

การใช้ระบบเชิงวัตถุกับระบบงานบริหารพัสดุสายช่างอากาศ

OBJECT-ORIENTED SYSTEM FOR AERONAUTICAL SUPPLY MANGEMENT



นาวาอากาศโทหญิงนพพร สงวนทรัพย์
WING COMMANDER NOPPORN SA-NGUANSUP

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ

บัณฑิตวิทยาลัย

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

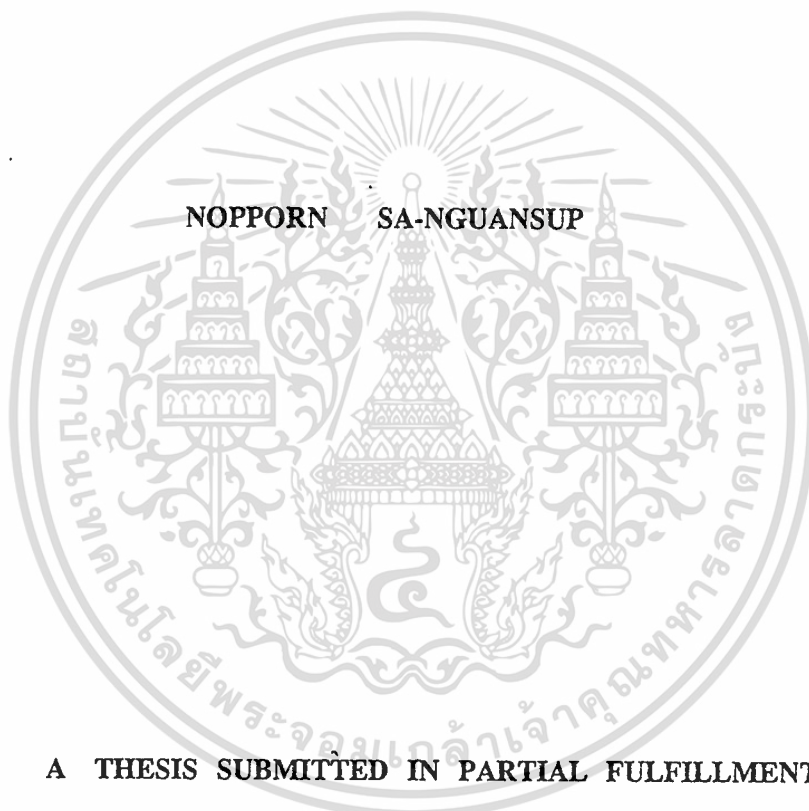
พ.ศ. 2539

ISBN 974-621-587-6

ลิขสิทธิ์ของบัณฑิตวิทยาลัย สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เลขที่.....
เลขทะเบียน.....27119
วัน, เดือน, ปี 1 ก.พ 2540

OBJECT-ORIENTED SYSTEM FOR AERONAUTICAL SUPPLY MANGEMENTIGN



**A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE
MASTER OF SCIENCE PROGRAM IN COMPUTER SCIENCE
AND INFORMATION TECHNOLOGY
SCHOOL OF GRADUATE STUDIES
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG**

1996

ISBN 974-621-587-6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อวิทยานิพนธ์ การใช้ระบบเชิงวัตถุกับระบบงานบริหารงานพัสดุสายช่างอากาศ
นักศึกษา น.ท.หญิง นพพร สงวนทรัพย์
อาจารย์ผู้ควบคุมวิทยานิพนธ์ ผศ.ดร.ศุภมิตร จิตตะยโสธร
ระดับการศึกษา วิทยาศาสตร์มหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ
ภาควิชา คณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
พ.ศ. 2539

บทคัดย่อ

ซอฟต์แวร์ในปัจจุบันมีการเพิ่มคุณสมบัติที่ทำให้สามารถเขียนโปรแกรมแบบเชิงวัตถุได้ ตัวอย่างเช่น Visual Basic, Visual C, Delphi เป็นต้น ดังนั้นเพื่อเป็นการเตรียมการรองรับสำหรับซอฟต์แวร์ใหม่ๆ ในอนาคตที่มีแนวโน้มเปลี่ยนไปเป็นแบบเชิงวัตถุจึงได้นำเอาหลักการเชิงวัตถุมาออกแบบระบบงานบริหารพัสดุสายช่างอากาศของกรมช่างอากาศ ซึ่งหลักการคือกำหนดข้อมูลในระบบงานเป็นออบเจกต์คลาสแทนการกำหนดเป็นฐานข้อมูลเช่นในวิธีการออกแบบระบบแบบเดิม ภายในออบเจกต์คลาสประกอบด้วยข้อมูลและเมธอด(หรือโพรซีเยอร์)รวมอยู่ด้วยกันในขณะที่แบบเก่าจะแยกส่วนของข้อมูลและโพรซีเยอร์(หรือโปรแกรมระบบนั่นเอง)ออกจากกัน ออกแบบระบบงานโดยใช้ออบเจกต์โมเดลลิ่งเทคนิคหรือโอเอ็มที (OMT : Object Modeling Technique) และพัฒนาระบบงานบนสมอลทอล์ค (Smalltalk) สาเหตุที่เลือกสมอลทอล์คเพราะว่าสมอลทอล์คเป็นต้นแบบของซอฟต์แวร์เชิงวัตถุ

Thesis Title Object-Oriented System for Aeronautical Supply Management
Student Wg.Cdr. Nopporn Sa-nuangsup
Thesis Advisor Asst.Prof. Dr. Supamit Jittayasothorn
Level of Study Master of Science Program in
 Computer Science and Information Technology
Department Mathematics and Computer Science
 Faculty of Science
 King Mongkut's Institute of Technology Ladkrabang
Year 1996



ABSTRACT

Today application developing tools such as Visual Basic, Visual C, Delphi are object-oriented. Applications implemented by these tools should therefore be object-oriented designed. In this thesis Aeronautical Engineering Depot supply management system is designed using Object Modeling Technique and implemented in Smalltalk. Smalltalk is chosen because object-oriented features are fully implemented and supported by the language.

กิตติกรรมประกาศ

วิทยานิพนธ์นี้ประสบความสำเร็จลงได้ด้วยดีก็เพราะว่าได้รับความกรุณาจากท่านผู้ช่วยศาสตราจารย์ ดร.ศุภมิตร จิตตะยโสธร ท่านกรุณาให้ข้อเสนอแนะและเป็นกำลังใจผลักดันจนกระทั่งทำให้ดิฉันได้มีวันนี้ ดังนั้นดิฉันจึงขอกราบขอบพระคุณท่านอาจารย์เป็นอย่างสูงไว้

ขอขอบพระคุณผู้บังคับบัญชาทุกท่าน เพื่อนทุกคน รวมทั้งเจ้าหน้าที่ของสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุกคน ที่ได้มีส่วนช่วยสนับสนุนผลงานวิทยานิพนธ์นี้

ท้ายที่สุดขอขอบคุณ นอ.รัช สงวนทรัพย์ ที่เสียสละให้ดิฉันใช้เวลาของครอบครัวทำงานวิทยานิพนธ์จนประสบความสำเร็จ

นพพร สงวนทรัพย์

สารบัญ

	หน้า
บทคัดย่อภาษาไทย.....	I
บทคัดย่อภาษาอังกฤษ.....	II
กิตติกรรมประกาศ.....	III
สารบัญ.....	IV
สารบัญภาพ.....	VIII
บทที่	
1 บทนำ.....	1
ความเป็นมาและทฤษฎีที่ใช้.....	1
สาเหตุที่เลือกออกแบบเจดโมเดลลิงเทคนิค.....	2
โครงสร้างวิทยานิพนธ์.....	2
2 ทฤษฎีที่เกี่ยวข้อง.....	4
ออกแบบเจดโมเดลลิงเทคนิค.....	4
ออกแบบเจดโมเดล.....	4
-ออกแบบเจดไดอะแกรม(Objectdiagram).....	4
- แอสโซซิเอชัน(Association).....	6
- อะกรีเกชัน (Aggregation).....	11
- เจนเนอรัลไลเซชัน (Generalization).....	13
- เปรียบเทียบอะกรีเกชันกับแอสโซซิเอชัน.....	13
- เปรียบเทียบอะกรีเกชันกับเจนเนอรัลไลเซชัน.....	14
- ข้อบังคับ (Constraints).....	15
ไดนามิกโมเดล (Dynamic Model).....	16
- อีเวนท์ (Event).....	16
- สเตท (State).....	19
- โอเปอเรชัน (Operation).....	20
- สเตทไดอะแกรม.....	21

สารบัญ (ต่อ)

บทที่	หน้า
ฟังก์ชันนัลโมเดล (Functional Model).....	22
- โพรเซส (Process).....	23
- ดาต้าโฟล (Data Flow).....	23
- การกำหนดโอเปอเรชัน.....	26
- ข้อบังคับ (Constraint).....	27
โปรแกรมภาษาสมอลทอล์ค (Smalltalk).....	29
- ชนิดของตัวแปร.....	29
- คำสั่ง.....	29
- บล็อกอาร์กิวเมนต์.....	34
- ตัวอย่างโปรแกรม.....	35
การใช้สมอลทอล์คไฟร์ (Smalltalk / V).....	35
- เมนู.....	36
- ออบเจกต์และเมสเสจ.....	37
- คลาสและเมธอด.....	38
3 ระบบงานบริหารพัสดุสายช่างอากาศ.....	39
โพรเซส 1.0 การรับเอกสาร.....	39
โพรเซส 2.0 การรับพัสดุส่งคืน.....	39
โพรเซส 3.0 การหักล้างค้างจ่ายพัสดุ.....	39
โพรเซส 4.0 การส่งซ่อมพัสดุ.....	40
โพรเซส 5.0 การรับพัสดุจากซ่อม.....	40
โพรเซส 6.0 การจำหน่ายพัสดุ.....	40
โพรเซส 7.0 การจ่ายพัสดุ.....	40
โพรเซส 8.0 การจัดหาพัสดุ.....	40
โพรเซส 9.0 การรับพัสดุจากการจัดหา.....	41
4 การพัฒนาระบบงาน.....	50

สารบัญ (ต่อ)

บทที่	หน้า
การวิเคราะห์เชิงวัตถุ.....	50
ประโยคปัญหา (Problem statement).....	50
ออบเจกโมเดลลิ่ง.....	51
- กำหนดคลาส.....	52
- กำหนดแอสโซซิเอชัน.....	54
- กำหนดเอสโซซิเอชัน.....	63
ไดนามิกโมเดลลิ่ง.....	67
- การกำหนดอีเวนต์.....	69
- การสร้างสเตตโอดีอะแกรม.....	70
ฟังก์ชันนัลโมเดลลิ่ง.....	88
- การกำหนดค่าอินพุตและเอาต์พุต.....	88
- สร้างด้าด้าไฟล์โอดีอะแกรม.....	88
- การกำหนดข้อบังคับระหว่างออบเจค.....	89
- การเพิ่มโอเปอเรชัน.....	90
5 การบำรุงรักษาระบบ.....	103
การบำรุงรักษา.....	103
การบันทึกการเปลี่ยนแปลงแบบอัตโนมัติ.....	103
ความสำคัญของการเก็บอิมเมจ.....	103
การลดขนาดไฟล์เซนจ์ล็อก (Compressing Change Log).....	104
การลดขนาดของไฟล์ซอส (Compressing source).....	104
การกู้ระบบ (Surviving a System Crash).....	104
การกู้ข้อมูล (Recovering from a System Crash).....	106
6 สรุปและข้อเสนอแนะ.....	107
สรุปผลการวิจัย.....	107
ข้อเสนอแนะ.....	108

สารบัญ (ต่อ)

บทที่	หน้า
ภาคผนวก ก บทความที่ตีพิมพ์ในวารสาร.....	109
ภาคผนวก ข การใช้ระบบงาน.....	119
ภาคผนวก ค ตัวอย่างรายงาน.....	132
เอกสารอ้างอิง.....	136
ประวัติผู้เขียน.....	137



สารบัญภาพ

	หน้า
1 ออบเจคไดอะแกรม.....	5
2 ออบเจคคลาสไดอะแกรม.....	5
3 ตัวอย่างความสัมพันธ์แบบ หนึ่ง-ต่อ-หนึ่ง (one-to-one association).....	6
4 ตัวอย่างความสัมพันธ์แบบ many-to-many association.....	7
5 ตัวอย่างความสัมพันธ์แบบ ternary association.....	8
6 ลิงค์แอสซิซิเอชันสำหรับความสัมพันธ์แบบ many-to-many.....	9
7 ลิงค์แอสซิซิเอชันสำหรับความสัมพันธ์แบบ one-to-many.....	9
8 ลิงค์แอสซิซิเอชันสำหรับความสัมพันธ์แบบ ternary.....	10
9 ลิงค์แอสซิซิเอชันเปรียบเทียบกับออบเจคแอสซิซิเอชัน.....	11
10 ตัวอย่างการกำหนดควอลลิไฟเออร์.....	12
11 ตัวอย่างอะกรีเกชัน.....	12
12 ตัวอย่างอะกรีเกชันหลายระดับ.....	12
13 ตัวอย่างเจนเนอรัลไลเซชัน.....	13
14 ตัวอย่างอะกรีเกชัน.....	13
15 ตัวอย่างอะกรีเกชันและเจนเนอรัลไลเซชัน.....	14
16 รีเคอซีฟอะกรีเกชัน.....	15
17 ตัวอย่างของข้อบังคับ.....	15
18 ภาพเหตุการณ์สำหรับ phone call.....	18
19 ทางเดินของภาพเหตุการณ์สำหรับ phone call.....	19
20 สเตทไดอะแกรม.....	21
21 ตัวอย่างดาต้าไฟล์.....	24
22 ตัวอย่างดาต้าไฟล์.....	24
23 ตัวอย่างดาต้าไฟล์ไดอะแกรม.....	25
24 ดาต้าไฟล์ไดอะแกรมของระบบงาน ระดับ 0	42
25 โพรเซส 1.0 การรับเอกสาร ระดับ 1.....	43

สารบัญภาพ (ต่อ)

	หน้า
26 โพรเซส 2.0 การรับพัสดุส่งคืน ระดับ 1	44
27 โพรเซส 3.0 การหักล้างค้างจ่าย ระดับ 1	45
28 โพรเซส 4.0 การส่งซ่อมพัสดุ ระดับ 1	46
29 โพรเซส 5.0 การรับพัสดุจากซ่อม ระดับ 1	46
30 โพรเซส 6.0 การจำหน่ายพัสดุ ระดับ 1	47
31 โพรเซส 8.0 การจัดหาพัสดุ ระดับ 1	47
32 โพรเซส 7.0 การจ่ายพัสดุ ระดับ 1	48
33 โพรเซส 9.0 การรับพัสดุดำเนินการ ระดับ 1	49
34 ออบเจคโมเดล.....	66
35 ไดนามิกโมเดล.....	73
36 ฟังก์ชันนัลโมเดล.....	94

บทที่ 1

บทนำ

1.1 ความเป็นมาและทฤษฎีที่ใช้

งานวิจัยนี้ได้นำเสนอการออกแบบระบบงานโดยใช้หลักการเชิงวัตถุ ซึ่งแตกต่างไปจากการออกแบบระบบงานแบบเดิม (Conventional) ที่ใช้วิธีการเก็บรวบรวมข้อมูลต่างๆ เป็นแฟ้มข้อมูล (File) หรือฐานข้อมูล (Database) แยกกันกับโปรแกรมระบบงาน ในขณะที่หลักการเชิงวัตถุ [1] จะกำหนดสิ่งต่างๆ ที่อยู่ภายในขอบเขตของระบบงานเป็นออบเจกต์ (Object) ซึ่งในออบเจกต์จะประกอบด้วยข้อมูล (Data) และเมธอด (Method) รวมกันอยู่ภายในออบเจกต์ (Encapsulation) เมธอดเปรียบเทียบกับทำนองเดียวกันกับโพรซีเจอร์ (Procedure) ในโปรแกรมระบบงานแบบเดิมนั้นเอง การเรียกใช้เมธอดหรือการสั่งให้เกิดการกระทำตามที่กำหนดไว้ในเมธอดคือการส่งเมสเสจ (Message) ไปยังออบเจกต์นั้นๆ ดังนั้นหากมีการเปลี่ยนแปลงวิธีการกระทำ (Function) ใดๆ ของ ออบเจกต์จะไม่มีผลกระทบต่อเมสเสจที่ส่งไปยังออบเจกต์นั้นและจะไม่มีผลกระทบต่อออบเจกต์อื่นๆ ด้วย ซึ่งนับว่าเป็นผลดีอย่างมากกับระบบงานเพราะโดยปกติแล้วพบว่าระบบงานมักจะมีการเปลี่ยนแปลงวิธีการปฏิบัติอยู่เสมอๆ เมธอดสามารถใช้อธิบายถึงพฤติกรรมของออบเจกต์แต่ละออบเจกต์ได้ ออบเจกต์ที่มีพฤติกรรมแบบเดียวกันสามารถจัดรวมกลุ่มเป็นกลุ่มเดียวกันได้เรียกว่าคลาส (Class) ในระหว่างคลาสสามารถที่จะถ่ายทอดคุณสมบัติ (Inheritance) จากคลาสหนึ่งไปยังอีกคลาสหนึ่งได้ โดยเรียกคลาสที่ถ่ายทอดคุณสมบัติว่าซูเปอร์คลาส (Super class) และเรียกคลาสที่ได้รับการถ่ายทอดคุณสมบัติว่าซับคลาส (Subclass) ซึ่งทำให้การบำรุงรักษาระบบงานทำได้ง่าย เริ่มต้นงานวิจัยด้วยการศึกษาหลักการเชิงวัตถุต่อจากนั้นทำการวิเคราะห์เชิงวัตถุ (Object-Oriented Analysis) โดยใช้วิธีออบเจกต์โมเดลลิงเทคนิค (Object Modeling Technique) หรือโอเอ็มที (OMT) [2] ในระบบงานบริหารพัสดุสายช่างอากาศกำหนดคลาส Item เป็น คลาสสำหรับพัสดุทั้งหมด คลาส Item ประกอบด้วยสองซับคลาส คือคลาส Repairable และคลาส Consumable โดยคลาส Repairable คือคลาสของพัสดุประเภทพัสดุซ่อมหมุนเวียนและคลาส Consumable คือคลาสของพัสดุประเภทพัสดุลิ้นเปลือง หลังจากนั้นสร้างแบบจำลองเพื่อใช้อธิบายระบบงานทั้งระบบงานซึ่งมีอยู่ด้วยกัน 3 แบบจำลอง คือหนึ่งออบเจกต์โมเดล (Object Model) ใช้อธิบายรายละเอียดเกี่ยวกับออบเจกต์นั้นมีข้อมูลและเมธอดอะไรบ้างและแสดงความสัมพันธ์ระหว่างออบเจกต์แต่ละออบเจกต์ภายในระบบงาน แบบจำลองที่สองคือไดนามิก

โมเดล (Dynamic Model) ซึ่งใช้สเตตไดอะแกรม (State Diagram) อธิบายพฤติกรรมของออบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่นับผูกมัดให้เข้าระบบสารสนเทศ

ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เจตที่กระทำโต้ตอบกันภายในระบบ ไดนามิกโมเดลนี้ไม่ค่อยมีความจำเป็นมากนักสำหรับระบบงานบริหารพัสดุสายช่างอากาศ เพราะว่าเป็นระบบงานที่เกี่ยวกับการเก็บข้อมูลเสียเป็นส่วนใหญ่ แบบจำลองสุดท้ายคือฟังก์ชันนัลโมเดล(Functional Model) ซึ่งใช้ดาต้าโฟลไดอะแกรม(Data Flow Diagram) อธิบายถึงวิธีการคำนวณของเมธอดภายในออบเจกต์แต่ละออบเจกต์ หลังจากสร้างแบบจำลองเสร็จแล้วอันดับต่อไปก็คือการพัฒนากระบวนการงาน (Implementation) บนสมอลทอค [3]

1.2 สาเหตุที่เลือกใช้ออบเจกต์โมเดลลิ่งเทคนิค

วิธีการเชิงวัตถุ (Object-Oriented Methodology) มีอยู่หลายวิธีด้วยกัน เช่น ออบเจกต์โอเรียนเตดอะนาไลซิส(Object-Oriented System Analysis) ซึ่งสร้างขึ้นโดยแชลเลอร์และเมลเลอร์ (Shlaer and Mellor) [Shlaer 1988] [4] โดยใช้อีอาร์ไดอะแกรม(ER-diagram) อธิบายโครงสร้างของข้อมูล ต่อมาในปี 1990 Coad-Yourdon ได้สร้างออบเจกต์โอเรียนเตดอะนาไลซิส (Object-Oriented Analysis) หรือโอโอเอ (OOA) [5] ซึ่งคล้ายๆ กับออบเจกต์โอเรียนเตดซิสเต็มอะนาไลซิส (Object-Oriented System Analysis) หรือโอเอสเอ (OSA) ซึ่งพัฒนามาจาก ออบเจกต์รีเลชันชิพโมเดล (Object Relationship Model) หรือโออาร์เอ็ม (ORM) [6] โออาร์เอ็มเป็นผลงานวิจัยของเคิร์ตซ์ (Kurtz) [Kurtz 1988] โอโอเอแตกต่างกับโอเอสเอคือโอเอสเอเน้นสะท้อนถึงรูปแบบของการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming) ในปี 1991 เจมส์รามบอจ (James Rumbaugh) สร้างออบเจกต์โมเดลลิ่งเทคนิค (Object Modeling Technique-OMT) ซึ่งใช้เอ็กซ์เทนเดดอีอาร์ไดอะแกรม(extended ER diagram) อธิบายโครงสร้างของออบเจกต์ ใช้สเตตไดอะแกรม (state diagram) อธิบายถึงพฤติกรรมโต้ตอบระหว่างออบเจกต์ และใช้ดาต้าโฟลไดอะแกรม (data flow diagram) อธิบายวิธีการคำนวณของเมธอดในออบเจกต์ เราจะสังเกตเห็นได้ว่าการวิเคราะห์เชิงวัตถุแต่ละวิธีจะคล้ายๆ กัน ดังนั้นผู้ทำวิจัยจึงเลือกใช้วิธีออบเจกต์โมเดลลิ่งเทคนิคในงานวิจัยนี้ การออกแบบระบบงานโดยใช้หลักการเชิงวัตถุเท่าที่ผ่านมามักจะอยู่ในรูปของปัญหาสั้นๆ หรือเป็นเพียงบางส่วนเท่านั้น ดังนั้นผู้วิจัยจึงคิดที่จะทดลองออกแบบระบบงานบริหารพัสดุสายช่างอากาศแบบครบวงจร (Real Lift System) โดยใช้ออบเจกต์โมเดลลิ่งเทคนิค และพัฒนาระบบงานบนสมอลทอค เพราะว่สมอลทอคเป็นต้นแบบของซอฟต์แวร์เชิงวัตถุ

1.3 โครงสร้างวิทยานิพนธ์

รายละเอียดของเนื้อหาในแต่ละบทถัดไปมีดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2 ออบเจกโมเดลลิ่งเทคนิคและสมอลทอค กล่าวถึงหลักการของออบเจกโมเดลลิ่งเทคนิคและโปรแกรมภาษาสมอลทอค

บทที่ 3 ระบบงานบริหารพัสดุสายช่างอากาศ กล่าวถึงลักษณะของระบบงานที่จะใช้ศึกษาในงานวิจัยนี้

บทที่ 4 การพัฒนาระบบงาน กล่าวถึงวิธีการออกแบบโดยใช้วิธีของออบเจกโมเดลลิ่งเทคนิค และการพัฒนาระบบงานที่ใช้ศึกษาโดยใช้โปรแกรมสมอลทอค

บทที่ 5 การข้อมนำร่องระบบเชิงวัตถุ กล่าวถึงการข้อมนำร่องระบบเชิงวัตถุของสมอลทอค

บทที่ 6 สรุปและข้อเสนอแนะ กล่าวถึงผลสรุปของงานวิจัย และข้อเสนอแนะ



บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

2.1 ออบเจกต์โมเดลลิ่งเทคนิค (Object Modeling Technique : OMT)

ออบเจกต์โมเดลลิ่งเทคนิคหรือโอเอ็มที ประกอบด้วยโมเดลที่ใช้ในการอธิบายระบบงานที่แตกต่างกัน 3 โมเดลด้วยกันคือ

1. ออบเจกต์โมเดล (Object Model)
2. ไดนามิกโมเดล (Dynamic Model)
3. ฟังก์ชันนัลโมเดล (Functional Model)

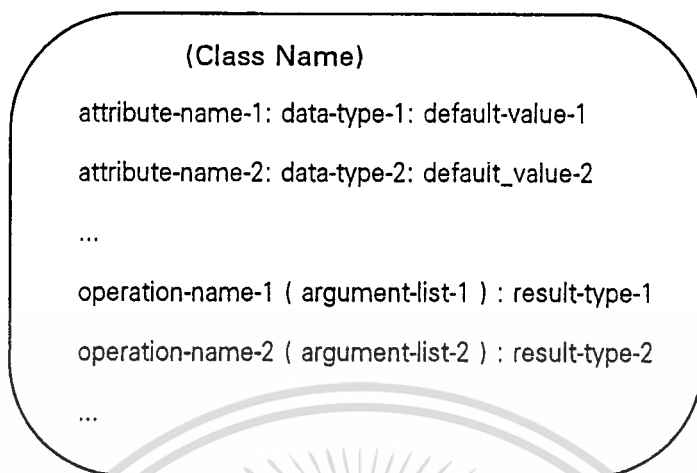
2.1.1 ออบเจกต์โมเดล

ออบเจกต์โมเดลเป็นโมเดลที่สำคัญที่สุดในจำนวนโมเดลทั้งสามโมเดล ออบเจกต์โมเดลอธิบายโครงสร้างที่ไม่มีการเปลี่ยนแปลงของระบบ โดยแสดงออบเจกต์ต่างๆ ภายในระบบ ความสัมพันธ์ระหว่างออบเจกต์ (Association) แอททริบิวต์ (Attribute) และโอเปอเรชัน (Operation) ทั้งหมดของแต่ละออบเจกต์

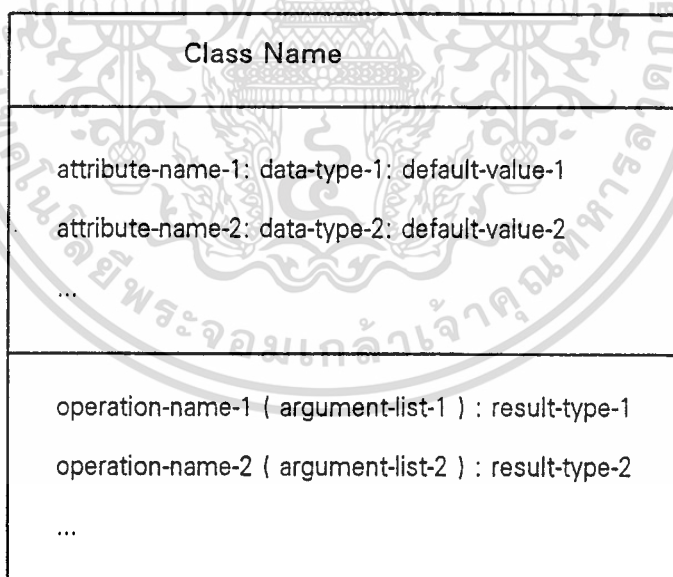
2.1.1.1 ออบเจกต์ไดอะแกรม (Object diagram)

แผนผังที่ใช้แทนออบเจกต์คือออบเจกต์ไดอะแกรม โดยใช้สัญลักษณ์รูปสี่เหลี่ยมผืนผ้ามุมมน ภายในรูปสี่เหลี่ยมผืนผ้ามุมมนเขียนชื่อของคลาสด้วยตัวอักษรทึบและอยู่ภายในเครื่องหมายวงเล็บ บรรทัดถัดมาเป็นชื่อของออบเจกต์ แอททริบิวต์ และโอเปอเรชันทั้งหมดของออบเจกต์ ดังแสดงในภาพที่ 1

แผนผังที่ใช้แทนคลาสคือรูปสี่เหลี่ยมผืนผ้า ภายในรูปสี่เหลี่ยมผืนผ้าเขียนชื่อของคลาสด้วยตัวอักษรทึบ บรรทัดถัดมาเป็นแอททริบิวต์ และโอเปอเรชันทั้งหมดของออบเจกต์ โดยระหว่างชื่อคลาส แอททริบิวต์ และโอเปอเรชันจะมีเส้นตรงแบ่งคั่นเพื่อให้แผนผังของคลาสแตกต่างไปจากแผนผังของออบเจกต์ ดังแสดงตัวอย่างในภาพที่ 2



ภาพที่ 1 ออบเจคไดอะแกรม



ภาพที่ 2 คลาสไดอะแกรม

2.1.1.2 แอสโซซิเอชัน (Association)

สัญลักษณ์ของแอสโซซิเอชันคือเส้นตรงที่ลากต่อระหว่างคลาส ซึ่งใช้แสดงความสัมพันธ์ของออบเจกต์ภายในคลาส มัลติพลิซิตี (Multiplicity) คือการกำหนดจำนวนสมาชิกของคลาสหนึ่งซึ่งมีความสัมพันธ์กับสมาชิกของอีกคลาสหนึ่ง โดยวิธีเขียนสัญลักษณ์บนแอสโซซิเอชันชิดกับคลาสนั้น โดยปกติมักใช้คำว่า one หรือ many

สัญลักษณ์ 1+ หมายถึงหนึ่งหรือมากกว่า

2,4 หมายถึง 2 และ 4

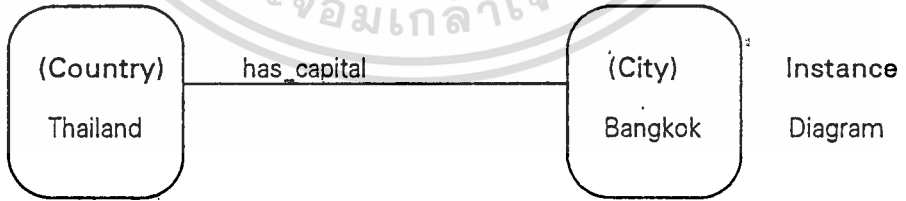
○— หมายถึงศูนย์หรือหนึ่ง

●— หมายถึงศูนย์หรือมากกว่าศูนย์

ดังตัวอย่างในภาพที่ 3 ถึงภาพที่ 5 ในภาพที่ 3 แสดงความสัมพันธ์แบบ หนึ่งต่อหนึ่ง นั่นคือประเทศหนึ่งมีเมืองหลวงได้เพียงหนึ่งเมืองเท่านั้น ภาพที่ 4 แสดงความสัมพันธ์แบบ มากต่อมาก นั่นคือจุด ๆ หนึ่งมีเส้นตรงตัดผ่านมากกว่าหนึ่งเส้น และเส้นตรงเส้นหนึ่งผ่านจุดตัดมากกว่าหนึ่งจุดหรือไม่ผ่านจุดใดเลย ภาพที่ 5 แสดงความสัมพันธ์แบบ Ternary ใช้สัญลักษณ์รูปสี่เหลี่ยมขนมเปียกปูนเป็นจุดเชื่อมต่อแอสโซซิเอชัน

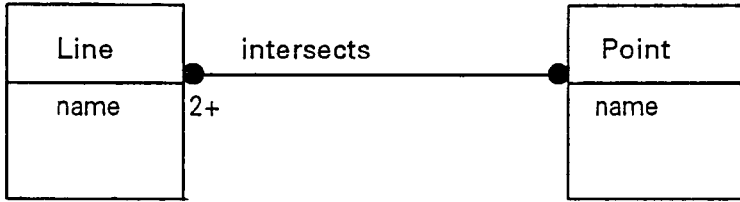


Class Diagram

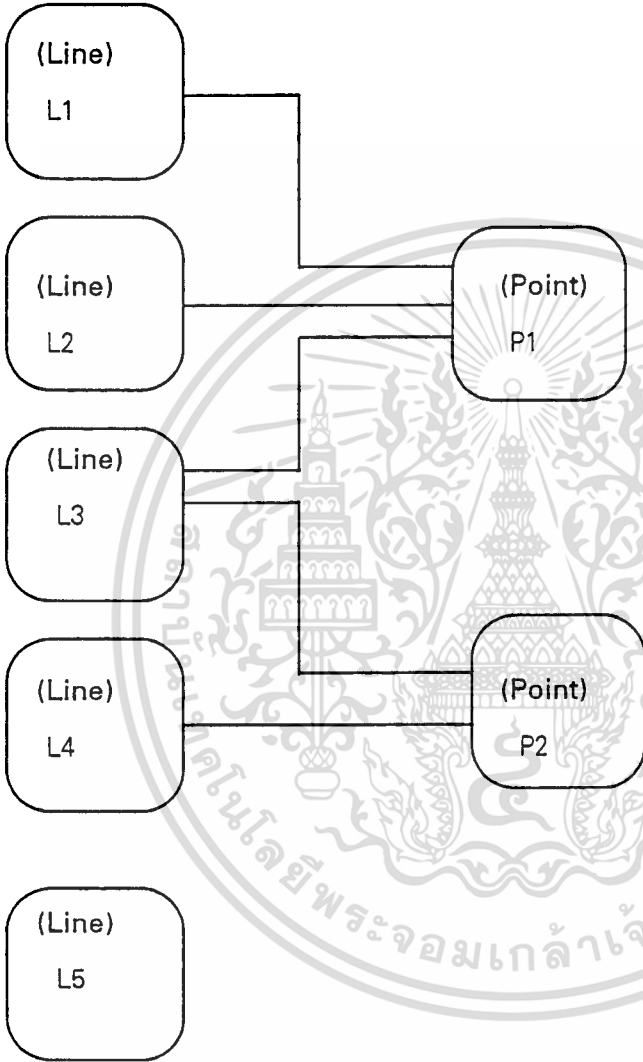


Instance Diagram

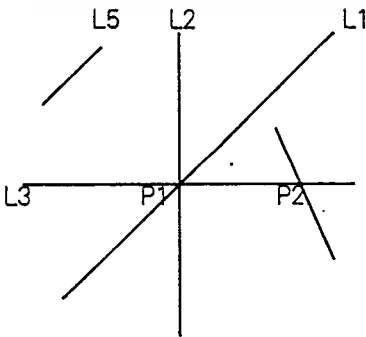
ภาพที่ 3 ตัวอย่างความสัมพันธ์แบบ หนึ่ง-ต่อ-หนึ่ง (one-to-one association)



Class Diagram



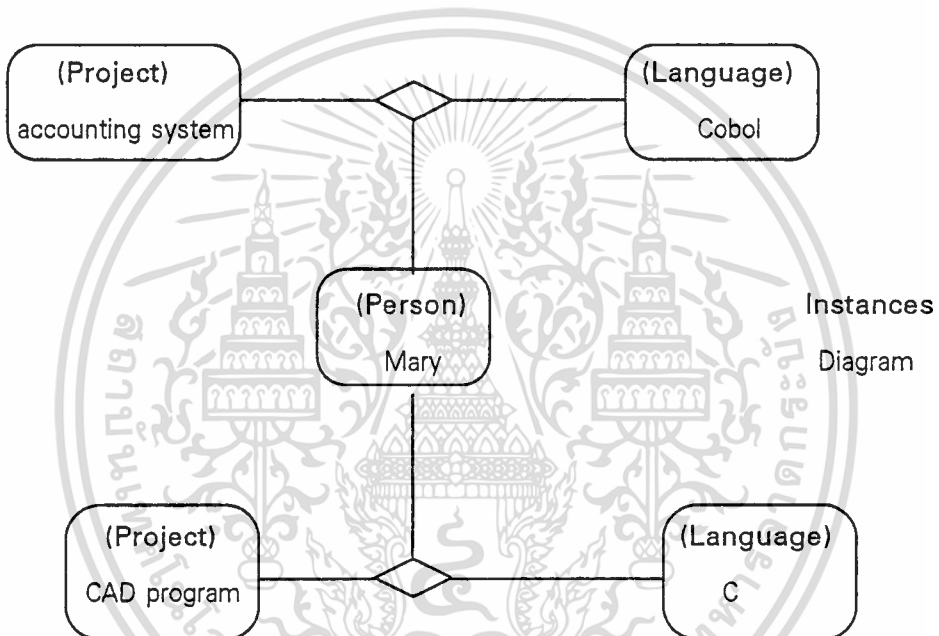
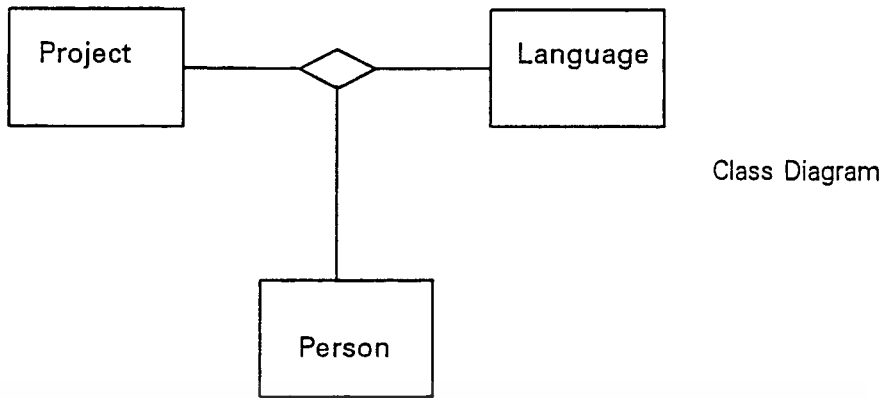
Instances Diagram



Sample Data

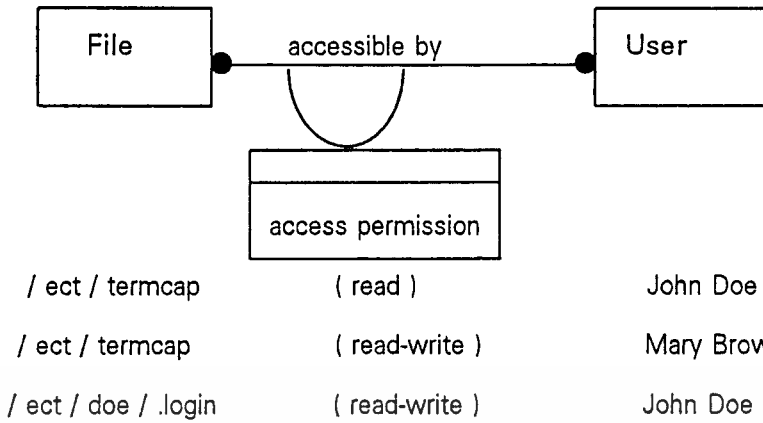
ภาพที่ 4 ตัวอย่างความสัมพันธ์แบบ many-to-many association

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

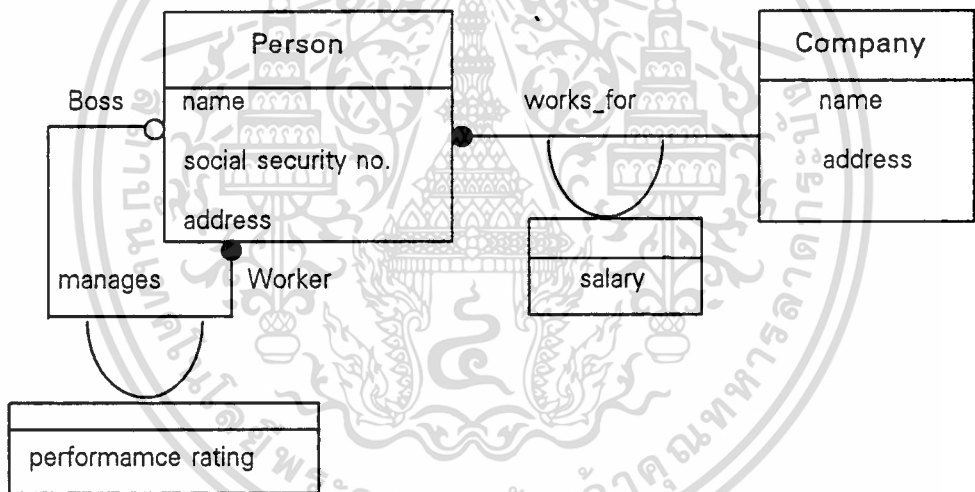


ภาพที่ 5 ตัวอย่างความสัมพันธ์แบบ ternary association

ลิงค์แอททริบิวต์ (Link Attribute) คือคุณสมบัติของลิงค์ในแอสโซซิเอชัน ในทำนองเดียวกับแอททริบิวต์คือคุณสมบัติของออบเจกต์ในคลาส สัญญลักษณ์ของลิงค์แอททริบิวต์คือรูปสี่เหลี่ยมสัมผัสกับแอสโซซิเอชันด้วยเส้นโค้งที่ลากต่อระหว่างรูปสี่เหลี่ยมสัมผัสกับแอสโซซิเอชันดังตัวอย่างในภาพที่ 6 ถึงภาพที่ 8

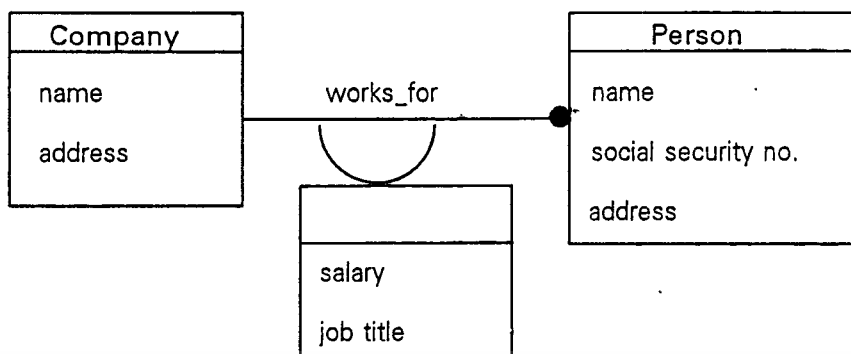


ภาพที่ 6 ลิงค์แอทริบิวท์สำหรับความสัมพันธ์แบบ many-to-many

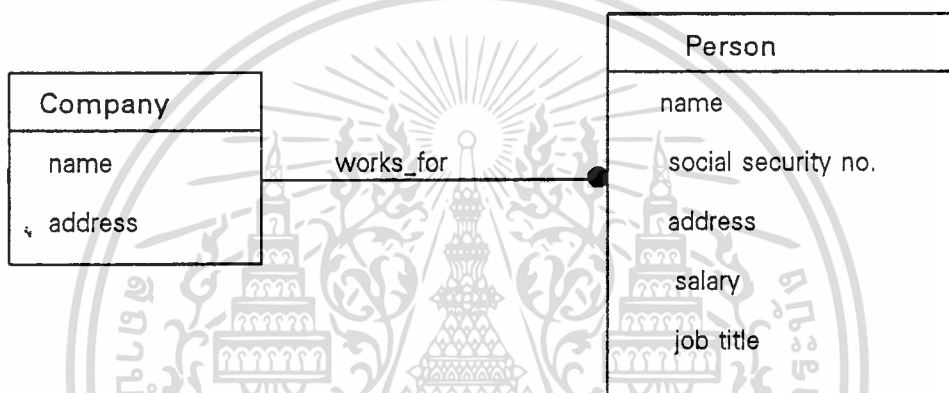


ภาพที่ 7 ลิงค์แอทริบิวท์สำหรับความสัมพันธ์แบบ one-to-many

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแบบที่ควรทำ



รูปแบบที่ไม่ควรทำ

ภาพที่ 9 ลิงค์แอสซิเอชันเปรียบเทียบกับออบเจกต์แอสซิเอชัน

ชื่อของบทบาท (Role Name) ใช้อธิบายบทบาทของคลาสในแอสซิเอชันนั้น โดยเขียนชื่อของบทบาทที่เหนือแอสซิเอชันใกล้ๆ กับคลาสที่แสดงบทบาท เช่นตัวอย่างในภาพที่ 7

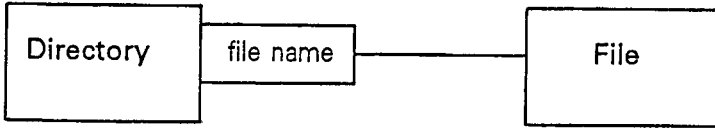
ควอลิฟิเคชัน (Qualification) คือแอสซิเอชันพิเศษซึ่งลดมัลติพลิซิตีของแอสซิเอชันแบบ one-to-many และ many-to-many โดยใช้สัญลักษณ์เป็นรูปกล่องสี่เหลี่ยมเล็กๆ ที่ปลายของแอสซิเอชันใกล้กับคลาสที่ถูกกำหนดควอลิฟิเคชัน เช่น ตัวอย่างในภาพที่ 10

2.1.1.3 อะกรีเกชัน (Aggregation)

คือความสัมพันธ์แบบ "part-whole" หรือ "a-part-of" ซึ่งใช้อธิบายความสัมพันธ์ในรูปแบบที่ออบเจกต์หนึ่งเป็นส่วนประกอบของอีกออบเจกต์หนึ่ง อะกรีเกชันใช้สัญลักษณ์เช่นเดียวกับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

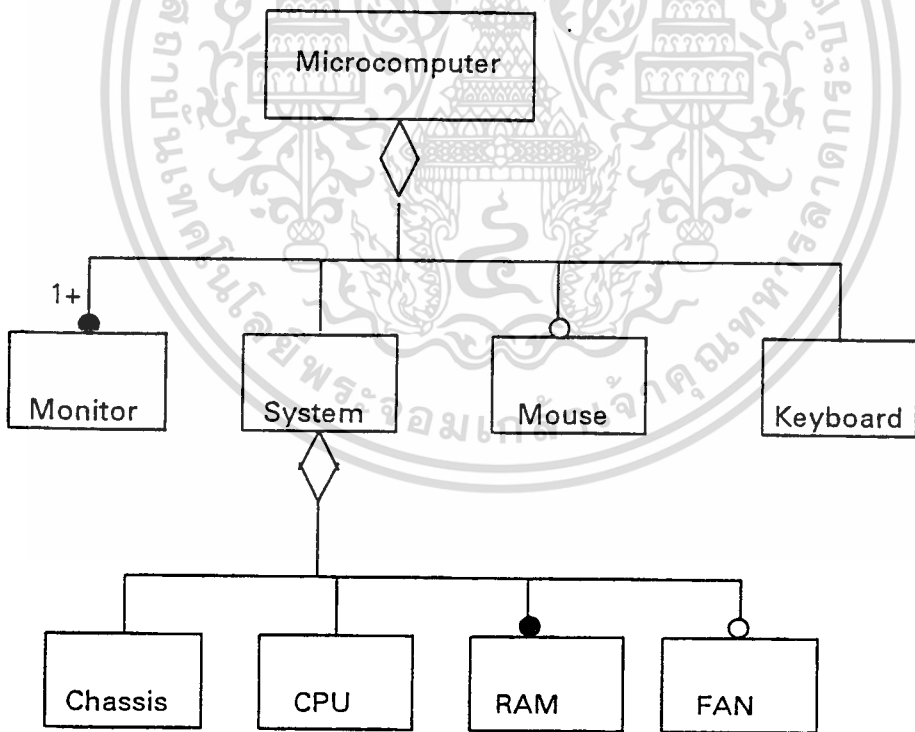
แอตทริบิวต์แต่มีสัญลักษณ์ที่เชื่อมขมมเป็ยกปุ่นที่ปลายของแอตทริบิวต์เพื่อระบุอบเจคที่เป็นทั้งหมด(whole) เช่น ตัวอย่างในภาพที่ 11 และภาพที่ 12



ภาพที่ 10 ตัวอย่างการกำหนดควอลลิไฟเออร์



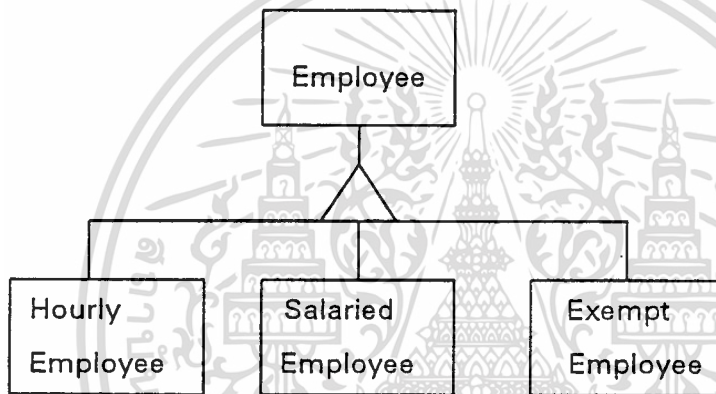
ภาพที่ 11 ตัวอย่างอะกรีเกชัน



ภาพที่ 12 อะกรีเกชันหลายระดับ

2.1.1.4 เจนเนอรัลไลเซชัน (Generalization)

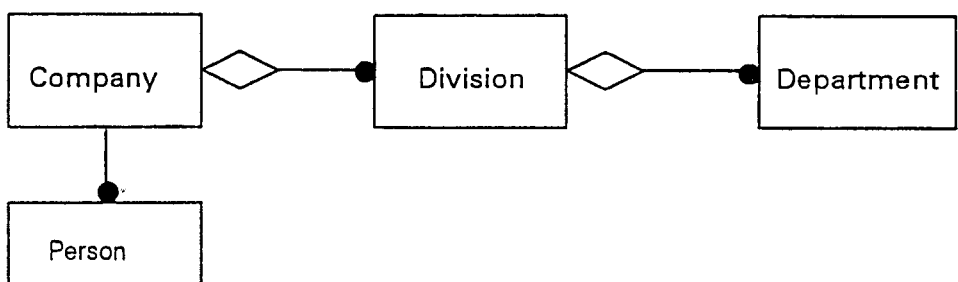
คือความสัมพันธ์ระหว่างคลาสซึ่งคลาสใหม่ได้มาจากการดัดแปลงคลาสที่มีอยู่แล้วโดยเรียกคลาสที่เป็นต้นฉบับว่าซูปเปอร์คลาส (Super Class) คลาสที่ดัดแปลงขึ้นมาใหม่เรียกว่าซับคลาส (Subclass) สัญญลักษณ์ที่ใช้ในแอสโซซิเอชันคือรูปสามเหลี่ยม โดยยอดของสามเหลี่ยมชี้ไปที่ซูปเปอร์คลาสและฐานของสามเหลี่ยมสัมผัสกับแอสโซซิเอชันซึ่งติดต่อกับซับคลาส เช่นตัวอย่างในภาพที่ 14 เจนเนอรัลไลเซชันนี้บางที่เรียกว่าความสัมพันธ์แบบ "a-kind-of" หรือ "is-a" แต่ละซับคลาสจะถ่ายทอดลักษณะทั้งหมดของซูปเปอร์คลาส และเพิ่มแอททริบิวต์, โอเปอเรชันที่เป็นลักษณะเฉพาะของซับคลาสนั้นเข้าไป



ภาพที่ 13 ตัวอย่างเจนเนอรัลไลเซชัน

2.1.1.5 เปรียบเทียบอะกรีเกชันกับแอสโซซิเอชัน

อะกรีเกชันคือรูปแบบพิเศษของแอสโซซิเอชัน ซึ่งมีแนวความคิดที่ไม่ได้แยกอบเจกต์ออกจากกันอย่างเป็นอิสระถ้าอบเจกต์ของอบเจกต์มีความสัมพันธ์กันแบบอบเจกต์หนึ่งเป็นส่วนประกอบของอีกอบเจกต์หนึ่งซึ่งโดยปกติแล้วอบเจกต์ทั้งสองอบเจกต์สามารถที่จะแยกออกจากกันเป็นสองอบเจกต์ที่เป็นอิสระต่อกันได้ อะกรีเกชันสามารถอธิบายความสัมพันธ์ในกรณีดังกล่าวนี้ได้ เช่น ตัวอย่างในภาพที่ 15

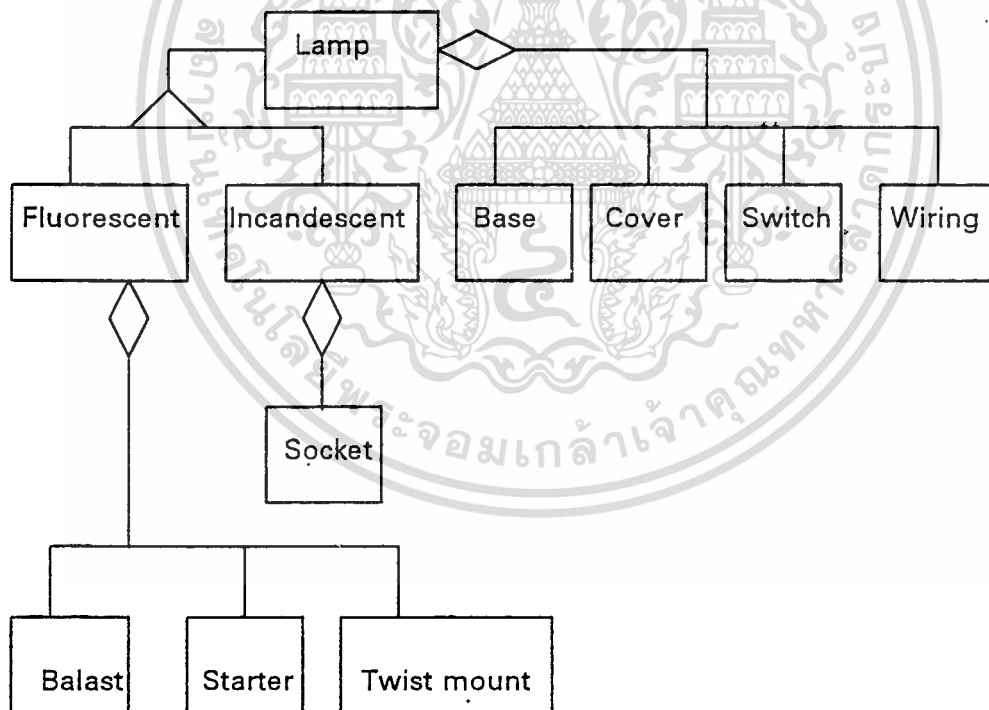


ภาพที่ 14 ตัวอย่างอะกรีเกชัน

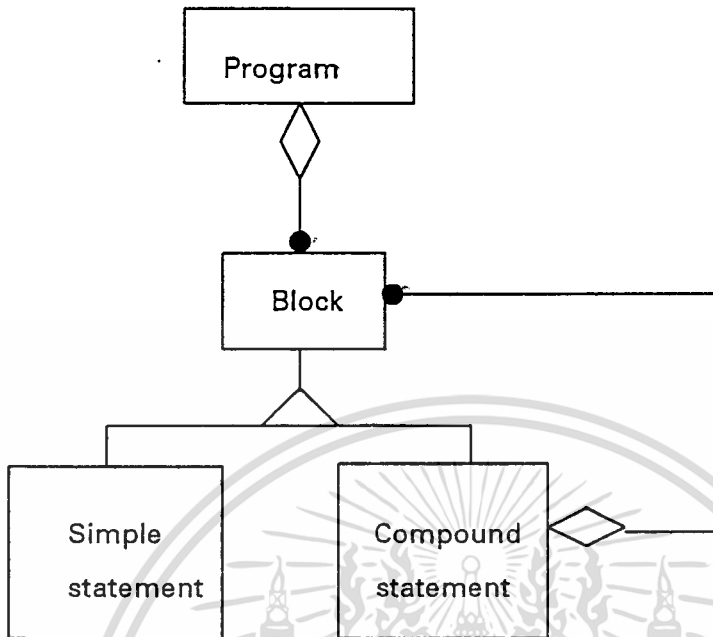
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาค้นคว้าเท่านั้น เมื่อผู้ญาติให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.1.6 เปรียบเทียบอะกรีเกชันกับเจนเนอรัลไลเซชัน

