ห้องสมุดคณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การพัฒนาแอพพลิเคชั่นฐานข้อมูลบนระบบไคลเอ็นท์เซอร์ฟเวอร์

DATABASE APPLICATION DEVELOPMENT ON CLIENT/SEVER SYSTEM

โดย

นาย ไกรสิทธิ์ วิตตินานนท์

นาย ณัฐพงษ์ มงคลนาวิน

วัน เดือน ปี 19 ม.ค. 2539
เลขทะเบียน 034928
เลขเรียกหนังสือ T 57.228 ก9

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2537

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
034928

การพัฒนาแอพพลิเคชั่นฐานข้อมูลบนระบบไคลเอ็นท์เซอร์ฟเวอร์

DATABASE APPLICATION DEVELOPMENT ON CLIENT/SEVER SYSTEM

โดย

นาย ไกรสิทธิ์ วิตตินานนท์ 34101039

นาย ณัฐพงษ์ มงคลนาวิน 34102115

อาจารย์ที่ปรึกษา

ผ.ศ.ค.ร.ศุภมิตร จิตตะยโศธร

วิทยานิพนธ์สำหรับปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้า วิทยาเขตคุณทหาร ลาดกระบัง

ปีการศึกษา 2537

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2537

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การพัฒนาแอพพลิเคชั่นฐานข้อมูลบนระบบไคลเอ็นท์เซอร์ฟเวอร์

ผู้จัดทำ

นาย ไกรสิทธิ์ วิตตินานนท์
นาย ณัฐพงษ์ มงคลนาวิน

อาจารย์ที่ปรึกษา

(ผศ. ดร. ศุภมิตร จิตตะยโศธร)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การพัฒนาแอพพลิเคชั่นฐานข้อมูลบนระบบไคลเอ็นท์เซอร์ฟเวอร์

ไกรสิทธิ์ วิตตินานนท์
ณัฐพงษ์ มงคลนาวิน
ผ.ศ.ค.ร. สุภมิตร จิตตะยโสธร อาจารย์ที่ปรึกษา
ปีการศึกษา 2537

บทคัดย่อ

วิทยานิพนธ์ฉบับนี้ เรียบเรียงขึ้นจากการค้นคว้าเพื่อหาวิถีทางในการพัฒนาแอพพลิเคชั่น (Application) บนระบบไคลเอ็นท์เซอร์ฟเวอร์ (Client-Server) เพื่อให้เหมาะสมกับเครื่องมือที่ใช้ในการพัฒนา เนื่องจากเครื่องมือที่ใช้ในการพัฒนาระบบ (Gupta) เป็นเครื่องมือที่ได้รับการบริจาคโดยบริษัท จักรวาล จำกัด ซึ่งสนับสนุนการโปรแกรมแบบ ออพเจกต์โอเรียนเต็ด (Object-Oriented) จึงทำให้เราได้ไปค้นคว้าวิธีการออกแบบระบบ และการพัฒนา แอพพลิเคชั่นแบบ ออพเจกต์โอเรียนเต็ดมา วิธีการนี้ชื่อว่า โอเอ็มที (OMT ; Object Modeling Technique) หลักการของโอเอ็มที ใช้แนวความคิดในการออกแบบโดยมองตัวระบบเป็น ออพเจกต์ และทำการออกแบบ แบบจำลอง (Model) ตามลักษณะจริงของระบบ จากนั้นจึงทำการเพิ่มรายละเอียดลงไป และในขั้นสุดท้ายจึงทำการประยุกต์ใช้กับภาษาโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Database Application Develope On Client-Server System

Kraisit Vittinanon

Nattapong Mongkolnavin

Asst.Prof.Suppamitr Chittayasothorn

1994

Abstract

This Thesis is written for finding an appropriate methodology to develope an database application on Client-Server system.The methodology must be appropriate to the tools (GUPTA) donated by Universal Co,Ltd. This tools supports for Object-Oriented programing. The methodology , called OMT (Object Modeling Technique) , presented in this thesis is appropriate to the tools.This methodology uses models to simulate the system which it concern.After design models the methodology will add detail to each models and then immplement it to a programming language.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

ในปัจจุบันนี้แนวทางที่ได้รับความนิยมมากอย่างหนึ่งในการพัฒนาระบบก็คือแนวความคิดแบบไคล์เอ็นเซอร์ฟเวอร์นั่นเอง แนวความคิดแบบนี้มีข้อดีอย่างไร และ อะไรคือ ไคลเอ็นท์เซอร์ฟเวอร์ ในปริญญานิพนธ์เล่มนี้ได้จัดหาคำตอบไว้ให้แล้ว นอกจากจะมีการอธิบายแนวความคิดที่ว่านี้แล้ว ในปริญญานิพนธ์เล่มนี้ยังได้จัดยกตัวอย่างในการพัฒนาระบบงานให้ดูอีกด้วย โดยใช้วิธีการพัฒนาแบบ ออพเจกต์โอเรียนเต็ด พร้อมทั้งยังมีการเปรียบเทียบ การพัฒนาระหว่างการใช้เครื่องมือในการพัฒนาที่ไม่เหมือนกันอีกด้วย

เนื้อหาต่างๆในปริญญานิพนธ์สามารถแบ่งออกได้เป็นหัวข้อดังนี้

- บทที่1 บทนำ กล่าวถึงรายละเอียดเกี่ยวกับปริญญานิพนธ์ โดยสังเขป
- บทที่ 2 หลักการที่เกี่ยวข้อง กล่าวถึงทฤษฎีที่เกี่ยวข้อง เช่น แนวความคิดของระบบฐานข้อมูล , แนวความคิดแบบออพเจกต์โอเรียนเต็ด และ สถาปัตยกรรมของระบบ
- บทที่ 3 วิธีการออกแบบ กล่าวถึงวิธีการออกแบบ (Methodology) ที่ได้ทำการศึกษามา และ ตัวอย่างการประยุกต์ใช้ วิธีการออกแบบกับระบบลงทะเบียน
- บทที่ 4 การเปรียบเทียบ โอเอ็มทีกับระเบียบวิธีการอื่นๆ กล่าวถึงการเปรียบเทียบการออกแบบ โดยวิธีการแบบ โอเอ็มที กับ วิธีการออกแบบในวิธีการอื่นๆ เช่น การออกแบบ แบบโครงสร้างวิเคราะห์ (Sturcture Analysis)
- บทที่ 5 ระบบพยากรณ์ดวงชะตา กล่าวถึง การประยุกต์ใช้วิธีการที่ได้ศึกษาร่วมกับเครื่องมือการออกแบบของ กุปต้า เพื่อออกแบบระบบทำนายดวงชะตา โดยมีจุดประสงค์หลักเพื่อออกแสดงในงาน ภาคกระบังนิทรรศน์ ครั้งต่อไปที่จะถึงนี้
- บทที่ 6 การสร้างระบบ กล่าวถึงการสร้างระบบ จากการออกแบบที่ได้ทำไว้ในบทที่ 3 มาทำการสร้างให้เป็นแอพพลิเคชั่นจริงๆ โดยจะใช้เครื่องมือในการพัฒนาสองชนิดเพื่อเป็นการเปรียบเทียบกัน เครื่องชนิดแรกเป็นของ กุปต้า ส่วนชนิดที่สอง เป็นการใช้ภาษาวิชวลเบสิก (Visual Basic) ในการพัฒนา
- ภาคผนวก กล่าวถึง วิธีการใช้ โปรแกรมการทำนายดวงชะตา พร้อมทั้ง โค้ดการโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

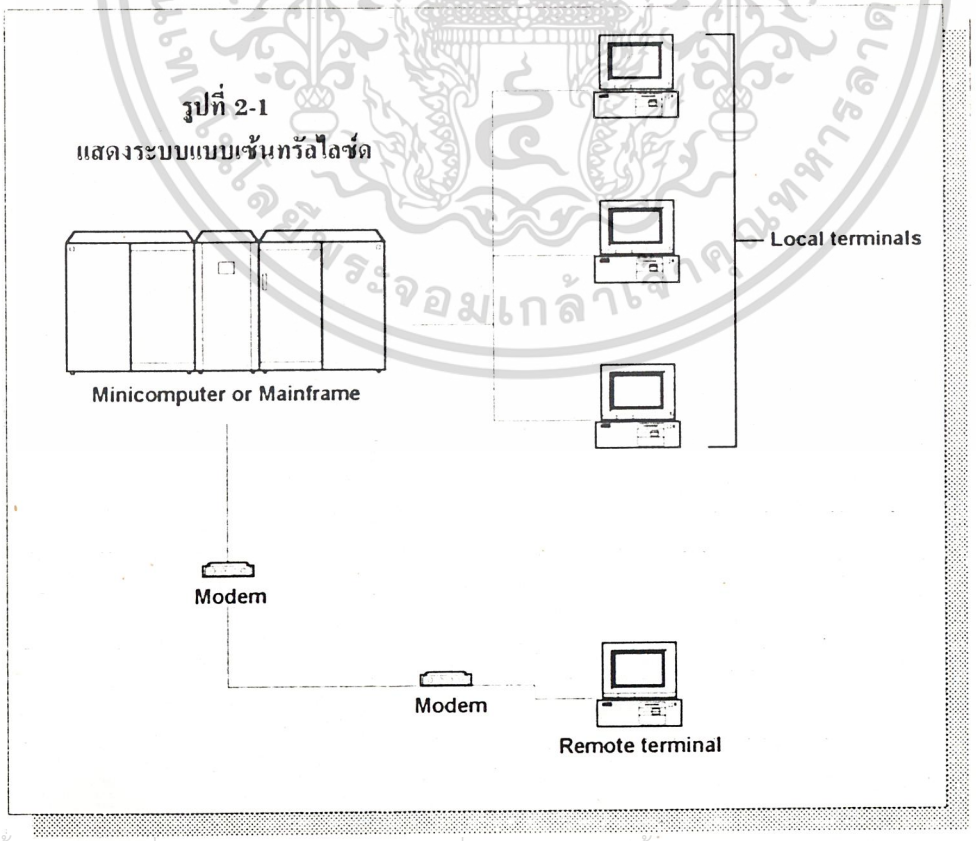
บทที่ 2
หลักการที่เกี่ยวข้อง.

ภายในบทนี้ผมจะข้อกล่าวถึงหลักการและทฤษฎีต่างๆที่เกี่ยวข้องกับโครงงาน เพื่อเป็นพื้นฐานของความเข้าใจในเรื่องอื่นๆที่จะกล่าวต่อๆไปหลังจากบทนี้

2.1 สถาปัตยกรรมของระบบฐานข้อมูล

ระบบคอมพิวเตอร์ที่วิ่งงานฐานข้อมูลนี้เราสามารถจำแนกมันออกเป็น ระดับอย่างกว้างๆได้ 4 ชั้น คือ *เซ็นทรัลไลซ์ด* (Centralized), *พีซี* (PC), *ไคลเอ็นท์เซอร์ฟเวอร์* (Client/Server), และ *ดิสทริบิวเต็ด* (Distributed). ระบบทั้งสี่นี้แตกต่างกันตรงที่ว่า การประมวลผลจริงๆนั้นเกิดที่ใด

2.1.1 ระบบแบบเซ็นทรัลไลซ์ด



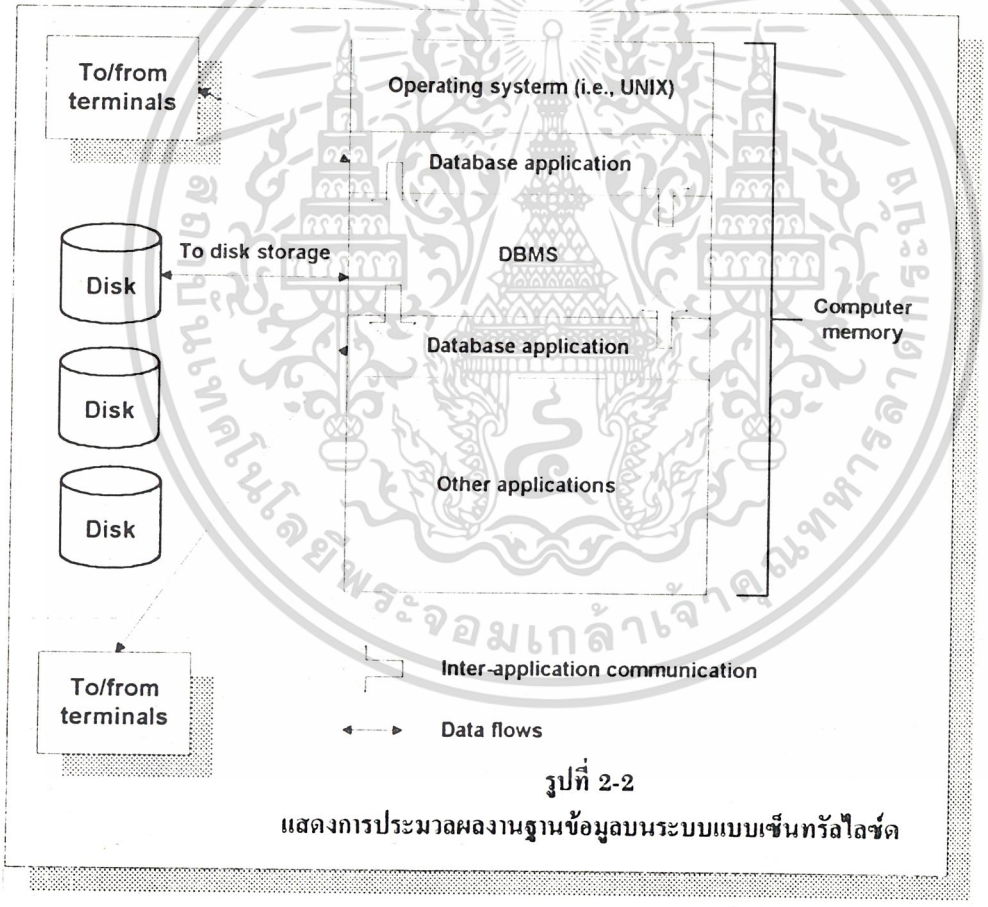
รูปที่ 2-1
แสดงระบบแบบเซ็นทรัลไลซ์ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในระบบที่เป็นแบบ เซ็นทรัลไลซ์ด นั้น โปรแกรมทั้งหมดจะวิ่งอยู่บนระบบหลักที่เรียกว่า โฮสท์(Host) ซึ่งโปรแกรมทั้งหมดนี้ก็รวมไปถึงตัวโปรแกรมดีบีเอ็มเอส (DBMS), โปรแกรมประยุกต์ (Application Program) ที่ทำการเข้าถึง (access) ฐานข้อมูล ตลอดจนแฟกซิลิตี้ส์ (Facilities) ที่ทำหน้าที่ทางการสื่อสารข้อมูลกับเทอร์มินัล (Terminal) ของผู้ใช้ด้วย

ผู้ใช้ในระบบเซ็นทรัลไลซ์ดสามารถเข้าถึงฐานข้อมูลได้โดยผ่านการติดต่อของเทอร์มินัล (Terminal) ทั้งแบบโลคอล (Locally Connected) หรือแบบติดต่อผ่านสายโทรศัพท์ (dial-up หรือ remote) ดังรูปที่ 2-1 เทอร์มินัลเหล่านี้ โดยทั่วไปแล้วจะเป็นแบบดัมพ์ (dump) คือไม่สามารถประมวลผลอะไรเองได้ มีแต่เพียงหน้าจอและคีย์บอร์ด และ ฮาร์ดแวร์ที่ใช้ในการติดต่อกับโฮสท์เท่านั้น

การประมวลผลต่างๆทั้งหมดบนระบบเซ็นทรัลไลซ์ด จะเกิดบนฝั่งเครื่องโฮสท์ และจะต้องมีดีบีเอ็มเอสวิ่งอยู่ก่อนที่โปรแกรมประยุกต์จะสามารถติดต่อเข้าถึงฐานข้อมูลได้. โปรแกรมประยุกต์และดีบีเอ็มเอสจะวิ่งอยู่บนระบบพร้อมกันและติดต่อสื่อสารกันผ่านทางหน่วยความจำที่ใช้ร่า



รูปที่ 2-2
แสดงการประมวลผลงานฐานข้อมูลบนระบบแบบเซ็นทรัลไลซ์ด

มกันหรือ แอพพลิเคชั่นทาสก์แอเรีย (Application Task Areas) ที่จัดการโดยระบบปฏิบัติการ (Operating System) ดีบีเอ็มเอสมีหน้าที่เคลื่อนย้ายข้อมูลไปมาระหว่างระบบดิสก์เก็บข้อมูล รูปที่ 2-2 จะแสดงภาพการติดต่อสื่อสารกันระหว่างโปรแกรมต่างๆบนระบบ โปรแกรมประยุกต์จะติดต่อกับผู้ใช้โดยผ่านเทอร์มินัล และติดต่อกับดีบีเอ็มเอส ในขณะที่ดีบีเอ็มเอสจะติดต่อกับสตอลเล็จดีไวส์ (Storage Device) และกับโปรแกรมประยุกต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดีบีเอ็มเอสที่วิ่งอยู่บนโฮสท์นั้นสามารถเป็นฐานข้อมูลที่มีดาต้าโมเดล (Data Model) ได้ทั้งแบบไฮลาจิกคอล (Hierarchical), เน็ตเวอร์ค (Network) หรือ รีเลชั่นนอล (Relational) ก็ได้

ข้อดีที่สำคัญของระบบแบบเซ็นทรัลไลซ์คก็อยู่ที่ความปลอดภัยของข้อมูล และความสามารถในการจัดการข้อมูลจำนวนมหาศาลบนสตอลเล็งดีไวส์
นอกจากนี้ระบบแบบเซ็นทรัลไลซ์คยังสามารถรับใช้ผู้ใช้พร้อมๆกันได้เป็นจำนวนมากด้วย
ส่วนข้อเสียของระบบแบบเซ็นทรัลไลซ์คก็จะอยู่ที่ค่าใช้จ่ายที่จะใช้ไปในการซื้อ และบำรุงรักษาระบบ และระบบแบบนี้ยังต้องการพนักงานควบคุมระบบที่ได้รับการฝึกสอนมาเป็นอย่างดีแล้วด้วย

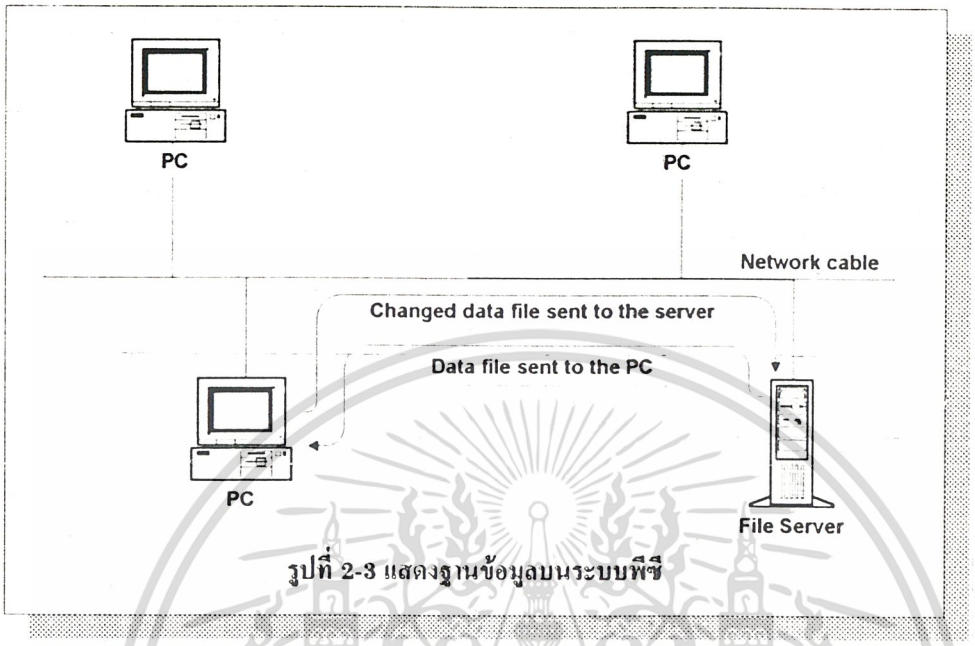
2.1.2 ระบบแบบพีซี

คอมพิวเตอร์ส่วนบุคคล (Personal) หรือที่เรียกกันว่า พีซี นี้ เกิดขึ้นครั้งแรกในสมัยปี 1970 และได้พัฒนาให้มีความสามารถมากขึ้นในเวลาต่อมา
สำหรับโปรแกรมดีบีเอ็มเอสบนพีซีตัวแรกที่ประสบความสำเร็จนั้น ก็ได้แก่ dBASE II ของบริษัท Aston-Tate
เมื่อดีบีเอ็มเอสวิ่งบนเครื่องพีซีนั้น พีซีจะทำหน้าที่เป็นทั้งโฮสท์และเทอร์มินัล
แต่หน้าที่ของทั้งดีบีเอ็มเอสและโปรแกรมประยุกต์จะถูกรวมเข้าเป็นอันเดียวกันซึ่งไม่เหมือนกับบนระบบใหญ่ๆ
โปรแกรมประยุกต์ทางฐานข้อมูลบนเครื่องพีซีจะจัดการกับอินพุต (input) ของผู้ใช้, ผลลัพธ์ที่ออกหน้าจอ, และการเข้าถึงข้อมูลบนดิสก์ด้วย การรวมหน้าที่ของจากดีบีเอ็มเอสและโปรแกรมประยุกต์เข้าด้วยกันนี้
ทำให้ดีบีเอ็มเอสมีความสามารถมากขึ้น, ยืดหยุ่นมากขึ้น, และมีความเร็วสูงขึ้น
อย่างไรก็ตามความสามารถที่เพิ่มขึ้นนี้ก็ต้องแลกกับการลดลงของความปลอดภัย (security) และความถูกต้อง (integrity) ของข้อมูล

ในระยะแรกพีซีเป็นระบบที่วิ่งเป็นเครื่องโดดๆ
แต่เมื่อไม่กี่ปีที่ผ่านมานี้ได้มีความพยายามที่จะเชื่อมต่อเครื่องพีซีหลายๆเครื่องเข้าด้วยกันเป็นระบบแลน (LAN)
บนระบบแลนนี้ ข้อมูลและโปรแกรมประยุกต์ต่างๆจะอยู่บนไฟล์เซอร์ฟเวอร์ (File Server)
และไฟล์เซอร์ฟเวอร์จะทำหน้าที่จัดการการเข้าถึงข้อมูลบนฮาร์ดดิสก์ (hard disks) และทรัพยากรอื่นๆเช่น เครื่องพิมพ์ ที่ใช้ร่วมกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แม้ว่าแลนจะทำให้ผู้ใช้ฐานข้อมูลสามารถใช้ไฟล์ฐานข้อมูลร่วมกันก็ตาม
มันก็ไม่ได้เปลี่ยนแปลงการทำงานของดีบีเอ็มเอสเลย



รูปที่ 2-3 แสดงฐานข้อมูลบนระบบพีซี

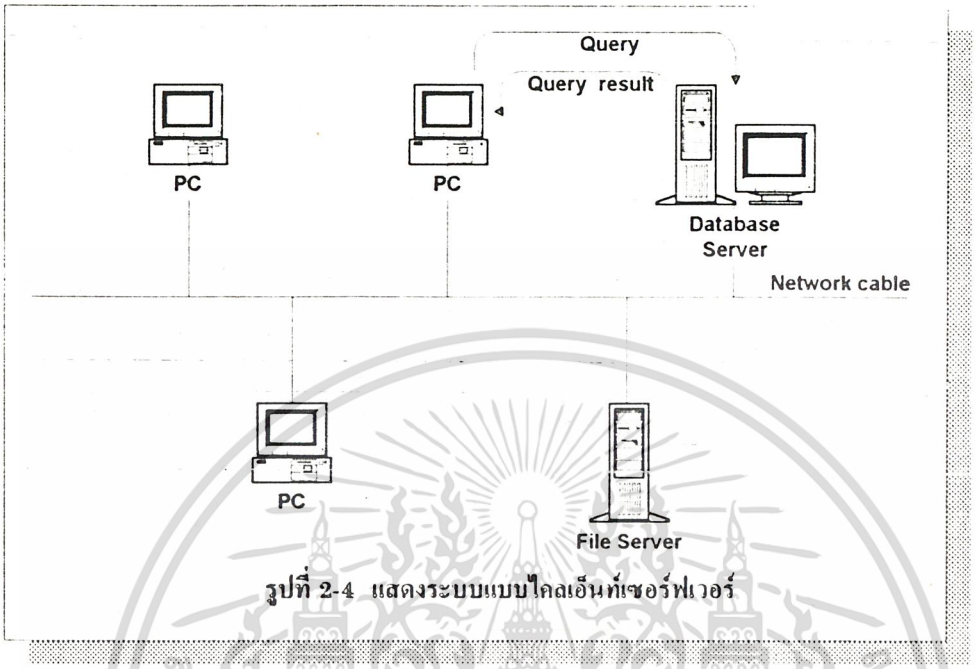
เพราะการประมวลผลก็ยังคงอยู่ที่พีซีที่วิ่งโปรแกรมประยุกต์นั่นเอง
ไฟล์เซอร์ฟเวอร์มีหน้าที่เพียงค้นหาไฟล์ที่ผู้ใช้ต่องการแล้วส่งผ่านระบบเครือข่ายไปยังเครื่องพีซีของผู้ใช้ที่ต้องกา
รไฟล์นั้นนั่นเอง ดังที่สามารถแสดงได้ดังรูปที่ 2-3
แม้ว่าจะมีการเพิ่มเติมความสามารถให้สามารถเข้าถึงข้อมูลพร้อมๆกันได้หลายๆคนก็ตาม
ข้อด้อยของระบบแบบนี้ก็คือประสิทธิภาพของระบบไม่ได้อยู่ที่ความสามารถของไฟล์เซอร์ฟเวอร์
แต่กลับไปอยู่ที่พีซีที่วิ่งตัวดีบีเอ็มเอสจริงๆอยู่ และเมื่อมีผู้ใช้ไฟล์ข้อมูลเดียวกันหลายๆคน
ก็จะเกิดการส่งไฟล์ข้อมูลเหล่านั้นไปให้กับพีซีทุกเครื่องที่ต้องการ
เมื่อเป็นเช่นนี้จึงเป็นสาเหตุที่ทำให้ระบบเครือข่ายช้าลง.

2.1.3 ระบบแบบไคลเอ็นท์เซอร์เวอร์

ในรูปแบบที่ง่ายที่สุดของระบบแบบไคลเอ็นท์เซอร์เวอร์
จะมีการแบ่งการประมวลผลฐานข้อมูลออกกันระหว่าง ไคลเอ็นท์พีซี (Client PC) กับ คาต้าเบสเซอร์ฟเวอร์
(Database Server) โดยไคลเอ็นท์พีซีจะวิ่งโปรแกรมประยุกต์ฐานข้อมูล และ
คาต้าเบสเซอร์ฟเวอร์จะวิ่งโปรแกรมดีบีเอ็มเอส คาต้าเบสเซอร์ฟเวอร์นี้จะวิ่งอยู่บนไฟล์เซอร์ฟเวอร์ด้วย
ไฟล์เซอร์ฟเวอร์ของระบบแลนก็ยังคงต้องให้บริการการใช้รีซอร์สร่วมกันเหมือนเดิม
โปรแกรมประยุกต์ที่วิ่งอยู่บนไคลเอ็นท์พีซีนี้เราจะเรียกว่า ฟร้อนท์เอ็นด์ซิสเต็ม (front-end system)
ซึ่งจะจัดการกับหน้าจอและการประมวลผลอินพุตเอาร์พุต ส่วนคาต้าเบสเซอร์ฟเวอร์เราจะเรียกว่า
แบ็กเอ็นด์ซิสเต็ม (back-end system) ซึ่งจะดูแลเรื่องการประมวลผลข้อมูลและการติดต่อกับดิสก์
ดังตัวอย่างต่อไปนี้ เมื่อผู้ใช้ที่อยู่บนฟร้อนเอ็นด์สร้างรีเควิส (request) ว่าต้องการข้อมูลจากคาต้าเบสเซอร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมแอพพลิเคชั่นก็จะส่งรีเควิสผ่านเน็ตเวอร์คไปที่เซอร์ฟเวอร์
คาต้าเบสเซอร์ฟเวอร์ก็จะทำการค้นหาข้อมูล และส่งเฉพาะข้อมูลเหล่านั้นกลับมาตอบคำถามของผู้ใช้ ดังรูปที่ 2-



รูปที่ 2-4 แสดงระบบแบบไคลเอ็นท์เซอร์ฟเวอร์

4

ข้อดีที่เห็นได้ชัดของระบบแบบไคลเอ็นท์เซอร์ฟเวอร์นี้ก็คือ สามารถลดแทรฟฟิก (traffic)
บนสายเคเบิล (cable)
นอกจากข้อดีนี้แล้วเราจะมาพูดกันถึงข้อดีข้อเสียของระบบแบบไคลเอ็นท์เซอร์ฟเวอร์กันอีกทีในภายหลัง

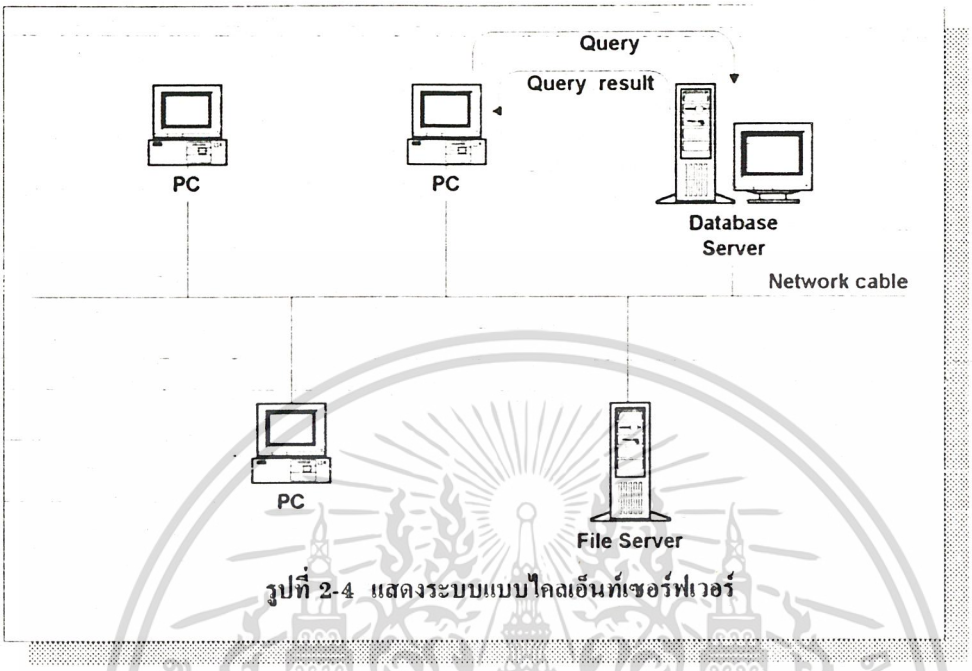
ข้อเสียที่เห็นได้ชัดคือระบบที่มีการเก็บข้อมูลไว้ในที่ที่เดียวนี้
(ไม่ได้เจาะจงถึงไคลเอ็นท์เซอร์ฟเวอร์เท่านั้น แต่รวมถึงทั้งแบบเซ็นทรัลไลซ์ และพีซีด้วย)
ก็คือปัญหาที่จะเกิดกับบริษัทขนาดใหญ่ที่มีสาขาแตกกระจายกันไปตามที่ต่างๆห่างไกลกันแต่มีความต้องการใช้
ข้อมูลร่วมกันบางส่วน จากข้อเสียนี้นำมาซึ่งระบบต่อไป ก็คือระบบแบบดิสทริบิวเต็ดนั่นเอง

2.1.4 ระบบแบบดิสทริบิวเต็ด

สำหรับระบบแบบนี้ในทางอุดมคติ ผู้ใช้จะส่งรีเควิสเพื่อขอข้อมูลไปที่โลคอลโฮสท์ (Local Host)
ถ้าโลคอลโฮสท์พิจารณาแล้วพบว่ามันไม่มีข้อมูลดังกล่าว
มันก็จะไปค้นหาข้อมูลดังกล่าวจากที่ต่างๆบนระบบเครือข่าย
แล้วส่งกลับมาให้ผู้ใช้ได้โดยที่ผู้ใช้ไม่จำเป็นต้องรู้เลยว่า ข้อมูลดังกล่าวนั้นอยู่ที่ใดบนระบบเครือข่าย
แต่ว่าระบบในปัจจุบันนี้ยังสร้างได้ไม่ถึงความสามารถดังกล่าวอย่างแท้จริง รูปที่ 2-5
จะแสดงรูปแบบของระบบแบบดิสทริบิวเต็ด เมื่อผู้ใช้สร้างและส่งคาต้าคิวรี่ (data query)
ไปยังโลคอลคาต้าเบสเซอร์ฟเวอร์ (Local Database Server)
คาต้าเบสเซอร์ฟเวอร์ก็จะส่งรีเควิสข้อมูลส่วนที่มันไม่มีไปบนเครือข่ายไปถึงเครื่องเมนเฟรม (Mainframe)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมแอพพลิเคชั่นก็จะส่งรีเควิสผ่านเน็ตเวอร์คไปที่เซอร์ฟเวอร์
คาต้าเบสเซอร์ฟเวอร์ก็จะทำการค้นหาข้อมูล และส่งเฉพาะข้อมูลเหล่านั้นกลับมาตอบคำถามของผู้ใช้ ดังรูปที่ 2-



รูปที่ 2-4 แสดงระบบแบบไคลเอ็นท์เซอร์ฟเวอร์

4

ข้อดีที่เห็นได้ชัดของระบบแบบไคลเอ็นท์เซอร์ฟเวอร์นี้ก็คือ สามารถลดแทรฟฟิก (traffic)
บนสายเคเบิล (cable)
นอกจากข้อดีนี้แล้วเราจะมาพูดกันถึงข้อดีข้อเสียของระบบแบบไคลเอ็นท์เซอร์ฟเวอร์กันอีกทีในภายหลัง

