจริง จึงสามารถเข้ามาทำกิจกรรมภายใต้เงื่อนไขซ้ำไปเรื่อยๆ จนกระทั่งเงื่อนไขนั้นเป็นเท็จจึงจะไม่ทำกิจกรรมในเงื่อนไข

2. Do-Until

เป็นการกระทำกิจกรรมใดๆ ซ้ำจนกระทั่งเงื่อนไขนั้นเป็นจริง ดังตัวอย่างต่อไปนี้

```
Do
        Read Employee Record
        Print Employee Record
Until End-of-File
```

จากตัวอย่างจะเห็นว่า มีการกระทำกิจกรรมก่อนอย่างน้อย 1 ครั้ง แล้วจึงมีการตรวจสอบเงื่อนไขที่ได้ระบุไว้ หากเป็นจริงตามเงื่อนไขจึงหยุดกระทำกิจกรรม

8.2.2. ตารางการตัดสินใจ (Decision Table)

การอธิบายการทำงานของ Process ด้วย Structured English อาจจะไม่สามารถอธิบายการทำงานของ Process ที่มีเงื่อนไขซับซ้อนให้เข้าใจได้ง่ายพอ หรือหากสามารถอธิบายได้จะทำให้ Structured

English นั้นมีความซับซ้อนมากเกินไปเทคนิคในการอธิบาย Process ที่สามารถอธิบายเงื่อนไขที่มีความซับซ้อนให้ดูง่ายขึ้นในหัวข้อนี้คือ “ตารางการตัดสินใจ (Decision Table)”

Decision Table คือ แผนภาพที่ใช้ในการอธิบายการทำงานของ Process ที่มีเงื่อนไขการตัดสินใจที่ซับซ้อน โดยแสดงเงื่อนไข (Conditions) การกระทำ (Actions) และกิจกรรมที่เป็นไปได้ตามกฎเกณฑ์ (Rules) ของเงื่อนไขนั้นอยู่ในรูปของตาราง

- Conditions คือ เงื่อนไขต่างๆ ที่กำหนดขึ้น
- Action คือ ผลของเงื่อนไข ซึ่งได้จากเงื่อนไขต่างๆ มาประมวลจนได้ผลลัพธ์
- Rule คือ กฎเกณฑ์ เป็นการรวมกันของเงื่อนไขและการกระทำอันใดอันหนึ่งที่ระบุว่ากิจกรรมใดที่จะต้องกระทำตามเงื่อนไขใด

Conditions	Rules 1. 2. 3. 4. 5. 6. 7. 8. Etc.
1….. 2…. 3…. 4…. 5… etc.	
Actions 1…. 2…. 3… etc.	

		1	2	3	4	5	6	7	8
Conditions	Credit Limit exceeded	Y	Y	Y	Y	N	N	N	N
	Customer with good payment history	Y	Y	N	N	Y	Y	N	N
	Purchase above $200	Y	N	Y	N	Y	N	Y	N
Actions	Allow Credit					X	X	X	X
	Refuse Credit	X		X	X				
	Refer to manager		X						

Y = Yes, condition true

N = No, condition not true

อธิบายการประมวลผลในลักษณะของ Decision Table ของการอนุมัติบัตรเครดิตโดย Conditions คือ เงื่อนไขต่างๆ ซึ่งถ้าเงื่อนไขเป็น Y จะหมายถึงถูกตรงเงื่อนไข ถ้าเป็น N จะไม่ตรงกับเงื่อนไข ส่วน Action คือ การกระทำหรือผลที่เกิดจาก Condition ต่างๆ ว่าจะได้ผลอย่างไร

Conditions

- ถ้าหากลูกค้ามีการใช้วงเงินเกินเครดิต -→ ใช่
- ถ้าลูกค้ามีประวัติการชำระเงินที่ดี --→ ใช่
- ถ้ามียอดค่าใช้จ่ายเกิน 200 เหรียญ -→ ใช่

Action

- ไม่อนุมัติการใช้บัตรเครดิต

ตัวอย่างที่ 2

	Conditions/Causes of Action	Rules					
		1	2	3	4	5	6
Condition	Employee type	S	H	S	H	S	H
	Hours worked	<40	<40	40	40	>40	>40
Action	Pay base salary	X		X		X	
	Calculate hourly wage		X		X		X
	Calculate overtime						X
	Produce Absence Report		X				

อธิบายการประมวลผลเป็นตารางการตัดสินใจของระบบจ่ายเงินเดือน

Condition มี 2 เงื่อนไข คือ

1. ประเภทของลูกจ้าง (Employee Type) ซึ่งมีค่าที่เป็นไปได้ 2 ค่า คือ

- "S" ลูกจ้างรายเดือน และ
- "H" ลูกจ้างรายวัน

2.จำนวนชั่วโมงทำงาน (Hours worked) ซึ่งมีค่าเป็นไปได้ 3 ค่า คือน้อยกว่า 40 ชม. (<40) เท่ากับ 40 ชม. และมากกว่า 40 ชม. (>40)

Action มี 4 กิจกรรม คือ

1. จ่ายเงินเดือนตามปกติ (Pay base salary)
2. คำนวณจำนวนชั่วโมงทำงาน (Calculate hourly wage)
3. คำนวณจำนวนชั่วโมงล่วงเวลา (Calculate Overtime)
4. จัดพิมพ์รายงานการขาดงาน (Produce Absence Report)

Rule การอ่านตารางการตัดสินใจตามกฎเกณฑ์มีขั้นตอนดังนี้ เริ่มจากอ่านค่าของเงื่อนไข (Condition) ในคอลัมน์ที่ 1

คอลัมน์ที่ 1 ถ้าเป็นลูกจ้างรายเดือน (S) และจำนวนชั่วโมงน้อยกว่า 40 ชั่วโมง ระบบจะต้องจ่ายเงินเดือนตามปกติ

คอลัมน์ที่ 2 ถ้าเป็นลูกจ้างรายวัน (H) และจำนวนชั่วโมงทำงานน้อยกว่า 40 ชั่วโมง ระบบจะต้องคำนวณค่าแรงเป็นชั่วโมง และจัดพิมพ์รายงานการขาดงาน

คอลัมน์ที่ 3 ถ้าเป็นลูกจ้างรายเดือน (S) และจำนวนชั่วโมงทำงานเท่ากับ 40 ชั่วโมง ระบบจะต้องจ่ายเงินเดือนตามปกติ

คอลัมน์ที่ 4 ถ้าเป็นลูกจ้างรายวัน (H) และจำนวนชั่วโมงทำงานเท่ากับ 40 ชั่วโมง ระบบะต้องคำนวณค่าแรงตามชั่วโมงทำงาน

คอลัมน์ที่ 5 ถ้าเป็นลูกจ้างรายเดือน (S) และจำนวนชั่วโมงทำงานมากกว่า 40 ชั่วโมง ระบบจะต้องจ่ายเงินเดือนตามปกติ

คอลัมน์ที่ 6 ถ้าเป็นลูกจ้างรายวัน (H) และจำนวนชั่วโมงทำงานมากกว่า 40 ชั่วโมง ระบบจะต้องคำนวณค่าแรงตามชั่วโมงทำงาน และคำนวณชั่วโมงล่วงเวลา

จากตัวอย่างสังเกตได้ว่า ที่คอลัมน์ที่ 1,3 และ 5 ภายใต้เงื่อนไขของลูกจ้างรายเดือน (S) เหมือนกันทั้ง 3 คอลัมน์ มีการกระทำกิจกรรมที่เหมือนกันหรือกระทำกิจกรรมเดียวกันคือ จ่ายเงินเดือนตามปกติ (Pay base salary) ซึ่งเงื่อนไขที่เป็นจำนวนชั่วโมงทำงาน (Hours worked) นั้นไม่ว่าจะเป็นกี่ชั่วโมงก็ตามจะไม่มีผลกระทบต่อการกระทำกิจกรรม กล่าวคือ ยังกระทำกิจกรรมชนิดเดียวกันทั้ง 3 คอลัมน์ คือ จ่ายเงินเดือนตามปกติ กรณีที่เงื่อนไขการตัดสินใจไม่มีผลกระทบต่อการกระทำกิจกรรม เรียกว่า Indifferent condition

หากตารางการตัดสินใจมีกรณี Indifferent condition เกิดขึ้น นั่นหมายความว่า มีคอลัมน์ที่มีการกระทำกิจกรรมชนิดเดียวกันตั้งแต่ 2 คอลัมน์ขึ้นไป ให้ทำการยุบคอลัมน์เหล่านั้นให้เหลือเพียงคอลัมน์เดียวดังนี้

Conditions/Causes of Action	Rules			
	1	2	3	4
Employee type	S	H	H	H
Hours worked	-	<40	40	>40
Pay base salary	X			
Calculate Hourly wage		X	X	X
Calculate overtime				X
Produce Absence Report	-	X		

จากตารางสังเกตคอลัมน์ที่ 1 จะไม่แสดงค่าของเงื่อนไข Hours worked (-) เนื่องจาก เงื่อนไขประเภทนี้ไม่มีผลต่อการกระทำกิจกรรม Pay base salary ภายใต้เงื่อนไขเดียวกันคือ ประเภทลูกจ้างรายเดือน (S) จะทำให้สามารถลดจำนวนคอลัมน์ Rules ลงได้จาก 6 Rules เหลือเพียง 4 Rules เท่านั้น

ขั้นตอนการสร้างตารางการตัดสินใจ (Decision Tables) ในการสร้างตารางการตัดสินใจ มีขั้นตอนดังนี้

1. กำหนดเงื่อนไขการตัดสินใจ (Conditions) และกำหนดค่าที่สามารถเป็นไปได้ของเงื่อนไข โดยที่ค่าของเงื่อนไขที่สามารถเป็นไปได้นั้นอาจจะมีเพียง 2 ค่า คือ จริง (Y) กับเท็จ (N)
2. กำหนดกิจกรรมที่อาจจะเกิดขึ้น (Actions) เนื่องจากเงื่อนไขใดๆ
3. กำหนดจำนวน Rule หรือจำนวนคอลัมน์โดยวิธีการดังนี้
    3.1. จำนวน Rule หาได้จาก นำจำนวนค่าที่เป็นไปได้ของแต่ละเงื่อนไขมาคูณกัน
    3.2. จำนวนครั้งในการเขียนค่าที่เป็นไปได้ของเงื่อนไขที่ 1 ให้เขียนสลับกันไป
    3.3. จำนวนครั้งในการเขียนค่าที่เป็นไปได้ของเงื่อนไขที่ 2 แต่ละค่า ให้เขียนครอบคลุมค่าที่เป็นไปได้ของเงื่อนไขที่ 1

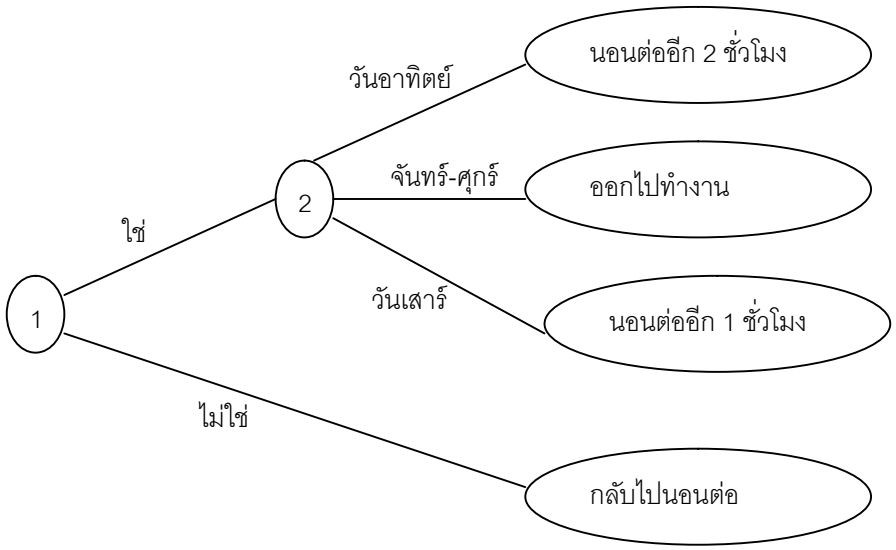
4. กำหนดกิจกรรมที่ต้องกระทำตามกฎ (Rule) แต่ละกฎที่กำหนดขึ้น โดยทำเครื่องหมายกากบาท (X) ในช่องนั้นๆ

5. พยายามเขียนตารางใหม่ให้กะทัดรัด โดยใช้กฎการลดคอลัมน์ที่ไม่จำเป็นลงด้วยในกรณีที่เกิด Indifferent Conditions

8.2.3. การตัดสินใจแบบต้นไม้ (Decision Tree)

เทคนิคที่ใช้ในการอธิบายการทำงานของ Process อีกวิธีหนึ่งที่สามารถทำความเข้าใจได้โดยง่าย คือ "การตัดสินใจแบบต้นไม้ (Decision Tree)"

Decision Tree คือ แผนภาพที่ใช้ในการอธิบายการทำงานของ Process ที่มีเงื่อนไขการตัดสินใจแสดงอยู่ในรูปของโหนด (Nodes) เชื่อมต่อกับเงื่อนไขการตัดสินใจอีกเงื่อนไขหนึ่งด้วยเส้นตรง โดยเส้นทางการตัดสินใจในแต่ละเงื่อนไขจะสิ้นสุดลงที่กิจกรรมซึ่งแสดงอยู่ในรูปวงรี



ส่วนประกอบของการตัดสินใจแบบต้นไม้

1. Decision Points เป็นจุดของเงื่อนไขการตัดสินใจ ซึ่งจะแสดงอยู่ในรูปของโหนด (Nodes)
2. Actions เป็นการกระทำที่อยู่ภายใต้จุดเงื่อนไขการตัดสินใจ ซึ่งจะแสดงอยู่ในรูปวงรี (Ovals) โดยเชื่อมต่อกับ Nodes ด้วยเส้นตรง

ขั้นตอนการสร้างการตัดสินใจแบบต้นไม้

การสร้างการตัดสินใจแบบต้นไม้ เริ่มต้นการแสดงเงื่อนไขการตัดสินใจแต่ละเงื่อนไขด้วยโหนด แต่ละโหนดจะมีหมายเลขกำกับ โดยแสดงคำอธิบายโหนดเงื่อนไขไว้ต่างหาก (Legends) และโหนดแรกจะเรียกว่า "Root Nodes" ค่าที่เป็นไปได้ของเงื่อนไขจะแตกแขนงออกไปเป็นเส้นตรงตามจำนวนค่าที่เป็นไปได้ โดยที่แต่ละโหนดจะต้องมีค่าที่เป็นไปได้อย่างน้อยที่สุด 2 ค่า และสิ้นสุดลงที่การกระทำที่เกิดขึ้นตามเงื่อนไขและค่าที่เป็นไปได้ ซึ่งการกระทำนั้นแสดงไว้ในรูปวงรี

สรุป

ในบทเรียนนี้เป็นการใช้เทคนิคในการเขียนคำอธิบายขั้นตอนการทำงานของระบบ (Process) ที่ปรากฏอยู่บนแผนภาพกระแสข้อมูล (Data Flow Diagram:DFD) ซึ่งในส่วนของ DFD เองไม่สามารถแสดงรายละเอียดลักษณะการทำงานของ Process ที่ต้องมีการตัดสินใจได้ จึงต้องอาศัยเทคนิคในการอธิบายลักษณะการทำงานดังกล่าวได้แก่ ภาษาอังกฤษแบบโครงสร้าง (Structured English) ตารางการตัดสินใจ (Decision Table) และการตัดสินใจแบบต้นไม้ (Decision Tree)

Structured English เป็นการเขียนเงื่อนไขและการกระทำด้วยภาษาอังกฤษในลักษณะที่คล้ายกับการเขียนโปรแกรมด้วยภาษาชนิดต่างๆ ซึ่งจะทำให้โปรแกรมเมอร์สามารถอ่านการทำงานได้ง่ายขึ้น

Decision Table เป็นการแสดงเงื่อนไขการตัดสินใจ (Conditions) การกระทำ (Actions) และกฎเกณฑ์ในการตัดสินใจ (Rules) ในรูปแบบตาราง เป็นเทคนิคที่รองรับการทำงานที่มีเงื่อนไขซับซ้อนได้โดยไม่ทำให้ผู้อ่านสับสน

Decision Tree เป็นการแสดงเงื่อนไขการตัดสินใจในรูปแบบโหนด แสดงการกระทำที่เป็นผลลัพธ์จากการทดสอบเงื่อนไขต่างๆในรูปวงรี และเชื่อมต่อกันด้วยเส้นตรงเป็นเส้นทางที่แตกแขนงคล้ายกับต้นไม้ เทคนิคชนิดนี้ทำให้ผู้ใช้ระบบสามารถเข้าใจการทำงานได้โดยง่าย

บทที่ 9

แบบจำลองข้อมูล (Data Modeling)

9.1. แนะนำแผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (Entity Relationship Diagram : E-R Diagram)

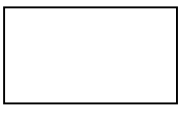
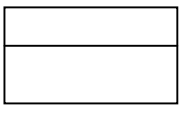
การสร้างแผนภาพจำลองข้อมูลและกระบวนการดำเนินงานนั้นมีบทบาทสำคัญในการพัฒนาระบบ เนื่องจากสามารถแสดงโครงสร้างของข้อมูลและการทำงานภายในระบบได้ชัดเจน ซึ่งจะช่วยให้ทั้งนักวิเคราะห์ระบบและผู้ใช้งานเกิดความเข้าใจในการทำงานของระบบอย่างถูกต้อง แบบจำลองข้อมูลที่สร้างขึ้นในขั้นตอนการวิเคราะห์ความต้องการของระบบนี้ยังเรียกว่าเป็น "การออกแบบฐานข้อมูลในระดับแนวคิด (Conceptual Database Design)" ของขั้นตอนการออกแบบ (Design Phase) ในกิจกรรมการออกแบบฐานข้อมูลซึ่งจะนำ Conceptual Data Model ที่ได้จากกิจกรรมย่อยนี้ไปทำการปรับปรุงและออกแบบฐานข้อมูลในระดับ Logical และ Physical ต่อไป ในที่นี้เพื่อความสะดวกจะเรียก Conceptual Data Model ว่า "Data Model"

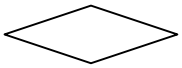
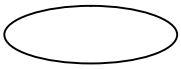
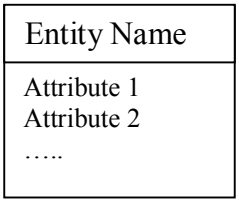
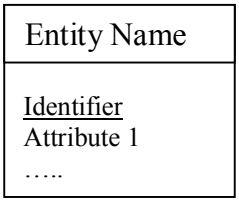
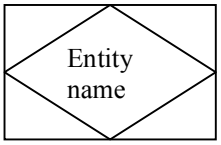
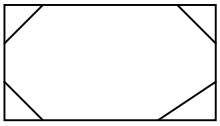
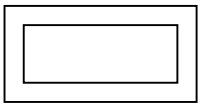
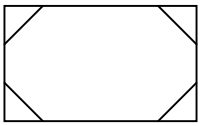
แบบจำลองข้อมูล (Data Model) หมายถึง การจำลองข้อมูลที่เกิดขึ้นทั้งหมดในระบบ พร้อมทั้งจำลองความสัมพันธ์ระหว่างข้อมูลที่เกิดขึ้นนั้น โดยใช้ "แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (Entity Relationship Diagram: E-R Diagram)"

แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (E-R Diagram) หมายถึง แผนภาพที่ใช้เป็นเครื่องมือสำหรับจำลองข้อมูลซึ่งจะประกอบไปด้วย Entity (แทนกลุ่มของข้อมูลที่เป็นเรื่องเดียวกัน/เกี่ยวข้องกัน) และความสัมพันธ์ระหว่างข้อมูล (Relationship) ที่เกิดขึ้นทั้งหมดในระบบ

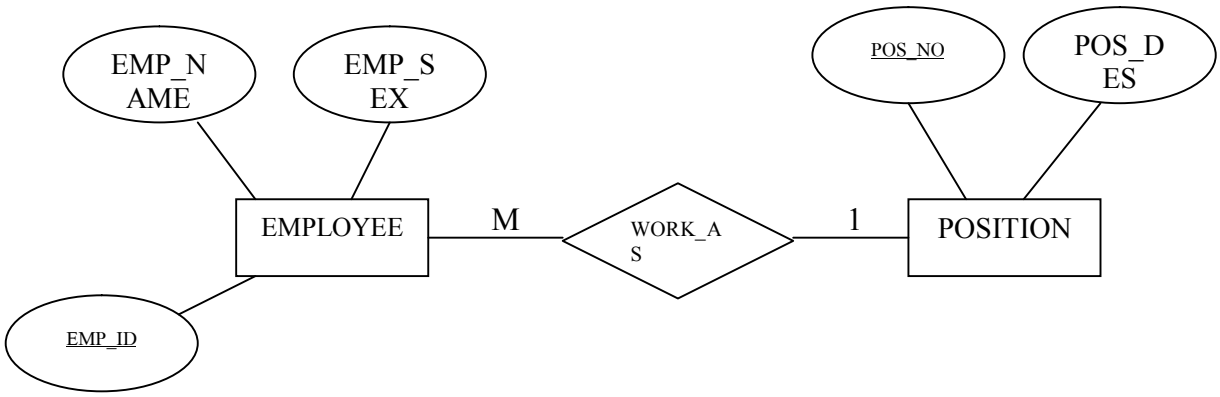
9.2. สัญลักษณ์ที่ใช้ใน E-R Diagram

สัญลักษณ์ที่ใช้ในแผนภาพ E-R Diagram ที่ใช้ในการจำลองแบบข้อมูลมีหลายรูปแบบ ในที่นี้ขอยกตัวอย่าง 2 รูปแบบ ได้แก่ Chen Model และ Crow's Foot Model สำหรับหนังสือเล่มนี้จะเลือกใช้แบบ Chen Model

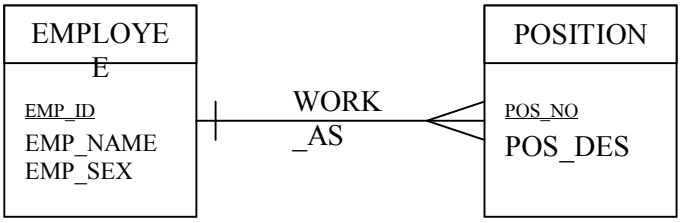
Chen Model	Crow's Foot Model	ความหมาย
		ใช้แสดง Entity
		Relationship Line เส้นเชื่อมความสัมพันธ์ระหว่าง Entity

	-	Relationship ใช้แสดงความสัมพันธ์ระหว่าง Entity สำหรับ Crow's Foot Model ใช้ตัวอักษรเขียนแสดงความสัมพันธ์
		Attribute ใช้แสดง Attribute ของ Entity
		ใช้แสดงคีย์หลัก (Identifier)
		Associative Entity
		Weak Entity

ตัวอย่าง E-R Diagram ของรูปแบบ Chen Model ดังรูป



ตัวอย่าง E-R Diagram ของรูปแบบของ Crow's Foot Model ดังรูป



9.3. องค์ประกอบของ E-R Diagram

การสร้างแผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (E-R Diagram) นั้นมีองค์ประกอบต่างๆ ดังนี้

- Entities
- Attributes
- ความสัมพันธ์ระหว่าง Entity (Relationship)
- ระดับความสัมพันธ์ระหว่าง Entity(Degree of Relationship)
- Cardinalities ใน Relationship
- Associative Entities
- Generalization Hierachy
- Aggregation

9.3.1. Entities

Entity หมายถึง องค์ประกอบส่วนหนึ่งของ E-R Diagram ที่ใช้สำหรับเก็บข้อมูลแต่ละรายการที่มีคุณสมบัติร่วมกันภายใต้ขอบเขตของระบบหนึ่งที่กำลังสนใจ เช่น ระบบโรงเรียน ซึ่งประกอบไปด้วย Entity นักเรียน (Student) Entity อาจารย์ (Teacher), Entity หลักสูตร (Course), Entity ห้องเรียน (Room) เป็นต้น โดยที่ Entity นักเรียนจะถูกบรรยายด้วยคุณสมบัติต่างๆ เช่น ชื่อ-สกุล (Name-Surname) ระดับชั้น (Level) เป็นต้น กล่าวได้ว่า Entity สามารถเป็นได้ทั้งสิ่งที่จับต้องได้และสิ่งที่จับต้องไม่ได้ในระบบ Entity ที่รวบรวมได้จากระบบสามารถแยกแยะและจัดเป็นหมวดหมู่ได้ตามชนิดของ Entity ได้ เช่น

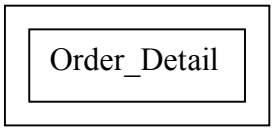
1. หมวดบุคคล (Person):
   EMPLOYEE,STUDENT,PATIENT,CUSTOMER,DEPARTMENT,DIVISION

2. หมวดสถานที่ (Place):
   STATE,REGION,COUNTRY,BRANCH,BUILDING,ROOM,CAMPUS
3. หมวดเหตุการณ์ (Event):
   SALE,REGISTRATION,RENEWAL,ORDER,INVOICE,FLIGHT,CANCELLATION
4. สิ่งของ (Object): Machine, Building , Automobile, Product
5. หมวดของแนวคิด : Account, Course, Work, Center

ใน E-R Diagram สามารถจำแนก Entity ได้ 2 ประเภท ดังนี้

1. Regular Entity : หรือบางครั้งเรียกว่า Strong Entity เป็น Entity ที่ประกอบด้วยสมาชิกที่มีคุณสมบัติ ซึ่งบ่งบอกถึงเอกลักษณ์ของแต่ละสมาชิกนั้น เช่น Entity ประชากร (POPULATION) ซึ่งสมาชิกภายใน Entity นี้ได้แก่ ประชากรแต่ละคนในประเทศไทยที่มีหมายเลขบัตรประชาชนไม่ซ้ำกันเลย เป็นต้น สำหรับสัญลักษณ์ที่ใช้แทน Entity ประเภทนี้ คือรูปสี่เหลี่ยมผืนผ้า โดยมีชื่อของ Entity นั้นอยู่ภายใน ดังรูป

2. Weak Entity : คือ Entity ที่มีลักษณะตรงกันข้ามกับ Regular Entity กล่าวคือ สมาชิกของ Entity ประเภทนี้ จะสามารถมีคุณสมบัติที่บ่งบอกถึงเอกลักษณ์ของแต่ละสมาชิกได้นั้น จะต้องอาศัยคุณสมบัติใดคุณสมบัติหนึ่งของ Regular Entity มาประกอบกับคุณสมบัติของ Weak Entity เอง เช่น "Order_Detail" ซึ่งสมาชิกของ Entity นี้ได้แก่ รายละเอียดของสินค้าที่สั่งซื้อภายใต้ใบสั่งซื้อแต่ละใบ ซึ่งเมื่อพิจารณาดูจะพบว่า สินค้า ก อาจถูกสั่งซื้อในใบสั่งซื้อได้หลายใบ ดังนั้น ถ้าระบุเพียงต้องการทราบจำนวนของสินค้า ก จะไม่สามารถทราบได้ว่าต้องการทราบจำนวนสินค้า ก ในใบสั่งซื้อใด แต่ถ้ามีการระบุเลขที่ใบสั่งซื้อประกอบกับสินค้า ก แล้ว จะสามารถทราบได้ทันทีว่าหมายถึงจำนวนของสินค้า ก ในใบสั่งซื้อใด ซึ่งเลขที่ใบสั่งซื้อนี้คือคุณสมบัติของ Regular Entity ที่นำมาประกอบกับคุณสมบัติของ Weak Entity "ORDER_DETAIL" เพื่อทำให้สมาชิกของ Entity นี้สามารถมีคุณสมบัติที่บ่งบอกถึงเอกลักษณ์ของแต่ละสมาชิกได้ สำหรับสัญลักษณ์ที่ใช้แทน Entity ประเภทแสดงดังรูป



9.3.2. Attributes

Attributes (Property/Element/Field) หมายถึง คุณสมบัติหรือลักษณะของ Entity หรือ Relationship ที่สนใจ

ตัวอย่างที่ 1 Entity "บัตรประชาชน" จะมีคุณสมบัติหรือมีลักษณะ (Attributes) ดังนี้

- หมายเลขบัตรประชาชน
- ชื่อ-สกุล
- วันเดือนปีเกิด
- ภูมิลำเนา
- วันที่ออกบัตร
- วันที่บัตรหมดอายุ เป็นต้น

ตัวอย่างที่ 2 Entity "Employee" มี Attribute ที่ทำให้ทราบว่าถ้ามีสิ่งเหล่านี้แล้วจึงเรียกได้ว่า "Employee" ได้แก่

- Employee_ID
- Employee_Name
- Address
- Skill

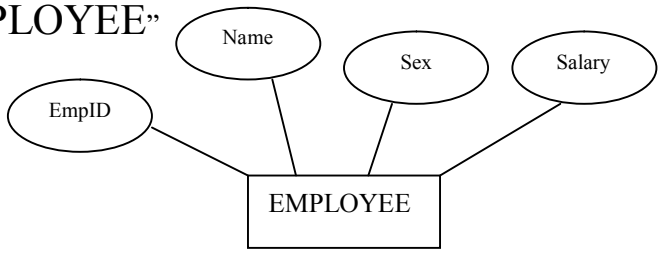
สำหรับ Attributes สามารถจำแนกได้เป็น 6 ประเภทดังนี้

9.3.2.1. Simple Attribute
9.3.2.2. Composite Attribute
9.3.2.3. Identifier/Key
9.3.2.4. Single-Valued Attribute
9.3.2.5. Multi-Valued Attribute
9.3.2.6. Derived Attribute

9.3.2.1. Simple Attribute

Simple Attribute คือ Attribute ที่ค่าภายใน Attribute นั้นไม่สามารถแบ่งย่อยได้อีก เช่น เพศ, เงินเดือน, อายุ, จังหวัด เป็นต้น

สำหรับสัญลักษณ์ที่ใช้แทน Attribute ประเภทนี้ ได้แก่ วงรีที่มีเส้นเชื่อมต่อไปยัง Entity ที่เป็นเจ้าของ Attribute นั้น โดยมีชื่อของ Attribute นั้นอยู่ภายใน เช่น Attribute "EmpID","NAME","SEX" และ "SALARY" ของ Entity "EMPLOYEE"

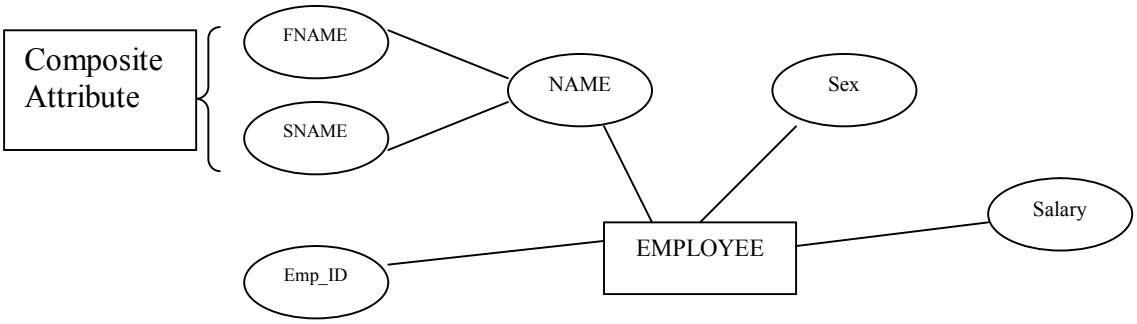


9.3.2.2. Composite Attribute

Composite Attribute คือ Attribute ที่ค่าภายใน Attribute นั้น สามารถแยกเป็น Attribute ย่อยได้อีก ซึ่งมีลักษณะตรงข้ามกับ Simple Attribute

ตัวอย่างที่ 1 Attribute “ชื่อ” ที่สามารถแบ่งย่อยออกเป็น “คำนำหน้าชื่อ”, “ชื่อ” และ “นามสกุล”

ตัวอย่างที่ 2 Attribute “ที่อยู่” ที่สามารถแบ่งย่อยออกเป็น “เลขที่บ้าน”, “ซอย” ,” ถนน” ,”แขวง/ตำบล”, “เขต/อำเภอ”,”จังหวัด” เป็นต้นดังรูป



9.3.2.3. Identifier/Key

Identifier หรือ Key คือ Attribute หรือกลุ่มของ Attribute ที่มีค่าในแต่ละ Attribute ของ Entity ไม่ซ้ำกันเลย ซึ่งถูกนำมาใช้กำหนดความเป็นเอกลักษณ์ให้กับแต่ละ Attribute ใน Entity

ตัวอย่าง Attribute “EmpID” ของ Entity “Employee” ที่ใช้แทนรหัสประจำตัวพนักงาน ซึ่งโดยทั่วไปแล้วการเก็บรหัสของพนักงานในองค์กรต่างๆ ค่าของรหัสพนักงานจะไม่มีรหัสพนักงานคนใดที่ซ้ำกันเลย

Identifier/Key สามารถจำแนกได้ 3 ประเภทดังนี้

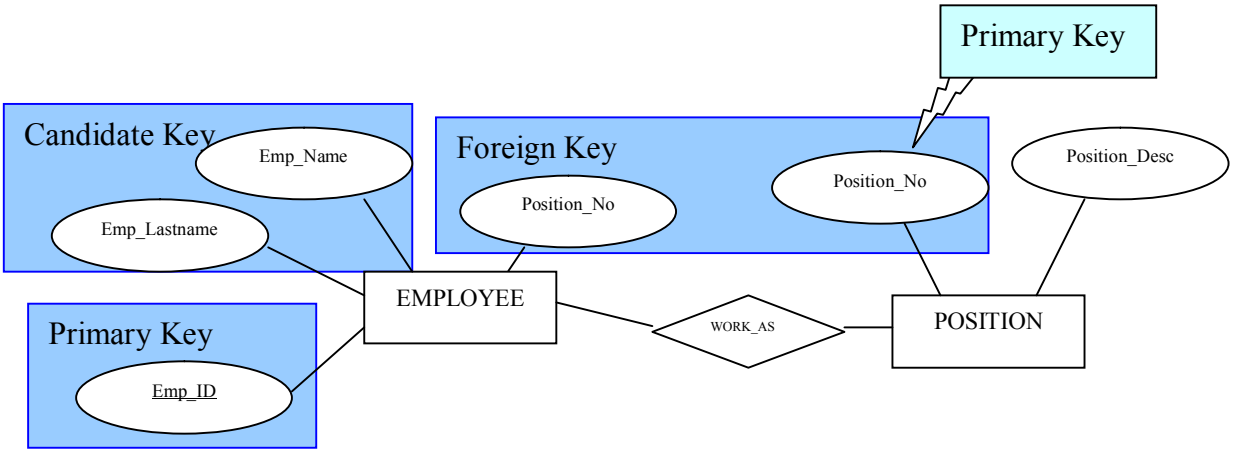
1. Candidates Keys
2. Primary Key
3. Foreign Key

1. **<u>Candidate Keys</u>** คือ Attribute ใดๆ หรือ Attribute ที่รวมกันแล้วทำให้ค่าของ Attribute ของ Entity นั้นไม่ซ้ำกันเลย เช่น

Entity “Employee” มี Candidate Key คือ “Employee_Name” และ “Employee_Lastname” ซึ่งหากมี Candidate Key เป็น “Employee_Name” เพียง Attribute เดียว จะทำให้เกิดข้อมูลซ้ำซ้อน กล่าวคือชื่อของพนักงานอาจจะเกิดการซ้ำกันได้ ดังนั้นเมื่อระบุชื่อของพนักงานที่ต้องการ

ให้แสดงรายงานออกมาเพียงคนเดียว แต่ผลที่ออกมาจะมีพนักงานหลายคนที่มีชื่อเดียวกัน แต่คนละนามสกุลกัน ดังนั้นเพื่อให้เกิดการไม่ซ้ำกันของค่าของ Attribute จึงกำหนด Candidate Key เป็น “Employee_Name” และ “Employee_Lastname

2. Primary Key คือ Candidate Key ที่ถูกเลือกให้เป็น Key หลักที่มีค่าของสมาชิกใน Attribute ไม่ซ้ำกันเลย การที่เลือก Key ที่มีค่าไม่ซ้ำกันเลยมาเป็น Primary Key เพื่อจะให้ Primary Key นี้สามารถไประบุค่าในอีก Attribute อื่นเพื่อประโยชน์ในการค้นหาข้อมูลได้โดยไม่เกิดข้อมูลซ้ำซ้อนกัน
3. Foreign Key คือ Primary Key ของ Entity หนึ่งที่สามารถระบุค่าสมาชิกของอีก Entity หนึ่งที่มีความสัมพันธ์กันได้ แสดงดังรูป

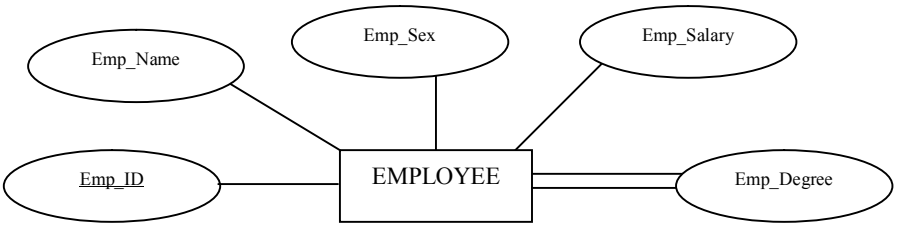


9.3.2.4. Single-Valued attribute

Single-Valued Attribute คือ Attribute ที่มีค่าของข้อมูลภายใต้ Attribute ใด Attribute หนึ่งเพียงค่าเดียว เช่น Attribute “Salary” ซึ่งที่ใช้เก็บเงินเดือนของพนักงาน และพนักงานแต่ละคนจะมีเงินเดือนเพียงค่าเดียว

9.3.2.5. Multi-Valued Attribute

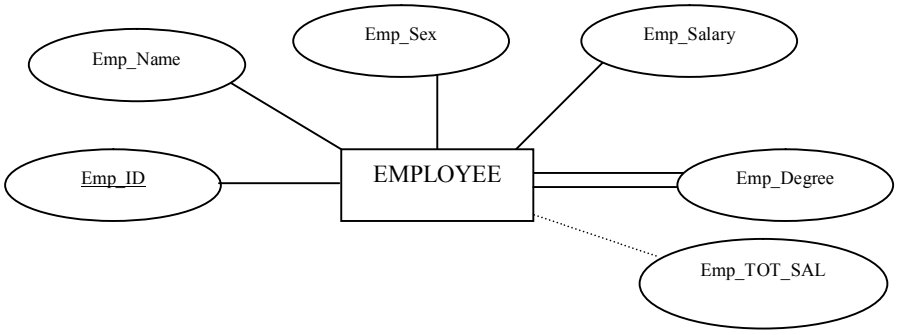
Muti-Valued Attribute คือ Attribute ที่มีค่าของข้อมูลได้หลายค่าภายใต้ค่าของ Attribute ใด Attribute หนึ่งเช่น Attribute “DEGREE” ที่ใช้ระบุระดับการศึกษาของพนักงานแต่ละคน ซึ่งพนักงานแต่ละคน จะมีระดับการศึกษาได้หลายระดับ สำหรับสัญลักษณ์ที่ใช้แทน Attribute ประเภทนี้ จะใช้เส้น 2 เส้นเชื่อมระหว่างรูปภาพของ Attribute กับ Entity ดังรูป