อะกรีเกชันมักจะอ้างถึงออบเจกต์ที่แตกต่างกันแต่มีความสัมพันธ์กันในแง่ที่ออบเจกต์หนึ่งเป็นส่วนประกอบของอีกออบเจกต์หนึ่ง ส่วนเจนเนอรัลไลเซชันมักจะอ้างถึงคลาสโดยออบเจกต์ที่เป็นสมาชิกของชั้นคลาสจะเป็นสมาชิกของซูเปอร์คลาสด้วยเสมอ ดังนั้นบางครั้งจึงเรียกความสัมพันธ์แบบอะกรีเกชันว่า "and" และเรียกความสัมพันธ์แบบเจนเนอรัลไลเซชันว่า "or" อะกรีเกชันสามารถเป็นได้ทั้งแบบอะกรีเกชันคงที่ (Fixed aggregation), อะกรีเกชันเปลี่ยนแปลงได้ (Variable aggregation), หรือรีเคอซีฟอะกรีเกชัน (Recursive aggregation) ตัวอย่างของ Lamp ในภาพที่ 15 คืออะกรีเกชันแบบคงที่ ตัวอย่างของ Company ในภาพที่ 15 คืออะกรีเกชันแบบเปลี่ยนแปลงได้ เพราะบริษัทมีได้หลายฝ่ายและแต่ละฝ่ายมีได้หลายแผนก ตัวอย่าง ของรีเคอซีฟอะกรีเกชันคือสมาชิกของอะกรีเกชันมีชนิดเป็นแบบเดียวกันกับอะกรีเกชันนั้น ตัวอย่างเช่นโปรแกรมคอมพิวเตอร์ในภาพที่ 16



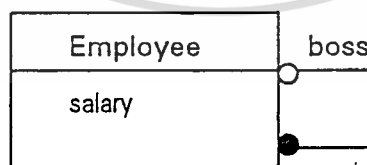
ภาพที่ 15 ตัวอย่างอะกรีเกชันและเจนเนอรัลไลเซชัน



ภาพที่ 16 รีเคอร์ซีฟอะกรีเกชัน

2.1.1.7 ข้อบังคับ (Constraints)

คือข้อกำหนดการกระทำระหว่างเอนทิตี (Entities) ในออบเจกโมเดล เอนทิตีหมายถึง ออบเจกคลาส แอททริบิวต์ ลิงค์ (Link) และแอตทริบิวต์ ข้อบังคับจะกำหนดค่าที่เอนทิตีสามารถจะมีได้ ตัวอย่างเช่น ไม่มีพนักงานคนไหนมีเงินเดือนมากกว่าหัวหน้า(ข้อบังคับระหว่างสองสิ่งในเวลาเดียวกัน) ดังแสดงในภาพที่ 17



{ salary <= boss.salary }

ภาพที่ 17 ตัวอย่างของข้อบังคับ

2.1.2 ไดนามิกโมเดล

โครงสร้างแบบคงที่ทำให้เราสามารถเข้าใจระบบได้ง่ายที่สุดเป็นอันดับแรก ซึ่งโครงสร้างนั้นก็คือโครงสร้างของออบเจกต์และความสัมพันธ์ระหว่างออบเจกต์ ในเวลาเดียวกันเราพบว่ามีการเปลี่ยนแปลงเกิดขึ้นกับออบเจกต์และความสัมพันธ์ระหว่างออบเจกต์ตลอดเวลา ลักษณะของระบบซึ่งเกี่ยวข้องกับเวลาและการเปลี่ยนแปลงคือไดนามิกโมเดล ซึ่งตรงกันข้ามกับความคงที่หรือออบเจกต์โมเดล ไดนามิกโมเดลอธิบายลำดับของโอเปอเรชันที่เกิดขึ้นเพื่อตอบสนองต่อสิ่งเร้าจากภายนอกโดยไม่คำนึงถึงว่าโอเปอเรชันนั้นทำอะไร ทำกับอะไร หรือโอเปอเรชันถูกสร้างขึ้นมาอย่างไร แนวความคิดหลักของไดนามิกโมเดลคืออีเวนต์ (Event) ซึ่งแทนสิ่งเร้าจากภายนอกและสเตต (State) ซึ่งแทนค่าของออบเจกต์

ออบเจกต์โมเดลอธิบายรูปแบบที่เป็นไปได้ของออบเจกต์ แอทริบิวต์ และลิงค์ซึ่งมีอยู่ในระบบ ค่าของแอทริบิวต์และลิงค์ที่อยู่ในออบเจกต์เรียกว่าสเตต ตลอดเวลาออบเจกต์มีการกระตุ้นซึ่งกันและกันผลลัพธ์คือลำดับของการเปลี่ยนแปลงสเตต สิ่งเร้าจากออบเจกต์หนึ่งถึงอีกออบเจกต์หนึ่งคืออีเวนต์ การตอบสนองอีเวนต์ขึ้นกับสเตตของออบเจกต์ขณะกำลังรับอีเวนต์นั้น สเตตไดอะแกรมคือแผนผังเน็ตเวิร์คของสเตตและอีเวนต์ ในทำนองเดียวกับออบเจกต์ไดอะแกรมคือแผนผังเน็ตเวิร์คของคลาสกับความสัมพันธ์ของคลาส ไดนามิกโมเดลประกอบด้วยสเตตไดอะแกรมหลาย สเตตไดอะแกรม โดยสเตตไดอะแกรมหนึ่งสำหรับคลาส ๆ หนึ่ง ซึ่งมีการเปลี่ยนแปลงพฤติกรรมที่สำคัญและแสดงกิจกรรมของระบบทั้งระบบ

2.1.2.1 อีเวนต์

อีเวนต์คือบางสิ่งบางอย่างที่เกิดขึ้น ณ เวลาหนึ่ง อีเวนต์ไม่มีช่วงเวลา ถึงแม้ว่าจะไม่มีอะไรเกิดขึ้นทันทีทันใดก็ตาม อีเวนต์หนึ่งอาจจะเกิดหลังอีเวนต์หนึ่งหรืออีเวนต์สองอีเวนต์อาจจะไม่มีความเกี่ยวข้องกันเลย เช่น เทียวบิน 123 ออกจากซิดนีย์ก่อนหรือหลังเที่ยวบิน 456 ถึงโรม เรียกสองอีเวนต์ที่ไม่เกี่ยวข้องกันนี้ว่าพร้อมกัน (Concurrent) ซึ่งไม่มีผลกระทบซึ่งกันและกันในโมเดลของระบบจะต้องไม่เขียนลำดับของอีเวนต์เพราะว่ามันสามารถเกิดขึ้นในลำดับใดก็ได้ อีเวนต์เป็นวิธีหนึ่งของการส่งผ่านข้อมูลจากออบเจกต์หนึ่งไปยังออบเจกต์อื่นซึ่งไม่เหมือนกับการเรียกใช้ซับรูทีน (Subroutine) ที่มีการส่งค่ากลับคืน ในความเป็นจริงออบเจกต์มีอยู่พร้อมๆ กัน ออบเจกต์ที่ส่งอีเวนต์ถึงอีกออบเจกต์หนึ่งอาจจะคาดหวังที่จะได้รับคำตอบ แต่คำตอบก็คืออีเวนต์อีกอีเวนต์หนึ่งที่แยกกันอยู่ภายใต้การควบคุมของออบเจกต์ที่สองซึ่งอาจจะเลือกส่งหรือไม่ส่งคำตอบก็ได้ ทุกๆ อีเวนต์คือสิ่งที่เกิดขึ้นไม่ซ้ำกันเรารวมกลุ่มของอีเวนต์เป็นอีเวนต์คลาสและตั้งชื่อเพื่อระบุ

โครงสร้างและพฤติกรรม ซึ่งโครงสร้างจะเป็นแบบระดับชั้น (Hierarchical) เช่นเดียวกับโครงสร้างของออบเจกต์คลาส ตัวอย่างเช่น เที่ยวบิน 123 จากชิคาโก และเที่ยวบิน 456 จากโรมคือสมาชิกของอีเวนต์คลาส "เที่ยวบินขึ้น" อีเวนต์คลาสมีแอสริบิวท์ระบุข้อมูลที่นำไปเช่นเดียวกับข้อมูลในออบเจกต์ โดยเขียนแอสริบิวท์ในวงเล็บหลังชื่อของอีเวนต์คลาส ตัวอย่างเช่น

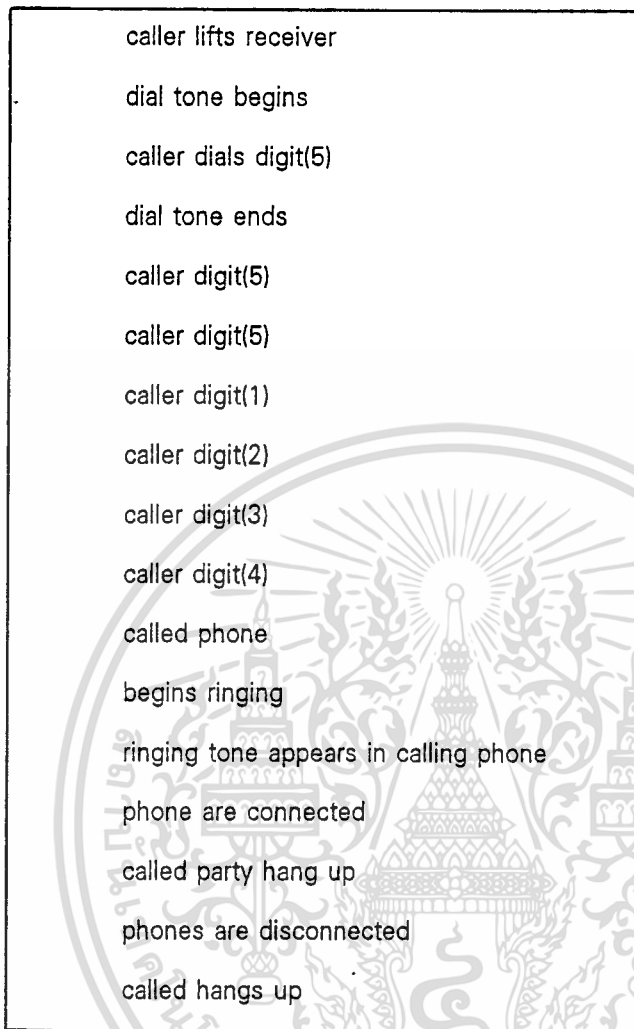
airplane flight departs (airline,flight number,city)

input string entered(text)

phone receiver lifted

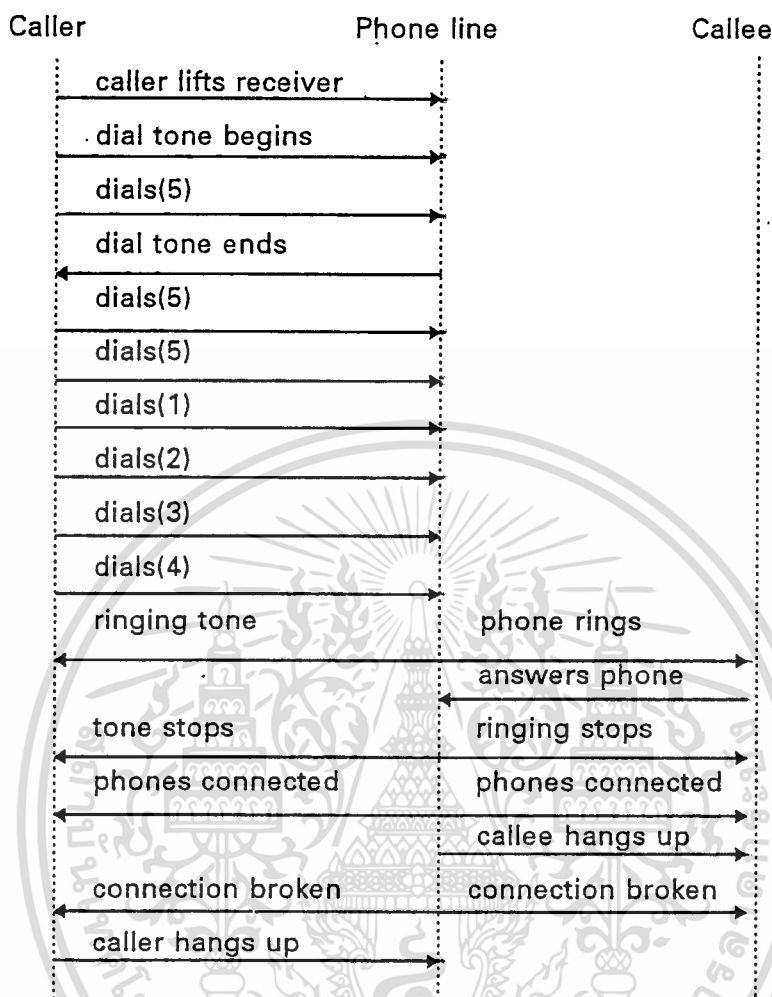
อีเวนต์รวมเงื่อนไขที่ผิดพลาดได้ด้วยเช่นเดียวกันกับอีเวนต์ที่เกิดขึ้นแบบปกติ ตัวอย่างเช่น การจราจรติดขัด อีเวนต์ที่ผิดพลาดไม่ได้มีอะไรแตกต่างไปจากอีเวนต์ปกติเพียงแต่บอกว่ามันคือข้อผิดพลาด

ภาพเหตุการณ์ (Senarios) คือลำดับของอีเวนต์ซึ่งเกิดขึ้นในเวลาทีระบบกำลังทำงานขอบเขตของภาพเหตุการณ์สามารถเปลี่ยนแปลงได้ ซึ่งอาจจะรวมทุกอีเวนต์ในระบบหรืออาจจะรวมเฉพาะบางออบเจกต์ในระบบ ภาพเหตุการณ์สามารถเป็นรายการในอดีตของการทำงานของระบบ หรือแนวความคิดที่ได้จากประสบการณ์ในการทำงาน ตัวอย่างภาพเหตุการณ์สำหรับการใช้โทรศัพท์ในภาพที่ 18 ซึ่งแสดงเฉพาะอีเวนต์ที่มีผลกระทบกับ phone line เท่านั้น



ภาพที่ 18 ภาพเหตุการณ์สำหรับ phone call

แต่ละอีเวนต์ส่งผ่านข้อมูลจากออบเจกต์หนึ่งไปยังออบเจกต์อื่น ตัวอย่างเช่น dial tone begins ส่งสัญญาณจากโทรศัพท์ไปยังผู้เรียก ขั้นตอนต่อไปหลังจากเขียนภาพเหตุการณ์คือการกำหนดออบเจกต์ที่เป็นผู้รับและผู้ส่งของแต่ละอีเวนต์ ลำดับของอีเวนต์และออบเจกต์สามารถแสดงได้ทั้งสองแบบในภาพเหตุการณ์ที่ขยายเพิ่มขึ้นเรียกว่าแผนผังทางเดินของเหตุการณ์ (Event Trace Diagram) ซึ่งแสดงออบเจกต์แต่ละออบเจกต์ด้วยเส้นดิ่งและแต่ละอีเวนต์ด้วยลูกศรในแนวนอนจากออบเจกต์ผู้ส่งถึงออบเจกต์ผู้รับ เริ่มจากบรรทัดแรกจนถึงบรรทัดสุดท้ายตามระยะเวลา โดยช่องว่างระหว่างลูกศรไม่มีความหมายอะไรเพียงแต่เป็นลำดับของอีเวนต์เท่านั้น ตัวอย่างทางเดินของเหตุการณ์สำหรับ phone call ดังแสดงในภาพที่ 19



ภาพที่ 19 ทางเดินของเหตุการณ์สำหรับ phone call

2.1.2.2 สเตท

สเตทคือนามธรรมของค่าแอสรีบิวท์และลิ่งค์ของออบเจคซึ่งรวมค่าต่างๆ เข้าด้วยกันเป็นสเตท โดยรวมค่าเป็นกลุ่มตามคุณสมบัติที่มีอิทธิพลกับพฤติกรรมส่วนใหญ่ของออบเจค ตัวอย่างเช่น สเตทของธนาคารคือสามารถชำระหนี้ได้หรือไม่สามารถชำระหนี้ได้ซึ่งขึ้นกับทรัพย์สินของธนาคาร สเตทกำหนดการตอบสนองของออบเจคต่ออีเวนท์ที่เข้ามายังออบเจคซึ่งการตอบสนองอาจจะเปลี่ยนแปลงตามค่าของแอสรีบิวท์ แต่การตอบสนองจะเป็นแบบเดียวกันสำหรับทุกๆ ค่าของข้อมูลในสเตทเดียวกันและอาจจะแตกต่างกันสำหรับค่าที่อยู่ในสเตทที่ต่างกัน การตอบสนองของออบเจคต่ออีเวนท์อาจจะรวมแอสรีบิวท์หรือการเปลี่ยนสเตทโดยออบเจค ตัวอย่างเช่น ถ้าตัวเลขถูกหมุนในสเตท Dial tone และออบเจค phone line ออกจากสเตท Dial เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

tone เข้าสู่สแตต Dialing ถ้าออบเจค receiver เข้ามาแทนที่ ออบเจค phone line จะตายไป และเข้าสู่สแตต Idle สแตตอยู่ตรงกลางระหว่างสองอีเวนท์ซึ่งเป็นอีเวนท์ที่ออบเจคได้รับ อีเวนท์แทนจุดในเวลานั้น สแตตแทนช่วงระยะเวลา ตัวอย่างเช่น หลังจากยกหูฟังและก่อนหมุนตัวเลขตัวแรก phone line จะอยู่ในสแตต Dial tone สแตตของออบเจคขึ้นกับลำดับของอีเวนท์ที่ออบเจคได้รับ ในเกือบทุกกรณีอีเวนท์ในอดีตจะถูกซ่อนไว้โดยอีเวนท์ที่ตามมา ตัวอย่างเช่น อีเวนท์ที่เกิดขึ้นก่อนวางหูฟังไม่มีผลกับพฤติกรรมในอนาคต สแตต Idle ลืมอีเวนท์ที่ได้รับก่อนอีเวนท์วางหู สแตตที่มีช่วงระยะเวลามักเกิดขึ้นในช่วงเวลาหนึ่ง สแตตเกี่ยวข้องกับกรกระทำที่ต่อเนื่อง เช่น การทำให้เสียงโทรศัพท์ดัง หรือสแตตเกี่ยวข้องกับแอคติวิตี (Activity) ที่ต้องใช้เวลาทำให้เสร็จสมบูรณ์ เช่น เทียบบินจากชิคาโกถึงซานฟรานซิสโก อีเวนท์และสแตตเป็นสิ่งที่คู่กันอีเวนท์หนึ่งจะแบ่งแยกสองสแตตและสแตตหนึ่งจะแบ่งแยกสองอีเวนท์ สแตตมักเกี่ยวข้องกับค่าของออบเจคซึ่งตรงกับเงื่อนไขใดเงื่อนไขหนึ่ง ค่าของแอทริบิวต์แต่ละค่ากำหนดสแตตที่แตกต่างกัน

เงื่อนไข (Condition) คือบูลีนฟังก์ชัน (Boolean Function) ของค่าออบเจค เช่น 'อุณหภูมิต่ำกว่าจุดแข็ง' เงื่อนไขจะเป็นจริงตามระยะเวลา เช่น "อุณหภูมิต่ำกว่าจุดแข็งจาก 15 พ.ย 1921 จนกระทั่งถึง 3 มี.ค 1922" การแยกเงื่อนไขจากอีเวนท์ซึ่งไม่มีช่วงของระยะเวลาเป็นเรื่องที่สำคัญ สแตตสามารถกำหนดในรูปของเงื่อนไขได้ ในทางกลับกันสแตตคือเงื่อนไข เงื่อนไขสามารถใช้เป็นเสมือนผู้คุมทรานซิชัน (Transition) ทรานซิชันที่มีผู้คุมนี้จะเกิดขึ้นได้ก็ต่อเมื่อมีอีเวนท์เกิดขึ้น และอีเวนท์เกิดขึ้นได้ก็เพราะเงื่อนไขเป็นจริง เช่น "เมื่อคุณออกจากบ้านในตอนเช้า ถ้าฝนตก (เงื่อนไข) ดังนั้นคุณกางร่ม (สแตตถัดไป)" เงื่อนไขที่คุมทรานซิชันแสดงในรูปของนิพจน์บูลีน ภายในวงเล็บตามหลังอีเวนท์

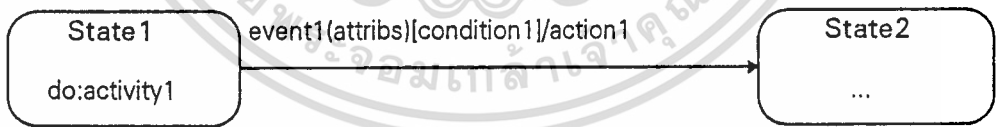
2.1.2.3 โอเปอเรชัน (Operation)

สแตตไดอะแกรมจะมีประโยชน์เพียงเล็กน้อยเท่านั้นถ้าใช้อธิบายเฉพาะรูปแบบของอีเวนท์ การอธิบายพฤติกรรมของออบเจคต้องกำหนดว่าออบเจคตอบสนองต่ออีเวนท์อย่างไร การที่โอเปอเรชันถูกกระทำก็เพื่อตอบสนองต่อสแตตหรืออีเวนท์ แอคติวิตีคือโอเปอเรชันซึ่งต้องใช้เวลาเพื่อให้การกระทำสมบูรณ์ แอคติวิตีเกี่ยวข้องกับสแตต แอคติวิตีรวมโอเปอเรชันทั้งแบบต่อเนื่องและแบบเรียงลำดับ ตัวอย่างโอเปอเรชันแบบต่อเนื่องเช่น ภาพบนจอโทรทัศน์ สัญลักษณ์ของแอคติวิตีคือการเขียนชื่อของแอคติวิตีตามหลัง "do:" ภายในสัญลักษณ์รูปกล่องสี่เหลี่ยมของสแตต เช่น "do: A" โดยแอคติวิตี A จะเริ่มทำงานเมื่อเข้าสู่สแตตและจะสิ้นสุดการทำงานเมื่อออกจากสแตต สแตตอาจจะควบคุมลำดับของแอคติวิตี เช่น การเคลื่อนที่ของหุ่นยนต์ซึ่งจะกระทำจนเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่นอนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กว่าจะสมบูรณ์ หรือจนกว่าจะถูกเบรคทำให้ต้องหยุดการกระทำก่อนเวลา แอคชัน(Action) คือ โอเปอเรชันชั่วขณะ แอคชันเกี่ยวข้องกับอีเวนต์ แอคชันแทนโอเปอเรชันซึ่งช่วงระยะเวลาไม่มีความสำคัญในสเตตไดอะแกรม ความจริงแล้วโอเปอเรชันชั่วขณะจริงๆ ไม่มีแต่จำลองโอเปอเรชันเป็นแอคชันนี้เพื่อชี้ว่าเราไม่สนใจช่วงระยะเวลา ถ้าหากว่าเราสนใจช่วงระยะเวลาเราจะจำลองโอเปอเรชันเป็นแอคติวิตี้ซึ่งมีอีเวนต์เริ่มต้น อีเวนต์จบ และอีเวนต์ภายในช่วงระยะเวลา เขียนชื่อของแอคชันตามหลังชื่อของอีเวนต์ โดยหน้าชื่อของแอคชันมีเครื่องหมาย "/"


2.1.2.4 สเตตไดอะแกรม

สเตตไดอะแกรมเกี่ยวข้องกับอีเวนต์และสเตต อีเวนต์ทำให้เกิดการเปลี่ยนสเตต ซึ่งเรียกการเปลี่ยนสเตตว่าทรานซิชัน (Transition) สเตตไดอะแกรมคือกราฟที่มีสเตตเป็นโหนด (Node) และเส้นที่ลากต่อระหว่างโหนดคือทรานซิชัน โดยเขียนชื่อของอีเวนต์ที่สเตตนั้นได้รับกำกับบนทรานซิชัน สเตตใช้สัญลักษณ์เป็นรูปสี่เหลี่ยมมุมมน และเขียนชื่อของสเตตภายในรูปสี่เหลี่ยมหรือจะไม่เขียนก็ได้โดยเขียนด้วยตัวอักษรทึบ เส้นลูกศรลากออกจากสเตตหนึ่งไปยังสเตตเป้าหมายแทนทรานซิชัน เขียนชื่อของอีเวนต์บนทรานซิชัน ซึ่งอาจจะมีแอทริบิวต์เขียนอยู่ภายในวงเล็บตามหลังหรือไม่ก็ได้ ถ้ามีเงื่อนไขจะเขียนภายในเครื่องหมายวงเล็บใหญ่ต่อจากชื่อของอีเวนต์ และเขียนชื่อของแอคชันต่อหลังโดยมีเครื่องหมาย "/" นำหน้าชื่อแอคชัน ส่วนชื่อของแอคติวิตี้จะเขียนอยู่ในรูปกล่องสเตตหลังคำสั่ง "do:" ดังแสดงในภาพที่ 20



ภาพที่ 20 สเตตไดอะแกรม

สเตตไดอะแกรมกำหนดลำดับของสเตตซึ่งเป็นผลมาจากลำดับของอีเวนต์ ลำดับของอีเวนต์ตรงกับเส้นทางบนกราฟ สเตตไดอะแกรมอธิบายพฤติกรรมของแต่ละออบเจกต์คลาส เมื่อทุกสมาชิกของคลาสมีพฤติกรรมแบบเดียวกัน (โดยคำจำกัดความ) สมาชิกจะใช้สเตตไดอะแกรมร่วมกันเช่นเดียวกับที่ใช้ลักษณะของคลาสร่วมกัน เพียงแต่ว่าแต่ละออบเจกต์จะมีสเตตเป็นของตนเองและลำดับของอีเวนต์ที่ออบเจกต์ได้รับจะแตกต่างกัน แต่ละออบเจกต์จะเป็นอิสระต่อกันและมีการกระทำในตำแหน่งของตนเอง สเตตไดอะแกรมสามารถแทนวงรอบแบบสั้นๆ (One-shot life เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

cycle) หรือแบบต่อเนื่องได้ ไดอะแกรมแบบวงรอบสั้นๆ จะมีสเตทเริ่มต้น (Initial) และสเตทสุดท้าย (Final) สเตทเริ่มต้นสร้างขอบเขตแต่สเตทสุดท้ายทำลายขอบเขต สัญลักษณ์ของสเตทเริ่มต้นเป็นวงกลมทึบ ซึ่งอาจจะกำหนดชื่อเพื่อระบุเงื่อนไขการเริ่มต้นที่แตกต่างกัน สเตทสุดท้ายใช้สัญลักษณ์รูปลูกตาวัว (Bull's eye ) โดยจะกำหนดชื่อเพื่อให้สามารถที่จะแยกความแตกต่างของเงื่อนไขการจบได้ เราสามารถที่จะมองไดอะแกรมของวงรอบสั้นๆ เป็นขั้นบันไดของสเตทไดอะแกรมได้ ซึ่งสามารถจะอ้างถึงได้หลายแห่งในไดอะแกรม ไดนามิกโมเดลคือสเตทไดอะแกรมหลายๆ ไดอะแกรมรวมกัน ออบเจกต์โมเดลแสดงโครงสร้างของระบบแบบคงที่ ขณะที่ไดนามิกโมเดลแสดงการควบคุมโครงสร้างของระบบ สเตทไดอะแกรมเหมือนกับออบเจกต์คลาสที่เป็นรูปแบบอธิบายรายละเอียดทั้งหมดและภาพเหตุการณ์เหมือนกับเป็นสมาชิกของไดนามิกโมเดล

2.1.3 ฟังก์ชันนัลโมเดล

ฟังก์ชันนัลโมเดลแสดงผลลัพธ์การคำนวณโดยไม่ได้กำหนดว่าคำนวณอย่างไรและเมื่อไร ฟังก์ชันนัลโมเดลกำหนดความหมายของโอเปอเรชันในออบเจกต์โมเดล และแอคชันในไดนามิกโมเดล รวมทั้งข้อบังคับในออบเจกต์โมเดล โปรแกรมแบบไม่มีการโต้ตอบ เช่น คอมไพเลอร์มีไดนามิกโมเดลเพียงเล็กน้อยเพราะจุดมุ่งหมายของโปรแกรมคือฟังก์ชันการคำนวณซึ่งฟังก์ชันนัลโมเดลคือโมเดลหลักสำหรับโปรแกรมเช่นนี้ ออบเจกต์โมเดลสำคัญสำหรับปัญหาแบบโครงสร้างข้อมูล โปรแกรมแบบที่มีการโต้ตอบหลายโปรแกรมก็มีฟังก์ชันนัลโมเดลที่สำคัญด้วยเช่นกัน ตรงกันข้ามกับฐานข้อมูลมักจะมีฟังก์ชันนัลโมเดลเพียงเล็กน้อยเมื่อจุดประสงค์คือการเก็บและการจัดข้อมูลไม่ใช่การเคลื่อนย้ายข้อมูล ฟังก์ชันนัลโมเดลประกอบด้วยดาต้าโฟลไดอะแกรม (Data Flow Diagram) หลายไดอะแกรมซึ่งกำหนดความหมายของโอเปอเรชันและข้อบังคับ ดาต้าโฟลไดอะแกรมแสดงความสัมพันธ์แห่งการกระทำของค่าที่คำนวณโดยระบบรวมทั้งค่าของอินพุต ค่าของผลลัพธ์ และที่เก็บข้อมูลภายในระบบ ดาต้าโฟลไดอะแกรมคือกราฟที่แสดงดาต้าโฟลจากแหล่งของข้อมูลในออบเจกต์ผ่านทางโพรเซส (Process) โพรเซสจะส่งข้อมูลต่อไปยังจุดหมายปลายทางในออบเจกต์อื่นๆ ดาต้าโฟลไดอะแกรมไม่ได้แสดงข้อมูลข่าวสารการควบคุมเช่นเวลาที่โพรเซสทำงานหรือการตัดสินใจเลือกดาต้าโฟล การตัดสินใจเลือกนี้อยู่ในไดนามิกโมเดล ดาต้าโฟลไดอะแกรมไม่ได้แสดงการจัดข้อมูลในออบเจกต์ การจัดข้อมูลนี้อยู่ในออบเจกต์โมเดล ดาต้าโฟลไดอะแกรมประกอบด้วย

- โพรเซส (Process) ซึ่งทำการส่งผ่านข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

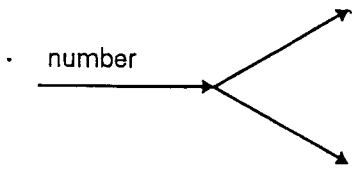
- ดาต้าโฟล (Data Flow) ซึ่งเคลื่อนย้ายข้อมูล
- ผู้กระทำ (actor) ออบเจกต์ซึ่งสร้างและใช้ข้อมูล
- แหล่งเก็บข้อมูล (data store) ออบเจกต์ซึ่งเก็บข้อมูล

2.1.3.1 โพรเซส (Process)

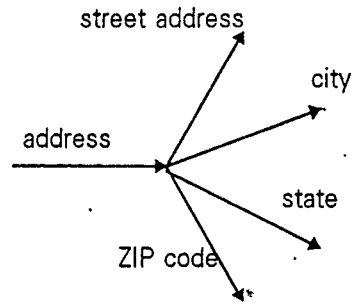
โพรเซสจะส่งผ่านค่าของข้อมูล โพรเซสในระดับต่ำสุดก็คือฟังก์ชันแท้ๆ แผนผังของดาต้าโฟลทั้งหมดคือโพรเซสในระดับสูง ฟังก์ชันโมเดลระบุเฉพาะเส้นทางของการกระทำที่เป็นไปได้เท่านั้นแต่ไม่ได้แสดงว่าเส้นทางใดจะเกิดการกระทำขึ้น ผลลัพธ์ของโพรเซสขึ้นกับพฤติกรรมของระบบตามที่กำหนดไว้ในไดนามิกโมเดล สัญลักษณ์ของโพรเซสคือวงรีรูปไข่โดยเขียนชื่อของโพรเซสไว้ภายใน แต่ละโพรเซสมีลูกศรที่เป็นข้อมูลอินพุตและเอาต์พุตเป็นจำนวนที่แน่นอน ซึ่งแต่ละลูกศรมีค่าของข้อมูลตามชนิดที่กำหนดให้ เราสามารถที่จะกำหนดชื่อของอินพุตและเอาต์พุตเพื่อระบุบทบาทในการคำนวณได้ โพรเซสสามารถมีผลลัพธ์ได้มากกว่าหนึ่งผลลัพธ์ แผนผังแสดงเฉพาะรูปแบบของอินพุตและเอาต์พุต ต้องกำหนดการคำนวณค่าของผลลัพธ์จากค่าของอินพุตด้วย โพรเซสในระดับสูงสามารถขยายออกเป็นดาต้าโฟลไดอะแกรมได้มากเท่าที่ซับซ้อน สามารถแตกออกเป็นซับซ้อนในระดับต่ำ โพรเซสซึ่งแยกย่อยออกไปอีกไม่ได้แล้วต้องอธิบายด้วยภาษาที่เข้าใจได้ง่ายหรือเป็นสมการการคำนวณ โพรเซสสร้างขึ้นเป็นเมธอด (หรือส่วนหนึ่งของเมธอด) ของโอเปอเรชันในออบเจกต์

2.1.3.2 ดาต้าโฟล (Data Flow)

ดาต้าโฟลเชื่อมต่อระหว่างผลลัพธ์ของออบเจกต์ หรือโพรเซสกับอินพุตจากออบเจกต์หรือโพรเซสอื่น ดาต้าโฟลแทนค่าของข้อมูลที่เกิดจากการคำนวณ ค่าของข้อมูลไม่ได้เปลี่ยนแปลงไปเพราะดาต้าโฟล ดาต้าโฟลแทนด้วยเส้นลูกศรซึ่งอยู่ระหว่างผู้สร้างและผู้ใช้ค่าของข้อมูล โดยบนลูกศรจะมีคำอธิบายเกี่ยวกับค่าของข้อมูล ซึ่งปกติจะเขียนเป็นชื่อหรือเป็นชนิดของข้อมูล ค่าเดียวกันสามารถส่งไปได้หลายที่ ซึ่งจะใช้สัญลักษณ์เป็นช่อมที่มีหลายหัวลูกศรรวมกัน โดยที่ลูกศรผลลัพธ์ไม่ได้เขียนชื่อกำกับเพราะว่าใช้แทนค่าอินพุตอันเดียวกัน ดังแสดงตัวอย่างในภาพที่ 21 ในบางครั้งค่าของข้อมูลแบบอะกรีเกชันแตกออกเป็นส่วนประกอบซึ่งแต่ละส่วนประกอบถูกส่งไปยังโพรเซสที่แตกต่างกัน ในกรณีนี้จะแสดงด้วยสัญลักษณ์ช่อมซึ่งแต่ละลูกศรจะเขียนชื่อของส่วนประกอบกำกับ ส่วนประกอบที่รวมกันเป็นค่าอะกรีเกชันคือค่าที่อยู่ด้านตรงข้าม ดังแสดงตัวอย่างในภาพที่ 22



ภาพที่ 21 ตัวอย่างดาต้าไฟล์



ภาพที่ 22 ตัวอย่างดาต้าไฟล์

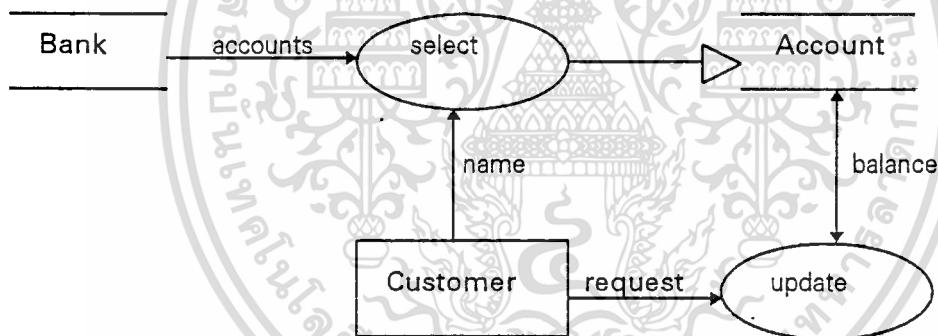
แต่ละดาต้าไฟล์จะแทนค่าซึ่งได้จากการคำนวณที่จุดใดจุดหนึ่ง ซึ่งในความเป็นจริงค่านี้ไม่จำเป็นต้องมีความสำคัญ ไฟล์ที่อยู่ตรงส่วนขอบของดาต้าไฟล์ไดอะแกรมคืออินพุตและเอาต์พุต ไฟล์เหล่านี้อาจจะไม่ถูกเชื่อมต่อ (ถ้าไดอะแกรมเป็นเพียงส่วนหนึ่งของระบบ) หรืออาจจะเชื่อมต่อกับออบเจกต์

ผู้กระทำ (Actor) คือออบเจกต์ที่มีการกระทำที่ขับเคลื่อนดาต้าไฟล์โดยการสร้างและการใช้ค่า ผู้กระทำติดต่อกับอินพุตและเอาต์พุตของดาต้าไฟล์ผู้กระทำวางอยู่บนขอบนอกของกราฟดาต้าไฟล์แต่จบการไหลของข้อมูลด้วยแหล่งข้อมูลและเก็บข้อมูลไว้ ดังนั้นในบางครั้งจึงเรียกผู้กระทำนี้ว่าเทอร์มินเนเตอร์ (Terminators) การกระทำของผู้กระทำอยู่นอกขอบเขตของดาต้าไฟล์ไดอะแกรมแต่ควรจะเป็นส่วนหนึ่งของไดนามิกโมเดล ผู้กระทำแทนด้วยรูปสามเหลี่ยมเพื่อแสดงว่าเป็นออบเจกต์ ลูกศรระหว่างผู้กระทำและไดอะแกรมคืออินพุตและเอาต์พุตของไดอะแกรม

แหล่งเก็บข้อมูล (Data Store) คือออบเจกต์ที่ไม่มีการกระทำภายในดาต้าไฟล์ไดอะแกรมแต่เก็บข้อมูลสำหรับการกระทำล่าสุด แหล่งเก็บข้อมูลไม่เหมือนกับผู้กระทำเพราะแหล่งเก็บข้อมูลไม่ได้สร้างโอเปอเรชันใดๆ บนแหล่งเก็บข้อมูลเพียงแต่ตอบรับที่จะเก็บข้อมูล แหล่งเก็บข้อมูลเขียนแทนด้วยเส้นคู่ขนานที่เขียนชื่อแหล่งเก็บข้อมูลไว้ภายใน ลูกศรเข้าไปที่แหล่งเก็บข้อมูลระบุข้อมูลข่าวสารหรือโอเปอเรชันซึ่งปรับปรุงแก้ไขข้อมูลที่เก็บไว้ในแหล่งเก็บข้อมูล ได้แก่ การเพิ่ม การเปลี่ยนแปลงค่า หรือการลบค่าออก ลูกศรออกจากแหล่งเก็บข้อมูลระบุข้อมูลข่าวสารที่ถูกดึงออกมาจากแหล่งเก็บข้อมูล ซึ่งอาจจะเป็นค่าทั้งหมดหรือค่าเพียงบางส่วน โครงสร้างของออบเจกต์จะต้องอธิบายในออบเจกต์โมเดล รวมทั้งต้องอธิบายโอเปอเรชันการแก้ไขและการกระทำ

ทั้งผู้กระทำและแหล่งเก็บข้อมูลคือออบเจกต์ แต่ที่เรากำหนดให้แตกต่างกันก็เพราะพฤติกรรมและการใช้มีความแตกต่างกันถึงแม้ว่าในภาษาเชิงวัตถุจะสร้างเป็นออบเจกต์เหมือนกันหรือแหล่งเก็บข้อมูลอาจจะสร้างเป็นเพิ่มข้อมูลและผู้กระทำเป็นอุปกรณ์ภายนอก ดาต้าไฟล์ดาต้าไฟล์เป็นออบเจกต์ด้วย ถึงแม้ว่าในหลายกรณีที่ดาต้าไฟล์คือค่าแท้ๆ เช่น เลขจำนวนเต็ม (ในภาษาเชิงวัตถุออบเจกต์และค่าแท้ๆ มักจะสร้างเหมือนกัน)

มุมมองที่แตกต่างกันระหว่างออบเจกต์คือออบเจกต์มีข้อมูลได้ค่าเดียวแต่แหล่งเก็บข้อมูลมีข้อมูลได้มากกว่าหนึ่งค่า ตัวอย่างเช่นในภาพที่ 22 นำชื่อของลูกค้าไปเลือกบัญชีจากธนาคารได้ผลลัพธ์เป็นออบเจกต์บัญชีนั้นซึ่งใช้เป็นแหล่งเก็บข้อมูลสำหรับโอเปอเรชันการแก้ไข ดาต้าไฟล์ซึ่งสร้างออบเจกต์และจะใช้เป็นเป้าหมายของโอเปอเรชันอื่นระบุโดยใช้รูปสามเหลี่ยมกลวงที่ปลายของดาต้าไฟล์ ตรงกันข้ามโอเปอเรชันการแก้ไขซึ่งปรับยอดบัญชีในออบเจกต์บัญชีระบุโดยใช้หัวลูกศรเล็กๆ รูปสามเหลี่ยมกลวงที่ระบุค่าผลลัพธ์ของดาต้าไฟล์เป็นออบเจกต์โดยปกติแล้วก็คือแหล่งเก็บข้อมูล



ภาพที่ 23 ตัวอย่างดาต้าไฟล์ไดอะแกรม

การควบคุมไหลของข้อมูล (Control Flow) ดาต้าไฟล์ไดอะแกรมแสดงเส้นทางการคำนวณที่เป็นไปได้สำหรับค่าข้อมูล โดยไม่แสดงว่าเส้นทางใดถูกกระทำและลำดับของการกระทำเป็นอย่างไร การตัดสินใจและการเรียงลำดับคือการควบคุมซึ่งเป็นส่วนหนึ่งของไดนามิกโมเดล การตัดสินใจนั้นมีผลกับฟังก์ชันว่ามีหนึ่งฟังก์ชันหรือมากกว่าหนึ่งฟังก์ชันที่จะถูกกระทำ ไม่ใช่เป็นการให้ค่ากับฟังก์ชัน ถึงแม้ว่าการตัดสินใจเลือกฟังก์ชันนี้จะได้ไม่มีค่าอินพุตแต่ในบางครั้งก็มีประโยชน์ที่จะรวมการตัดสินใจเลือกฟังก์ชันนี้ไว้ในฟังก์ชันโมเดลเพื่อว่าจะได้ไม่ลืมและยังเป็นการแสดงข้อมูลที่ขึ้นต่อกันอีกด้วยวิธีทำคือรวมการควบคุมดาต้าไฟล์นี้เข้าไปไว้ดาต้าไฟล์ไดอะแกรมด้วย การควบคุมดาต้าไฟล์คือค่าบูลีน (Boolean) ซึ่งจะมีผลกับโพเซสว่าว่าโพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ย่อยลงไปได้อีก (Leaf Process) ก็คือโอบเจกต์ในออบเจกต์ ออบเจกต์โมเดลแสดงผู้กระทำซึ่งก็คือออบเจกต์นั่นเอง แต่ละโพรเซสถูกสร้างขึ้นมาจากเมธอด ไดนามิกโมเดลแสดงลำดับของโอบเจกต์ที่ถูกกระทำ เมธอดได้มาจากทั้งสามโมเดลรวมกันแต่ฟังก์ชันนัลโมเดลเน้นแนวทางเมธอด โพรเซสในฟังก์ชันนัลโมเดลตรงกับโอบเจกต์ในออบเจกต์โมเดล การจับคู่กันในเน็ตเวิร์คโพรเซสระดับบนสุดจะตรงกับโอบเจกต์ของออบเจกต์ที่ซับซ้อนและโพรเซสระดับต่ำสุดจะตรงกับออบเจกต์ขั้นพื้นฐานซึ่งเป็นส่วนหนึ่งของออบเจกต์ที่ซับซ้อน บางครั้งโพรเซสเดียวอาจจะตรงกับหลายโอบเจกต์ และบางครั้งโอบเจกต์เดียวจะตรงกับหลายโพรเซส โพรเซสในฟังก์ชันนัลโมเดลแสดงออบเจกต์ซึ่งเกี่ยวข้องกันโดยฟังก์ชัน บ่อยครั้งที่อินพุตหนึ่งของโพรเซสสามารถถูกกำหนดเป็นออบเจกต์เป้าหมาย และอินพุตส่วนที่เหลือกำหนดเป็นพารามิเตอร์ของโอบเจกต์เป้าหมายคือไคลเอ็นท์ (Client) ของออบเจกต์อื่นซึ่งเรียกว่าซัพพลายเออร์ (Supplier) เพราะออบเจกต์เป้าหมายใช้ออบเจกต์อื่นทำโอบเจกต์ไคลเอ็นท์ถูกสร้างในเทอมของซัพพลายเออร์และขึ้นกับซัพพลายเออร์ โดยปกติโพรเซสจะพัฒนาไปเป็นเมธอด ถ้าอินพุตและเอาต์พุตเป็นออบเจกต์ในคลาสเดียวกันดังนั้นออบเจกต์คือเป้าหมายและอินพุตที่เหลือคืออาร์กิวเมนต์ ถ้าผลลัพธ์ของโพรเซสคือแหล่งเก็บข้อมูล แหล่งเก็บข้อมูลคือเป้าหมาย ถ้าอินพุตหรือผลลัพธ์คือผู้กระทำ ผู้กระทำคือออบเจกต์ที่กำหนดขึ้นในออบเจกต์โมเดล ดาต้าโฟลว์เข้าไปยังผู้กระทำหรือออกจากผู้กระทำใช้แทนโอบเจกต์ ค่าของดาต้าโฟลว์คืออาร์กิวเมนต์ของโอบเจกต์หรือเป็นผลลัพธ์ของโอบเจกต์ ผู้กระทำคือออบเจกต์ที่มีการเคลื่อนไหวด้วยตนเอง ในฟังก์ชันนัลโมเดลไม่ได้ระบุเมื่อผู้กระทำกระทำแต่กำหนดในไดนามิกโมเดล แหล่งเก็บข้อมูลก็คือออบเจกต์ในออบเจกต์โมเดลเช่นกัน หรือเป็นส่วนหนึ่งของออบเจกต์ เช่น แอทริบิวต์ ดาต้าโฟลว์ที่เข้าไปยังแหล่งเก็บข้อมูลคือโอบเจกต์การแก้ไข ดาต้าโฟลว์ที่ออกจากแหล่งเก็บข้อมูลคือโอบเจกต์คิวรีซึ่งไม่มีผลกระทบต่อแหล่งเก็บข้อมูล ดาต้าโฟลว์คือค่าในออบเจกต์โมเดล ดาต้าโฟลว์หลายดาต้าโฟลว์คือค่าแบบธรรมดา เช่น จำนวน ข้อความ เป็นต้น ค่าต่างๆสามารถถูกจำลองเป็นคลาสได้ และพัฒนาเป็นออบเจกต์ในภาษาส่วนมากแต่ไม่เป็นเอกลักษณ์ (Identity) ค่าต่างๆไม่ใช่เป็นภาษาอะไรซึ่งจะเก็บค่าที่สามารถเปลี่ยนแปลงได้แต่เป็นเพียงค่าเท่านั้น เพราะฉะนั้นค่าต่างๆไม่มีสเตตและไม่มีไดนามิกโมเดล โอบเจกต์บนค่าต่างๆ ให้ผลลัพธ์เป็นค่าต่างๆ และไม่มีผลกระทบต่อ ตัวอย่างเช่นโอบเจกต์การคำนวณทางคณิตศาสตร์

ฟังก์ชันนัลโมเดล: ออบเจกต์โมเดลแสดงโครงสร้างของผู้กระทำ, แหล่งเก็บข้อมูล และดาต้าโฟลว์ในฟังก์ชันนัลโมเดล ส่วนไดนามิกโมเดลแสดงลำดับของการทำโพรเซส

ออบเจกต์โมเดล: ฟังก์ชันนำโมเดลแสดงโอเปอเรชันบนคลาสและอาร์กิวเมนต์ของแต่ละโอเปอเรชัน ดังนั้นฟังก์ชันนำโมเดลแสดงความสัมพันธ์ของไคลเอ็นท์และเซิร์ฟเวอร์ในกลุ่มคลาส ไดนามิกโมเดลแสดงสแตทของออบเจกต์และโอเปอเรชันซึ่งกระทำเมื่อได้รับอีเวนต์และเปลี่ยนสแตท

ไดนามิกโมเดล: ฟังก์ชันนำโมเดลแสดงคำจำกัดความของแอคชันและแอคติวิตี้ที่แยกย่อยลงไปอีกไม่ได้ซึ่งไม่ได้กำหนดไว้ในไดนามิกโมเดล ออบเจกต์แสดงว่าอะไรเปลี่ยนสแตทและอะไรได้รับโอเปอเรชัน

2.2 โปรแกรมภาษาสโมลทอล์ค (Smalltalk)

โปรแกรมภาษาสโมลทอล์คนั้นเป็นโปรแกรมเชิงวัตถุอย่างแท้จริง (pure Object-Oriented Programming)

2.2.1 ชนิดของตัวแปร

แบ่งออกเป็น

1) ตัวแปรแบบชั่วคราว (Tempolary Variable)

ตัวแปรแบบชั่วคราวจะกำหนดอยู่ในภายในเครื่องหมาย "" ที่บรรทัดแรกของชุดคำสั่ง ชื่อของตัวแปรแบบชั่วคราวต้องขึ้นต้นด้วยอักษรตัวเล็ก ส่วนตัวถัดไปจะเป็นตัวอักษรเล็กหรือตัวใหญ่ก็ได้

2) ตัวแปรแบบโกลบอล (Global Variable)

ชื่อของตัวแปรแบบโกลบอลต้องขึ้นต้นด้วยตัวพิมพ์ใหญ่ ส่วนตัวถัดไปจะเป็นตัวเล็ก, ตัวใหญ่ หรือตัวเลขก็ได้

2.2.2 คำสั่ง

- การกำหนดค่า โดย

1) ใช้เครื่องหมาย := (Assignment Operator) กำหนดค่าให้กับตัวแปร

2) ใช้เครื่องหมาย . (Period) แบ่งคำสั่ง(statement) แต่ละคำสั่ง ยกเว้นคำสั่งสุดท้ายในชุดคำสั่ง ตัวอย่างเช่น

```
l temp index factorial l.
```

```
factorial := #(3 4 5 6).
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

index := 1.
temp := factorial at:index.
index := index + 1

```

สองบรรทัดแรกกำหนดขอบเขตให้กับตัวแปรแบบชั่วคราว สองบรรทัดต่อมา กำหนดผลลัพธ์ของเมสเสจ (Message) ซึ่งผลลัพธ์ของเมสเสจมักจะเป็นขอบเขตเดียวเสมอ โดยปกติมักจะกำหนดขอบเขตให้กับตัวแปรแบบชั่วคราว

- การเรียกใช้ฟังก์ชัน

1) แบบมีอาร์กิวเมนต์เดียว

การเรียกใช้ฟังก์ชัน size ในภาษาปาสคาล (Pascal) จะเขียนเป็น

```
a := size(array);
```

แต่การเรียกใช้ฟังก์ชันในภาษาสมอลทอล์คคือการส่งเมสเสจ ตัวอย่างเช่น

```
a := array size
```

คือการส่งเมสเสจ size ไปยังค่าของตัวแปร array ซึ่งเป็นอาร์กิวเมนต์

2) แบบสองอาร์กิวเมนต์

ในสมอลทอล์คอาร์กิวเมนต์จะนำหน้าและตามหลังเมสเสจ ตัวอย่างเช่น

Smalltalk

```
x := x1 max:x2.
```

```
y := p + q.
```

Pascal

```
x := max(x1,x2);
```

```
y := p + q;
```

เมสเสจ max: ถูกส่งไปยังค่าของ x1 (อาร์กิวเมนต์แรก) กับค่าของ x2 (อาร์กิวเมนต์ที่สอง) แล้วกำหนดผลลัพธ์ที่ได้กับ x

3) แบบอาร์กิวเมนต์ตั้งแต่สามขึ้นไป

ตัวอย่างเช่น

```
a := x between:x1 and:x2
```

เปรียบเทียบกับ Pascal จะเขียนเป็น

```
a := between(x,x1,x2);
```

ในสมอลทอล์คชื่อของเมสเสจจะ แยกออกเป็นส่วนๆ และชื่อของเมสเสจแต่ละส่วนจะนำหน้าแต่ละอาร์กิวเมนต์ที่ต่อมาจากอาร์กิวเมนต์แรก ในตัวอย่างชื่อเมสเสจคือ between:and:

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และอาร์กิวเมนต์คือ x , $x1$, และ $x2$.