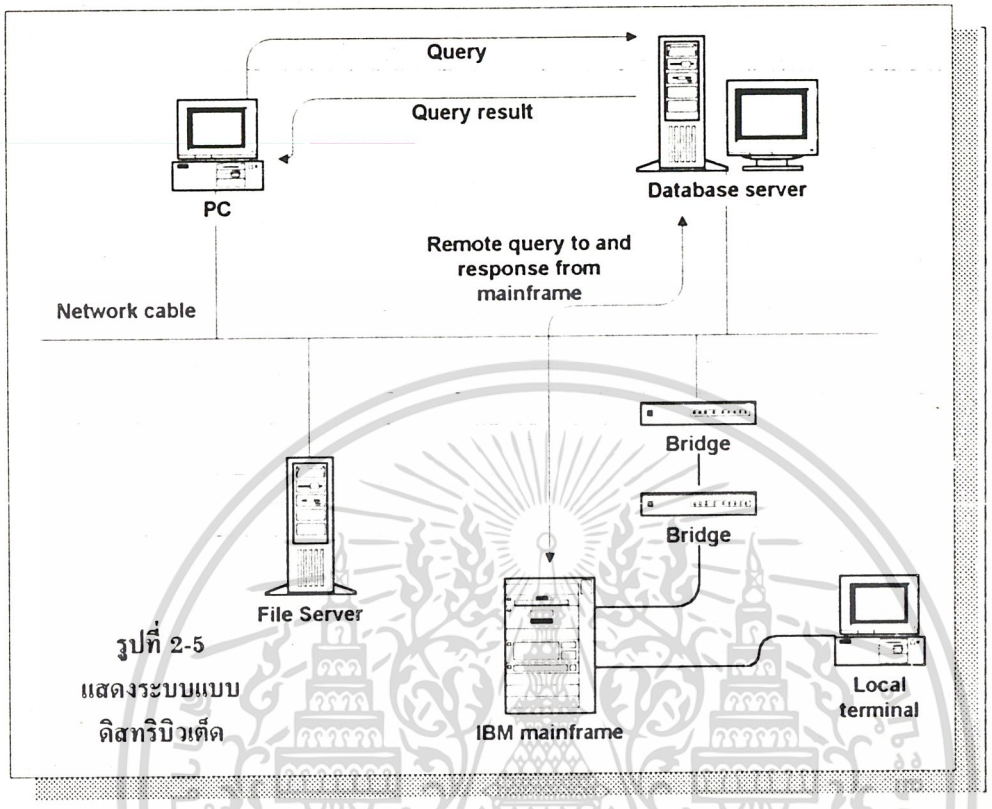
ข้อเสียที่เห็นได้ชัดก็ระบบที่มีการเก็บข้อมูลไว้ในที่ที่เดียวนี้
(ไม่ได้เจาะจงถึงไคลเอ็นท์เซอร์ฟเวอร์เท่านั้น แต่รวมถึงทั้งแบทเช่นทรัลไอซ์ด และพีซีด้วย)
ก็คือปัญหาที่จะเกิดกับบริษัทขนาดใหญ่ที่มีสาขาแตกกระจายกันไปตามที่ต่างๆห่างไกลกันแต่มีความต้องการใช้
ข้อมูลร่วมกันบางส่วน จากข้อเสียนี้นำมาซึ่งระบบต่อไป ก็คือระบบแบบดิสทริบิวเต็ดนั่นเอง

2.1.4 ระบบแบบดิสทริบิวเต็ด

สำหรับระบบแบบนี้ในทางอุดมคติ ผู้ใช้จะส่งรีเควิสเพื่อขอข้อมูลไปที่โลคอลโฮสท์ (Local Host)
ถ้าโลคอลโฮสท์พิจารณาแล้วพบว่ามันไม่มีข้อมูลดังกล่าว
มันก็จะไปค้นหาข้อมูลดังกล่าวจากที่ต่างๆบนระบบเครือข่าย
แล้วส่งกลับมาให้ผู้ใช้ได้โดยที่ผู้ใช้ไม่จำเป็นต้องรู้เลยว่า ข้อมูลดังกล่าวนั้นอยู่ที่ใดบนระบบเครือข่าย
แต่ว่าระบบในปัจจุบันนี้ยังสร้างได้ไม่ถึงความสามารถดังกล่าวอย่างแท้จริง รูปที่ 2-5
จะแสดงรูปแบบของระบบแบบดิสทริบิวเต็ด เมื่อผู้ใช้สร้างและส่งคาต้าคิวรี่ (data query)
ไปยังโลคอลคาต้าเบสเซอร์ฟเวอร์ (Local Database Server)
คาต้าเบสเซอร์ฟเวอร์ก็จะส่งรีเควิสข้อมูลส่วนที่มันไม่มีไปบนเครือข่ายไปถึงเครื่องเมนเฟรม (Mainframe)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากนั้นมันก็จะได้รับคำตอบของคิวรี่นั้นกลับมาที่โลคอลคาต้าเบสเซอร์ฟเวอร์



รูปที่ 2-5
แสดงระบบแบบ
ดิสทริบิวเต็ด

คาต้าเบสเซอร์ฟเวอร์ก็จะทำการรวมข้อมูลเหล่านั้นกับข้อมูลที่มันมี และส่งกลับข้อมูลทั้งหมดกลับมาให้ผู้ใช้

2.2

เทคโนโลยีของระบบฐานข้อมูลแบบไคลเอ็นท์เซอร์ฟเวอร์

2.2.1 ความหมายของคำว่า ไคลเอ็นท์เซอร์ฟเวอร์

โดยปรกติแล้วเรามักจะมองหรือเรียกระบบแลนว่าเป็นระบบแบบไคลเอ็นท์เซอร์ฟเวอร์ เพราะว่าเวอร์คสเตชั่น (Workstation) หรือไคลเอ็นท์ จะขอบริการพวก ข้อมูล, ไฟล์โปรแกรม หรือ การพิมพ์ จากเซอร์ฟเวอร์ แต่อย่างไรก็ตาม คำว่า ไคลเอ็นท์เซอร์ฟเวอร์ ในปัจจุบันนี้ถูกยอมรับโดยแพร่หลายว่าหมายถึงระบบใดๆที่มีการแบ่งการประมวลผลข้อมูลกันระหว่างคอมโพเนนท์ (Component) หรือส่วนประกอบที่แยกกัน.

จากคำจัดความดังกล่าวนี้, ระบบไคลเอ็นท์เซอร์ฟเวอร์ไม่ได้ถูกจำกัดกับคาต้าเบสแอพพลิเคชั่น (Database Application) เพียงอย่างเดียว แอพพลิเคชั่นใดๆที่มีส่วนที่เป็นยูสเซอร์อินเตอร์เฟส (user interface) หรือที่เรียกได้อีกอย่างว่า ฟร้อนท์เอ็นด์ วิ่งอยู่ที่ไคลเอ็นท์ และมีส่วนการประมวลผลวิ่งอยู่บนเซอร์ฟเวอร์ หรือแบ็กเอ็นด์ ระบบเหล่านี้ก็เป็นรูปแบบของไคลเอ็นท์เซอร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.2 ความสามารถของระบบฐานข้อมูลไคลเอ็นท์เซอร์ฟเวอร์

2.2.2.1 ข้อดีของระบบฐานข้อมูลไคลเอ็นท์เซอร์ฟเวอร์

ข้อดีอย่างแรกของระบบไคลเอ็นท์เซอร์ฟเวอร์เกิดจากการแบ่งการประมวลผลระหว่างไคลเอ็นท์และเซอร์ฟเวอร์ เพราะการประมวลผลฐานข้อมูลถูกกระทำที่ฝั่งแบ็กเอ็นด์
ดังนั้นจึงทำให้ความเร็วของดีบีเอ็มเอสไม่ถูกยึดติดกับความเร็วของเวอร์คสเตชั่น
นอกจากนี้เวอร์คสเตชั่นยังต้องการเพียงให้สามารถวิ่งฟร้อนท์เอ็นด์ซอฟแวร์ให้ได้เท่านั้นจึงช่วยยืดอายุของเครื่องพีซีรุ่นเก่า เพราะไม่จำเป็นต้องใช้เครื่องที่มีความสามารถสูงๆ

การแบ่งดังกล่าวยังช่วยลดแทรฟฟิกของระบบเครือข่าย
เนื่องจากระบบไม่ต้องส่งไฟล์ฐานข้อมูลทั้งหมดกลับไปกลับมา
จึงทำให้การติดต่อกันละหว่างไคลเอ็นท์และเซอร์ฟเวอร์เป็นไปอย่างรวดเร็วด้วย
นอกเหนือจากเรื่องประสิทธิภาพแล้ว
การแบ่งดังกล่าวทำให้ผู้ใช้ไม่ต้องยึดติดกับระบบหรือแพล็ทฟอร์มใดแบบหนึ่ง ในระบบไคลเอ็นท์เซอร์ฟเวอร์เวอร์คสเตชั่นอาจเป็นคอมแพททิเบิลพีซี (compatible PC), แม็คอินทอชส์ (Machintoshes), ยูนิกส์ (UNIX) หรือหลายๆแบบรวมกันก็ได้ นอกจากนี้อาจวิ่งโปรแกรมปฏิบัติการที่ไม่เหมือนกันก็ได้

ข้อดีอีกอย่างหนึ่งของไคลเอ็นท์เซอร์ฟเวอร์ก็คือการสามารถรักษาความถูกต้องของข้อมูลได้
เนื่องจากปัจจุบันนี้คาต้าเบสเซอร์ฟเวอร์จะวิ่งดีบีเอ็มเอสที่ใช้ค้าต้าโมเดลแบบรีเลชั่นนอล
และผู้ใช้จะถูกกั้นไม่ให้สามารถเข้าถึงข้อมูลได้โดยตรงโดยไม่ผ่านดีบีเอ็มเอส
นอกจากนี้ดีบีเอ็มเอสก็สามารถใช้บริการอื่นๆที่ช่วยรักษาความปลอดภัยของข้อมูลได้ดีด้วย
ไม่ว่าจะเป็นการเข้ารหัสข้อมูล การทำเรียลไทม์แบ็กค์อัพ (realtime backup) ข้อมูลลงสู่เทป เป็นต้น

2.2.2.2 ข้อเสียของระบบฐานข้อมูลไคลเอ็นท์เซอร์ฟเวอร์

ข้อเสียหลักๆของระบบไคลเอ็นท์เซอร์ฟเวอร์ก็อยู่ที่ค่าใช้จ่ายที่เพิ่มขึ้นในการจัดการระบบ และการหาบุคคลากรที่มาคอยดูแลคาต้าเบสเซอร์ฟเวอร์ บนระบบเครือข่ายขนาดเล็ก ผู้ควบคุมเครือข่าย (network administrator) อาจเป็นผู้ดูแลคาต้าเบสเซอร์ฟเวอร์เองเลยก็ได้
แต่อย่างไรก็ตามเมื่อมีจำนวนผู้ใช้มากขึ้นหรือตัวคาต้าเบสเองโตขึ้นเรื่อยๆ ก็ควรจะต้องมีผู้มาดูแลเฉพาะ

นอกจากนี้ยังต้องมีค่าใช้จ่ายเพิ่มเติมในค้ายฮาร์ดแวร์ด้วย คือ
ถึงแม้ว่าผู้ขายดีบีเอ็มเอสแบบไคลเอ็นท์เซอร์ฟเวอร์จะอ้างว่ามันสามารถวิ่งอยู่บนฮาร์ดแวร์ตัวเดียวกันกับไฟล์เซอร์ฟเวอร์ได้เลยก็ตาม
คาต้าเบสเซอร์ฟเวอร์ก็ควรจะถูกวิ่งบนเครื่องที่อุทิศเพื่องานนี้โดยเฉพาะเพื่อสร้างความมั่นใจในเรื่องประสิทธิภาพและความถูกต้องและปลอดภัยของข้อมูล

ราคาของซอฟแวร์ไคลเอ็นท์เซอร์ฟเวอร์ก็มักจะมีราคาสูงกว่าโปรแกรมโดยทั่วไป
และจะต้องมีค่าใช้จ่ายในค้านการฝึกอบรมโปรแกรมเมอร์ (programmer) เพื่อทำงานกับระบบใหม่อีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

นอกจากเรื่องต่างๆที่ได้กล่าวมาแล้ว
ความซับซ้อนของระบบไคลเอ็นท์เซอร์ฟเวอร์ก็เป็นอีกเรื่องหนึ่งที่ต้องคำนึงถึงด้วย
เมื่อระบบไคลเอ็นท์เซอร์ฟเวอร์มีองค์ประกอบมาก
จึงทำให้เป็นการยากมากขึ้นในการตรวจหาข้อผิดพลาดที่อาจเกิดขึ้น
และการติดตั้งระบบแบบไคลเอ็นท์เซอร์ฟเวอร์ก็ยากกว่าด้วยเหตุผลในทำนองเดียวกัน

2.2.3 แพล็ทฟอร์ม (Platforms) ของระบบฐานข้อมูลไคลเอ็นท์เซอร์ฟเวอร์

แพล็ทฟอร์มนี้หมายถึงการประกอบกันของฮาร์ดแวร์และซอฟแวร์ที่ดีบีเอ็มเอสแบบไคลเอ็นท์เซอร์ฟเวอร์วิ่งบน เราสามารถแบ่งแพล็ทฟอร์มของระบบออกได้เป็น 4 แบบคือ พีซี, ยูนิกส์เวอร์คสเตชั่น, มินิคอมพิวเตอร์ (minicomputer), เมนเฟรม
แม้ว่าระบบไคลเอ็นท์เซอร์ฟเวอร์มักจะใช้กับแพล็ทฟอร์มแบบพีซี แต่ว่าแพล็ทฟอร์มทั้ง 4 ก็มีข้อดีและข้อเสียของมันเฉพาะแตกต่างกันไป

ในขณะที่ฮาร์ดแวร์ของระบบมีให้เลือกใช้มากมายตามคุณสมบัติ และ ความสามารถ แต่มีคุณสมบัติที่ต้องมีเหมือนๆกันของระบบปฏิบัติการที่ใช้สำหรับระบบเหล่านี้
ระบบปฏิบัติการหรือที่เรียกกันว่า โอเอส (OS) นี้ เป็นซอฟแวร์สำคัญของระบบ โดยทำหน้าที่เป็นตัวประสานฮาร์ดแวร์และโปรแกรมประยุกต์เข้าด้วยกันเพื่อให้สามารถปฏิบัติการตามที่ต้องการได้ โดยปกติแล้วแอพพลิเคชั่นมักจะถูกเขียนขึ้นมาเฉพาะสำหรับโอเอสแต่ละตัว
สำหรับโอเอสที่เป็นที่รู้จักกันดีก็ได้แก่ MS/PC-DOS, OS/2, ยูนิกส์ชนิดต่างๆ, VMS ของบริษัทเด็ค (DEC) และ MVS/XA ที่วิ่งอยู่บนเครื่องเมนเฟรมของ ไอบีเอ็ม (IBM) เป็นต้น

คุณสมบัติหลักที่โอเอสต้องมีเพื่อรองรับดีบีเอ็มเอสแบบไคลเอ็นท์เซอร์ฟเวอร์ก็คือ มัลติทาสกิ้ง (Multitasking) มัลติทาสกิ้งคือการที่มีโปรแกรมประยุกต์สามารถวิ่งบนระบบไปพร้อมๆกันนั่นเอง
มัลติทาสกิ้งจะทำให้ดีบีเอ็มเอสสามารถจัดการกับคิวรี่ของผู้ใช้หลายๆคนได้พร้อมๆกันโดยใช้การแบ่งเวลาที่ใช้ในการประมวลผลของซีพียู (CPU) ให้กับแต่ละโพรเซส (Process)

นอกจากนี้โอเอสอาจจะมีความสามารถที่เรียกว่า มัลติยูสเซอร์ (Multiuser) ด้วย คือสามารถรองรับผู้ใช้ที่ทำงานอยู่บนระบบหลายๆคนพร้อมกันได้
แต่ความสามารถทางมัลติยูสเซอร์ไม่ได้เป็นข้อดีหรือข้อเสียพิเศษใดๆสำหรับดีบีเอ็มเอสแบบไคลเอ็นท์เซอร์ฟเวอร์ โอเอสจะมีความสามารถนี้หรือไม่ก็ได้

2.2.4 การสื่อสารข้อมูล

เนื่องจากระบบฐานข้อมูลแบบไคลเอ็นท์เซอร์ฟเวอร์นี้จะแยกการประมวลผลระหว่างฟร้อนเอ็นท์และแบ็กเอ็นท์หรือล้ำเต้าเบสเซอร์ฟเวอร์
ระบบเครือข่ายจึงเข้ามาเป็นตัวกลางในการสื่อสารกันระหว่างทั้งสองส่วนของระบบ
แม้ว่าเรื่องเครือข่ายคอมพิวเตอร์จะอยู่นอกเหนือจากขอบเขตของเรื่องที่เรากำลังศึกษาอยู่นี้

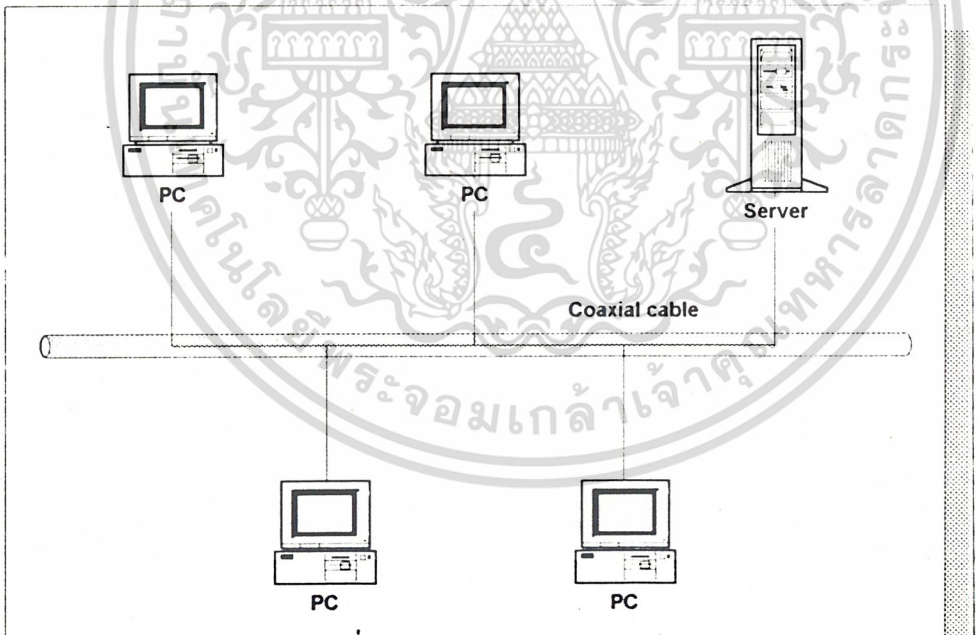
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
034928

แต่ขอยกแนวความคิดพื้นฐานบางเรื่องมากล่าวเพื่อเพิ่มความเข้าใจถึงการสื่อสารข้อมูลในระบบไคลเอ็นท์เซอร์ฟเวอร์ให้ดียิ่งขึ้น

2.2.4.1 ฮาร์ดแวร์และซอร์ฟแวร์ของระบบเครือข่าย

ไคลเอ็นท์ติดต่อกับเซอร์ฟเวอร์โดยผ่านเครือข่ายที่เป็นการรวมกันของฮาร์ดแวร์และซอฟแวร์ โดยเครื่องพีซี, เวอร์คสเตชั่น และเซอร์ฟเวอร์จะมีเน็ตเวอร์คอินเตอร์เฟสการ์ด (network interface card) หรือแผ่น
วงจรที่ใช้ในการเชื่อมต่อกับเครือข่ายติดตั้งอยู่กับตัวเครื่อง
โดยปกติแล้วถ้าเครือข่ายของเครื่องคอมพิวเตอร์อยู่ภายในอาคารเดียวกันเราจะเรียกมันว่าแลน หรือ โลคอลแอเรียเน็ตเวอร์ค (local area network) แต่ถ้าระบบเครือข่ายนั้นถูกขยายให้มีการติดต่อกันกว้างขึ้นไป เราก็จะเรียกมันว่าเป็นแวน (WAN) หรือ ไวด์แอเรียเน็ตเวอร์ค (wide area network)

ในปัจจุบันนี้มีโทโพโลจีส์ (topology) ของระบบแลนหรือลักษณะการเชื่อมต่อของระบบแลนใช้กันอย่างแพร่หลายอยู่สามแบบด้วยกันคือ อีเทอร์เน็ท (Ethernet), อาร์คเน็ท (ARCnet), และ โทเค็นริง (Token Ring) โดยปกติ ระบบที่เป็นอีเทอร์เน็ทจะวิ่งที่ความเร็วประมาณ 10 เมกะบิทต่อวินาที (Mbps) และใช้สายเคเบิลแบบโคเอ็กเชียล (coaxial) ต่อกันในแบบบัส (Bus) ดังในรูปที่ 2-6
หรืออาจใช้สายเคเบิลแบบทวิสท์แพร์ (twisted-pair, TP) ต่อเข้ากับฮับ (hup) ในรูปแบบสตาร์ (Star) ดังในรูปที่

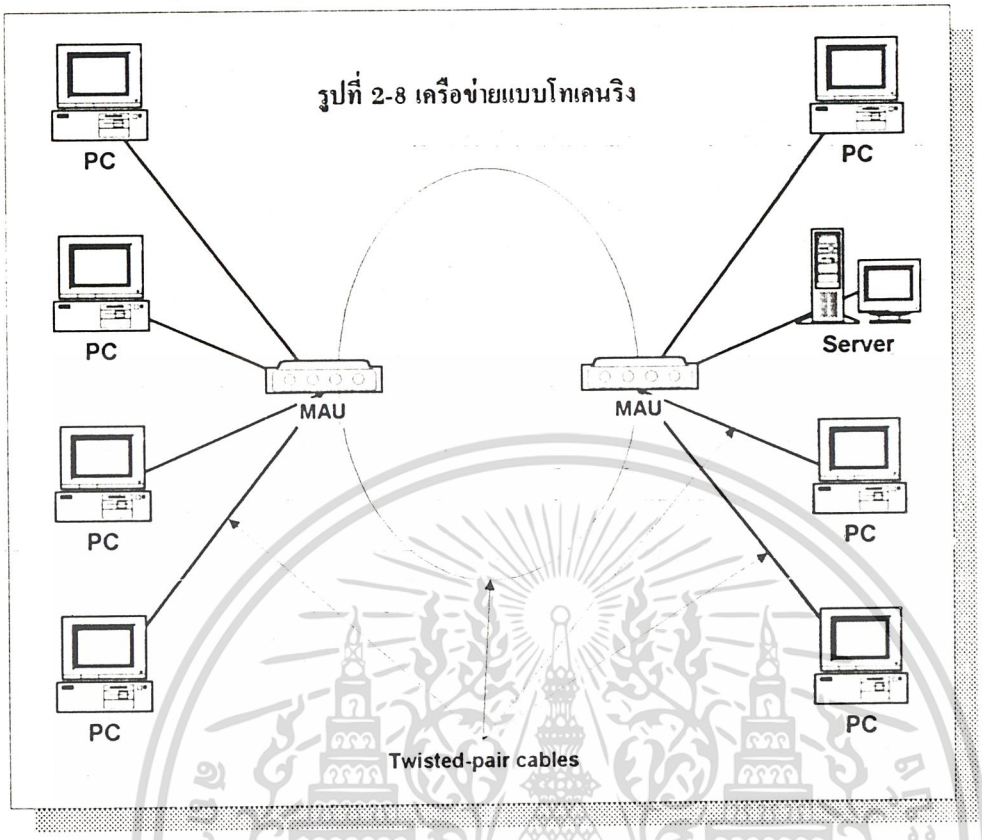


รูปที่ 2-6 แสดงเครือข่ายแบบอีเทอร์เน็ท

2-7 ก็ได้

สำหรับโทเค็นริงจะวิ่งที่ความเร็ว 4 หรือ 16 Mps ดังรูปที่ 2-8
สังเกตว่าเครือข่ายที่เชื่อมต่อกันแบบโทเค็นริงจะดูคล้ายกับการต่อแบบสตาร์ที่ใช้ฮับของอีเทอร์เน็ท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2-8 เครือข่ายแบบโทเคนริง

อาร์คเน็ท เป็นเครือข่ายแบบโทเค็นบัส (token-bus network) ที่วิ่งด้วยความเร็ว 2 Mps โดยใช้แอ็กทีฟ (active) และ แพสซีฟ (passive) ฮับต่อกับสายเคเบิลแบบโคแอ็กเชียล ดังรูปที่ 2-9

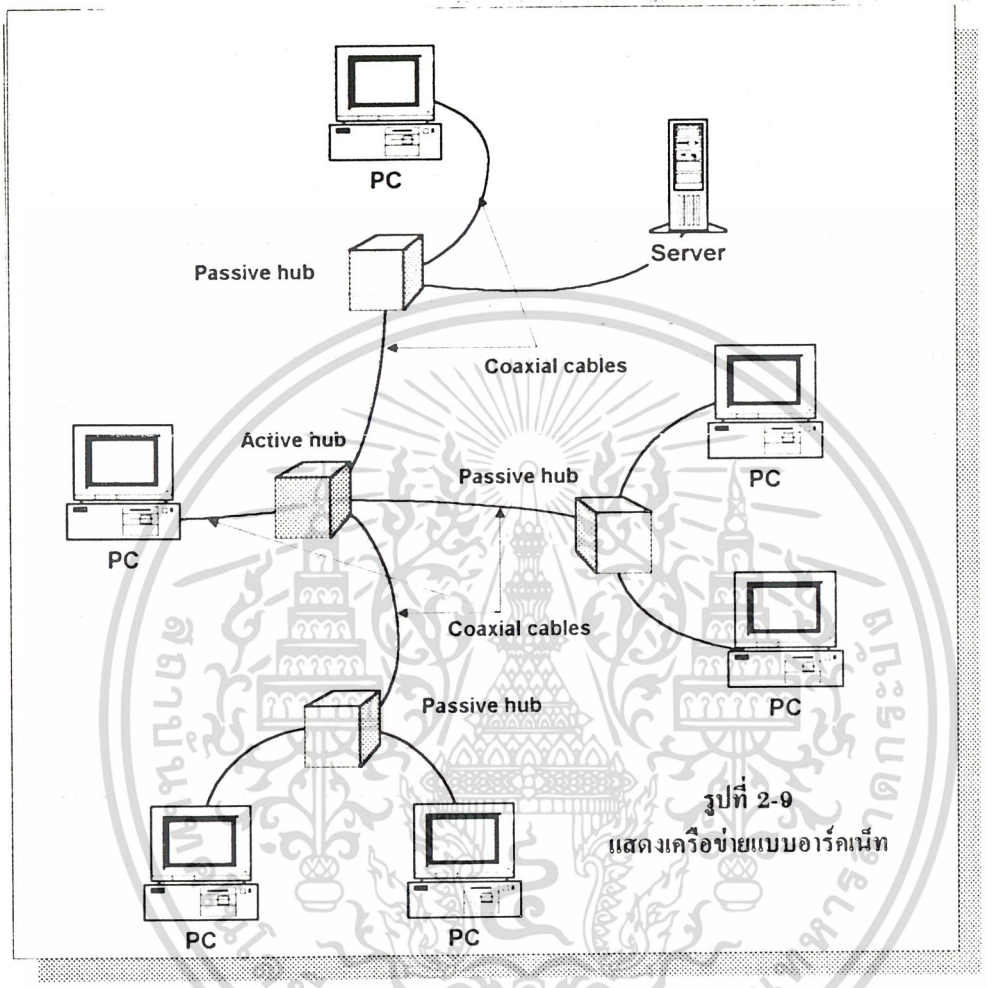
เมื่อพีซีต่อเข้ากับเครือข่าย เราจะต้องทำการติดตั้ง ซอฟแวร์ไดร์เวอร์ (software driver) เพื่อทำให้พีซีเครื่องนั้นสามารถติดต่อผ่านสายเคเบิลของเครือข่ายได้ นอกจากนี้อาจต้องติดตั้งไดร์เวอร์อื่นๆเข้าไปด้วยเพื่อที่จะทำให้เครื่องสามารถเข้าถึงข้อมูลหรือไฟล์ต่างๆบนไฟล์เซอร์ฟเวอร์, ส่งงานไปพิมพ์ที่เครื่องพิมพ์ที่ใช้ร่วมกัน, หรือติดต่อกับดาต้าเบสเซอร์ฟเวอร์ได้

2.2.4.2 เน็ตเวอร์คโพรโตคอล (Network Protocols)

โพรโตคอลคือมาตรฐานที่ใช้ในการติดต่อสื่อสารกันระหว่างเครื่องคอมพิวเตอร์ผ่านระบบเครือข่าย มีเน็ตเวอร์คโพรโตคอลที่เป็นของเฉพาะสำหรับแต่ละผู้ผลิต อยู่มากมายในปัจจุบัน แต่มีเพียงไม่กี่อันที่ถูกออกแบบมาเพื่อระบบฐานข้อมูลแบบไคลเอ็นท์เซอร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อไปนี้เป็นตัวอย่างโพรโตคอลที่ใช้กันอยู่ทั่วไปในการสื่อสารบนระบบเครือข่าย : NetWare LAN ใช้ IPX/SPX ของบริษัทโนเวล (Novell) เป็นโพรโตคอล LAN Manager ของบริษัทไมโครซอฟท์ (Microsoft)



รูปที่ 2-9
แสดงเครือข่ายแบบอาร์คเน็ท

และ LAN Server ของไอบีเอ็ม ใช้โพรโตคอล NetBIOS DEC-based LAN ก็ใช้โพรโตคอล DECNet เป็นต้น

โพรโตคอลที่ใช้ในการสื่อสารข้ามแพล็ทฟอร์มมีชื่อว่า TCP/IP ซึ่งย่อมาจาก Transmission Control Protocol/Internet Protocol แต่เดิม TCP/IP ถูกพัฒนามาเป็นเน็ตเวอร์คโพรโตคอลสำหรับยูนิกส์ ปัจจุบัน TCP/IP มีใช้บนทุกๆแพล็ทฟอร์มและทุกระบบปฏิบัติการ และได้รับความนิยมอย่างแพร่หลายในการนำมาใช้ในการเชื่อมต่อพีซี, เวอร์คสเตชั่น, มินิคอมพิวเตอร์ และเมนเฟรมเข้าด้วยกัน

ถึงแม้ว่าโพรโตคอลที่กล่าวมาแล้วข้างต้นจะเพียงพอที่จะใช้ในการสื่อสารบนแลนแล้วก็ตาม แต่โพรโตคอลเหล่านี้ไม่เพียงพอสำหรับระบบไคลเอ็นท์เซอร์ฟเวอร์เลยเพราะต้องการโพรโตคอลที่มีระดับสูงกว่านั้น ในปัจจุบันดีบีเอ็มเอสของแต่ละบริษัทจึงต้องกำหนดโพรโตคอลเฉพาะของเขาเองขึ้นมาเพื่อทำหน้าที่นี้ เช่น โพรโตคอลที่มีชื่อว่า Pipes เป็นโพรโตคอลที่ใช้สำหรับคาต้าเบสเซอร์ฟเวอร์ที่วิ่งอยู่บน OS/2 เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3 แนวความคิดแบบออพเจ็กต์โอเรียนเต็ด (Object Oriented)

ต่อไปนี้เราจะมาศึกษาแนวความคิดที่กำลังได้รับความนิยมในวงการคอมพิวเตอร์ในปัจจุบัน ซึ่งแนวความคิดดังกล่าวก็คือแนวความคิดของออพเจ็กต์โอเรียนเต็ดนั่นเอง มันดูเหมือนว่า, การใช้คำว่า 'ออพเจ็กต์โอเรียนเต็ด' จะแสดงให้เห็นถึงความทันสมัย ความสามารถที่ดี และ มีประโยชน์ต่อวงการสารสนเทศในปัจจุบันมาก อย่างไรก็ตามการนำแนวความคิดนี้มาใช้ให้เกิดประโยชน์อย่างมีประสิทธิภาพนั้น จำต้องมีความเข้าใจถึงแนวความคิดดังกล่าวอย่างถ่องแท้ด้วยเช่นกัน

คำว่าออพเจ็กต์โอเรียนเต็ดที่เรากำลังพูดถึงนี้ ไม่ได้ถูกจำกัดอยู่เฉพาะกับการโปรแกรมมิ่งแบบออพเจ็กต์โอเรียนเต็ดเพียงอย่างเดียว แต่ยังรวมไปถึงปรัชญาทั้งหมดในการวิเคราะห์ระบบ (system analysis), การออกแบบระบบ (system design), การออกแบบฐานข้อมูล (database design), และเรื่องอื่นๆที่เกี่ยวข้องในสาขาเดียวกันนี้ด้วย ภายในบทนี้จะแนะนำแนวความคิดพื้นฐานและคำเฉพาะที่ใช้ในระเบียบวิธีทางออพเจ็กต์โอเรียนเต็ด (object-oriented methods) โดยเราจะเริ่มต้นดูกันถึงเรื่องราวที่เป็นภูมิหลังของแนวความคิด ดังต่อไปนี้

2.3.1 ภูมิหลัง

การเกิดขึ้นของออพเจ็กโอเรียนเต็ดเป็นการสะท้อนประวัติศาสตร์ทั้งหมดของการคำนวณ (computing) โดยงานทางการคำนวณในยุคแรกๆ ซึ่งย้อนหลังไปถึงในยุคปลายของทศวรรษที่ 1940 จะเกี่ยวข้องกับสิ่งที่เราเรียกกันในปัจจุบันว่า การโปรแกรม หรือ โปรแกรมมิ่ง (programming) ตราบจนในภายหลังที่เพิ่มจะมีความสนใจเกี่ยวกับ การออกแบบและการวิเคราะห์ระบบขึ้นมาเป็นเรื่องที่แยกออกไป ในทำนองเดียวกันนี้, มันเป็นเพราะการโปรแกรมแบบออพเจ็กต์โอเรียนเต็ดนั่นเองที่ดึงดูดความสนใจในระยะแรก และ ในภายหลัง การออกแบบและการวิเคราะห์โดยวิธีออพเจ็กต์โอเรียนเต็ดจึงค่อยเกิดขึ้นมาเป็นที่สนใจของผู้คนในวงการคอมพิวเตอร์