สำหรับกลุ่มข้อมูลที่สามารถมีหลายค่าได้นี้ (Multi-Valued Attribute) เรียกอีกอย่างหนึ่งว่า “Repeating Group”

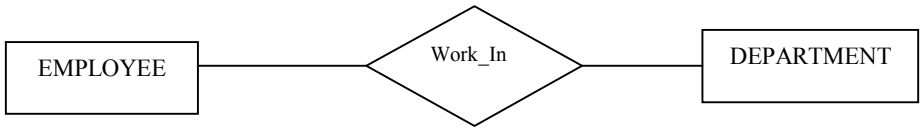
9.3.2.6. Derived Attribute

Derived Attribute คือ Attribute ที่ค่าของข้อมูลได้มาจากการนำเอาค่าของ Attribute อื่นมาทำการคำนวณ ซึ่งค่าของ Attribute ประเภทนี้จะต้องเปลี่ยนแปลงทุกครั้ง เมื่อมีการเปลี่ยนแปลงค่าของ Attribute ที่ถูกนำค่ามาคำนวณ
เช่น Attriute “TOT_SAL” ของ Entity “Employee” ที่ใช้เก็บเงินเดือนทั้งหมดของพนักงานแต่ละคนของพนักงานแต่ละคนเพื่อนำไปคำนวณภาษี ซึ่งได้มาจากผลรวมของค่าใน Attribute “INCOME” ของ Entity “MTHLY_SALARY” ซึ่งเป็นเงินเดือนที่พนักงานแต่ละคนได้รับในแต่ละเดือน
สำหรับสัญลักษณ์ที่ใช้แทน Attribute ประเภทนี้จะใช้สัญลักษณ์เส้นปะเชื่อมระหว่าง Entity และ Attribute ดังรูป



9.3.3. ความสัมพันธ์ระหว่าง Entity (Relationship)

Relationship คือ ความสัมพันธ์ระหว่าง Entity 2 Entity ที่มีการเชื่อมโยงข้อมูลซึ่งกันและกัน สมาชิกของ Relationship จึงเกิดการจับคู่กันระหว่างสมาชิกของ Entity ที่มีการร่วมกันของ Relationship นั้น สำหรับสัญลักษณ์จะใช้รูปสี่เหลี่ยมข้าวหลามตัดที่มีชื่อของ Relationship นั้นอยู่ภายในสัญลักษณ์จะต้องเชื่อมระหว่าง Entity เสมอ ดังรูป

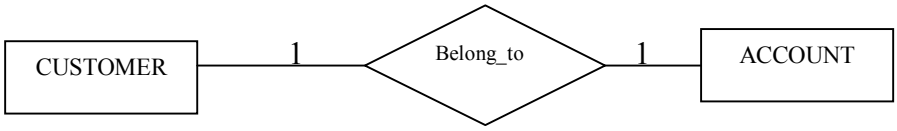


ประเภทของ Relationship ประเภทของ Relationship สามารถจำแนกได้ 3 ประการดังนี้

1. One-to-One Relationship
2. One-to-Many Relationship
3. Many-to-Many Relationship

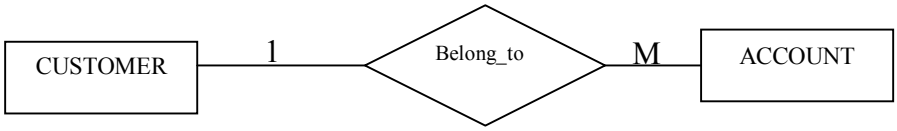
1. One-to-One Relationship : เป็น Relationship ที่แต่ละ Participant ของ Entity หนึ่งจะมีความสัมพันธ์กับอีก Participant ของอีก Entity หนึ่งเพียง Participant เดียว

เช่น กรณีลูกค้าสามารถมีบัญชีเงินฝากได้เพียงบัญชีเดียว และแต่ละบัญชีเงินฝากจะมีเจ้าของบัญชีได้เพียงคนเดียว ดังรูป



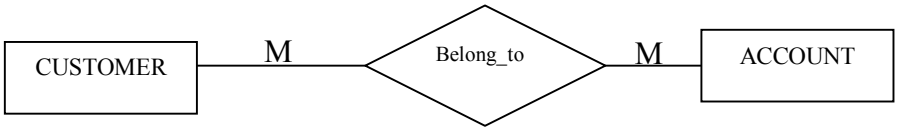
2. One-to-Many Relationship : เป็น Relationship ที่แต่ละ Participant ของ Entity หนึ่งมีความสัมพันธ์กับ Participant ของอีก Entity หนึ่งมากกว่า 1Participant

เช่นกรณีลูกค้าสามารถมีบัญชีเงินฝากได้มากกว่า 1 บัญชี และแต่ละบัญชีเงินฝากจะต้องมีเจ้าของบัญชีเพียงคนเดียว ดังรูป



3. Many-to-Many Relationship : เป็น Relationship ที่ Participant มากกว่า 1 Participant ของ Entity หนึ่ง มีความสัมพันธ์กับ Participant ของอีก Entity หนึ่งมากกว่า 1 Participant

เช่น กรณีลูกค้าสามารถมีบัญชีเงินฝากได้มากกว่า 1 บัญชี และแต่ละบัญชีเงินฝากสามาถมีเจ้าของบัญชีได้มากกว่า 1 คน

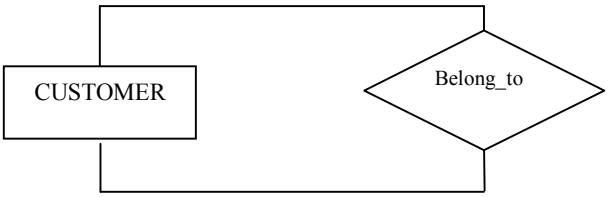


9.3.4. ระดับความสัมพันธ์ระหว่าง Entity (Degree of a Relationship)

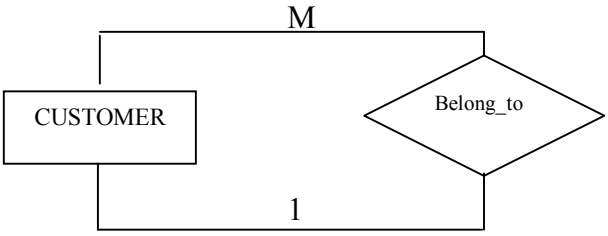
Entiy คือสิ่งที่สนใจในระบบ ซึ่งอาจจะเป็นข้อมูล สิ่งของ แผนก หรือสถานที่ ซึ่งจะต้องมีความสัมพันธ์กับอีก Entity หนึ่งเพื่อให้ระบบเกิดการทำงานเป็นตามขั้นตอน ดังนั้นจึงต้องมีสิ่งที่ใช้วัดความเข้มข้นของความสัมพันธ์ระหว่าง Entity ว่ามีความสัมพันธ์กันลักษณะอย่างไรหรือมีความสัมพันธ์ที่ซับซ้อนเพียงใด ซึ่งการวัดจำนวน Entity ที่มีความสัมพันธ์กันนั่นเอง ที่เรียกว่า Degree of a Relationship คือ ขนาดของความสัมพันธ์ระหว่าง Entity สามารถจำแนกได้เป็น 3 ขนาด ได้แก่

1. Unary Relationship/Recursive Relationship : เป็นความสัมพันธ์ที่เกิดขึ้นระหว่างสมาชิกภายใน Entity ของตัวเองซึ่งเกิดในกรณีที่ Attribute ของ Entity นั้น สามารถสร้างความสัมพันธ์กับอีก Attribute หนึ่งภายใน Entity เดียวกัน

ตัวอย่าง ความสัมพันธ์แบบ One-to-One จะเห็นว่า Person หนึ่ง Person จะสามารถแต่งงาน (IS Married to) กับอีก Person ได้เพียงคนเดียวในแต่ละครั้ง ดังรูป



ตัวอย่าง ความสัมพันธ์เป็นแบบ One-to-Many ซึ่งจะเห็นได้ว่า Employee หนึ่งคนสามารถจะบริหารงาน Employee คนอื่นๆได้ เช่นหัวหน้างาน ดังรูป



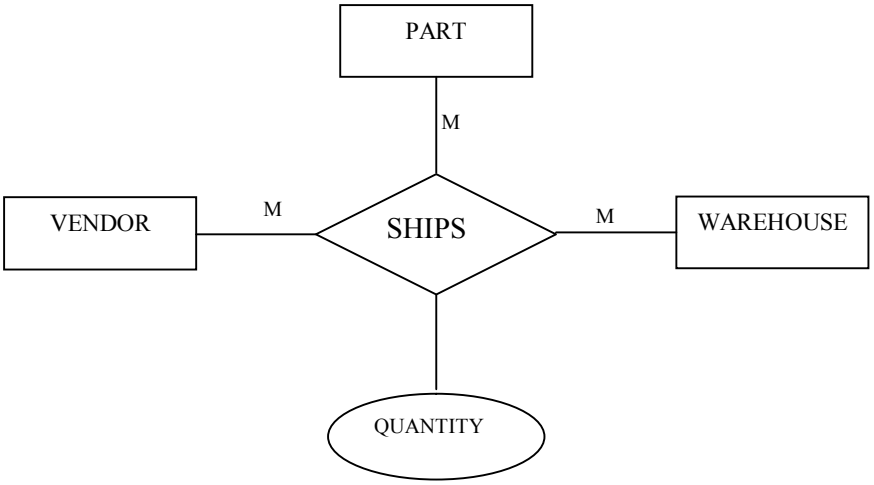
2. Binary Relationship : คือ Relationship ที่เกิดขึ้นระหว่าง 2 Entity กรณีเช่นนี้เรียกได้ว่ามี Degree ของความสัมพันธ์เท่ากับ 2 เนื่องจากเป็นความสัมพันธ์ระหว่าง Entity 2 จำนวน

ตัวอย่าง ความสัมพันธ์แบบ One-to-One Relationship ซึ่งพนักงานหนึ่งคนจะสามารถมีที่จอดรถ (Parking Place) ได้ 1 ที่เท่านั้นและที่จอดรถ 1 ที่เป็นของพนักงานหนึ่งคนดังรูป

ตัวอย่าง แบบ One-to-Many ในหนึ่งสายผลิตภัณฑ์ (Product Line) จะประกอบไปด้วย (Contains) สินค้าได้ตั้งแต่ 1 ผลิตภัณฑ์ (Product) ขึ้นไป และสินค้าตั้งแต่ 1 ผลิตภัณฑ์ขึ้นไปจะต้องอยู่ในสายการผลิตเพียง 1 สายเท่านั้น ดังรูป

ตัวอย่าง แบบ Many-to-Many คือ นักศึกษา (Student) ตั้งแต่ 1 คนขึ้นไปสามารถลงทะเบียน (Register for) เรียนในรายวิชา (Course) ได้ตั้งแต่ 1 รายวิชาขึ้นไป และในรายวิชาตั้งแต่ 1 รายวิชาขึ้นไปนี้นักศึกษาลงทะเบียนเรียนได้มากกว่า 1 คน ดังรูปด้านล่าง

3. Ternary Relationship : คือ Relationship ที่เกิดขึ้นระหว่าง Entity มากกว่า 2 Entity ขึ้นไป
   ตัวอย่าง Entity ความสัมพันธ์กัน 3 Entity ด้วยกันได้แก่ PART,VENDOR และ WAREHOUSE ซึ่งมีความสัมพันธ์ในส่วนของการส่งสินค้า ผู้จัดจำหน่าย (VENDOR) สามารถส่งชิ้นส่วนสินค้า (PART) ได้ตั้งแต่ 1 ชิ้นส่วนขึ้นไป เพื่อไปเก็บไว้ในคลังสินค้า (WAREHOUSE) ได้ตั้งแต่ 1 คลังสินค้าขึ้นไป ดังรูป



9.3.5. Cardinalities ใน Relationships

Cardinality หมายถึง จำนวนสมาชิกที่เป็นไปได้ใน Entity หนึ่งที่มีความสัมพันธ์กับสมาชิกของอีก Entity หนึ่ง

ตัวอย่าง หากมี Entity "ภาพยนตร์(MOVIE)" และ"ม้วนวิดีโอ(VIDEO TAPE)" ซึ่งมีความสัมพันธ์กันคือ ภาพยนตร์จะถูกบันทึกไว้ในม้วนวิดีโอ(SAVE AS) โดยมีเงื่อนไขคือ ภาพยนตร์หนึ่งเรื่องสามารถบันทึกไว้ในม้วนวิดีโอได้อย่างน้อยที่สุด 1 ม้วน ส่วนม้วนวิดีโอ 1 ม้วน สามารถบันทึกภาพยนตร์ได้สูงสุด 1 เรื่องหรือไม่บันทึกเลยได้ แสดงได้ดังรูป

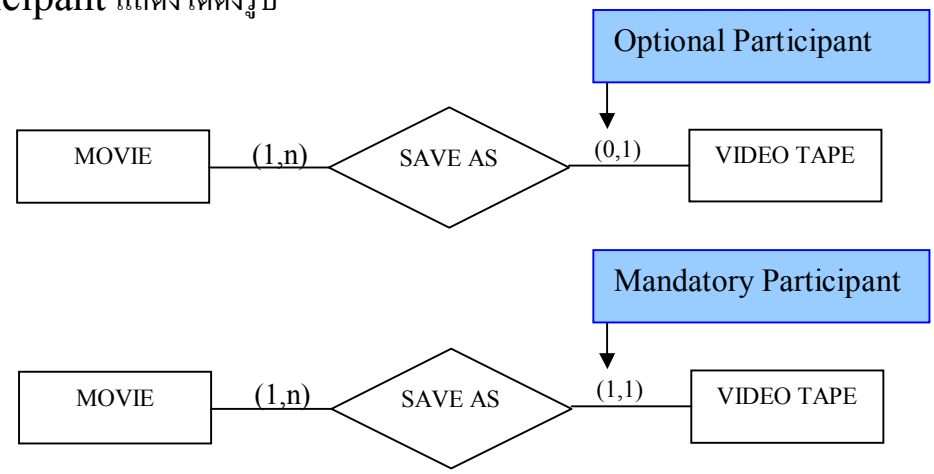
แสดงแบบ One-to-Many

แสดงแบบกำหนดจำนวนสมาชิกของทั้งสอง Entity

Mapping Cardinality หมายถึง การกำหนดขอบเขตหรือจำนวนสมาชิกของ Entity ใดๆ ที่มีความสัมพันธ์กัน โดยจะสามารถกำหนดได้ก็ต่อเมื่อทราบประเภทของ Relationship ระหว่าง Entity นั้นๆ ซึ่งเขียนอยู่ในรูปของคู่ลำดับคือ (min_card,max_card) ดังรายละเอียดต่อไปนี้

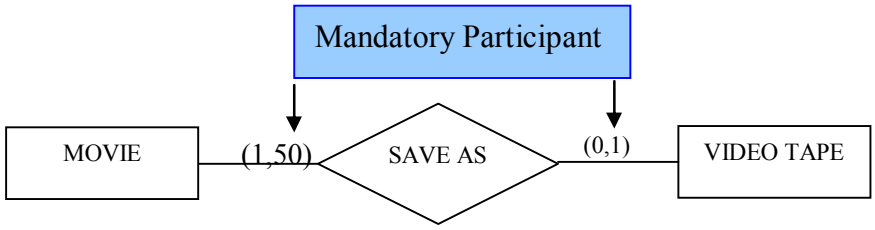
- Minimum Cardinality ของความสัมพันธ์ (min_card) คือ การกำหนดจำนวนสมาชิกน้อยที่สุดที่เป็นไปได้ของ Entity หนึ่งที่มีความสัมพันธ์กับสมาชิกของอีหนึ่ง Entity

ตัวอย่าง ความสัมพันธ์ระหว่าง Movie กับ Video Tape มีความสัมพันธ์แบบ One-to-Many คือ ภาพยนตร์ 1 เรื่องสามารถบันทึกไว้ในม้วนวิดีโอได้หลายม้วน แล้วสามารถกำหนดจำนวนสมาชิกที่น้อยที่สุดของ Entity ภาพยนตร์ และ Entity ม้วนวิดีโอ ทั้งนี้ขึ้นอยู่กับเงื่อนไขของแต่ละองค์กร เช่น ม้วนวิดีโอ 1 ม้วนไม่จำเป็นต้องบันทึกภาพยนตร์ไว้ กรณีเช่นนี้ จะเรียก Entity ม้วนวิดีโอว่า Optional Participant และกำหนดให้ภาพยนตร์ 1 เรื่องจะต้องบันทึกไว้ในม้วนวิดีโออย่างน้อยที่สุด 1 ม้วน กรณีเช่นนี้เรียกว่า Mandatory Participant แสดงได้ดังรูป



- Maximum Cardinality ของความสัมพันธ์ (max_card) คือ การกำหนดขอบเขตหรือจำนวนสมาชิกมากที่สุดที่เป็นไปได้ของ Entity หนึ่งที่มีความสัมพันธ์กับอีกหนึ่ง Entity

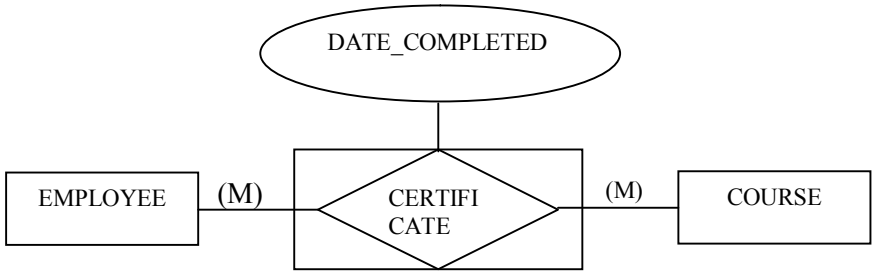
ตัวอย่าง หากเงื่อนไขกำหนดว่าม้วนวิดีโอ 1 ม้วนไม่จำเป็นต้องบันทึกภาพยนตร์ไว้เลยได้ แต่หากบันทึกจะต้องไม่เกิน 1 เรื่องต่อ 1 ม้วน และภาพยนตร์ 1 เรื่องจะต้องบันทึกไว้ในม้วนวิดีโออย่างน้อยที่สุด 1 ม้วนแต่ไม่เกิน 50 ม้วนของภาพยนตร์เรื่องนั้น แสดงดังรูป



9.3.6. Associative Entities

Associative Entity หมายถึง Relationship ที่มี Attribute เกิดขึ้นใหม่ โดยที่ Attribute นั้นเกิดจากความสัมพันธ์ระหว่าง Entity ตั้งแต่ 2 Entity ขึ้นไป ในสัญลักษณ์สี่เหลี่ยมข้าวหลามตัดที่ล้อมรอบด้วยสี่เหลี่ยมผืนผ้า

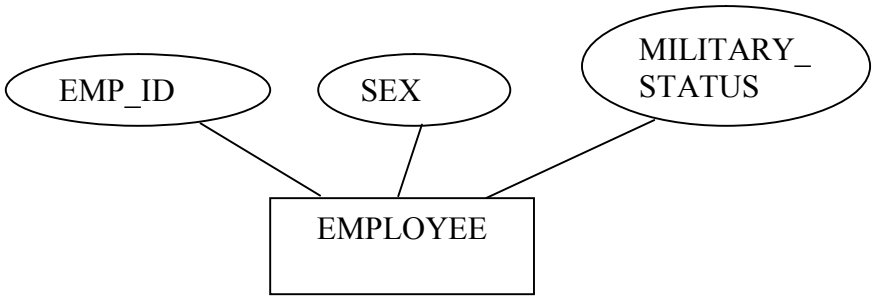
ตัวอย่าง ความสัมพันธ์ระหว่าง Employee และ Course กล่าวคือ Employee 1 คนสามารถสำเร็จหลักสูตรฝึกอบรมได้หลาย Course และ Course 1 Course จะมี Employee เข้าฝึกได้หลายคน (จะไม่มีพนักงานสำเร็จหลักสูตรได้ และ หลักสูตรจะไม่มีพนักงานเข้าอบรมเลยได้) ดังนั้น Attribute "DATE_COMPLETE" จะสามารถบอกให้ทราบได้ว่า "EMPLOYEE" คนใดสำเร็จหลักสูตร (Course) ใด และสำเร็จเท่าใด ดังนั้นเพื่อให้ E-R Diagram สมบูรณ์คือสามารถมองเห็น Entity ที่แอบแฝงมากับ Relationship ได้ จึงทำการแปลง Relationship นั้นให้เป็น Relationship ที่เรียกว่า Associative Entity ดังรูป



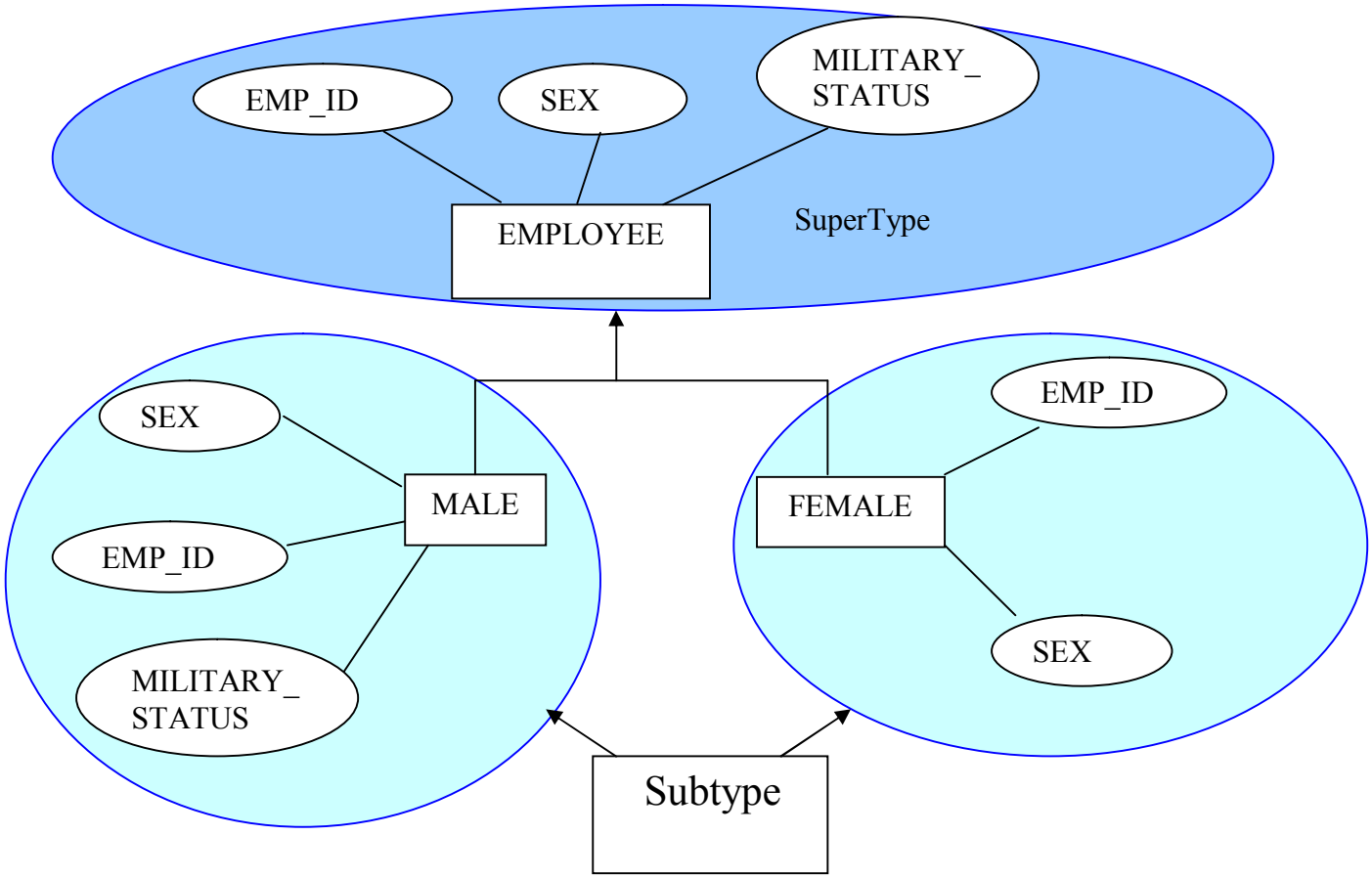
9.3.7. Generalization Hierarchy

Generalization Hierarchy เป็นการแสดงถึงการจัดลำดับของ Entity ที่มีความสัมพันธ์กันหรือ Relationship ที่มีความสัมพันธ์กัน ได้ถูกนำมาใช้กับ E-R Diagram เพื่อแสดงถึง Entity หรือ Relationship ซึ่งมีสมาชิกที่สามารถแยกออกเป็นกลุ่มย่อยๆ ภายใต้ Entity หรือ Relationship นั้น ดังนั้น Entity หรือ Relation นี้จึงเรียกว่า "Supertype Entity"

ตัวอย่าง Entity "EMPLOYEE" ที่มี Attribute "EMP_ID", "SEX" และ MILITARY_STATUS" ดังรูป

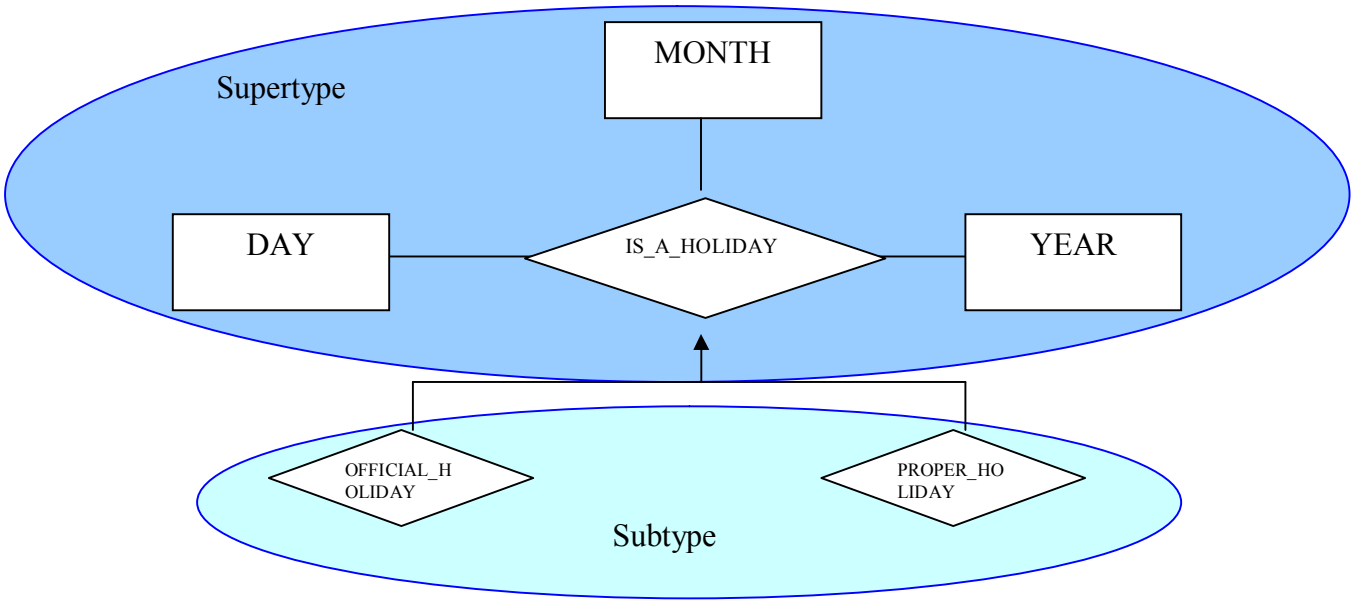


ซึ่ง Attribute “SEX” นั้นสามารถเป็นได้ 2 ค่าคือ MALE และ FEMALE และหากต้องการทราบว่าพนักงานชายคนใดที่ผ่านการเกณฑ์ทหารแล้วบ้าง ค่าที่ปรากฏจะมีเฉพาะพนักงานชายเท่านั้นที่มีข้อมูลของการเกณฑ์ทหาร ดังนั้น Entity “EMPLOYEE” จึงจำเป็นต้องมีสมาชิกหรือ Attribute ที่ต้องแยกออกเป็นกลุ่มย่อย ได้แก่ Entity “MALE” และ “FEMALE” ส่งผลให้ Entity “Employee” เป็น Supertype Entity และทำให้เกิด Subtype Entity ได้แก่ Entity “MALE” ที่ใช้สำหรับเก็บข้อมูลพนักงานชาย และ “FEMALE” ที่ใช้เก็บข้อมูลพนักงานหญิง แสดงดังรูป



ตัวอย่าง Generalization Hierarchy สามารถใช้กับ Relationship ได้ Relationship “IS_A_HOLIDAY” ที่สามารถแยกออกเป็น 2 Relationship

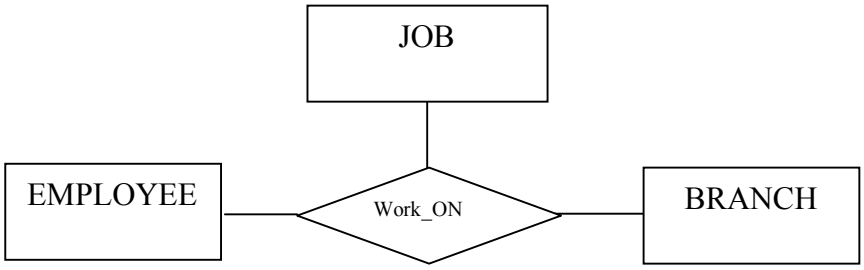
ย่อย คือ Relationship “OFFICIAL_HOLIDAY” ซึ่งเป็นวันหยุดประจำปี และ “PROPER_HOLIDAY” ซึ่งเป็นวันหยุดเฉพาะของบริษัท แสดงดังรูป



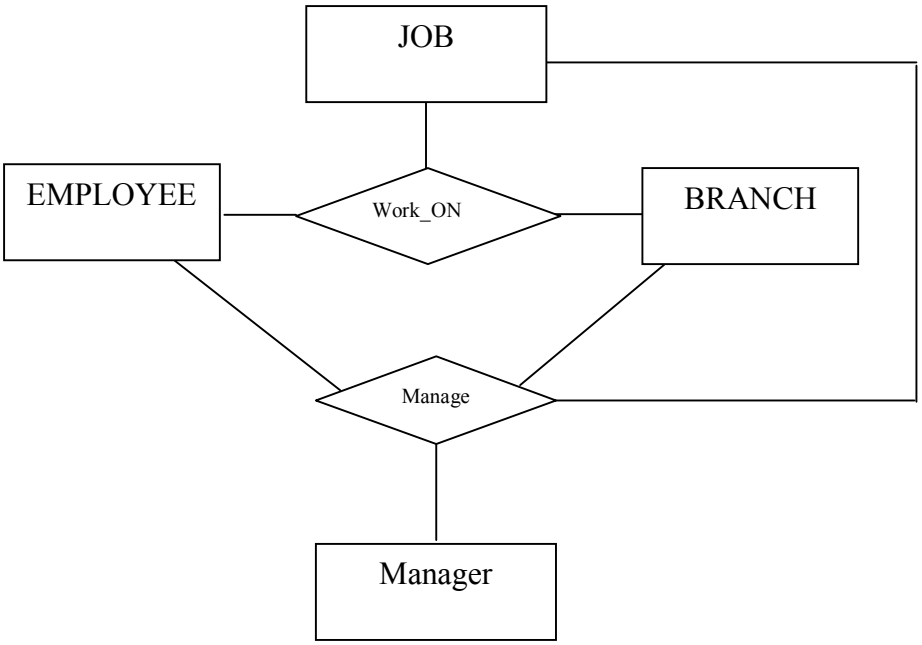
9.3.8. Aggregation

Aggreation คือ การทำให้ Relationship และ Entity ที่ทำให้เกิด Relationship นั้นอยู่ในภาวะรวมกลุ่มกันเสมือนเป็นอีก Entity หนึ่งเพื่อให้สามารถนำไปใช้สร้างความสัมพันธ์กับ Entity อื่นได้

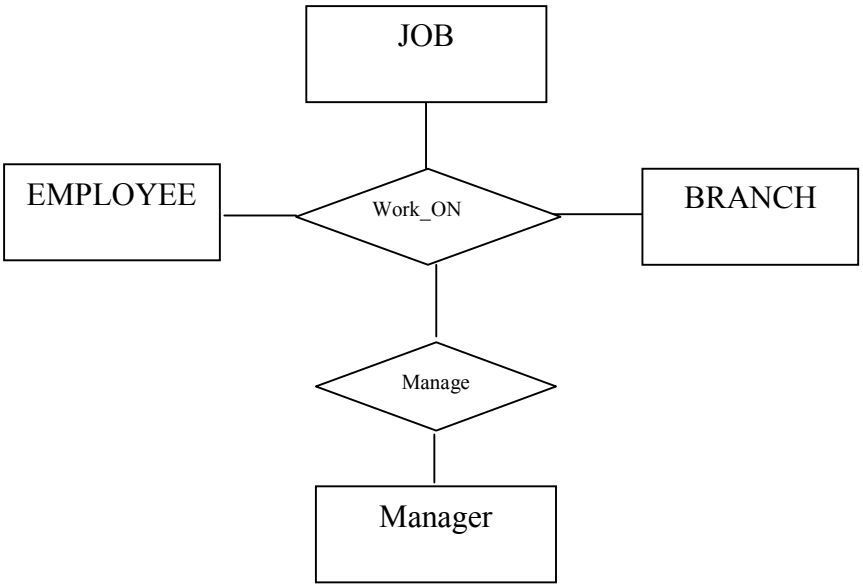
ตัวอย่าง Entity “Employee” “JOB” และ “BRANCH” ซึ่งเป็น Ternary Relationship มีความสัมพันธ์กันคือ พนักงาน (Employee) ทำงาน (Work_on) ที่ได้รับมอบหมายงาน (Job) ในแต่ละสาขา (Branch) ของสำนักงานดังรูป



สมมติว่ามีอีก Entity หนึ่งเพิ่มเข้ามาคือ “Manager” เพื่อบริหารงานดังกล่าว โดยมีความสัมพันธ์กับทุก Entity กล่าวคือคอยบริหารจัดการ “พนักงาน” แต่ละคน บริหารจัดการงานแต่ละ “สาขา” และบริหารจัดการ “งาน” ที่พนักงานต้องทำ ดังรูป



ฉะนั้นกรณีดังกล่าว E-R Diagram นั้นค่อนข้างซับซ้อน เนื่องจากเกิดความสัมพันธ์คาบเกี่ยวกัน (Redundancy) สามารถแก้ปัญหาโดยใช้วิธี Aggregation ดังรูป



ใช้วิธี Aggregation เพื่อทำให้เขียน E-R Diagram ได้ง่ายขึ้นมีหลักเกณฑ์ดังนี้

1. ให้สมมติ Relationship ที่เกิดจากกลุ่ม Entity นั้นเป็นเสมือน Entity 1 Entity โดยการวาดกรอบสี่เหลี่ยมล้อมรอบ Relationship และกลุ่ม Entity เข้าไว้ด้วยกัน
2. ลากเส้นตรงเชื่อมความสัมพันธ์ระหว่าง Relationship ที่ Aggregation แล้วกับ Relationship ที่เกิดจากความสัมพันธ์ของอีก Entity หนึ่ง
3. กลุ่ม Entity และ Relationship ที่ถูกรวมเข้าไว้ด้วยกันมีค่าเท่ากับ 1 Entity

ดังนั้น ความสัมพันธ์ที่ใช้วิธี Aggregation แล้วสามารถอ่านได้ดังนี้

1. พนักงานทำงานที่ได้รับมอบหมายตามสาขาที่กำหนดไว้
2. การทำงานที่ได้รับมอบหมายของพนักงานในแต่ละสาขาถูกบริหารจัดการโดยผู้จัดการ

9.4. วิธีการสร้าง E-R Diagram

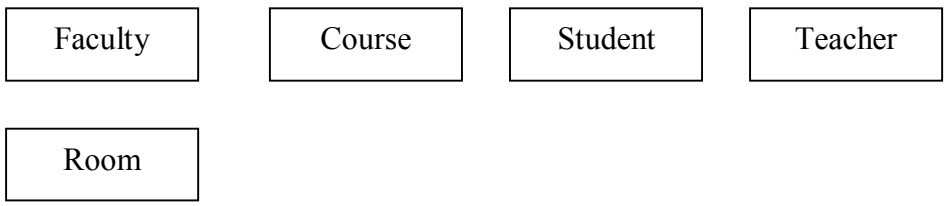
วิธีสร้าง E-R Diagram ตามลำดับดังนี้

- กำหนด Entity ทั้งหมดของระบบ
- สร้าง Relationship ระหว่าง Entity
- กำหนดเงื่อนไข (Contraints) ของความสัมพันธ์ระหว่าง Entity ต่างๆ
- กำหนด Attribute ให้กับแต่ละ Entity พร้อมทั้งกำหนด Primary Key

ตัวอย่าง การเก็บรวบรวมข้อมูลสมมติของมหาวิทยาลัยแห่งหนึ่ง เปิดสอบหลักสูตรปริญญาตรีหลายคณะแต่ละคณะเปิดสอนหลายรายวิชา ซึ่งทำการสอนโดยอาจารย์ที่มีคุณภาพ แต่ละรายวิชาจะสามารถเปิดสอนได้ต่อเมื่อมีนักศึกษามาลงทะเบียนในรายวิชานั้นอย่างน้อย 20 คน อาจารย์ 1 ท่านสามารถสอนได้หลายวิชา และห้องเรียนแต่ละห้องสามารถใช้สอนวิชาต่างๆ ได้หลายวิชา จากข้อมูลนี้สามารถสร้าง E-R Diagram โดย

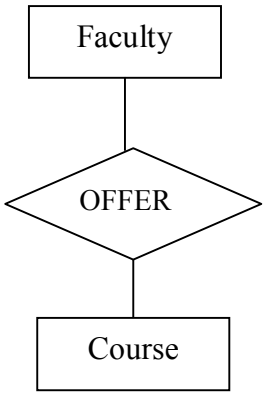
9.4.1. กำหนด Entity ทั้งหมดในระบบ ได้ดังนี้

1. คณะ (Faculty)
2. รายวิชา (Course)
3. อาจารย์ (Teacher)
4. นักเรียน (Student)
5. ห้องเรียน (Room)