- การใช้ค่าดัชนี (Subscript) ในสมอลทอคใช้เมสเสจ `at:` และ `at:put:` ตัวอย่างเช่น

Smalltalk	Pascal
<code>x := a at:i.</code>	<code>x := a[i];</code>
<code>a at:i put:y.</code>	<code>a[i] := y;</code>
<code>a at:i+1 put:(a at:i)</code>	<code>a[i+1] := a[i]</code>

จากตัวอย่างในคำสั่งบรรทัดแรกค่าของ i เป็น index ของอาร์เรย์เพื่อกำหนดค่าของ a และส่งผลลัพธ์ไปเก็บที่ x คำสั่งในบรรทัดที่สองคือแทนค่าของ a ที่ตำแหน่ง i ด้วยค่าของ y คำสั่งในบรรทัดที่สามเป็นการแทนค่าในทำนองเดียวกันเครื่องหมายวงเล็บในสมอลทอคเป็นการกำหนดลำดับของการกระทำ และในสมอลทอคด้านซ้ายมือของเครื่องหมาย `:=` ต้องเป็นตัวแปรแบบสเกลลาร์ (Scalar) เท่านั้น

การเปรียบเทียบออบเจกต์ ในสมอลทอคเปรียบเทียบออบเจกต์โดยการส่งเมสเสจ `<`, `<=`, `=`, `>`, `>=` และ `==` ซึ่งพัฒนาเป็นไบนารีเมสเสจ ดังนั้นเครื่องหมายวงเล็บมักจะต้องใช้ถ้ามีไบนารีเมสเสจอื่น เช่น

```
3<4
#(1 2 3 4) = #(1 2 3 4)
5=(2+3)
```

การเปรียบเทียบมักจะส่งค่ากลับเป็นจริงหรือเท็จ

การทดสอบออบเจกต์ ออบเจกต์หลายออบเจกต์เข้าใจเมสเสจซึ่งในบางครั้งทดสอบสเตตัสของออบเจกต์หรือเงื่อนไข เช่น

```
$a isUpperCase
('Hello' at:1) isVowel
```

- คำสั่งเงื่อนไข

ภาษาส่วนมากใช้คำสั่ง `if` สำหรับประโยคเงื่อนไข สมอลทอคใช้บล็อกของโค้ด (Code) และเมสเสจสำหรับประโยคเงื่อนไข เช่น

```
| max a b |
a := 5 squared.
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

b := 4 factorial.

a < b

ifTrue:[max := b]

ifFalse:[max := a].

^max

เมสเสจการเปรียบเทียบ a < b: ส่งค่ากลับเป็นจริงหรือเท็จซึ่งเป็นรีซีพเวอร์ของเมสเสจ

ifTrue:ifFalse: การเปรียบเทียบมักจะไม่ใช่แบบเดี่ยวแต่มักจะเป็นแบบผสม (Compound) โดยใช้เมสเสจ and: และ or: ตัวอย่างเช่น (c < \$0 or:[c > \$9])

รีซีพเวอร์ของเมสเสจ or: คือผลลัพธ์ของการเปรียบเทียบแรกซึ่งเป็นจริงหรือเท็จ ส่วนอาร์กิวเมนต์คือบล็อกของโค้ดซึ่งเป็นการเปรียบเทียบสุดท้ายมีผลลัพธ์เป็นจริงหรือเท็จเช่นกัน

- คำสั่งทำซ้ำ (Iterative Statement)

คำสั่งวนรูปแบบธรรมดาของสมอลทอล์คใช้เมสเสจ timeRepeat: กำหนดจำนวนครั้งในการทำซ้ำเช่น

```
| string index c |
```

```
string:='Now is the time'.
```

```
index:=1.
```

```
string size timesRepeat:[
```

```
  c:=string at:index.
```

```
  string at:index put:
```

```
    (c isVowel
```

```
      ifTrue:[c UpperCase]
```

```
      ifFalse:[c asLowerCase]).
```

```
  index:=index + 1].
```

```
^string
```

เมสเสจจนวนลูปรกรรมดาแบบอื่น เช่น

```
"copy a disk file"
| input output |
input:=File pathName:'go'.
output:=File pathName:'junk'.
[input atEnd]

whileFalse:[output nextPut:input next].

input close.
output close
```

ในภาษาสมอลทอคข้อความในเครื่องหมายคำพูดคือคำอธิบาย (Comment) ภายในบล็อก input atEnd จะส่งค่ากลับเป็นจริงหรือเท็จ โดยไฟล์ input จะถูกอ่านค่าถัดไปด้วยเมสเสจ next ไฟล์ output จะเขียนค่าอาร์กิวเมนต์ (คือค่าที่ได้จากการอ่านค่าในไฟล์ input) ด้วยเมสเสจ nextPut:

ตัวอย่างของ whileTrue: เช่น

```
...
[ i < 10 ]
whileTrue:[sum:=sum + (a at: i)].
i:=i+1]
```

เมสเสจทำซ้ำ to:do: มีอาร์กิวเมนต์ 2 อาร์กิวเมนต์ ตัวอย่างเช่น

```
"draw several polygons"
3 to: 6 do:[:side| side timeRepeat:[Turtle go:6; turn:360//side]]
```

จากตัวอย่างข้างบน 3 คือรัศมีพเวอร์รอบเจด 6 คืออาร์กิวเมนต์ตัวแรกซึ่งเป็นค่าสูงสุดของการทำซ้ำ และ 2 คือบล็อกของโคดซึ่งมี side เป็นบล็อกอาร์กิวเมนต์ (จะได้กล่าวถึงรายละเอียดในหัวข้อถัดไป) จะเพิ่มค่าที่ละ 1 ในแต่ละรอบของการทำซ้ำ แต่ถ้าต้องการจะกำหนดค่าที่จะเพิ่มขึ้นในแต่ละรอบของการทำซ้ำเป็นค่าอื่น สามารถใช้เมสเสจ to:by:do: ได้ตัวอย่างเช่น

```
| sum |
sum:=0.
1/2 to: 1 by 1/8 do:[:il sum:=sum + 1].

^sum
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.3 บล็อกอาร์กิวเมนต์

บล็อกอาร์กิวเมนต์คือตัวแปรชั่วคราวแบบหนึ่ง แต่ไม่ต้องกำหนดไว้ที่บรรทัดแรกของชุดคำสั่ง บล็อกอาร์กิวเมนต์เป็นตัวแปรที่ใช้ภายในบล็อกโดยกำหนดเป็นส่วนแรกของบล็อก ซึ่งหน้าตัวแปรมีเครื่องหมาย : (colon) และตัวแปรเขียนแยกกับคำสั่งภายในบล็อกด้วยเครื่องหมาย | (vertical bar)

บล็อกและอาร์กิวเมนต์ทำให้สมอลทอคมีเมสเซจการทำซ้ำหลายเมสเซจ เช่นเมสเซจ do:, select:, reject: และ collect:

- เมสเซจ do:

ตัวอย่างเช่น

```
| vowel |
vowel:=0.
'Now is the time' do:[:char char isVowel
isTrue:[vowel:=vowel + 1]].
^vowel
```

เมสเซจการทำซ้ำ do: มีผลทำให้สตริงค์ส่งตัวอักษรทีละตัวเข้าไปยังบล็อก

- เมสเซจ select:

ตัวอย่างเช่น ('Now is the time' select:[:clc isVowel]) size

เมสเซจการทำซ้ำ select: มีผลทำให้รีซีฟเวอร์ส่งค่ากลับเป็นสมาชิกทั้งหมดเข้าไปยังบล็อก ผลลัพธ์คืออักขรระทั้งหมด และเมสเซจ size จะให้ค่าจำนวนสมาชิกทั้งหมดที่ถูกเลือก

- เมสเซจ reject:

ตัวอย่างเช่น #(1 2 3 4 5 6 7 8 9) reject:[:il i factorial >=(i*i*i)]

การทำงานของเมสเซจ reject: จะทำนองเดียวกับ select: แต่จะส่งค่ากลับเฉพาะสมาชิกทั้งหมดของรีซีฟเวอร์ที่ได้ผลลัพธ์จากบล็อกเป็นเท็จ ในขณะที่เมสเซจ select: ผลลัพธ์จากบล็อกเป็นจริง

- เมสเซจ collect:

ตัวอย่างเช่น #(1 13 7 10) collect:[:ili*i]

การทำงานของเมสเสจ collect: มีผลทำให้บล็อกของโค้ดทำงานสำหรับแต่ละสมาชิกของรีซีฟเวอร์ และให้ผลลัพธ์เป็นกลุ่มของผลลัพธ์ทั้งหมดที่ส่งกลับโดยบล็อก

- การส่งค่ากลับของฟังก์ชัน ในสมอลทอล์คใช้เครื่องหมาย ^ (caret) นำหน้านิพจน์ซึ่งเป็นผลลัพธ์ของเมธอด เมธอดจะหยุดการทำงานและส่งค่าผลลัพธ์ เช่น

```
^ answer
```

เปรียบเทียบกับภาษาปาสคาลผลลัพธ์ของฟังก์ชันจะเขียนเป็น

```
functionName:=answer;
return
```

2.2.4 ตัวอย่างโปรแกรม

```
l s c f k l
f:=Array new:26.
s:=Prompter prompt:'Enter line' default:".
1 to: 26 do:[:ilf at:i put:0].
1 to: s size do:[:il c:=(s at:i) asLowerCase.
c isLetter
ifTrue:[k:=c asciiValue - $a asciiValue + 1.
f at:k put:(f at:k) + 1]].
^f
```

2.3 การใช้สมอลทอล์คไฟว์ (Smalltalk/v)

สมอลทอล์คไฟว์คือระบบแบบเมนู ดังนั้นคุณสามารถใช้คำสั่งเพื่อเลือกสิ่งต้องการได้จากเมนูโดยการใช้เมาส์ (mouse) หรือคีย์บอร์ด

1) การเรียกเมนู (pop up menu)

โดยการคลิกปุ่มขวาของเมาส์ หรือคีย์บอร์ดปุ่ม 'Del'

2) การเลือกสิ่งที่อยู่ในเมนู

เลื่อนเคอร์เซอร์ (cursor) ไปที่สิ่งที่ต้องการเลือกแล้วกดปุ่มซ้ายของเมาส์หรือคีย์บอร์ด

ปุ่ม '+' บนคีย์บอร์ดด้านตัวเลข (numeric keypad)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.1. เมนู

สมอลทอคไฟร์ประกอบด้วยเมนูหลายชนิด คือ

1) เมนูระบบ (System Menu)

เป็นเมนูที่เราต้องใช้บ่อยๆ การเรียกเมนูระบบทำได้โดยการเลื่อนเคอร์เซอร์ออกนอกหน้าต่าง (window) ไปที่พื้นจอภาพ (background) แล้วกดปุ่มขวาของเมาส์ หรือกดคีย์บอร์ดปุ่ม 'Del' หน้าต่างของเมนูระบบจะเปิดขึ้นใหม่และกระทำฟังก์ชันระดับระบบเช่น ออกจากระบบวาดจอภาพใหม่ เป็นต้น

2) วินโดว์เมนู (Window Menu)

การเรียกวินโดว์เมนูทำได้โดยการเลื่อนเคอร์เซอร์ไปที่ป้ายของวินโดว์หรือขอบของวินโดว์ แล้วกดปุ่มขวาของเมาส์ หรือกดคีย์บอร์ดปุ่ม 'Del' ออกจากวินโดว์เมนูโดยการเลื่อนเคอร์เซอร์ออกจากวินโดว์เมนู วินโดว์เมนูของเมนูที่แตกต่างกันก็จะมีรายละเอียดที่แตกต่างกันด้วย

3) เพนเมนู(Pane menu)

การเรียกเพนเมนูทำได้โดยการเลื่อนเคอร์เซอร์ไปที่จุดใดจุดหนึ่งของวินโดว์เพน แล้วกดปุ่มขวาของเมาส์หรือกดคีย์บอร์ดปุ่ม 'Del' ซึ่งจะมีรายละเอียดแตกต่างกันสำหรับเมนูแต่ละชนิด

เราสามารถเปิดวินโดว์ได้พร้อมๆกันหลายวินโดว์ เมื่อต้องการใช้วินโดว์ใดก็เลื่อนเคอร์เซอร์เข้าไปในขอบเขตของวินโดว์นั้น แล้วกดปุ่มซ้ายของเมาส์หรือกดคีย์บอร์ดปุ่ม '+' บนคีย์บอร์ดด้านตัวเลข ป้ายของวินโดว์จะเปลี่ยนเป็นสีเข้มขึ้นเพื่อแสดงว่าเป็นแอคทีฟวินโดว์ (active window)

ถ้าต้องการเพิ่มข้อความบนเพนของวินโดว์ให้เลื่อนเคอร์เซอร์ไปที่ตำแหน่งที่ต้องการแล้วกดปุ่มซ้ายของเมาส์หรือกดคีย์บอร์ดปุ่ม '+' บนคีย์บอร์ดด้านตัวเลขจะปรากฏเครื่องหมายที่ตำแหน่งนั้นเรียกว่าจุดเพิ่ม (insert point) เราสามารถพิมพ์สิ่งที่ต้องการเพิ่มเข้าไปได้

ถ้าต้องการลบข้อความบางอย่างออกจะใช้วิธีคล้ายๆการเพิ่มคือเลื่อนเคอร์เซอร์ไปที่หลังข้อความนั้นแล้วกดปุ่มซ้ายของเมาส์หรือกดปุ่มคีย์ '+' บนคีย์บอร์ดด้านตัวเลข แล้วใช้คีย์ลบถอยหลัง (backspace) ลบข้อความที่ต้องการ

ถ้าต้องการเลือกข้อความใดข้อความหนึ่งให้เลื่อนเคอร์เซอร์ไปที่ตัวเริ่มต้นของข้อความ กดปุ่มซ้ายของเมาส์ค้างไว้ และลากเมาส์ไปที่ท้ายสุดของข้อความแล้วจึงปล่อยปุ่มซ้ายของเมาส์ จะปรากฏเป็นแถบสว่างบนข้อความที่เลือก แต่ถ้าใช้คีย์บอร์ดให้กดปุ่ม '+' บนคีย์บอร์ดด้านตัวเลขที่จุดเริ่มต้นของข้อความ และกดปุ่ม '-' บนคีย์บอร์ดด้านตัวเลขที่ท้ายสุดของข้อความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.2. ออบเจกต์และเมสเสจ

หลักการพื้นฐานของภาษาส몰ทอคคือออบเจกต์และเมสเสจ ออบเจกต์คือบล็อกพื้นฐานของส몰ทอค ตัวอย่างเช่น

1234 คือเลขจำนวนเต็ม

\$A คือตัวอักษรเอ

#(1 2 3) คืออาร์เรย์ของเลขจำนวนเต็ม ซึ่งมีอยู่ด้วยกัน 3 ออบเจกต์

#('array' 'of' 5 'string' 'and' '2') คืออาร์เรย์ของออบเจกต์ที่ต่างชนิดและต่าง

ขนาด

#(1 ('two' 'three') 4) คืออาร์เรย์ซึ่งมี 3 ออบเจกต์ และออบเจกต์ที่ 2 คืออาร์เรย์ซึ่งแน่นอนที่สุดว่าออบเจกต์ไม่สามารถทำอะไรได้ด้วยตัวเอง ในส몰ทอคเราส่งเมสเสจไปยังออบเจกต์เพื่อทำให้เกิดสิ่งหนึ่งขึ้นมา เมสเสจนั้นจะทำงานองเดียวกับการเรียกฟังก์ชันในภาษาอื่น ตัวอย่างเช่น 20 factorial คือการส่งเมสเสจ factorial ไปยังออบเจกต์ 20

เมสเสจประกอบด้วย

- 1) รีซีฟเวอร์ออบเจกต์ (Receiver Object)
- 2) เมสเสจซีเลคเตอร์ (Message Selector)
- 3) อาร์กิวเมนต์ ซึ่งอาจจะมีหรือไม่มีก็ได้

ตัวอย่างเช่น 'Now is the time' size ข้อความเป็นรีซีฟเวอร์ออบเจกต์ และ size คือเมสเสจซีเลคเตอร์ซึ่งไม่มีอาร์กิวเมนต์ เมสเสจที่ไม่มีอาร์กิวเมนต์เรียกว่ายูนารีเมสเสจ (Unary Message) เมสเสจที่มีหนึ่งอาร์กิวเมนต์ หรือมากกว่าหนึ่งอาร์กิวเมนต์เรียกว่าคีย์เวิร์ดเมสเสจ (Keyword Message)

การคำนวณในภาษาส몰ทอคมองดูคล้ายกับภาษาอื่นๆ โดยทั่วไป แต่ความจริงคือเมสเสจตัวอย่างเช่น 3 + 4 ตัวเลข 3 คือรีซีฟเวอร์ออบเจกต์ และ + คือเมสเสจซีเลคเตอร์ ซึ่งมีเลข 4 เป็นอาร์กิวเมนต์ โดยที่เมสเสจคล้ายฟังก์ชันดังนั้นจึงส่งค่ากลับเป็นออบเจกต์ เช่น 'Hello' size + 4 เมสเสจ size ส่งค่ากลับเป็นออบเจกต์ตัวเลขและเป็นรีซีฟเวอร์ออบเจกต์ของเมสเสจ + ซึ่งมี 4 เป็นอาร์กิวเมนต์ การแทนเมสเสจมักจะเป็นชุดเพื่อสั่งให้ทำงานใดๆ เช่น

Turtle blank.

Turtle home.

Turtle go:100.

Turtle turn:120.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในกรณีที่เมสเสจส่งถึงออบเจกต์เดียวกันเราสามารถเขียนแบบย่อได้เรียกว่า **เคสเคด เมสเสจ** (cascaded message) ซึ่งแต่ละเมสเสจจะเขียนคั่นด้วยเครื่องหมาย ; ตัวอย่างเช่น

```
Turtle blank;
    home;
    go:100;
    turn:120;
    go:100;
    turn:120;
    go:100;
    turn:120.
```

ออบเจกต์มีสเตตที่ออบเจกต์สามารถทำได้ เมสเสจเปลี่ยนสเตตของออบเจกต์ จาก ตัวอย่างของออบเจกต์ Turtle จำกัดตำแหน่งของมันเอง หัวเรื่อง และสีได้ เมสเสจ black และ white เปลี่ยนสีของออบเจกต์ Turtle เมสเสจ go: และ home เปลี่ยนตำแหน่งของมัน และเมสเสจ turn: เปลี่ยนหัวเรื่อง

2.3.3. คลาสและเมธอด

แนวความคิดของคลาสและเมธอดทำให้สมอลทอคไม่เหมือนโปรแกรมภาษาอื่น

1) คลาส

การแก้ไขปัญหาโดยสมอลทอคนั้นรวมการจัดประเภทของออบเจกต์ซึ่งขึ้นกับความคล้ายกัน และความแตกต่างกัน คลาสกำหนดพฤติกรรมของออบเจกต์แบบเดียวกัน โดยการกำหนดตัวแปร และเมธอดภายในออบเจกต์ ซึ่งเมธอดสามารถตอบกับเมสเสจที่ส่งมายังออบเจกต์ได้

ทุกๆ ออบเจกต์คืออินสแตนซ์ (instance) หรือสมาชิกของคลาส ตัวอย่าง เช่น #(1 2 3) และ #(sam joe) คืออินสแตนซ์ของคลาสอาร์เรย์ แต่ 'north' และ 'south' คืออินสแตนซ์ของ คลาสสตริงค์ ทุกออบเจกต์รู้ว่าตนเป็นสมาชิกคลาสใด ตัวแปรในออบเจกต์เรียกว่าตัวแปรอิน สแตนซ์

2) เมธอด

เมธอดคือ สมอลทอคโคด อะกอร์ทิทึมที่อธิบายพฤติกรรมของออบเจกต์ และการกระทำ ของออบเจกต์ เมธอดคล้ายกับฟังก์ชันในภาษาอื่น ๆ เมื่อเมสเสจส่งถึงออบเจกต์เมธอดจะถูก กระทำ และส่งผลลัพธ์กลับเป็นออบเจกต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

ระบบงานบริหารพัสดุสายช่างอากาศ

3.1 ระบบงานบริหารพัสดุสายช่างอากาศ

ระบบงานบริหารพัสดุสายช่างอากาศเป็นระบบงานที่ใช้ควบคุมการเบิกจ่ายพัสดุซึ่งเกี่ยวกับการบินทั้งหมด [7] ประกอบด้วยขั้นตอนการทำงานดังแสดงในดาต้าโฟลไดอะแกรมภาพที่ 24

3.2 โพรเซส 1.0 การรับเอกสาร

ฝ่ายรับเอกสารตรวจสอบความถูกต้องของเอกสารตามโพรเซส 1.1 ถ้าเอกสารไม่ถูกต้องจะส่งคืนเอกสารตามโพรเซส1.4 ส่งคืนเอกสารกลับคืนไปยังหน่วยที่เป็นเจ้าของเอกสาร ถ้าเอกสารถูกต้องใส่เลขที่รับเอกสารตามโพรเซส1.2 แยกประเภทของเอกสารตามโพรเซส1.3 ถ้าเป็นเอกสารเบิกจะส่งต่อไปยังโพรเซส7.0 ถ้าเป็นเอกสารส่งคืนจะส่งต่อไปยังโพรเซส2.0 ถ้าเป็นเอกสารรับพัสดุจัดหาจะส่งต่อไปยังโพรเซส9.0 หรือถ้าเป็นเอกสารรับพัสดุจากการซ่อมจะส่งต่อไปยังโพรเซส5.0 ดังแสดงในภาพที่ 25

3.3 โพรเซส 2.0 การรับพัสดุส่งคืน

หน่วยผู้ใช้ส่งคืนพัสดุที่เกินความจำเป็นของหน่วยให้กับคลังใหญ่ หรือในกรณีที่คลังใหญ่เรียกพัสดุคืนเพื่อจ่ายให้กับหน่วยผู้ใช้ที่มีความจำเป็นเร่งด่วนกว่า เช่น เครื่องบินขึ้นบินไม่ได้ เป็นต้น เจ้าหน้าที่พัสดุจะตรวจสอบสภาพของพัสดุที่ส่งคืนว่าเป็นพัสดุดีหรือพัสดุชำรุดตามโพรเซส2.1 ถ้าเป็นพัสดุดีเจ้าหน้าที่เพิ่มยอดของพัสดุดีตามโพรเซส2.2 และส่งต่อไปยังโพรเซส 3.0 เพื่อดำเนินการทำให้คลังข้างจ่ายพัสดุต่อไป ถ้าเป็นพัสดุชำรุดเพิ่มยอดจำนวนพัสดุชำรุดตามโพรเซส2.3 และส่งพัสดุชำรุดเข้าไปในโพรเซส 4.0 เพื่อดำเนินการส่งซ่อมพัสดุต่อไปดังแสดงในภาพที่ 26

3.4 โพรเซส 3.0 การหักล้างค้างจ่ายพัสดุ

รายการรับพัสดุดีจากแหล่งการจัดหา หรือจากหน่วยซ่อมพัสดุจะถูกตรวจสอบว่ามีรายการค้างจ่ายพัสดุนหมายเลขนี้ในบัญชีค้างจ่ายหรือไม่ตามในโพรเซส 3.1 ถ้าไม่มีส่งพัสดุไปเก็บในคลังพัสดุตามโพรเซส 3.2 ถ้ามีรายการค้างจ่ายพัสดุเลือกรายการค้างจ่ายพัสดุ โดยจะพิจารณา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ว่าหน่วยใดเบิกก่อนหรือหน่วยใดมีความจำเป็นเร่งด่วนตามในโพรเซส 3.3 เมื่อเลือกรายการค้างจ่ายได้แล้วจะลบบรายการค้างจ่ายนั้นออกจากบัญชีค้างจ่ายตามโพรเซส 3.4 ลดยอดพัสดุตีในบัญชีคุมพัสดุตามโพรเซส 3.5 ทำการจ่ายพัสดุตามโพรเซส 3.6 ถ้ามีพัสดุเหลือจะส่งพัสดุด่วนที่เหลือนั้นเข้าไปเก็บในคลังพัสดุตามโพรเซส 3.7 ดังแสดงในภาพที่ 27

3.5 โพรเซส 4.0 การส่งซ่อมพัสดุ

รายการพัสดุชำรุดจากโพรเซส 2.0 ทำรายการรอรับงานซ่อมตามโพรเซส 4.1 ลดจำนวนพัสดุชำรุดและเพิ่มจำนวนพัสดुरอรับงานซ่อมตามโพรเซส 4.2 และส่งพัสดุด่วนไปซ่อมตามประเภทของพัสดุ เช่น ซ่อมในประเทศหรือส่งซ่อมต่างประเทศดังแสดงในรูปที่ 28

3.6 โพรเซส 5.0 การรับพัสดุจากซ่อม

รายการรับพัสดุจากการซ่อมซึ่งได้มาจากโพรเซส 1.0 ลบบรายการพัสดुरอรับงานซ่อมออกจากบัญชีรอรับงานซ่อมตามโพรเซส 5.1 ตรวจสอบภาพของพัสดุที่รับตามโพรเซส 5.2 ถ้าเป็นพัสดุดีเพิ่มจำนวนพัสดุตีในบัญชีคุมพัสดุตามโพรเซส 5.3 แล้วส่งต่อไปยังโพรเซส 3.0 ถ้าเป็นพัสดุชำรุดจะเพิ่มจำนวนพัสดุชำรุดในบัญชีคุมตามโพรเซส 6.0 ดังแสดงในภาพที่ 29

3.7 โพรเซส 6.0 การจำหน่ายพัสดุ

รายการพัสดุชำรุดจากโพรเซส 5.0 ลดจำนวนพัสดุชำรุดในบัญชีคุมพัสดุ ตามโพรเซส 6.1 เพิ่มจำนวนพัสดุนำจำหน่ายในบัญชีคุมพัสดุตามโพรเซส 6.2 และส่งไปเก็บในคลังจำหน่ายพัสดุเพื่อรอการจำหน่ายต่อไป ดังแสดงในภาพที่ 30

3.8 โพรเซส 7.0 การจ่ายพัสดุ

เอกสารเบิกพัสดุจากโพรเซส 1.0 ตรวจสอบพัสดุเพื่อหาประเภทของพัสดุว่าเป็นพัสดุประเภทพัสดุซ่อมหมุนเวียนหรือพัสดุลิ้นเปลืองตามโพรเซส 7.1 ถ้าเป็นพัสดุซ่อมหมุนเวียนจะตรวจสอบว่ามีพัสดุชำรุดมาแลกเปลี่ยนจึงจะขอเบิกพัสดุนิวได้ตามโพรเซส 7.7 ถ้าไม่มีพัสดุดังคืนจะส่งเอกสารเบิกกลับคืนหน่วยผู้ใช้ตามโพรเซส 7.8 แต่ถ้ามีพัสดุดังคืนจะตรวจสอบยอดพัสดุนิวว่ามีจำนวนเพียงพอที่จะจ่ายหรือไม่ตามโพรเซส 7.2 ถ้ามีจำนวนพัสดุเพียงพอที่จะจ่ายจะลดจำนวนพัสดุในบัญชีคุมตามโพรเซส 7.3 ส่งจ่ายพัสดุตามโพรเซส 7.4 ถ้ามีพัสดุไม่พอจ่ายจะตรวจสอบว่ามีพัสดุที่สามารถใช้แทนกันได้หรือไม่ เพื่อนำมาพิจารณาจ่ายแทนพัสดุที่เบิกตามโพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เชส 7.5 แต่ถ้าไม่สามารถที่จะจ่ายพัสดุได้จะทำรายการพัสดุดังจ่ายไว้ตามโพรเชส 7.6 ส่งรายการค้างจ่ายเข้าไปยังโพรเชส 8.0 ต่อไป ดังแสดงในภาพที่ 32

3.9 โพรเชส 8.0 การจัดหาพัสดุ

ฝ่ายจัดหาคำนวณหาจำนวนพัสดุที่ต้องจัดหาเพิ่มเติม เนื่องจากพัสดุมีจำนวนมากกว่าแผนรายการ แต่งบประมาณมีจำกัด ดังนั้นในทางปฏิบัติจึงใช้วิธีเลือกรายการพัสดุที่มีจำนวนพัสดุน้อยกว่าจุดเบิก(reorder point) มาคำนวณหาจำนวนพัสดุที่ต้องจัดหาเพิ่มเติม การคำนวณมีดังนี้

จำนวนพัสดุที่ใช้ต่อวัน = จำนวนพัสดุที่ใช้ไปทั้งหมด/จำนวนวัน

$$[DDR=(consumption1+consumption2+consumption3) / (365x3)]$$

โดย DDR : Daily Demand Rate ,consumption: ความสิ้นเปลืองพัสดุ]

จำนวนพัสดุดังคลัง = จำนวนพัสดุที่ใช้ต่อวัน x 365 [stock-level = DDR x 365]

จำนวนพัสดุที่จุดเบิก = จำนวนพัสดุที่ใช้ต่อวัน x 270 [reorder-point = DDR x 270]

จำนวนพัสดุที่จุดวิกฤติ = จำนวนพัสดุที่ใช้ต่อวัน x 90 [safety-level = DDR x 90]

requirement = (stock-level + dueout) - (onhand + duein + workorder)

โดย จำนวนพัสดุที่ต้องมี : จำนวนพัสดุดังคลังรวมกับจำนวนพัสดุที่ค้างจ่าย

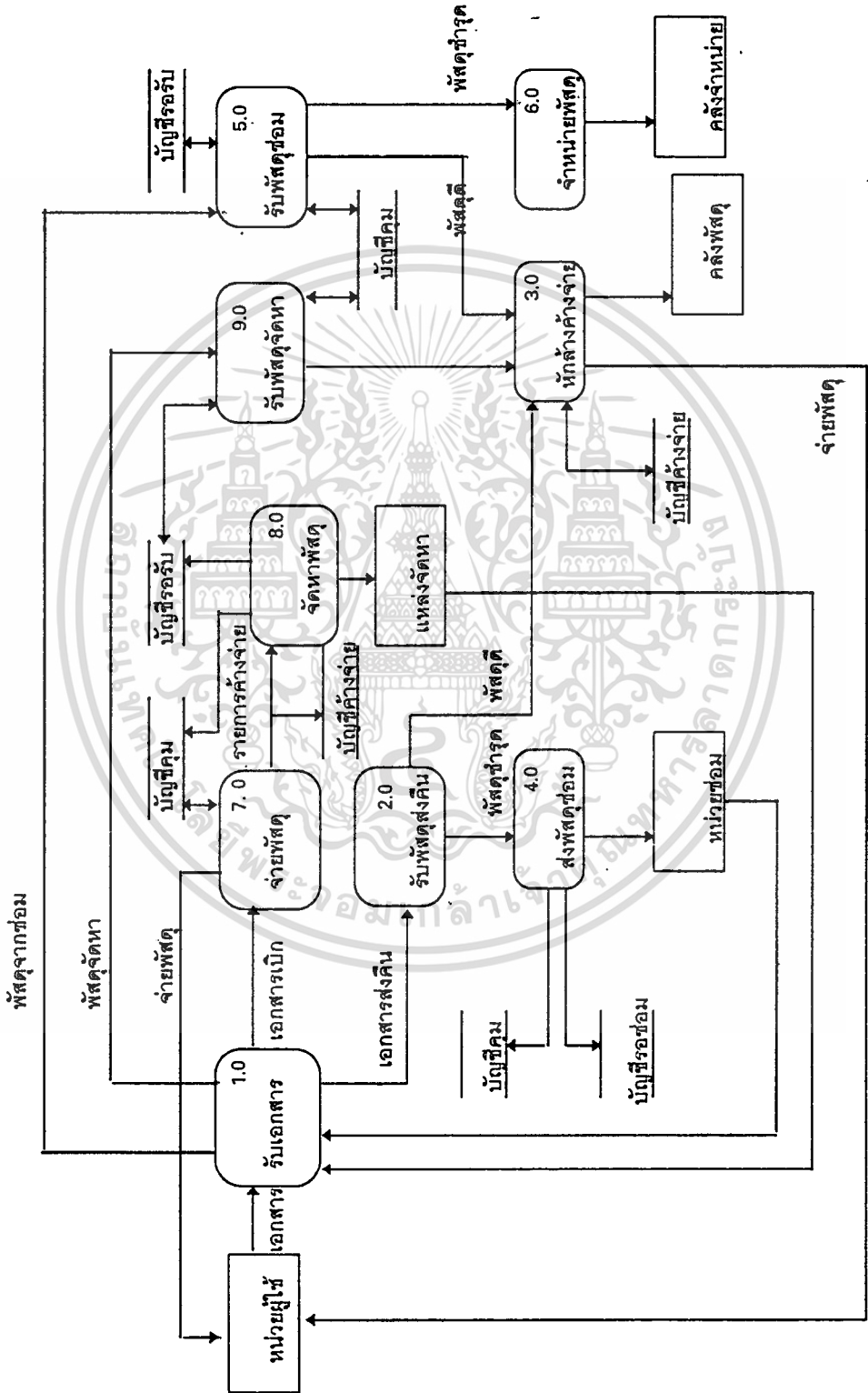
จำนวนพัสดุที่มีอยู่: จำนวนพัสดุดังคลังรวมกับจำนวนพัสดุที่รอรับจากการจัดหา และ

จากการซ่อม

การคำนวณกำหนดไว้ในโพรเชส 8.1 ทำรายการรอรับการจัดการในบัญชีรอรับการจัดการตามโพรเชส 8.2 ดำเนินการจัดหาตามโพรเชส 8.3 ส่งเอกสารการจัดการไปที่แหล่งจัดหา ดังแสดงในภาพที่ 31

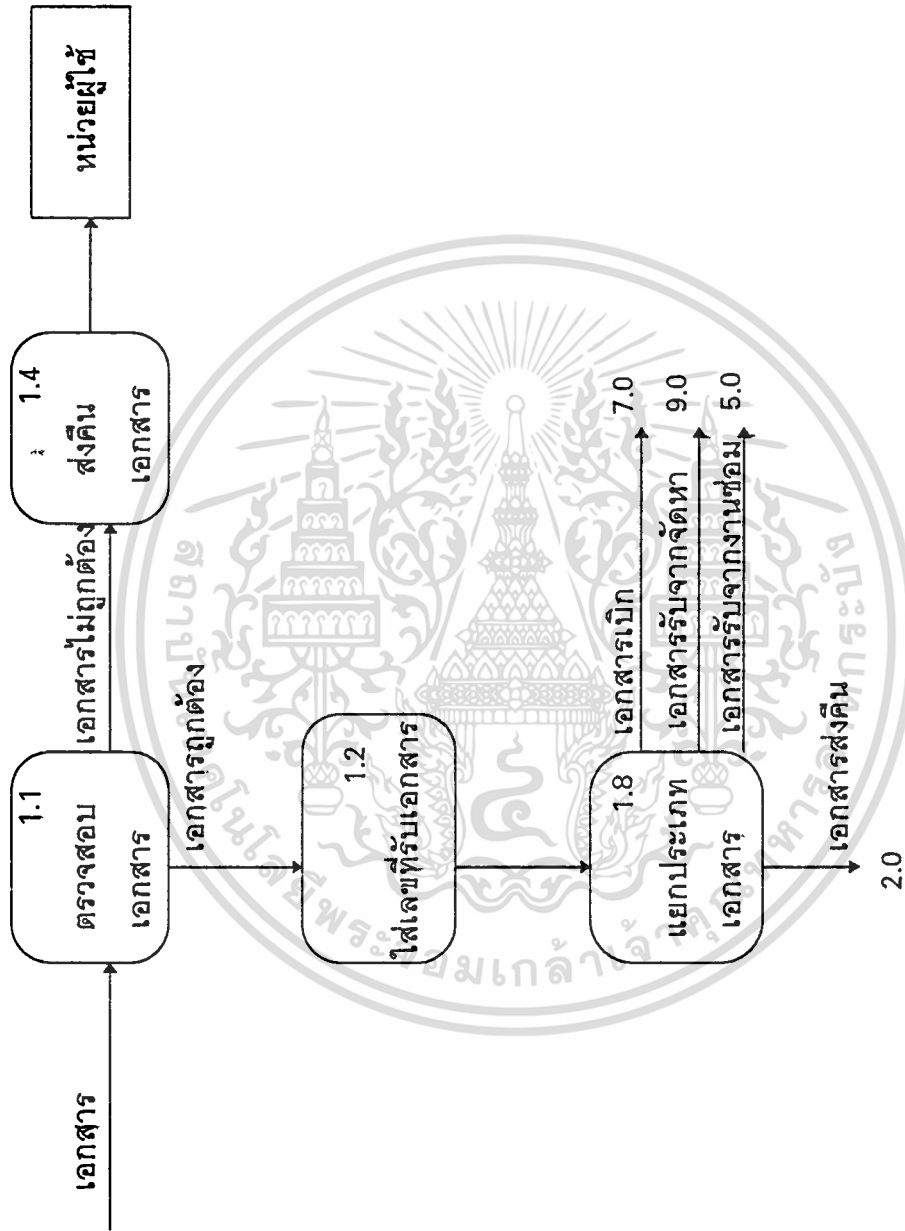
3.10 โพรเชส 9.0 การรับพัสดุจากการจัดหา

รายการรับพัสดุจากการจัดหาจากโพรเชส 1.0 ลบรายการรอรับพัสดุจากการจัดหาในบัญชีรอรับพัสดุจากการจัดหาตามโพรเชส 9.1 เพิ่มจำนวนพัสดุในบัญชีคุมพัสดุตามโพรเชส 9.2 และส่งรายการพัสดุดีเข้าไปยังโพรเชส 3.0 ดังแสดงในภาพที่ 33



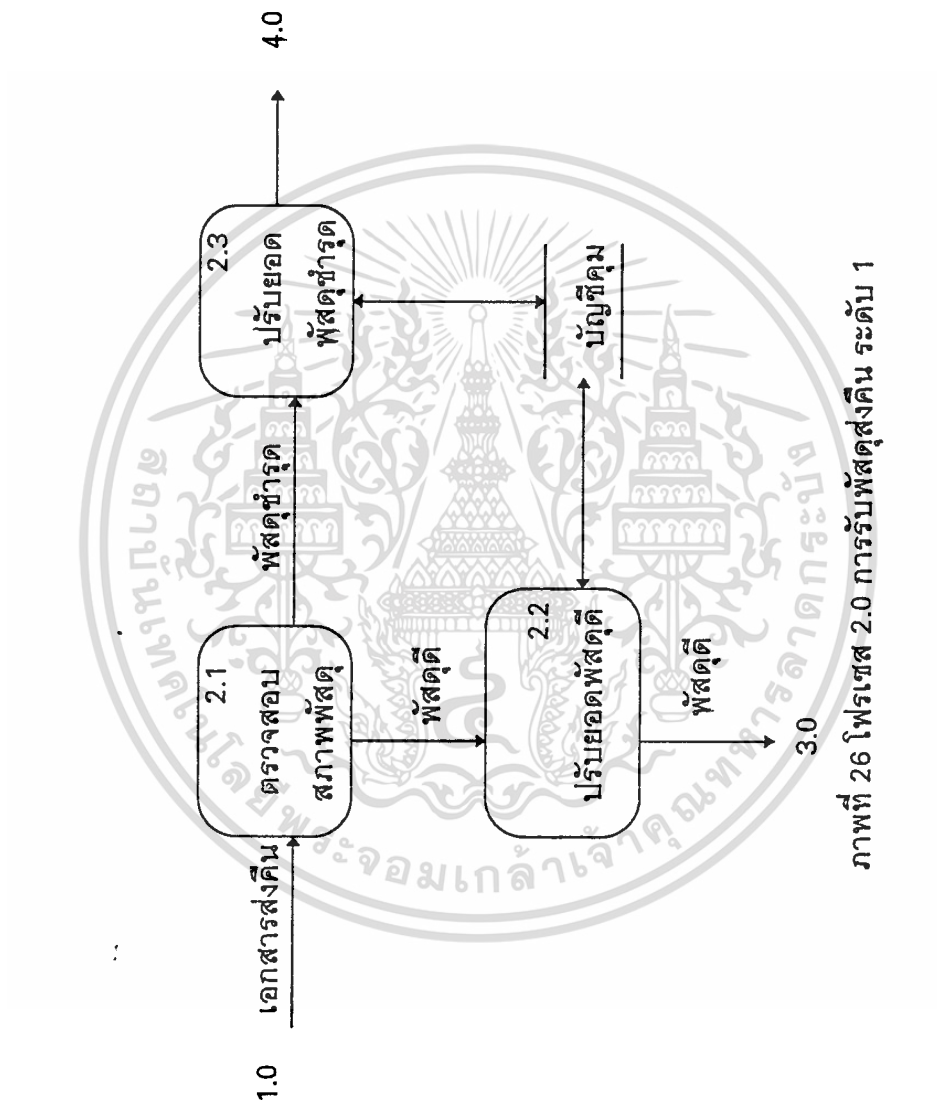
ภาพที่ 24 ดาต้าโฟลไดอะแกรมของระบบงาน ระดับ 0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 25 โฟรเซส 1.0 การรับเอกสาร ระดับ 1

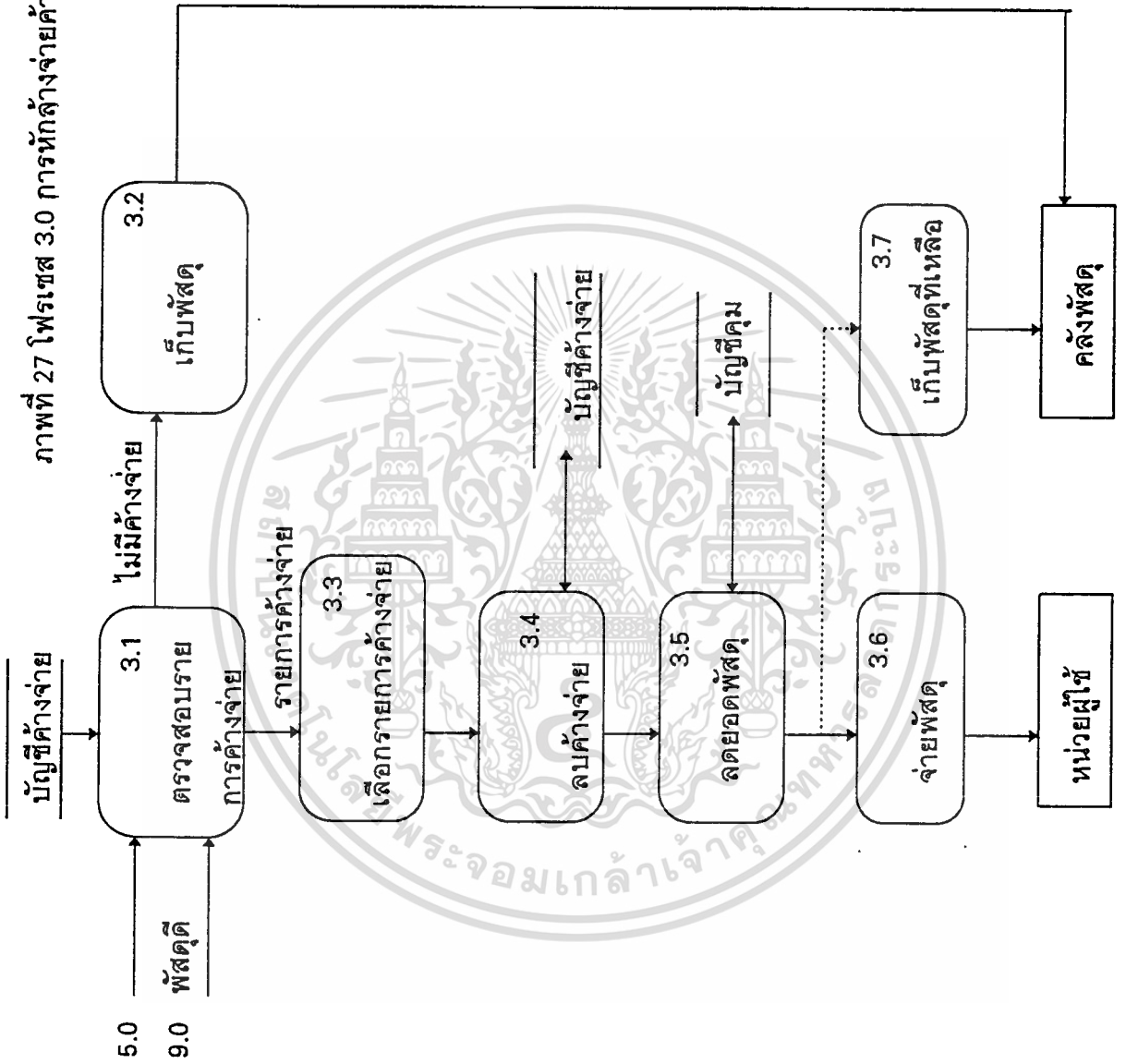
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



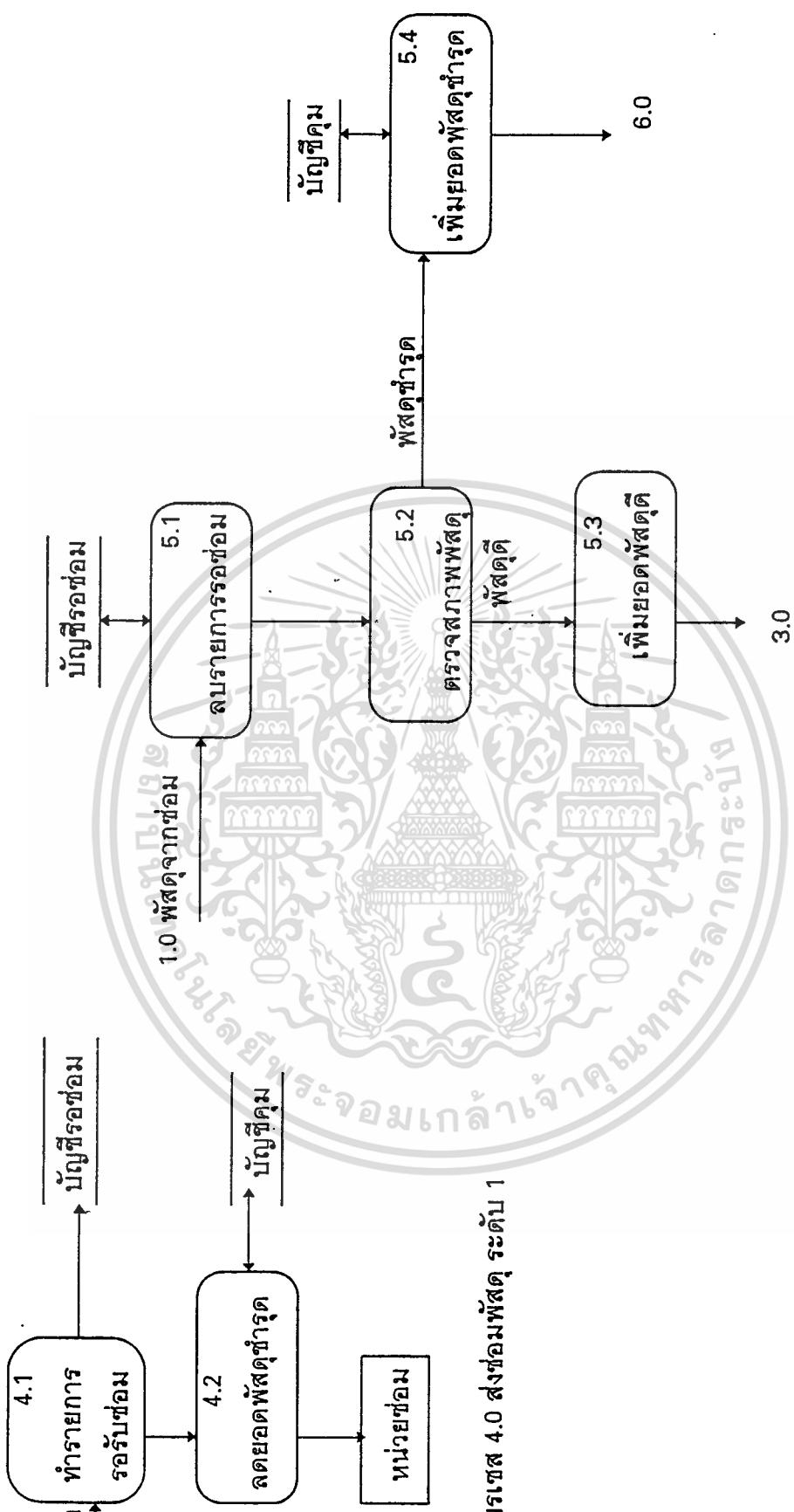
ภาพที่ 26 โพรเซส 2.0 การรับผู้สวดสงวนใจ ระดับ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 27 โพรเซส 3.0 การนำค่างจ่ายค่างจ่าย ระดับ 1

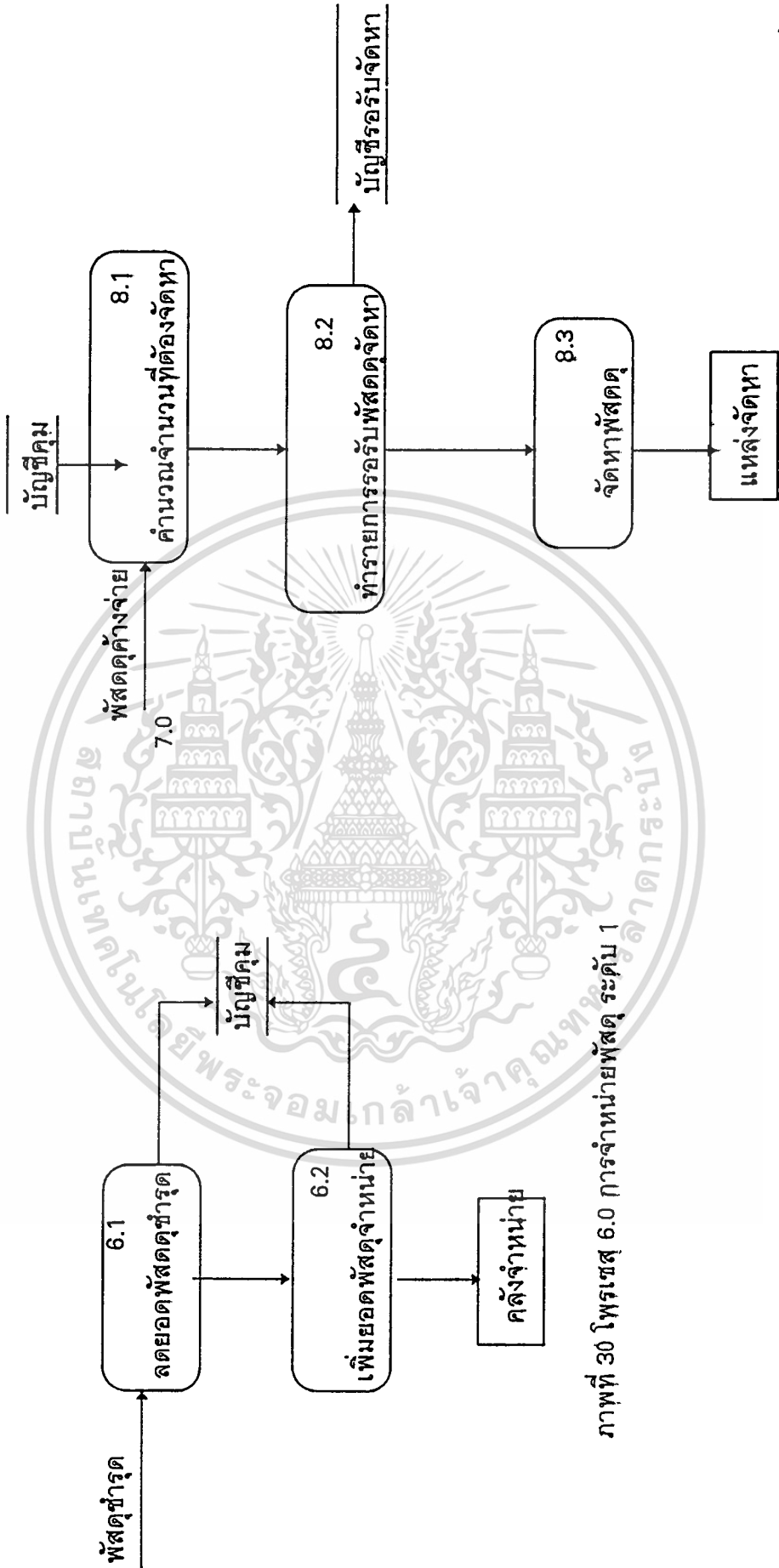


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



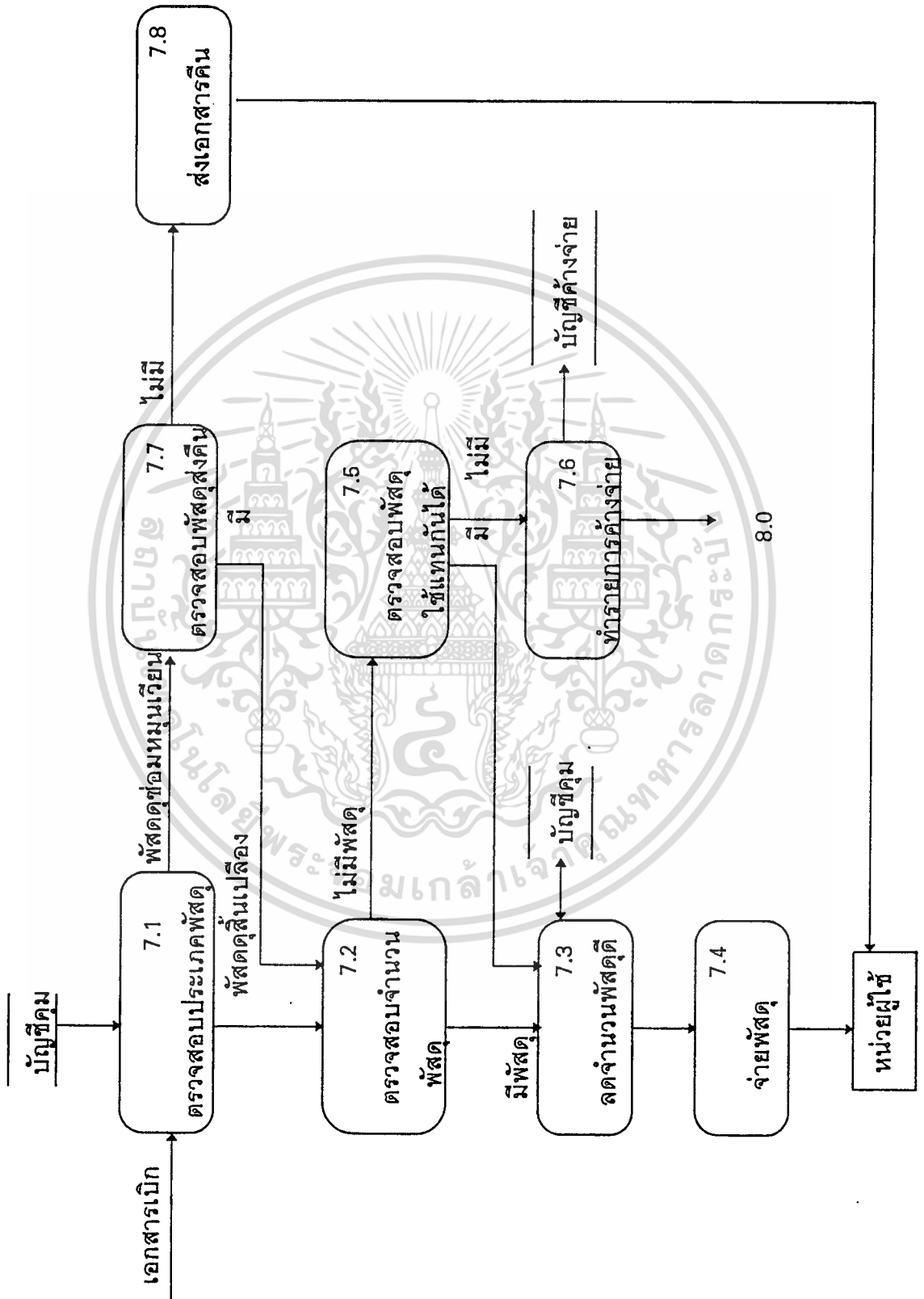
ภาพที่ 28 โพรเซส 4.0 ส่งซ่อมพัสดุ ระดับ 1

