หอสมุดกลาง สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

บทที่ 3
แนวความคิดและทฤษฎีที่เกี่ยวข้อง

จากปัญหาที่เกิดขึ้นธนาคารแห่งประเทศไทยจึงได้นำระบบการหักบัญชีทางอิเล็กทรอนิกส์ (Electronic Clearing System : ECS) มาใช้กับธนาคารสมาชิก ซึ่งระบบการหักบัญชีทางอิเล็กทรอนิกส์นี้มีผลต่อธนาคารสมาชิกคือต้องมีหน้าที่เตรียมระบบงานภายใน และเครื่องมือที่จำเป็นสำหรับรองรับระบบใหม่นี้ ดังนั้นการวิจัยครั้งนี้จึงมีการออกแบบและพัฒนาระบบงานภายในของธนาคารสมาชิกด้วยวิธีการเชิงวัตถุ (Object Oriented Methodology) โดยมีทฤษฎีที่เกี่ยวข้องคือ

การเรียกเก็บเงินระหว่างธนาคาร

เช็ค (ดังแสดงในภาคผนวก ก) ของธนาคารอื่นที่ลูกค้านำมาฝากเข้าบัญชีที่ธนาคาร โดยปกติธนาคารจะทำการเรียกเก็บเงินระหว่างธนาคาร ได้ด้วยวิธีดังต่อไปนี้

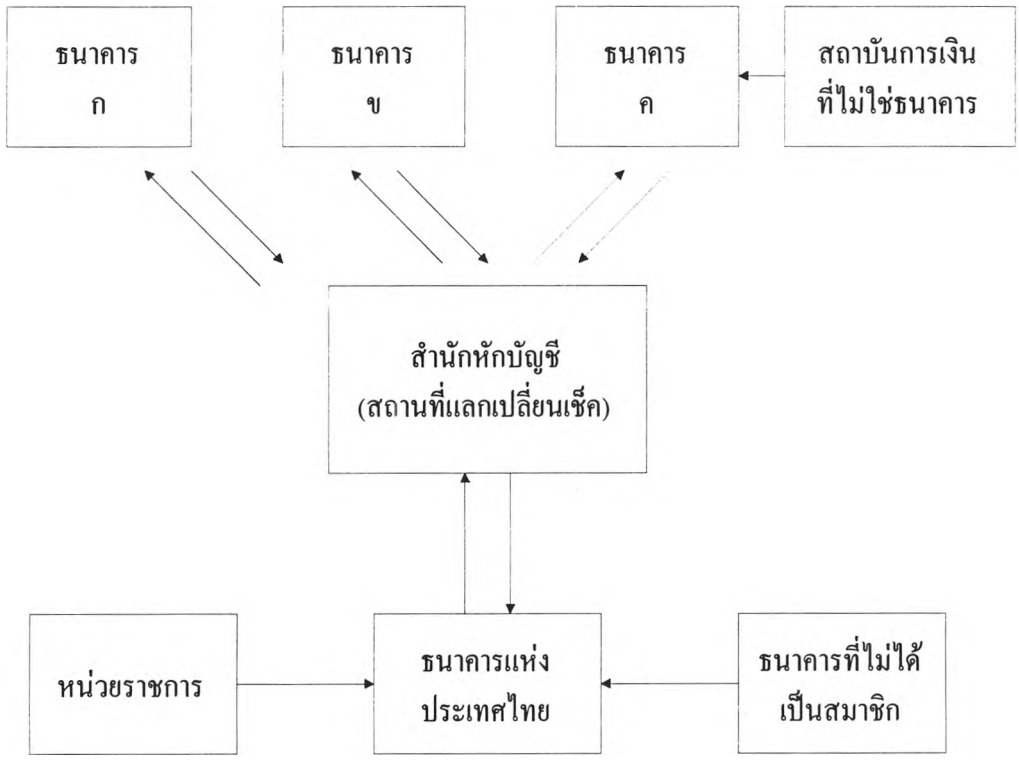
1. เรียกเก็บเงินระหว่างธนาคารผ่านสำนักหักบัญชี

เป็นการเรียกเก็บเงินระหว่างธนาคาร โดยธนาคารผู้ทรงเช็คเรียกเก็บเงินเอาจากธนาคารเจ้าของเช็คนั้นผ่าน"สำนักหักบัญชี"ซึ่งเป็นสถาบันที่จัดตั้งขึ้นโดยธนาคารแห่งประเทศไทยมีทั้งในกรุงเทพและในต่างจังหวัด เพื่อดำเนินการและควบคุมระบบการหักบัญชีระหว่างธนาคาร และเป็นแหล่งกลางในการแลกเปลี่ยนเช็คระหว่างธนาคารที่ได้เข้าเป็นสมาชิกแห่งระบบการหักบัญชีเพื่อทำการหักกลบลบหนี้อันพึงเกิดขึ้นในการแลกเปลี่ยน และชำระหรือเรียกเก็บหนี้ โดยนำผลต่างหลังจากที่ได้หักกลบกันแล้วไปเข้าบัญชีหรือหัก "บัญชีเงินฝาก" ของธนาคารต่าง ๆ ที่มีอยู่ ณ ธนาคารแห่งประเทศไทย ดังแสดงในรูปที่ 3.1

2. เรียกเก็บเงินระหว่างธนาคารโดยตรง

โดยธนาคารผู้ทรงเช็คจะยื่นเช็คเรียกเก็บเงินเอาจากธนาคารเจ้าของเช็คนั้นโดยตรง หรือธนาคารผู้ทรงเช็คใช้วิธีนำเช็คเหล่านั้นฝากเข้าบัญชีของตนที่ธนาคารแห่งประเทศไทย วิธีการเช่นนี้ใช้สำหรับธนาคารที่อยู่นอกระบบการหักบัญชี หรือสำหรับธนาคารที่ดำเนินการอยู่ในต่างจังหวัด

โดยการส่งเช็คไปเรียกเก็บที่ธนาคารนั้นในฐานะตัวแทนเรียกเก็บเงินให้ ในกรณีที่ไม่มีสำนักงานของตนตั้งอยู่



รูปที่ 3.1 ระบบการเรียกเก็บเงินระหว่างธนาคารผ่านสำนักหักบัญชี

ระบบการหักบัญชีทางอิเล็กทรอนิกส์

ในปัจจุบันธนาคารต่าง ๆ ได้พัฒนาเทคโนโลยีเพื่อดึงดูดใจลูกค้า ด้วยบริการที่ดีกว่า สะดวกรวดเร็วกว่า และหวังส่วนแบ่งการตลาดที่มากกว่า ไม่เว้นแม้แต่ธนาคารแห่งประเทศไทยที่พยายามนำระบบเทคโนโลยีสมัยใหม่มาใช้ เพื่อเพิ่มประสิทธิภาพในการทำงานเช่นกัน

สำหรับเทคโนโลยีชิ้นใหม่ที่ธนาคารแห่งประเทศไทยเปิดใช้ไปนั่นคือระบบการหักบัญชีทางอิเล็กทรอนิกส์ (ECS) นับเป็นอีกก้าวหนึ่งที่ธนาคารแห่งประเทศไทยได้พัฒนาขึ้นเพื่อทดแทนระบบเก่าที่ใช้แรงงานคน ซึ่งช้าทำให้ไม่สามารถสนองความต้องการของลูกค้าได้ นับเป็นการ "ปฏิวัติการทำงาน" ครั้งสำคัญในระบบการเคลียริ่งเช็คของไทย

ECS เป็นระบบการหักบัญชีผ่านสื่ออิเล็กทรอนิกส์ โดยรวบรวมข้อมูลเช็คจากธนาคารพาณิชย์ทั่วประเทศเข้ามาเคลียริ่งที่ศูนย์หักบัญชีอิเล็กทรอนิกส์ของธนาคารแห่งประเทศไทยเพื่อคำนวณดุลสุทธิเป็นรายธนาคาร และชำระดุลให้แก่ธนาคารลูกข่ายก่อนที่จะส่งข้อมูลไปให้ธนาคารสมาชิกตัดบัญชีลูกค้าต่อไป (ทัศนัย ตระกูลตระกล, 2539) โดยธนาคารธนาคารแห่งประเทศไทยได้ออก "ระเบียบธนาคารแห่งประเทศไทยว่าด้วยการหักบัญชีระหว่างธนาคารในกรุงเทพมหานครด้วยอิเล็กทรอนิกส์ พ.ศ.2539" (ดังแสดงในภาคผนวก ค) เพื่อให้ธนาคารสมาชิกได้ปฏิบัติตาม (ธนาคารแห่งประเทศไทย, 2539)

<u>โครงข่ายของระบบการหักบัญชีทางอิเล็กทรอนิกส์</u>

ระบบโครงข่ายของระบบการหักบัญชีทางอิเล็กทรอนิกส์ จะต้องมีศูนย์หักบัญชีอิเล็กทรอนิกส์เป็นศูนย์กลางรองรับข้อมูลทั้งหมด โดยต้องมีส่วนที่เชื่อมโยงไปถึงธนาคารลูกข่ายและยังมีส่วนที่เชื่อมไปถึงสาขาของธนาคารลูกข่ายด้วย พอจะแยกการทำงานออกเป็น 3 ส่วนดังนี้

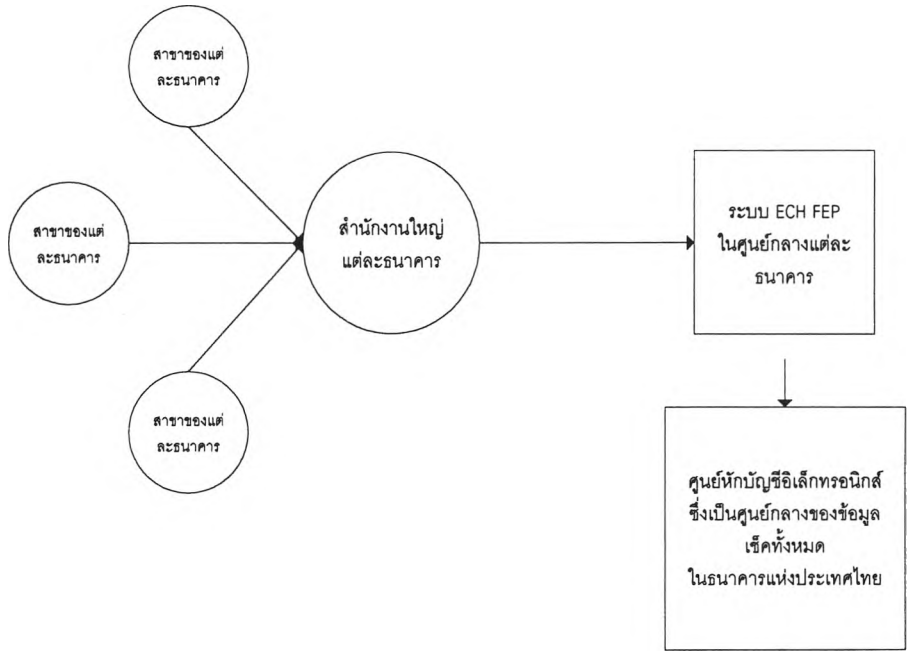
1. ศูนย์หักบัญชีอิเล็กทรอนิกส์ เป็นศูนย์กลางของข้อมูลเช็คทั้งหมดโดยจะมีคอมพิวเตอร์ ยูนิกซ์ โฮส (UNIX Host) 2 เครื่อง เป็นตัวเก็บข้อมูล เครื่องอ่านเช็ค (Reader) 6 เครื่อง มีความสามารถในการคัดแยกเช็คได้ 1,700 ฉบับต่อนาที อุปกรณ์เทอร์มินัลสำหรับรีเจค (Reject)และรีเอนทรี (Reentry) ข้อมูลจากธนาคารลูกข่ายอีก 45 เครื่องที่ปรับให้รองรับข้อมูลจำนวนมากในชั่วโมงเร่งด่วนได้

2. ธนาคารพาณิชย์ลูกข่าย จะมีส่วนของระบบเคลียริ่งเชื่อมต่อกับระบบของศูนย์หักบัญชีอิเล็กทรอนิกส์ และสาขาของธนาคารเอง ด้วยระบบออนไลน์ (Online) ผ่านสายเช่าเฉพาะ (Leased Line) ขององค์การโทรศัพท์แห่งประเทศไทย และมีศูนย์กลางข้อมูลเพื่อรองรับข้อมูลเช็คจากสาขาต่าง ๆ ทั่วกรุงเทพฯ และส่งข้อมูลไปในรูปแบบมาตรฐานของธนาคารแห่งประเทศไทยที่เรียกว่า Electronic Clearing House Front End Processor (ECH FEP) เพื่อให้มีรูปแบบในการส่งข้อมูลเดียวกัน ซึ่งจะทำให้การประมวลข้อมูลทำได้ง่ายขึ้น

3. สาขาของธนาคารลูกข่ายจะต้องมีระบบการส่งข้อมูลเดียวกัน และมีระบบเชื่อมต่อไปถึงสำนักงานใหญ่ ซึ่งระบบของสาขานี้ต้องมีประสิทธิภาพสามารถส่งข้อมูลได้อย่างถูกต้องแม่นยำ เพราะถ้าเกิดความผิดพลาดในการส่งข้อมูลจะมีผลไปถึงการคำนวณดุลสุทธิที่คลาดเคลื่อนได้

การทำงานของทั้ง 3 ส่วนดังแสดงในรูปที่ 3.2 จะต้องมีความสัมพันธ์เป็นชุดเดียวกัน ดังนั้นทางธนาคารแห่งประเทศไทยจึงได้กำหนดรูปแบบของการเคลียริ่งออกมาเรียกว่า อีซีเอช เฟพ "ECH FEP" เพื่อให้มีความเป็นหนึ่งเดียวกันทั้งระบบ ลดค่าใช้จ่ายและความยุ่งยากช่วยให้

การประสานงานเป็นไปได้ดี สามารถประกันความสำเร็จของระบบได้ในระดับหนึ่ง โดยระบบอีซีเอช เฟพ (ECH FEP) เป็นระบบที่มีความคล่องตัวในการขยายขีดความสามารถเชื่อมโยงกับระบบของธนาคารสมาชิกได้ง่าย ส่งถ่ายข้อมูลได้ถูกต้อง รวดเร็ว และแม่นยำ อีกทั้งเป็นระบบที่ใช้แรงงานและเวลาในการพัฒนาน้อยอีกด้วย



รูปที่ 3.2 โครงข่ายของระบบการหักบัญชีทางอิเล็กทรอนิกส์

เนื่องจากระบบการหักบัญชีทางอิเล็กทรอนิกส์เป็นระบบ ที่ต้องอาศัยข้อมูลที่ถูกต้องแม่นยำ ธนาคารแห่งประเทศไทยจึงต้องเตรียมการพัฒนาระบบมาเป็นขั้นตอน โดยขั้นแรกจะต้องให้ธนาคารลูกข่ายทำเช็คอิเล็กทรอนิกส์ที่มีคุณภาพ ต้องมีแถบโค๊ดไลน์ (Codeline) ที่สามารถใช้หมึกแม่เหล็ก (Magnetic Ink Character Recognition) พิมพ์เลขบัญชี รหัส และจำนวนเงินลงไปได้ หลังจากนั้นจึงให้ธนาคารทดสอบการส่งข้อมูลพร้อมกับตัวเช็คจริง ๆ มาที่ศูนย์หักบัญชีฯ เมื่อธนาคารมีความเข้าใจระบบและส่งข้อมูลไม่ผิดพลาดแล้วจึงเปิดใช้ระบบอย่างเต็มรูปแบบ

จากตารางที่ 3.3 ระบบการเคลียริ่งเช็ค ECS จะเริ่มรับเช็คจากลูกค้าในเวลาทำการของธนาคารเรื่อยไปจนถึงประมาณ 13.00 - 14.00 น. ก็จะปิดรับเช็ค ซึ่งระหว่างที่เปิดรับเช็คนั้นจะส่ง

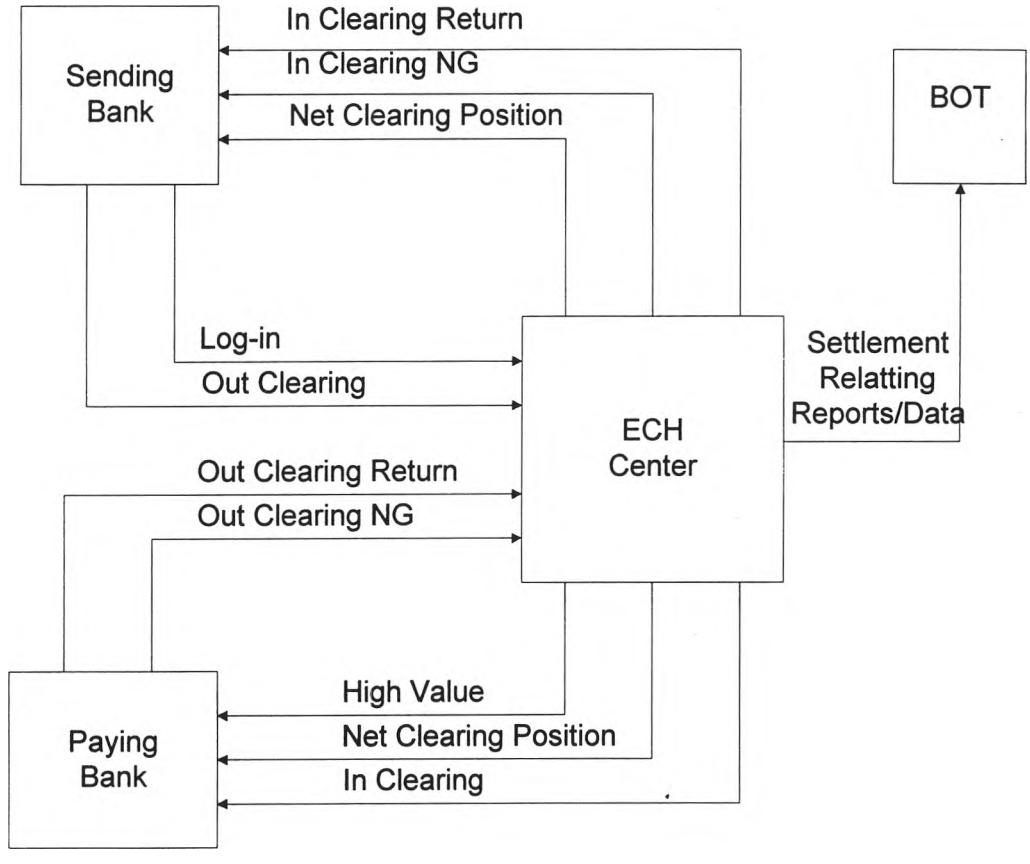
ข้อมูลที่บรรจุในแถบโค๊ดไลน์ของเช็คแต่ละใบไปที่ธนาคารลูกข่าย (สำนักงานใหญ่ของธนาคารพาณิชย์)ธนาคารลูกข่ายจะรวบรวมข้อมูลแล้วส่งตามสายเช่าเฉพาะไปที่ศูนย์หักบัญชีอิเล็กทรอนิกส์ ศูนย์หักบัญชีฯก็จะประมวลข้อมูลทั้งหมดออกมาเป็นดุลสุทธิของแต่ละธนาคารแล้วจึงส่งข้อมูลกลับไปที่ธนาคารลูกข่ายเพื่อตัดบัญชีลูกค้า

ในด้านของตัวเช็คหลังจากที่ปิดรับเช็คแล้วสาขาจะส่งเช็คมาที่ธนาคารลูกข่าย (สำนักงานใหญ่) แล้วจะรวบรวมเช็คส่งไปศูนย์หักบัญชีฯในตอนเย็น เพื่อเข้าเครื่องอ่านเช็ค (Reader) ที่มีความเร็วในการคัดแยกเช็ค 1,700 ฉบับต่อนาที คัดแยกเช็คออกเป็นธนาคารและสาขา แล้วนำตัวเช็คที่คัดแยกมายืนยันข้อมูลออนไลน์ (Online) ที่ส่งมาก่อนหน้านี้

หลังจากนั้นจะเข้ามาสู่ระบบ Electronic Cheque Return หรือ ECR โดยที่ตัวเช็คถูกส่งกลับไปที่ธนาคารลูกข่าย ธนาคารลูกข่ายจะกระจายเช็คไปตามสาขาต่าง ๆ เพื่อตรวจสอบความถูกต้องของการตัดบัญชีว่า บัญชีใดสามารถตัดได้ บัญชีใดตัดไม่ได้ และส่งข้อมูลบัญชีที่ตัดไม่ได้กลับมาที่ธนาคารลูกข่าย และศูนย์หักบัญชีฯ ต่อจากนั้นศูนย์หักบัญชีจะทำดุลสุทธิรอบเช็คคืนแล้วส่งข้อมูลกลับไปยังธนาคารลูกข่ายเพื่อปรับยอดบัญชีในเวลาประมาณ 12.00 น. และจะปลดเคลียริ่งส่วนตัวเช็คจะตามมาที่ศูนย์หักบัญชีฯทีหลัง เพื่อที่จะนำตัวเช็คมาคืนกันที่ศูนย์หักบัญชีฯในช่วงบ่าย โดยมีแผนภาพที่แสดงถึงทางเดินของข้อมูลเช็คในระบบการหักบัญชีทางอิเล็กทรอนิกส์แสดงในรูปที่ 3.4