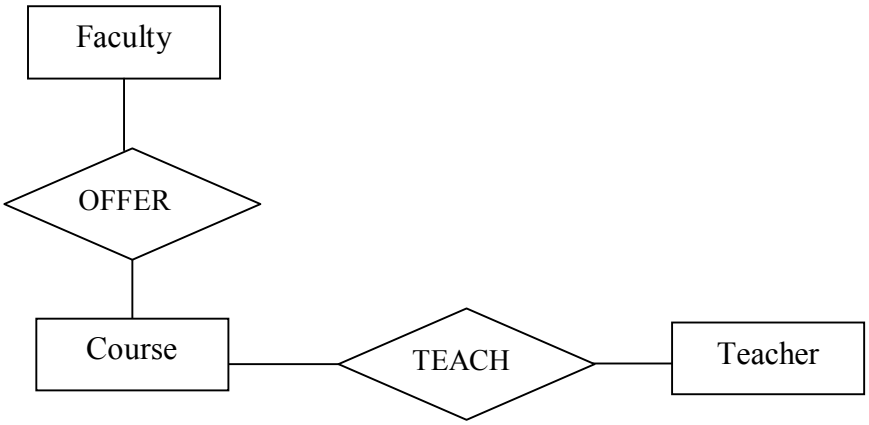


9.4.2. สร้าง Relationship ให้กับ Entity ได้ดังต่อไปนี้

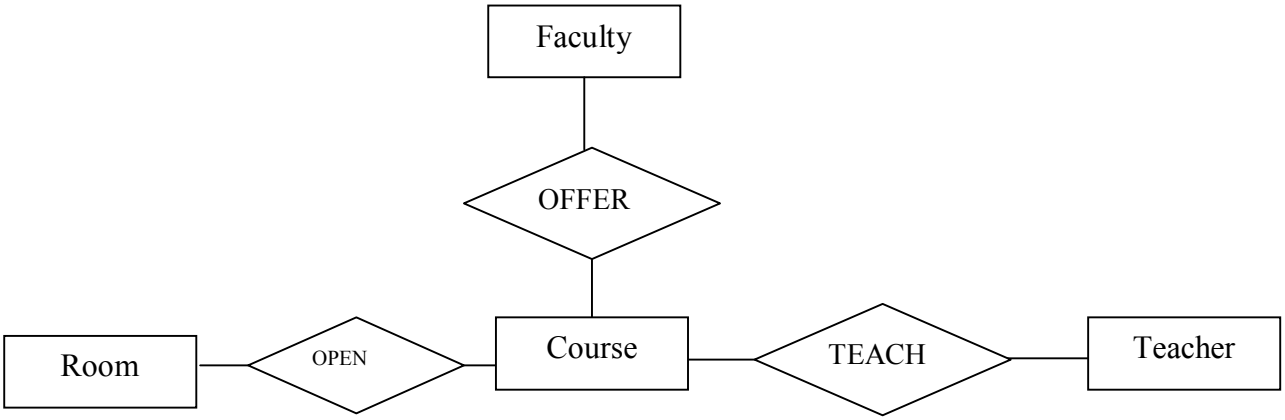
1. ทางคณะ (Faculty) จะต้องเสนอเรื่อง (Offer) เพื่อขอเปิดวิชาเรียน (Course)



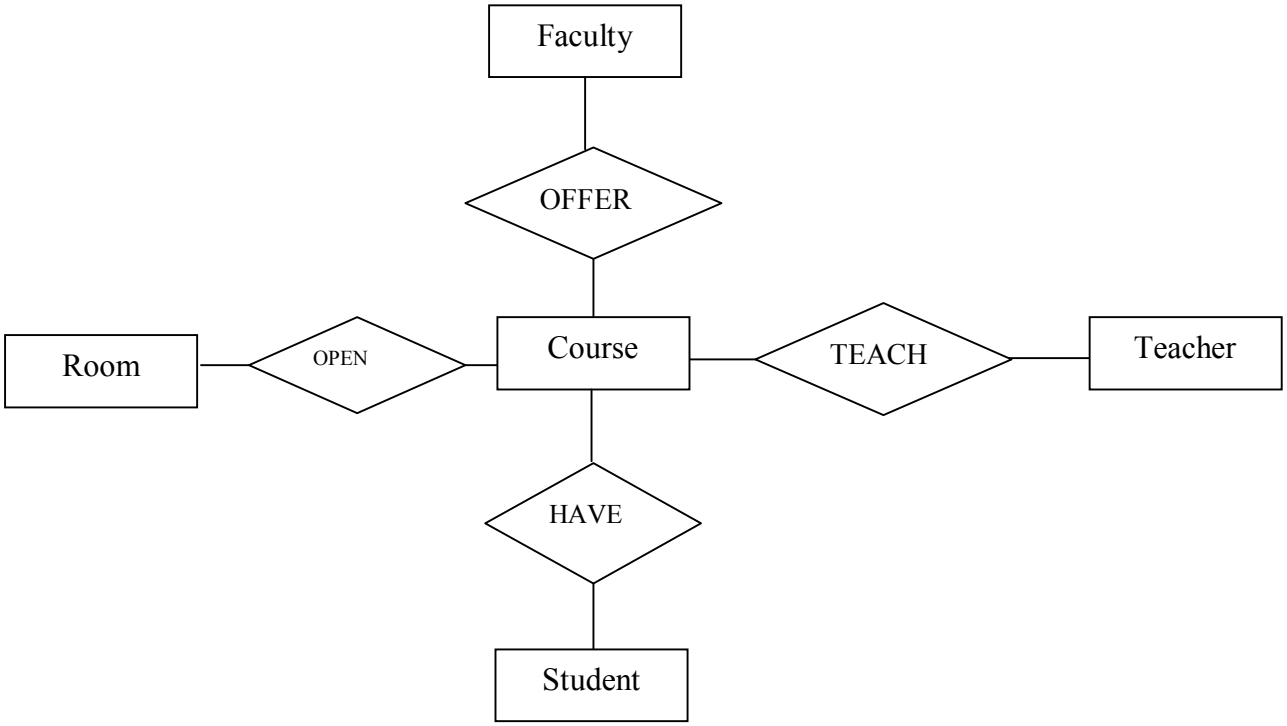
2. แต่ละรายวิชา (Course) จะต้องมีอาจารย์ (Teacher) เป็นผู้ทำการสอน (Teach)



3. ห้องเรียน (Room) เปิด (Open) ทำการเรียนการสอนทุกรายวิชา (Course)

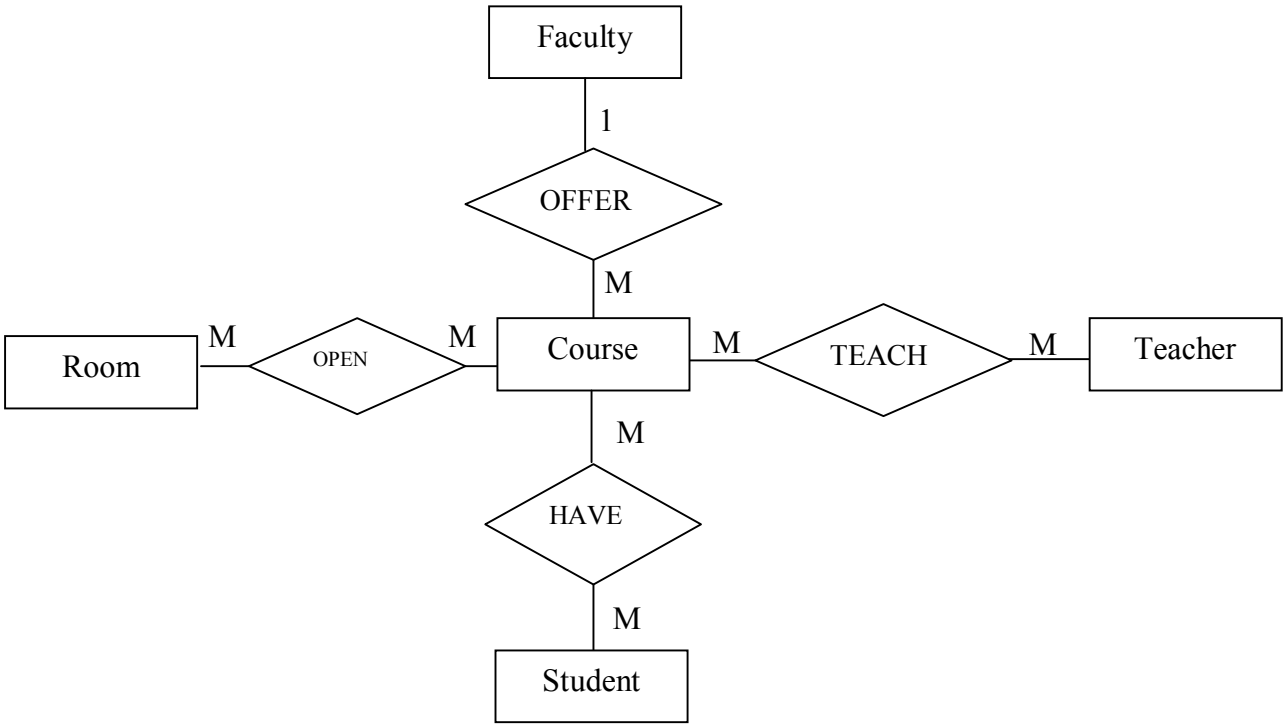


4. รายวิชาใดๆ (Course) จะสามารถเปิดสอนได้ต้องมี (Have) นักเรียน (Student) ลงทะเบียนเรียนอย่างน้อย 20 คน



9.4.3. กำหนดเงื่อนไขของความสัมพันธ์ระหว่าง Entity สามารถกำหนดได้ดังนี้

1. แต่ละคณะสามารถเสนออธิบดีเพื่อขอเปิดสอนได้หลายรายวิชา แต่รายวิชา 1 รายวิชาจะต้องอยู่ในคณะเดียวเท่านั้น กล่าวคือ ชื่อรายวิชาจะซ้ำกับคณะอื่นไม่ได้
2. อาจารย์ 1 ท่าน สามารถสอนได้หลายวิชา และ 1 รายวิชาสามารถมีอาจารย์สอนได้หลายท่านเช่นกัน
3. ห้องเรียน 1 ห้องสามารถเปิดทำการเรียนการสอนได้หลายรายวิชา และ 1 รายวิชาสามารถใช้ห้องเรียนหลายห้องเพื่อทำการเรียนการสอนได้เช่นเดียวกัน
4. รายวิชา 1 รายวิชาจะสามารถเปิดได้ จะต้องมีนักเรียนลงทะเบียนอย่างน้อย 20 คน และนักเรียน 1 คนสามารถมีวิชาเรียนได้หลายวิชา แสดงได้ดังรูป



9.4.4. กำหนด Attribute และ Primary Key ให้กับแต่ละ Entity ดังต่อไปนี้

Faculty (FAC_ID,FAC_NAME) โดยที่ FAC_ID เป็น Primary Key

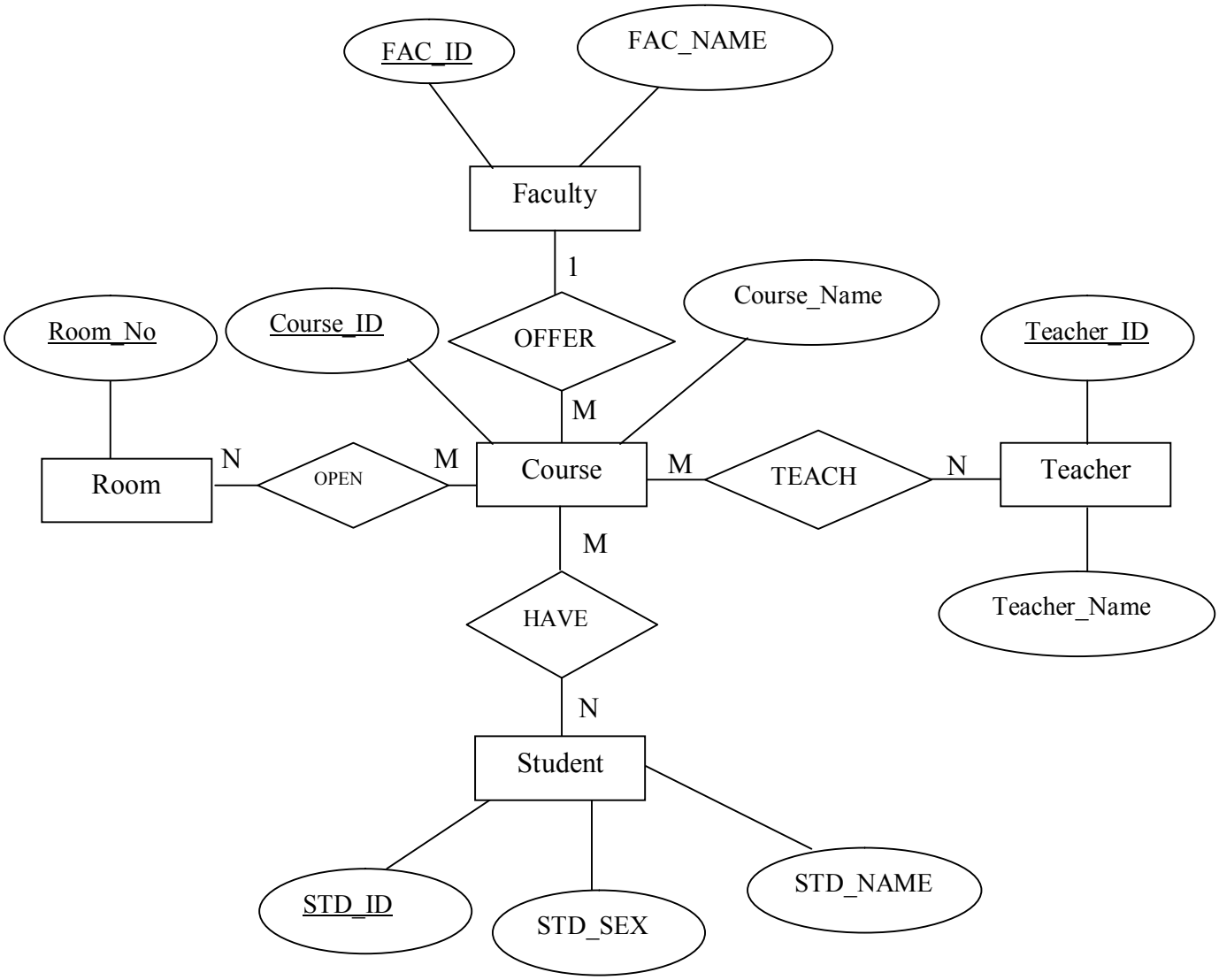
Course (Course_ID,Course_Name)โดยที่ Course_ID เป็น Primary Key

Teacher(Teacher_ID, Teacher_Name)โดยที่ Teacher_ID เป็น Primary Key

Student(STD_ID,STD_SEX,STD_NAME)โดยที่ STD_ID เป็น Primary Key

Room(Room_No) โดยที่ Room_No เป็น Primary Key

แสดงได้ดังรูป


FAC_ID
FAC_NAME
Faculty
1
OFFER
Course_Name
Room_No
Course_ID
Teacher_ID
M
Room
N
OPEN
M
Course
M
TEACH
N
Teacher
M
HAVE
Teacher_Name
N
Student
STD_ID
STD_SEX
STD_NAME

บทที่ 10

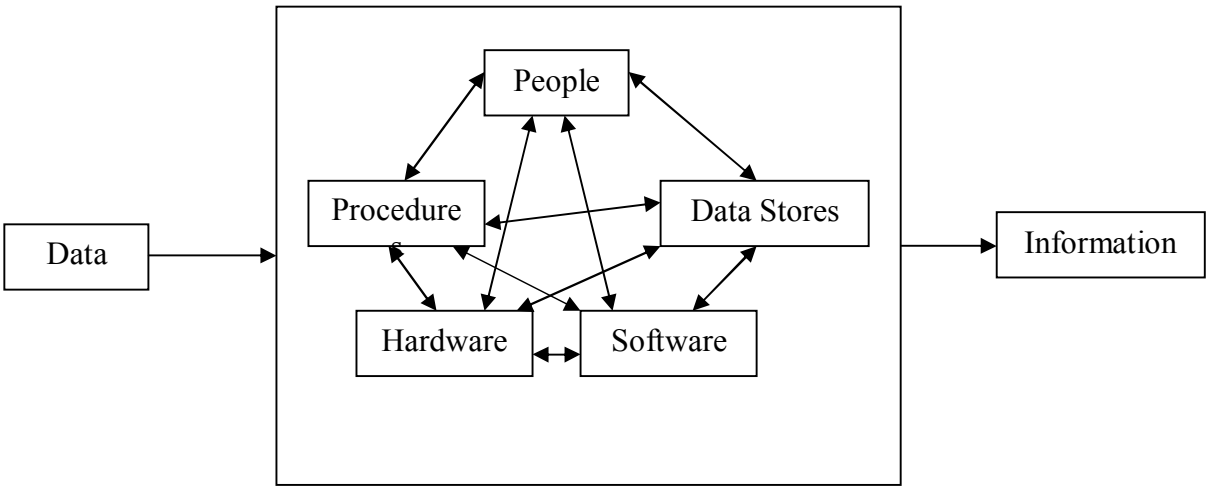
การออกแบบ Output และ Input

Logical Design เป็นเพียงการวิเคราะห์ในรายละเอียด ด้วยการกำหนดความสัมพันธ์ของส่วนประกอบต่างๆ ในระบบ ขั้นตอนต่อไปคือ กระบวนการออกแบบเพื่อพัฒนาให้เป็นรูปร่าง (Physical) ด้วยการนำ Logical Design ที่ได้มาจากขั้นตอนของการวิเคราะห์ มาออกแบบให้อยู่ในรูปของการปฏิบัติงานได้จริง (Physical Design)

ผลลัพธ์หรือเอาต์พุตจากการประมวลผลเพื่อให้ได้สารสนเทศที่ต้องการ จะต้องมีความถูกต้องและมีความน่าเชื่อถือ ตรงกับความต้องการของผู้ใช้งาน

10.1. กิจกรรมหลักที่ต้องมีในการออกแบบระบบ

ลักษณะระบบสารสนเทศ ประกอบไปด้วย ข้อมูลดิบ (Data) ส่วนจัดเก็บข้อมูล (Data Stores)บุคลากรทางคอมพิวเตอร์ (people) กระบวนการ (Procedures) ฮาร์ดแวร์ (Hardware) ซอฟต์แวร์ (Software) และสารสนเทศ (Information) ซึ่งส่วนประกอบต่างๆ เหล่านี้มีความสัมพันธ์กัน ในส่วนแรกของการออกแบบ คือ การออกแบบตรรกศาสตร์ ต่อมาจึงเริ่มออกแบบทางกายภาพโดยในการออกแบบต้องนำเอกสารต่างๆ ที่แสดงถึงความต้องการของผู้ใช้ระบบที่ได้มาจากการวิเคราะห์ระบบขั้นสุดท้ายมาใช้ เพื่อนำมากำหนดในการออกแบบทางกายภาพ การออกแบบทางกายภาพนั้นต้องเริ่มจากการออกแบบการแสดงผลลัพธ์ (Output Design) ก่อนเพราะจะมีผลกับการออกแบบการนำเข้าข้อมูล (Input Design) แล้วค่อยออกแบบการจัดเก็บข้อมูล (Data Stores) ซึ่งประกอบด้วยแฟ้มข้อมูล และฐานข้อมูล (File and Database) จากนั้นเป็นการออกแบบการประมวลผล (System Procession) และการออกแบบซอฟต์แวร์ (Software Design) ตามลำดับ ซึ่งในการออกแบบแต่ละส่วนต้องสัมพันธ์กัน ขั้นตอนสุดท้ายของการออกแบบระบบ คือการทำรายงานและการนำเสนอผลการออกแบบ โดยสรุปกิจกรรมในการออกแบบระบบได้ดังรูป



10.2. การออกแบบ Output

การออกแบบ Output จะต้องดำเนินการก่อนการออกแบบข้อมูลเข้า (Input Design) ด้วยเหตุผลสำคัญคือ รูปแบบของรายงานที่ทำการออกแบบ ทำให้ได้ข้อมูลนำเข้า ดังนั้นการออกแบบรายงานจึงทำให้ทราบถึงข้อมูลนำเข้าที่จำเป็นต้องป้อนเข้าสู่ระบบ

การออกแบบ Output ควรเริ่มต้นด้วยการเขียนลงในแบบฟอร์ม (Report Layout Form) ซึ่งถือเป็นสิ่งสำคัญเพื่อใช้ในการตัดสินใจได้ว่า จะมีรายละเอียดของข้อมูลใดบ้างในรายงาน โดย Output ที่ออกแบบนั้นจะต้องตรงกับความต้องการ (Requirements) รวมทั้งพจนานุกรมข้อมูล (Data Dictionary) ที่ทำให้ทราบถึงประเภทของข้อมูล ทราบถึงขนาดความกว้างข้อมูล ทำให้สามารถกำหนดจำนวนคอลัมน์ที่ต้องการใช้ในรายงานได้อย่างถูกต้อง โดยรายละเอียดที่เขียนลงในแบบฟอร์มอาจใช้ข้อความ หรือใช้ตัวอักษร X เพื่อแทนข้อมูลตัวอักษร และเลข 9 เพื่อแทนข้อมูลตัวเลข

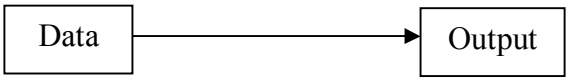
10.2.1. ชนิดของ Output

Output คือ ข้อความ เอกสาร หรือรายงานชนิดหนึ่ง โดยนำข้อมูลจากไฟล์มาพิมพ์ หรือนำข้อมูลมาผ่านการประมวลผลเพื่อได้เอาต์พุตที่ต้องการ ดังนั้นเอาต์พุตจึงอาจหมายถึง

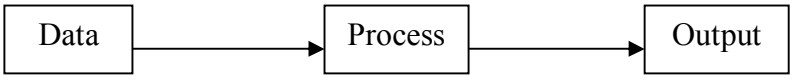
- รายงาน (Report)
- เอกสาร (Document)
- ข้อความ (Message)

กระบวนการต่างๆ เพื่อให้ได้มาซึ่งเอาต์พุตที่ต้องการนั้น อาจมาจากแหล่งข้อมูลด้วยวิธีการต่างๆดังต่อไปนี้

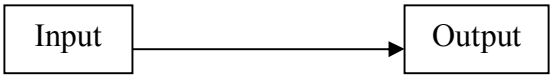
1. เรียงจากแฟ้มข้อมูลโดยตรง (Retrieval from a data store) เป็นรายงานที่สามารถทำการแสดง (List) หรือสั่งพิมพ์ได้ด้วยการนำข้อมูลในแฟ้มข้อมูลนั้นมาพิมพ์ได้ทันที



2. นำข้อมูลมาผ่านการประมวลผลเพื่อให้ได้รายงานที่ต้องการ (Transmission from a process) เป็นเอาต์พุตที่มีการนำข้อมูลจากแฟ้มข้อมูล ผ่านกระบวนการประมวลผลเพื่อให้ได้เอาต์พุตที่ต้องการ

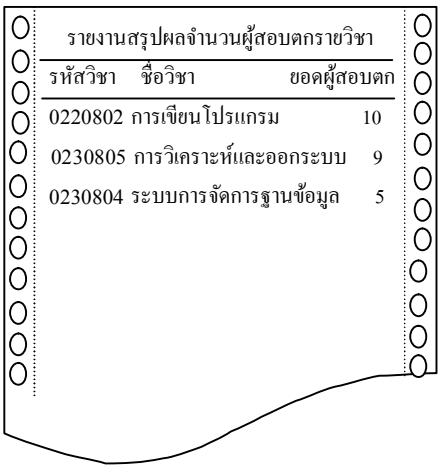


3. รับข้อมูลโดยตรงจากการคีย์ข้อมูลเข้า (Direct from an input source) เป็นเอาต์พุตที่ได้จากการคีย์ข้อมูลเข้าโดยตรง



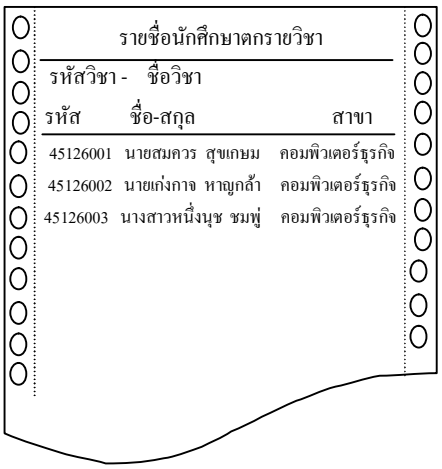
การออกแบบ Output หรือการออกแบบรายงานที่ดีนั้น ควรมีการออกแบบให้มีข้อมูลที่สัมพันธ์กันกับรายงานต่างๆ ที่เกี่ยวข้อง กล่าวคือ ในรายงานฉบับหนึ่งอาจนำเสนอข้อมูลเพียงเพื่ออธิบายสรุปผลบางอย่าง ถ้าผู้ใช้งานต้องการรายละเอียดข้อมูลในรายงาน อาจจำเป็นต้องดูรายงานอีกฉบับหนึ่งเพื่อทำการตรวจสอบ ดังนั้นข้อมูลในรายงานที่เกี่ยวข้องจึงต้องมีความสัมพันธ์กัน

ตัวอย่างเช่น หลังจากมีการประเมินผลเป็นที่เรียบร้อยแล้ว จะพิมพ์รายงานสรุปผลจำนวนผู้สอบตกรายวิชา ซึ่งเป็นข้อมูลสรุปจำนวนนักศึกษาที่สอบตกในรายวิชาต่างๆ คณบดีอาจต้องการเพียงยอดจำนวนนักศึกษา เพื่อนำไปประกอบการพิจารณาว่า เห็นสมควรต่อการเปิดลงทะเบียนในรายวิชาที่นักศึกษาสอบตกในภาคการศึกษาต่อไปหรือไม่ แต่ถ้าต้องการทราบในรายละเอียดว่า มีนักศึกษารายใดบ้างที่สอบตกอาจจำเป็นต้องใช้รายงานอื่นประกอบ เช่น รายงานรายชื่อนักศึกษาสอบตกรายวิชา เป็นต้น



รายงานสรุปผลจำนวนผู้สอบตกรายวิชา

รหัสวิชา	ชื่อวิชา	ยอดผู้สอบตก
0220802	การเขียนโปรแกรม	10
0230805	การวิเคราะห์และออกระบบ	9
0230804	ระบบการจัดการฐานข้อมูล	5



รายชื่อนักศึกษาตกรายวิชา

รหัสวิชา - ชื่อวิชา

รหัส	ชื่อ-สกุล	สาขา
45126001	นายสมควร สุขเกษม	คอมพิวเตอร์ธุรกิจ
45126002	นายเก่งกาจ หาญกล้า	คอมพิวเตอร์ธุรกิจ
45126003	นางสาวหนึ่งนุช ชมพู่	คอมพิวเตอร์ธุรกิจ

ถึงแม้เทคโนโลยีทางคอมพิวเตอร์ในปัจจุบัน จะมีส่วนช่วยทำให้การออกแบบงานมีรูปแบบที่สวยงาม แต่รายงานที่นำเสนอรูปแบบที่สวยงามใช่ว่าจะเป็นรายงานที่ดีเสมอไป รายงานที่ดีจะพิจารณาถึงข้อมูลหรือสารสนเทศที่ประกอบกันเป็นรายงาน และตรงกับความต้องการของผู้ใช้เป็นหลัก

รายงานในระบบมีจำนวนมากมาย ต่างก็มีความสำคัญในการนำเสนอข้อมูลต่อผู้ใช้งานในแง่มุมต่างๆดังนั้นการออกแบบรายงานควรพิจารณาถึงสิ่งต่อไปนี้

1. ใครเป็นผู้ใช้รายงานนี้
2. ใช้ประโยชน์จากรายงานนี้อย่างไร
3. รายละเอียดข้อมูลในรายงานต้องการมากเพียงใด
4. รายงานนี้มีความต้องการใช้บ่อยแค่ไหน เช่น ทุกวัน ทุกสัปดาห์ หรือทุกเดือน
5. รายงานแสดงออกทางใด เช่น ทางจอภาพ หรือทางเครื่องพิมพ์

ตัวอย่างรายงานการลงทะเบียนแต่ละรายวิชา

1. ใครเป็นผู้ใช้รายงานนี้

-ฝ่ายทะเบียนและวัดผล

-คณบดี

-นักศึกษา

-อาจารย์ผู้สอน

2. ใช้ประโยชน์จากรายงานนี้อย่างไร

- ฝ่ายทะเบียนและวัดผล เพื่อต้องการทราบรายชื่อและจำนวนนักศึกษาที่ลงทะเบียนเรียนในแต่ละรายวิชา
- คณบดี เพื่อต้องการทราบจำนวนนักศึกษาที่ลงทะเบียนเรียนแต่ละรายวิชา
- นักศึกษา ตรวจสอบรายวิชาที่ตนลงทะเบียนจากบอร์ดว่ามีหรือไม่ และเรียนอยู่กลุ่มใด
- อาจารย์ผู้สอน ตรวจสอบจำนวนนักศึกษาในรายวิชาพร้อมทั้งพิมพ์รายชื่อนักศึกษาในกลุ่มที่สอน

3. รายละเอียดข้อมูลในรายงาน

4. รายงานนี้ต้องการใช้บ่อยแค่ไหน

- ใช้ทุกๆ ภาคการศึกษาเมื่อมีการเปิดลงทะเบียนแล้วเสร็จ

5. รายงานแสดงออกทางใด

- ทางเครื่องพิมพ์ และทางจอภาพ

10.2.2. การจัดรูปแบบรายงาน

การจัดรูปแบบรายงาน เป็นเรื่องที่สำคัญอย่างหนึ่งที่ทำให้สามารถสื่อความหมายให้กับผู้ใช้งานได้ครบถ้วน รวมทั้งผู้พัฒนาระบบงาน ดังนั้นรูปแบบรายงานที่ดีควรมีรายละเอียดดังนี้

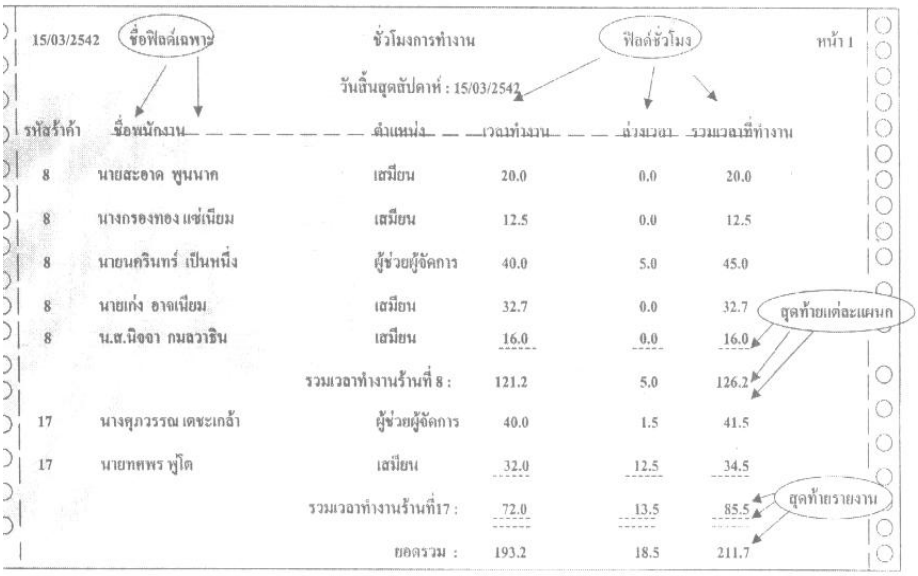
1. หัวรายงาน (Heading) :

หัวรายงานหรือชื่อรายงาน จำเป็นต้องมีในทุกๆ รายงานและทุกหน้า เพื่อสื่อความหมายข้อมูลในรายงานว่าเป็นรายงานอะไร หากรายงานมีมากกว่าหนึ่งหน้า ในหน้าถัดๆ ไปจะต้องมีชื่อรายงานและเลขหน้ากำกับไว้เสมอ โดยเลขหน้าอาจกำกับได้โดยแสดงตัวอย่างตามลำดับ เช่น 1,2,3.....หรือ 1 of 10, 2 of 10..... เป็นต้น ในส่วนของหัวรายงานซึ่งอยู่บรรทัดแรกๆ นั้น นอกจากจะมีชื่อรายงาน เลขหน้าแล้ว อาจจะมีชื่อบริษัทหรือหน่วยงาน วันที่พร้อมเวลาที่พิมพ์รายงาน เพื่อสามารถตรวจสอบได้ว่าเป็นรายงานที่ทันสมัย (Update) หรือไม่ และอาจใส่ชื่อโปรแกรมไว้เป็นได้เพื่อเป็นประโยชน์ต่อผู้พัฒนา เช่น หากในวันข้างหน้าจำเป็นต้องมีการปรับหรือแก้ไขรายงานนี้ จะทำให้ทราบชื่อโปรแกรมจากรายงานนี้ได้ทันที ทำให้ไม่ต้องเสียเวลากับการค้นหาโปรแกรมเพื่อทำการแก้ไข

2. รายละเอียด (Details) :

พื้นที่ในส่วนของรายละเอียดจะมีพื้นที่มากที่สุด เพื่อใช้แสดงรายละเอียดหรือข้อมูลต่างๆ ซึ่งรายละเอียดหรือข้อมูลนั้นอาจจะแสดงต่อเนื่องกันไป หรือมีการรวมกลุ่มและมีการควบคุมข้อมูล (Control Break) เป็นส่วนๆ การกำหนดเงื่อนไขการพิมพ์โดยมีรายละเอียดดังนี้

- การควบคุม (Control Break) คือมีการรวมข้อมูลที่เกี่ยวข้องและแตกเป็นกลุ่มย่อย ในที่นี้คือการควบคมการแสดงรายละเอียดของข้อมูลด้วยหมายเลขรหัสลูกค้า
- เงื่อนไขการพิมพ์ (Conditions) คือมีการพิมพ์รายงานด้วยการกำหนดเงื่อนไข ในที่นี้คือการกำหนดเงื่อนไขด้วยวันที่สิ้นสุดสัปดาห์ 15/03/47
- ยอดสรุป (Summaries) คือ ยอดรวมเวลาทำงานของร้านที่.....