ภาพที่ 29 โพรเซส 5.0 การรับพัสดุจากซ่อม ระดับ 1

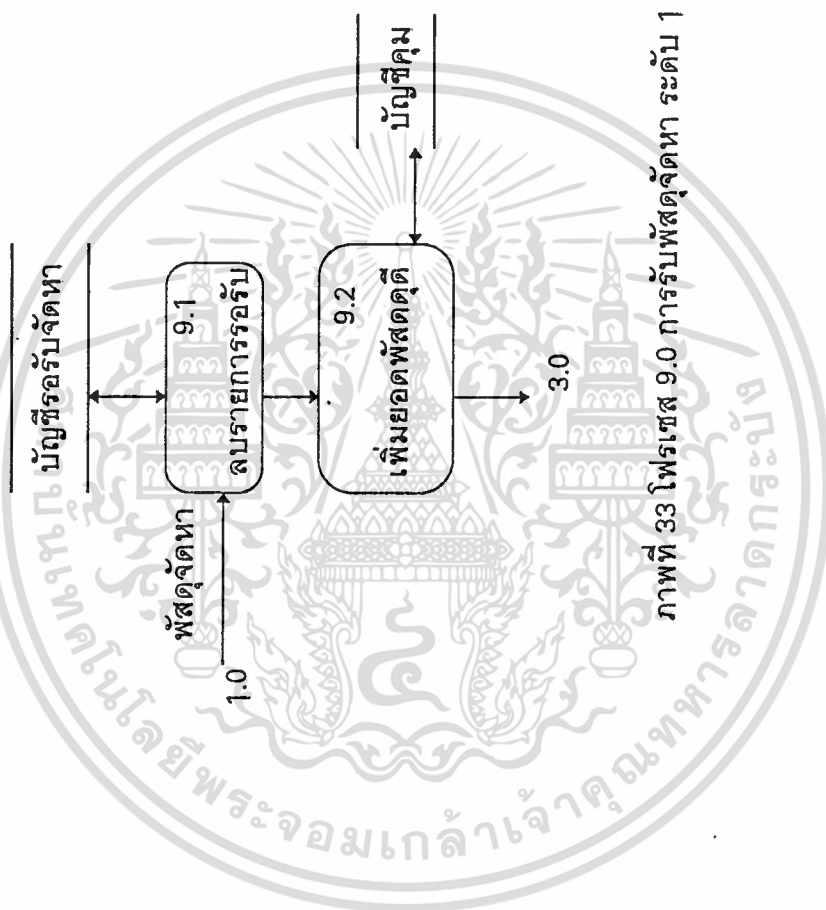


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาพที่ 32 โพรเซส 7.0 การจ่ายพัสดุ



เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ 1.0 สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ภาพที่ 33 โปรแกรม 9.0 การรับพัสดุจัดหา ระดับ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การพัฒนาระบบงาน

4.1 การวิเคราะห์เชิงวัตถุ

ขั้นตอนในการวิเคราะห์เชิงวัตถุมีดังต่อไปนี้คือ

1) การกำหนดประโยคปัญหา (Problem Statement) ซึ่งประโยคปัญหานี้อาจจะยังไม่สมบูรณ์หรือไม่อยู่ในรูปแบบฟอร์มที่ถูกต้อง ดังนั้นการวิเคราะห์จะต้องทำให้ประโยคปัญหานี้สมบูรณ์ ถูกต้องครบถ้วน ไม่คลุมเครือ และมีความคงเส้นคงวา (consistent) ประโยคปัญหาไม่ได้เป็นสิ่งที่ไม่มีวันเปลี่ยนแปลงแต่เป็นพื้นฐานที่สามารถแต่งเติมความต้องการที่แท้จริงได้ จนในที่สุดประโยคปัญหานี้สามารถใช้อธิบายระบบได้

2) การสร้างแบบจำลองของการวิเคราะห์ (Analysis Model) แบบจำลองนี้ต้องสามารถใช้แทนประโยคปัญหาได้อย่างครบถ้วนแม่นยำ สั้นกระชับรัด แต่จุใจความ ซึ่งทำให้หาคำตอบและวิธีการแก้ปัญหาจากแบบจำลองปัญหานั้นได้ การสร้างแบบจำลองภายในขอบเขตของปัญหา ทำให้ผู้พัฒนาระบบสามารถแก้ไขความเข้าใจที่อาจจะผิดพลาดได้ตั้งแต่ในระยะแรกๆ ของการสร้างระบบงาน ซึ่งยังสามารถที่จะแก้ไขได้ง่ายอยู่

4.2 ประโยคปัญหา (Problem Statement)

ขั้นตอนแรกของการสร้างระบบงานคือการกำหนดความต้องการ ประโยคของความ ต้องการจะประกอบด้วย ขอบเขตของปัญหา อะไรคือสิ่งที่ต้องการ รายละเอียดของระบบงาน สมมุติฐาน และความต้องการเกี่ยวกับการทำงาน ประโยคปัญหาต้องกำหนดว่าจะทำอะไรไม่ใช่ จะทำอย่างไร นั่นคือเป็นประโยคของความ ต้องการไม่ใช่เป็นสมมุติฐานของการแก้ปัญหา คู่มือการใช้ระบบงานที่ต้องการสร้างคือประโยคปัญหาที่ดี เจ้าของระบบงานควรระบุว่าจะอะไรที่จะต้อง มีในระบบงานและอะไรคือสิ่งที่ให้เลือกได้ เพื่อหลีกเลี่ยงการออกแบบระบบที่มีข้อบังคับมากเกินไป เจ้าของระบบงานไม่ควรอธิบายรายละเอียดภายในระบบงานมากเกินไป เพราะจะทำให้การสร้างระบบขึ้นมาใช้งานจริงไม่มีความยืดหยุ่นเท่าที่ควร ข้อกำหนดการทำงานและรูปแบบของ การติดต่อเพื่อโต้ตอบกับภายนอกระบบคือความต้องการที่เป็นไปตามทฤษฎี ประโยคปัญหา หลายประโยคจากเอกชน บริษัท และหน่วยงานของรัฐ รวมเข้าด้วยกันเป็นความต้องการที่แท้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จริงสำหรับการออกแบบระบบ ในบางครั้งอาจมีเหตุผลที่บังคับว่าต้องการเครื่องคอมพิวเตอร์ชนิดนี้ หรือต้องใช้ภาษาคอมพิวเตอร์ภาษานี้หรือมีความจำเป็นที่จะต้องใช้อะกอริทึม (algorithm) แบบนี้ นักวิเคราะห์ระบบจะต้องแยกความต้องการที่แท้จริงและความต้องการที่ใช้สร้างระบบ นักวิเคราะห์ระบบจะต้องมองความต้องการแฝงเหล่านั้นเป็นข้อจำกัดของความยืดหยุ่น อาจจะมีเหตุผลที่เป็นนโยบายหรือเป็นระเบียบการจัดองค์การซึ่งเป็นความต้องการแฝงแต่ในที่สุดนักวิเคราะห์ระบบต้องรู้ว่าสิ่งเหล่านี้ไม่ใช่ลักษณะที่สำคัญของปัญหา ประโยคปัญหาอาจจะมีรายละเอียดมากหรือมีรายละเอียดน้อย ตัวอย่างเช่นโปรแกรมบัญชีเงินเดือน หรือระบบการเงิน ต้องการรายละเอียดมาก ส่วนงานวิจัยสิ่งใหม่ๆ ไม่ต้องการรายละเอียดมาก แต่ต้องกำหนดจุดมุ่งหมายที่ชัดเจน ประโยคปัญหาส่วนมากมักจะคลุมเครือ ไม่สมบูรณ์ หรือไม่คงเส้นคงวา ความต้องการบางอย่างผิดอย่างชัดเจน ความต้องการบางอย่างกำหนดถูกต้องแต่ผลลัพธ์ที่ได้ไม่ใช่พฤติกรรมของระบบ หรือค่าใช้จ่ายในการสร้างระบบไม่สมเหตุผล ความต้องการบางอย่างดูเหมือนว่ามีเหตุผลในครั้งแรกแต่กลับทำงานออกมาแล้วไม่ตรงกับที่ต้องการหรือที่คิดไว้ ประโยคปัญหา คือจุดเริ่มต้นของการเข้าใจปัญหาไม่ใช่เป็นเอกสารที่ไม่มีวันเปลี่ยนแปลง จุดมุ่งหมายของการวิเคราะห์คือการเข้าใจปัญหาและการสร้างระบบขึ้นมาอย่างครบถ้วน ไม่มีเหตุผลที่จะคิดว่าการจัดเตรียมประโยคปัญหาจะถูกต้องโดยไม่ต้องมีการวิเคราะห์ นักวิเคราะห์ระบบต้องทำงานร่วมกับเจ้าของระบบงานเพื่อขจัดเกลาความต้องการจนกระทั่งสามารถใช้แทนความต้องการที่แท้จริงของเจ้าของระบบงานได้

ประโยคปัญหาของงานวิจัยในที่นี้ก็คือระบบงานบริหารพัสดุสายช่างอากาศที่ได้กล่าวถึงรายละเอียดของระบบงานที่ต้องการไว้ในบทที่ 3

4.3 ออบเจกโมเดลลิ่ง (Object Modeling)

ขั้นตอนแรกในการวิเคราะห์ความต้องการคือการสร้างออบเจกโมเดล ซึ่งแสดงโครงสร้างของข้อมูลแบบคงที่ของระบบ และแบ่งออกเป็นการทำงานส่วนย่อยๆ ออบเจกโมเดลใช้อธิบายคลาส และความสัมพันธ์ระหว่างคลาส สิ่งที่สำคัญที่สุดคือการจัดระเบียบในระดับบนสุดของระบบ เป็นคลาส ซึ่งเชื่อมโยงกันด้วยแอสโซซิเอชัน (association) การแบ่งในระดับต่ำกว่าภายในคลาสมีความสำคัญน้อยกว่า ออบเจกโมเดลจะถูกสร้างขึ้นก่อนไดนามิกโมเดล (dynamic model) และฟังก์ชันนัลโมเดล (functional model) เพราะว่าออบเจกโมเดลสามารถกำหนดได้ดีกว่าขึ้นกับรายละเอียดในระบบงานน้อยกว่า มีความคงที่ขณะแก้ปัญหามากกว่า และสามารถทำความเข้าใจได้ง่ายกว่าข่าวสารสำหรับออบเจกโมเดลได้มาจากประโยคปัญหา ผู้เชี่ยวชาญระบบงาน ความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รู้ทุกๆไป ถ้าผู้ออกแบบระบบไม่ใช่ผู้เชี่ยวชาญในเรื่องนั้นข่าวสารต้องได้รับมาจากผู้เชี่ยวชาญระบบงาน และต้องตรวจสอบแบบจำลองซ้ำอีก ออบเจกโมเดลช่วยให้นักคอมพิวเตอร์และผู้เชี่ยวชาญระบบงานได้มีการติดต่อสื่อสารกัน อันดับแรกกำหนดคลาสและแอตทริบิวต์ตามที่มีอิทธิพลเหนือโครงสร้างทั้งหมดและเข้าใกล้กับปัญหา ต่อไปเพิ่มแอตทริบิวต์ (attribute) เพื่อให้อธิบายเน็ตเวิร์กชั้นพื้นฐานของคลาสและแอตทริบิวต์ ต่อจากนั้นรวบรวมและจัดระเบียบของคลาสโดยใช้อินเฮริทนต์ (inheritance) การพยายามจะกำหนดอินเฮริทนต์โดยตรงโดยไม่มีกำหนดคลาสและแอตทริบิวต์ของคลาสในระดับต่ำมาก่อนมักจะทำให้โครงสร้างที่คิดไว้ผิดเพี้ยนไป ต่อจากนั้นเพิ่มโอเปอเรชัน (Operation) ในคลาสตามผลลัพธ์ของการสร้างไดนามิกโมเดลและฟังก์ชันนัลโมเดล โอเปอเรชันไม่สามารถกำหนดได้อย่างครบถ้วนจนกว่าจะเข้าใจไดนามิกโมเดลและฟังก์ชันโมเดลแล้ว วิธีที่ดีที่สุดคือเขียนความคิดต่างๆลงไปกระดาษก่อน อย่าพึ่งพยายามที่จะจัดระเบียบความคิดเหล่านั้นมากจนเกินไปถึงแม้ว่าความคิดเหล่านั้นอาจจะซ้ำซ้อนหรือไม่คงเส้นคงวา เพื่อว่าจะได้ไม่ตกล้นรายละเอียดที่สำคัญ แบบจำลองการวิเคราะห์ระบบเริ่มแรกนี้ดูเหมือนว่ายังมีข้อผิดพลาดที่จะต้องแก้ไขดังนั้นจะต้องมีการทำซ้ำ แบบจำลองทั้งหมดไม่จำเป็นต้องสร้างขึ้นมาเป็นแบบเดียวกันหมด

ขั้นตอนการสร้างออบเจกโมเดลมีดังนี้

- กำหนดออบเจก และคลาส
- จัดทำพจนานุกรมข้อมูล (data dictionary)
- กำหนดแอตทริบิวต์ (รวมทั้ง aggregation) ระหว่างออบเจก
- กำหนดแอตทริบิวต์ของออบเจกและลิงค์ (link)

4.3.1 กำหนดคลาส

ขั้นตอนแรกในการสร้างออบเจกโมเดลคือกำหนดคลาสที่อยู่ในขอบเขตของระบบงาน ออบเจกหมายถึงสิ่งของที่มีตัวตน เช่น บ้าน ลูกจ้าง และเครื่องจักร เป็นต้น ออบเจกยังหมายถึงแนวความคิด เช่น การจองที่นั่งโดยสารสายการบิน ตารางการจ่ายเงินเดือน เป็นต้น ทุกๆ คลาสต้องอยู่ภายในขอบเขตของระบบงาน หลีกเลี้ยงสิ่งที่เป็นโครงสร้างทางคอมพิวเตอร์ เช่น ลิงค์ลิสต์ (link list) ซับรูทีน (subroutine) เป็นต้น ทุกๆ คลาสไม่ได้มีอยู่อย่างชัดเจนในประโยคปัญหา บางคลาสได้มาจากขอบเขตของระบบงานหรือจากความรู้โดยทั่วไป เขียนชื่อของคลาสทั้งหมดที่คิดได้โดยไม่ต้องเลือกมากเกินไป คลาสมักจะตรงกับคำนาม อย่าไปกังวลเกี่ยวกับการอินเฮริทนต์หรือเกี่ยวกับคลาสในระดับสูงมากนัก ตัวอย่างเช่น การทำรายการสิ่งของและการตรวจสอบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับระบบงานห้องสมุด กำหนดชนิดสิ่งของที่แตกต่างกัน เช่น หนังสือ วารสารหนังสือพิมพ์ เทปวีดีโอ เป็นต้น ต่อจากนั้นรวมกลุ่มสิ่งของตามลักษณะที่คล้ายหรือแตกต่างกัน

ตัดคลาสที่ไม่จำเป็นและไม่ถูกต้องออกโดยใช้กฎเกณฑ์ดังต่อไปนี้

- คลาสที่ซ้ำซ้อนกัน (redundant classes) ถ้าคลาสแทนสิ่งเดียวกันเลือกชื่อคลาสที่มีความหมายอธิบายถึงสิ่งนั้นได้ชัดเจนดีที่สุดไว้ ตัวอย่างเช่น คลาส Customer และ Passenger ถ้าหมายถึงผู้โดยสารเครื่องบินเหมือนกันชื่อคลาส Passenger ย่อมอธิบายได้ตรงกว่า แต่ถ้าหมายถึงคนที่ติดต่อกับสายการบิน ชื่อคลาส Person ย่อมเหมาะสมกว่า เพราะความหมายใช้คลุมไปถึงผู้โดยสารได้ด้วย

- คลาสที่ไม่เกี่ยวข้องกับขอบเขตของปัญหา หรือเกี่ยวข้องน้อยมากควรที่จะตัดคลาสนั้นออก ตัวอย่างเช่น ระบบจองบัตรละคร คลาส "คนเก็บบัตร" ไม่เกี่ยวข้อง แต่คลาส "พนักงานโรงละคร" อาจจะมีเกี่ยวข้อง

- คลาสที่คลุมเครือไม่ชัดเจนบางคลาสอาจจะกำหนดขอบเขตกว้างเกินไป ตัวอย่างเช่น คลาส "เน็ตเวิร์ค" เป็นคลาสที่กว้างเกินไป

- คลาสที่ควรจะเป็นแอทริบิวท์ ตัวอย่างเช่น ชื่อ อายุ ที่อยู่ โดยปกติเป็นแอทริบิวท์ แต่ถ้ามีคุณสมบัติอิสระที่สำคัญควรจะกำหนดเป็นคลาสไม่ใช่แอทริบิวท์ ตัวอย่างเช่น ห้องทำงาน เป็นคลาสในระบบการจัดห้อง

- ชื่อคลาสซึ่งอธิบายโอเปอเรชันของออบเจกต์และไม่ได้มีการกระทำภายในตัวเองดังนั้น ไม่ใช่คลาส ตัวอย่างเช่น call เป็นผลจากการกระทำของผู้เรียกและเน็ตเวิร์คของโทรศัพท์ ถ้าเป็นระบบโทรศัพท์ธรรมดา Call เป็นเพียงส่วนหนึ่งของไดนามิกโมเดลเท่านั้นไม่ใช่คลาส แต่ถ้าโอเปอเรชันมีคุณลักษณะของตนเองควรกำหนดเป็นคลาส ตัวอย่างเช่น ระบบเก็บเงินค่าใช้โทรศัพท์ Call ควรจะเป็นคลาสและมีแอทริบิวท์เป็น วัน เวลา และจุดหมายปลายทาง เป็นต้น

- คลาสควรจะมีชื่อที่หมายถึงเนื้อหาของออบเจกต์ ไม่ใช่บทบาทในแอสซิซิเอชัน ตัวอย่างเช่น Owner เป็นชื่อคลาสที่ไม่ดีในฐานะข้อมูลของโรงงานผลิตรถ เพราะว่าถ้าจะเพิ่มคนขับหรือคนเช่ารถในภายหลังจะทำอย่างไร ควรจะใช้ชื่อว่า Person หรือ Customer ซึ่งจะมีความหมายรวมได้หลายบทบาทเช่น เจ้าของ คนขับ และผู้เช่า บางครั้งเอนทิตี (entity) หนึ่งตรงกับคลาสได้หลายคลาส ตัวอย่างเช่น Person กับ Employee อาจจะเป็นคลาสที่แตกต่างกันในบางพฤติกรรม หรืออาจจะเป็นคลาสที่ซ้ำซ้อนกันในพฤติกรรมอื่นๆ ถ้าเป็นมุมมองของฐานข้อมูลของลูกค้าบริษัท คลาสทั้งสองคลาสนี้ อาจจะเป็นคลาสเดียวกัน แต่สำหรับฐานข้อมูลเกี่ยวกับการเก็บภาษีของรัฐบาล คนสามารถทำงานได้มากกว่าหนึ่งงาน ดังนั้นจึงเป็นเรื่องที่สำคัญที่จะต้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กำหนดความแตกต่างของคลาส Person กับ Employee คนหนึ่งสามารถเป็นสมาชิกหนึ่งของ Employee หรือเป็นมากกว่าหนึ่งสมาชิก หรือไม่เป็นสมาชิกเลย

- คลาสที่เป็นโครงสร้างของการสร้างระบบงาน ควรจะตัดออกจากการวิเคราะห์ระบบ เพราะว่่าสิ่งเหล่านั้นยังไม่ต้องการตอนนี้ แต่จะต้องการในตอนออกแบบระบบ ตัวอย่างเช่น ซีพียู ทัชปด ทัชปด อะกอร์ทิม ลิงคิลิส ตรี (tree) เป็นต้น

4.3.2 กำหนดแอทริบิวต์

ต่อไปกำหนดแอทริบิวต์ แอทริบิวต์คือคุณสมบัติเฉพาะของแต่ละออบเจกต์เช่นชื่อ น้่าหนัก หรือสี แอทริบิวต์ไม่ควรจะเป็นออบเจกต์ เราควรใช้แอสไซซิเอชันแสดงความสัมพันธ์ ระหว่างออบเจกต์สองออบเจกต์ โดยปกติแอทริบิวต์จะตรงกับค่านามที่ตามด้วยวลีแสดงความเป็น เจ้าของเช่น สีของรถหรือ ตำแหน่งของเคอร์เซอร์ (cursor) คำคุณศัพท์ (adjectives) มักจะใช้แทน ค่าของแอทริบิวต์เช่น สีแดง(red) อยู่(on) หรือสิ้นสุด(expired) แอทริบิวต์แทบจะไม่มีอยู่ในประโยค ปัญหาซึ่งแตกต่างกับคลาสและแอสไซซิเอชัน ดังนั้นจึงต้องค้นหาเอาเองจากขอบเขตของระบบ งานและตามความเป็นจริง โชคดีที่นานๆ ครั้งแอทริบิวต์จะมีอิทธิพลเหนือโครงสร้างพื้นฐานของ ปัญหา อย่ากำหนดแอทริบิวต์มากเกินไป คิดเฉพาะแอทริบิวต์ที่อยู่ในระบบงานเท่านั้น อันดับ แรกให้เลือกแอทริบิวต์ที่สำคัญที่สุด ส่วนของรายละเอียดเพิ่มเติมในภายหลังได้ ในระหว่างการ วิเคราะห์หลักเลียงแอทริบิวต์ที่เกี่ยวกับการพัฒนา ชื่อของแอทริบิวต์ต้องมีความหมายที่ทำให้เข้าใจได้หรือกำหนดอย่างชัดเจน ควรจะละเว้นแอทริบิวต์ที่ได้มาจากแอทริบิวต์อื่น ตัวอย่างเช่น อายุ ได้มาจากวันเกิดและเวลาปัจจุบัน ดีไว้แอทริบิวต์สามารถใช้กำหนดคุณสมบัติแบบนามธรรมของ ระบบงาน แต่ว่าต้องกำหนดให้แตกต่างกับแอทริบิวต์หลักอย่างชัดเจน ซึ่งแอทริบิวต์หลักจะ กำหนดสถานะของออบเจกต์ ดีไว้แอทริบิวต์จะต้องไม่ดูเหมือนกับเป็นโอเปอเรชัน เช่น หาอายุ (get-age) ถึงแม้ว่ามันอาจจะมีการกระทำเป็นแบบนั้นก็ตาม กำหนดลิงค์แอทริบิวต์ด้วยเช่นกัน ลิงค์แอทริบิวต์คือคุณสมบัติของลิงค์ที่อยู่ระหว่างสองออบเจกต์ไม่ใช่เป็นคุณสมบัติของตัวออบเจกต์ ตัวอย่างเช่น แอสไซซิเอชันแบบ many-to-many ซึ่งอยู่ระหว่างผู้ถือหุ้นกับบริษัท มีลิงค์แอทริ บิวต์คือจำนวนการใช้ร่วมกัน

คัดเลือกแอทริบิวต์ที่ไม่จำเป็นและไม่ถูกต้องออกโดยใช้กฎเกณฑ์ดังต่อไปนี้

- ถ้าสิ่งที่มีอยู่แล้วอย่าง เป็นอิสระมีความสำคัญมากกว่าค่าของมัน ดังนั้นสิ่งนั้นคือ ออบเจกต์ ตัวอย่างเช่น หัวหน้าคือออบเจกต์ และเงินเดือนคือแอทริบิวต์ การแยกความแตกต่างนี้มัก จะขึ้นกับระบบงาน ตัวอย่างเช่น ระบบการส่งทางไปรษณีย์ เมืองคือแอทริบิวต์ ขณะที่ในระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำมะโนประชากร เมืองควรจะเป็นออบเจกต์ สิ่งใดที่มีลักษณะเป็นของตนเองภายในระบบงานที่กำหนดให้สิ่งนั้นคือออบเจกต์

- ถ้าค่าของแอทริบิวท์ขึ้นกับข้อความที่ระบุเฉพาะดังนั้นแอทริบิวท์นี้เป็นควอลลิไฟเออร์ตัวอย่างเช่น หมายเลขพนักงานไม่ได้มีเพียงค่าเดียวสำหรับคนที่ทำงานสองงาน หมายเลขพนักงานจะทำให้แอตทริบิวท์ใน "บริษัทจ้างคนทำงาน" ชัดเจนขึ้น

- ชื่อมักจะถูกกำหนดเป็นควอลลิไฟเออร์มากกว่าเป็นแอทริบิวท์ของออบเจกต์ ทดสอบดูว่าชื่อนั้นใช้ระบุออบเจกต์ที่อยู่ในกลุ่มได้ไหม ออบเจกต์ภายในกลุ่มสามารถมีชื่อได้มากกว่าหนึ่งชื่อใหม่ ถ้าตอบว่าใช่ชื่อนั้นทำให้แอตทริบิวท์มีความชัดเจนดีขึ้น ถ้าปรากฏว่าชื่อนี้มีค่าไม่ซ้ำกัน แสดงว่ามีคลาสที่ถูกควอลลิไฟเออร์ไปตัวอย่างเช่น ชื่อของแผนกจะไม่ซ้ำกันในบริษัทหนึ่ง แต่ในบางโปรแกรมอาจจะต้องการทำงานกับหลายบริษัทซึ่งมีชื่อของแผนกที่ซ้ำกัน ชื่อจะเป็นแอทริบิวท์ของออบเจกต์ก็ต่อเมื่อชื่อนั้นไม่ได้ขึ้นกับเนื้อหาในข้อความ และโดยเฉพาะเมื่อชื่อนั้นมีค่าซ้ำกันได้ ชื่อของคนต่างกับชื่อของบริษัทซึ่งชื่อคนอาจจะมีค่าซ้ำกันได้ เพราะฉะนั้นจึงเป็นแอทริบิวท์ของออบเจกต์

- ภาษาเชิงวัตถุเน้นได้รวมความคิดเกี่ยวกับค่าที่ใช้ระบุตัวตนของออบเจกต์หนึ่งๆสำหรับใช้อ้างถึงออบเจกต์นั้นได้อย่างไม่คลุมเครือเข้าใจในภาษาดู ในออบเจกต์โมเดลไม่ต้องแสดงค่าที่ใช้ระบุตัวตนของออบเจกต์เหล่านี้ เหมือนกับว่ามีค่าเหล่านี้ในออบเจกต์โมเดลอยู่แล้ว แสดงเฉพาะค่าของแอทริบิวท์ที่อยู่ในขอบเขตของระบบงานเท่านั้น ตัวอย่างเช่น หมายเลข serial number ของพัสดุใช้อ้างถึงพัสดุแต่ละชิ้นส่วน

- ถ้าคุณสมบัติขึ้นกับการแทนค่าของลิงค์ ดังนั้นคุณสมบัตินั้นคือแอทริบิวท์ของลิงค์และไม่เกี่ยวข้องกับออบเจกต์ แอทริบิวท์ของลิงค์โดยปกติปรากฏชัดเจนบนแอตทริบิวท์แบบ many-to-many ซึ่งแอทริบิวท์ของลิงค์ไม่สามารถจะติดเข้ากับคลาสทั้งสองเพราะมัลติพลิซิตีของคลาสทั้งสอง แอทริบิวท์ของลิงค์บนแอตทริบิวท์แบบ many-to-one นั้นดีกว่าเพราะสามารถติดเข้ากับออบเจกต์ด้านที่เป็น many ได้โดยข่าวสารไม่หาย แอทริบิวท์บนแอตทริบิวท์แบบ one-to-one ก็สมารถทำได้เช่นเดียวกัน

- ถ้าแอทริบิวท์อธิบายสถานะภายในของออบเจกต์ซึ่งภายนอกออบเจกต์จะมองไม่เห็นดังนั้นตัดมันออกจากการวิเคราะห์

- ละเว้นแอทริบิวท์ของหลายๆมาซึ่งไม่น่าจะเป็นไปได้ว่าจะมีอิทธิพลเหนือโอเปอเรชันส่วนมาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่นิยมนำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- แอทริบิวต์ที่ดูเหมือนว่าแตกต่างกับแอทริบิวต์อื่น และไม่เกี่ยวข้องกับแอทริบิวต์อื่นๆ อาจจะชี้คลาสซึ่งควรจะแตกออกเป็นสองคลาสที่แตกต่างกัน คลาสควรจะแบบธรรมดาและมีการติดต่อกัน

ขั้นตอนต่อไปคือจัดคลาสโดยการใช้อินเฮริทนต์เพื่อใช้โครงสร้างแบบสามัญร่วมกัน อินเฮริทนต์สามารถเพิ่มเข้ามาได้สองทางด้วยกัน คือโดยกำหนดลักษณะแบบธรรมดาทั่วไปของคลาสที่มีอยู่แล้วใส่เข้าไปในซูเปอร์คลาส(superclass) เรียกว่าวิธีทำจากข้างล่างขึ้นมา(bottom up)หรือโดยการขัดเกลากลุ่คลาสที่มีอยู่แล้วใส่เข้าไปในซับคลาสพิเศษเรียกว่าวิธีทำจากข้างบนลงไป(top down)

เราสามารถจะค้นพบอินเฮริทนต์จากวิธีทำจากข้างล่างขึ้นมาโดยการหาคลาสที่มีแอทริบิวต์ แอสโซซิเอชัน หรือโอเปอเรชันคล้ายๆ กันกำหนดเป็นซูเปอร์คลาสที่ใช้ลักษณะแบบสามัญร่วมกันสำหรับแต่ละเจเนอรัลไลเซชัน(generalization) ตัวอย่างเช่น พัสดูลิ้นเปลือง พัสดูล้อมหมุนเวียน สามารถที่จะสร้างแบบรวมๆ ได้คือ "พัสดู" บางแอทริบิวต์หรือคลาสที่เหมือนกันอาจจะถูกกำหนดใหม่เพียงเล็กน้อยเท่านั้นที่จะทำให้เหมาะสมกันโดยถูกต้อง แต่อย่าพยายามที่จะทำให้เข้ากันได้มากจนเกินไปนัก เพราะอาจจะได้เจเนอรัลไลเซชันที่ผิด บางเจเนอรัลไลเซชันซึ่งมีอยู่จริงอาจจะพบได้ในที่กำลังจัดกลุ่ม ใช้แนวความคิดที่มีอยู่แล้วนี้เมื่อไรก็ตามที่เป็นไปได้ ความสมดุลจะเสนอแนะคลาสที่ขาดหายไปจากเจเนอรัลไลเซชันนั้น

วิธีทำจากข้างบนลงไป สเปเชียลไลเซชัน (specialization) มักจะปรากฏขึ้นมาจากขอบเขตของระบบงาน ให้มองหาวลีค่านามประกอบด้วยคำคุณศัพท์หลายชนิดบนชื่อคลาส เช่น fluorescent lamp, incandescent lamp ให้หลีกเลี่ยงการขัดเกลากลุ่ที่มากจนเกินไป ถ้าสเปเชียลไลเซชันที่เสนอขึ้นมาขัดแย้งกับคลาสที่มีอยู่แล้ว คลาสที่มีอยู่แล้วอาจจะถูกกำหนดไว้ไม่ถูก กรณีร่องๆ ลงมาในขอบเขตของระบบงานทั้งหมดคือแหล่งกำเนิดของสเปเชียลไลเซชัน บ่อยครั้งที่เขียนเฉพาะเขตของกรณีร่องๆ ที่มีอยู่ก็เป็นการเพียงพอแล้วโดยไม่ต้องเขียนที่เป็นจริงทั้งหมด ตัวอย่างเช่น Bank account สามารถขัดเกลากลุ่เป็น Checking account และ Savings account ในขณะที่ในระบบงานธนาคารบางระบบความแตกต่างนี้ไม่มีอิทธิพลเหนือพฤติกรรมของระบบ เพียงแค่เพิ่มแอทริบิวต์ธรรมดาคือ "account type" ลงใน account เท่านั้นก็เพียงพอ

มัลติเพิลอินเฮริทนต์(multiple inheritance) อาจจะนำมาใช้เพื่อเพิ่มการใช้ร่วมกันแต่ต้องเฉพาะเมื่อจำเป็นเท่านั้น เพราะว่ามันเพิ่มทั้งแนวความคิดและความซับซ้อนในการสร้างระบบ

เมื่อชื่อของแอทริบิวต์ที่เหมือนกันปรากฏมากกว่าหนึ่งครั้งด้วยความหมายแบบเดียวกัน ดังนั้นพยายามกำหนดอย่างกว้างๆ สำหรับคลาสที่รวมกัน ตัวอย่างเช่น Transaction ถูกส่งเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่นอนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เข้ามาทั้งทาง Cashier และ ATM กำหนดเงินเนอรัลไลเซชันคือ Entry station สำหรับ Cashier station และ ATM ในบางครั้งคลาสไม่มีอะไรที่คล้ายๆ กันเลยนอกจากแอสไซซิเอชัน แต่ก็มีบ่อยครั้งที่พบลักษณะโดยทั่วไปที่เคยถูกมองข้าม

แอสไซซิเอชันและแอสไซซิเอชันต้องกำหนดกับคลาสที่ระบุเฉพาะ ซึ่งอยู่ในแผนผังของคลาส(class hierarchy) แต่ละแอสไซซิเอชันและแอสไซซิเอชันต้องถูกกำหนดกับคลาสแบบทั่วไปส่วนมากที่เหมาะสม การทำทุกอย่างให้ถูกต้องอาจจะต้องการการแก้ไขบางอย่างให้เหมาะสม ความสมมูลอาจจะเสนอแนะแอสไซซิเอชันเพิ่มขึ้นเพื่อบอกความแตกต่างระหว่างชั้นคลาสให้เห็นชัดเจนขึ้น

ให้ลองไล่ตามเส้นทางต่างๆ ตลอดทั้งขอบเขตโมเดลเพื่อมองดูว่าเส้นทางนั้นให้ผลลัพธ์ที่สมเหตุสมผลหรือไม่ ในที่ๆ ที่คิดว่ามีเพียงแค่ค่าเดียวจะมีเส้นทางที่ให้ผลลัพธ์เพียงค่าเดียวหรือไม่ สำหรับมัลติพลิซิตี้ (multiplicity) แบบ "many" มีหนทางที่จะได้ค่าเดียวเมื่อต้องการหรือไม่ คิดถึงคำถามที่อยากจะถามมีคำถามที่มีประโยชน์ซึ่งตอบไม่ได้หรือไม่ สิ่งเหล่านี้จะบ่งชี้ถึงข่าวสารที่ขาดหายไป ถ้ามีบางสิ่งซึ่งดูธรรมดาในความเป็นจริง แต่กลับซับซ้อนในแบบจำลอง แสดงว่าอาจจะมียางบางสิ่งซึ่งขาดหายไป (แต่ต้องมั่นใจว่ามันไม่ได้ซับซ้อนจริงๆ)

มันยากมากที่ขอบเขตโมเดลจะถูกต้องหลังการทำเพียงครั้งเดียว การพัฒนาซอฟต์แวร์ทั้งหมดคือหนึ่งครั้งของการทำซ้ำ ส่วนที่แตกต่างกันของแบบจำลองมักจะอยู่ที่ชั้นของความสำเสร็จซึ่งแตกต่างกันด้วย ถ้าพบความบกพร่องให้ย้อนกลับไปที่ยุ่ก่อนหน้านั้นถ้ามันจำเป็นต้องแก้ไข การขัดเกลางานบางอย่างจะสามารถทำได้ก็ต่อเมื่อทำไดนามิกโมเดลและฟังก์ชันนัลโมเดลเสร็จเรียบร้อยแล้ว

สัญญาณที่บอกว่ามีขอบเขตขาดหายไป

- ความสมมูลในแอสไซซิเอชันและเงินเนอรัลไลเซชัน: ให้เพิ่มคลาสใหม่โดยใช้ความคล้ายคลึงกัน
- กระจายแอสไซซิเอชันและโอเปอเรชันของคลาส: ให้แตกคลาสออก ดังนั้นแต่ละส่วนจะมีการติดต่อกัน
- คุณสมบัติร่วมกันของคลาสหาร่าบาก: คลาสหนึ่งอาจแสดงสองบทบาท แตกคลาสนั้นออก
- โอเปอเรชันที่ไม่มีคลาสเป้าหมายที่ดี: ให้เพิ่มคลาสเป้าหมายที่ขาดไป
- แอสไซซิเอชันที่ซ้ำกันทั้งชื่อและความประสงค์: ให้สร้างซูปเปอร์คลาสขึ้น
- บทบาทของคลาส: อาจจะต้องเป็นคลาสที่แยกกัน ซึ่งมักจะหมายถึงการเปลี่ยนแอสไซซิเอชันไปเป็นคลาส ตัวอย่างเช่น คนสามารถที่จะทำงานได้หลายบริษัท ดังนั้น Employee คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คลาสของคนทำงานบริษัทซึ่งระบุว่าเป็นบริษัทอะไร ซึ่งเป็นคลาสที่เพิ่มขึ้นมาจากคลาส Person และ Company

สัญญาณที่บอกว่าคลาสนั้นไม่จำเป็น

- แอทริบิวท์ ไอเปอเรชัน และ แอสไซซิเอชัน บนคลาสที่ขาดไป: ให้ตั้งคำถามว่าทำไมถึงต้องการสิ่งนี้

สัญญาณที่บอกว่าแอสไซซิเอชันนั้นขาดไป

- เส้นทางสำหรับไอเปอเรชันขาดไป: ให้เพิ่มแอสไซซิเอชันเพื่อให้สามารถตอบคำถามได้

สัญญาณที่บอกว่าแอสไซซิเอชันนั้นไม่จำเป็น

- ข่าวสารในแอสไซซิเอชันที่ซ้ำซ้อน: ให้ตัดแอสไซซิเอชันที่ไม่ได้เพิ่มข่าวสารใหม่หรือเปลี่ยนไปเป็นดิโรว์แทน

- แอสไซซิเอชันที่ไม่มีไอเปอเรชัน: ถ้าไม่มีไอเปอเรชันใช้เส้นทางนั้นอาจจะไม่ต้องการข่าวสารนั้น ซึ่งการทดสอบนี้ต้องคอยจนกระทั่งกำหนดไอเปอเรชันเสร็จแล้ว

สัญญาณที่บอกว่ามีแอสไซซิเอชันวางไม่ถูกที่

- ชื่อของบทบาทซึ่งกว้างหรือแคบจนเกินไป: ให้เคลื่อนย้ายแอสไซซิเอชันขึ้นหรือลงในแผนผังของคลาสเพื่อหาตำแหน่งที่เหมาะสม

สัญญาณที่บอกว่ามีแอทริบิวท์วางไม่ถูกที่

- ต้องการเข้าถึงออบเจกต์โดยผ่านทางค่าแอทริบิวท์ค่าหนึ่งของออบเจกต์นั้น: ให้กำหนดควอลิไฟแอสไซซิเอชัน (qualified association) ขึ้น

ในวิธีการปฏิบัติ การสร้างแบบจำลองไม่ได้ต้องทำตามลำดับที่กล่าวมาข้างต้นอย่างเข้มงวด เราสามารถที่จะรวมหลายชั้นตอนเข้าด้วยกันได้เมื่อเรามีประสบการณ์ ตัวอย่างเช่นเราสามารถกำหนดคลาส ตัดคลาสที่ไม่ถูกต้องออกโดยไม่ต้องเขียนคลาสนั้นลงไป และใส่คลาสเหล่านั้นลงในแผนผังออบเจกต์พร้อมกับแอสไซซิเอชัน เราสามารถเลือกบางส่วนของแบบจำลองและพัฒนาในรายละเอียดบางอย่าง ในขณะที่ส่วนอื่นๆ ยังเป็นร่างอยู่ ลำดับชั้นตอนสามารถจะสลับกันได้ถ้าเหมาะสม แต่ถ้าฟังจะเรียนรู้ออบเจกต์โมเดล ขอแนะนำให้ทำตามลำดับชั้นตอน

ขั้นตอนสุดท้ายของออบเจกต์โมเดลคือจัดกลุ่มคลาสลงในหน้ากระดาษและในโมดูล (module) แผนผังอาจจะแบ่งออกเป็นหน้ากระดาษที่มีขนาดและรูปแบบเดียวกันเพื่อความสะดวกในการวาด การเขียน และการดู คลาสที่ยึดติดกันต้องรวมกลุ่มเข้าด้วยกัน แต่หน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กระดาษมีขนาดจำกัด ดังนั้นต้องมีการตัดแบ่งเป็นครั้งคราวตามแต่จะกำหนดเอาเอง โมดูลคือกลุ่มของคลาส (หนึ่งหน้ากระดาษหรือมากกว่า) ซึ่งเป็นบางส่วนของแบบจำลองทั้งหมด ตัวอย่างเช่น แบบจำลองของระบบปฏิบัติการคอมพิวเตอร์อาจประกอบด้วยโมดูลของควบคุมการทำงาน ควบคุมอุปกรณ์ บำรุงรักษาเพิ่มข้อมูล และจัดหน่วยความจำ ขนาดของโมดูลเปลี่ยนแปลงได้

แต่ละแอสซีซีไอจะอยู่ในหน้ากระดาษเดียวกัน แต่บางคลาสอาจจะปรากฏมากกว่าหนึ่งครั้งเพื่อใช้เป็นจุดต่อหน้ากระดาษ หากจุดตัดในระหว่างคลาส คลาสซึ่งเป็นจุดต่อจุดเดียวในระหว่างสองหน้ากระดาษหรือโมดูล หรือคือจุดที่แยกขอบเขตเน็ตเวิร์ค คลาสนั้นเป็นเหมือนสะพานระหว่างสองหน้ากระดาษหรือโมดูล ตัวอย่างเช่นในระบบจัดการเพิ่มข้อมูล เพิ่มข้อมูลคือจุดตัดระหว่างโครงสร้างของโคเรคทอรี และเนื้อเรื่องของการเพิ่มข้อมูล ถ้าหากจุดตัดเพียงจุดเดียวไม่ได้ก็พยายามให้มีจุดเชื่อมต่อน้อยที่สุด พยายามเลือกใช้โมดูลเพื่อลดจำนวนของการข้ามหน้ากระดาษในแผ่นผังขอบเขต แผ่นผังขอบเขตส่วนมากสามารถเขียนเป็นกราฟได้โดยที่ไม่มีเส้นตัดกัน

นำโมดูลที่ออกแบบก่อนหน้านี้กลับมาใช้อีกถ้าเป็นไปได้ แต่อย่าพยายามที่จะทำให้มันเป็นเช่นนั้น การนำกลับมาใช้ใหม่(reuse) เป็นสิ่งที่ง่ายที่สุดเมื่อส่วนหนึ่งของปัญหาเข้ากันได้กับปัญหาก่อนหน้านี้ ถ้าปัญหาใหม่คล้ายๆ กับปัญหาก่อนหน้านี้ การออกแบบเริ่มแรกจะต้องขยายกว้างออกไปเพื่อให้ครอบคลุมได้ทั้งสองปัญหา

รายละเอียดของคลาสต่างๆ ที่กำหนดขึ้นได้ในระบบงานบริหารพัสดุสายช่างอากาศมีดังนี้คือ

Item
Aircraft type
FSC/NIIN
Name
ERRC
Cost
UI
SOS
ISG
OnHand
Add
Delete
Update
Issue
Issue to Shop
Turn in
Receive from Dueln
Receive from Shop
BOR
Procuring
Item detail

คลาส Item เป็นคลาสของพัสดุ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AED
Action code
Display add screen Display delete screen Display issue screen Display issue to shop screen Display turn in screen Display receive from duein screen Display receive from shop screen Display BOR screen Display procuring screen Display item detail screen

คลาส AED เป็นคลาสที่ใช้ในการรับข้อความติดต่อจากผู้ใช้โดยตรงแล้วส่งผ่านต่อไปยังคลาสต่างๆ ภายในระบบงาน หรือเป็นคลาสของระบบงานนั่นเอง

DuelN
Document SOS code Quantity Date
Create Delete

คลาส Dueln เป็นคลาสของเอกสารรับการจัดหา เอกสารนี้เป็นเอกสารที่ส่งวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DueOut
Document Organization code Quantity Date
Create Delete

คลาส DueOut เป็นคลาสของเอกสารการค้างจ่าย

WO
Document Shop code Quantity Date
Create Delete

คลาส WO เป็นคลาสของเอกสารรรับงานซ่อมพัสดุ

4.3.2 การจัดทำพจนานุกรมข้อมูล

เนื่องจากว่าคำแต่ละคำมีความหมายได้หลายอย่างดังนั้นควรทำพจนานุกรมข้อมูลสำหรับแบบจำลองทั้งหมด เขียนคำอธิบายแต่ละคลาสอย่างครบถ้วนอธิบายขอบเขตของคลาสภายในปัญหาขณะนั้น

พจนานุกรมข้อมูลของระบบงานบริหารพัสดุดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชื่อข้อมูล	ความหมาย
Aircraft type	รหัสชนิดของเครื่อง
FSC/NIIN	รหัสหมายเลขของพัสดุ
Name	ชื่อของพัสดุ
ERRC	รหัสขีดความสามารถในการซ่อมได้ของพัสดุ
Cost	ราคาต่อหน่วยของพัสดุ
UI	รหัสหน่วยนับของพัสดุ
SOS	รหัสของแหล่งที่จัดหาพัสดุ
ISG	รหัสของหมายเลขพัสดุที่ใช้แทนกันได้
OnHand	จำนวนพัสดุดี
Repair	จำนวนพัสดุนำรอด
Action code	รหัสระบุประเภทของการทำงาน (รายละเอียดอยู่ในรูปที่ 5.4)
Document	หมายเลขของเอกสาร แบ่งออกเป็นของ คลังใหญ่ หน่วยผู้ใช้ เอกสารการซ่อมพัสดุ เอกสารการจัดหาพัสดุ
SOS code	รหัสของแหล่งที่จัดหาพัสดุ
Shop code	รหัสของหน่วยที่ซ่อมพัสดุ
Organization code	รหัสของหน่วยผู้ใช้
Quantity	จำนวนของพัสดุที่จ่าย ส่งซ่อม หรือจัดหา
Date	วันที่จ่าย ส่งซ่อม หรือจัดหาพัสดุ

4.3.3 กำหนดแอสซิซิเอชัน

ต่อไปกำหนดแอสซิซิเอชันระหว่างคลาส ความสัมพันธ์ระหว่างคลาสสองคลาสหรือมากกว่าสองคลาสคือแอสซิซิเอชัน คลาสหนึ่งอ้างถึงคลาสอื่นโดยใช้แอสซิซิเอชัน จะต้องไม่อ้างถึงคลาสด้วยแอทริบิวต์ตัวอย่างเช่น คลาส Person ต้องไม่มีแอทริบิวต์เป็น Employer แต่ใช้แอสซิซิเอชันแทนเช่น แอสซิซิเอชัน Work-for ใช้อ้างถึงคลาส Company จาก Person แอสซิซิเอชันเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แสดงความสัมพันธ์ระหว่างคลาสในระดับเดียวกัน แอสโซซิเอชันถูกพัฒนาขึ้นมาได้หลายทางด้วยกัน แต่ยังไม่ควรเลือกว่าจะทำวิธีใดในแบบจำลองการวิเคราะห์เพื่อให้การออกแบบมีอิสระ แอสโซซิเอชันมักจะตรงคำกริยาหรือวลีกริยารวมทั้งสถานที่ (next to, part of, contained in) การกระตุ้น (drives) การติดต่อสื่อสาร (talk to) การเป็นเจ้าของ (has, part of) หรือการสอดคล้องกับเงื่อนไขใดเงื่อนไขหนึ่ง (work for, married to, manages) เลือกคำกริยาเหล่านี้จากประโยคปัญหาเป็นแอสโซซิเอชันแล้วเขียนลงในกระดาษ ยังไม่ต้องรีบขีดเกลาะอะไร อย่าเพิ่งเสียเวลาที่จะพยายามแยกความแตกต่างระหว่างแอสโซซิเอชันกับอะกรีเกชัน (aggregation) อะกรีเกชันก็คือแอสโซซิเอชันพิเศษแบบหนึ่ง

ตัดแอสโซซิเอชันที่ไม่จำเป็นและไม่ถูกต้องออกโดยใช้กฎเกณฑ์ดังต่อไปนี้

- แอสโซซิเอชันที่อยู่ระหว่างคลาสที่ถูกตัดออก ถ้าคลาสใดคลาสหนึ่งในแอสโซซิเอชันถูกตัดออก ดังนั้นแอสโซซิเอชันต้องถูกตัดออกไปด้วย หรือต้องเปลี่ยนไปเป็นของคลาสอื่น
- ตัดแอสโซซิเอชันที่ไม่ถูกต้องในขอบเขตของปัญหาหรือเกี่ยวข้องกับโครงสร้างของการพัฒนา ตัวอย่างเช่น system handles concurrent access
- ตัดแอสโซซิเอชันที่อธิบายถึงการกระทำหรือเหตุการณ์ เพราะว่าแอสโซซิเอชันควรอธิบายถึงคุณสมบัติที่เป็นโครงสร้างของระบบงาน ตัวอย่างเช่น แผนกควบคุมรับเอกสารเบิกพัสดุ ซึ่งอธิบายถึงส่วนหนึ่งของการโต้ตอบกันระหว่างแผนกควบคุมกับหน่วยผู้ใช้ไม่ได้เป็นความสัมพันธ์ที่ถาวรระหว่างแผนกควบคุมกับเอกสารเบิกพัสดุแต่อย่างใด
- แอสโซซิเอชันที่เป็นชนิดเทอร์นารี(ternary)หรืออยู่ในระหว่างคลาสตั้งแต่สามคลาสขึ้นไปสามารถแตกออกเป็นชนิดไบนารีแอสโซซิเอชัน (binary) หรือควอลิไฟแอสโซซิเอชัน(qualified association) ได้ก็ควรจะทำ ยกเว้นว่าไม่สามารถจะทำได้จริงๆ เพราะว่าข่าวสารบางส่วนจะสูญหายไป เช่น Professor teaches course in room ถ้าในเทอร์นารีแอสโซซิเอชันไม่มีลักษณะอะไรที่เป็นของตนเองดังนั้นสามารถทำเป็นแอทริบิวท์ของลิงค์ (link attribute) บนไบนารีแอสโซซิเอชันได้เช่น Company pays salary to person สามารถทำเป็น Company employs person มีค่า salary เป็นแอทริบิวท์ของลิงค์ Company-Person
- ตัดแอสโซซิเอชันที่ได้มาจากแอสโซซิเอชันอื่นๆ (derived association) เพราะว่าซ้ำซ้อนกัน ตัวอย่างเช่น Grandparent of นี้ตัดออกได้ เพราะว่าสามารถถูกกำหนดเป็นคู่ของ Parent of ได้ และถ้าเป็นแอสโซซิเอชันที่กำหนดมาจากเงื่อนไขของแอทริบิวท์ในออบเจกต์ ตัดออกได้เช่นกัน เช่น younger than แทนเงื่อนไขวันเกิดของบุคคล (Person) สองคน ไม่ได้มีข่าวสารใหม่เพิ่ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขึ้น คลาส แอทริบิวท์ และแอสโซซิเอชันในออบเจกต์โมเดลควรจะแทนค่าของข่าวสารที่เป็นอิสระต่อกันมากที่สุดเท่าที่เป็นไปได้ เส้นทางระหว่างคลาสที่มีมากกว่าหนึ่ง (multiple paths) มักจะเป็นดิริวเอดแอสโซซิเอชัน (derived association) ซึ่งเป็นส่วนประกอบของแอสโซซิเอชันเริ่มแรก (primitive association) แต่ต้องระวังเพราะว่าไม่ใช่ทุกแอสโซซิเอชันที่มีรูปแบบเป็นมัลติเพลทาสแล้วจะกำหนดซ้ำซ้อนเสมอไป บางครั้งแอสโซซิเอชันสามารถมาจากสองแอสโซซิเอชันหรือมากกว่าสองได้ และทำมัลติพลิซิตี้ (multiplicity) ไม่ได้ให้เก็บแอสโซซิเอชันพิเศษนั้นไว้ ถ้าข้อบังคับของมัลติพลิซิตี้ที่สำคัญ ตัวอย่างเช่น บริษัทจ้างคนทำงานหลายคนและเป็นเจ้าของคอมพิวเตอร์หลายเครื่อง ลูกจ้างบางคนไม่ถูกกำหนดให้ใช้คอมพิวเตอร์เลย แต่บางคนถูกกำหนดให้ใช้คอมพิวเตอร์ได้มากกว่าหนึ่งเครื่องคอมพิวเตอร์บางเครื่องมีคนใช้หนึ่งคนหรือไม่มีใครใช้เลยซึ่งแอสโซซิเอชัน Employs และ Owns ไม่สามารถอ้างถึงมัลติพลิซิตี้ของแอสโซซิเอชัน Assigned-to ได้ ถึงแม้ว่าดิริวเอดแอสโซซิเอชันจะไม่ได้เพิ่มข่าวสารใหม่ แต่มีประโยชน์ในความเป็นจริงและในการออกแบบ ตัวอย่างเช่น ความสัมพันธ์กันในเครือญาติ เช่น ลุง แม่นาย และหลาน อธิบายถึงความสัมพันธ์ของครอบครัวซึ่งมีความสำคัญในสังคมของเรา ในออบเจกต์โมเดลใช้สัญลักษณ์เส้นประระบุดิริวเอดแอสโซซิเอชันเพื่อให้แตกต่างกับแอสโซซิเอชันหลัก ต่อจากนั้นกำหนดแอสโซซิเอชันขั้นสูงดังต่อไปนี้