วันที่	เวลา	รายละเอียด
1	18.30 -...(15.30) น.	ธนาคารสมาชิกรับฝากเช็คจากลูกค้า พร้อมทั้ง encode จำนวนเงิน และส่งข้อมูลเช็คมายังศูนย์หักบัญชีฯ
	15.45 น.	ศูนย์หักบัญชีฯ ปิดรับข้อมูลเช็คจากธนาคารสมาชิก
	15.45 น. - 16.00 น.	ศูนย์หักบัญชีฯ ส่งข้อมูลดุลการหักบัญชีเบื้องต้น และเช็คมูลค่าสูงให้ธนาคารสมาชิก
	16.00 น. - 16.30 น.	ธนาคารสมาชิกตอบรับปฏิเสธการจ่ายเงินตามเช็คมูลค่าสูงที่มีเงินในบัญชีไม่พอจ่าย
	16.45 น.	ศูนย์หักบัญชีฯ คำนวณดุลการหักบัญชีและแจ้งดุลฯ ให้ธนาคารต่าง ๆ
	17.15 น.	ธนาคารแห่งประเทศไทยชำระดุลการหักบัญชี
	19.00 น.	ศูนย์หักบัญชีฯ ปิดรับเช็คจากธนาคารสมาชิก
	19.00 น.	ศูนย์หักบัญชีฯ อ่านและคัดแยกเช็คเป็นรายสาขาเรียงลำดับตามหมายเลขบัญชี ฯลฯ
2	8.30 -...(11.30) น.	ธนาคารสมาชิกส่งข้อมูลเช็คคืนที่ปฏิเสธการจ่ายเงินและศูนย์หักบัญชีฯ คำนวณดุลการหักบัญชีรอบเช็คคืน (ECR)
	12.00 น.	ธนาคารแห่งประเทศไทยชำระบัญชีรอบเช็คคืน (ECR)
	13.00 น.	ลูกค้าสามารถเบิกถอนเงินจากเช็คที่นำฝาก

ตารางที่ 3.3 การดำเนินการหักบัญชีในระบบการหักบัญชีทางอิเล็กทรอนิกส์



รูปที่ 3.4 แผนภาพที่แสดงถึงทางเดินของข้อมูลเช็คในระบบการหักบัญชีทางอิเล็กทรอนิกส์

แนวคิดหลักของวิธีการเชิงวัตถุ

เทคโนโลยีเชิงวัตถุ (Object Technology) เป็นแนวคิดของการพัฒนาโปรแกรมแบบหนึ่ง โดยจะผนึกเอาข้อมูล (Data) และการปฏิบัติการ (Operation) เข้าด้วยกันเป็นวัตถุ (Object) และกำหนดตัวประสาน (Interface) ระหว่างวัตถุต่าง ๆ โดยถ้าวัตถุหนึ่งต้องการจะใช้ข้อมูลในอีกวัตถุหนึ่ง จะต้องติดต่อผ่านส่วนติดต่อนี้เท่านั้น จากหลักการนี้จะเห็นว่า การเปลี่ยนแปลงแก้ไขโปรแกรมและข้อมูลใด ๆ ในวัตถุหนึ่งจะมีผลกระทบต่ออีกวัตถุหนึ่งน้อยมาก ซึ่งจะทำให้การแก้ไขโปรแกรมทำได้สะดวก และรวดเร็วยิ่งขึ้น นอกจากนี้ยังมีการถ่ายทอดคุณสมบัติจากวัตถุหนึ่งไปยังอีกวัตถุหนึ่งได้อีกด้วย ทำให้สามารถนำโปรแกรมที่มีอยู่เดิมกลับมาใช้ใหม่ได้ โดยผู้พัฒนาโปรแกรมไม่ต้องทราบเลยว่าโปรแกรมที่มีอยู่นั้นเขียนขึ้นมาอย่างไร เพียงแค่ทราบว่าโปรแกรมนั้นทำอะไร และทำการเพิ่มคุณสมบัติอื่น ๆ ที่ผู้พัฒนาโปรแกรมต้องการเข้าไป ดังนั้นการพัฒนาโปรแกรมใหม่ ๆ จะทำได้รวดเร็วขึ้น (ศรัณย์ อินทโกสุม, 2536) ซึ่งมีแนวคิดที่เกี่ยวข้อง เช่น

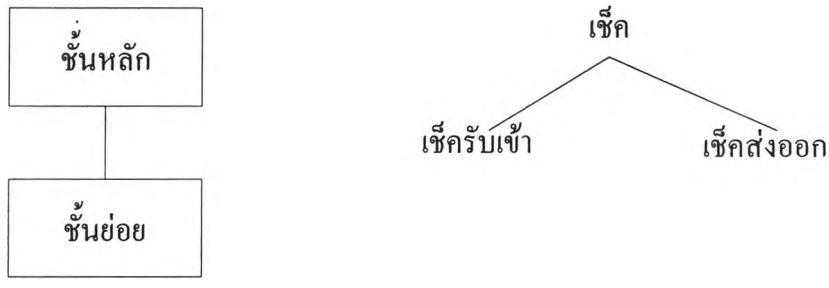
1. การแปลงเป็นนามธรรม (Abstraction) คือการจำกัดขอบเขตให้มองเฉพาะสิ่งที่เกี่ยวข้องเท่านั้น

2. การห่อหุ้ม (Encapsulation) คือการนำเอาข้อมูลที่ได้จากการ Abstraction มาเก็บไว้รวมกับชุดของการปฏิบัติการที่จะใช้กับข้อมูลนั้น โดยที่การเข้าถึงข้อมูลดังกล่าวจะกระทำได้ผ่านปฏิบัติการที่เก็บไว้ด้วยกันนี้เท่านั้น เปรียบเสมือนกับว่าทั้งข้อมูลและปฏิบัติการถูกห่อไว้ด้วยกัน เช่น ถ้าเรามีวัตถุซึ่งเรียกว่า เช็ค เราอาจเก็บคุณลักษณะของเช็คได้แก่ หมายเลขเช็ค หมายเลขบัญชีที่ต้องถูกหักเงิน หมายเลขบัญชีที่จะรับเงิน รหัสธนาคารและรหัสสาขาที่เป็นผู้ส่งไปเรียกเก็บเงิน รหัสธนาคารและรหัสสาขาที่เป็นผู้จ่ายเงิน และจำนวนเงินที่เรียกเก็บ และเก็บปฏิบัติการที่กระทำได้กับเช็ค ได้แก่ การรับเข้าเช็ค การส่งออกเช็ค การปฏิเสธการจ่ายเงินกับเช็คแต่ละฉบับ การรับการปฏิเสธการจ่ายเงินจากธนาคารอื่นที่ส่งไปเรียกเก็บ เป็นต้น

3. ข้อความปฏิบัติการ (Message) คือการที่วัตถุติดต่อกับวัตถุอื่นโดยการส่งข้อความ โดยข้อความจะประกอบด้วยชื่อผู้รับ, ข้อความ และพารามิเตอร์ (Parameter) ซึ่งจะมีการตรวจสอบว่าผู้รับอยู่ในคลาสไหน คลาสนั้นมีการปฏิบัติการนี้หรือไม่ และหากมีการปฏิบัติการนี้อยู่จะถูกกระตุ้นให้ทำงาน ถ้าไม่มีจะเกิดการข้อผิดพลาด

4. การจำแนกชั้น (Classification) คือการจำแนกวัตถุเป็นชั้น (Class) ที่มีคุณลักษณะเหมือนกัน โดยจัดการจำแนกเป็นชั้นลดหลั่นกันไปทำให้เห็นได้ว่ามีชั้นย่อยที่ประกอบกันเป็นชั้นในระดับที่สูงขึ้นมา เช่น คลาสของเช็คอาจมีชั้นย่อย (Subclass) เป็น เช็ครับเข้า เช็คส่งออก ซึ่ง

ความสัมพันธ์จะทำให้เกิดการถ่ายทอดมรดกด้วย วัตถุส่วนใหญ่อาจจำแนกเป็นชั้นได้มากกว่า 1 วิธี ขึ้นอยู่กับขอบเขตมุมมองของแต่ละคน ตัวอย่างการจำแนกชั้นแสดงในรูปที่ 3.5



รูปที่ 3.5 การจำแนกวัตถุให้เป็นชั้น

5. การรับมรดก (Inheritance) คือการถ่ายทอดทั้งข้อมูลและปฏิบัติการไปยังชั้นย่อย ซึ่งเป็นการรับคุณลักษณะที่มีอยู่ในชั้นระดับสูงกว่ามาเป็นคุณลักษณะในชั้นระดับต่ำกว่า เปรียบได้กับการรับมรดก ถ้าเราจำแนกชั้นของเช็คเป็นชั้นย่อยได้แก่ เช็ครับเข้า และเช็คส่งออก จะเห็นได้ว่าเช็คในระดับย่อยนั้นก็มีคุณลักษณะที่ได้จากระดับในก่อนหน้านั้น และแต่ละตัวก็จะมีรายละเอียดที่เป็นคุณลักษณะของชั้นนั้นนั่นเอง อันเป็นสิ่งที่จำแนกความแตกต่างของชั้นย่อยที่มาจากชั้นหลักเดียวกัน การรับมรดกมี 2 ชนิดคือ

5.1 การรับมรดกทางเดียว (Single Inheritance) โดยแต่ละชั้นย่อยจะมี 1 ชั้นหลักเท่านั้น

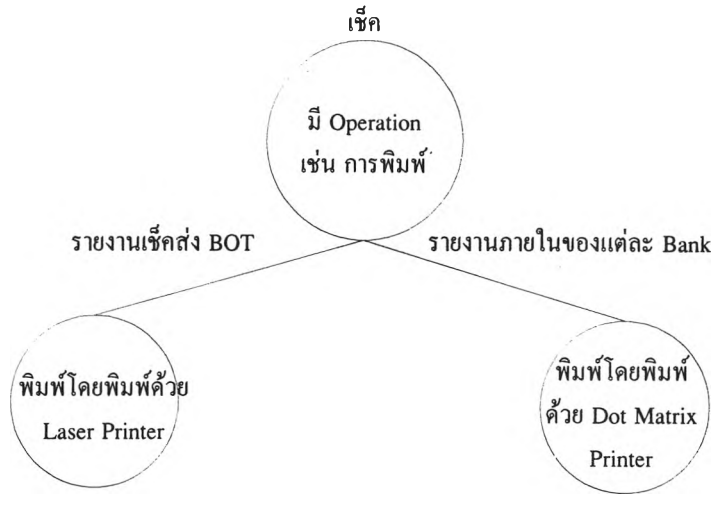
5.2 การรับมรดกหลายทาง (Multiple Inheritance) โดยใน 1 ชั้นย่อย มีชั้นหลักได้มากกว่า 1 ชั้นหลัก

6. การนำมาใช้ได้อีก (Reuse) โดยมีแนวความคิดจากการรับมรดก ดังนั้นจึงไม่ต้องมีการเริ่มต้นสร้างใหม่ จะสามารถถ่ายทอดจากของเดิมได้

7. โพลีมอฟีซึม (Polymorphism) มี 2 รูปแบบ

7.1 ในคลาสต่างคลาสกันมีชื่อการปฏิบัติการชื่อเดียวกัน ขึ้นอยู่กับว่าจะส่งข้อความปฏิบัติการให้กับวัตถุไหน ซึ่งการที่มีชื่อการปฏิบัติการเหมือนกันไม่จำเป็นว่าต้องมีการทำงานเหมือนกันดังแสดงในรูปที่ 3.6

7.2 จากการรับมรดกสามารถนำการปฏิบัติการมาดัดแปลงแก้ไขได้ โดยข้อความปฏิบัติการเหมือนเดิม



รูปที่ 3.6 การเปลี่ยนแปลงการปฏิบัติการโดยข้อความยังคงเดิม

8. คอนเคอเรนซี (Concurrency) การส่งข้อความจากวัตถุหนึ่งไปยังวัตถุอื่น ๆ พร้อมกัน

9. ความคงอยู่ (Persistant) การที่วัตถุใด ๆ จะคงอยู่หรือไม่หลังจากการปฏิบัติการ

10. วัตถุ (Object) เป็นการรวมข้อมูลและการปฏิบัติการเข้าด้วยกัน ซึ่งมีการนิยามเกี่ยวกับวัตถุดังนี้

10.1 วัตถุคือแอ็บสแทรคชัน (Abstraction) ของบางสิ่งบางอย่างในโดเมน (Domain) ซึ่งสามารถเก็บข้อมูลข่าวสาร (Information) ได้และสามารถทำงานติดต่อประสานงานกันได้ โดยจะรวมทั้งข้อมูลและการปฏิบัติการไว้ด้วยกัน

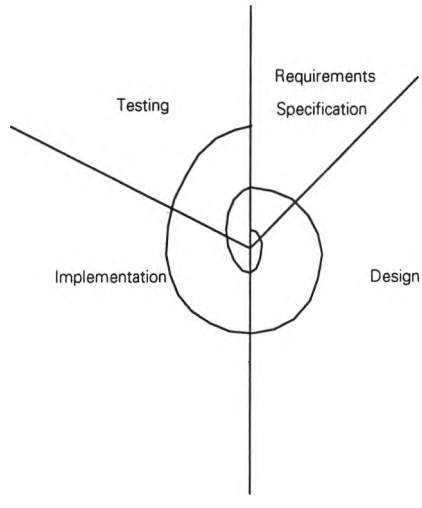
10.2 วัตถุคืออะไรก็ตามที่อาจสัมผัสได้หรือสัมผัสไม่ได้ จะเป็นเหตุการณ์หรือการกระทำ (Action) อะไรก็ได้ ซึ่งจะมีสถานะ, พฤติกรรม และคุณสมบัติเฉพาะ โดยคุณสมบัติเฉพาะนี้ในแต่ละวัตถุต้องไม่ซ้ำกันสามารถแยกแยะได้

10.3 วัตถุเป็นสิ่ง ๆ หนึ่งที่สร้างมาจากแบบของวัตถุ (Object Type)

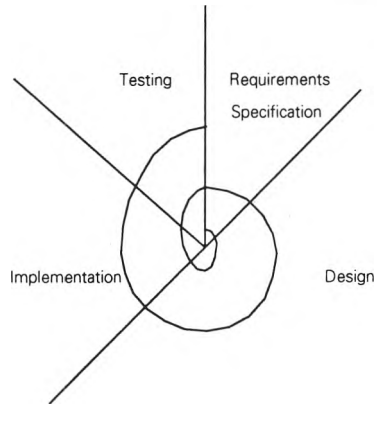
<u>วงจรชีวิตของการพัฒนาซอฟต์แวร์</u>

วงจรชีวิตของการพัฒนาซอฟต์แวร์จะเริ่มเกิดขึ้น เมื่อมีการกำหนดความต้องการ (Requirements Specification) แล้วต้องมีการออกแบบระบบ (Design) การนำไปใช้ให้เกิดผลตามความต้องการ (Implemented) และการทดสอบ (Testing)

ถ้าซอฟต์แวร์นั้นผ่านการทดสอบ จะมีการนำซอฟต์แวร์ที่ได้ไปติดตั้งเป็นผลิตภัณฑ์ให้กับผู้ใช้ที่ร้องขอมา แต่เมื่อผู้ใช้มีปัญหาในการใช้งาน เช่น ผลลัพธ์ที่ได้ผิดพลาด หรือซอฟต์แวร์ที่นำไปใช้ไม่สามารถตอบสนองความต้องการได้ซึ่งมีผลกระทบโดยตรงกับระบบของผู้ใช้ ดังนั้นต้องมีการปรับปรุงพัฒนาซอฟต์แวร์อีกครั้ง ซึ่งจะมีขบวนการจากกำหนดความต้องการ วิเคราะห์ออกแบบ และทดสอบซ้ำอีกครั้ง ได้จำลองวงจรชีวิตของการพัฒนาซอฟต์แวร์โดยใช้การขดเป็นวง (Spiral) ดังแสดงในรูปที่ 3.7 และรูปที่ 3.8 ซึ่งสามารถทำซ้ำในขั้นตอนต่าง ๆ ได้ (Rebecca Wirf-Brock,1990)



รูปที่ 3.7 วงจรชีวิตของการพัฒนาซอฟต์แวร์ที่ใช้ตั้งแต่แรกเริ่มและใช้สืบเนื่องกันมา



รูปที่ 3.8 วงจรชีวิตของการพัฒนาซอฟต์แวร์เชิงวัตถุ

จากรูปที่ 3.7 จะเห็นว่าใช้เวลาส่วนใหญ่ในการออกแบบระบบ (Design) และการนำไปใช้ให้เกิดผลตามความต้องการ (Implemented) แต่การออกแบบซอฟต์แวร์ด้วยวิธีเชิงวัตถุตามรูปที่ 3.8 มีเป้าหมายเพื่อให้สามารถทำได้ง่ายเมื่อมีการใช้ซ้ำ (reused),การปรับปรุงเมื่อถูกใช้จากซอฟต์แวร์อื่น ๆ (refined), การทดสอบ (tested), การแก้ไขเมื่อพบความผิดพลาด (maintained), การเพิ่มเติมการทำงานต่าง ๆ ในโปรแกรมเดิม (extended) ดังนั้นการออกแบบซอฟต์แวร์ด้วยวิธีเชิงวัตถุจะใช้เวลาในการออกแบบมากกว่าขั้นตอนอื่น ๆ เพื่อให้เข้าใจปัญหาให้มากขึ้น และการออกแบบจะคำนึงถึงการนำไปใช้ซ้ำ, การปรับปรุง, การแก้ไข และการเพิ่มเติม

อาจมีการใช้เทคนิคการออกแบบแบบหมุนวน (iterative) ซึ่งเป็นการออกแบบแบบหมุนวนทวนซ้ำจนกว่าจะได้รูปแบบที่ใช้งานได้อย่างแท้จริง ทั้งนี้เพราะรูปแบบของระบบที่ออกมาครั้งแรกอาจยังไม่เป็นที่ยอมรับ ต้องนำมาทดลองใช้งานจริงกับผู้ใช้ระบบ และนำเอาเสียงสะท้อนกลับ (feedback) จากผู้ใช้กลับไปปรับแต่งใหม่ จนกว่าจะได้รูปแบบที่พอยอมรับกันได้ทั้งสองฝ่าย (จรณิต แก้วกังวาล, 2540)

<u>การวิเคราะห์และออกแบบระบบด้วยวิธีเชิงวัตถุ</u>

การพัฒนาระบบเป็นงานที่ยุ่งยากซับซ้อน แต่จำเป็นต้องแยกความซับซ้อนให้อยู่ในแต่ละโมเดล เพื่อให้สามารถจัดการกับระบบได้ (Ivar Jacobson, 1992) โดยมีโมเดลที่แตกต่างกันดังนี้

1) โมเดลความต้องการ (requirements model)
2) โมเดลวิเคราะห์ (analysis model)
3) โมเดลออกแบบ (design model)
4) โมเดลนำไปใช้ให้เกิดผลตามความต้องการ (implementation model)
5) โมเดลทดสอบ (testing model)

1) <u>โมเดลความต้องการ</u> (requirements model)

ความต้องการต่อระบบที่ร้องขอมาจะเป็นสิ่งสำคัญ เพราะมีผลต่อขอบเขตของงานที่จะเกิดขึ้นกับระบบ และยังมีผลต่อการวางแผน การวิเคราะห์ รวมไปถึงการออกแบบ

ความเข้าใจในความต้องการแบบไม่กระจ่างจะมีผลต่อการพัฒนาโครงการ หลักใหญ่ๆ ของความต้องการมี 2 ประการที่มีความสำคัญต่อการพัฒนากระบวนการปฏิบัติการ

1. ความต้องการของผู้ใช้ อธิบายสิ่งที่จำเป็นต่อความต้องการของผู้ใช้ ความต้องการของผู้ใช้จะนำไปสู่การวิเคราะห์ กระบวนการพัฒนาเชิงวัตถุมีการแนะนำให้ใช้ยูสเคสในขั้นตอนนี้

2. ความต้องการที่ไม่เกี่ยวกับฟังก์ชันมีความเกี่ยวข้องกับสิ่งต่าง ๆ ดังนี้

2.1 คุณสมบัติในการใช้ของเครื่องใด ๆ : ความต้องการจะทำให้ช่องว่าง/เวลาและระบบอื่น ๆ ชัดเจนขึ้นในระบบงานที่ผู้ใช้ร้องขอ ความต้องการคุณสมบัติในการใช้เครื่องสามารถผลักดันให้เกิดกิจกรรมในระหว่างขั้นตอนการออกแบบ

2.2 กรอบรองรับระบบ: รายละเอียดของความต้องการเป็นสิ่งจำเป็นที่จะส่งไปสู่ระบบ เช่นฮาร์ดแวร์และภาษาที่ใช้ กรอบการรองรับระบบจะนำไปสู่ทั้งการกำหนดตัวเชื่อมโยงและกิจกรรมของการนำไปใช้ให้เกิดผลตามความต้องการ