ประวัติศาสตร์ของออพเจ็กต์โอเรียนเต็ดเริ่มต้นที่การพัฒนาภาษา Simula ในนอร์เวย์ ตอนปี 1967 ภาษา Simula นี้เป็นภาษาที่ใช้ในการเลียนแบบเหตุการณ์ (discrete event simulation) และ ได้ถูกพัฒนาต่อมาเป็นอีกภาษาหนึ่งที่สมควรจะได้รับว่าใช้แนวคิดทางออพเจ็กต์โอเรียนเต็ดอย่างแท้จริง ซึ่งภาษานั้นก็คือ ภาษา Smalltalk ที่เกิดในช่วงทศวรรษที่ 1970

ซิมมูเลชั่นโมเดลลิ่ง (Simulation modelling) เป็นปัญหาที่ยากมากสำหรับโปรแกรมเมอร์ที่ใช้ภาษาแบบคอนเวนชั่นนอล (conventional) หรือภาษาในยุคที่สาม (third-generation language) เพราะโปรแกรมเมอร์จะต้องเข้าใจถึงฟังก์ชั่นนอลโฟลว์ (functional flow) ของการควบคุมอย่างลึกซึ้ง ซึ่งเป็นสิ่งที่เป็นเรื่องง่ายในภาษาที่คอนโทรลโฟลว์ (control flow)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถูกอธิบายในรูปแบบของออพเจ็กต์ที่ซับซ้อนซึ่งเปลี่ยนสถานะและมีอิทธิพลต่อเหตุการณ์จากช่วงหนึ่งไปถึงอีกช่วงหนึ่ง ในภาษาแบบออพเจ็กต์โอเรียนเต็ด, ฟังก์ชั่นนอลโฟลว์จะถูกแทนด้วยการส่งผ่านแมสเสจ (message) กันระหว่างออพเจ็กต์ต่างๆ ซึ่งเป็นสาเหตุให้เกิดการเปลี่ยนแปลงสถานะของออพเจ็กต์
ดังนั้นการโปรแกรมแบบออพเจ็กต์โอเรียนเต็ดจึงเป็นวิธีที่ดูเป็นธรรมชาติเพราะโครงสร้างของโปรแกรมจะสะท้อนถึงปัญหาที่ต้องการแก้โดยตรง
นอกจากนี้มันยังเป็นเรื่องที่ง่ายกว่าสำหรับปัญหาการซิมมูเลชั่นที่จะแบ่งแยกวัตถุต่างๆอย่างเด่นชัด เป็นต้นว่า รถบนถนน, เครื่องยนต์ในสายการผลิต และอื่นๆ

คำว่า 'ออพเจ็กต์โอเรียนเต็ด' ได้มาสู่ภาษาคอมพิวเตอร์ก็เมื่อมีการสร้างภาษา Smalltalk ขึ้น ส่วนใหญ่ของ Smalltalk ได้ถูกพัฒนาขึ้นที่ศูนย์วิจัยของบริษัทซีรอกส์ (Xerox) ที่ Palo Alto (PARC) ในช่วงแรกๆนี้ Smalltalk เป็นซอฟแวร์หลักๆตัวหนึ่งของ Dynabook ซึ่งเป็นเครื่องแบบหนึ่งของ PARC และได้รับอิทธิพลจากทั้งแนวความคิดเรื่องคลาสส์และการสืบทอดหรืออินเฮียริแทนส์ (inheritance) ของภาษา Simula และคุณสมบัติที่เป็นแบบโครงสร้าง (structural feature) ของภาษา LISP แม้ว่า Smalltalk จะดึงความคิดของภาษามาจาก LISP ค้วยก็ตามแต่ภาษา Smalltalk กับภาษา LISP ก็แตกต่างกันมาก

ในระยะต่อมาถึงในช่วงทศวรรษที่ 1980
ก็ความสนใจเกี่ยวกับคอมพิวเตอร์ก็มุ่งไปสู่ในเรื่องยูสเซอร์อินเตอร์เฟส ผู้นำทางการค้าที่เป็นที่รู้จักมากที่สุด ได้แก่ ซีรอกส์ และ แอปเปิ้ล (Apple) ก็ได้นำระบบยูสเซอร์อินเตอร์เฟสแบบใหม่มาใช้ ยูสเซอร์อินเตอร์เฟสแบบใหม่นี้เรียกกันว่า WIMP และแนวความคิดใน Smalltalk ก็ถูกในมาใช้ในการพัฒนาระบบดังกล่าว (WIMP ย่อมาจาก Windows, Icons, Mice และ Pointer ซึ่งอ้างอิงถึงรูปแบบการติดต่อกับผู้ใช้โดยการใช้ภาพกราฟฟิก (Graphical User Interface, GUI))
ในอีกนัยหนึ่งก็คือว่าการโปรแกรมมิ่งแบบออพเจ็กต์โอเรียนเต็ดนี้สนับสนุนการพัฒนาระบบยูสเซอร์อินเตอร์เฟสในลักษณะนี้ ในทางกลับกันรูปแบบของภาษาแบบออพเจ็กต์โอเรียนเต็ดก็ได้รับอิทธิพลมาจาก WIMP ค้วยเช่นกัน สิ่งที่สะท้อนให้เห็นในเรื่องนี้ก็ได้แก่การที่มีออพเจ็กต์ไลบรารี่ (library) สำหรับการพัฒนาอินเตอร์เฟสเป็นจำนวนมาก
สิ่งที่เป็นสาเหตุที่ทำให้ออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่งประสบความสำเร็จเป็นอย่างมากก็คือ ความซับซ้อนของกราฟฟิกยูสเซอร์อินเตอร์เฟส และ ค่าใช้จ่ายที่มากในการสร้างระบบการติดต่อดังกล่าว มีการออกความเห็นกันว่าถ้าปราศจากความสามารถที่จะนำกลับมาใช้ได้อีก (reuseablity) ของโปรแกรมแบบออพเจ็กโอเรียนเต็ดแล้ว ระบบการติดต่อแบบนี้ก็ไม่สามารถสร้างขึ้นมาใช้งานในระดับกว้างขวางได้

เมื่อออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่งเริ่มต้นที่จะเติบโตเต็มที่ ความสนใจก็ได้เลื่อนไปอยู่ที่วิธีในการออกแบบโดยใช้แนวคิดของออพเจ็กต์โอเรียนเต็ด และตลอดไปจนถึงการวิเคราะห์แบบออพเจ็กต์โอเรียนเต็ด และการกำหนดรายละเอียด (specification) ข้อดีของการนำกลับมาใช้ได้อีกและการที่สามารถขยายได้ (extensibility) ก็ได้ถูกนำมาประยุกต์ใช้กับการออกแบบและการกำหนดรายละเอียดเช่นเดียวกับการโปรแกรม แต่ยังมีการถกเถียงกันอยู่ในสาขานี้ด้วย เป็นต้นว่า การออกแบบแบบออพเจ็กต์โอเรียนเต็ดนี้จะต้องนำมาสร้างหรือ อิมพลีเมนต์ (implement) ด้วยภาษาแบบออพเจ็กต์โอเรียนเต็ดด้วยหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับในด้านฐานข้อมูล ภายหลังจากที่รีเลชั่นนอลดาต้าเบสได้รับการยอมรับแล้ว ผู้ผลิตรายใหญ่ๆต่างได้นำส่วนเพิ่มเติมรีเลชั่นนอลดาต้าเบสที่เรียกว่า โพสรีเลชั่นนอล (post-relational) ออกมาอีกหลายแบบ โดยแต่ละแบบต่างก็มีฟิลด์ (field) ที่เป็นต้นกำเนิดแตกต่างกันออกไปเช่น ระบบผู้เชี่ยวชาญ, ฟังก์ชั่นนอลโปรแกรมมิ่ง (functional programming) และปัจจุบันออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่ง สำหรับออพเจ็กต์โอเรียนเต็ด มันเพิ่งเริ่มที่จะมีการสร้างออกมาใช้ในทางธุรกิจเมื่อไม่นานนี้

2.3.2 อะไรคือระเบียบวิธีทางออพเจ็กต์โอเรียนเต็ด

ดังที่เราได้เน้นมาแล้วถึงคำว่า ระเบียบวิธีทางออพเจ็กต์โอเรียนเต็ด นี้หมายถึงหลายๆสิ่ง เช่นเดียวกับคำว่า ออพเจ็กต์โอเรียนเต็ด แต่ความหมายที่เฉพาะที่เรากำลังสนใจอยู่ก็จะได้แก่ในเรื่องของ ออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่ง, การออกแบบแบบออพเจ็กต์โอเรียนเต็ด, การวิเคราะห์แบบออพเจ็กต์โอเรียนเต็ด, และฐานข้อมูลแบบออพเจ็กต์โอเรียนเต็ด หรือกล่าวอีกนัยหนึ่งก็คือ ทุกๆเรื่องที่เกี่ยวข้องกับการพัฒนาระบบที่ใช้หลักการของออพเจ็กต์โอเรียนเต็ดนั่นเอง

ในหัวข้อต่อๆไปเราจะแนะนำให้ทราบถึงคำเฉพาะที่ใช้อยู่ในระเบียบวิธีทางออพเจ็กต์โอเรียนเต็ด แต่ผู้อ่านควรตระหนักถึงว่าคำเฉพาะเหล่านี้มีการนำมาใช้ในความหมายที่แตกต่างกันสำหรับผู้เขียนแต่ละคน เหตุที่เป็นเช่นนี้ก็เนื่องจากว่าเรื่องของออพเจ็กต์โอเรียนเต็ดนี้เป็นเรื่องที่เกิดขึ้นมาใหม่นั่นเอง

2.3.3 คำศัพท์เฉพาะและแนวความคิดพื้นฐานเบื้องต้น

แนวความคิดแบบออพเจ็กต์โอเรียนเต็ดไม่ว่าจะเป็นในเรื่องของการออกแบบ การโปรแกรม หรือ อื่นๆ จะถูกกำหนดอยู่ภายใต้คุณสมบัติสำคัญอยู่ 2 คุณสมบัติ ซึ่งคุณสมบัติทั้งสองนั้นก็คือ

- แอปสแตรกชั่น (abstraction)
- อินเฮียริแทนส์ (inheritance)

ทั้งคำว่าแอปสแตรกชั่นและอินเฮียริแทนส์ต่างก็ร่วมกันปกปิดแนวความคิดที่สำคัญอื่นๆไว้ภายใต้มัน ในหัวข้อต่อๆไปเราจะได้เปิดเข้าไปถึงแนวความคิดที่ถูกปิดบังไว้เหล่านี้ เพื่อเป็นการสร้างความเข้าใจถึงคำเฉพาะต่างๆที่จะต้องใช้กันในภายหลัง

2.3.3.1 แอปสแตรกชั่น

แอปแสตรกชั่นมีความหมายถึง การแทนคุณสมบัติที่สำคัญๆของบางสิ่งบางอย่างโดยไม่นำรวมรายละเอียดที่ไม่จำเป็น ในทางโปรแกรมมิ่งคำนี้จึงหมายถึงว่า ออพเจ็กต์ควรจะถูกแทนด้วยการรวมข้อมูลและโพรเซสที่เกี่ยวข้องกับมันเข้าไว้ด้วยกันเพื่อเป็นตัวแทนของออพเจ็กต์นั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แนวความคิดพื้นฐานก็คือว่าออพเจ็กต์จะถูกกำหนดโดยกลุ่มของแอททริบิวส์ (attributes) ที่ใช้รวมกันใช้แทนออพเจ็กต์นั้น แอททริบิวส์ดังกล่าวนี้อาจมีชื่อเรียกว่าเป็น **อินสแตนส์วาไรเอเบิล** (instance variable) หรือ **คลาสส์วาไรเอเบิล** (class variable)
นอกจากกลุ่มของแอททริบิวส์แล้วออพเจ็กต์ก็จะถูกกำหนดไปพร้อมๆกันด้วยกลุ่มของโพรซีเดอร์ (procedures) ที่ได้รับอนุญาติที่สามารถปฏิบัติการกับแอททริบิวส์ของออพเจ็กต์ได้
โดยปกติแล้วแอททริบิวส์ของออพเจ็กต์จะไม่สามารถเข้าถึงได้จากภายนอก
ต้องเข้าถึงแอททริบิวส์เหล่านั้นโดยผ่านทางโพรซีเดอร์ของออพเจ็กต์นั้นเพียงอย่างเดียว

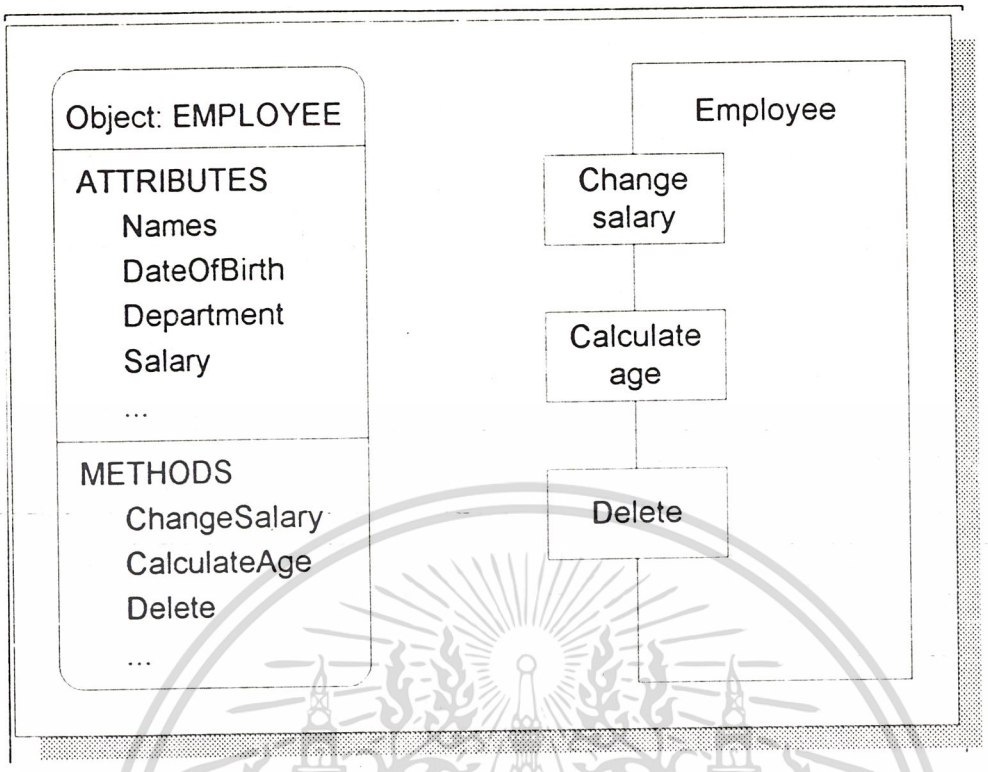
แนวความคิดที่ทำการผูกฟังก์ชันและข้อมูลเข้าด้วยกันนี้มีความสัมพันธ์อย่างใกล้ชิดกับแนวความคิดเรื่องชนิดของข้อมูลในภาษาโปรแกรมมิ่งทั่วไป ในขณะที่ 3 เป็นอินสแตนส์ของข้อมูลชนิดจำนวนเต็ม และจำนวนเต็มถูกกำหนดให้มีแอททริบิวต์หนึ่งตัวคือค่าของมันเอง แต่อนุญาติให้มีโอเปอเรชั่น (operation) ทางคณิตศาสตร์ได้หลายตัว ตัวเลขจำนวนจริงถูกกำหนดให้มีสิ่งต่างๆในลักษณะเดียวกัน แต่การอิมพลีเมนต์ (implementation) ของการคูณสำหรับตัวเลขแบบโฟรตติ้งพอยต์ (floating point) จะแตกต่างกับการคูณตัวเลขจำนวนเต็มธรรมดา ในลักษณะนี้เราจึงสามารถจัดการกับออพเจ็กต์ที่ซับซ้อน เป็นต้นว่า ลูกช้าง ได้โดยกำหนดแอททริบิวส์ให้มันและกำหนด**เมทตอดส์** (methods), **โอเปอเรชั่น**, หรือ **เซอร์วิสส์** (services) ให้กับมัน ทั้งคำว่า เมทตอดส์, โอเปอเรชั่น และเซอร์วิสส์ นั้นเป็นคำที่ใช้เรียกโพรซีเดอร์ที่มีการปกปิดถึงการอิมพลีเมนต์จริงๆไว้

คำว่า **คลาสส์** (class) ในความหมายทางออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่งจะหมายถึง กลุ่มของออพเจ็กต์ที่ใช้แอททริบิวส์และเมทตอดส์พื้นฐานร่วมกัน

ออพเจ็กต์มีมุมมองจากทั้งภายในและภายนอก มุมมองภายในจะบอกถึงสถานะ, การอิมพลีเมนต์ของออพเจ็กต์ ส่วนมุมมองภายนอกจะแสดงชื่อของเมทตอดส์และชนิดของพารามิเตอร์ (parameter) รูปต่อไปนี้เป็นสัญลักษณ์ที่ใช้ในการเขียนอธิบายออพเจ็กต์หนึ่งๆ

จากแนวความคิดในการปกปิดข้อมูลไม่ให้สามารถติดต่อได้โดยตรงจากภายนอกของออพเจ็กต์ การติดต่อระหว่างออพเจ็กต์จึงเป็นการใช้ **แมสเสจ** ติดต่อกันระหว่างออพเจ็กต์
แมสเสจหนึ่งจะประกอบด้วยแอดเดรส (address) ของออพเจ็กต์ที่จะส่งแมสเสจนั้นไปถึง และคำสั่งที่ประกอบด้วยชื่อของแมทตอดส์และตามด้วยพารามิเตอร์ถ้ามี
ถ้าออพเจ็กต์ที่รับแมสเสจนั้นมีแมทตอดส์ที่ต้องทำการตอบข้อมูลบางตัวก็สามารถส่งกลับไปให้ออพเจ็กต์ที่ส่งแมสเสจนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



อีกคำเฉพาะหนึ่งที่เราจะพบก็คือคำว่า **โพลีมอร์ฟิซึม** (Polymorphism) และ **โอเปอเรเตอร์โอเวอร์โหลดดิ้ง** (Operator Overloading)
คำทั้งสองนี้หมายถึงความสามารถที่จะใช้เครื่องหมายสัญลักษณ์เดียวกันสำหรับความหมายที่แตกต่างกันได้ เช่น เมื่อส่งแมสเสจ 'บวก 5' ไปยังตัวเลขจำนวนเต็มและไปยังตัวเลขจำนวนจริง
ออพเจ็กต์ของตัวเลขแต่ละแบบทั้งสองก็จะทำการกระตุ้นโพรซีเดอร์ในการบวกที่แตกต่างสำหรับจำนวนเต็มกับจำนวนจริง แต่มันเป็นการง่ายที่จะใช้เครื่องหมาย +
แทนความหมายทั้งสองและยังทำให้ภาษาง่ายต่อการเรียนรู้และใช้งานอีกด้วย

2.3.3.2 อินเฮียริแทนส์

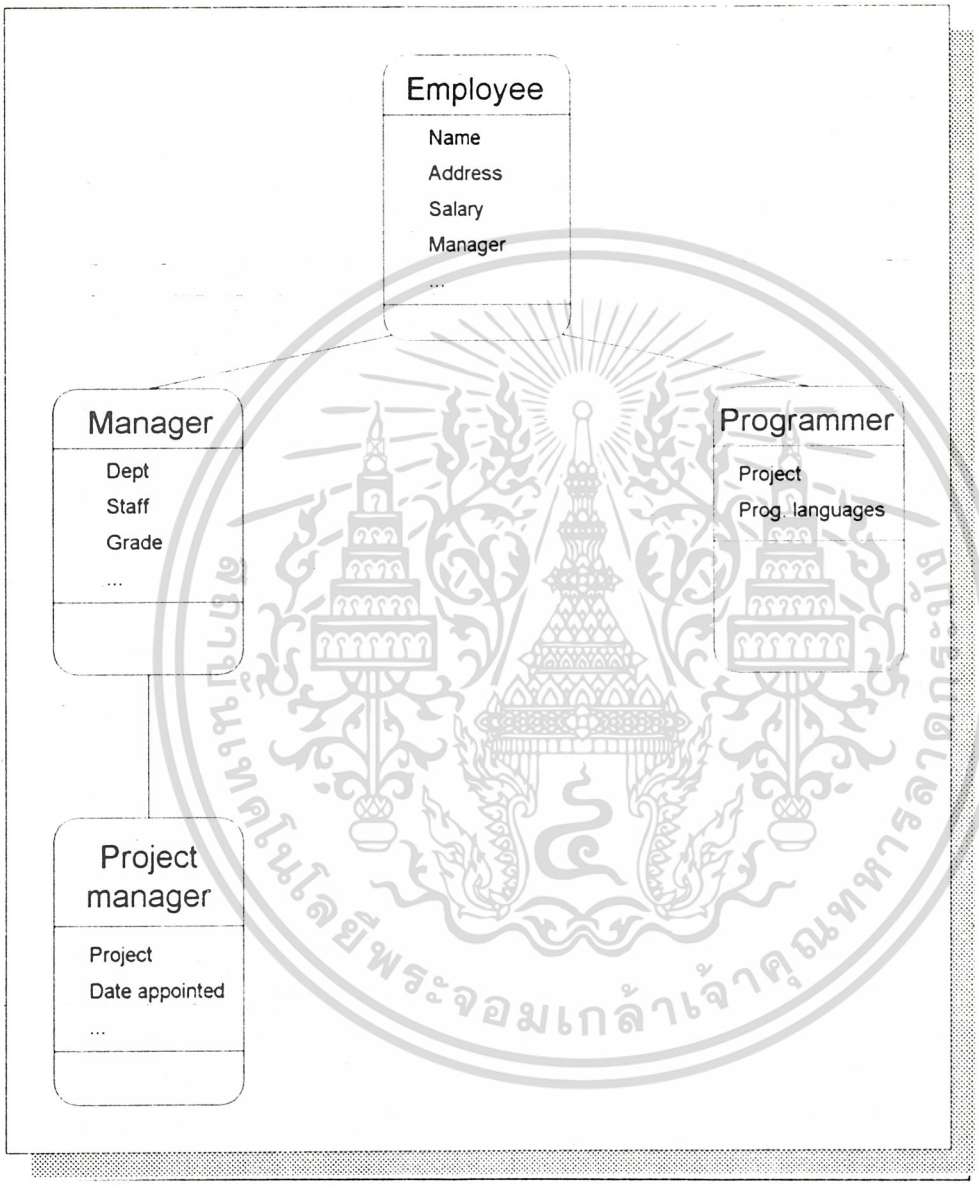
คุณสมบัติของออพเจ็กต์โอเรียนเต็ดอีกอันหนึ่งก็คือ **อินเฮียริแทนส์**
ซึ่งเป็นวิธีการในการจัดการกับความสัมพันธ์ทางโครงสร้างและความหมายระหว่างออพเจ็กต์และคลาสส์
และตัดความซ้ำซ้อนของการที่ต้องเก็บข้อมูลหรือโพรซีเดอร์มากเกินกว่าที่จำเป็น

ออพเจ็กต์หนึ่งๆจะทำการสืบทอดคุณสมบัติหรืออินเฮียริทจากคลาสส์ของมัน
และเนื่องจากคลาสส์ก็เป็นออพเจ็กต์หนึ่งก็สามารถสืบทอดคุณสมบัติจากคลาสส์อื่นได้ด้วยซึ่งคลาสส์นั้นจะเป็นซุปเปอร์คลาสส์ของคลาสส์ที่ทำการอินเฮียริทคุณสมบัติมา
จากการที่มีการสืบทอดคุณสมบัติต่อๆกันนี้จึงทำให้เกิด**อินเฮียริแทนส์เน็ตเวอร์ค** (inheritance network) ขึ้น

ออพเจ็กต์หนึ่งๆสามารถอินเฮียริทมาจากคลาสส์มากกว่าหนึ่งคลาสส์ได้ในเวลาเดียวกัน
ซึ่งเราจะเรียกความสามารถในการอินเฮียริทแบบนี้ว่า **มัลติเพิลอินเฮียริแทนส์** (multiple inheritance)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่มัลติเพิลอินเฮียริแทนส์ก็มีปัญหาที่อาจเกิดขึ้นได้เช่นกัน
ปัญหาของมัลติเพิลอินเฮียริแทนส์จะเกิดขึ้นเมื่อคุณสมบัติที่ออพเจ็กต์ทำการอินเฮียริทมาจากหลายๆคลาสส์นั้นต่างขัดแย้งกัน ในทางปฏิบัติก็สามารถแก้ปัญหาเหล่านี้ได้ เช่น กำหนดดีเฟ้าท์ (default) ของคุณสมบัติเหล่านั้น เป็นต้น รูปต่อไปนี้จะแสดงตัวอย่างของอินเฮียริแทนส์เน็ตเวอร์คของลูกจ้างของบริษัท



2.3.4 ออพเจ็กต์โอเรียนเต็ดแอนนาไลซิส และ ออพเจ็กต์โอเรียนเต็ดดีไซน์ (OOA & OOD)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากคุณประโยชน์ของแนวความคิดแบบออพเจ็กต์โอเรียนเต็ด
ที่ช่วยให้มีการนำกลับมาใช้ใหม่ของส่วนต่างๆ และสามารถแก้ไขเพิ่มเติมได้ง่ายนี้เอง
จึงมีการนำออพเจ็กต์โอเรียนเต็ดมาใช้ในการวิเคราะห์และออกแบบระบบ
โดยการวิเคราะห์และออกแบบระบบโดยวิธีการแบบออพเจ็กต์โอเรียนเต็ดนี้จะแตกต่างจากการวิเคราะห์ออกแบบแบบดั้งเดิม
โดยมีมุมมองระบบเป็นออพเจ็กต์ทำการติดต่อกันมากกว่าที่จะมองเป็นกระบวนการไหลของข้อมูลในการวิเคราะห์และออกแบบแบบดั้งเดิม

เนื่องจากการวิเคราะห์และออกแบบระบบด้วยวิธีการทางออพเจ็กต์โอเรียนเต็ดนั้นมีด้วยกันหลากหลายวิธี ประกอบกับยังไม่มีวิธีการใดที่เป็นที่ยอมรับอย่างแท้จริงเลย
ดังนั้นเราจึงสามารถพบเห็นวิธีการสร้างระบบสารสนเทศ (Information System)
ด้วยวิธีการทางออพเจ็กต์โอเรียนเต็ดได้อย่างหลากหลายวิธีการตามแต่ผู้เชี่ยวชาญต่างๆที่จะออกแบบโมเดลที่คิดว่าเหมาะสมขึ้นมารองรับ, อย่างไรก็ตามกรรมวิธีที่แตกต่างกันเหล่านี้ก็ใช้หลักการเดียวกัน
ที่จะแทนวัตถุต่างๆที่เกี่ยวข้องในระบบงานลงเป็นองค์ประกอบต่างๆบนแผ่นกระดาษและตัวโปรแกรม.

ต่อไปนี้ก็จะขอยกวิธีการแบบออพเจ็กต์โอเรียนเต็ดแบบหนึ่งที่ Mr. James Martin และ Mr. James J. Odell ได้แนะนำไว้ในหนังสือของเขา ดังต่อไปนี้.

โอโอโมเดลส์ (OO Models)

เมื่อเราต้องการวิเคราะห์ระบบ, เราก็จะสร้างโมเดลที่ใช้แทนอาณาเขตของระบบที่เราสนใจนั้นขึ้นมา.
โมเดลดังกล่าว,นอกจากจะต้องแทนความเป็นจริงแล้ว ยังต้องช่วยให้เราเข้าใจระบบนั้นๆได้อีกด้วย.
ด้วยการใช้ ออพเจ็กต์โอเรียนเต็ดแอนนาไลซิส (Object Oriented Analysis),
รูปแบบของโมเดลที่แทนความจริงนั้นจะแตกต่างไปจากการวิเคราะห์แบบดั้งเดิม เราจะแทนระบบในรูปของชนิดของวัตถุ (Object type) และสิ่งที่จะเกิดขึ้นกับมัน. เมื่อนักวิเคราะห์ได้สร้างโมเดลของระบบขึ้นมาแล้ว, โมเดลเหล่านี้ก็จะถูกเปลี่ยนไปอยู่ในรูปของดีไซน์และโปรแกรมต่อๆไป.

ออพเจ็กต์สตรักเจอร์แอนนาไลซิส (Object Structure Analysis)

ในการทำการวิเคราะห์โครงสร้างของออบเจ็กต์นี้, ข้อมูลต่อไปนี้เป็นสิ่งที่เรามุ่งค้นหา:

- **What are the object types and how are they associated?** The identification of objects and their associations are represented by an object schema. This information guides the designer in class and data structure definition.
- **How are the object types organized into supertypes and subtypes?** Generalization hierarchies can be diagrammed and indicate directions of inheritance to the designer.
- **What is the composition of complex objects?** Composed-of hierarchies can be diagrammed. Composition guides the designer in defining mechanisms that properly manage objects-within-objects.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลต่างๆที่ได้จากการทำการวิเคราะห์, จะถูกนำไปสร้างเป็นออพเจ็กต์สคีมม่า (Object Schema) ออพเจ็กต์สคีมม่าจะเป็นแผนภาพแบบหนึ่งที่ใช้แสดง ความสัมพันธ์ของออบเจ็กต์ต่างๆ; ซุปเปอร์ไทปส์ (supertypes) และซับไทปส์ (subtypes) ; คอมโพสด์ออฟสตรักเจอร์ (Composed-of structures) ของออบเจ็กต์.

ออพเจ็กต์บีเฮฟวิเออร์แอนนาไลซิส (Object Behavior Analysis)

ในการทำการวิเคราะห์พฤติกรรมของออบเจ็กต์นี้, ข้อมูลต่อไปนี้เป็นสิ่งที่เราต้องค้นหา:

- **What states can an object be in?** One object can have multiple sets of states.
- **What state transitions occur?** These are drawn on state transition diagrams.
- **What events occur?** (An event is the change in state of an object.)
- **What operations take place?** An event schema is drawn, showning the sequence of operations and events.
- **What interactions occur between objects?** A diagram can be drawn showing messages passing among classes.
- **What trigger rules are used to react to the event?**
- **How are the operations represented in methods?** Diagrams, declarative statements, or other means are used to specify the methods in sufficient detail to generate code.

ข้อมูลต่างๆที่ได้จากการวิเคราะห์เรื่องพฤติกรรมของออบเจ็กต์นี้จะถูกนำไปสร้างเป็นอีเวนท์สคีมมา (Event Schema) ซึ่งจะแสดงลำดับของเหตุการณ์ที่จะกระตุ้นโอเปอร์เรชั่นต่างๆ
ซึ่งอีเวนท์สคีมมานี้จะสัมพันธ์กับออพเจ็กต์สคีมมาของขั้นตอนออพเจ็กต์สตรักเจอร์แอนนาไลซิสด้วย.