- การกำหนดชื่อของแอสโซซิเอชันนั้นเป็นเรื่องที่สำคัญ ไม่ต้องบอกว่าเป็นอย่างไรหรือทำไม บอกว่าเป็นอะไร ควรที่จะเลือกอย่างระมัดระวังและให้ความหมายที่สามารถเข้าใจได้เช่น Bank computer maintains account เป็นประโยคของการกระทำ ประโยคของวลีกริยาคือ Bank holds account

- เพิ่มชื่อของบทบาทในที่ๆ เหมาะสม ชื่อของบทบาทอธิบายถึงบทบาทของคลาสในแอสโซซิเอชันจากมุมมองของอีกคลาสหนึ่ง ตัวอย่างเช่นแอสโซซิเอชัน Work-for คลาส Company มีบทบาทเป็น employer และคลาส Person มีบทบาทเป็น employee แต่ถ้ามีเพียงแอสโซซิเอชันเดียวระหว่างคลาสคู่หนึ่งและชื่อของคลาสอธิบายบทบาทอย่างชัดเจนดีแล้วไม่ต้องมีชื่อของบทบาทเช่น Central computer communicates with ATM แอสโซซิเอชันระหว่างสองสมาชิกในคลาสเดียวกัน (reflexive association) ต้องมีชื่อบทบาท เพื่อแยกความแตกต่าง เช่น Person manages person ต้องมีบทบาท boss และ worker

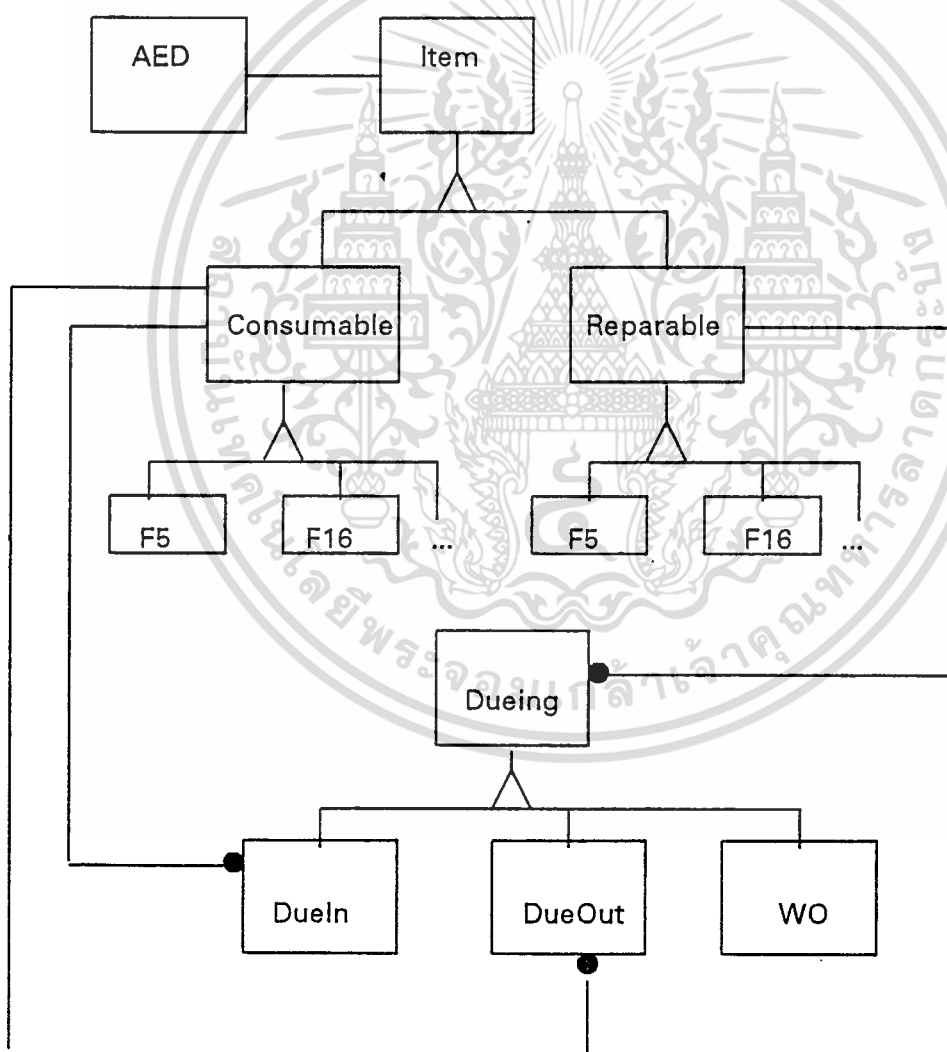
- ควอลิไฟแอสโซซิเอชัน(qualified association) โดยปกติชื่อจะบอกได้ว่าออบเจกต์นั้นๆ คืออะไร แต่ชื่อส่วนใหญ่แล้วจะซ้ำกัน เนื้อหาพร้อมกับชื่อจะระบุให้ออบเจกต์นั้นไม่ซ้ำกัน ตัวอย่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เช่นชื่อของบริษัทในเมืองต้องไม่ซ้ำกัน แต่อาจจะซ้ำกับในเมืองอื่น ชื่อของบริษัทควอลลิไฟแอสโซซิเอชัน State charters company ชื่อของเมืองและบริษัทจะทำให้บริษัทไม่ซ้ำกัน ค่าควอลลิไฟจะบอกความแตกต่างของออบเจกต์ด้านที่เป็นชนิด many ของแอสโซซิเอชัน ตัวอย่างเช่น หมายเลขเอกสารจะทำให้สามารถแยกเอกสารไปยังแผนกที่ต้องการ

- กำหนดมัลติพลิซิตี แต่อย่าพยายามที่จะทำให้มันถูกต้องมากจนเกินไปเพราะว่ามัลติพลิซิตีมักจะเปลี่ยนแปลงในเวลาวิเคราะห์เสมอ
- เพิ่มแอสโซซิเอชันที่ขาดไป

รายละเอียดออบเจกต์โมเดลของระบบงานบริหารพัสดุสายช่างอากาศแสดงอยู่ในภาพที่ 34



ภาพที่ 34 ออบเจกต์โมเดล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 ไดนามิกโมเดลลิง (Dynamic modelling)

ไดนามิกโมเดลแสดงพฤติกรรม (behavior) ที่ขึ้นกับเวลาของระบบและออบเจกต์ในแบบจำลอง เริ่มต้นการวิเคราะห์ไดนามิกโดยการหาอีเวนต์ (event) สิ่งเร้าและการตอบสนองที่มองเห็นได้จากภายนอกระบบ ลำดับของอีเวนต์สำหรับแต่ละออบเจกต์ลงในสเตตไดอะแกรม (state diagram) การทำอะกอริทึม (algorithm) สำเร็จไม่ใช่สิ่งที่การวิเคราะห์ต้องการถ้าไม่มีเหตุการณ์ที่มองเห็นได้จากภายนอก ระบบ อะกอริทึมเป็นเพียงส่วนหนึ่งของการสร้างระบบเท่านั้น

ไดนามิกโมเดลไม่มีความสำคัญสำหรับงานซึ่งเกี่ยวกับกรเก็บข้อมูลแท้ๆ เช่น ระบบฐานข้อมูล (database) แต่ไดนามิกโมเดลมีความสำคัญสำหรับงานที่มีการกระทำโต้ตอบกัน สำหรับปัญหาส่วนมากนั้นความถูกต้องแห่งตรรกวิทยา (logical) ขึ้นกับลำดับของการกระทำโต้ตอบกัน ไม่ใช่เวลาที่แน่นอนของการกระทำโต้ตอบกัน แต่อย่างไรก็ตามในระบบเวลาจริง (real-time) ต้องการเวลาที่ระบุเฉพาะของการกระทำโต้ตอบกัน ซึ่งจะต้องคิดถึงในขณะวิเคราะห์ แต่ในที่นี้จะไม่พูดถึงการวิเคราะห์เวลาจริง

อันดับแรกจัดเตรียมภาพสถานการณ์ (scenarios) ถึงแม้ว่าอาจจะไม่ครอบคลุมทั้งหมดที่อาจจะเกิดขึ้นได้แต่อย่างน้อยก็มั่นใจได้ว่าการกระทำโต้ตอบแบบธรรมดาจะไม่ถูกมองข้ามไปเหตุการณ์ได้มาจากภาพสถานการณ์ โดยปกติจะเป็นการดีที่สุดที่จะกำหนดอีเวนต์เป็นอันดับแรกและจากนั้นจึงกำหนดแต่ละอีเวนต์ให้กับออบเจกต์เป้าหมายของอีเวนต์นั้น แล้วจัดลำดับของอีเวนต์และสเตต (state) ลงในสเตตไดอะแกรม (state diagram) สุดท้ายก็เปรียบเทียบสเตตไดอะแกรมสำหรับออบเจกต์ต่างๆ เพื่อให้เกิดความมั่นใจว่าออบเจกต์แลกเปลี่ยนอีเวนต์เป็นคู่ที่เข้ากันได้ผลลัพธ์ของสเตตไดอะแกรมสร้างขึ้นเป็นไดนามิกโมเดล

ขั้นตอนการสร้างไดนามิกโมเดลมีดังต่อไปนี้คือ

- จัดเตรียมภาพสถานการณ์ของลำดับการกระทำโต้ตอบกัน
- กำหนดอีเวนต์ระหว่างออบเจกต์
- เตรียมทางเดินของอีเวนต์สำหรับแต่ละภาพสถานการณ์
- สร้างสเตตไดอะแกรม
- จับคู่ระหว่างอีเวนต์กับออบเจกต์เพื่อตรวจสอบความคงเส้นคงวา

จัดเตรียมบทสนทนาระหว่างผู้ใช้และระบบ เพื่อใช้กำหนดพฤติกรรมของระบบ ภาพสถานการณ์เหล่านี้แสดงการกระทำโต้ตอบหลัก รูปแบบการแสดงต่อภายนอกและการแลกเปลี่ยนข่าวสาร ภาพสถานการณ์ทำให้เข้าใจไดนามิกโมเดลได้มากกว่าการพยายามเขียนแบบ

จำลองแบบทั่วไปโดยตรงและยังรับประกันได้ว่าจะไม่ข้ามขั้นตอนที่สำคัญ รวมทั้งเส้นทางของการกระทำโต้ตอบกันทั้งหมดถูกต้อง

ในบางครั้งประโยคปัญหาอธิบายลำดับการกระทำโต้ตอบกันได้ทั้งหมด แต่เวลาที่ใช้ส่วนมากใช้ในการกำหนดรูปแบบของการกระทำโต้ตอบกัน ประโยคปัญหาอาจจะกำหนดข่าวสารที่ต้องการแต่ไม่ระบุวิธีการได้รับข่าวสาร ในระบบงานหลายระบบการรวบรวมข้อมูลอินพุตคืองานส่วนใหญ่ ไดนามิกโมเดลมีความสำคัญสำหรับระบบงานเช่นนี้

เริ่มต้นเตรียมภาพสถานการณ์สำหรับกรณี "ปกติ" การกระทำโต้ตอบกันที่ไม่มีข้อมูลอินพุตที่ไม่ปกติหรือเงื่อนไขข้อผิดพลาด จากนั้นจึงคิดถึงกรณี "พิเศษ" เช่น ลำดับของข้อมูลอินพุตที่ละเว้นได้ ค่าสูงสุดและต่ำสุด และค่าซ้ำ ต่อไปคิดถึงกรณีข้อผิดพลาดของผู้ใช้ ค่าที่หาค่าไม่ได้และความไม่สำเร็จที่จะต้องตอบ สำหรับระบบงานชนิดมีการกระทำโต้ตอบกันหลายระบบการเก็บข้อผิดพลาดคือส่วนที่ร่าบากที่สุดของการพัฒนา ถ้าเป็นไปได้ควรยินยอมให้ผู้ใช้ยกเลิกการกระทำนั้นหรือถอยหลังกลับไปตั้งต้นที่จุดที่ไม่มีปัญหาก่อนหน้านั้น ท้ายสุดคิดถึงชนิดของการกระทำโต้ตอบกันหลายชนิดที่สามารถกำหนดเป็นการกระทำโต้ตอบขั้นพื้นฐานได้เช่น การร้องขอความช่วยเหลือ และสถานะของคำถาม(status queries)

ภาพสถานการณ์คือลำดับของอีเวนต์ อีเวนต์เกิดขึ้นเมื่อมีการแลกเปลี่ยนข่าวสารระหว่างออบเจกต์ในระบบและตัวกระทำภายนอกเช่น user, sensor หรืองานอื่น ค่าของข่าวสารที่แลกเปลี่ยนคือพารามิเตอร์ของอีเวนต์ อีเวนต์ที่ไม่มีพารามิเตอร์เป็นอีเวนต์ที่มีความหมายและเป็นแบบธรรมดา ข่าวสารในอีเวนต์คือความจริงที่เกิดขึ้นเป็นสัญญาณ (signal) เมื่อข่าวสารอินพุตในระบบ หรือเอาวิพุทจากระบบมีอีเวนต์เกิดขึ้น

สำหรับแต่ละอีเวนต์กำหนดผู้กระทำ (เช่น ระบบ ผู้ใช้ หรือผู้กระทำภายนอกอื่นๆ) ที่ทำให้เกิดอีเวนต์และพารามิเตอร์ของอีเวนต์ รูปแบบของจอภาพ หรือรูปแบบการแสดงผลลัพธ์ไม่ได้มีอิทธิพลเหนือการกระทำโต้ตอบกันหรือการแลกเปลี่ยนค่า ไม่ต้องกังวลถึงรูปแบบผลลัพธ์สำหรับการเริ่มต้นออกแบบไดนามิกโมเดล เอาไว้อธิบายตอนขัดเกลาแบบจำลอง

การกระทำโต้ตอบกันสามารถแบ่งออกได้เป็นสองส่วน คือลอจิกของระบบงานและการติดต่อกับผู้ใช้ นักวิเคราะห์ระบบควรพึงเล็งเป็นอันดับแรกที่การไหลของข่าวสารและการควบคุมข่าวสารมากกว่ารูปแบบการแทนค่า ลอจิกของโปรแกรมเดียวกันสามารถจะรับการอินพุตจากบรรทัดของคำสั่ง เพิ่มข้อมูล เมาส์ การสัมผัสจอภาพ กดปุ่มคีย์บอร์ด หรือการเชื่อมต่อระยะไกล ถ้าหากว่ารายละเอียดของการติดต่อแบ่งแยกกันอย่างระมัดระวัง ไดนามิกโมเดลจะเก็บลอจิกการควบคุมของระบบงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นการยากที่จะหาวิธีของการติดต่อกับผู้ใช้โดยไม่มี การทดสอบ ลอจิกระบบงาน สามารถจำลองขึ้นโดยสร้างเป็นตุ๊กตาของวิธีดำเนินงาน (dummy procedure) ลอจิกของระบบงาน ที่แยกกันกับการติดต่อกับผู้ใช้ยอมให้หาวิธีการติดต่อกับผู้ใช้ได้ในขณะที่ระบบงานกำลังพัฒนา อยู่

4.4.1 การกำหนดอีเวนต์

ตรวจสอบภาพสถานการณ์เพื่อกำหนดอีเวนต์ภายนอกทั้งหมด ได้แก่ สัญญาณทั้งหมด ข้อมูลอินพุท การตัดสินใจ การขัดจังหวะ การเปลี่ยนสถานะ และการกระทำกับผู้ใช้หรือ จากผู้ใช้ การกระทำกับอุปกรณ์ภายนอกหรือจากอุปกรณ์ภายนอก แต่ขั้นตอนการคำนวณภายในไม่ใช่อีเวนต์ยกเว้นสำหรับจุดตัดสินใจซึ่งกระทำโต้ตอบกับโลกภายนอก ใช้ภาพสถานการณ์ ค้นหาอีเวนต์ปกติ แต่อย่าลืมกรณีที่เกิดผิดพลาดและอีเวนต์ที่ไม่ปกติ

การกระทำของออบเจกต์ที่ส่งผ่านข่าวสารคืออีเวนต์ ตัวอย่างเช่นในระบบ ATM การใส่รหัสผ่านคืออีเวนต์ส่งจากผู้กระทำภายนอกคือ User ถึง ATM การกระทำโต้ตอบกันระหว่างออบเจกต์กับออบเจกต์และโอเปอเรชันตรงกับอีเวนต์ ตัวอย่างเช่น การใส่บัตรคืออีเวนต์ซึ่งส่งจาก User ถึง ATM การไหลของข่าวสารบางอย่างนั้นมีอยู่ชัดเจน ตัวอย่างเช่น การจ่ายเงินสดคืออีเวนต์จาก ATM ถึง User

อีเวนต์ซึ่งมีผลแบบเดียวกันบนทิศทางของการควบคุมกำหนดให้มีชื่อเดียวกันถึงแม้ว่าค่าพารามิเตอร์จะแตกต่างกัน ตัวอย่างเช่น การใส่รหัสผ่านควรจะเป็นคลาสของอีเวนต์เมื่อค่าของรหัสผ่านไม่มีอิทธิพลบนทิศทางของการควบคุม ในทำนองเดียวกันการจ่ายเงินสดคือคลาสของอีเวนต์ด้วยเมื่อจำนวนเงินไม่ได้มีอิทธิพลบนทิศทางของการควบคุม อีเวนต์ที่มีอิทธิพลเหนือทิศทางของการควบคุมควรจะถูกกำหนดแตกต่างกัน เช่น บัญชีถูกต้อง, บัญชีไม่ถูกต้อง, และรหัสผ่านผิดคืออีเวนต์ที่แตกต่างกัน อย่างไรก็ตามกลุ่มเข้าด้วยกันโดยใช้สถานะของบัตร

ถ้าค่าที่เป็นปริมาณแตกต่างกันมีความสำคัญเพียงพอให้กำหนดเป็นอีเวนต์ที่มีความแตกต่าง ตัวอย่างเช่น การกดปุ่มตัวเลขแต่ละตัวที่แตกต่างกันปกติจะคิดว่าเป็นอีเวนต์เดียวกันเมื่อการควบคุมไม่ได้ขึ้นกับค่าของตัวเลข การกดปุ่ม "enter" อาจจะเป็นอีเวนต์ที่แตกต่างกันเมื่อระบบงานตอบสนองแตกต่างกัน การกำหนดความแตกต่างกันขึ้นกับประเภทระบบงาน เราต้องสร้างสแตทโคอะแกรมก่อนจึงจะสามารถกำหนดอีเวนต์ทั้งหมดได้ ความแตกต่างระหว่างอีเวนต์บางอย่างจะไม่มีผลกับพฤติกรรมดังนั้นไม่เอาใจใส่ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กำหนดอีเวนต์แต่ละชนิดให้กับคลาสซึ่งส่งและรับอีเวนต์นั้น อีเวนต์เป็นแบบอินพุตสำหรับผู้รับและเป็นแบบเอาต์พุตสำหรับผู้ส่ง บางครั้งออกแบบเจดส่งอีเวนต์ให้กับตนเอง ในกรณีนี้อีเวนต์จะเป็นทั้งอินพุตและเอาต์พุตสำหรับคลาสเดียวกัน

แสดงแต่ละภาพสถานการณ์เหมือนเป็นทางเดินของอีเวนต์ รายการคำสั่งของอีเวนต์ระหว่างออกแบบเจดต่างๆกำหนดในคอลัมน์ของตาราง ถ้ามีออกแบบเจดมากกว่าหนึ่งออกแบบเจดของคลาสที่มีส่วนร่วมในภาพสถานการณ์เดียวกัน กำหนดเป็นคอลัมน์ที่แยกกันสำหรับแต่ละออกแบบเจดโดยการพิจารณาคอลัมน์ที่ระบุเฉพาะในทางเดิน เราสามารถมองเห็นอีเวนต์ซึ่งมีอิทธิพลโดยตรงเหนือออกแบบเจดที่ระบุเฉพาะได้เฉพาะอีเวนต์เหล่านี้เท่านั้น ซึ่งจะปรากฏในสเตทไดอะแกรมสำหรับออกแบบเจด

แผนผังการไหลของอีเวนต์ (event flow diagram) แสดงอีเวนต์ระหว่างกลุ่มของคลาส (เช่น โมดูล) แผนผังนี้สรุปอีเวนต์ระหว่างคลาสโดยไม่สนใจการเรียงลำดับคือรวมอีเวนต์จากภาพสถานการณ์ทั้งหมดและอีเวนต์ของความผิดพลาด แผนผังการไหลของอีเวนต์คือสำเนาที่เปลี่ยนแปลงอยู่เสมอของออกแบบเจดไดอะแกรม เส้นทางในออกแบบเจดไดอะแกรมแสดงการไหลของข่าวสารที่เป็นไปได้ แต่เส้นทางในแผนผังการไหลของอีเวนต์แสดงการไหลของการควบคุมที่เป็นไปได้

4.4.2 การสร้างสเตทไดอะแกรม

เตรียมสเตทไดอะแกรมสำหรับแต่ละคลาสกับพฤติกรรมที่เปลี่ยนแปลงอยู่เสมอ แสดงอีเวนต์ที่ออกแบบเจดรับและส่ง ทุกภาพสถานการณ์หรือทางเดินของอีเวนต์ตรงกับเส้นทางในสเตทไดอะแกรม แต่ละกึ่งในการไหลของการควบคุมถูกแทนที่โดยสเตทซึ่งการเปลี่ยนสเตทมีทางออกมากกว่าหนึ่งทาง เริ่มต้นด้วยแผนผังเส้นทางอีเวนต์ซึ่งมีอิทธิพลเหนือคลาสที่จะจำลองขึ้น เลือกเส้นทางที่แสดงการกระทำโต้ตอบกันและอีเวนต์ที่มีอิทธิพลเหนือออกแบบเจดเพียงออกแบบเจดเดียว จัดอีเวนต์เข้าไปในเส้นทางและเขียนชื่อของอีเวนต์ที่เป็นแบบอินพุตหรือเอาต์พุตซึ่งได้มาจากคอลัมน์ในแผนผังเส้นทางอีเวนต์ สเตทจะอยู่ระหว่างสองอีเวนต์ ใส่ชื่อของสเตทแต่ละสเตท ถ้าชื่อมีความหมายแต่ถ้าไม่มีก็ไม่ต้องกังวล แผนผังเริ่มแรกจะเป็นลำดับของอีเวนต์และสเตท ถ้าภาพสถานการณ์สามารถทำซ้ำได้อย่างไม่จำกัดให้ปิดเส้นทางในสเตทไดอะแกรม

หาวงรอบในแผนผัง ถ้าลำดับของอีเวนต์สามารถทำซ้ำได้อย่างไม่จำกัดดังนั้นมันจะอยู่ในรูปของวงรอบ (loop) แทนที่ลำดับของอีเวนต์ด้วยวงรอบเมื่อเป็นไปได้ ในวงรอบสเตทแรกและสเตทสุดท้ายเหมือนกันทุกอย่าง แต่ถ้าออกแบบเจด "จำได้" ว่ามันได้ข้ามวงรอบ ดังนั้นสองสเตท

ไม่ได้เหมือนกันทุกอย่างจริงและวงรอบแบบธรรมดาไม่น่าถูกต้อง อย่างน้อยที่สุดสเตทในวงรอบต้องมีหลายทรานแซคชัน (transaction) ออกจากออบเจกต์หรือไม่เช่นนั้นวงรอบนั้นจะไม่มีวันเล็กลง

รวมภาพสถานการณ์อื่นๆ เข้าในสเตทไดอะแกรม หากดูในแต่ละภาพสถานการณ์ซึ่งมันแผ่ออกจากภาพสถานการณ์ก่อนหน้านั้น จุดนี้ตรงกับสเตทที่มีอยู่แล้วในไดอะแกรม อีเวนต์ใหม่ที่เกิดขึ้นตามหลังสเตทที่มีอยู่แล้วคือเส้นทางที่ให้เลือกได้ ในขณะที่ตรวจสอบสเตทและภาพสถานการณ์ เราอาจจะคิดถึงอีเวนต์ที่เป็นไปได้อื่นๆ ซึ่งสามารถเกิดขึ้นได้ที่แต่ละสเตท เพิ่มอีเวนต์เหล่านั้นเข้าไปในสเตทไดอะแกรมด้วย

สิ่งที่ยากที่สุดคือการตัดสินใจว่าสเตทไหนที่เส้นทางที่เลือกได้มารวมกับไดอะแกรมที่มีอยู่แล้ว เส้นทางสองเส้นทางมารวมกันที่สเตทถ้าออบเจกต์ "ลิม" ว่าเป็นเส้นทางไหนในหลายกรณีจากระบบงานปรากฏอย่างชัดเจนว่าสองสเตทเหมือนกันทุกอย่าง ตัวอย่างเช่น ใส่ 2 nikels ในเครื่องขายของเท่ากับใส่ 1 dime

ระวังเส้นทางสองเส้นทางที่ปรากฏเหมือนกันทุกอย่าง แต่สามารถที่จะแยกความแตกต่างได้ภายใต้พฤติกรรมบางอย่าง ตัวอย่างเช่น บางระบบทำซ้ำลำดับการใส่ข้าวสารถ้าผู้ใช้ใส่ผิดจะยกเลิกการทำงานหลังจำนวนครั้งจำนวนหนึ่งที่ทำผิด ลำดับการทำซ้ำเกือบเหมือนกันเว้นแต่มันจำความผิดพลาดที่ผ่านไปได้ ความแตกต่างสามารถทำได้โดยการเพิ่มพารามิเตอร์ไว้จำข้าวสาร เช่นจำนวนความผิดพลาด อย่างน้อยที่สุดการเปลี่ยนสเตทต้องขึ้นกับค่าของพารามิเตอร์การใช้พารามิเตอร์ที่ฉลาดและการเปลี่ยนสเตทโดยมีเงื่อนไขสามารถทำให้สเตทไดอะแกรมง่ายลงอย่างมากมาย แต่ต้องเสียค่าใช้จ่ายในการรวมข้าวสารของสเตทและข้อมูล สเตทไดอะแกรมที่ข้อมูลมีความไม่เป็นอิสระมากเกินไปจะทำให้สับสนและไม่เป็นดังที่คิด สิ่งที่ให้เลือกอื่นๆ คือการแบ่งสเตทไดอะแกรมเป็นสองซับไดอะแกรม (subdiagram) ที่เกิดขึ้นพร้อมๆ กัน ใช้ส่วนหนึ่งสำหรับแนวทางหลักและอีกส่วนสำหรับแยกความแตกต่างข้าวสาร ตัวอย่างเช่น ซับไดอะแกรมที่ยอมสำหรับความผิดพลาดของผู้ใช้อาจจะไม่มีข้อผิดพลาดและมีข้อผิดพลาด

ภายหลังที่คิดถึงอีเวนต์ปกติแล้วเพิ่มกรณีที่อยู่ในขอบเขตและกรณีพิเศษ คิดถึงอีเวนต์ที่เกิดขึ้นล่าช้า ตัวอย่างเช่น การร้องขอเพื่อจะยกเลิกทรานแซคชันภายหลังที่ได้ส่งเข้ามาสำหรับการทำงานแล้วในกรณีเมื่อผู้ใช้ (หรือผู้กระทำภายนอกอื่นๆ) อาจจะไม่ได้รับการตอบสนองและทรัพยากร (resource) บางอย่างต้องถูกเรียกคืน และอีเวนต์นั้นหมดเวลา การจับข้อผิดพลาดของผู้ใช้มักจะต้องการความคิดและการบันทึกมากกว่ากรณีปกติ การจับข้อผิดพลาดมักจะทำให้โครงสร้างของโปรแกรมสับสนแต่จำเป็นต้องทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สเตตไดอะแกรมของคลาสทำสำเร็จเมื่อไดอะแกรมครอบคลุมภาพสถานการณ์ทั้งหมดและไดอะแกรมเก็บอีเวนท์ทั้งหมดที่สามารถมีอิทธิพลเหนือขอบเขตของคลาสในแต่ละสเตตของขอบเขต เราสามารถใช้สเตตไดอะแกรมเสนอแนะภาพสถานการณ์ใหม่โดยการคิดว่าบางอีเวนท์ที่ไม่ได้มีอยู่ในไดอะแกรมจะมีอิทธิพลเหนือสเตตของขอบเขตอย่างไร การวางคำถามว่า 'อะไรถ้า' คือวิธีที่ดีที่จะทดสอบความครบถ้วนและขีดความสามารถในการจับข้อผิดพลาดของคลาส (และอาจจะทำซ้ำที่ในระดับโมดูลและระบบได้ด้วย)

ถ้ามีการกระทำโต้ตอบกันที่ซับซ้อนกับการอินพุตที่ไม่ขึ้นต่อกันเราสามารถจัดไดนามิกโมเดลโดยใช้ nested state diagram หรือมีอะนั่นสเตตไดอะแกรมแบบธรรมดาาก็เพียงพอแล้วถ้ามันไม่ซับซ้อน

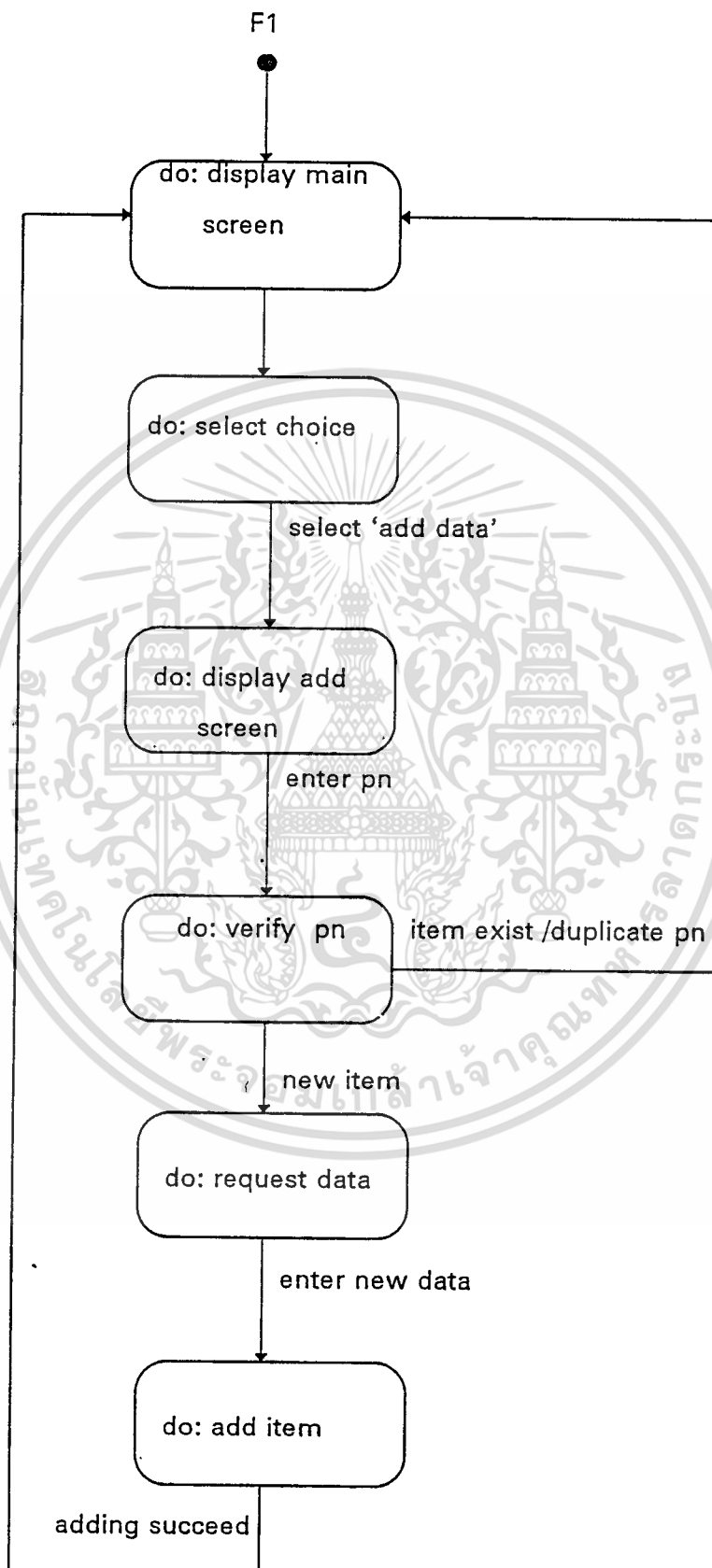
ให้ทำซ้ำตามวิธีการสร้างสเตตไดอะแกรมที่กล่าวมาข้างต้น สำหรับแต่ละคลาสของขอบเขต เฟ่งเล็งเฉพาะคลาสที่มีการกระทำโต้ตอบที่สำคัญ ไม่ต้องเขียนสเตตไดอะแกรมสำหรับทุกคลาส ขอบเขตหลายขอบเขตที่ตอบสนองต่ออีเวนท์ที่เป็นอินพุตอย่างเป็นอิสระ ประวัติที่ผ่านมาของขอบเขตที่ไม่ได้มีอิทธิพลเหนือการควบคุมคือพารามิเตอร์ ขอบเขตอาจจะรับหรือส่งอีเวนท์ เขียนรายชื่อของอีเวนท์ที่เป็นแบบอินพุตสำหรับแต่ละขอบเขต และอีเวนท์ที่เป็นแบบเอาต์พุตส่งไปตอบสนองต่ออีเวนท์ที่เป็นแบบอินพุต

ในที่สุดเราอาจจะเขียนสเตตไดอะแกรมได้โดยไม่ต้องเตรียมเส้นทางอีเวนท์ โดยปกติภาพสถานการณ์เพียงเล็กน้อยก็มีประโยชน์ในบางกรณี

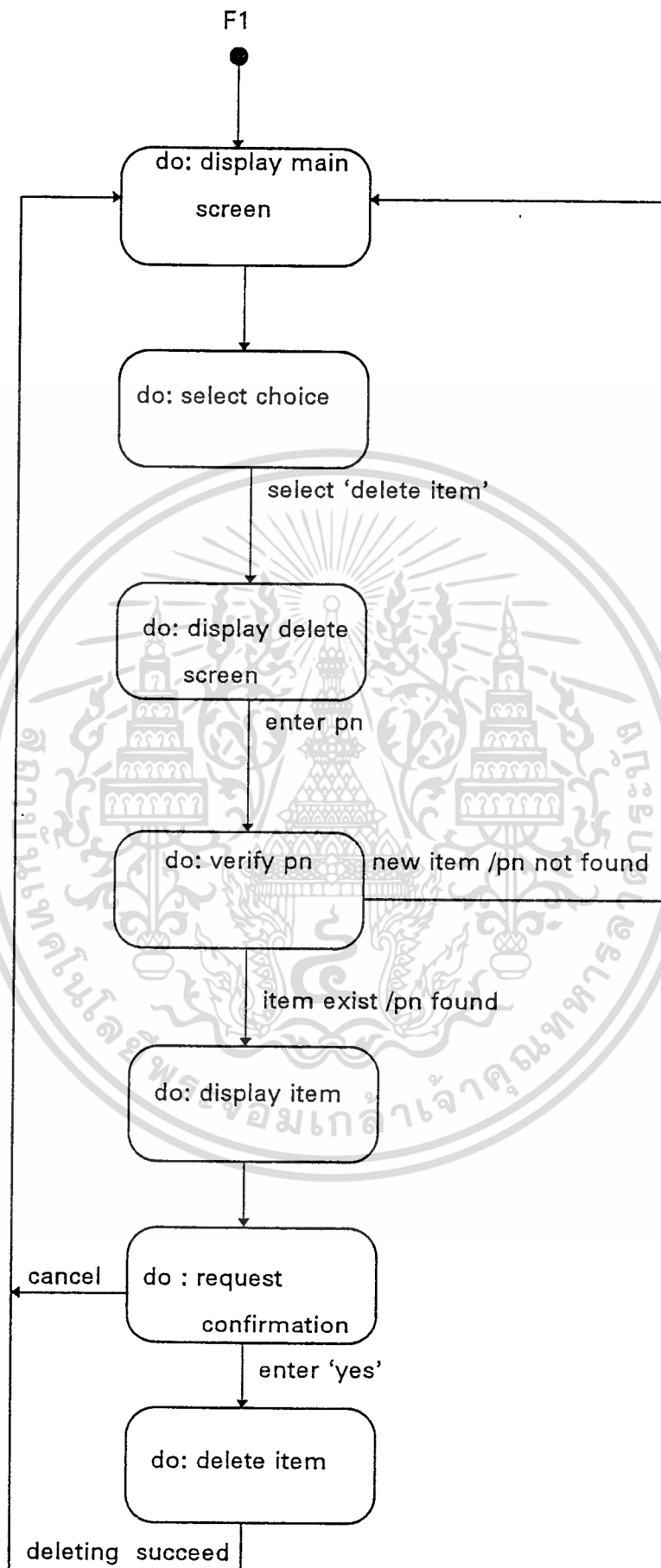
ตรวจสอบความครบถ้วนและคงเส้นคงวาเมื่อสร้างสเตตไดอะแกรมสำหรับแต่ละคลาสเสร็จ ทุกอีเวนท์ต้องมีผู้ส่งและผู้รับ บางครั้งทั้งผู้ส่งและผู้รับคือคลาสเดียวกัน สเตตที่ไม่มีผู้ให้กำเนิดและผู้รับต่อนี้นำสงสัยถ้ามั่นใจว่าสเตตเหล่านั้นไม่ใช่จุดเริ่มต้นหรือจุดจบของลำดับการกระทำโต้ตอบกัน ติดตามผลของอีเวนท์ที่อินพุตจากขอบเขตถึงขอบเขตตลอดทั้งระบบ เพื่อให้เกิดความมั่นใจว่ามันเข้ากันได้กับภาพสถานการณ์ ขอบเขตที่เกิดขึ้นพร้อมๆ กันให้ระวังข้อผิดพลาดของการทำพร้อมกันเมื่อมีอินพุตที่ล่าช้าไป ต้องมั่นใจว่าการตรงกันของอีเวนท์บนสเตตไดอะแกรมต่างๆ มีความคงเส้นคงวา กลุ่มของสเตตไดอะแกรมสำหรับคลาสที่มีพฤติกรรมสำคัญที่เปลี่ยนแปลงอยู่เสมอสร้างไดนามิกโมเดลสำหรับระบบงาน

รายละเอียดของสเตตไดอะแกรมสำหรับขอบเขตแต่ละขอบเขตในระบบงานบริหารพัสดุสายช่างอากาศแสดงอยู่ในภาพที่ 35

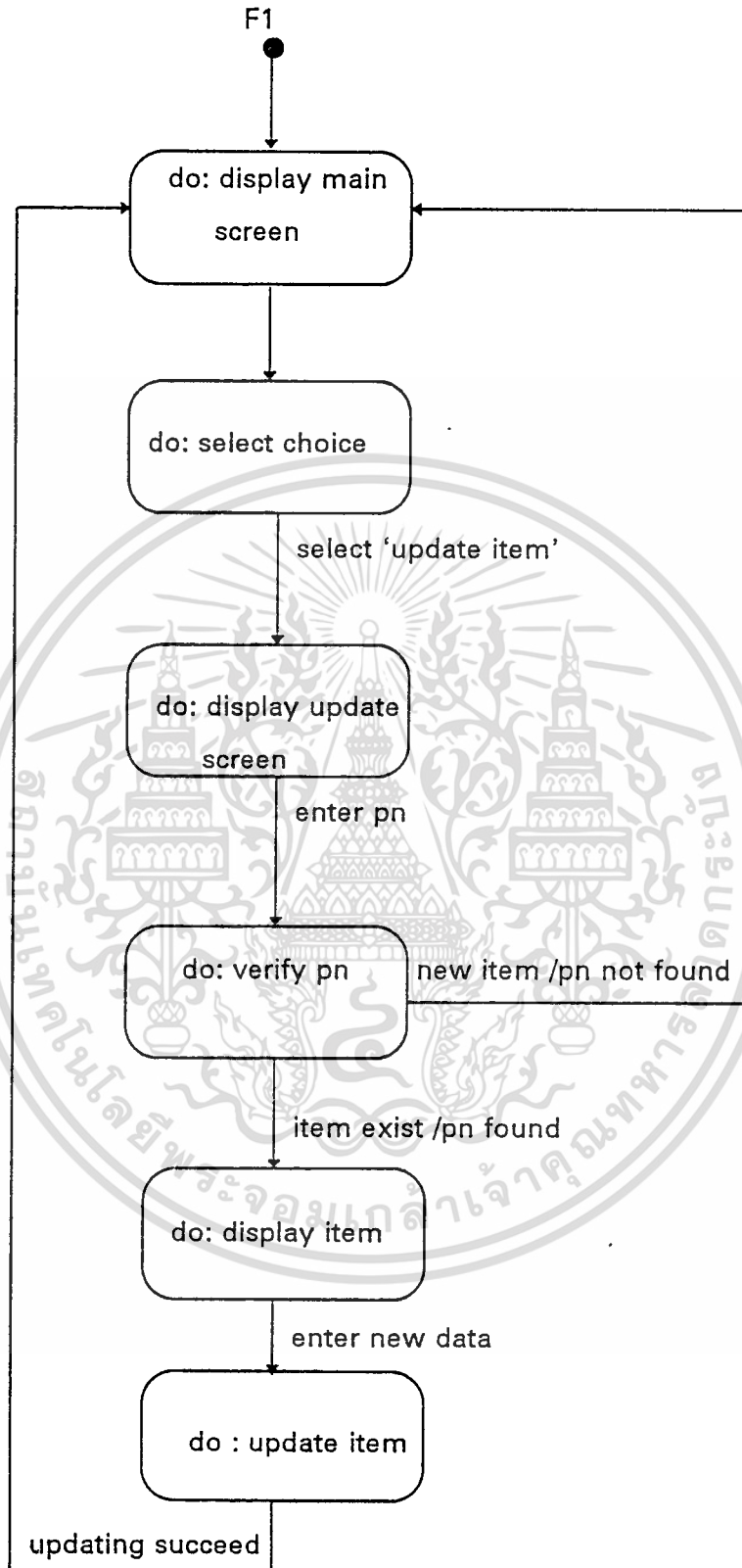
ภาพที่ 35 ไดนามิกโมเดล



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้ภายในเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 สเดทไดอะแกรมของคลาส AED
 ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

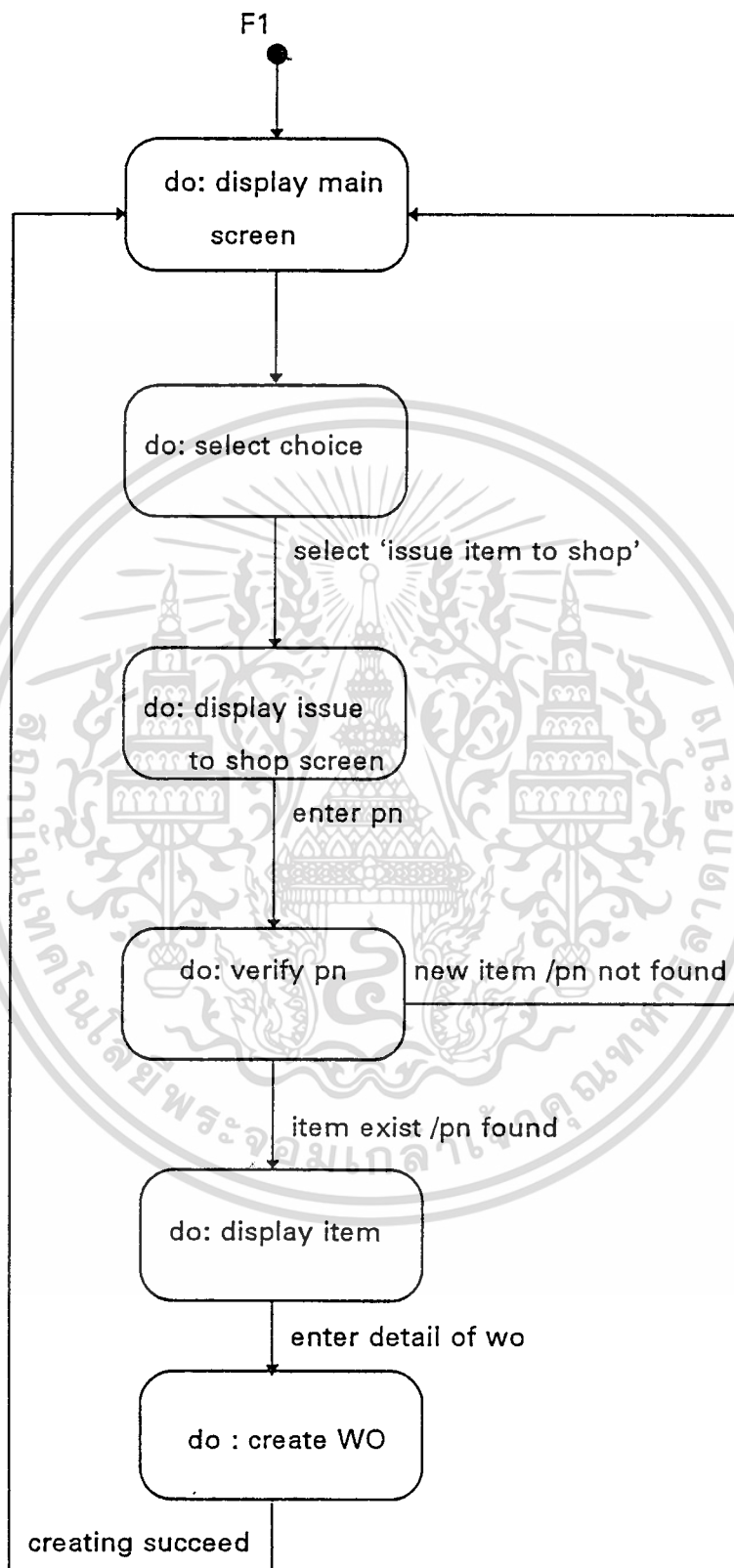


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้ภายในระบบเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

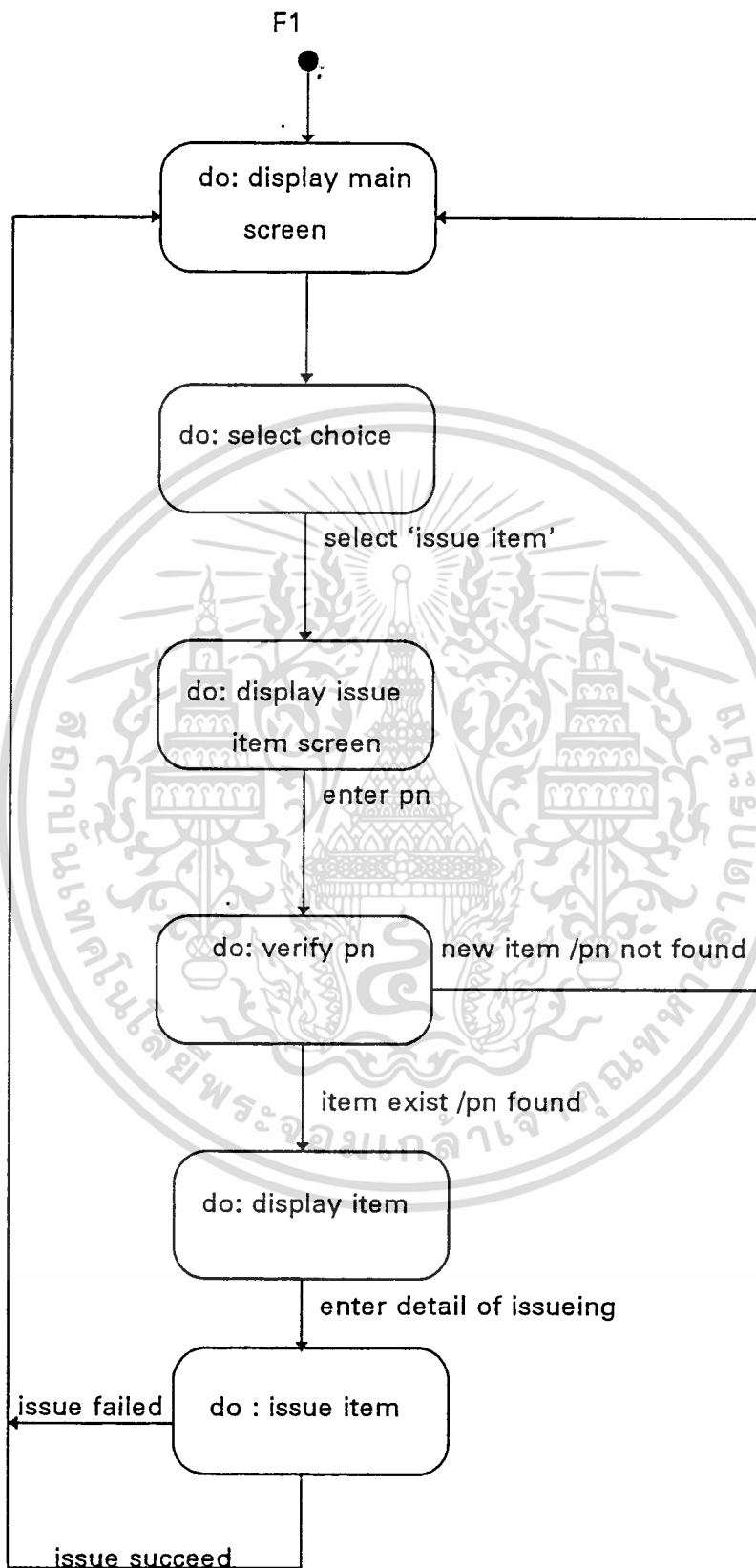


สแตทไดอะแกรมของคลาส AED

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

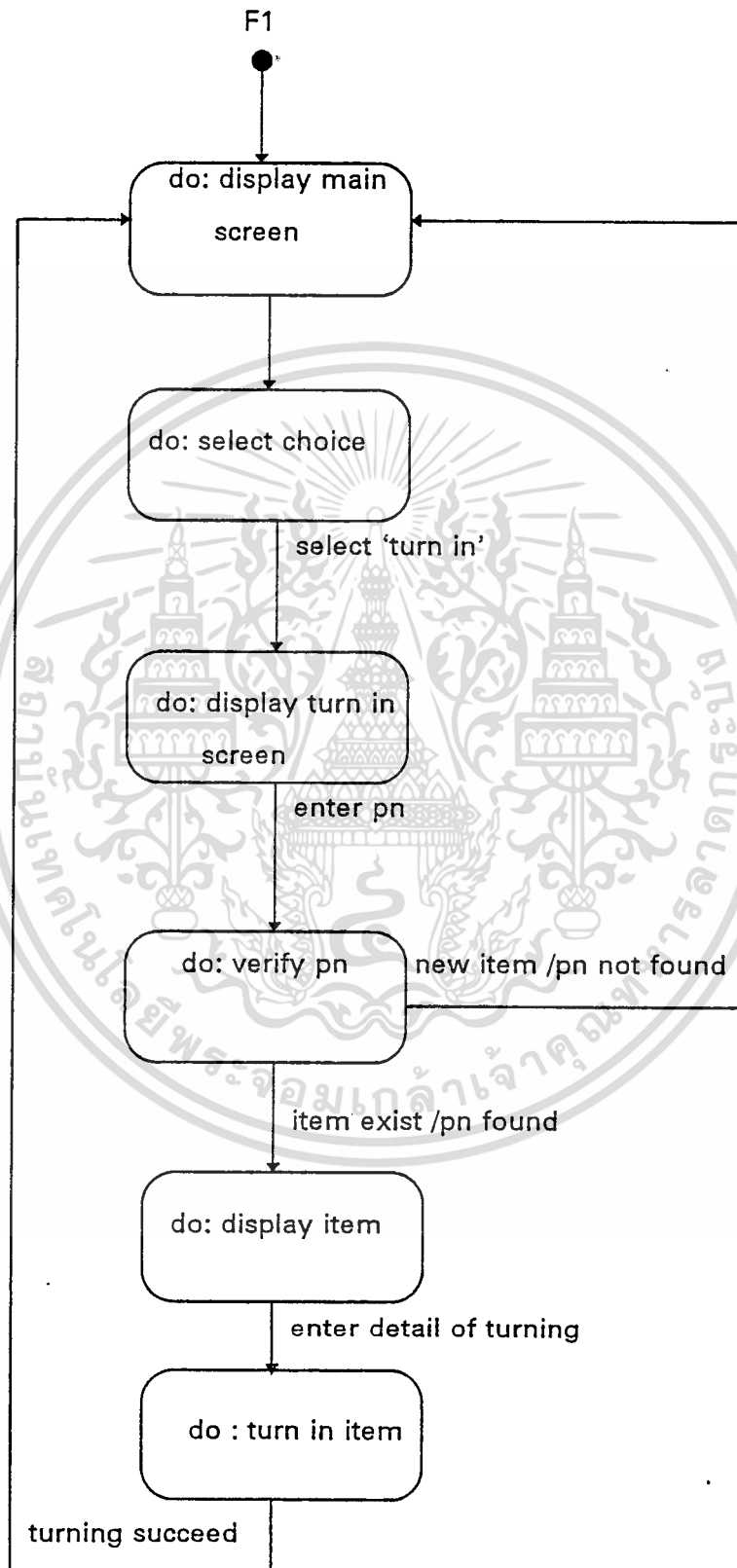


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้โดยนักเรียนของคลาส AED ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