ความต้องการที่ไม่เกี่ยวกับฟังก์ชันถ้าไม่ทราบในช่วงแรกของโครงการ สามารถเลื่อนการเตรียมการไปจนกว่าจะทราบในเวลาออกแบบ เวลาเริ่มต้นในการออกแบบการพิจารณาเพื่อตัดสินใจจำเป็นต้องทำก่อนการพัฒนา ซึ่งในเวลานี้ความต้องการที่ไม่เกี่ยวกับฟังก์ชันต้องทำเป็นลำดับก่อนที่ทีมพัฒนาเริ่มทำการพัฒนา

การกำหนดความต้องการอาจแสดงรายละเอียดของโครงการส่วนอื่น ๆ เช่น ส่วนสำคัญของการติดต่อระหว่างผู้ร้องขอและทีมพัฒนาโครงการ ความผูกพันและสัญญาต่าง ๆ ผู้ร้องขอในที่นี้อาจไม่ใช่ผู้ร้องขอโดยตรง โดยอาจหมายถึงบุคคลที่เป็นตัวแทนของผู้ร้องขอ เช่น ผู้ใช้ระบบ

ผู้ร้องขอมีส่วนร่วมในบางฟอร์มหรือบางส่วนซึ่งสำคัญมากในการพัฒนาเชิงวัตถุ โดยเฉพาะอย่างยิ่งในช่วงของการรวบรวมความต้องการ การติดต่อระหว่างทีมพัฒนาและผู้ร้องขอควรเป็นทางการหรือไม่เป็นทางการนั้นแนวความคิดนี้ไม่มีสำคัญมากนัก การที่จะเป็นทางการหรือไม่เป็นทางการจะปรากฏความสำคัญได้อย่างชัดเจน เมื่อทุกหน่วยงานที่เกี่ยวข้องเห็นด้วยในรายละเอียดดังกล่าวที่จะดำเนินการต่อไป การกำหนดความต้องการนั้นจำเป็นอย่างยิ่งในการใช้ถ้อยคำที่ชัดเจน ง่ายและไม่กำกวม ทั้งผู้ร้องขอและทีมพัฒนาต้องเข้าใจในความต้องการของระบบ และความเข้าใจของทั้งสองฝ่ายต้องตรงกัน ความเข้าใจแบบไม่กระจ่างหรือการเปลี่ยนแปลงความต้องการบ่อย ๆ จะเป็นเหตุให้ระบบงานเสียไป หากเข้าใจในความต้องการของระบบแล้วมีผลทำให้การพัฒนาระบบนั้นประสบความสำเร็จได้

ในส่วนของการกำหนดความต้องการ เอกสารของระบบงานรวมควรประกอบด้วยเอกสารย่อย ๆ ในส่วนของ

- การกำหนดปัญหา (Problem Statement)
- ยูสเคส โมเดล (Use Case Model)
- ความต้องการที่ไม่เกี่ยวกับฟังก์ชัน (Nonfunctional Requirements)

- การออกแบบตัวเชื่อมโยงกับผู้ใช้ (User Interface Design)

ซึ่งเนื้อหาของการกำหนดปัญหาเป็นการสรุปปัญหาที่เกิดขึ้น ซึ่งระบบต้องแก้ไขปัญหาดังกล่าว

เอกสารที่เกิดขึ้นในช่วงการกำหนดความต้องการ อาจไม่มีความสัมพันธ์โดยตรงกับการพัฒนาระบบเชิงวัตถุ แต่แบบฟอร์มบางฟอร์มของการกำหนดความต้องการทำให้เพิ่มความสะดวกยิ่งขึ้นในการเริ่มพัฒนาระบบเชิงวัตถุมากกว่าแบบอื่น เช่น ยูสเคสมีประโยชน์มากในการหาความต้องการของฟังก์ชัน

1.1 การกำหนดปัญหา (Problem Statement)

การกำหนดปัญหาเป็นการสรุปปัญหาที่เกิดขึ้นซึ่งระบบที่ต้องการพัฒนาต้องแก้ไขปัญหาดังกล่าว โดยในขั้นตอนนี้ทุกคนในทีมงานควรเข้าใจให้ตรงกัน และยอมรับในเนื้อหาของปัญหาทั้งหมด ควรรีบแก้ไขทันทีหากมีความผิดพลาดเกิดขึ้น เช่นความเข้าใจไม่ตรงกันของทีมงาน

การกำหนดปัญหาต้องทำก่อนงานส่วนอื่น ๆ โดยการกำหนดปัญหาต้องเขียนสรุปแล้วพยายามหาว่าเราต้องการอะไร ทำไมต้องการอย่างนั้น ซึ่งไม่ควรเกี่ยวข้องในส่วนของปัญหาว่าจะถูกแก้ไขอย่างไร

ในส่วนของการกำหนดปัญหาสามารถช่วยชี้ให้เห็นว่ามีวัตถุที่เกี่ยวข้อง (object) และพฤติกรรม (behaviors) อย่างไรบ้าง รวมไปถึงการกำหนดขอบเขตของระบบงานที่ต้องการพัฒนา

การกำหนดปัญหาไม่ควรเขียนจากบุคคลที่เกี่ยวข้องกับระบบในบางส่วนเท่านั้น ควรเขียนจากบุคคลที่รู้ระบบทั้งหมด หรือผู้บริหาร หรือผู้จัดการโครงการ ซึ่งเมื่อเขียนแล้วควรมีการทบทวนข้อความทั้งหมดจากทีมงานที่เกี่ยวข้องอีกครั้ง (James Rumbaugh, 1991)

1.2 ยูสเคสโมเดล (Use Case Model)

ยูสเคสโมเดลเป็นฟอร์มที่ช่วยให้สะดวกขึ้นในการหาความต้องการของฟังก์ชันในระดับบนสุด (top level) ยูสเคสโมเดลประกอบไปด้วยกลุ่มแอ็คเตอร์ (actor) แสดงให้เห็นเฉพาะผู้กระทำภายนอกระบบเท่านั้น และยูสเคส (Use Cases) ซึ่งเป็นการใช้ระบบโดยแอ็คเตอร์และส่วนเชื่อมต่อระหว่างแอ็คเตอร์กับยูสเคส

ยูสเคสโมเดลถือว่าเป็นศูนย์กลางของเอกสารความต้องการ สำหรับการพัฒนาระบบเชิงวัตถุ ซึ่งเสนอว่าระบบควรจะทำอะไร โดยตรงกันข้ามกับการกำหนดความต้องการที่ไม่เกี่ยวกับ

ฟังก์ชันเพราะจะกำหนดให้มีคุณสมบัติการใช้ของเครื่องใด ๆ (Performance), ความน่าเชื่อถือ (Reliability), ความเป็นไปได้ (Availability) เข้าไว้ในระบบด้วย

ยูสเคส (Use Cases) จะอธิบายอย่างละเอียด มองเห็นได้รวมไปถึงพฤติกรรมของระบบ ซึ่งพฤติกรรมสามารถเห็นได้ชัดจากภายนอกโดยการเชื่อมต่อกับระบบ กลุ่มของพฤติกรรมที่มองเห็นได้เป็นตัวกำหนดความต้องการฟังก์ชันของระบบ ยูสเคสอาจต้องเชื่อมต่อกับแอ็คเตอร์หนึ่งคนหรือมากกว่า เช่น ยูสเคสคือการควบคุมการบินของอากาศยานทุกชนิด ดังนั้นแอ็คเตอร์คือนักบินและเรดา (Radar) โดยที่แอ็คเตอร์อาจเป็นบุคคลหรือระบบอื่น ๆ เช่นระบบจัดการฐานข้อมูล

ยูสเคสโมเดลคือแบบฟอร์มซึ่งแสดงฟังก์ชันของระบบที่ผู้ใช้ต้องการ ซึ่งทุกฝ่ายที่เกี่ยวข้องต้องมีความเข้าใจอย่างชัดเจนในรายละเอียด หากผู้ใช้ไม่เข้าใจงานหลักของระบบแล้ว ผู้ใช้ก็ไม่สามารถตรวจสอบความต้องการได้อย่างถูกต้อง

ยูสเคสโมเดลเป็นแบบฟอร์มที่ตรงกันข้ามกับฟอร์แมตอื่น คือใช้ความต้องการฟังก์ชันในระดับบนสุดเพราะเน้นถึงการเชื่อมต่อและฟังก์ชันจากจุดแรกถึงจุดสุดท้าย เมื่อได้กำหนดแอ็คเตอร์ในยูสเคสโมเดลแล้ว ต้องรู้ว่าขอบเขตของระบบเป็นอย่างไร และการเชื่อมต่อที่จำเป็นของระบบ กำหนดยูสเคสในยูสเคสโมเดลแล้วต้องอธิบายว่าระบบใช้แอ็คเตอร์หนึ่งคนหรือมากกว่าอย่างไร หรือแอ็คเตอร์จะใช้ระบบอย่างไร

ความผิดพลาดไม่ค่อยมีโอกาสเกิดขึ้นในเรื่องของการออกแบบภายในเพราะว่าพฤติกรรมที่เห็นได้ใช้ยูสเคสและต้องเชื่อมโยงโดยตรงกับแอ็คเตอร์ ซึ่งช่วยให้แน่ใจในยูสเคสโมเดลที่กำหนดขอบเขตของระบบ แต่ไม่รวมถึงโครงสร้างภายใน, เครื่องมือเครื่องใช้ หรืออัลกอริทึม (Algorithm) ถ้ายอมให้รายละเอียดภายในดังกล่าวรวมเข้าไปด้วยในเอกสารของความต้องการ จะทำให้มีเอกสารมากขึ้นผู้ใช้อาจสับสนได้ และเกิดการบังคับการออกแบบในส่วนที่ไม่จำเป็น

ยูสเคสโมเดลควรเขียนจากทีมเล็ก ๆซึ่งเสนอว่าควรเป็นทั้งผู้ใช้และทีมพัฒนา (วิเคราะห์) รวมไปถึงผู้จัดการโครงการ และหัวหน้าทีม อย่างน้อยควรมีผู้เชี่ยวชาญระบบหนึ่งคน หากเป็นไปได้ควรมีผู้ใช้ระบบหนึ่งคน

ยูสเคสโมเดลเป็นส่วนหนึ่งของการติดต่อระหว่างผู้ใช้กับทีมพัฒนา ดังนั้นควรเขียนในช่วงแรก ๆ ของการรวบรวมความต้องการ

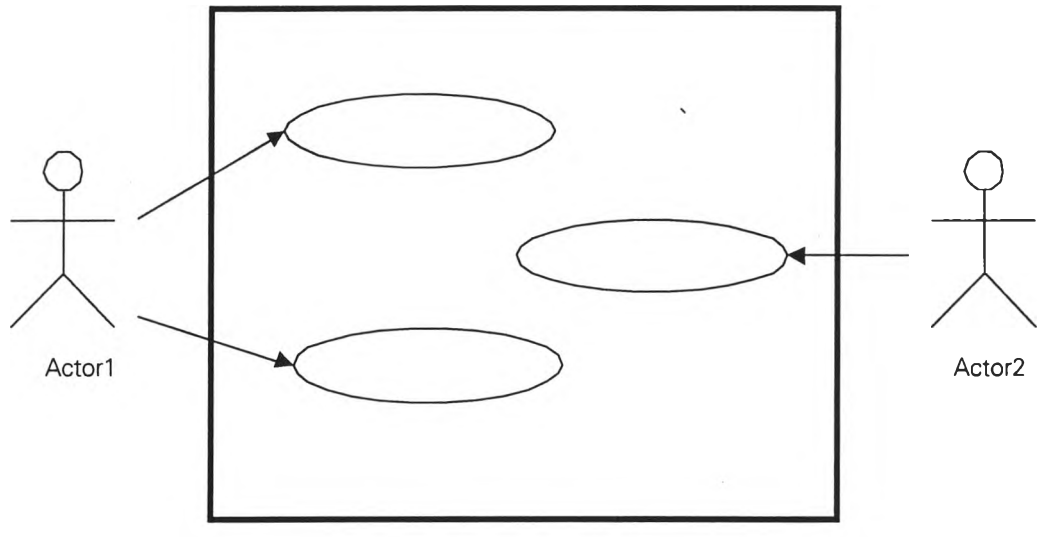
จุดเริ่มสำหรับยูสเคสโมเดลคือ การกำหนดปัญหาซึ่งเป็นการสรุปข้อความของปัญหาที่ระบบควรจะแก้ไข ดัดแปลงปัญหาที่มีไปสู่ยูสเคสโมเดลซึ่งเกี่ยวข้องโดยตรงว่าขอบเขตของระบบอยู่ที่ไหน ใครเป็นผู้ใช้ของระบบ และผู้ใช้คาดหวังอะไรจากระบบ สิ่งเหล่านี้ควรทำโดยการสัมภาษณ์หรือสังเกตผู้ใช้หรือผู้เชี่ยวชาญระบบ บุคคลที่เกี่ยวข้องมีความสำคัญมากในการตรวจสอบยูสเคส

I18077031

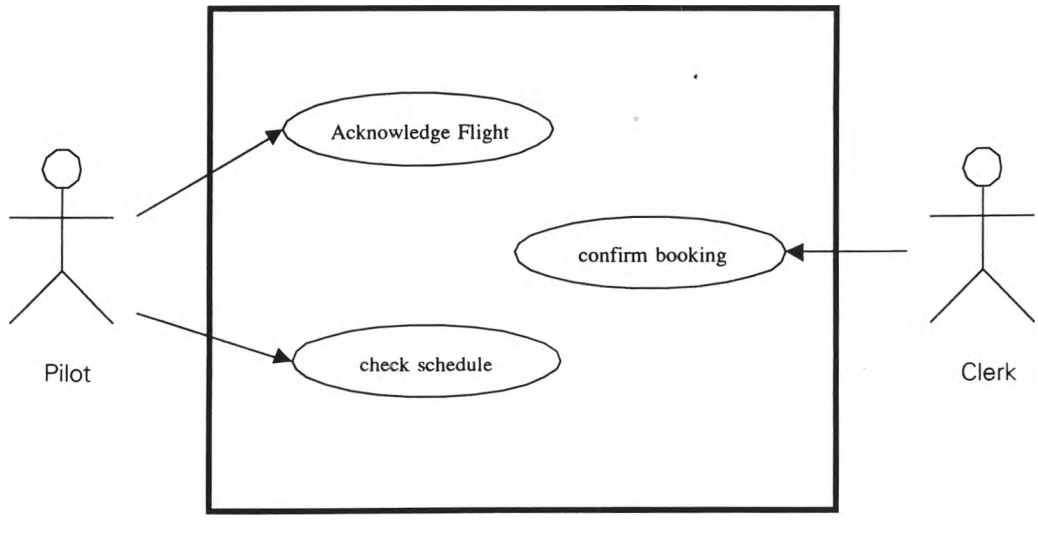
โมเดล

การกำหนดปัญหา และความต้องการทางธุรกิจจะช่วยในการเขียนยูสเคสโมเดลตามความต้องการ หรือความคาดหวังของผู้ใช้ได้อย่างมีเหตุมีผลขึ้น

สัญลักษณ์ที่ใช้สำหรับยูสเคสโมเดลของความต้องการฟังก์ชัน คือการผสมกันระหว่างไดอะแกรมและข้อความ ดังแสดงในรูปที่ 3.9



รูปที่ 3.9 ไดอะแกรมยูสเคส



รูปที่ 3.10 ตัวอย่างไดอะแกรมยูสเคส

จากรูปที่ 3.10 แสดงให้เห็นว่ายูสเคสโมเดลถูกแสดงในไดอะแกรมอย่างไร กรอบสี่เหลี่ยมจะแสดงระบบที่ถูกสร้างขึ้น ข้างในกรอบสี่เหลี่ยมคือยูสเคสซึ่งเป็นสิ่งที่ระบบมีให้ ยูสเคสจะแสดงส่วนที่ติดต่อกับภายนอกคือแอ็คเตอร์ (actor) รูปของแอ็คเตอร์กับยูสเคสโมเดลอาจจะแตกต่างจากตัวอย่างที่เห็น ซึ่งขึ้นอยู่กับชนิดของแอ็คเตอร์ เช่น อาจเป็นระบบอื่น ๆ หรืออาจเป็นบุคคลก็ได้

ถ้าเป็นระบบที่เล็กมากอาจมียูสเคสโมเดลเพียงแค่หนึ่งไดอะแกรม แต่หากเป็นระบบที่ใหญ่อาจมียูสเคสโมเดลหลายไดอะแกรม

ส่วนเชื่อมต่อระหว่างแอ็คเตอร์จะมีการเชื่อมต่อโดยตรงกับยูสเคส ซึ่งการเชื่อมต่อโดยตรงของทั้งสองฝ่าย (ระบบ หรือแอ็คเตอร์) จะเป็นจุดเริ่มของการติดต่อสื่อสารกัน แอ็คเตอร์หนึ่งคนจะเชื่อมกับยูสเคสได้มากกว่าหนึ่งยูสเคส และหนึ่งยูสเคสจะเชื่อมต่อกับแอ็คเตอร์มากกว่าหนึ่งคน

ส่วนเพิ่มเติมสำหรับยูสเคสไดอะแกรม คือข้อความที่จะอธิบายในแต่ละยูสเคสรวมไปถึงแอ็คเตอร์ซึ่งควรมีส่วนนี้ด้วย โดยแต่ละยูสเคสควรจะมีแบบฟอร์มดังแสดงในรูปที่ 3.11

Use case name	:	การเช่าอุปกรณ์กีฬาและสนาม
Definition	:	ยูสเคสนี้เกี่ยวข้องกับลูกค้าที่ต้องการเช่าอุปกรณ์กีฬาและสนามหากเช่าอุปกรณ์กีฬาและสนามได้ แล้วทางคลับต้องเปลี่ยนสถานะของอุปกรณ์กีฬาและสนามเป็นถูกเช่า การเช่าต้องตรวจสอบว่าลูกค้าเป็นสมาชิกของทางยอร์ชคลับหรือไม่ และสามารถเช่าอุปกรณ์กีฬาและสนามได้หรือไม่
Notes	:	- ถ้าลูกค้าเป็นสมาชิกจะได้ส่วนลดพิเศษ - ถ้าลูกค้าคืนอุปกรณ์กีฬาและสนามช้ากว่ากำหนดต้องถูกปรับเงินชั่วโมงละ 50 บาท

รูปที่ 3.11 ตัวอย่างของยูสเคส

Actor name	: พนักงานเก็บเงินร้านวีดีโอ
Definition	: เป็นบุคคลที่ทำหน้าที่เก็บเงินลูกค้าที่มาเช่าวีดีโอ หรือเก็บเงินค่าปรับหากลูกค้าคืนวีดีโอช้ากว่ากำหนด
Notes	:

รูปที่ 3.12 ตัวอย่างของแอ็คเตอร์

ซึ่งชื่อของแอ็คเตอร์และยูสเคสที่เชื่อมต่อกันในรายละเอียดของข้อความดังแสดงในรูปที่ 3.11 นั้นถูกแสดงมาแล้วจากยูสเคสไดอะแกรม ส่วนรายละเอียดที่จะอธิบายแต่ละยูสเคสควรดูตามความเหมาะสมว่ามากหรือน้อยเท่าใด การออกแบบหรือส่วนที่เชื่อมโยงกับผู้ใช้ (interface), ข้อคิดและ/หรือข้อบังคับต่าง ๆ จะเกิดขึ้นระหว่างการรวบรวมความต้องการ ในส่วนหมายเหตุ (Notes) จะแยกความต้องการจากรายละเอียดการออกแบบ การหมายเหตุในเรื่องของส่วนที่เชื่อมโยงกับผู้ใช้จะประกอบด้วยหรืออ้างอิงโครงร่างของผังจอภาพ (Screen Layouts), ต้นแบบของการติดต่อกับผู้ใช้ด้วยภาพกราฟฟิกส์ (GUI Protypes) หรือความสัมพันธ์กับมาตรฐานทั่วไปเช่น RS-232C

สำหรับแอ็คเตอร์มีแบบฟอร์มดังแสดงในรูปที่ 3.12

ใช้ยูสเคสโมเดลเป็นเครื่องมือที่สำคัญ เพื่อสื่อสารกันระหว่างแอ็คเตอร์กับระบบและผู้เชี่ยวชาญระบบ การที่ใช้โมเดลทำให้สะดวกขึ้นเพื่อตรวจสอบความถูกต้องและความสมบูรณ์แอ็คเตอร์กับระบบ เป็นเรื่องง่ายมากถ้าใช้อธิบายในแต่ละกลุ่มของยูสเคส ไม่ว่าจะเป็นกลุ่มของซับเจค (Subject area) หรือโดยกลุ่มของแอ็คเตอร์ และควรพยายามหาช่องว่าง, การวางสิ่งที่เกี่ยวข้องผิดตำแหน่งในขอบเขตของระบบ, การเพิ่มแอ็คเตอร์ หรือข้อผิดพลาดอื่น ๆ

ยูสเคสโมเดลของ Ivar Jacobson and others (Ivar Jacobson, 1992) เป็นการแสดงยูสเคสเต็มรูปแบบ และมีหน้าที่ในการผลักดันกระบวนการพัฒนา Jacobson ใช้ยูสเคสในทิศทางที่ต่างกับวิทยานิพนธ์ฉบับนี้ โดยตำแหน่งยูสเคสของ Jacobson เหมือนเป็นศูนย์กลางและมีงานพัฒนาทั้งหมดเป็นโครงสร้าง แต่ในวิทยานิพนธ์ฉบับนี้จะใช้บทโครงการ (Scenario) แทนที่ยูสเคสในบางส่วน โดยมีเหตุผลดังนี้