ออพเจ็กต์โอเรียนเต็ดดีไซน์ (Object Oriented Design)

เมื่อเราได้ทำการวิเคราะห์โครงสร้างและพฤติกรรมของออบเจ็กต์จนได้โมเดลต่างๆที่ใช้แทนระบบงานออกมาแล้ว
ก็มาถึงขั้นตอนที่นำโมเดลเหล่านี้มาออกแบบระบบโดยจะมีการจำแนกออพเจ็กต์อื่นๆเพิ่มเติมที่จะต้องใช้ในระบบใหม่ แล้วทำการเชื่อมต่อมันเข้ากับออพเจ็กต์ในส่วนเดิมเพื่อให้สามารถทำงานร่วมกันได้

ภายหลังจากที่เราทำการออกแบบระบบใหม่ขึ้นมาได้แล้วเราก็จะนำระบบนั้นมาทำการอิมพลีเมนต์ ซึ่งการอิมพลีเมนต์นี้ไม่จำเป็นต้องใช้ภาษาหรือฐานข้อมูลแบบออพเจ็กต์โอเรียนเต็ดจริงๆก็ได้ เราสามารถนำระบบที่ออกแบบแบบออพเจ็กต์โอเรียนเต็ดนี้มาทำการปรับให้อยู่ในรูปแบบที่สามารถนำไปใช้การอิมพลีเมนต์ที่เราต้องการได้

ต่อไปนี้จะขอแสดงตัวอย่างสัญลักษณ์ต่างๆที่ใช้ในวิธีการวิเคราะห์และออกแบบวิธีนี้ ในหน้าต่อๆไป

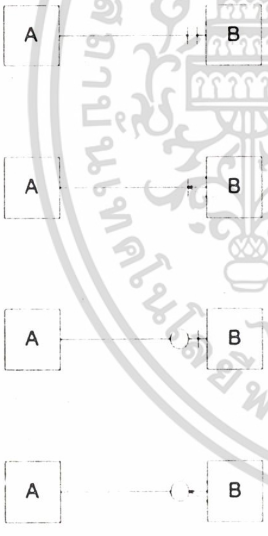
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญลักษณ์ที่ใช้ในการทำ Object Structure Analysis

แทน Object Type

แทน Association

Common notation for Object and association



An A is always associated with one B

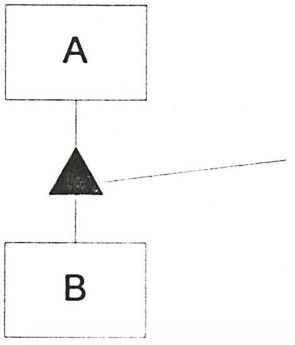
An A is always associated with one or many of B

An A is always associated with zero or one of B

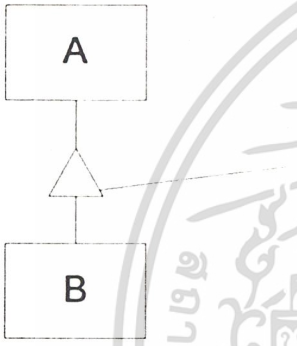
An A is always associated with any number of B

Common notation for cardinality

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

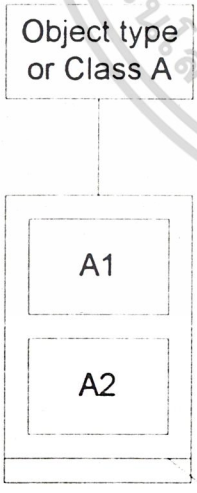


GENERALIZATION (and inheritance) B is a subtype of A



COMPOSITION (a special type of relationship) B is a component of A

SUBTYPE BOXES



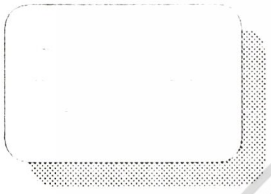
A1 and A2 are disjoint subtypes of type (or class) A

This vertical line indicates that there are subtypes other than those specified in the box.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญลักษณ์ที่ใช้ในการทำ Object Behavior Analysis

ACTIVITIES (Operations, processes, procedures, methods)

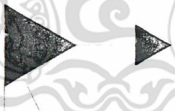


EXTERNAL ACTIVITIES (External to the system being diagrammed)

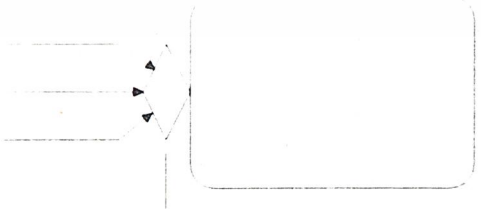
An Event Type

A Clock Event Type

Trigger rules

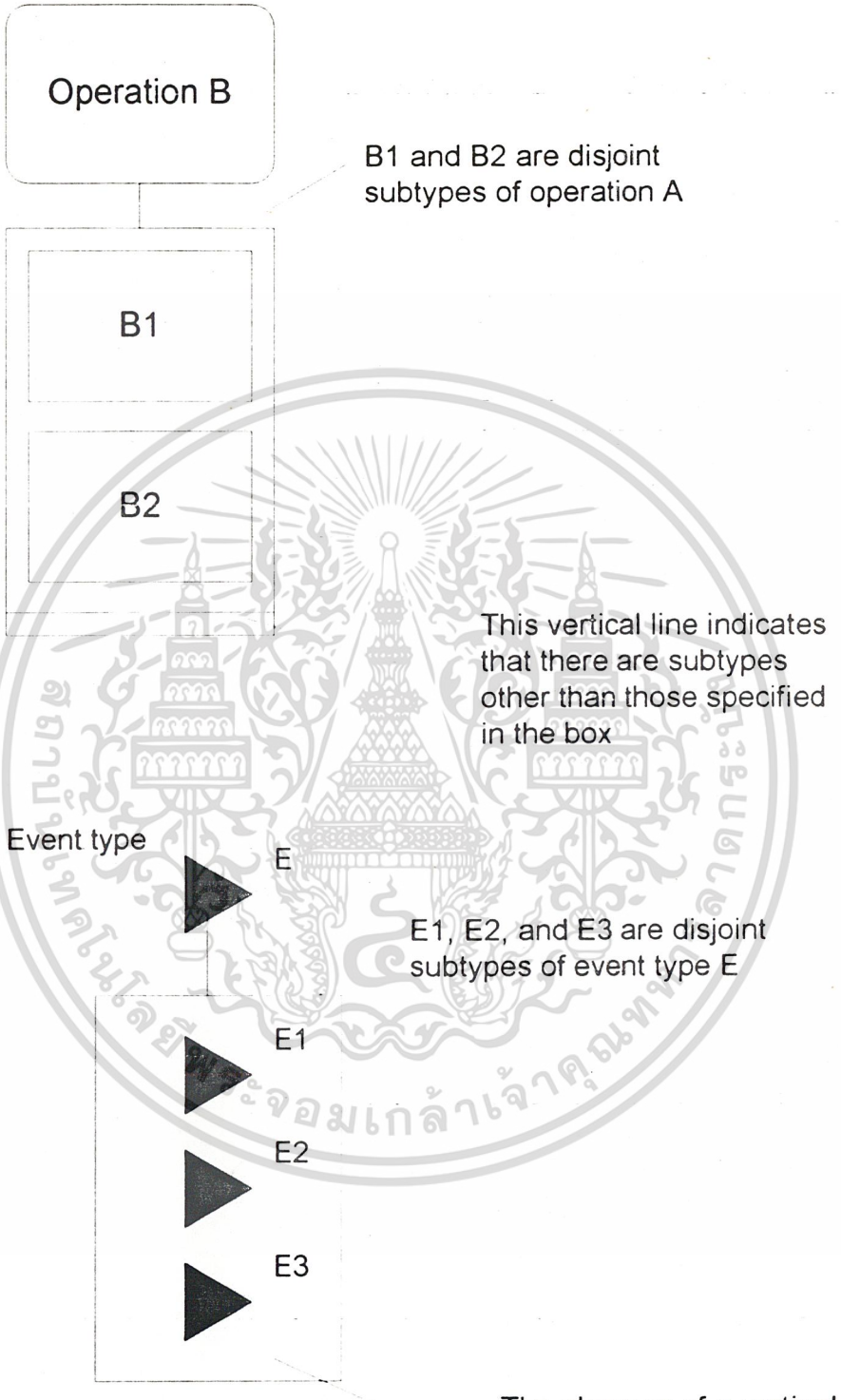


Operation results in this event



This control condition governs whether the operation occurs

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4 สรุป

สถาปัตยกรรมของระบบฐานข้อมูลบนคอมพิวเตอร์ที่ใช้กันอยู่ในปัจจุบันมีอยู่หลายระดับ
ซึ่งแต่ละระดับก็จะมีคุณประโยชน์และการอิมพลีเมนเทชั่นที่แตกต่างกันออกไป
ระบบแบบเซ็ลทรัลไลซ์คเป็นระบบแบบแรกๆที่มีการนำมาใช้งานและก็ยังมีการใช้งานในปัจจุบันในระบบใหญ่
ที่ใช้เครื่องคอมพิวเตอร์เมนเฟรม
ระบบแบบพีซีเป็นระบบฐานข้อมูลที่ถูกประยุกต์ใช้เข้ากับเครื่องคอมพิวเตอร์ส่วนบุคคลที่ราคาไม่แพง
แม้ว่าประสิทธิภาพและความปลอดภัยของข้อมูลหรืออื่นๆจะยังเป็นปัญหาสำหรับระบบแบบพีซีอยู่ก็ตาม
ระบบแบบพีซีก็ยังคงตอบสนองความต้องการของระบบฐานข้อมูลขนาดที่ไม่ใหญ่นักได้ดี
ระบบแบบไคลเอ็นท์เซอร์ฟเวอร์เป็นแนวความคิดที่ต้องการแบ่งการประมวลผลระหว่างเครื่องที่เป็นเซอร์ฟเวอร์
และเครื่องที่เป็นไคลเอ็นท์ให้แยกออกจากกัน
เพื่อประโยชน์ในแง่ของประสิทธิภาพโดยรวมของระบบเพราะช่วยลดแทรฟฟิกของระบบเครือข่าย
ส่วนระบบแบบดิสทริบิวเต็ดเป็นระบบที่ถูกออกแบบสำหรับองค์กรที่มีหน่วยงานกระจายกันอยู่หลายสถานที่ที่ห่
างกัน

เนื่องจากเราได้มุ่งเน้นถึงการออกแบบพัฒนาระบบแบบไคลเอ็นท์เซอร์ฟเวอร์
ดังนั้นเราจำเป็นต้องมาศึกษาว่าเราจะใช้วิธีการใดมาทำการออกแบบระบบประเภทนี้แล้วมีประสิทธิภาพมากที่สุด
สังเกตว่าระบบแบบไคลเอ็นท์เซอร์ฟเวอร์ต้องการแยกโพรเซสที่เป็นยูสเซอร์อินเตอร์เฟสมาใช้ไคลเอ็นท์ทำ
และในปัจจุบันยูสเซอร์อินเตอร์เฟสเหล่านั้นต่างก็เป็นแบบจียูไอ (GUI) เป็นส่วนมาก
เราจึงมุ่งหาวิธีที่ใช้ในการออกแบบสร้างที่สามารถมาประยุกต์ใช้กับระบบแบบนี้นี่เอง
เราพบว่าการใช้แนวความคิดแบบออพเจ็กต์โอเรียนเต็ดจะช่วยทำให้ระบบงานมีความยืดหยุ่นต่อการแก้ไขปัญหาแ
ละยังเหมาะสมกับระบบปัจจุบันที่ยูสเซอร์อินเตอร์เฟสเป็นจียูไอที่เน้นแนวความคิดของออพเจ็กต์ค้าย
ดังนั้นในบทนี้เราจึงศึกษาถึงแนวความคิดและคำเฉพาะที่ใช้ในออพเจ็กต์โอเรียนเต็ดเม็ดตอด

ในตอนท้ายเราได้ยกตัวอย่างวิธีการวิเคราะห์และออกแบบของ Mr. James Martin และ Mr. James J. Odell
แม้ว่าในความเป็นจริงแล้วยังมีวิธีการวิเคราะห์ออกแบบแบบอื่นๆด้วยที่ใช้แนวความคิดแบบออพเจ็กโอเรียนเต็ดเ
ช่นเดียวกัน วิธีการที่แตกต่างกันเหล่านี้ จริงๆแล้วก็แตกต่างกันไม่มาก
และสิ่งที่แตกต่างก็จะเป็นในเรื่องของสัญลักษณ์ คำเฉพาะที่ใช้ในวิธีนั้นๆ
แต่ในส่วนของแนวความคิดแล้วนั้นไม่ได้มีความแตกต่างกันไปมากเลย
และในบทต่อไปเราจะได้นำเสนอวิธีการออกแบบอีกแบบหนึ่งที่มีชื่อว่า OMT
ที่ผู้เขียนใช้ในการออกแบบสร้างโครงงานนี้จริงๆ ซึ่งในหลักการแล้ว
ก็ใช้แนวความคิดเหมือนกับตัวอย่างวิธีการที่ยกให้เห็นภายในบทนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

ระเบียบวิธีการออกแบบ (Methodology)

การออกแบบและการสร้างแบบจำลอง (Object-oriented Modeling and Design)

เป็นแนวคิดใหม่เพื่อการวิเคราะห์ปัญหาโดยการสร้างแบบจำลองอิงกับปัญหาในรูปแบบที่เป็นรูปธรรม วิธีการออกแบบที่นำมาประยุกต์ใช้มีชื่อว่า Object Modeling Technique (OMT) ซึ่งเป็นการแก้ปัญหาโดยเริ่มจากการ วิเคราะห์ (Ananlysis) , ออกแบบ (Design) และ ประยุกต์ (Implementation) ขั้นตอนของ OMT เริ่มจากการออกแบบ แบบจำลองของปัญหา แล้วจึงเพิ่มรายละเอียดเข้าไปในแบบจำลองในขั้นตอนของการวิเคราห์

แบบจำลองของ OMT

3.1การวิเคราะห์

ขั้นตอนการวิเคราะห์เริ่มจากการเก็บรวบรวม ข้อกำหนดและลักษณะของปัญหา จากระบบ และนำมาทำการสร้างแบบจำลองทั้งสามดังนี้

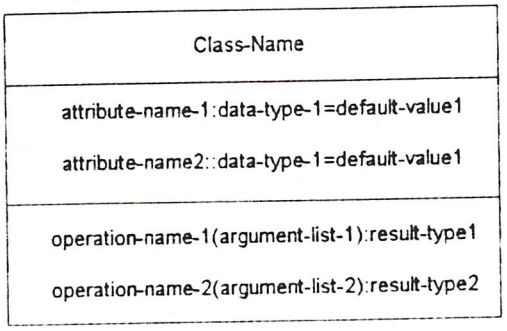
- Object Modeling
- Dynamic Modeling
- Functional Modeling

3.1.1 แบบจำลองออพเจกต์ (Object Modeling)

เป็นแบบจำลองที่แสดงถึงตัว ออพเจกต์ และ ความสัมพันธระหว่าง ออพเจกต์ รวมทั้ง แอททริบิวท์(Attribute) และ โอเปอร์เรชั่น ของ ออพเจกต์ คลาสส์ ด้วย แบบจำลองออพเจกต์ ถือว่าเป็นแบบจำลองที่มีความสำคัญมากที่สุดในแบบจำลองทั้งสามแบบที่จะกล่าวต่อไป เพราะว่า เป็นแบบจำลองที่แสดงถึงโครงสร้างของระบบโดยใช้รูปภาพและสัญลักษณ์ต่างสื่อความหมายทำให้ทั้งผู้ออกแบบระบบและผู้ใช้ระบบสามารถทำความเข้าใจกับตัวระบบได้อย่างเป็นรูปธรรม

สัญลักษณ์ต่างๆที่ใช้ใน แบบจำลองออพเจกต์ มีรูปแบบตามรูปต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดงสัญลักษณ์ที่ใช้แทน ออพเจกต์ คลาสส์

สัญลักษณ์ที่ใช้แทน ออพเจกต์ คลาสส์ จะประกอบด้วยส่วนหลักๆ สามส่วนด้วยกัน ส่วนแรกใช้แสดงชื่อคลาสส์ ส่วนที่สองแสดงรายการของ แอททริบิวท์s ของ ออพเจกต์ คลาสส์ และส่วนสุดท้ายใช้แสดงรายการชื่อของ โอเปอร์เรชั่นs ของ ออพเจกต์ คลาสส์ ในแต่ละ แอททริบิวท์ ที่แสดงในส่วนที่สองจะมีรายละเอียดเพิ่มเติมอีกเล็กน้อยเช่นชนิดของข้อมูลและค่าเริ่มต้น เช่นเดียวกันในส่วนที่สามหรือส่วนของโอเปอร์เรชั่น (Operation) ก็จะประกอบด้วยรายละเอียดปลีกย่อยเช่น รายการของ Arguments และชนิดของผลลัพธ์ อย่างไรก็ตามเมื่อถึงเวลาการเขียน แบบจำลอง แล้วชื่อของ แอททริบิวท์ และ โอเปอร์เรชั่น บางตัวอาจละไว้ได้ตามสมควร

ความสัมพันธ์ระหว่าง ออพเจกต์ สามารถได้ด้วยสัญลักษณ์ด้านล่างต่อไปนี้ ซึ่งมีทั้งแบบ หนึ่งต่อหนึ่ง และแบบ Manty-to-Many โดยทั่วไปแล้วความสัมพันธ์ที่ปรากฏใน Diagram จะได้มาจากคำกริยาของข้อกำหนดของปัญหา (Problem Statement) ที่ได้ตระเตรียมไว้

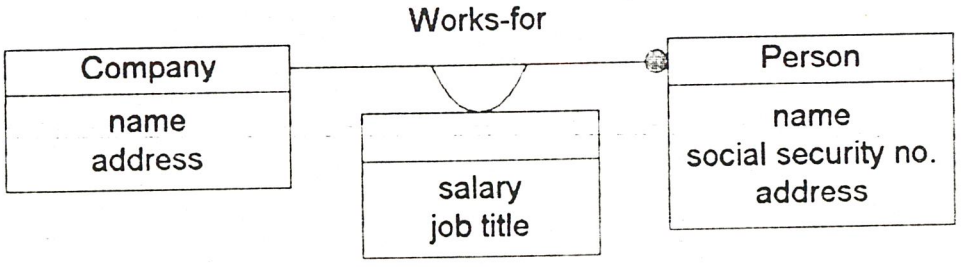
ความสัมพันธ์แบบ 1 1

ความสัมพันธ์แบบ 1 M

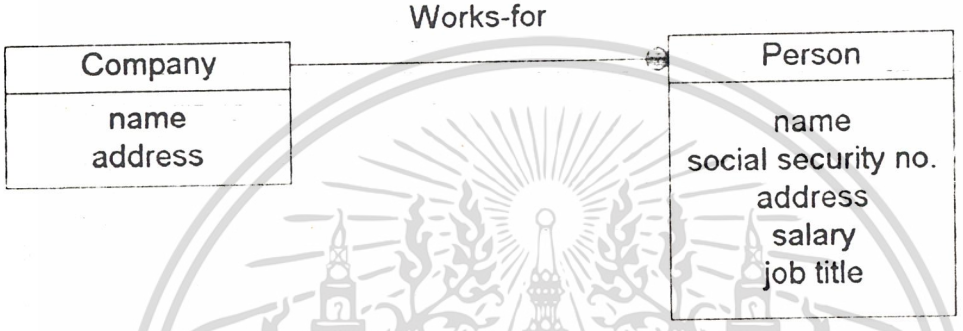
ความสัมพันธ์แบบ M M

แสดงความสัมพันธ์ (Association) ระหว่าง คลาสส์ ในรูปแบบต่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแบบที่เหมาะสม

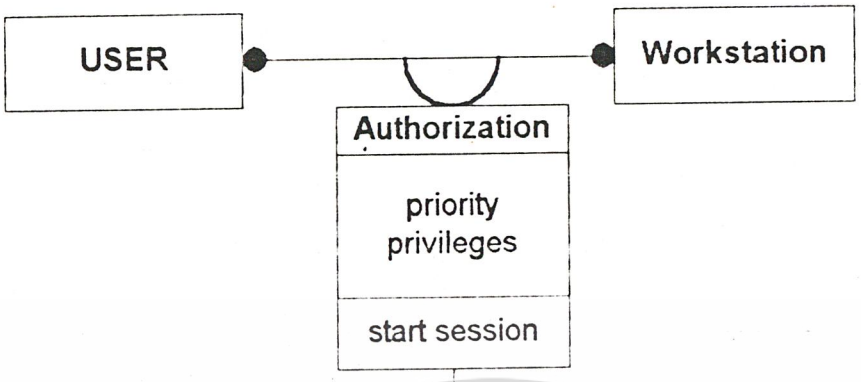


รูปแบบที่ไม่เหมาะสม

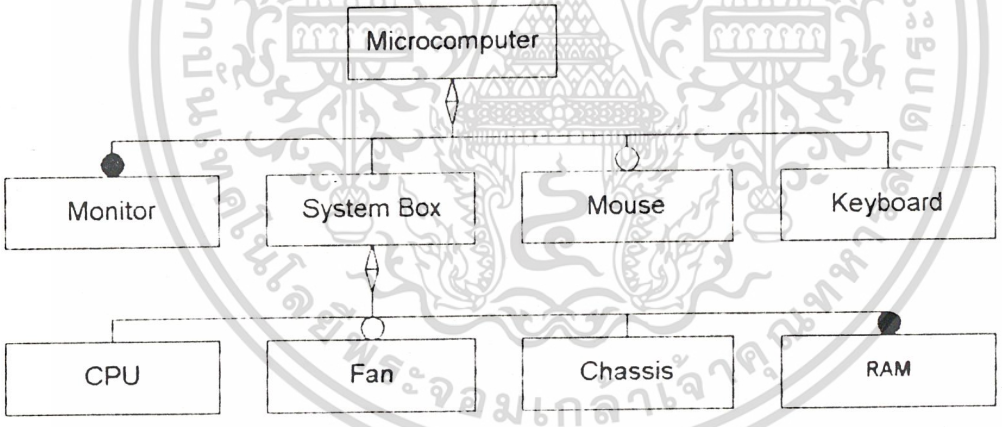
แสดง Link หรือ Association ที่มี แอททริบิวท์

การใส่ แอททริบิวท์ ให้กับ Link หรือ Association ทำให้เราสามารถเปลี่ยนความสัมพันธ์ระหว่าง ออพเจกท์ คลาสส์ ได้โดยไม่กระทบกับ ออพเจกท์ คลาสส์ ของเดิม และนอกจากนี้แล้วยังทำให้การจัดเก็บข้อมูลในรูปของฐานข้อมูลลดความซ้ำซ้อนลงอีกด้วย ในบางครั้งเราสามารถที่จะกำหนดความสัมพันธ์ให้เป็น คลาสส์ ก็ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

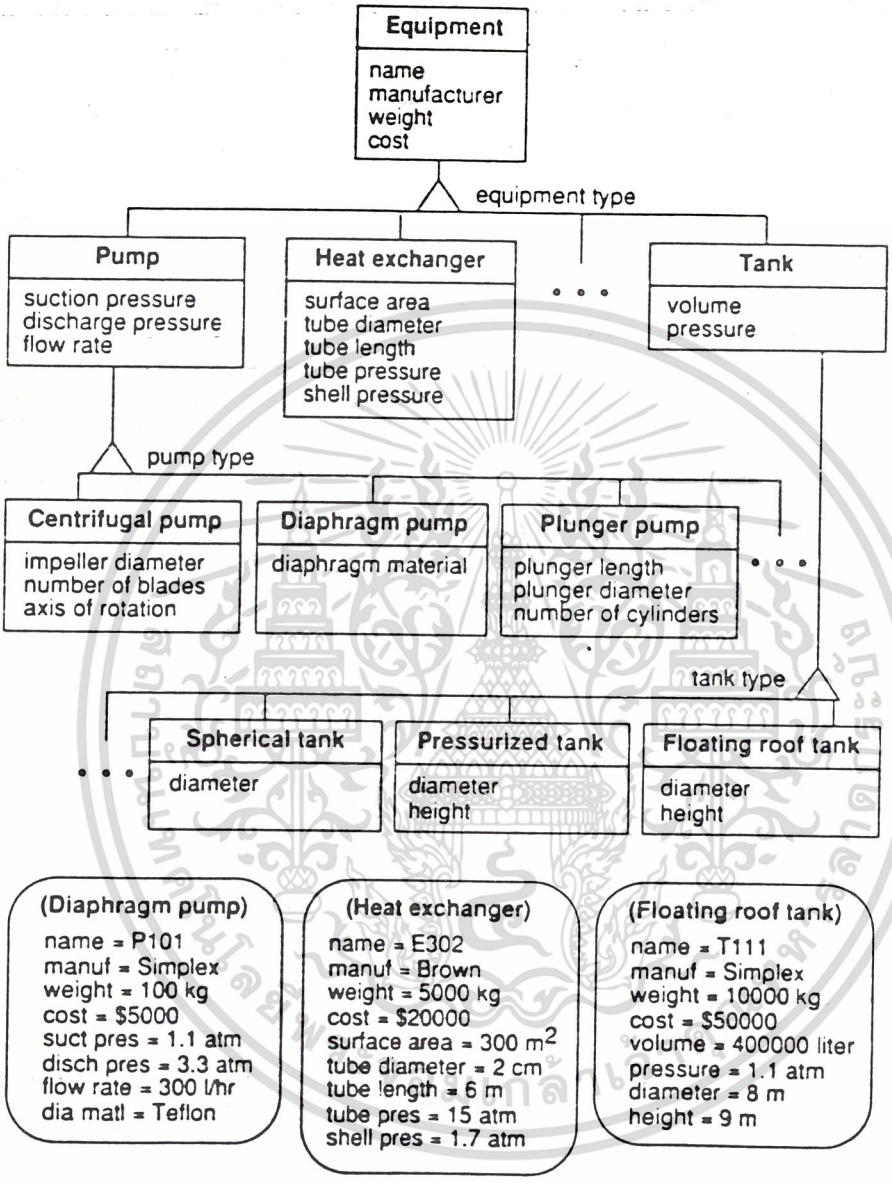


แสดง Link ที่เป็น คลาสส์



แสดง ความสัมพันธ์แบบ Aggregation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



[Fig 3.23]

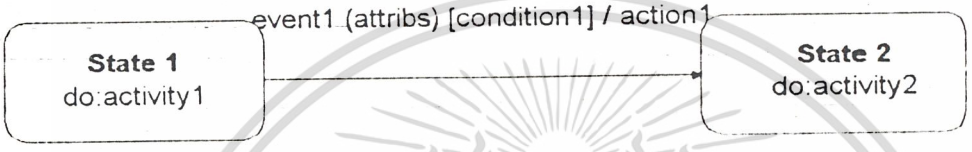
แสดง ความสัมพันธ์ในรูป Inheritance

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.2 แบบจำลองไดนามิค (Dynamic Modeling)

ลำพังแต่ Object Model เราไม่สามารถที่จะอธิบายความหมายของระบบทั้งหมดได้ เพราะ Object Model อธิบายแต่ความสัมพันธ์ และ โครงสร้างของ ออพเจกค์ เท่านั้น แต่ แบบจำลองไดนามิค ทำหน้าที่อธิบายถึงลำดับการเปลี่ยนแปลงของ โอเปอร์เรชั่น ของ ออพเจกค์ ที่ถูกกระตุ้นโดย Events โดยไม่สนใจว่าโอเปอร์เรชั่น นั้นจะทำอะไรหรือทำขึ้นมาอย่างไร

หลักการสำคัญของ แบบจำลองไดนามิคคือ States (แสดงค่าของ ออพเจกค์ ณ จุดหนึ่ง) และ Events (ตัวกระตุ้นให้เกิดการเปลี่ยน State) รวมกันเป็น State Diagram สัญลักษณ์ที่ใช้แสดงใน State Diagram มีรูปแบบดังต่อไปนี้



แสดงส่วนประกอบของ State Diagram

ส่วนประกอบของ State Diagram ประกอบด้วยตัว State ซึ่งชื่อของ State จะถูกเขียนด้วยตัวพิมพ์หนา ส่วนชื่อของเหตุการณ์ (Events) จะอยู่บนเส้นตรงระหว่าง States และอาจตามด้วย แอททริบิวท์ , เงื่อนไข และ Actions ในส่วนของตัว States จะมี Activity ที่ต้องทำอยู่ภายใน

อย่างที่ได้ทราบกันแล้วว่า แบบจำลองไดนามิคทำหน้าที่แสดงจังหาะและเวลาการเกิดขึ้นของเหตุการณ์ต่างๆในระบบ โดยเฉพาะกับระบบที่มีการโต้ตอบกับผู้ใช้เป็นหลักนั้น แบบจำลองไดนามิคจะมีความสำคัญมากเพราะจะใช้เป็นตัวแสดงขั้นตอนการโต้ตอบระหว่างระบบกับผู้ใช้

ขั้นตอนในการสร้าง แบบจำลองไดนามิคเริ่มจากการเตรียมโครงร่างของเหตุการณ์ (Scenario) ขึ้นมาก่อน โครงร่างที่จัดเตรียมขึ้นมาไม่จะเป็นต้องครอบคลุมเหตุการณ์ทั้งหมดของระบบ แต่อย่างน้อยจะต้องครอบคลุมถึงเหตุการณ์การโต้ตอบของระบบในขั้นพื้นฐาน เมื่อได้จัดเตรียมโครงร่างของระบบเสร็จเรียบร้อยแล้ว หน้าที่ที่จะต้องทำต่อไปก็คือ ทำการแยกแยะและกำหนดเหตุการณ์ (Events) เสียก่อนจากนั้นก็กำหนดเหตุการณ์เหล่านั้นให้กับ ออพเจกค์ ที่เหมาะสม แล้วจึงมาจัดเรียงลำดับเหตุการณ์ให้เป็น State Diagram ส่วนในขั้นสุดท้ายจะทำการเปรียบเทียบ State Diagram กับ ออพเจกค์ อื่นๆ เพื่อตรวจสอบการส่ง เหตุการณ์ ระหว่าง ออพเจกค์ นั้นมีความถูกต้อง

1.การเตรียมโครงร่างเหตุการณ์ (Preparing Scenario)

เขียนขั้นตอนการโต้ตอบระหว่างระบบและผู้ใช้เพื่อสังเกตุรูปแบบและแนวทางของระบบ โครงร่างที่เขียนขึ้นจะแสดงถึงการโต้ตอบหลักๆของระบบ , รูปแบบของหน้าจอ และ การแลกเปลี่ยนข้อมูลระหว่างระบบกับผู้ใช้ เป็นต้น ข้อที่ควรคำนึงถึงของการเขียนโครงร่างเหตุการณ์ก็คือต้องเขียนขั้นตอนหลักๆลงไปในโครงร่างนั้นๆ เหตุการณ์ในระบบจะเกิดขึ้นได้ก็ต่อเมื่อมีการแลกเปลี่ยนข้อมูลระหว่าง ออพเจกค์ ภายในหรือนอกระบบ ค่าของข้อมูลที่แลกเปลี่ยนกับนั้นเราเรียกว่า Patameter ของเหตุการณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเขียนโครงร่างของเหตุการณ์ที่ดีนั้นจะต้องมีทั้งในกรณีที่เหตุการณ์เกิดขึ้นอย่างปกติ และโครงร่างของเหตุการณ์ที่เกิดขึ้นในกรณีพิเศษ เช่นการเกิดข้อผิดพลาดของระบบ

2.รูปแบบการติดต่อ (Interface Format)

โดยส่วนใหญ่แล้วการติดต่อจะแบ่งออกเป็นสองส่วน ก) Application Logic ข) User Interface การเขียน แบบจำลองไดนามิคจะทำให้เราสามารถกำหนดตรรก การควบคุมการติดต่อ (Control Logic) ของระบบได้

3.การเลือกกำหนดเหตุการณ์ (Identifying Events)

เหตุการณ์ที่กล่างถึงนี้หมายถึง สัญญาณ (Signals) , Input , Decision , Interupt , Transition และการกระทำโดย หรือ จาก ผู้ใช้ระบบหรือจากสิ่งอื่นภายนอกระบบ การคำนวณใดก็ตามที่เกิดขึ้นในระบบเราไม่ถือว่าเป็นเหตุการณ์เว้นแต่ว่าจะเป็นจุดที่ทำการตัดสินใจเกี่ยวกับสิ่งอื่นภายนอกระบบ การกระทำใดก็ตามที่เกิดจาก ออพเจกต์ ที่ทำให้เกิดการเคลื่อนย้ายหรือส่งต่อข้อมูลเราจะถือว่าสิ่งนั้นเป็นเหตุการณ์

เมื่อสามารถกำหนดเหตุการณ์ของระบบออกมาได้แล้ว เราก็จะมาทำการกำหนดเหตุการณ์เหล่านี้ให้กับ ออพเจกต์ คลาสส์ เหตุการณ์หนึ่งๆอาจเป็นได้ทั้ง Input และ Output ให้กับ ออพเจกต์ คลาสส์ ใดๆ และในบางครั้ง ออพเจกต์ คลาสส์ นั้นก็สามารถที่จะส่งเหตุการณ์ย้อนกลับไปให้ตัวมันเองก็ได้

โครงร่างของเหตุการณ์ที่เกิดขึ้นจะถูกนำมาแสดงในรูปของ Event Trace ซึ่งเป็นการเขียนลำดับของเหตุการณ์ที่เกิดขึ้นระหว่าง ออพเจกต์ ที่แตกต่างกัน โดย ออพเจกต์ แต่ละตัวจะถูกเขียนแยกไว้เป็นคอลัมน์ๆ จากการเขียน Event Trace จะทำให้เห็นลักษณะและผลกระทบของเหตุการณ์ที่เกิดขึ้นกับ ออพเจกต์ หนึ่ง

แผนผังที่จะขาดไม่ได้เลยของการทำ Dynamic Modeling ก็คือ Event Flow Diagram มีหน้าที่แสดงเหตุการณ์ที่เกิดขึ้นระหว่าง ออพเจกต์ คลาสส์ โดยไม่กล่าวถึงลำดับการเกิดขึ้นของเหตุการณ์นั้นๆ เส้นทาง (Paths) ต่างๆที่เกิดขึ้นใน Event Flow Diagram จะเป็นตัวแสดง Control Flow ของระบบที่เป็นไปได้

4.การสร้าง State Diagram

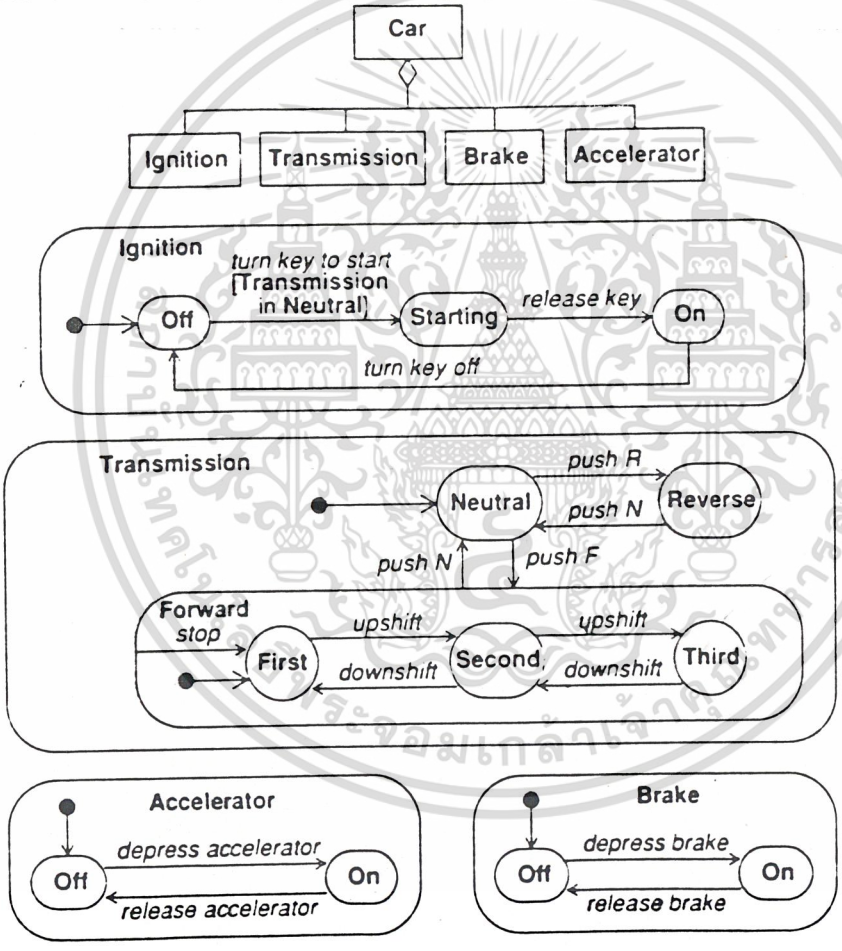
จาก Event Trace Diagram เลือก Diagram สำหรับ ออพเจกต์ คลาสส์ ที่ต้องการเขียน State Diagram ขึ้นมาแล้วนำเหตุการณ์จาก Diagram นั้นในคอลัมน์เดียวกันมาเขียนลงใน State Diagram โดยช่วงเวลาระหว่างเหตุการณ์สองเหตุการณ์เราจะกำหนดให้เป็น State และถ้า State นั้นสามารถกำหนดชื่อได้เราก็จะกำหนดชื่อให้ แต่ถ้าไม่สามารถหาชื่อที่มีความหมายให้ได้ก็ไม่จำเป็นต้องกำหนดก็ได้

หลังจากการเขียน State และ Event เรียบร้อยแล้วก็ให้ทำการหา Loop ที่เกิดขึ้นใน Diagram คือให้ State แรกและ State สุดท้ายเป็น State เดียวกัน State Diagram ที่ดีควรจะต้องแสดงรายละเอียดของการเกิดข้อผิดพลาดของระบบด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.ตรวจสอบความถูกต้อง