3. ผลสรุป (Summaries) :

รายงานบางรายงานถูกออกแบบมาเพื่อสรุปผลสารสนเทศ โดยจะไม่มีการแสดงรายละเอียดใดๆ ซึ่งในบางครั้งผู้บริหารหรือผู้ที่ใช้รายงานประเภทนี้ต้องการทราบยอดจำนวนเท่านั้น โดยไม่ต้องการทราบรายละเอียด

4. หมายเหตุ (Remarks) :

รายงานบางประเภทอาจจำเป็นต้องมีหมายเหตุหรือคำแนะนำ เพื่อให้ผู้อ่านหรือผู้ใช้รายงานเข้าใจในรายละเอียดมากขึ้น

10.2.3. ประเภทของรายงานมีดังนี้

1. รายงานแสดงรายละเอียด (Detailed Reports)

รายงานแสดงรายละเอียด คือ รายงานที่อ่านข้อมูลจากแฟ้มข้อมูล ซึ่งอาจจะไม่ได้มีการปรุงแต่งข้อมูลเลย หรืออาจจะมีการปรับแต่งเล็กน้อย เช่น รายงานแสดงรายชื่อนักศึกษา, รายงานวิชาที่เปิดลงทะเบียน เป็นต้น

2. รายงานแสดงประวัติข้อมูล (Historical Reports)

รายงานแสดงประวัติข้อมูลจะมีลักษณะคล้ายกับรายงานแสดงรายละเอียด แต่มีข้อแตกต่างคือ ตัวรายงานจะแสดงข้อมูลประจำวัน (Transaction) เพื่อใช้ในการตรวจสอบ หรือ ยืนยันว่าระบบได้ดำเนินการและเก็บข้อมูลได้อย่างถูกต้อง ตัวอย่างรายงานประเภทนี้ เช่น รายงานการเบิกของจากคลังประจำวัน รายงานการเบิกถอนเงินธนาคารของลูกค้า รายงานการลงทะเบียนรายวิชาในแต่ละวัน เป็นต้น

3.รายงานสรุปผลข้อมูล (Summary Reports)

รายงานสรุปผลข้อมูลจัดเป็นรายงานสำหรับผู้บริหารที่ต้องการดูผลสรุปเพียงอย่างเดียว โดยไม่ต้องการดูข้อมูลในรายละเอียด เพื่อใช้ในการพิจารณาและตัดสินใจ เช่น รายงานงบกำไรขาดทุน, รายงานยอดสรุปจำนวนผู้สอบตกรายวิชา เป็นต้น

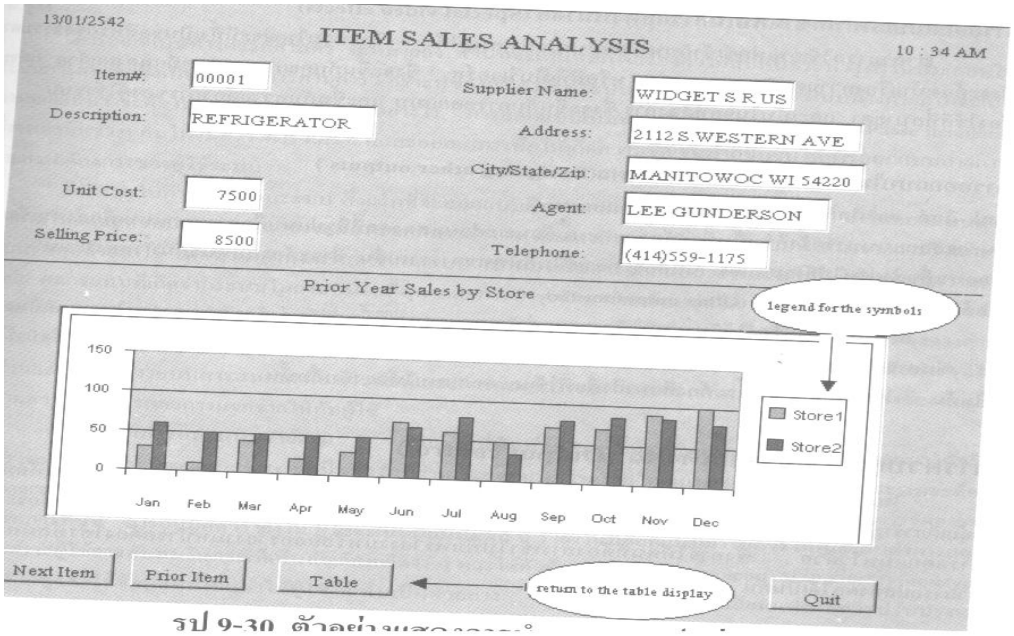
4.รายงานข้อยกเว้น หรือกรณีพิเศษ (Exception Reports)

รายงานข้อยกเว้นหรือกรณีพิเศษนี้ เป็นรายงานที่มีการกรองข้อมูล (Filter) บางอย่างออกไป เพื่อคงไว้เพียงแต่ข้อมูลที่ต้องการ เช่นรายงานผลเฉลี่ยสะสมรายวิชาที่ต่ำกว่าเกณฑ์ จะแสดงข้อมูลนักศึกษาเฉพาะที่มียอดคะแนนเฉลี่ยสะสมที่ไม่ถึง 2.00 ส่วนที่อยู่ในเกณฑ์หรือสูงกว่าในเกณฑ์ จะไม่นำมาพิมพ์

10.2.4. การนำเสนอข้อมูลในรายงาน แบ่งเป็น 2 ชนิด

1. รูปแบบตาราง (Tabular Format) : รายงานในลักษณะรูปแบบตารางนี้เป็นรูปแบบที่ใช้ทั่วไป โดยรูปแบบจะแบ่งเป็นแถว (Row) และคอลัมน์ (Column) และอาจมียอดรวมตัวเลขที่อาจเสนอเป็นแนวนอน (Horizontal) หรือ แนวตั้ง (Vertical)

2. รูปแบบกราฟ (Graph Format) : รายงานในลักษณะกราฟนั้นมักใช้งานทางธุรกิจที่สรุปผลการดำเนินงานในเชิงเปรียบเทียบ ทำให้เห็นภาพชัดเจนกว่าในลักษณะข้อมูล เช่น การเปรียบเทียบผลกำไรในแต่ละไตรมาส เมื่อเทียบกับการเปรียบเทียบในลักษณะตัวเลขอาจเห็นได้ไม่ชัดเจน แต่ถ้าเปรียบเทียบในลักษณะกราฟจะเปรียบเทียบได้ชัดเจนขึ้น โดยกราฟที่นำเสนออาจเป็นได้ทั้งกราฟเส้น กราฟแท่ง และกราฟวงกลม เป็นต้น



รูป 9-30 ตัวอย่าง

10.2.5. การออกแบบรายงานทางจอภาพ

การติดต่อระบบงานระหว่างผู้ใช้ระบบ จะเป็นไปในลักษณะของการโต้ตอบ มีการแลกเปลี่ยนข้อมูลกันทั้งส่วน Input และOutput ผู้ใช้ระบบเป็นผู้ที่ป้อนข้อมูลให้กับระบบ เมื่อคอมพิวเตอร์ได้รับข้อมูลเข้าจะทำการประมวลผลเป็น Output ให้กับผู้ใช้ระบบ โดย Output ที่ระบบประมวลผลออกมาอาจเป็น Input ของผู้ใช้งาน เพื่อทำการรับข้อมูลต่อและประมวลผลอีกครั้ง

การโต้ตอบระหว่างคอมพิวเตอร์กับผู้ใช้งาน ในส่วนของคอมพิวเตอร์จะมีข้อจำกัดที่มากกว่าเนื่องจากมีพื้นที่จำกัดอยู่ในหน้าจอคอมพิวเตอร์ เช่น การนำเสนอรายงานทางจอภาพ หน้าจออาจมีขนาดเล็กเกินไปกับผลลัพธ์ที่ต้องการแสดงทั้งหมด ดังนั้นจึงควรทำการออกแบบผลลัพธ์ที่สามารถแสดงได้ทีละหน้าหรือทีละส่วนที่ต่อเนื่องกันไป เพื่อให้ผู้ใช้งานสามารถดูข้อมูลได้ทั้งหมดและต่อเนื่องได้ทางจอภาพ ซึ่งเป็นไปได้ที่ผู้ใช้งานต้องการดูรายละเอียดข้อมูลทางจอภาพเพียงอย่างเดียว หรือดูรายละเอียดทางจอภาพเพื่อความแน่นอนก่อนที่จะพิมพ์ลงเครื่องพิมพ์

การนำเสนอรายงานทางจอภาพมีข้อดี คือช่วยลดปริมาณกระดาษ และช่วยลดค่าใช้จ่ายลงได้มาก แต่รายงานบางประเภทจำเป็นต้องนำเสนอทางกระดาษ เพื่อใช้ในการอ้างอิงข้อมูลและจัดเก็บเพื่อเป็นหลักฐานไว้ใช้งานต่อไป ซึ่งผิดกับรายงานทางจอภาพที่เป็นเพียงการนำเสนอผ่านทางจอภาพ ไม่สามารถจัดเก็บได้ ในส่วนของรายงานทางเครื่องพิมพ์จะต้องคำนึงถึงประเภทของรายงาน ขนาดกระดาษ และการใช้เครื่องพิมพ์

10.3. การออกแบบ Input

การออกแบบ Input อาจเริ่มต้นเขียนด้วยแบบฟอร์ม (Screen Layout Form) ซึ่งมีลักษณะคล้ายกับแบบฟอร์มของการออกแบบเอาต์พุต แต่จำนวนแถวและคอลัมน์จะถูกจำกัด กล่าวคือจอภาพปกติที่แสดงผลในลักษณะของข้อความ (Text) จะมีอยู่จำนวน 25 แถว 80 คอลัมน์ ซึ่งการออกแบบ Input จัดเป็นขั้นตอนแรกในการเตรียมข้อมูลเพื่อป้อนให้กับระบบ และทำการประมวลผลเพื่อให้ได้ผลลัพธ์ที่ต้องการ โดยวัตถุประสงค์ของการออกแบบอินพุต (Objective of Input Design) มีดังนี้

1. ควบคุมจำนวนอินพุต : การเตรียมข้อมูล (Data Preparation) และการป้อนข้อมูล (Data entry) ซึ่งจะต้องมีพนักงานป้อนข้อมูล (Data Entry Operator) และจำเป็นต้องใช้กำลังคน อาจใช้เวลามากในการตระเตรียมข้อมูลเพื่อป้อนเข้าสู่ระบบเพื่อการประมวลผล นั่นหมายถึงการมีค่าใช้จ่ายที่เพิ่มขึ้น ในช่วงของการตระเตรียมข้อมูล หน่วยประมวลผลกลางจะอยู่ในสภาวะนิ่ง (Sit Idle) จนกระทั่งมีการป้อนข้อมูลเพื่อให้ระบบทำการประมวลผล ดังนั้นการควบคุมจำนวนอินพุต ทำให้อินพุตต่างๆที่จำเป็น จึงมีส่วนทำให้มีการประมวลผลข้อมลได้รวดเร็วยิ่งขึ้น
2. หลีกเลี่ยงความล่าช้า : ผลของการประมวลผลที่ล่าช้า เกิดจากการตระเตรียมข้อมูลเพื่อป้อนข้อมูลเข้า และคอมพิวเตอร์จะอยู่ในสภาวะรอข้อมูลเพื่อ Input ทำให้ไม่ได้ใช้ประโยชน์จากซีพียูได้อย่างเต็มที่ ควรมีการจัดการ Input ข้อมูลที่ดี มีการพิจารณาความล่าช้าของระบบว่าเกิดจากสาเหตุใด เช่น ในบางช่วงเวลาที่วิกฤตที่อาจมีข้อมูลเข้าหรือมีการใช้บริการมากเป็นพิเศษ ในช่วงนี้อาจเกิดปัญหาคอขวด (Bottleneck) ได้
3. หลีกเลี่ยงการป้อนข้อมูลที่ผิดพลาด : สารสนเทศที่ถูกต้อง จะมีความเชื่อถือได้มากหรือไม่นั้นขึ้นอยู่กับข้อมูลที่ป้อนเข้าไปในระบบ เช่น สมมติว่าในหนึ่งสัปดาห์มีการบันทึกข้อมูลการขายจำนวนประมาณ 10,000 รายการ และมีรายการประมาณ 300 รายการที่มีข้อมูลผิดพลาด นั่นหมายถึง มีข้อผิดพลาดในข้อมูล 3 เปอร์เซ็นต์ นักวิเคราะห์ระบบจะต้องทำการลดจำนวนข้อผิดพลาดจากการป้อนข้อมูลเข้า ซึ่งจำเป็นต้องมีการตรวจสอบข้อมูลเข้าโดยโปรแกรม เพื่อแจ้งให้กับผู้ใช้งานว่าเกิดการป้อนข้อมูลผิดพลาดในระบบแล้ว
4. หลีกเลี่ยงขั้นตอนพิเศษหรือขั้นตอนที่ไม่จำเป็น : ในบางครั้งจำนวนรายการและจำนวนข้อมูลมีปริมาณมากเกินเกิดจากการไม่มีการควบคุมข้อมูลเช่น การสั่งจ่ายเช็ค ซึ่งในความเป็นจริงลูกค้าอาจมีการสั่งจ่ายเช็คที่ถูกต้องและแบบกรณีพิเศษที่ปะปนกันไป ทำให้ขั้นตอนการประมวลผลโดยรวมอาจเกิดประสิทธิผลที่ต่ำลง ดังนั้นขั้นตอนพิเศษนี้อาจทำการแยกส่วนเฉพาะไป เพื่อจะได้ไม่ปะปนกัน การแตกส่วนงานในขั้นตอนของการสั่งจ่ายเช็คแบบกรณีพิเศษออกไปอีกส่วนหนึ่ง จะทำให้ขั้นตอนการประมวลผลของการสั่งจ่ายเช็คแบบปกติ สามารถทำงานได้รวดเร็วและสะดวก และลดปริมาณข้อมูลที่ไม่จำเป็นออกไป
5. มีขั้นตอนการใช้งานง่าย : เป็นไปได้ว่าหากมีการควบคุมข้อมูลเข้า จะมีส่วนทำให้ผู้ใช้งานใช้งานระบบยากขึ้น แต่อย่างไร การควบคุมข้อมูลเข้าจะทำให้ระบบดีขึ้น ในด้านของการประมวลผลข้อมูลในขั้นต่อไปที่มีความถูกต้องสูง และง่ายต่อการตรวจสอบ ควรมีรูปแบบการควบคุมข้อมูลและแสดงข้อมูลแนะนำให้ชัดเจน การช่วยลดความยุ่งยากในการทำงานให้กับผู้ใช้งานและใช้งานง่าย ทำให้ผู้ใช้งานมีการยอมรับในระบบ

10.3.1. การออกแบบหน้าจอรับข้อมูล (Data Entry Screens Design) : ในการออกแบบหน้าจอรับข้อมูลมีแนวทาง 14 ข้อ คือ

1. เมื่อเข้าไปที่หน้าจอรับข้อมูลเคอร์เซอร์ (Cursor) ต้องอยู่ที่ตำแหน่งแรกของข้อมูลรับตามเนื้อหาของหน้าจอนั้น ตำแหน่งนั้นควรอยู่มุมบนซ้ายมือ หลังจากผู้ใช้ใส่ข้อมูลแล้วเคอร์เซอร์ (Cursor) ควรเลื่อนไปที่ช่องรับข้อมูลถัดไปเสมอ คือด้านล่างหรือด้านข้างถัดไปเพื่อให้การป้อนข้อมูลเป็นธรรมชาติจากบนซ้ายไปขวา แล้วจึงลงมาบรรทัดถัดไป

2. ทุกช่องรับข้อมูลต้องมีคำอธิบาย โดยแสดงกำกับที่ช่องนั้น ซึ่งจะต้องระบุให้ชัดเจนและอยู่ในตำแหน่งที่ถูกต้อง และควรมีรูปแบบเดียวกันเพื่อให้ผู้ใช้ไม่สับสนในการทำงาน

3. ถ้าช่องรับข้อมูลนั้นมีรูปแบบเฉพาะให้แสดงรูปแบบเฉพาะนั้นเพื่อให้ผู้ใช้ป้อนข้อมูลไม่ผิดพลาด ตัวอย่างเช่น ถ้าเป็นวันที่เริ่มต้น (Start Date) ซึ่งกำหนดฟิลด์เป็น DD/MM/YYYY คือใส่วันนี้เป็นเลข 2 หลัก ตามด้วย "/" ตามด้วยเดือนซึ่งเป็นเลข 2 หลักตามด้วย "/" ตามด้วยปีเป็นเลข 4 หลัก

4. ในทุกครั้งที่ป้อนข้อมูลในช่องรับข้อมูลนั้นครบถ้วนให้ผู้ใช้กด "Enter" เพื่อไปยังช่องรับข้อมูลถัดไป หรือในกรณีที่ใส่ข้อมูลเท่าจำนวนของข้อมูลนั้นแล้ว เคอร์เซอร์ควรไปยังช่องรับข้อมูลถัดไปอัตโนมัติเพื่อรับข้อมูลต่อไป เพื่ออำนวยความสะดวกต่อผู้ใช้

5. กรณีข้อมูลรับเข้าเป็นตัวอักษรแล้วควรออกแบบให้สามารถใส่ตัวอักษรพิเศษอัตโนมัติโดยผู้ใช้ไม่ต้องใส่เอง เช่น ถ้าต้องการให้ผู้ใช้ป้อนรหัสสินค้าที่จะคั่นด้วย "-" คือรหัสสินค้ามาตรฐานคือ 99-999-99 แล้วผู้ใช้เพียงป้อนตัวเลขคือ 9999999 ค่าแสดงผลการป้อนจะเป็น 99-999-99 ให้ผู้ใช้เห็นทันที โดยผู้ใช้ไม่ต้องยุ่งยากกับการป้อน "-" เป็นต้น

6. กรณีข้อมูลรับเข้าต้องการแสดงผลรับเป็นเลข "0" ให้แสดงโดยผู้ใช้ไม่ต้องป้อนเอง เช่นถ้าป้อนรหัสโครงการที่มีการแสดงผลเป็น ZZZZ คือแสดงตัวเลข "0" ถ้าค่านั้นเป็น "0" ดังนั้นเมื่อผู้ใช้ป้อนเลข "45" เมื่อแสดงผลรับข้อมูลนั้นจะแสดง "0045" เป็นต้น

7. กรณีนี้ข้อมูลรับเข้าเป็นจำนวนเลขที่นำไปคำนวณค่า เช่น ถ้าเป็นตัวเลขที่แทนจำนวนเงิน ควรมี "," คั่นหลักพันหรือจุดทศนิยมขึ้นอัตโนมัติ คือ เมื่อผู้ใช้ใส่ จำนวนเงิน 1980 บาท จะเป็น 1,980.00 บาท ทันที

8. กรณีช่องรับข้อมูลนั้นมีค่าที่สามารถกำหนด (Default) ได้ ระบบนั้นควรแสดงค่าที่กำหนดนั้น (Default) ที่ช่องนั้นเพื่ออำนวยความสะดวกกับผู้ใช้โดยผู้ใช้เพียงกด "Enter" เพื่อรับค่านั้นเลยโดยผู้ใช้ไม่ต้องป้อนข้อมูลนั้น เช่นถ้าเป็นวันที่ใช้คือวันที่ปัจจุบันเป็นค่าที่กำหนดให้ผู้ใช้เห็นซึ่งผู้ใช้สามารถกด "Enter" เพื่อรับค่าเข้าระบบได้ทันที แต่ถ้าต้องการเปลี่ยนสามารถเปลี่ยนได้ตามความต้องการของผู้ใช้ เป็นต้น

9. สำหรับข้อมูลรับที่เป็นรหัสที่ต้องมีการตรวจสอบก่อนบันทึก คือค่านั้นมีความสัมพันธ์กับค่าอื่นที่บันทึกไว้เดิมแล้ว เมื่อมีการป้อนรหัสนั้นต้องรับค่านั้นไปตรวจสอบที่แฟ้มข้อมูลที่อ้างถึง เพื่อตรวจสอบว่าเป็นรหัสเดิมหรือไม่ ถ้าใช่ให้แสดงผลพร้อมค่าที่ต้องการแสดงเพื่อระบุรายละเอียดของรหัสนั้นให้ผู้ใช้ทราบว่าป้อนรหัสไม่ผิด เช่น กรณีในการใส่รหัสลูกค้าคือ "001" ซึ่งหมายถึง "บริษัท ก" เมื่อผู้ใช้ป้อนรหัสลูกค้าคือ "001" ควรแสดง "บริษัท ก" ให้ผู้ใช้ได้เห็นทางหน้าจอด้วย เป็นต้น

10. ควรมีปุ่มคำสั่งต่างๆ ตามความเหมาะสมของการใช้งานหน้าจอนั้น เช่น มีปุ่มคำสั่งเพื่อออกจากหน้าจอ Data Entry (Data Entry Screen) โดยไม่ต้องบันทึกข้อมูล (Exit without Adding record) หรือมีปุ่มคำสั่งในการเคลียร์หน้าจอ (Clear Current Record) เป็นต้น

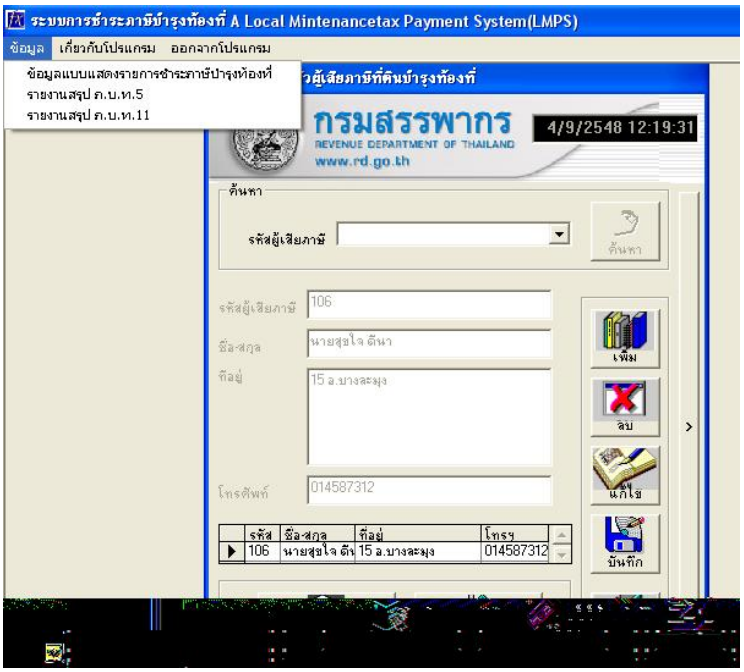
11. หลังจากป้อนข้อมูลในหน้าจอนั้นสมบูรณ์แล้วและได้รับการตรวจสอบแล้วควรมีการให้ผู้ใช้ได้ตรวจสอบ และยืนยันหรือยกเลิกก่อนบันทึกข้อมูลเข้าเครื่องเช่น "บันทึกข้อมูลที่เพิ่มเติมนี้หรือไม่" (Add this record?(y/n) เป็นต้น

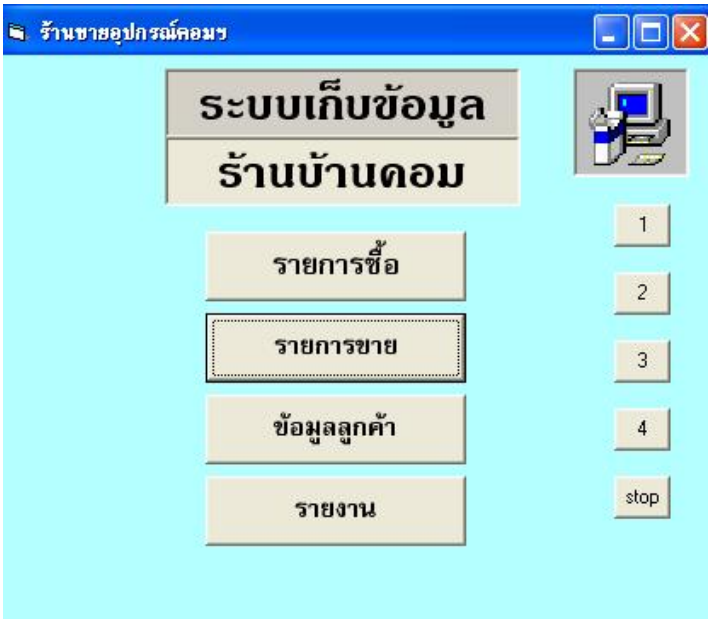
12. ควรเตรียมปุ่มรับคำสั่งเพื่อเคลื่อนย้ายเคอร์เซอร์ในฟิลด์ต่างๆ ในหน้าจอเพื่อให้ผู้ใช้ได้แก้ไขก่อนจะบันทึกเข้าเครื่อง เช่นอาจใช้ปุ่ม "TAB" ในการเคลื่อนไปยังฟิลด์ถัดไป และใช้ "SHIFT+TAB" ในการเคลื่อนไปยังฟิลด์ก่อนหน้า เป็นต้น

13. ถ้าผู้ใช้ดาต้าแคปเจอร์เพื่อป้อนข้อมูล ในการออกแบบหน้าจอดาต้าเอนทรี นั้นควรให้สอดคล้องกับดาต้าแคปเจอร์นั้น เช่น หน้าจอ ดาต้าเอนทรี ซึ่งสอดคล้องกับเอกสาร (Source Document)

14. ในการออกแบบหน้าจอควรมีคำสั่งต่อไปนี้คือ เพิ่มข้อมูล (Add) ปรับแก้ข้อมูล (Change) ลบข้อมูล (Deletes) และแสดงผลข้อมูล (View) ของรายการต่างๆ คำสั่งที่ต้องระวังเป็นพิเศษควรมีการยืนยันคำสั่งนั้นก่อนจะปฏิบัติคำสั่งนั้นด้วย

10.3.2. ประเภทของการออกแบบความสัมพันธ์กับผู้ใช้





1. การออกแบบความสัมพันธ์กับผู้ใช้ด้วยเมนู

ระบบเมนูถือเป็นการออกแบบความสัมพันธ์ประเภทหนึ่งที่นิยมใช้กันมาก เนื่องจากในชีวิตประจำวันมักใช้ในลักษณะนี้ เช่น ในร้านอาหาร หากเราต้องการสั่งอาหารมักจะเลือกรายการอาหารที่ตนต้องการจากเมนู เช่นเดียวกับระบบเมนูในคอมพิวเตอร์ ที่ผู้ใช้งานสามารถเลือกรายการที่ตนต้องการได้ โดยรูปแบบของเมนูมีลักษณะต่างๆ

2. การออกแบบความสัมพันธ์กับผู้ใช้ด้วยคำสั่ง

การออกแบบความสัมพันธ์กับผู้ใช้ด้วยคำสั่ง ผู้ใช้งานจะต้องมีความรู้ในการใช้คำสั่งต่างๆ เป็นอย่างดีจึงจะสามารถทำการโต้ตอบกับระบบได้ เช่น ในระบบปฏิบัติการดอส (DOS) หรือระบบปฏิบัติการยูนิกส์ (Unix) ผู้ใช้งานต้องรู้คำสั่งการใช้งานต่างๆ เพื่อติดต่อกับระบบ จึงจะสามารถปฏิบัติการโต้ตอบกับระบบได้หรือโปรแกรมฐานข้อมูล เช่น dBASE หรือ FoxPro ที่ออกแบบความสัมพันธ์กับผู้ใช้ ด้วยการป้อนคำสั่งเพื่อใช้งาน

3. การออกแบบความสัมพันธ์กับผู้ใช้แบบกราฟิก

การออกแบบความสัมพันธ์กับผู้ใช้แบบกราฟิก หรือที่เรียกว่า Graphics User Interface ในปัจจุบันเป็นที่นิยมมาก เนื่องจากสามารถสื่อสารกับผู้ใช้ในลักษณะข้อความได้แล้ว ยังสามารถสื่อสารในรูปแบบของภาพต่างๆได้อีกด้วย ทำให้ผู้ใช้งานเกิดความเข้าใจและใช้งานได้ง่ายกว่า ซึ่งตรงกับคำว่า รูปภาพเพียงหนึ่งรูป สามารถแสดงความหมายได้มากมาย เช่น ในระบบสินค้าคงคลังในการตรวจสอบคลังสินค้าอะไหล่หรือเครื่องมือต่างๆ อาจมีการออกแบบให้สามารถดูลักษณะสินค้าในคลังได้ เป็นต้น

บทที่ 11
แนวความคิดเชิงวัตถุ (Object – Oriented Concept)

11.1. ความหมายของ Object

ความหมายของ Object แบ่งได้เป็น 2 ประเภทคือ สิ่งที่เป็นรูปธรรมและนามธรรม ที่มีอยู่จริงบนพื้นโลก (real-world)

- สิ่งที่มีลักษณะเป็นรูปธรรม (จับต้องได้) เช่น จักรยาน รถ สุนัข องค์กร ใบรายการสินค้า เป็นต้น
- สิ่งที่มีลักษณะเป็นนามธรรม (จับต้องไม่ได้) เช่น ความเป็นเจ้าของ เที่ยวบิน การวิ่ง แสง เป็นต้น

Object ทุก object จะประกอบด้วย 2 ส่วนปฏิบัติการคือ attribute และ method โดย attribute หรือเรียกได้อีกอย่างว่า data และ method ที่เรียกได้ว่า behavior จะมีรายละเอียดดังต่อไปนี้

Object data :

ในทาง o-o programming จะเรียก data ของ object ว่า attribute และในการเขียนโปรแกรม ส่วน attribute ของ object จะเป็นส่วนข้อมูลลับของ object ที่จะทราบเฉพาะภายใน object เท่านั้น (จะได้ทราบต่อไปว่าทำไม) ซึ่งในที่นี้จะขอยกตัวอย่างบางตัวอย่างของ attribute บน Object บางส่วนดังนี้

Object คน จะมี attribute คือ หลายเลขบัตรประชาชน, วันเกิด, เพศ, เบอร์โทรศัพท์ เป็นต้น

Object สุนัข จะมี attribute คือ ชื่อ, สี, พันธ์, อาหาร เป็นต้น

หมายเหตุ คำว่า data และ attribute นั้นมีความใกล้เคียงกันในความหมาย โดยถ้าพูดถึง "สี" ความหมายของ attribute คือ สี และ data คือ แดง, ดำ, เหลือง, เขียว เป็นต้น ดังนั้นคำว่า attribute จะครอบคลุมกว่า

Object Behavior :

Behavior ของ Object คือสิ่งที่ Object นั้นๆ สามารถทำได้ในทาง Procedural language จะเรียก Behavior เป็น Procedure , Function หรือ Subroutine ส่วนในทาง O-O programming จะเรียก Behavior เป็น Method ซึ่ง Method คือลำดับคำสั่งในการทำงานของ Computer ตัวอย่าง เช่น

Object คน จะมี Methods คือ การเดิน, ยืน , มีบัตรประจำตัว, มีรถ เป็นต้น

Object สุนัข จะมี Methods คือ การเห่า, การหายใจ ,การกระดิกหาง เป็นต้น

หมายเหตุ คำว่า Method sหรือ Behavior จะมีความหมายเหมือนกัน

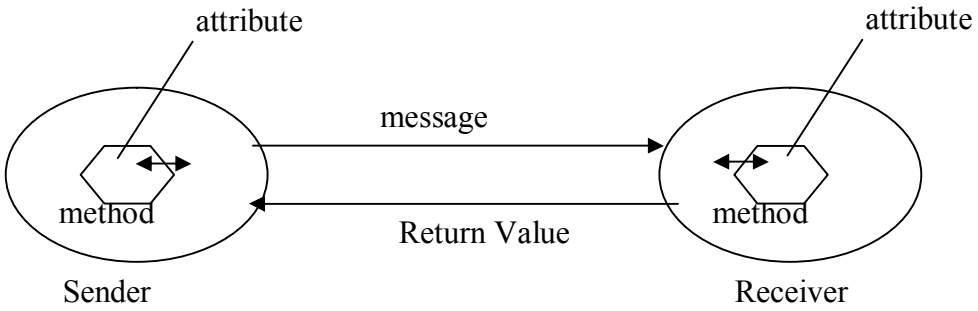
ภายใน Object หนึ่ง นั้นจะประกอบด้วย attribute และ method ซึ่งได้ทราบแล้วในข้างต้น โดยโครงสร้างของ Object แล้วนั้น สามารถเปรียบเทียบได้ว่าส่วนของ attribute จะได้รับการห่อหุ้ม (Encapsulate) ด้วย method ดังนั้นเมื่อ object ใดๆ ต้องการจะติดต่อกันจะต้องผ่านส่วนที่เรียกว่า method เท่านั้น จากลักษณะดังกล่าวส่วนของ method จึงต้องมีข้อมูลที่สามารถเปิดเผยให้ object อื่นทราบได้ดังนี้

- ชื่อของ method ที่ใช้ในการอ้างถึง
- คุณลักษณะ หรือ properties ของ object นั้นๆ
- Parameters ที่ใช้ในการรับส่งค่าระหว่าง method

หมายเหตุ คำว่า Object Orientation นั้นมาจากคำว่า Object ซึ่งแปลว่าวัตถุทั้งที่จับต้องได้ และจับต้องไม่ได้ และ Orientation ซึ่งมาจากคำว่า Orient ซึ่งเป็นคำกริยาที่แปลว่า นำทาง นำไป ซึ่งเมื่อนำคำทั้งสองมารวมกันแล้ว Object Orientation จะหมายถึง การใช้ Object เป็นตัวหลักเพื่อการพิจารณาความเป็นจริงต่างๆที่เกิดขึ้นในโลก

11.2. การสื่อสารระหว่าง Object

ในการสื่อสารระหว่างกันของ Object จำเป็นต้องมีส่วนที่เรียกว่า Message เป็นส่วนประกอบสำคัญร่วมด้วย เพื่อให้ทราบถึงจุดประสงค์ในการติดต่อระหว่างกัน สมมติว่า Object หนึ่งเป็น Sender และอีก Object หนึ่งเป็น Receiver ดังนั้นเมื่อ Sender ส่ง message ไปเพื่อร้องขอการทำงานบางอย่าง receiver จะส่งผลเป็นค่า return value กลับไปยัง sender



ตัวอย่าง ระบบการทำงานที่มี Object Employee และ Object Payroll ทำงานร่วมกัน โดยที่ Object Payroll ทำหน้าที่คำนวณค่าเงินเดือนของ Object Employee ส่วนประกอบภายในของทั้ง 2 Object ได้แก่

1. Object Employee ประกอบด้วย
- attribute : SocialSecurityNumber, Gender, DateofBirth
- method : GetsocialSecurityNumber, GetGender, getDateOfBirth
2. Object Payroll ประกอบด้วย
- attribute : Pay
- method : calculatePay

ในกรณีนี้ object Payroll จะต้องทำการร้องขอ SocialSecurityNumber (หมายเลขบัตรประชาชน) ที่เป็นส่วน attribute ของ object Employee โดย Object Payroll จะทำการส่ง message ไปยัง method GetSocialSecurityNumber เพื่อร้องขอข้อมูล SocialSecurityNumber เพื่อทำการคำนวณค่าเงินเดือนของพนักงาน

จากหลักการสื่อสารระหว่างกันของ Object 2 Object สามารถแยกประเภทของการสื่อสารออกได้เป็น 2 กรณี ดังนี้ กรณีที่ 1 Object Sender ร้องขอข้อมูลจาก Object receiver และจำเป็นต้องรอการคืนค่ากลับ ตัวอย่าง เช่น object School ต้องการถามรายชื่อของนักเรียน โดยส่ง message ไปยัง Object student และทางฝ่าย Object Student จะทำการส่งคืนค่า (Return) รายชื่อกลับไปยัง School Object

กรณีที่ 2 Sender ต้องการส่งค่าบางค่าไปเปลี่ยนแปลงข้อมูลที่ฝั่ง Receiver ซึ่งในกรณีนี้ Sender ไม่จำเป็นต้องสนใจการคืนค่ากลับ เช่น Object School ต้องการเปลี่ยนชื่อนักเรียน โดยส่ง message ไปที่ object Student เพื่อแจ้ง

การเปลี่ยนชื่อ จากนั้น object student จะทำการเปลี่ยน attribute ที่เป็นชื่อนักเรียนให้เป็นชื่อใหม่ ในกรณีนี้ object school จะไม่สนใจว่าจะมีค่า return กลับมาหรือไม่

11.3. Class

Object ทุกตัวจะต้องอยู่ใน Class ซึ่ง Class กับ Object เป็นสิ่งที่คู่กันเสมอ สามารถทราบรายละเอียดและคุณสมบัติ (Characteristic) ของ Object ได้ด้วยการดูที่ Class

Class คือ กลุ่มของ Object ที่มีโครงสร้างพื้นฐานพฤติกรรมเดียวกัน ดังนั้น Object ที่มีคุณสมบัติลักษณะเดียวกันนี้จะรวมกลุ่มอยู่ใน Class เดียวกัน (Object are grouped in class) จึงสามารถสรุปได้ว่า Class คือ ต้นแบบข้อมูล (Information) ที่มีไว้เพื่อสร้าง Object นั่นเอง Class นอกจากจะมีชื่อ Class ที่บอกคุณสมบัติของ Class นั้นแล้ว ยังมี Attribute และ Operation ต่างๆ ซึ่งเป็นตัวอธิบายรายละเอียด และหน้าที่ต่างๆโดยสามารถแสดงในลักษณะ Template ดังนี้

Name
Attributes
Operations
Responsibilities

ตัวอย่าง Class ของ Student

Student {abstract}
StudentIdNumber FirstName LastName Address City State Zipcode TelephoneNumber DateofBirth GradPointAverage Etc….
RegisterForCourse DropCourse ChangeAddressPhone RequestTranscript Etc…

ตัวอย่าง Class คือแบบรูปดาวที่สร้างจากแผ่นไม้ (Template) และ Object คือรูปดาวที่สร้างจากตัวแบบรูปดาวที่ทำหน้าที่เป็น Class นั้น ซึ่ง Object ที่ถูกสร้างขึ้นมาจะมีลักษณะเหมือนกับ Class ที่เป็นต้นแบบ แต่โดยคุณสมบัติของ Object แล้วจะสามารถเพิ่มคุณสมบัติเฉพาะของตัวเองขึ้นมาได้ ในที่นี้ได้แสดงให้เห็นว่าแต่ละ Object จะเพิ่มสีขึ้นมาเป็นคุณสมบัติเฉพาะตัวเองขึ้นมาได้ ในที่นี้แสดงให้เห็นว่าแต่ละ Object จะเพิ่มสีขึ้นมาเป็นคุณสมบัติเฉพาะของตัวเองขึ้นมาได้ แต่คุณสมบัติพื้นฐานจะได้รับการสืบทอดมาจาก class ที่เป็นต้นแบบ

ตัวอย่าง มีแม่นกตัวหนึ่ง แม่นกตัวนั้นสามารถมองได้ว่าเป็น Object แต่เมื่อมีลูกนกเกิดขึ้น ลูกนกก็มองได้ว่าเป็น Object เนื่องจากลูกนกจะสืบทอดคุณสมบัติจากแม่ ซึ่งทำให้เราสามารถมองได้ว่า แม่นกเป็น Class ๆ หนึ่งเช่นกัน

11.4. Encapsulation (Protection)

Encapsulation คือรากฐานอย่างหนึ่งของแนวความคิดในเชิง Object-Oriented ซึ่งข้อดีของ Encapsulation คือการป้องกัน attribute (data) ของ object จากความเสียหาย เพราะถ้าส่วนของโปรแกรมทั้งหมด อนุญาตให้มีการเข้าถึง Attribute ได้ตามที่ต้องการแล้วนั้น จะส่งผลให้ attribute นั้นง่ายต่อการถูกใช้อย่างผิดๆ ทำให้ค่า attribute เปลี่ยนแปลงไป ซึ่งก่อให้เกิดความเสียหายตามมา

Encapsulation คือการห่อหุ้ม attributes และ methods เข้าไว้ด้วยกัน

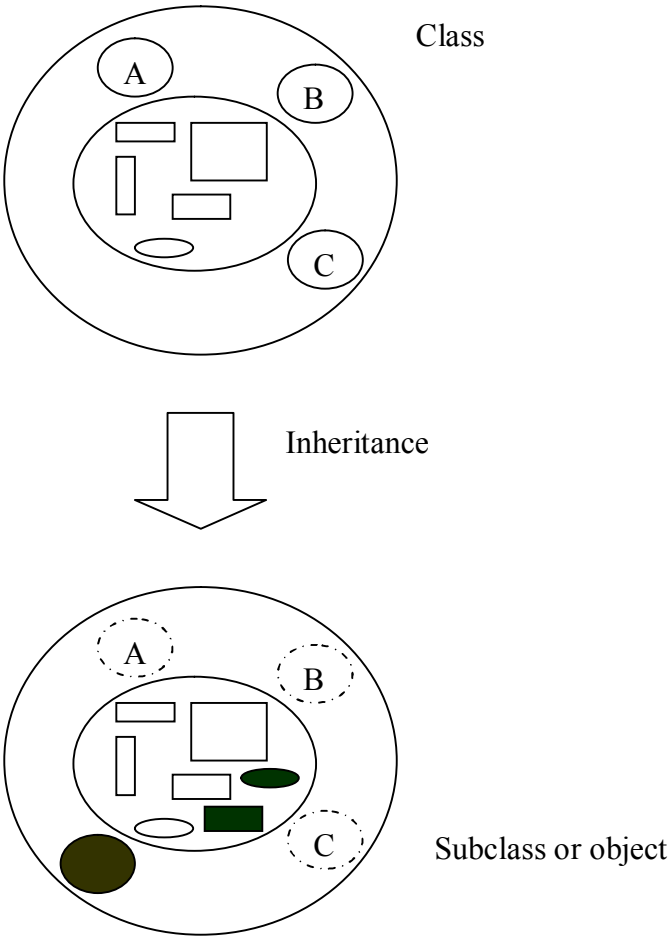
Encapsulation จะทำหน้าที่ป้องกันมิให้ Object อื่นที่อยู่ภายนอก เข้าถึง Object หนึ่งๆ ได้อย่างอิสระจะมีเฉพาะ methods ที่อยู่ใน Object เท่านั้นจะสามารถติดต่อกับ attribute ที่อยู่ใน Object เดียวกันได้ เรียกได้ว่าการ Encapsulation มีคุณสมบัติของ Information hiding

Information Hiding คือการจำกัดการมองเห็นข้อมูลภายใน Object เช่นการกำหนดคุณลักษณะเป็น “public” เพื่อให้สามารถเชื่อมต่อกับ Object ภายนอกได้ และกำหนดเป็น “private” เพื่อจำกัด attribute ให้อยู่ภายใน object เท่านั้น

11.5. Inheritance (การสืบทอดคุณสมบัติ)

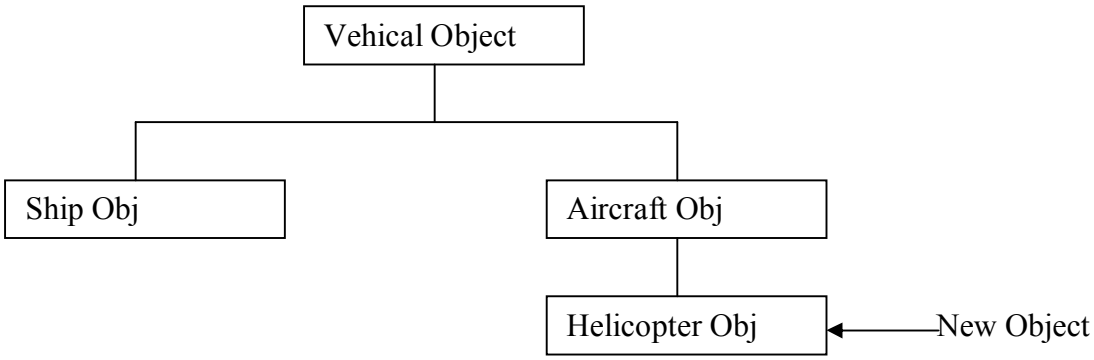
ใน Procedural Programming จะทำการจัดหาส่วนของโปรแกรมที่มีการใช้ซ้ำบ่อยๆ เพื่อนำกลับมาใช้ใหม่ (reuse) ในรูปของโปรแกรมย่อย (Procedure หรือ Function) ซึ่งเราจะสามารถเรียกใช้ซ้ำได้หลายครั้งตามต้องการ อย่างไรก็ตาม ใน O-O Programming มีคุณสมบัติข้อนี้เช่นกัน แต่ยังมีการออกแบบที่พิเศษมากขึ้นกว่าการ reuse คือความสามารถในการจัดการกับ Class และกลุ่มของ Class ต่างๆ โดยคุณสมบัติที่เรียกว่า “Inheritance” ที่ประกอบด้วยหลักการของ Superclass/Subclass, Abstraction และ is-a relationship ซึ่งหลักการต่างๆ เหล่านี้จะช่วยให้การออกแบบโครงสร้างโปรแกรมแบบ O-O ทำได้ง่ายขึ้น