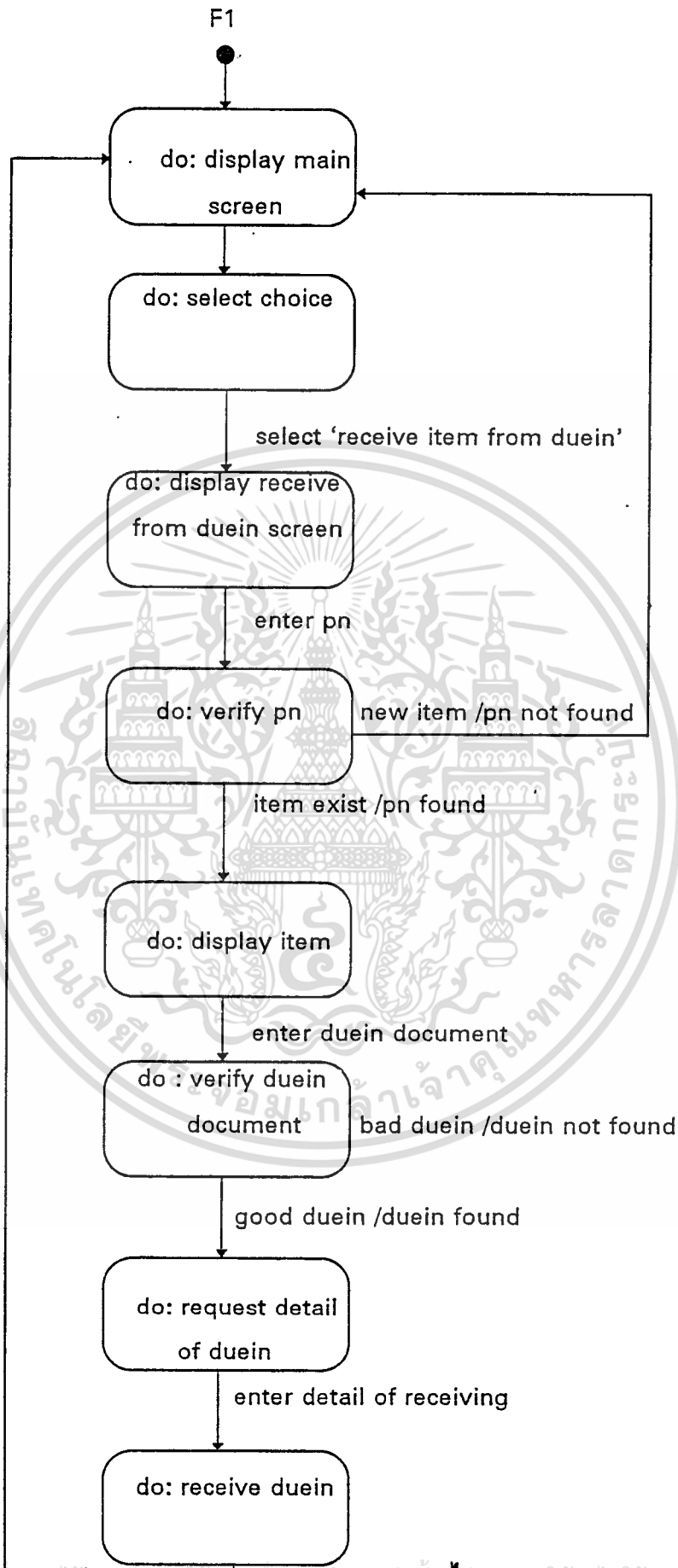


สแตทไดอะแกรมของคลาส AED

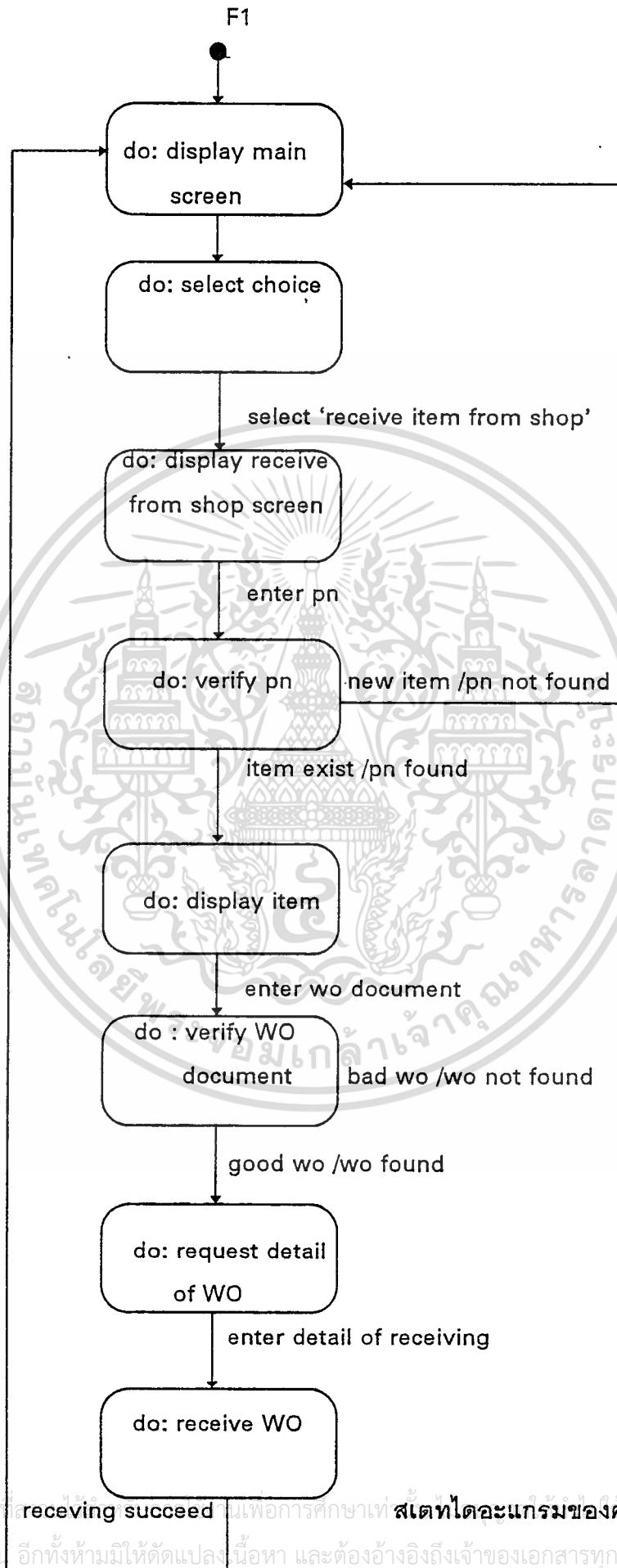
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



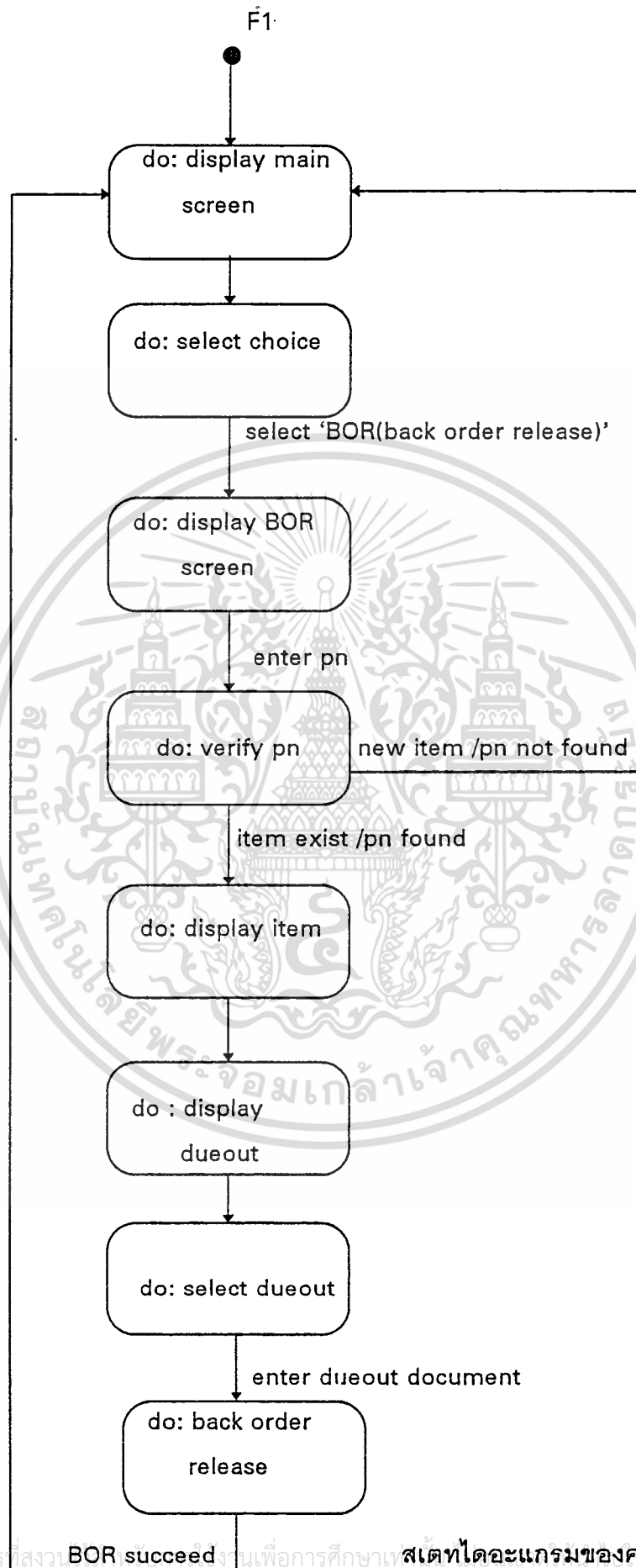
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้เฉพาะในหน่วยงานของ AED เท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



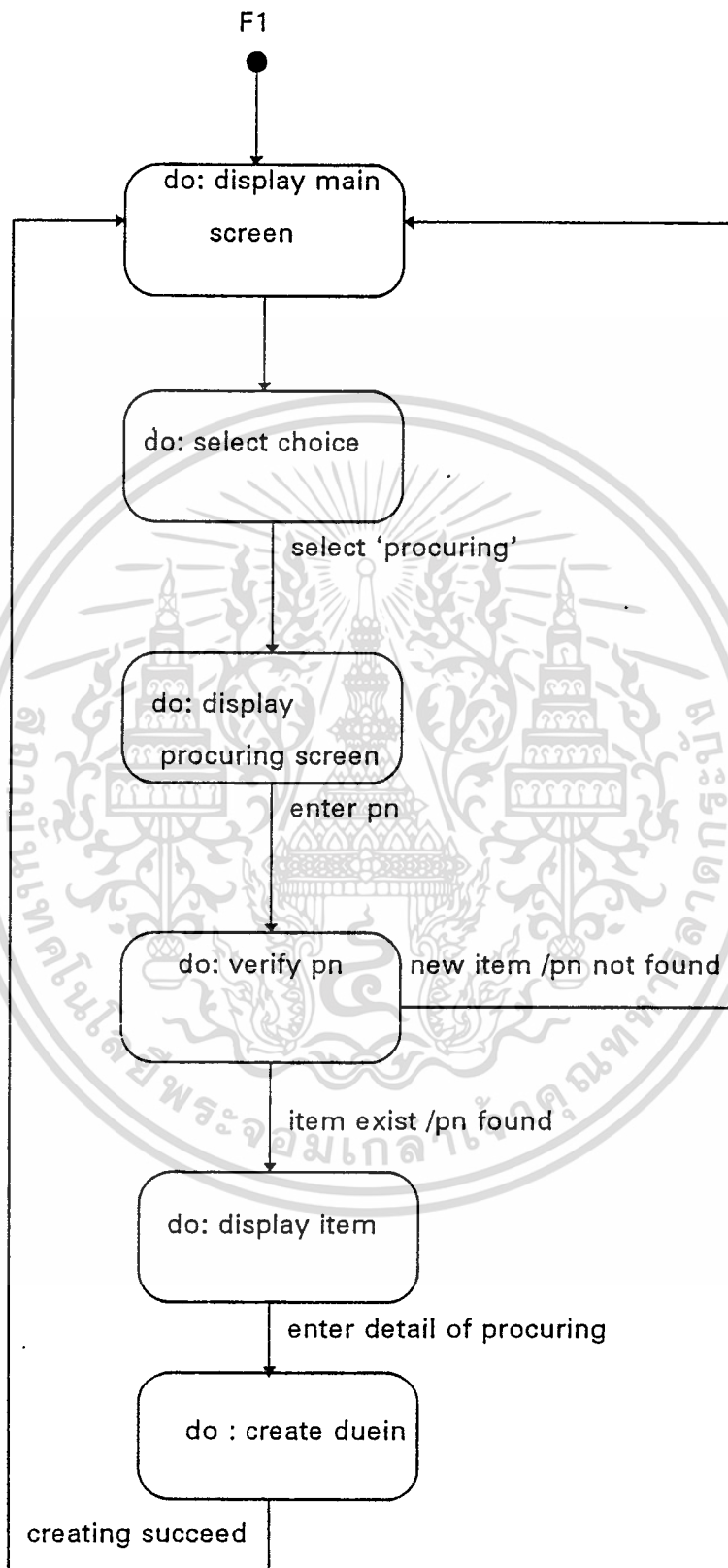
เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ไว้เพื่อการศึกษาเท่านั้น ไม่สามารถนำข้อมูลไปใช้ในการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



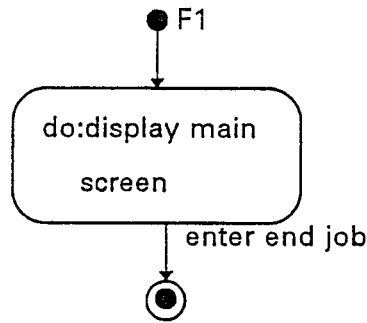
เอกสารนี้เป็นเอกสารเพื่อการศึกษาเท่านั้น
 ไม่ว่ากรณีใดๆทั้งสิ้น
 อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



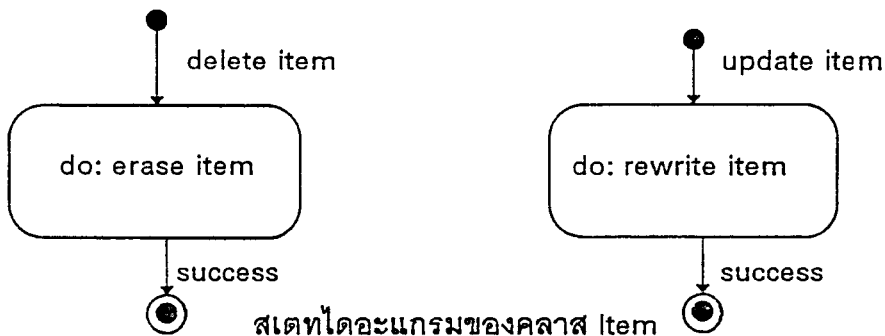
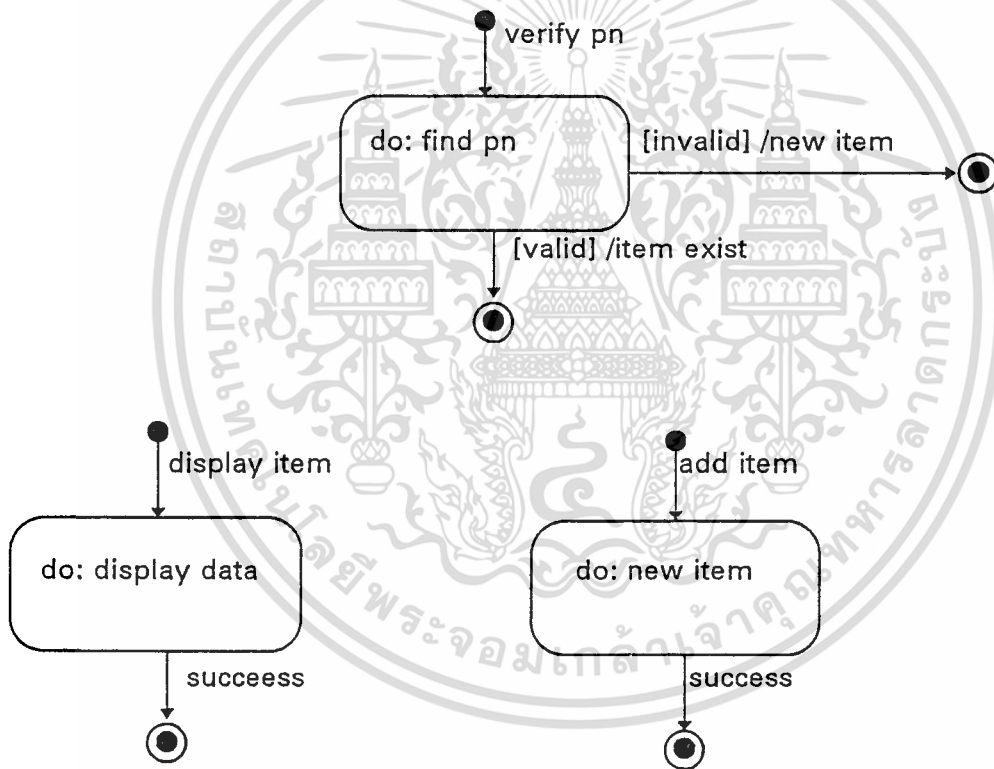
เอกสารนี้เป็นเอกสารที่สงวน BOR succeed ใช้งานเพื่อการศึกษาเท่านั้น ไม่สามารถเผยแพร่โดยไม่ขออนุญาตของ AED
 ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้เฉพาะของคณะ AED ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

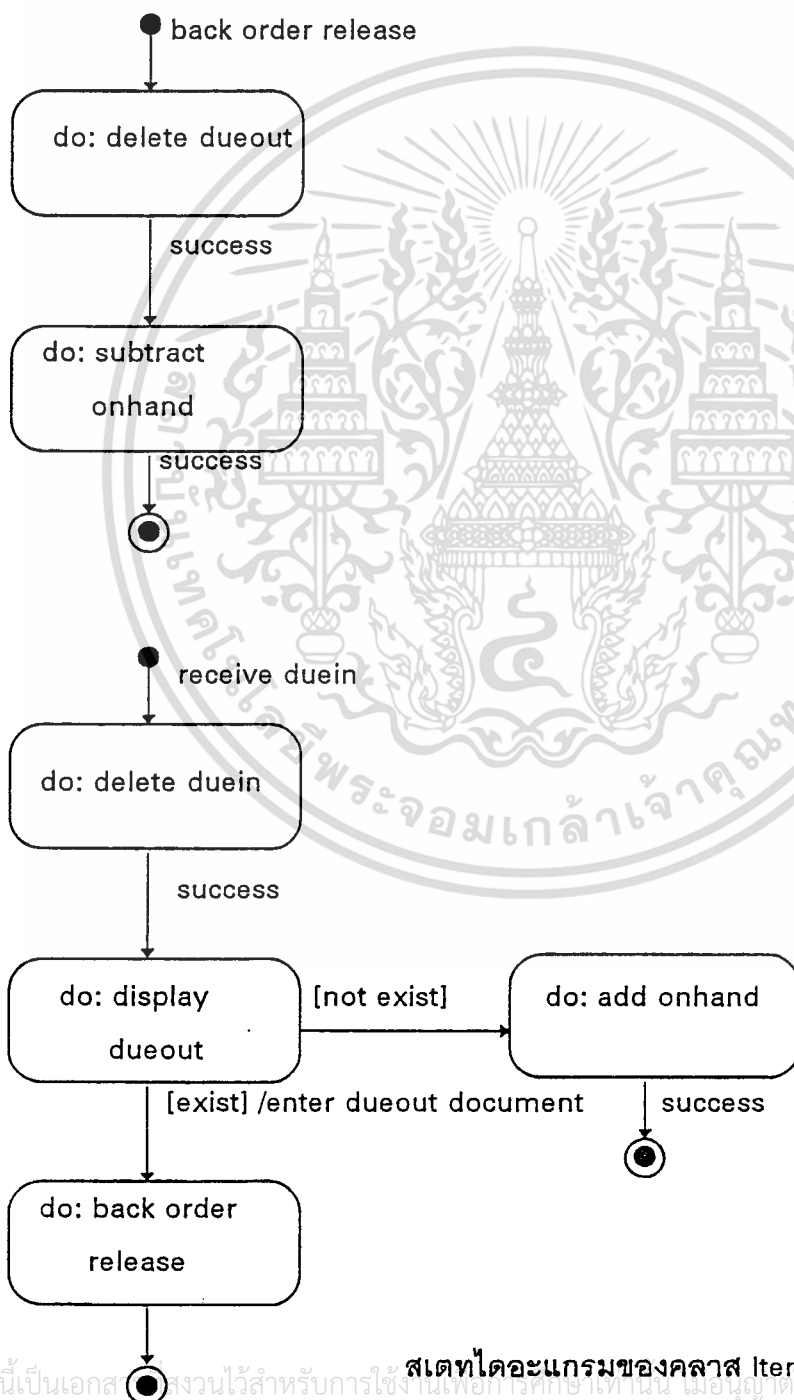
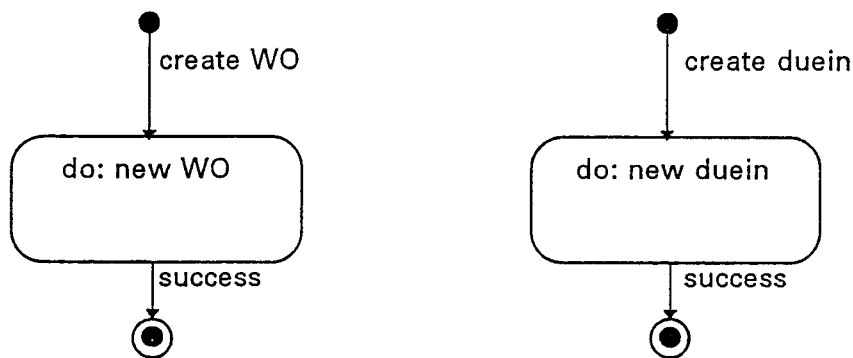


สเตตโด้อะแกรมของคลาส AED



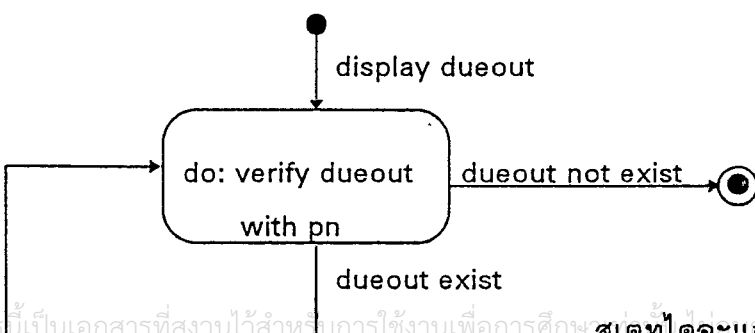
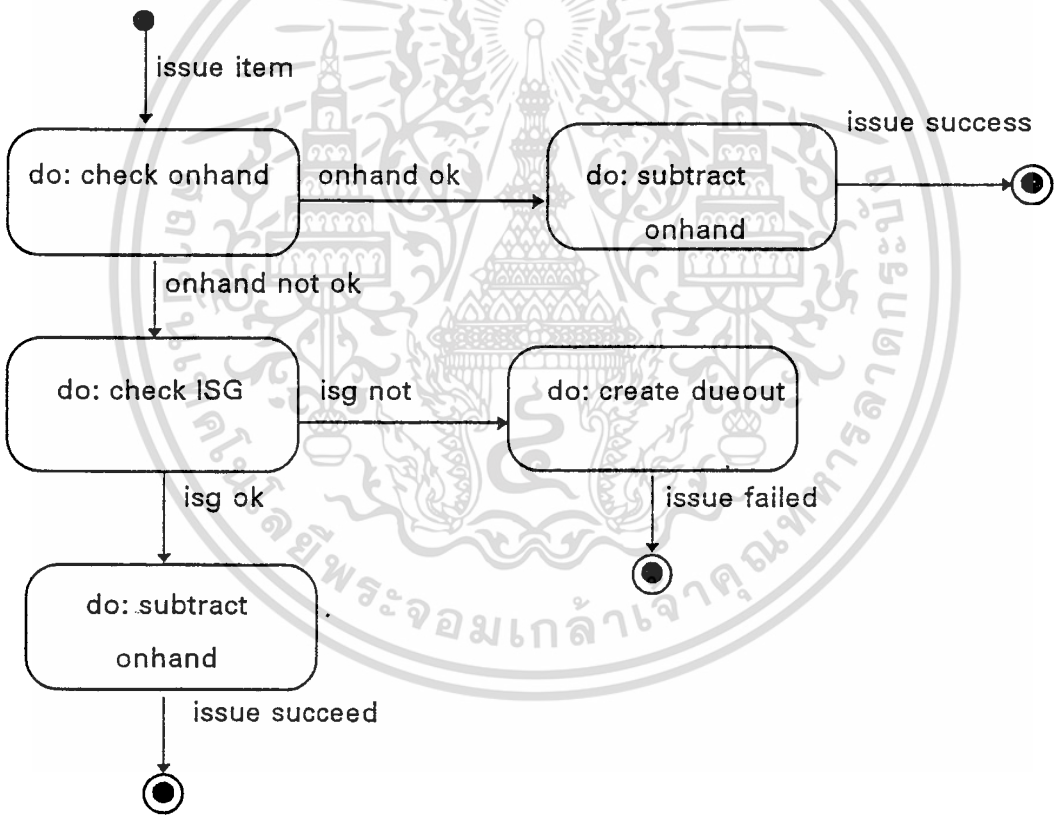
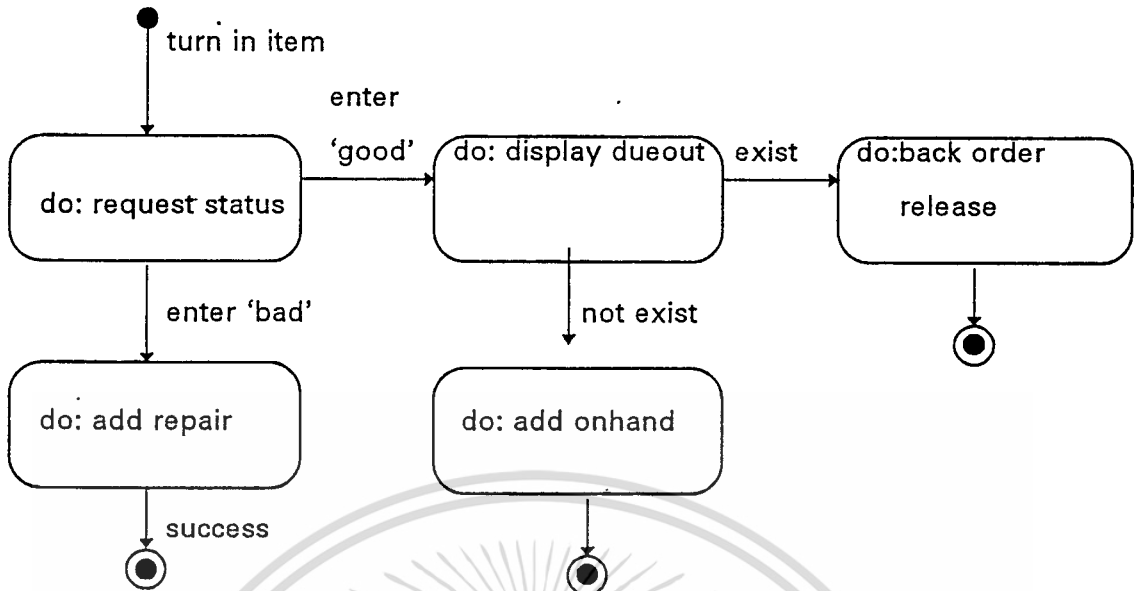
สเตตโด้อะแกรมของคลาส Item

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

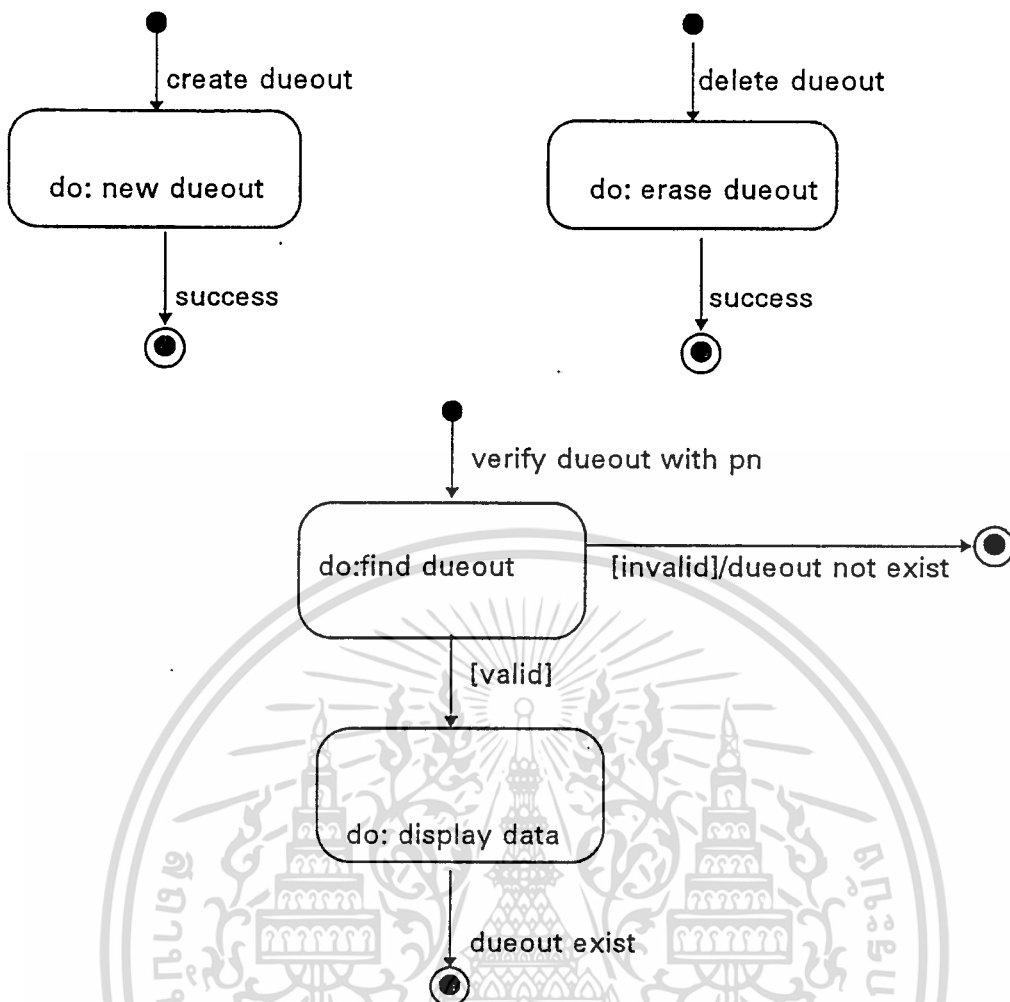


สแตตโคอะแกรมของคลาส Item

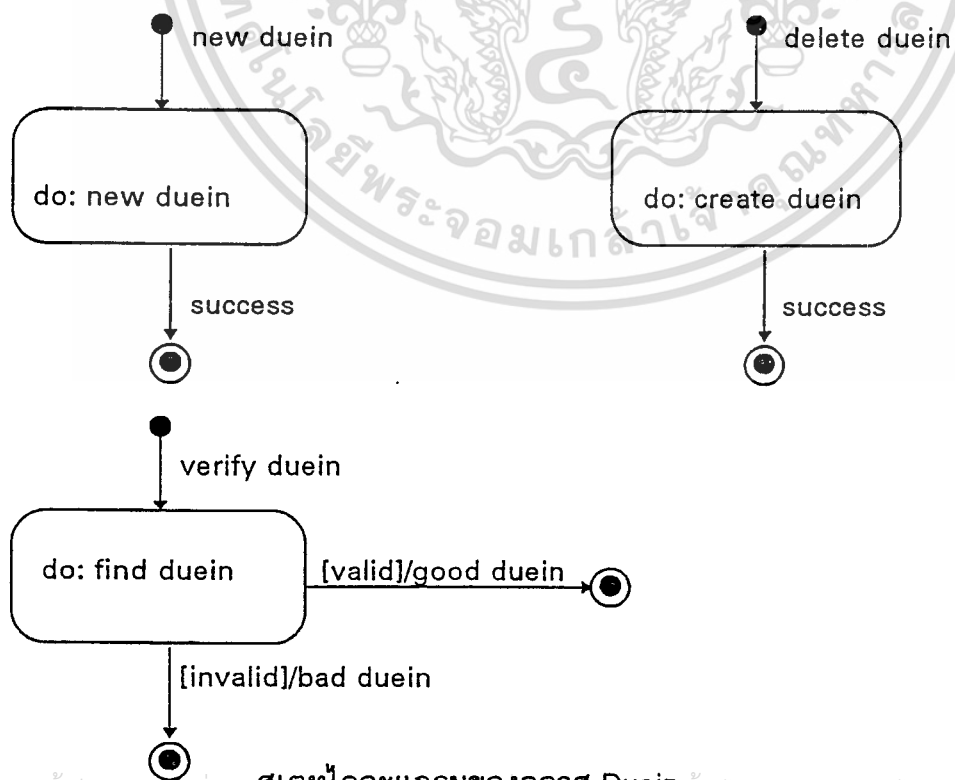
เอกสารนี้เป็นเอกสารสงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น เมื่ออนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษา **สเตตัสและแกรมของคลาส Item** ด้านการค้า
 ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

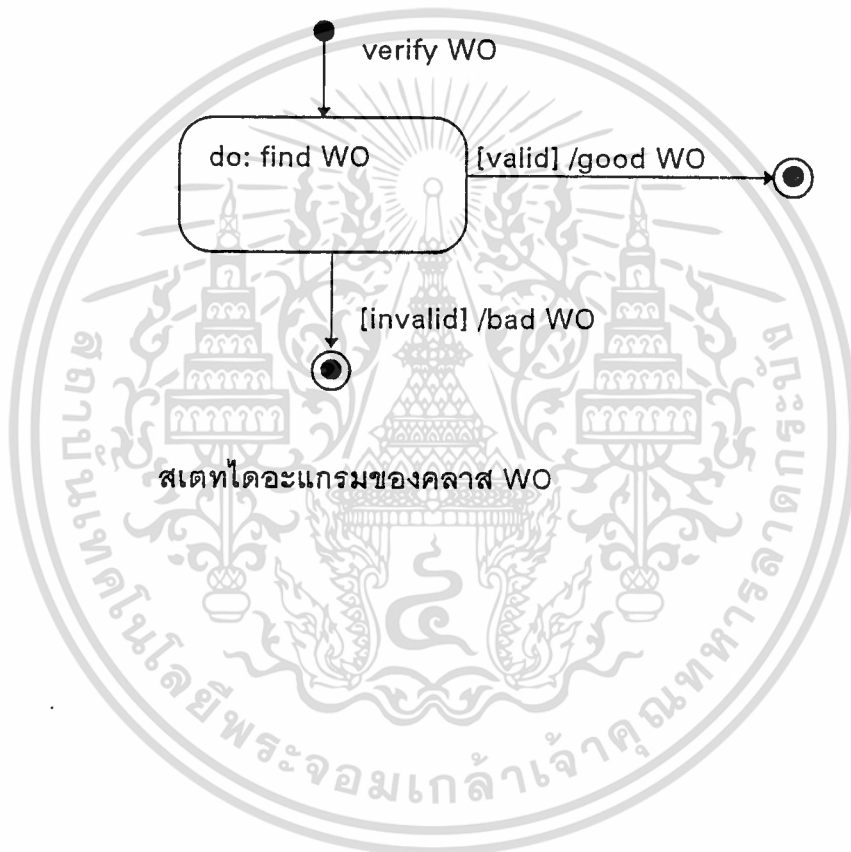
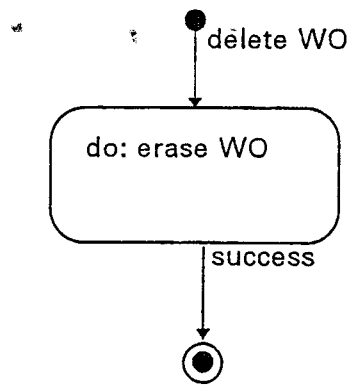
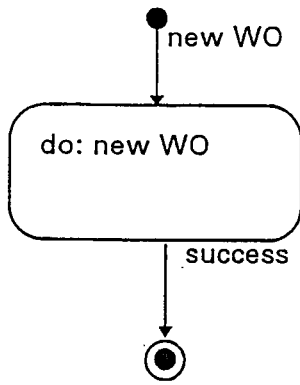


สเตตโตะแกรมของคลาส Dueout



สเตตโตะแกรมของคลาส Duein

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์หรือการเขียนเพื่อการค้าเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 ฟังก์ชันนัลโมเดลลิ่ง (Functional modelling)

ฟังก์ชันนัลโมเดลแสดงการคำนวณค่าว่าได้มาอย่างไร โดยไม่สนใจลำดับ การตัดสินใจ, หรือโครงสร้างของออบเจกต์ ฟังก์ชันนัลโมเดลแสดงค่าซึ่งขึ้นกับค่าและการกระทำอื่นๆ ที่มีความเกี่ยวพันกัน ดาต้าไฟล์ไดอะแกรม(Data flow diagram) มีประโยชน์สำหรับแสดงการกระทำที่ไม่มีอิสระ การกระทำแสดงได้หลายทาง ได้แก่ ภาษาธรรมชาติ สมการทางคณิตศาสตร์ และซูโดโคด (pseudocode)

โพรเซส(process) ในดาต้าไฟล์ไดอะแกรมตรงกับแอคทิวิตีหรือแอคชันในสเตทไดอะแกรมของคลาส การไหล (flow) บนดาต้าไฟล์ไดอะแกรมตรงกับออบเจกต์หรือค่าของแอทริบิวต์ในออบเจกต์ไดอะแกรม วิธีที่ดีที่สุดคือสร้างฟังก์ชันนัลโมเดลภายหลังสร้างออบเจกต์และไดนามิกโมเดลเสร็จแล้ว

ขั้นตอนการสร้างฟังก์ชันนัลโมเดลมีดังต่อไปนี้

- กำหนดอินพุตและเอาต์พุต
- สร้างดาต้าไฟล์ไดอะแกรมแสดงความขึ้นต่อกันของฟังก์ชัน
- กำหนดข้อบังคับ (constraints)

4.5.1 การกำหนดอินพุตและเอาต์พุต

เริ่มต้นโดยการเขียนรายการอินพุตและเอาต์พุต อินพุตและเอาต์พุตคือพารามิเตอร์ของอินเทอร์เฟซระหว่างระบบกับโลกภายนอก ตรวจสอบประโยคปัญหาเพื่อหาค่าอินพุตหรือค่าที่เราทำขาดหายไป

4.5.2 สร้างดาต้าไฟล์ไดอะแกรม

สร้างดาต้าไฟล์ไดอะแกรม แสดงเอาต์พุตว่าถูกคำนวณจากอินพุตได้อย่างไรดาต้าไฟล์ไดอะแกรมโดยปกติจะถูกสร้างเป็นชั้น ชั้นบนสุดอาจจะประกอบด้วยโพรเซสเดียวหรือบางทีหนึ่งโพรเซสรวมอินพุต ค่าวนควม และสร้างผลลัพธ์ ในแต่ละชั้นของดาต้าไฟล์ไดอะแกรมใช้วิธีการทำงานถอยหลังจากแต่ละเอาต์พุตเพื่อแสดงฟังก์ชันที่คำนวณเอาต์พุต ถ้าการอินพุตไปยังโอเปอเรชันคือการอินพุตทั้งหมดของทั้งโปรแกรมที่เราทำ มิฉะนั้นบางส่วนของอินพุตที่เป็นโอเปอเรชันคือค่าที่อยู่ระหว่างกลางซึ่งต้องถูกลากเส้นถอยหลัง เราสามารถใช้วิธีการเดินหน้าจากการอินพุตถึงการเอาต์พุตได้ แต่ปกติมันยากที่จะกำหนดการใช้การอินพุตทั้งหมดมากกว่าที่จะกำหนดทรัพยากรทั้งหมดของการเอาต์พุต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กระจายแต่โพรเซสที่มีขนาดใหญ่ในโปรแกรมระดับบนสุดไปเป็นดาต้าไฟล์ไดอะแกรมระดับต่ำกว่า ถ้าโพรเซสระดับที่สองยังคงประกอบด้วยโพรเซสที่มีขนาดใหญ่อยู่สามารถกระจายออกไปได้อีก

ระบบส่วนมากมีออบเจกต์ภายในที่เป็นที่เก็บข้อมูล ซึ่งเก็บค่าไว้ได้ระหว่างการทำซ้ำ ที่เก็บข้อมูลภายในสามารถที่จะแยกความแตกต่างกับการไหลของข้อมูลหรือโพรเซสได้ เพราะว่าที่เก็บรับข้อมูลซึ่งไม่ได้ให้ผลลัพธ์เป็นการเอว้พทุทในทันทีทันใดแต่ถูกใช้ในเวลาที่ต่างกันในอนาคต

ดาต้าไฟล์ไดอะแกรมกำหนดความไม่ขึ้นต่อกันระหว่างโอเปอเรเตอร์เท่านั้น ไม่ได้แสดงการตัดสินใจ หรือลำดับของโอเปอเรเตอร์ ในความเป็นจริงบางโอเปอเรเตอร์สามารถจะให้เลือกได้หรือไม่ได้เกิดขึ้นพร้อมๆ กัน ตัวอย่างเช่น รหัสผ่านต้องถูกตรวจสอบก่อนที่จะแก้ไขบัญชี ถ้ารหัสไม่ถูกต้องดังนั้นก็แก้ไขบัญชี ลำดับการตัดสินใจนี้เป็นส่วนหนึ่งของไดนามิกโมเดลไม่ใช่ฟังก์ชันนัลโมเดล

ค่าของข้อมูลบางค่ามีอิทธิพลเหนือการตัดสินใจในไดนามิกโมเดล การตัดสินใจไม่ได้มีอิทธิพลเหนือเอว้พทุทโดยตรงในดาต้าไฟล์โมเดล ในขณะที่ดาต้าไฟล์โมเดลแสดงเส้นทางเกี่ยวกับการคำนวณที่เป็นไปได้ แต่อย่างไรก็ตามสามารถหาฟังก์ชันเกี่ยวกับการตัดสินใจได้ในดาต้าไฟล์โมเดลเมื่อฟังก์ชันเหล่านั้นอาจจะเป็นฟังก์ชันที่สับสนของอินพุท ฟังก์ชันเกี่ยวกับการตัดสินใจสามารถถูกแสดงบนดาต้าไฟล์ไดอะแกรม แต่เอว้พทุทของฟังก์ชันเกี่ยวกับการตัดสินใจคือสัญญาควบคุม ระบบด้วยลูกศรเอว้พทุทเป็นเส้นประ ฟังก์ชันเหล่านี้เป็น "ข้อมูลจมลง" ภายในดาต้าไฟล์ไดอะแกรม เอว้พทุทของฟังก์ชันเหล่านั้นมีอิทธิพลเหนือการไหลของการควบคุมในไดนามิกโมเดลและไม่ใช้เอว้พทุทโดยตรง ตัวอย่างเช่น การตรวจสอบรหัสผ่านคือฟังก์ชันเกี่ยวกับการตัดสินใจ เราแสดงสัญญาข้อมูลผิดพลาดซึ่งอาจจะถูกสร้างขึ้นมา แต่ปล่อยให้ลากลูกศรควบคุมถึงโพรเซสแก้ไขบัญชี ถ้าเราต้องการเราก็สามารถที่จะลากเส้นลูกศรควบคุมถึงโพรเซสที่ถูกควบคุมโดยการตัดสินใจได้

เมื่อดาต้าไฟล์ไดอะแกรมถูกขีดเกลากจนเพียงพอแล้ว เขียนคำอธิบายของแต่ละฟังก์ชัน คำอธิบายสามารถเขียนเป็นภาษาธรรมชาติ สมการคณิตศาสตร์ ชูโดโคด ตารางการตัดสินใจหรือรูปแบบอื่นๆ ที่เหมาะสม โดยเพ่งเล็งที่ฟังก์ชันทำอะไร ไม่ใช่ว่าจะทำให้เป็นจริงขึ้นได้อย่างไร คำอธิบายจะเป็นแบบบอกเล่าหรือแบบวิธีดำเนินการ คำอธิบายแบบบอกเล่ากำหนดความสัมพันธ์ระหว่างอินพุทและเอว้พทุท และความสัมพันธ์ระหว่างเอว้พทุท ตัวอย่างเช่น คำอธิบายของฟังก์ชัน "เรียงลำดับและลบค่าที่ซ้ำกันออก" อาจเป็นดังนี้คือ "ทุกๆ ค่าในรายการของอินพุทปรากฏอย่างแน่นอนเพียงครั้งเดียวในรายการเอว้พทุท รายการในเอว้พทุทมีค่ามาจากรายการอินพุท" เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

พูดเท่านั้น และคำในรายการเออาร์พหมีอันดับเพิ่มขึ้น" คำอธิบายแบบวิธีดำเนินการกำหนด ฟังก์ชันโดยให้อะกอร์ทิมของการคำนวณฟังก์ชันนั้น จุดประสงค์ของอะกอร์ทิมคือกำหนดว่า ฟังก์ชันทำอะไรเท่านั้น ในระหว่างการสร้างระบบอะกอร์ทิมอื่นๆ ซึ่งคำนวณค่าเดียวกันสามารถ นำมาใช้แทนกันได้ คำอธิบายแบบบอกเล่าดีกว่าคำอธิบายแบบวิธีดำเนินการ เพราะว่าคำ อธิบายแบบบอกเล่าไม่ได้สื่อให้เห็นวิธีการสร้างระบบ แต่คำอธิบายแบบวิธีดำเนินการเขียนได้ง่าย กว่า ซึ่งควรที่จะเลือกใช้วิธีนี้

4.5.3 การกำหนดข้อบังคับระหว่างออบเจค

กำหนดข้อบังคับระหว่างออบเจค ข้อบังคับคือความไม่เป็นอิสระแห่งการกระทำ ระหว่างออบเจคซึ่งไม่ได้เกี่ยวพันกันเนื่องจากความไม่เป็นอิสระของค่าอินพุตและค่าเออาร์พหมี ข้อ บังคับสามารถอยู่บนออบเจคสองออบเจคที่เวลาเดียวกัน, ระหว่างสมาชิกของออบเจคเดียวกันใน เวลาต่างกัน, หรือระหว่างสมาชิกของออบเจคที่แตกต่างกันในเวลาต่างกัน (สิ่งที่ตามมาภายหลัง โดยปกติคือการกระทำเกี่ยวกับการอินพุตและเออาร์พหมี) เงื่อนไขก่อนหน้าการกระทำคือข้อบังคับ ข้อมูลอินพุต และเงื่อนไขหลังการกระทำคือข้อบังคับข้อมูลเออาร์พหมี

กำหนดค่าที่เป็นค่าสูงสุด, ค่าต่ำสุด, หรือมีค่านั้นค่าที่ดี ถ้ามีมาตรฐานที่ดีหลายอย่างซึ่ง ขัดแย้งกันระบุว่าประโยชน์อย่างไรที่ต้องการ อย่างกังวลเกี่ยวกับการทำสิ่งนี้ให้ถูกต้องมากจนเกิน ไปนัก ปกติเราจะไม่สามารถทำได้ และมาตรฐานอาจเป็นไปได้ว่าเปลี่ยนก่อนที่จะทำโครงการ เสร็จ

4.5.4 การเพิ่มโอเปอเรชัน

วิธีของการวิเคราะห์เชิงวัตถุไม่เน้นการกำหนดโอเปอเรชัน รายการของโอเปอเรชันที่มี ประโยชน์ที่อาจเป็นไปได้คือเริ่มต้นและจบ มันยากที่จะรู้ว่าเมื่อไร ที่จะหยุดเพิ่มโอเปอเรชัน โอเปอเรชันในภาษาการเขียนโปรแกรมเชิงวัตถุตรงกับคำถาม (query) เกี่ยวกับแอมริบิวท์หรือแอส โซซิเอชันในออบเจคโมเดล อีเวนทีโนไดนามิกโมเดล และโพเรเซลในฟังก์ชันนัลโมเดล เราพบว่ามี ประโยชน์มากที่จะแยกชนิดความแตกต่างของโอเปอเรชันนัลในระหว่างการวิเคราะห์ ในภาษาเชิง วัตถุบางภาษา เช่น Smalltalk โอเปอเรชันเหล่านี้ทั้งหมดจะถูกพัฒนาขึ้นมาแบบเดียวกันแต่ภาษา อื่นๆ เช่น DSM แต่ละโอเปอเรชันจะไม่เหมือนกัน โอเปอเรชันสามารถเพิ่มได้จาก

- ออบเจคโมเดล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งยังมีให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โอเปอเรชันจากโครงสร้างออบเจกต์ การอ่าน, การเขียนค่าแอทริบิวต์ และแอคซิเอชันลิงค์ โอเปอเรชันเหล่านี้ไม่ต้องการถูกแสดงบนออบเจกต์โมเดล แต่เห็นได้จากการแทนค่าของแอทริบิวต์ ในระหว่างการวิเคราะห์สมมุติว่าแอทริบิวต์ทั้งหมดเข้าถึงได้ ใช้สัญลักษณ์ "จุด" แสดงการเข้าถึงของแอทริบิวต์เช่น ATM.cash โมเดลสามารถถูกแสดงเป็นชุดของการเข้าถึง "pseudo-attribute" ได้เช่น account.bank การเข้าถึงควอลิไฟลิงค์(qualified link) ได้โดยใช้สัญลักษณ์ "ดัชนี" เช่น consortium.bank[bank-code].account[account-code] สัญลักษณ์นี้สามารถถูกใช้ในชุดโคโดเพื่อกำหนดฟังก์ชันและแอคชันได้

- อีเวนท์

แต่ละอีเวนท์ที่ส่งถึงออบเจกต์ตรงกับโอเปอเรชันบนออบเจกต์ขึ้นกับสถาปัตยกรรมระบบอีเวนท์สามารถถูกสร้างขึ้นโดยตรงโดยจับอีเวนท์เป็นส่วนหนึ่งของพื้นฐานระบบ, หรือสามารถเปลี่ยนไปเป็นเมทอด (method) ซึ่งมีอยู่ชัดเจนแล้ว ในระหว่างการวิเคราะห์อีเวนท์เขียนกำกับบนการเปลี่ยนสแตท และไม่ต้องเขียนในออบเจกต์โมเดล

- สแตท แอคชัน และแอคทิวิตี

แอคชันและแอคทิวิตีในสแตทไดอะแกรมอาจเป็นฟังก์ชัน ฟังก์ชันเหล่านี้มีโครงสร้างแห่งการคำนวณที่น่าสนใจควรถูกกำหนดเป็นโอเปอเรชันบนออบเจกต์โมเดล

- ฟังก์ชัน

แต่ละฟังก์ชันในดาต้าไฟล์ไดอะแกรมตรงกับโอเปอเรชันบนออบเจกต์ (หรือเป็นไปได้ว่าหลายออบเจกต์)ฟังก์ชันเหล่านี้มีโครงสร้างแห่งการคำนวณที่น่าสนใจและจะสรุปบนออบเจกต์โมเดล จัดฟังก์ชันลงในโอเปอเรชันบนออบเจกต์

ถ้าชุดของสมการหรือส่วนหนึ่งของชุดโคโด อธิบายฟังก์ชันได้มากกว่าหนึ่งฟังก์ชัน ดังนั้นสามารถสร้างโอเปอเรชันใหม่เพื่อทำให้ฟังก์ชันนั้นโมเดลง่ายขึ้น

- ซอปปิงลิสต์โอเปอเรชัน(Shopping List Operation)

บางครั้งพฤติกรรมของคลาสในความเป็นจริงเสนอแนะโอเปอเรชัน เมเยอร์ [Meyer-88] เรียกว่า "ซอปปิงลิสต์" เพราะว่าโอเปอเรชันไม่ได้ขึ้นกับทั้งระบบงานใดระบบงานหนึ่งและเรื่องลำดับใดลำดับหนึ่งของการทำให้สำเร็จ แต่มีความหมายในตัวเอง ซอปปิงลิสต์โอเปอเรชันอนุญาตให้คิดถึงความต้องการที่เป็นไปได้ในอนาคตขณะที่การจัดของคลาสนี้ยังคงเป็นแผนผังอยู่ ซอปปิงลิสต์โอเปอเรชันเตรียมโอกาสให้ฐานของค่านิยามออบเจกต์กว้างออก พ้นจากความต้องการของปัญหาที่คับแคบทันทีทันใด

ตรวจสอบออกแบบเจดโมเดลสำหรับโอเปอเรชันที่คล้ายๆกันและการแปรผันรูปแบบบนโอเปอเรชันเดี่ยว พยายามทำให้คำนิยามของโอเปอเรชันกว้างออกเพื่อครอบคลุมถึงการแปรผันและกรณีพิเศษ ใช้อินแวนชันชันในที่ๆ เป็นไปได้เพื่อลดจำนวนของโอเปอเรชันที่แตกต่างกัน สร้างซูปเปอร์คลาสใหม่เมื่อต้องการทำให้โอเปอเรชันง่ายขึ้น, ซึ่งการสร้างซูปเปอร์ใหม่นี้จะต้องไม่เป็นการบีบบังคับและผิดธรรมดา กำหนดที่ตั้งของแต่ละโอเปอเรชันในระดับที่ถูกต้องภายในแผนผังคลาสผลลัพธ์ของการซัดเกลานี้มักจะน้อยมาก โอเปอเรชันมีประสิทธิภาพมากกว่าและกำหนดได้ง่ายกว่าโอเปอเรชันเริ่มแรก เพราะว่ามันมีรูปแบบเดียวกันหมดและเป็นแบบธรรมดามากกว่า

โมเดลการวิเคราะห์ส่วนมากต้องการการวิเคราะห์มากกว่าหนึ่งครั้งที่จะสมบูรณ์ ประโยคปัญหาส่วนมากเป็นแบบวนรอบ และระบบงานส่วนมากไม่สามารถจะทำแบบเป็นเส้นตรง (linear) ได้ เพราะว่าส่วนต่างๆของปัญหากระทำโต้ตอบกัน เพื่อที่จะเข้าใจปัญหาความสัมพันธ์ของมันทั้งหมดเราต้องทำวิเคราะห์อย่างซ้ำๆ เพื่อเพิ่มความเข้าใจของเรา มันไม่มีเขตที่แน่นอนระหว่างการวิเคราะห์และการออกแบบ ดังนั้นอย่าทำให้มันมากจนเกินไป ตรวจสอบการวิเคราะห์สุดท้ายกับเจ้าของระบบงาน และผู้เชี่ยวชาญระบบงาน

