

การพัฒนาระบบคิดเงินลูกค้าโดยใช้ ยูเอ็มแอล

Billing System Development Using UML

โดย

นายอนันตวุฒิ เอื้อพัฒน์วงศ์

รหัส 42067172



H001828

อาจารย์ที่ปรึกษา

รศ.ดร.วิเชียร เปรมชัยสวัสดิ์

วัน เดือน ปี.....	16 ส.ค. 2550
เลขทะเบียน.....	01828
เลขเรียกหนังสือ.....	จุฬ ๑174 ก 2544
"ห้องสมุดคณะเทคโนโลยีสารสนเทศ สอจ."	

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการการพัฒนาระบบงาน
หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
ภาคเรียนที่ 2 ปีการศึกษา 2544
คณะเทคโนโลยีสารสนเทศ
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อหัวข้อ	การพัฒนาระบบคิดเงินลูกค้าโดยใช้ ยูเอ็มแอล
นักศึกษา	นายอนันท์วุฒิ เอื้อพัฒน์วงศ์
อาจารย์ที่ปรึกษา	รศ.ดร. วิเชียร เปรมชัยสวัสดิ์
ระดับการศึกษา	วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2544

บทคัดย่อ

โครงการนี้ เป็นการพัฒนาระบบสารสนเทศ โดยใช้หลักวิธีในการพัฒนาระบบแบบเชิงวัตถุ ตั้งแต่การวิเคราะห์และออกแบบ และนำสิ่งที่ได้ออกแบบมาพัฒนาเป็นระบบที่ใช้งานได้จริง โดยใช้ ยูเอ็มแอล ทั้งภาษา และวิธีการ โดยในโครงการนี้จะพัฒนาระบบงานโดยอาศัยกรณีศึกษาของการพัฒนาระบบการคิดเงินค่าบริการลูกค้าของบริษัทยูไนต์ค บรอดแบนด์ เทคโนโลยี จำกัด

Title	Billing System Development Using UML
Student	Mr.Ananthawut Auaphathanawong
Advisor	Prof. Vichian Pramchaisawad
Level of Study	Master of Science in Information Technology
Major	Information Science
Academic Year	2001

ABSTRACT

This project's main objective is to developing an information system by using an object oriented development methodology, which covered from analysis to designing. The result of design then developing into a working system by using a relational database system and relational database tools. The object-oriented analysis and design is based on UML in language, methodology and tools. This project is using a case study form a Billing System of United Broadband Technology Co., Ltd.

กิตติกรรมประกาศ

โครงการพัฒนาระบบงานนี้ประสบความสำเร็จได้ เพราะได้รับการส่งเสริมและสนับสนุนจากบุคคลหลายฝ่ายเป็นอย่างดี ผู้จัดทำจึงใคร่ขอขอบพระคุณบุคคลดังนี้

1. บิดา, มารดา และพี่ ที่คอยให้ความช่วยเหลือ และให้กำลังใจมาตลอด
2. รศ.ดร.วิเชียร เปรมชัยสวัสดิ์ อาจารย์ที่ปรึกษาโครงการที่กรุณาให้คำปรึกษาและแนะนำในด้านต่างๆ ทั้งในการวิเคราะห์และออกแบบระบบ และให้คำแนะนำวิธีการแก้ปัญหาที่เกิดขึ้น
3. คุณกิตติพงษ์ กลมกล่อม ที่ช่วยให้คำแนะนำและปรึกษาเกี่ยวกับการเขียนโปรแกรม
4. คุณชนะพล กิณีมาน และ คุณนคร ไชยแสนชมภู ที่ช่วยให้คำแนะนำและให้ยืมหนังสือ
5. คุณพัฒนะ วรรณวิไล, คุณราเชนทร์ ศิริเรียรไชย และคุณครองพงศ์ แจ่มใจ ที่คอยให้ความช่วยเหลือต่างๆ และเป็นเพื่อนเที่ยวในเวลาว่างที่เรียดจากการทำงาน

อนันตวุฒิ เอื้อพัฒน์วงศ์

ผู้จัดทำ

สารบัญ

	หน้า
บทคัดย่อภาษาไทย.....	I
บทคัดย่อภาษาอังกฤษ.....	II
กิตติกรรมประกาศ.....	III
สารบัญ.....	IV
สารบัญตาราง.....	VII
สารบัญภาพ.....	X
บทที่	
1. บทนำ.....	1
1.1 วัตถุประสงค์.....	1
1.2 แผนการดำเนินงาน.....	1
1.3 ประโยชน์ที่คาดว่าจะได้รับ.....	2
1.4 ขอบเขตของระบบ.....	2
2. ทฤษฎีที่เกี่ยวข้อง.....	3
2.1 ความรู้พื้นฐานเกี่ยวกับ UML.....	3
2.1.1 Use-Case Diagram.....	4
2.1.2 Class Diagram.....	9
2.1.3 Object Diagram.....	11
2.1.4 State Diagram.....	12
2.1.5 Sequence Diagram.....	13
2.1.6 Collaboration Diagram.....	13
2.1.7 Activity Diagram.....	14
2.1.8 Component Diagram.....	15
2.1.9 Deployment Diagram.....	15

สารบัญ (ต่อ)

	หน้า
2.2 ความรู้พื้นฐานเกี่ยวกับเทคโนโลยีของออปเจกต์.....	16
2.2.1 หลักการพื้นฐานที่เกี่ยวกับออปเจกต์.....	16
2.2.2 การสื่อสารระหว่างออปเจกต์.....	18
2.2.3 การสืบทอดมรดก (Inheritance).....	20
2.2.4 โพลิมอร์ฟิซึม (Polymorphism).....	22
3. การวิเคราะห์และออกแบบระบบ.....	24
3.1 Billing System.....	24
3.2 การทำงานของระบบงานเดิม.....	24
3.3 ขั้นตอนการดำเนินการพัฒนาระบบ.....	25
3.3.1 Problem Definition.....	25
3.3.2 Requirement Analysis.....	26
3.3.3 Specification.....	26
3.3.4 Design.....	26
3.3.4.1 Use Case diagram.....	26
3.3.4.2 Activity Diagram.....	33
3.3.4.3 Sequence Diagram.....	35
3.3.4.4 Class Diagram.....	47
3.3.4.5 State Diagram.....	72
3.3.4.6 Deployment Diagram.....	74
4. การสร้างระบบ.....	75
4.1 เครื่องมือที่ใช้.....	75
4.2 การสร้างฐานข้อมูล.....	76
4.3 ขั้นตอนการใช้เครื่องมือของ Rational 2000.....	76

สารบัญ (ต่อ)

	หน้า
4.4 คุณสมบัติต่างๆในการทำ DDL Code Generation.....	77
4.5 Generated Code.....	82
4.6 การดำเนินการของคลาส.....	84
5. สรุป.....	87
5.1 ข้อดีของการพัฒนาระบบเชิงวัตถุ.....	87
5.2 ปัญหาที่เกิดขึ้น.....	87
6. คู่มือการใช้.....	88
6.1 การเข้าสู่โปรแกรม.....	88
6.2 Customer access control.....	89
6.3 การ Generate Invoice.....	91
6.4 การ Cancel Invoice.....	93
6.5 การ Generate Receipt.....	96
6.6 การ Cancel Receipt.....	100
6.7 การ Generate Credit Note.....	101
6.8 การ Cancel Credit Note.....	102
6.9 การ Generate Debit Note.....	103
6.10 การ Cancel Debit Note.....	105
บรรณานุกรม.....	107
ภาคผนวก ก ตัวอย่างรายงาน.....	108
ประวัติผู้เขียน.....	109

สารบัญตาราง

	หน้า
ตารางที่	
1.1 แผนการดำเนินงาน.....	2
3.1 แอทธิบิต์ของคลาส PROMOTION.....	50
3.2 การดำเนินการของคลาส PROMOTION.....	50
3.3 แอทธิบิต์ของคลาส PROMOTION_PRODUCT.....	51
3.4 การดำเนินการของคลาส PROMOTION_PRODUCT.....	51
3.5 แอทธิบิต์ของคลาส PROMOTION_SERVICE.....	51
3.6 การดำเนินการของคลาส PROMOTION_SERVICE.....	52
3.7 แอทธิบิต์ของคลาส PRODUCT.....	52
3.8 การดำเนินการของคลาส PRODUCT.....	52
3.9 แอทธิบิต์ของคลาส SERVICE_PRICE.....	53
3.10 การดำเนินการของคลาส SERVICE_PRICE.....	53
3.11 แอทธิบิต์ของคลาส SERVICE_TYPE.....	53
3.12 การดำเนินการของคลาส SERVICE_TYPE.....	54
3.13 แอทธิบิต์ของคลาส PERSON.....	54
3.14 แอทธิบิต์ของคลาส EMPLOYEE.....	54
3.15 การดำเนินการของคลาส EMPLOYEE.....	55
3.16 แอทธิบิต์ของคลาส CUSTOMER.....	55
3.17 แอทธิบิต์ของคลาส CUSTOMER_SERVICE_EXPENSE.....	55
3.18 การดำเนินการของคลาส CUSTOMER_SERVICE_EXPENSE.....	56
3.19 แอทธิบิต์ของคลาส FORM.....	56
3.20 แอทธิบิต์ของคลาส FORM_DETAIL.....	56
3.21 แอทธิบิต์ของคลาส INVOICE.....	57

สารบัญตาราง (ต่อ)

ตารางที่	หน้า
3.22 การดำเนินการของคลาส INVOICE.....	58
3.23 แอทริบิวต์ของคลาส INVOICE_DETAIL.....	58
3.24 การดำเนินการของคลาส INVOICE_DETAIL.....	58
3.25 แอทริบิวต์ของคลาส RECEIPT.....	59
3.26 การดำเนินการของคลาส RECEIPT.....	60
3.27 แอทริบิวต์ของคลาส RECEIPT_DETAIL.....	60
3.28 การดำเนินการของคลาส RECEIPT_DETAIL.....	61
3.29 แอทริบิวต์ของคลาส BANK.....	61
3.30 การดำเนินการของคลาส BANK.....	61
3.31 แอทริบิวต์ของคลาส BRANCH_BANK.....	62
3.32 การดำเนินการของคลาส BRANCH_BANK.....	62
3.33 แอทริบิวต์ของคลาส APPLICATION.....	63
3.34 การดำเนินการของคลาส APPLICATION.....	63
3.35 แอทริบิวต์ของคลาส ADDRESS.....	63
3.36 การดำเนินการของคลาส ADDRESS.....	64
3.37 แอทริบิวต์ของคลาส DEBIT_NOTE.....	64
3.38 การดำเนินการของคลาส DEBIT_NOTE.....	65
3.39 แอทริบิวต์ของคลาส DEBIT_NOTE_DETAIL.....	65
3.40 การดำเนินการของคลาส DEBIT_NOTE_DETAIL.....	65
3.41 แอทริบิวต์ของคลาส CREDIT_NOTE.....	66
3.42 การดำเนินการของคลาส CREDIT_NOTE.....	66

สารบัญตาราง (ต่อ)

ตารางที่		หน้า
3.43	การดำเนินการของคลาส DB.....	70
4.1	Project DDL Generation Properties.....	80
4.2	Property DDL Generation Properties.....	82



สารบัญภาพ

รูปที่	หน้า
2.1 View ของ UML.....	3
2.2 Actor	5
2.3 Use Case.....	5
2.4 สัญลักษณ์แสดงขอบเขตของระบบ.....	5
2.5 ความสัมพันธ์ระหว่าง Use Case กับ Use Case แบบ <<uses>>.....	6
2.6 ความสัมพันธ์ระหว่าง Use Case กับ Use Case แบบ <<extends>>.....	7
2.7 ความสัมพันธ์ระหว่าง Actor กับ Actor.....	7
2.8 ความสัมพันธ์ระหว่าง Actor กับ Use Case.....	8
2.9 ตัวอย่างของ Use Case Diagram.....	8
2.10 Class.....	9
2.11 ความสัมพันธ์แบบ Generalization.....	9
2.12 ความสัมพันธ์แบบ Association.....	10
2.13 ความสัมพันธ์แบบ Aggregation.....	10
2.14 ตัวอย่าง Class Diagram.....	11
2.15 Object Diagram.....	11
2.16 State Diagram.....	12
2.17 Sequence Diagram.....	13
2.18 Collaboration Diagram.....	14
2.19 Activity Diagram.....	14
2.20 Component Diagram.....	15
2.21 Deployment Diagram.....	15
2.22 ตัวอย่างของคลาสและออบเจกต์ในระบบโรงพยาบาล.....	17
2.23 Message และ Methods.....	18

สารบัญภาพ (ต่อ)

รูปที่	หน้า
2.24 แสดงคุณสมบัติและประโยชน์ของ Message และ Encapsulation.....	19
2.25 แสดงตัวอย่างความสัมพันธ์ของคลาสต่างๆเป็นโครงสร้างแบบลำดับชั้น.....	21
2.26 เปรียบเทียบระหว่างการออกแบบคลาสที่ไม่ใช้ Inheritance และใช้ Inheritance	21
3.1 Use Case Diagram ระบบคิดเงินลูกค้า.....	28
3.2 กรณีลูกค้าติดตั้งใหม่.....	33
3.3 กรณีลูกค้าต้องการเปลี่ยนแปลงวงจร.....	33
3.4 กรณีลูกค้าต้องการยกเลิกการใช้งานวงจรชั่วคราว.....	34
3.5 กรณีลูกค้าต้องการเปลี่ยนค่าของวงจร.....	34
3.6 กรณีลูกค้าต้องการยกเลิกวงจร.....	35
3.7 กรณีลูกค้าต้องการขอใช้บริการต่อ.....	35
3.8 Sequence Diagram สำหรับ Use Case Login.....	36
3.9 Sequence Diagram สำหรับ Use Case Set Product Pricing.....	37
3.10 Sequence Diagram สำหรับ Use Case Set Service Pricing.....	38
3.11 Sequence Diagram สำหรับ Use Case Set Product Promotion.....	39
3.12 Sequence Diagram สำหรับ Use Case Set Service Promotion.....	40
3.13 Sequence Diagram สำหรับ Use Case Setup Promotion.....	41
3.14 Sequence Diagram สำหรับ Use Case Set Record Customer Data.....	42
3.15 Sequence Diagram สำหรับ Use Case Set Generate Service Invoice.....	43
3.16 Sequence Diagram สำหรับ Generate Service Receipt.....	44
3.17 Sequence Diagram สำหรับ Generate Credit Note.....	45
3.18 Sequence Diagram สำหรับ Use Case Generate Debit Note.....	46
3.19 Class Diagram สำหรับระบบคิดเงินลูกค้า.....	49
3.20 Class DB.....	49
3.21 Class Diagram สำหรับส่วนติดต่อกับผู้ใช้งาน.....	71

สารบัญภาพ (ต่อ)

รูปที่	หน้า
3.22 State Diagram สำหรับออบเจ็ก INVOICE.....	72
3.23 State Diagram สำหรับออบเจ็ก RECEIPT.....	73
3.24 State Diagram สำหรับออบเจ็ก CREDIT_NOTE.....	73
3.25 State Diagram สำหรับออบเจ็ก DEBIT_NOTE.....	74
3.26 Deployment Diagram ระบบคิดเงินลูกค้า.....	74
4.1 หน้าจอ Class Specification.....	77
4.2 หน้าจอ Options.....	78
4.3 Class Attribute Specification.....	78
4.4 Project Proties.....	79
4.5 Attribute Properties.....	81
4.6 DDL Setup ก่อนการ generate code.....	82
4.7 หน้าจอแสดงผลการ generate DDL.....	83
4.8 หน้าจอผลลัพธ์จากการทำ DDL Generate Code.....	83
4.9 Activity Diagram สำหรับการดำเนินการ AddInvoice.....	85
4.10 Activity Diagram สำหรับการดำเนินการ AddDebitNote.....	85
4.11 Activity Diagram สำหรับการดำเนินการ AddCreditNote.....	86
4.12 Activity Diagram สำหรับการดำเนินการ AddDebitNote.....	86
6.1 หน้าจอ Login.....	88
6.2 หน้าจอแรกของระบบ.....	89
6.3 หน้าจอ Customer Account Control.....	90
6.4 หน้าจอ Generate Invoice.....	91
6.5 หน้าจอขึ้นชั้นการ Generate Invoice.....	92
6.6 หน้าจอ Generate Invoice.....	92
6.7 หน้าจอหลังจาก Generate Invoice.....	93

สารบัญภาพ (ต่อ)

รูปที่	หน้า
6.8 หน้าจอ View & Cancel Invoice.....	94
6.9 หน้าจอ Relate Document ในการทำ View & Cancel Invoice.....	95
6.10 หน้าจอ Reson of Canceling เมื่อทำการ Cancel Invoice.....	95
6.11 หน้าจอ บอกสถานะการทำการ Cancel Invoice.....	96
6.12 หน้าจอ Bank Description.....	96
6.13 หน้าจอ Generate Receipt.....	98
6.14 หน้าจอ Select Invoice.....	98
6.15 หน้าจอ Confirm Generate Receipt.....	99
6.16 หน้าจอแสดงผลการ Generate Receipt.....	100
6.17 หน้าจอ View & Cancel Receipt.....	100
6.18 หน้าจอ Reson of Canceling Receipt.....	101
6.19 หน้าจอแสดงผลการ Cancel Receipt.....	101
6.20 หน้าจอ Generate Credit Note.....	102
6.21 หน้าจอ View & Cancel Credit Note.....	103
6.22 หน้าจอ Generate Debit Note.....	104
6.23 หน้าจอ ยืนยันการ Generate Debit Note.....	104
6.24 หน้าจอแสดงผลการ Generate Complete.....	105
6.25 หน้าจอ View & Cancel Debit Note.....	105
6.26 หน้าจอ Reson of Canceling.....	106
6.27 หน้าจอแสดงผลการ Cancel Debit Note.....	106

บทที่ 1

บทนำ

บริษัท UBT จำกัดดำเนินธุรกิจด้านการให้บริการสื่อสารด้วยระบบความเร็วสูง ADSL โดยมีการขายสินค้าและบริการให้กับลูกค้า โดยสินค้านั้นประกอบด้วยอุปกรณ์ประกอบต่างๆของวงจร ADSL และในส่วนของการบริการเป็นการให้บริการขายบริการวงจรเชื่อมโยงสู่อินเทอร์เน็ตโดยมีการแบ่งประเภทของลูกค้าออกเป็น 3 ประเภทซึ่งมีลักษณะการให้บริการที่ต่างกัน

ระบบการการคิดเงินลูกค้าของบริษัท UBT จำกัดซึ่งใช้ในการออกใบแจ้งหนี้เพื่อเรียกเก็บเงินจากลูกค้าที่ได้ซื้อสินค้าและบริการจากบริษัทซึ่งในระบบนี้จะมีการออกใบแจ้งหนี้แก่ลูกค้า (Invoice) ใบเสร็จรับเงิน (Receipt) ใบลดหนี้ (Credit Note) ใบเพิ่มหนี้ (Debit Note) และจดหมายเตือน (Notice) เมื่อลูกค้าไม่ได้มีการชำระเงินตามกำหนด

1.1 วัตถุประสงค์

1. เพื่อให้การเรียกเก็บเงินจากลูกค้าเป็นไปได้อย่างถูกต้อง
2. ช่วยลดภาระของหน่วยงานที่ต้องคิดเงินและเรียกเก็บเงินจากลูกค้า
3. สร้างความพึงพอใจและความน่าเชื่อถือให้กับลูกค้า

1.2 แผนการดำเนินงาน

1. ศึกษาความต้องการของผู้ใช้ระบบ
2. วิเคราะห์และออกแบบระบบสแตติกโมเดล และไดนามิกโมเดล โดยใช้ Unified Modeling Language (UML)
3. พัฒนาโปรแกรมตามระบบที่ได้ออกแบบไว้ ซึ่งมีกำหนดการดังแสดงในตารางที่ 1.1

ตารางที่ 1.1 ตารางแผนการดำเนินงาน

ขั้นตอนการดำเนินงาน	ต.ค.	พ.ย	ธ.ค	ม.ค.	ก.พ.	มี.ค.
1. ศึกษาทฤษฎี UML						
2. วิเคราะห์ความต้องการของระบบ						
3. วิเคราะห์และออกแบบจำลอง						
4. ออกแบบฐานข้อมูล						
5. พัฒนาระบบ						

1.1 ประโยชน์ที่คาดว่าจะได้รับ

- 1. การคิดเงินลูกค้าเป็นไปอย่างถูกต้องและรวดเร็ว
- 2. ลดภาระการทำงานของเจ้าหน้าที่เกี่ยวข้องกับระบบ
- 3. ช่วยให้บริษัทสามารถตรวจสอบรายรับได้ถูกต้องมากขึ้น

1.2 ขอบเขตของระบบ

- 1. ออกใบเรียกเก็บเงินลูกค้า (Service Invoice)
- 2. ออกใบเสร็จรับเงินให้แก่ลูกค้า (Invoice)
- 3. ออกใบลดหนี้ (Credit Note)
- 4. ออกใบเพิ่มหนี้ (Debit Note)

บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

2.1 ความรู้พื้นฐานเกี่ยวกับ UML

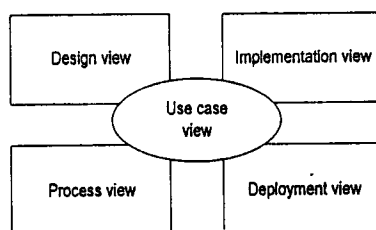
UML เป็นภาษาสำหรับบรรยายแบบจำลองของซอฟต์แวร์ (Modeling Language) โดยเป็นแนวคิดเชิงวัตถุโดยเป็นการรวมวิธีการของ Grady Booch (Booch method) James Rumbaugh (OMT-2 : Object Modeling Technique) และ Iva Jacobson (OOSE/Objectory method)

ความสัมพันธ์ของสถาปัตยกรรมของระบบกับ UML

UML กำหนดสถาปัตยกรรมของระบบโดยพิจารณาจากมุมมองคือ Use case View, Design View, Implement View, Process View และ Deployment View โดยแต่ละมุมมองประกอบด้วย Diagram ต่างๆซึ่งแสดงแบบจำลองผ่านมุมมองของระบบนั้นๆ ซึ่งรายละเอียดของแต่ละมุมมองมีดังต่อไปนี้

1. Use Case View เป็นมุมมองที่แสดงพฤติกรรมของระบบโดยพิจารณาจากมุมมองของผู้ใช้ (User) ที่อยู่นอกระบบ Use Case View ประกอบด้วย Use Case Diagram ซึ่งอธิบายหน้าที่ของระบบและองค์ประกอบต่างๆที่เกี่ยวข้อง ซึ่งเป็นมุมมองของผู้ใช้, ผู้วิเคราะห์ระบบ และผู้มีหน้าที่ทดสอบ

2. Design View เป็นมุมมองที่แสดงการออกแบบภายในระบบเพื่อให้ได้หน้าที่ของระบบตามความต้องการของผู้ใช้ โดยประกอบไปด้วย Class Diagram และ Object Diagram เพื่อใช้อธิบายโครงสร้างของระบบ และ State Diagram, Sequence Diagram, Collaboration Diagram และ Activity Diagram ต่างๆ ซึ่งใช้อธิบายการทำงานร่วมกัน โดยบุคคลที่มีส่วนเกี่ยวข้องกับ Design View ได้แก่ ผู้ออกแบบ และผู้พัฒนาระบบ



รูปที่ 2.1 View ของ UML

3. Process View เป็นมุมมองของระบบซึ่งใช้ดูการทำงานร่วมกันของ Process ต่างๆเพื่อดูประสิทธิภาพของระบบ โดยประกอบไปด้วย Diagram ต่างๆ คล้ายๆกับ Design View แต่จะสนใจที่ threads และ Process ของ Active Class

4. Implementation View เป็นมุมมองที่มองระบบที่มีการนำ Component และ file ซึ่งเป็นส่วนประกอบของระบบ มาประกอบร่วมกัน ดัดแปลงค่าต่างๆ เพื่อให้ระบบสามารถทำงานได้โดยประกอบไปด้วย Component Diagram, Interaction diagram, State Diagram และ Activity Diagram

5. Deployment View เป็นมุมมองที่แสดงสถาปัตยกรรมทางกายภาพของระบบ ประกอบด้วย Deployment Diagram ซึ่งอธิบายโครงสร้างและความสัมพันธ์ทางกายภาพขององค์ประกอบภายในระบบ (คอมพิวเตอร์ และอุปกรณ์อื่นๆ) บุคคลที่มีส่วนเกี่ยวข้องกับ Deployment View ได้แก่ ผู้พัฒนา และผู้ทดสอบระบบ

UML Diagram

UML Diagram คือ แบบจำลองซึ่งแสดงเป็นรูปภาพ เพื่อใช้ในการบรรยายแบบจำลอง ซึ่งมีทั้งหมด 9 ไคอะแกรม คือ

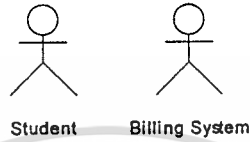
1. Use Case Diagram
2. Class Diagram
3. Object Diagram
4. State Diagram
5. Sequence Diagram
6. Collaboration Diagram
7. Activity Diagram
8. Component Diagram
9. Deployment Diagram

ในแต่ละไคอะแกรมมีรายละเอียดดังต่อไปนี้

2.1.1 Use-Case Diagram

Use-Case Diagram เป็นแบบจำลองซึ่งใช้อธิบายกิจกรรมของระบบ โดยมีการนำเสนอในมุมมองรูปภาพแสดงความสัมพันธ์ระหว่างสิ่งที่อยู่ภายนอกกับสิ่งที่อยู่ภายในระบบ โดย Use-Case Diagram ประกอบไปด้วย

1. Actor จะใช้สัญลักษณ์เป็นรูปคน (Stickman) โดยมีชื่อของ Actor อยู่ใต้รูปดังแสดงในรูปที่ 2 โดยที่ Actor จะแสดงบทบาท (Role) ที่ผู้ใช้หรือระบบย่อย ที่อยู่นอกระบบแสดงต่อระบบที่จะสร้าง มีข้อสังเกต คือ คำว่า “Actor” กับ “User” นั้นไม่เหมือนกัน สำหรับ User คนหนึ่งอาจแสดงได้หลายบทบาท เช่นผู้ดูแลระบบอาจเป็นได้ทั้ง User ที่เป็นคน หรือฮาร์ดแวร์ หรือระบบย่อยอื่น



รูปที่ 2.2 Actor

2. Use Case คือฟังก์ชันหลักของระบบที่มองจากมุมมองของผู้ใช้ โดย Use Case จะทำการโต้ตอบกับผู้ใช้ (Actors) ของระบบ แทนด้วยสัญลักษณ์รูปวงรี จะแสดงกิจกรรมหนึ่งๆที่เกิดขึ้นในระบบนั้น



รูปที่ 2.3 Use Case

3. Boundary ขอบเขตของระบบมีลักษณะเป็นเส้นที่ตีกรอบแสดงขอบเขตของระบบ เพื่อให้เห็นชัดเจนว่าส่วนไหนอยู่ภายในระบบ และส่วนไหนอยู่นอกระบบ



รูปที่ 2.4 สัญลักษณ์แสดงขอบเขตของระบบ

4. Relationship ความสัมพันธ์ภายใน Use Case Diagram

Bi-directional Association เป็นเส้นแสดงความสัมพันธ์ หรือการสื่อสารระหว่าง Actor กับ Use Case หรือระหว่าง Use Case กับ Use Case โดยสัญลักษณ์ที่ต้นลูกศรจะเป็นผู้เริ่มต้นการสื่อสารนั้นก่อน

Extend เป็นเส้นแสดงความสัมพันธ์ระหว่าง Use Case ทำหน้าที่ของส่วนยกเว้นหรือส่วนขยายขอบข่ายการทำงานให้มากกว่า Use Case พื้นที่

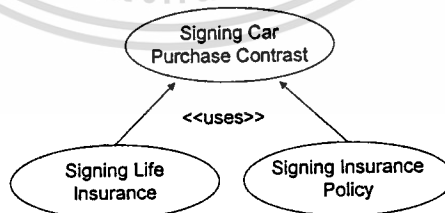
Include Use Case แสดงความสัมพันธ์จาก Use Case พื้นฐานไปยังที่เป็นส่วนประกอบอยู่ใน Use Case พื้นฐานนั้น

Generalization เส้นแสดงความสัมพันธ์ระหว่าง Use Case ที่เฉพาะเจาะจงกับ Use Case ที่เป็นทั่วไป

Uni-directional Association เป็นเส้นแสดงความสัมพันธ์หรือการสื่อสารระหว่าง Actor กับ Use Case หรือระหว่าง Use Case กับ Use Case โดยสัญลักษณ์ที่ต้นลูกศรจะเป็นผู้เริ่มต้นการสื่อสารนั้นก่อน

ซึ่งมีความสัมพันธ์ 3 แบบ คือ

1. ความสัมพันธ์ระหว่าง Use Case กับ Use Case มีด้วยกัน 3 ลักษณะ คือ Use , Extended และ Grouping ซึ่งเป็นความสัมพันธ์แบบ Generalization ความสัมพันธ์แบบ Uses จะแทนด้วย หัวลูกศรที่ชี้ไปยัง Use Case ที่จะใช้ พร้อมกับมีคำอธิบายว่า <<Uses>> ซึ่งหมายถึงว่า Use Case นั้นเป็นส่วนหนึ่งของ Use Case ที่ใช้งานดังรูปข้างล่าง

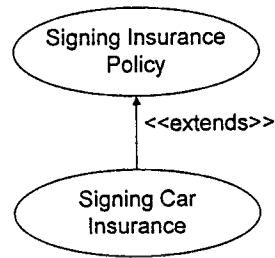


รูปที่ 2.5 ความสัมพันธ์ระหว่าง Use Case กับ Use Case แบบ <<uses>>

ส่วนความสัมพันธ์ในลักษณะ Extended เป็นความสัมพันธ์ระหว่าง Use Case ดันแบบ กับ Use Case ซึ่งขอใช้บริการโดยที่ Use Case ซึ่งขอใช้บริการจาก Use Case ดันแบบจะทำหน้าที่ต่าง ซึ่งใน Use Case ดันแบบมีแล้วยังเพิ่มเติมหน้าที่ของตัวเองเพิ่ม ต่อจาก Use case ดันแบบเข้าไปอีก

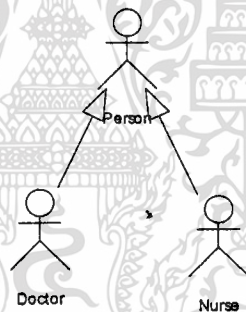
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดังตัวอย่างในรูปที่ 5 Signing Insurance ก็อาจจะมีหน้าที่ให้บริการเพิ่มเติมจาก Signing Insurance Policy โดยการให้บริการตรวจเช็คสภาพรถยนต์ให้ฟรีเพิ่มขึ้นมา



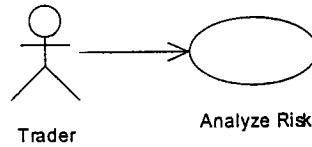
รูปที่ 2.6 ความสัมพันธ์ระหว่าง Use Case กับ Use Caseแบบ <<extends>>

2. ความสัมพันธ์ระหว่าง Actor กับ Actor เป็นความสัมพันธ์แบบ Generalization จะแทนด้วยลูกศรหัวทึบชี้ไปยัง Actor ที่เป็นซูเปอร์คลาส (Super Class)



รูปที่ 2.7 ความสัมพันธ์ระหว่าง Actor กับ Actor

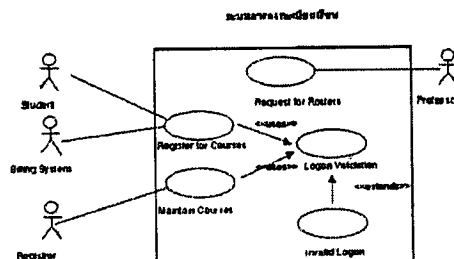
ความสัมพันธ์ระหว่าง Actor กับ Use Case มีเพียงความสัมพันธ์แบบ Association ซึ่งแทนด้วยลูกศรหัวเปิดซึ่งหางของลูกศรคือผู้ที่เริ่มการติดต่อ หรือไม่มีหัวลูกศรในกรณีที่ทั้ง Actor และ Use Case มีความสัมพันธ์แบบ 2 ทาง