Inheritance คือ คุณสมบัติที่ Class ๆ หนึ่ง สามารถสืบลักษณะของ attribute และ method ของ อีก Class หนึ่งได้การทำเช่นนี้ทำให้คุณสามารถ Create Class ใหม่ขึ้นโดยนำสาระสำคัญ ที่เหมือนกันของ attribute และ behavior(method) จาก class อื่นมาใช้ได้ ดังรูป

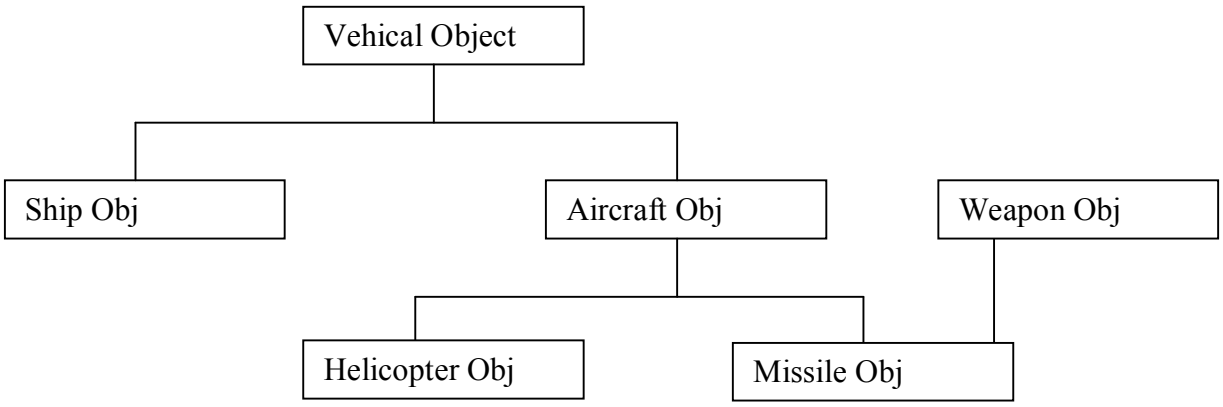


จากรูปที่ข้างต้น A, B, C คือ method ที่สืบทอดมาจาก Class เดิม ส่วน  คือ method ที่ Subclass สร้างขึ้น และ  , คือ attribute ที่สร้างขึ้นใช้ใน SubClass เพื่อแสดงลักษณะของตัวเองลักษณะการ Inheritance จะมี 2 แบบ คือ

แบบที่ 1 Object ใหม่ 1 Object ที่ถูกสร้างขึ้นมามีการสืบทอดลักษณะจาก Class แม่เพียง 1 Class จะเรียกว่า Single inheritance จะเห็นได้ว่า Object Helicopter จะสืบทอดลักษณะมาจาก Object Aircraft เพียง Object เดียว



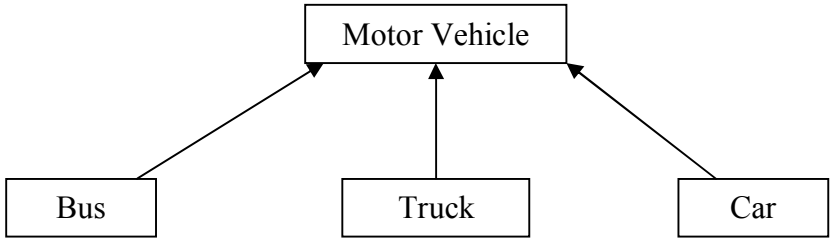
แบบที่ 2 บาง Object จะมีการสืบทอดลักษณะจาก Class แม่ 2 Class ขึ้นไป ซึ่งจะเรียกว่า Multiple inheritance ดังรูป



Single inheritance หมายถึง การที่ Object หนึ่งๆ ได้รับการสืบทอดลักษณะจาก Object อื่น เพียง 1 Object
Multiple inheritance หมายถึง การที่ Object หนึ่งๆ ได้รับการสืบทอดลักษณะจาก Object อื่นมากกว่า 1 Object จากรูปด้านบนจะเห็นว่า Object Missile จะเกิดจาก Object Aircraft และ Object Weapon การสืบทอดลักษณะแบบนี้มีข้อควรระวังคือ ชื่อของ attribute ในแต่ละ Object ที่ได้รับการสืบทอดมา ถ้าชื่อซ้ำกันจะเกิดความกำกวมขึ้น ซึ่งควรหลีกเลี่ยงการตั้งชื่อกันในแต่ละ Object สรุปได้ว่าคุณสมบัติทุกๆ สิ่ง (attribute,method) ที่อยู่ใน Parent Class จะถูกถ่ายทอดไปสู่ Childclass

11.5.1.Superclasses&Supclasses

Superclass เป็นคุณสมบัติหนึ่งที่เกิดจากการ Inheritance โดย Superclass หรือ Parent class หมายถึง class ที่มี attribute และ method ที่จะให้ class ทั่วไปสืบทอดคุณสมบัติจากตัวเอง ส่วน subclass หรือ child class หมายถึง class ที่สืบคุณสมบัติมาจาก parent class ในการสืบทอดคุณสมบัติจาก Superclass ไปยัง Subclass นั้น Subclass จะมีคุณสมบัติของ attribute และ method ทุกอย่างจาก Superclass
ตัวอย่างดังรูป



11.6. Polymorphism

Polymorphism หมายถึง การบอกแบบเดียว แต่ได้รับการตอบสนองหลายรูปแบบ (Poly=”many”, morph = “form” ซึ่งเป็นไปตามหลักการของเทคโนโลยีเชิงวัตถุที่กล่าวมาแล้ว คือ จะเน้นที่ตัวปฏิบัติการมากกว่าการปฏิบัติการ ดังนั้น รายละเอียดหรือหน้าที่ต่างๆ จะขึ้นอยู่กับเราว่าจะไปมีความสัมพันธ์กับตัวปฏิบัติการเหล่านั้นอย่างไร

ตัวอย่าง DrawChart() คือ ฟังก์ชันการวาด หมายรวมถึงความสามารถในการวาดรูปได้ในหลายๆ ลักษณะมิได้เฉพาะเจาะจง เช่น การวาดรูปวงกลม วงรี สามเหลี่ยม สี่เหลี่ยม ฯลฯ ซึ่งหมายความว่าหากต้องการให้มีการตอบสนองรูปวาดต่างๆ นั้น ผู้ใช้สามารถติดต่อได้ด้วยการใช้ฟังก์ชัน DrawChart() เพียงฟังก์ชันเดียว ส่วนจะมีการตอบสนองการวาดในลักษณะใดนั้นขึ้นอยู่กับรายละเอียด ดังนั้นผู้ใช้งานไม่จำเป็นต้องจดจำฟังก์ชันในการวาดต่างๆ มากมาย ซึ่งแตกต่างจากการเขียนโปรแกรมแบบเดิม ที่จำเป็นต้องมีฟังก์ชันการวาดต่างๆ มากมายในการวาดรูปเหล่านั้น เช่น ฟังก์ชันวาดรูปสี่เหลี่ยมฟังก์ชันวาดรูปสามเหลี่ยม วงกลมฯลฯ ดังนั้นหลักการของ Object-Oriented นี้จะสามารถลดความยุ่งยากในการจดจำคำสั่ง และลดคำสั่งลงได้มาก แต่ละ Class สามารถมีการตอบสนองที่ต่างกันต่อ method เดียวกันนี้เรียกว่า polymorphism

11.7. ความสัมพันธ์ในแบบ Composition

ตัวอย่าง Employee คือ Person และการทำงานของ Employee จะไม่มีการส่ง message ไปยัง person object เพราะ Employee object มีหน้าที่เพียงให้บริการต่อ person object นั้นสรุปได้ว่า Employee object คือ person object หนึ่งนั่นเองดังรูปแสดงความสัมพันธ์ Person Class และ Employee class ในลักษณะ inheritance อย่างไรก็ตาม Composition จะอยู่ในสถานะที่ต่างจาก inheritance เพราะ Composition จะแทนได้ด้วยการติดต่อกันระหว่าง Object ต่างๆ มีการรับส่ง message แลกเปลี่ยนกันเพื่อร้องขอการทำงานหรือให้บริการ ซึ่ง Composition จะช่วยให้ออกแบบระบบได้ง่ายขึ้น

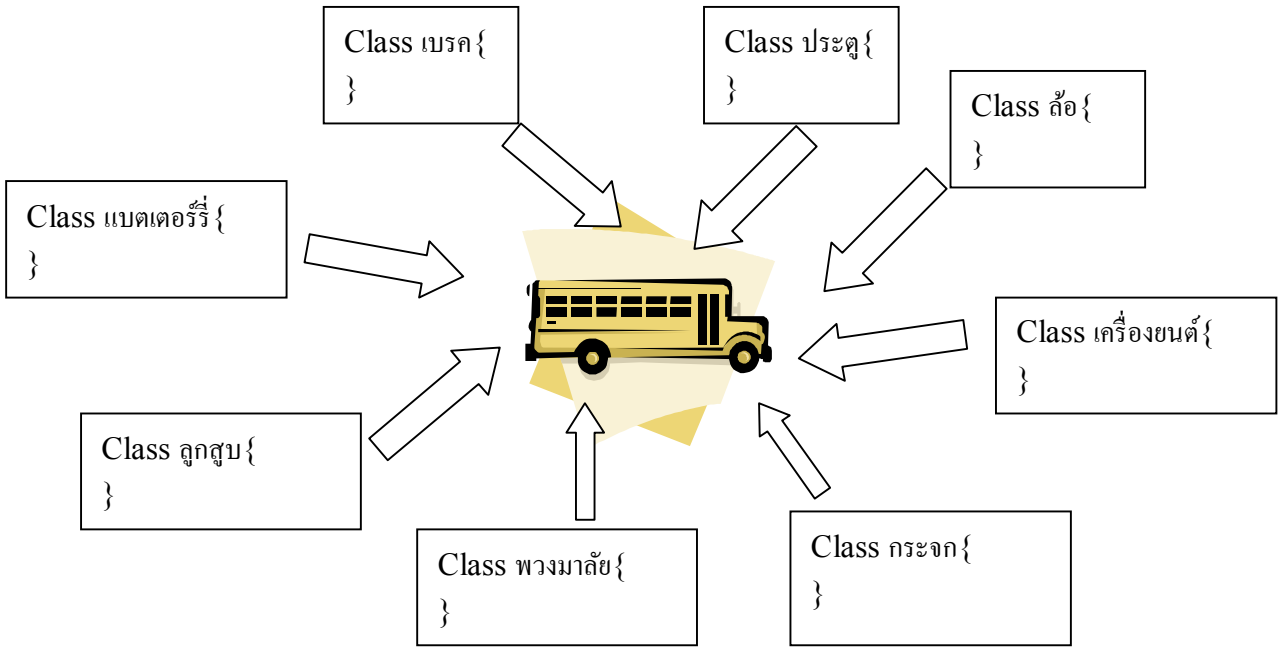
Person
- Name : String - SSNum :String - Age :int
+ getName : Stirng + getSSNum:string + getAge :int + setName :void + setSSNum : void + setAge :void

Employee
- CompanyID:string - จากอ้อtring - SSNum :String - Age :int
+ getName : Stirng

+ getSSNum:string
+ getAge :int
+ setName :void
+ setSSNum : void
+ setAge :void

ข้อดีของ Composition มีดังนี้

1. ทำให้ระบบมีความซับซ้อนน้อยที่สุด โดยการจดจำเพียงข้อมูลชิ้นใหญ่ๆ ที่ประกอบจากส่วนประกอบย่อยๆ หลายๆ ส่วน เช่น เมื่อพูดถึง พวงมาลัย เกียร์ เครื่องยนต์ เบรก เบาะนั่ง กระจก ฯลฯ สิ่งที่กล่าวมาทั้งหมดนี้สามารถพูดได้ง่ายๆว่า ‘‘รถยนต์’’ และสามารถรู้ได้ทันทีว่าภายในรถยนต์ประกอบด้วยส่วนต่างๆ เหล่านี้ โดยไม่จำเป็นต้องบรรยายถึงส่วนประกอบย่อยๆ ทั้งหมด
2. Composition มีส่วนช่วยในการ reuse เช่น การแลกเปลี่ยนส่วนต่างๆ ของระบบสามารถทำได้เพราะชิ้นส่วนต่างๆ เป็น object ที่มีการทำงานเป็นของตนเอง และไม่ถูกควบคุมจาก object อื่น ยกตัวอย่าง รถยนต์หนึ่งคัน ล้อที่ทำหน้าที่ช่วยในการเคลื่อนที่ของรถยนต์นั้นสามารถถอดเปลี่ยนได้และสามารถใช้ล้อยี่ห้ออื่นได้เช่นกัน ไม่จำเป็นจะต้องเป็นชิ้นส่วนเดิมเท่านั้น
3. การแบ่งระบบออกเป็นส่วนการทำงานย่อยๆ นั้น ส่วนต่างๆ จะสามารถทำงานได้เป็นอิสระต่อกันส่งผลให้การทดสอบหรือปรับปรุงแก้ไขแต่ละส่วนสามารถทำได้อย่างอิสระ ไม่ส่งผลกระทบต่อ object หรือส่วนย่อยอื่นๆ ในระบบ จะเห็นได้ว่า class ของรถยนต์หนึ่งคันจะประกอบไปด้วย class ต่างๆ โดยที่แต่ละ class เหล่านี้มีหน้าที่เฉพาะในการทำงานของตนเอง เช่น Class ล้อ จะทำหน้าที่หมุนเพื่อให้รถเคลื่อนที่ class ประตู จะทำหน้าที่เปิด และปิดเมื่อต้องการเข้าหรือออกจากรถ เป็นต้น เมื่อจะต้องทำการแก้ไขส่วนประกอบภายในของ class รถยนต์เช่น แบตเตอรี่หมดทำการเปลี่ยนแบตเตอรี่ (ข้องเกี่ยวกับ class แบตเตอรี่เพียง class เดียว)หรือต้องการให้เครื่องยนต์มีความแรงเพิ่มมากขึ้น สามารถเพิ่มกำลังความเร่งให้กับเครื่องยนต์ได้ (ข้องเกี่ยวกับ class เครื่องยนต์เพียง classเดียว) หรือเมื่อยางล้อแบน สามารถสูบลมเพิ่มให้กับยางรถยนต์ได้ (ข้องเกี่ยวกับ class ยางรถยนต์เพียง class เดียว) จะเห็นได้ว่าเมื่อแยกส่วนประกอบต่างๆ ออกเป็น Class จะสะดวกในการทำงานของระบบมากขึ้น ดังภาพแสดงส่วนประกอบของ Class รถยนต์ในแบบ composition



11.7.1. ชนิดของ Composition สามารถแบ่งออกได้เป็น 2 ประเภทดังนี้คือ

11.7.1.1. Aggregations

11.7.1.2. Association

ซึ่งทั้ง 2 ประเภทนี้จะแทนที่การทำงานร่วมกันของ object โดย aggregation จะแสดงให้เห็นส่วนประกอบย่อยๆ ที่ประกอบขึ้นเป็นส่วนประกอบหลัก 1 ชิ้นงานที่สมบูรณ์ เช่น Computer จะประกอบไปด้วย monitor, CD-ROM, Harddisk, Power Supply, Mainboard, Sound Card เป็นต้น ส่วน Association จะแสดงให้เห็นถึงส่วนประกอบย่อยหลายๆ ส่วนที่เป็นอิสระต่อกัน แต่มาประกอบร่วมกันขึ้นเพื่อให้เป็นชุดงาน 1 ชิ้นที่สมบูรณ์ เช่น Stereo จะประกอบด้วย ลำโพง, หูฟัง, รีโมต เป็นต้น ซึ่งทั้งสองวิธีนี้จะต่างกันที่วิธีการคิด ดังนี้

11.7.1.1. Aggregations

ส่วนใหญ่แล้วิธีการของ Composition จะคิดในแนวของ Aggregation ซึ่งหมายถึงการนำส่วนประกอบย่อยๆ หลายๆ อย่างมาประกอบเป็นหนึ่ง object ที่สามารถใช้งานได้อย่างสมบูรณ์ เช่น TV เครื่องหนึ่ง ผู้คนที่ผ่านไปมาคงสังเกตแต่เพียงว่าใช้งานอย่างไร ขนาดจอกี่นิ้ว แต่จะไม่มีใครสนใจว่าภายในประกอบด้วยอะไรบ้าง มีหลอดภาพเป็นอย่างไร มี transistor เป็นอย่างไร หรือเมื่อต้องการเลือกซื้อรถยนต์คันหนึ่ง โดยทั่วไปจะพิจารณาเพียงว่า มีเครื่องยนต์แรงหรือไม่ มีสิ่งอำนวยความสะดวกเช่น Stereo หรือไม่ ซึ่งทั้งหมดนี้เป็นการมองภาพรวมทั้งหมดของวัตถุ มองวัตถุว่าเป็นชิ้นงานที่สมบูรณ์หนึ่งชิ้น นั่นคือการมองในแง่มุมของ Programmer ซึ่งเป็นผู้ทำการติดต่อกับ Class หรือเป็นผู้ใช้ Class ถ้าเป็นตัวอย่างโครงสร้างของ object car เมื่อ Programmer ต้องการจะติดต่อกับ Object car หรือเรียกใช้ object car จะสามารถทำได้โดยสะดวกมากกว่าที่จะต้องเรียกใช้การทำงานของ object Engine, object door, object stereo เป็นต้น ที่เป็นส่วนประกอบของ object car เรียกได้ว่า object car คือส่วน interface ที่สร้างไว้เพื่อใช้ติดต่อกับ programmer หรือ object อื่น แต่ทั้งนี้ในมุมมองของผู้ออกแบบระบบจะต้องมองถึงรูปแบบการสร้าง object car ภายในระบบด้วย นั่นคือส่วนประกอบภายในจะต้องประกอบด้วย object

ย่อยๆ เพื่อสะดวกในการทำงานและการแก้ไขระบบในภายหลัง เช่น โครงสร้าง object รถยนต์ในมุมมองของผู้ออกแบบระบบจะต้องสร้าง object ต่างๆ ซึ่งเรียกได้ว่าเป็นการออกแบบด้วยวิธี Composition แบบ Aggregation ดังนั้นวิธี composition แบบ aggregation คือการสร้าง object ย่อยๆ ไว้ภายในระบบเพื่อพึ่งพาอาศัยกันในการทำงาน โดยยึดหลักที่ว่าให้ส่วน interface แก่ user (programmer หรือ object อื่น) ให้น้อยที่สุด และจากลักษณะของ object ย่อยๆ ยังช่วยให้การเปลี่ยนแปลงแก้ไขระบบแก่ผู้ออกแบบและพัฒนาระบบให้เป็นไปได้ง่ายขึ้นด้วย อีกตัวอย่างหนึ่งของวิธี Composition แบบ aggregation คือ โครงสร้างการทำงานของระบบคอมพิวเตอร์ ซึ่งจะสามารถออกแบบในแบบ aggregation ได้ เมื่อกล่าวถึงโครงสร้าง Object Computer ผู้ออกแบบระบบจำเป็นที่จะต้องนึกถึงส่วนประกอบต่างๆ เหล่านี้อย่างเป็น Object ย่อยๆ

11.7.1.2. Association

ในส่วนของ Aggregation นั้นจะแสดง object ย่อยภายใน ที่อยู่ภายใต้ Object หนึ่ง แต่ Association จะแสดงส่วนของ object ทั้งหมดที่เป็นอิสระต่อกัน ที่มีการทำงานเฉพาะไม่ขึ้นกับส่วนของ object อื่นๆ แต่นำ Object เหล่านี้มาประกอบกันขึ้น เพื่อให้มีการทำงานที่สมบูรณ์ เช่นระบบคอมพิวเตอร์หนึ่งเครื่องจะประกอบด้วย object ของ monitor, keyboard, mouse และกล่องๆ หนึ่งที่ทำการควบคุม (main box) เป็นต้น โดยแต่ละส่วนเหล่านี้มีหน้าที่ให้บริการเฉพาะอย่างแก่คอมพิวเตอร์ mouse ทำหน้าที่ให้บริการแก่ main box ในการเคลื่อนที่ของลูกศรบนจอคอมพิวเตอร์ หรือจอภาพทำหน้าที่ให้บริการแสดงผลภาพผ่านหน้าจอแก่ main box ฯลฯ Aggregation คือ object ที่ซับซ้อนซึ่งประกอบจาก object อื่น ส่วน Association คือ object หนึ่งที่ต้องการบริการจาก object อื่น ดังนั้น object ที่แยกออกมาได้จึงต่างกัน aggregation จะเน้นให้แสดงส่วน interface ให้น้อยที่สุดคือ ส่วน association จะเน้นให้แสดงส่วนของ object ต่างๆ ที่นำมาประกอบกัน และแต่ละส่วนของ object เหล่านี้สามารถแยกทำงานได้อย่างอิสระ โดยขอบริการใช้งานกันระหว่าง object โดยมีส่วน interface เป็นของตนเอง

11.7.2. ความเหมือนและต่างกันระหว่าง Association และ Aggregation

ความเหมือนระหว่าง Association และ Aggregation

- เนื่องจากการออกแบบด้วยวิธี Composition นั้น ทั้งแบบ Aggregation และ Association แต่ละส่วนจะได้รับการสร้างเป็น Object ทำให้ง่ายในการ reuse เช่น ในแบบ association เมื่อ computer เครื่องอื่นต้องการใช้ printer สามารถร้องขอบริการมายัง object printer ได้ หรือ ในแบบ aggregation การทำงานของ object mainboard ที่ได้สร้างขึ้นไว้แล้วนั้น สามารถนำโครงสร้างของ Object mainboard นี้ไปใช้กับ object อื่นได้ โดยไม่จำเป็นต้องสร้าง object mainboard ขึ้นมาใหม่
- ลักษณะของ Object ทั้ง Aggregation และ Association จะมีการทำงานของแต่ละ object เป็นลักษณะเฉพาะที่สามารถปรับเปลี่ยน หรือแก้ไข โครงสร้างภายใน object นั้นได้โดยสะดวก ไม่กระทบต่อการทำงานของ object อื่น

ความแตกต่างระหว่าง Association และ Aggregation

- ความสัมพันธ์ระหว่าง Object ส่วนของ aggregation นั้นวัตถุแต่ละชิ้นจะทำงานโดยพึ่งพาอาศัยกัน ซึ่งถ้าวัตถุชิ้นใดขาดหายไปจากส่วนประกอบจะทำให้การทำงานรวมทั้งหมดกระทบกระเทือนไปด้วย เช่นระบบคอมพิวเตอร์ เมื่อขาด object mainboard ไป จะส่งผลให้ระบบคอมพิวเตอร์ไม่สามารถทำงานต่อไปได้

- ส่วนของ Association วัตถุแต่ละส่วนจะทำงานในแบบการขอบริการซึ่งกันและกัน ดังนั้นเมื่อขาดการทำงานของวัตถุชิ้นใดไป อาจจะทำให้การทำงานโดยรวมที่เกิดขึ้นไม่สมบูรณ์ แต่ไม่ถึงกับทำให้การทำงานทั้งหมดหยุดชะงัก เช่น ระบบคอมพิวเตอร์เมื่อขาด Object Printer ไประบบคอมพิวเตอร์ยังคงทำงานได้เป็นปกติ สามารถเปิดเครื่อง เปิดโปรแกรมการทำงานต่างๆ ได้ จะมีเพียงการขอบริการไปยัง Object Printer นั้นถูกตัดขาดไป ทำให้ object printer ไม่ทำงานเท่านั้น
- สิ่งหนึ่งที่ต้องสังเกตคือ ต้องแยกความแตกต่างระหว่าง Association และ Aggregation ให้ได้อย่างชัดเจน ซึ่งทั้งสองคำนี้มีหลักการที่แตกต่างกัน ดังนั้นในการออกแบบระบบโปรแกรม จะต้องตัดสินใจให้ได้ว่าจะใช้วิธีออกแบบด้วยวิธี Association หรือ Aggregation ซึ่งโดยปกติจะขึ้นอยู่กับระบบที่จะทำการออกแบบและการวิเคราะห์ของการออกแบบวิธีใดเหมาะสมมากกว่ากัน

บทที่ 12
Object Oriented Programming and UML(Unified Modeling Language)

ในสมัยก่อนการเขียนโปรแกรมคอมพิวเตอร์ยังเป็นเรื่องยุ่งยากและไม่แพร่หลาย ผู้คนส่วนใหญ่ยังนิยมใช้งานทางด้าน hardware ทั้งๆ ที่ software สามารถช่วยลดต้นทุน และทำการเปลี่ยนแปลงได้ง่ายกว่าทั้งนี้เนื่องจากข้อจำกัดของ Software ที่ซับซ้อน ภาษาที่ใช้ในการเขียนโปรแกรมที่ค่อนไปในทางภาษาเครื่องซึ่งยากที่มนุษย์จะเข้าใจ รวมทั้งการลงทุนในการเปลี่ยนแปลงระบบในช่วงแรกค่อนข้างราคาสูง เป็นเหตุให้การนำ Software มาใช้งานไม่เป็นที่นิยม ต่อมาเมื่อวิวัฒนาการทางด้านคอมพิวเตอร์มีความก้าวหน้ามากขึ้นเรื่อยๆ ควบคู่ไปกับการพัฒนาทางด้าน Software จากคอมพิวเตอร์ระดับ mainframe มาเป็นคอมพิวเตอร์ส่วนบุคคล ที่มีราคาถูกลง จนกระทั่งคนทั่วไปสามารถมีคอมพิวเตอร์ส่วนบุคคลใช้งานภายในบ้านได้นั้นซึ่ง Software ได้รับการพัฒนาให้สามารถใช้งานบนคอมพิวเตอร์ส่วนบุคคลได้เช่นกัน การพัฒนาทางด้าน Software ในระยะหลังจะเพิ่มความสะดวกในการใช้งานให้ผู้ใช้มากขึ้น ภาษาที่ใช้ในโปรแกรมเป็นภาษาชั้นสูง นอกจากนี้ยังมีเทคนิคมากมายในการสร้าง Software ที่สามารถนำไปใช้งานได้หลายรูปแบบ ดังนั้น Software จึงได้รับความนิยมสูงขึ้น ดังจะเห็นได้จากการที่มี Software ใหม่ๆ เกิดขึ้นอยู่เสมอในปัจจุบัน

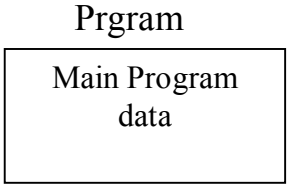
12.1. เทคนิคในการเขียนโปรแกรม

ตั้งแต่อดีตจนถึงปัจจุบันนี้มีภาษาทาง Programming มากมายให้เลือกใช้งาน เช่น Assembly, Pascal, C, C++, LISP, Smalltalk, Java, VB เป็นต้น โดยเทคนิคในการเขียนโปรแกรมภาษาต่างๆ เหล่านี้สามารถสรุปอย่างกว้างๆ ได้ 4 ประเภทดังนี้

1. Unstructured Programming
2. Procedural Programming
3. Modular Programming
4. Object-Oriented Programming

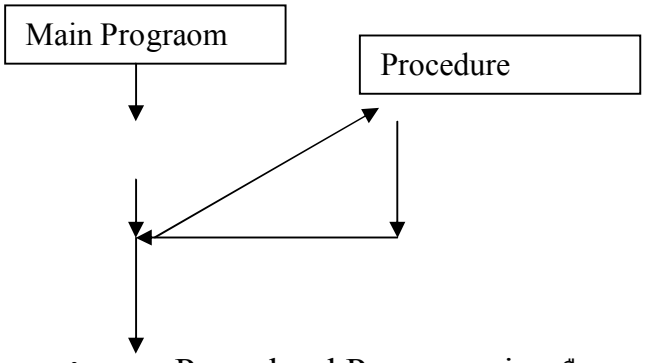
12.2. Unstructured Programming

การเขียนโปรแกรมแบบไม่มีโครงสร้าง จะประกอบด้วยส่วนของโปรแกรมหลัก (main program) มีกลุ่มของข้อมูล (data) อยู่ภายในโปรแกรมหลัก ซึ่ง data นี้สามารถเรียกใช้ได้ทั่วทั้งโปรแกรม ส่วนของโปรแกรมที่มีการใช้งานซ้ำในตำแหน่งที่แตกต่างกันในโปรแกรมจะต้องคัดลอกข้อมูลมาวางไว้อีกที่หนึ่ง (เพราะวิธีการนี้จะยังไม่มีการเขียนโปรแกรมย่อย) การเขียนโปรแกรมในลักษณะนี้จะมีข้อเสีย เนื่องจากใช้เนื้อที่ในการเก็บโปรแกรมค่อนข้างมาก ตัวโปรแกรมจะมีขนาดใหญ่ แสดงภาพโปรแกรมแบบไม่มีโครงสร้าง จะรวม main program และ data ไว้เป็นก้อนเดียวกัน



12.3. Procedural Programming

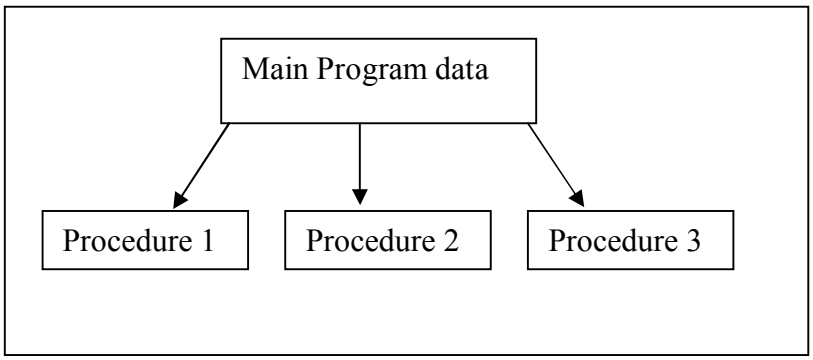
การเขียนโปรแกรมในลักษณะนี้ จะนำประโยค (Statement) ที่ใช้ซ้ำหลายๆ ครั้งรวมเข้าเป็นกลุ่มเดียวกัน เรียกว่า Procedure โดยที่ Procedure จะสามารถเรียกใช้ซ้ำได้ตามต้องการในตำแหน่งอื่นๆ ภายในโปรแกรม ผ่านฟังก์ชัน call (ในภาษา C) ดังรูป



จากรูปลักษณะการทำงานของ Procedural Programming คือ main program สามารถกระโดดมาทำงานที่ Procedure และกลับไปทำงานที่โปรแกรมหลักต่อไปได้ การทำงานจะเกิดเริ่มแรกที่โปรแกรมหลัก (main program) ซึ่งจะทำการเรียก procedure โดย

ผ่านฟังก์ชัน call และข้อมูล (data) เข้าไปใน Procedure สุดท้ายผลลัพธ์ของข้อมูลจะเปลี่ยนแปลงไปเมื่อการทำงานของ Procedure เสร็จสิ้น (ในที่นี้ procedure หมายถึงไม่มี subprocedure อยู่ภายใน) ดังรูป

Program

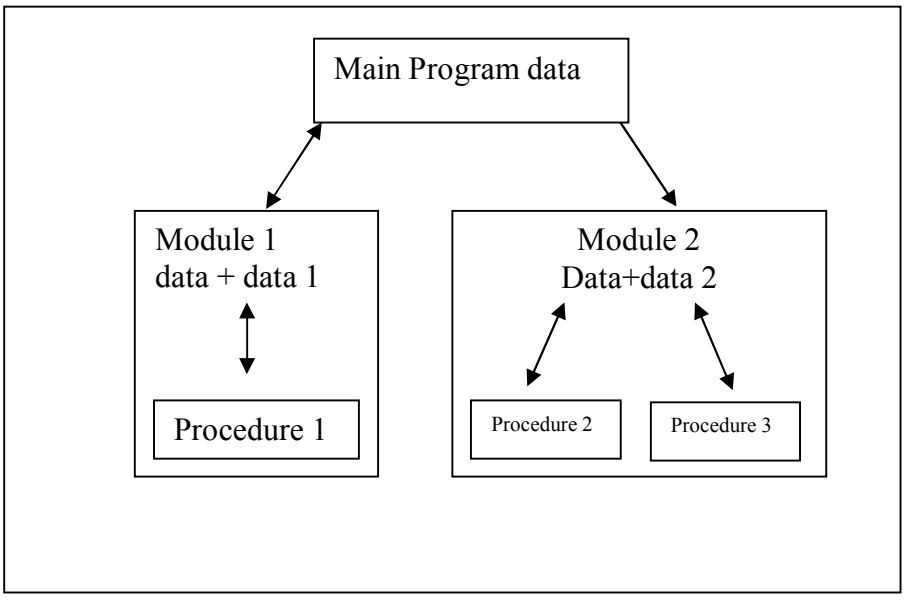


โครงสร้างการทำงานในลักษณะนี้มีข้อดีคือ สามารถหาจุดผิดพลาด (error) ที่เกิดขึ้นได้ง่ายตัวอย่างเช่น ถ้ามี procedure ถูกสร้างขึ้นภายในโปรแกรม และ procedure นี้สามารถทำหน้าที่ได้อย่างถูกต้อง แน่นอนว่าต่อไปในทุกๆ ครั้งที่เรียก procedure นี้การทำงานจะถูกต้องเสมอ ดังนั้นถ้ามีจุดผิดพลาดที่เกิดขึ้นในโปรแกรม เป็นไปได้ว่าจะเกิดขึ้นที่จุดอื่นที่ไม่ได้อยู่ใน procedure นี้ ทำให้การค้นหาจุดผิดพลาดมีช่วงการค้นหาที่แคบลง จากรูปด้านบน จะเห็นได้ว่าในขณะนั้นโปรแกรมได้แบ่งออกเป็นโปรแกรมย่อยๆหลายๆ โปรแกรมย่อยซึ่งแต่ละโปรแกรมย่อย จะทำงานขึ้นกับการควบคุมกับการควบคุมของโปรแกรมหลัก (ให้สังเกตว่า data จะอยู่ที่โปรแกรมหลัก)

12.4. Modular Programming

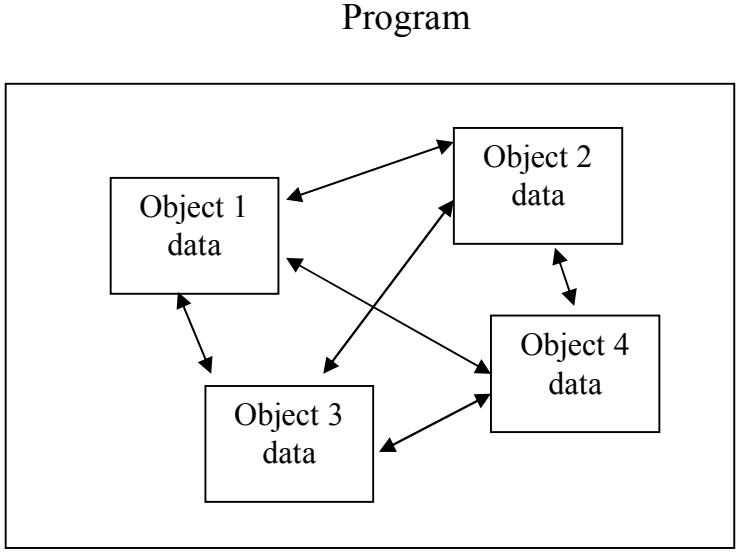
ในส่วนของ Modular Programming จะต่อเนื่องมาจาก Procedure Programming คือ จะนำ Procedure หลายๆ procedure มารวมกลุ่มกันเป็น modules และในแต่ละ module จะมี data เป็นของตนเอง โปรแกรมจะทำโดยเรียก procedure ที่อยู่ในแต่ละ module รวมทั้งมีการผ่านค่า parameter(data) ร่วมด้วย ซึ่งสภาวะการทำงานภายใน module จะเกิดขึ้นเพียงหนึ่ง procedure ต่อหนึ่ง module และภายในหนึ่งโปรแกรมจะเกิดสภาวะการทำงานเพียงหนึ่ง module ต่อโปรแกรมทั้งหมด Modular Programming ซึ่งประกอบไปด้วยหลายๆ module และแต่ละ module จะประกอบไปด้วย procedure และ data ที่เป็น data ของโปรแกรมหลัก และเฉพาะที่ใช้งานใน module นั้น ดังแสดงในรูป

Program



12.5. Object-Oriented Programming (o-o programming)

o-o programming เป็นวิธีการเขียนโปรแกรมแบบใหม่ ซึ่งจะเรียกโปรแกรมย่อยที่ทำหน้าที่เฉพาะว่า object และยังได้รวม data เข้าเป็นส่วนของ object ด้วย นอกจากนี้ o-o ยังมีโครงสร้างที่พิเศษนอกเหนือจาก procedure programming หรือ module programming คือ การทำงานของแต่ละ object จะไม่สิ้นสุดลง เมื่อกระบวนการทำงานของ โปรแกรมจบการทำงานบน object นั้นแล้ว ซึ่ง object ที่อยู่ในโปรแกรมเหล่านี้ จะพร้อมเสมอในการเริ่มทำงาน ต่อจากตำแหน่งเดิมเมื่อเกิดการเรียกใช้งาน object อีกครั้ง ซึ่งแตกต่างกับการเขียนโปรแกรมในแบบอื่นๆ ที่โปรแกรมย่อยจะต้องเริ่มการทำงานใหม่เมื่อถูกเรียกใช้อีกครั้ง ดังนั้นเทคนิคนี้จึงสามารถช่วยลดปัญหาที่จะเกิดขึ้นจากการเขียนโปรแกรมด้วยวิธีอื่นๆได้ Object-Oriented Programming โดยแต่ละ object สามารถติดต่อกันได้โดยการส่ง message ไปที่ object อื่น ดังรูป



12.6. ความแตกต่างระหว่าง Procedural Programming และ o-o Programming

วิธีการเขียนโปรแกรมที่กล่าวมาข้างต้น สามารถแบ่งออกได้เป็น 2 แนวคิดหรือ 2 วิธีการในการพัฒนา Software ได้แก่วิธีการทาง procedural programming และวิธีการทาง o-o programming ซึ่งมีความแตกต่างกันดังนี้