1. บทโครงการอธิบายจากจุดเริ่มต้นไปถึงจุดสุดท้ายของพฤติกรรม ซึ่งจะเชื่อมโยงกับความต้องการ, การพัฒนา และเคสที่ใช้ทดสอบโดยวิธีทางตรง โดยใช้ยูสเคสเพื่อกำหนดขอบเขตของระบบ และใช้ยูสเคสเป็นส่วนบนสุดจากบทโครงการซึ่งได้มาภายหลัง ผังยูสเคสจากยูสเคส

โมเดลของ Jacobson ไม่จำเป็นจะต้องไหลจากจุดเริ่มต้นสู่จุดสุดท้าย ยูสเคสของ Jacobson หลายยูสเคสเป็นส่วนประกอบของระดับบนสุด และเป็นจุดเริ่มของยูสเคส

2. บทโครงการอนุญาตให้ใช้ประโยชน์จากการแยกส่วนที่เกี่ยวข้องคือ ยูสเคสโมเดลใช้กำหนดขอบเขตของระบบ และพฤติกรรมของระดับบนสุดของระบบ บทโครงการจะใช้เพื่อแยกแยะการสันนิษฐาน, ผลลัพธ์ และการเปลี่ยนแปลงในสิ่งที่เกี่ยวข้อง

3. บทโครงการยอมให้อธิบายรายละเอียดของระบบในเรื่องพฤติกรรม

1.3 ความต้องการที่ไม่เกี่ยวกับฟังก์ชัน (Nonfunctional Requirements)

ความต้องการที่ไม่เกี่ยวกับฟังก์ชันเป็นที่รวบรวมความต้องการของระบบ ซึ่งไม่มีความสัมพันธ์โดยตรงว่าระบบควรทำอะไร ตัวอย่างของความต้องการที่ไม่เกี่ยวกับฟังก์ชันจะรวมไปถึงความน่าเชื่อถือ, ความเป็นไปได้, คุณสมบัติในการใช้ของเครื่องใด ๆ และรายละเอียดของส่วนประกอบ (ฮาร์ดแวร์ และซอฟต์แวร์) ที่ใช้ โดยปกติความต้องการที่ไม่เกี่ยวกับฟังก์ชันจะใช้ฟอร์มกำหนดว่าระบบควรจะปฏิบัติการอย่างไร

ความต้องการที่ไม่เกี่ยวกับฟังก์ชันเกี่ยวข้องกับส่วนประกอบของฮาร์ดแวร์ และซอฟต์แวร์ การนำเสนอความต้องการส่วนใหญ่จะเสนอออกมาเป็นรูปภาพ

ความต้องการที่ไม่เกี่ยวกับฟังก์ชัน เป็นส่วนประกอบที่สำคัญมากของเอกสารความต้องการ ตัวอย่างเช่น คุณสมบัติในการใช้ของเครื่องใด ๆ ถือว่าเป็นปัจจัยหนึ่งที่กำหนดสถาปัตยกรรมโปรแกรม แต่ไม่มีผลกระทบต่อการวิเคราะห์ปัญหา ตัวอย่างอื่นของความต้องการที่ไม่เกี่ยวกับฟังก์ชัน เช่น เครื่องลูกข่ายแบบกระจายจากศูนย์กลางและเครื่องแม่ข่าย (Server) ใช้ TCP/IP สำหรับการสื่อสาร

นักวางแผน, ผู้จัดการโครงการ และหัวหน้าทีมควรเป็นคนกำหนดความต้องการที่ไม่เกี่ยวกับฟังก์ชันกับข้อเสนอของผู้ใช้

ความต้องการที่ไม่เกี่ยวกับฟังก์ชันถูกกำหนดขึ้นและถูกยอมรับ ซึ่งถือว่าเป็นส่วนหนึ่งในช่วงการรวบรวมความต้องการ

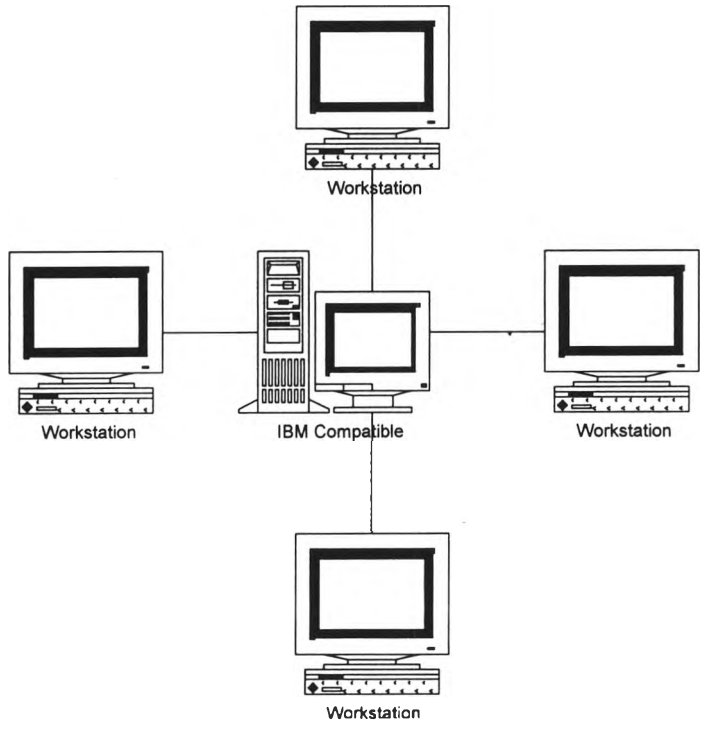
มีวิธีที่จะสร้างหรือตรวจสอบความคลอบคลุมของความต้องการที่ไม่เกี่ยวกับฟังก์ชัน คือผ่านเข้ายูสเคสโมเดล ซึ่งแสดงอยู่ระดับบนสุดของความต้องการฟังก์ชัน สำหรับแต่ละยูสเคสและแต่ละแอ็คเตอร์ไปจนถึงการเชื่อมต่อกับยูสเคส สอบถามว่าข้อบังคับอะไรที่เหมาะสม สำหรับแต่ละจุดที่เชื่อมต่อโดยความต้องการที่ไม่เกี่ยวกับฟังก์ชันจะสัมพันธ์กัน ซึ่งถูกกำหนดก่อนด้วยกลุ่มของหัวเรื่อง เช่น ความน่าเชื่อถือ, ความเป็นไปได้, ระบบความปลอดภัย, คุณสมบัติของการใช้

เครื่องใด ๆ และมาตรฐานต่าง ๆ รวมไปถึงส่วนประกอบอื่น ๆ คือฮาร์แวร์และซอฟต์แวร์ โดยระบบจะต้องหรือควรที่จะแสดงให้เห็นโดยรูปภาพ

สัญลักษณ์ที่ใช้ในไดอะแกรมจะไม่มีรูปแบบที่แน่นอน ถ้าอยากให้เหมาะสมควรแสดงข้อบังคับของฮาร์ดแวร์และซอฟต์แวร์ที่กำหนดการใช้อุปกรณ์ต่าง ๆ ความต้องการที่ไม่เกี่ยวกับฟังก์ชันควรเป็นรายละเอียดในกลุ่มหัวเรื่องที่ได้กล่าวถึงแล้ว

ตัวอย่างของความต้องการที่ไม่เกี่ยวกับฟังก์ชัน สำหรับโปรแกรมกับการใช้ภาพเป็นตัวประสานกับผู้ใช้ (GUI) เช่น

- การสอบถามทุกรายการต้องเรียบร้อยภายใน 0.5 วินาที
- การปรับปรุงทุกรายการต้องเรียบร้อยภายใน 1 วินาที
- ระบบควรจะรองรับผู้ใช้พร้อม ๆ กันได้ 40 เครื่อง
- การพัฒนาเทคโนโลยีควรใช้เทคโนโลยีเชิงวัตถุ เพื่อให้ง่ายในการปรับปรุง การแก้ไขและการบำรุงรักษาในอนาคต
- สถาปัตยกรรมคอมพิวเตอร์ทางฮาร์ดแวร์ดังแสดงในรูปที่ 3.13 ตามความต้องการของผู้ใช้



รูปที่ 3.13 ตัวอย่างไดอะแกรมของความต้องการทางสภาพแวดล้อม

1.4 การออกแบบตัวเชื่อมโยงกับผู้ใช้ (User Interface Design)

โมเดลเชื่อมโยงกับผู้ใช้อธิบายว่าผู้ใช้ของระบบสามารถติดต่อกับฟังก์ชัน ซึ่งเป็นส่วนของโมเดลการวิเคราะห์อย่างไร โมเดลการวิเคราะห์สันนิษฐานว่าผู้ใช้สามารถติดต่อโดยตรงกับวัตถุซึ่งเป็นปัญหาหลักเพื่อการบริการ และรายละเอียดของบริการถูกกำหนด ถูกจัดการ และจุดสนใจคือให้ง่ายในการติดต่อขอใช้บริการ

ผู้ใช้สามารถนำการเชื่อมโยงไปทดสอบ และสามารถตรวจสอบโมเดลในช่วงการวิเคราะห์ได้ โดยในส่วนการออกแบบตัวเชื่อมโยงกับผู้ใช้ประกอบด้วย

- แผนผังของจอภาพ (Screen Flows)
- แบบของจอภาพ (Screen Layouts)

1.4.1 แผนผังของจอภาพ (Screen Flows)

การเชื่อมโยงกับผู้ใช้ (UI) โดยแผนผังของจอภาพสามารถอธิบายลำดับของจอภาพ ตามที่ผู้ใช้คาดหวังว่าอาจได้เห็นจากระบบจริง ๆ ที่เสร็จสมบูรณ์แล้ว ซึ่งแผนผังของจอภาพเหมือนกับการวาดบรรยายให้เห็นภาพการโต้ตอบจากระบบและงานในส่วนของผู้ใช้

แผนผังของจอภาพเป็นการแสดงในส่วนของหน้าที่หลักของระบบเท่านั้น โดยอธิบายถึงความสัมพันธ์ในการเชื่อมโยงระหว่างผู้ใช้กับงานที่มีอยู่ ซึ่งจะไม่แสดงในกรณีที่มีความผิดพลาดเกิดขึ้น

แผนผังของจอภาพที่เชื่อมโยงกับผู้ใช้ จำเป็นที่ต้องออกแบบการเชื่อมโยงกับผู้ใช้ให้ตรงกับความความต้องการของผู้ใช้ และการมีประสิทธิภาพในการโต้ตอบระหว่างผู้ใช้กับซอฟต์แวร์ รูปแบบของผังจอภาพอาจจะมุ่งพัฒนาไปในแนวให้คำแนะนำกับผู้ใช้หรือเหมือนเป็นเพื่อนกับผู้ใช้ คือสามารถใช้งานได้ง่าย เช่น มีเมนูให้เลือก หรือมีข้อความช่วยเหลืออธิบายการทำงาน

ผังของจอภาพมีการโต้ตอบกับยูสเคส โดยรายละเอียดข้อมูลอาจมาจากผู้ใช้ว่าสามารถเข้าสู่ระบบได้อย่างไร และรายการแต่ละฟังก์ชัน

ผังของจอภาพที่เชื่อมโยงกับผู้ใช้โดยปกติจะออกแบบและทำเอกสารโดยทีมงานหรือบุคคลที่ใช้ส่วนเชื่อมโยงนี้โดยตรง นักวิเคราะห์และผู้ใช้ ซึ่งงานของการกำหนดผังของจอภาพสามารถเริ่มใช้ในช่วงแรก ๆ ของโครงการ อาจเป็นก่อน, ระหว่าง หรือหลังการวิเคราะห์โครงการโดยทั่ว ๆ ไปแล้วผังของจอภาพถูกพัฒนาก่อนแบบของจอภาพ แต่ไม่ได้หมายความว่าผังจอภาพ

จะหมดโอกาสในการอ้างอิงถึงแบบจอภาพที่มีอยู่แล้ว เพราะอาจมีการแก้ไขโดยเพิ่มส่วนที่เกี่ยวข้องในช่วงการพัฒนา

เทคนิคในการสร้างผังของจอภาพ โดยจากรายการของบทโครงการให้เลือกบทโครงการที่เป็นงานหลัก แล้วรวมกลุ่มบทโครงการที่เหมือนกันในแต่ละกลุ่มให้วาดภาพง่าย ๆ ของจอภาพ ซึ่งแสดงให้เห็นจุดเริ่มต้นของสถานะของจอภาพก่อนที่บทโครงการอื่นจะทำต่อไป การเขียนผังงานอันใหม่เพื่อจะบรรยายจอภาพถัดไป ซึ่งผู้ใช้สามารถมองเห็นภาพได้และเชื่อมต่อส่วนใหม่นี้กับผังเดิมโดยใช้เส้นตรงดังแสดงในรูปที่ 3.14

ผลลัพธ์ที่ได้จะเป็นลำดับของเรื่องราวซึ่งเป็นส่วนหนึ่งของลำดับชั้น หรือการอธิบายถึงพฤติกรรมภายนอกของระบบ (Dave Collins, 1995)

1.4.2 <u>แบบของจอภาพ</u> (Screen Layouts)

แบบของจอภาพที่เชื่อมโยงกับผู้ใช้จะแสดงให้เห็นจอภาพจริงที่เกิดขึ้น ซึ่งผู้ใช้สามารถเห็นได้เมื่อทำงานกับระบบ โดยปกติแล้วแบบของจอภาพถูกเตรียมไว้สำหรับงานหลักโดยจะหลีกเลี่ยงเงื่อนไขต่าง ๆ แบบของจอภาพจะแสดงข้อความ, รายละเอียด และไอคอน คือมีการจัดการในเรื่องของช่องว่าง, ข้อความ, ไอคอน และสัญลักษณ์ควบคุมต่าง ๆ เช่น ปุ่ม แถบเลื่อน ฟิลด์ ฯลฯ โดยตัวอย่างของแบบของจอภาพดังแสดงในรูปที่ 3.15

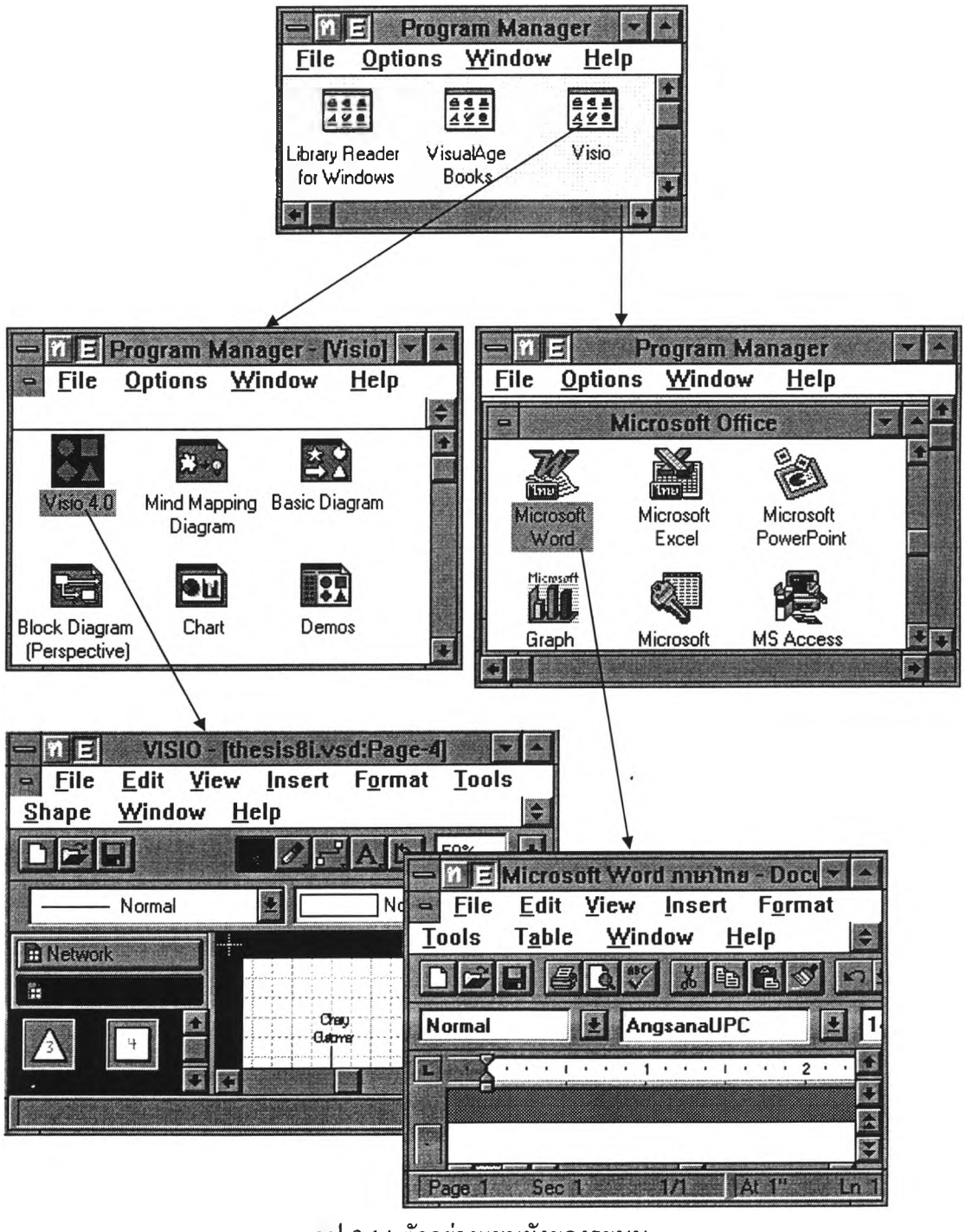
แบบของจอภาพที่เชื่อมโยงกับผู้ใช้มีความจำเป็นเหมือนแผนผังจอภาพ เพื่อตรวจสอบฟังก์ชันและความสามารถในการเชื่อมโยงระหว่างผู้ใช้กับระบบ ควรทำตามมาตรฐานการเชื่อมโยงกับผู้ใช้, ความสะดวกในการใช้โปรแกรม, การใช้ถ้อยคำ และการใช้สี เพื่อให้เหมาะสมกับผู้ใช้ ควรมีการแลกเปลี่ยนความคิดเห็นกันระหว่างผู้ใช้ ผู้เชี่ยวชาญด้านมนุษย์ศาสตร์ และนักออกแบบ

แบบของจอภาพที่เชื่อมโยงกับผู้ใช้ควรออกแบบโดยผู้เชี่ยวชาญทางมนุษย์ศาสตร์ นักออกแบบส่วนเชื่อมโยงกับผู้ใช้ หรือผู้ใช้ระบบโดยตรง แบบของจอภาพควรถูกทบทวนดูอีกครั้งจากผู้ใช้

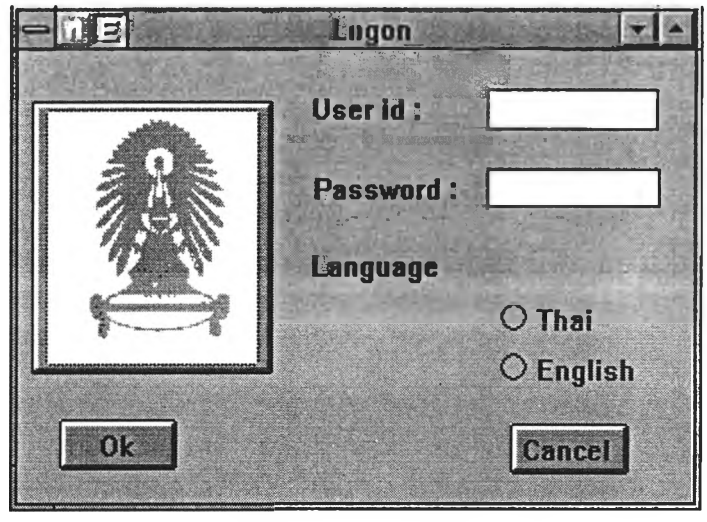
แบบของจอภาพควรเสร็จเรียบร้อยก่อนการออกแบบในส่วนประกอบของระบบ ซึ่งถูกผลักดันโดยตรง หรือควบคุมส่วนเชื่อมโยงกับผู้ใช้ เช่นวัตถุของภาพ และวัตถุที่เป็นตัวควบคุม โดยปกติแล้วแบบของจอภาพจะถูกพัฒนาหลังจากแผนผังของจอภาพ เพราะว่าแผนผังระบุความต้องการในแต่ละจอ แต่ทั้งแผนผังของจอภาพและแบบของจอภาพสามารถพัฒนาก่อน, ระหว่าง หรือหลังจากการวิเคราะห์ระบบ

หากทำก่อนการวิเคราะห์ ดังนั้นแผนผังและแบบของจอภาพจะช่วยค้นหาฟังก์ชัน (โมเดลของยูสเคส) และความต้องการที่ไม่เกี่ยวกับฟังก์ชัน ซึ่งอาจเป็นส่วนเชื่อมโยงกับผู้ใช้ผลักดันให้เกิดการรวบรวมความต้องการ