ตรวจสอบความถูกต้องและความสมบูรณ์ของ State Diagram ของ ออพเจกต์ คลาสส์ แต่ละตัว โดยที่ทุกๆเหตุการณ์จะต้องมีทั้งผู้ส่งและผู้รับ ใน State ใดๆถ้าไม่มี State นำหน้าหรือตามหลังให้ทำการตรวจสอบ State นั้นให้รอบคอบว่าเป็น State เริ่มต้นหรือ สิ้นสุดเท่านั้น นอกจากนี้ยังต้องทดลองไล่ตาม State และ Event ในแผนผังเพื่อเปรียบเทียบว่าสอดคล้องกับโครงร่างของเหตุการณ์ที่ได้เขียนไว้ก่อนแล้ว



แสดง ความสัมพันธ์ระหว่าง คลาสส์ และ State Diagram

วิธีการสร้าง State Diagram เราจะกำหนดให้ คลาสส์ หนึ่ง คลาสส์ จะมี State Diagram ประจำตัวหนึ่งอัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1.3 แบบจำลองฟังก์ชั่น (Functional Modeling)

เป็นแบบจำลองที่ใช้แสดงวิธีการคำนวณค่าต่างๆในระบบว่ามีวิธีการคำนวณอย่างไร โดยไม่คำนึงถึงลำดับขั้นตอน , การตัดสินใจ หรือ โครงสร้างของ ออพเจกต์ แบบจำลองที่เขียนขึ้นนี้ก็คือ Data Flow Diagram ซึ่งเรานำมาอธิบายความสัมพันธ์ของข้อมูลและฟังก์ชั่น การประมวลผลต่างๆ(Process) ที่ปรากฏใน Diagram จะสอดคล้องกับ Activities และ Actions ใน State Diagram ที่เขียนขึ้น ส่วนข้อมูลที่ไหลผ่านใน Diagram ก็จะสอดคล้องกับ ออพเจกต์s หรือ แอททริบิวท์s ใน ออพเจกต์ Diagram การเขียน Functional Model ที่เหมาะสมที่สุดนั้นควรจะเขียนหลังจากการเขียน แบบจำลองออพเจกต์ และ แบบจำลองไดนามิคแล้ว

1.การกำหนดค่า Input และ Output

เริ่มต้นโดยการเขียนรายการ Input และ Output ที่จะเกิดขึ้นโดยนำมาจากข้อมูลระหว่างระบบและสิ่งต่างๆภายนอกระบบ ค่าข้อมูลเข้าและออกเหล่านี้จะถูกใช้เป็น Parameters ของเหตุการณ์ที่เกิดขึ้น ค่าของ Input ใดๆ ถ้ามีผลกระทบต่อการไหลของข้อมูลเพียงอย่างเดียวเราจะไม่นำมาคิดเป็น Inputs

2.การสร้าง Data Flow Diagram

โดยปกติแล้ว Data Flow Diagram จะถูกสร้างขึ้นเป็นชั้นๆ ในชั้นแรกอาจประกอบไปด้วย Process เพียง Process เดียว ในแต่ละระดับของ Data Flow Diagram เราจะทำการออกแบบย้อนกลับโดยเริ่มจากการดู Output ก่อนแล้วค่อยมากำหนดฟังก์ชั่นที่ทำหน้าที่ให้ Output นั้นออกมา และจากโปรเซสใหญ่ๆเราก็จะแตกออกเป็นโปรเซสย่อยลงไปจนกระทั่งสามารถทำการประยุกต์ (Implement) ได้จากโปรเซสย่อยๆนั้น

3.อธิบายฟังก์ชั่น

Data Flow Diagram ที่ผ่านการตรวจสอบจนสมบูร์แล้วให้เขียนคำอธิบายให้กับฟังก์ชั่นแต่ละตัว ซึ่งคำอธิบายอาจเป็นคำพูดธรรมดา , สมการคณิตศาสตร์ , Pseudocode , ตารางการตัดสินใจ หรือ ในรูปแบบอื่นที่เหมาะสม โดยอธิบายความเหมาะสมของ Input และ Output

4.การกำหนดข้อบังคับระหว่าง ออพเจกต์

เป็นการกำหนดข้อบังคับของข้อมูลซึ่งแบ่งได้เป็นสองชนิด 1.Preconditon เป็นการกำหนดข้อบังคับให้กับข้อมูลเข้า (INPUT) ให้กับฟังก์ชั่น 2.Postconditon เป็นการกำหนดข้อบังคับให้กับข้อมูลออก (OUTPUT) การกำหนดข้อบังคับนี้จะต้องทำควบคู่กันไประหว่าง ไดนามิคโมเดล และ ฟังก์ชั่นเนลโมเดล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.การกำหนดขอบเขตุการ Optimize

การกำหนดขอบเขตุการ Optimize ในบางครั้งอาจเกิดความขัดแย้งขึ้นระหว่างจุดสองจุดหรือมากกว่า เพราะฉะนั้นก่อนทำการตัดสินใจก็ขอให้ทำการศึกษาของผลกระทบที่จะตามมา , ดูผลได้ผลเสีย , ดูลำดับความสำคัญ เสียก่อนแล้วจึงตัดสินใจเลือกกระทำการ Optimize ในส่วนที่ต้องการ

3.2 การออกแบบระบบ (system design)

การออกแบบระบบ (system design) เป็นการวางกลยุทธ์ในการแก้ปัญหาและจัดหาวิธีการแก้ไขให้กับระบบ รายละเอียดในขั้นตอนนี้ได้แก่ 1.การแบ่งระบบออกเป็นระบบย่อย (Subsystem) 2. จัดสรรระบบย่อยเหล่านี้ให้กับฮาร์ดแวร์ และ ซอฟท์แวร์ 3.กำหนดแนวคิดและนโยบายการตัดสินใจให้กับระบบเพื่อการออกแบบรายละเอียดของระบบ

ในขั้นตอนของการวิเคราะห์ จุดประสงค์ของในขั้นนี้คือพยามที่จะกำหนดว่าเราจะต้องทำอะไรบ้างในระบบโดยไม่สนใจวิธีการทำ ในทางกลับกัน ในขั้นตอนของการออกแบบระบบจุดประสงค์ในขั้นนี้คือพยายามหาวิธีการแก้ปัญหา โดยในขั้นแรกจะเริ่มในระดับสูงก่อน แล้วจึงค่อยๆเจาะลงไปในรายละเอียด

ขั้นตอนต่างๆต่อไปนี้เป็นขั้นตอนที่ผู้ออกแบบระบบจะต้องทำ

- แตกระบบออกเป็นระบบย่อย
- กำหนดขั้นตอนที่จะต้องทำงานพร้อมกัน (Concurrency)
- จัดสรรระบบย่อยให้แก่โปรเซสเซอร์
- เลือกวิธีการและการบริหารการจัดเก็บข้อมูล
- จัดสรรการให้ทรัพยากรณ์ของระบบโดยรวม
- เลือกวิธีการการควบคุมของซอฟท์แวร์
- จัดการเงื่อนไขขอบ (Boundary conditions)
- กำหนดผลได้ ผลเสียของระบบ

3.2.1 การแตกระบบออกเป็นระบบย่อย

การแตกระบบเป็นขั้นตอนแรกของการออกแบบระบบ องค์ประกอบหลักของระบบจะถูกแบ่งออกมาเป็นระบบย่อย ในแต่ละระบบย่อยๆจะมีลักษณะบางอย่างที่เหมือนกัน เช่น มีหน้าที่ (functional)ที่เหมือนกัน , มีตำแหน่งทางกายภาพที่เดียวกัน , หรือใช้ ฮาร์ดแวร์ตัวเดียวกัน

ระบบย่อยโดยทั่วไปแล้วจะถูกกำหนดโดย บริการ(Service) ของตัวมัน บริการที่ว่านี้ก็คือกลุ่มของฟังก์ชั่นที่มีการใช้อุปกรณ์บางอย่างร่วมกัน เช่น อุปกรณ์ อินพุท เอาท์พุท ตัวอย่างของระบบย่อยที่ว่านี้อาจจะพอยกตัวอย่างได้เช่น ระบบจัดการแฟ้มข้อมูลในระบบปฏิบัติการก็ถือว่าเป็นระบบย่อยได้อย่างหนึ่ง ระบบย่อยที่การติดต่อและโต้ตอบต่างควรจะเกิดขึ้นอยู่ภายในระบบ เพื่อลดการขึ้นแก่กัน (Dependency) ระหว่างระบบย่อย

ความสัมพันธ์ระหว่างระบบย่อยเหล่านี้จะอยู่ในรูปแบบ Client-Supplier หรือ Peer-to-Peer ก็ได้ ในความสัมพันธ์แบบแรกนั้นผู้ขอใช้บริการ (Client) จะต้องรู้ถึงรายละเอียดขั้นตอนวิธีการติดต่อกับฝ่ายผู้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ให้บริการ (Supplier) ในขณะที่ฝ่ายผู้ให้บริการไม่จะเป็นจะต้องสนใจขั้นตอนเหล่านี้เลย เพราะว่าการติดต่อโต้ตอบนี้ฝ่ายผู้ขอใช้บริการจะกระทำโดยใช้รูปแบบของฝ่ายผู้ให้บริการ

ในระบบการติดต่อแบบ Peer-to-Peer ฝ่ายใดก็ได้ของระบบย่อยอาจเป็นผู้เรียกใช้บริการ การติดต่อในรูปแบบนี้จะมีความซับซ้อนมากกว่าในแบบแรกเพราะว่า แต่ละฝ่ายจะต้องรู้วิธีการและข้อกำหนดในการติดต่อซึ่งกันและกัน ดังนั้นคงพอจะเห็นภาพแล้วว่าการติดต่อแบบ Client-Server จะมีความสะดวก และง่ายในการประยุกต์ใช้มากกว่าการติดต่อแบบ Peer-to-Peer เพราะการติดต่อ**แบบทางเดียว (one-way interaction)** สามารถสร้างขึ้นได้ง่าย ทำความเข้าใจได้ง่าย และ ทำการเปลี่ยนแปลงได้ง่ายกว่าการติดต่อแบบสองทาง (two-way interaction)

การแบ่งระบบย่อยสามารถทำได้ด้วยกันสองแนวคิดแนวตั้งและแนวนอน

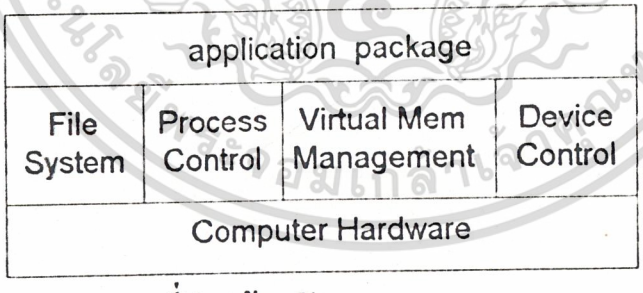
3.2.1.1การแบ่งระบบในแนวนอน

การแบ่งระบบในแนวนอน ระบบที่อยู่ด้านล่างจะเป็นตัวให้บริการแก่ระบบที่อยู่เหนือมัน รายละเอียดต่างๆ ระบบที่อยู่เหนือกว่าจะรู้รายละเอียดและข้อมูลของระบบที่อยู่ด้านล่าง ในขณะที่ระบบที่อยู่ด้านล่างจะไม่รู้รายละเอียดของระบบที่อยู่เหนือมัน (เช่นทำนอง Client-Sever)

3.2.1.2การแบ่งระบบในแนวตั้ง

การแบ่งระบบในแนวตั้งเป็นการแบ่งแยกหน้าที่ให้แก่แต่ละระบบย่อยให้มีหน้าที่อย่างเดียว ซึ่งในแต่ละระบบย่อยเหล่านี้ อาจรู้ถึงรายละเอียดซึ่งกันและกันแต่ไม่ได้รู้มากนัก ตัวอย่างของการแบ่งระบบในลักษณะนี้เช่น ระบบปฏิบัติการประกอบไปด้วย ระบบจัดการแฟ้มข้อมูล , ระบบการควบคุมการประมวลผล , ระบบการบริหารหน่วยความจำเสมือน , และ ระบบควบคุมอุปกรณ์ต่อพ่วงนอก

ลักษณะการแบ่งระบบอาจทำได้ทั้งในแนวตั้ง และ ในแนวนอนพร้อมๆกันดังตัวอย่างดังรูปต่อไปนี้



3.2.2 การกำหนดการทำงานที่ทำพร้อมกัน (Concurrency)

เป็นการกำหนดว่า ออพเจกต์ ใดบ้างที่กระทำงานพร้อมกัน และ ออพเจกต์ ใดบ้างที่ทำงานไม่พร้อมกับ ออพเจกต์ อื่น ลักษณะของ ออพเจกต์ ที่จะทำงานพร้อมกัน จะต้องสามารถรับเหตุการณ์ได้ในเวลาเดียวกันโดยไม่มีการโต้ตอบซึ่งกันและกัน ตัวอย่างของการทำงานพร้อมกันเช่น การควบคุมเครื่องยนต์ของเครื่องบินและการควบคุมปีกของเครื่องบิน ระบบย่อยที่ดีนั้นจะต้องไม่ขึ้นกับระบบย่อยอื่นๆ เพราะว่าจะได้ทำการจัดสรรฮาร์ดแวร์ให้กับแต่ละระบบ โดยไม่ต้องมีการติดต่อระหว่างระบบ ทำให้ประหยัดค่าใช้จ่าย และ เวลาในส่วนนี้ลงไปได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.3 การจัดสรรระบบย่อยให้กับโปรเซสเซอร์และทาสก์

หลักของการจัดสรรผู้ออกแบบระบบจะใช้หลักการดังนี้

1.ประมาณจำนวน และ ชนิดของฮาร์ดแวร์ที่ต้องการ จำนวนของโปรเซสเซอร์ ฮาร์ดแวร์จะขึ้นอยู่กับปริมาณการคำนวนของระบบ และความเร็วของเครื่อง

2.การเลือกใช้ระหว่าง ฮาร์ดแวร์และซอฟท์แวร์ เป็นหน้าที่ของผู้ออกแบบระบบที่จะต้องเลือกระหว่างการประยุกต์ใช้ ระบบย่อยเป็นฮาร์ดแวร์ หรือ ซอฟท์แวร์ ระบบย่อยที่เป็นฮาร์ดแวร์มีเหตุผลที่จะใช้ดังนี้

- ฮาร์ดแวร์ชิ้นนั้นตอบสนองความต้องการตามหน้าที่ที่ต้องการ เช่นการใช้ Math-Co Processor ทำการคำนวณตัวเลขจำนวนจริงแทนการประยุกต์ใช้โดยซอฟท์
- ต้องการประสิทธิภาพการคำนวณที่สูงกว่าการใช้หน่วยประมวลผลทั่วๆไป เช่นการใช้ ชิพที่ทำการคำนวณ Fast Fourier Transform (FFT)

3.การกำหนดงานให้กับโปรเซสเซอร์ อัตราการคำนวณในโปรเซสเซอร์หนึ่งอาจจะมากเกินไป เพราะฉะนั้นการแบ่งงานออกมากำหนดให้แก่โปรเซสเซอร์ สักสอง สามตัว ก็จะทำให้โหลด ลดลงได้ อีกประการหนึ่งระบบย่อยที่มีการติดต่อระหว่างระบบด้วยกันบ่อยๆ เราก็จะจัดระบบเหล่านี้กำหนดให้กับโปรเซสเซอร์ตัวเดียวกัน เพื่อเป็นการลดการติดต่อระหว่างโปรเซส ส่วนระบบที่เป็นเอกเทศไม่ได้ติดต่อกับใครเราก็จะจัดให้ใช้ โปรเซสเซอร์ต่างหาก

4.กำหนดการเชื่อมต่อทางกายภาพ การเชื่อมต่อทางกายภาพอาจดูได้จากความสัมพันธ์ของ ผู้ขอใช้บริการ-ผู้ให้บริการ ในฟังก์ชันเนลโมเดล ลักษณะของการเชื่อมต่อ (Topology) อาจเป็นแบบ Matrix , Star , Tree โดยที่ผู้ออกแบบระบบจะต้องพิจารณาจากการไหลไปของข้อมูล และ Algorithm ที่ทำการประมวลผลข้อมูล

3.2.4 การบริหารและการจัดการข้อมูล

แฟ้มข้อมูลคงจะเป็นการจัดการเก็บข้อมูลในรูปแบบที่ง่ายที่สุด อย่างไรก็ตามการปฏิบัติงานกับแฟ้มข้อมูลคงจะเป็นเรื่องที่ค่อยในระดับล่างๆ และเนื่องจากระบบแฟ้มข้อมูลในระบบคอมพิวเตอร์ที่ต่างกันก็จะไม่เหมือนกันเพราะฉะนั้นถ้าระบบของเรามีแนวโน้มที่จะต้องโยกย้าย ถ่ายเทไปหลายระบบก็คงจะไม่เป็นการดีแน่ในการใช้ แฟ้มข้อมูลเป็นตัวจัดการ

การใช้ระบบจัดการฐานข้อมูล (DBMS) คงจะเป็นทางเลือกที่ดีทางหนึ่งในการบริหารและจัดการข้อมูล การใช้ DBMS จะทำให้ฐานข้อมูลสามารถโยกย้าย ไปมาระหว่างระบบคอมพิวเตอร์ที่แตกต่างกันได้

3.2.5 การจัดการการใช้ทรัพพยากรณ์

ผู้ออกแบบระบบจะต้องทำการจัดสรรการใช้ทรัพพยากรณ์ของระบบ ซึ่งแบ่งออกเป็นสองชนิด

1.Physical Object เช่น ตัวประมวลผล , เครื่องอ่านเทป , การใช้Disk เป็นต้น การจัดการการใช้ในส่วนนี้สามารถทำได้โดยการกำหนด โปรโตคอล การใช้งาน

2.Logical names เช่น Object , ฐานข้อมูลที่ใช้ร่วมกัน เป็นต้น การจัดการในส่วนนี้จะต้องใช้ การ Lock เข้ามาช่วยไม่ให้เกิดการชนกันของการใช้ทรัพพยากรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.6 เลือกวิธีการควบคุมทางซอฟท์แวร์

Procedure-driven Systems

วิธีการควบคุมชนิดนี้การควบคุมจะอยู่ในตัวโค้ดของโปรแกรม โปรซีเจอร์จะรอรับอินพุทจากภายนอกเมื่อได้รับอินพุท โปรซีเจอร์ ก็จะทำงานต่อไปตามการควบคุม ข้อดีของการประยุกต์ใช้โดยวิธีนี้ก็คือสามารใช้ภาษาโปรแกรมธรรมดา (Conventional) ทำการประยุกต์ได้ ข้อเสียของวิธีนี้ก็คือเราจะต้องทำการ Map โปรซีเจอร์ที่ทำงานพร้อมๆกัน (Concurrency) ให้เป็นแบบ Sequential Flow Control นอกจากข้อเสียที่ว่านี้แล้ว ระบบการติดต่อกับผู้ใช้ และ การควบคุมระบบค่อนข้างที่จะไม่มีความยืดหยุ่นในการใช้งานอีกด้วย

Event-driven Systems

การควบคุมจะมีมากับตัว Dispather (ตัวตอบสนอง เหตุการณ์) ที่อยู่ในภาษา การรอรับอินพุท และ เอาท์พุทจะไม่เกิดขึ้นในวิธีการนี้ เพราะโปรซีเจอร์ต่างๆในระบบจะถูกควบคุมโดย Dispather ให้ตอบสนองตาม เหตุการณ์ที่เหมาะสม การใช้วิธีการนี้ให้ความยืดหยุ่นในรูปแบบของการควบคุมมากกว่า วิธีการแรก ตัวอย่างของระบบที่ใช้การควบคุมแบบนี้ ได้แก่ X-Window ของ SUNS MICROSYSTEM.

Concurrent Systems

การควบคุมชนิดนี้ ออพเจกต์ ต่างๆจะทำงานไปพร้อมๆกัน การส่ง เหตุการณ์ระหว่าง ออพเจกต์จะเป็นในรูปแบบ ทางเดียว (ONE-WAY) ทากส์ใดๆในระบบสามารถหยุดรอรับอินพุทได้ ในขณะที่ทากส์อื่นยังคงทำงานต่อไป ดังนั้นระบบปฏิบัติการจะต้องมีกลไกจัดลำดับ (queueing) ของเหตุการณ์ให้กับทาสก์ที่หยุดการทำงานไปตัวอย่างของระบบชนิดนี้คือ ภาษา Ada และ ตัวระบบปฏิบัติการ

อย่างไรก็ตามการควบคุมชนิดนี้ก็ยังเป็นเพียงการทดลอง และการศึกษาอยู่ภายในห้องทดลองเท่านั้นและยังไม่มีการประยุกต์ใช้อย่าเป็นทางการ

3.2.7 การกำหนดเงื่อนไขขอบ

ถึงแม้ว่าการออกแบบระบบของเราจะเน้นไปทางด้านลักษณะการใช้งานตามปกติ แต่มีอีกสิ่งหนึ่งที่ผู้ออกแบบระบบต้องคำนึงถึงด้วยนั่นก็คือ เงื่อนไขขอบต่างๆ ดังนี้

เงื่อนไขการเริ่มต้น เป็นการกำหนดค่าเริ่มต้นต่างๆไม่ว่าจะเป็นค่าคงที่ , ค่าของพารามิเตอร์ , การกำหนดตัวแปรโกลบอล เป็นต้น

เงื่อนไขการสิ้นสุด เป็นการกำหนดให้ระบบปล่อยทรัพพยากรณ์ที่ใช้ออก เมื่อสิ้นสุดการทำงานในส่วนของตัวเอง

เงื่อนไขการล้มเหลว เป็นการกำหนดมาตรการเพื่อรับรอง ความผิดพลาดที่อาจเกิดขึ้นได้ในระบบ ผู้ออกแบบระบบที่ดีควรจะรักษาสภาพเดิมของระบบให้ได้มากที่สุดเมื่อเกิด ความผิดพลาดขึ้น และถ้าเป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไปได้ควรจะบันทึก หรือ พิมพ์ ข้อมูลที่ก่อให้เกิดความผิดพลาด ออกมาด้วยก่อนที่จะทำการสิ้นสุดโปรซีเจอร์

3.2.8 กำหนดผลได้ผลเสียที่จะได้รับ

การกำหนดผลได้ผลเสียจะต้องทำก่อนการออกแบบระบบ เพื่อให้การออกแบบนั้นบรรลุผลตามต้องการ ตัวอย่างเช่น การผลิตเครื่องเล่น วิดีโอ เกมส์ในสมัยเริ่มแรกนั้น จะให้ความสำคัญกับ ขนาดของหน่วยความจำมากที่สุด ดังนั้นโปรแกรมเมอร์จึงจะต้องอาศัยความชำนาญและทักษะในการเขียน โปรแกรมให้มากที่สุด เป็นต้น

3.3 การออกแบบ ออพเจกค์ (Object Design)

ในขั้นตอนนี้เป็นเพียงการเพิ่มรายละเอียดจากสองขั้นที่ได้กล่าวมา ฉะนั้นจึงขออธิบายเป็นหัวข้อคร่าวๆดังต่อไปนี้

1.เพิ่มเติม โอเปอร์เรชั่น ให้กับ ออพเจกค์ model จากโมเดลอื่น

- เลือก โอเปอร์เรชั่น จาก Process ใน Functional model
- เลือก Operatoin จาก เหตุการณ์ ใน Dynamic model

2.ออกแบบ Algorithms สำหรับ โอเปอร์เรชั่น

- เลือกใช้ Algorithmsที่สามารถ implement ได้ง่าย
- กำหนด โครงสร้างข้อมูลให้เหมาะกับ Algorithms
- กำหนด คลาสส์ ภายใน และ โอเปอร์เรชั่น ใหม่ถ้าจำเป็น
- กำหนด หน้าที่ของแต่ละ โอเปอร์เรชั่น ให้ชัดเจน

3.กำหนด ให้ การเข้าใช้ข้อมูลสามารถทำได้อย่างรวดเร็ว

- เพิ่ม Association เข้าไปเพื่อความสะดวกในการเข้าใช้ข้อมูล
- จัดระบบการคำนวณใหม่เพื่อเพิ่มประสิทธิภาพ
- เลี่ยงการใช้ Derived Value เพื่อเลี่ยงการคำนวณซ้ำของ Expression ที่ซับซ้อน

4.พัฒนาการควบคุมของ Software จากขั้นตอน System Design

5.จัดโครงสร้างของ คลาสส์ เพื่อเพิ่ม inheritance

6.ออกแบบการใช้งาน Association

- วิเคราะห์เส้นทางของ Association หมายถึงการการประยุกต์ใช้ Pointer แทน Association ที่กำหนด โดยแบ่งเป็น Association แบบทางเดียว และ Association แบบสองทาง

7.กำหนด ออพเจกค์ แอททริบิวท์ ให้ถูกต้อง

8.รวบ คลาสส์ และ Association ให้เป็น Module

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

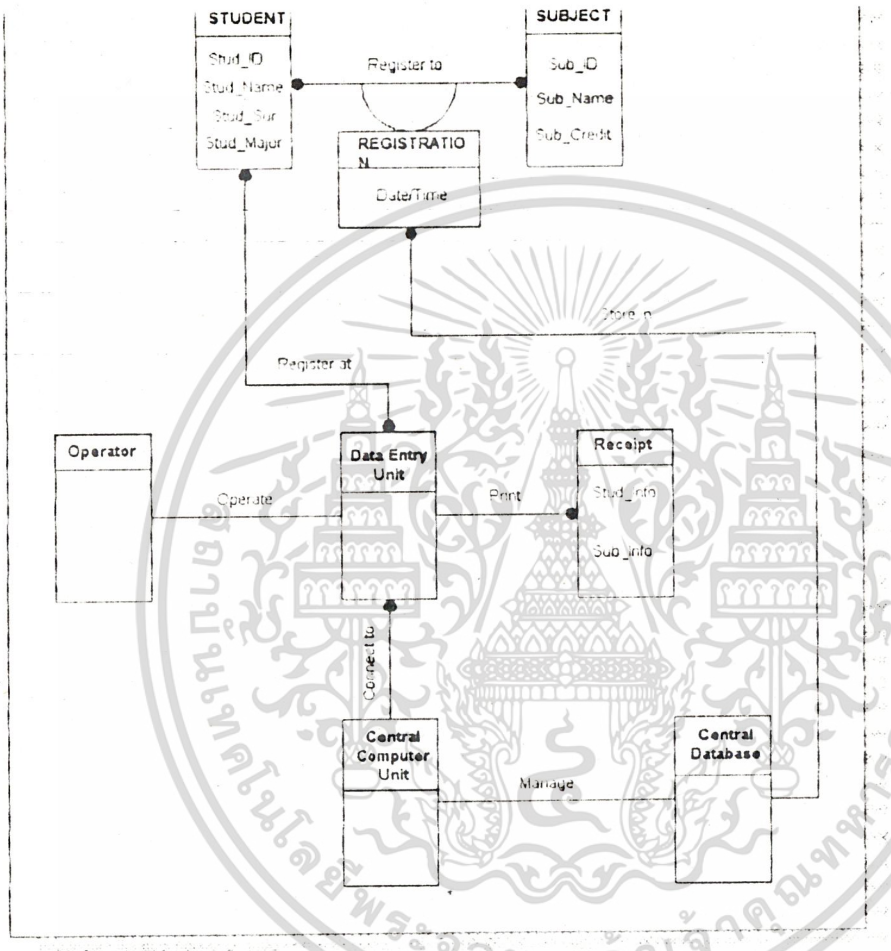
ตัวอย่างการออกแบบระบบการลงทะเบียนด้วยวิธีโอเอ็มที (OMT)

ต่อไปนี้ขอยกตัวอย่างการวิเคราะห์และออกแบบระบบโดยวิธีโอเอ็มที ซึ่งเราได้พูดถึงไปแล้วในตอนก่อนหน้านี้ ตัวอย่างนี้เป็นการวิเคราะห์และออกแบบระบบรับลงทะเบียนเรียนของนักศึกษาของมหาวิทยาลัยหนึ่งซึ่งมีนิยามความต้องการ (Requirement definition) ดังต่อไปนี้

ระบบการลงทะเบียนนี้จะถูกใช้ในการลงทะเบียนเรียนของนักศึกษา โดยผู้ใช้โปรแกรมหรือเจ้าหน้าที่รับการลงทะเบียนจะรับใบลงทะเบียนจากนักศึกษา แล้วทำการป้อนรหัสนักศึกษาและรหัสวิชาที่นักศึกษากรอกไว้ในใบลงทะเบียนเข้าสู่เครื่องคอมพิวเตอร์ จากนั้นระบบก็จะนำข้อมูลรหัสนักศึกษาและรหัสวิชาไปค้นหาชื่อ, นามสกุล, ภาควิชา, ชื่อวิชา, จำนวนหน่วยกิต เป็นต้น จากฐานข้อมูลกลาง เมื่อได้ข้อมูลที่ทำการค้นมาแล้วระบบก็จะแสดงขึ้นสู่หน้าจอเพื่อให้เจ้าหน้าที่ทำการตรวจสอบ เมื่อตรวจสอบเรียบร้อยแล้วพนักงานก็จะสั่งให้รับการลงทะเบียนซึ่งระบบก็จะทำการเพิ่มเรคคอร์ดที่เก็บข้อมูลการลงทะเบียนนั้นในฐานข้อมูลกลาง และก็จะพิมพ์ใบรับการลงทะเบียนออกมาให้นักศึกษาด้วย ระบบนี้จะมีเครื่องคอมพิวเตอร์ที่มาทำหน้าที่รับการลงทะเบียนอยู่หลายเครื่อง โดยทุกเครื่องจะเชื่อมต่อกับเครื่องคอมพิวเตอร์กลางเพื่อใช้ข้อมูลร่วมกัน

จากนิยามความต้องการในข่อหน้าที่ผ่านมานี้ เราก็จะนำมันมาใช้ในการเขียนออพเจ็กต์โมเดล (object model), ไดนามิกส์โมเดล (dynamics model) หรืออีเวนท์โฟลไดอะแกรม (event flow diagram), อีเวนท์เทรสไดอะแกรม (event trace diagram), สเตทไดอะแกรม (state diagram), ฟังก์ชันนอลโมเดล (functional model) หรือ ดาต้าโฟลไดอะแกรม (data flow diagram) ซึ่งใช้ร่วมกันในการอธิบายระบบการลงทะเบียนนี้ ออพเจ็กต์โมเดลนี้แสดงให้เห็นว่าภายในระบบรับการลงทะเบียนนี้จะประกอบด้วยออพเจ็กต์ต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

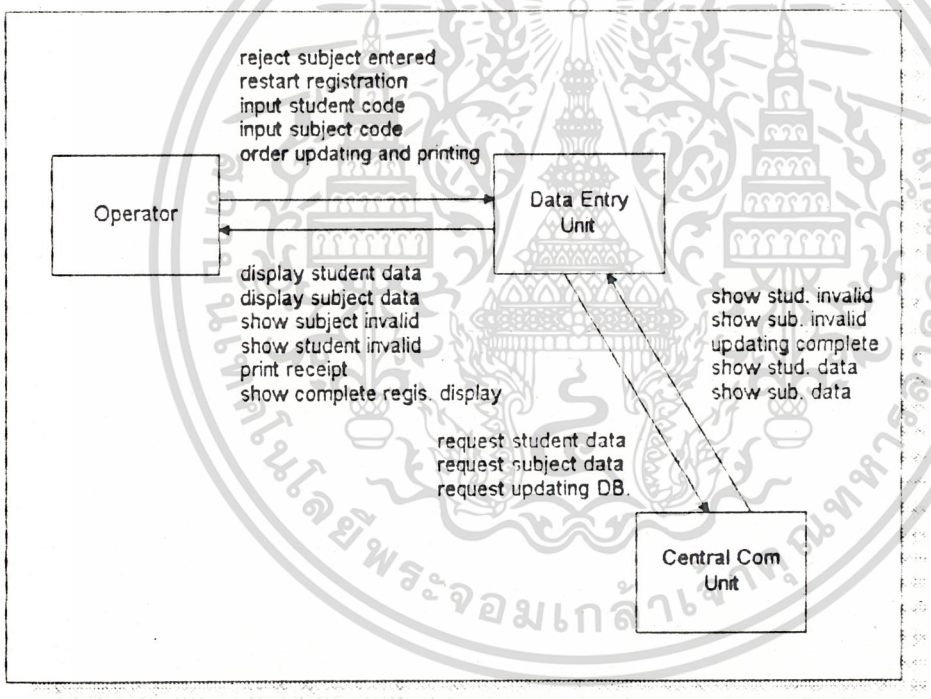


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ออพเจ็กต์โมเดลนี้แสดงให้เห็นว่าภายในระบบรับการลงทะเบียนนี้จะประกอบด้วยออพเจ็กต์ต่างๆอะไรบ้างและความสัมพันธ์ระหว่างออพเจ็กต์เหล่านี้เป็นอย่างไร เช่น ออพเจ็กต์ของส่วนรับการลงทะเบียนหรือคาต้าเอ็นทรี่ (data entry) จะสัมพันธ์กับนักศึกษาที่มาลงทะเบียนในลักษณะที่คาต้าเอ็นทรี่หนึ่งๆมีนักศึกษาหลายคนมาทำการลงทะเบียนที่นั้น และนักศึกษาสามารถลงทะเบียนที่คาต้าเอ็นทรี่ใดก็ได้ แต่ละคาต้าเอ็นทรี่จะทำการพิมพ์ใบรับการลงทะเบียนหลายๆใบ แต่คาต้าเอ็นทรี่หนึ่งๆจะมีเจ้าพนักงานควบคุมอยู่หนึ่งคน คอมพิวเตอร์กลางหรือเซ็นทรัลคอมพิวเตอร์ยูนิต (central computer unit) จะเชื่อมต่ออยู่กับคาต้าเอ็นทรี่หลายๆเครื่อง และเซ็นทรัลคอมพิวเตอร์ยูนิตจะจัดการฐานข้อมูลกลางหรือเซ็นทรัลคาต้าเบส (central database)