12.6.1. โครงสร้างของ Programming

แนวคิดแบบ procedural programming ภายในโครงสร้างของโปรแกรมจะแบ่งเป็น 2 ส่วนคือส่วนของโปรแกรมหลัก (main program) ที่มี data เป็นส่วนประกอบ และส่วนของโปรแกรมย่อย (procedures หรือ functions) โดย data ที่ประกาศใช้อยู่ภายในโปรแกรมหลักนั้น จะถูกเรียกโดยโปรแกรมย่อยต่างๆ ที่อยู่ภายในโปรแกรม ลักษณะของ data ที่มีการประกาศใช้ทั่วทั้งโปแกรมนั้นเรียกว่าเป็นการประกาศใช้แบบ global แนวคิดแบบ o-o programming dataจะได้รับการประกาศใช้เฉพาะภายใน object แต่ละ object เท่านั้น (จะไม่ประกาศเป็น global) ซึ่ง object อื่นสามารถเข้าถึง data ได้โดยผ่าน method ของ object ที่เป็นเจ้าของ data เท่านั้น

จะเห็นได้ว่า โครงสร้างตามแนวคิด o-o programming จะดีกว่าแบบ procedural programming เนื่องจากการประกาศใช้ data แบบ global จะมีผลเสียต่อการควบคุมการเปลี่ยนแปลง data ของโปรแกรมซึ่งยากต่อการแก้ไขโปรแกรมในภายหลัง

12.6.2. ช่วงชีวิต (Life Time)

ใน procedural programming ช่วงชีวิตของโปรแกรมย่อยนั้นจะเท่ากับช่วงเวลาการทำงานของโปรแกรมที่ผ่านเข้ามา และออกจากโปรแกรมย่อย ดังนั้นเมื่อโปรแกรมหลักดำเนินงานผ่านโปรแกรมย่อยไปแล้วค่าที่อยู่ในโปรแกรมย่อยนั้นจะกลับสู่สภาพเดิม เหมือนขณะโปรแกรมย่อยอยู่ในสภาพเริ่มต้น ใน o-o programming ช่วงชีวิตของ object จะเริ่มตั้งแต่โปรแกรมเริ่มทำงานจนกระทั่งสิ้นสุดการทำงาน หรือเมื่อ object นั้นถูกต้องตัดขาดออกจากโปรแกรม ซึ่ง object จะถูกตัดขาดจากโปรแกรมเมื่อไม่มี object ใดๆ เรียกใช้ object นั้นอีก(ไม่มีคำสั่งจาก object ใดๆที่เรียกใช้ object นั้นอีก)

12.6.3. การส่งข้อมูลบน network (เครือข่าย)

procedural programming จะแบ่งส่วน data ของ program ออกจากโปรแกรมย่อยอย่างชัดเจนดังนั้นการส่งข้อมูลข้ามระบบในทาง Procedural programming จะเข้าใจว่าปลายทางของอีกฝั่งจะมี program ที่สามารถจัดการกับ data ได้ ดังนั้นจะมีเพียง data เท่านั้นที่จะถูกส่งข้าม network ไป ใน o-o programming ส่วนของ data และ method จะถูกห่อหุ้มเข้าด้วยกันเป็น object หนึ่งๆ ซึ่งเป็นข้อดีอย่างหนึ่งของ o-o programming เช่นในระบบ network ข้อมูลของ object ที่ถูกส่งข้ามไปนั้นจะมีทั้ง data และ operation รวมกันอยู่ จากความแตกต่างในด้านต่างๆระหว่าง procedural programming และ o-o programming ซึ่งได้เปรียบเทียบให้เห็นความแตกต่าง และบอกถึงข้อด้อยของระบบแบบเดิม (procedural programming) รวมทั้งแนวทางแก้ไขด้วยแนวคิดแบบใหม่ (o-o programming) ที่ให้ประสิทธิภาพในการทำงานที่ดีกว่า

12.7. เทคนิคต่างๆ ของ o-o programming เทคนิคต่างๆ เหล่านี้เป็นคุณสมบัติที่มีอยู่ใน o-o programming ได้แก่

12.7.1. ความสามารถในการเพิ่มหรือขยายโปรแกรม (Extendibility)

12.7.2. ความสามารถในการนำกลับมาใช้ใหม่ (Reusability)

12.7.3. ความสามารถในการเข้ากันได้กับส่วนประกอบอื่นๆ (Compatibility)

12.7.4. ประสิทธิภาพ (Efficiency)

12.7.5. ง่ายในการใช้ (Easy to uses)

12.7.6. ง่ายในการออกแบบ (Easy to designs)

12.7.1. ความสามารถในการเพิ่มหรือขยายโปรแกรม (Extendibility)

การเปลี่ยนแปลงแก้ไข หรือเพิ่มเติมบางส่วนของโปรแกรม สำหรับโปรแกรมขนาดเล็ก มักจะทำได้ไม่ยุ่งยากนัก แต่สำหรับโปรแกรมที่มีขนาดใหญ่ ซึ่งค่อนข้างลำบากอย่างมากในการทำการเปลี่ยนแปลงเนื่องจากการแก้ไข-เพิ่มเติมบางส่วนลงในโปรแกรมเพียงเล็กน้อย จะเสี่ยงมากต่อผลกระทบที่อาจจะทำให้ระบบการทำงานของโปรแกรมทั้งระบบล่มได้ ในระบบการทำงานแบบเดิม (procedural programming) ก่อนที่จะทำการเปลี่ยนแปลงโปรแกรมได้ จะต้องทำการศึกษาการออกแบบและการสร้างโปรแกรม ที่ค่อนข้างจะยุ่งยากมาก ด้วยเหตุที่ data ของระบบสามารถจะถูกเปลี่ยนแปลงได้โดยทุกส่วนของโปรแกรม ซึ่งเป็นการยากที่จะควบคุมผลกระทบที่อาจเกิดขึ้น ดังนั้น o-o programming จึงได้แก้ปัญหาโดยใช้เทคนิคบางประการในการ implement คุณสมบัติต่างๆ ของ o-o programming มีส่วนสนับสนุนความสามารถในการเพิ่มหรือขยายโปรแกรม ดังนี้

- การออกแบบระบบ : หลักการในการออกแบบที่มองทุกอย่างให้เป็น object นั้นทำให้ง่ายในการออกแบบ เพราะทุกชิ้นส่วนของโปรแกรมสามารถนำมาเชื่อมต่อกันได้ และไม่ว่าจะเป็นการเปลี่ยนแปลงระบบหรือแก้ไขใดๆในระบบจะทำได้ง่ายขึ้น เนื่องจากแต่ละส่วนในโปรแกรมมีการทำงานเป็นส่วนๆเฉพาะ
- ไม่รวมที่ศูนย์กลาง : แต่ละ object มีความเป็นอิสระต่อกัน ทำให้การเปลี่ยนแปลงต่อ object หนึ่งๆ ไม่ส่งผลกระทบต่อ object อื่นๆ ทั้งหมดภายในระบบ วิธีของ o-o จะมีสถาปัตยกรรมที่ช่วยให้ผู้ออกแบบโปรแกรมสามารถออกแบบโครงสร้างได้ง่ายขึ้น โดยจะไม่รวมการทำงานให้อยู่ที่ศูนย์กลางเท่านั้น ซึ่งเป็นผลดีต่อการทำงานของโปรแกรมเป็นอย่างมาก

12.7.2. ความสามารถในการนำกลับมาใช้ใหม่ (Reusability)

บ่อยครั้งที่ระบบการทำงานของโปรแกรม จะมีการทำงานในรูปแบบเดิมซ้ำๆ ดังนั้นการนำบางส่วนของโปรแกรมกลับมาใช้ใหม่ ย่อมส่งผลดีต่อรูปแบบการเขียนโปรแกรม รวมทั้งลดข้อผิดพลาดที่อาจเกิดขึ้นจากการเขียนโปรแกรมที่มีขนาดใหญ่ ในระบบแบบเดิมจะไม่สนับสนุนความสามารถในลักษณะนี้ แต่ใน o-o programming สามารถตอบสนองคุณสมบัติข้อนี้ได้

12.7.3. ความสามารถในการเข้ากันได้กับส่วนประกอบอื่นๆ (Compatibility)

การเข้ากันได้กับส่วนประกอบอื่นนั้นจะมีความสำคัญมาก เพราะผลิตภัณฑ์ Software จำเป็นต้องทำงานกับส่วนประกอบอื่นๆด้วย และบ่อยครั้งมักจะประสบปัญหาในการเชื่อมต่อกัน เนื่องจากความแตกต่างในรูปแบบที่สนับสนุนโดย OS(Operating systems) ของระบบที่ต่างกัน จุดสำคัญหลักที่จะทำให้การออกแบบ Software ให้สามารถเข้ากับส่วนประกอบอื่นๆ ได้ ขึ้นอยู่กับมาตรฐานการสื่อสารภายในโปรแกรมที่ต้องคำนึงถึงดังนี้

- มาตรฐานของระบบไฟล์ (Standardized file formats) : ทำให้ไฟล์ที่ต่างกันสามารถทำงานได้บน OS เดียวกันได้
- มาตรฐานของโครงสร้างข้อมูล (Standardized data structures) : ข้อมูลที่มีโครงสร้างการทำงานต่างกันจะสามารถนำมาใช้ได้บนเครื่อง PC เดียวกัน ด้วยคุณสมบัติที่สนับสนุนต่อ o-o เช่น โปรแกรมที่มีโครงสร้างข้อมูลสนับสนุนการทำงานแบบ Big andian และlittle andian จะสามารถดำเนินการบนเครื่อง PC เครื่องเดียวกันได้

- มาตรฐานในส่วนติดต่อกับผู้ใช้ (Standardized user interfaces) : การติดต่อกับผู้ใช้ใน o-o programming จะมีส่วนที่เรียกว่า framework ช่วยในการติดต่อสื่อสารระหว่าง user ทำให้ user สามารถเข้าใจและเรียนรู้การทำงานได้ง่ายขึ้นด้วย โดยทั่วๆ ไปแล้วมาตรฐานการจัดการที่เรียกว่า protocol ที่มีส่วนสำคัญต่อการจัดการกับ Software จะเรียกว่า middleware protocol ซึ่งมีอยู่ใน Microsoft's OLE-COM (ActiveX) เป็นต้น

12.7.4. ประสิทธิภาพ (Efficiency)

ประสิทธิภาพคือความสามารถของระบบ Software ที่ขึ้นอยู่กับความเป็นไปได้ของ hardware เช่นความถี่ในการทำงานของ procedure ที่ว่างของ memory ที่มีอยู่ เป็นต้น หลักการที่สำคัญในการสร้าง Software ไม่ได้ขึ้นอยู่กับความเร็วในการทำงานของ Software แต่ขึ้นอยู่กับความถูกต้องในการทำงานที่สมดุลกันกับจุดมุ่งหมายอื่น เช่น ความสามารถในการขยายโปรแกรม (Extendibility) การนำบางส่วนของโปรแกรมกลับมาใช้ใหม่ (Reusability)

12.7.5. ง่ายในการใช้ (Easy to uses)

หมายถึง คนทั่วไปๆ ที่มีพื้นฐานความรู้ต่างกัน ควรจะเรียนรู้การใช้ Software และสามารถประยุกต์ในการแก้ปัญหาเล็กๆน้อยๆ ได้ เช่น การลงโปรแกรม เป็นต้น ซึ่งระบบที่ได้รับการออกแบบมาดี ต้องมีความชัดเจนในการสร้าง (มีเป้าหมายในการสร้าง) ต้องง่ายในการเรียนรู้และการใช้งานของผู้ใช้ (พื้นฐานของ Software ต้องสามารถอ่านได้, เคลื่อนไหว mouse ได้, คลิกที่ปุ่มได้, พิมพ์ได้ เป็นต้น)

12.7.6. ง่ายในการออกแบบ (Easy to designs)

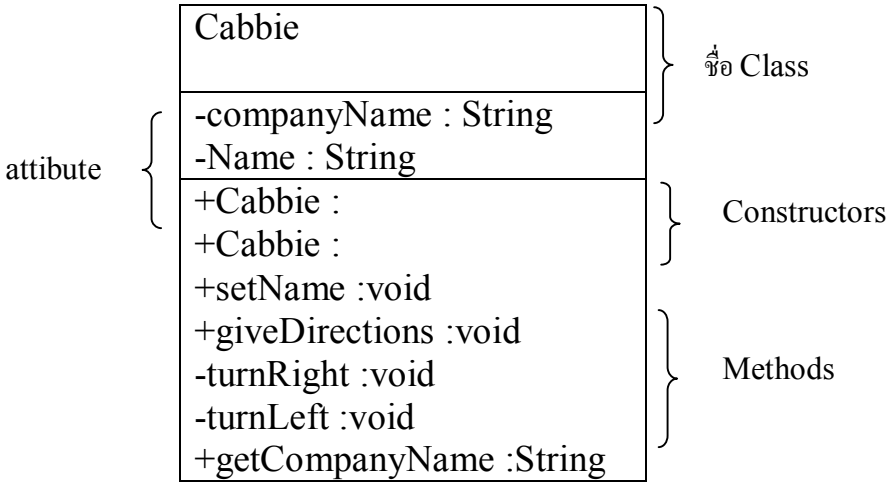
ใน o-o programming นั้น จะแบ่งส่วนของโปรแกรมออกเป็นโปรแกรมย่อยๆที่เรียกว่า Object ซึ่งการออกแบบส่วนประกอบย่อยๆ นั้นง่ายกว่าการสร้าง Software ขนาดใหญ่ขึ้นเป็นชิ้นเดียวโดยไม่มีการแบ่งส่วนการทำงานออกเป็นกลุ่มๆ (Object) นอกจากนี้ในการแก้ไขหรือปรับปรุงสามารถทำได้ง่าย เนื่องจากการทำงานแต่ละส่วนได้แยกออกจากกันอย่างชัดเจนเป็นแต่ละ object

12.8. ความหมาย Unified Modeling Language (UML)

UML หมายถึง ภาษารูปภาพที่ทำการกำหนดลักษณะของ Class การสร้าง class และเป็นเอกสารที่บอกถึงรายละเอียดของระบบโครงสร้างโปรแกรม ถ้าเปรียบไปแล้ว UML คล้ายๆ กับพิมพ์เขียวของระบบ UML จะสามารถแสดงโครงสร้างของระบบ object-oriented ในรายละเอียดลึกๆ ได้ กับพิมพ์เขียวของระบบ UML จะสามารถแสดงโครงสร้างของระบบ object-oriented ในรายละเอียดลึกๆ ได้ดีในรูปแบบของแผนภาพ (diagram) แผนภาพเหล่านี้จะทำให้เกิดความเข้าใจที่ตรงกันระหว่างผู้ออกแบบระบบ และ programmer ทำให้การปรับปรุงแก้ไขโปรแกรมทำได้ง่ายขึ้น ในการศึกษาการออกแบบ o-o ให้เกิดประโยชน์นั้น จำเป็นต้องสามารถอ่าน เขียน และเข้าใจภาษารูปภาพ UML พร้อมทั้งเข้าใจการออกแบบและวิเคราะห์ทาง o-o ควบคู่กันไปด้วย

โครงสร้าง Class diagram

โครงสร้างแผนภาพของ class หรือ class diagram สามารถแบ่งได้เป็น 3 ส่วน คือ class, attribute และ method โครงสร้างของ Class diagram จะแทนที่ได้ด้วยรูปสี่เหลี่ยมผืนผ้า ที่ได้แบ่งออกเป็น 3 ส่วน ดังรูป

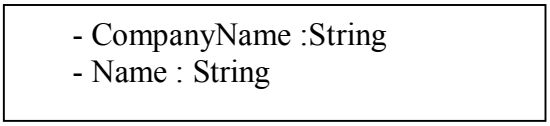


ชื่อ Calss, Attributes และ Methods

ภายใน class diagram จะประกอบด้วยข้อมูล 3 ส่วนคือ ชื่อ class, attributes และ method ที่จะได้กล่าวถึงดังนี้

ชื่อ Class : ควรตั้งให้สอดคล้องกับการทำงาน เช่น class ที่ใช้คำนวณเกรด ควรตั้งชื่อเป็น CalGrade เป็นต้น สัญลักษณ์ของ class และ object จะคล้ายๆ กัน คือ จะต้องขึ้นต้นด้วยตัวอักษรตัวใหญ่ ใช้อักษรตัวหนา และไม่ขีดเส้น

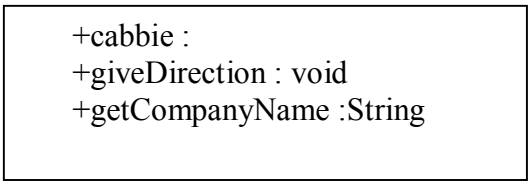
Attributes : โดยปกติแล้วส่วนของ attribute จะมีตัวบ่งบอกชนิด (type) ที่ใช้บอกชนิดของข้อมูล ซึ่งแสดงได้บน class diagram ถ้าพิจารณาจะมี Attributes ดังนี้



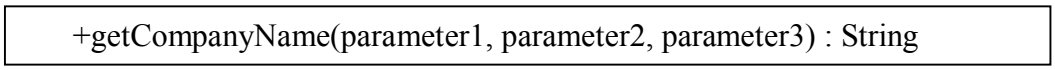
ทั้ง 2 attribute นี้ กำหนดชนิดของข้อมูลเป็น string (ตัวอักขระ) แต่จริงๆ แล้วการกำหนดชนิดของ attribute อาจจะกำหนดเป็นข้อมูลชนิดอื่นได้เช่น integer, float เป็นต้น ทั้งนี้ขึ้นกับความเหมาะสมในการใช้งานของข้อมูลชนิดนั้นๆ คุณสมบัติของ attribute มีดังนี้

1. attribute ที่แสดงบน class diagram สามารถบ่งบอกได้ถึงชนิดของค่ารับ และค่าส่ง (parameter) ที่ใช้โดย method ที่อยู่ภายใน class นั้นๆ
2. ในหนึ่ง attribute สามารถเป็นได้ทั้งค่ารับ และค่าส่ง (parameter) ของ object
3. ถือได้ว่า attribute เป็นตัวเก็บข้อมูลที่เกี่ยวกับ object นั้นๆ

Methods : เรียกได้ว่าเป็นส่วนที่เก็บขั้นตอนการดำเนินงานของระบบในส่วนนั้นๆ และแต่ละ method สามารถแสดงการรับค่า และคืนค่า parameter ซึ่งจะกำหนดชนิด (type) ของการคืนค่าไว้ที่ method โดยใช้สัญลักษณ์ : คั่นเช่น



ในบางกรณีที่ method ประกอบด้วยการรับค่า และส่งค่า parameter จะสามารถเขียนโครงสร้างการทำงานบน UML ได้ด้วยการคั่นระหว่าง parameter ด้วยสัญลักษณ์ , เช่น



มีกรณีพิเศษของ method ที่การตั้งชื่อ method สามารถใช้ชื่อซ้ำกันได้ทั้งๆที่อยู่ใน class เดียวกัน ในลักษณะนี้จะเรียกว่า constructors จะเห็นว่า cabbie method อยู่ 2 method

เครื่องหมาย : จะใช้แสดงลักษณะการมองเห็นจากภายนอกด้วยเครื่องหมาย +, - และไม่แสดงเครื่องหมาย อธิบายได้ดังนี้

- สัญลักษณ์ (+) แสดงความเป็น public หมายความว่า attribute และ method นั้นสามารถมองเห็นได้จาก object อื่น
- สัญลักษณ์ (-) แสดงความเป็น private หมายความว่า attribute และ method นั้นสามารถมองเห็นได้เฉพาะภายใน object เดียวกัน หรือสามารถเรียกใช้ภายใน object เดียวกันเท่านั้น
- ไม่แสดงเครื่องหมาย แสดงความเป็น protect หมายความว่า attribute หรือ method นี้จะอนุญาตให้ข้อมูลใน package เท่านั้นที่จะสามารถเข้าถึง attribute หรือ method นี้ได้ และ package นี้คือกลุ่มของ class ที่ได้รวมไว้ด้วยกันเป็นกลุ่มๆ หนึ่งโดยผู้ออกแบบระบบ หรือผู้พัฒนาระบบ

บทที่ 13
การพัฒนาและติดตั้งระบบ (System Implementation)

หลังจากที่ผ่านขั้นตอนการวิเคราะห์และออกแบบระบบมาแล้ว ขั้นตอนต่อไปคือการพัฒนาและติดตั้งระบบ (System Implementation) ซึ่งในที่นี้จะมีความหมายรวมถึง การเขียนโปรแกรม (Coding) การทดสอบโปรแกรม (Testing) การติดตั้ง(Installation) การจัดทำเอกสาร(Documentation) การฝึกอบรม(Training) และการสนับสนุนการหลังการติดตั้งโปรแกรม(Support) ซึ่งภายในเนื้อหาจะไม่เน้นสอนการเขียนโปรแกรมแต่อย่างใด แต่เป็นการแนะนำแนวทางในการทดสอบโปรแกรมหลังจากที่เขียนโปรแกรมเสร็จสิ้นแล้ว ซึ่งโดยทั่วไปหน้าที่ของนักวิเคราะห์ระบบจะไม่ได้รับผิดชอบในการเขียนโปรแกรมโดยตรง เพียงแต่เป็นผู้ประสานงานในระหว่างที่โปรแกรมเมอร์ทำการเขียนโปรแกรมเท่านั้น แต่ในบางครั้งหรือในบางโอกาสนักวิเคราะห์และออกแบบระบบยังรับหน้าที่ในการเขียนโปรแกรมเองอีกด้วย อย่างไรก็ตามนักวิเคราะห์ระบบจะต้องมีแนวทางในการตรวจสอบหรือทดสอบโปรแกรมที่เขียนเสร็จสิ้นแล้ว ก่อนที่จะนำไปติดตั้งระบบเพื่อใช้งานจริง เพื่อป้องกันข้อผิดพลาดที่อาจจะเกิดขึ้นได้หรือเพื่อเป็นการลดข้อผิดพลาดให้มีน้อยที่สุด จะทำให้ได้ระบบที่สมบูรณ์มากที่สุดต่อไป หลังจากการติดตั้งและได้รับการแก้ไขแล้ว

การพัฒนาระบบจะประสบผลสำเร็จและเป็นที่ยอมรับของผู้ใช้ได้หรือไม่นั้น ไม่ใช่เพียงการได้รับการยอมรับจากผู้ใช้งานการทดสอบใช้ต้นแบบที่สร้างขึ้นมาในระหว่างขั้นตอนการออกแบบระบบเท่านั้น แต่ยังหมายถึงการสร้างเอกสารประกอบการใช้งานระบบ การจัดฝึกอบรมพนักงาน การแก้ไขและปรับปรุงระบบหลังจากการติดตั้ง จึงจะทำให้ทุกส่วนที่เกี่ยวข้องกับระบบใหม่เกิดการทำงานที่สอดคล้องกันและกันได้อย่างลงตัวมากที่สุด

13.1. แนะนำขั้นตอนการพัฒนาและติดตั้งระบบ

ในขั้นตอนการพัฒนาและติดตั้งระบบ มีวัตถุประสงค์เพื่อการปรับเปลี่ยนจากระบบงานเดิมเข้าสู่ระบบงานใหม่ที่ได้ผ่านการวิเคราะห์และออกแบบมาแล้ว โดยเริ่มจากเขียนโปรแกรมของระบบงาน ทดสอบโปรแกรมที่เขียนขึ้น เพื่อให้เป็นโปรแกรมที่น่าเชื่อถือ สามารถทำงานได้อย่างมีข้อผิดพลาดน้อยที่สุด พร้อมทั้งจัดทำเอกสารคู่มือการใช้ระบบ เพื่อเตรียมจัดฝึกอบรมให้กับพนักงาน ซึ่งจะส่งผลให้เป็นระบบที่สามารถทำงานได้อย่างมีประสิทธิภาพ ทั้งนี้ขั้นตอนเหล่านี้โดยส่วนมากแล้ว จะไม่อยู่ในความรับผิดชอบของนักวิเคราะห์ระบบ ยกเว้นกรณีที่องค์กรนั้นแต่งตั้งให้นักวิเคราะห์ระบบรับผิดชอบการเขียนโปรแกรมและติดตั้งระบบรวมอยู่ด้วย อย่างไรก็ตาม เนื่องจากนักวิเคราะห์ระบบเป็นผู้ประสานงานระหว่างผู้เกี่ยวข้องกับการพัฒนาระบบทั้งหมด ดังนั้นหน้าที่ในส่วนนี้จัดว่ายังเป็นส่วนหนึ่งของความรับผิดชอบ เพื่ออำนวยความสะดวกให้กับทีมงานเขียนโปรแกรมและติดตั้งระบบ นอกจากนี้ยังคอยดูแลการทำงานให้เป็นไปตามสิ่งที่ได้ออกแบบและวิเคราะห์มาเพื่อให้ตรงกับความต้องการของผู้ใช้มากที่สุด

ในการทำให้ระบบที่วิเคราะห์และออกแบบมาแล้วนั้นเกิดผลขึ้นคือ การพัฒนาและติดตั้งระบบ (System Implementation) แต่ก่อนการติดตั้งระบบ นักวิเคราะห์ระบบควรมีการวางแผนงานในการติดตั้ง ซึ่งแผนงานนั้นจะต้องครอบคลุมกิจกรรมตั้งแต่การทดสอบโปรแกรม การทดสอบระบบ การติดตั้งระบบ วิธีการติดตั้ง การจัดทำเอกสารคู่มือ การจัดฝึกอบรม และการสนับสนุนหลังการติดตั้ง

สืบเนื่องจากในการพัฒนาระบบนั้นมีหลายแนวทางหรือวิธีการ (Methodology) ที่จะทำให้แต่ละขั้นตอน ของวงจรพัฒนาระบบนั้นลุล่วงได้

15.1.1. การพัฒนาระบบแบบเร่งด่วน (Rapid Application Development : RAD)

ในขั้นตอนการพัฒนาและติดตั้งระบบ มีวัตถุประสงค์เพื่อการปรับเปลี่ยนจากระบบงานเดิมเข้าสู่ระบบงานใหม่ที่ได้ผ่านการวิเคราะห์และออกแบบมาแล้ว โดยเริ่มจากเขียนโปรแกรมของระบบงาน ทดสอบโปรแกรมที่เขียนขึ้น เพื่อให้เป็นโปรแกรมที่น่าเชื่อถือ สามารถทำงานได้อย่างมีข้อผิดพลาดน้อยที่สุด พร้อมทั้งจัดทำเอกสารคู่มือการใช้ระบบ เพื่อเตรียมจัดฝึกอบรมให้กับพนักงาน ซึ่งจะส่งผลให้เป็นระบบที่สามารถทำงานได้อย่างมีประสิทธิภาพ ทั้งนี้ขั้นตอนเหล่านี้โดยส่วนมากแล้ว จะไม่อยู่ในความรับผิดชอบของนักวิเคราะห์ระบบ ยกเว้นกรณีที่องค์กรนั้นแต่งตั้งให้นักวิเคราะห์ระบบรับผิดชอบการเขียนโปรแกรมและติดตั้งระบบรวมอยู่ด้วย อย่างไรก็ตาม เนื่องจากนักวิเคราะห์ระบบเป็นผู้ประสานงานระหว่างผู้เกี่ยวข้องกับการพัฒนาระบบทั้งหมด ดังนั้นหน้าที่ในส่วนนี้จัดว่ายังเป็นส่วนหนึ่งของความรับผิดชอบ เพื่ออำนวยความสะดวกให้กับทีมงานเขียนโปรแกรมและติดตั้งระบบ นอกจากนี้ยังคอยดูแลการทำงานให้เป็นไปตามสิ่งที่ได้ออกแบบและวิเคราะห์มาเพื่อให้ตรงกับความต้องการของผู้ใช้มากที่สุด

ในการทำให้ระบบที่วิเคราะห์และออกแบบมาแล้วนั้นเกิดผลขึ้นก็คือ การพัฒนาและติดตั้งระบบ (System Implementation) แต่ก่อนการติดตั้งระบบ นักวิเคราะห์ระบบควรมีการวางแผนงานในการติดตั้ง ซึ่งแผนงานนั้นจะต้องครอบคลุมกิจกรรมตั้งแต่การทดสอบโปรแกรม การทดสอบระบบ การติดตั้งระบบ วิธีการติดตั้ง การจัดทำเอกสารคู่มือ การจัดฝึกอบรม และการสนับสนุนหลังการติดตั้ง

15.1.1. การพัฒนาระบบแบบเร่งด่วน (Rapid Application Development : RAD)

RAD เป็นวิธีการพัฒนาระบบ (Methodology) วิธีการหนึ่ง ที่รวบรวมเทคนิค (Techniques) เครื่องมือ (Tools) และ เทคโนโลยี เพื่อผสมผสานและประยุกต์ใช้ในการสนับสนุนการพัฒนาระบบให้สามารถลุล่วงโดยใช้เวลาน้อยที่สุด ทั้งนี้ขึ้นอยู่กับความพร้อมขององค์กรในขณะนั้น ไม่ว่าจะเป็นเรื่องค่าใช้จ่าย บุคลากร รวมทั้งความต้องการที่แน่นอนของผู้ใช้ระบบ

จากแนวคิดในวิธีการแบบ RAD ที่มีวัตถุประสงค์เพื่อให้การพัฒนาระบบในวงจรการพัฒนา สามารถดำเนินการเสร็จสิ้นได้อย่างรวดเร็ว จึงทำให้การแบ่งขั้นตอนในวงจรการพัฒนาระบบของแต่ละวิธีการที่สนับสนุนแนวคิด RAD แตกต่างกันออกไปรวมถึงขั้นตอนการพัฒนาและติดตั้งระบบ (System Implementation) จะใช้เวลาในการดำเนินงานน้อยกว่าวิธีการแบบ Waterfall ซึ่งเป็นวิธีการที่ใช้เวลาในการพัฒนาระบบค่อนข้างนาน เนื่องจากแต่ละขั้นตอนจะเริ่มได้ก็ต่อเมื่อขั้นตอนก่อนหน้าเสร็จสิ้นแล้ว อย่างไรก็ตามเนื้อหาในหนังสือ ไม่ได้มีเจตนาชี้นำว่าแนวทางใดที่ดีที่สุด หรือประสบความสำเร็จมากที่สุด เนื่องจากการพัฒนาระบบในความเป็นจริงแล้วจะต้องเลือก ผสมผสานและประยุกต์ใช้เทคนิค เครื่องมือ และเทคโนโลยีที่มีอยู่มากมายเพื่อให้เหมาะสมกับความพร้อมขององค์กรในขณะนั้น ซึ่งเป็นตัวแปรสำคัญในการกำหนดวิธีการ (Methodology) ต่างๆ เพื่อเป็นแนวทางในการพัฒนาระบบ

ต่อไปนี้เป็นวิธีการต่างๆ ในกลุ่มของการพัฒนาระบบด้วยวิธีการแบบ RAD ที่ช่วยลดระยะเวลาในการพัฒนาลง เช่น

1. การพัฒนาด้วยการสร้างตัวต้นแบบ (Prototype Approach to Development) : เป็นกระบวนการเพื่อสร้างการทำงานบางส่วนหรือทั้งหมดของระบบ ที่เหมือนระบบจริงมากที่สุดเท่าที่จะเป็นไปได้ แต่ต้นแบบนั้นจะยังไม่สมบูรณ์จนกว่าจะได้รับการเพิ่มเติมการทำงานทีละส่วนจนครบทุกส่วนของระบบ ขั้นตอนการพัฒนาระบบด้วยตัวต้นแบบ แบ่งออกเป็น 3 ขั้นตอน ดังนี้

ขั้นตอนที่ 1 การวางแผน (Planning)

ขั้นตอนที่ 2 วิเคราะห์(Analysis)

ขั้นตอนที่ 3 ออกแบบและพัฒนา(Design and Implementation)

2. การพัฒนาด้วยวิธีการวนรอบ (Spiral Approach to Development): เป็นวิธีการพัฒนาระบบแบบทำซ้ำ (Iterative Development) โดยในการทำซ้ำแต่ละรอบนั้นอาจจะทำซ้ำในขั้นตอนการวางแผน (Planning) วิเคราะห์(Analysis) ออกแบบ(Design) และการพัฒนา (Development) รวมอยู่ด้วย โดยจะเริ่มต้นที่การวางแผนการทำซ้ำก่อน โดยการทำซ้ำแต่ละรอบจะทำการสร้างตัวต้นแบบเพิ่มเติมจนครบทั้งระบบ ในการสร้างตัวต้นแบบแต่ละรอบจะแบ่งขั้นตอนในการสร้าง

1. วางแผนเพื่อทำซ้ำรอบต่อไป (Plan next Iteration)
2. วิเคราะห์และออกแบบ (Analysis and Design
3. ทดสอบและรวมตัวต้นแบบ

การพัฒนาระบบด้วยวิธีการแบบ Spiral เริ่มต้นที่ขั้นตอนการวางแผน (Plan first Iteration) เป็นขั้นตอนที่จะรวบรวมสารสนเทศให้เพียงพอต่อการสร้างตัวต้นแบบให้ได้มากที่สุด ศึกษาความเป็นไปได้ สำรวจความต้องการของระบบกำหนดทางเลือกในการแก้ปัญหา และกำหนดจำนวนรอบในการทำซ้ำตัวต้นแบบ หลังจากนั้นจึงเริ่มต้นสร้างตัวต้นแบบรวบที่1 ทดสอบและประเมินต้นแบบ และวางแผนเพื่อสร้างต้นแบบรอบที่ 2 เพิ่มเติม ทำเช่นนี้ต่อไปจนกระทั่งครบทุกส่วนงานของระบบ

จากวิธีการพัฒนาระบบแบบ RAD ทั้งในรูปแบบของการใช้ตัวต้นแบบและแบบ Spiral จะเห็นว่าการพัฒนาระบบนั้นจะรวดเร็วกว่าวิธีการพัฒนาระบบแบบ Waterfall เนื่องจากมีการรวมกิจกรรมบางอย่างให้สามารถดำเนินการขนานกันไปได้เข้าไว้ด้วยกัน จึงทำให้ขั้นตอนในการพัฒนาระบบลดน้อยลง อีกทั้งยังเลือกใช้การสร้างต้นแบบเพื่อจำลองการทำงานของระบบจริงทีละส่วนจนกว่าจะครบทุกส่วน และจะกลายเป็นระบบที่สมบูรณ์ที่สุด

15.1.2. การพัฒนาระบบด้วยเครื่องมือ (Tool-based Development)

เป็นการพัฒนาระบบที่ใช้วิธีการเลือกเครื่องมือสนับสนุนการพัฒนาระบบที่ตรงกับความต้องการที่สุด และจะไม่พัฒนาระบบที่มีความต้องการที่ยากแก่การพัฒนาด้วยเครื่องมือ

องค์กรต่างๆ ไม่นิยมเลือกใช้วิธีการพัฒนาระบบวิธีนี้ เนื่องจากองค์กรเองไม่สามารถระบุความต้องการที่ยากต่อการพัฒนาระบบได้ ซึ่งในความเป็นจริงแล้ว ระบบงานที่จะพัฒนาขึ้นมานั้นเพื่อตอบสนองความต้องการในระหว่างการทำงาน ดังนั้นวิธีการนี้ถึงแม้จะพัฒนาระบบได้ง่ายแต่ไม่สามารถตอบสนองความต้องการสำหรับระบบใหม่ได้อย่างเต็มที่

15.1.3. การซื้อซอฟต์แวร์สำเร็จรูปมาใช้งาน (Packaged Software)

กรณีที่องค์กรเลือกที่จะซื้อซอฟต์แวร์สำเร็จรูปมาใช้ในระบบงานใหม่ ก่อนที่จะมีการติดตั้งระบบโปรแกรมเมอร์จะต้องพิจารณาว่าซอฟต์แวร์ที่ซื้อมานั้น จะต้องได้รับการแก้ไขอย่างไรจึงจะมีลักษณะการทำงานที่ตรงกับความต้องการขององค์กร การแก้ไขซอฟต์แวร์ที่ซื้อมาสามารถจำแนกได้ 3 ลักษณะดังต่อไปนี้

- การปรับฟังก์ชันการทำงานของซอฟต์แวร์สำเร็จรูป (Customize Packages)
- การรวมซอฟต์แวร์สำเร็จรูปให้ทำงานร่วมกัน (Integration Packages)
- การยกระดับความสามารถของซอฟต์แวร์สำเร็จรูป (Upgrading Packages)

1. การปรับฟังก์ชันการทำงานของซอฟต์แวร์สำเร็จรูป (Customize Packages) : เป็นการแก้ไขซอฟต์แวร์ที่ซื้อจากภายนอกองค์กร ด้วยการเพิ่มหรือลดฟังก์ชันการทำงานของซอฟต์แวร์ที่ซื้อมา ให้สามารถทำงานได้เหมาะสมกับระบบงานขององค์กรที่กำลังพัฒนา เนื่องจากซอฟต์แวร์สำเร็จรูปบางผลิตภัณฑ์มีฟังก์ชันครอบคลุมการทำงานในด้านตางๆ มากมายจนเกินความต้องการของระบบ ดังนั้นจึงต้องมีการลดฟังก์ชันการทำงานลง หรือเพิ่มฟังก์ชันการทำงานสำหรับซอฟต์แวร์ที่มีการทำงานไม่ครบตามระบบงาน ซึ่งลักษณะการเพิ่มหรือลดฟังก์ชันการทำงานมี 3 ชนิดดังนี้

1. Configuration : เป็นการเลือกตัวเลือกลักษณะการทำงานของซอฟต์แวร์ที่มีการเตรียมตัวเลือกนั้นไว้ให้แล้ว หรือเป็นการเปลี่ยนแปลงค่าพารามิเตอร์ต่างๆ ของซอฟต์แวร์เพื่อให้รูปแบบการทำงานเหมาะกับระบบงานขององค์กร เช่น การเลือกรูปแบบของ Interfaces จากตัวเลือกที่ซอฟต์แวร์มีให้ เป็นต้น

2.Modification : เป็นการเปลี่ยนแปลงโค้ดของซอฟต์แวร์ เพื่อให้มีรูปแบบการทำงานที่เหมาะสมกับระบบงานขององค์กร การแก้ไขซอฟต์แวร์ในลักษณะ Modification นี้จะต้องอาศัยโปรแกรมเมอร์ที่มีความชำนาญเป็นพิเศษ เนื่องจากซอฟต์แวร์สำเร็จรูปบางชนิดไม่อนุญาตให้มีการแก้ไขโค้ด หรือหากอนุญาตให้แก้ไข แต่อาจจะมีเงื่อนไขคือจะไม่รับผิดชอบต่อความเสียหายที่อาจจะเกิดขึ้น และข้อเสียของ Modification คือ เมื่อแก้ไขโค้ดไปแล้วจะทำให้เกิดปัญหาในการ Upgrade ซอฟต์แวร์จากทางผู้ขาย

3.Enhancement : เป็นการเพิ่มโค้ดหรือเพิ่มโมดูลการทำงานของซอฟต์แวร์ เพื่อเพิ่มฟังก์ชันการทำงานให้กับซอฟต์แวร์สำเร็จรูปที่ซื้อมา การ Customize ซอฟต์แวร์สำเร็จรูปในลักษณะนี้ไม่ใช่การแก้ไขโค้ด แต่เป็นการเพิ่มฟังก์ชันการทำงานเข้าไปในซอฟต์แวร์