เมื่อทำคู่ขนานไปกับการวิเคราะห์ แผนผังและแบบจอภาพจะช่วยหาวัตถุในระบบ, สถานะและการโต้ตอบระหว่างวัตถุ (โมเดลวัตถุ, โมเดลสถานะและOIDs) (Dave Collins, 1995)



รูป 3.14 ตัวอย่างแผนผังของระบบ



รูปที่ 3.15 ตัวอย่างแบบของจอภาพ

2) โมเดลวิเคราะห์ (analysis model)

เอกสารการวิเคราะห์จะถูกสร้างขึ้นในระหว่างขั้นตอนการวิเคราะห์โครงการ โดยการวิเคราะห์จะแยกส่วนประกอบแต่ละส่วนคือ การตรวจสอบปัญหา องค์ประกอบและความสัมพันธ์

การวิเคราะห์ด้วยวิธีการเชิงวัตถุ เป็นกระบวนการที่ต้องการระบุวัตถุที่สัมพันธ์กับปัญหาซึ่งควรถูกแก้ไข รวมไปถึงการแบ่งแยกวัตถุและหาความสัมพันธ์ระหว่างแต่ละคลาส

ระหว่างการวิเคราะห์เชิงวัตถุสามารถประยุกต์วิธีการเชิงวัตถุและเทคนิค เพื่อความเข้าใจในฟังก์ชันของระบบ, การพัฒนาฟังก์ชัน และสื่อสารความต้องการทางฟังก์ชันของระบบ

การวิเคราะห์ได้รวมความสนใจไปที่ปัญหาหลัก (Problem Domain) และไม่ได้รวมความสนใจไปที่โปรแกรม ดังนั้นถ้าเป็นไปได้ควรมีคลาสเท่านั้นที่รวมอยู่ในการวิเคราะห์ ซึ่งผู้ใช้ควรจะรู้ขอบเขตของระบบ ด้านหนึ่งของการวิเคราะห์อาจถูกจำกัดโดยขอบเขตของโปรแกรม ส่วนด้านอื่น ๆ เช่นการออกแบบ การเชื่อมต่อกับระบบไม่มีส่วนเกี่ยวข้องกับการวิเคราะห์เลย

ตามแง่คิดนี้แอ็คเตอร์ของยูสเคสโมเดลจะมาเป็นคลาส ซึ่งถูกวิเคราะห์ถึงความรับผิดชอบ และความร่วมมือโดยเหมือนกับการวิเคราะห์คลาสอื่น ๆ หลังจากนั้นอาจเป็นคลาสแรกที่มีอยู่ในปัญหาหลัก การสื่อสารระหว่างขอบเขตของระบบถูกอธิบายในระหว่างการวิเคราะห์

ในส่วนของการวิเคราะห์ระบบ เอกสารของระบบงานรวมควรประกอบด้วยเอกสารย่อย ๆ ในส่วนของ

- ขอบเขตของซับเจค (Subject Area)

- การวิเคราะห์โมเดลเชิงวัตถุ (Analysis Object Model)
- การวิเคราะห์บทโครงการ (Analysis Scenarios)
- การวิเคราะห์รายละเอียดของคลาส (Analysis Class Descriptions)

2.1 ขอบเขตของซับเจค (Subject Area)

ขอบเขตของซับเจคคือโดเมนที่ชัดเจนขึ้นในจุดที่สนใจ ที่สามารถระบุได้ในช่วงเวลาของการวิเคราะห์ ซึ่งเป็นส่วนหนึ่งของปัญหาหลักที่สามารถวิเคราะห์แยกเป็นส่วนได้ ขอบเขตของซับเจคถูกวิเคราะห์ว่าระบบย่อยอะไรที่จะถูกออกแบบ ขอบเขตของซับเจคอาจเป็นหรืออาจไม่เป็นระบบย่อย ๆ ในช่วงเวลาการออกแบบ

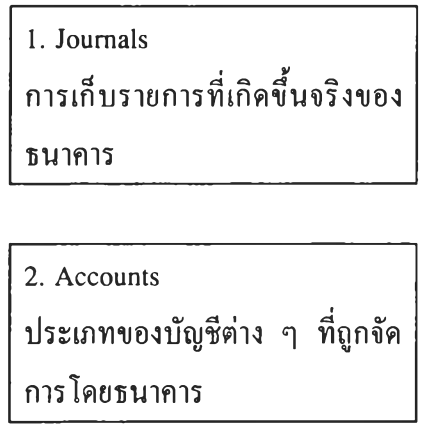
ในวิธีการเชิงวัตถุ ขอบเขตของซับเจคมักถูกกำหนดเป็นกลุ่มของคลาสซึ่งมีความสัมพันธ์กับคลาสอื่นโดยการรับมรดก (inheritance : "is-a"), ผลสรุป (aggregation : "has-a"), และความสัมพันธ์อื่น ๆ ("uses")

อย่างไรก็ตามอาจมีบางคลาสที่ไม่ได้มีความสัมพันธ์ เพียงแค่กับคลาสในขอบเขตของซับเจคตนเอง โดยอาจมีบางคลาสที่อยู่ในขอบเขตของซับเจคนั้นติดต่อกับคลาสในขอบเขตของซับเจคอื่น ๆ ซึ่งเป็นการสนับสนุนความสัมพันธ์แบบ "uses" จากระดับคลาสไปยังระดับของขอบเขตซับเจคดังนั้นขอบเขตของซับเจคจะใช้หรือขึ้นอยู่กับคลาสอื่น ๆ

การใช้ของขอบเขตซับเจคยอมให้ระบบใหญ่ ๆ แบ่งเป็นส่วนย่อย ๆ ในช่วงแรกของการพัฒนา ซึ่งจะช่วยแยกการวิเคราะห์และเตรียมวิถีทางขององค์กร หากปราศจากขอบเขตซับเจคหรือแนวความคิดขององค์กรตรงกัน เอกสารการวิเคราะห์อาจไม่สามารถควบคุมขนาดและปริมาณได้ โดยขอบเขตของซับเจคจะเตรียมวิถีทางของการวิเคราะห์ให้สามารถจัดการในแต่ละส่วนได้

การแบ่งแยกการวิเคราะห์ โดยใช้ขอบเขตของซับเจคจะเป็นการง่ายในการใช้ซ้ำอีกครั้งเพราะสะดวกในการตรวจสอบ ตัวอย่างของขอบเขตซับเจคดังแสดงในรูปที่ 3.16

ผู้ชำนาญการวิเคราะห์ควรขอความเห็นกับผู้ใช้ ผู้เชี่ยวชาญทางด้านโดเมน เพื่อจะแบ่งการวิเคราะห์ออกเป็นขอบเขตของซับเจค ซึ่งควรอยู่ในช่วงเวลาของการวิเคราะห์ และเพื่อใช้จัดการงานการวิเคราะห์ (Peter Coad and Ed Yourdon, 1990)



รูปที่ 3.16 ตัวอย่างของขอบเขตซับเจค

2.2 การวิเคราะห์โมเดลเชิงวัตถุ (Analysis Object Model)

การวิเคราะห์โมเดลเชิงวัตถุ คือโมเดลคงที่ซึ่งเป็นส่วนหนึ่งของปัญหาหลักและสัมพันธ์กับการกำหนดปัญหา โดยทั่วไปการออกแบบโมเดลเชิงวัตถุประกอบด้วยคลาส และความสัมพันธ์ระหว่างคลาส ความสัมพันธ์ส่วนใหญ่ที่ใช้มี 3 แบบคือ การเกี่ยวเนื่องสัมพันธ์กัน (Associations) การไหลมารวมกัน (Aggregations) และการรับมรดก (generalizations/specialications/inheritance)

โมเดลเชิงวัตถุคือวิถีทางที่สำคัญ เพื่อเอกสารจะได้มีรูปแบบที่คงที่ของวัตถุในปัญหาหลัก ภาพจำลองของวัตถุคือการพัฒนาเชิงวัตถุให้แตกต่างจากการพัฒนาแบบเดิม ความคิดพื้นฐานที่จะแยกระบบออกเป็นส่วน ๆ สู่คลาสของวัตถุ

โมเดลเชิงวัตถุคืองานในเชิงสถาปัตยกรรมและการวิเคราะห์ มีความสำคัญมากในการได้รับความร่วมมืออย่างเต็มที่จากผู้ที่เกี่ยวข้อง ตั้งแต่วัตถุโดยทั่วไปที่ถูกเสนอให้อยู่ในโมเดลตามปัญหาหลัก ผู้ที่เกี่ยวข้องทั้งหมดควรให้อินพุทและตรวจสอบโมเดล ผู้ที่เกี่ยวข้องควรเข้าใจปัญหาเพื่อลดความผิดพลาดในการวิเคราะห์

การวิเคราะห์โมเดลเชิงวัตถุควรพัฒนาในช่วงแรกของการวิเคราะห์ แต่มีความจำเป็นที่ต้องปรับปรุงระหว่างการออกแบบหรือการนำไปใช้ให้เกิดผลตามความต้องการ ถ้าโดเมนมีการเปลี่ยนแปลง

ในทางเทคนิคควรจะ

1. ระบุคีย์ของขอบเขตของปัญหาหลักและพฤติกรรม โดยพฤติกรรมของวัตถุในช่วงเวลาของการวิเคราะห์สามารถเป็นไปได้ทั้งกิจกรรมและการให้บริการ

2. กำหนดคลาสที่เกี่ยวข้อง โดยได้จากคำนามสั้น ๆ หรือสิ่งที่เป็นนามธรรม

3. เชื่อมต่อคลาสที่เลือกแล้วโดยกำหนดความสัมพันธ์ระหว่างคลาส การเกี่ยวข้องสัมพันธ์กัน และการรับมรดก

4. เพิ่มความรับผิดชอบ

- คีย์แอททริบิวส์
- พฤติกรรม
- การไหลไปรวมกัน/ผลสรุป

5. วนซ้ำจนกระทั่งโมเดลมีเสถียรภาพ

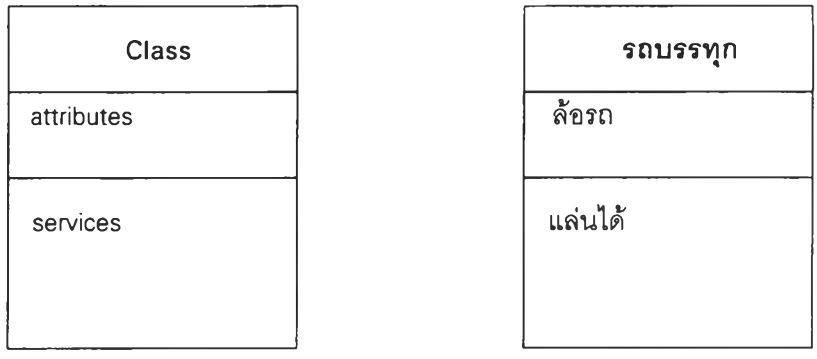
6. ปรับปรุงรายละเอียดของคลาส

หลังจากวิเคราะห์โมเดลเชิงวัตถุเรียบร้อยแล้ว ถ้าโดเมนมีขนาดใหญ่ควรเลือกแบ่งโมเดลสู่ขอบเขตของซับเจค หรือระบบย่อย ๆ ในเทคโนโลยีโมเดลเชิงวัตถุ (OMT) โมเดลขนาดใหญ่ต้องการการจัดการภายใน และการแบ่งเป็นส่วน ๆ จะช่วยให้รวมคลาสเป็นกลุ่มและพิจารณาไปยังกลุ่มของโมเดลนั้น

การวิเคราะห์โมเดลเชิงวัตถุคือฟอร์มพิเศษของเอกสาร เพื่อให้ได้ความสัมพันธ์ระหว่างคลาสของวัตถุในระบบหรือโปรแกรม การวิเคราะห์โมเดลเชิงวัตถุเสนอไดอะแกรมของคลาสได้อย่างดีโดยมีข้อมูลที่แสดงในไดอะแกรมของคลาสคือ

- คลาส
- ความสัมพันธ์
    1. การรับมรดก (generalizations/specialications/inheritance [IsA : คือ])
    2. การเกี่ยวเนื่องสัมพันธ์กัน (association, knowsAbout : รู้เกี่ยวกับ)
    3. การไหลไปรวมกัน/ผลสรุป (aggregation [HasA : มี])
- แอททริบิวส์
- พฤติกรรม

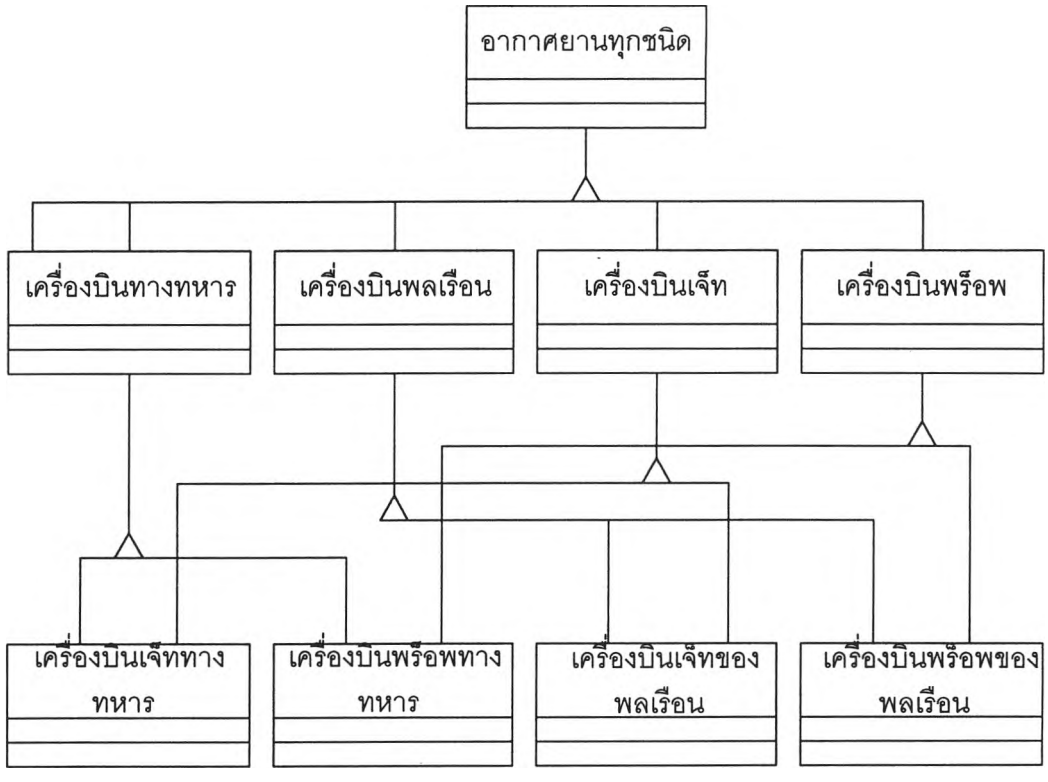
โดยที่คลาสสามารถวาดเป็นกล่องโดยแบ่งเป็น 3 ส่วน ซึ่งชื่อคลาสอยู่ในส่วนแรก รายการของแอททริบิวส์อยู่ในส่วนกลาง และรายการของบริการหรือการปฏิบัติการอยู่ส่วนท้ายของกล่องสี่เหลี่ยม ดังแสดงในรูปที่ 3.17



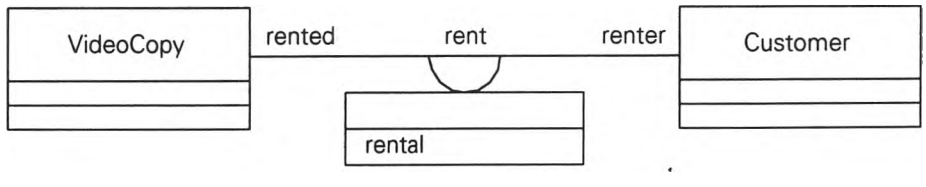
รูปที่ 3.17 สัญลักษณ์ของคลาส

การรับมรดกโดยปกติแล้วอาจแสดงเป็นลำดับชั้นความสัมพันธ์ของซุปเปอร์/ซับไทพ์ โดยซับไทพ์ถูกถ่ายทอดคุณสมบัติ(ทั้งแอททริบิวส์และบริการ) ของซุปเปอร์ไทพ์ดังแสดงในรูปที่ 3.18 เป็นการแสดงสัญญลักษณ์ของลำดับชั้นการรับมรดกของคลาส โดยเครื่องบินทางทหาร เครื่องบินของพลเรือน เครื่องบินเจ็ท และเครื่องบินพร็อพจะรับมรดกจากคลาสอากาศยานทุกชนิด

การเกี่ยวข้องสัมพันธ์กันแสดงถึงความคงอยู่ของวัตถุอื่น ๆ ความเกี่ยวข้องสัมพันธ์กันเหมือนกับคลาสนั้นได้รับคุณสมบัติด้วยตนเอง ดังนั้นสามารถสนับสนุนแอททริบิวส์ บริการ และความสัมพันธ์อื่น ๆ ความแตกต่างกันในสัญลักษณ์ทำให้จัดการด้วยวิธีการที่ต่างกันด้วย แต่ความคิดหลักคือความเกี่ยวข้องสัมพันธ์กันคือวิธีการที่สำคัญที่อธิบายว่าวัตถุหนึ่ง ๆ สามารถใช้ส่วนอื่นเพื่อให้งานสำเร็จได้อย่างไร ถ้าคลาส 2 คลาสมีความเกี่ยวเนื่องสัมพันธ์กัน ดังนั้นอินซแท็นซ์ของคลาสเหล่านี้ควรมีการเชื่อมต่อกัน การเชื่อมต่อกันระหว่างอินซแท็นซ์เหมือนกับความเกี่ยวเนื่องกันระหว่างคลาส บ่อยครั้งที่โมเดลเชื่อมต่อกับสเตท (State) และพฤติกรรม ดังแสดงดังรูปที่ 3.19 โดยที่ VideoCopy และ Customer มีความเกี่ยวเนื่องสัมพันธ์กันซึ่งมีการเชื่อมต่อกับแอททริบิวส์ Rental ด้วย



รูปที่ 3.18 สัญลักษณ์การรับมรดก

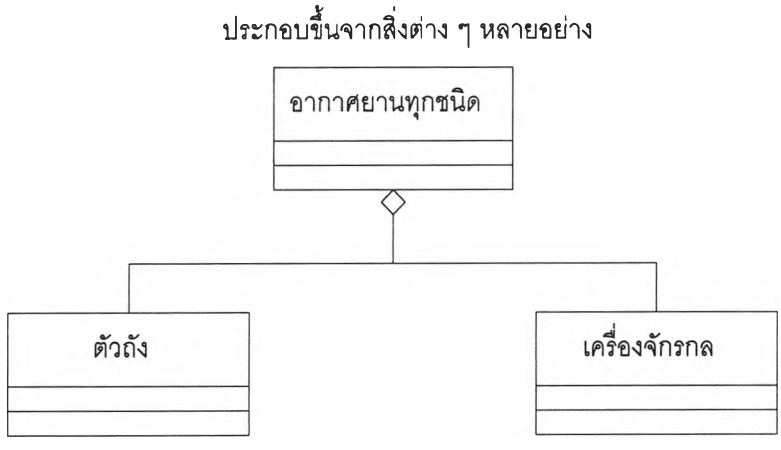


รูปที่ 3.19 สัญลักษณ์ของความเกี่ยวเนื่องสัมพันธ์กันโดยมีการเชื่อมโยงแอททริบิวส์กัน

ความเกี่ยวเนื่องสัมพันธ์กันนั้นมีการระบุการนับจำนวนด้วย โดยการนับจำนวนเป็นการแสดงว่าอินซแท็นซ์ที่สัมพันธ์กับอินซแท็นซ์ของคลาสอื่น ๆ เป็นจำนวนเท่าไร การนับจำนวนสามารถเป็น 0 หรือ 1 (ลูกกลมที่ว่างเปล่า), 1 (ไม่ทำเครื่องหมาย), 0 หรือมากมาย (ลูกกลมระบายทึบ) หรืออาจเป็นขอบเขตของตัวเลขก็ได้ การนับจำนวนมีความสำคัญสำหรับปัญหาหลัก เช่น ลูกค้าสามารถมีชื่อตามบัตรประชาชนได้เพียงชื่อเดียว

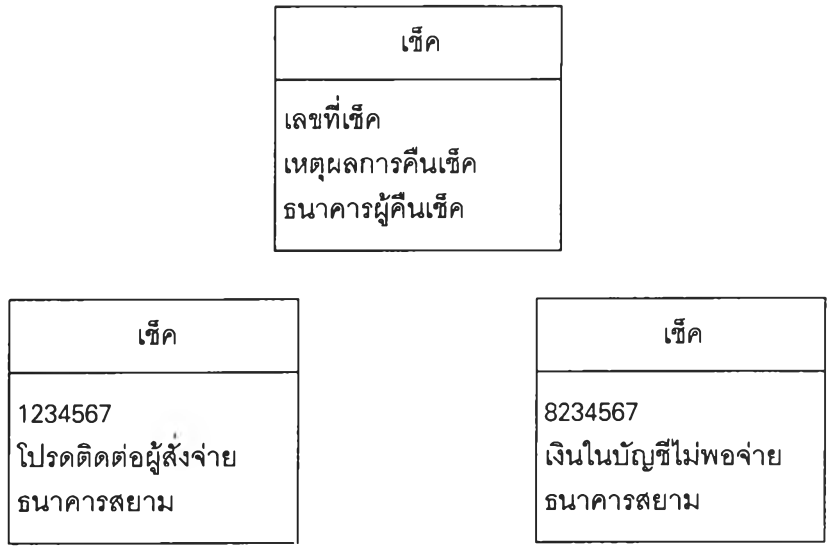
ความเกี่ยวเนื่องสัมพันธ์กันเหมือนกับการเชื่อมผ่าน โดยข้อความจะถูกผ่านให้เข้าถึงแอททริบิวส์ และบริการของส่วนประกอบของโมเดลอื่น ๆ