รูปที่ 2.8 ความสัมพันธ์ระหว่าง Actor กับ Use Case

ตัวอย่างการใช้ Use Case Diagram

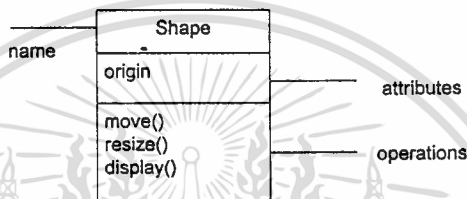
จากรูป จะเห็นได้ว่า Actor ที่ติดต่อกับระบบการลงทะเบียนเรียนนี้มี อยู่ 4 Actor และมี 5 กิจกรรม อยู่ในระบบ ยกตัวอย่าง เช่น Actor ที่แสดงบทบาทของ Student ที่ต้องการลงทะเบียน ก็มีความสัมพันธ์กับกระบวนการลงทะเบียนเรียนวิชาต่างๆ (Register for Courses) และมีอีก Actor หนึ่งที่เกี่ยวข้องกับ กิจกรรม นี้คือ ระบบการออกใบเสร็จ (Billing System) เป็นระบบภายนอก ซึ่งอาจมีขั้นตอนซับซ้อนเช่นกัน แต่ถือว่าไม่อยู่ในระบบการลงทะเบียน ส่วน Actor ที่แสดงบทบาทเจ้าหน้าที่ทะเบียน (Registrar) ก็มีหน้าที่จัดสรรวิชาต่างๆที่เปิดสอน เพื่อให้นักเรียนมาลงทะเบียน ซึ่ง Transaction นี้คือ Maintain Courses และ Actor สุดท้าย คือ Professor ก็แสดงความต้องการบัญชีรายชื่อนักเรียน ที่ลงทะเบียนวิชาที่ตนเองรับผิดชอบอยู่ ซึ่ง Professor ก็จะเกี่ยวข้องกับการ Request for Rosters สิ่งที่น่าสนใจคือ ความสัมพันธ์ระหว่าง Transaction แบบ <<uses>> และ <<extends>> ทั้งสองความสัมพันธ์นี้ ใช้ในการเชื่อมโยงระหว่าง Use Case ที่มีความสัมพันธ์กัน สำหรับ <<extends>> นั้น ใช้กรณีเฉพาะ หรือทำอะไรเพิ่มเติมจาก Use Case ที่ถูกขยาย หรือพูดง่ายๆว่าเป็น Option จะทำหรือไม่ทำก็ได้ ในตัวอย่างนี้ เมื่อมีการทำกิจกรรม Valid Logon แล้ว กิจกรรม Invalid Logon จะทำ หรืออาจจะไม่ทำก็ได้ ถ้าทำ อาจเป็นการเพิ่มเติมส่วนแสดงข้อความ Error ขึ้นมาที่หน้าจอ สำหรับกรณี Invalid Logon เป็นต้นในกรณีของ <<uses>> นั้นใช้ในกรณีที่ Use Case หลายอัน มีการทำงานบางอย่างที่เหมือนกันอยู่ จึงแยกการทำงานบางส่วนที่เหมือนกันนั้นออกมา ดังนั้น กรณี <<uses>> จะต้องทำ Use Case ที่ <<uses>> ออกมาเสมอ



รูปที่ 2.9 ตัวอย่างของ Use Case Diagram

2.1.2 Class Diagram

เป็น Diagram ที่สำคัญซึ่งแสดงถึงองค์ประกอบหรือความต้องการของระบบที่เราพัฒนา ประกอบด้วย Class และความสัมพันธ์กันระหว่าง Classes (Relationship) ในขณะที่ Relationship แสดงความสัมพันธ์แบบต่างๆ ระหว่าง Class ซึ่งได้แก่ Generalization, Association และ Dependency และ Inheritance



รูปที่ 2.10 Class

ส่วนประกอบของ Class

1. Names แสดงชื่อ Class ภายในระบบ
2. Attributes อธิบายถึงโครงสร้างและลักษณะทั้งหมดที่มีใน Class
3. Operations เป็นส่วนที่ใช้อธิบายถึงลักษณะการทำงาน (Method) หรือ พฤติกรรมของ (Behavior) ของ Class

ความสัมพันธ์ระหว่าง Class (Relationships)

ประกอบไปด้วย

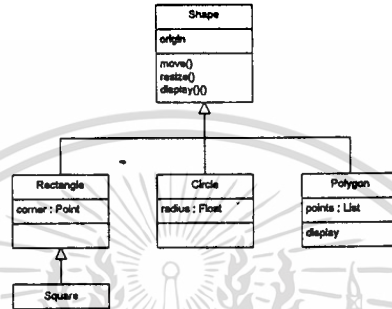
1. Dependency

เป็นความสัมพันธ์ในลักษณะที่มีการขึ้นตรงต่อกัน คือเมื่อมีการแก้ไข Class หนึ่งจะมีผลกับอีก Class หนึ่ง ในลักษณะที่เมื่อมีการเปลี่ยนแปลงของ Class ที่เป็นต้นแบบแล้วจะทำให้เกิดการเปลี่ยนแปลงกับ Class ที่ไปขึ้นอยู่ กับ Class ต้นแบบ สัญลักษณจะเป็นลูกศรรูปปะซึ่งหัวลูกศรชี้ไปยัง Class ซึ่งมีการขึ้นกับอีก Class หนึ่งซึ่งอยู่ตรงปลายลูกศร

2. Generalization

เป็นความสัมพันธ์ระหว่าง Sub Class และ Super Class ในลักษณะของการถ่ายทอดคุณสมบัติจาก Super Class ซึ่งเป็น Class ต้นแบบ มายัง Sub Class จากรูปที่ 9 จะเห็นว่าถ้าเรามีการ

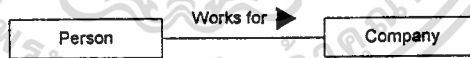
รวบรวม คุณสมบัติต่างๆที่มีการซ้ำกันของในแต่ละ Sub Class ย่อยๆเข้ามาเราจะได้ Class ต้นแบบ ขึ้นไปได้เรื่อยๆ เช่น Class Shape ก็คือ Super Class ที่จะมี Attribute และ Operation ต่างๆที่ Class Rectangle Circle และ Polygon มี หรือกล่าวในอีกทางหนึ่งว่า Class Rectangle Circle และ Polygon มีการสืบทอดไปจาก Class Shape



รูปที่ 2.11 ความสัมพันธ์แบบ Generalization

3. Association

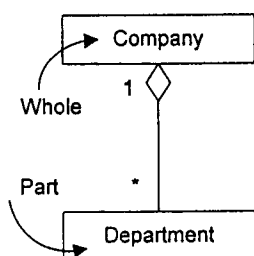
Name คือ ชื่อแสดงถึงความสัมพันธ์ระหว่าง Class และบอกทิศทางของความสัมพันธ์ด้วยเครื่องหมายสามเหลี่ยมทึบ



รูปที่ 2.12 ความสัมพันธ์แบบ Association

4. Aggregation

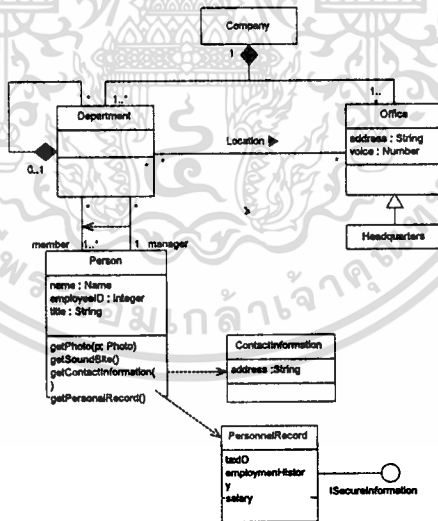
เป็นรูปแบบพิเศษของ Association คือเป็นความสัมพันธ์ที่แสดงว่า Class หนึ่งเป็นส่วนหนึ่งของอีก Class หนึ่ง



รูปที่ 2.13 ความสัมพันธ์แบบ Aggregation

จำนวนความสัมพันธ์ (Multiplicity)

แต่ละ Class จะมีความสัมพันธ์ระหว่างกัน โดยกำหนดเป็นตัวเลข หรือช่วงของตัวเลข โดยแสดงในรูปแบบค่าต่ำสุด..ค่าสูงสุดไว้ที่ปลายของเส้นความสัมพันธ์ ถ้าเป็นเลขตัวเดียวหมายถึงค่าที่เป็นได้ค่าเดียว แต่ถ้าเป็นช่วงจะใช้เครื่องหมายดอกจัน (*) หมายถึงสามารถมีค่าความสัมพันธ์ได้หลายค่าตามช่วงที่กำหนด เช่น 1..*



รูปที่ 2.14 ตัวอย่าง Class Diagram

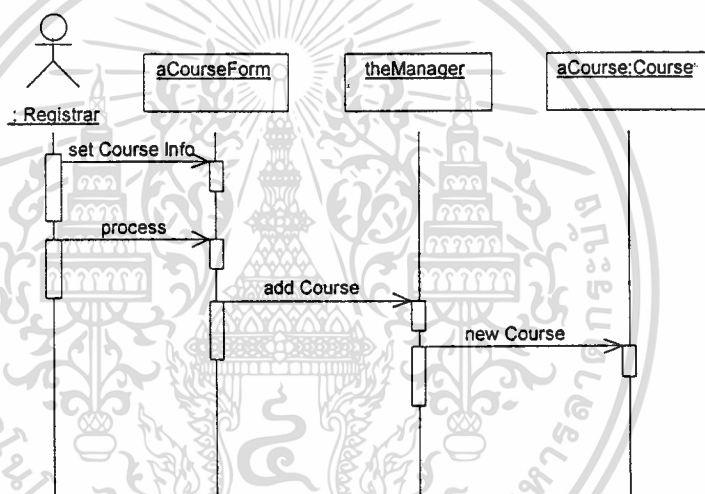
2.1.3 Object Diagram

Object Diagram ประกอบด้วย Object และ Relationship ระหว่าง Object โดยแต่ละ Object แสดง Instance ของแต่ละ Class ที่มีในระบบ ในขณะที่ Relationship แสดงความสัมพันธ์แบบต่างระหว่าง Object ซึ่งได้แก่ Generalization, Association และ Dependency

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.5 Sequence Diagram

เป็นการอธิบายคลาสที่สร้างขึ้น โดยอธิบายถึงความสัมพันธ์ของออบเจกต์ในมิติของเวลา กล่าวคือจะการเรียงลำดับออบเจกต์ตามแนวนอน และส่วนในแนวตั้งมีลำดับการส่งผ่านข้อความจากออบเจกต์หนึ่งไปยังออบเจกต์หนึ่ง ที่เกิดขึ้นเมื่อเวลาผ่านไป โดยมีหมายเลขกำกับกับการส่งผ่านข้อความเหล่านั้นด้วย ออบเจกต์ใน Sequence Diagram เป็นสัญลักษณ์สี่เหลี่ยมผืนผ้าแทนออบเจกต์หรือคลาส สัญลักษณ์เส้นประในแนวตั้งเรียกว่าเส้นชีวิตของออบเจกต์ (Object 's lifeline) เป็นการกำหนดขอบเขตการทำงานของออบเจกต์นั้น ตัวอย่างแสดงได้ดังรูปที่ 2.17

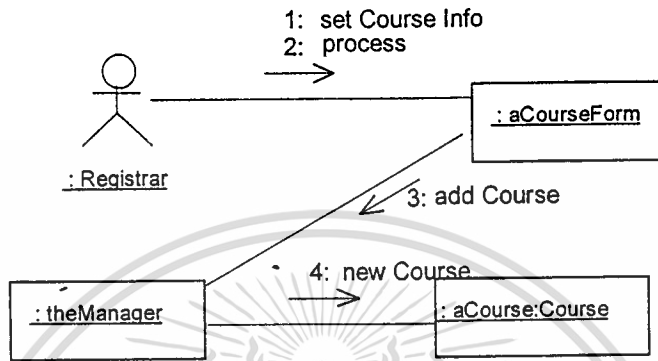


รูปที่ 2.17 Sequence Diagram

2.1.6 Collaboration Diagram

Collaboration Diagram เป็น Diagram ซึ่งแสดงการปฏิสัมพันธ์ (Interaction) ระหว่าง Object ตามลำดับของการปฏิสัมพันธ์และเหตุการณ์ต่างๆที่เกิดขึ้น ณ เวลาที่กำหนด เช่นเดียวกับ Sequence Diagram แต่ Collaboration Diagram จะเน้นหนักในการอธิบายบริบท (Context) ของระบบ สัญลักษณ์ที่ใช้เหมือนกับ Sequence Diagram ซึ่งสามารถเลือกแสดงได้ในรูปแบบใดแบบหนึ่ง กล่าวคือทั้งสองแผนผังสามารถเปลี่ยนแปลงรูปแบบระหว่างกันได้ ซึ่งสามารถเรียกแผนผังทั้งสองได้อีกชื่อว่า Interaction Diagram ดังเช่นตัวอย่างในรูปจะเป็นระบบลงทะเบียนของมหาวิทยาลัย ซึ่งแสดงลำดับของการลงทะเบียนเพิ่มข้อมูลวิชา ติดต่อเพื่อให้นักศึกษา ซึ่งนักศึกษาที่ต้องการจะทำการลง Course ตามลำดับ คือ Register หรือนักศึกษาจะต้องเข้ามาในขั้นตอนของการลงทะเบียน

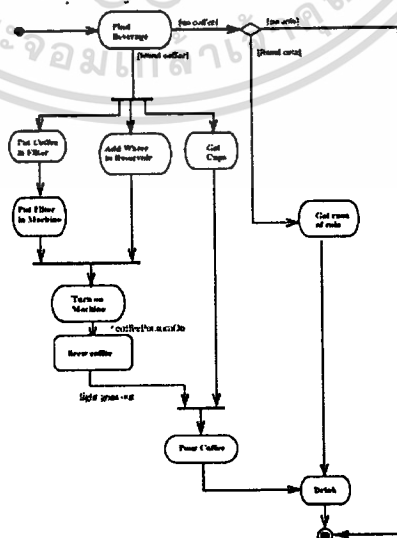
โดยการนำข้อมูลลงฟอร์มของ Course โดยการกรอกข้อมูลแล้ว ข้อมูลนั้นจะส่งไปถึงผู้จัดการเพื่ออนุมัติ หลังจากนั้นข้อมูลของ Course จะถูกส่งเข้าไปในฐานข้อมูล Course



รูปที่ 2.18 Collaboration Diagram

2.1.7 Activity Diagram

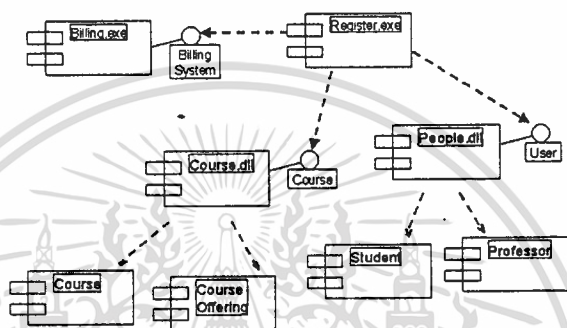
Activity Diagram เป็น Diagram ซึ่งแสดงขั้นตอนการทำงานในการปฏิบัติการ (Operation) โดยประกอบด้วยสถานะ (State) ต่างๆที่เกิดขึ้นระหว่างการทำงาน และผลจากการทำงานในขั้นตอนต่าง ๆ ที่เกิดขึ้นระหว่างการทำงาน และผลการทำงานในขั้นตอนต่าง ๆ



รูปที่ 2.19 Activity Diagram

2.1.8 Component Diagram

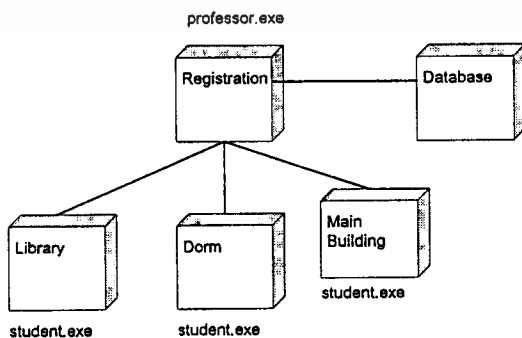
Component Diagram เป็น Diagram ซึ่งแสดงโครงสร้างทางกายภาพของซอฟต์แวร์ซึ่งอยู่ในรูปแบบต่างๆ (Text, Binary และ Executable) ดังรูปที่ 18 เป็นตัวอย่างของระบบลงทะเบียนของมหาวิทยาลัยซึ่งประกอบด้วยส่วนที่เป็น Text ได้แก่ Professor, Student, Course Offering, Course และส่วนที่เป็น Binary ได้แก่ Billing.exe, Register.exe, Course.dll และ People.dll



รูปที่ 2.20 Component Diagram

2.1.9 Deployment Diagram

เป็น Diagram ซึ่งแสดงสถาปัตยกรรมของฮาร์ดแวร์และซอฟต์แวร์ในระบบ ตลอดจนความสัมพันธ์ระหว่างฮาร์ดแวร์และซอฟต์แวร์ในระบบ ตลอดจนความสัมพันธ์ระหว่างฮาร์ดแวร์และซอฟต์แวร์ จากรูป เป็น Deployment Diagram ของระบบลงทะเบียนนักศึกษาของมหาวิทยาลัย ซึ่งจะมีการติดตั้งฮาร์ดแวร์กระจายไปยังอาคารต่าง ๆ



รูปที่ 2.21 Deployment Diagram

2.2 ความรู้พื้นฐานเกี่ยวกับเทคโนโลยีของออปเจกต์

2.2.1 หลักการพื้นฐานที่เกี่ยวกับออปเจกต์

วิธีการพัฒนาระบบสารสนเทศในอดีต นักพัฒนาจะพิจารณาวิเคราะห์ระบบในส่วนของการกระบวนการทำงานนั้นแยกจากกัน วิธีการ เครื่องมือ และโคดอะแกรมที่ถูกลำเอียงมาใช้ในการวิเคราะห์ก็มักจะเป็นวิธีการที่เน้นให้ผู้พัฒนาระบบทำการพิจารณาเนื้อหาของข้อมูล (Data Approach) ออกจากวิธีการที่ใช้วิเคราะห์กระบวนการวิธี (Process Approach) แต่หลังจากนั้นมาในช่วงกลางศตวรรษที่ 1980 นักพัฒนาก็เริ่มตระหนักถึงวิธีการพัฒนาระบบเชิงออปเจกต์ (Object Approach) ซึ่งแต่ละออปเจกต์ก็คือส่วนย่อยของโปรแกรมซึ่งภายในประกอบด้วยทั้งฟิลด์ของข้อมูลและฟังก์ชันพื้นฐานที่กระทำกับข้อมูลนั้นในที่นี้จะกล่าวถึงวิธีการเชิงออปเจกต์ในหลักการพื้นฐานสำคัญ ดังต่อไปนี้

1. หลักการของคลาสและออปเจกต์

คลาสหรือออปเจกต์ ก็คือองค์ประกอบของระบบที่มีความสมบูรณ์ในตัวเองโดยภายในประกอบด้วยทั้งข้อมูลและฟังก์ชันพื้นฐานที่ใช้ในการประมวลผลกับข้อมูลนั้น โดยที่ออปเจกต์อาจจะใช้เพื่อเป็นตัวแทนของบุคคล สิ่งของ สถานที่ หรือเหตุการณ์ที่เราต้องการทำการบันทึกข้อมูล เราอาจเปรียบเทียบคำว่า คลาสและออปเจกต์ได้ใกล้เคียงกับคำว่า “เอนทิที” (Entity) ในโคดอะแกรมชนิด “เอนทิที-รีเลชันชิพ” (Entity Relationship Diagram)

ตัวอย่างเช่น ถ้าเราต้องการพัฒนาระบบสารสนเทศสำหรับโรงพยาบาล คลาสและออปเจกต์ดังกล่าวอาจประกอบด้วย คลาส “นายแพทย์” (Doctor) “คนไข้” (Patient) และ “การนัดหมาย” (Appointment) แต่ถ้าเราพิจารณาถึงคนไข้คนใดคนหนึ่งเจาะจงไป เช่น คนไข้ที่ชื่อว่า “สมควร แซ่ลิ้ม” หรือ “สุเมธ ชัยกร” นั้น เราจะเรียกว่าเป็นออปเจกต์

ในแต่ละคลาสหรือออปเจกต์นั้นจะต้องมี “คุณสมบัติ” (Properties หรือ Attributes) หรือคือรายละเอียดของข้อมูล รายละเอียดต่างๆที่สนใจ สำหรับคลาสหรือออปเจกต์นั้น เช่น สำหรับออปเจกต์คนไข้ก็ควรจะมียุทธศาสตร์ของข้อมูล เช่น ชื่อ (Name), ที่อยู่ (Address), วันเดือนปีเกิด (Birthdate), และเบอร์โทรศัพท์ (Phone number) ค่าต่างๆของฟิลด์ของข้อมูลเหล่านี้จะแสดงสถานะของออปเจกต์นั้นๆ

ความแตกต่างระหว่างเอนทิทีและออปเจกต์ ก็คือออปเจกต์ นอกจากจะประกอบด้วยฟิลด์ของข้อมูลแล้ว ยังประกอบด้วย “พฤติกรรม” (Behavior) ซึ่งแสดงฟังก์ชันพื้นฐานที่กระทำหรือประมวลผลกับข้อมูลพื้นฐานภายในออปเจกต์นั้น ฟังก์ชันพื้นฐานนี้เรียกว่าเป็น “เมธอด” (Methods) หรือ “ออปอเรชัน” (Operations) ตัวอย่างเช่น ออปเจกต์การนัดหมายนั้นนอกจากจะมีข้อมูลเกี่ยวข้องกับการนัดหมายแล้วยังประกอบด้วยฟังก์ชันการ “เพิ่ม” (Insert) รายการนัดหมาย รายการใหม่ และ “พิมพ์” (print) การนัดหมายของนายแพทย์แต่ละท่าน เป็นต้น

โดยปกติแล้วภายในแต่ละออปเจ็กต์จะไม่มีฟิลด์ที่เป็น Primary key และ Foreign key เหมือนอย่างที่มีการใช้กันในฐานะข้อมูลแบบสัมพันธ์ แต่เทคโนโลยีของออปเจ็กต์จะใช้วิธีการกำหนดด้วย “รหัสเฉพาะ” หรือ “Unique Identifier” (UID) โดยระบบภายในจะกำหนดค่าดังกล่าวสำหรับแต่ละออปเจ็กต์โดยอัตโนมัติ และค่า UID นี้จะเป็นค่าที่ซ่อนไว้จากผู้ใช้งานและจะถูกนำมาใช้ในการอ้างอิง และแยกแยะความแตกต่างของแต่ละออปเจ็กต์ ดังนั้น เมื่อมีการสร้างออปเจ็กต์ขึ้นมาสองออปเจ็กต์ และถึงแม้ค่าภายในแต่ละออปเจ็กต์จะถูกกำหนดให้เหมือนกัน แต่ก็จะมี UID แตกต่างกันเพื่อใช้ในการอ้างอิงออปเจ็กต์ ที่แตกต่างกัน

คลาส (Class) ก็คือ กรอบในภาพรวมของออปเจ็กต์ที่ผู้พัฒนาระบบได้กำหนดขึ้นเพื่อจะเป็นกรอบรายละเอียดของออปเจ็กต์ที่จะถูกสร้างขึ้นมา ดังนั้น ออปเจ็กต์หนึ่งๆ ตัวอย่างเช่น ออปเจ็กต์ทั้งหลายที่เก็บข้อมูลคนไข้แต่ละคนจะถูกนิยามขึ้น หรือถูกเรียกโดยรวมว่าเป็น “คลาส” (Class) เนื่องจากออปเจ็กต์ต่างๆ เหล่านั้นมีฟิลด์ของข้อมูล (Attribute) เช่น name, birthDate, address และ phoneNumber และเมธอด (Methods) ที่ถูกกำหนดขึ้นตามนิยามของคลาส Patient นั้น ดังจากตัวอย่างในรูปที่ 2.22

คลาส Patient

Patient
name
birthDate
address
phoneNumber
- insert() - delete()

ออปเจ็กต์ Patient

a : Patient
name = "สมหญิง ใจดี"
birthDate = "26 มีนาคม 2510"
address = "50 ถนนงามวงศ์วาน นนทบุรี"
phoneNumber = "(662)541-5122"
- insert() - delete()

คลาส Appointment

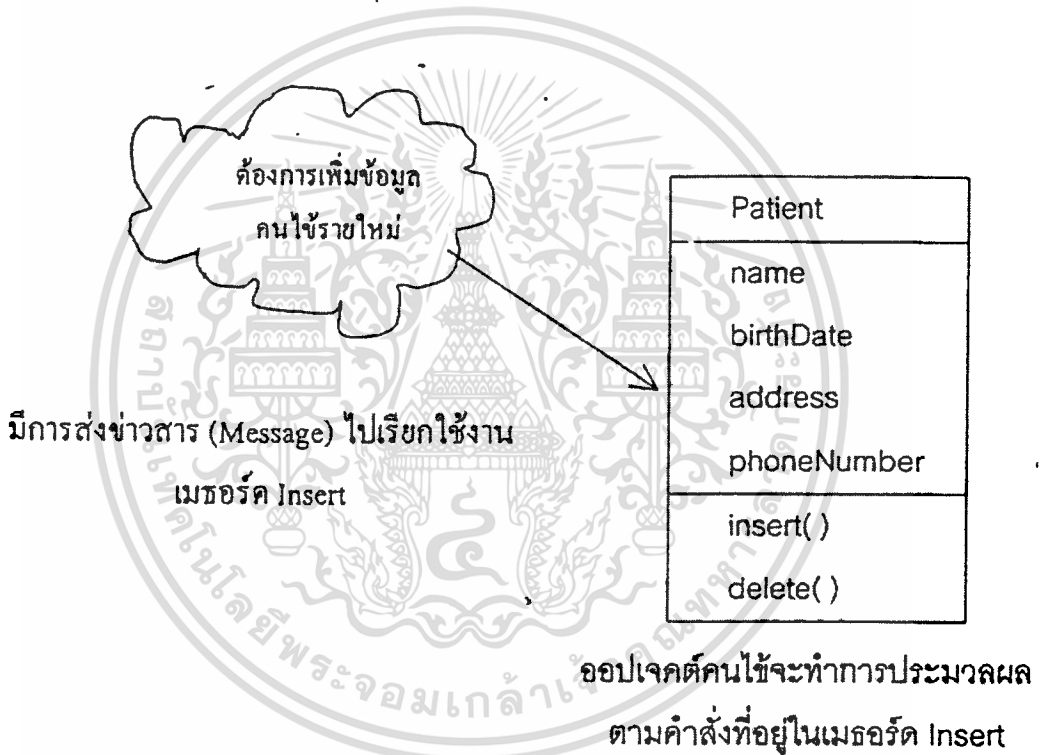
Appointment
patientName
doctorName
date
time
- insert() - delete() - print ()

ออปเจ็กต์ Appointment

: Appointment
patientName = "สมควร แซ่ลิ้ม"
doctorName = "สุชาติ เรืองรอง"
date = "20 ธันวาคม 2543"
time = "04:00 pm"
- insert() - delete() - print ()

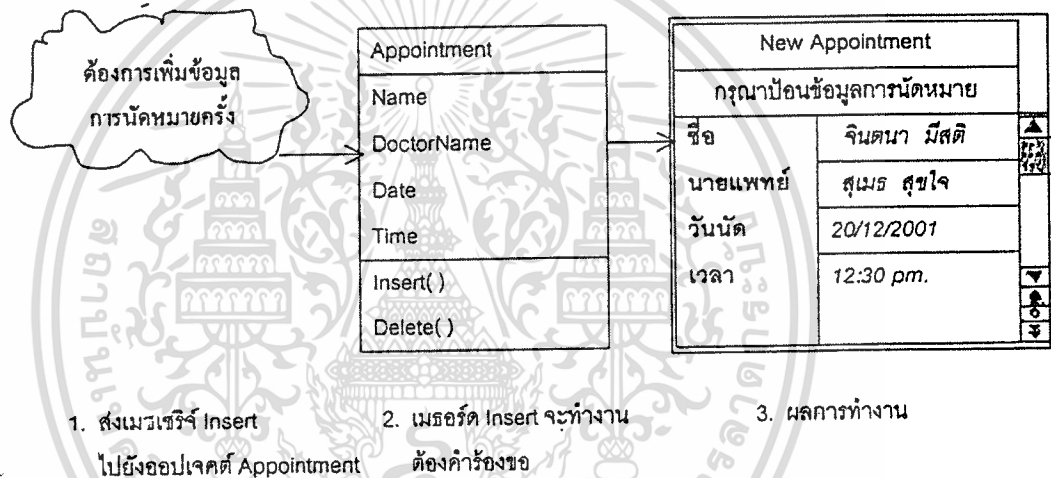
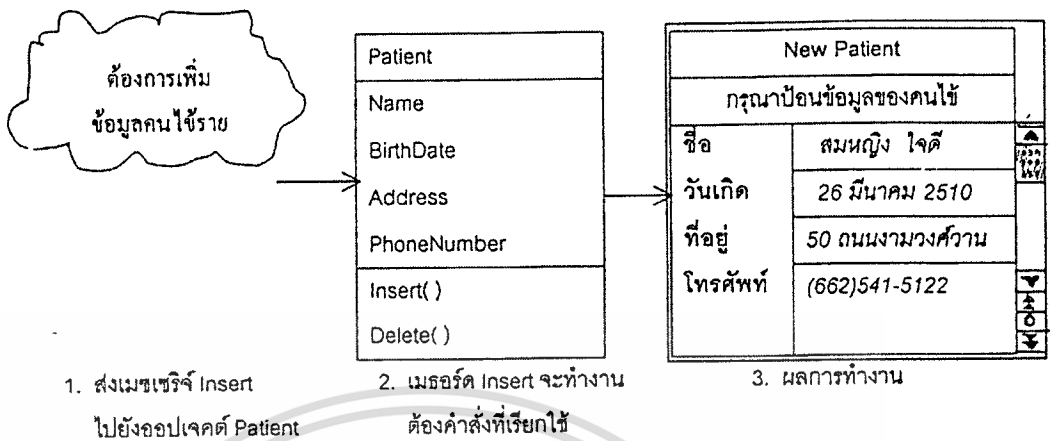
2.2.2 การสื่อสารระหว่างออปเจกต์

เมซเซจ (Message) หมายถึง การส่งข่าวสารไปยังออปเจกต์เพื่อให้ออปเจกต์ทำการประมวลผลตามเมธอดที่ถูกเรียกใช้งานนั้น เมธอดนี้คือฟังก์ชันหรือคำสั่งพื้นฐานที่จะกระทำการประมวลผลกับข้อมูลภายในออปเจกต์นั้น ตัวอย่างเช่น ถ้ามีคนไข้รายใหม่เข้ามาที่สำนักงานของแพทย์ ระบบจะทำการส่งข่าวสารเรียกใช้งานเมธอด Insert ของออปเจกต์คนไข้ ออปเจกต์คนไข้จะได้รับคำสั่งผ่านทางเมซเซจนี้แล้วจะดำเนินการประมวลผลต่างๆตามที่กำหนดไว้ในเมธอด Insert, ดูรูปที่ 2.23



รูปที่ 2.23 Message และ Methods

ผู้ใช้งานไม่จำเป็นต้องพะวงหรือไม่ต้องทราบรายละเอียดวิธีการที่อยู่ในเมธอด Insert ผู้ใช้เพียงแต่ทราบว่าเมธอดอะไรให้ใช้งานบ้างแล้วทำการส่งเมซเซจที่ต้องการไปยังออปเจกต์นั้นๆ ตัวอย่างเช่น ถ้าเราส่งเมซเซจที่ชื่อว่า “Draw Figure” (วาดรูปของตนเอง) ไปยังออปเจกต์ “Square” (สี่เหลี่ยมจตุรัส) “Circle” (วงกลม) และ “Triangle” (สามเหลี่ยม) ผลลัพธ์ของเมธอด “Draw Figure” จะให้ผลต่างกันขึ้นอยู่กับว่าเราเรียกใช้เมซเซจนั้นกับออปเจกต์ชนิดไหน



รูปที่ 2.24 แสดงคุณสมบัติและประโยชน์ของ Message และ Encapsulation

Encapsulation การส่งเมสเซจจ์ระหว่างออบเจกต์ แล้วออบเจกต์ผู้รับจะดำเนินการประมวลผลตามคำสั่งที่กำหนดไว้ภายใน แนวทางการสื่อสารระหว่างออบเจกต์ในลักษณะนี้ ทำให้เราสามารถใช้งานออบเจกต์ได้ในรูปแบบของกล่องดำ (Black Box) กล่าวคือ ผู้ใช้เมธอร์ดของออบเจกต์นั้นไม่จำเป็นต้องเข้าใจว่าเมธอร์ดนั้นทำงานในรายละเอียดอย่างไร แนวคิดในการซ่อนรายละเอียดของข้อมูลและวิธีการประมวลผลภายในของแต่ละเมธอร์ดในออบเจกต์โดยที่ผู้ใช้หรือออบเจกต์ภายนอกไม่จำเป็นต้องรู้รายละเอียดเรียกว่า “Encapsulation”

ความจริงที่ว่าเราสามารถใช้งานออบเจกต์ได้โดยการเรียกใช้ฟังก์ชันต่างๆ ที่ต้องการนี้เป็นหัวใจสำคัญประการหนึ่งใน “การนำกลับมาใช้ใหม่” (Resuability) เนื่องจากการซ่อนรายละเอียดจากโปรแกรมภายนอกได้ และเมื่อมีการเปลี่ยนแปลงใดๆ ภายในออบเจกต์ ก็จะไม่กระทบการเรียกใช้งานจากภายนอกได้ ดังแสดงในรูปที่ 2.23 สังเกตว่าผู้เรียกใช้งานเมสเซจจ์ Insert ได้ โดยไม่