2.การรวมซอฟต์แวร์สำเร็จรูปเฉพาะด้านให้ทำงานร่วมกัน (Integration Packages) : เป็นการรวมซอฟต์แวร์สำเร็จรูปในงานเฉพาะด้านของแต่ละหน่วยงาน ให้สามารถทำงานร่วมกันได้ด้วยการใช้ข้อมูลร่วมกันแต่เดิมการใช้ซอฟต์แวร์สำเร็จรูปอาจจะนำมาใช้งานในแต่ละแผนกแตกต่างกันไป โดยมีการสร้างและจัดเก็บข้อมูลกันเองแต่ละแผนก การแก้ไขซอฟต์แวร์สำเร็จรูปลักษณะนี้มีแนวคิดคือการสร้างและใช้ขอ้มูลร่วมกันได้จากทุกแผนกไม่ว่าจะเป็นแผนกบุคคล แผนกบัญชี การเงิน ผลิต จัดซื้อ เป็นต้น เช่นข้อมูลพนักงานจะถูกสร้างและจัดเก็บโดยแผนกบุคคล แต่แผนกบัญชีสามารถนำข้อมูลพนักงานไปใช้เพื่อคำนวณเงินเดือนและภาษีได้ โดยไม่ต้องสร้างข้อมูลพนักงานเอง ข้อดีของการแก้ไขซอฟต์แวร์สำเร็จรูปประเภทนี้คือ แต่ละแผนกจะได้ข้อมูลที่มีรูปแบบเป็นมาตรฐานเดียวกันในการใช้งานนั่นเอง

การรวมซอฟต์แวร์สำเร็จรูปเฉพาะด้านให้ทำงานร่วมกันได้ในองค์กร เรียกอีกอย่างหนึ่งว่า “EAI (Enterprise Application Integration)” เป็นกระบวนการที่ใช้เชื่อมโยงแต่ละหน่วยการทำงานขององค์กร ทั้งหน่วยการทำงานทางธุรกิจและสารสนเทศให้สามารถทำงานร่วมกันได้ โดยเลือกใช้วิธีการ 2 วิธี ดังนี้

- ใช้ Middleware ในการเชื่อมโยงการทำงานของแต่ละซอฟต์แวร์

- จัดซื้อซอฟต์แวร์ ERP ที่มี Middleware เพื่อการเชื่อมโยงการทำงานเตรียมไว้ให้เรียบร้อยแล้ว

Middleware คือ ซอฟต์แวร์อรรถประโยชน์ที่มีลักษณะเป็นกึ่งโปรแกรมประยุกต์และกึ่งโปรแกรมระบบหรือที่เรียกว่า Utility Software ที่คอยเชื่อมการทำงานร่วมกันระหว่างเทคโนโลยีที่แตกต่างกันได้ตัวอย่างเช่น ODBC Drive, CORBA และ DCOM เป็นต้น

ERP (Enterprise Resource Planning) คือกระบวนการในการใช้ซอฟต์แวร์เพื่อเชื่อมโยงการทำงานแต่ละหน่วยงานขององค์กร (Middleware) เพื่อปรับปรุง พัฒนาประสิทธิภาพและประสิทธิผลขององค์กร

3.การยกระดับความสามารถของซอฟต์แวร์สำเร็จรูป (Upgrading Packages) : การแก้ไขซอฟต์แวร์สำเร็จรูปเพื่อการติดตั้งระบบประเภทนี้ อาจมีหลายลักษณะ เช่น ยกระดับความสามารถของซอฟต์แวร์ให้ทำงานง่ายขึ้น การแก้ไขข้อผิดพลาดที่พบจากซอฟต์แวร์เวอร์ชัน (Version) เดิม การเพิ่มฟังก์ชันการทำงานของซอฟต์แวร์ เป็นต้น แต่การ Upgrade ซอฟต์แวร์บ่อยครั้งนั้น เป็นสิ่งที่ไม่ดีนัก เนื่องจากถูกมองว่าองค์กรเลือกซอฟต์แวร์ที่ไม่มีประสิทธิภาพมาใช้งานได้ และจากที่ได้กล่าวไว้แล้วว่าหากซอฟต์แวร์สำเร็จรูปได้รับการ Customize หรือการ Integrate มาแล้ว จะทำให้การ Upgrade นั้นยุ่งยากขึ้นจนถึงไม่สามารถ Upgrade ได้เลย ดังนั้น ก่อนที่จะมีการแก้ไขซอฟต์แวร์สำเร็จรูปเพื่อการติดตั้งระบบ ควรมีการวางแผนอย่างรอบคอบ เพื่อป้องกันการเกิดปัญหาลักษณะนี้

จากเนื้อหาในข้อนี้ เป็นการแนะนำให้ผู้อ่านได้รู้จักกับขั้นตอนการพัฒนาและติดตั้งระบบ (System Implementation) ซึ่งถ้าพิจารณาสืบเนื่องมาจากการเลือกวิธีการ (Methodologies) ในการพัฒนาระบบนั้น จะเห็นว่าองค์กรสามารถเลือกพัฒนาระบบได้หลายทางด้วยกัน ดังนั้นก่อนการพัฒนาและติดตั้งระบบควรมีการวางแผนการติดตั้งระบบอย่างรอบคอบ หากเป็นกรณีที่องค์กรเลือกที่จะพัฒนาระบบด้วยทีมงานพัฒนาระบบที่จัดตั้งขึ้นเอง ต้องมีทีมโปรแกรมเมอร์ เพื่อเขียนโปรแกรมของระบบขึ้นมา และเมื่อกระบวนการของการพัฒนาโปรแกรมดำเนินการเสร็จสิ้นแล้ สิ่งสำคัญก่อนการติดตั้งระบบคือ การทดสอบโปรแกรม

15.2. การทดสอบโปรแกรม (Software Testing)

การทดสอบโปรแกรมเป็นขั้นตอนที่สำคัญขั้นตอนหนึ่งหลังจากที่โปรแกรมเมอร์ได้เขียนโปรแกรมเสร็จสิ้นแล้ว จะต้องทดสอบว่าโปรแกรมนั้นให้ผลลัพธ์ที่ถูกต้องหรือไม่ เมื่อพบข้อผิดพลาดจะได้ทำการแก้ไขและป้องกันข้อผิดพลาดที่จะเกิดขึ้นต่อไป ทั้งนี้เพื่อเป็นการทดสอบความสมบูรณ์ของโปรแกรม รวมทั้งความน่าเชื่อถือและความถูกต้องของผลลัพธ์จากโปรแกรมที่พัฒนาขึ้น ซึ่งเทคนิคในการทดสอบโปรแกรมนั้นมีหลากหลายวิธี ในที่นี้จะขอแบ่งออกเป็น 2 กลุ่ม ดังนี้

15.2.1. การทดสอบโดยไม่ใช้เครื่องคอมพิวเตอร์ (Manual Testing)

- การทดสอบแบบตรวจการณ์ (Inspection)
- การทดสอบตามลำดับคำสั่งในโปรแกรม (Desk Checking)

15.2.2. การทดสอบด้วยเครื่องคอมพิวเตอร์ (Automated Testing)

- การทดสอบด้วยการตรวจสอบไวยากรณ์ (Syntax Checking)
- การทดสอบทีละโมดูล (Unit Testing)
- การทดสอบแบบเพิ่มโมดูล (Integration Testing)
- การทดสอบด้วยโมดูลตัวแทน(Stub Testing)
- การทดสอบรวม (System Testing)

15.2.1. การทดสอบโดยไม่ใช้เครื่องคอมพิวเตอร์ (Manual Testing)

เป็นการทดสอบโปรแกรมโดยการตรวจสอบจากโปรแกรมเมอร์เอง ไม่มีการใช้เครื่องคอมพิวเตอร์เพื่อทดสอบโดยอัตโนมัติซึ่งการทดสอบแบบโดยไม่ใช้เครื่องคอมพิวเตอร์ แบ่งได้เป็น 2 ชนิด คือ การทดสอบแบบตรวจการณ์ (Inspection) และการทดสอบตามลำดับคำสั่งในโปรแกรม (Desk Checking)

- การทดสอบแบบตรวจการณ์ (Inspection) : เป็นเทคนิคการทดสอบโปรแกรมโดยโปรแกรมเมอร์ตรวจสอบเองด้วยการเปรียบเทียบโค้ดของโปรแกรมที่เขียนขึ้นกับรายการ Error ที่โปรแกรมเมอร์ทราบแล้วว่าจะต้องเกิดขึ้นจากโปรแกรมภาษาที่ใช้ในการพัฒนาโปรแกรม โดยการตรวจสอบว่าโค้ดที่เขียนนั้นมี Error เกิดขึ้นตามรายการหรือไม่ เช่น Error ที่สามารถเกิดขึ้นได้ในภาษา COBOL ซึ่งจะมีรายการ Error ในคู่มือ ดังนั้นสามารถนำคู่มือนั้นมาทำการเปรียบเทียบกับโปรแกรมที่พัฒนาขึ้นได้ เทคนิคชนิดนี้ใช้ในการป้องกันการเกิดข้อผิดพลาดในรูปแบบเดิมไม่ให้เกิดขึ้นซ้ำอีกครั้ง โดยในบางกรณีโปรแกรมเมอร์สามารถทดสอบโปรแกรมแบบตรวจการณ์ในขณะเขียนโปรแกรมได้ จะทำให้ลดเวลาในการทดสอบโปรแกรมชนิดนี้ได้ภายหลัง แต่การทดสอบชนิดนี้ไม่ได้ทำให้โปรแกรมเมอร์ทราบว่าได้ว่ามีผลลัพธ์ที่ถูกต้องหรือไม่ เนื่องจากไม่ได้ทดสอบการทำงานของโปรแกรม เป็นเพียงการทดสอบความผิดพลาดของโค้ดเท่านั้น
- การทดสอบตามลำดับคำสั่งในโปรแกรม (Desk Checking) : เทคนิคการทดสอบโปรแกรมชนิดนี้ กระทำโดยผู้ที่ได้รับการแต่งตั้งให้เป็นผู้ทดสอบโปรแกรมซึ่งอาจจะเป็นโปรแกรมเมอร์หรือไม่ก็ได้ แต่จะต้องมีความเข้าใจในการทำงานทางตรรกะของโปรแกรม ด้วยการตรวจสอบโค้ดของโปรแกรมตามลำดับคำสั่งในโปรแกรม ว่ามีตรรกะที่ผิดปกติหรือไม่ แต่วิธีการนี้จะทำให้เสียเวลาในการทดสอบโปรแกรมค่อนข้างมากถ้าระบบงานที่มีความซับซ้อนสูง

15.2.2. การทดสอบด้วยเครื่องคอมพิวเตอร์ (Automated Testing)

เป็นการทดสอบโปรแกรมด้วยเครื่องคอมพิวเตอร์ ซึ่งจะทำให้เสียเวลาในการทดสอบ แบ่งได้เป็น 4 ชนิด ได้แก่ การทดสอบด้วยการตรวจสอบไวยากรณ์ (Syntax Checking) การทดสอบทีละโมดูล (Unit Testing) การทดสอบแบบเพิ่มโมดูล (Integration Testing) การทดสอบด้วยโมดูลตัวแทน (Stub Testing) และการทดสอบรวม (System Testing)

1. การทดสอบด้วยการตรวจสอบไวยากรณ์ (Syntax Checking) : เป็นการทดสอบโปรแกรม ด้วยการตรวจสอบไวยากรณ์ (Syntax) ที่เขียนขึ้น โดยปกติแล้วจะได้รับการตรวจสอบด้วย Compiler ซึ่งจะใช้เวลาไม่นานสามารถทราบผลได้ แต่วิธีการนี้ไม่ทำให้ทราบได้ว่าผลลัพธ์จากการทำงานของโปรแกรมนั้นจะถูกต้องหรือไม่ เนื่องจากเป็นทดสอบเพียงไวยากรณ์ของโปรแกรมเท่านั้น
2. การทดสอบทีละโมดูล (Unit Testing) : Unit Testing หรือบางครั้งเรียกอีกอย่างว่า Module Testing เป็นการทดสอบโปรแกรมทีละโมดูล เพื่อหาข้อผิดพลาดที่จะเกิดขึ้นภายในการทำงานของแต่ละโมดูล
3. การทดสอบแบบเพิ่มโมดูล (Integration Testing) : เป็นการทดสอบโปรแกรมโดยการเพิ่มจำนวนโมดูลเพื่อการทดสอบ ซึ่งวิธีการนี้จะอาศัย Structure Chart ที่มีอยู่แล้ว ช่วยในการทดสอบโปรแกรม ซึ่งวิธีการในการทดสอบแบบ Integration นี้แบ่งออกเป็น 2 ลักษณะได้แก่ การทดสอบแบบเพิ่มโมดูลจากบนลงล่าง (Top-down Approach) และการทดสอบแบบเพิ่มโมดูลจากล่างขึ้นบน (Bottom-up Approach)

- การทดสอบแบบเพิ่มโมดูลจากบนลงล่าง (Top- down Approach) : เป็นการทดสอบโปรแกรม โดยทดสอบโมดูลจากบนลงล่าง

- การทดสอบแบบเพิ่มโมดูลจากล่างขึ้นบน (Bottom-up Approach) : เป็นการทดสอบโปรแกรม โดยทดสอบโมดูลจากล่างขึ้นบน

4. การทดสอบด้วยโมดูลตัวแทน (Stub Testing) : โดยทั่วไปแล้ว โมดูลที่อยู่ในระดับบนจะเรียกใช้ข้อมูลจากโมดูลระดับล่าง แต่การทดสอบโปรแกรมแบบเพิ่มโมดูล (Integration) นั้นจะทดสอบโมดูลทีละระดับ ฉะนั้น Stub Testing คือ กลุ่มคำสั่งสั้นๆ ที่เขียนขึ้นมาเพื่อเป็นโมดูลตัวแทนในการทดสอบโปรแกรม
5. การทดสอบรวม (System Testing) : เป็นการทดสอบโปรแกรมที่มีวิธีการคล้ายกับการทดสอบแบบ Integration แตกต่างกันตรงที่ Integration Testing จะทดสอบโดยใช้โมดูลเพิ่มไปเรื่อยๆ จนกระทั่งครบทุกโมดูลของโปรแกรม แต่ System Testing จะทดสอบจากโปรแกรมเพิ่มไปเรื่อยๆ จนกระทั่งครบทุกโปรแกรมของระบบงาน ว่าโปรแกรมทุกโปรแกรมเมื่อทำงานร่วมกันแล้วจะให้ผลลัพธ์ที่ถูกต้องหรือไม่ นอกจากนี้แล้ว System Testing ยังเป็นการทดสอบระบบงานว่า สามารถทำงานให้ผลลัพธ์ที่มีประสิทธิภาพเป็นที่ยอมรับหรือไม่ และเพื่อทำให้มั่นใจได้ว่าระบบงานนั้นสามารถตอบสนองความต้องการของผู้ใช้ได้อย่างตรงจุดมากที่สุด

15.2.3. กลยุทธ์ในการทดสอบระบบ

กลยุทธ์ที่ใช้ในการทดสอบประสิทธิภาพในการทำงานของระบบ สามารถจำแนกเป็น การทดสอบการทำงานสูงสุด (Peak Load Testing) การทดสอบประสิทธิภาพของเวลา (Performance Testing) การทดสอบการกู้ระบบ (Recovery Testing) การทดสอบการเก็บข้อมูล (Storage Testing) การทดสอบกระบวนการ (Procedure Testing) และการทดสอบผู้ใช้ (User Testing)

1. การทดสอบการทำงานสูงสุด(Peak Load Testing): เป็นการทดสอบประสิทธิภาพในการประมวลผลของระบบเมื่อมีการทำรายการมากที่สุด ณ ช่วงเวลาใดเวลาหนึ่ง เพื่อทดสอบว่าระบบจะสามารถรองรับการทำรายการคำสั่งมากที่สุดได้เพียงใด และนานเท่าใดเมื่อต้องประมวลผลจำนวนรายการคำสั่งที่มากที่สุดดังกล่าวในช่วงเวลาหนึ่ง

   เช่น ระบบลงทะเบียนนักศึกษาผ่านทางอินเตอร์เน็ต สามารถรองรับการทำรายการของนักศึกษาได้สูงสุด 1000 รายการต่อวัน ในการทดสอบสมมติให้มีการทำรายการ 1200 รายการต่อวัน ในช่วงภาคการศึกษาฤดูร้อน การทดสอบลักษณะนี้จะทำให้ทราบได้ว่าระบบสามารถรองรับการทำรายการมากกว่าขีดสูงสุดที่กำหนดได้หรือไม่ และได้นานเพียงใด มีปัญหาอะไรเกิดขึ้นบ้าง เป็นการปรับปรุงระบบหากมีการเพิ่มจำนวนระเบียนข้อมูลในฐานข้อมูลในอนาคต
2. การทดสอบประสิทธิภาพของเวลา (Performance Testing) : เป็นการทดสอบระบบ เพื่อพิจารณาถึงช่วงเวลาที่ใช้ในการประมวลผลรายการ ว่าใช้ระยะเวลานานเพียงใดในการทำรายการไม่ว่าจะเป็นการประมวลผลแบบกลุ่ม (Batch Processing) หรือการประมวลผลแบบออนไลน์ (On-line Processing) รวมทั้งทดสอบช่วงเวลาที่ใช้ในการเข้าถึงข้อมูลแบบลำดับ (Sequential Access) และแบบสุ่ม (Random Access) ด้วย
3. การทดสอบการกู้ระบบ (Recovery Testing) : เป็นการทดสอบความสามารถในการกู้ระบบกรณีที่ระบบล้ม ความสามารถในการกู้ระบบนี้รวมทั้งการกู้ข้อมูลด้วย
4. การทดสอบการเก็บข้อมูล (Storage Testing) : เป็นการทดสอบความสามารถของระบบในการเก็บข้อมูล ว่าสามารถเก็บข้อมูลได้สูงสุดเป็นจำนวนเท่าใด เพื่อจะได้เตรียมการรองรับจำนวนข้อมูลที่อาจจะเพิ่มมากขึ้นในอนาคต

5. การทดสอบกระบวนการ (Procedure Testing) : เป็นการทดสอบการจัดทำเอกสารคู่มือการดำเนินงานของระบบและคู่มือการใช้งานสำหรับผู้ใช้ ว่าสามารถสร้างความเข้าใจให้กับผู้ใช้ได้มากน้อยเพียงใด และเมื่อเกิดปัญหาในเบื้องต้นขึ้น ผู้ใช้สามารถอ่านคู่มือเพื่อแก้ไขปัญหานั้นได้หรือไม่
6. การทดสอบผู้ใช้ (User Testing) : หรือบางครั้งเรียกว่า Human Factors Testing เป็นการทดสอบการใช้งานจริงของระบบเพื่อต้องการทราบว่าผู้ใช้จะทำอย่างไรเมื่อพบปัญหาที่เกิดขึ้น

นอกจากการทดสอบดังกล่าวข้างต้นทั้งหมดแล้ว ยังจะต้องมีการทดสอบการยอมรับของผู้ใช้ที่มีต่อระบบใหม่ จึงจะถือว่าระบบนั้นได้ผ่านการทดสอบอย่างครบถ้วนแล้ว

15.2.4. การทดสอบการยอมรับของระบบโดยผู้ใช้

หลังจากที่ได้ทดสอบความสมบูรณ์และความถูกต้องของโปรแกรมแล้ว ยังจะต้องทำการทดสอบการยอมรับของระบบจากผู้ใช้ที่มีต่อระบบอีกด้วย ซึ่งเป็นการทดสอบที่สำคัญเทียบเท่ากับการทดสอบโปรแกรม เนื่องจากการพัฒนาระบบนั้นเพื่อตอบสนองความต้องการในการดำเนินงานของผู้ใช้ระบบ ดังนั้นระบบงานใหม่ จะสามารถติดตั้งได้ต้องผ่านการยอมรับจากผู้ใช้ โดยวิธีการทดสอบการยอมรับของระบบนั้นสามารถแบ่งออกเป็น 2 ประเภท คือ Alpha Testing และ Beta Testing

1. Alpha Testing : คือ การทดสอบความสมบูรณ์ของระบบโดยผู้ใช้ และใช้ข้อมูลสมมติในการทดสอบ ในการทดสอบประเภทนี้ จะสมมติให้ระบบอยู่ในสถานการณ์ที่อาจจะเกิดขึ้นได้จาก Alpha Testing นี้จะทำให้ทราบได้ว่าระบบมีข้อผิดพลาดอะไรเกิดขึ้นบ้าง โดยการทดสอบจะมี 4 ประเภทคือ

   Recovery Testing : เป็นการทดสอบการกู้ระบบ หากเกิดกรณีที่ระบบล้ม เพื่อทดสอบว่าระบบมีประสิทธิภาพในการกู้ข้อมูลและดำเนินระบบต่อไปได้น่าพอใจหรือไม่

   Security Testing : เป็นการทดสอบความปลอดภัยของระบบ ว่ามีเครื่องมือรักษาความปลอดภัยที่มีประสิทธิภาพเป็นที่น่าพึงพอใจหรือไม่จากสถานการณ์การลักลอบเรียกใช้ข้อมูล

   Stress Testing : เป็นการทดสอบประสิทธิภาพการทำงานของระบบภายใต้ความกดดัน เช่นจะเกิดอะไรขึ้นเมื่อผู้ใช้ป้อนข้อมูลไม่ครบทุกField หรือจะเกิดอะไรขึ้นเมื่อมีการเข้าถึงข้อมูลในเวลาเดียวกันจากผู้ใช้หลายคน เป็นต้น

   Performance Testing : เป็นการทดสอบประสิทธิภาพการทำงานของระบบภายใต้สภาพแวดล้อมของคอมพิวเตอร์ เช่นภายใต้ระบบปฏิบัติการคอมพิวเตอร์ที่แตกต่างกัน ภายใต้ระบบเครือข่ายคอมพิวเตอร์ที่แตกต่างกัน ว่าระบบมี Response time มากน้อยเพียงใด

ทั้งนี้ข้อมูลที่ใช้ในการทดสอบแบบ Alpha Testing เป็นข้อมูลที่สมมติขึ้น แต่จะมีอีกประเภทหนึ่งที่ใช้ข้อมูลจริงจากการดำเนินงานในชีวิตประจำวันในการทดสอบ นั่นคือ Beta Testing

2. Beta Testing : คือ การทดสอบความสมบูรณ์ของระบบโดยผู้ใช้ และใช้ข้อมูลจริงในการทดสอบและภายใต้สถานการณ์ที่เกิดขึ้นจริง การทดสอบประเภทนี้ถือว่าเป็นการซ้อมติดตั้งระบบเพื่อใช้งานจริง เนื่องจากเป็นการทดสอบระบบอย่างสมจริงไม่ว่าจะเป็นสถานการณ์ ข้อมูล ขั้นตอนการดำเนินงาน เอกสารคู่มือ การฝึกอบรม การสนับสนุนการทำงาน รวมทั้งยังเป็นการแก้ปัญหาที่พบจากการทดสอบแบบ Alpha Testing ด้วย

15.3. การติดตั้งระบบ (Installation)

เมื่อทดสอบโปรแกรมและระบบ จนผู้ใช้ยอมรับแล้ว ระบบงานที่ได้รับการพัฒนาพร้อมที่จะนำมาใช้งานจริง ด้วยการติดตั้ง (Installation) การติดตั้งระบบ คือ การเปลี่ยนการทำงานจากระบบงานเดิมไปเป็นระบบงานใหม่ แต่การ

เปลี่ยนแปลงไปสู่สิ่งใหม่ย่อมมีผลกระทบต่อผู้ใช้งานบางกลุ่ม ที่ยังคงมีความคุ้นเคยกับวิธีการดำเนินงานแบบเก่า รวมทั้งข้อจำกัดในเรื่องของความพร้อมในการเปลี่ยนแปลง ดังนั้นจึงควรเลือกแนวทางที่เหมาะสมในการติดตั้งด้วย ซึ่งแบ่งออกเป็น 4 แนวทางดังนี้

1. การติดตั้งระบบทันทีทันใด (Direct Installation)
2. การติดตั้งแบบขนาน (Parallel Installation)
3. การติดตั้งแบบนำร่อง (Single Location Installation/Pilot Installation)
4. การติดตั้งแบบยอมติดตั้งเป็นระยะ (Phased Installation)

15.3.1. การติดตั้งระบบทันทีทันใด (Direct Installation)

เป็นวิธีการติดตั้งที่มีการใช้ระบบงานใหม่ทันที และระบบงานเก่ายกเลิกการใช้งานทันทีเช่นเดียวกัน วิธีการแบบนี้องค์กรเสียค่าใช้จ่ายน้อยแต่มีความเสี่ยงสูง เนื่องจากในการใช้งานจริงนั้นข้อมูลที่นำเข้าสู่ระบบมีความเป็นไปได้ในการเกิดข้อผิดพลาดขึ้นได้มากกว่าข้อมูลที่สมมติขึ้นเพื่อการทดสอบ และหากเกิดกรณีเช่นนี้จะส่งผลกระทบต่อผู้ใช้ระบบ จนทำให้ระบบหยุดชะงักได้ วิธีการแก้ปัญหาอย่างหนึ่งคือนำระบบเก่าเข้ามาใช้งานอีกครั้ง แต่จะเสียเวลามาก เนื่องจากต้องปรับปรุงข้อมูลในระบบเก่าทั้งหมด แต่การติดตั้งวิธีนี้ยังมีข้อดีคือ สามารถเลื่อนกำหนดการใช้งานระบบใหม่ออกไปได้หากองค์กรหรือผู้ใช้งานยังไม่มีความพร้อมสำหรับระบบใหม่

อย่างไรก็ตามระบบที่พร้อมจะติดตั้งเพื่อใช้งาน จะต้องผ่านการทดสอบอย่างหนักในทุกสถานการณ์ และผ่านการแก้ไขข้อผิดพลาดที่เกิดขึ้นจากการทดสอบมาแล้วอย่างครบถ้วน

15.3.2. การติดตั้งแบบขนาน (Parallel Installation)

เป็นวิธีการที่มีการใช้ระบบงานใหม่ไปพร้อมๆ กับการใช้ระบบงานเก่า จนกว่าผู้ใช้และผู้บริหารจะพอใจระบบใหม่และตัดสินใจที่จะหยุดใช้ระบบเก่า การติดตั้งแบบขนาน เป็นวิธีการติดตั้งที่ต้องใช้ค่าใช้จ่ายค่อนข้างสูง เนื่องจากการดำเนินงานสองระบบในเวลาเดียวกันหมายถึงการเพิ่มพนักงานบางตำแหน่งเพื่อดูแลระบบใหม่ ทั้งยังต้องคอยดูแล บำรุงรักษาระบบให้มีความสมบูรณ์และมีประสิทธิภาพมากที่สุด ในบางครั้งอาจทำให้ผู้ใช้ระบบเกิดความสับสนได้ แต่วิธีการนี้จะทำให้สามารถเปรียบเทียบผลของการดำเนินงานระหว่างระบบใหม่กับระบบเก่าได้

15.3.3. การติดตั้งแบบนำร่อง (Single Location Installation/Pilot Installation)

เป็นวิธีการที่มีการใช้ระบบงานใหม่เพียงหน่วยเดียวขององค์กรก่อนเพื่อเป็นการนำร่อง แล้วจึงค่อยปรับเปลี่ยนทั้งหมดเมื่อเห็นว่าระบบใหม่นั้นลงตัวแล้ว เช่น การใช้ระบบงานใหม่เฉพาะแผนกจัดซื้อแผนกเดียวกัน การใช้ระบบงานใหม่เฉพาะสาขาเดียวก่อน หรือเฉพาะบางภูมิภาคก่อน เป็นต้น

ข้อดีของวิธีการนี้คือ เสียค่าใช้จ่ายน้อยกว่าสองวิธีแรก และเมื่อเกิดข้อผิดพลาดหรือความเสียหายเนื่องจากระบบงานใหม่ ความเสียหายนั้นจะจำกัดอยู่เพียงแค่สถานที่ขององค์กรที่จัดให้มีการติดตั้งระบบใหม่แบบนำร่องเท่านั้น หรือจำกัดอยู่เพียงสาขาเดียว ไม่ทำให้การดำเนินธุรกิจของทั้งองค์กรเสียหายไปด้วย ทั้งยังสามารถติดตามผล และดูแลระบบใหม่ได้อย่างเต็มที่ เพื่อปรับปรุงและแก้ไขให้เป็นระบบที่สมบูรณ์ก่อนที่จะนำไปปรับเปลี่ยนกับส่วนที่เหลือขององค์กร

อย่างไรก็ตาม ทีมนักพัฒนาหรือนักวิเคราะห์ระบบ ยังคงต้องมีการดูแลรักษาระบบใหม่ไปพร้อมๆ กับที่ให้ความสนใจกับระบบเก่าที่ยังคงใช้งานอยู่ส่วนใหญ่ซึ่งคล้ายกับวิธีการอื่น หากแต่ความสับสนที่เกิดขึ้นนั้นน้อยกว่า

15.3.4. การติดตั้งแบบทยอยติดตั้งเป็นระยะ (Phased Installation)

เรียกอีกอย่างหนึ่งว่า “Staged Installation” เป็นวิธีการที่มีการใช้ระบบงานใหม่เพียงบางส่วนก่อนระยะหนึ่งควบคู่ไปกับระบบงานเก่า แล้วจึงค่อยๆ ทยอยใช้ระบบงานใหม่เพิ่มขึ้นทีละส่วนจนกระทั่งครบทุกส่วนของระบบงานใหม่อย่างเต็มรูปแบบในที่สุด วิธีการนี้มีลักษณะคล้ายกับแบบนำร่อง คือเริ่มจากจุดเดียวก่อน แตกต่างกันตรงที่วิธีแบบทยอยติดตั้งเป็นระยะ ไม่คำนึงถึงสถานที่ แต่คำนึงถึงระบบงานย่อยโดยการติดตั้งทีละระบบ ซึ่งอาจจะกระจายไปตามสาขาต่างๆ ที่มีการใช้ระบบงานย่อยนั้นเมื่อระบบย่อยที่ใช้เริ่มแรกสมบูรณ์แล้ว จึงเริ่มนำระบบย่อยต่อไปมาใช้งาน เป็นเช่นนี้ไปเรื่อยๆ จนครบทั้งระบบในที่สุด

ข้อดี : ของวิธีการนี้คือ สามารถจำกัดความเสี่ยงที่เกิดขึ้นได้ เนื่องจากการปรับเปลี่ยนเป็นช่วงระยะเวลาและบางสถานที่ความเสี่ยงหรือข้อผิดพลาดจึงเกิดขึ้นในช่วงที่มีการเปลี่ยนแปลงและสถานที่ที่ใช้ระบบงานใหม่เท่านั้น

ข้อเสีย : คือ ความไม่สอดคล้องของการดำเนินงานในระบบใหม่และระบบเก่า เนื่องจากมีการนำระบบงานย่อยของระบบใหม่ เข้ามาใช้ร่วมกับระบบเก่า เช่น หากมีการนำระบบการสั่งซื้อซึ่งเป็นระบบใหม่ เข้ามาใช้งานก่อนอาจจะเกิดปัญหาในการติดต่อระหว่างฐานข้อมูลของระบบใหม่กับฐานข้อมูลของระบบเก่าได้

-การวางแผนการติดตั้งระบบ (Planning Installation) : การติดตั้งระบบงานใหม่หมายถึง การเปลี่ยนแปลงที่จะเกิดขึ้นกับการดำเนินงานในชีวิตประจำวันของผู้ใช้ระบบ ซึ่งมีความคุ้นเคยกับระบบเก่ามานาน การเปลี่ยนแปลงนั้นยังรวมถึงวิธีการดำเนินงาน การเปลี่ยนแปลงรูปแบบเอกสาร รูปแบบการนำเข้าข้อมูล การเปลี่ยนแปลงโปรแกรม หรือแม้กระทั่งการเปลี่ยนแปลงรูปแบบของข้อมูลบางอย่างที่จะต้องนำเข้าสู่ระบบซึ่งนับได้ว่าเป็นขั้นตอนที่สำคัญขั้นตอนหนึ่ง ดังนั้น ก่อนการติดตั้งระบบ นักวิเคราะห์จึงจำเป็นต้องมีการวางแผนเพื่อเลือกวิธีการติดตั้งอย่างรอบคอบและระมัดระวังมากที่สุด เพื่อป้องกันไม่ให้เกิดการต่อต้านระบบใหม่จากผู้ใช้ ซึ่งอาจจะเกิดขึ้นได้อย่างฉับพลันในระหว่างใช้ระบบใหม่ ทำให้เกิดความเสียหายแก่องค์กรในที่สุด โดยการเลือกวิธีการในการติดตั้งให้เหมาะสมกับองค์กร ซึ่งอาจจะมีการผสมผสานวิธีการติดตั้งได้มากกว่า 2 วิธี ขึ้นอยู่กับสถานการณ์และความพร้อมขององค์กร ผนวกกับปัจจัยด้านบุคลากร ความคล่องตัว ความกล้าที่จะเสี่ยงกับความสับสนที่จะเกิดขึ้นกับลูกค้า นักวิเคราะห์ระบบควรมีการตั้งคำถามเพื่อการตัดสินใจเลือกใช้วิธีการติดตั้งระบบใหม่ และเมื่อเลือกวิธีการได้แล้ว ควรจัดทำตารางการติดตั้งระบบใหม่ ประชาสัมพันธ์ให้ทุกคนในองค์กรรับทราบถึงตารางการติดตั้งระบบนั้น เพื่อการเตรียมพร้อมในการเตรียมงานของทุกฝ่าย รวมทั้งวางแผนเพื่อจัดเตรียมทีมงานสำหรับดูแลระบบใหม่อย่างชัดเจนเพื่อรองรับปัญหาที่จะเกิดขึ้นได้อย่างทันท่วงที สิ่งที่สำคัญของการวางแผนในการติดตั้งระบบใหม่สำหรับการดำเนินธุรกิจ คือ ไม่ควรวางแผนให้มีการติดตั้งระบบใหม่ในช่วงเวลาที่ทำเงินให้ธุรกิจได้มากที่สุดของปี เนื่องจากหากมีการติดตั้งระบบใหม่ในช่วงนั้น ซึ่งแน่นอนว่า ระบบใหม่นั้นย่อมจะเกิดปัญหาขึ้นในระหว่างการดำเนินงาน และส่งผลให้สร้างความไม่พอใจแก่ลูกค้าในที่สุด ดังนั้นไม่ว่าจะเป็นวิธีการติดตั้งระบบแบบใดก็ตาม นักวิเคราะห์ระบบควรมีความเข้าใจเกี่ยวกับการดำเนินธุรกิจเป็นอย่างดี เพื่อให้การติดตั้งระบบนั้นส่งผลกระทบต่อผลประโยชน์ในการดำเนินธุรกิจขององค์กร

15.4. การจัดทำเอกสาร (Documentation)

ในระหว่างการพัฒนาระบบงานใหม่นั้น จะเกิดเอกสารแสดงรายละเอียดของระบบงานมากมาย ไม่ว่าจะเป็นเอกสารที่เกิดจากขั้นตอนการวิเคราะห์ระบบ หรือออกแบบระบบก็ตาม เมื่อมาถึงขั้นตอนการติดตั้งระบบแล้ว ชุดเอกสารเหล่านั้นจะต้องทำการปรับปรุงให้เป็นเอกสารที่แสดงรายละเอียดของระบบใหม่ที่ถูกต้องมากที่สุด นอกจากนี้ในขั้นตอนการติดตั้งนักวิเคราะห์ระบบยังจะต้องจัดทำเอกสารคู่มือสำหรับผู้ใช้ เพื่อแสดงรายละเอียด

ขั้นตอนการทำงาน ภาพรวมของระบบใหม่ทั้งหมด เพื่อใช้อ้างอิงในระหว่างการใช้งานระบบใหม่ สามารถจำแนกการจัดทำเอกสารออกเป็น 2 ประเภท ดังนี้

- การจัดทำเอกสารของระบบ (System Documentation)
- การจัดทำเอกสารของผู้ใช้ (User Documentation)

15.4.1. การจัดทำเอกสารของระบบ (System Documentation)

เอกสารของระบบ คือ เอกสารที่แสดงขั้นตอนการทำงานภายในของระบบ และรายละเอียดข้อมูลเฉพาะของการออกแบบระบบ เช่น แบบจำลองชนิดต่างๆ ที่ใช้เป็นเครื่องมือในระหว่างการวิเคราะห์ระบบ (Diagrams) พจนานุกรมข้อมูล (Data Dictionary) แบบร่างสำหรับการออกแบบแบบฟอร์ม รายงาน หน้าจอ และต้นแบบ เป็นต้น ซึ่งสามารถจำแนกเอกสารของระบบนี้ออกเป็น 2 กลุ่ม ได้แก่ เอกสารภายใน (Internal Documentation) และเอกสารภายนอก (External Documentation)

เอกสารภายใน (Internal Documentation) : คือเอกสารของระบบที่แสดงรายละเอียดส่วนของโค้ดโปรแกรมที่เขียนขึ้น พร้อมทั้งคำอธิบายโปรแกรม ผลการรันโปรแกรมและอื่นๆที่เกี่ยวข้องกับโปรแกรมระบบ

เอกสารภายนอก (External Documentation) : คือเอกสารที่เป็นแผนภาพ และแบบจำลองชนิดต่างๆของระบบ เช่นแผนภาพกระแสข้อมูล (Data Flow Diagram: DFD) แผนภาพแสดงความสัมพันธ์ของข้อมูล (Entity Relationship Diagram :E-R diagram) แผนผังโครงสร้าง (Structure Chart) เป็นต้น

เอกสารของระบบ จะช่วยให้การบำรุงรักษาระบบทำได้สะดวกขึ้น เนื่องจากโดยขอบเขตของเอกสารแล้วสามารถเป็นข้อมูลสำหรับอ้างอิงการทำงานของระบบเมื่อเกิดปัญหาได้ และหากเกิดกรณีที่องค์กรเปลี่ยนทีมงานพัฒนาระบบ เอกสารเหล่านั้นจะช่วยให้ทีมพัฒนาทีมใหม่ ทำความเข้าใจกับโปรแกรม ขั้นตอนการทำงานและอื่นๆที่เกิดขึ้นในระบบได้รวดเร็วยิ่งขึ้น

15.4.2. การจัดทำเอกสารของผู้ใช้ (User Documentation)