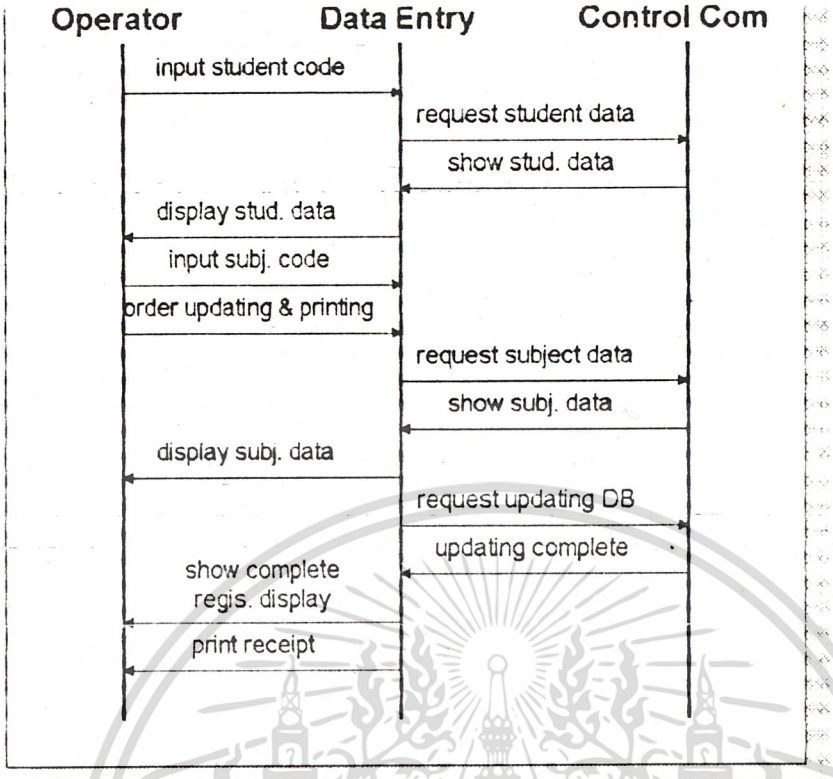
จากออพเจ็กต์โมเดลแรกนี้เราต้องทำการปรับปรุงเพิ่มเติมอีกโดยให้ผู้วิเคราะห์ (analyst) มาทำการตรวจสอบว่ามีความซ้ำซ้อนของคลาสส์ต่างๆหรือไม่ จำเป็นต้องมีการรวมคลาสส์ต่างๆเข้าด้วยกันหรือต้องนำแนวคิดของอินเฮียริแทนส์เข้ามาร่วมใช้หรือไม่ เมื่อทำการตรวจสอบเรียบร้อยแล้วจึงสามารถยอมรับออพเจ็กต์โมเดลดังกล่าวในการอธิบายระบบทางด้านสแตติก (static) ได้

ต่อไปก็เป็นอีเวนท์โฟลไดอะแกรมของระบบรับการลงทะเบียนที่เขียนได้จากลักษณะของระบบงานดังกล่าวได้ดังรูปต่อไปนี้



ไดอะแกรมนี้เป็นส่วนหนึ่งของไดนามิกส์โมเดล ซึ่งเป็นแบบจำลองที่ช่วยอธิบายการปฏิสัมพันธ์ (interaction) กันระหว่างออพเจ็กต์ต่างๆในระบบ สำหรับอีกส่วนหนึ่งที่เราจะใช้ร่วมกับอีเวนท์โฟลไดอะแกรมก็คือ อีเวนท์เทรสไดอะแกรม สำหรับอีเวนท์เทรสไดอะแกรมจะแสดงลำดับขั้นตอนของการติดต่อกันระหว่างออพเจ็กต์ เมื่อรวมทั้งสองไดอะแกรมเข้าด้วยกันเราก็จะใช้เป็นไดนามิกส์โมเดล ซึ่งใช้อธิบายระบบทางด้านไดนามิกส์ ต่อไปนี้จะแสดงอีเวนท์เทรสไดอะแกรมของระบบรับการลงทะเบียนดังกล่าว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



จากอีเวนท์โฟลไดอะแกรมนี้สามารถอธิบายได้ว่า ระหว่างออพเจ็กต์โอเปอเรเตอร์กับคาต้าเอ็นทรี่ยูนิต มีการติดต่อกันด้วยอีเวนท์อะไรบ้าง อีเวนท์ใดใครส่งไปให้ใคร เช่น input student code, input subject code ฯ เป็นอีเวนท์ที่โอเปอเรเตอร์ส่งไปให้คาต้าเอ็นทรี่ยูนิตเพื่อกระตุ้นให้เกิดการทำงานถึงการเปลี่ยนสถานะ และ display student data, show student invalid ฯ เป็นอีเวนท์ที่คาต้าเอ็นทรี่ยูนิตส่งกลับไปให้โอเปอเรเตอร์ ระหว่างคู่ของออพเจ็กต์คาต้าเอ็นทรี่กับเซ็นทรัลคอมพิวเตอร์ยูนิตกับคาต้าเอ็นทรี่ก็มีชุดของอีเวนท์ที่ใช้ในการติดต่อซึ่งกันและกัน เช่น request student data, request updating DB ฯ เป็นอีเวนท์ที่คาต้าเอ็นทรี่ส่งไปให้เซ็นทรัลคอมพิวเตอร์ยูนิต และอีเวนท์ show student invalid, updating complete ฯ เป็นอีเวนท์ที่เซ็นทรัลคอมพิวเตอร์ยูนิตส่งกลับมาให้คาต้าเอ็นทรี่ยูนิต

ในรูปอีเวนท์เทรสไดอะแกรมก็จะแสดงลำดับเหตุการณ์ที่แต่ละออพเจ็กต์ติดต่อกันผ่านทางอีเวนท์ต่างๆ ถ้าจะยกตัวอย่างจากรูปนี้ เมื่อตอนเริ่มต้นโอเปอเรเตอร์ก็จะส่ง input student code หรืออีเวนท์ที่แสดงว่าได้ทำการป้อนรหัสนักศึกษาไปให้คาต้าเอ็นทรี่ จากนั้นคาต้าเอ็นทรี่ก็จะตอบสนองโดยการส่งอีเวนท์ request student data ไปยังเซ็นทรัลคอมพิวเตอร์ยูนิตเพื่อขอข้อมูลนักศึกษารหัสประจำตัวดังกล่าวมาตรวจสอบ เมื่อเซ็นทรัลคอมพิวเตอร์ได้ค้นข้อมูลนักศึกษาตามที่มีการขอมาได้แล้วก็จะส่งอีเวนท์ show stud. data กลับมาให้คาต้าเอ็นทรี่ยูนิตพร้อมข้อมูลของนักศึกษาคนดังกล่าว

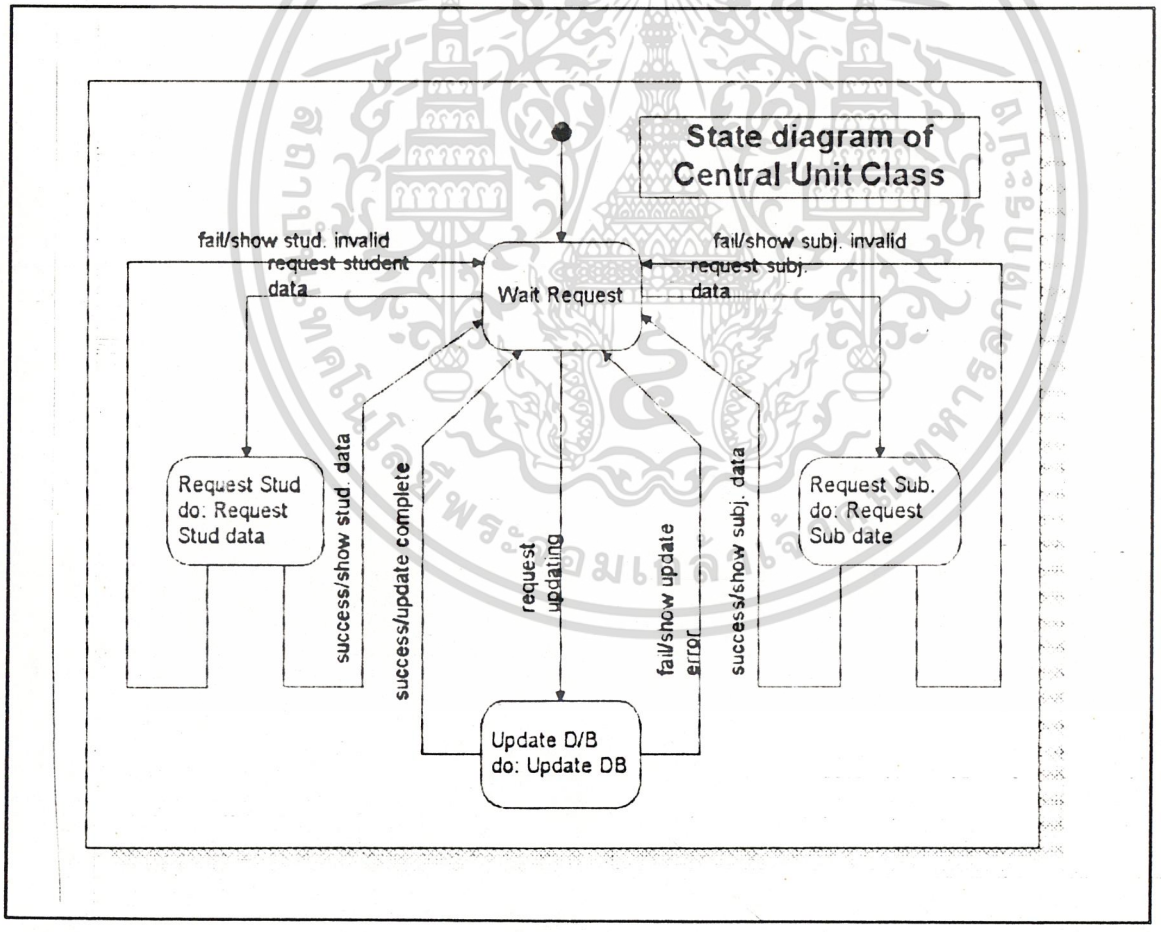
ต่อจากนั้นคาต้าเอ็นทรี่ยูนิตก็จะส่งอีเวนท์ display student data เพื่อแสดงข้อมูลของนักศึกษาคนดังกล่าวมาให้โอเปอเรเตอร์ตรวจสอบ เมื่อโอเปอเรเตอร์ตรวจสอบข้อมูลดังกล่าวเรียบร้อยแล้วโอเปอเรเตอร์ก็จะทำการป้อนรหัสวิชาซึ่งจะเกิดเป็นอีเวนท์ input sub. code ส่งไปให้คาต้าเอ็นทรี่ และจากนั้นโอเปอเรเตอร์ก็ส่งอีเวนท์ order updating and printing เพื่อให้ทำการเพิ่มข้อมูลการลงทะเบียนดังกล่าวลงในฐานข้อมูลและพิมพ์ใบรับการลงทะเบียน เมื่อคาต้าเอ็นทรี่ได้รับอีเวนท์ดังกล่าวแล้วก็จะส่งอีเวนท์ request subject data ไปให้เซ็นทรัล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

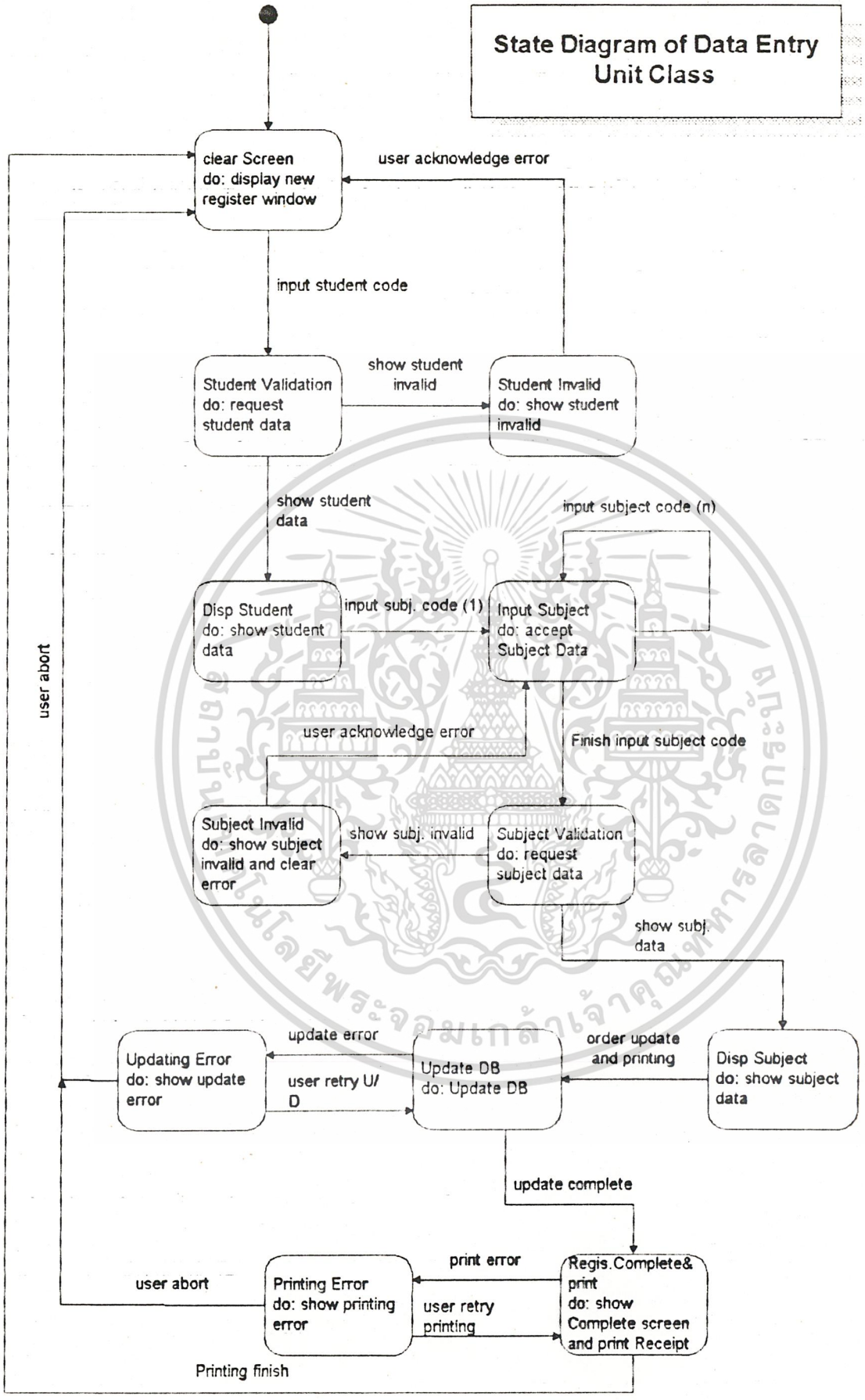
คอมพิวเตอร์ยูนิตเพื่อค้นหาข้อมูลของวิชาจากฐานข้อมูล เซ็นทรัลคอมพิวเตอร์ยูนิตก็จะทำการค้นหาข้อมูลแล้วส่งกลับให้กับคาต้าเอ็นทรี่ยูนิตซึ่งเกิดขึ้นพร้อมกับอีเวนท์ show sub. data เมื่อคาต้าเอ็นทรี่ได้รับข้อมูลวิชาแล้วก็จะแสดงผลให้กับโอเปอเรเตอร์ซึ่งคืออีเวนท์ display subject data จากนั้นคาต้าเอ็นทรี่ก็จะส่ง request updating DB ไปให้เซ็นทรัลคอมพิวเตอร์ยูนิต และเมื่อทำการเพิ่มข้อมูลในฐานข้อมูลเสร็จก็จะส่งอีเวนท์ updating complete กลับมาบอกคาต้าเอ็นทรี่ยูนิต เมื่อได้รับอีเวนท์ดังกล่าวแล้วมันจึงทำการพิมพ์ใบรับการลงทะเบียนซึ่งทำให้เกิดอีเวนท์ print receipt กลับไปยังโอเปอเรเตอร์

จากรูปอีเวนท์โฟลไดอะแกรมเราจะสังเกตว่ามันไม่ได้รวมออพเจ็กต์ทุกๆอันบนออพเจ็กต์โมเดล ที่เป็นเช่นนี้เพราะว่า ส่วนที่นำมาอยู่ในไดอะแกรมเป็นส่วนที่เราสนใจ หรือ นำมาสร้างเป็นโปรแกรมที่จะใช้งานในระบบรับการลงทะเบียนนี้

จากอีเวนท์โฟลและอีเวนท์เทรสไดอะแกรมที่กล่าวมาแล้ว เราจะทำการกำหนดสเตทไดอะแกรม (state diagram) ซึ่งเป็นไดอะแกรมที่ใช้แสดงสถานะต่างๆของออพเจ็กต์ในระบบว่าจะมีการเปลี่ยนสถานะอย่างไรเมื่อมีเอเวนท์เกิดขึ้น ด้วยสเตทไดอะแกรมนี้เราก็สามารถอธิบายระบบทางด้านไดนามิกส์ได้ลึกลงไปถึงระดับออพเจ็กต์ซึ่งเป็นองค์ประกอบของระบบ ต่อไปนี้แสดงสเตทไดอะแกรมของเซ็นทรัลคอมพิวเตอร์ยูนิตและคาต้าเอ็นทรี่ยูนิตตามลำดับ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากสเตทไดอะแกรมของเซ็นทรัลคอมพิวเตอร์ยูนิต ตอนเริ่มต้นเซ็นทรัลคอมพิวเตอร์ยูนิตจะอยู่ในสเตท Wait Request คอยรับรีเควส (request) จากคาต้าเอ็นทรี่ตัวต่างๆ สมมุติว่ามีอีเวนท์ request subj. data เพื่อขอข้อมูลวิชามาจากคาต้าเอ็นทรี่ เซ็นทรัลคอมพิวเตอร์ยูนิตก็จะเปลี่ยนสเตทมาเป็น Request Sub. ซึ่ง ณ.ที่สเตทนี้เซ็นทรัลคอมพิวเตอร์ยูนิตจะทำการค้นหาข้อมูลของวิชาที่ได้รับการร้องขอข้อมูลมา ถ้ามันสามารถค้นหาข้อมูลของวิชาดังกล่าวขึ้นมาได้ เซ็นทรัลคอมพิวเตอร์ยูนิตก็จะส่งข้อมูลดังกล่าวนั้นกลับไปให้คาต้าเอ็นทรี่ เกิดเป็นอีเวนท์ success/show subj. data แล้วเซ็นทรัลคอมพิวเตอร์ยูนิตก็จะเปลี่ยนสถานะกลับไปเป็น Wait Request อีกครั้งหนึ่ง เมื่อเกิดอีเวนท์อื่นๆอีกก็จะมีการเปลี่ยนสถานะในลักษณะดังที่กล่าวมาแล้ว

สำหรับกรณีของสเตทไดอะแกรมของคาต้าเอ็นทรี่ยูนิตนั้น เมื่อเริ่มต้นคาต้าเอ็นทรี่ยูนิตจะอยู่ในสเตท Clear Screen ซึ่งจะทำการจัดเตรียมหน้าต่างของระบบให้ว่างเปล่ารอรับการป้อนข้อมูลจากพนักงาน เมื่อพนักงานป้อนรหัสนักศึกษาเสร็จเกิดเป็นอีเวนท์ input student code มากระตุ้นให้คาต้าเอ็นทรี่ยูนิตเปลี่ยนสเตทไปเป็น Student Validation ซึ่ง ณ.สเตทนี้คาต้าเอ็นทรี่จะทำการตรวจสอบข้อมูลของนักศึกษาคนนี้ ถ้าข้อมูลของนักศึกษาคนนี้มีอย่างถูกต้อง สเตทของคาต้าเอ็นทรี่ก็จะถูกกระตุ้นให้เปลี่ยนไปอีกเป็นสเตท Disp Student แต่ถ้าคาต้าเอ็นทรี่พบว่าข้อมูลของนักศึกษาคนดังกล่าวไม่ถูกต้อง สเตทของคาต้าเอ็นทรี่ก็จะถูกเปลี่ยนเป็น Student Invalid เพื่อแสดงข้อผิดพลาดนั้นให้พนักงานรับลงทะเบียนทราบ

ในสเตท Disp Student คาต้าเอ็นทรี่ก็จะแสดงข้อมูลเหล่านั้นขึ้นให้พนักงานทราบ จากสเตทนี้เมื่อพนักงานเริ่มป้อนรหัสของวิชา สเตทของคาต้าเอ็นทรี่ยูนิตก็จะถูกเปลี่ยนไปอีกเป็น Input Subject เพื่อรับรหัสวิชาต่างๆที่นักศึกษาต้องการจะลงทะเบียน เมื่อพนักงานป้อนรหัสวิชาสุดท้ายเสร็จ สถานะของคาต้าเอ็นทรี่ยูนิตก็จะถูกเปลี่ยนเป็น Subject Validation ซึ่งเป็นสเตทที่คาต้าเอ็นทรี่ยูนิตจะทำการตรวจสอบข้อมูลวิชาที่นักศึกษาต้องการลงทะเบียนเหล่านั้น ถ้าข้อมูลวิชาไม่ถูกต้องสเตทก็ถูกเปลี่ยนไปเป็น Subject Invalid เพื่อบอกแก่พนักงาน ถ้าข้อมูลถูกต้องก็จะเปลี่ยนไปเป็นสเตท Disp Subject เพื่อแสดงข้อมูลของวิชาเหล่านั้น

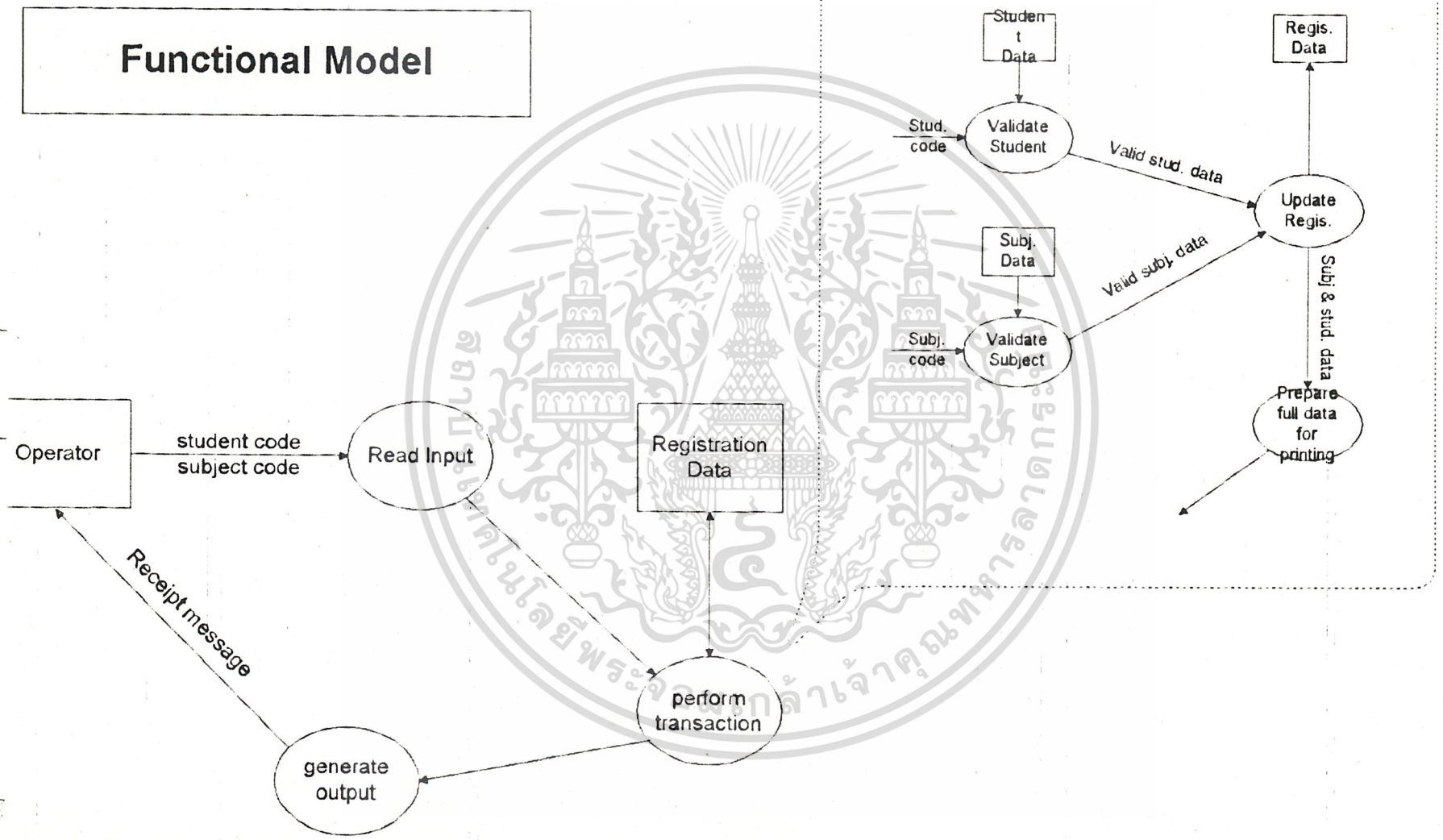
เมื่อพนักงานรับทราบข้อมูลวิชาเหล่านั้นแล้วก็จะมีการสั่งให้อัพเดทฐานข้อมูลและพิมพ์ใบรับการลงทะเบียน ซึ่งก็คืออีเวนท์ order update and printing เพื่อกระตุ้นให้คาต้าเอ็นทรี่ยูนิตเปลี่ยนสเตทไปเป็น Update DB ณ.สเตทนี้คาต้าเอ็นทรี่จะทำการเพิ่มข้อมูลการลงทะเบียนของนักศึกษาลงในฐานข้อมูล (โดยผ่านเซ็นทรัลคอมพิวเตอร์ยูนิต) ถ้าเกิดปัญหาสเตทก็จะเปลี่ยนไปเป็น Updating Error แต่ถ้าไม่เกิดปัญหาใดก็จะเปลี่ยนสเตทเป็น Regis. Complete&print ที่สเตท Updating Error ก็จะบอกให้พนักงานทราบถึงข้อผิดพลาด ถ้าพนักงานพยายามจะอัพเดทฐานข้อมูลอีกครั้งก็จะเกิดอีเวนท์ user retry U/D แล้วสถานะก็จะกลับไปเป็น Update DB อีกครั้ง แต่ถ้าพนักงานไม่ต้องการอัพเดทอีกก็สามารถสั่งยกเลิกได้โดยเกิดอีเวนท์ user abort แล้วสถานะก็จะเปลี่ยนไปเป็น Clear Screen อีกครั้ง

เมื่ออยู่ที่สเตท Regis Complete & Print คาต้าเอ็นทรี่ยูนิตก็จะทำการแสดงการอัพเดทฐานข้อมูลว่าเป็นไปอย่างเรียบร้อย แล้วทำการพิมพ์ใบรับการลงทะเบียนออกมา ถ้าการพิมพ์มีปัญหาสเตทก็จะถูกเปลี่ยนไปเป็น Printing Error แต่ถ้าการพิมพ์ถูกต้องก็จะเกิดอีเวนท์ Printing finish ทำให้สเตทกลับไปเป็น Clear Screen อีกครั้ง ณ.สเตท Printing Error พนักงานสามารถจะยกเลิกการพิมพ์หรือพยายามพิมพ์อีกครั้งก็ได้ ในลักษณะทำนองเดียวกันกับสเตท Update DB

มาถึงโมเดลที่สามในการใช้วิธีในการวิเคราะห์และออกแบบแบบโอเอ็มที ซึ่งก็คือฟังก์ชันนอลโมเดล ด้วยฟังก์ชันนอลโมเดลนี้จะช่วยอธิบายการเปลี่ยนแปลงสภาพของข้อมูลต่างๆในระบบของเราว่า จากจุดหนึ่งไปอีกจุดหนึ่งข้อมูลเหล่านั้นผ่านโพรเซสอะไร มีการเปลี่ยนแปลงไปเป็นข้อมูลอะไร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Functional Model



จากรูปฟังก์ชันนอลโมเดลที่แสดงนั้น จะเห็นว่าพนักงานจะทำการส่งข้อมูลรหัสนักศึกษาและรหัสวิชาผ่านกระบวนการอ่านอินพุต แล้วข้อมูลเหล่านั้นจะถูกส่งผ่านต่อไปทำการโพรเซสในโพรเซส perform transaction ทำการอัพเดทข้อมูลการลงทะเบียน แล้วส่งข้อมูลของการลงทะเบียนทั้งหมดมาสู่โพรเซสของการสร้างเอาท์พุตซึ่งในระบบนี้ก็คือการพิมพ์ออกเป็นใบรับการลงทะเบียนให้กับนักศึกษานั่นเอง

ภายในโพรเซส perform transaction จะสามารถอธิบายถึงการเปลี่ยนแปลงสภาพของข้อมูลได้ในรายละเอียดดังกรอบเส้นประในรูปฟังก์ชั่นนอลโมเดลที่แสดงในตอนต้น ภายในกรอบเส้นประนั้นแสดงให้เห็นว่ารหัสนักศึกษาและรหัสวิชาจะถูกส่งผ่านโพรเซส Validate Student และ Validate Subject ตามลำดับ จากนั้นส่งข้อมูลที่รับการตรวจสอบแล้วไปยังโพรเซส Update Regis เพื่อทำการอัพเดทฐานข้อมูลของการลงทะเบียน ภายหลังจากโพรเซส Update Regis. ข้อมูลนักศึกษาและวิชาก็จะถูกส่งไปยังโพรเซส Prepare full data for printing เพื่อเตรียมข้อมูลทั้งหมดเพื่อจะใช้ในการพิมพ์ต่อไป

โดยการใช้วิธีการแบบโอเอ็มทีนี้ระบบที่ทำการวิเคราะห์และออกแบบจะถูกแสดงอยู่ในรูปของโมเดลทั้งสาม ดังที่แสดงให้เห็นเป็นตัวอย่างไปแล้วถึงระบบรับการลงทะเบียนที่ถูกวิเคราะห์ออกแบบโดยใช้วิธีการนี้ ต่อจากขั้นตอนของการวิเคราะห์และออกแบบถึงจะเป็นการนำโมเดลเหล่านี้ไปทำการอิมพลีเมนต์ให้เกิดเป็นระบบงานที่วิ่งจริงๆ ซึ่งการอิมพลีเมนต์นั้นอาจทำได้โดยใช้ภาษาที่มีการสนับสนุนออพเจ็กต์โอเรียนเต็ด (ที่เรียกภาษาแบบนี้ว่าภาษาโอโอพี (OOP)) หรือจะใช้โพรซีเดอรอลโปรแกรมมิ่ง (Procedural Programming) ธรรมดาก็สามารถทำได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การเปรียบเทียบระหว่างโอเอ็มทีกับระเบียบวิธีการแบบอื่นๆ

4.1 กล่าวนำ

ในบทนี้จะเป็นการสรุประเบียบวิธีการทางวิศวกรรมซอฟต์แวร์แบบอื่นๆ
และเปรียบเทียบพวกมันกับวิธีโอเอ็มที (OMT, Object Modeling Technique)
โดยที่เราจะเน้นสตรักเจอร์ดแอนนาไลซิส/สตรักเจอร์ด
ดีไซน์ (SA/SD, Structured Analysis / Structured Design), แจ็กสันสตรักเจอร์ดดีไซน์ (JSD, Jackson Structured Design) และเราจะมาดูกันถึงจุดอ่อนจุดแข็งของวิธีการเหล่านี้

เป้าหมายหลักของเราในบทนี้ก็คือ
เราจะพยายามชี้ให้เป็นความแตกต่างระหว่างโอเอ็มทีกับวิธีการอื่นๆ
ซึ่งจะส่งผลให้เราสามารถเข้าใจโอเอ็มทีได้ดียิ่งขึ้น การกล่าวถึงระเบียบวิธีการแบบอื่นๆนั้น
เราจะกล่าวอย่างสั้นและสรุปเพื่อที่จะพอให้เห็นเป็นแนวทาง
แต่ถ้าผู้อ่านมีความสนใจในรายละเอียดของระเบียบวิธีการแบบใดเป็นพิเศษ
ก็สามารถค้นคว้าต่อได้จากบัญชีรายชื่อหนังสือหรือเอกสารอ้างอิงที่จะมีกำกับไว้ให้ภายในบทนี้

4.2 สตรักเจอร์ดแอนนาไลซิส/สตรักเจอร์ดดีไซน์ (SA/SD)

ในปัจจุบันนี้ ระเบียบวิธีการในทางวิศวกรรมซอฟต์แวร์จะอยู่บนพื้นฐานของดาต้าโฟลว์ไดอะแกรม (data flow diagram) เป็นส่วนใหญ่
ในทางปฏิบัติแล้วก็จะมีดาต้าโฟลว์ไดอะแกรมอยู่หลากหลายแบบที่มีการนำมาใช้กัน
เราจะพูดถึงสตรักเจอร์ดแอนนาไลซิส /
สตรักเจอร์ดดีไซน์ในแง่ที่มันเป็นตัวแทนของระเบียบวิธีการแบบที่ใช้ดาต้าโฟล์วไดอะแกรมแบบอื่นๆ

ทั้งระเบียบวิธีการแบบโอเอ็มทีและเอสเอ / เอสดี (SA/SD)
ต่างก็รวมส่วนประกอบในการโมเดลสิ่งระบบที่คล้ายคลึงกัน
วิธีการทั้งสองสนับสนุนการมองระบบในสามมุมมองเช่นกัน คือ ออพเจ็กต์, ไดนามิกส์ และ ฟังก์ชั่นนอล โมเดล สิ่งที่ต่างกันระหว่างวิธีการทั้งสองนี้ก็คือ โอเอ็มทีจะเน้นหนักที่ออพเจ็กต์โมเดล
ซึ่งออพเจ็กต์จะเป็นตัวแทนสิ่งต่างๆในโลกแห่งความเป็นจริง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ออพเจ็กต์และความสัมพันธ์ระหว่างกันจะถูกใช้เป็นฐานในการทำความเข้าใจพฤติกรรมทางไดนามิกส์และฟังก์ชั่นนอลของระบบ ถ้าเปรียบเทียบกับเอสเอ / เอสดีแล้ว เอสเอ / เอสดีจะเน้นที่การแบ่งแยกย่อย (Decomposition) ระบบตามหน้าที่หรือฟังก์ชันนอลของระบบ
ระบบจะถูกมองเสมือนเป็นสิ่งที่ใช้ฟังก์ชันแก่ผู้ใช้มากกว่าจะเป็นออพเจ็กต์