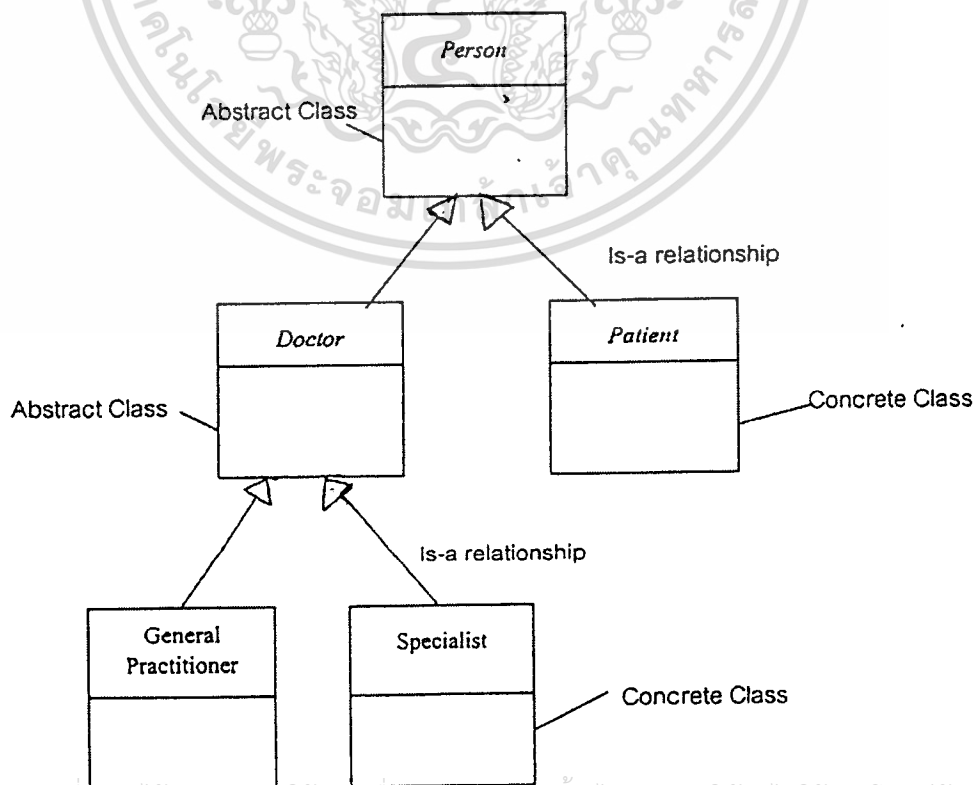
เือกสาร... ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต้องไปพ่วงถึงขั้นตอนวิธีในรายละเอียดของเมธอร์คี่นั้นๆ สิ่งที่เราเขียนโปรแกรมจำเป็นจะต้องรู้ในการเรียกใช้ออปเจ็คก็แค่มีแต่ทำความเข้าใจว่าแต่ละออปเจ็คมีชื่อว่าอะไร และมีเมธอร์คี่อะไร และมีเมธอร์คี่อะไร มีการส่งผ่านค่า และเอาต์พุตอะไรที่จะเกิดขึ้นหลังจากการเรียกใช้ เมธอร์คี่นั้นๆ

2.2.3 การสืบทอดมรดก (Inheritance)

โดยปกติแล้วคลาสต่างๆ ในระบบจะถูกจัดระเบียบและโยงความสัมพันธ์ในลักษณะลำดับชั้น (Hierarchy) โดยที่ซูเปอร์คลาส (Super Class) หรือคลาสที่กล่าวครอบคลุมแบบทั่วไป (General Class หรือ Generalization) จะอยู่ด้านบนของโครงสร้าง ส่วนซับคลาส (Sub Class) หรือคลาสที่กล่าวถึงแบบเจาะจง (Specific Class หรือ Specialization) จะอยู่ด้านล่างของโครงสร้าง

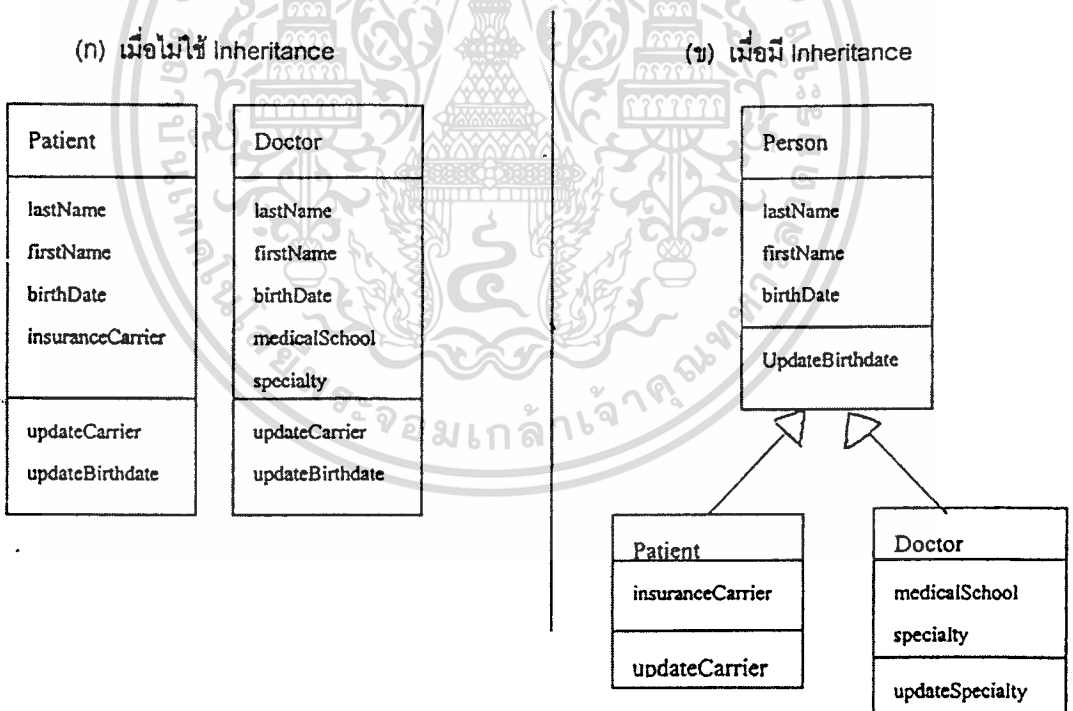
ดูตัวอย่างจากรูปที่ 2.23 แสดงคลาส “Person” (บุคคล) เป็นซูเปอร์คลาส (Super class) ของซับคลาส “Doctor” (นายแพทย์) และ “Patient” (คนไข้) และในขณะเดียวกันคลาส “Doctor” ก็ทำหน้าที่เป็นซูเปอร์คลาสของซับคลาส “General Practitioner” (แพทย์ทั่วไป) และ “Specialist” (แพทย์ชำนาญการเฉพาะด้าน)



รูปที่ 2.25 แสดงตัวอย่างความสัมพันธ์ของคลาสต่างๆเป็น โครงสร้างแบบลำดับชั้น (Hierarchy)

ชั้นคลาสจะได้รับการถ่ายทอดทั้งคุณสมบัติ (Attribute) และเมธอด (Methods) ที่กำหนดไว้ในซูเปอร์คลาส กล่าวคือ ชั้นคลาสจะมีแอทริบิวต์และเมธอดเหมือนกับที่มีอยู่ในซูเปอร์คลาส จากตัวอย่างในรูปที่ 2.24 เนื่องจากทั้ง Doctor และ Patient ต่างก็เป็นชั้นคลาสของ Person ดังนั้น ทั้งคลาส Doctor และ Patient จะมีแอทริบิวต์และเมธอดที่กำหนดไว้ในคลาส Person ด้วย

จากรูปที่ 2.24 แสดงความสัมพันธ์ระหว่างคลาสที่กล่าวถึงเนื้อหาแบบครอบคลุม โดยทั่วไป ซึ่งจะอยู่ด้านบนของโครงสร้าง และคลาสที่กล่าวถึงเนื้อหาแบบเฉพาะเจาะจงจะอยู่ด้านล่างของโครงสร้าง สังเกตการใช้สัญลักษณ์ลูกศรที่ปลายเป็นสามเหลี่ยมสีขาว ความสัมพันธ์ดังกล่าวนี้จะเรียกว่าเป็นสัญลักษณ์แสดงความสัมพันธ์ที่เรียกว่า “Is-a relationship” เช่น เราอาจกล่าวว่า “Patient” is a “Person” (Patient เป็นชั้นคลาส และเป็นซูเปอร์คลาส)



รูปที่ 2.26 เปรียบเทียบระหว่างการออกแบบคลาสแบบที่ไม่ใช้ Inheritance ในรูป(ก) และแบบที่ใช้ Inheritance ในรูป (ب)

จากรูปที่ 2.25 จะเห็นว่าการออกแบบคลาสในลักษณะที่มีการใช้ Inheritance นั้น จะช่วยทำเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ให้ง่ายต่อการกำหนดแอทริบิวต์และเมธอดที่มีลักษณะร่วมกัน เช่น Last Name, First Name, Birth ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมีเหตุดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

date และเมธอร์ค update birth date ซึ่งเราจะนำมารวมกันไว้ภายในซูเปอร์คลาส Person สำหรับคลาส Patient และคลาส Doctor จะมีทั้งแอททริบิวและเมธอร์คข้างต้นได้ โดยการประกาศให้ทั้งสองคลาสเป็นซับคลาสของ Person (สังเกตจากรูปเราใช้สัญลักษณ์เป็นเส้นลูกศรเป็นสามเหลี่ยมสีขาว) ให้สังเกตประสิทธิภาพในการประกาศตามรูปที่ 2.25 (ข) ดีกว่าการประกาศในรูปที่ 2.25 (ก)

คลาสส่วนใหญ่ในโครงสร้างลำดับชั้นนั้นจะเป็นต้นแบบที่ทำให้เราสามารถสร้างออบเจกต์ที่จะบรรจุด้วยค่าหนึ่งๆที่กำหนดให้เราเรียกคลาสที่สามารถสร้าง Instance หรือสร้างเป็นออบเจกต์ได้นี้ว่า “คลาสรูปธรรม” (Concrete Class) ตัวอย่าง เช่น “สมชาย” และ “สมหญิง” สามารถเป็นออบเจกต์ของคลาส Patient ได้ เราก็เรียกว่า Patient ที่เป็น “คลาสชนิดรูปธรรม” (Concrete Class)

อย่างไรก็ตาม มีคลาสบางคลาสที่ถูกออกแบบขึ้นมาโดยที่อาจจะไม่สามารถนำไปสร้างออบเจกต์ได้ เนื่องจากคลาสถูกออกแบบขึ้นมาเพื่อเป็นกรอบของข้อมูลเพื่อให้เป็นซูเปอร์คลาสอื่นมาสัมพันธ์เป็นซับคลาส (โดยเฉพาะอย่างยิ่งคือคลาสที่อยู่ในตอนบนๆ ของโครงสร้างลำดับชั้น) คลาสประเภทนี้เรียกว่า “คลาสนามธรรม” (Abstract Class) Person จะเป็นตัวอย่างหนึ่งของคลาสนามธรรม เนื่องจากแทนที่จะสร้างออบเจกต์ในชนิดของ Person เราจะสร้างออบเจกต์ที่มีรูปธรรมหรือเจาะจงมากกว่านั้น คือ เราจะเลือกสร้างออบเจกต์เป็นชนิด Doctor หรือ Patient ซึ่งทั้งสองเป็นซับคลาสของ Person เป็นต้น

2.2.4 โพลิมอร์ฟิซึม (Polymorphism)

ในโพลิมอร์ฟิซึม เป็นคุณสมบัติที่สำคัญมากอีกประการหนึ่งของการพัฒนาระบบเชิงออบเจกต์ โพลิมอร์ฟิซึม หมายถึงการที่เมซเซจ (Message) หรือการเรียกชื่อเมธอร์คหนึ่งๆอาจมีความหมายหรือมีการทำงานได้หลายแบบขึ้นอยู่กับชนิดของออบเจกต์ที่ถูกเรียกใช้งานนั้นๆ เช่น พิจารณาตัวอย่างจากรูปที่ 2.23 นั้น เมซเซจที่เรียก “Insert” จะมีความหมายแตกต่างกันไประหว่าง Insert ของออบเจกต์คนไข้ (Patient) และ Insert ของออบเจกต์การนัดหมาย (Appointment) (ข้อมูลที่จะถูกบันทึกจะแตกต่างกันระหว่างข้อมูลคนไข้และข้อมูลการนัดหมาย แต่ด้วยหลักการที่เป็นข้อดีสำคัญของระบบเชิงออบเจกต์ ก็คือผู้ใช้เมธอร์คดังกล่าวไม่ต้องพะวงถึงขั้นตอนในรายละเอียดว่าระบบจะทำงานอย่างไร (How) เราเพียงแค่ทำการส่งเมซเซจ (Message) ไปยังออบเจกต์ที่ถูกเรียกใช้นั้นๆ จะรับหน้าที่ในการตีความหมายเมซเซจนั้นๆ ว่าเป็นเมซเซจของออบเจกต์ในคลาสใดแล้วดำเนินการกระบวนการประมวลผลที่เหมาะสมต่อไป

ตัวอย่าง เช่น ถ้าเราส่งเมซเซจว่า “Draw Your Self” ไปยังออบเจกต์สี่เหลี่ยมจัตุรัส (Square Object) ออบเจกต์วงกลม (Circle Object) และออบเจกต์สามเหลี่ยม (“Triangle Object”) เราจะได้

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลลัพธ์ที่แตกต่างกัน ถึงแม้จะใช้ชื่อเมทธีเดียวกัน เนื่องจากแต่ละอปเจกต์จะประมวลผลตาม
 นิยามของเมธอร์คของตัวเองแตกต่างกันไป

หลักการของโพลิเมอร์พีซีมจะช่วยให้การพัฒนาโปรแกรมมีความยืดหยุ่น และกระทำได้ง่าย
 ตามการบำรุงรักษาโปรแกรม



บทที่ 3

การวิเคราะห์และการออกแบบระบบงาน ระบบคิดเงินลูกค้า (Billing System)

3.1 Billing System

ระบบคิดเงินลูกค้า เป็นระบบที่ช่วยให้ระบบงานธุรกิจซึ่งมีการขายสินค้าและบริการให้กับลูกค้าแล้ว ต้องมีการคิดเงินลูกค้าโดยการคำนวณค่าใช้จ่ายต่างๆของลูกค้าแต่ละคน แล้วทำการออกใบแจ้งหนี้ไปยังลูกค้า โดยเมื่อลูกค้าได้ชำระหนี้มาแล้ว ระบบก็จะมีการออกใบเสร็จรับเงินให้กับลูกค้า แต่ในกรณีที่มีการคิดเงินลูกค้าเกินจากมูลค่าจริงที่ลูกค้าต้องจ่าย ระบบจะออกใบ Credit Note ส่วนในกรณีที่มีการออกใบแจ้งหนี้ไปเป็นมูลค่าต่ำกว่าค่าใช้จ่ายจริงของลูกค้า ระบบจะออกใบ Debit Note ให้กับลูกค้า

3.2 การทำงานของระบบงานเดิม

แผนก Sales

ลักษณะการดำเนินงาน

1. การให้บริการลูกค้าจะมี 2 แบบคือ ลูกค้ามาติดต่อที่ Shop หรือเข้ามาติดต่อที่สำนักงานใหญ่โดยตรงเลย ซึ่งทาง Sales จะยื่นเอกสาร Application Form ให้ลูกค้ากรอกรายละเอียดลูกค้าจ่ายค่าธรรมเนียมแรกเข้า Pay-in Slip และยื่นเอกสารประกอบต่างๆ และมีการออกใบเสร็จรับเงินชั่วคราว
2. เมื่อได้ Application Form จากทางลูกค้าแล้ว Sale ก็จะกรอกข้อมูลต่างๆลงใน Computer (Program ที่ทาง Sales จัดทำขึ้นเอง) โดยรายละเอียดของข้อมูลต่างๆมีดังนี้

-หมายเลขโทรศัพท์ ADSL

-บริษัท ISP ที่จะทำการเชื่อมต่อ

-รายละเอียด ชื่อ-ที่อยู่ลูกค้า , สถานที่ส่งใบแจ้งหนี้ , สถานที่ติดตั้งบริการ โดย ลูกค้าจะ

แบ่งเป็น 2 ประเภทคือ บุคคลธรรมดา/ในนามบริษัท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
-แผนที่ของสถานที่ติดตั้งบริการ
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

-โมเด็มที่ใช้ (ยี่ห้อ/รุ่น/ชนิดของ Interface)

-ความเร็วที่ใช้

-อัตราค่าใช้บริการจากนั้น Sales จะทำการส่ง Application Form/สำเนาบัตรประชาชน/Pay-in Slip/ แผนที่ลูกค้า ให้กับทางแผนก Terminal Operation และแผนก บัญชี/การเงินต่อไป

แผนกบัญชี/การเงิน

ลักษณะการดำเนินงาน

1. เมื่อ Sales ส่งเอกสารข้อมูลลูกค้ามา บัญชีจะต้องทำการออก ใบเสร็จรับเงินชั่วคราวให้แก่ลูกค้า โดย Running เลขที่ใบเสร็จรับเงินชั่วคราวเอง
2. จากนั้นทำการบันทึกข้อมูลลง Excel Format
3. เมื่อได้รับเอกสารการติดตั้งเสร็จเรียบร้อยจากทาง Terminal Operation ก็จะมีการออก Invoice โดยคิดจากวันที่เริ่ม In Service ได้ ค่าใช้จ่ายที่ลูกค้าต้องชำระมีดังนี้

-ค่าหลักประกัน

-เงินมัดจำอุปกรณ์ เช่น โมเด็ม

-ค่าธรรมเนียมแรกเข้า

3.3 ขั้นตอนการดำเนินการพัฒนาระบบ

ประกอบไปด้วยขั้นตอนการดำเนินงานดังต่อไปนี้ คือ

3.3.1 Problem Definition

เป็นขั้นตอนในการหาปัญหาที่แท้จริงซึ่งเกิดขึ้นกับระบบงานเดิม ซึ่งสรุปปัญหาที่พบได้ดังนี้ คือ

1. เนื่องจากข้อมูลที่ได้จาก Sales ยังเป็นข้อมูลที่มาในรูปเอกสารทั้งหมด เมื่อเจ้าหน้าที่ประสานงานรับเอกสาร ก็จะมาทำการบันทึกข้อมูลลงใน MS Excel ซึ่งเป็นการทำงานที่ซ้ำซ้อน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ระบบการออก ใบแจ้งหนี้, ใบเสร็จรับเงิน ยังคงเก็บข้อมูลไว้ใน Excel Format จึงทำให้เกิดความไม่คล่องตัวในการทำงานทั้งในด้าน Process งานประจำวัน และการสรุปรายงานประจำเดือน

3.3.2 Requirement Analysis

จากการวิเคราะห์ความต้องการของผู้ใช้ สามารถสรุปได้ดังนี้

ความต้องการของแผนก Sale

- สามารถเก็บข้อมูลวงจร ADSL ของลูกค้าได้
- สามารถสร้างโปรโมชั่นต่างๆจากสินค้าและบริการ
- ความต้องการของแผนกบัญชี
- สามารถออกใบแจ้งหนี้ได้ถูกต้องทันความต้องการ
- สามารถดึงข้อมูลต่างๆมาใช้ได้จากแผนก Sales ได้ทันที
- ต้องการรายงาน

3.3.3 Specification

สามารถสรุปได้ดังนี้

- ทำการวิเคราะห์และออกแบบระบบโดยใช้ยูเอ็มแอล
- ใช้ Rational Rose 2000 เป็นเครื่องมือในการวิเคราะห์และออกแบบระบบ
- ใช้เครื่องมือในการเขียนโปรแกรม คือ Delphi Client/Server Suite 4.0
- ใช้ระบบฐานข้อมูลแบบ Relational Database เข้ามาช่วยในการเก็บข้อมูล โดยใช้

Oracle 8.0

3.3.4 Design

ไดอะแกรมต่างๆของยูเอ็มแอลที่เลือกมาใช้ในการออกแบบ คือ

3.3.4.1 Use Case Diagram

จากการศึกษาระบบงาน แล้วในระบบจะมีผู้ที่มีบทบาทต่อระบบ (Actor) ดังต่อไปนี้ คือ

Sales คือ แผนก Sales มีหน้าที่ในการกำหนดโปรโมชั่นออกมาเตรียมไว้ก่อนเพื่อสนับสนุนกับกลุ่มลูกค้าประเภทต่างๆ ซึ่งในระบบนี้ พนักงานขายส่วนหนึ่งจะเป็นผู้กำหนดโปรโมชั่นซึ่งประกอบไปด้วยการทำโปรโมชั่นเกี่ยวกับสินค้าและโปรโมชั่นในส่วนของการที่จำแนกเป็นค่าบริการ

Account คือแผนกบัญชี มีหน้าที่ในการออกใบแจ้งหนี้ (Invoice), ใบเสร็จรับเงิน (Receipt), ใบลดหนี้ (Credit Note) และใบเพิ่มหนี้ (Debit Note)

Use Case ประกอบด้วย

Use Case:Login

หน้าที่

ใช้ทุกครั้งที่มีการ Login เข้าสู่ ระบบเพื่อตรวจสอบผู้ใช้ก่อนการเข้าใช้งานระบบ

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และเจ้าหน้าที่แผนกขาย ซึ่งเป็นผู้ใช้งานระบบโดยตรง

ลำดับการทำงาน

เริ่มจากเจ้าหน้าที่แผนกบัญชี/การเงิน หรือเจ้าหน้าที่แผนกขายต้องการเข้าใช้งานระบบก็ทำการใส่ Login และรหัสผ่าน โดยการกระทำผ่านหน้าจอ Login เข้าสู่ระบบ โดยถ้าผู้ใช้งานใส่ Login หรือรหัสผ่านไม่ถูกต้องก็จะมีการแจ้งเตือนและผ่านไปทำ Use Case Invalid Login

Use Case:Invalid Login

หน้าที่

ใช้ในการจัดการเกี่ยวกับการ Login เข้าสู่ระบบ ไม่ถูกต้องของผู้ใช้งาน เช่นแสดง ข้อความแสดงความคิดเห็น และไม่ยอมให้เข้าสู่ระบบ

Actor ที่เกี่ยวข้อง

ไม่มี

ลำดับการทำงาน

จะเริ่มการทำงานใน Use Case นี้ก็ต่อเมื่อผู้ใช้งานใช้งาน Use Case Login ไม่ถูกต้อง

Use Case:Set Product Promotion

หน้าที่

ใช้ในการจัดโปรโมชัน ซึ่งต้องมีการจัดโปรโมชันของสินค้า และโปรโมชันของค่าบริการ มาก่อนนั่นคือ ต้องมีการเรียกใช้ Use Case Set Promotion Product และ Set Promotion Service ซึ่งเป็นความสัมพันธ์แบบ Include

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกขาย

ลำดับการทำงาน

เจ้าหน้าที่แผนกขายคิดโปร โมชันเกี่ยวกับสินค้าซึ่งจะมีส่วนลดแบบต่างๆให้กับสินค้าซึ่งจะระบุว่าเป็นอีพ็อค รูนไคเพื่อจัดทำส่วนลดแบบต่างๆให้ เช่นลดเป็นเปอร์เซ็นต์ หรือลดเป็นจำนวนเงิน

Use Case: Set Product Pricing

หน้าที่

ใช้ในการกำหนดราคาสินค้าที่จะขายให้กับลูกค้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกขาย

ลำดับการทำงาน

ก่อนที่จะมีการทำโปรโมชั่น เจ้าหน้าที่แผนกขาย ต้องทำการตั้งราคาสินค้าไว้ก่อน

Use Case: Set Service Pricing

หน้าที่

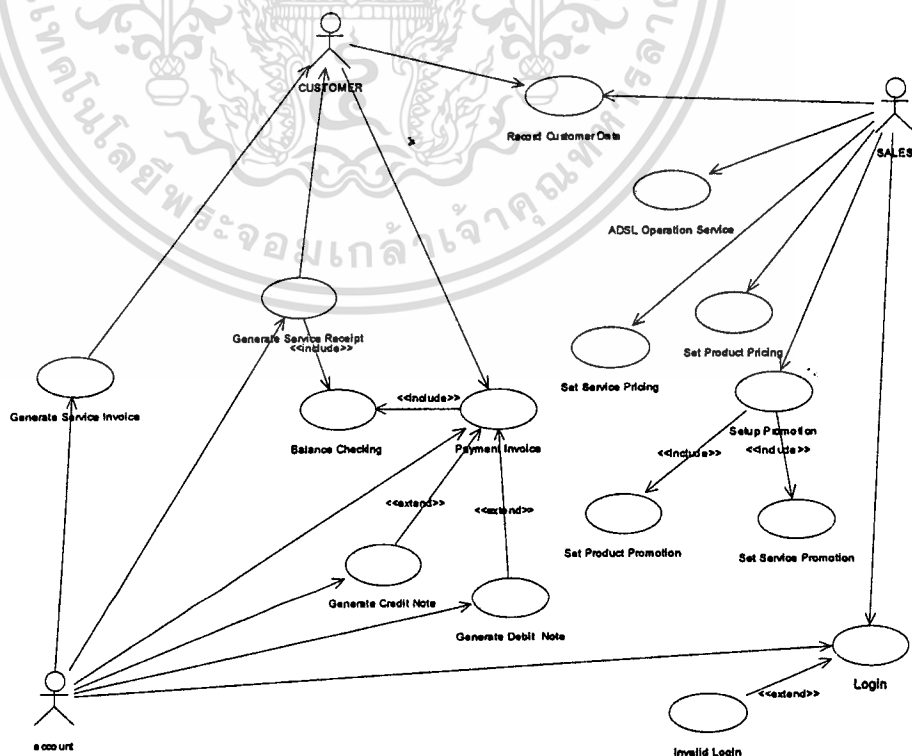
ใช้ในการกำหนดราคาที่เป็นส่วนของค่าบริการ เช่น ค่าธรรมเนียมแรกเข้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกขาย

ลำดับการทำงาน

ก่อนที่จะมีการทำโปรโมชั่น เจ้าหน้าที่แผนกขาย ต้องทำการตั้งราคาค่าบริการ ADSL ก่อน



Use Case:Record Customer Data

หน้าที่

ใช้ในการจัดการเกี่ยวกับข้อมูลลูกค้า เกี่ยวกับการเพิ่มข้อมูลลูกค้า ลบข้อมูลลูกค้า และการแก้ไขข้อมูลลูกค้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกขาย และลูกค้า

ลำดับการทำงาน

เมื่อลูกค้าติดต่อกับเจ้าหน้าที่แผนกขายเพื่อซื้อบริการ ADSL แล้ว เจ้าหน้าที่แผนกขายก็จะทำการบันทึกข้อมูลลูกค้า จากเอกสารที่ได้ คือ ใบสมัคร ซึ่งประกอบไปด้วย ชื่อลูกค้า ที่อยู่ รายละเอียดวงจร ADSL ลูกค้า และรายละเอียดโปรโมชันที่ลูกค้าเลือกซื้อ

Use Case: Generate Service Invoice

หน้าที่

ทำการคำนวณค่าใช้จ่ายเกี่ยวกับบริการ ADSL และออกใบ Service Invoice ให้กับลูกค้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เจ้าหน้าที่แผนกบัญชี/การเงิน เข้ามาทำการเรียกใช้งานโดยใช้ข้อมูลการใช้ค่าบริการ ADSL ของลูกค้าเพื่อทำการคำนวณค่าใช้จ่ายต่างๆของลูกค้า แล้วก็ทำการเก็บข้อมูลการคำนวณนั้นเข้าสู่ฐานข้อมูลเพื่อเตรียมที่จะพิมพ์ออกใบ Service Invoice แล้วส่งให้ลูกค้าเพื่อการเรียกเก็บเงินต่อไป

Use Case:Generate Product Invoice

หน้าที่

ทำการคำนวณค่าใช้จ่ายเกี่ยวกับบริการ ADSL และออกใบ Product Invoice ให้กับลูกค้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เจ้าหน้าที่แผนกบัญชี/การเงิน เข้ามาทำการเรียกใช้งานโดยใช้ข้อมูลเกี่ยวกับการสั่งซื้อ

อุปกรณ์ ADSL ของลูกค้า เพื่อทำการคำนวณค่าใช้จ่ายต่างๆของลูกค้า แล้วก็ทำการเก็บข้อมูลการเอกสารนี้เป็นเอกสารที่ส่งงานเวลาสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำนวณนั้นเข้าสู่ฐานข้อมูลเพื่อเตรียมที่จะพิมพ์ออกในใบ Product Invoice แล้วส่งให้ลูกค้าเพื่อการเรียกเก็บเงินต่อไป

Use Case:Balance Checking.