การไหลไปรวมกัน/ผลสรุปคือฟอร์มพิเศษของความสัมพันธ์ และแสดงในอีกรูปแบบหนึ่ง คือคลาสของวัตถุมีความสัมพันธ์เป็นลำดับชั้นในโมเดล ดังแสดงในรูปที่ 3.20 เมื่ออากาศยานทุกชนิด (aircraft) แยกเป็นส่วน ๆ คือตัวถังและเครื่องจักรกล ซึ่งคือส่วนประกอบต่าง ๆ แต่ละส่วนของอากาศยานนั้นเอง



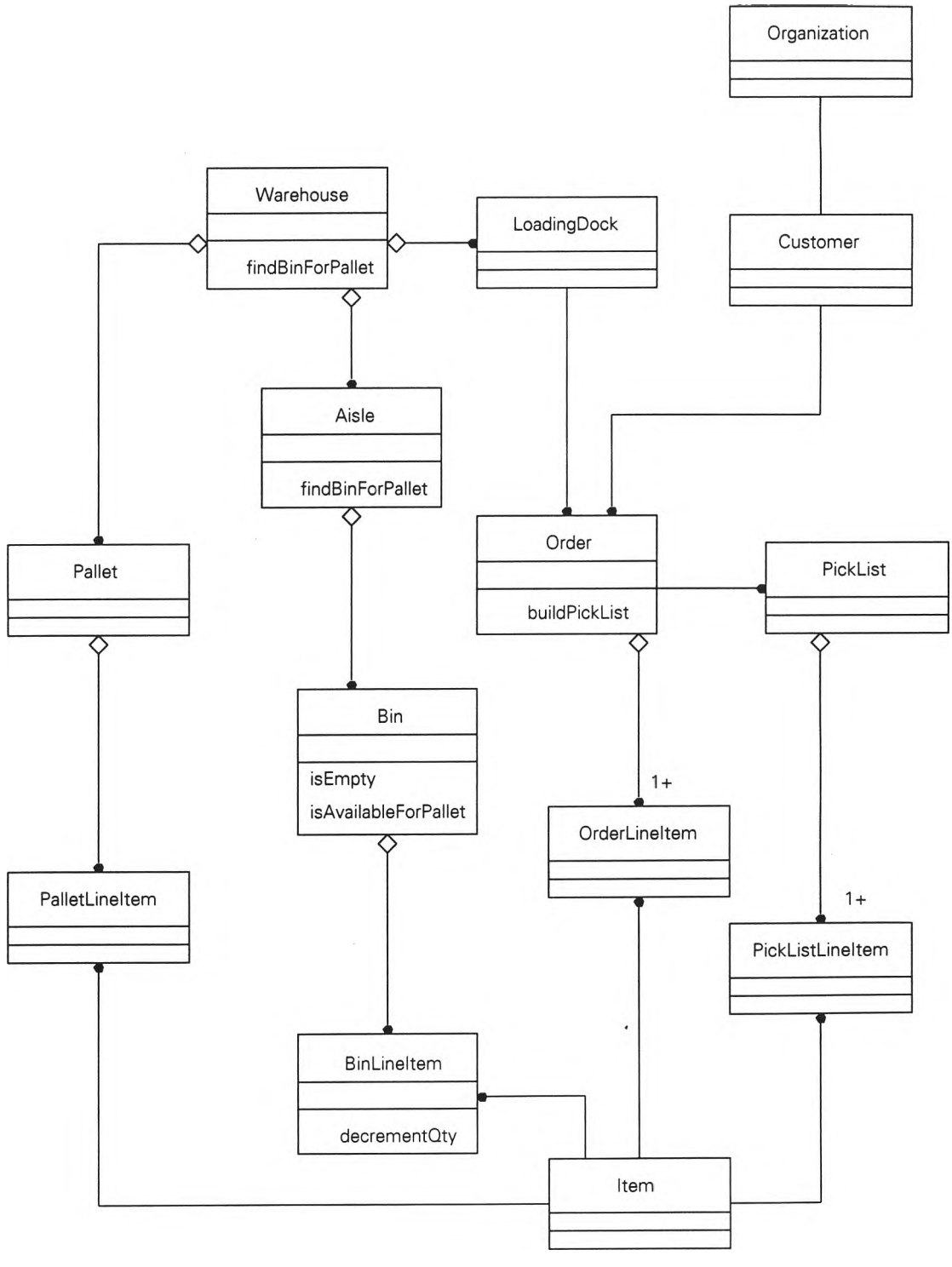
รูปที่ 3.20 สัญลักษณ์การไหลไปรวมกัน/ผลสรุป

แอททริบิวส์จะแสดงถึงโครงสร้างคุณสมบัติของคลาส จากตัวอย่างดังแสดงในรูปที่ 3.21 แสดงถึงคุณสมบัติคงที่ของคลาสเช็คคือ เลขที่เช็ค เหตุผลการคืนเช็ค และธนาคารผู้คืนเช็ค โดยแต่ละอินซแท็นซ์ของคลาสจะมีค่าของข้อมูลอยู่ด้วย เช่น “1234567”, “โปรดติดต่อผู้สั่งจ่าย”, “ธนาคารสยาม” ซึ่งเป็นแอททริบิวส์



รูปที่ 3.21 แอททริบิวส์ของคลาส

พฤติกรรมของคลาสซึ่งอาจเป็นบริการ หรือการปฏิบัติการ หรือความรับผิดชอบของคลาส ความสำคัญของคลาสวัตถุคือการห่อหุ้มข้อมูล และหน้าที่เข้าเป็นชุดเดียวกัน และหาประโยชน์จากการรับมรดก โพลิมอฟีซึม และการนำไปใช้ซ้ำอีกครั้ง การบริการจะห่อหุ้มฟังก์ชันเหล่านี้ไว้ โดยมีตัวอย่างของไดอะแกรมของคลาสของโมเดลเชิงวัตถุ ดังแสดงในรูปที่ 3.22 (James Rumbaugh, 1991)



รูปที่ 3.22 ไดอะแกรมของคลาสของโมเดลเชิงวัตถุในช่วงของการวิเคราะห์

2.3 การวิเคราะห์บทโครงการ (Analysis Scenarios)

บทโครงการคือการต่อเติมยูสเคสให้ละเอียดยิ่งขึ้น ยูสเคส (Ivar Jacobson, 1992) คือ กลุ่มข้อความของความต้องการในระดับบน โดยบทโครงการ (J.M. Spivey, 1988) จะเพิ่มรายละเอียดให้มากขึ้นและอธิบายปัจจัยซึ่งอาจมีผลต่อการเปลี่ยนแปลงพฤติกรรมของยูสเคส

บทโครงการสามารถกำหนดได้โดย

บทโครงการ = ยูสเคส + การสันนิษฐาน (เงื่อนไขในตอนแรก) + ผลลัพธ์ที่ได้

บทโครงการจะอธิบายพฤติกรรมของระบบอย่างละเอียด ซึ่งยูสเคสโมเดลและกลุ่มของบทโครงการทั้งหมดต่างก็เป็นความต้องการฟังก์ชันของระบบ ความต้องการเหล่านี้จะเริ่มอย่างเป็นทางการหรือไม่เป็นทางการให้พิจารณาตามความเหมาะสม

ยูสเคสคือข้อความที่ผู้ใช้ต้องการ แต่ยูสเคสไม่มีรายละเอียดเพียงพอสำหรับพัฒนาโมเดลการวิเคราะห์ บทโครงการจะแก้ไขยูสเคสและใช้พัฒนาไดอะแกรมการโต้ตอบเชิงวัตถุ (OIDs) หนึ่งยูสเคสสามารถเกิดบทโครงการได้หลายบท และบทโครงการที่ถูกผลักดันจากยูสเคสที่เหมือนกันสามารถสัมพันธ์กันในคลาสที่ต่างกัน

บทโครงการควรถูกสร้างจากบุคคลในทีมซึ่งมีความรู้ในเรื่องของเทคโนโลยีเชิงวัตถุและควรเริ่มในช่วงวิเคราะห์ หลังจากมียูสเคสแล้ว

ตัวอย่างดังแสดงในรูปที่ 3.23 เป็นบทโครงการจากระบบธนาคารซึ่งถูกผลักดันจากยูสเคส "ลูกค้าต้องการกู้เงิน"

Use Case 1: ลูกค้าต้องการกู้เงิน

Scenario 1.1: ธนาคารอนุญาตให้ลูกค้ากู้เงินได้

Assumptions:

- เป็นลูกค้าของธนาคาร
- ลูกค้าเป็นลูกจ้างบริษัทเอกชน
- เป็นลูกค้าชั้นดีของธนาคาร

Outcomes:

- อนุญาติให้ลูกค้ากู้เงินได้

รูปที่ 3.23 ตัวอย่างบทโครงการจากระบบเงินกู้

Use Case 1: ลูกค้าต้องการกู้เงิน

Scenario 1.2: ธนาคารปฏิเสธการกู้เงิน

Assumptions:

- ไม่ได้เป็นลูกค้าของธนาคาร
- ลูกค้าเป็นลูกจ้างบริษัทเอกชน
- เป็นลูกค้าที่เครดิตไม่ดีเท่าที่ควร

Outcomes:

- ไม่อนุญาติให้ลูกค้ากู้เงิน

รูปที่ 3.23 ตัวอย่างบทโครงการจากระบบเงินกู้ (ต่อ)

2.4 การวิเคราะห์รายละเอียดของคลาส (Analysis Class Description)

การวิเคราะห์รายละเอียดของคลาส คือบทสรุปของข้อมูลทั้งหมดเกี่ยวกับคลาสในระดับการวิเคราะห์ หากมีการเพิ่มความรับผิดชอบใหม่ให้กับคลาสในโมเดลเชิงวัตถุ ดังนั้นในรายละเอียดของคลาสควรจะมีข้อมูลที่เพิ่มใหม่นี้ด้วยเช่นกัน

การวิเคราะห์รายละเอียดของคลาสเพื่อเก็บข้อมูลของคลาส เช่น รายละเอียดสั้น ๆ, แอททริบิวส์, ความรับผิดชอบ หรือรายละเอียดอื่น ๆ ที่เกิดขึ้นในช่วงวิเคราะห์ และเพื่อเก็บข้อมูลที่ไม่ได้อยู่ในไดอะแกรมอื่น ๆ ซึ่งอาจเป็นมาตรฐาน หรือความสัมพันธ์ หรืออื่น ๆ

นักวิเคราะห์ผู้เป็นเจ้าของคลาส ควรรับผิดชอบในการปรับปรุงรายละเอียดของคลาสนั้น โดยที่รายละเอียดของคลาสควรเปลี่ยนแปลงทันทีเมื่อโมเดลเชิงวัตถุและสถานะโมเดลถูกแก้ไขรายละเอียดของคลาส จัดการจากการวิเคราะห์ออกแบบและช่วงการนำไปใช้ โดยช่วงการนำไปใช้สามารถนำไปใช้ได้ เพราะว่ารายละเอียดของคลาสเป็นที่รวบรวมข้อมูลใหม่ของคลาส ตัวอย่างรายละเอียดของคลาสดังแสดงในรูปที่ 3.24 (Rebecca Wirfs-Brock, 1990)

Class name	: BankAccount
Definition	: บัญชีธนาคารคือการยอมรับระหว่างธนาคารและลูกค้า ซึ่ง ลูกค้าสามารถฝากเงินกับธนาคาร และสามารถถอนเงินได้ตาม สิทธิที่มีผ่านบัญชีที่มีอยู่กับธนาคาร
Operations	: - ถอนเงิน - โอนเงิน - ถามยอด
Key attributes	: - รายละเอียดของลูกค้า - ยอดคงเหลือ
Relations	: ซับคลาส: บัญชีออมทรัพย์, บัญชีกระแสรายวัน เกี่ยวเนื่องสัมพันธ์กับ: เจ้าของบัญชี
States	: - Open - Active - Closed
Documentation	: กฎเกณฑ์การถอนเงินควรขึ้นกับแต่ละประเภทบัญชี

รูปที่ 3.24 รายละเอียดของคลาส

3) การออกแบบ (System Design)

การออกแบบเชิงวัตถุคือกระบวนการในการกำหนดสถาปัตยกรรมที่ใช้ในระบบและกำหนดแต่ละคลาสที่จำเป็น เพื่อนำไปใช้ในการผลิตซอฟต์แวร์ ซึ่งอาจเกี่ยวข้องกับการเลือกทั้งส่วนกลางและภายในในการวางแผนการนำไปใช้โดยยึดจากข้อบังคับ, ความต้องการที่ไม่เกี่ยวกับฟังก์ชัน และทางเลือกที่เป็นไปได้

ระหว่างการวิเคราะห์อาจมีการรวมความสนใจไปยังปัญหาหลักของวัตถุ อย่างไรก็ตามในช่วงออกแบบควรรวมความสนใจไปสู่วิธีการเชิงวัตถุ ในระหว่างการออกแบบจะเน้นหนักในเรื่องของการกำหนดวิธีการแก้ปัญหา

ปัญหาหลักที่พบระหว่างวิเคราะห์จะถูกแก้ไขในช่วงการออกแบบ วัตถุใหม่และคลาสจะถูกสร้างระหว่างการออกแบบ

ในภาพรวมของการออกแบบ

- การออกแบบเชิงวัตถุจะเปลี่ยนรูปโมเดลวิเคราะห์ไปสู่การออกแบบวิธีการ
- กำหนดสถาปัตยกรรมของซอฟต์แวร์จะทำระหว่างการออกแบบ
- แอททริบิวส์, การปฏิบัติการและอัลกอริทึมถูกกำหนดระหว่างการออกแบบ

ในช่วงการออกแบบประกอบด้ วย

- ออกแบบโมเดลเชิงวัตถุ (Design Object Model)
- ออกแบบไดอะแกรมการโต้ตอบเชิงวัตถุ (Design Object Interaction Diagram)
- ออกแบบโมเดลสถานะ (Design State Models)

3.1 การออกแบบโมเดลเชิงวัตถุ (Design Object Model)

การออกแบบโมเดลเชิงวัตถุคือการแสดงวัตถุซอฟต์แวร์ ซึ่งประกอบด้วยการนำไปใช้ของระบบหรือโปรแกรม โมเดลคงที่จะประกอบด้วยการออกแบบคลาสวัตถุ, แอททริบิวส์, ความรับผิดชอบ, การปฏิบัติการ และความสัมพันธ์ ซึ่งเป็นไปได้ทั้งแบบความเกี่ยวเนื่องสัมพันธ์กัน การรวมกัน และการรับมรดก

โมเดลเชิงวัตถุเป็นวิธีการเชิงวัตถุที่สำคัญในการเข้าถึงปัญหา ถ้ารวมความสนใจไปที่โมเดลเชิงวัตถุระหว่างการออกแบบคือโครงสร้างของระบบซอฟต์แวร์

การออกแบบโมเดลเชิงวัตถุคือการพัฒนาในช่วงแรกของการออกแบบ เทคนิคของการออกแบบโมเดลเชิงวัตถุวิธีที่ดีที่สุดคือพัฒนาโมเดลเชิงวัตถุในช่วงภารออกแบบ โดยเริ่มจากโมเดลเชิงวัตถุจากช่วงวิเคราะห์โดยขยายส่วนการออกแบบโมเดลเข้าไป ซึ่งมีขั้นตอนคือ

1. เริ่มจากการลอกแบบของโมเดลเชิงวัตถุในช่วงการวิเคราะห์
2. เพิ่มคลาสใหม่จากวิธีการโดเมน เช่น ภาพและคลาสที่ให้ความสะดวกต่าง ๆ
3. ตรวจสอบและกำหนดความรับผิดชอบจากการออกแบบ OIDs และโมเดลสถานะในเรื่องของ
   - แอททริบิวส์
   - การปฏิบัติการ หรือเมธอด
4. สำหรับทุกคลาสและความสัมพันธ์ในโมเดล พิจารณาการปฏิบัติการโดยผลของอินซแท็นซ์อาจเป็นการสร้างหรือลบ

5. โมเดลเชิงวัตถุควรทำให้เหมาะสมกับการปฏิบัติงาน และการนำกลับมาใช้อีกครั้ง แนะนำว่าความสัมพันธ์ใหม่ หรือการแก้ไขสิ่งเติมที่มีอยู่ควรเหมาะสมกับความต้องการที่ไม่เกี่ยวกับฟังก์ชัน ในเรื่องของความสัมพันธ์ควรเป็นลักษณะถาวรของคลาสสู่การเชื่อมโยงแบบชั่วคราวของวัตถุ ซึ่งจะผ่านอาร์กิวเมนต์ในเมธอด หรือสร้างเหมือนเป็นวัตถุชั่วคราวในเมธอด

6. ตัดหรือยุบโครงสร้างซึ่งแสดงข้อมูลที่ไม่จำเป็น

7. เปลี่ยนรูปของโครงสร้างการรับมรดกไปสู่ภาวะที่เป็นตัวแทน เมื่ออยู่ในลำดับการแยกข้อมูลของการออกแบบ ถ้าเป็นซับคลาสไม่ควรผ่านการทำหน้าที่แทน (Is-A), เปลี่ยน Is-A ไปใช้หรือมี (Has)

8. กำหนดว่าคลาสไหนควรมีความคงอยู่

9. กำหนดความสามารถในการเข้าถึงของการปฏิบัติการ

- ของสาธารณะ (Public) : ทุกคลาสสามารถเรียกใช้การปฏิบัติการได้
- เชิงป้องกัน (Protected) : ซับคลาสเท่านั้นที่สามารถเรียกใช้การปฏิบัติการ
- เฉพาะตัว (Private) : ไม่มีคลาสไหนที่จะเรียกใช้การปฏิบัติการ

10. กำหนดรายละเอียดการนำไปใช้ของความสัมพันธ์

11. กำหนดความเป็นเจ้าของคลาส เช่นใครสร้างคลาส ใครสามารถลบคลาส

12. สร้างชนิดของการอ้างอิงถึงกัน ซึ่งจะใช้โดยแอททริบิวส์ในการนำไปใช้ในความสัมพันธ์และการรวมกัน

- โดยการอ้างอิง : ใช้พอยน์เตอร์ หรือการอ้างถึง
- โดยค่าข้อมูล : มีวัตถุ
- โดยคีย์ : มีคีย์ที่สามารถแปลงข้อมูลสู่การอ้างอิง

13. กำหนดขอบเขตของความสัมพันธ์

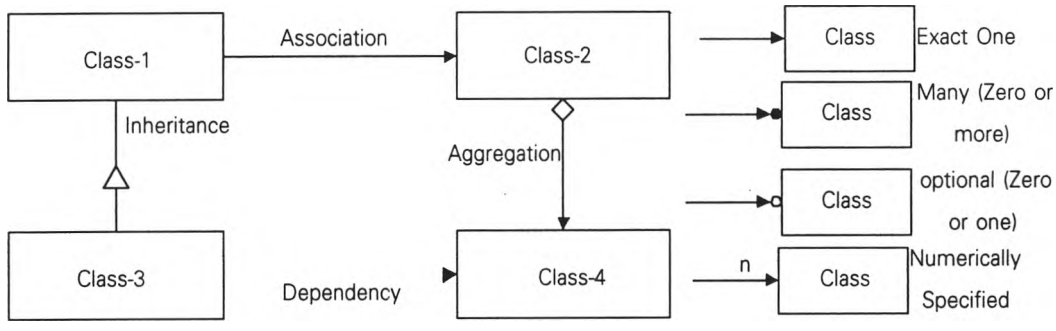
- ขอบเขตของการปฏิบัติการ (อ้างอิงแบบไดนามิก) : เมื่อวัตถุ A ได้อ้างอิงไปยังวัตถุ B โดยผ่านพารามิเตอร์เมธอด หรือโดยตัวแปรภายในของเมธอด
- ขอบเขตของคลาส (อ้างอิงแบบคงอยู่) : อ้างอิงจากวัตถุ A ไปยังวัตถุ B จำเป็นที่จะคงอยู่ระหว่างเรียกเมธอด