4.2.1 สรุปของวิธีแบบเอสเอ/เอสดี

เอสเอ/เอสดีมีสัญลักษณ์อยู่มากมายที่ใช้ในการอธิบายซอฟต์แวร์ให้เป็นระบบระเบียบ ในระหว่างขั้นตอนการวิเคราะห์ คาต้าโฟล์วไดอะแกรม, โพรเซสสเปกซิฟิเคชั่น (process specification), คาต้าดิกชั่นนารี่ (data dictionary), สเตททรานซิชั่นไดอะแกรม (state transition diagram), และ เอ็นติตี้รีเลชั่นชิพไดอะแกรมจะถูกใช้เพื่ออธิบายระบบที่ทำการวิเคราะห์ให้อยู่ในรูปแบบนามธรรม (logical) ในขั้นตอนการออกแบบ,รายละเอียดของระบบจะถูกเพิ่มเติมลงไปในโมเดลต่างที่สร้างในขั้นตอนการวิเคราะห์ และคาต้าโฟลว์ไดอะแกรมก็จะถูกแปลงไปเป็นสตรักเจอร์ชาร์ท (structure chart) และก็จะถูกแปลงไปเป็นโค๊ด (code) ของภาษาที่ใช้ในการโปรแกรม

คาต้าโฟลว์ไดอะแกรมจะเป็นการจำลองของการแปลงของข้อมูลเมื่อมันไหลผ่านระบบ และเอสเอ/เอสดีเน้นตรงเรื่องของคาต้าโฟลว์นี้มาก คาต้าโฟลว์ไดอะแกรมจะประกอบด้วยโพรเซส, คาต้าโฟลว์, แอ็กเตอร์ (actors) และคาต้าสตอร์ (data store) เริ่มต้นจากคาต้าโฟลว์ไดอะแกรมระดับบนสุด
วิธีการแบบเอสเอ/เอสดีจะทำการวนแตกโพรเซสที่มีความซับซ้อนลงไปเป็นซับไดอะแกรม (subdiagrams) จนกระทั่งโพรเซสที่ถูกแยกย่อยแล้วนั้นง่ายพอที่จะทำการสร้างได้
เมื่อโพรเซสที่เป็นผลจากการแยกย่อยนั้นง่ายพอที่จะสร้างแล้ว การวนเพื่อทำการแตกโพรเซสก็จะหยุดลง และโพรเซสสเปกซิฟิเคชั่นก็จะถูกเขียนสำหรับแต่ละโพรเซสในระดับต่ำสุดนั้นๆ
โพรเซสสเปกซิฟิเคชั่นที่ทำขึ้นนี้อาจถูกอธิบายโดยใช้ดิซิชั่นเทเบิล (decision table), ซูโดโค๊ด (pseudocode) หรือเทคนิควิธีการอื่นๆก็ได้

คาต้าดิกชั่นนารี่จะเป็นสิ่งที่ใช้เก็บรายละเอียดที่ไม่ได้ใส่ไว้ในคาต้าโฟลว์ไดอะแกรม
คาต้าดิกชั่นนารี่จะเก็บความหมายหรือคำอธิบายของคาต้าโฟลว์ คาต้าสตอร์
และความหมายของชื่อกับคำสำคัญอื่นๆ

สเตททรานซิชั่นไดอะแกรมจะแสดงพฤติกรรมที่ขึ้นกับเวลาของระบบหรือโปรแกรม
ซึ่งจะคล้ายคลึงกับสเตทโมเดลของวิธีโอเอ็มที
สเตทไดอะแกรมส่วนใหญ่จะอธิบายโพรเซสการควบคุมหรือไทม์มิ่ง (timing)
ของการปฏิบัติการของฟังก์ชันและการเข้าถึงข้อมูลที่ถูกกระตุ้นโดยเหตุการณ์หรืออีเวนท์หนึ่งๆ

เอ็นติตี้รีเลชั่นชิพไดอะแกรมจะเน้นถึงความสัมพันธ์กันระหว่างคาต้าสตอลต่างๆที่เห็นอยู่ในโพรเซสสเปกซิฟิเคชั่น แต่ละคาต้าเอลลิเมนท์ (data element)
ในอีอาร์ไดอะแกรมจะสัมพันธ์กับคาต้าสตอลอันหนึ่งบนคาต้าโฟลว์ไดอะแกรม

เครื่องมือต่างๆที่กล่าวมาแล้วนี้
ถูกใช้ในระหว่างกระบวนการวิเคราะห์ระบบหรือสตรักเจอร์ดแอนนาไลซิส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สตรักเจอร์คดีไซน์จะตามมาหลังจากสตรักเจอร์คแอนนาไลซิส
สตรักเจอร์คดีไซน์นี้จะจัดการกับระบบหรือโปรแกรมในรายละเอียดที่ต่ำลงไปอีกจากขั้นตอนการวิเคราะห์
เป็นต้นว่า ในระหว่างการทำสตรัก
เจอร์คดีไซน์, โพเซสบนคาต้าโฟลว์ไดอะแกรมจะถูกจับรวบกับเป็นกลุ่มของงาน (tasks) และถูกกำหนดไปยัง
โพรเซสของระบบปฏิบัติการและซีพียู (CPU)
คาต้าโฟลว์ไดอะแกรมจะถูกแปลงไปเป็นฟังก์ชั่นในภาษาที่ใช้ในการโปรแกรม
และสตรักเจอร์คชาร์ทจะถูกสร้างเพื่อแสดงให้เห็นถึงลำดับการเรียกใช้ในรูปแบบโพรซีเดอร์คอลล์ทรี (procedure call tree)

4.2.2 การเปรียบเทียบกับโอเอ็มที

เอสเอ/เอสดีกับโอเอ็มทีมีสิ่งที่คล้ายกันอยู่หลายอย่าง
ระเบียบวิธีการทั้งสองใช้โครงสร้างการจำลองระบบที่คล้ายคลึงกัน
อีกทั้งยังให้มุมมองระบบทั้งสามมุมมองเหมือนกันด้วย ความแตกต่างระหว่างเอสเอ/เอสดีกับโอเอ็มทีนั้น
โดยหลักๆก็จะเป็นในเรื่องของสไตล์และการเน้นหนัก
ในวิธีเอสเอ/เอสดีนั้นการเน้นหนักจะตกอยู่ที่ฟังก์ชั่นนอลโมเดล รองลงมาก็จะเป็นไดนามิกส์โมเดล
และออพเจ็กต์โมเดลเป็นอันดับสุดท้าย เมื่อเทียบกับโอเอ็มทีแล้ว โอเอ็มทีจะเน้นที่ตัวออพเจ็กต์โมเดลมากที่สุด
ไดนามิกส์และฟังก์ชั่นนอลโมเดลน้อยรองกันลงมาตามลำดับ

เอสเอ/เอสดีจะจัดระบบในรูปแบบโพรซีเดอร์ประกอบเข้าด้วยกัน
ในขณะที่โอเอ็มทีจะจัดระบบอยู่ในรูปแบบของวัตถุในโลกแห่งความเป็นจริง
หรือออพเจ็กต์ในจิตนาการที่อยู่ในความคิดเกี่ยวกับโลกแห่งความเป็นจริงของผู้ใช้
เนื่องจากการเปลี่ยนรีควายเมนต์มักจะเป็นการเปลี่ยนแปลงทางฟังก์ชันของระบบมากกว่าในทางวัตถุหรือออพเจ็กต์ของระบบ
ดังนั้นการเปลี่ยนแปลงจึงมักจะมีผลกระทบอย่างมากในการออกแบบที่อยู่บนพื้นฐานของโพรซีเดอร์
เมื่อเปรียบเทียบแล้วการเปลี่ยนแปลงทางฟังก์ชันถูกทำให้เหมาะสมในการออกแบบแบบออพเจ็กต์โอเรียนเต็ดโดยการเพิ่มเติมหรือเปลี่ยนแปลงโอเปอร์เรชั่น ปล่อยให้โครงสร้างพื้นฐานของออพเจ็กต์ไม่ถูกเปลี่ยนแปลง
เอสเอ/เอสดีจึงเหมาะสมสำหรับปัญหาที่ฟังก์ชันมีความสำคัญมากกว่าและซับซ้อนกว่าตัวข้อมูล
ที่เป็นเช่นนี้ก็เนื่องจากว่าเอสเอ/เอสดีได้ถูกคิดขึ้นบนสมมุติฐานของปัญหาประเภทดังกล่าว

เอสเอ/เอสดีดีไซน์ (ผลการออกแบบโดยวิธีเอสเอ/เอสดี)
จะมีขอบเขตของระบบที่ถูกนิยามไว้อย่างชัดเจน
คือแบ่งระหว่างซอฟต์แวร์โพรซีเดอร์ที่ต้องติดต่อกับโลกแห่งความเป็นจริง
โครงสร้างของเอสเอ/เอสดีดีไซน์จะถูกสร้างขึ้นเป็นส่วนๆจากขอบเขตของระบบ
ดังนั้นมันจึงเป็นการยากที่จะขยายดีไซน์ให้เป็นไปตามขอบเขตใหม่
มันจะเป็นการง่ายกว่าที่จะขยายออพเจ็กต์โอเรียนเต็ดดีไซน์
โดยอาจเพิ่มออพเจ็กต์และรีเลชั่นชิพใกล้ๆกับขอบเขตเพื่อที่จะแสดงออพเจ็กต์ที่มีอยู่ก่อนหน้านี้แต่อยู่ภายนอกขอ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

งขอบเขตเดิม ออพเจ็กต์โอเรียนเต็ดดีไซน์จึงเหมาะสมกว่าเมื่อมีเหตุการณ์ที่ต้องเปลี่ยนแปลง
และสามารถขยายขอบเขตระบบได้ง่ายกว่าด้วย

การเปรียบกันอย่างตรงไปตรงมาระหว่างออพเจ็กต์ในออพเจ็กต์โอเรียนเต็ดดีไซน์และออพเจ็กต์ในขอบเขตของปัญหาจริง ทำให้ระบบที่อธิบายนี้สามารถทำความเข้าใจได้ง่ายจากโมเดลที่ใช้แสดงระบบนั้น
สิ่งนี้ทำให้ดีไซน์สามารถเข้าใจได้ดีกว่าและเป็นธรรมชาติมากกว่าและทำให้การตรวจสอบระหว่างรีควายเมนท์และซอฟต์แวร์โค๊ดเป็นไปได้อย่างสะดวก
นอกจากนั้นการเปรียบเทียบกับปัญหาจริงยังทำให้คนที่ไม่ได้ร่วมอยู่ในทีมที่วิเคราะห์ออกแบบสามารถทำความเข้าใจกับโมเดลของระบบนั้นได้ง่ายขึ้นด้วย

ในเอสเอ/เอสดี
การแยกย่อยโพรเซสหนึ่งๆออกเป็นซับโพรเซสเป็นอะไรที่ตามแต่ผู้ออกแบบจะนึกคิดเอาเอง
ผู้ออกแบบคนละคนกันสามารถแยกย่อยโพรเซสให้ออกมาในแบบที่ไม่เหมือนกันได้
ในออพเจ็กต์โอเรียนเต็ดดีไซน์ การแยกย่อยจะอยู่บนพื้นฐานของออพเจ็กต์หรือวัตถุในสภาวะแวดล้อมของปัญหา
ดังนั้นผู้พัฒนาที่ต่างคนหรือต่างชุดกันก็มีแนวโน้มที่จะกำหนดออพเจ็กต์ได้เหมือนๆกัน
สิ่งนี้ทำให้เกิดการที่สามารถนำเอาส่วนประกอบจากระบบหนึ่งนำกลับมาใช้ได้อีกในอีกระบบหนึ่ง

วิธีการแบบออพเจ็กต์โอเรียนเต็ดจะสามารถรวมฐานข้อมูลกับโปรแกรมมิ่งโค๊ดได้ดีกว่า
โดยเป็นแบบที่เหมือนๆกัน ออพเจ็กต์สามารถโมเดลได้ทั้งฐานข้อมูลและโครงสร้างการโปรแกรม
การวิจัยเกี่ยวกันออพเจ็กต์โอเรียนเต็ดดาต้าเบสในอนาคตจะยิ่งช่วยให้สิ่งที่กล่าวถึงนี้เป็นไปได้ดีและชัดเจนยิ่งขึ้น
เมื่อเปรียบเทียบกับวิธีการดีไซน์แบบโพรซีเดอร์รอดแล้ว
วิธีการแบบโพรซีเดอร์รอดจะยุ่งยากมากกว่าในการจัดการกับฐานข้อมูล
มันเป็นการยากที่จะรวมโปรแกรมมิ่งโค๊ดที่ถูกจัด (organize) เพื่อฟังก์ชั่นกับฐานข้อมูลที่ถูกจัดเพื่อดาต้าข้อมูล

มีอยู่หลายเหตุผลที่ว่าทำไมวิธีการแบบดาต้าโฟลว์จึงเป็นที่นิยมใช้กันอย่างแพร่หลาย
อย่างแรกก็คือโปรแกรมเมอร์มีแนวโน้มที่จะคิดในแบบที่เป็นฟังก์ชันมากกว่า
ดังนั้นระเบียบวิธีที่มีพื้นฐานอยู่บนฟังก์ชันหรือโพรเซสอย่างดาต้าโฟลว์ไดอะแกรมจึงเรียนรู้ได้ง่ายกว่า
อีกเหตุผลหนึ่งก็เป็นไปตามประวัติศาสตร์ที่เอสเอ/เอสดีเป็นระเบียบวิธีที่เป็นทางการ (formal)
วิธีแรกที่ถูกคิดขึ้นมาเพื่อการพัฒนาซอฟต์แวร์หรือระบบ ดังนั้นจึงมีผู้ใช้วิธีการนี้อยู่เดิมแล้วมาก
แต่ในอนาคตเมื่อวิธีการแบบออพเจ็กต์โอเรียนเต็ดถูกพัฒนาให้ดีมากขึ้น
เราเชื่อว่าเทคโนโลยีแบบออพเจ็กต์โอเรียนเต็ดจะถูกนำมาใช้ในการวิเคราะห์ออกแบบและสร้างระบบอย่างแพร่หลายมากขึ้น

4.3 แจ็กสันสตรักเจอร์ดดีวีลอพเมนต์ (เจเอสดี)

แจ็กสันสตรักเจอร์ดดีวีลอพเมนต์หรือเจเอสดีนี้ก็เป็นอีกระเบียบวิธีหนึ่งที่ถูกพัฒนาขึ้นเพื่อนำมาใช้ในการวิเคราะห์และออกแบบระบบ
ซึ่งวิธีเจเอสดีนี้มีลักษณะแตกต่างออกไปจากเอสเอ/เอสดีและโอเอ็มทีมากพอสมควร
ระเบียบวิธีแบบเจเอสดีถูกพัฒนาโดย Michel Jackson และได้รับความนิยมากในประเทศแถบยุโรป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เจเอสดีจะไม่แยกขั้นตอนระหว่างการวิเคราะห์และออกแบบระบบ แต่ได้รวมทั้งสองขั้นตอนเข้าไว้ด้วยกัน
แล้วกำหนดให้เรียกขั้นตอนที่รวมกันนี้ว่าขั้นตอนการกำหนดรายละเอียดของระบบ (specification)
เจเอสดีแบ่งการพัฒนาระบบออกเป็น 2 ขั้นตอนคือ การกำหนดรายละเอียดของระบบ และจากนั้นก็เป็น
การสร้างระบบ ในขั้นตอนแรกเจเอสดีจะวิเคราะห์ว่าระบบนี้ "คืออะไร (what)"
แล้วตอนหลังจึงพิจารณาว่าจะแก้ไขปัญหาได้ "อย่างไร (how)"
เจเอสดีถูกพัฒนาขึ้นโดยเฉพาะสำหรับแอพพลิเคชั่นที่ไทม์มิ่งมีความสำคัญ

เจเอสดีใช้โมเดลที่เป็นแผนภาพเหมือนกันกับเอสเอ/เอสดี, โอเอ็มที และวิธีอื่นใช้
แต่เราขอที่จะไม่แสดงแผนภาพของเจเอสดีไว้ ณ.ที่นี้
ทั้งนี้เนื่องจากแผนภาพไม่ได้เป็นจุดที่ทำให้เราเห็นถึงข้อดีของเจเอสดีแต่อย่างใด
ในความเป็นจริงแล้วเราคิดว่าเจเอสดีมีลักษณะที่เป็นกราฟฟิคหรือแผนภาพน้อยกว่าเอสเอ/เอสดี หรือ
โอเอ็มทีเสียด้วยซ้ำไป

4.3.1 สรุปวิธีการแบบเจเอสดี

วิธีการแบบเจเอสดีเริ่มต้นด้วยการพิจารณาโลกแห่งความเป็นจริง
แม้ว่าจุดมุ่งหมายของระบบจะเพื่อการใช้ฟังก์ชันนอลลิตี้ (functionality) หรือสิ่งที่ให้การทำงานก็ตาม
แต่วิธีการของแจ็กสันคิดว่าในลำดับแรก
ผู้ที่จะวิเคราะห์และออกแบบจะต้องทำการพิจารณาว่าฟังก์ชันนอลลิตี้ดังกล่าวนั้นจะเหมาะสมกับโลกแห่งความเ
ป็นจริงอย่างไร เจเอสดีโมเดลจะอธิบายโลกแห่งความเป็นจริงในรูปแบบของเอ็นติตี้ (entity), แอ็คชั่น (action)
และลำดับของแอ็คชั่นเหล่านั้น
โดยปกติแล้วเอ็นติตี้จะปรากฏในลักษณะที่เป็นคำนามในคำบรรยายความต้องการของระบบ
และแอ็คชั่นจะปรากฏในรูปของคำกริยา การพัฒนาซอฟต์แวร์ด้วยเจเอสดีจะประกอบไปด้วยขั้นตอน 6
ขั้นตอนที่เรียงลำดับกันดังนี้ ขั้นตอนเอ็นติตี้แอ็คชั่น (entity action step), ขั้นตอนเอ็นติตี้สตรักเจอร์ (entity
structure step), ขั้นตอนอินนิเทียลโมเดล (initial model step), ขั้นตอนฟังก์ชั่น (function step),
ขั้นตอนซิสเต็มไทม์มิ่ง (system timing step) และขั้นตอนในการสร้าง (implementation step)

ระหว่างขั้นตอนเอ็นติตี้แอคชั่น,
ผู้พัฒนาจะเขียนรายการของเอ็นติตี้และแอ็คชั่นสำหรับส่วนของโลกแห่งความเป็นจริงโดยใช้จุดมุ่งหมายรวมของ
ระบบทั้งหมดเป็นแนวทางในการเลือกเอ็นติตี้และแอ็คชั่น
อินพุตสำหรับขั้นตอนเอ็นติตี้แอ็คชั่นคือคำบรรยายความต้องการของระบบ
และเอาร์พุตจะเป็นรายการของเอ็นติตี้และแอ็คชั่น

ในบทความของแจ็กสัน [Jackson-83] ได้แสดงตัวอย่างไว้หลายตัวอย่าง
ซึ่งตัวอย่างหนึ่งก็เป็นการออกแบบระบบควบคุมลิฟท์
เราจะอ้างอิงกับตัวอย่างระบบลิฟท์ของแจ็กสันในบทนี้ด้วย ระบบควบคุมลิฟท์นี้จะควบคุมลิฟท์ 2
ตัวที่คอยให้บริการชั้น 6 ชั้น ในลิฟท์แต่ละตัวจะมีปุ่มอยู่ 6 ปุ่มซึ่งแต่ละปุ่มจะสำหรับชั้นแต่ละชั้น
ที่แต่ละชั้นก็จะมีปุ่มให้กดขึ้นและลงในบริเวณที่คอยลิฟท์ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แจ็กสันกำหนดเอ็นติตี้ขึ้นมาสองเอ็นติตี้สำหรับตัวอย่างการควบคุมลิฟท์นี้ -- ปุ่ม และ ลิฟท์
เขายังกำหนดแอ็คชั่นขึ้นอีกสามอันคือ กดปุ่ม, ลิฟท์ถึงชั้นที่ N และลิฟท์ออกจากชั้นที่ N

แอ็คชั่นจะเป็นสิ่งเกิดขึ้นในโลกแห่งความเป็นจริงและไม่ได้เป็นสิ่งที่มนุษย์สร้างขึ้นเอง (artifact)
แอ็คชั่นจะเกิดขึ้น ณ.จุดหนึ่งของเวลาและเป็นอะตอมมิก (atomic) ซึ่งก็คือไม่สามารถแบ่งแยกแอ็คชั่นต่อไปได้
ขั้นตอนเอ็นติตี้สตรักเจอร์จะทำการจัดลำดับของแอ็คชั่นบางส่วนโดยใช้เวลาเป็นตัวจัดลำดับ
ระบบควบคุมลิฟท์เป็นตัวอย่างที่แสดงให้เห็นความสำคัญของการจัดลำดับแอ็คชั่น
มันยอมรับได้สำหรับลิฟท์ที่จะมาถึงชั้น 3 แล้วออกจากชั้นที่ 3, มาถึงชั้นที่ 2 แล้วออกจากชั้นที่ 2 และอื่นๆ
แต่มันจะเป็นไปไม่ได้ที่จะเกิดแอ็คชั่นแบบที่ลิฟท์มาถึงชั้นที่ N สองครั้งติดต่อกัน

ขั้นตอนอินนิเทียลโมเดลจะกล่าวถึงการเชื่อมต่อระหว่างโลกแห่งความเป็นจริงและแบบจำลองว่าเชื่อมต่อกันอย่างไร เจเอสดีสนับสนุนการเชื่อมต่อกันทั้งแบบเสตทเวคเตอร์ (state-vector connection) และดาต้าสตรีม (data stream connection) ซึ่งจะได้อธิบายในย่อหน้าต่อไป

ระบบการควบคุมลิฟท์แสดงให้เห็นการเชื่อมต่อกันแบบสเตทเวคเตอร์
อะไรจะเกิดขึ้นเมื่อมีบางคนกดปุ่มขึ้น 5 ครั้งติดต่อกัน?
ผู้ใช้ลิฟท์ไม่ได้ต้องการให้ระบบควบคุมลิฟท์จำการกดแต่ละครั้งแล้วส่งลิฟท์มา 5 ครั้งตามการกดนั้น
แต่แทนที่จะเป็นเช่นนั้น การกดปุ่มขึ้นจะทำการตั้งแฟล็กการขึ้น (up-flag) ให้เป็นจริง (True)
การกดปุ่มขึ้นหลายๆครั้งจะไม่มีผลกระทบอื่นใด
ระบบคอมพิวเตอร์ของโมเดลเจเอสดีจะไม่รับรู้ถึงจำนวนการกดปุ่ม
แต่จะติดต่อกับโลกแห่งความเป็นจริงโดยผ่านแฟล็กการขึ้นเท่านั้น
แจ็กสันเรียกแฟล็กการขึ้นนี้ว่าเป็นการติดต่อแบบเสตทเวคเตอร์

พริ้นท์บัฟเฟอร์ของคอมพิวเตอร์ (computer print buffer) จะแสดงให้เห็นการเชื่อมต่อแบบดาต้าสตรีม
ผู้ใช้คอมพิวเตอร์ไม่ต้องการให้เกิดการสูญหายของข้อมูลขึ้นถ้าคอมพิวเตอร์สามารถส่งข้อมูลได้เร็วกว่าที่เครื่องพิมพ์จะสามารถพิมพ์ได้
พริ้นท์บัฟเฟอร์จะเป็นสิ่งที่กั้นระหว่างเครื่องคอมพิวเตอร์กับเครื่องพิมพ์และทำให้การประมวลผลของซีพียูกับการพิมพ์สามารถซ้อนเหลื่อมกันได้ พริ้นท์บัฟเฟอร์จะมีขนาดที่จำกัดและเมื่อบัฟเฟอร์เต็ม
คอมพิวเตอร์จะต้องรอก่อนที่จะส่งข้อมูลต่อไปได้อีก
การติดต่อแบบดาต้าสตรีมของเจเอสดีก็จะเป็นในลักษณะเดียวกันกับบัฟเฟอร์ที่มีขนาดจำกัด
ขั้นตอนอินนิเทียลของเจเอสดีจะไม่ได้กล่าวถึงข้อจำกัดทางกายภาพของบัฟเฟอร์เอง

ขั้นตอนฟังก์ชั่นจะใช้ซูโดโค้ด (pseudocode) ในการอธิบายผลลัพธ์หรือเอาร์พุตของแอ็คชั่น
ในตอนท้ายของขั้นตอนนี้ผู้พัฒนาจะมีสเปคซิฟิเคชั่นที่สมบูรณ์ของระบบ ในตัวอย่างระบบลิฟท์,
การเปิดปิดแสดงที่แสดงว่าลิฟท์ได้มาถึงแต่ละชั้นจะเป็นฟังก์ชันที่จะต้องกำหนด

ขั้นตอนชิสเต็มไทม์มิ่งจะพิจารณาว่าจะอนุญาตให้โมเดลห่อหุ้มโลกแห่งความเป็นจริงได้มากน้อยเพียงใด สำหรับส่วนใหญ่แล้วผลลัพธ์ของขั้นตอนไทม์มิ่งจะเป็นกลุ่มของบันทึกที่ไม่เป็นทางการ (informal note)
ในเรื่องเกี่ยวกับข้อจำกัดทางประสิทธิภาพของระบบ (performance constraints)
เป็นต้นว่าระบบควบคุมลิฟท์ต้องตรวจจับว่าเมื่อใดปุ่มถูกกดลงและปล่อยขึ้น
ระยะเวลายาวนานเท่าใดที่ผู้ใช้กดปุ่มให้หน้าสัมผัสต่อกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มันจะน่ารำคาญถ้ากดปุ่มไปแล้วระบบกลับไม่มีการตอบสนองแต่อย่างใดดังนั้นเราจะต้องกำหนดเรื่องพวกนี้ด้วย
ระยะเวลาที่กำหนดไว้สั้นจะหมายความว่าระบบควบคุมมักจะตรวจพบเซอร์วิสรีเควิส (service request)
อย่างไรก็ตามถ้าการกดปุ่มถูกตรวจจับโดยใช้วิธีการแบบพูลลิ่ง (polling scheme)
ค่าที่ต่ำจะหมายถึงว่าต้องใช้ทรัพยากรของเครื่องคอมพิวเตอร์มากขึ้นในการตรวจสอบ
ผู้ออกแบบจะต้องหาจุดสมดุลย์ของประสิทธิภาร (performance trade-offs)
อย่างชัดเจนในระหว่างขั้นตอนซิสเต็มไทมมิ่งนี้

ขั้นตอนการสร้าง หรืออิมพลีเมนต์เทชั่นจะมุ่งความสนใจในปัญหาของการจัดกระบวนการ (process scheduling) และการกำหนดตัวประมวลผลสำหรับแต่ละกระบวนการ (allocates processors to processes)
จำนวนของโพรเซสอาจจะไม่เท่ากันกับจำนวนของตัวประมวลผลก็ได้
โมเดลในการควบคุมลิฟท์ของแจ็คสันจะประกอบไปด้วยกระบวนการ 50 กระบวนการ
ผู้พัฒนาจะต้องตัดสินใจว่าจะให้แต่ละกระบวนการถูกทำโดยแต่ละซีพียูซึ่งจะต้องมีทั้งหมดถึง 50 ตัว
หรือจะใช้ตัวประมวลผลร่วมกันระหว่างกระบวนการหลายๆกระบวนการ
หลังจากผ่านขั้นตอนทั้งหกของเจเอสดีก็จะมาถึงการเขียนโค๊ดและการออกแบบฐานข้อมูล

4.3.2 การเปรียบเทียบกับโอเอ็มที

ผู้เขียนบางคนกล่าวถึงเจเอสดีว่าเป็นออพเจ็กต์โอเรียนเต็ดซึ่งเราไม่เห็นด้วย
แม้ว่าเจเอสดีจะเริ่มต้นด้วยการพิจารณาโลกแห่งความเป็นจริงซึ่งในแง่นี้เป็นเหมือนแนวความคิดแบบออพเจ็กต์โอเรียนเต็ด แต่อย่างไรก็ตามแจ็กสันกำหนดเอ็นติตี้ (ออพเจ็กต์) และแสดงโครงสร้างของมันน้อยเกินไป
แต่ละตัวอย่างจากสามตัวอย่างที่แจ็กสันนำเสนอ มีเอ็นติตี้เพียงสองหรือสามเอ็นติตี้เท่านั้น
เราเชื่อว่าโมเดลแบบออพเจ็กต์โอเรียนเต็ดจะต้องมีการประกอบกันของโครงสร้างข้อมูลและความสัมพันธ์ระหว่างกันอย่างมากๆ (rich mixture of data structure and relationships)

เราพบว่าวิธีการของแจ็กสันมีความซับซ้อนมากเกินไปและเป็นการยากที่จะสามารถเข้าใจได้ทั้งหมด
พวกเราคิดว่าเจเอสดีมีลักษณะที่คลุมเครือมากกว่าวิธีการแบบคาต้าโฟลว์และออพเจ็กต์โอเรียนเต็ด
สาเหตุหนึ่งที่ทำให้เจเอสดีมีความซับซ้อนก็เนื่องจากว่าเจเอสดีต้องพึ่งการใช้ซูโดโค๊ดมากนั่นเอง
ทั้งที่โมเดลที่เป็นแผนภาพจะสามารถทำให้เข้าใจได้ง่ายกว่า
นอกจากนี้เจเอสดียังซับซ้อนเพราะว่ามันถูกออกแบบมาเพื่อจัดการกับปัญหาแบบเรียลไทม์ (real time)
ซึ่งเป็นปัญหาที่ยากนั่นเอง สำหรับปัญหาประเภทดังกล่าว
เจเอสดีอาจจะทำให้ได้ผลการออกแบบที่ดีกว่าและคุ้มค่ากับความซับซ้อนของโมเดลที่เพิ่มขึ้น
อย่างไรก็ตามความซับซ้อนของเจเอสดีไม่มีความจำเป็นหรือมากเกินไปสำหรับปัญหาทั่วๆไปหรือปัญหาที่ง่ายกว่าปัญหาแบบเรียลไทม์

วิธีการของแจ็กสันจะเน้นที่แอ็คชั่นมากกว่าที่แอททริบิวส์อย่างที่วิธีของโอเอ็มทีที
แอ็คชั่นของเจเอสดีบางอันดูเหมือนแอ็คโซซิเอชั่นของโอเอ็มที ตัวอย่างเช่น พนักงานจัดผลิตภัณฑ์ตามคำสั่งซื้อ (a clerk allocates product to an order) ตามวิธีโอเอ็มทีจะเรียก "จัด" (allocate) ว่าเป็นแอ็คโซซิเอชั่น
แต่แจ็กสันเรียกมันว่าเป็นแอ็คชั่น แจ็กสันคิดว่าแอททริบิวส์น่าสับสนและพยายามหลีกเลี่ยงมัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แอ็คชั่นมีบทบาทอย่างมากในการโมเดลลิ่งแบบเจเอสดีจนมันไม่เน้นถึงแอททริบิวส์
ซึ่งก็เหมือนกันกับที่แอททริบิวส์มีบทบาทมากกว่าในออพเจ็กต์โมเดลของโอเอ็มที

เจเอสดีจะเป็นระเบียบวิธีการที่มีประโยชน์สำหรับแอพพลิเคชั่นประเภทต่อไปนี้

- คอนเคอร์เรนท์ซอฟแวร์ (Concurrent software) ซึ่งโพรเซสจะต้องซิงโครไนซ์ (synchronize) ระหว่างกัน
- เรียลไทม์ซอฟแวร์ (Real time software)
  เจเอสดีโมเดลลิ่งจะบอกรายละเอียดได้อย่างดีและมุ่งความสนใจกับเวลาได้ดี
- ไมโครโค๊ด (Microcode) เจเอสดีจะมีรายละเอียดครอบคลุม,
  ไม่มีการตั้งสมมุติฐานเกี่ยวกับการมีระบบปฏิบัติการและพิจารณาถึงการคอนเคอร์เรนท์กระบวนการและไทม์มิ่ง
- การโปรแกรมคอมพิวเตอร์แบบขนาน (Programming parallel computer)
  รูปแบบการมีหลายกระบวนการของเจเอสดีจะช่วยในการออกแบบแอพพลิเคชั่นแบบนี้มาก

เจเอสดีจะไม่เหมาะสำหรับแอพพลิเคชั่นต่อไปนี้

- แอพพลิเคชั่นที่ต้องมีการวิเคราะห์ระดับสูง (High level analysis)
  เนื่องจากเจเอสดีไม่สนับสนุนการทำความเข้าใจในปัญหาอย่างกว้างครอบคลุมทั้งหมด
  เจเอสดีจะเป็นวิธีที่ไม่มีประสิทธิภาพดีพอในการทำแอปสแตรกชั่นและการทำซิมพลิฟิเคชั่น (abstraction and simplification)
  เจเอสดีจัดการรายละเอียดอย่างพิถีพิถันแต่ไม่ได้ช่วยให้ผู้พัฒนาเข้าใจถึงแก่นแท้ของปัญหา
- ฐานข้อมูล การออกแบบฐานข้อมูลจะเป็นเรื่องที่ยากถ้าใช้วิธีการของแจ็กสัน
  โมเดลแบบเจเอสดีเน้นที่แอ็คชั่นกว่าที่เอ็นติตี้และแอททริบิวท์มากจนเกินไป
  จึงเป็นเหตุให้มันเป็นวิธีการที่แย่ในการออกแบบฐานข้อมูล
- ซอฟต์แวร์ทั่วไปที่วิ่งอยู่ภายใต้ระบบปฏิบัติการตัวหนึ่งๆ
  เนื่องจากเจเอสดีจะมองระบบในแบบที่เป็นกระบวนการนับร้อยนับพันซึ่งที่ให้เกิดความสับสนและบางส่วนก็ไม่จำเป็น