หน้าที่

ทำการตรวจสอบยอดที่ลูกค้าเป็นหนี้บริษัทว่ามูลค่าเท่าไร เมื่อเทียบกับมูลค่าของเงินที่ลูกค้าได้ชำระหนี้เข้ามา

Actor ที่เกี่ยวข้อง

ไม่มี

ลำดับการทำงาน

Use Case นี้จะถูกเรียกใช้โดย Use Case Generate Service Receipt Generate Product Receipt และ Use Case Payment Invoice

Use Case:Generate Product Receipt

หน้าที่

ทำการออกใบเสร็จรับเงินที่เกี่ยวกับสินค้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เจ้าหน้าที่แผนกบัญชี/การเงินเข้ามาร้องขอเพื่อทำการออกใบเสร็จรับเงินเกี่ยวกับสินค้า โดยจะทำการออกใบเสร็จรับเงินก็ต่อเมื่อลูกค้าได้ชำระเงินตามใบ Invoice ที่ได้ออกให้กับลูกค้าไปก่อนแล้ว

Use Case:Generate Service Receipt

หน้าที่

ทำการออกใบเสร็จรับเงินที่เกี่ยวกับบริการ เช่น ค่าธรรมเนียมแรกเข้า

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เจ้าหน้าที่แผนกบัญชี/การเงินเข้ามาร้องขอเพื่อทำการออกใบเสร็จรับเงินเกี่ยวกับบริการ โดยจะทำการออกใบเสร็จรับเงินก็ต่อเมื่อลูกค้าได้ชำระเงินตามใบ Invoice ที่ได้ออกให้กับลูกค้าไปก่อนแล้ว

Use Case:Payment Invoice

หน้าที่

รับชำระเงินค่าสินค้าและบริการ ADSL

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เมื่อลูกค้าได้รับใบแจ้งหนี้แล้วลูกค้าได้ไปชำระเงินและทางบริษัทได้รับเงินแล้วทางบริษัทจะออกใบเสร็จรับเงินค่าสินค้าหรือบริการตามที่บริษัทเรียกเก็บเงินไปให้แก่ลูกค้า ซึ่งใน Use Case นี้ได้เรียกใช้ Use Case Balance Checking เข้ามาใช้เพื่อตรวจสอบยอดเงินด้วย Generate Credit Note ถูกเรียกใช้โดย Account เพื่อทำการออกใบ Credit Note ให้กับลูกค้า โดยใบ Credit Note นี้จะออกให้ลูกค้ากรณีที่ไม่ใบ Invoice มีการเรียกเก็บเงินลูกค้ามากกว่าค่าใช้จ่ายจริงที่ลูกค้าต้องชำระ โดยที่ในกรณีปกติถ้าลูกค้าจ่ายเงินตรงตามใบแจ้งหนี้ก็จะไม่เกิด Credit Note ขึ้น ซึ่งถือว่า User Case Generate Credit Note เป็นกระบวนการทำงานเพิ่มเติมจาก Payment Invoice

Use Case:Generate Credit Note

หน้าที่

ถ้ามีการเรียกเก็บเงินตามใบแจ้งหนี้ที่ทำให้ยอดเงินไม่พอดี คือยอดเรียกเก็บเงินในใบแจ้งหนี้มีค่ามากกว่ายอดที่ลูกค้าต้องชำระจริง ทางบริษัทจะดำเนินการออกใบ Credit Note ให้กับลูกค้าเพื่อเป็นการลดหนี้ให้กับลูกค้า และในกรณีที่ลูกค้าชำระเงินมาพอดีกับยอดเงินที่เรียกเก็บไปในใบ Service Invoice ทางบริษัทก็ไม่ต้องออกใบ Credit Note

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เมื่อเจ้าหน้าที่แผนกบัญชี/การเงินได้ตรวจสอบการชำระเงินจากลูกค้าแล้ว ก็ต้องเลือกกว่าควรจะออกใบ Credit Note ให้กับลูกค้าตามมูลค่าที่แตกต่าง

Use Case:Generate Debit Note

หน้าที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้ามีการเรียกเก็บเงินตามใบแจ้งหนี้ที่ทำให้ยอดเงินไม่พอดี คือยอดเรียกเก็บเงินในใบแจ้งหนี้มีค่าน้อยกว่ายอดที่ลูกค้าต้องชำระจริง ทางบริษัทจะดำเนินการออกใบ Debit Note ให้กับลูกค้า เพื่อเป็นการเพิ่มหนี้ให้กับลูกค้า และในกรณีที่ลูกค้าชำระเงินมาพอดีกับยอดเงินที่เรียกเก็บไปในใบ Service Invoice ทางบริษัทก็ไม่ต้องออกใบ Debit Note

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

เมื่อเจ้าหน้าที่แผนกบัญชี/การเงินได้ตรวจสอบการชำระเงินจากลูกค้าแล้ว ก็ต้องเลือกว่าจะออกใบ Debit Note ให้กับลูกค้าตามมูลค่าที่แตกต่างกัน

Use Case: ADSL Operation Service

หน้าที่

ใช้ในการสนับสนุนการร้องขอดำเนินการวงจร ADSL เพื่อขอดำเนินการต่างๆ เช่น ขอเปลี่ยนความเร็ว ซึ่งทาง Sale จะทำการรับคำร้องขอจากลูกค้าแล้วทำการบันทึกข้อมูลเก็บไว้ และหลังจากนั้นก็ส่งเรื่องให้กับแผนก NMC เพื่อดำเนินการต่อไป

Actor ที่เกี่ยวข้อง

เจ้าหน้าที่แผนกบัญชี/การเงิน และลูกค้า

ลำดับการทำงาน

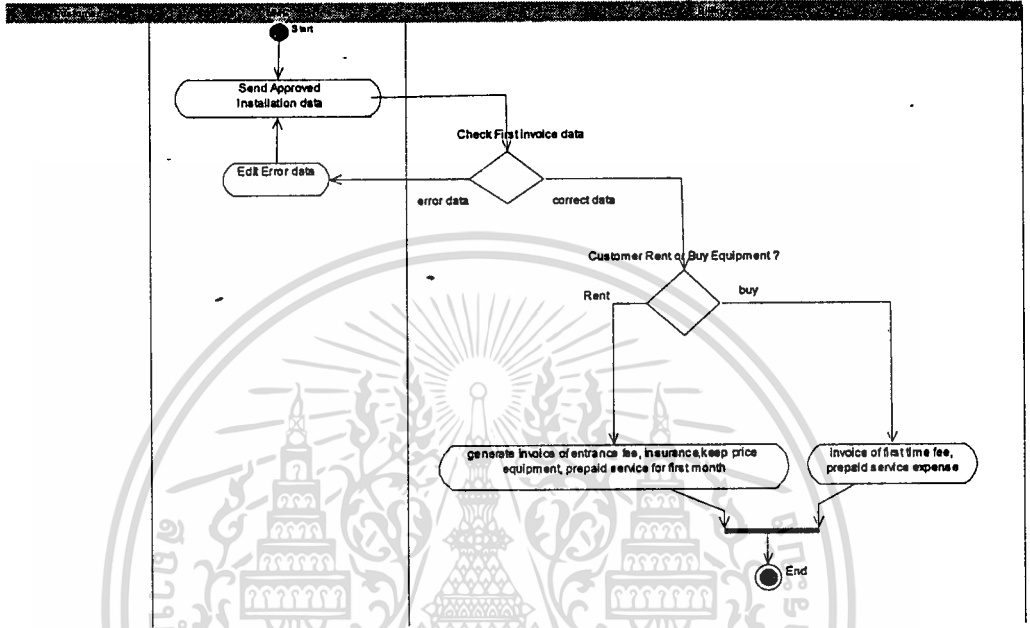
ลูกค้าร้องขอดำเนินการต่างๆ กับวงจร ADSL ซึ่งมีการดำเนินการต่างๆดังต่อไปนี้ คือ

- 1.ติดตั้งวงจรใหม่
- 2.ขอเปลี่ยนแปลงโมเด็ม
- 3.ขอเปลี่ยนแปลงหมายเลข ADSL
- 4.ขอเปลี่ยนแปลงความเร็ว
- 5.ขอระงับบริการชั่วคราว
- 6.ขอเปิดใช้บริการ
- 8.ขอยกเลิกบริการ

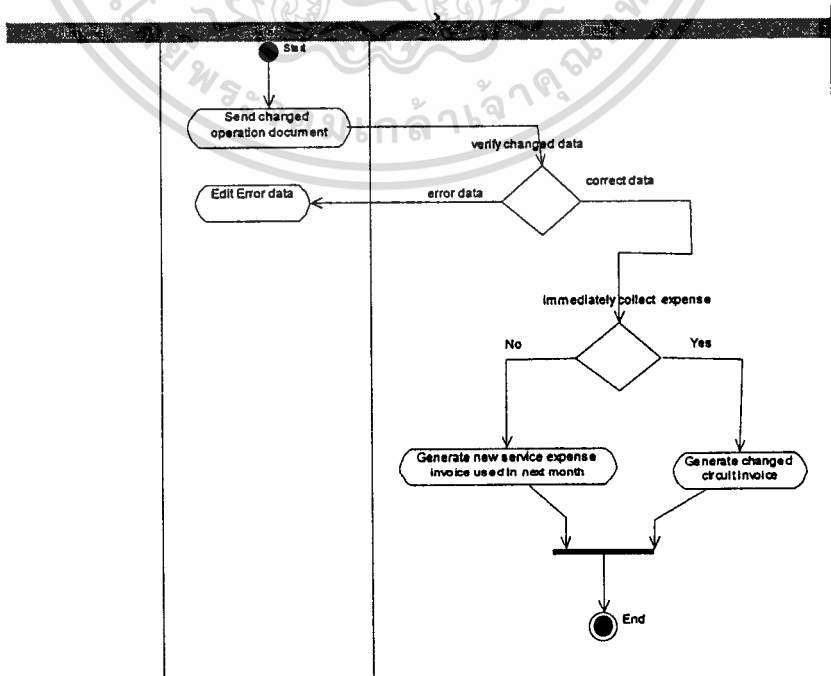
โดยที่ลูกค้าจะร้องขอดำเนินการผ่านมาทางเจ้าหน้าที่ฝ่ายขาย หลังจากนั้นเจ้าหน้าที่ฝ่ายขายก็จะนำคำร้องขอดำเนินการมาบันทึกข้อมูลเข้าระบบเพื่อให้ฝ่าย NMC ดำเนินการต่อไป ซึ่งหลังจากดำเนินการแล้ว จะมีผลกระทบต่อออกใบแจ้งหนี้เรียกเก็บเงินกับลูกค้า

3.3.4.2 Activity Diagram

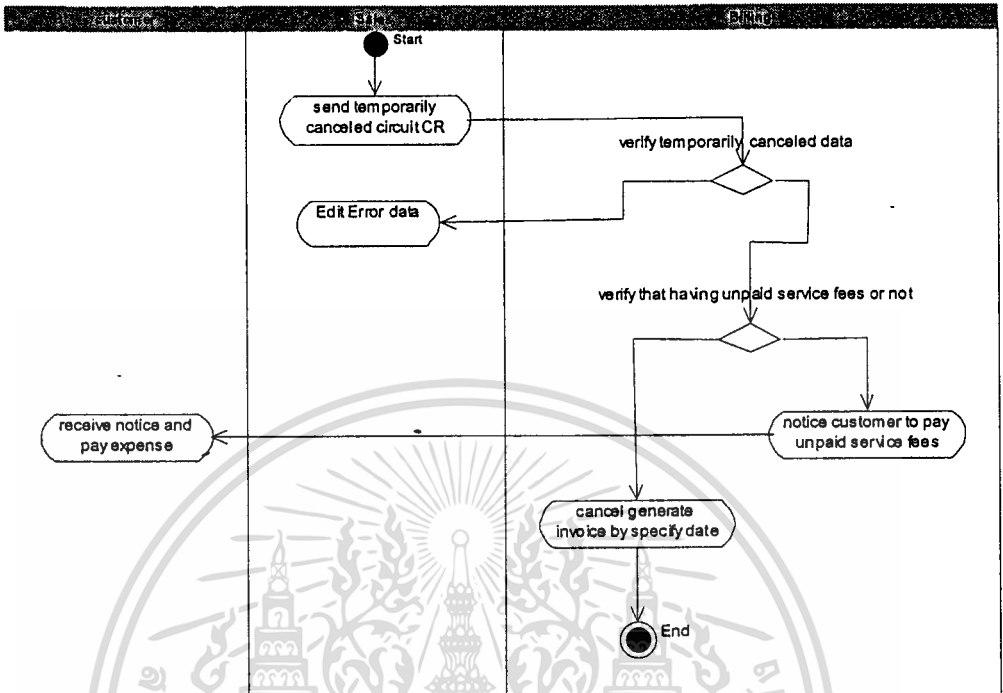
ใช้อธิบายลำดับการทำงานของระบบคิดเงินลูกค้า ซึ่งมีกรณีต่างๆที่เกิดขึ้นดังต่อไปนี้ คือ



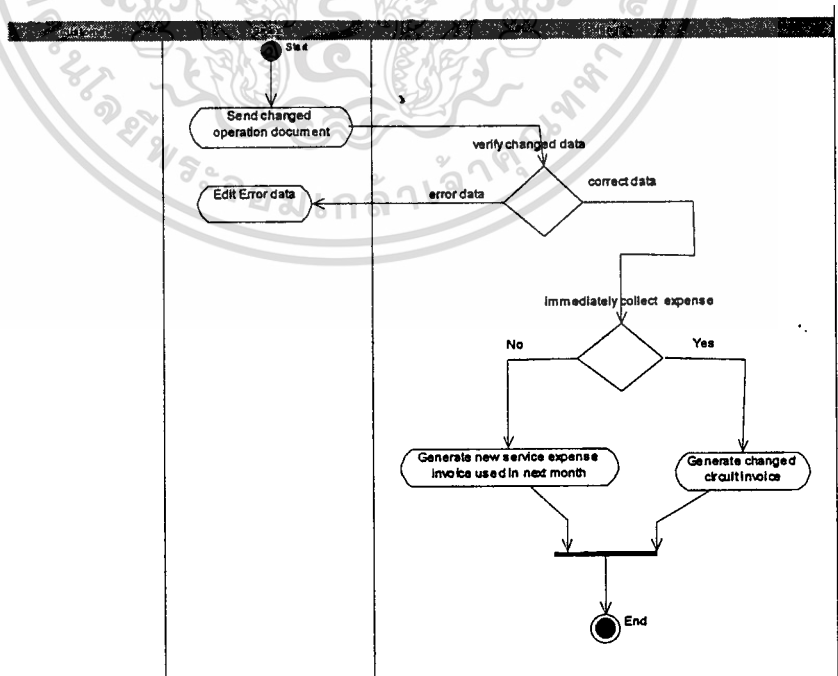
รูปที่ 3.2 กรณีลูกค้าติดตั้งใหม่



รูปที่ 3.3 กรณีลูกค้าต้องการเปลี่ยนแปลงวงจร



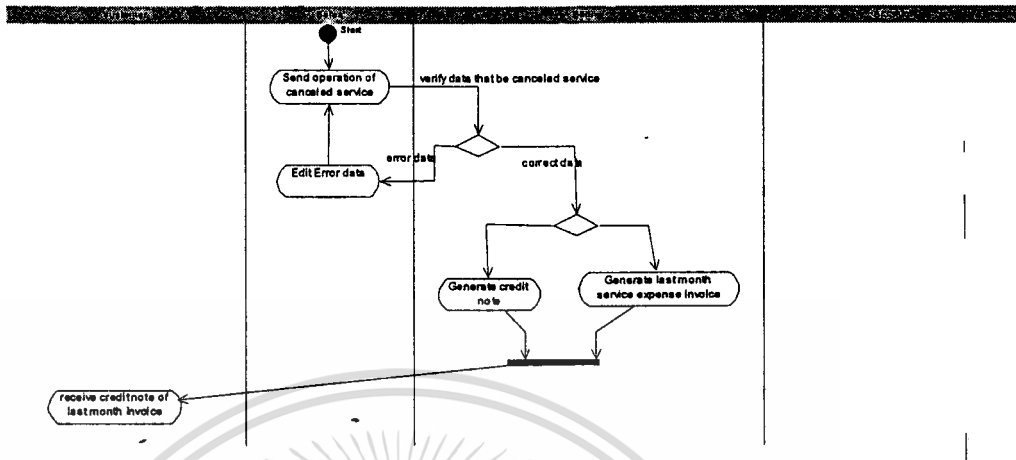
รูปที่ 3.4 กรณีลูกค้าต้องการยกเลิกการใช้งานวงจรชั่วคราว



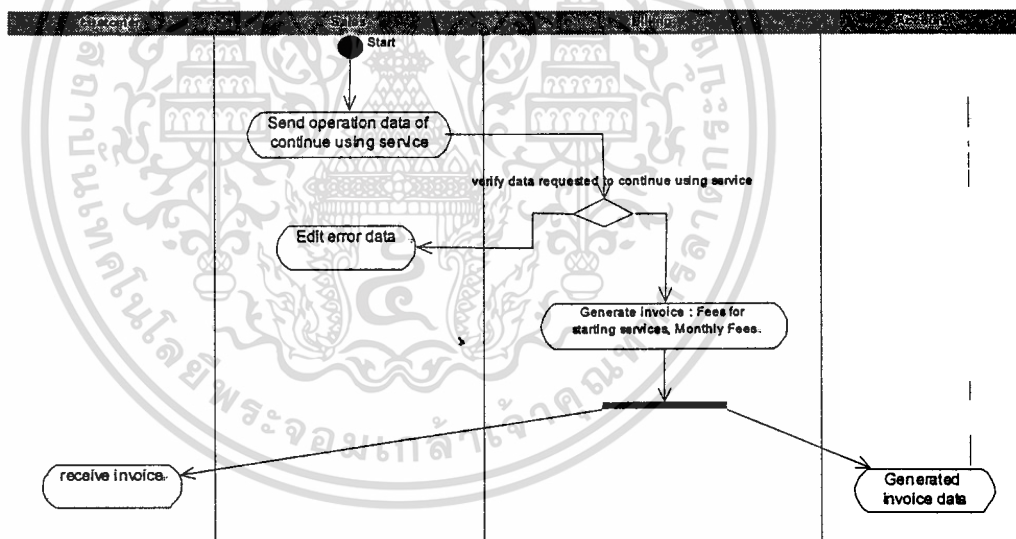
รูปที่ 3.5 กรณีลูกค้าต้องการเปลี่ยนค่าของวงจร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.6 กรณีลูกค้าต้องการยกเลิกวงจร

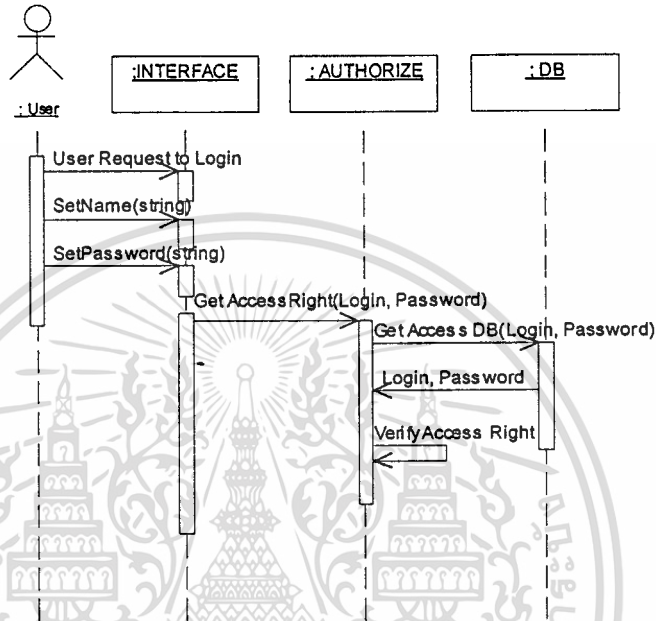


รูปที่ 3.7 กรณีลูกค้าต้องการขอใช้บริการต่อ

3.3.4.3 Sequence Diagram

ในระบบงานมี Sequence Diagram เพื่ออธิบายพฤติกรรมของระบบดังต่อไปนี้ คือ

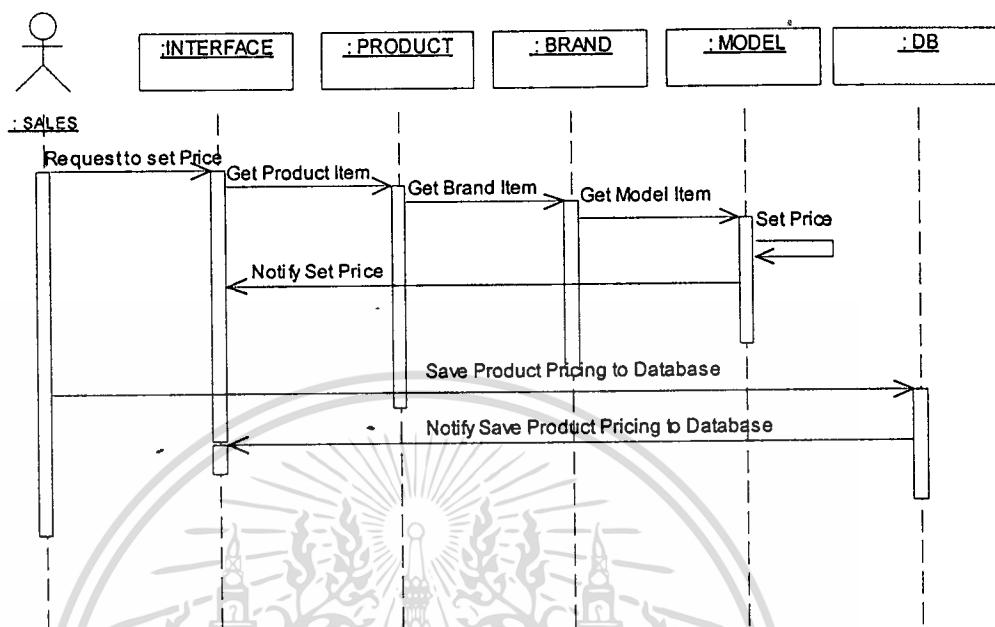
3.3.4.3.1 Login



จากรูป 3.8 แสดง Sequence Diagram ของ Use Case Login

Sequence Diagram ของการเข้าสู่ระบบของผู้ใช้งาน โดยผู้ใช้งานจะทำการร้องขอเข้าสู่ระบบผ่านคลาส INTERFACE หลังจากที่คุณใช้งานป้อน Login และ Password แล้วก็จะส่ง Message ไปหาคลาส AUTHORIZE เพื่อทำการตรวจสอบ Login และ Password ว่าถูกต้องหรือไม่โดยการที่คลาส AUTHORIZE จะส่ง message ไปถามจากฐานข้อมูล ถ้าถูกต้องผู้ใช้งานก็จะสามารถเข้าสู่ระบบเพื่อเริ่มต้นใช้งานได้

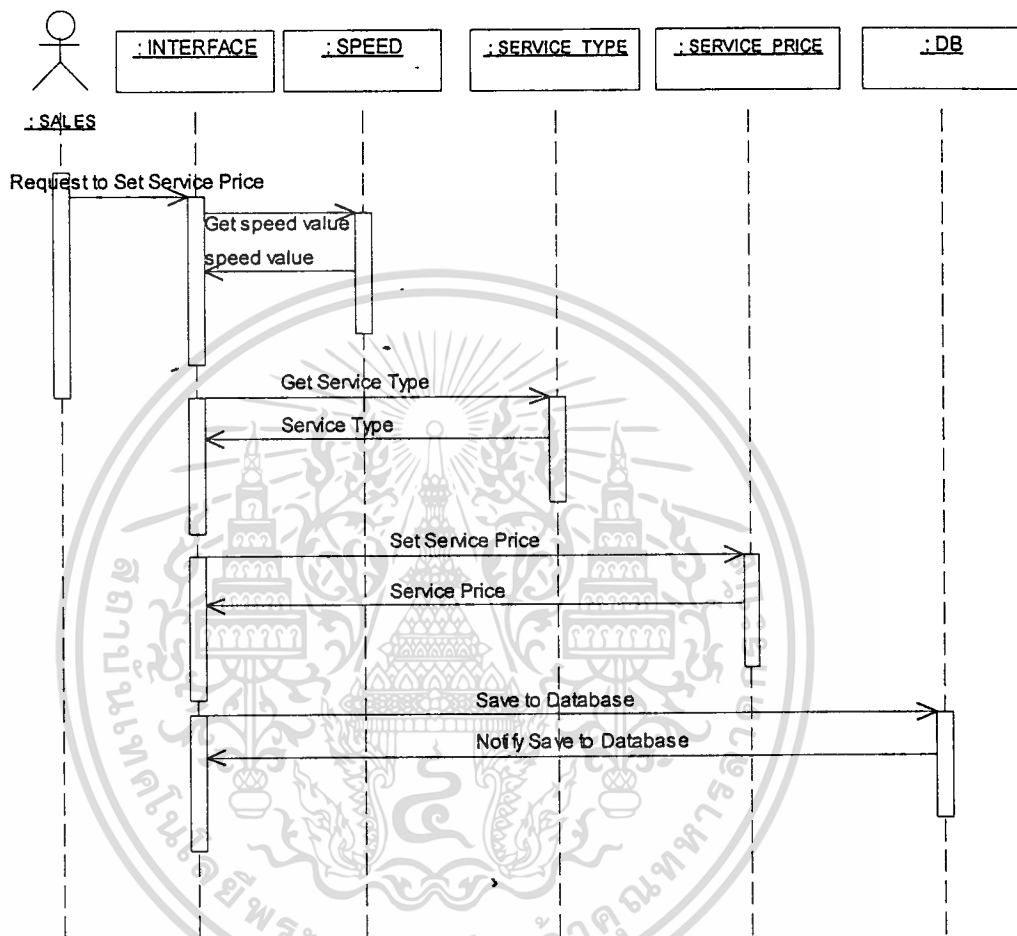
3.3.4.3.2 Set Product Pricing



รูปที่ 3.9 Sequence Diagram สำหรับ Use Case Set Product Pricing

จากรูปที่ 3.3 จุดประสงค์ของ Use Case นี้เพื่อทำการกำหนดราคาสินค้าที่เป็นอุปกรณ์ซึ่งใช้กับ ADSL โดยเริ่มแรก Sales จะร้องขอเพื่อทำการกำหนดราคาผ่านคลาส INTERFACE ซึ่งเป็นแบบฟอร์มซึ่งเป็นส่วนติดต่อกับผู้ใช้โดยจะต้องมีการเลือกชื่อ Product เลือกชื่อ Brand และเลือกชื่อ Model ที่ต้องการจากคลาส PRODUCT, BRAND และ MODEL ตามลำดับ ซึ่งหลังจากนั้นก็จะมี การกำหนดราคาของอุปกรณ์ที่ได้เลือกมาผ่านฟังก์ชัน SetPrice() ของคลาส Model ซึ่งเมื่อกำหนดราคาได้แล้วขั้นตอนต่อไปก็จะเป็นการบันทึกข้อมูลนี้ลงสู่ฐานข้อมูล

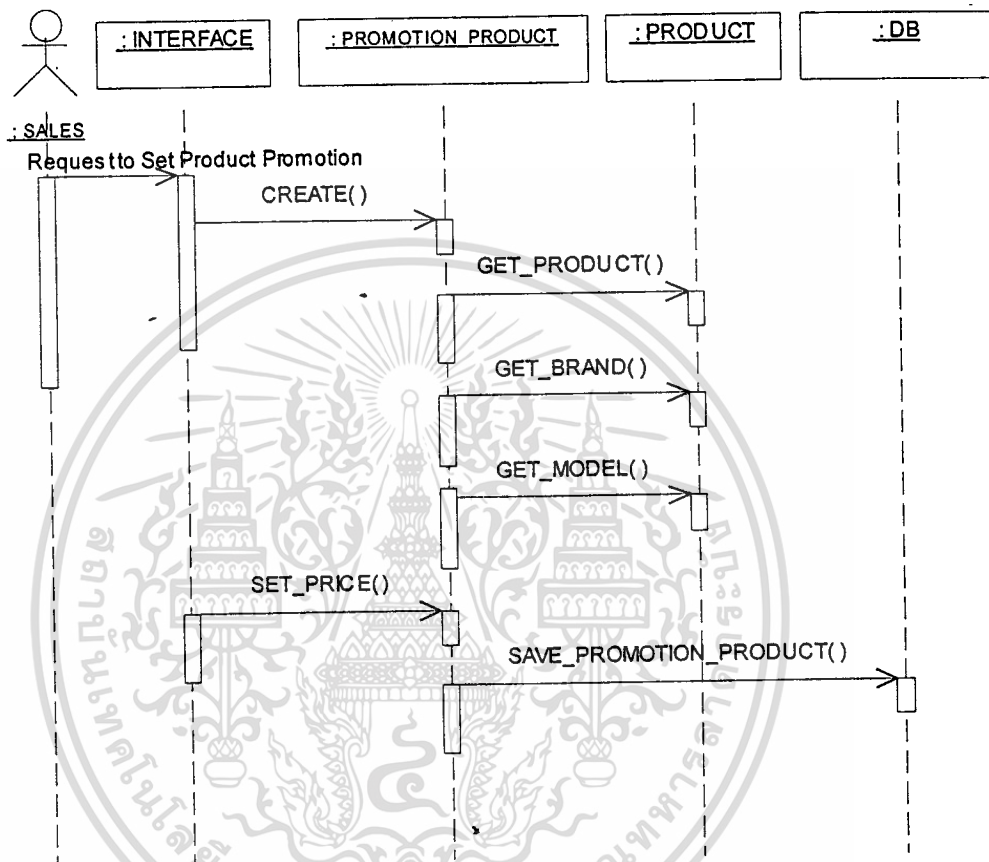
3.3.4.3.3 Set Service Pricing



รูปที่ 3.10 Sequence Diagram สำหรับ Use Case Set Service Pricing

จุดประสงค์ของการทำงานนี้คือ เพื่อเป็นการกำหนดราคาค่าบริการสำหรับลูกค้าที่เข้ามาใช้บริการ ADSL ขั้นตอนคือ Sales ทำการร้องขอการตั้งราคาบริการผ่านทางหน้าจอที่ติดต่อกับผู้ใช้ โดย sales จะกำหนด speed และ ประเภทการให้บริการ เช่น HOME ผ่านคลาส SPEED และ คลาส SERVICE_TYPE ตามลำดับ หลังจากนั้นเป็นการกำหนดราคาผ่านคลาส SERVICE_PRICE และ บันทึกผลลัพธ์ลงฐานข้อมูล

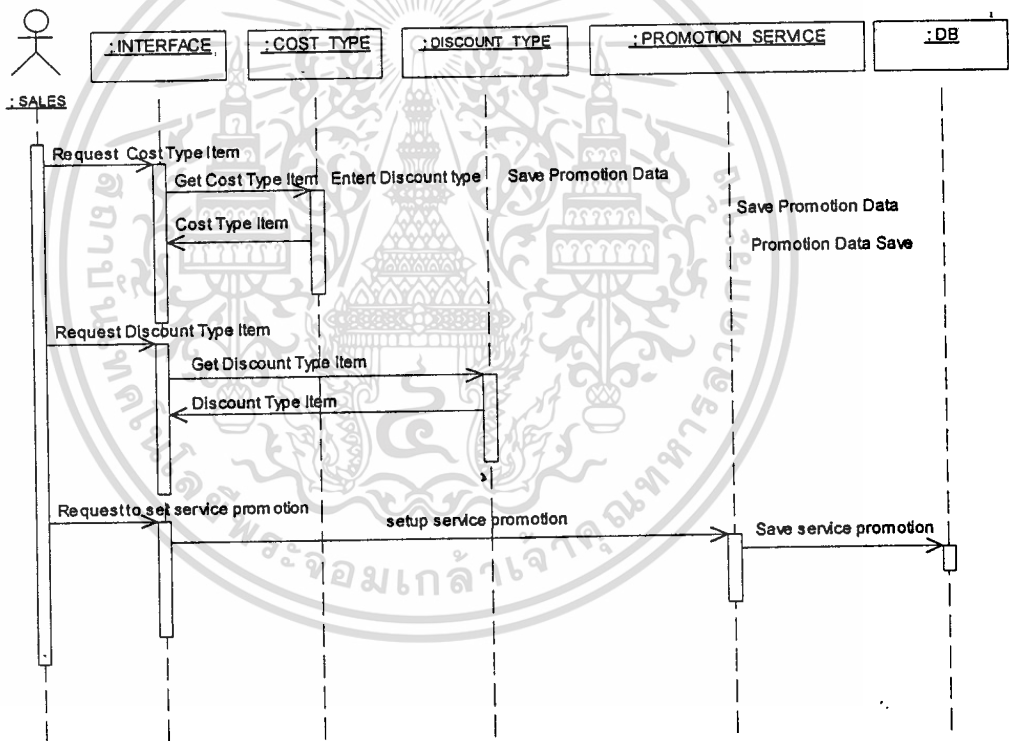
3.3.4.3.4 Set Product Promotion



รูปที่ 3.11 Sequence Diagram สำหรับ Use Case Set Product Promotion

จุดประสงค์ของการทำก็คือ เพื่อเป็นการจัด Promotion ของอุปกรณ์ ลำดับการทำงาน คือ Sales ทำการเลือกชนิดของ Product, Brand และ Model ตามลำดับแล้วส่งข้อมูลเหล่านี้ให้กับคลาส PROMOTION_PRODUCT หลังจากนั้นบันทึกผลลัพธ์ลงสู่ฐานข้อมูล

3.3.4.3.5 Set Service Promotion

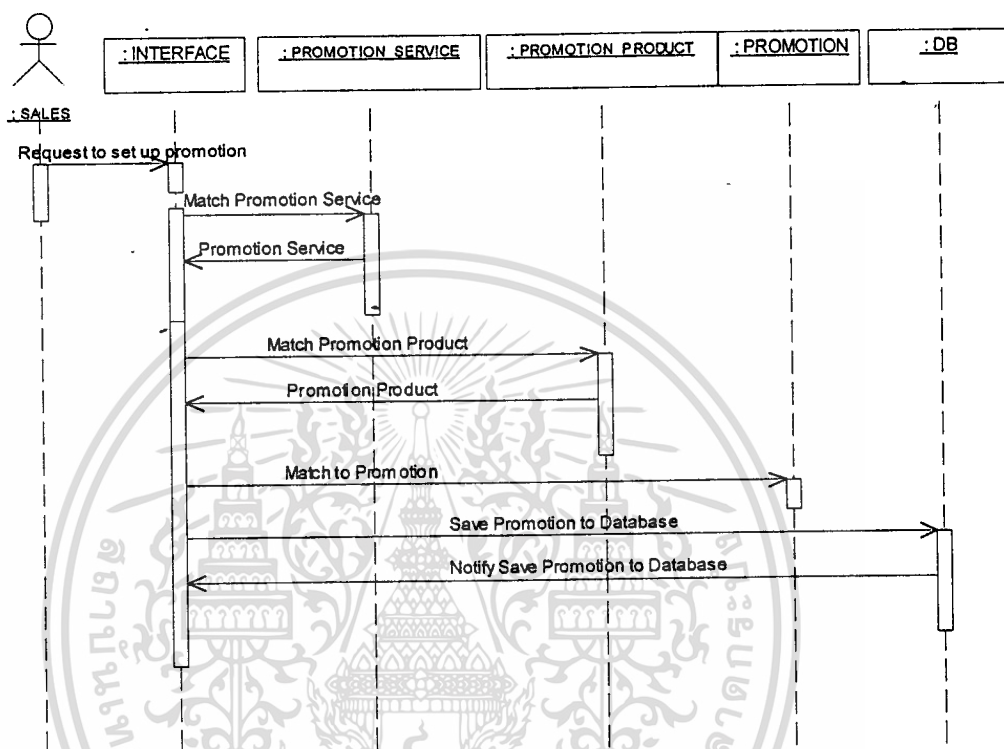


รูปที่ 3.12 Sequence Diagram สำหรับ Use Case Set Service Promotion

จุดประสงค์ คือ เพื่อเป็นการสร้าง Promotion ในส่วนของค่าบริการลำดับการทำงาน จะเริ่มจาก Sales เลือกประเภทค่าบริการและส่วนลดจากคลาส COST_TYPE และคลาส DISCOUNT_TYPE ตามลำดับ โดยการกระทำผ่านคลาส INTERFACE แล้วนำข้อมูลเหล่านี้ส่งให้คลาส PROMOTION_SERVICE เพื่อการสร้าง Promotion สำหรับค่าบริการแล้วบันทึกผลลัพธ์ลงสู่ฐานข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