โมเดลการวิเคราะห์ทั้งหมดอาจแสดงความไม่คงเส้นคงวาและความไม่สมดุลภายในโมเดลและข้ามโมเดล ทวนสเตตต่างๆเพื่อไม่ให้มีสิ่งอื่นที่ไม่ต้องการ,ดูความต่อเนื่องกันในการออกแบบ พยายามซัดเกลาคำจำกัดความของออกแบบเพื่อการใช้ร่วมกัน และทำโครงสร้างให้ดีขึ้น เพิ่มรายละเอียดที่เราข้ามไปในตอนที่ทำครั้งแรก

บางโครงสร้างจะดูเหมือนมีสิ่งที่ไม่จำเป็นน่าที่จะปรับให้รัดกุมกว่านี้ ตรวจสอบใหม่ด้วยความระมัดระวังเราอาจจะมีแนวความคิดที่ผิดไป ในบางครั้งมีความต้องการจัดโครงสร้างหลักใหม่อีกครั้งในโมเดลเมื่อความเข้าใจของเราเพิ่มขึ้น มันทำได้ง่ายกว่าตอนที่ยังไม่เคยทำ ดังนั้นอย่าหลีกเลี่ยงที่จะเปลี่ยนที่ดูเหมือนเพียงจะถูกต้องเพียงเพราะว่าเรามีโมเดลเรียบร้อยแล้ว เมื่อมีโครงสร้างหลายโครงสร้างที่คล้ายๆ กันแต่เข้ากันไม่ได้เราอาจจะขาดแนวความคิดแบบธรรมดาหรือผิดกรณี ตัดส่วนประกอบของเงินเนอรัลไลเซชันบนแอทริบิวท์ที่ผิดออก

โดยปกติมักจะลืมออกแบบทางกายภาพ ซึ่งมีลักษณะทางลจิจัลที่แตกต่างกันสองลักษณะ ซึ่งควรจะจำลองกับออกแบบที่แตกต่างกันสำหรับแต่ละลักษณะ การบ่งของสิ่งนี้คือออกแบบที่ไม่เหมาะเพราะมันต้องการใส่สองบทบาท

การบ่งบอกอื่นๆที่จะต้องดูคือข้อยกเว้นที่รวมอยู่, กรณีพิเศษหลายกรณีขาดความสมดุลที่คาดไว้ และออกแบบที่มีชุดของแอทริบิวท์หรือโอเปอเรชันที่ไม่เกี่ยวพันกันสองชุดหรือมาก

กว่า ลองคิดถึงการจัดโครงสร้างโมเดลใหม่อีกครั้ง เพื่อให้เก็บข้อบังคับภายในโครงสร้างได้ดีขึ้น ลบขอบเขตหรือแอสโซซิเอชันที่ดูเหมือนมีประโยชน์ในครั้งแรกแต่ตอนนี้พบว่าเป็นของที่อื่น บ่อยครั้งที่ขอบเขตสองขอบเขตที่แตกต่างกันในการวิเคราะห์สามารถนำมารวมกันได้เพราะว่าความแตกต่างระหว่างขอบเขตนั้นไม่ได้มีอิทธิพลเหนือส่วนที่เหลือของโมเดลในทางที่มีความหมายใดๆ

โมเดลที่ดีจะรู้สึกว่าคุณถูกต้องและไม่ปรากฏว่ามีรายละเอียดของผู้อื่น อย่างกังวลถ้ามันดูเหมือนว่าไม่สมบูรณ์ แม้แต่โมเดลที่ดีและมีขนาดเล็กๆ ซึ่งการออกแบบเพียงพอแล้วก็ยังรู้สึกว่าคุณถูกต้องเสียทีเดียว

เมื่อการวิเคราะห์สมบูรณ์ โมเดลเหมือนเป็นพื้นฐานสำหรับความต้องการและกำหนดขอบเขตของการสนทนาในอนาคต ความต้องการที่หลายๆส่วนมากจะเป็นส่วนหนึ่งของโมเดล ความต้องการบางอย่างกำหนดข้อบังคับของการกระทำ สิ่งเหล่านี้ต้องถูกกำหนดอย่างชัดเจนร่วมกับมาตรฐานที่ดี ความต้องการอื่นๆ อาจจะกำหนดวิธีการแก้ปัญหา

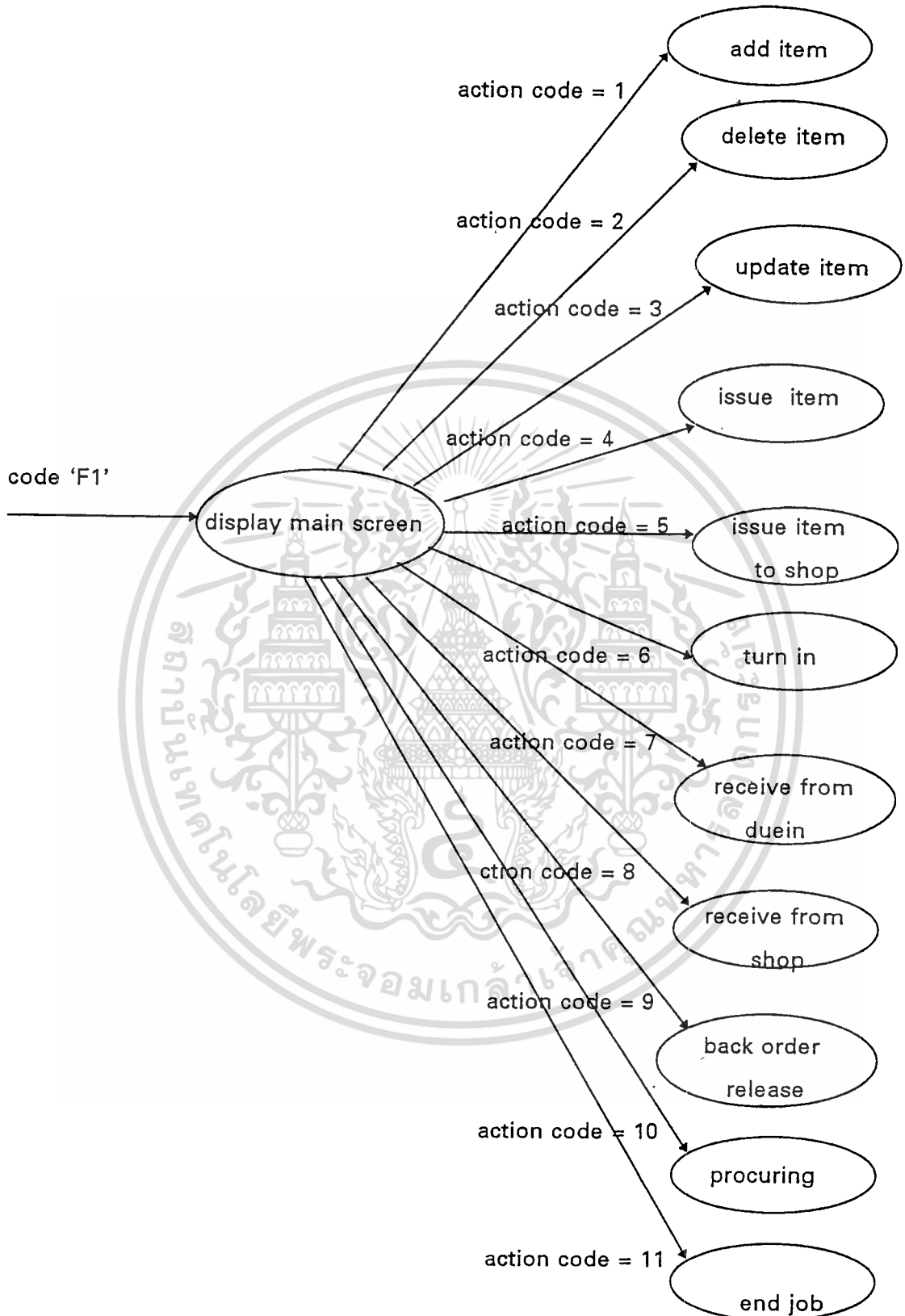
โมเดลขั้นสุดท้ายต้องถูกตรวจสอบกับเจ้าของระบบงาน ในระหว่างการวิเคราะห์ ความต้องการบางอย่างอาจปรากฏว่าไม่ถูกต้องหรือทำไม่ได้ การแก้ไขความต้องการต้องได้รับการยืนยัน โมเดลการวิเคราะห์ต้องถูกตรวจสอบโดยผู้เชี่ยวชาญระบบงานด้วยเพื่อสร้างความมั่นใจว่าโมเดลถูกต้องและเป็นจริง เราพบว่าโมเดลการวิเคราะห์คือวิธีของการติดต่อสื่อสารกับผู้เชี่ยวชาญระบบงานที่ไม่ใช่ผู้เชี่ยวชาญคอมพิวเตอร์ที่ได้ผล

โมเดลการวิเคราะห์ที่ถูกตรวจสอบขั้นสุดท้ายเพื่อเป็นพื้นฐานสำหรับสถานปัตยกรรมระบบ การออกแบบ และการสร้างระบบ ประโยคปัญหาเริ่มแรกต้องถูกทวนเพื่อรวมความถูกต้องเข้าด้วยกันกับความเข้าใจที่พบระหว่างการวิเคราะห์

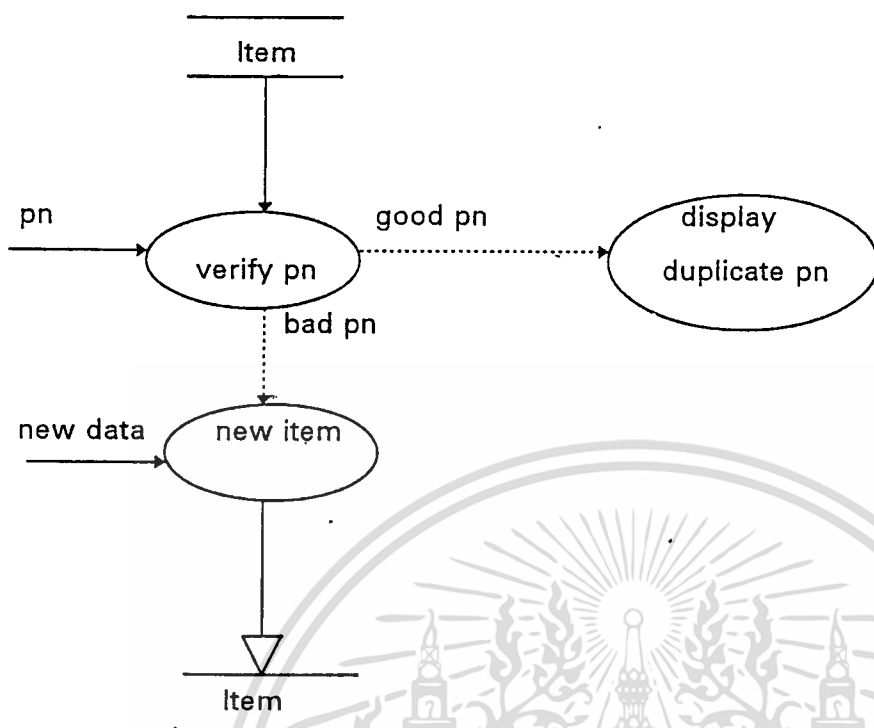
เป้าหมายของการวิเคราะห์คือกำหนดปัญหาและขอบเขตระบบงานอย่างครบถ้วนโดยไม่มีกัการพยายามสร้างให้สอดคล้องกับการสร้างระบบแต่อย่างใด แต่ในการปฏิบัติมันอาจจะเป็นไปได้ที่จะหลีกเลี่ยงไม่ให้สิ่งที่เกี่ยวข้องกับการสร้างระบบเลย ไม่มีเส้นแบ่งที่แน่นอนในแต่ละตอนที่ออกแบบว่าขั้นตอนนี้คือการวิเคราะห์ พันเขตแดนนี้ไปเป็นออกแบบระบบ เวลาปฏิบัติจริงจะมีการคาบเกี่ยวกัน ไม่ได้เป็นไปตามกฎที่กล่าวมาข้างต้นอย่างเข้มงวดแต่สามารถยืดหยุ่นได้ เป้าหมายของการจำลองแบบคือทำงานทั้งหมดให้สำเร็จ

รายละเอียดฟังก์ชันนัลโมเดลของระบบงานบริหารพัสดุสายช่างอากาศแสดงไว้ในภาพที่ 36

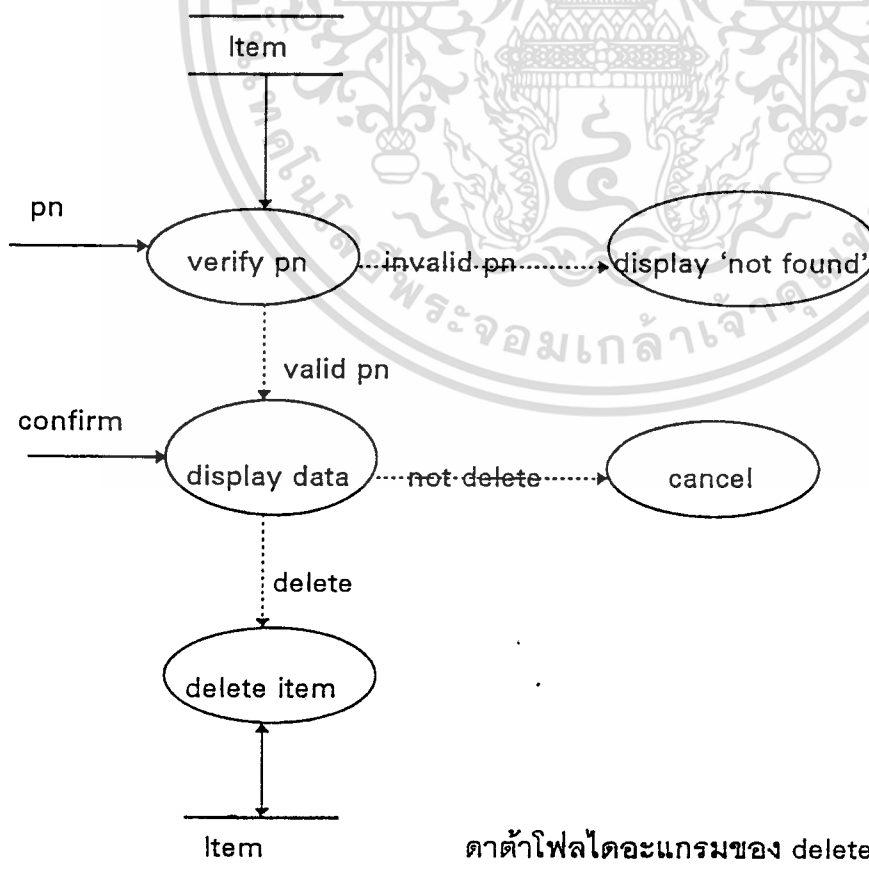
ภาพที่ 36 ฟังก์ชันนำโมเดล



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

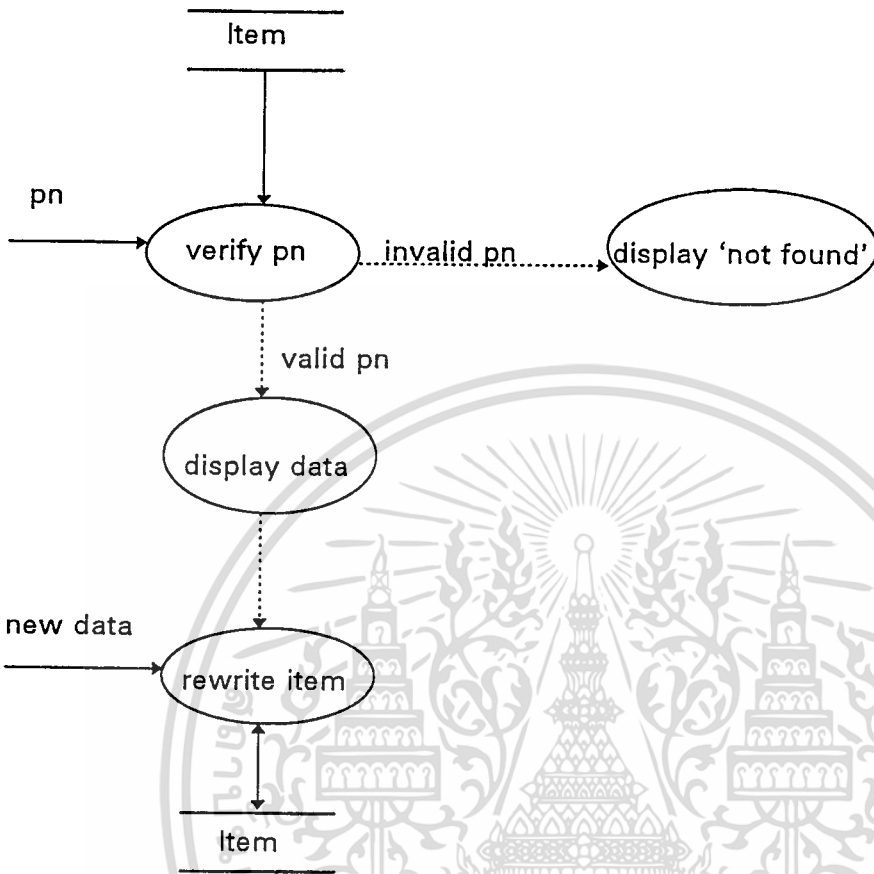


ดาต้าไฟล์ไดอะแกรมของ add item process

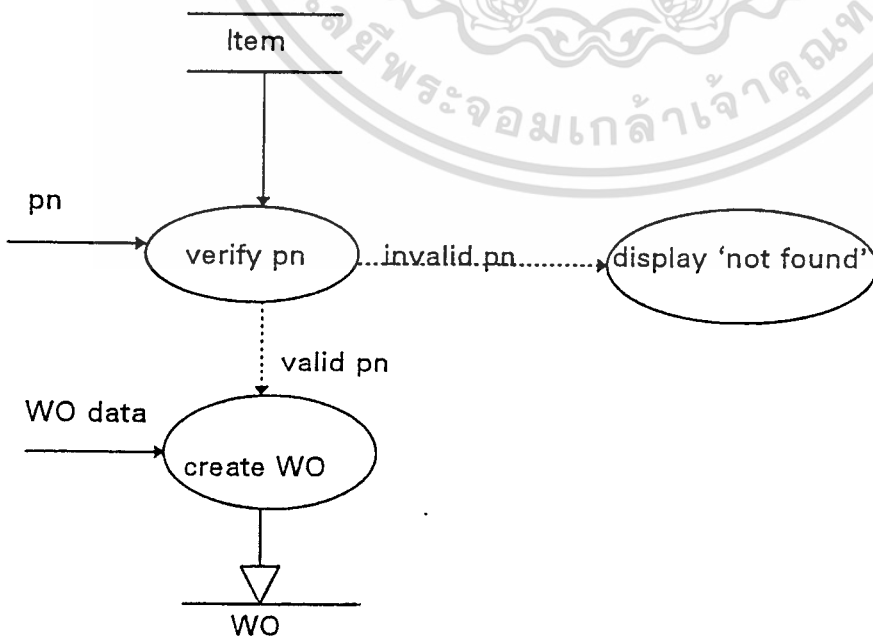


ดาต้าไฟล์ไดอะแกรมของ delete item process

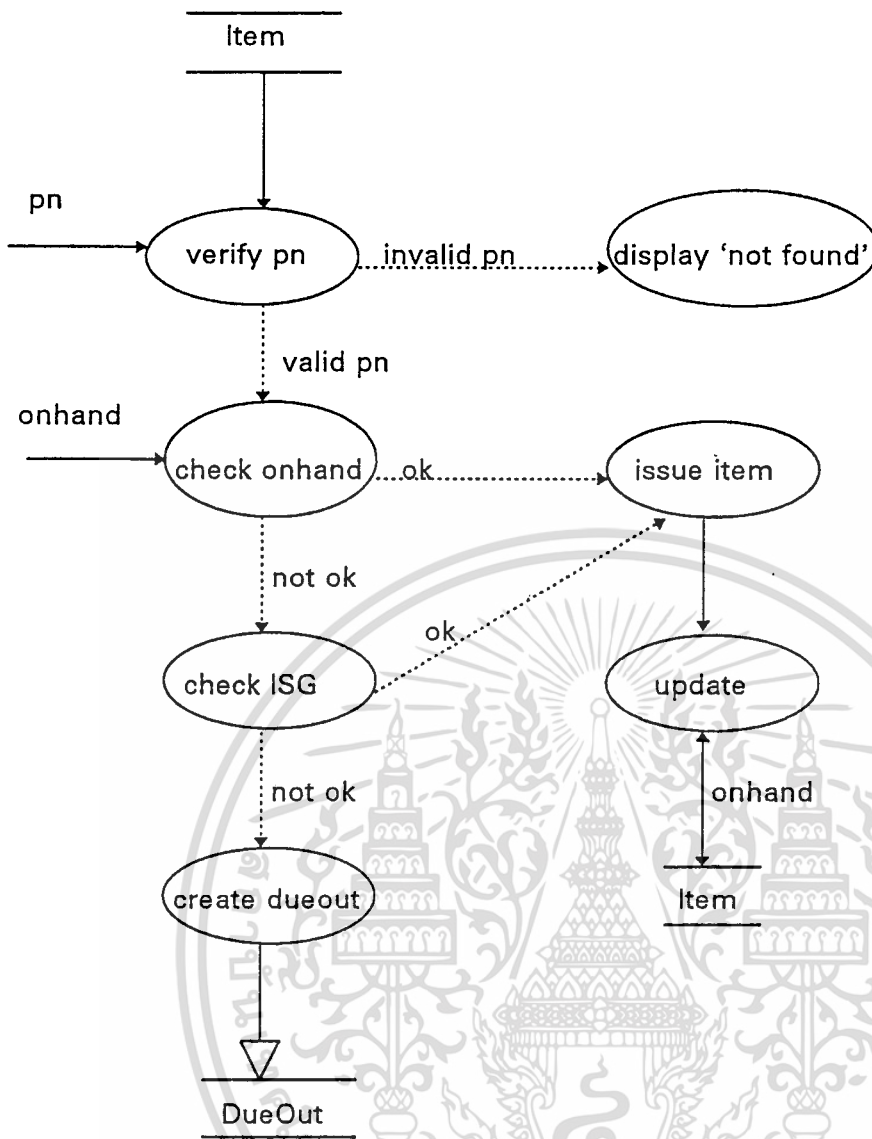
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ดาต้าไฟล์ไดอะแกรมของ update item process

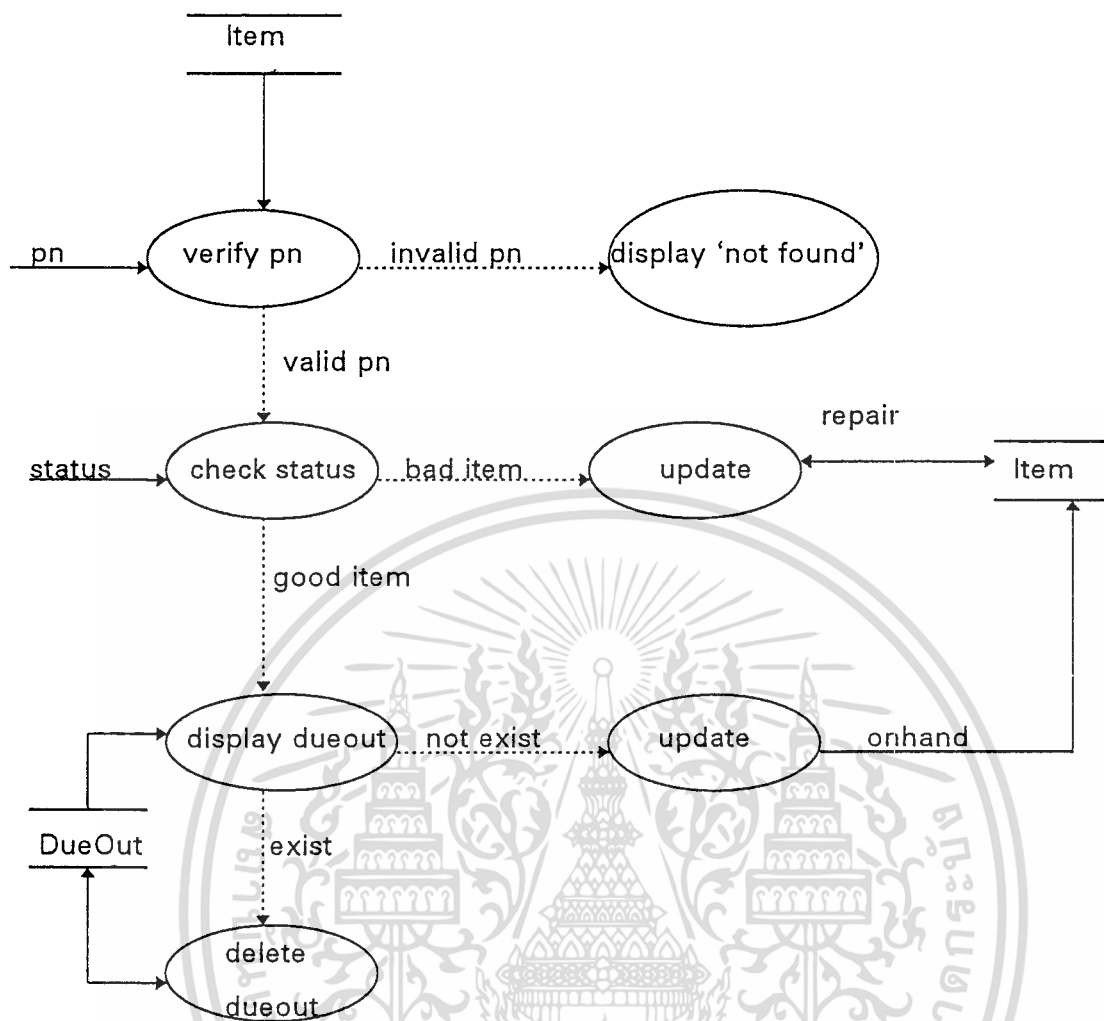


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้ **ดาต้าไฟล์ไดอะแกรมของ issue to shop process** โยชน์ด้านการค้า
 ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



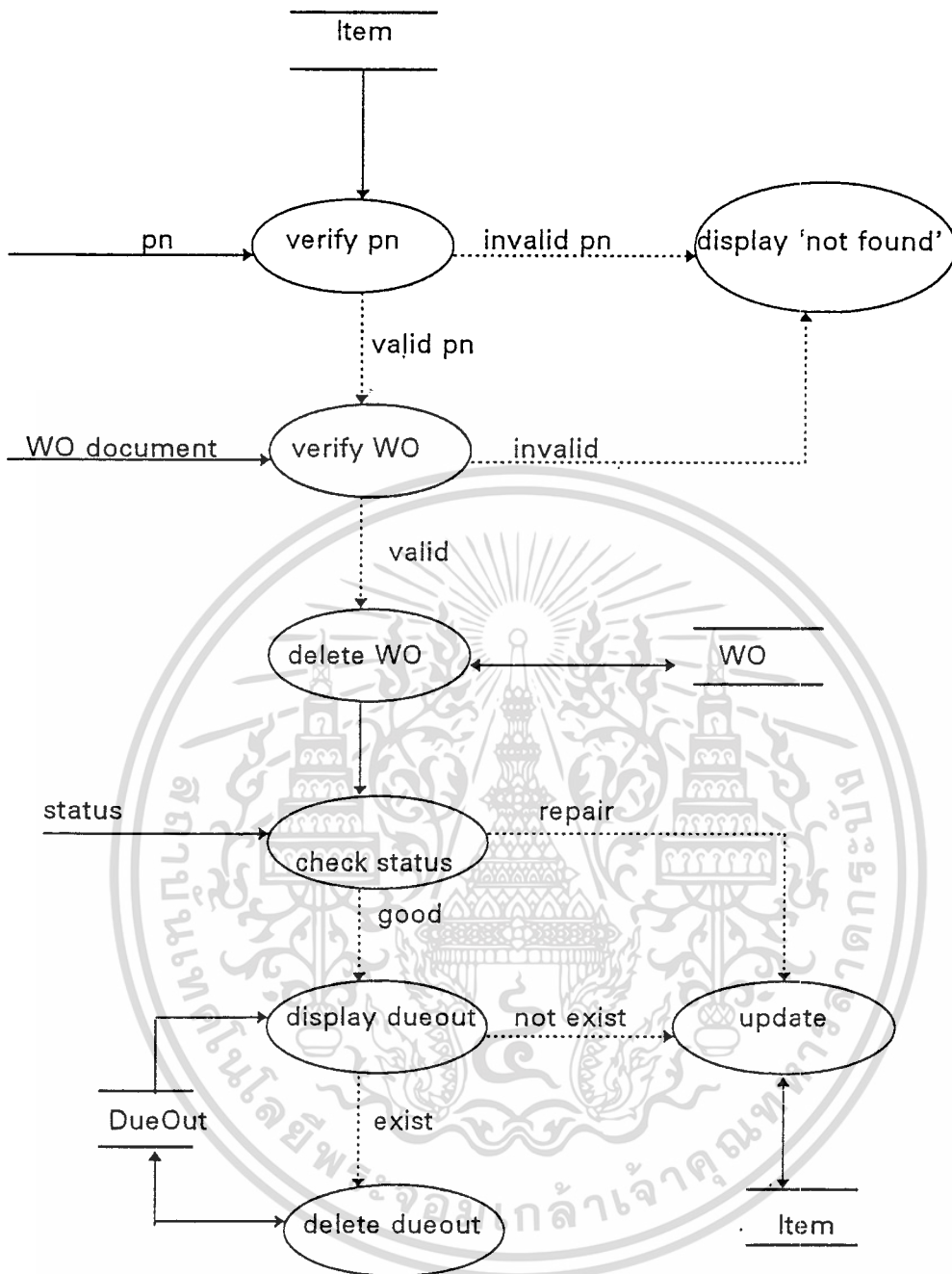
ดาต้าไฟล์ไดอะแกรมของ issue item process

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



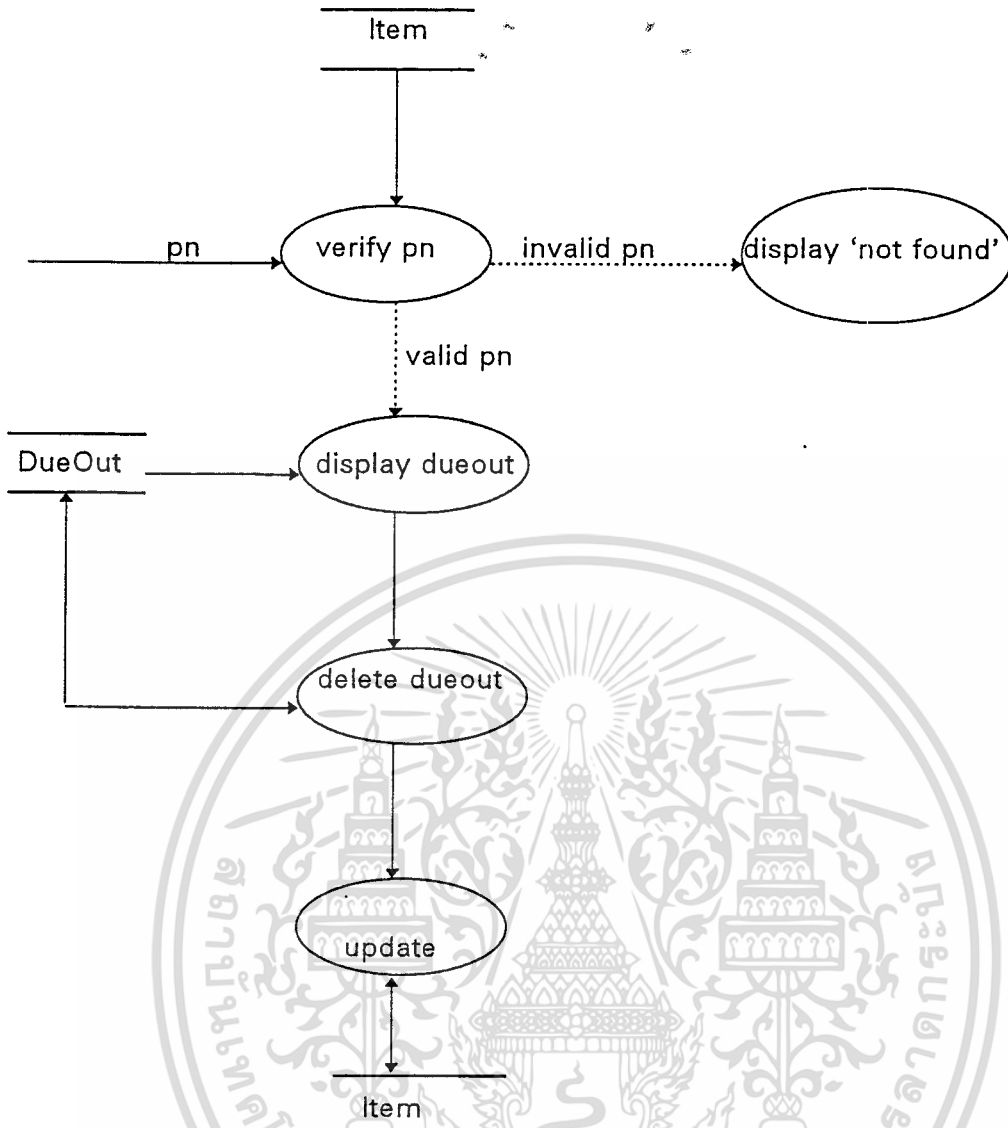
ดาต้าไฟล์ไดอะแกรมของ turn in process

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

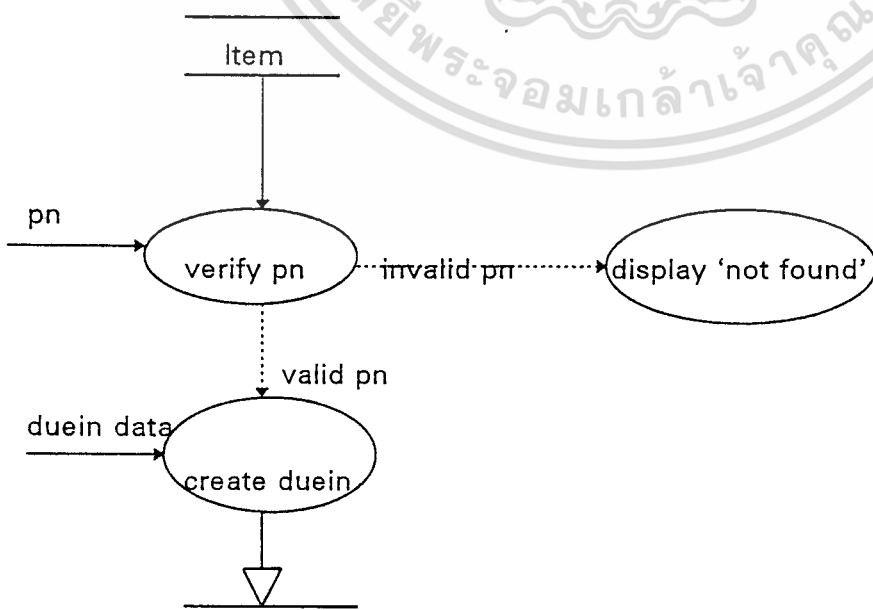


ดาต้าไฟล์ไดอะแกรมของ receive from shop process

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

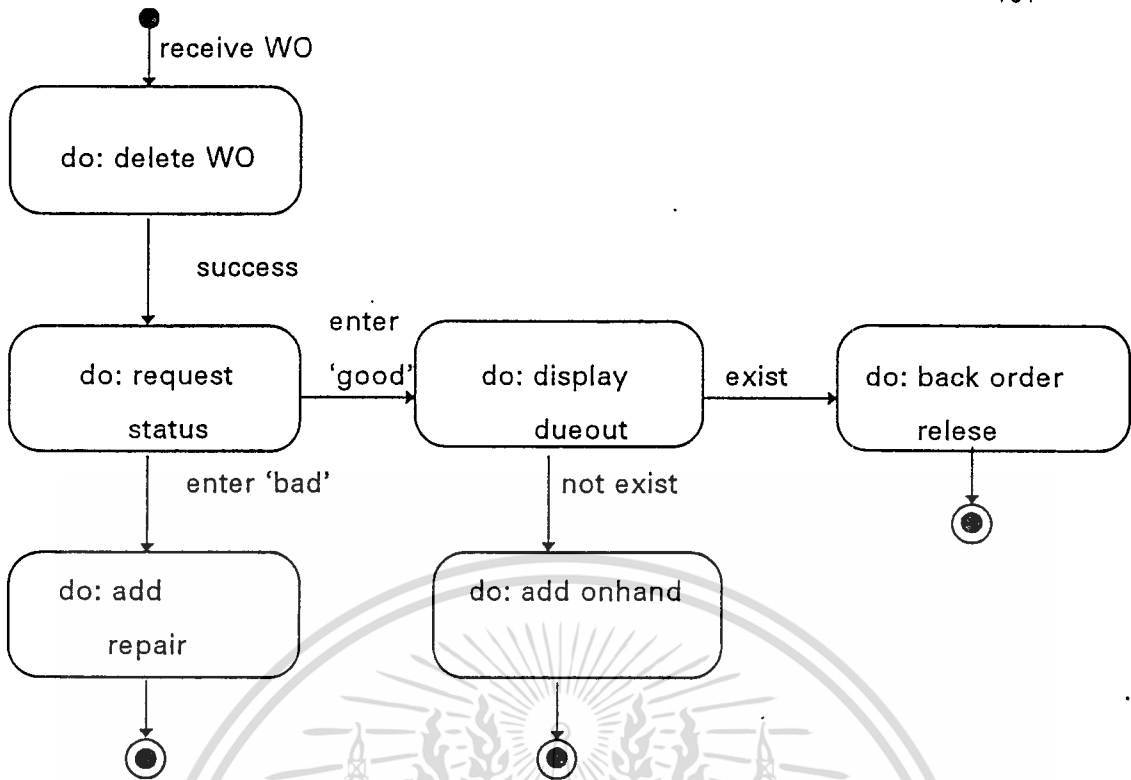


ดาต้าไฟล์ไดอะแกรมของ BOR



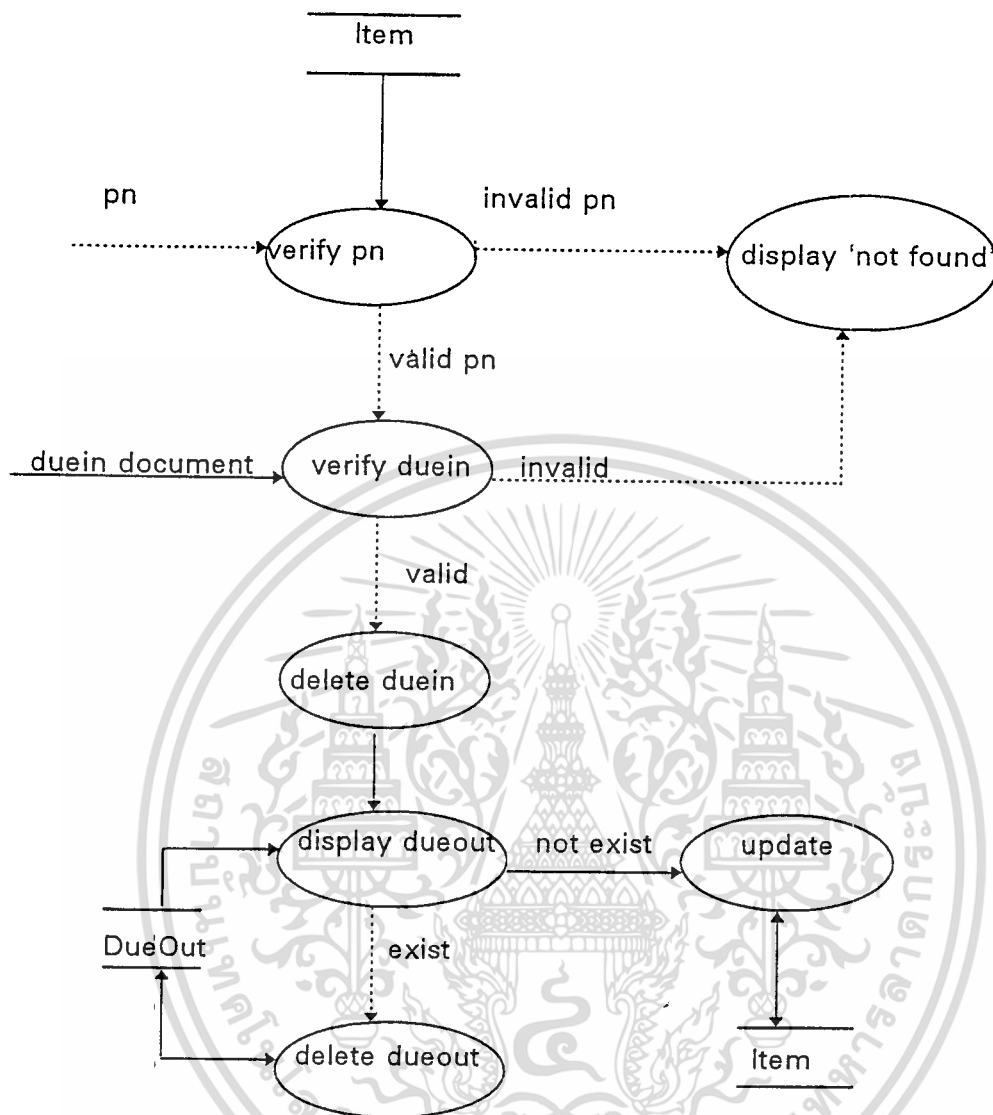
ดาต้าไฟล์ไดอะแกรมของ procuring process

เอกสารนี้เป็นเอกสารของบริษัทสำหรับการใช้งานเพื่อการศึกษารายงาน เมื่อผู้จัดทำเห็นว่าประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



สเตตโอดีอะแกรมของคลาส Item

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ดาต้าไฟล์ไดอะแกรมของ receive from duein process

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

การบำรุงรักษาระบบ

5.1 การบำรุงรักษา

สิ่งที่จะต้องทำในเรื่องของการบำรุงรักษาคือ

- 1) จะเก็บรักษาไฟล์ของระบบอย่างไรถ้าขนาดของไฟล์ใหญ่ขึ้น
- 2) ขั้นตอนปกติที่เราสามารถป้องกันไม่ให้งานสูญหายไป ถ้าหากว่าระบบเกิดมี
ปัญหาทำงานไม่ได้
- 3) จะทำอย่างไรถ้าระบบเกิดปัญหาทำงานไม่ได้

5.2 การบันทึกการเปลี่ยนแปลงแบบอัตโนมัติ

สมอลทอคจะบันทึกการเปลี่ยนแปลงไว้ในไฟล์เซนจ์ล็อก (Change Log) ส่วนโคดที่ไม่มี
การเปลี่ยนแปลงจะเก็บไว้ในไฟล์ซอส (Source) และเก็บพจนีย์เตอร์ของไฟล์เซนจ์ล็อก ไฟล์ซอสไว้
ในไฟล์อิมเมจ (Image) ดังนั้นเราต้องเก็บรักษาไฟล์ทั้งสามไว้ด้วยกัน จากเหตุผลดังกล่าวเราจึงไม่
ควรที่จะปรับปรุงแก้ไขไฟล์เซนจ์ล็อก เราสามารถเรียกมาดูได้และสามารถที่จะนำเมธอดและการ
กำหนดคลาสกลับเข้ามาในระบบได้อีกในกรณีที่มีการกู้ข้อมูล (Recovery) อันเนื่องมาจากระบบ
เกิดปัญหาทำงานไม่ได้ เหตุการณ์ที่สำคัญอื่นๆ ที่ทำให้สมอลทอคมีการจดบันทึกแบบอัตโนมัติ
เช่น ทุกครั้งที่สั่งสำเนาไฟล์อิมเมจ (ใช้คำสั่ง Save Image) จะบันทึกข้อความพร้อมวันเวลาลงใน
ไฟล์เซนจ์ล็อก ทุกครั้งที่สั่งให้มีการทำงาน (ใช้คำสั่ง do it หรือ show it) จะมีการจดบันทึกรวมทั้ง
เมื่อมีการสั่งให้ลบเมธอดออกจากคลาสก็จะมีกรจดบันทึกเช่นกัน

5.3 ความสำคัญของการเก็บอิมเมจ

ไฟล์อิมเมจคือกลุ่มของออบเจกต์ทุกออบเจกต์ในสมอลทอคซึ่งถูกสร้างขึ้นมา เมื่อเราสั่ง
สำเนาข้อมูลในไฟล์อิมเมจคำอธิบายของออบเจกต์ทั้งหมดจะบันทึกลงในไฟล์อิมเมจ เมื่อเราเริ่ม
ต้นระบบงานใหม่ในภายหลังทุกๆ ออบเจกต์ในไฟล์อิมเมจจะถูกสร้างขึ้นมาอีกครั้งหนึ่งเฉพาะตรง
จุดที่ไฟล์อิมเมจถูกสั่งให้ทำสำเนาไว้ก่อนที่จะออกจากระบบงานครั้งล่าสุด ดังนั้นระบบจะเริ่มต้น
ใหม่เหมือนกับที่เราได้เคยใช้งานระบบก่อนหน้านั้น เมื่อเราเปลี่ยนแปลงโคด (Code) หรือสร้าง
ออบเจกต์ใหม่และเปลี่ยนแปลงรายละเอียดของออบเจกต์เดิมเราจะต้องสั่งทำสำเนาไฟล์อิมเมจเพื่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้การเปลี่ยนแปลงนั้นถาวร เราสามารถส่งสำเนาไฟล์อิมเมจด้วยการเลือกใช้ฟังก์ชันเซฟอิมเมจในเมนูระบบ หรือเลือกฟังก์ชันเซฟอิมเมจเมื่อตอนที่จะออกจากโปรแกรมสมอลทอค ถ้าเราไม่ได้สั่งให้ทำสำเนาไฟล์อิมเมจไว้เราจะไม่ได้การเปลี่ยนแปลงต่างๆ เหล่านั้นเมื่อเราเริ่มต้นระบบใหม่ในคราวหน้า

5.4 การลดขนาดไฟล์เซจล็อก (Compressing Change Log)

การลดขนาดของไฟล์เซจล็อกนั้นจะมีการสร้างสำเนาของเมธอดล่าสุดของแต่ละเมธอดที่เป็นเมธอดใหม่หรือมีการเปลี่ยนแปลง ไฟล์อิมเมจใหม่จะถูกเขียนขึ้นโดยอัตโนมัติเราควรลดขนาดของไฟล์เซจล็อกเมื่อมันเริ่มจะใหญ่ขึ้น ซึ่งในขณะที่จะทำการลดขนาดของไฟล์เซจล็อกนี้เนื้อที่ในดิสก์จะต้องมีพอสำหรับไฟล์เซจล็อกเก่าและใหม่ในเวลาเดียวกันด้วย ดังนั้นเราต้องมั่นใจว่าเราได้ลดขนาดของไฟล์เซจล็อกก่อนที่ดิสก์จะเต็ม คำสั่งที่ใช้คือ `Smalltalk compressChanges`

5.5 การลดขนาดของไฟล์ซอส (Compressing source)

การลดขนาดของไฟล์ซอสจะมีการสร้างซอสไฟล์ใหม่สำหรับทุกๆ เมธอดในระบบในขณะนั้น โดยที่จะคัดลอกสำเนาล่าสุดของซอสโคดสำหรับแต่ละเมธอดจากไฟล์ซอสเก่าหรือจากไฟล์เซจล็อก หลังจากการทำให้ขนาดของไฟล์ซอสลดลงแล้วจะได้ไฟล์อิมเมจและไฟล์เซจล็อกใหม่เกิดขึ้นโดยอัตโนมัติด้วย ไฟล์เซจล็อกใหม่ที่ได้จะเป็นไฟล์ที่ว่างเปล่า ไฟล์ซอสจะถูกเขียนในรูปแบบบีบเพื่อประหยัดเนื้อที่ในดิสก์ ดังนั้นเราไม่สามารถแก้ไขไฟล์ซอสได้นอกจากโปรแกรมสมอลทอคเอง แต่เราสั่งทำสำเนาไฟล์ซอสได้ หลังจากที่เราบีบซอสเราจะได้ไฟล์อิมเมจใหม่ ไฟล์ซอสใหม่ และไฟล์เซจล็อกใหม่ แต่ไฟล์เซจล็อกจะว่างเปล่า ดังนั้นเราจึงควรที่จะทำสำเนาของไฟล์ทั้งสามไว้ก่อนที่จะทำการลดขนาดของไฟล์ซอสเผื่อว่าอาจจะจะมีข้อผิดพลาดอะไรเกิดขึ้นจะได้สามารถกู้ข้อมูลคืนได้ คำสั่งที่ใช้คือ `Smalltalk compressSources`

5.6 การกู้ระบบ (Surviving a System Crash)

สมอลทอคมีความยืดหยุ่นกับข้อผิดพลาดมาก ซึ่งออกจะโชคไม่ดีนักที่ความเสียหายสามารถที่จะเกิดขึ้นได้โดยเฉพาะถ้าหากว่าเรามีการปรับปรุงแก้ไขในสมอลทอค แต่ว่าคุณสมบัติการบันทึกสำเนาแบบอัตโนมัติของสมอลทอคทำให้การกู้ข้อมูลสามารถทำได้ ความเสียหายไม่

ได้เกิดขึ้นเพราะว่าในโปรแกรมสมอลทอคมีบั๊ก (bug) แต่เพราะว่ามีการเปลี่ยนแปลงสภาวะภายในของโปรแกรมสมอลทอคอันเนื่องมาจากโปรแกรมสมอลทอคยอมให้เราเปลี่ยนแปลงอะไรก็ได้ ทุกอย่างแม้แต่การทำลายสภาวะของระบบ แต่สิ่งเหล่านี้ก็ไม่ได้เป็นปัญหาอะไร เพราะว่าเราสามารถกู้ข้อมูลกลับคืนมาได้เกือบทั้งหมด ในความเป็นจริงการทดลองทำกับระบบคือวิธีการเรียนรู้ที่ดีที่สุด แต่ว่าเราควรจะมีใจว่าเราได้ทำสิ่งต่อไปนี้เป็น การป้องกันไว้ก่อนล่วงหน้าแล้วคือ

1) ต้องทำสำเนาของซอสไฟล์ไว้เสมอ

2) ต้องทำสำเนาที่เชื่อถือได้ของไฟล์อิมเมจและเซ่นจล๊อค จะต้องจำไว้ว่าไฟล์อิมเมจและไฟล์เซ่นจล๊อคใช้ในการทำงานร่วมกัน ซึ่งถ้าหากว่าเราใช้ไฟล์อิมเมจที่เป็นเวอร์ชันเก่ากว่าเวอร์ชันของไฟล์เซ่นจล๊อคเราก็จะไม่สามารถใช้ซอสโคดสำหรับบางเมธอดได้ เพราะว่าหลังจากที่เราลดขนาดของไฟล์เซ่นจล๊อคแล้วจะทำให้ไฟล์อิมเมจและไฟล์เซ่นจล๊อคจะถูกเปลี่ยนไปด้วย เราจึงไม่สามารถใช้ไฟล์อิมเมจเก่ากับไฟล์เซ่นจล๊อคใหม่ที่ถูกลดขนาดลงแล้วได้แต่เราจะใช้ไฟล์อิมเมจเก่ากับไฟล์เซ่นจล๊อคเวอร์ชันล่าสุดได้ถ้าหากว่าไฟล์เซ่นจล๊อคนั้นยังไม่ได้ถูกลดขนาดลงในช่วงระยะเวลาที่ไฟล์อิมเมจกำลังใช้ไฟล์เซ่นจล๊อค สรุปก็คือจะต้องทำสำเนาไฟล์อิมเมจและไฟล์เซ่นจล๊อคร่วมกันแล้วเราจะไม่มีปัญหา

3) เราควรจะทำสำเนาไฟล์อิมเมจเก็บไว้ในดิสก์ก่อนที่เราจะพยายามทำบางสิ่งบางอย่างที่อาจจะทำให้ระบบเกิดปัญหาจนไม่สามารถทำงานได้ ซึ่งเราสามารถที่จะใช้คำสั่งทำสำเนาไฟล์อิมเมจได้จากเมนูระบบ การทำเช่นนี้จะช่วยให้เราสามารถรักษางานทั้งหมดของเราไว้ได้ถ้าหากว่าระบบเกิดปัญหาจนทำงานไม่ได้ขึ้นมาเราก็จะสามารถเริ่มต้นใหม่อีกครั้งได้โดยการใช้สำเนาไฟล์อิมเมจนี้ แต่ถ้าหากว่าเราทำสำเนาไฟล์อิมเมจยังไม่สำเร็จและระบบเกิดมีปัญหามาทำงานขึ้นมาเราก็ยังสามารถที่จะใช้ไฟล์เซ่นจล๊อคช่วยได้

4) ถ้าเราทดลองกับโปรแกรมสมอลทอคและเราเกิดไปทำระบบเสียหรือว่าเราไม่ต้องการให้การเปลี่ยนแปลงนั้นถาวรเราก็ไม่ต้องทำไฟล์สำเนาอิมเมจ โดยการที่เราออกจากโปรแกรมสมอลทอคด้วยการใช้ฟังก์ชันฟอร์เกทอิมเมจ (forget image) แต่ว่าการเปลี่ยนแปลงทั้งหมดเหล่านั้นถูกบันทึกไว้ในไฟล์เซ่นจล๊อคแล้วและเราก็สามารถที่จะเรียกกลับเข้ามาในระบบอีกครั้งได้

5) ถ้าระบบเกิดมีปัญหามาทำงาน อย่าตกใจ ให้ทำสำเนาของไฟล์อิมเมจและไฟล์เซ่นจล๊อค โดยปกติเราสามารถที่จะกู้ข้อมูลกลับคืนมาได้ เพราะถ้าเราทำสำเนาเอาไว้เราก็สามารถที่จะค่อยๆ อ่านไปที่ละส่วนเพื่อหาข่าวสารที่ต้องการจะกู้ข้อมูลคืนมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.7 การกู้ข้อมูล (Recovering From a System Crash)