เอกสารของผู้ใช้ คือ เอกสารที่จัดทำขึ้นเพื่อแสดงขั้นตอนการใช้งานระบบ และวิธีการใช้งานโปรแกรม เอกสารสำหรับผู้ใช้ยังรวมถึงคู่มือการใช้งาน (User's Guide) คู่มือสำหรับบริหารระบบ (System Administration's Guide) ซึ่งคู่มือเหล่านี้อาจจะอยู่ในรูปของเอกสารเป็นเล่ม และแสดงไว้ในส่วนช่วยเหลือ (Help) ภายในโปรแกรมก็ได้ แต่การจัดรูปแบบของการอธิบายรายละเอียดการใช้งานโปรแกรม จะต้องมีรูปแบบที่น่าสนใจ อ่านเข้าใจง่าย ค้นหาง่าย แสดงหัวข้อชัดเจน รวมทั้งแสดงถึงปัญหาที่อาจจะเกิดขึ้นเนื่องจากการใช้งานและวิธีแก้ปัญหา โดยการจัดรูปแบบจะมีหลักเกณฑ์คล้ายกับการออกแบบส่วนช่วยเหลือของโปรแกรมในขั้นตอนการออกแบบระบบนั่นเอง

15.5. การฝึกอบรมพนักงาน (Training)

ปัจจุบันนี้องค์กรต่างๆ ได้เล็งเห็นถึงความสำคัญของการฝึกอบรมพนักงานของตนมากขึ้น สังเกตได้จากการขยายตัวของกิจการสถาบันอบรมหลักสูตรคอมพิวเตอร์ ทั้งนี้อาจจะเนื่องมาจากความคิดเห็นที่ว่า ระบบที่พัฒนามาอย่างสมบูรณ์แล้วจะยังไม่สามารถก่อให้เกิดประสิทธิภาพให้กับองค์กรได้อย่างเต็มที่ หากขาดความเข้าใจและทักษะในการใช้งานระบบเป็นอย่างดีของผู้ใช้ระบบ

กรณีที่ระบบงานนั้นเป็นระบบใหม่ทั้งหมด การอาศัยความเข้าใจจากคู่มือการใช้งานโปรแกรมเพียงอย่างเดียว อาจจะยังไม่เพียงพอ แต่ยังจะต้องสร้างทักษะเริ่มต้นในการดำเนินงานให้แก่ผู้ใช้งานอีกด้วย ดังนั้น

องค์กรและนักวิเคราะห์ระบบจึงต้องร่วมมือกัน จัดหลักสูตรฝึกอบรมการใช้งานโปรแกรม ซึ่งมีวิธีการและเทคนิคแตกต่างกันไปแต่ละองค์กร ขึ้นอยู่กับสภาพแวดล้อมและสถานการณ์ในขณะนั้นองค์กร และปัจจัยสำคัญที่ส่งผลกระทบต่อการจัดหลักสูตรฝึกอบรมให้แก่พนักงานนั่นก็คือ “ค่าใช้จ่าย” ดังนั้นรูปแบบในการฝึกอบรมพนักงานจึงมีแตกต่างกันไป ซึ่งอาจจะมีการจัดฝึกอบรมในลักษณะต่างๆ ดังนี้

1. จัดเป็นหลักสูตรฝึกอบรมเป็นกลุ่ม
2. ฝึกอบรมด้วยคอมพิวเตอร์ช่วยสอนเช่น CAI(Computer-aided Instruction), CBI(Computer-base Training) หรือ WBT(Web-Base Training)
3. ฝึกอบรมด้วย CAI พร้อมกับมีผู้เชี่ยวชาญคอยแนะนำ
4. ผู้ใช้ฝึกอบรมด้วยตัวเองจากส่วนช่วยเหลือของโปรแกรม
5. ฝึกอบรมโดยผู้จำหน่ายโปรแกรมของระบบ หรือกลุ่มผู้รับพัฒนาระบบภายนอกองค์กร
6. ฝึกอบรมผ่านทางอินเตอร์เน็ตกรณีที่องค์กรมีสาขาอยู่ต่างจังหวัด และต้องการลดค่าใช้จ่ายในการเดินทาง

การจัดหลักสูตรฝึกอบรมพนักงาน จะต้องได้รับการวางแผนเป็นอย่างดี โดยจะต้องทราบถึงกลุ่มพนักงานเป้าหมายที่จะจัดฝึกอบรม หน้าที่การทำงานของกลุ่มเป้าหมายนั้น จัดหัวข้อการฝึกอบรม จัดตั้งวัตถุประสงค์ของหลักสูตรที่พนักงานจะต้องได้รับหลังจากจบหลักสูตรแล้ว การประเมินผลหลังการฝึกอบรม

15.6. การบริการให้ความช่วยเหลือหลังการติดตั้งระบบ (Support)

การบริการให้ความช่วยเหลือหลังการติดตั้งระบบ คือ การจัดเตรียมความช่วยเหลือเพื่อการแก้ปัญหาที่เกิดขึ้นกับผู้ใช้หลังจากติดตั้งและใช้งานระบบใหม่

การให้การสนับสนุนแก่ผู้ใช้ภายหลังจากการใช้ระบบงานใหม่ สำหรับองค์กรอาจจะมีการแต่งตั้งทีมงานเพื่อคอยให้ความช่วยเหลือ ซึ่งอาจจะมีการจำแนกหน้าที่ในการช่วยเหลือได้หลายลักษณะ เช่น ช่วยเหลือเมื่อมีการติดตั้งโปรแกรมใหม่หรือฮาร์ดแวร์ใหม่ คอยให้คำปรึกษาและความช่วยเหลือแก่ผู้ใช้งานเมื่อเกิดปัญหาในเบื้องต้น แบ่งแยกข้อมูลจากฐานข้อมูลไปสู่เครื่องคอมพิวเตอร์ของผู้ใช้งาน เป็นต้น

15.6.1. การให้ความช่วยเหลือแบบอัตโนมัติ (Automating Support)

สำหรับองค์กรที่มีขนาดเล็ก การจัดตั้งทีมงานเพื่อคอยให้ความช่วยเหลือเมื่อเกิดปัญหาขึ้นนั้น จะทำให้เสียค่าใช้จ่ายในส่วนของค่าตอบแทนแรงงานส่วนนั้น การลดค่าใช้จ่ายในส่วนนั้น องค์กรอาจจะใช้วิธีการให้การช่วยเหลือแบบอัตโนมัติ ด้วยเทคโนโลยีการติดต่อสื่อสารที่ทันสมัยในปัจจุบันเช่น

- On-line Support Forum ผู้ใช้สามารถตั้งกระทู้ไว้เพื่อแนะนำเทคนิคการใช้งานให้กับผู้ใช้คนอื่นได้ทราบ หรือสามารถเปิดประเด็นตั้งคำถามได้โดยผ่านเครือข่ายอินเตอร์เน็ต หรือกรณีที่เป็นองค์กรขนาดเล็กจะสามารถทำได้โดยผ่านเครือข่ายอินทราเน็ต
- Bulletin Board กระดานฝากข้อความหรือข่าวสาร กรณีเช่นนี้ทางองค์กรเองสามารถให้ความรู้หรือข่าวสารแก่ผู้ใช้งานได้โดยฝากเป็นข้อความไว้ที่กระดานฝากข้อความนี้ได้อัตโนมัติ
- Voice-response System : ระบบเสียงตอบรับอัตโนมัติ ผู้ใช้ระบบสามารถเรียกไปยังหมายเลขที่ทางองค์กรกำหนดไว้ โดยระบบเสียงตอบรับอัตโนมัติจะมีเมนูให้เลือกโดยการกดปุ่มโทรศัพท์ตามหัวข้อที่ผู้ใช้ต้องการ

นอกจากนี้องค์กรยังสามารถให้การสนับสนุนหลังการติดตั้งระบบโดยผ่านทาง E-mail ไปยังผู้ใช้ระบบที่เป็นลูกค้าขององค์กรได้อีกด้วย

15.6.2. Help Desk

คือ จุดให้ความช่วยเหลือแก่ผู้ใช้ระบบ ไม่ว่าจะเป็นปัญหาด้านใดก็ตามที่เกิดขึ้นจากการดำเนินงานของระบบ ในปัจจุบันจะสามารถพบกับ Help Desk ได้มากมายซึ่งจะมีชื่อเรียกแตกต่างกันไป และองค์กรเหล่านี้ส่วนใหญ่จะเป็นองคกรที่มีผู้ใช้ระบบเป็นลูกค้า ซึ่งจะเรียก Help Desk ขององค์กรแตกต่างกันไป เช่น Call Center, Customer Service, Web call center, Customer Care เป็นต้น

องค์กรที่จัดให้มี Help Desk จะต้องจัดฝึกอบรมให้กับทีมงานที่จะมาทำหน้าที่ในส่วนนี้เป็นอย่างดี ทีมงานนั้นจะต้องมีความรู้และเข้าใจเกี่ยวกับการดำเนินงานของระบบ และมีทักษะในการตอบคำถามที่ผู้ใช้ระบบซักถาม พร้อมทั้งสามารถยอมรับฟังปัญหาต่างๆ ที่เกิดขึ้นได้อย่างเต็มใจ

การยอมรับระบบใหม่ของผู้ใช้ นับว่ามีอิทธิพลมากต่อการเปลี่ยนแปลงการดำเนินงานจากระบบเดิมมาเป็นระบบใหม่ เพื่อป้องกันไม่ให้เกิดความล้มเหลวในการใช้งานระบบใหม่ จะต้องตระหนักถึงว่าระบบที่พัฒนาขึ้นมาแม้ว่าจะสมบูรณ์ที่สุดแล้วก็ตาม ไม่อาจเป็นระบบที่ดีที่สุดได้ หากผู้ใช้ไม่ยอมรับระบบนั้น และการสร้างการยอมรับจะต้องสร้างความคุ้นเคยให้แก่ผู้ใช้งานได้เห็นว่า การเปลี่ยนแปลงลักษณะการทำงานมาเป็นระบบใหม่นั้นไม่ใช่เรื่องยากหรือซับซ้อน และไม่ใช่เรื่องของการลดบทบาทการทำงานของผู้ใช้ แต่ช่วยให้ผู้ใช้ทำงานได้ดียิ่งขึ้นอันเป็นการเพิ่มพูนประสิทธิภาพให้กับองค์กร และนำประโยชน์สูงสุดให้เกิดกับองค์กรของตนได้ ดังนั้นการจัดฝึกอบรมและการให้การสนับสนุนหลังการติดตั้งระบบจึงเป็นเรื่องที่ไม่ควรมองข้าม

บทที่ 14
การซ่อมบำรุงระบบ (System Maintenance)

การบำรุงรักษาระบบ (System Maintenance) เป็นขั้นตอนสุดท้ายของวงจรการพัฒนาระบบ (SDLC) หลังจากติดตั้งระบบใหม่เพื่อใช้งานแทนระบบเก่าแล้วเป็นที่ทราบกันดีว่า การดำเนินงานกับระบบใหม่ซึ่งผู้ใช้ยังไม่มีความคุ้นเคยมากพอ อาจมีปัญหาในระหว่างการทำงานเกิดขึ้นมากมาย ปัญหาเหล่านั้นสามารถแก้ไขและให้คำปรึกษาได้จากทีมงานที่ถูกแต่งตั้งขึ้นให้เป็นฝ่ายบริการให้ความช่วยเหลือหลังการใช้งานระบบ (Support) ซึ่งรายละเอียดได้กล่าวไว้แล้ว แต่เมื่อปัญหาดังกล่าวได้รับการแก้ไขจนกระทั่งลงตัวและผู้ใช้มีความคุ้นเคยกับระบบใหม่แล้ว เมื่อเวลาผ่านไปนักวิเคราะห์ระบบจะต้องรักษาและพัฒนาประสิทธิภาพการทำงานของระบบให้ยังคงตอบสนองความต้องการของผู้ใช้ได้อยู่เสมอ เนื่องจากการเปลี่ยนแปลงไปของสภาพเศรษฐกิจ ทำให้เงื่อนไขหรือรูปแบบการดำเนินงานอาจเปลี่ยนแปลงตามไปด้วย อีกทั้งเทคโนโลยีทางด้านคอมพิวเตอร์ที่มีการพัฒนาให้ทันสมัยและมีศักยภาพในการทำงานเพิ่มขึ้นเรื่อยๆ ดังนั้นจึงต้องมีการซ่อมบำรุงและรักษาระบบให้ยังคงมีประสิทธิภาพในการทำงานอยู่เสมอ

14.1. กิจกรรมในขั้นตอนการซ่อมบำรุงระบบ (System Maintenance)

การซ่อมบำรุงระบบ เป็นขั้นตอนเพื่อการดูแลระบบเมื่อมีข้อผิดพลาดเกิดขึ้น ทีมงานที่รับผิดชอบหน้าที่ในส่วนนี้ จะต้องทำการแก้ไขข้อผิดพลาดนั้น เช่น ข้อผิดพลาดของโปรแกรม (Errors) ระหว่างผู้ใช้ปฏิบัติงาน เป็นต้น รวมทั้งยังเป็นขั้นตอนเพื่อการปรับปรุง ดัดแปลง หรือแก้ไขทั้งโปรแกรมและขั้นตอนการทำงานของระบบ

นอกจากนี้ยังจะต้องปรับปรุงการทำงานบางส่วนของระบบ ให้สามารถทำงานตามความต้องการของผู้ใช้ที่เพิ่มขึ้นอยู่เสมอ ทั้งนี้เนื่องจากการเปลี่ยนแปลงไปของสภาพเศรษฐกิจ ทำให้เงื่อนไขหรือรูปแบบการดำเนินงานอาจเปลี่ยนแปลงตามไปด้วยอีกทั้งเทคโนโลยีทางด้านคอมพิวเตอร์ที่มีการพัฒนาให้ทันสมัยและมีศักยภาพในการทำงานเพิ่มขึ้นเรื่อยๆ ดังนั้นจึงต้องมีการซ่อมบำรุงและรักษาระบบให้ยังคงมีประสิทธิภาพในการทำงานอยู่เสมอ

การเริ่มดำเนินการซ่อมบำรุงระบบนั้น สามารถเริ่มได้ทันทีหลังจากที่เริ่มใช้งานระบบใหม่ แต่จะซ่อมบำรุงเป็นระยะเวลานานเพียงใดนั้น ขึ้นอยู่กับองค์กร ผู้ใช้ระบบ และทีมนักพัฒนาที่มีความเห็นพ้องกันว่าระบบใหม่ที่ใช้งานมาในช่วงระยะเวลาหนึ่งนั้น ได้กลายเป็นระบบเก่าที่ไม่สามารถทำงานได้อย่างมีประสิทธิภาพอีกต่อไป และทางองค์กรจะพิจารณาเพื่อตัดสินว่าจะพัฒนาระบบขึ้นมาใหม่หรือจะซื้อจากผู้ขายภายนอกองค์กรดี ซึ่งทำให้ครบรอบของวงจรการพัฒนาระบบ (SDLC)

หลังจากใช้งานระบบใหม่ไปได้ช่วงระยะเวลาหนึ่ง ทีมงานซ่อมบำรุงระบบ อาจจะต้องมีการปรับปรุงการทำงานของระบบให้ทันสมัย ตามสภาพการณ์ของธุรกิจและจากการร้องขอของผู้ใช้ แต่เนื่องจากการปรับปรุงระบบในแต่ละครั้งนั้น จะต้องเกิดค่าใช้จ่ายซึ่งหมายถึงต้นทุนเข้ามาเกี่ยวข้องด้วย ดังนั้นในการร้องขอให้ปรับปรุงระบบแต่ละครั้ง จะต้องได้รับการอนุมัติจากผู้บริหาร นักวิเคราะห์ระบบ โปรแกรมเมอร์และผู้เกี่ยวข้องทุกฝ่าย และเพื่อให้การดำเนินการซ่อมบำรุงเป็นไปอย่างมีลำดับขั้นตอนจึงมีกระบวนการในการซ่อมบำรุงระบบ ทั้งหมด 4 ขั้นตอน ดังนี้

14.1.1. เก็บรวบรวมคำร้องขอให้ปรับปรุงระบบ

14.1.2. วิเคราะห์ข้อมูลการร้องขอเพื่อการปรับปรุง

14.1.3. ออกแบบการทำงานที่ต้องการปรับปรุง

14.1.4. ปรับปรุงระบบ

14.1.1. เก็บรวบรวมคำร้องขอให้ปรับปรุงระบบ : เป็นขั้นตอนแรกของการซ่อมบำรุงระบบ เริ่มจากองค์กรจะต้องจัดเตรียมแบบฟอร์มการร้องขอให้ปรับปรุงระบบ ซึ่งแต่ละองค์กรอาจจะมีรูปแบบแตกต่างกันไป หรืออาจจะใช้แบบฟอร์มเดียวกันกับแบบฟอร์มเพื่อขอให้พัฒนาระบบใหม่ โดยผู้ใช้จะระบุถึงปัญหาที่เกิดขึ้น ซึ่งเป็นสาเหตุให้มีความต้องการให้ปรับปรุงระบบ ซึ่งมีปัญหาที่อาจมีความแตกต่างกันออกไป ดังนั้นในการเก็บรวบรวม จะต้องจัดกลุ่มของปัญหาหรือจัดการข้อมูลที่รวบรวมมาได้ให้มีขอบเขตอย่างชัดเจน

14.1.2. วิเคราะห์ข้อมูลการร้องขอเพื่อการปรับปรุง : หลังจากรวบรวมข้อมูลร้องขอได้แล้ว ทีมงานซ่อมบำรุงระบบจะต้องวิเคราะห์ข้อมูลเหล่านั้น เพื่อเพิ่มความเข้าใจในปัญหาซึ่งการวิเคราะห์จะต้องประเมินถึงผลกระทบที่จะเกิดขึ้นกับระบบ หากมีการปรับปรุงการทำงานบางส่วนของระบบ รวมถึงการประเมินถึงความเสี่ยงที่จะเกิดขึ้นและความเป็นไปได้ของการปรับปรุง และทำการพิจารณาอนุมัติว่าคำร้องขอเพื่อปรับปรุงระบบในส่วนใดที่เหมาะสมที่สุด

14.1.3. ออกแบบการทำงานที่ต้องการปรับปรุง : เป็นการออกแบบการทำงานในบางส่วนของระบบที่ต้องการปรับปรุงหรือดัดแปลง กรณีที่การร้องขอนั้นได้รับการอนุมัติแล้วในขั้นตอนนี้มีลักษณะและวิธีการออกแบบระบบคล้ายกับวิธีการออกแบบโดยทั่วไป

14.1.4. ปรับปรุงระบบ : เป็นขั้นตอนสุดท้ายหลังจากที่ได้มีการออกแบบแล้ว ไม่ว่าจะเป็นการออกแบบเพื่อปรับปรุงโปรแกรมหรือปรับปรุงเงื่อนไขในการตรวจเช็คข้อมูลใดก็ตาม ทีมงานจะต้องนำแบบร่างที่ได้ออกแบบไว้มาพัฒนาให้สัมฤทธิ์ผล เช่น ดัดแปลงโปรแกรมเพื่อเพิ่มเติมสิ่งต่างๆ หรือเพิ่ม Table ในฐานข้อมูล เป็นต้น

จากกระบวนการของการซ่อมบำรุงระบบ สิ่งที่ได้ คือระบบที่ได้รับการปรับปรุงหรือดัดแปลงแล้ว ซึ่งไม่ว่าจะเป็นการปรับปรุงครั้งที่เท่าใดก็ตาม สิ่งที่นักวิเคราะห์และทีมงานจะต้องไม่ลืมคือ จะต้องปรับปรุงชุดเอกสารทั้งเอกสารระบบ (System Document) และเอกสารของผู้ใช้ (User Document) ให้เป็นปัจจุบันเสมอ

14.2. ประเภทของการซ่อมบำรุงระบบ

ในระหว่างการซ่อมบำรุงระบบ ทีมงานที่รับผิดชอบอาจจะต้องดำเนินการหลายอย่างตลอดช่วงเวลาของการใช้งานระบบ เช่นปรับปรุง แก้ไขข้อผิดพลาด หรือเปลี่ยนแปลงหน้าที่การทำงานบางส่วนของระบบ เป็นต้น เพื่อให้สามารถใช้งานระบบได้อย่างมีประสิทธิภาพมากขึ้น การดำเนินการดังกล่าวได้ถูกนำมาจัดกลุ่มเพื่อแบ่งประเภทของการซ่อมบำรุงระบบ ซึ่งมีทั้งหมด 4 ประเภทดังนี้

14.2.1. Corrective Maintenance

14.2.2. Adaptive Maintenance

14.2.3. Perfective Maintenance

14.2.4. Preventive Maintenance

14.2.1. Corrective Maintenance

เป็นประเภทที่มีความสำคัญที่สุด เนื่องจากเป็นการซ่อมบำรุงเพื่อความถูกต้องของระบบทันทีที่มีข้อผิดพลาดเกิดขึ้น หรือแก้ไขข้อผิดพลาดให้ถูกต้องนั่นเอง การซ่อมบำรุงระบบประเภทนี้ มักจะเกิดขึ้นทันทีหลังจากที่ได้ติดตั้งระบบงานใหม่ และต้องแก้ไขทันทีที่เกิดข้อผิดพลาดขึ้น เนื่องจากเป็นการแก้ไขเมื่อมีปัญหาเกิดขึ้นทันทีจากการใช้งานระบบใหม่ จึงอาจทำให้เกิดความไม่คล่องตัวในระหว่างการทำงาน หรืออาจทำให้การดำเนินงานหยุดชะงัก ซึ่งเป็นข้อเสียของการซ่อมบำรุงระบบประเภทนี้ แต่เป็นข้อเสียที่องค์กรไม่อาจจะเลี่ยงได้นอกจากต้องการลดข้อผิดพลาดที่จะเกิดขึ้นในการพัฒนาระบบให้ได้มากที่สุด

14.2.2. Adaptive Maintenance

เป็นการซ่อมบำรุงระบบ เพื่อดัดแปลงขั้นตอนการทำงานบางส่วนของระบบตามความต้องการและตามเงื่อนไขในการดำเนินธุรกิจที่เพิ่มขึ้น การซ่อมบำรุงระบบประเภทนี้ไม่จำเป็นต้องดำเนินการในทันทีหลังจากการติดตั้งระบบเหมือนกับแบบแรก เนื่องจากเงื่อนไขทางธุรกิจหรือเทคโนโลยีต่างๆ นั้นค่อยๆ เปลี่ยนแปลงไปตามเวลาและสภาพการณ์ ซึ่งองค์กรสามารถเตรียมพร้อมที่จะเปลี่ยนแปลงได้จากการติดตามข่าวสารเทคโนโลยีและวงการธุรกิจ

14.2.3. Perfective Maintenance

เป็นการซ่อมบำรุงระบบ เพื่อเพิ่มเติมลักษณะของการทำงานบางอย่างเข้าไปในระบบ เพื่อให้ดูแปลกใหม่และสามารถใช้งานได้ง่ายมากขึ้น เช่น เพิ่มลักษณะ Interface ของโปรแกรมใหม่ให้สามารถใช้งานง่ายมากขึ้นยิ่งกว่าเดิม หรือลดขั้นตอนบางอย่างในการป้อนข้อมูลเข้าสู่ระบบ เพื่อให้สามารถนำข้อมูลเข้าสู่ระบบได้เร็วขึ้น เป็นต้น การซ่อมบำรุงระบบประเภทนี้เสมือนเป็นการเติมเต็มประสิทธิภาพการทำงานให้แก่ระบบนั่นเอง

14.2.4. Preventive Maintenance

เป็นการซ่อมบำรุงระบบเพื่อป้องกันหรือลดโอกาสที่จะเกิดข้อผิดพลาดขึ้นในระหว่างการดำเนินงานในอนาคต เช่น การเพิ่มความสามารถในการเก็บข้อมูล ให้สามารถรองรับจำนวนลูกค้าที่เพิ่มมากขึ้นในอนาคต เป็นต้น การซ่อมบำรุงระบบประเภทนี้จะต้องมีการวางแผนไว้ล่วงหน้าว่าจะดำเนินการซ่อมบำรุงระบบเป็นประจำในช่วงเวลาใดบ้าง เช่น เป็นประจำทุกอาทิตย์ หรือทุกเดือน เป็นต้น

จากประเภทของการซ่อมบำรุงระบบทั้ง 4 ประเภท จะเห็นว่า Corrective Maintenance เป็นประเภทที่มีความสำคัญต่อระบบมากที่สุด เนื่องจากเป็นการซ่อมบำรุงที่จะต้องดำเนินการเป็นลำดับแรกหลังจากการติดตั้งระบบเพื่อแก้ไขสิ่งที่ผิดพลาดให้ถูกต้องทันทีที่พบ เพื่อให้ระบบมีความถูกต้องอย่างสมบูรณ์มากที่สุด จึงจะสามารถซ่อมบำรุงระบบในประเภทต่างๆ ไปได้โดยไม่ต้องย้อนกลับมาแก้ไขอีกครั้ง

14.3. ปัจจัยที่มีผลกระทบต่อต้นทุนในการซ่อมบำรุงระบบ

เมื่อใช้งานระบบใหม่ไปช่วงระยะเวลาหนึ่งแล้ว ระบบใหม่นั้นจะกลายเป็นระบบที่ผู้ใช้คุ้นเคยและกลายเป็นระบบเก่าในที่สุด เมื่อสภาพการณ์เปลี่ยนไปและจำเป็นต้องพัฒนาระบบขึ้นมาใหม่อีกครั้ง หากเห็นว่าระบบนั้นไม่สามารถแก้ไขได้แล้วหรือการแก้ไขนั้นต้องใช้เงินทุนเท่ากับการพัฒนาระบบใหม่ ซึ่งหมายถึงการลงทุนจำนวนมากที่มาพร้อมกับความเสี่ยง การซ่อมบำรุงรักษาระบบให้สามารถดำเนินงานอย่างมีประสิทธิภาพและสามารถปรับปรุงแก้ไขได้ง่ายนั้น ถึงแม้จะเกิดต้นทุนในการซ่อมบำรุงแต่เมื่อเปรียบเทียบกับการพัฒนาระบบงานใหม่ทั้งหมดแล้ว ยังนับว่าการซ่อมบำรุงระบบใช้ต้นทุนน้อยกว่ามาก และทำให้องค์กรไม่ต้องเผชิญกับความเสี่ยงที่อาจเกิดขึ้นอีกด้วย

ดังนั้นองค์กรและทีมนักพัฒนาจึงควรพัฒนาระบบให้เป็นระบบที่มีความง่ายต่อการซ่อมบำรุง (Maintainability) ซึ่งหมายถึงระบบนั้นจะต้องสามารถทำความเข้าใจ แก้ไข ดัดแปลงหรือปรับปรุงได้อย่างง่ายดาย ซึ่งปัจจุบันองค์กร ต่างๆ ได้เล็งเห็นความสำคัญของการซ่อมบำรุงระบบมากขึ้นเนื่องจากจะช่วยให้ไม่ต้องเสียค่าใช้จ่ายในการพัฒนาระบบใหม่นั่นเอง

อย่างไรก็ตามการซ่อมบำรุงระบบยังจะต้องเกิดต้นทุนอยู่เช่นกัน แต่จะเกิดต้นทุนสูงหรือต่ำนั้นขึ้นอยู่กับปัจจัยหลายประการที่แวดล้อมต่อระบบงานนั้น ปัจจัยเหล่านั้นได้แก่

14.3.1. จำนวนข้อผิดพลาดที่แฝงอยู่ภายในระบบ (Defects)

14.3.2. จำนวนลูกค้า (Customers)

14.3.3. คุณภาพของเอกสาร (Documentation)

14.3.4. คุณภาพของทีมงานซ่อมบำรุงระบบ (Personnel)

14.3.5. เครื่องมือที่ใช้สนับสนุนการซ่อมบำรุงระบบ (Tools)

14.3.1. จำนวนข้อผิดพลาดที่แฝงอยู่ภายในระบบ (Defects) : คือ จำนวนข้อผิดพลาดที่ไม่สามารถค้นพบได้ภายหลังจากการติดตั้งระบบ ปัจจัยข้อนี้เองที่ทำให้ต้องมีการซ่อมบำรุงระบบแบบ Corrective เพื่อหาข้อผิดพลาดที่เกิดขึ้นและแก้ไขให้ถูกต้องทันที จึงทำให้เกิดต้นทุนในการซ่อมบำรุงระบบขึ้น ถ้าภายในระบบไม่มีข้อผิดพลาด การซ่อมบำรุงแบบ Corrective จะไม่เกิดขึ้นหรือถ้ามี จะใช้เวลาในการค้นหาไม่นาน ทำให้ต้นทุนน้อยไปด้วย

14.3.2. จำนวนลูกค้า (Customers) : ปัจจัยเนื่องจากจำนวนผู้ใช้ที่เป็นลูกค้ามีผลกระทบต่อต้นทุนในการซ่อมบำรุง กล่าวคือ หากองค์กรมีจำนวนลูกค้าน้อยกลุ่มจำนวนการร้องขอเพื่อให้ปรับปรุงระบบจะมีจำนวนน้อยตามไปด้วย ต้นทุนในการซ่อมบำรุงจะต่ำ ไม่ว่าจะเป็นการแก้ไขโปรแกรม การแก้ไขเอกสาร หรือการฝึกอบรม เนื่องจากมีกลุ่มผู้ใช้ซึ่งเป็นลูกค้าเพียงกลุ่มเล็กๆ นั่นเอง ในทางกลับกันหากผู้ใช้ระบบมีจำนวนมาก หลากหลายกลุ่ม จำนวนการร้องขอเพื่อปรับปรุงมีมากตามไปด้วย ทีมงานซ่อมบำรุงระบบจะต้องแยกแยะกลุ่มลูกค้าออกเป็นหลายกลุ่มเพื่อฝึกอบรมสำหรับการปรับปรุงในแต่ละครั้ง ทำให้เกิดค่าใช้จ่ายสูง แต่อย่างไรก็ตามการที่มีจำนวนผู้ใช้ระบบที่เป็นลูกค้ามากราย ยิ่งจะทำให้องค์กรพบข้อผิดพลาดของระบบได้เร็วขึ้นและสามารถแก้ไขได้ตามจำนวนข้อผิดพลาดที่ถูกค้นพบ รวมทั้งยังสามารถค้นพบเงื่อนไขต่างๆ จากลูกค้าซึ่งหมายถึงความต้องการของลูกค้าผู้ใช้ระบบ เพื่อนำมาปรับปรุงให้ตรงต่อความต้องการของลูกค้ามากที่สุด ถึงแม้จะเกิดต้นทุนที่สูงก็ตาม

14.3.3. คุณภาพของเอกสาร (Documentation) : ปัจจัยที่มีผลกระทบต่อต้นทุนในการซ่อมบำรุงในข้อนี้คือ คุณภาพของเอกสาร กรณีที่ทีมนักพัฒนาระบบจัดทำเอกสารของระบบ (System Documentation) ไม่มีคุณภาพ กล่าวคือ ไม่มีการจัดการกับเอกสารที่ดี หรือไม่ปรับปรุงเอกสารให้เป็นปัจจุบัน จะทำให้ทีมงานการปรับปรุงหรือแก้ไขระบบงานในภายหลังนั้นเป็นไปด้วยความลำบาก และใช้เวลาปรับปรุงระบบนานกว่าที่ควรจะเป็น ทำให้ต้นทุนในการซ่อมบำรุงสูงขึ้นไปด้วย

14.3.4. คุณภาพของทีมงานซ่อมบำรุงระบบ (Personnel) : คุณภาพของทีมงานซ่อมบำรุงระบบ เป็นอีกปัจจัยหนึ่งที่ช่วยให้ต้นทุนในการซ่อมบำรุงระบบต่ำลงได้ หากบุคลากรในทีมนี้มีประสบการณ์และความชำนาญ เช่น โปรแกรมเมอร์ที่มีประสบการณ์และความชำนาญจะสามารถดูแลโปรแกรมของระบบงานได้ดีกว่าโปรแกรมเมอร์ที่มีประสบการณ์น้อยกว่า เนื่องจากสามารถทำความเข้าใจโปรแกรมได้เร็วกว่า จึงทำให้ไม่เสียเวลาในการแก้ไขมากนัก

14.3.5. เครื่องมือที่ใช้สนับสนุนการซ่อมบำรุงระบบ (Tools) : เครื่องมือสนับสนุนการซ่อมบำรุงระบบ จะช่วยให้การดูแลระบบและโปรแกรมมีความสะดวกมากขึ้น เช่น ซอฟต์แวร์บางชนิดสามารถสร้างเอกสารต่างๆ ของระบบได้อัตโนมัติ ทำให้ไม่เสียเวลาในการสร้างกรณีที่เอกสารชนิดนั้นยังไม่มีส่งผลให้ต้นทุนในการรักษาระบบน้อยลง

จากปัจจัยดังกล่าวข้างต้น ล้วนแต่ส่งผลกระทบต่อต้นทุนในการซ่อมบำรุงระบบ อย่างไรก็ตามหากในระหว่างการพัฒนาระบบ ทีมนักพัฒนาได้ตระหนักถึงขั้นตอนในการซ่อมบำรุงระบบภายหลังจากการติดตั้งใช้งานระบบแล้ว ควรจะพัฒนาระบบนั้นอย่างมีประสิทธิภาพ มีโครงสร้างของโปรแกรมที่ดี มีการทดสอบโปรแกรมและแก้ไขข้อผิดพลาดให้ได้มากที่สุดเท่าที่จะมากได้ ก่อนที่จะติดตั้งโปรแกรมเพื่อใช้งานจริง จะทำให้ซ่อมบำรุงรักษาระบบได้ง่าย และต้นทุนที่เกิดขึ้นจะน้อยตามไปด้วย

14.4. การจัดการการซ่อมบำรุงระบบ (Maintenance Management)

องค์กรต่างๆ อาจจะมีความตั้งใจที่จะซ่อมบำรุงระบบให้สามารถดำเนินงานอย่างมีประสิทธิภาพให้นานที่สุด เพื่อหลีกเลี่ยงการพัฒนาระบบใหม่ทั้งหมดที่มีต้นทุนสูง ดังนั้นการจัดการซ่อมบำรุงระบบ จึงเป็นข้อหนึ่งที่ควรตระหนักถึงหากองค์กรต้องการให้มีการซ่อมบำรุงระบบอย่างมีประสิทธิภาพแต่เกิดค่าใช้จ่ายไม่มากนัก ในการบริหารงานด้านการซ่อมบำรุงระบบจะมีการจัดการอยู่ทั้งหมด 3 ด้านด้วยกันดังนี้

14.4.1. บุคลากรในทีมงานซ่อมบำรุงระบบ (Maintenance Personnel Management)

องค์กรอาจจะต้องพิจารณาตัดสินใจว่าจะแยกทีมพัฒนาระบบ ออกจากทีมงานซ่อมบำรุงระบบหรืออาจจะตัดสินใจให้เป็นทีมเดียวกัน ซึ่งขึ้นอยู่กับสถานการณ์ของแต่ละองค์กรที่จะตัดสินใจเลือกแนวทางใด ที่จะทำให้การซ่อมบำรุงระบบนั้นมีประสิทธิภาพมากที่สุด ประกอบกับโปรแกรมเมอร์ผู้เชี่ยวชาญบางคนถนัดที่จะพัฒนาโปรแกรมมากกว่าที่จะแก้ไขโปรแกรมที่สร้างขึ้นจากโปรแกรมเมอร์คนอื่น อาจจะเนื่องด้วยเหตุผลใดก็ตาม ดังนั้นจึงขึ้นอยู่กับองค์กรเองว่าจะบริหารงานบุคลากรด้านนี้อย่างไร ให้การซ่อมบำรุงระบบมีประสิทธิภาพมากที่สุดตามความเหมาะสมกับสถานการณ์ขององค์กร

14.4.2. การประเมินผลประสิทธิภาพในการซ่อมบำรุงระบบ (Maintenance Effectiveness Measurement) : นอกจากการบริหารงานด้านบุคลากรในทีมงานซ่อมบำรุงระบบจะต้องมีประสิทธิภาพ และเหมาะสมกับองค์กรแล้ว ในการบริหารจัดการการซ่อมบำรุงระบบยังจะต้องมีการประเมินผลการซ่อมบำรุงระบบด้วย โดยสามารถประเมินได้จากหลักเกณฑ์ดังต่อไปนี้

1. จำนวนข้อผิดพลาดที่เกิดขึ้น
2. ระยะห่างของเวลาที่เกิดข้อผิดพลาดแต่ละครั้ง
3. ชนิดของข้อผิดพลาด

จากหลักเกณฑ์ในการประเมินจำนวนข้อผิดพลาดที่เกิดขึ้น จะมีความสัมพันธ์กับระยะห่างของเวลาที่เกิดข้อผิดพลาดแต่ละครั้ง กล่าวคือ การซ่อมบำรุงระบบที่ดีนั้น หลังจากเริ่มต้นซ่อมบำรุงระบบแบบ Corrective ภายหลังจากติดตั้งระบบแล้ว จำนวนของข้อผิดพลาดจะต้องลดลง และระยะห่างของเวลาที่เกิดข้อผิดพลาดจากจุดหนึ่งไปยังอีกจุดหนึ่ง จะต้องยาวนานขึ้น จึงจะถือว่าการซ่อมบำรุงระบบนั้นได้ผลและมีประสิทธิภาพ

นอกจากนี้การซ่อมบำรุงระบบที่ดี เมื่อเกิดข้อผิดพลาดชนิดใดขึ้นแล้ว จะต้องไม่มีข้อผิดพลาดชนิดนั้นซ้ำอีกเมื่อเวลาผ่านไป

14.4.3. การควยคุมการร้องขอให้ปรับปรุงระบบของผู้ใช้ (Maintenance Requests Control) การบริหารจัดการด้านการควบคุมการร้องขอให้ปรับปรุงระบบของผู้ใช้ เป็นการจัดการประการสำคัญเนื่องจากบางองค์กรที่มีผู้ใช้งานระบบหลายกลุ่ม จะเกิดการร้องขอให้ปรับปรุงระบบมากมายแตกต่างกันไปตามกลุ่มของผู้ใช้ ดังนั้นการควบคุมการร้องขอจึงเป็นสิ่งสำคัญที่จะต้องพิจารณาว่าการร้องขอจากกลุ่มผู้ใช้กลุ่มใดที่จำเป็นมากที่สุด หรือการร้องขอให้แก้ไขข้อผิดพลาดใดที่เร่งด่วนที่สุด

ดังนั้นองค์กรจะต้องมีกระบวนการในการพิจารณาคัดเลือกการร้องขอเหล่านั้น ซึ่งอาจจะเริ่มจากพิจารณาที่การร้องขอเพื่อแก้ไขข้อผิดพลาดที่ร้ายแรงก่อน แล้วจึงพิจารณาคัดเลือกการร้องขอเพื่อการปรับปรุงระบบหรือเพื่อการดัดแปลงระบบองค์กรและทีมงานจะต้องพิจารณาว่า การดัดแปลงนั้นจะมีผลกระทบต่อขั้นตอนการทำงาน และเงื่อนไขในการดำเนินธุรกิจมากน้อยเพียงใด โดยอาจจะเริ่มจากการประเมินผลกระทบเหล่านั้น จัดหมวดหมู่ของการร้องขอ จัดลำดับความสำคัญ และจัดลำดับในการปรับปรุงให้แก่คำร้องขอเหล่านั้น อย่างไรก็ตามกระบวนการในการพิจารณาคำร้องขอต่างๆ ควรมีการวิเคราะห์ถึงความเสี่ยง ความเป็นไปได้ของการปรับปรุงแต่ละโครงการอย่างรอบคอบด้วย