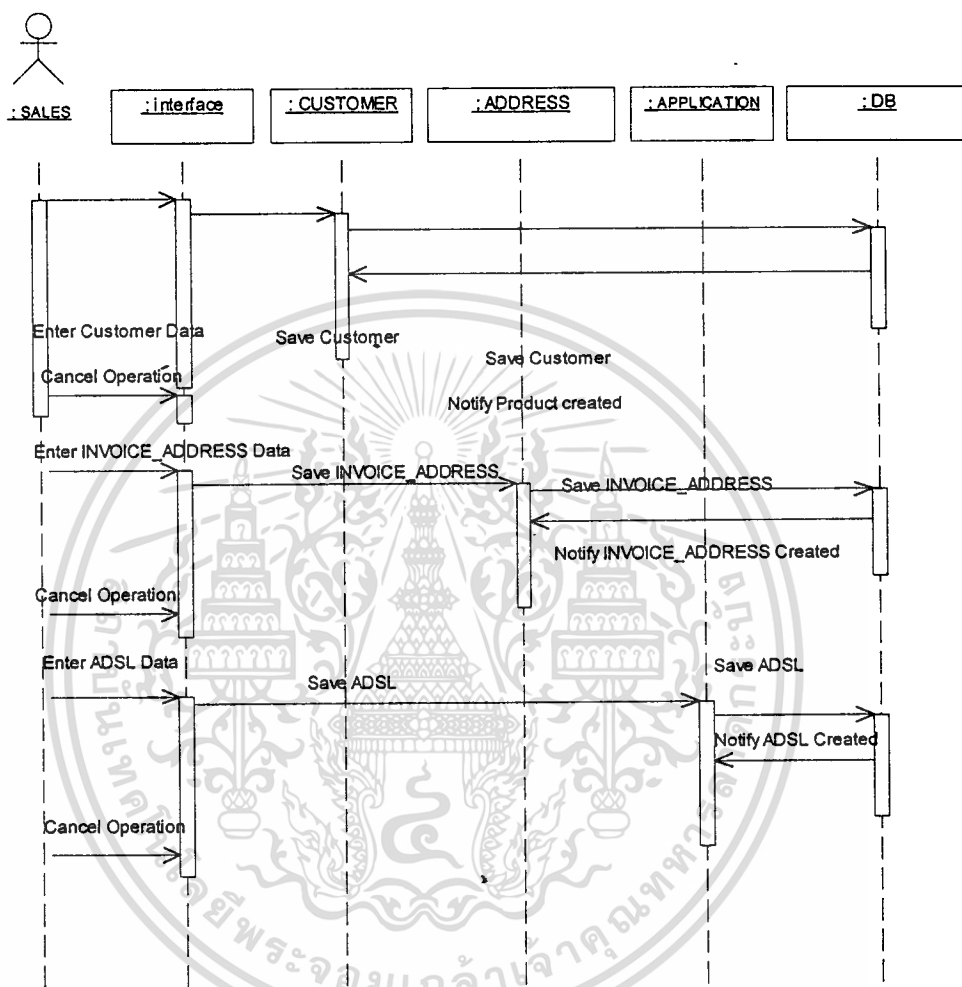
3.3.4.3.6 Setup Promotion



รูปที่ 3.13 Sequence Diagram สำหรับ Use Case Setup Promotion

จุดประสงค์ คือ เพื่อเป็นการกำหนดชุดของ Promotion โดยรวมโดยที่ Promotionหลักจะประกอบไปด้วย Promotion ย่อยของสินค้า และ Promotion ย่อยของค่าบริการ ขั้นตอนการทำงานเริ่มจาก Sales เลือก Promotion Service และ Promotion Product ที่ได้ setup ไว้ก่อนหน้านี้แล้ว แล้วเก็บข้อมูลของ Promotion หลักไว้ในคลาส Promotion และบันทึกส่งฐานข้อมูล

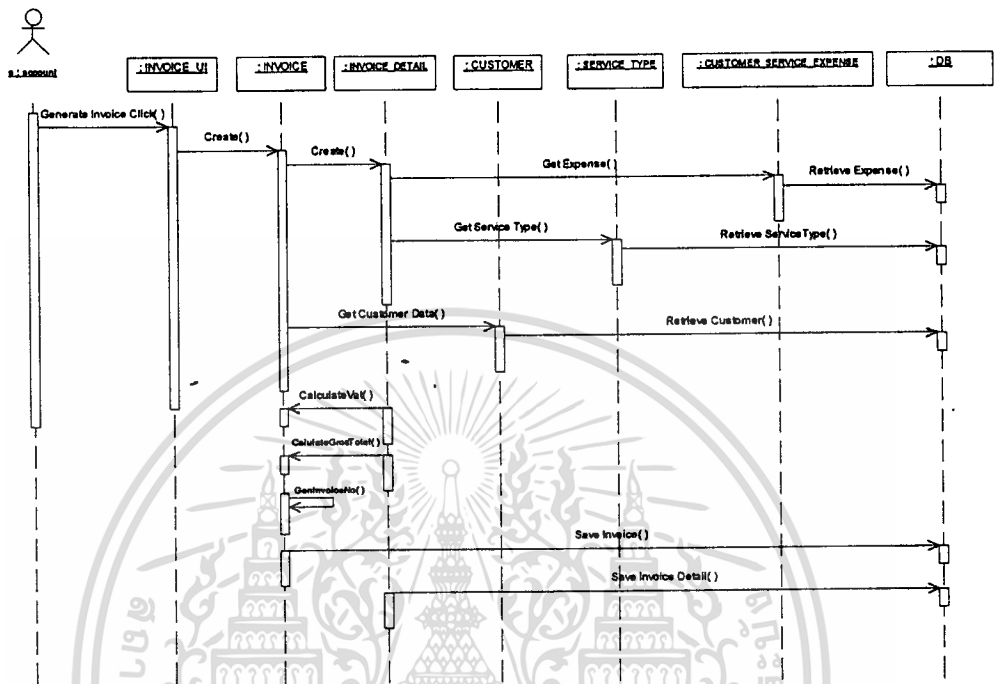
3.3.4.3.7 Record Customer Data



รูปที่ 3.14 Sequence Diagram สำหรับ Use Case Set Record Customer Data

จุดประสงค์คือ เพื่อเป็นการจัดการข้อมูลลูกค้าโดยมีการดำเนินการคือ เพิ่มข้อมูลลูกค้า แก้ไขข้อมูลลูกค้า ลบข้อมูลลูกค้า โดยลำดับการทำงานเริ่มจาก Sales บันทึกข้อมูลลูกค้า บันทึกข้อมูลสถานที่อยู่รับใบแจ้งหนี้ บันทึกข้อมูล ADSL ผ่าน User Interface เข้าไปยังคลาส CUSTOMER คลาส INVOICE_ADDRESS และคลาส ADSL ตามลำดับ หลังจากนั้นบันทึกข้อมูลลงฐานข้อมูล

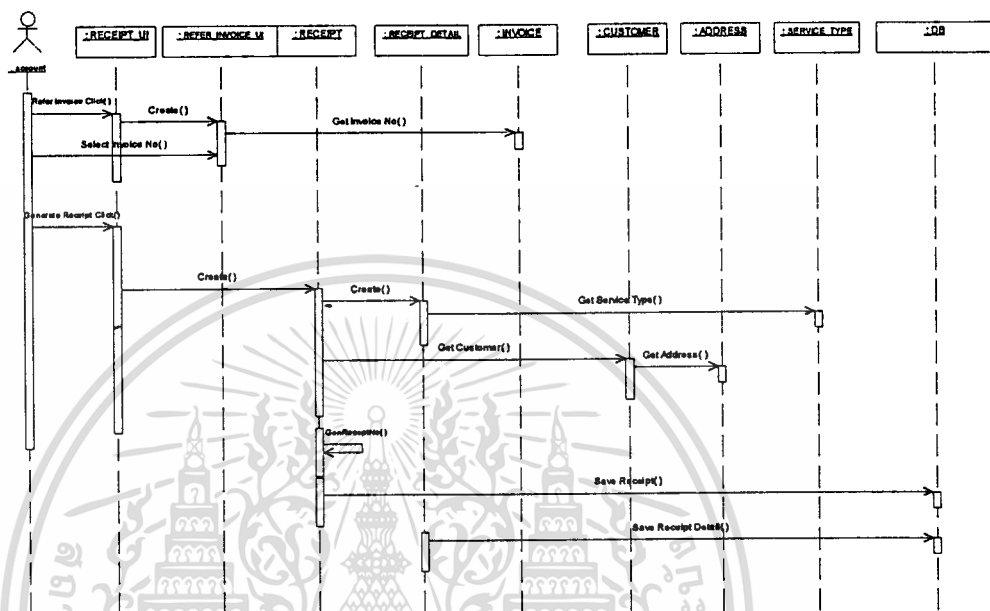
3.3.4.3.8 Generate Service Invoice



รูปที่ 3.15 Sequence Diagram สำหรับ Use Case Set Generate Service Invoice

จุดประสงค์ คือ เพื่อทำการคำนวณค่าใช้จ่ายต่างๆของลูกค้าแต่ละคน เก็บไว้ในฐานข้อมูลรอการเรียกใช้งานเพื่อออก INVOICE ตามรอบบัญชีที่ต้องการ ลำดับการทำงาน เริ่มจากเจ้าหน้าที่ฝ่ายบัญชีรอขอการ generate service invoice ผ่าน User Interface โดยในการคำนวณจะต้องมีการดึงข้อมูลที่เกี่ยวข้องมาใช้ในการคำนวณคือ ข้อมูล ADSL ข้อมูลลูกค้า ที่อยู่ สถานที่ส่งใบแจ้งหนี้จากคลาส ADSL คลาส CUSTOMER และคลาส INVOICE_ADDRESS ตามลำดับ หลังจากนั้นนำข้อมูลที่ได้มาคำนวณ เก็บผลลัพธ์ไว้ในคลาส INVOICE และ INVOICE_DETAIL แล้วบันทึกผลลัพธ์ลงสู่ฐานข้อมูล

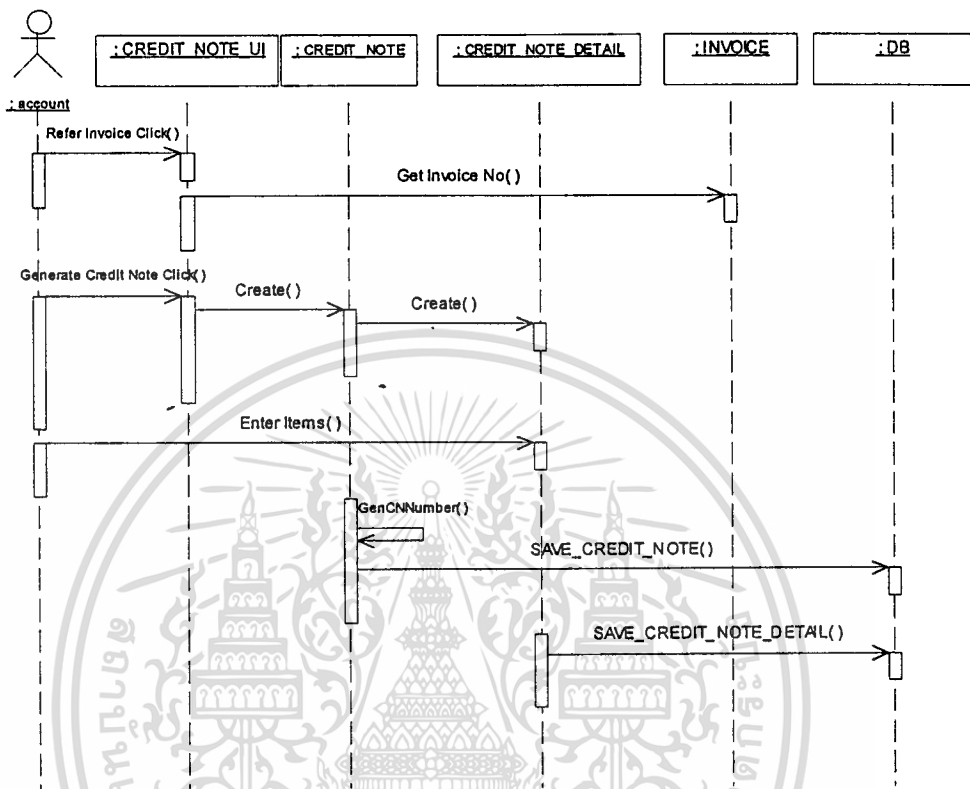
3.3.4.3.9 Generate Service Receipt



รูปที่ 3.16 Sequence Diagram สำหรับ Generate Service Receipt

จุดประสงค์ คือ เพื่อทำการออกใบเสร็จรับเงินค่าบริการให้กับลูกค้า ภายหลังจากที่ลูกค้าได้ชำระเงินตามใบแจ้งหนี้ โดยลำดับการทำงานเริ่มจากเจ้าหน้าที่ฝ่ายบัญชีได้ทำการกดปุ่ม Generate ในหน้าจอ RECEIPT_UI ซึ่งต้องมีการเลือกหมายเลขใบแจ้งหนี้เดิมอ้างอิงในการออกใบเสร็จรับเงินค่าบริการ ซึ่งจะปรากฏหน้าจอสำหรับให้เลือกหมายเลขใบแจ้งหนี้ขึ้นมาก่อน เมื่อเลือกหมายเลขใบแจ้งหนี้เดิมได้แล้วก็จะมีการดึงข้อมูลลูกค้า ที่อยู่ และข้อมูลการชำระเงินค่าบริการของลูกค้าจากใบแจ้งหนี้เข้ามา เพื่อจัดการบันทึกข้อมูลเหล่านี้กลับเข้าไปยังฐานข้อมูล เพื่อเตรียมการพิมพ์ออกใบเสร็จรับเงินค่าบริการต่อไป

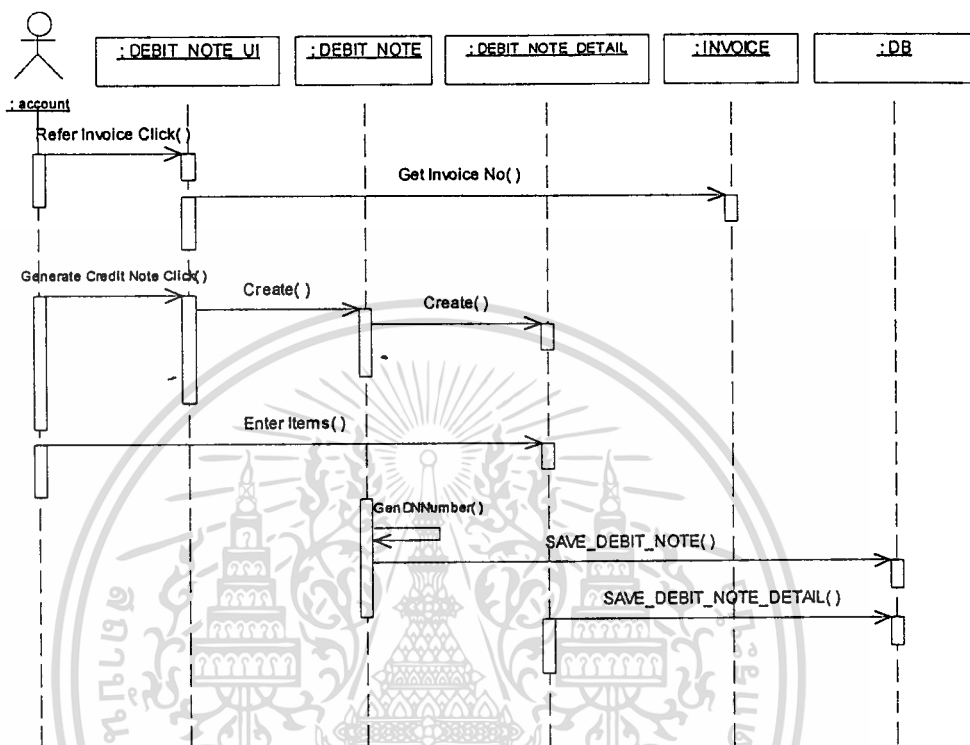
3.3.4.3.10 Generate Credit Note



รูปที่ 3.17 Sequence Diagram สำหรับ Generate Credit Note

จุดประสงค์ คือ เพื่อทำการออกใบ Credit Note ซึ่งจะเกิดขึ้นเมื่อลูกค้าได้ชำระเงินมากกว่ายอดในใบแจ้งหนี้ค่าบริการ หรืออาจเกิดจากลูกค้าได้เข้ามาร้องขอทำการลดความเร่งด่วนซึ่งมีผลให้ค่าใช้จ่ายลดลงจากเดิม ลำดับขั้นตอนการทำงานเริ่มจากเจ้าหน้าที่ฝ่ายบัญชีตรวจสอบยอดเงินในใบแจ้งหนี้จากหน้าจอ Account Control แล้วทราบว่าต้องทำการออกใบ Credit Note ให้กับลูกค้า จึงเข้ามายังหน้าจอ CREDIT_NOTE_UI หลังจากนั้นทำการเลือกหมายเลขใบแจ้งหนี้ค่าบริการที่ต้องการออกใบ Credit Note ซึ่งที่หน้าจอนี้จะมีการแสดงข้อมูลใบแจ้งหนี้ค่าบริการที่เกี่ยวข้องเข้ามา แล้วก็จะมีการสร้างใบ Credit Note ขึ้นมาใหม่ ให้ทำการป้อนรายการ พร้อมทั้งจำนวนเงินที่ต้องการลดหนี้ให้กับลูกค้า หลังจากนั้นเจ้าหน้าที่ฝ่ายบัญชีจะทำการกดปุ่ม Generate ซึ่งจะเป็นการบันทึกข้อมูลใบ Credit Note ไปนี้ลงสู่ฐานข้อมูลเพื่อเตรียมไว้สำหรับการพิมพ์ใบ Credit Note ให้กับลูกค้า

3.3.4.3.11 Generate Debit Note



รูปที่ 3.18 Sequence Diagram สำหรับ Use Case Generate Debit Note

จุดประสงค์ คือ เพื่อทำการออกใบ Credit Note ให้กับลูกค้าหลังจากที่เจ้าหน้าที่ฝ่ายบัญชีได้
 เลขออกใบแจ้งหนี้ค่าบริการไปแล้ว และปรากฏว่าลูกค้าได้มีการชำระเงินเข้ามาน้อยกว่าในใบแจ้ง
 หนี้ค่าบริการ หรืออาจเกิดจากลูกค้าได้มีค่าใช้จ่ายเพิ่มขึ้นจากการไปขอดำเนินการเพิ่มความเร่งด่วน
 ลำดับขั้นตอนการทำงานเริ่มจากเจ้าหน้าที่ฝ่ายบัญชีเข้ามายังหน้าจอ DEBIT_NOTE_UI แล้วทำการ
 เลือกหมายเลขใบแจ้งหนี้เดิมที่ต้องการออกใบ Debit Note หลังจากนั้นจะปรากฏข้อมูลอ้างอิงแสดง
 อยู่บน DEBIT_NOTE_UI แล้วเจ้าหน้าที่ฝ่ายบัญชีจะป้อนรายการ พร้อมทั้งจำนวนเงินที่ต้องการทำ
 การเพิ่มหนี้ให้กับลูกค้า หลังจากป้อนข้อมูลเรียบร้อย เจ้าหน้าที่ฝ่ายบัญชีจะทำการกดปุ่ม Generate
 จะทำให้มีการสร้าง DEBIT_NOTE และ DEBIT_NOTE_DETAIL ขึ้นมาหลังจากนั้นจะมีการสร้าง
 หมายเลขใบ Debit Note และข้อมูลของ DEBIT_NOTE และ DEBIT_NOTE_DETAIL จะถูก
 บันทึกกลับไปยังฐานข้อมูลเพื่อเตรียมไว้สำหรับการพิมพ์ใบ Debit Note ให้กับลูกค้า

3.3.4.4 Classs Diagram

เมื่อทราบความต้องการของผู้ใช้ระบบ และขั้นตอนการทำงานของระบบแล้ว เราก็มาออกแบบ ว่าในระบบของเราควรจะประกอบไปด้วยคลาสอะไรบ้าง และควรมี Attribute อะไรบ้างก่อน จากนั้นก็มาหาความสัมพันธ์ของแต่ละคลาส โดยพิจารณาว่าควรจะเป็นความสัมพันธ์แบบไหน และมี Multiplicity เท่าไร ซึ่งจากการออกแบบระบบ แล้วประกอบด้วย คลาสดังต่อไปนี้ คือ-

1. LOGIN จัดการเรื่อง Login , Password และสิทธิในการใช้งานระบบคิดเงินลูกค้า
2. INVOICE_ADDRESS เก็บที่อยู่ของลูกค้าเพื่อใช้เป็นที่อยู่สำหรับการส่งใบแจ้งหนี้
3. ISP เก็บรายชื่อ ISP
4. CUSTOMER เก็บข้อมูลลูกค้าที่มาซื้อบริการ ADSL
5. EMPLOYEE เก็บข้อมูลของพนักงานในบริษัท
6. APPLICATION เป็นข้อมูลรายละเอียดวงจรและโปรโมชันของลูกค้าแต่ละรายซึ่งสมัครใช้บริการ
7. PRODUCT ข้อมูลผลิตภัณฑ์ต่างๆที่ใช้ในวงจร ADSL หรืออุปกรณ์ประกอบต่างๆ
8. SERVICE_TYPE ประเภทของค่าใช้จ่ายเกี่ยวกับบริการ
9. SERVICE_PRICE ราคาของแต่ละรายการที่ไม่ใช่สินค้า ซึ่งถือเป็นค่าบริการ เช่น ค่าธรรมเนียมแรกเข้า
10. CUSTOMER_SERVICE_EXPENSE เก็บรายละเอียดค่าใช้จ่ายเกี่ยวกับบริการของลูกค้าที่ได้สมัครมาในแต่ละครั้ง
11. PROMOTION_PRODUCT เก็บรายการสินค้าประเภทอุปกรณ์ที่จัดเข้าโปรโมชันโดยสามารถระบุได้ว่าจะให้เป็นการทำโปรโมชันว่าจะให้มีการทำส่วนลดเป็นแบบใด คือลดเป็นเปอร์เซ็นต์หรือ ลดเป็นจำนวนเงิน หรือฟรี
12. PROMOTION_SERVICE เก็บรายการทำโปรโมชันเฉพาะของรายการที่ถือเป็นค่าบริการทั้งหมด โดยสามารถระบุได้ว่าจะให้เป็นการทำโปรโมชันแบบใด คือลดเป็นเปอร์เซ็นต์หรือ ลดเป็นจำนวนเงิน หรือฟรี
13. PROMOTION เก็บรายชื่อรายการ โปรโมชันหลัก ซึ่งในแต่ละ โปรโมชันประกอบไปด้วยรายการทำโปรโมชันจากสินค้า และรายการทำโปรโมชันจากค่าบริการ
14. INVOICE เก็บข้อมูลส่วนหัวของใบแจ้งหนี้
15. INVOICE_DETAIL เก็บข้อมูลรายการต่างๆที่มีในใบแจ้งหนี้นั้นๆ
16. DEBIT_NOTE เก็บข้อมูลส่วนหัวของใบเพิ่มหนี้

17. DEBIT_NOTE_DETAIL เก็บข้อมูลรายการของใบเพิ่มหนี้
18. RECEIPT เก็บข้อมูลส่วนหัวของใบเสร็จรับเงิน
19. RECEIPT_DETAIL เก็บข้อมูลรายการของใบเสร็จรับเงิน
20. DB เป็นคลาสพิเศษซึ่งมีหน้าที่ในการติดต่อกับฐานข้อมูลกับ Business คลาสจากรูปที่ สามารถอธิบายความสัมพันธ์ของคลาสต่างๆ ได้ดังนี้

1. CUSTOMER กับ INVOICE_ADDRESS
2. EMPLOYEE กับ APPLICATION คือพนักงานขายแต่ละคนสามารถขายบริการ ADSL

ได้หลายๆเบอร์

3. APPLICATION กับ CUSTOMER คือ ลูกค้าแต่ละคนสามารถซื้อบริการได้หลายครั้ง โดยการเขียนใบสมัครเข้ามา

4. PROMOTION กับ PROMOTION_SERVICE เป็นความสัมพันธ์แบบ PROMOTION ประกอบด้วยหลายๆ PROMOTION_SERVICE

5. PROMOTION กับ PROMOTION_PRODUCT เป็นความสัมพันธ์แบบ PROMOTION ประกอบด้วยหลายๆ PROMOTION_PRODUCT

6. INVOICE กับ INVOICE_DETAIL เป็นความสัมพันธ์แบบ INVOICE ประกอบด้วยหลายๆ INVOICE_DETAIL

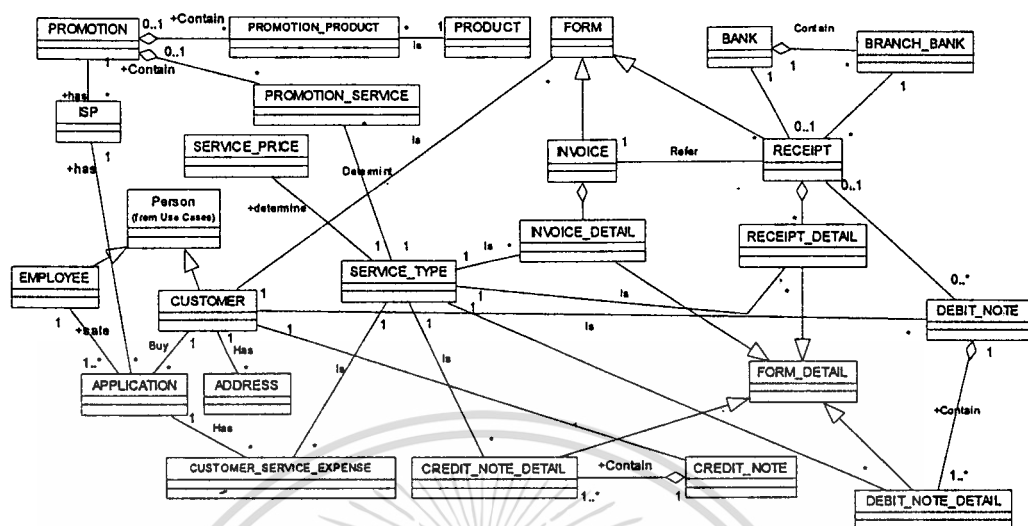
7. CREDIT_NOTE กับ CREDIT_NOTE_DETAIL เป็นความสัมพันธ์แบบ CREDIT_NOTE ประกอบด้วยหลายๆ CREDIT_NOTE_DETAIL

8. DEBIT_NOTE กับ DEBIT_NOTE_DETAIL เป็นความสัมพันธ์แบบ DEBIT_NOTE ประกอบด้วยหลายๆ DEBIT_NOTE_DETAIL

9. RECEIPT กับ RECEIPT_DETAIL เป็นความสัมพันธ์แบบ RECEIPT ประกอบด้วยหลายๆ RECEIPT_DETAIL

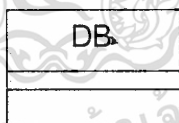
10. INVOICE กับ CREDIT_NOTE เป็นความสัมพันธ์แบบ INVOICE ประกอบด้วยหลายๆ CREDIT_NOTE_HEADER โดยที่ถ้าลูกค้ามีการจ่ายเงินตามยอดที่ปรากฏในใบแจ้งหนี้ จะส่งผลทำให้ยอดการชำระเงินพอดี ทำให้ระบบไม่ต้องออกใบ Credit Note

11. INVOICE กับ DEBIT_NOTE เป็นความสัมพันธ์แบบ INVOICE ประกอบด้วยหลายๆ DEBIT_NOTE



รูปที่ 3.19 Class Diagram สำหรับระบบคิดเงินลูกค้า (Billing System)

สำหรับคลาสที่เป็นคลาสสำหรับให้บริการในการติดต่อกับฐานข้อมูลโดยเฉพาะ คือ คลาส DB ซึ่งจะให้บริการหลักๆคือ การเพิ่มข้อมูล การแก้ไขข้อมูล และการค้นหาข้อมูลขึ้นมาจากฐานข้อมูล



รูปที่ 3.20 คลาส DB

รายละเอียดของคลาส

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
PROMOTION_ID	NUMBER(2,0)	รหัสโปรโมชั่น
PROMOTION_NAME	VARCHAR2(50)	ชื่อโปรโมชั่น
PROMOTION_DETAIL	VARCHAR2(255)	รายละเอียดโปรโมชั่น
PROMOTION_START	DATE	วันที่เริ่มโปรโมชั่น
PROMOTION_STOP	DATE	วันที่สิ้นสุดโปรโมชั่น
PROMOTION_PRICE	NUMBER(9,2)	ราคาโปรโมชั่น

ตารางที่ 3.1 แอทริบิวต์ของคลาส PROMOTION

การดำเนินการ	คำอธิบาย
GetPromotion(PROMOTION_ID)	ค้นหาโปรโมชั่นจากรหัสโปรโมชั่น
AddPromotion	เพิ่มข้อมูลโปรโมชั่น
การดำเนินการ	คำอธิบาย
UpdatePromotion(PROMOTION_ID)	แก้ไขข้อมูลโปรโมชั่น
DeletePromotion(PROMOTION_ID)	ลบข้อมูลโปรโมชั่น

ตารางที่ 3.2 การดำเนินการของคลาส PROMOTION

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
PROM_PROD_ID	NUMBER(2, 0)	รหัสโปรโมชั่นสินค้า
PROM_PROD_START	DATE	วันที่เริ่มโปรโมชั่น
PROM_PROD_STOP	DATE	วันที่สิ้นสุดโปรโมชั่น
PMPD_AMOUNT	NUMBER(7, 2)	ราคาโปรโมชั่น
DISCOUNT_TYPE	VARCHAR2(10)	ประเภทส่วนลด
COST_TYPE	VARCHAR2(10)	ประเภทค่าใช้จ่าย

ตารางที่ 3.3 แอทริบิวต์ของคลาส PROMOTION_PRODUCT

การดำเนินการ	คำอธิบาย
GetPromotionProduct(PROMOTION_ID)	ค้นหาโปรโมชั่นจากรหัสโปรโมชั่นสินค้า
AddPromotionProduct	เพิ่มข้อมูลโปรโมชั่นสินค้า
UpdatePromotionProduct(PROMOTION_ID)	แก้ไขข้อมูลโปรโมชั่นสินค้า
DeletePromotionProduct(PROMOTION_ID)	ลบข้อมูลโปรโมชั่นสินค้า

ตารางที่ 3.4 การดำเนินการของคลาส PROMOTION_PRODUCT

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
PROM_SERV_ID	PROM_SERV_ID	รหัสโปรโมชั่นบริการ
PMSV_START_DATE	DATE	วันโปรโมชั่น
PMSV_AMOUNT	NUMBER(7, 2)	จำนวนเงินหรือเปอร์เซ็นต์ที่ลด
DISCOUNT_TYPE	VARCHAR2(10)	ประเภทส่วนลด
COST_TYPE	VARCHAR2(10)	ประเภทค่าใช้จ่าย

ตารางที่ 3.5 แอทริบิวต์ของคลาส PROMOTION_SERVICE

การดำเนินการ	คำอธิบาย
GetPromotionService(PROMOTION_ID)	ค้นหาโปรโมชั่นจากรหัสโปรโมชั่นบริการ
AddPromotionService	เพิ่มข้อมูลโปรโมชั่นบริการ
UpdatePromotionService(PROMOTION_ID)	แก้ไขข้อมูลโปรโมชั่นบริการ
การดำเนินการ	คำอธิบาย
DeletePromotionService(PROMOTION_ID)	ลบข้อมูลโปรโมชั่นบริการ

ตารางที่ 3.6 การดำเนินการของคลาส PROMOTION_SERVICE

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
PROD_ID	NUMBER(2,0)	รหัสสินค้า
PRODUCT	VARCHAR2(50)	ประเภทสินค้า
BRAND	VARCHAR2(50)	ยี่ห้อสินค้า
MODEL	VARCHAR2(50)	รุ่นสินค้า
PRICE	NUMBER(9,2)	ราคาสินค้า

ตารางที่ 3.7 แอทริบิวต์ของคลาส PRODUCT

การดำเนินการ	คำอธิบาย
GetProduct(PRODUCT_ID)	ค้นหาสินค้า
AddProduct	เพิ่มข้อมูลสินค้า
UpdateProduct(PROMOTION_ID)	แก้ไขข้อมูลสินค้า
DeleteProduct(PROMOTION_ID)	ลบข้อมูลสินค้า

ตารางที่ 3.8 การดำเนินการของคลาส PRODUCT

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
SERV_PL_ID	NUMBER(2, 0)	รหัสค่าใช้จ่าย
SERVICE_PRICE	NUMBER(7, 2)	ราคาค่าบริการ
START_DATE	DATE	วันที่เริ่มใช้ราคา
END_DATE	DATE	วันที่สิ้นสุดราคา
SPEED	VARCHAR2(10)	ความเร็ว

ตารางที่ 3.9 แอทริบิวต์ของคลาส SERVICE_PRICE

การดำเนินการ	คำอธิบาย
GetServicePrice	ค้นหารายการบริการ
AddServicePrice	เพิ่มรายการบริการพร้อมราคา
UpdateServicePrice	แก้ไขรายการบริการพร้อมราคา
DeleteServicePrice	ลบข้อมูลรายการบริการ

ตารางที่ 3.10 การดำเนินการของคลาส SERVICE_PRICE

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
SERVICE_TYPE_ID	NUMBER(2, 0)	รหัสประเภทค่าบริการ
SERVICE_TYPE_NAME	VARCHAR2(50)	ชื่อประเภทค่าบริการ

ตารางที่ 3.11 แอทริบิวต์ของคลาส SERVICE_TYPE

การดำเนินการ	คำอธิบาย
GetServiceType(SERVICE_TYPE_ID)	ค้นหาสินค้า
AddProduct	เพิ่มข้อมูลสินค้า
UpdateProduct(PROMOTION_ID)	แก้ไขข้อมูลสินค้า
DeleteProduct(PROMOTION_ID)	ลบข้อมูลสินค้า

๕

ตารางที่ 3.12 การดำเนินการของคลาส SERVICE_TYPE

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
FIRST_NAME	VARCHAR2(30)	ชื่อ
LAST_NAME	VARCHAR2(40)	นามสกุล

ตารางที่ 3.13 แอทริบิวต์ของคลาส PERSON

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
EMP_ID	NUMBER(7, 0) ,	รหัสพนักงาน
FIRST_NAME	VARCHAR2(40)	ชื่อ
LAST_NAME	VARCHAR2(40)	นามสกุล
TEL_NO	VARCHAR2(10)	โทรศัพท์
POSITION	VARCHAR2(20)	ตำแหน่ง

ตารางที่ 3.14 แอทริบิวต์ของคลาส EMPLOYEE

การดำเนินการ	คำอธิบาย
GetEmployee(EMP_ID)	ค้นหาพนักงาน
AddEmployee	เพิ่มข้อมูลพนักงาน
UpdateEmployee(EMP_ID)	แก้ไขข้อมูลพนักงาน
DeleteEmployee(EMP_ID)	ลบข้อมูลพนักงาน