14. กำหนดหน้าที่ของการปฏิบัติ และแอททริบิวส์ของคลาส

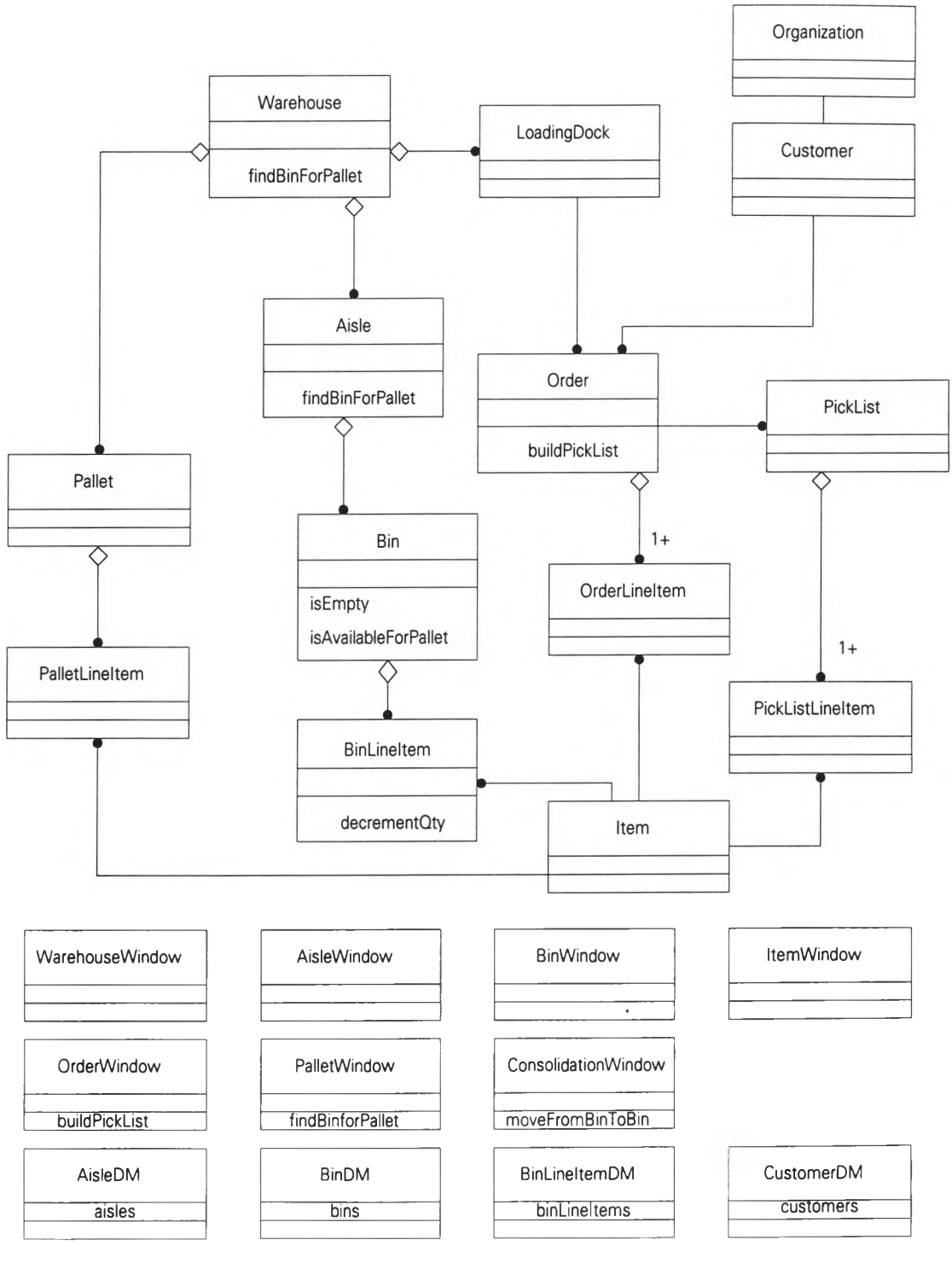
15. กำหนดคุณสมบัติพิเศษของคลาสและการปฏิบัติการ ซึ่งบางครั้งคุณสมบัติพิเศษขึ้นอยู่กับการใช้ภาษา

16. กำหนดประเภทของทุกแอททริบิวส์

สัญลักษณ์ที่ใช้ในการออกแบบโมเดลเชิงวัตถุ จะเพิ่มขึ้นจากสัญลักษณ์ที่ใช้ในการวิเคราะห์ โดยมีส่วนที่เพิ่มขึ้นคือดังแสดงในรูปที่ 3.25 และมีตัวอย่างของการออกแบบโมเดลเชิงวัตถุดังแสดงในรูปที่ 3.26 (James Rumbaugh, 1991)



รูปที่ 3.25 สัญลักษณ์การออกแบบเชิงวัตถุ



รูปที่ 3.26 ตัวอย่างการออกแบบโมเดลเชิงวัตถุ

3.2 การออกแบบไดอะแกรมการโต้ตอบเชิงวัตถุ (Design Object Interaction Diagram)

การออกแบบไดอะแกรมการโต้ตอบเชิงวัตถุ (OIDs) คือการแสดงโดยเส้นจากการวิเคราะห์บทโครงการ โดยแสดงในส่วนของการโต้ตอบระหว่างความเป็นจริงหรือวัตถุที่วิเคราะห์ ซึ่งการออกแบบ OIDs จะแสดงส่วนเปลี่ยนแปลงของบทโครงการโดยแสดงว่าวัตถุมีส่วนเกี่ยวข้องอย่างไรในบทโครงการ มีส่วนร่วมในลำดับใดเพื่อให้ได้ผลลัพธ์ตามความต้องการ การวิเคราะห์บทโครงการจะถูกผลักดันโดยตรงจากยูสเคส เมื่อออกแบบ OIDs เรียบร้อยแล้วทำให้เกิดการเชื่อมต่อระหว่างความต้องการและการออกแบบโมเดลเชิงวัตถุ ซึ่งการออกแบบ OIDs เป็นการแสดงส่วนเปลี่ยนแปรของบทโครงการโดยใช้เส้นการโต้ตอบ

การออกแบบ OIDs เป็นการเตรียมภาพของวัตถุหรืออินซแท็นซ์ของคลาสที่กำหนดในการออกแบบโมเดลเชิงวัตถุว่าโต้ตอบในลำดับอย่างไร ทำตามบทโครงการซึ่งคือความต้องการในระบบ ภาพที่แสดงโดยเส้นจะแสดงผังทำงานจากจุดเริ่มต้นถึงจุดสุดท้ายในรูปแบบง่าย ๆ การออกแบบ OIDs เป็นการให้ความรับผิดชอบซึ่งจำเป็นต้องทำตามบทโครงการ และถูกกำหนดความรับผิดชอบไปยังคลาสเหมือนกับการออกแบบ OIDs จะเชื่อมบทโครงการไปยังการออกแบบโมเดลเชิงวัตถุ ซึ่งการออกแบบ OIDs จะใช้ผลักดันโมเดลเชิงวัตถุหรือตรวจสอบโมเดลเชิงวัตถุที่มีอยู่แล้ว

ความยืดหยุ่นและความสามารถในการขยายของระบบขึ้นอยู่กับความสามารถในการจัดสรรพฤติกรรมในส่วนของการเชื่อมต่อกันและการรวมกัน การจัดสรรที่ดีของพฤติกรรมสามารถนำกลับมาใช้ซ้ำได้อีกและสับเปลี่ยนกันของวัตถุ จุดเด่นของการออกแบบ OIDs คือการแสดงการควบคุมจากจุดเริ่มต้นสู่จุดสุดท้ายและผังข้อมูลระหว่างวัตถุ ซึ่งอยู่ภายใต้สถาปัตยกรรมของระบบ

ภาพจำลองกับการออกแบบ OIDs คือการผลักดันที่มีประสิทธิภาพของกระบวนการพัฒนา ซึ่งเป็นเครื่องมือหลักในการจัดสรรความรับผิดชอบของวัตถุ ค้นพบปัญหา พิจารณาการออกแบบที่มีอยู่

ไดนามิกของสถาปัตยกรรมระบบสามารถใช้ออกแบบ OIDs ได้อย่างดี ความรู้ในส่วนของวัตถุโต้ตอบกับสถาปัตยกรรมที่มีอยู่อย่างไร และภาพจำลองแบบไดนามิกกับการออกแบบ OIDs คือขั้นตอนที่สำคัญที่สุด เพราะจะกระทบโดยตรงกับการนำไปใช้ การออกแบบอื่น ๆ ควรถูกแก้ไขอีกครั้งในระดับการออกแบบ OIDs

สรุปแล้วการออกแบบ OIDs หมายถึง

- การใช้ภาพเพื่อบรรยายการออกแบบพฤติกรรมและความรับผิดชอบของวัตถุ
- ค้นพบ แสดง และเข้าใจฟังก์ชันที่ทำสำเร็จในแต่ละวัตถุ

- ภาพการกระจายจากศูนย์กลางของความรับผิดชอบของระบบระหว่างวัตถุ
- ระบุ ประยุกต์ และแสดงแบบสำหรับโครงสร้างการออกแบบ

การพัฒนา OIDs ในช่วงการออกแบบคืองานของสถาปนิกระบบ นักออกแบบ และนักพัฒนา ควรนำทีมโดยสถาปนิกระบบ นักเขียนโปรแกรมควรมีส่วนร่วมด้วยในขั้นตอนนี้ เพื่อให้เข้าใจข้อบังคับของระบบ สิ่งแวดล้อมของระบบ และความจำกัดของภาษา ปัจจัยเหล่านี้จะต้องทำให้เห็นในช่วงการออกแบบ OIDs

การออกแบบ OIDs ควรจะพัฒนาพร้อม ๆ กับการออกแบบโมเดลเชิงวัตถุ การออกแบบ OIDs ควรใช้เครื่องมือผลักดันการพัฒนาโครงสร้างของระบบ โดยการวิเคราะห์โมเดลเชิงวัตถุจะเตรียมพื้นฐานในสิ่งเหล่านี้สำหรับช่วงการออกแบบ เมื่อออกแบบโมเดลเชิงวัตถุก็เป็นเวลาสำหรับพัฒนาการออกแบบ OIDs สำหรับวัตถุของระบบเช่นกัน ในการออกแบบกระบวนการโมเดล OIDs ทำให้วัตถุถูกสร้างขึ้นมากมาย

ในกระบวนการวนซ้ำและการเพิ่มส่วน โมเดลการออกแบบ โดยเฉพาะการออกแบบโมเดลเชิงวัตถุควรปฏิบัติตามวงจรการพัฒนาอื่น ๆ แม้แต่ในเรื่องของการนำไปใช้ การออกแบบ OIDs ถูกอ้างอิงหรือแก้ไขบ่อย ๆ สำหรับการออกแบบใหม่ หรือตามแนวคิดของสถาปัตยกรรม

การออกแบบ OIDs จะพิจารณาภายใต้สถาปัตยกรรม กรอบการทำงาน และข้อบังคับของระบบ โดยปกติส่วนประกอบของการออกแบบ OIDs มีรายละเอียดมากมาย ตั้งแต่ใช้เป็นส่วนที่ระบุการเขียนโปรแกรม เมื่อมีสถาปัตยกรรมแล้วการออกแบบ OIDs จะใช้เพื่อกำหนดความรับผิดชอบของคลาส การพัฒนาการออกแบบ OIDs จะเกี่ยวข้องกับ

- เริ่มด้วยยูสเคส และการวิเคราะห์บทโครงการ
- หาความรับผิดชอบของวัตถุในการออกแบบโมเดลเชิงวัตถุ ซึ่งควรตรงกับวัตถุออกแบบอาจมีการทำต่อกับความรับผิดชอบของวัตถุบางวัตถุโดยเพิ่มบางส่วนจากสถาปัตยกรรมระบบ
- วิเคราะห์การโต้ตอบของวัตถุออกแบบ ตรวจสอบความรับผิดชอบของการขึ้นกับวัตถุอื่น ๆ เช่น ถ้าความรับผิดชอบของวัตถุคือการทำกิจกรรม แต่ไม่ควบคุมความรู้ทั้งหมดที่จำเป็นสำหรับความสำเร็จของกิจกรรม ซึ่งต้องอาศัยความช่วยเหลือของวัตถุที่ถูกกำหนดในคลาสอื่นให้ควบคุมความรู้นั้น โดยทั้งสองวัตถุมีความช่วยเหลือกันซึ่งเป็นความสัมพันธ์โดยตรง หรือเห็นได้ชัดว่าต้องขึ้นกับวัตถุอื่น ๆ ซึ่งจะถูกกำหนดในการออกแบบโมเดลเชิงวัตถุ
- กำหนดความช่วยเหลือกันโดยตั้งคำถามสำหรับความรับผิดชอบของทุกคลาส เช่น วัตถุของคลาสมีความสามารถสำเร็จในความรับผิดชอบด้วยตัวเอง ? ถ้าไม่ใช่อะไรคือสิ่งที่จำเป็น ?

จากคลาสอื่นจะได้สิ่งที่จำเป็นหรือไม่ ? สามารถออกแบบให้แต่ละความรับผิดชอบสามารถช่วยเหลือกันระหว่างคลาสได้

- ตรวจสอบพารามิเตอร์ซึ่งถูกกำหนดสำหรับข่าวสารที่รวมเมธอดการคืนค่า ดังนั้นผังข้อมูลและหน่วยเก็บข้อมูลควรเขียนออกมาอย่างชัดเจนในเอกสาร

- เพิ่มข้อมูลการควบคุม กำหนดในสัญลักษณ์ส่วนย่อย ๆ ในการออกแบบ OIDs ของระบบว่าจะแสดงเมื่อไร ที่ไหน และพฤติกรรมของวัตถุจะปฏิบัติการอย่างไร หรือรอการตอบรับ

สัญญลักษณ์ที่ใช้ในการออกแบบ OIDs สามารถอธิบายได้ดังนี้

1. วัตถุ : วัตถุหรืออินซแท็นซ์ที่เกี่ยวข้องในบทโครงการนี้ถูกแสดงในไดอะแกรม แต่ละวัตถุถูกระบุโดยชื่อ โดยปกติชื่อควรอยู่บนสุดของเส้นแบ่งเวลาสำหรับวัตถุ

2. เส้นแบ่งเวลา :เส้นแบ่งเวลาคือเส้นตรงในแนวตั้ง ซึ่งจะแสดงแต่ละวัตถุโดยใช้เหมือนเป็นแหล่งที่มาและเป็นที่หมายของลำดับเวลาก่อนหลังของข่าวสารที่ส่งระหว่างวัตถุ

3. ข่าวสาร : ข่าวสารจะแสดงถึงการโต้ตอบของวัตถุ ทุก ๆ ข่าวสารในการออกแบบ OIDs มีหนึ่งผู้ส่งและหนึ่งผู้รับ บางครั้งผู้ส่งสามารถเว้นการส่งได้ถ้าวัตถุมากกว่าหนึ่งคลาสจะส่งข่าวสาร ข่าวสารสามารถขอข้อมูลจากวัตถุ หรือเปลี่ยนสถานะก็ได้ และข่าวสารสามารถส่งถึงตัวเองได้คือวัตถุปัจจุบันซึ่งจะแสดงโดยข่าวสารที่ด้นทางและปลายทางซึ่งก็คือวัตถุปัจจุบัน ข่าวสารสามารถเป็นไปได้ทั้งซิงโครนัสและอซิงโครนัส

- ข่าวสารแบบซิงโครนัส : ผู้ส่งข่าวสารแบบซิงโครนัสคาดหวังว่าการแสดงผลด้วยเส้นข่าวสารกลับมาพร้อมพารามิเตอร์ ผู้ส่งข่าวสารต้องคอยผลกลับมาก่อนที่จะทำกระบวนการอื่น

- ข่าวสารแบบอซิงโครนัส : ผู้ส่งข่าวสารแบบอซิงโครนัสจะไม่คอยข่าวสารที่ส่งกลับมาก่อนทำกระบวนการอื่น ซึ่งเป็นข้อแตกต่างระหว่างข่าวสารแบ่บซิงโครนัสและอซิงโครนัส

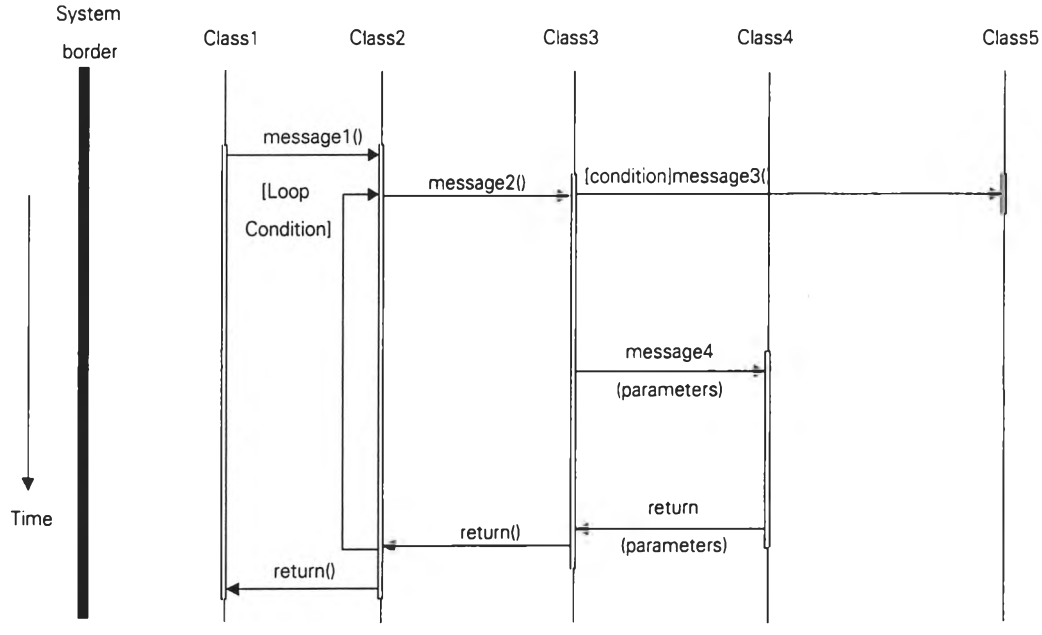
4. ข่าวสารพารามิเตอร์ : ข่าวสารสามารถนำพารามิเตอร์ไปในการโต้ตอบได้

5. เงื่อนไข : จะแสดงถึงเงื่อนไขในข่าวสารของวัตถุผู้ส่ง ถ้าเงื่อนไขเป็นจริงข่าวสารจะถูกส่งทันที

6. ลูป (loop) : ในลำดับของการทำงานซ้ำ ๆ กัน ลูปสามารถผูกติดกับเส้นเวลาของวัตถุ โดยลูปมีความสัมพันธ์กับเงื่อนไขของลูป ซึ่งเงื่อนไขจะอยู่ภายใต้การทำงานของลูปอีกครั้ง

7. การรวมความสนใจไปที่การควบคุมเพื่อแสดงว่าวัตถุกำลังปฏิบัติการอยู่ (active) ซึ่งอยู่ในสถานะของการกระทำการ หรือในสถานะที่รอข่าวสารกลับ โดยจะใช้รูปสี่เหลี่ยมผืนผ้าแบบยาวซึ่งเกี่ยวข้องกับเส้นแบ่งเวลาวัตถุ ในระบบที่ซับซ้อนมีการประมวลผลและขอบเขตของระบบย่อย เหมือนเป็นการทำงานหลายงาน ซึ่งสามารถเสนอได้ในการออกแบบ OIDs การออก

แบบ OIDs สามารถแสดงการประมวลผลและระบบย่อยของวัตถุ การติดต่อกันในช่วงเวลาหนึ่งของวัตถุ รวมไปถึงกิจกรรมที่เกิดขึ้น ดังแสดงในรูปที่ 3.27 (Ivar Jacobson, 1992)



รูปที่ 3.27 แบบฟอร์มของการออกแบบไดอะแกรมการโต้ตอบเชิงวัตถุ

ตัวอย่างการออกแบบไดอะแกรมการโต้ตอบเชิงวัตถุดังแสดงในรูปที่ 3.29 ซึ่งเป็นสถานการณ์ของส่วนช่วยเหลือลูกค้าของบริษัทเช่าซื้อรถยนต์ โดยรับโทรศัพท์จากลูกค้าซึ่งมีความต้องการแจ้งให้บริษัททราบถึงการเปลี่ยนแปลงที่อยู่ โดยมีรายละเอียดบทโครงการที่ 1.2 ดังแสดงในรูปที่ 3.28

Analysis Scenario 1.2: ลูกค้าต้องการเปลี่ยนแปลงที่อยู่

Assumptions:

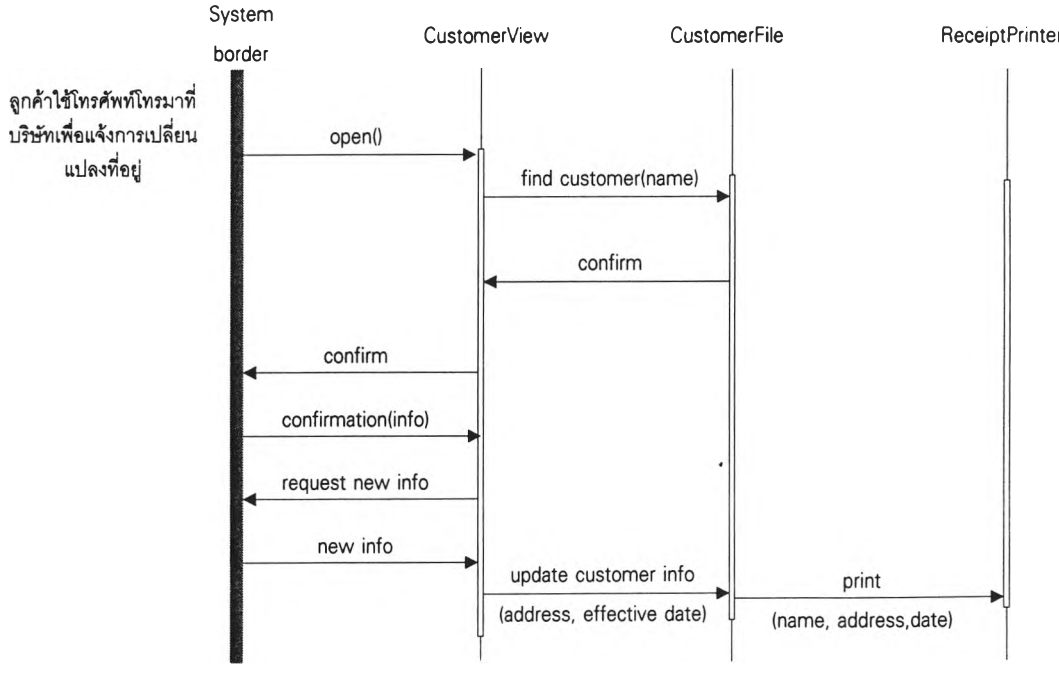
- เป็นลูกค้าของบริษัทเช่าซื้อรถยนต์
- ที่อยู่ต้องเปลี่ยนแปลงทันที

Outcomes:

- การเปลี่ยนแปลงที่อยู่เสร็จเรียบร้อย

Description: การเปลี่ยนแปลงที่อยู่ลูกค้าจะเกี่ยวข้องกับไฟล์ลูกค้า และต้องออกใบแจ้งการเปลี่ยนแปลงที่อยู่ไปยังลูกค้าและฝ่ายขายด้วย

รูปที่ 3.28 ตัวอย่างบทโครงการของลูกค้าต้องการเปลี่ยนแปลงที่อยู่



รูปที่ 3.29 ไดอะแกรมการโต้ตอบเชิงวัตถุในกรณีที่ลูกค้าต้องการเปลี่ยนที่อยู่

3.3 การออกแบบโมเดลสถานะ (Design State Models)

โมเดลสถานะจะใช้ในการออกแบบเชิงวัตถุเพื่ออธิบายวงจรชีวิตของคลาส ซึ่งอธิบายถึงสถานะของคลาสที่ได้มา และช่วงหัวเลี้ยวหัวต่อที่เป็นสาเหตุของการเปลี่ยนสถานะ การเปลี่ยนสถานะจะแสดงถึงปัจจัยภายนอกหรือเหตุการณ์ที่ทำให้สถานะวัตถุเปลี่ยนไป โมเดลสถานะเป็นการแสดงโดยไดอะแกรมสถานะหรือตารางการเปลี่ยนสถานะ

โมเดลสถานะเป็นการแสดงวงจรชีวิตของวัตถุในสัญญลักษณ์รูปภาพ หรือแบบฟอร์มตารางซึ่งเป็นการให้ภาพรวมของการเปลี่ยนแปลงของวัตถุกับเหตุการณ์ภายนอกอย่างไร โดยปราศจากรายละเอียดอื่น ๆ โมเดลสถานะจะง่ายแก่การพัฒนาและง่ายแก่การเข้าใจหากเปรียบเทียบกับการอธิบายโดยข้อความ

การออกแบบ OIDs จะเตรียมเงื่อนไขปัญหาสำหรับสร้างการออกแบบโมเดลสถานะ เมื่อข่าวสารถูกส่งจากวัตถุหนึ่งไปสู่วัตถุอื่น สองสิ่งจะเกิดขึ้น โดยสิ่งแรกคือวัตถุที่ส่งข่าวสารจะปฏิบัติการในสถานะนั้นเพื่อส่งข่าวสารได้สำเร็จ สองคือกรณีจัดรูปแบบและไปถึงวัตถุที่รับข่าวสารนั้น โดยวัตถุที่รับข่าวสารจะปฏิบัติการ และถ้าเป็นไปได้จะมีการเปลี่ยนสถานะ

เทคนิคในการเลือกคลาสสู่โมเดลสถานะคือ

1. ควรมองหาคลาสที่น่าสนใจ หรือสนใจในวงจรชีวิต เช่นในระบบการเช่า มีคลาส “Loan” เป็นคลาสที่ควรให้ความสนใจ

2. ระบุให้เห็นว่าวัตถุเข้ามาสู่ระบบได้อย่างไร ซึ่งจะเป็นการเปลี่ยนสถานะมาสู่สถานะแรกเริ่ม

3. จากสถานะแรกเริ่ม เพิ่มการเปลี่ยนแปลงทั้งหมดซึ่งจะเกิดขึ้นและสถานะใหม่ที่ควรจะเป็น

4. วนซ้ำในกระบวนการเพื่อให้ได้สถานะทั้งหมด

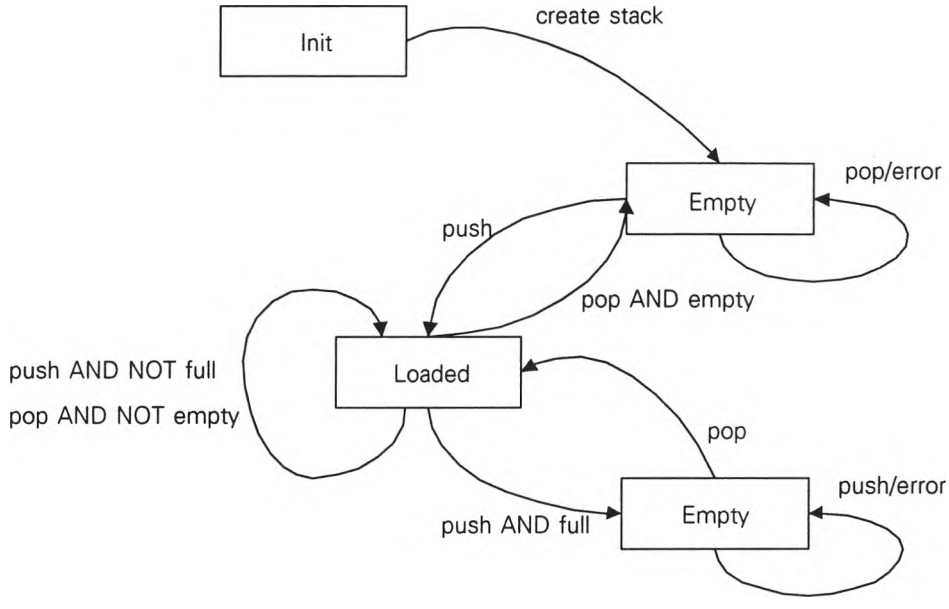
5. ควรตรวจสอบเพื่อเพิ่มการเปลี่ยนแปลง กรณีที่วนซ้ำกับสถานะเดิม (ลูป)

ในระดับการออกแบบ โมเดลสถานะควรกำหนดโดยสถาปนิกระบบ และนักออกแบบถ้าเป็นไปได้นักเขียนโปรแกรมควรมีส่วนร่วม เพื่อให้แน่ใจว่าการออกแบบสามารถนำไปใช้ได้

การออกแบบโมเดลสถานะโดยปกติจะปฏิบัติการตามแบบวนทำซ้ำ กับการพัฒนาของการออกแบบโมเดลเชิงวัตถุ และการออกแบบ OIDs

สัญลักษณ์ที่ใช้ในโมเดลสถานะโดยสถานะของคลาสอาจใช้รูปวงกลม หรือรูปสี่เหลี่ยม ส่วนการเปลี่ยนแปลงอาจแสดงโดยใช้เส้นเชื่อมระหว่างสถานะ โดยการเปลี่ยนแปลงควรมีคำ

อธิบายด้วย เช่น สถานะเพื่ออธิบายการทำงานของสแต็คดังแสดงในรูปที่ 3.30 และตัวอย่างตารางการเปลี่ยนสถานะของคลาส Account ดังแสดงในรูปที่ 3.31 (Ivar Jacobson, 1991)



รูปที่ 3.30 ไดอะแกรมสถานะเพื่ออธิบายสแต็ค

State	Event	Next State
Open	Clerk Closes	Closed
Open	Customer Closes	Closed
Open	Customer Deposits	Active
Active	Customer Closes	Closed
Active	Customer Deposits	Active

รูปที่ 3.31 ตารางการเปลี่ยนสถานะของคลาส Account

4) การนำไปใช้ให้เกิดผลตามความต้องการ (Implementation)

ในขั้นตอนนี้เกี่ยวข้องกับการเปลี่ยนรูปแบบของเอกสารจากการออกแบบซึ่งเป็นรายละเอียดของแผนการเพื่อแก้ปัญหา สู่โปรแกรมที่ผู้เขียนโปรแกรมเขียนขึ้นด้วยภาษาระดับสูง (Source Code) และผลัตภัณฑ์อื่น ๆ ซึ่งในการพัฒนาเชิงวัตถุจะเกี่ยวข้องกับการนำคลาสที่สร้างใหม่ไปใช้ในโปรแกรมภาษาเชิงวัตถุ โดยมีการกำหนดคลาสแล้วในการออกแบบ แม้ว่ารายละเอียดการออกแบบจะรวมความสนใจไปที่คุณสมบัติภายนอก แต่ก็รวมสถานะภายใน, การปฏิบัติการ (รายละเอียดของเมธอด) และการแสดง หากการออกแบบมีเอกสารดังกล่าวไม่ครบการนำไปใช้ให้เกิดผลตามความต้องการอาจใช้เฉพาะเอกสารการออกแบบโมเดลเชิงวัตถุ และเอกสารการออกแบบไดอะแกรมการโต้ตอบเชิงวัตถุ

5) การทดสอบ (Testing)

การทดสอบในวิธีการเชิงวัตถุมีความแตกต่างเล็กน้อย กับการทดสอบซอฟต์แวร์ที่ใช้ตั้งแต่แรกเริ่ม คือการทดสอบไม่ใช่ขั้นตอนที่เกิดขึ้นเฉพาะในช่วงเวลาที่การพัฒนาเรียบร้อยเท่านั้น

ในโครงการเชิงวัตถุการทดสอบจะไม่มีความต่อเนื่อง ขึ้นอยู่กับความต้องการว่าอยากทดสอบอะไร เช่น อาจทดสอบกระบวนการทำงานของระบบ, ทดสอบระบบรวม, ทดสอบตรวจรับของผู้ใช้ คือ การทดสอบความสามารถในการทำงานของเครื่องคอมพิวเตอร์และอุปกรณ์ ซึ่งกระทำโดยผู้ใช้ว่าสามารถทำงานได้ครบถ้วนตามรายละเอียดที่ตกลงกันหรือไม่, ทดสอบข้อบังคับต่าง ๆ และทดสอบคุณสมบัติในการใช้งานของเครื่อง

การวางแผนและควบคุมโครงการ

การทำงานด้านการวิเคราะห์และออกแบบระบบ มีขั้นตอนการทำงานหลายขั้นตอนด้วยกันซึ่งจะต้องดำเนินไปอย่างมีระบบ เมื่อเริ่มต้นวิเคราะห์ระบบแล้ว การวางแผนโครงการและการควบคุมโครงการให้ดำเนินไปตามแผนจะต้องเริ่มต้นในทันทีเช่นเดียวกัน

การวางแผนโครงการคือ ความพยายามที่จะคาดคะเนเวลาและค่าใช้จ่ายที่จะใช้ในการดำเนินโครงการใดโครงการหนึ่งรวมทั้งผลประโยชน์ที่จะได้รับจากโครงการด้วยแผนงานของโครงการจะรวมถึงขั้นตอนการทำงาน กิจกรรมที่จะต้องทำ เวลาที่ใช้ในแต่ละกิจกรรม รวมทั้งบุคคลากรที่เหมาะสมในแต่ละกิจกรรมด้วย แต่ละโครงการควรจะวางแผนในรายละเอียดให้มากก่อนที่จะเริ่มทำงานจริง และเมื่อทำงานจริง ๆ แล้วควรจะติดตามและควบคุมให้เป็นไปตามไปตามแผนที่วางไว้ด้วย

แผนงานของโครงการวิเคราะห์และออกแบบระบบ จะประกอบแผนงานย่อยของกิจกรรมเช่น การวิเคราะห์ การออกแบบ การพัฒนาโปรแกรม ฝึกอบรม และการนำระบบมาใช้งานจริง

การควบคุมโครงการจะตรงข้ามกับการวางแผนโครงการ การควบคุมโครงการเป็นการติดตามการทำงานเมื่อโครงการเริ่มต้นแล้ว ซึ่งจะทำให้ทราบว่าการทำงานเป็นไปตามจุดประสงค์ของแผนงานหรือไม่ ซึ่งต้องมีการติดตามการทำงานอย่างต่อเนื่องเพื่อควบคุมการทำงานให้เป็นไปตามตารางแผนงานและค่าใช้จ่ายให้อยู่ในงบประมาณที่ตั้งไว้ การติดตามการทำงานอย่างต่อเนื่องช่วยให้ค้นพบปัญหาใดปัญหาหนึ่งได้ทันท่วงทีในกรณีที่มีปัญหาเกิดขึ้น และจะทำให้การแก้ไขเป็นไปได้ง่ายและประหยัดค่าใช้จ่ายด้วย

การวางแผนงานจะมีประโยชน์มาก ถ้าการคาดคะเนถูกต้อง การคาดคะเนที่ถูกต้องทั้งเวลา ค่าใช้จ่าย และผลประโยชน์ จะเป็นหัวเลี้ยวหัวต่อสำคัญของการส่งมอบระบบให้ตรงเวลาและค่าใช้จ่ายไม่บานปลาย วิธีที่จะช่วยในการคาดคะเนได้แก่ การนับลักษณะพิเศษของระบบตัวอย่างเช่นจำนวนระบบงานที่จะต้องทำ จำนวนระบบที่จะเชื่อมต่อกัน รูปแบบของโครงสร้างข้อมูล (อำพร พรประเสริฐสกุล, 2537)

ในส่วนของการวางแผนและควบคุมโครงการสามารถแบ่งเป็นหัวข้อย่อย ๆ ได้ดังนี้

- การวางแผนทรัพยากร (Resource Plan)
- ตารางกำหนดการ (Schedule)
- เคสทางธุรกิจ

1 การวางแผนทรัพยากร (Resource Plan)

การวางแผนทรัพยากรสามารถระบุถึงความต้องการของโครงการในแง่ของทีมงาน, การ อบรม, อุปกรณ์, การบริการ และงบประมาณ ซึ่งควรแสดงถึงประเภทและปริมาณของทรัพยากรเมื่อมีการร้องขอ

ช่วงแรกของวงจรชีวิตของโครงการมีความสำคัญมากในการระบุทรัพยากรทั้งหมด ที่ร้องขอมาในช่วงระหว่างวงจรชีวิตของโครงการเพื่อความสำเร็จ ระหว่างวงจรชีวิตของโครงการโดยการวางแผนทรัพยากรจำเป็นที่ต้องมีการแก้ไขและทบทวนใหม่เป็นระยะ ๆ เพื่อจัดการทรัพยากร

การวางแผนทรัพยากร หากเป็นไปได้ควรทำให้เสร็จในช่วงแรก ๆ การวางแผนควรมีการปรับปรุงและแก้ไขขณะที่ทำโครงการ รวมไปถึงรายละเอียดของโครงการ

มีเครื่องมือมากมายที่สนับสนุนการวางแผนทรัพยากรเช่น PERT Chart, ตารางเวลา และการวิเคราะห์จุดวิกฤต ซึ่งเครื่องมือเหล่านี้มีความสำคัญและสามารถที่จะจัดการโครงการได้ดี

ตัวอย่างจากรูปที่ 3.32 จะแสดงให้เห็นว่าการวางแผนทรัพยากรควรจะกำหนดหน้าที่ของทีมงานในแต่ละคน

สมาชิกในทีมพัฒนา	หน้าที่ในโครงการ
คุณสมชาย รักงานดี	ผู้จัดการโครงการ
คุณสมศรี สวยสม	นักออกแบบ, หัวหน้าทีม
คุณสมชาติ ตระกูล	นักวิเคราะห์, นักออกแบบ, นักพัฒนา, นักทดสอบ
คุณนารี ผลงาม	นักวิเคราะห์, นักออกแบบ
คุณอภิชาติ พูนผล	นักออกแบบ, นักพัฒนา, นักทดสอบ
คุณอารี แย้มยิ้ม	นักพัฒนา
คุณมนัส เก่งจริง	นักพัฒนา

รูปที่ 3.32 ตัวอย่างการวางแผนทรัพยากรของทีมพัฒนา

2. ตารางกำหนดการ (Schedule)

ตารางกำหนดการคือกลุ่มของกิจกรรม วันเริ่มต้น กิจกรรมที่เกิดขึ้นในช่วงเวลาดังกล่าว การกำหนดการทำงาน และขั้นของวงจรชีวิตดังแสดงในรูปที่ 3.33

ตารางกำหนดการควรถูกสร้างขึ้นมาตอนเริ่มโครงการ เพื่อเป็นฐานในการสร้างฟังก์ชันต่าง ๆ และวันสุดท้ายของโครงการควรถูกกำหนดโดยใช้การประเมินตัวเลขของยูสเคส หรือทุกส่วนที่เกี่ยวข้อง เปรียบเทียบโครงการที่คล้ายกันและอาศัยประสบการณ์ที่มีอยู่

ตารางกำหนดการถูกสร้างเพื่อควบคุมงานให้เสร็จเรียบร้อยตามช่วงเวลาที่กำหนด โดยใช้แผนและเครื่องมือในการจัดการ

เริ่มแรกควรหาวันสุดท้ายของโครงการ ระหว่างวงจรชีวิตของการพัฒนาโครงการควรเตรียมการในส่วนดำเนินการของโครงการ เปรียบเทียบกับวันจริงที่ต้องการให้โครงการเสร็จเรียบร้อยสำหรับแต่ละหัวข้อในตารางกำหนดการถูกวางแผนเรื่องวันแล้วส่งกลับไปยังผู้จัดการโครงการเพื่อตามผล ถ้าโครงการอยู่หลังตารางกำหนดการซึ่งต้องถูกแก้ไข ความแตกต่างระหว่างการวางแผนและความจริงคือการวิเคราะห์ถึงสาเหตุซึ่งต้องแก้ไขให้เกิดความเหมาะสม ถ้าเหมาะสมแล้วข้อมูลเกี่ยวกับการวัดใช้สำหรับการประเมินโครงการซึ่งควรปรับปรุง นำไปสู่ข้อมูลตารางกำหนดการจริง

ผู้จัดการโครงการและนักวางแผนควรแบ่งความรับผิดชอบสำหรับตารางกำหนดการและควรได้รับความช่วยเหลือจากหัวหน้าทีมโครงการ

หากเป็นไปได้ตารางกำหนดการควรเสร็จเรียบร้อยในช่วงแรก ๆ ซึ่งตารางกำหนดการควรมีการปรับปรุงและแก้ไข ขณะที่ทำโครงการรายละเอียดจริงของโครงการควรมีความเป็นไปได้และมีการทบทวนเป็นระยะ ๆ เพื่อระบุสิ่งที่ผิดแผกไปจากการวางแผน เพื่อให้เกิดความถูกต้องและเหมาะสม (จรณิต แก้วกังวาล, 2540)

Activitys	#Weeks	1998					
		Jan	Feb	Mar	Apr	May	Jun
Staff	2						
Requirements	6						
Analysis	4						
Design	7						
Test	3						
Install and Train	1						

รูปที่ 3.33 ตัวอย่างตารางกำหนดการ

3. <u>เคสทางธุรกิจ</u> (Business Case)

เคสทางธุรกิจเป็นส่วนที่เตรียมเหตุผลสำหรับการยอมรับโครงการ โดยเคสทางธุรกิจเป็นส่วนประกอบกับการกำหนดปัญหาคือ การกำหนดปัญหาเป็นการแสดงปัญหาที่ต้องแก้ไข ส่วนเคสทางธุรกิจให้เหตุผลทางธุรกิจ เคสทางธุรกิจมี 2 แบบคือ ปริมาณซึ่งเกี่ยวข้องกับการวัดและเชิงคุณภาพโดยเป็นสิ่งที่จับต้องไม่ได้และไม่จำเป็นในการวัด

เคสทางธุรกิจมีเหตุผลของโครงการจากภาพพจน์ของธุรกิจ บ่อยครั้งที่นำแบบฟอร์มของการวิเคราะห์ต้นทุน/ผลประโยชน์ โดยต้นทุนของโครงการคือการประเมินจากทรัพยากรที่ต้องใช้ และผลประโยชน์ของโครงการที่มีหลายแบบ เช่น รายได้, ผลิตผล, ความนิยมของผู้ใช้, ได้ผู้มีประสบการณ์ตามต้องการมาพัฒนาโครงการ, การนำส่วนประกอบที่มีอยู่มาใช้ใหม่ ตัวอย่างเช่น การวิเคราะห์โดเมน (Domain) ของระบบโทรศัพท์ และโครงประกอบ (Framework) ของโปรแกรมในระบบโทรศัพท์แบบง่าย ๆ ซึ่งสามารถจะนำกลับมาใช้กับโปรแกรมในอนาคตได้