ถ้าระบบไม่ทำงานเราจะต้องทำสำเนาของไฟล์อิมเมจและไฟล์เซจลือกไว้ก่อนแล้ว การกู้ข้อมูลมีลำดับขั้นตอนดังนี้

1) ตรวจสอบถ้าหากว่าเรามีไฟล์อิมเมจที่ดีแล้ว พยายามที่จะเริ่มต้นโปรแกรมสมอลทอคแบบปกติโดยใช้ไฟล์อิมเมจนี้ แต่ถ้ามันเริ่มต้นไม่ได้เราควรจะหาสำเนาของไฟล์อิมเมจและไฟล์เซจลือกที่ใหม่ที่สุดแล้วเริ่มต้นโปรแกรมสมอลทอคด้วยเวอร์ชันนั้น

2) ในขณะที่โปรแกรมสมอลทอคกำลังทำงาน ให้มองหาในไฟล์เซจลือกที่กำลังใช้ในขณะนั้นเมื่อระบบเกิดไม่ทำงาน ในไฟล์เซจลือกจะมีการเปลี่ยนแปลงทั้งหมดที่เราทำและการทำงานทั้งหมดที่เราสั่ง

3) มองหาจุดที่เราทำสำเนาไฟล์อิมเมจล่าสุดในไฟล์เซจลือกเพราะว่าทุกครั้งที่เราทำสำเนาไฟล์อิมเมจโปรแกรมสมอลทอคจะเขียนข้อความบอกวันเวลาที่ทำในไฟล์เซจลือก ไล่หาในไฟล์เซจลือกจากข้างท้ายขึ้นมาถ้าต้องการสำเนาอิมเมจล่าสุด

4) เมื่อได้จุดที่ทำสำเนาอิมเมจซึ่งงานของเราได้หายไปทีจุดนี้ไปจนจบไฟล์เซจลือก ใช้ฟังก์ชันอินสทอล (install) จากเมนูเพน (pane menu) เราต้องระวังที่จะไม่นำเอาส่วนที่มีข้อผิดพลาดกลับเข้ามาในระบบอีก

บทที่ 6

สรุปและข้อเสนอแนะ

6.1 สรุปผลการวิจัย

ผลการทำวิจัยสรุปได้ดังนี้

1. การศึกษาทฤษฎี

การออกแบบระบบโดยใช้หลักการเชิงวัตถุแตกต่างไปจากการออกแบบระบบงานแบบเดิม ซึ่งใช้วิธีการเก็บรวบรวมข้อมูลต่างๆ เป็นแฟ้มข้อมูล(File) หรือฐานข้อมูล(Database) แยกกันกับโปรแกรมระบบงาน ในขณะที่หลักการเชิงวัตถุจะกำหนดสิ่งต่างๆ ที่อยู่ภายในขอบเขตของระบบงานเป็นออบเจกต์ และในออบเจกต์จะประกอบด้วยข้อมูลและเมธอด รวมกันอยู่ภายในออบเจกต์ การเรียกใช้เมธอดหรือการสั่งให้เกิดการกระทำตามที่กำหนดไว้ในเมธอดคือการส่งแอสเซสไปยังออบเจกต์นั้นๆ ดังนั้นหากมีการเปลี่ยนแปลงวิธีการกระทำใดๆ ของออบเจกต์จะไม่มีผลกระทบต่อข้อความที่ส่งไปยังออบเจกต์นั้นและจะไม่มีผลกระทบต่อออบเจกต์อื่นๆ ด้วย นั่นคือโปรแกรมของระบบไม่มีการเปลี่ยนแปลง ซึ่งนับว่าเป็นผลดีอย่างมากกับระบบงานเพราะโดยปกติแล้วพบว่าระบบงานมักจะมีการเปลี่ยนแปลงวิธีการปฏิบัติอยู่เสมอๆ เมธอดสามารถใช้อธิบายถึงพฤติกรรมของออบเจกต์แต่ละออบเจกต์ได้ ออบเจกต์ที่มีพฤติกรรมแบบเดียวกันสามารถจัดรวมกลุ่มเป็นกลุ่มเดียวกันได้เรียกว่าคลาส ในระหว่างคลาสสามารถที่จะถ่ายทอดคุณสมบัติ จากคลาส หนึ่งไปยังอีกคลาสหนึ่งได้ โดยเรียกคลาสที่ถ่ายทอดคุณสมบัติว่าซูเปอร์คลาส และเรียกคลาสที่ได้รับการถ่ายทอดคุณสมบัติว่าซับคลาส ซึ่งทำให้การพัฒนาการระบบงานทำได้ง่าย ในการทำวิจัยนี้เลือกใช้โมเดลของออบเจกต์โมเดลลิ่งเทคนิค(Object Modeling Technique) หรือโอเอ็มที(OMT) และเลือกใช้โปรแกรมภาษาสโมลทอคเพราะเป็นต้นแบบของภาษาเชิงวัตถุโดยแท้จริง

2. การวิเคราะห์เชิงวัตถุ(Object-Oriented Analysis)

ระบบงานบริหารพัสดุสายช่างอากาศ กำหนดคลาส Item เป็นคลาสของพัสดุทั้งหมด คลาส Item ประกอบด้วยซับคลาส 2 ซับคลาส คือคลาส Repairable และคลาส Consumable โดยคลาส Repairable คือคลาสของพัสดุประเภทพัสดุดูซ่อมหมุนเวียนและคลาส Consumable คือคลาสของพัสดุประเภทพัสดุดิ้นเปลือง หลังจากนั้นสร้างแบบจำลองเพื่อใช้อธิบายระบบงานทั้งระบบงาน ซึ่งมีอยู่ด้วยกัน 3 แบบจำลองคือหนึ่งออบเจกต์โมเดล (Object Model) ใช้อธิบายรายละเอียดเกี่ยวกับออบเจกต์นั้นว่ามีข้อมูลและเมธอดอะไรบ้าง และแสดงความสัมพันธ์ระหว่างออบเจกต์แต่ละเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ออบเจกต์ภายในระบบงาน แบบจำลองที่สองคือไดนามิกโมเดล (Dynamic Model) ซึ่งใช้สเตทไดอะแกรม (State Diagram) อธิบายพฤติกรรมของออบเจกต์ที่กระทำโต้ตอบกับอีเวนต์ต่างๆ ไดนามิกโมเดลนี้ไม่ค่อยมีความจำเป็นมากนักสำหรับระบบงานบริหารพัสดุสายช่างอากาศ เพราะว่าเป็นระบบงานที่เกี่ยวกับการเก็บข้อมูลเสียเป็นส่วนใหญ่ และแบบจำลองสุดท้ายคือฟังก์ชันนัลโมเดล (Functional Model) ซึ่งใช้ไดอะแกรมไหล (Data Flow Diagram) อธิบายถึงวิธีการคำนวณค่าของเมธอดภายในออบเจกต์แต่ละออบเจกต์

3. การพัฒนาระบบงานด้วยโปรแกรมสมอลทอค

จากผลของการทำวิจัยเริ่มตั้งแต่การออกแบบระบบงานไปจนกระทั่งถึงการพัฒนาระบบงานด้วยโปรแกรมสมอลทอค พบว่าสามารถที่จะวิเคราะห์และออกแบบระบบงานได้ดีเช่นเดียวกับวิธีการแบบเดิมและจะดีกว่า, ออกแบบได้ง่ายกว่าสำหรับระบบงานบางประเภทเสียด้วยซ้ำเช่นระบบงานที่มีข้อมูลซับซ้อนมากๆ หรือระบบงานที่มีการเปลี่ยนแปลงวิธีการทำ (procedure) บ่อยๆ ระบบงานที่ได้จากการวิจัยสามารถทำงานได้ดีเช่นเดียวกับวิธีการแบบเดิม ในด้านของผู้ใช้ก็ไม่รู้สึกว่ามีอะไรเปลี่ยนแปลงเปรียบเสมือนการสังคอกไม้ ผู้ขายจะใช้วิธีการใดลูกค้าไม่รู้ รู้แต่ว่าได้รับดอกไม้ตามที่ต้องการ ตรงกันข้ามกับวิธีการแบบเดิมที่ส่วนมากแล้วเมื่อมีการแก้ไขผู้ใช้อาจมีการอบรมใหม่ ในระยะแรกๆ ของการทำอาจจะรู้สึกติดขัดบ้างเพราะว่ายังไม่คุ้นกับแนวความคิดแบบใหม่ แต่เมื่อเริ่มคุ้นเคยจะพบว่าสามารถทำได้ง่ายและสะดวกกว่าวิธีการแบบเดิม การสร้างความคุ้นเคยกับแนวความคิดแบบใหม่นี้ก็ใช้เวลาไม่มากนักเพราะหลักการเข้ากับชีวิตประจำวันของเราที่มักจะมองสิ่งต่างๆ รอบตัวเราเป็นออบเจกต์ๆ หนึ่ง

6.2 ข้อเสนอแนะ

งานวิจัยนี้มุ่งที่จะทดลองใช้หลักการเชิงวัตถุมาออกแบบระบบงานแทนวิธีการแบบเดิมๆ เพื่อจะศึกษาหาข้อสรุปผลการทำงาน ดังนั้นจึงเลือกใช้โปรแกรมสมอลทอค ซึ่งเป็นโปรแกรมเชิงวัตถุโดยแท้ แต่ว่าในปัจจุบันมีซอฟต์แวร์เชิงวัตถุใหม่ๆ เกิดขึ้นมากมาย ซึ่งใช้งานได้ง่ายและสะดวกกว่า จึงขอเสนอแนะให้ผู้สนใจได้ทดลองใช้โปรแกรมเชิงวัตถุใหม่ๆ เหล่านั้น



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ปีที่ 20 ฉบับที่ 105 / มกราคม - กุมภาพันธ์ 2537

คอมพิวเตอร์

JOURNAL OF COMPUTER ASSOCIATION OF THAILAND UNDER THE ROYAL PATRONAGE OF HIS MAJESTY THE KING

กรณีศึกษาของสื่อสารระยะไกล กับเครื่องแปลภาษา

คอมพิวเตอร์กับจิตสำนึก

กรณีออกแบบฐานข้อมูลข้าราชการและพลเรือน

- ระบบเครื่องราชฯ ของข้าราชการมหาลัย

- กรณีอุทกกรรม

การใช้โปรแกรมกับงานบริหารด้านพยากรณ์อากาศ

นำทฤษฎี Fuzzy มาใช้ปรับปรุงทฤษฎี

แบบแผนการจำลองสี่และแปด

การใช้โปรแกรมเชิงวัตถุกับ ระบบงานบริหารพัสดุสายช่างอากาศ

บทความนี้ได้นำเสนอแนวคิดของการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming) ซึ่งกำหนดข้อมูลประเภทนามธรรม (Abstraction Data Type หรือ ADT) ออบเจกต์ (Object) สร้างมาจากเอ็ดที (ADT) ออบเจกต์ประกอบด้วยตัวแปรที่มีฟังก์ชันที่จำเป็นที่กระทำกับตัวแปรเหล่านั้นรวมอยู่ด้วย ตัวแปรใช้แทนข้อมูลข่าวสารภายใน ฟังก์ชันใช้กำหนดการกระทำที่กระทำกับออบเจกต์นั้นออบเจกต์ใหม่สามารถถ่ายทอด (Inheritance) คุณสมบัติจากออบเจกต์ที่มีอยู่เดิมได้ คำสั่งเดียวกันให้ผลการทำงานต่างกันให้ออบเจกต์ที่ต่างกัน (Polymorphism) การเปลี่ยนแปลงข้อมูลภายในออบเจกต์ไม่ทำให้โปรแกรมที่เรียกใช้ฟังก์ชันต้องเปลี่ยนแปลง ทำให้การบำรุงรักษาและการขยายโปรแกรมทำได้ง่ายซึ่งเหมาะสมกับข้อมูลที่มีความ

ซับซ้อนและระบบที่มีขนาดใหญ่ ดังนั้นจึงได้นำเอาหลักการนี้มาใช้ออกแบบระบบงานบริหารพัสดุสายช่างอากาศ ซึ่งประสบปัญหาในการเปลี่ยนแปลงโครงสร้างข้อมูลและต้องแก้ไขโปรแกรมด้วยเสมอ ทำให้มีปัญหาในการปรับปรุงระบบงาน

Object-Oriented Programming (OOP) relies on three basic concepts: data abstraction, inheritance, and polymorphism. Data abstraction refers to ability to define abstract data type or ADT

that encapsulate some data together with a set of well-defined operations. Such user-defined data types predefined hierarchy with a view to share can represent in software. Inheritance is the mechanism that enables one object to behave just like another, except for some modifications. Polymorphism refers to the fact the different objects react differently to the same message. The aeronautical supply management system organizes programs using a collection of objects whose classes are organized in code and data through inheritance.

1. บทนำ

ตั้งแต่คอมพิวเตอร์ยุคแรกโปรแกรมเมอร์พยายามหาวิธีการเขียนโปรแกรมที่มีความซับซ้อนโปรแกรมคอมพิวเตอร์ในระยะแรกคือ ลำดับของคำสั่งภาษาเครื่องที่ถูกนำ (Fetch) เข้าไปในหน่วยความจำโดยหน่วยประมวลผลกลางและทำงานตามคำสั่งนั้น

ภาษาแอสเซมบลีได้ปรับปรุงให้สามารถใช้ชื่อนิโมนิค (mnemonic) แทนคำสั่งภาษาเครื่อง และใช้ชื่อของสัญลักษณ์แทนตำแหน่งของหน่วยความจำได้ ตัวแปลที่เรียกว่าแอสเซมเบลเลอร์จะเปลี่ยนโปรแกรมภาษาแอสเซมบลีให้เป็นภาษาเครื่อง ในเวลาต่อมาแอสเซมเบลเลอร์เริ่มมีคำสั่งเฉพาะหรือไดเรกทีฟ (directive) ซึ่งยอมให้โปรแกรมเมอร์จัดกลุ่มของข้อมูลพื้นฐานเป็นโครงสร้างของข้อมูลแล้วตั้งชื่อโครงสร้างของข้อมูลนั้น และเขียนโปรแกรมโครงสร้างของข้อมูลนั้น และเขียนโปรแกรมโครงสร้าง แต่แอสเซมเบลเลอร์ยังคงบังคับให้แนวความคิดในการเขียนโปรแกรมอยู่ในรูปแบบของคำสั่งภาษาเครื่อง ซึ่งซีพียูสามารถทำงานตามคำสั่งนี้ได้

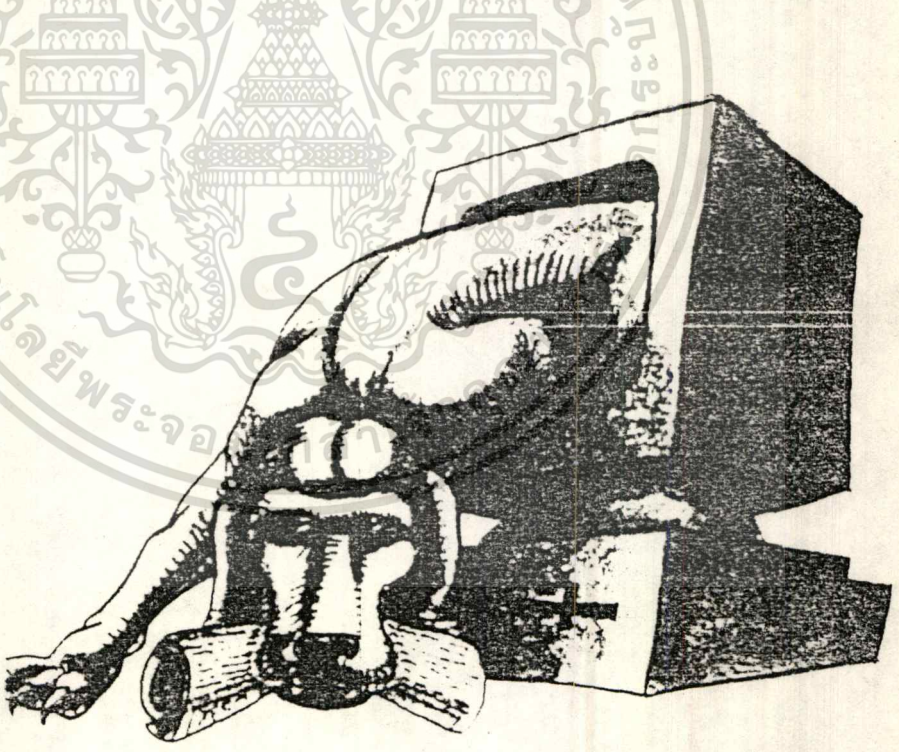
ภาษาในระดับที่สูงกว่า เช่น ภาษาฟอร์แทรน เบสิก ปาสกาล และภาษา C ได้จัดการผูกติดกับคำสั่งภาษาเครื่องของซีพียู โดยการกำหนดประเภทของข้อมูลมาตรฐานเช่นเลขจำนวนเต็ม เลขจำนวนทศนิยม และตัวอักษร ฯลฯ ซึ่งสามารถใช้ในประโยคคำสั่งที่ถูกแปลให้เป็นคำสั่งภาษาเครื่อง โปรแกรมใช้งานที่มีอยู่ส่วนมากเขียนขึ้นในรูปแบบผสมกันระหว่างภาษาระดับสูงกับภาษาแอสเซมบลี

รูปแบบการเขียนโปรแกรมที่ได้รับการยอมรับมีการจัดข้อมูลที่มีความสัมพันธ์กันโดยใช้คำสั่งโครงสร้าง เช่น คำสั่ง Record ในภาษาปาสกาล หรือ คำสั่ง Struct ในภาษา C แล้วจัดกลุ่มของข้อมูลที่ได้ให้เป็นหน่วยเดียว เมื่อมีการจัดโครงสร้างของข้อมูลตั้งนี้การเขียนโปรแกรมใช้งานจึงเขียนในรูปของชุดคำสั่งซึ่งกระทำกับโครงสร้างของข้อมูลเรียกชุดคำสั่งนี้ว่าโพรซีเยอร์ (Procedure)

ถึงแม้ว่าการเขียนโปรแกรมด้วยวิธี “ออกแบบโครงสร้างของข้อมูลและเขียนฟังก์ชันที่กระทำกับข้อมูล” สามารถเขียนได้ดี แต่ความซับซ้อนของซอฟต์แวร์ได้เพิ่มมากขึ้นเพื่อให้ฮาร์ดแวร์มีขีดความสามารถเพิ่มขึ้นเช่นซีพียูเร็วขึ้น ทำกราฟิกให้ดีขึ้นทำระบบเครือข่ายง่ายขึ้น ฯลฯ ผู้ใช้ยอมคาดหวัง

ว่าความสามารถของฟังก์ชันจะต้องเพิ่มขึ้น และโปรแกรมจะรวมคุณลักษณะต่าง ๆ เอาไว้ด้วย เช่น ระบบการติดต่อของผู้ใช้กับระบบวินโดว์ ระบบการบันทึกข้อมูลในคอมพิวเตอร์ ทั้งขนาดเล็กและขนาดใหญ่ และสามารถทำงานในสภาวะของระบบเครือข่าย ฯลฯ จากสาเหตุของความซับซ้อนเหล่านี้ทำให้มีโปรแกรมเมอร์หลายคนหันมาใช้เทคนิคของการเขียนโปรแกรมเชิงวัตถุ (Object-Oriented Programming หรือ OOP)

ไม่ได้มีอะไรเป็นสิ่งใหม่ในการเขียนโปรแกรมเชิงวัตถุ แนวความคิดก็คือการกำหนดข้อมูลชนิดนามธรรม (data abstraction) การถ่ายทอด (inheritance) และการใช้คำสั่งเดียวกันแต่ให้ผลการทำงานที่แตกต่างกัน (polymorphism) แล้วอะไรคือความใหม่ที่



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นสิ่งที่เพิ่มความสนใจในหมู่ของโปรแกรมเมอร์ทั่วไปและโปรแกรมเมอร์ภาษา C เหตุผลข้อหนึ่งก็คือความนิยมการเขียนโปรแกรมภาษา C++ ซึ่งโปรแกรมเมอร์หลายคนมองเหมือนว่าเป็นส่วนที่พัฒนาต่อมาจากภาษา C และเป็นภาษาที่ในปัจจุบันผู้พัฒนาซอฟต์แวร์ส่วนมากเลือกใช้ ภาษา C++ ได้เพิ่มคำสั่งโครงสร้างเข้าไปในภาษา C หลายอย่างซึ่งช่วยให้เทคนิคเชิงวัตถุได้โดยตรง ถ้าเป็นโปรแกรมเมอร์ภาษา C อยู่แล้วจะเรียนรู้เกี่ยวกับกฎเกณฑ์ของภาษา C++ ได้ง่ายขึ้นเพียงแต่ต้องเปลี่ยนแปลงแนวความคิดเสียใหม่ถ้าต้องการจะใช้หลักการเขียนโปรแกรมเชิงวัตถุ วิธีที่ดีที่สุดในการเรียนรู้ภาษา C++ คือการทำความเข้าใจกับแนวความคิดของโอโอพี (OOP) แล้วดูว่าภาษา C++ ทำตามแนวความคิดนั้นได้อย่างไร

คือวิธีการออกแบบซอฟต์แวร์และการทำให้ซอฟต์แวร์นั้นเป็นจริงขึ้นมาเท่านั้น การใช้เทคนิคของการเขียนโปรแกรมเชิงวัตถุไม่ได้บ่งบอกถึงความสำเร็จที่สามารถมองเห็นได้ของซอฟต์แวร์ อย่างไรก็ตามเมื่อโปรแกรมเมอร์จะทำซอฟต์แวร์ให้เป็นจริงขึ้นมาก็สามารถที่จะหวังประโยชน์จากการใช้วิธีการเขียนโปรแกรมเชิงวัตถุได้โดยเฉพาะซอฟต์แวร์ของโครงการขนาดใหญ่ เพราะว่าโอโอพีทำให้สามารถอยู่ในระดับแนวความคิด (Conceptual) ได้และสร้างแบบจำลองระดับสูงของปัญหาที่กำลังพยายามจะแก้ปัญหานั้นได้ซึ่งสามารถจัดการกับความซับซ้อนต่าง ๆ

ได้ดีกว่าวิธีอื่น ๆ ที่มักจะบังคับให้จัดปัญหาให้เข้ากับลักษณะของภาษานั้น สามารถได้รับประโยชน์จากความเปลี่ยนแปลง ของออบเจกต์ และทำโปรแกรมให้เป็นจริงขึ้นมาในรูปของหน่วยที่มีความสัมพันธ์แบบเป็นอิสระไม่ขึ้นต่อกัน ซึ่งสามารถจัดการดูแลและขยายได้ง่ายโดยสามารถจะใช้รหัสในออบเจกต์ร่วมกันได้โดยวิธีการถ่ายทอด

ชั้นสองคือโอโอพีไม่ได้ขึ้นกับภาษาโปรแกรมใด ๆ ถึงแม้ว่าภาษาโปรแกรมนั้นจะทำเทคนิคของโอโอพีได้ก็เพียงแต่ทำให้เทคนิคนั้นเป็นจริงขึ้นมาได้ง่ายขึ้นเท่านั้น

2. อะไรคือการเขียนโปรแกรมเชิงวัตถุ

การเขียนโปรแกรมเชิงวัตถุได้นิยามข้อมูลชนิดนามธรรมเพื่อใช้แทนข้อมูลจริงที่มีความซับซ้อนหรือใช้แทนออบเจกต์ชนิดนามธรรมแล้วสร้างโปรแกรมโดยใช้ข้อมูลแอบสแทรคชัน (Data abstraction) กล่าวถึงการใช้นิยามของข้อมูลชนิดนามธรรม ส่วนในหัวข้อของ Inheritance และ Polymorphism นั้นกล่าวถึงกลไกที่ทำให้สามารถได้รับประโยชน์จากคุณลักษณะของข้อมูลชนิดนามธรรมซึ่งก็คือ ออบเจกต์ในโอโอพีนั่นเอง

กันที่จะเข้าไปสู่หัวข้อของโอโอพีพิจารณาข้อสังเกตสองข้อ ข้อหนึ่งโอโอพี



52 คอหพิวตอร์

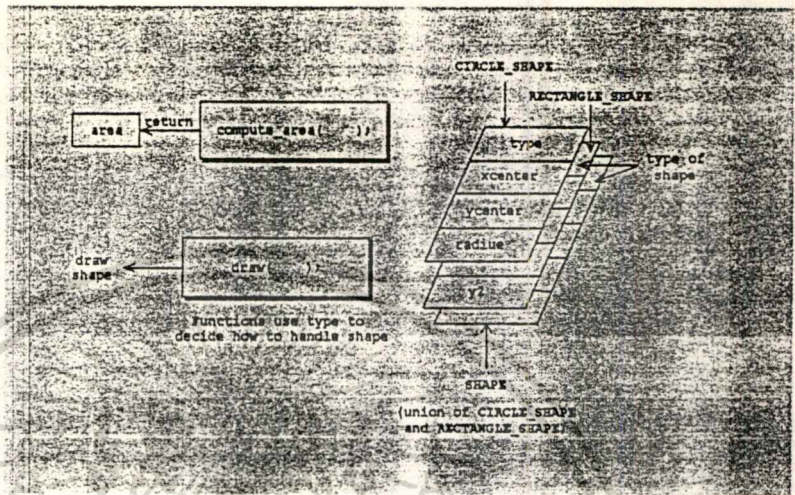
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. การเขียนโปรแกรมโครงสร้าง

Procedure Oriented Programming)

ก่อนที่จะพูดถึงหัวข้อไอโอพี มาดูภาษาที่ใช้เขียนโปรแกรมโครงสร้าง เช่น ภาษา C วิธีเขียนโปรแกรมโครงสร้างคือจัดลำดับของปัญหาเป็นลำดับของขั้นตอนที่จะต้องทำ จัดข้อมูลที่มีความสัมพันธ์กันโดยใช้คำสั่งในภาษา C หรือ STRUCT เขียนฟังก์ชันหรือโพรซีเจอร์ที่จำเป็นกระทำกับข้อมูลและต้องใช้ในการทำงาน จัดลำดับของงานให้สมบูรณ์เพื่อใช้งานการแก้ปัญหา ถึงแม้ว่าข้อมูลจะถูกจัดเป็นแบบโครงสร้าง จุดสนใจจุดแรกอยู่ที่ฟังก์ชัน ตัวอย่างโปรแกรมภาษา C เช่น การจัดการเกี่ยวกับรูปทรงเรขาคณิต คือรูปสี่เหลี่ยมและวงกลม โปรแกรมต้องสามารถที่จะวาดรูปทรงใด ๆ และคำนวณพื้นที่ของรูปนั้นได้ นั่นคือสามารถแบ่งการทำงานของโปรแกรมได้เป็นสองฟังก์ชันคือวาดรูปและคำนวณพื้นที่ เริ่มต้นกำหนดโครงสร้างข้อมูลทั้งสองรูปทรงแล้วเชื่อมให้เป็นหน่วยเดียวกันด้วยคำสั่ง UNION โดยมีรหัสกำหนดว่าขณะนั้นใช้โครงสร้างข้อมูล ของรูปทรงใด จากนั้นเขียนฟังก์ชันทั้งสองฟังก์ชันของแต่ละรูปทรงโดยใช้คำสั่ง SWITCH เป็นตัวระบุว่าจะใช้รูปทรงใดดังรูปที่ 1

ปัญหาข้อหนึ่งของการเขียนโปรแกรมโครงสร้างคือเมื่อต้องการเพิ่มโครงสร้างของข้อมูลแบบใหม่ เช่นรูปสามเหลี่ยม จะต้องทำดังต่อไปนี้



รูปที่ 1 การจัดการเกี่ยวกับรูปทรงเรขาคณิตด้วยภาษา C

- ขั้นที่ 2 เพิ่มสมาชิกใหม่ในส่วนของ UNION
- ขั้นที่ 3 เพิ่มรูปทรงใหม่ในฟังก์ชันทั้งสอง
- ขั้นที่ 4 แก้ไขตัวโปรแกรมเพิ่มส่วนที่ต้องการใช้โครงสร้างของข้อมูลที่เพิ่มใหม่

เหตุผลที่แสดงตัวอย่างข้างต้นเพื่อชี้ให้เห็นว่าเมื่อต้องการเพิ่มประเภทของข้อมูลแบบใหม่ต้องมีการแก้ไขตลอดทั้งโปรแกรม และในความเป็นจริงการเขียนโปรแกรมเกี่ยวข้องกับการเพิ่มข้อมูลมากมายนั่นคือ เมื่อมีการเปลี่ยนแปลงจะต้องแก้ไขเพิ่มข้อมูลทุก ๆ เพิ่มข้อมูล

ในหัวข้อถัดไปจะเห็นว่าการใช้วิธีของโปรแกรมเชิงวัตถุสามารถหลีกเลี่ยงปัญหานี้ได้ โดยโครงสร้างของข้อมูลและฟังก์ชันที่กระทำกับข้อมูลนั้นเหมือนเดิม มีการแก้ไขโปรแกรมน้อยที่สุด นั่นคือข้อดีของไอโอพี

4. คำศัพท์ของไอโอพี

มีแนวความคิด 3 ข้อภายใต้หัวข้อไอโอพี คือ

1. ข้อมูลแอบสแทรกชัน
2. อินเฮริทนต์
3. โพลิมอร์ฟิซึม

คำศัพท์เหล่านี้ไม่ใช่คำศัพท์ใหม่แต่การนำมาใช้เป็นรากฐานของไอโอพีเป็นเรื่องใหม่

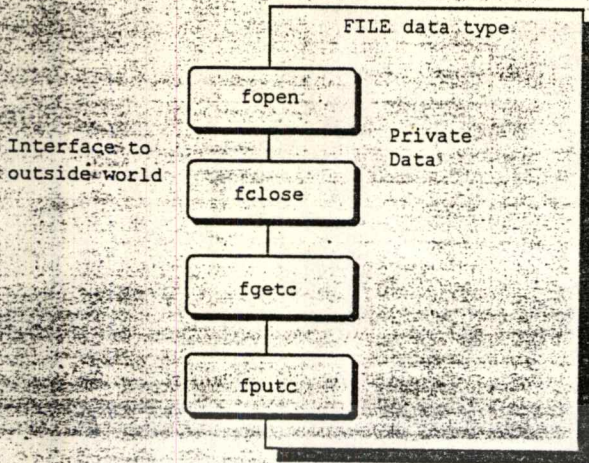
4.1 ข้อมูลแอบสแทรกชัน

เพื่อทำความเข้าใจกับคำศัพท์ข้อมูลแบบแอบสแทรกชัน พิจารณารูทีนของไฟล์ไอโอ (file I/O routine) ในไลบรารี (Library) ของภาษา C ซึ่งรูทีนเหล่านี้มองเห็นไฟล์เป็นเพียงตัวอักษรเรียงติดต่อกันและสามารถถูกกระทำได้หลายวิธีโดยการเรียกใช้รูทีนของไฟล์ไอโอ เช่น fopen เพื่อเปิดไฟล์ fputs เพื่อเขียนข้อมูลในไฟล์ การกำหนดชนิดของข้อมูลเป็น File ซึ่งเก็บข้อมูลเกี่ยวกับไฟล์ ทำให้แบบจำลองนามธรรมของไฟล์เป็นจริงขึ้นมา ในภาษา C ใช้คำสั่ง STRUCT และ TYPEDEF กำหนดข้อมูลชนิดไฟล์

สามารถคิดถึงข้อกำหนดของไฟล์รวมอยู่กับฟังก์ชันที่กระทำข้อมูลนั้น (encapsulation) เป็นประเภทของข้อมูล

ขั้นที่ 1 กำหนดโครงสร้างของข้อมูลแบบใหม่ เช่นรูปสามเหลี่ยม

แม้ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2 ข้อมูลชนิด File ในภาษา C ตัวอย่างขอเอ็ดที่

หรือ char การใช้ข้อมูลประเภทไฟล์นั้นไม่จำเป็นต้องรู้โครงสร้างจริง ๆ ของไฟล์ เพราะว่าโครงสร้างของไฟล์อาจเปลี่ยนแปลงไปตามประเภทของระบบคอมพิวเตอร์แต่การทำงานของรูทีนไฟล์ไอโอจะเหมือนกันในทุกๆ ระบบที่เป็นเช่นนี้ได้เพราะว่าไม่ได้กระทำกับโครงสร้างของข้อมูลโดยตรงแต่ทำบนฟังก์ชันและแมคโคร (Macros) ซึ่งได้ซ่อนรายละเอียดภายในของไฟล์ทั้งสิ้นไว้

ข้อมูลแอบสแทรกชันคือการกำหนด ชนิดของข้อมูลรวมทั้งข้อมูลส่วนที่ซ่อนไว้ (เรียกว่า Abstract Data Type หรือ ADT) คำนิยามของเอ็ดที่คือ กำหนดข้อมูลภายในเอ็ดที่และฟังก์ชันที่กระทำกับเอ็ดที่นั้น ข้อมูลส่วนที่ซ่อนไว้ทำให้ไม่ต้องเปลี่ยนแปลงโปรแกรมที่ใช้ฟังก์ชันที่กระทำกับเอ็ดที่นั้นเมื่อมีการเปลี่ยนแปลงโครงสร้างภายในของเอ็ดที่ ตัวอย่างของเอ็ดที่คือ ข้อมูลประเภท File ในภาษา C ดังรูปที่ 2

4.2 ออบเจกต์ คลาส เมธอด

ออบเจกต์ในโอโอพีสร้างมาจากเอ็ดที่ ซึ่งเอ็ดที่คือกลุ่มของตัวแปรที่มีฟังก์ชันที่จำเป็นที่กระทำกับตัวแปรเหล่านั้นรวมอยู่ด้วย ตัวแปรใช้แทนข้อมูลข่าวสารภายในออบเจกต์ แต่ฟังก์ชันใช้กำหนดการกระทำที่จะกระทำกับออบเจกต์นั้น สามารถคิดถึงเอ็ดที่เหมือนกับเป็นเทมเพลท (Template) ที่ใช้แทนสมาชิก ของออบเจกต์ มักเรียกเทมเพลทว่าคลาส (Class) เพราะฉะนั้นคลาสจึงมีความหมายเหมือนกับเอ็ดที่

ฟังก์ชันที่กระทำกับออบเจกต์เรียกว่า Method เป็นคำที่มาจากภาษา Smalltalk ซึ่งเป็นต้นใช้กำหนดความประพฤติของออบเจกต์ ในภาษา C++ เรียกว่าเมมเบอร์ฟังก์ชัน (member functions) คำศัพท์คำอื่น ๆ ในโอโอพีที่มีที่มาจากภาษา Smalltalk แนวความคิดในการเรียกใช้ method คือการส่งข้อความไปยังออบเจกต์เพื่อระบุให้ตัวใด

ตัวหนึ่งใดทำงานในภาษา C++ ก็คือการเรียกเมมเบอร์ฟังก์ชันในออบเจกต์

4.3 อินเฮริแทนซ์

ในการกำหนดข้อมูลชนิดนามธรรมไม่สามารถกำหนดคุณสมบัติที่สำคัญของออบเจกต์ได้ครบทั้งหมดและในความเป็นจริงนั้นออบเจกต์ไม่ได้แยกกันอยู่แบบเดี่ยว ๆ แต่มีความสัมพันธ์กับออบเจกต์อื่น ๆ เสมอ ดังนั้นจึงสามารถที่จะกำหนดออบเจกต์ใหม่โดยอธิบายคุณสมบัติที่แตกต่างจากออบเจกต์เดิมที่มีอยู่แล้ว ซึ่งเป็นแนวความคิดที่สำคัญของโอโอพีเรียกว่าอินเฮริแทนซ์ คือการถ่ายทอดข้อมูลและความประพฤติจากคลาสแม่ (Parent class) ไปสู่คลาสลูก (child class) โดยในภาษา C++ เรียกคลาสแม่ว่า เบสคลาส (base class) และเรียกคลาสลูกว่าดิริวไวด์คลาส (derived class)

4.4 การอินเฮริแทนซ์มากกว่าหนึ่ง (Multiple Inheritance)

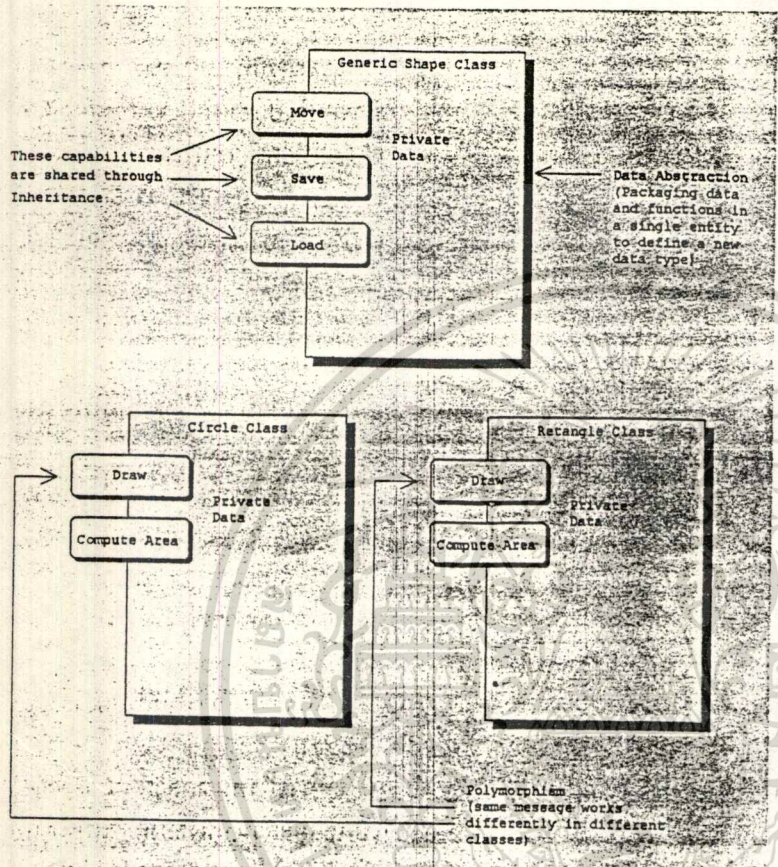
หมายความว่าดิริวไวด์คลาสมีการถ่ายทอดมาจากหลาย ๆ เบสคลาสซึ่งในภาษาเชิงวัตถุหลายภาษาไม่สามารถทำได้ แต่ในภาษา C++ สามารถทำได้

4.5 โพลิมอร์ฟิซึม

ความหมายตามคำศัพท์ของโพลิมอร์ฟิซึมคือมีคุณสมบัติหลายอย่าง แต่ความหมายโอโอพีหมายถึงการสั่งให้กระทำด้วยคำสั่งเดียวกันกับออบเจกต์ที่แตกต่างกันจะมีการกระทำที่แตกต่างกัน เช่น ในตัวอย่างข้างต้นสั่งให้วาดรูปจะได้รูปวงกลมหรือสี่เหลี่ยมขึ้นกับประเภทของออบเจกต์นั้น

จากตัวอย่างที่แล้วสามารถเขียนด้วยภาษา C++ โดยใช้เทคนิคของโอโอพี ดังรูปที่ 3

2 แนวคิด



รูปที่ 3 Data abstraction, inheritance และ Polymorphism ในโอโอพี

5. ระบบงานบริหารพัสดุสายช่างอากาศ

ในปัจจุบันระบบงานบริหารพัสดุสายช่างอากาศของกรมช่างอากาศควบคุมด้วยระบบคอมพิวเตอร์ที่ใช้โปรแกรมโครงสร้าง ซึ่งปัญหาที่ประสบอยู่เสมอ ๆ คือเมื่อมีการเปลี่ยนแปลงโครงสร้างของข้อมูลจะต้องเปลี่ยนแปลงโปรแกรมทั้งหมดที่เกี่ยวข้องด้วยดังเช่นตัวอย่างที่อธิบายในหัวข้อที่ผ่านมา และต้องการที่จะเพิ่มบางส่วนเข้าไปในระบบงานการดำเนินงานก่อนข้างลำบากยุ่งยากเพราะต้องคำนึงถึงโครงสร้างของแฟ้มข้อมูลเดิม และผลกระทบที่จะเกิด

กับโปรแกรมที่ใช้แฟ้มข้อมูลนั้น ดังนั้นจึงได้นำเอาหลักการของการเขียนโปรแกรมเชิงวัตถุมาใช้ในการออกแบบระบบงานบริหารพัสดุช่างอากาศ

5.1 ขั้นตอนของระบบงาน

เมื่อมีผู้ใช้เบิกพัสดุ เจ้าหน้าที่พัสดุตรวจสอบประเภทของพัสดุว่าเป็นพัสดุลีนเปลือง พัสดุขอมหุนเวียนหรือพัสดุที่ต้องมีอัตราจ่าย (เช่น เครื่องมือช่าง) ถ้าเป็นพัสดุขอมหุนเวียนต้องตรวจว่ามีของส่งคือหรือไม่ เพราะถ้าไม่มีของส่งคืนจะไม่จ่ายให้จากนั้นจึงตรวจสอบว่ามีของจ่ายหรือไม่ ถ้ามีจ่าย ถ้าไม่มีตรวจว่ามีพัสดุที่ใช้แทนกันได้หรือไม่ ถ้าจ่ายพัสดุใช้แทน ถ้าไม่มี

ทำรายการค้างจ่ายไว้ และพยายามหาหนทางนำพัสดุมาจ่ายเช่น ตรวจสอบรายการรอรับจากการจัดหา รายการรอรับจากงานซ่อม เพื่อที่จะได้เร่งรัดและติดตามให้ได้พัสดุ หากมีความจำเป็นเร่งด่วน เช่น เครื่องบินบินไม่ได้ อาจจะต้องตามขอคืนจากผู้ใช้งานหน่วยก่อนหรือจัดหาโดยวิธีพิเศษ เมื่อได้รับพัสดุจากการจัดหาตรวจสอบว่าหมายเลขตรงกับที่จัดหาหรือไม่ ถ้าไม่ตรงตรวจสอบว่าเป็นพัสดุที่ใช้แทนกันได้หรือไม่ ถ้าไม่ได้ไม่รับ ถ้าได้จึงรับพัสดุนั้นและตรวจสอบว่ามีรายการค้างจ่ายหรือไม่ จะได้ทำหักล้างค้างจ่ายเสียก่อนนำไปเก็บคลังพัสดุต่อไป

เมื่อได้รับพัสดุจากการส่งคืนหรืองานซ่อมตรวจสอบว่าเป็นพัสดุดี หรือพัสดุชำรุด ถ้าเป็นพัสดุชำรุดตรวจสอบว่าสามารถซ่อมได้ไหม ถ้าไม่ได้นำจ่ายพัสดุแล้วจัดหา ถ้าเป็นพัสดุดีตรวจสอบว่ามีรายการค้างจ่ายหรือไม่เพื่อทำหักล้างค้างจ่ายก่อนนำไปเก็บในคลังพัสดุดังแผนผังรูปที่ 4

5.2 ออกแบบระบบงานโดยใช้หลักการของระบบเชิงวัตถุ

จากขั้นตอนการบริหารพัสดุสายช่างอากาศตั้งที่กล่าวมาข้างต้น ออกแบบระบบงานโดยใช้หลักการของระบบเชิงวัตถุ

กำหนดคลาสประเภทพัสดุ (Generic Item Class) ซึ่งมีข้อมูลรายละเอียดของพัสดุ เช่น หมายเลขพัสดุ ชื่อจำนวนพัสดุดี จำนวนพัสดุชำรุด รายการรอรับจากการจัดหา (Due In) รายการรอรับจากงานซ่อม (Work Order) รายการค้างจ่าย (Due Out) รายการที่มีการเปลี่ยนแปลงต่าง ๆ (Transaction) เป็นต้น ดังรูปที่ 5

2 ทศวรรษอีอีพี

	พัสดุสิ้นเปลือง	พัสดุซ่อมหมุนเวียน
การรับพัสดุ	ตรวจสอบว่ามีรายการค้างจ่ายหรือไม่ มีทำหักค้างจ่ายที่เหลือเพิ่มจำนวนพัสดุนั้นหรือไม่ ไม่มีเพิ่มจำนวนพัสดุนั้น	ตรวจสอบสภาพพัสดุ - พัสดุนั้น ตรวจสอบว่ามีรายการค้างจ่ายหรือไม่ ถ้ามีทำหักค้างจ่ายที่เหลือเพิ่มจำนวนพัสดุนั้น ถ้าไม่มีเพิ่มจำนวนพัสดุนั้น - พัสดุชำรุด เพิ่มจำนวนพัสดุชำรุด
การจ่ายพัสดุ	ตรวจสอบว่ามีของจ่ายหรือไม่ ถ้ามีจ่าย ไม่มีตรวจสอบว่ามีหมายเลขแทนหรือไม่ ถ้ามีจ่ายหมายเลขแทน ไม่มีทำรายการค้างจ่าย	ตรวจสอบสภาพพัสดุ - พัสดุนั้น ตรวจสอบว่ามีพัสดุสงคืนหรือไม่ ถ้าไม่มีไม่จ่าย ถ้ามี และมีของจ่ายจึงจ่าย ถ้าไม่มีของ ตรวจสอบหมายเลขแทนเพื่อจ่ายแทน ถ้าไม่มีหมายเลขแทนทำรายการค้างจ่าย - พัสดุชำรุด ตรวจสอบว่าซ่อมได้หรือไม่ ได้ส่งซ่อม ไม่ได้จำหน่ายพัสดุและจัดหาพัสดุ

รูปที่ 7 ตาราง

แบ่งพัสดุออกตามประเภทของพัสดุ คือเป็นพัสดุสิ้นเปลืองและพัสดุซ่อมหมุนเวียน แล้วกำหนดคลาสพัสดุสิ้นเปลือง (Consumable Item) และคลาสพัสดุซ่อมหมุนเวียน (Reparable Item) เป็นไดรฟ์คลาสสามารถถ่ายทอดคุณสมบัติมาจากเบสคลาสคือ คลาสประเภทพัสดุ มี method การรับพัสดุ (Receive Item) และการจ่ายพัสดุ (Issue-Item) ดังรูปที่ 6 การเรียกใช้ method ในคลาสพัสดุสิ้นเปลือง และคลาสพัสดุซ่อมหมุนเวียนจะมีการกระทำที่แตกต่างกันดังรูปที่ 7

6 สรุป

นอกจากคลาสที่กล่าวมาข้างต้นแล้วยังสามารถกำหนดคลาสเพิ่มขึ้นใหม่ได้อีกตามความจำเป็นโดยไม่ต้อง

กำหนดเผื่อไว้ล่วงหน้าเช่นวิธีเก่าบางครั้งรุนแรงถึงขั้นต้องรื้อสร้างใหม่ถ้าเผื่อไว้ไม่เพียงพอทำให้ไม่สามารถจะขยายได้ แต่วิธีการของระบบเชิงวัตถุไม่มีปัญหาเช่นนี้ ทำให้การออกแบบระบบทำได้สะดวกและง่าย และสามารถที่จะกำหนดคลาสชั้นใหม่โดยการอ้างอิงคุณสมบัติของคลาสเดิม และการเพิ่มคลาสใหม่ไม่มีผลกระทบต่อส่วนอื่น ๆ ที่มีอยู่เดิมภายในระบบ ทำให้การออกแบบระบบสามารถที่จะเริ่มต้นจากการทดสอบแล้วจึงค่อย ๆ ขยายจนครบสมบูรณ์ทั้งระบบ นอกจากนี้สามารถละเว้นส่วนที่ข้อมูลยังไม่สมบูรณ์ไว้ก่อนแล้วมาเพิ่มเติมเข้าไปในระบบภายหลัง

แบบระบบที่มีขนาดใหญ่เช่น ระบบงานบริหารพัสดุช่างอากาศ เป็นต้น เอกสารอ้างอิง

- [1] Bertrand Meyer, 1988, "Object-Oriented Software Construction" (Hertfordshire, Great Britain : Prentice-Hall International UK)
- [2] Stroustrup, B., 1986, "The C++ Programming Language", Addison-Wesley.
- [3] Rumbaugh, J., Blaha, H., Premerlani, W., Eddy F. and Lorensen, W., 1991, "Object-Oriented Modeling and Design" Prentice Hall, Englewood Cliffs, NJ
- [4] กองทัพอากาศ 2508 "คู่มือว่าด้วยการพัสดุ"
- [5] Shloer, S. and Mellor, S.J., 1988 "Object-Oriented System Analysis : Modeling the world in Data", Yourdon Press, Englewood Cliff, NJ.

สามารถจัดการกับความซับซ้อนของข้อมูลได้ดีและเหมาะกับการออกสาย



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมนูการใช้งาน

สามารถเลือกประเภทของการทำงานได้จากเมนูหลัก ซึ่งมีรายละเอียดดังนี้

MAIN SCREEN

1. Add Item
2. Delete Item
3. Update Item
4. Issue Item
5. Issue to Shop
6. Turn In
7. Receive from Dueln
8. Receive from Shop
9. Back Order Release
10. Procuring
11. Item Detail
99. End Job

Select action code ___

จอภาพของระบบงาน

จอภาพของระบบงานแต่ละจอภาพจะมีลักษณะและวิธีการใช้ที่คล้ายๆ กัน เพื่อให้ผู้ใช้สามารถใช้งานได้ง่าย ไม่ต้องจดจำวิธีการมากมาย โดยที่แต่ละจอภาพจะเริ่มต้นการทำงานด้วยการให้ผู้ผู้ใช้ใส่รหัสประเภทของเครื่องบินและหมายเลขพัสดุ โปรแกรมจะนำรายละเอียดของพัสดุนั้นมาแสดงที่จอภาพ จากนั้นผู้ใช้ก็จะใส่รายละเอียดเกี่ยวกับการทำงานในแต่ประเภทที่ต้องการต่อไป จอภาพแบ่งออกเป็น 11 จอภาพคือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การลบรายการพัสดุ

กรณีที่มีการจำหน่ายพัสดุจะต้องลบพัสดุนั้นออกไป โปรแกรมจะตรวจสอบก่อนว่ามีรายการรับการจัดหาพัสดุหรือรับงานซ่อมพัสดุหรือไม่ ถ้ามีโปรแกรมจะไม่ยอมลบพัสดุนั้นให้ ทั้งนี้เพื่อเป็นการป้องกันข้อผิดพลาดที่อาจเกิดขึ้นได้ รายละเอียดดังนี้

DELETE ITEM

CONSUMABLE

Aircraft type _ onHand ____

FSC/NIIN or PN _____

Nomenclature _____ ERRC _

Unit Price _____ UI _

SOS _____

ISG _____

Are you sure?(Y/N)___

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. การจ่ายพัสดุไปซ่อม

การส่งพัสดุชำรุดไปซ่อม ซึ่งจะทำรายการรอรับงานซ่อมเอาไว้ รายละเอียดดังนี้

ISSUE TO SHOP

REPARABLE

Aircraft type _____ onHand _____

repair _____

FSC/NIIN or PN _____

Nomenclature _____ ERRC _____

Unit Price _____ UI _____

Document of Depot _____

Shop _____

Shop Code _____

Quantity _____

Date _____

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. การรับพัสดุส่งคืน

เมื่อผู้ใช้หมดความจำเป็นใช้งานพัสดุจะส่งคืนมายังคลังใหญ่ รายละเอียดดังนี้

TURN IN

REPARABLE

Aircraft type _____ onHand _____

repair _____

FSC/NIIN or PN _____

Nomenclature _____ ERRC _____

Unit Price _____ UI _____

Document of Depot _____

Organization _____

Organization Code _____

Quantity _____

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DETAIL OF ITEM

REPARABLE

Aircraft type _ onHand ____

repair ____

FSC/NIIN or PN _____

Nomenclature _____ ERRC _

Unit Price _____ UI _

>>> DUE OUT <<<

>>> DUE IN <<<

>>> WO <<<



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

- [1] Bertrand Meyer, Object-Oriented Software Construction, Hertfordshire, Great Britain: Prentice hall International, UK, 1988
- [2] Rumbaugh, J., Blaha, H., Premerlani, W., Eddy F, and Lorenzen, W. , Object-Oriented Modeling and Design, Prentice Hall, Englewood Cliffs, New Jersey, 1991
- [3] Digitalk Inc., Smalltalk/v Tutorial and Programming Handbook, Digitalk Inc., 1986
- [4] Shlaer, s., and Mellor, s.J., Object-Oriented System Analysis: Modeling the word in Data, Prentice Hall, Englewood Cliffs, New Jersey, 1991
- [5] Peter Coad and Edward Yourdon, Object-Oriented Analysis, Prentice Hall, Englewood Cliffs, New Jersey, 1991
- [6] Embley, w., Kurtz, d., and Woodfield, n., Object-Oriented Systems Analysis A Model-Driven Approach, Prentice Hall, Englewood Cliffs, New Jersey, 1992
- [7] กองทัพอากาศ, คู่มือว่าด้วยการพัสดุ, โรงพิมพ์กองทัพอากาศ, กรุงเทพฯ 2508
- [8] Mark Mullin, Object-Oriented Program Design with Example in c++, Addison-Wesley Publishing Company, Inc., 1989
- [9] J. g. Hughes, Object-Oriented Database, Prentice hall International, UK, 1991
- [10] Cardenas, f., and Mcleod, Research Foundations in Object-Oriented and Semantic Database Systems

ประวัติผู้เขียน

ชื่อผู้เขียน	น.ท.หญิง นพพร สงวนทรัพย์
วันเดือนปีเกิด	22 เมษายน 2499
สถานที่เกิด	จ.ปทุมธานี
วุฒิการศึกษาระดับปริญญาตรี	วท.บ.(คณิตศาสตร์)
สถานที่สำเร็จการศึกษา	คณะวิทยาศาสตร์ มหาวิทยาลัยเกษตรศาสตร์
ปีที่สำเร็จการศึกษา	ปีการศึกษา 2522
ผลงานวิชาการ	วารสารสมาคมคอมพิวเตอร์ “การใช้โปรแกรมเชิงวัตถุกับ ระบบงานบริหารพัสดุสายช่างอากาศ”
สถานที่ทำงาน	กรมการسنเทศทหาร กองบัญชาการทหารสูงสุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้