ตารางที่ 3.15 การดำเนินการของคลาส EMPLOYEE

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
CUST_ID	NUMBER(7, 0)	รหัสลูกค้า
BUSINESS_TYPE	VARCHAR2(10)	ประเภทธุรกิจ

ตารางที่ 3.16 แอทริบิวต์ของคลาส CUSTOMER

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
ADSL_CUST_SEXP_ID	NUMBER(7, 0)	รหัสค่าใช้จ่ายของลูกค้า
COST_TYPE	VARCHAR2(10)	ประเภทค่าใช้จ่าย
ADSL_CUST_SEXP_PRICE	NUMBER(9, 2)	ราคาค่าบริการ
ADSL_CUST_SEXP_PROMOTION	NUMBER(9, 2)	ราคาโปรโมชั่น
ADSL_CUST_SEXP_REMARK	VARCHAR2(100)	หมายเหตุ
ADSL_CUST_SEXP_START_PROMOTION	DATE	วันที่เริ่มโปรโมชั่น
ADSL_CUST_SEXP_STOP_PROMOTION	DATE	วันที่สิ้นสุดโปรโมชั่น

ตารางที่ 3.17 แอทริบิวต์ของคลาส CUSTOMER_SERVICE_EXPENSE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การดำเนินการ	คำอธิบาย
GetCustomerServiceExpense(ADSL_CUST_SEXP_ID)	ค้นหาค่าใช้จ่ายของลูกค้า
AddCustomerServiceExpense	เพิ่มข้อมูลค่าใช้จ่ายของลูกค้า
UpdateCustomerServiceExpense (ADSL_CUST_SEXP_ID)	แก้ไขข้อมูลค่าใช้จ่ายของลูกค้า
DeleteCustomerServiceExpense (ADSL_CUST_SEXP_ID)	ลบข้อมูลค่าใช้จ่ายของลูกค้า

ตารางที่ 3.18 การดำเนินการของคลาส CUSTOMER_SERVICE_EXPENSE

แอททริบิวต์	ชนิดข้อมูล	คำอธิบาย
FORM_DATE	DATE	วันที่
TAX_INV_NO	VARCHAR2(10)	หมายเลขใบกำกับภาษี
DISCOUNT	NUMBER(9, 2)	ส่วนลด
NET_AMOUNT	NUMBER(9, 2)	ค่าบริการ
VAT	NUMBER(5, 2)	ภาษีมูลค่าเพิ่ม
TOTAL_AMOUNT	NUMBER(9, 2)	จำนวนเงินที่ต้องชำระ
REMARK	VARCHAR2(100)	หมายเหตุ

ตารางที่ 3.19 แอททริบิวต์ของคลาส FORM

แอททริบิวต์	ชนิดข้อมูล	คำอธิบาย
QUANTITY	NUMBER(4,0)	จำนวน
UNITPRICE	NUMBER(9,2)	ราคาต่อหน่วย
AMOUNT	NUMBER(9,2)	จำนวนเงิน

ตารางที่ 3.20 แอททริบิวต์ของคลาส FORM_DETAIL

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
INV_NO	VARCHAR2(15)	เลขที่ใบแจ้งหนี้
TAX_INV_NO	VARCHAR2(15)	เลขที่ใบกำกับภาษี
FORM_DATE	DATE	วันที่
PAYMENT_STATUS	VARCHAR2(1)	สถานะการจ่ายเงิน
DOC_STATUS	VARCHAR2(1)	สถานะของเอกสาร
GROSS_TOTAL	NUMBER(9,2)	จำนวนเงินรวม
DISCOUNT	NUMBER(9,2)	ส่วนลด
NET_AMOUNT	NUMBER(9,2)	ค่าบริการ
VAT	NUMBER(5,2)	ภาษีมูลค่าเพิ่ม
TOTAL_AMOUNT	NUMBER(9,2)	จำนวนเงินที่ต้องชำระ
REMARK	VARCHAR2(100)	หมายเหตุ

ตารางที่ 3.21 แอทริบิวต์ของคลาส INVOICE

การดำเนินการ	คำอธิบาย
AddInvoice	เพิ่มข้อมูลของใบแจ้งหนี้เก็บในฐานข้อมูล
UpdateInvoice(INV_NO)	แก้ไขข้อมูลของใบแจ้งหนี้
GetInvoice(INV_NO)	ค้นหาข้อมูลใบแจ้งหนี้ที่ระบุ
SetInvoicePaymentStatus(INV_NO)	กำหนดสถานะการจ่ายเงินของใบแจ้งหนี้ ซึ่งมีสถานะดังต่อไปนี้ คือ 1. จ่ายเงินแล้ว 2. จ่ายบางส่วน 3. ชยกเลิก
GetInvoicePaymentStatus(INV_NO)	อ่านสถานะการจ่ายเงินของใบแจ้งหนี้
SetInvoiceDocStatus(INV_NO)	กำหนดสถานะของใบแจ้งหนี้ ซึ่งมีสถานะต่างๆ ดังต่อไปนี้ คือ 1. สร้างใหม่ 2. ชยกเลิก
GenInvoiceNo	สร้างหมายเลขใบแจ้งหนี้

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
INV_SEQ	NUMBER(2,0)	ลำดับที่
QUANTITY	NUMBER(4,0)	จำนวน
UNITPRICE	NUMBER(7,2)	ราคาต่อหน่วย
AMOUNT	NUMBER(9,2)	จำนวนเงิน

ตารางที่ 3.23 แอทริบิวต์ของคลาส INVOICE_DETAIL

การดำเนินการ	คำอธิบาย
AddInvoiceDetail	เพิ่มข้อมูลรายละเอียดแจ้งหนี้เก็บในฐานข้อมูล
UpdateInvoiceDetail	แก้ไขข้อมูลรายละเอียดใบแจ้งหนี้เก็บในฐานข้อมูล
GetInvoiceDetail	ค้นหารายละเอียดใบแจ้งหนี้

ตารางที่ 3.24 การดำเนินการของคลาส INVOICE_DETAIL

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
REC_NO	VARCHAR2(15)	เลขที่ใบเสร็จรับเงิน
FORM_DATE	DATE	วันที่
TAX_INV_NO	VARCHAR2(15)	เลขที่ใบกำกับภาษี
PAYMENT_STATUS	VARCHAR2(1)	สถานะการจ่ายเงิน
DOC_STATUS	VARCHAR2(1)	สถานะของเอกสาร
GROSS_TOTAL	NUMBER(9,2)	จำนวนเงินรวม
DISCOUNT	NUMBER(5,2)	ส่วนลด
NET_AMOUNT	NUMBER(9,2)	รวมค่าบริการ
VAT	NUMBER(4,2)	ภาษีมูลค่าเพิ่ม
TOTAL_AMOUNT	NUMBER(9,2)	จำนวนเงินที่ต้องชำระ
PAID_BY	VARCHAR2(50)	ชำระเงินโดย
PAID_DATE	DATE	วันที่
TRANSFER_BANK	VARCHAR2(50)	ธนาคารที่รับชำระเงิน
TRANSFER_DATE	DATE	วันที่ชำระเงิน
TRANSFER_AMOUNT	NUMBER(9,2)	จำนวนเงินที่ชำระ
REMARK	VARCHAR2(100)	หมายเหตุ

ตารางที่ 3.25 แอทริบิวต์ของคลาส RECEIPT

การดำเนินการ	คำอธิบาย
AddReceipt	เพิ่มข้อมูลของใบเสร็จรับเงินเก็บในฐานข้อมูล
UpdateReceipt(REC_NO)	แก้ไขข้อมูลของใบเสร็จรับเงิน
GetReceipt(REC_NO)	ค้นหาข้อมูลใบเสร็จรับเงินที่ระบุ
SetReceiptPaymentStatus(REC_NO)	กำหนดสถานะการจ่ายเงินของใบเสร็จรับเงินซึ่งมีสถานะดังต่อไปนี้ คือ 1. จ่ายเงินแล้ว 2. จ่ายบางส่วน 3. ยกเลิก
GetReceiptPaymentStatus(REC_NO)	อ่านสถานะการจ่ายเงินของใบเสร็จรับเงิน
SetReceiptDocStatus(REC_NO)	กำหนดสถานะของใบเสร็จรับเงินซึ่งมีสถานะต่างๆ ดังต่อไปนี้ คือ 1. สร้างใหม่ 2. ยกเลิก
GenReceiptNo	สร้างหมายเลขใบเสร็จรับเงิน

ตารางที่ 3.26 การดำเนินการของคลาส RECEIPT

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
REC_SEQ	NUMBER(2,0)	ลำดับที่
QUANTITY	NUMBER(4,0)	จำนวน
UNITPRICE	NUMBER(7,2)	ราคาต่อหน่วย
AMOUNT	NUMBER(9,2)	จำนวนเงิน

ตารางที่ 3.27 แอทริบิวต์ของคลาส RECEIPT_DETAIL

การดำเนินการ	คำอธิบาย
AddReceiptDetail	เพิ่มข้อมูลรายละเอียดใบเสร็จรับเงินเก็บในฐานข้อมูล
UpdateReceiptDetail	แก้ไขข้อมูลรายละเอียดใบเสร็จรับเงินเก็บในฐานข้อมูล
GetReceiptDetail	ค้นหารายละเอียดใบเสร็จรับเงิน

ตารางที่ 3.28 การดำเนินการของคลาส RECEIPT_DETAIL

แอททริบิวต์	ชนิดข้อมูล	คำอธิบาย
BANK_ID	NUMBER(2,0)	รหัสธนาคาร
BANK_NAME	VARCHAR2(50)	ชื่อธนาคาร

ตารางที่ 3.29 แอททริบิวต์ของคลาส BANK

การดำเนินการ	คำอธิบาย
AddBank	เพิ่มข้อมูลธนาคาร
UpdateBank(BANK_ID)	แก้ไขข้อมูลธนาคาร
การดำเนินการ	คำอธิบาย
GetBank(BANK_ID)	ค้นหารายชื่อธนาคาร
DeleteBank(BANK_ID)	ลบธนาคาร

ตารางที่ 3.30 การดำเนินการของคลาส BANK

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
BRANCH_BANK_ID	NUMBER(2,0)	รหัสสาขานาการ
BRANCH_BANK_NAME	VARCHAR2(50)	ชื่อสาขานาการ

ตารางที่ 3.31 แอทริบิวต์ของคลาส BRANCH_BANK

การดำเนินการ	คำอธิบาย
AddBranchBank	เพิ่มข้อมูลสาขานาการ
UpdateBranchBank(BANK_ID)	แก้ไขข้อมูลสาขานาการ
GetBranchBank(BANK_ID)	ค้นหารายชื่อสาขานาการ
DeleteBranchBank(BANK_ID)	ลบสาขานาการ

ตารางที่ 3.32 การดำเนินการของคลาส BRANCH_BANK

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
APP_ID	VARCHAR2(10)	หมายเลขใบสมัคร
ADSL_NUMBER	VARCHAR2(20)	หมายเลข ADSL
ADSL_INSERVICE_DATE	DATE	วันที่เริ่มให้บริการ
USB_PORT	BOOLEAN	มี พอร์ต USB
ETHERNET_LAN_CARD	BOOLEAN	มี LAN CARD
CPU_TYPE	VARCHAR2(10)	ประเภท CPU
SPEED	VARCHAR2(10)	ความเร็ว

ตารางที่ 3.33 แอทริบิวต์ของคลาส APPLICATION

การดำเนินการ	คำอธิบาย
AddApplication	เพิ่มข้อมูลใบสมัคร
UpdateBranchBank(BANK_ID)	แก้ไขข้อมูลสาขาธนาคาร
การดำเนินการ	คำอธิบาย
GetBranchBank(BANK_ID)	ค้นหารายชื่อสาขาธนาคาร
DeleteBranchBank(BANK_ID)	ลบสาขาธนาคาร

ตารางที่ 3.34 การดำเนินการของคลาส APPLICATION

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
ADDR_ID	NUMBER(2,0)	รหัสที่อยู่
ADDR_NO	VARCHAR2(45)	เลขที่
MOO	VARCHAR2(10)	หมู่ที่
BLD_NAME	VARCHAR2(100)	ชื่ออาคาร
FLOOR	VARCHAR2(10)	ชั้นที่
SOI	VARCHAR2(50)	ซอย
ROAD_NAME	VARCHAR2(50)	ถนน
THUMBON	VARCHAR2(50)	ตำบล
AMPHUR	VARCHAR2(50)	อำเภอ
ZIP_CODE	VARCHAR2(15)	รหัสไปรษณีย์
TEL	VARCHAR2(20)	โทรศัพท์
MOBILE	VARCHAR2(20)	โทรศัพท์เคลื่อนที่
BUSINESS_TYPE	VARCHAR2(10)	ประเภทธุรกิจ

ตารางที่ 3.35 แอทริบิวต์ของคลาส ADDRESS

การดำเนินการ	คำอธิบาย
AddAddress	เพิ่มข้อมูลที่อยู่
UpdateAddress (ADDR_ID)	แก้ไขข้อมูลที่อยู่
GetAddress (ADDR_ID)	ค้นหารายชื่อที่อยู่
DeleteAddress (ADDR_ID)	ลบที่อยู่

ตารางที่ 3.36 การดำเนินการของคลาส ADDRESS

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
DN_NO	VARCHAR2(15)	รหัสเดบิต โนต
DN_TAX_NO	VARCHAR2(15)	หมายเลขใบกำกับภาษี
DN_DATE	DATE	วันที่
DN_TOTAL_AMT	NUMBER(9,2)	รวมมูลค่าสินค้าตามใบกำกับภาษีเดิม
DN_TRUE_PRICE	NUMBER(9,2)	มูลค่าที่ถูกต้อง
DN_DIFF_PRICE	NUMBER(7,2)	ผลต่าง
DN_VAT	NUMBER(7,2)	ภาษีมูลค่าเพิ่ม
DN_TOTAL	NUMBER(9,2)	รวม
DOC_STATUS	VARCHAR2(1)	สถานะใบเดบิต โนต

ตารางที่ 3.37 แอทริบิวต์ของคลาส DEBIT_NOTE

การดำเนินการ	คำอธิบาย
AddDebitNote	เพิ่มข้อมูลของใบ Debit Note เก็บในฐานข้อมูล
UpdateDebitNote (DN_NO)	แก้ไขข้อมูลของใบ Debit Note
GetDebitNote (DN_NO)	ค้นหาข้อมูลใบ Debit Note ที่ระบุ
GetReceiptPaymentStatus(DN_NO)	อ่านสถานะการจ่ายเงินของใบ Debit Note
SetDebitNoteDocStatus(DN_NO)	กำหนดสถานะของใบ Debit Note ซึ่งมีสถานะต่างๆ ดังต่อไปนี้ คือ 1. สร้างใหม่ 2. ยกเลิก
GenDNNo	สร้างหมายเลขใบ Debit Note

ตารางที่ 3.38 การดำเนินการของคลาส DEBIT_NOTE

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
DND_SEQ	NUMBER(2,0)	ลำดับที่
QUANTITY	NUMBER(4,2)	ปริมาณ
UNITPRICE	NUMBER(7,2)	ราคาต่อหน่วย
AMOUNT	NUMBER(9,2)	จำนวนเงิน
DND_DETAIL	VARCHAR2(30)	รายการ

ตารางที่ 3.39 แอทริบิวต์ของคลาส DEDIT_NOTE_DETAIL

การดำเนินการ	คำอธิบาย
AddDebitNoteDetail	เพิ่มข้อมูลรายละเอียดใบ Debit Note เก็บในฐานข้อมูล
UpdateDebitNoteDetail	แก้ไขข้อมูลรายละเอียดใบ Debit Note เก็บในฐานข้อมูล
GetDebitNoteDetail	ค้นหารายละเอียดใบ Debit Note

แอทริบิวต์	ชนิดข้อมูล	คำอธิบาย
CN_NO	VARCHAR2(15)	เลขที่ใบลดหนี้
CN_TAX_NO	NUMBER(9,2)	เลขที่ใบกำกับภาษี
CN_DATE	DATE	วันที่
CN_TOTAL_AMT	NUMBER(9,2)	รวมมูลค่าสินค้าตามใบกำกับภาษีเดิม
CN_TRUE_PRICE	NUMBER(9,2)	มูลค่าที่ถูกต้อง
CN_DIFF_PRICE	NUMBER(9,2)	ผลต่าง
CN_TOTAL	NUMBER(9,2)	รวมภาษีมูลค่าเพิ่มตามใบกำกับภาษีเดิม

ตารางที่ 3.41 แอทริบิวต์ของคลาส CREDIT_NOTE

การดำเนินการ	คำอธิบาย
AddCreditNote	เพิ่มข้อมูลของใบ Credit Note เก็บในฐานข้อมูล
UpdateCreditNote (CN_NO)	แก้ไขข้อมูลของใบ Credit Note
การดำเนินการ	คำอธิบาย
GetCreditNote (CN_NO)	ค้นหาข้อมูลใบ Credit Note ที่ระบุ
SetCreditNoteDocStatus(CN_NO)	กำหนดสถานะของใบ Credit Note ซึ่งมีสถานะต่างๆ ดังต่อไปนี้ คือ 1. สร้างใหม่ 2. ยกเลิก
GenCNNo	สร้างหมายเลขใบ Credit Note

ตารางที่ 3.42 การดำเนินการของคลาส CREDIT_NOTE

การดำเนินการ	คำอธิบาย
ADD_APPLICATION	เพิ่มข้อมูลใบสมัครลงในฐานข้อมูล
EDIT_APPLICATION	แก้ไขข้อมูลใบสมัครลงในฐานข้อมูล
DELETE_APPLICATION	ลบข้อมูลใบสมัครจากฐานข้อมูล
SAVE_APPLICATION	บันทึกข้อมูลใบสมัครลงฐานข้อมูล
GET_APPLICATION	ดึงข้อมูลใบสมัครจากฐานข้อมูล
ADD_EMPLOYEE	เพิ่มข้อมูลพนักงานลงฐานข้อมูล
EDIT_EMPLOYEE	แก้ไขข้อมูลพนักงานลงฐานข้อมูล
DELETE_EMPLOYEE	ลบข้อมูลพนักงานจากฐานข้อมูล
SAVE_EMPLOYEE	บันทึกข้อมูลพนักงานลงฐานข้อมูล
GET_EMPLOYEE	ดึงข้อมูลลูกค้าจากฐานข้อมูล
ADD_CUSTOMER	เพิ่มข้อมูลลูกค้าลงฐานข้อมูล
EDIT_CUSTOMER	แก้ไขข้อมูลลูกค้าฐานข้อมูล
DELETE_CUSTOMER	ลบข้อมูลลูกค้าจากฐานข้อมูล
GET_CUSTOMER	ดึงข้อมูลลูกค้าจากฐานข้อมูล
SAVE_CUSTOMER	บันทึกข้อมูลลูกค้าลงฐานข้อมูล
ADD_ADDRESS	เพิ่มข้อมูลที่อยู่ลงฐานข้อมูล
EDIT_ADDRESS	แก้ไขข้อมูลที่อยู่ลงฐานข้อมูล
DELETE_ADDRESS	ลบข้อมูลที่อยู่จากฐานข้อมูล
SAVE_ADDRESS	บันทึกข้อมูลที่อยู่ลงฐานข้อมูล
GET_ADDRESS	ดึงข้อมูลที่อยู่จากฐานข้อมูล
ADD_ISP	เพิ่มข้อมูล ISP ลงฐานข้อมูล
EDIT_ISP	แก้ไข ข้อมูล ISP ลงฐานข้อมูล
DELETE_ISP	ลบข้อมูล ข้อมูล ISP จากฐานข้อมูล
SAVE_ISP	บันทึกข้อมูล ข้อมูล ISP ลงฐานข้อมูล
GET_ISP	ดึง ข้อมูล ISP จากฐานข้อมูล
ADD_PRODUCT	เพิ่มข้อมูลสินค้าลงฐานข้อมูล
EDIT_PRODUCT	แก้ไขข้อมูลสินค้าลงฐานข้อมูล
DELETE_PRODUCT	ลบข้อมูลสินค้าจากฐานข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้เผยแพร่ไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การดำเนินการ	คำอธิบาย
SAVE_PRODUCT	บันทึกข้อมูลสินค้าลงฐานข้อมูล
GET_PRODUCT	ดึงข้อมูลสินค้าจากฐานข้อมูล
ADD_PROMOTION	เพิ่มข้อมูลโปรโมชั่นลงฐานข้อมูล
EDIT_PROMOTION	แก้ไขข้อมูลโปรโมชั่นลงฐานข้อมูล
SAVE_PROMOTION	บันทึกโปรโมชั่นลงฐานข้อมูล
GET_PROMOTION	ดึงข้อมูลโปรโมชั่นจากฐานข้อมูล
ADD_PROMOTION_PRODUCT	เพิ่มข้อมูลโปรโมชั่นสินค้าลงฐานข้อมูล
EDIT_PROMOTION_PRODUCT	แก้ไขข้อมูลโปรโมชั่นสินค้าลงฐานข้อมูล
DELETE_PROMOTION_PRODUCT	ลบข้อมูลโปรโมชั่นสินค้าลงฐานข้อมูล
SAVE_PROMOTION_PRODUCT	บันทึกข้อมูลโปรโมชั่นสินค้าลงฐานข้อมูล
GET_PROMOTION_PRODUCT	ดึงข้อมูลโปรโมชั่นจากฐานข้อมูล
ADD_PROMOTION_SERVICE	เพิ่มข้อมูลโปรโมชั่นบริการจากฐานข้อมูล
EDIT_PROMOTION_SERVICE	แก้ไขข้อมูลโปรโมชั่นบริการจากฐานข้อมูล
SAVE_PROMOTION_SERVICE	บันทึกข้อมูลโปรโมชั่นบริการจากฐานข้อมูล
GET_PROMOTION_SERVICE	ดึงข้อมูลโปรโมชั่นบริการจากฐานข้อมูล
ADD_INVOICE	เพิ่มข้อมูล INVOICE ลงฐานข้อมูล
EDIT_INVOICE	แก้ไขข้อมูล INVOICE ลงฐานข้อมูล
DELETE_INVOICE	ลบข้อมูล INVOICE จากฐานข้อมูล
SAVE_INVOICE	บันทึกข้อมูล INVOICE ลงฐานข้อมูล
GET_INVOICE	ดึงข้อมูล INVOICE จากฐานข้อมูล
ADD_INVOICE_DETAIL	เพิ่มข้อมูล INVOICE_DETAIL ลงฐานข้อมูล
EDIT_INVOICE_DETAIL	แก้ไขข้อมูล INVOICE_DETAIL ลงฐานข้อมูล
DELETE_INVOICE_DETAIL	ลบข้อมูล INVOICE_DETAIL จากฐานข้อมูล
SAVE_INVOICE_DETAIL	บันทึกข้อมูล INVOICE_DETAIL ลงฐานข้อมูล
GET_INVOICE_DETAIL	ดึงข้อมูล INVOICE_DETAIL จากฐานข้อมูล
ADD_RECEIPT	เพิ่มข้อมูล RECEIPT ลงฐานข้อมูล
EDIT_RECEIPT	แก้ไขข้อมูล RECEIPT ลงฐานข้อมูล
DELETE_RECEIPT	ลบข้อมูล RECEIPT จากฐานข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การดำเนินการ	คำอธิบาย
SAVE_RECEIPT	บันทึกข้อมูล RECEIPT ลงฐานข้อมูล
GET_RECEIPT	ดึงข้อมูล GET_RECEIPT จากฐานข้อมูล
ADD_RECEIPT_DETAIL	เพิ่มข้อมูล RECEIPT_DETAIL ลงฐานข้อมูล
EDIT_RECEIPT_DETAIL	แก้ไขข้อมูล RECEIPT_DETAIL ลงฐานข้อมูล
DELETE_RECEIPT_DETAIL	ลบข้อมูล RECEIPT_DETAIL จากฐานข้อมูล
SAVE_RECEIPT_DETAIL	บันทึกข้อมูล RECEIPT_DETAIL ลงฐานข้อมูล
GET_RECEIPT_DETAIL	ดึงข้อมูล GET_RECEIPT จากฐานข้อมูล
ADD_DEBIT_NOTE	เพิ่มข้อมูล DEBIT_NOTE ลงฐานข้อมูล
EDIT_DEBIT_NOTE	แก้ไขข้อมูล DEBIT_NOTE ลงฐานข้อมูล
DELETE_DEBIT_NOTE	ลบข้อมูล DEBIT_NOTE จากฐานข้อมูล
SAVE_DEBIT_NOTE	บันทึกข้อมูล DEBIT_NOTE ลงฐานข้อมูล
GET_DEBIT_NOTE	ดึงข้อมูล DEBIT_NOTE จากฐานข้อมูล
ADD_DEBIT_NOTE_DETAIL	เพิ่มข้อมูล DEBIT_NOTE_DETAIL ลงฐานข้อมูล
EDIT_DEBIT_NOTE_DETAIL	แก้ไขข้อมูล DEBIT_NOTE_DETAIL ลงฐานข้อมูล
DELETE_DEBIT_NOTE_DETAIL	ลบข้อมูล DEBIT_NOTE_DETAIL จากฐานข้อมูล
SAVE_DEBIT_NOTE_DETAIL	บันทึกข้อมูล DEBIT_NOTE_DETAIL ลงฐานข้อมูล
GET_DEBIT_NOTE_DETAIL	ดึงข้อมูล DEBIT_NOTE_DETAIL จากฐานข้อมูล
ADD_CREDIT_NOTE	เพิ่มข้อมูล CREDIT_NOTE ลงฐานข้อมูล
EDIT_CREDIT_NOTE	แก้ไขข้อมูล CREDIT_NOTE ลงฐานข้อมูล
DELETE_CREDIT_NOTE	ลบข้อมูล CREDIT_NOTE จากฐานข้อมูล
SAVE_CREDIT_NOTE	บันทึกข้อมูล CREDIT_NOTE ลงฐานข้อมูล
GET_CREDIT_NOTE	ดึงข้อมูล CREDIT_NOTE จากฐานข้อมูล
ADD_CREDIT_NOTE_DETAIL	เพิ่มข้อมูล CREDIT_NOTE_DETAIL ลงฐานข้อมูล

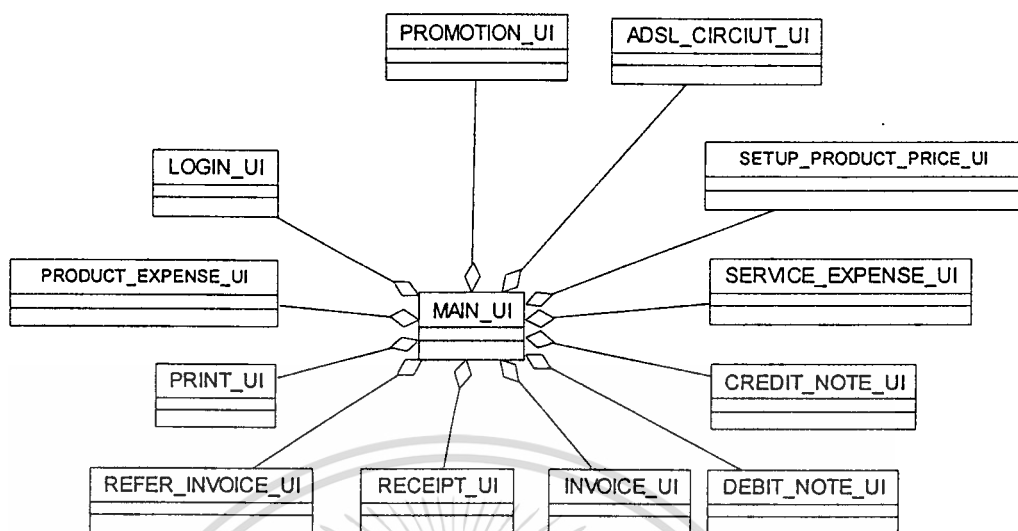
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การดำเนินการ	คำอธิบาย
EDIT_CREDIT_NOTE_DETAIL	แก้ไขข้อมูล CREDIT_NOTE_DETAIL ลงฐานข้อมูล
DELETE_CREDIT_NOTE_DETAIL	ลบข้อมูล CREDIT_NOTE_DETAIL จากฐานข้อมูล
SAVE_CREDIT_NOTE_DETAIL	บันทึกข้อมูล CREDIT_NOTE_DETAIL ลงฐานข้อมูล
GET_CREDIT_NOTE_DETAIL	ดึงข้อมูล CREDIT_NOTE_DETAIL จากฐานข้อมูล



ตารางที่ 3.43 การดำเนินการของDB



รูปที่ 3.21 Class Diagram สำหรับส่วนติดต่อกับผู้ใช้งาน

การออกแบบ View Layer

เป็นการออกแบบ Class ที่ทำหน้าที่เป็น Interface ของระบบ ซึ่งเป็น Interface ระหว่างผู้ใช้งานระบบ หรือเป็น User Interface ของระบบ และนำ Class เหล่านั้นมาพัฒนาต่อ ซึ่งออกมาในรูปแบบของ Graphic User Interface (GUI) ซึ่งประกอบไปด้วย Class ดังต่อไปนี้ คือ

1.MAIN_UI

เป็น User Interface ที่ทำหน้าที่ในการเป็นหน้าจอหลักเพื่อเชื่อมต่อกับหน้าจออื่นๆ

2. LOGIN_UI

เป็น User Interface ที่ทำหน้าที่ในการจัดการตรวจสอบให้ผู้ที่มิสิทธิ์ในการเข้าใช้งาน เข้าใช้งานระบบได้เท่านั้น

3. PROMOTION_UI

เป็น User Interface ที่ทำหน้าที่ในการจัดทำโปรโมชั่น

4. ADSL_CIRCIUT_UI

เป็น User Interface ที่ทำหน้าที่ในการเพิ่ม แก้ไข หรือดูรายละเอียดข้อมูลวงจร ADSL ของลูกค้า

5. SETUP_PRODUCT_PRICE_UI

เป็น User Interface ที่ทำหน้าที่ในการจัดการตั้งราคาสินค้า

6. SERVICE_EXPENSE_UI

เป็น User Interface ที่ทำหน้าที่ในการจัดการข้อมูลประเภทค่าใช้จ่ายเกี่ยวกับบริการ

7. PRODUCT_EXPENSE_UI

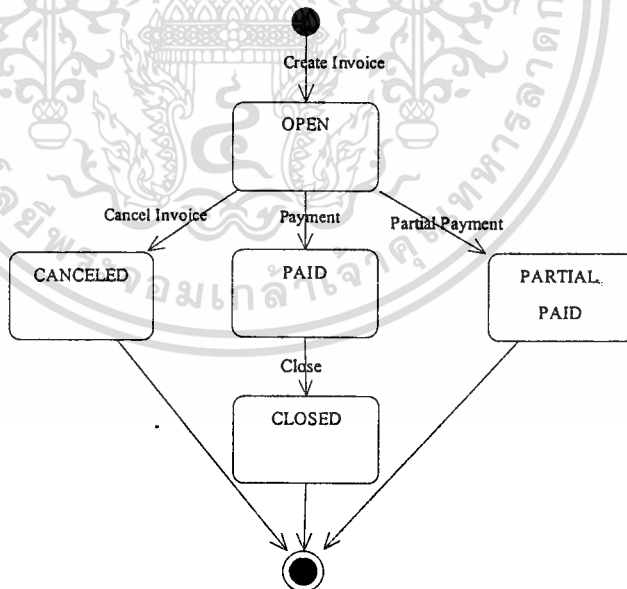
เป็น User Interface ที่ทำหน้าที่ในการเพิ่ม แก้ไข หรือดูรายละเอียดข้อมูลสินค้า

3.3.4.5 State Diagram

ใช้อธิบายสถานะของแต่ละออเบจกต์เมื่อมีการกระทำบางอย่างเกิดขึ้น

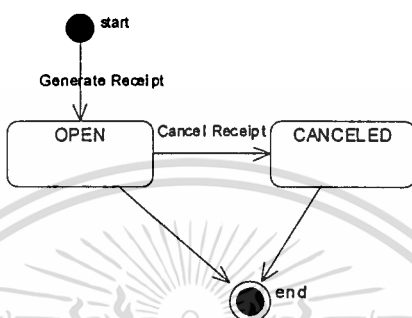
แสดง State Diagram ของ Invoice

จากรูปที่ 3.45 เมื่อมีค่าใช้จ่ายเกิดขึ้นประจำเดือนนั้น ระบบจะทำการสร้างใบเรียกเก็บเงินขึ้นมาใหม่ซึ่งสถานะในตอนแรกคือ “OPEN” หมายถึงการเปิดใบเรียกเก็บเงินใหม่เพื่อออกให้กับลูกค้า ในกรณีที่ไม่ต้องการเรียกเก็บเงินลูกค้าแล้วเราก็จะทำการยกเลิกใบเรียกเก็บเงินได้ซึ่งมีผลทำให้สถานะของใบเรียกเก็บเงินเป็น “CANCELED” แต่เมื่อลูกค้ามีการจ่ายเงินเข้ามาเต็มจำนวนก็จะมีสถานะเป็น “PAID” หมายถึงมีการจ่ายเงินแล้วและหลังจากนั้นเมื่อยอดเงินลงตัว ใบเรียกเก็บเงินใบนั้นๆก็จะถูกปิดซึ่งมีผลให้สถานะของใบเรียกเก็บเงินเป็น “CLOSE”



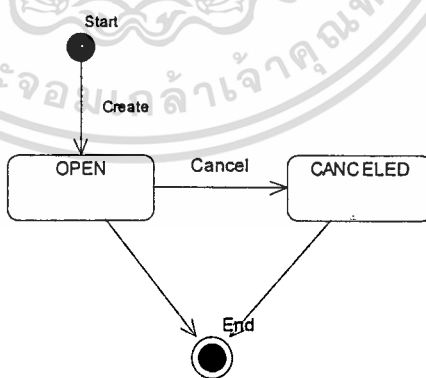
รูปที่ 3.22 State Diagram สำหรับออเบจกต์ INVOICE

จากรูปที่ 4.36 เมื่อออกใบแจ้งหนี้ให้กับลูกค้าแล้ว และลูกค้าเข้ามาชำระเงินระบบก็จะทำการออกใบเสร็จรับเงินให้กับลูกค้า ซึ่งในการสร้างใบเสร็จรับเงินในครั้งแรกใบเสร็จรับเงินจะมีสถานะเป็น “OPEN” และถ้าส่งให้กับลูกค้าเรียบร้อยแล้วก็เป็นการจบขั้นตอน ส่วนในกรณีที่ต้องทำการยกเลิกการออกใบเสร็จรับเงินสถานะของใบเสร็จรับเงินก็จะเปลี่ยนเป็น “CANCELED”



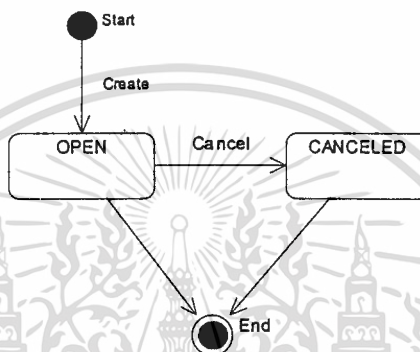
รูปที่ 3.23 State Diagram สำหรับ ขอบเจ็ท RECEIPT

จากรูปที่ 3.47 แสดงสถานะของใบลดหนี้เมื่อระบบทำการออกใบลดหนี้ให้กับลูกค้าในการสร้างใบลดหนี้สถานะคือ “OPEN” และถ้าส่งออกไปให้ลูกค้าแล้วก็เป็นจบขั้นตอน ส่วนในกรณีที่มีการยกเลิกใบลดหนี้สถานะก็จะเปลี่ยนเป็น “CANCELED”



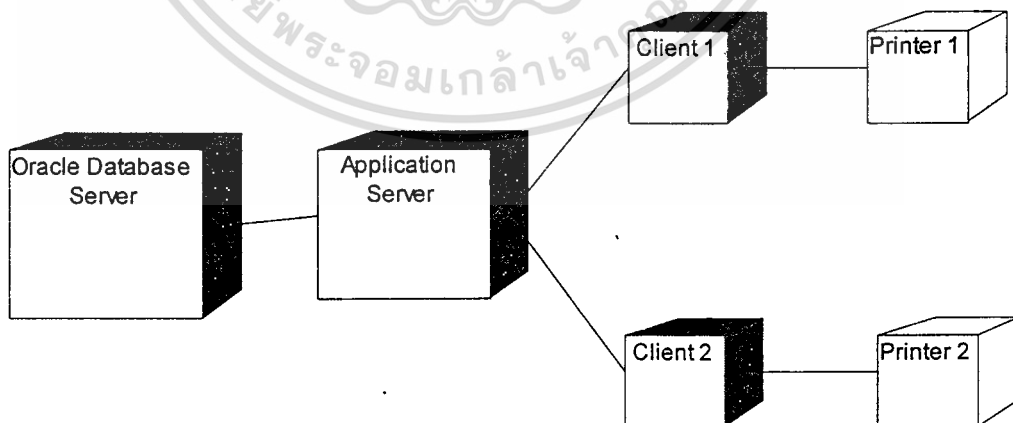
รูปที่ 3.24 State Diagram สำหรับ ขอบเจ็ท CREDIT_NOTE

จากรูปที่ 3.48 แสดงสถานะของใบเพิ่มหนี้เมื่อระบบทำการออกใบเพิ่มหนี้ให้กับลูกค้าในการสร้างใบเพิ่มหนี้สถานะคือ “OPEN” และถ้าส่งออกไปให้ลูกค้าแล้วก็เป็นกรจบขั้นตอน ส่วนในกรณีที่มีการยกเลิกใบเพิ่มหนี้สถานะก็จะเป็น “CANCELED”



รูปที่ 3.25 State Diagram สำหรับ ออบเจ็กต์ DEBIT_NOTE

3.3.4.6 Deployment Diagram



รูปที่ 3.26 Deployment Diagram ระบบคิดเงินลูกค้า

บทที่ 4

การสร้างระบบ

ในการออกแบบระบบนั้น ได้ออกแบบระบบให้มีความทำงานเป็นแบบ Client/Server ซึ่งโดยรวมแล้ว คลาสที่ออกแบบจะแบ่งออกเป็น 3 ประเภท คือ

1. คลาสส่วนที่เป็น View Layer คือส่วนที่ทำหน้าที่แสดง และติดต่อกับผู้ใช้โดยตรง
2. คลาสส่วนที่เป็น Business Layer ซึ่งตัวโปรแกรมจะถูกพัฒนาไว้ที่ Client
3. ส่วนที่เกี่ยวกับ Access Layer ซึ่งมีหน้าที่ในการติดต่อกับฐานข้อมูล

4.1 เครื่องมือที่ใช้

ในการสร้างระบบได้ใช้เครื่องมือดังต่อไปนี้

1. Rational Rose 2000 ใช้เขียนแบบจำลองเพื่อใช้ในการวิเคราะห์และออกแบบระบบ โดยใช้ในการวาดไดอะแกรมต่างๆ และยังใช้เป็นเครื่องมือช่วยในการสร้างฐานข้อมูล โดยการใช้เครื่องมือในส่วนของการแปลงจากคลาสดิอะแกรมไปเป็นภาษา DDL ซึ่งเป็นภาษาสคริปต์สำหรับการสร้างฐานข้อมูลสำหรับใช้ในระบบงาน
2. ฐานข้อมูลของระบบใช้ Oracle 8 ซึ่งเป็นการใช้งานในลักษณะของฐานข้อมูลเชิงสัมพันธ์
3. ในส่วนของการพัฒนาแอปพลิเคชัน ซึ่งเป็นส่วนติดต่อกับผู้ใช้ นั้นได้เลือกใช้ Delphi 4.0 ซึ่งมีคุณสมบัติดังนี้ คือ

3.1 สนับสนุนการเขียนโปรแกรมแบบ Object Oriented Programming

3.2 มีส่วนที่ช่วยอำนวยความสะดวกในการสร้างส่วนติดต่อผู้ใช้ในลักษณะการลากและ

3.3 มีส่วนช่วยในการสร้างโค้ดที่เกี่ยวข้องให้อัตโนมัติ ทำให้ลดภาระในการสร้างโค้ดของผู้พัฒนาระบบ

4.2 การสร้างฐานข้อมูล

จากการออกแบบในส่วนของ Access Layer ไว้ซึ่งทำงานติดต่อกับฐานข้อมูล ซึ่งมีการทำงานบางส่วนอยู่ที่ฝั่ง Server โดยมีวิธีการเตรียมดังต่อไปนี้

4.2.1 ติดตั้งฐานข้อมูล Oracle 8.0

4.2.2 ทำการสร้างฐานข้อมูลเตรียมไว้สำหรับรองรับการสร้างตารางข้อมูลเชิงสัมพันธ์

4.2.3 สร้าง User ของฐานข้อมูล พร้อมทั้งกำหนดสิทธิ์ต่างๆ ให้กับ User

4.2.3 กำหนดทางเชื่อมต่อ (Host String) เพื่อเชื่อมต่อกับฐานข้อมูล ซึ่งใช้ Net Easy Config ช่วยในการกำหนด

4.3 ขั้นตอนการใช้เครื่องมือของ Rational Rose 2000

ในการสร้างฐานข้อมูลเชิงสัมพันธ์โดยการใช้ เครื่องมือ DDL Code Generation โดยที่ถ้าเรามีคลาสซึ่งต้องการจัดเก็บลงฐานข้อมูล ซึ่ง Rational Rose 2000 สามารถสร้างภาษา Structured Query Language (SQL) ซึ่งเป็นภาษาที่ใช้ในการสร้างฐานข้อมูล โดยที่แต่ละฟิลด์ในคลาสจะถูกแปลงเป็นฟิลด์ในตารางของฐานข้อมูล ซึ่งมีขั้นตอนดังต่อไปนี้

1. กำหนดคุณสมบัติของคลาสให้เป็น persistent
2. กำหนดค่า DDL Code generation
3. เลือกคลาสที่ต้องการ generate จากคลาสไดอะแกรม
4. ไปที่เมนู Tools -> DDL -> Generate Code.

โดยปกติ Rose จะทำการ generate DDL สำหรับทุกคลาส แต่มีคลาสบางคลาสที่เป็น Boundary จะไม่ได้มีการแปลงไปเป็นตาราง

ขั้นตอนแรกให้ทำการเลือกคลาสที่ต้องการ generate DDL แล้วทำการกำหนดค่าเป็น persistent โดยในการกำหนดค่านี้ให้ทำการเปิด class specification window ของแต่ละคลาส แล้วเลือกไปยังหน้า Detail tab แล้วเลือกเลือกปุ่ม persistent ดังหน้าจอในรูปที่ 4.1

Class Specification for EMPLOYEE

Relations | Components | Nested | Files
General | Detail | Operations | Attributes

Multiplicity:

Space:

Persistence: ☒ Persistent ☐ Transient

Concurrency: ☒ Sequential ☐ Guarded ☐ Active ☐ Synchronous

☐ Abstract

Formal Arguments:

Name	Type	Default Value

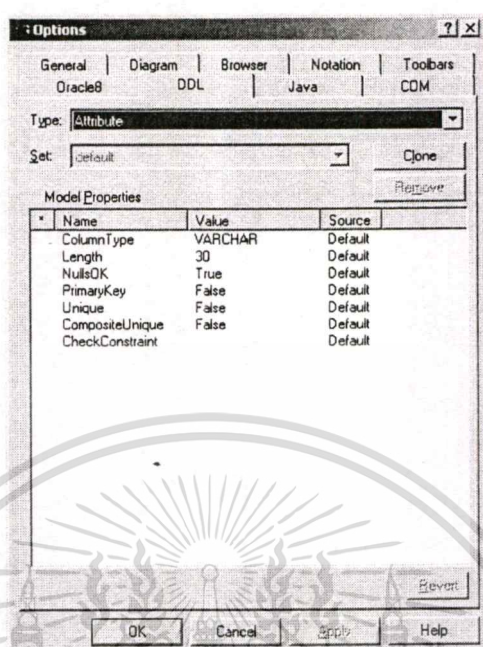
OK Cancel Apply Browse Help

รูปที่ 4.1 หน้าจอ Class Specification

นอกจากการกำหนดคุณสมบัติของคลาสให้เป็น persistent แล้วเรายังสามารถกำหนดคุณสมบัติในการทำ DDL generation โดยที่การดำเนินการของคลาสและส่วนต่างๆของโมเดล จะไม่มีผลต่อการทำ DDL generation

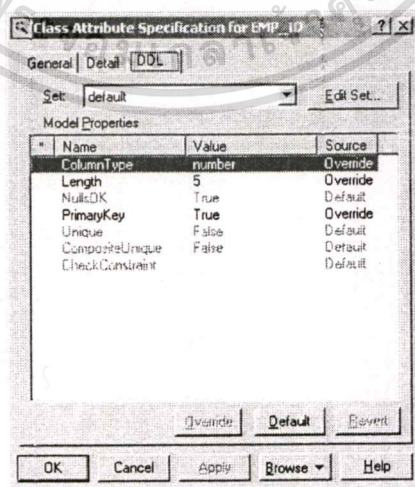
4.4 คุณสมบัติต่างๆในการทำ DDL Code Generation

คุณสมบัติของคลาสจะเป็นตัวกำหนดคุณสมบัติในการ DDL generation ซึ่งคุณสมบัติของ DDL จะกำหนดได้ในระดับของการระบุไปที่ attribute หรือ กำหนดได้ในระดับของการทำทั้ง project โดยการเข้าไปเลือกที่เมนู Tools -> Options แล้วเลือกไปยัง DDL Tab



รูปที่ 4.2 หน้าจอ Options

ในหน้าจอนี้สิ่งที่เรากำหนดไว้จะเป็นค่า default ให้กับทุกๆ attribute หรือ ทั้ง project เราสามารถกำหนดคุณสมบัติให้ทีละ attribute ได้โดยการเปิด attribute specification window ของ attribute นั้นๆ แล้วเลือก DDL tab โดยการเปลี่ยนแปลงนี้จะมีผลกับ attribute เดียว

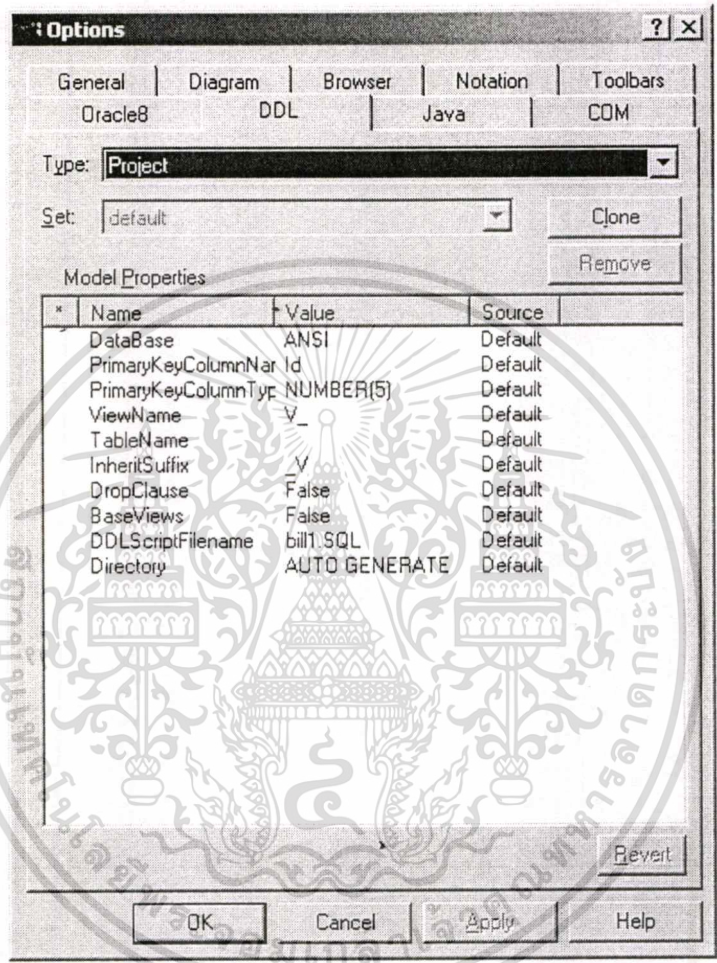


รูปที่ 4.3 Class Attribute Specification

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4.1 Project Properties

Project properties จะมีผลกระทบต่อ DDL code ของทั้ง project



รูปที่ 4.4 Project Properties

ในหน้าจอนี้ สามารถกำหนดค่า database default, defaultprimary key name, อักษรนำหน้าสำหรับ view, อักษรนำหน้าสำหรับตาราง โดยที่ค่าต่างๆที่มีให้กำหนดได้นั้นแสดงดังตารางที่ 4.5

Property	จุดประสงค์	Default
Directory	กำหนดค่า default ของไดเรกตอรีที่จะทำการ generate DDL	ไดเรกตอรีปัจจุบัน
Database	กำหนดค่าประเภทของฐานข้อมูลที่ใช้	ANSI

Property	จุดประสงค์	Default
PrimaryKeyColumnName	ถ้าไม่ได้ระบุไว้ว่า attribute ไหนของคลาสเป็น primary key, Rose จะสร้าง primary key ให้โดยใช้ชื่อของคลาสนั้นๆตามด้วยค่าที่ตั้งไว้ของ property นี้	Id
PrimaryKeyColumnType	ถ้าเราไม่ได้ระบุ attribute ในคลาสให้เป็น primary key, Rose จะสร้าง primary key ให้ โดยที่ data type ของ primary key จะเป็นค่าตามที่ระบุใน property นี้	NUMBER(5)
ViewName	กำหนดอักษรนำหน้าในการ generate ชื่อ view	V_
TableName	กำหนดอักษรนำหน้าในการ generate ชื่อตาราง	T_
InheritSuffix	กำหนดอักษรด้านหลังสำหรับ view เพื่อใช้สำหรับการสืบทอด	_V
DropClause	ควบคุมประโยค DROP TABLE ว่าให้ทำคำสั่งนี้ก่อนที่จะมีการทำคำสั่ง create table หรือไม่	False
BaseViews	ควบคุมว่าให้มีการ generate View สำหรับตาราง	False
DDLScriptFilename	ชื่อไฟล์ที่เกิดจากการ generate DDL	DDL1.SQL

ตารางที่ 4.1 Project DDL Generation Properties

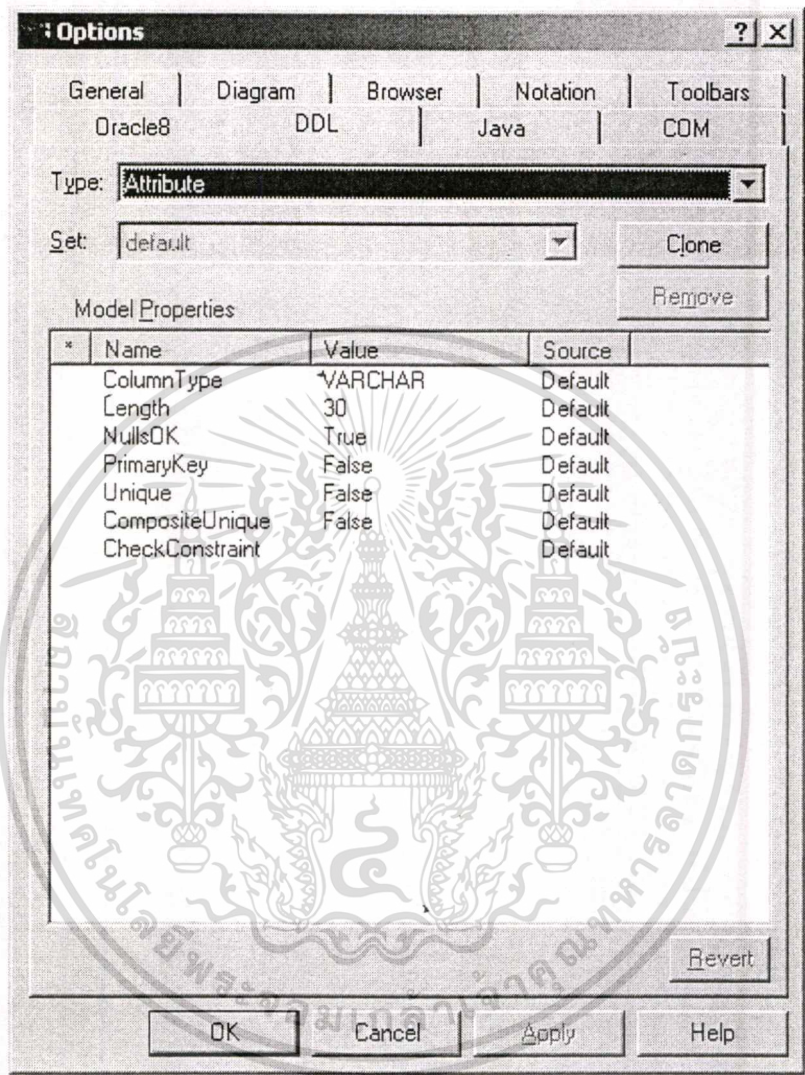
4.4.2 Attribute Properties

คุณสมบัติของ DDL จะกำหนดได้ด้วย attribute set ซึ่งเราสามารถกำหนด ค่า default ของแต่ละฟิลด์ได้

วิธีการในการกำหนดคุณสมบัติซึ่งมีผลกับทุกๆ attribute ทำได้โดยเข้าไปที่เมนู Tools ->

Options ไปที่ DDL tab แล้วเลือก attribute จาก drop-down list box แต่ถ้าเราต้องการจะกำหนดให้

มันมีผลเฉพาะบาง attribute ก็ทำได้โดยการไปเลือกที่ DDL tab ในหน้าจอ attribute specification ของ attribute ของคลาสนั้นๆและแก้ไขค่าที่หน้าจอ นั้น



รูปที่ 4.5 Attribute Properties

Property	จุดประสงค์	Default
ColumnType	กำหนด data type ของฟิลด์ที่จะทำ การ generate	VARCHAR
Length	กำหนดความยาว ของฟิลด์ที่จะทำ การ generate	Blank
NullsOK	ควบคุมว่าในฟิลด์นั้นจะยอมให้มี ค่าว่างได้หรือไม่	True

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้ภายในองค์กรเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Property	จุดประสงค์	Default
PrimaryKey	ควบคุมว่า ในฟิลด์นั้นๆเป็น primary key หรือเป็นส่วนหนึ่งของ primary key หรือเปล่า	False
Unique	กำหนดว่าต้องการให้ค่าในแต่ละเรคคอร์ดมีความเป็นเอกลักษณ์หรือเปล่า	False
CompositeUnique	ควบคุมว่าฟิลด์นี้เป็นส่วนหนึ่งของ คีย์สำหรับตารางหรือไม่	False
CheckConstraint	กำหนดการตรวจสอบ constraint สำหรับฟิลด์	Blank

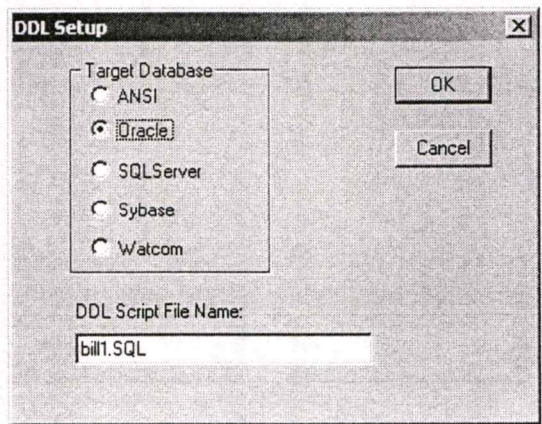
ตารางที่ 4.2 Property DDL Generation Properties

4.5 Generated Code

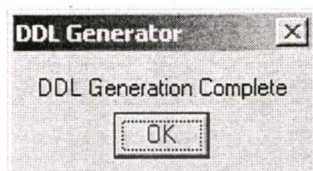
Rose จะ generate หนึ่งตารางต่อหนึ่งคลาสในกรณีที่ในแต่ละคลาสไม่มีความสัมพันธ์กัน แต่ถ้าในแต่ละ persistent คลาสมีความสัมพันธ์กัน Rose จะเพิ่มความสัมพันธ์โดยการใส่ foreign key และสร้าง views ขึ้นมา ซึ่งการ Generated Code ทำได้โดย

1.ให้เข้าไปที่ Class Diagram แล้วเลือกคลาสที่ต้องการ generate ซึ่งเราสามารถเลือกหลายๆคลาสได้

2.เข้าไปที่เมนู Tools -> DDL -> Generate Code แล้วจะปรากฏหน้าจอดังรูปที่ 4.6



3.เมื่อเรากดปุ่ม OK ในหน้าจอที่ 4.6 หลังจากนั้น Rose จะทำการ Generated Code เมื่อสำเร็จแล้วจะปรากฏหน้าจอดังรูปที่ 4.7



รูปที่ 4.7 หน้าจอแสดงผลการ Generate DDL

4.หลังจากนั้นเราจะได้ไฟล์ที่เป็นชื่อและนามสกุล ตามที่ได้กำหนดไว้ใน Project Property ซึ่งไฟล์นี้ในขั้นต่อไปทำการรันสคริปต์เพื่อสร้างตารางในฐานข้อมูล Oracle หรือในฐานข้อมูลชนิดอื่นๆได้

5.ถ้าเราต้องการดูผลลัพธ์ของ DDL Code ที่ Rose generate ให้ก็ทำได้โดยให้เข้าไปที่เมนู Tools -> DDL -> Browse DDL

```

bill1 - Notepad
File Edit Format Help
CREATE TABLE PROMOTION(
  PROMOTION_ID NUMBER(2),
  PROMOTION_NAME VARCHAR(30),
  PROMOTION_DETAIL VARCHAR(200),
  PROMOTION_START DATE(30),
  PROMOTION_STOP DATE(30),
  PROMOTION_PRICE NUMBER(7),
  ISP_ID number(2),
  FOREIGN KEY (ISP_ID) REFERENCES ISP,
  PROM_PROD_ID NUMBER(2),
  FOREIGN KEY (PROM_PROD_ID) REFERENCES PROMOTION_PRODUCT ON DELETE CASCADE,
  PROM_SERV_ID NUMBER(2),
  FOREIGN KEY (PROM_SERV_ID) REFERENCES PROMOTION_SERVICE ON DELETE CASCADE,
  PRIMARY KEY(PROMOTION_ID));

CREATE TABLE PROMOTION_PRODUCT(
  PROM_PROD_ID NUMBER(2),
  PMPD_START_DATE DATE(30),
  PMPD_END_DATE DATE(30),
  PMPD_AMOUNT NUMBER(7),
  DISCOUNT_TYPE VARCHAR(10),
  COST_TYPE VARCHAR(10),
  PROD_ID NUMBER(5),
  FOREIGN KEY (PROD_ID) REFERENCES PRODUCT,
  PROMOTION_ID NUMBER(2),
  FOREIGN KEY (PROMOTION_ID) REFERENCES PROMOTION,
  PRIMARY KEY(PROM_PROD_ID));

CREATE TABLE PROMOTION_SERVICE(
  PROM_SERV_ID NUMBER(2),
  PMSV_START_DATE DATE(30),
  PMSV_AMOUNT NUMBER(9),
  DISCOUNT_TYPE VARCHAR(10),
  COST_TYPE VARCHAR(10),

```

4.6 การดำเนินการของคลาส

ในที่นี้มีการแสดง Activity Diagram ของการดำเนินการบางการดำเนินการเท่านั้น

1. คลาส DBBusinessLogic

DBBusinessLogic::GetInvoice(INV_NO:VARCHAR2):INVOICE

มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง INVOICE ขึ้นมาโดยการระบุหมายเลขใบแจ้งหนี้ที่ต้องการส่งไปให้

DBBusinessLogic::GetInvoiceDetail(INV_NO:VARCHAR2):INVOICE_DETAIL

มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง INVOICE_DETAIL

DBBusinessLogic::GetReceipt(REC_NO:VARCHAR2):RECEIPT

มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง RECEIPT ขึ้นมาโดยการระบุหมายเลขใบเสร็จรับเงินที่ต้องการส่งไปให้

DBBusinessLogic::GetReceiptDetail(REC_NO:VARCHAR2):RECEIPT_DETAIL

มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง RECEIPT_DETAIL

DBBusinessLogic::GetDebitNote(DN_NO:VARCHAR2):DEBIT_NOTE

มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง DEBIT_NOTE ขึ้นมาโดยการระบุหมายเลขใบ Debit Note ที่ต้องการส่งไปให้

DBBusinessLogic::GetDebitNoteDetail(DN_NO:VARCHAR2):DEBIT_NOTE_DETAIL

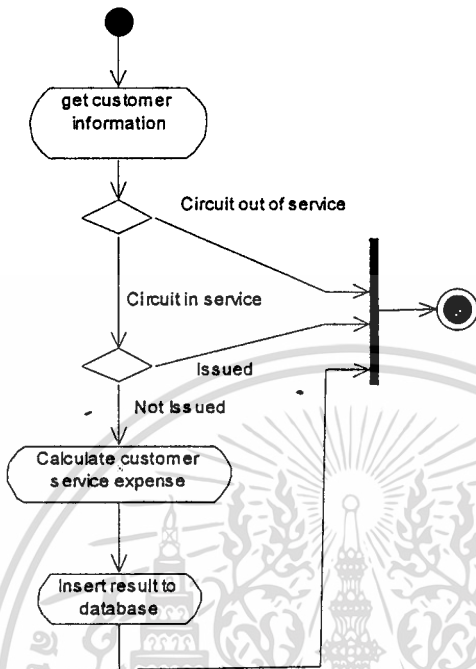
มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง DEBIT_NOTE_DETAIL

DBBusinessLogic::GetCreditNote(CN_NO:VARCHAR2):CREDIT_NOTE

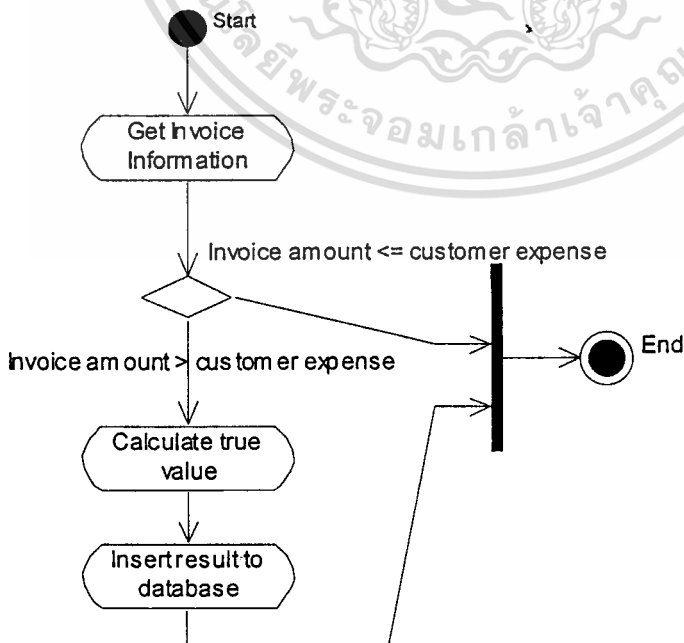
มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง CREDIT_NOTE ขึ้นมาโดยการระบุหมายเลขใบ Credit Note ที่ต้องการส่งไปให้

DBBusinessLogic::GetCreditNoteDetail(CN_NO:VARCHAR2):CREDIT_NOTE_DETAIL

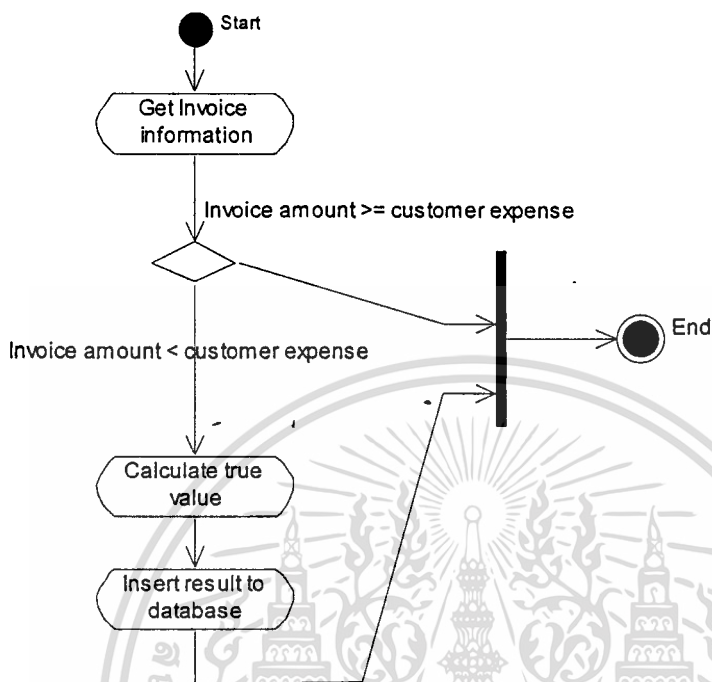
มีหน้าที่ในการดึงข้อมูลจากฐานข้อมูลในตาราง CREDIT_NOTE_DETAIL



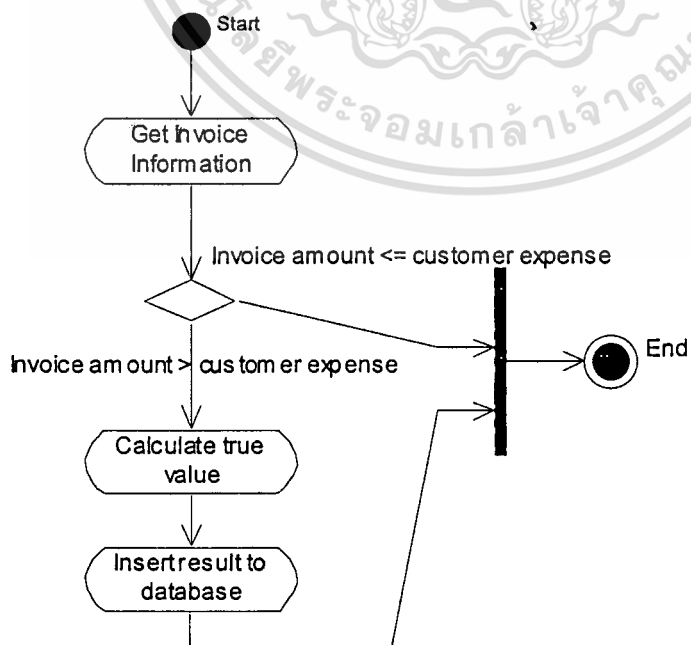
รูปที่ 4.9 Activity Diagram สำหรับการดำเนินการ AddInvoice



รูปที่ 4.10 Activity Diagram สำหรับการดำเนินการ AddDebitNote



รูปที่ 4.11 Activity Diagram สำหรับการดำเนินการ AddCreditNote



รูปที่ 4.12 Activity Diagram สำหรับการดำเนินการ AddDebitNote

บทที่ 5

สรุป

5.1 ข้อดีของการพัฒนาระบบเชิงวัตถุ

ในการทำการวิเคราะห์และออกแบบระบบ มักมีการปรับเปลี่ยนและแก้ไขระบบอยู่บ่อยๆ เพื่อให้สอดคล้องกับความต้องการที่เปลี่ยนไป ซึ่งเราสามารถทำการปรับเปลี่ยนได้ง่ายและไม่มีผลกระทบต่อส่วนอื่นๆ ของระบบ นอกจากนี้ระบบยังง่ายแก่การทำความเข้าใจและสะดวกในการสื่อสารกับทีมงาน

การนำกลับมาใช้ใหม่ สามารถทำได้ง่ายเพราะภายในวัตถุเองต้องประกอบด้วยข้อมูลและการดำเนินการกับข้อมูลนั้น ซึ่งแต่ละวัตถุสามารถแทนด้วยวัตถุในโลกแห่งความเป็นจริงได้

Case Tool แต่ละตัวสนับสนุน UML และสามารถสร้างโค้ดต้นแบบในภาษาโปรแกรมเชิงวัตถุได้ เช่น Rational Rose สามารถนำแบบจำลองที่ได้เป็นต้นแบบในการสร้างโค้ดภาษาต่างๆ เช่น C++, Java, Visual Basic และ Oracle8

5.2 ปัญหาที่เกิดขึ้น

เนื่องจากผู้จัดทำได้ใช้ภาษา Delphi5.0 ซึ่ง Case Tools Rational Rose 2000 ยังไม่สนับสนุนการทำงานกับภาษานี้ทำให้ผู้จัดทำได้ใช้ประโยชน์จาก Case Tools ตัวนี้ในการช่วยสร้าง Source Code โดยอัตโนมัติให้ไม่ได้ แต่ใช้ในการ Generate DDL สำหรับนำไปใช้ในการสร้างฐานข้อมูลบน Oracle เท่านั้น

บทที่ 6

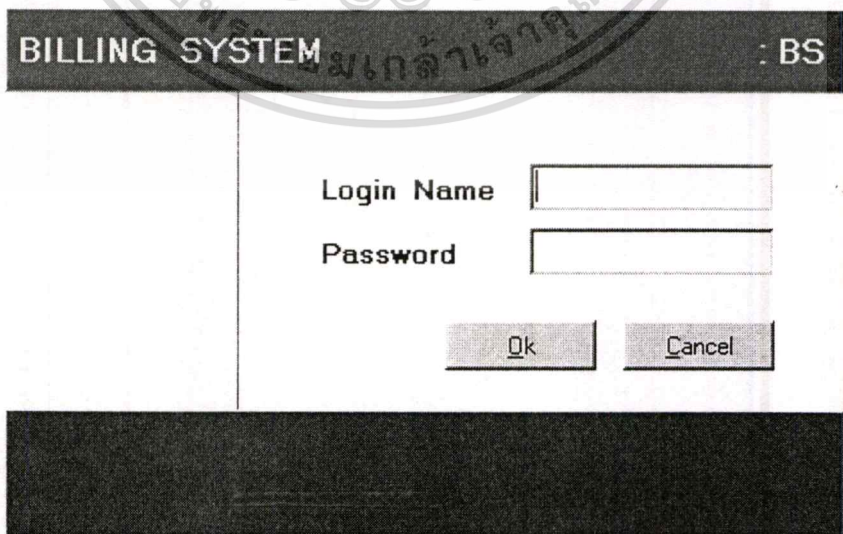
คู่มือการใช้งาน

โปรแกรม Billing System นี้เป็นโปรแกรมเพื่ออำนวยความสะดวกในการนำข้อมูลลูกค้ามาจัดทำใบแจ้งหนี้ (Invoice) ใบเสร็จรับเงิน(Receipt) ใบลดหนี้(Credit Note) และใบเพิ่มหนี้(Debit Note) เพื่อให้สามารถทำงานได้รวดเร็ว และถูกต้องแม่นยำ โดยในระบบจะประกอบไปด้วยระบบย่อยซึ่งมีหน้าที่ในการทำงานต่างๆดังนี้

- 1.ระบบเอกสารใบแจ้งหนี้
- 2.ระบบเอกสารใบเสร็จรับเงิน
- 3.ระบบเอกสารใบลดหนี้
- 4.ระบบเอกสารใบเพิ่มหนี้

6.1 การเข้าสู่โปรแกรม

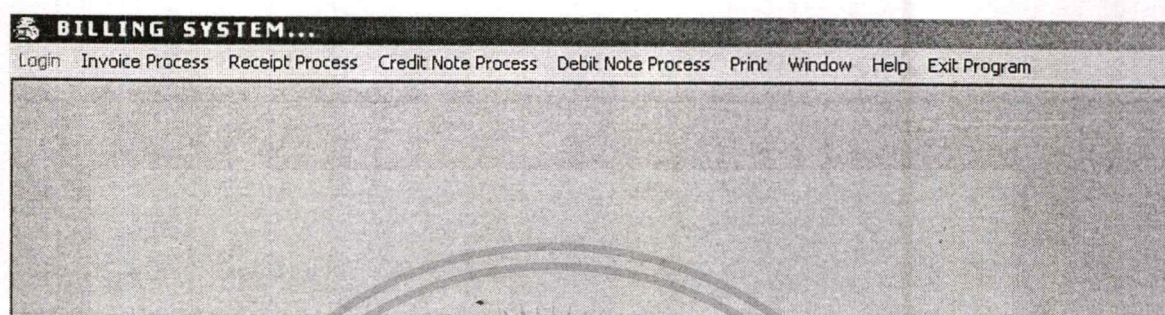
เมื่อเริ่มเข้าสู่โปรแกรมจะปรากฏหน้าต่าง Login ผู้ใช้ระบบจะต้องทำการกรอก User Name และ Password ใส่ลงไปแล้ว Click ปุ่ม Ok



รูปที่ 6.1 หน้าจอ Login

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลังจาก Login แล้วจะปรากฏหน้าจอแรกซึ่งจะประกอบด้วยระบบย่อยต่างๆตามที่กล่าวมาแล้วข้างต้น



รูปที่ 6.2 แสดงหน้าจอแรกของระบบ

ภายในระบบจัดทำเอกสารใบแจ้งหนี้จะประกอบด้วยส่วนทำงานดังนี้

6.2 Customer Account Control

ใช้ในการตรวจสอบยอดเงินปัจจุบันของลูกค้าว่ามียอดชำระเงินมาเกินหรือค้างชำระเป็นมูลค่าเท่าใด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Customer Account Control

Gen Account New Address Edit Save Undo Exit

Search Account

Customer Name

Customer Account ID Customer Account Name

Address No. Moo

Building Name Floor

Soi Road

Thumbon Amphur

Province Zip Code

Telephone Fax

Current Balance Baht

รูปที่ 6.3 หน้าจอ Customer Account Control

BILLING SYSTEM... - [Generate Automatic]

Login Invoice Process Receipt Process Credit Note Process Debit Note Process Print Window Help Exit Program

Generate Invoice

No. of Simulation

Service Type

Company

Invoice Date 13/ 9 / 2544

Invoice Due Date 13/ 9 / 2544

Detail of Simulation

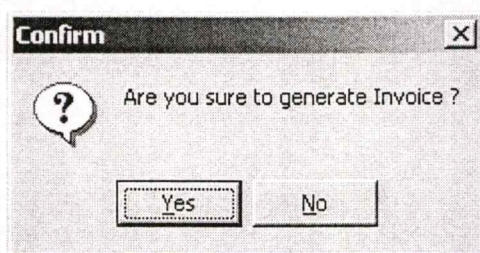
No.	Customer	Amount (฿)	Discount (฿)	Net Amount (฿)	Previous Balance (฿)
		Total	฿0.00	฿0.00	฿0.00

รูปที่ 6.4 หน้าจอ Generate Invoice

6.3 การ Generate Invoice

ขั้นตอนการทำงานของหน้าจอ Generate Invoice มีดังนี้

- 1.เลือก No. of Simulation แล้วจะปรากฏข้อมูลของลูกค้าเพื่อทำการ Generate Invoice ต่อไป
- 2.ทำการกดปุ่ม Generate แล้วโปรแกรมจะทำการคำนวณใบแจ้งหนี้ของลูกค้าและบันทึกข้อมูลกลับลงสู่ฐานข้อมูล
- 3.เมื่อกดปุ่ม Generate แล้วจะปรากฏหน้าจอขึ้นดังรูปเพื่อยืนยันการ Generate Invoice



รูปที่ 6.5 หน้าจอยืนยันการ Generate Invoice

4.เมื่อกดปุ่ม Yes แล้วจะปรากฏหน้าจอ ดังรูปที่ 6.4

Generate Invoice

No. of Simulation: Invoice Date: 13/9 / 2544

Service Type: Invoice Due Date: 13/9 / 2544

Company:

New Invoice		Invoice no.	Customer	Amount (\$)	Discount (\$)	Net Amount ...
Total Invoice	1	102000012	บริษัท จำกัด (มหาชน) จำกัด			0
Total Circuit	1					
Total Customer	1					
Total New Invoice	0					
Total New Circuit	0					
Total New Customer	0					
Total Hold	0					
Total Hold Circuit	0					
Total Hold Customer	0					
Total Amount	00.00					
Total Discount	00.00					
Total Net Amount	00.00					

รูปที่ 6.6 หน้าจอ Generate Invoice

BILLING SYSTEM (Generate Automatic)

Login Invoice Process Receipt Process Credit Note Process Debit Note Process Print Window Help Exit Program

Generate Invoice

No. of Simulation: Invoice Date: 13/9/2544

Service Type: Invoice Due Date: 13/9/2544

Company:

New Invoice		Invoice no.	Customer	Amount (฿)	Discount (฿)	Net Amount...
Total Invoice	3	1022000005	บริษัท อีที (ประเทศไทย) จำกัด			0
Total Circuit	3	1022000008	บริษัท อีที (ประเทศไทย) จำกัด			0
Total Customer	3	1022000002	บริษัท อีที (ประเทศไทย) จำกัด			0
Total New Invoice	0					
Total New Circuit	0					
Total New Customer	0					
Total Hold	0					
Total Hold Circuit	0					
Total Hold Customer	0					
Total Amount	฿0.00					
Total Discount	฿0.00					
Total Net Amount	฿0.00					

Project Billing X

Generate Complete

OK

..... ติดต่อฝ่ายนางสาวจุไรพร ณ โขนา เจ้าของระบบ

รูปที่ 6.7 หน้าจอหลังจาก Generate Invoice

6.4 การ Cancel Invoice

เมื่อต้องการยกเลิกใบ Invoice ทำได้โดยการเข้าไปที่หน้าจอหลักเมนู Invoice Process->Generate Invoice แล้วจะปรากฏหน้าจอ View & Cancel Invoice ดังปรากฏในรูปที่ 6.6

BILLING SYSTEM... [View & Cancel Invoice]

Login Invoice Process Receipt Process Credit Note Process Debit Note Process Print Window Help Exit Program

View & Cancel Invoice

Edit Save Undo Cancel Refresh Print Exit

Select Type: Search:

Invoice No. Name: Month: / /

Address: Inv Date: 17/8 / 254 Inv Due: 17/8 / 254

Sales: Doc Status: Pay Status:

Description

Expense Type	ADSL No.	Start Used Date	End Used Date	Speed	Discount	Amount

Reason to cancel!!! Previous Balance: ฿0.00 Total Amount: ฿0.00

Total Discount: ฿0.00

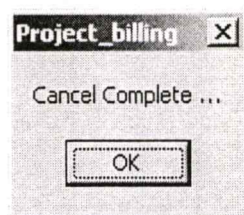
Vat: ฿0.00

Net: ฿0.00

รูปที่ 6.8 หน้าจอ View & Cancel Invoice

ขั้นตอนการทำงานของหน้าจอ View & Cancel Invoice มีดังนี้

- 1.เลือกประเภทของใบ Invoice
- 2.หลังจากนั้นที่ด้านขวาของจอจะปรากฏ Invoice No. ขึ้นมาซึ่งสามารถใช้เรียกดูข้อมูลของใบ Invoice และสามารถเลือกใบ Invoice ที่ต้องการยกเลิกได้ โดยที่ด้านล่างของจอจะปรากฏรายละเอียดค่าใช้จ่ายต่างๆของใบ Invoice นั้น พร้อมทั้งสถานะการจ่ายเงินของเอกสาร Previous Balance Total Amount Total Discount Vat และ Net Amount ของแต่ละใบ Invoice
3. กดปุ่ม Document reference เพื่อดูเอกสารที่เกี่ยวข้องกับใบ Invoice ใบนี้ ซึ่งจะปรากฏหน้าจอ ดังรูปที่ ซึ่งในหน้าจอนี้จะแสดงว่าใบ Invoice ใบนั้นๆมีใบลดหนี้ และใบเพิ่มหนี้ที่เกี่ยวข้องอะไรบ้าง



รูปที่ 6.11 แสดงหน้าจอบอกสถานะการทำการ Cancel Invoice

6.5 การ Generate Receipt

ในส่วนของระบบนี้มีหน้าที่ในการจัดทำใบเสร็จรับเงินซึ่งเป็นขั้นตอนต่อจากการการ Generate Invoice และส่งใบแจ้งหนี้ไปยังลูกค้า หลังจากนั้นลูกค้าจะมาชำระเงินค่าบริการ เข้ามา ซึ่งขั้นตอนต่อไปเป็นการออกใบเสร็จรับเงินให้กับลูกค้า โดยก่อนที่จะทำการ Generate Receipt ต้องทำการเพิ่ม หรือแก้ไขข้อมูล ธนาคาร และสาขาของธนาคารไว้ก่อน โดยการเข้าไปที่เมนู Generate Receipt-> Bank Description

Bank Description		
Bank Name	Add	Edit Save Close
Bank Code	Bank Name	
1	กรุงเทพพาณิชย์การ	
2	กรุงศรีอยุธยา	
3	ยูนิโคม มิตซูบิชิ	
4	กสิกรไทย	
5	กรุงไทย	
6	ออมสิน	
Branch Name	Add	Edit Save
Branch Code	Bank Name	Branch Name
1	กรุงเทพพาณิชย์การ	บางเขน
2	กรุงศรีอยุธยา	สุขุมวิท
3	girl	boy

รูปที่ 6.12 แสดงหน้าจอ Bank Description

ขั้นตอนการทำงานของหน้าจอ Bank Description

การเพิ่มข้อมูล Bank Name

1. กดปุ่ม Add
2. ป้อนข้อมูลรายชื่อธนาคาร
3. กดปุ่ม Save เพื่อทำการบันทึกข้อมูล

การแก้ไขข้อมูล Bank Name

1. กดปุ่ม Edit
2. เลือกข้อมูล Bank Name ที่ต้องการแก้ไขจากตาราง
3. ทำการแก้ไขข้อมูล
4. กดปุ่ม Save เพื่อทำการบันทึกข้อมูล

การเพิ่มข้อมูล Branch Name

1. กดปุ่ม Add
2. ป้อนข้อมูล
3. กดปุ่ม Save เพื่อทำการบันทึกข้อมูล

การแก้ไขข้อมูล Branch Name

1. กดปุ่ม Edit
2. เลือกข้อมูล Branch Name ที่ต้องการแก้ไขจากตาราง
3. ป้อนข้อมูล
4. กดปุ่ม Save เพื่อทำการบันทึกข้อมูล

เมื่อลูกค้าชำระเงินตามใบเรียกเก็บเงินมาแล้ว ขั้นตอนต่อไปก็จะเป็นการออกใบเสร็จรับเงินให้กับลูกค้าตามมูลค่าที่ลูกค้าได้ชำระเงินเข้ามา

BILLING SYSTEM.... [Generate Receipt]

Login Invoice Process Receipt Process Credit Note Process Debit Note Process Print Window Help Exit Program

Generate Receipt

Customer:

Receipt Date: 10/2 / 2545

Salesman:

Document No.	Document Date	Due Date	Amount (฿)	Pay amount (฿)

Amount:

Val:

Net Amount:

Type of Payment:

Bank:

Cheque No.:

Branch:

Date:

Paid Amount:

Remark:

รูปที่ 6.13 หน้าจอ Generate Receipt

Frm_SelectInv

Invoice

Invoice No.:

Invoice Date: 01/01/2002

Amount:

Discount:

Relate Document

Document No.	Document type	Document Date	Amount (฿)

Amount Balance:

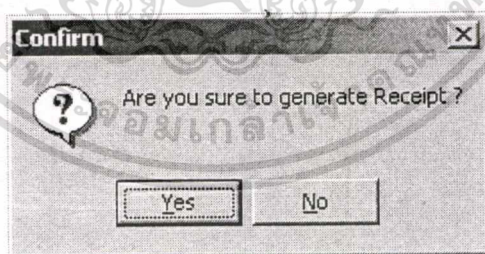
Pay Amount:

รูปที่ 6.14 หน้าจอ Select Invoice

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

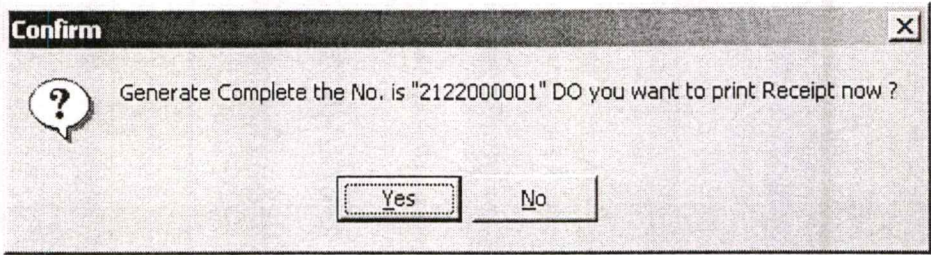
ขั้นตอนการทำงานของหน้าจอ Generate Receipt

- 1.เลือกเมนู Receipt Process -> Generate Receipt
- 2.เลือกรายชื่อลูกค้าที่ต้องการออกไปเสร็จรับเงินจาก List Box
- 3.กดปุ่ม Invoice เพื่อเลือกใบ Invoice ที่ตรงกับที่เราต้องการออกไปเสร็จรับเงิน หลังจากนั้นจะปรากฏหน้าจอดังรูปที่ 6.11 ให้ทำการเลือกหมายเลขใบ Invoice ที่ต้องการจาก Combo Box Invoice No.
- 4.ที่หน้าจอ Select Invoice ให้กดปุ่ม Ok
- 5.หลังจากนั้นที่หน้าจอ Generate Receipt จะปรากฏข้อมูลของเอกสารที่เกี่ยวข้องในตาราง ซึ่งจะแสดงข้อมูล Document Date Due Date Amount และ Pay Amount
- 6.เลือกวิธีการชำระเงินของลูกค้าจาก Type of Payment ซึ่งจะประกอบไปด้วย Cash Bank Transfer และ Cheque
- 7.เลือกชื่อธนาคาร และสาขาของธนาคาร
- 8.ป้อนจำนวนเงินที่รับชำระ Paid Amount
- 9.ถ้ามีหมายเหตุ ให้ใส่ลงในช่อง Remark
- 10.กดปุ่ม Generate
- 11.จะปรากฏหน้าจอ Confirm เพื่อยืนยันการ Generate Receipt ให้เลือกกดปุ่ม Yes



รูปที่ 6.15 หน้าจอ Confirm Generate Receipt

- 12.เมื่อระบบทำการ Generate Receipt เรียบร้อยแล้วจะมีหน้าจอ ดังรูปที่ แสดงว่าระบบได้ทำการ Generate Receipt เรียบร้อยแล้ว



รูปที่ 6.16 หน้าจอแสดงว่าได้มีการ Generate Receipt เรียบร้อยแล้ว

6.6 การ Cancel Receipt

มีหน้าที่ในการแก้ไข หรือยกเลิกใบเสร็จรับเงินที่ได้เคยจัดทำไว้ โดยการเลือกเมนู Receipt Process -> View & Cancel Receipt จะปรากฏหน้าจอดังรูปที่ 6.15

Document No.	Document Type	Amount

Type of Payment	Bank	Amount	฿0.00
Cheque No.	Branch	Vat	฿0.00
Date	Paid Amt	Net	฿0.00
Remark			
Reason to cancel !!!			

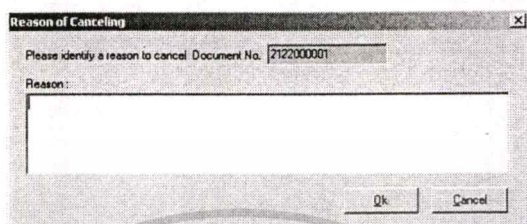
รูปที่ 6.17 หน้าจอ View & Cancel Receipt

ขั้นตอนการทำงานของหน้าจอ View & Cancel Receipt

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.ที่ปุ่ม Search ใส่หมายเลขใบเสร็จรับเงิน แล้วกดปุ่ม Search แล้วจะปรากฏรายละเอียดใบเสร็จที่เราค้นหา

2.กดปุ่ม Cancel แล้วจะปรากฏหน้าจอดังรูปที่ ให้ระบุเหตุผลในการยกเลิก



รูปที่ 6.18 หน้าจอ Reason of Canceling Receipt

3.ระบุเหตุผลในการ Cancel Receipt แล้วกดปุ่ม Ok

4.เมื่อทำการยกเลิกได้สำเร็จแล้วจะปรากฏหน้าจอดังรูปที่



รูปที่ 6.19 หน้าจอแสดงการ Cancel Receipt สำเร็จแล้ว

6.7 การ Generate Credit Note

จะเกิดขึ้นเมื่อ ลูกค้าชำระเงินมา ... กับยอดเงินที่ได้เรียกเก็บในใบแจ้งหนี้

BILLING SYSTEM....[Generate Credit Note] [F5] [X]

Login Invoice Process Receipt Process Credit Note Process Debit Note Process Print Window Help Exit Program

Generate Credit Note

Invoice Description

Invoice NO:

Name:

Address:

Date: / / Amount: ฿0.00

Due Date: / / Discount: ฿0.00

Relate Document			
Document No.	Doc. Type	Doc. Date	Amount (฿)

Amount Balance: ฿0.00

New Amount: ฿0.00

Diff Amount: ฿0.00

Credit Note Description

Credit Note Date: 22/8 / 2544 Salesman:

รายละเอียด:

จำนวนเงิน:

Remark:

Amount: ฿0.00

Vol: - ฿0.00

Net: ฿0.00

รูปที่ 6.20 หน้าจอ Generate Credit Note

ขั้นตอนการทำงานของหน้าจอ Generate Credit Note

1.เลือก Invoice No. Combo Box เพื่อเลือกหมายเลขใบแจ้งหนี้ที่เราต้องการทำการ ลดหนี้ หลังจากเลือก Invoice No. แล้วที่หน้าจอด้านบนจะแสดงข้อมูลของใบแจ้งหนี้ ส่วนที่ด้านล่างของจอให้พิมพ์รายละเอียดของรายการ และจำนวนเงินที่ต้องการลดหนี้

2.กดปุ่ม Generate เพื่อทำการ Generate Credit Note

6.8 การ Cancel Credit Note

BILLING SYSTEM... - [View & Cancel Credit Note]

Login Invoice Process Receipt Process Credit Note Process Debit Note Process Print Window Help Exit Program

View & Cancel Credit Note

Select Type [] Search: []

Credit Note

Name [] Address [] CN Date 24/8/2544 Salesman [] Ref. No. [] Doc Status []

Description

Remark	Amount	Vat	Net
	0.00	0.00	0.00

Reason to cancel !!! []

รูปที่ 6.21 หน้าจอ View & Cancel Credit Note

ขั้นตอนในการทำงานของหน้าจอ View & Cancel Credit Note

- 1.เข้าไปที่เมนู Credit Note Process -> View & Cancel Credit Note
- 2.ไปที่ Select Type เพื่อเลือกประเภทของใบ Credit Note
- 3.ที่ด้านซ้ายของจอ จะแสดงหมายเลข Credit Note ขึ้นมาให้เลือก หรืออาจเลือกใบ Credit Note ที่ต้องการ โดยการ
- 4.ถ้าต้องการใบแจ้งหนี้ที่เกี่ยวข้องกับใบ Credit Note นี้สามารถดูได้โดยการกดปุ่ม Invoice
- 5.กดปุ่ม Cancel เพื่อทำการยกเลิกใบ Credit Note

6.9 Generate Debit Note

รูปที่ 6.22 หน้าจอ Generate Debit Note

ขั้นตอนการ Generate Debit Note

1.เลือก หมายเลข Invoice จาก Invoice No. Combo Box แล้วจะปรากฏข้อมูล Invoice และเอกสารที่เกี่ยวข้องซึ่งจะแสดงในตารางทางด้านขวา

2.ป้อนรายละเอียดของรายการที่ต้องการทำ Debit Note คือรายละเอียด จำนวนเงิน และหมายเหตุ

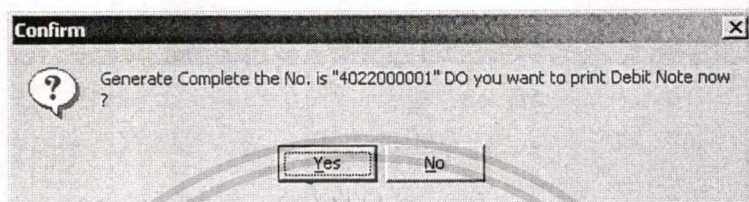
3.กดปุ่ม Generate เพื่อทำการ Generate Debit Note

รูปที่ 6.23 ยืนยันการ Generate Debit Note

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ทำซ้ำและต้องแจ้งเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.กดปุ่ม Yes เพื่อยืนยันการ Generate Debit Note

5.เมื่อระบบทำการ Generate Debit Note เรียบร้อยแล้วจะปรากฏหน้าจอดังรูปที่ ถ้าต้องการพิมพ์ใบ Debit Note ใบนั้นให้กดปุ่ม Yes ถ้าไม่พิมพ์ในภายหลังให้กดปุ่ม No



รูปที่ 6.24 หน้าจอแสดงว่า Generate Complete

6.10 การ Cancel Debit Note

รูปที่ 6.25 หน้าจอที่ View & Cancel Debit Note

ขั้นตอนการ View & Cancel Debit Note

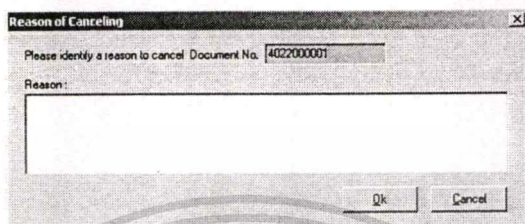
1.ป้อนหมายเลข Debit Note ที่ต้องการยกเลิกในช่อง Search จะปรากฏรายละเอียดของใบ

เอกสาร Debit Note ที่เราต้องการยกเลิกการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.กดปุ่ม Cancel

3.ที่หน้าจอ 6.24 Reason of Canceling ให้ป้อนเหตุผลในการยกเลิกใบ Debit Note

4.กดปุ่ม Ok เพื่อตกลง และถ้าต้องการยกเลิกให้กดปุ่ม Cancel



รูปที่ 6.26 หน้าจอ Reason of Canceling

5.ระบบเมื่อทำการ Cancel Debit Note เรียบร้อยจะปรากฏหน้าดังรูปที่ 6.25



รูปที่ 6.27 หน้าจอ Reason of Canceling

บรรณานุกรม

- กิตติ ภัคดีวัฒนะกุล และกิตติพงษ์ กลมกล่อม.2544.UML วิเคราะห์และออกแบบระบบเชิงวัตถุ.
กรุงเทพฯ: เคทีพี คอมพ์ แอนด์ คอนซัลท์.
- จิรายุ วิริยะพิบูล. 2537. คู่มือการเขียนเครื่องมือช่วยพัฒนาโปรแกรมด้วยเคสไฟล์.กรุงเทพฯ:ไทย
วัฒนาพานิช.
- สมนึก สิริโต.แนวทางการพัฒนาระบบเชิงออบเจกต์.กรุงเทพฯ:มหาวิทยาลัยเกษตรศาสตร์
- Bahrami, Ali. 1998. **Objected Systems Development using Unified ModelingLanguage.**
Singapore: McGraw-Hill.
- Craig, Larman. 1998. **Applying UML and Patterns.** USA: Prentice-Hall.
- Eliksson, Hans-Erik and Magnus Penker. 1998. UML Toolkit. USA: John Wiley & Sons.
- Jake, Sturm. 1999. **VB6 UML Design and Development.** Canada: Wrox Press.
- Rob, Pooley and Perdita, Stevens. 1999. **Using UML Software Engineering with Objects and
Components.** India: Eastern Press Pvt.
- Terry Quatrani. 1998. **Visual Modeling with rational rose 2000 and UML. 2nd.** USA: Addison
Wesley Longman.
- Wendy, Boggs and Michael, Boggs. 1999. **Mastering UML with Rational Rose.** USA: SYBEX.



ตัวอย่างรายงาน

ระบบคิดเงินลูกค้า มีการจัดทำรายงานออกเป็น 2 ประเภทด้วยกัน คือ

1. ใบเรียกเก็บเงินค่าบริการ (Service Invoice)
2. ใบเสร็จรับเงิน (Receipt)
3. ใบลดหนี้ (Credit Note)
4. ใบเพิ่มหนี้ (Debit Note)



ใบแจ้งหนี้

ชื่อและที่อยู่ลูกค้า/NAME & ADDRESS

เลขที่ใบแจ้งหนี้ :

วันที่:

เลขที่ใบกำกับภาษี

วันครบกำหนด:

พนักงานขาย:

รายละเอียด Description	จำนวน Quantity	ราคาต่อหน่วย Unit Price	จำนวนเงิน (บาท) Amount (Baht)
			
หมายเหตุ/Remark	จำนวนเงินรวม/Gross Total		
	ส่วนลด/Discount		
	ค่าสินค้า/บริการ/Net Amount		
	ภาษีมูลค่าเพิ่ม/Vat		

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใบเสร็จรับเงิน

ชื่อและที่อยู่ลูกค้า/NAME & ADDRESS

เลขที่ใบเสร็จรับเงิน :
เลขที่ใบกำกับภาษี

วันที่ :
วันครบกำหนด :
พนักงานขาย :

รายละเอียด Description	จำนวน Quantity	ราคาต่อหน่วย Unit Price	จำนวนเงิน (บาท) Amount (Baht)
หมายเหตุ/Remark	จำนวนเงินรวม/Gross Total		
	ส่วนลด/Discount		
	ค่าสินค้า/บริการ/Net Amount		
	ภาษีมูลค่าเพิ่ม/Vat		

ชำระเงินโดย

☐ เงินสด/CASH

☐ เช็ค/CHEQUE ธนาคาร.....สาขา...

☐ โอน/TRANSFER ทางธนาคาร.....วันที่.....จำนวนเงิน.....

อื่น ๆ/OTHER

ใบลดหนี้

ชื่อและที่อยู่ลูกค้า/NAME & ADDRESS

เลขที่ใบลดหนี้ :

วันที่ :

พนักงานขาย :

รายละเอียด Description	จำนวน Quantity	ราคาต่อหน่วย Unit Price	จำนวนเงิน (บาท) Amount (Baht)
หมายเหตุ/Remark	รวมมูลค่าตามใบกำกับภาษีเดิม		
	มูลค่าที่ถูกตัดออก		
	ผลต่าง		
	ภาษีมูลค่าเพิ่ม/Vat		
ตัวอักษร :	รวม		

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใบเพิ่มหนี้

ชื่อและที่อยู่ลูกค้า/NAME & ADDRESS

เลขที่ใบเพิ่มหนี้ :

วันที่ :

พนักงานขาย :

รายละเอียด Description	จำนวน Quantity	ราคาต่อหน่วย Unit Price	จำนวนเงิน (บาท) Amount (Baht)
หมายเหตุ/Remark	รวมมูลค่าตามใบกำกับภาษีเดิม		
	มูลค่าที่ถูกต้อง		
	ผลต่าง		
	ภาษีมูลค่าเพิ่ม/Vat		
	รวม		
ตัวอักษร :			

ประวัติผู้เขียน

ชื่อผู้เขียน	นายอนันทวุฒิ เอื้อพัฒนวงศ์
วันเดือนปีเกิด	27 มิถุนายน 2515
สถานที่เกิด	อ.ดำเนินสะดวก จ.ราชบุรี
วุฒิการศึกษาระดับปริญญาตรี	วท.บ. (วิทยาการคอมพิวเตอร์ประยุกต์)
สถานที่สำเร็จการศึกษา	คณะวิทยาศาสตร์ประยุกต์
	สถาบันเทคโนโลยีพระจอมเกล้า พระนครเหนือ
ปีที่สำเร็จการศึกษาในระดับปริญญาตรี	2538