4.4 สรุป

วิธีการทางวิศวกรรมซอฟต์แวร์ที่เป็นที่นิยมหลายวิธีการต่างก็มีพื้นฐานมาจากแนวคิดเกี่ยวกับการไหลของข้อมูลหรือดาต้าโฟลว์
และระเบียบวิธีการแบบเอสเอ/เอสดีก็เป็นตัวแทนของวิธีการที่ใช้แนวความคิดของดาต้าโฟลว์
เอสเอ/เอสดีเริ่มต้นด้วยกระบวนการหรือฟังก์ชันเพียงอันเดียวที่แสดงถึงเป้าหมายของระบบทั้งหมดที่ต้องการ
หลังจากนั้นเอสเอ/เอสดีก็จะเริ่มวนซ้ำเพื่อแบ่งกระบวนการที่ซับซ้อนนั้นออก
จนกระทั่งได้ฟังก์ชันที่มีขนาดเล็กและสามารถนำมาสร้างได้อย่างมีประสิทธิภาพ

เอสเอ/เอสดีกับโอเอ็มทีมีอะไรหลายอย่างที่เหมือนกัน
ซึ่งทั้งสองระเบียบวิธีการสนับสนุนมุมมองที่สมมาตรทั้งสามด้านของระบบ -- ออพเจ็กต์, ไดนามิกส์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และฟังก์ชันนอลโมเดล ความแตกต่างของมันอยู่ที่ว่าเอสเอ/เอสดีเน้นที่ฟังก์ชั่นนอลโมเดล
ในขณะที่โอเอ็มทีเน้นที่ออพเจ็กต์โมเดล พวกเราเชื่อว่าสำหรับปัญหาส่วนใหญ่แล้ว
วิธีการแบบออพเจ็กต์โอเรียนเต็ดจะเหมาะสมกว่าวิธีการแบบคาต้าโฟลว์
ผลการออกแบบด้วยวิธีออพเจ็กต์โอเรียนเต็ดจะสามารถเพิ่มขยายและสามารถตรวจสอบได้ง่ายกว่า
และยังสามารถรวมฐานข้อมูลกับโค๊ดที่โปรแกรมได้ดีกว่าด้วย

เจเอสดีโมเดลเริ่มต้นด้วยการพิจารณาโลกแห่งความเป็นจริง
จากนั้นผู้พัฒนาก็ทำการเขียนเอ็นติตี้และแอ็คชั่นที่สำคัญในโลกแห่งความเป็นจริงโดยใช้มุมมองจากตัวแอพพลิเคชั่น
ขั้นตอนต่อๆไปที่เหลือของเจเอสดีก็จะเป็นการเพิ่มเติมรายละเอียดของซูโดโค๊ดเพื่อที่จะสามารถกำหนดพฤติกรรมของตัวซอฟต์แวร์ที่ต้องการให้ตรงและสามารถตอบสนองกับโลกแห่งความเป็นจริงได้ดียิ่งขึ้น

เจเอสดีเป็นวิธีการหนึ่งที่ดี ดังเช่น เอสเอ/เอสดี และ โอเอ็มที
แต่ละระเบียบวิธีต่างก็มีประเภทของงานที่มันสามารถแสดงเป็นโมเดลได้ดีไม่เหมือนกัน
เจเอสดีจะเป็นวิธีการที่ดีมากสำหรับแอพพลิเคชั่นที่เป็นเรียลไทม์หรือเกี่ยวกับไมโครโค๊ด
แต่เจเอสดีจะไม่เหมาะสำหรับการวิเคราะห์ในระดับสูงและเกี่ยวกับฐานข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

ระบบพยากรณ์ดวงชะตา

5.1 กล่าวนำ

ระบบพยากรณ์ดวงชะตานี้เป็นอีกตัวอย่างหนึ่งที่เราได้นำวิธีการวิเคราะห์และออกแบบโดยวิธีโอเอ็มทีมาประยุกต์ใช้ ทั้งนี้ก็เพื่อการศึกษาวิธีการของโอเอ็มทีว่าสามารถนำมาประยุกต์ใช้กับระบบอื่นๆที่นอกเหนือจากระบบการลงทะเบียนที่ได้ยกตัวอย่างไว้แล้ว ได้อย่างมีประสิทธิภาพมากน้อยแค่ไหน การที่เราเลือกระบบพยากรณ์ดวงชะตามาเป็นตัวอย่างของปัญหาก็เพราะระบบนี้ตรงกับโครงงานของเราคือการออกแบบพัฒนาระบบไคลเอ็นท์เซชอร์ฟเวอร์ และเราตั้งใจที่จะสร้างระบบที่จะนำไปร่วมในงานลาดกระบังนิทรรศน์ในครั้งต่อไปนี้ด้วย ก่อนอื่นเราจะได้ทำความเข้าใจถึงระบบที่เรากำลังจะออกแบบอยู่ ว่ามีลักษณะเป็นอย่างไรดังนิยามความต้องการ (Requirement definition) ดังต่อไปนี้

ระบบการพยากรณ์นี้จะเป็นระบบที่ถูกใช้ในการทำนายดวงชะตาโดยมีพนักงานที่รับข้อมูลของวันเดือนปีเกิดของผู้ที่ต้องการให้ทำนายดวงชะตามาป้อนเข้าเครื่องคอมพิวเตอร์ โดยที่วันเดือนปีที่รับมานี้จะอยู่ในรูปของชื่อวัน (จันทร์, อังคาร, พุธ,...) และเดือนไทย (อ้าย, ยี่, สาม,...) และปี (ชวด, ฉลู, ขาล,...) หลังจากนั้นแล้วก็จะถามผู้ที่ต้องการให้ทำนายดวงชะตาว่าจะต้องการทราบเรื่องที่เกี่ยวกับอะไรซึ่งได้มีการกำหนดไว้เลือก เมื่อทำการเลือกเรื่องที่ต้องการให้ทำนายแล้ว เครื่องคอมพิวเตอร์ที่เป็นแทอร์มินัลนั้นก็จะทำการทำนายดวงชะตาโดยอาศัยข้อมูลจากฐานข้อมูล ผลของการทำนายจะแสดงขึ้นสู่จอภาพของเครื่องคอมพิวเตอร์และยังสามารถพิมพ์ออกทางเครื่องพิมพ์ได้ด้วย ในระบบนี้เราจะใช้เครื่องคอมพิวเตอร์ที่ทำหน้าที่เป็นที่ป้อน-รับข้อมูลในการทำนายหลายเครื่องติดต่ออยู่กับเครื่องที่เป็นตัวกลางที่ควบคุมฐานข้อมูลรวมหนึ่งเครื่อง และวิธีการพยากรณ์นี้จะใช้วิธีการผูกดวงแบบตัวเลข ๗ ตัวตามตำราพรหมชาติซึ่งได้ถูกอธิบายในหัวข้อต่อไป

5.2 วิธีการผูกดวงแบบตัวเลข ๗ ตัว

การผูกดวงแบบตัวเลข ๗ ตัวอาศัยหลักในการตั้งฐานจากเลขของวัน-เดือน-ปีเกิดของคนที่เราจะทำนายเขา โดยที่ใช้ตัวเลขวันเกิดมาตั้งเป็นฐานที่ ๑ เดือนเกิดมาตั้งเป็นฐานที่ ๒ และปีเกิดมาตั้งเป็นฐานที่ ๓ ในฐานที่ ๑ ซึ่งเป็นฐานของวันจะประกอบไปด้วยฐานทั้ง ๗ ที่มีชื่อว่า อัตตะ, หินะ, ธะนัง, ปีตา, มาตา, โภคา และมัชฌิมา ตามลำดับ ในฐานที่ ๒ ซึ่งเป็นฐานของเดือนจะประกอบไปด้วยฐานทั้ง ๗ ที่มีชื่อว่า ตนุ, กฎุมภะ, สุหัชชะ, พันธุ, ปุตตะ, หริ และปัตนิ ตามลำดับ ในฐานที่ ๓ ซึ่งเป็นฐานของปีจะประกอบไปด้วยฐานทั้ง ๗ ที่มีชื่อว่า มรณะ, สุภะ, กัมมะ, ลาภะ, พยายะ, ทาสี และทาสา ตามลำดับ

เมื่อได้วัน-เดือน-ปีเกิดมาแล้วผู้ทำนายก็จะตั้งทำการตั้งฐานทั้งสาม ซึ่งมีวิธีการดังต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในฐานที่ ๑ ฐานวันเกิด ตามตำราให้นับวันอาทิตย์เป็นเลข ๑ วันจันทร์เป็นเลข ๒ วันอังคารเป็นเลข ๓ วันพุธเป็นเลข ๔ วันพฤหัสบดีเป็นเลข ๕ วันศุกร์เป็นเลข ๖ วันเสาร์เป็นเลข ๗ ซึ่งผู้ใดเกิดวันอะไรก็ให้เอาเลขของวันนั้นมาตั้งเป็นอันดับแรก และเรียงลำดับต่อกันไปจนครบ ๗ วัน ดังตัวอย่างต่อไปนี้

	อัตตะ	หินะ	ธะนัง	ปิตา	มาตา	โภคา	มัชฌิมา
ผู้ที่เกิดวันอาทิตย์	๑	๒	๓	๔	๕	๖	๗
ผู้ที่เกิดวันจันทร์	๒	๓	๔	๕	๖	๗	๑
ผู้ที่เกิดวันอังคาร	๓	๔	๕	๖	๗	๑	๒
ผู้ที่เกิดวันพุธ	๔	๕	๖	๗	๑	๒	๓
ผู้ที่เกิดวันพฤหัสฯ	๕	๖	๗	๑	๒	๓	๔
ผู้ที่เกิดวันศุกร์	๖	๗	๑	๒	๓	๔	๕
ผู้ที่เกิดวันเสาร์	๗	๑	๒	๓	๔	๕	๖

ในฐานที่ ๒ ฐานเดือนเกิด ตามตำราให้นับเดือนอ้ายเป็นเลข ๑ เดือนยี่เป็นเลข ๒ เดือนสามเป็นเลข ๓ เดือนสี่เป็นเลข ๔ เดือนห้าเป็นเลข ๕ เดือนหกเป็นเลข ๖ เดือนเจ็ดเป็นเลข ๗ แต่เก้าเกิดเดือนแปดก็ให้เป็นเลข ๑ เดือนเก้าให้เป็นเลข ๒ เดือนสิบให้เป็นเลข ๓ เดือนสิบเอ็ดให้เป็นเลข ๔ เดือนสิบสองให้เป็นเลข ๕ ซึ่งผู้ใดเกิดเดือนอะไรก็ให้เอาเลขของเดือนนั้นมาตั้งเป็นอันดับแรก และเรียงลำดับต่อกันไปจนครบ ๗ ตัวฐาน ดังตัวอย่างต่อไปนี้

	ตนุ	กฏุมภะ	สุหัชชะ	พันธุ	ปุตตะ	หริ	ปัตนิ
ผู้ที่เกิดเดือนอ้าย	๑	๒	๓	๔	๕	๖	๗
ผู้ที่เกิดเดือนยี่	๒	๓	๔	๕	๖	๗	๑
ผู้ที่เกิดเดือนสาม	๓	๔	๕	๖	๗	๑	๒
ผู้ที่เกิดเดือนสี่	๔	๕	๖	๗	๑	๒	๓
ผู้ที่เกิดเดือนห้า	๕	๖	๗	๑	๒	๓	๔
ผู้ที่เกิดเดือนหก	๖	๗	๑	๒	๓	๔	๕
ผู้ที่เกิดเดือนเจ็ด	๗	๑	๒	๓	๔	๕	๖
ผู้ที่เกิดเดือนแปด	๑	๒	๓	๔	๕	๖	๗
ผู้ที่เกิดเดือนเก้า	๒	๓	๔	๕	๖	๗	๑
ผู้ที่เกิดเดือนสิบ	๓	๔	๕	๖	๗	๑	๒
ผู้ที่เกิดเดือนสิบเอ็ด	๔	๕	๖	๗	๑	๒	๓
ผู้ที่เกิดเดือนสิบสอง	๕	๖	๗	๑	๒	๓	๔

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในฐานที่ ๑ ฐานวันเกิด ตามตำราให้นับวันอาทิตย์เป็นเลข ๑ วันจันทร์เป็นเลข ๒ วันอังคารเป็นเลข ๓ วันพุธเป็นเลข ๔ วันพฤหัสบดีเป็นเลข ๕ วันศุกร์เป็นเลข ๖ วันเสาร์เป็นเลข ๗ ซึ่งผู้ใดเกิดวันอะไรก็ให้เอาเลขของวันนั้นมาตั้งเป็นอันดับแรก และเรียงลำดับต่อกันไปจนครบ ๗ วัน ดังตัวอย่างต่อไปนี้

	อัตตะ	หินะ	ธะนัง	ปีตา	มาตา	โภคา	มัชฌิมา
ผู้ที่เกิดวันอาทิตย์	๑	๒	๓	๔	๕	๖	๗
ผู้ที่เกิดวันจันทร์	๒	๓	๔	๕	๖	๗	๑
ผู้ที่เกิดวันอังคาร	๓	๔	๕	๖	๗	๑	๒
ผู้ที่เกิดวันพุธ	๔	๕	๖	๗	๑	๒	๓
ผู้ที่เกิดวันพฤหัสฯ	๕	๖	๗	๑	๒	๓	๔
ผู้ที่เกิดวันศุกร์	๖	๗	๑	๒	๓	๔	๕
ผู้ที่เกิดวันเสาร์	๗	๑	๒	๓	๔	๕	๖

ในฐานที่ ๒ ฐานเดือนเกิด ตามตำราให้นับเดือนอ้ายเป็นเลข ๑ เดือนยี่เป็นเลข ๒ เดือนสามเป็นเลข ๓ เดือนสี่เป็นเลข ๔ เดือนห้าเป็นเลข ๕ เดือนหกเป็นเลข ๖ เดือนเจ็ดเป็นเลข ๗ แต่ถ้าเกิดเดือนแปดก็ให้เป็นเลข ๑ เดือนเก้าให้เป็นเลข ๒ เดือนสิบให้เป็นเลข ๓ เดือนสิบเอ็ดให้เป็นเลข ๔ เดือนสิบสองให้เป็นเลข ๕ ซึ่งผู้ใดเกิดเดือนอะไรก็ให้เอาเลขของเดือนนั้นมาตั้งเป็นอันดับแรก และเรียงลำดับต่อกันไปจนครบ ๗ ตัวฐาน ดังตัวอย่างต่อไปนี้

	ตนุ	กฎุมภะ	สุหัชชะ	พันธุ	ปุตตะ	หริ	ปัตนิ
ผู้ที่เกิดเดือนอ้าย	๑	๒	๓	๔	๕	๖	๗
ผู้ที่เกิดเดือนยี่	๒	๓	๔	๕	๖	๗	๑
ผู้ที่เกิดเดือนสาม	๓	๔	๕	๖	๗	๑	๒
ผู้ที่เกิดเดือนสี่	๔	๕	๖	๗	๑	๒	๓
ผู้ที่เกิดเดือนห้า	๕	๖	๗	๑	๒	๓	๔
ผู้ที่เกิดเดือนหก	๖	๗	๑	๒	๓	๔	๕
ผู้ที่เกิดเดือนเจ็ด	๗	๑	๒	๓	๔	๕	๖
ผู้ที่เกิดเดือนแปด	๑	๒	๓	๔	๕	๖	๗
ผู้ที่เกิดเดือนเก้า	๒	๓	๔	๕	๖	๗	๑
ผู้ที่เกิดเดือนสิบ	๓	๔	๕	๖	๗	๑	๒
ผู้ที่เกิดเดือนสิบเอ็ด	๔	๕	๖	๗	๑	๒	๓
ผู้ที่เกิดเดือนสิบสอง	๕	๖	๗	๑	๒	๓	๔

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเราสามารถตั้งฐานทั้งสามขึ้นเป็นตารางได้แล้วเช่นตัวอย่างนี้ ต่อไปเราก็จะใช้ตารางนี้ในการทำนายดวงชะตา สำหรับการทำนายดวงชะตาจากตารางที่ได้ตั้งขึ้นนี้ ก่อนอื่นเราจะต้องเข้าใจความหมายของฐานแต่ละฐานก่อนว่าแต่ละฐานนั้นใช้ในการทำนายเกี่ยวกับเรื่องใด ความหมายของฐานต่างสามารถอธิบายได้ดังนี้

<u>ฐานวัน</u>

อัตตะ หมายถึง ตนเอง, ตัวเอง

หินะ หมายถึง ความต่ำช้า, ความชั่ว เลวทราม มีแต่ทางเสีย

ธะนัง หมายถึง ทรัพย์สิน เงินทอง รายได้ที่ผู้นั้นพึงมีพึงได้รับ

ปิตา หมายถึง พ่อ, บิดา ที่ให้กำเนิดมา อาจหมายรวมถึงพ่อเลี้ยงด้วย

มาตา หมายถึง แม่, มารดา ที่ให้กำเนิดมา อาจหมายรวมถึงผู้อุปการะเลี้ยงดูมาแต่เล็กๆด้วย

โภคา หมายถึง สมบัติพัสถาน ที่ดิน เรือกสวนไร่นา หรือกองมรดก ชี้ถึงฐานะบรรพบุรุษที่พอจะมียกให้ หรือเหลือเป็นมรดกตกทอดมาถึงจนหรือไม่ มากน้องเพียงไร

มัชฌิมา หมายถึง ลักษณะปานกลางไม่ดีนักแต่ก็ไม่เลวจนเกินไป

<u>ฐานเดือน</u>

ตนุ หมายถึง ตน ตัวของตัวเอง (เหมือนคำว่า “อัตตะ” ของฐานวันเกิด)

กฎุมภะ หมายถึง ทรัพย์สิน เงินทอง (เหมือนคำว่า “ธนัง” ของฐานวันเกิด)

สหัชชะ หมายถึง เพื่อนฝูง มิตรสหาย หมายรวมถึงคนที่สามารถช่วยเหลือในยามทุกข์ยากด้วย และอาจจะหมายความถึงผู้อุปถัมภ์ค้ำชู ให้ความอนุเคราะห์เวลาขัดสนด้วย

พันธุ หมายถึง ญาติพี่น้อง เครือญาติทั้งฝ่ายบิดาและมารดาให้หมายรวมถึงเครือญาติที่นับรวม ๗ ชั่วโคตร มี ทวด ปู่ย่า ตายาย พ่อแม่ ลุงป้าน้าอา ลูก หลาน ด้วย

ปุตตะ หมายถึง บุตร ลูก รวมทั้งลูกชาย หรือลูกสาว ให้รวมทั้งหลายๆด้วย

หริ หมายถึง ศัตรู ข้าศึก ให้หมายรวมถึง อุปสรรค สิ่งขัดขวางความสำเร็จและความผิดหวังนานาประการด้วย (คำว่า “หริ” บางตำราว่า “อริ” ซึ่งเหมือนกัน)

ปัตนิ หมายถึง คู่ครอง เนื้อคู่ คนที่เรารัก เพศตรงข้ามที่เราสนใจเขาหรือเขามาสนใจเรา หรือคนที่เราปรารถนาจะแต่งงานด้วย

<u>ฐานปี</u>

มรณะ หมายถึง ความตาย ความหมดหวังในชีวิต การหย่าร้าง ล้มละลายด้วย

สุภะ หมายถึง สิ่งที่ดีงาม น่ารัก พร้อมด้วยเกียรติยศ ชื่อเสียงเด่นในวงสังคม

กัมมะ หมายถึง การงาน ความขยันหมั่นเพียร หน้าที่ราชการ ตำแหน่งยศฐาน์บรรดาศักดิ์

ลาภะ หมายถึง โชคลาภ วาสนา ผลกำไร การลงทุน การติดต่อธุรกิจ การสอบได้ เสี่ยงโชค

พยายะ หมายถึง ความเจ็บไข้ได้ป่วย สุขภาพร่างการ โรคภัยพยาธิ ให้หมายรวมถึงอุปสรรคที่จะมาขัดขวางให้ต้องผิดหวังในวิถีชีวิตอีกด้วย (แต่มีความหมายอ่อนกว่า “หริ” ในฐานเดือน)

ทาสี หมายถึง หญิงข้าทาส หญิงคนใช้ ที่นำมาไว้ในบ้า หรือให้หมายถึงความสัมพันธ์ของเลขฐานวัน-เดือน ที่มาเป็นเลขเดียวกัน ทำให้สิ่งที่น่าจะดีกลับด้วยไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทาสา หมายถึง ข้าทาส บริวาร คนใช้ชาย หรือคนที่มาอาศัยอยู่ภายในบ้านที่เป็นเพศชาย

เวลาที่เราจะทำนายเรื่องหนึ่งเรื่องของฐานใดเราก็จะต้องจากตารางว่าเลขที่ฐานนั้นเป็นเลขใด และฐานที่จะทำนายนั้นไปสัมพันธ์กับฐานอื่น (คูได้จากการที่มีเลขของฐานเท่ากัน) อย่างไร

ตัวอย่างการทำนายดวงชะตาของคนเกิดวันอาทิตย์ เดือนยี่ ปีขาล ที่มีตารางเลข ๗ ตัว ต่อไปนี้

ฐานที่ ๑ ฐานวันเกิด	๑	๒	๓	๔	๕	๖	๗
ฐานที่ ๒ ฐานเดือนเกิด	๒	๓	๔	๕	๖	๗	๑
ฐานที่ ๓ ฐานปีเกิด	๓	๔	๕	๖	๗	๑	๒

การทำนาย เบื้องแรก ให้คูเลขฐานวัน ฐานเดือน และฐานปี ว่าตัวเลขที่เหมือนกันนั้นอยู่ในช่องความหมายของคำอะไรบ้าง ในที่นี้ เลข ๑ ฐานวันตก "อัตตะ" เลขที่ ๑ ฐานเดือนตก "ปัตนี" และเลขที่ ๑ ฐานปีตก "ทาสี" ทำนายได้ว่า ผู้นั้นหลักฐานฝ่ายตนดีมีเมียไม่ดูดีจะลำบากเพียงทาสถึงร่ำรวยก็ลำบากกาย อย่างนี้เป็นต้น

คูเลข ๒ ฐานวันตก "หินะ" เลข ๒ ฐานเดือนตก "คนุ" เลข ๒ ฐานปีตก "ทาสา" ทำนายได้ว่าผู้นั้นมีนิสัยนักเลง คบคนต่ำกว่าจน ว่านอนสอนยากหัวดื้อ คนในบ้านจะไม่มีใครเขารักนับถือตนเลย เพราะตนประพฤติไม่ดีนั่นเอง อย่างนี้เป็นต้น

ต่อไปก็คูเลข ๓ จากฐาน ๓ และเลข ๔,๕,๖,๗ จากทั้งสามฐาน ก็จะทำนายได้ลึกซึ้งต่อ

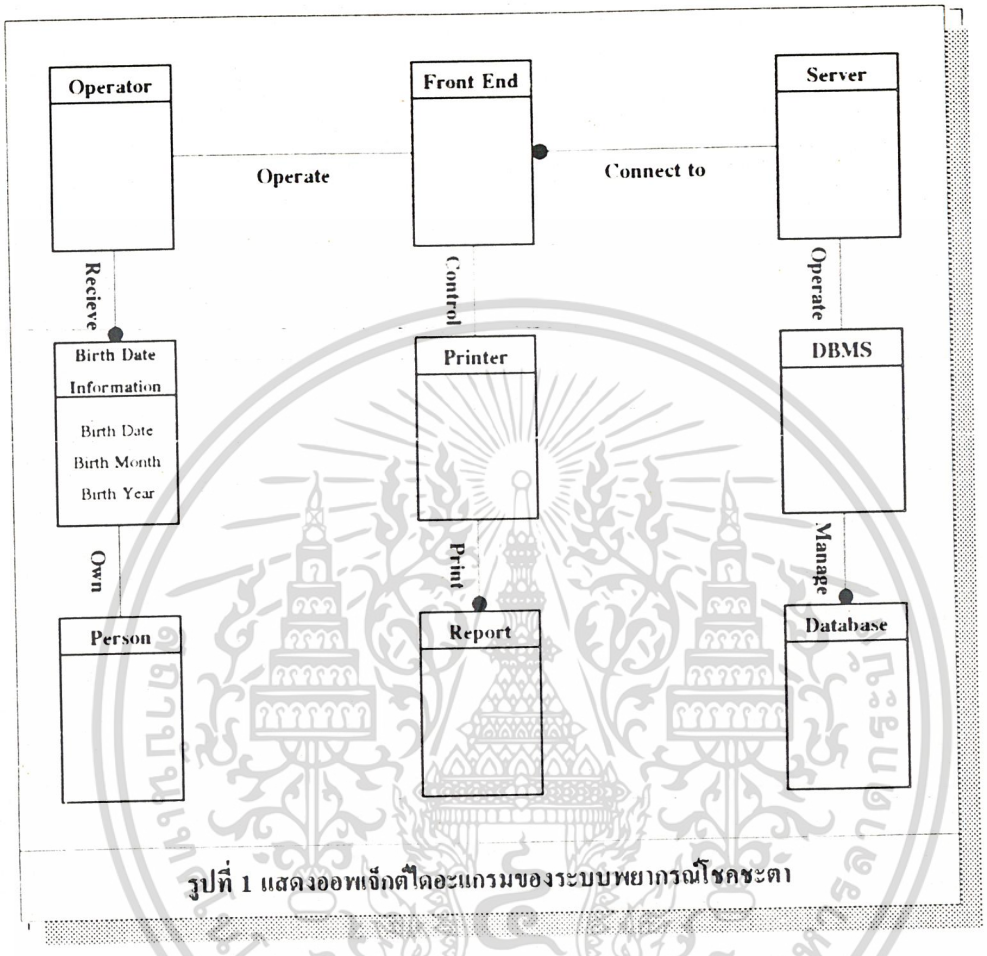
5.3 การวิเคราะห์และออกแบบระบบพยากรณ์ดวงชะตา

จากนิยามความต้องการของระบบเราสามารถนำมาวิเคราะห์ออกแบบโดยวิธีการโอเอ็มทีได้ โดยเราจะทำการเขียนออพเจ็กต์โมเดลที่แสดงโครงสร้างของออพเจ็กต์ภายในระบบ และไดนามิกส์โมเดลที่อธิบายพฤติกรรมของระบบที่เกี่ยวข้องกับเวลาและโอเปอร์เรชั่นต่างๆ และสุดท้ายฟังก์ชั่นนอลโมเดลที่อธิบายระบบในแง่ของการแปลงข้อมูลที่ไหลผ่านเข้ามาในระบบ ซึ่งโมเดลทั้งสามนี้จะถูกแสดงอยู่ในรูปของไดอะแกรม --ออพเจ็กต์ไดอะแกรม, อีเวนท์โฟลไดอะแกรม, อีเวนท์เทรสไดอะแกรม, เสตทไดอะแกรม และดาต้าโฟลไดอะแกรม สำหรับระบบพยากรณ์ดวงชะตาที่เรากำลังออกแบบนี้จะมีรูปของไดอะแกรมต่างดังที่ได้แสดงไว้ต่อไปนี้

รูปแรกจะเป็นออพเจ็กต์ไดอะแกรมซึ่งแสดงออพเจ็กต์โมเดลของระบบพยากรณ์ดวงชะตา จากแผนภาพนี้สามารถอธิบายได้ว่า ในระบบพยากรณ์ดวงชะตาจะประกอบไปด้วยออพเจ็กต์บุคคลที่ต้องการให้ทำนายดวงชะตา (Person), ออพเจ็กต์ข้อมูลวันเกิด (Birth Date Information), ออพเจ็กต์ผู้ปฏิบัติงานคอยควบคุมเครื่องคอมพิวเตอร์ (Operator), ออพเจ็กต์ฟร้อนท์เอ็นด์ (Front End), ออพเจ็กต์ของเครื่องพิมพ์ (Printer), ออพเจ็กต์ของรายงาน, ออพเจ็กต์เซอร์ฟเวอร์คอมพิวเตอร์ที่คอยให้ข้อมูลที่ใช้ในการพยากรณ์ (Server), ออพเจ็กต์ดีบีเอ็มเอส (DBMS), ออพเจ็กต์ดาต้าเบส (Database) จากแผนภาพบุคคลที่ต้องการให้ทำนายจะเป็นเจ้าของข้อมูลแสดงวันเกิดได้เพียงออพเจ็กต์เดียวเท่านั้น (เนื่องจากคนๆหนึ่งสามารถมีวันเกิดได้เพียงวันเดียว) แสดงข้อมูลแสดงวันเกิดหลายข้อมูลจะถูกส่งให้กับผู้ปฏิบัติการ ผู้ปฏิบัติการจะควบคุมเครื่องคอมพิวเตอร์ที่เป็นฟร้อนท์เอ็นด์เพียงเครื่อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เดียว สำหรับฟร้อนท์เอ็นด์แต่ละตัวจะมีเครื่องพิมพ์ต่ออยู่กับมันหนึ่งเครื่องเพื่อใช้ในการพิมพ์ผลการพยากรณ์



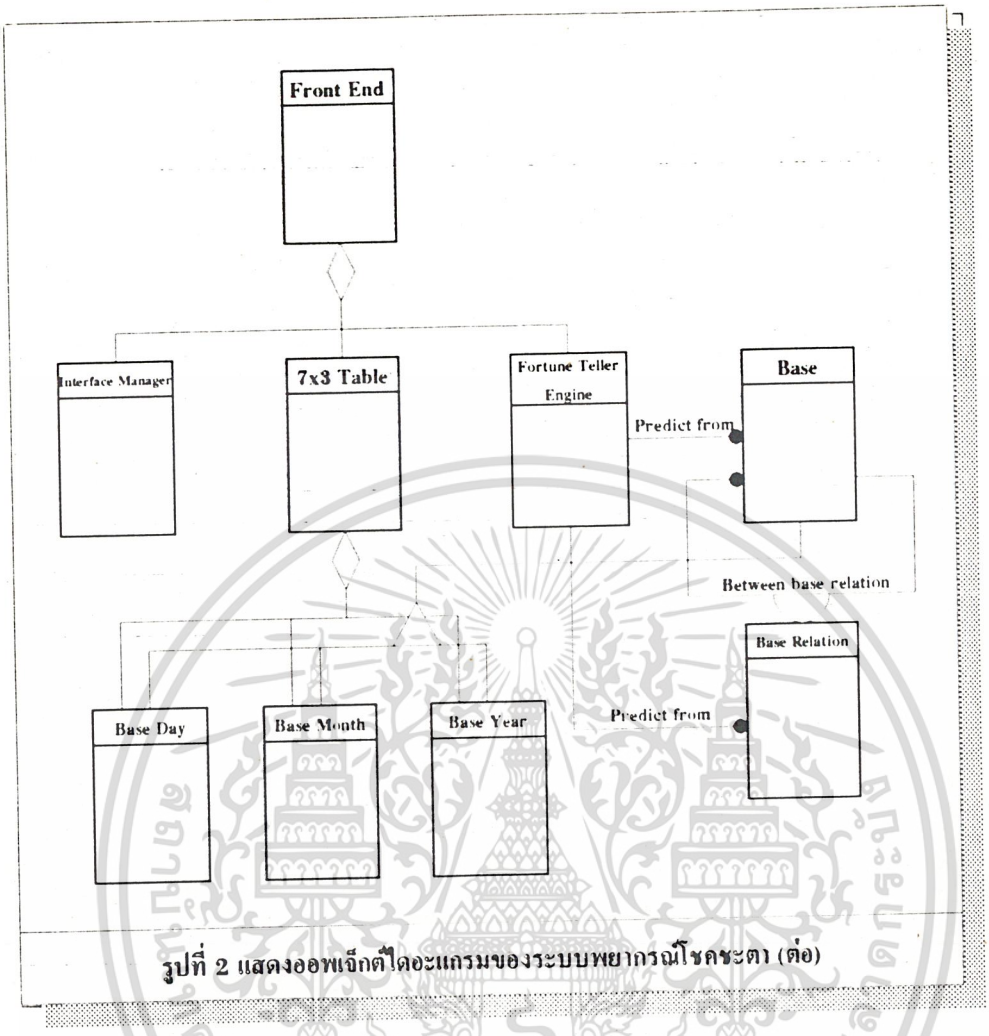
รูปที่ 1 แสดงออพเจ็กต์ไดอะแกรมของระบบพยากรณ์โชคชะตา

ออกมาบนกระดาษ ซึ่งเครื่องพิมพ์แต่ละเครื่องจะเป็นผู้พิมพ์รายงานหลายชุดด้วยกัน

ฟร้อนท์เอ็นด์หลายๆตัวจะต่อเข้ากับเครื่องเซอร์ฟเวอร์ (ซึ่งการต่อนั้นไม่จำเป็นต้องเป็นรูปแบบหรือโทโปโลจี้ใดอันหนึ่งเฉพาะ เพราะไดอะแกรมนี้แสดงระบบในรูปแบบของโลจิคอล (logical) เท่านั้น) เซอร์ฟเวอร์ตัวหนึ่งจะปฏิบัติการระบบฐานข้อมูลตัวหนึ่งซึ่งทำหน้าที่ในการจัดการตัวฐานข้อมูลหนึ่งที่เป็นศูนย์รวมข้อมูลที่ฟร้อนท์เอ็นด์แต่ละตัวต้องการ

รูปที่สองต่อไปนี้ก็จะเป็นแผนภาพที่แสดงออพเจ็กต์ไดอะแกรมต่อจากแผนภาพแรก โดยแผนภาพที่สองนี้เป็นส่วนขยายความเพิ่มเติมจากรูปที่หนึ่ง จากภาพแสดงให้เห็นว่าในออพเจ็กต์ของฟร้อนท์เอ็นด์จะประกอบไปด้วยออพเจ็กต์ที่ทำหน้าที่ติดต่อกับภายนอก (Interface Manger), ออพเจ็กต์ตาราง 7x3 ที่ใช้เก็บค่าของฐานต่างๆ (7x3 Table), และออพเจ็กต์ที่เป็นกลไกในการทำนายดวงชะตา (Fortune Teller Engine) ออพเจ็กต์กลไกในการทำนายจะทำนายผลจากข้อมูลที่ได้จากออพเจ็กต์ของฐานแต่ละฐาน (Base) และออพเจ็กต์ของความสัมพันธ์ระหว่างฐานแต่ละฐาน (Base Ralation) ออพเจ็กต์ของฐานสามารถแบ่งออกได้เป็นชนิดฐานวัน (Base

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2 แสดงออพเจ็กต์ไดอะแกรมของระบบพยากรณ์โชคชะตา (ต่อ)

Day) ชนิดฐานเดือน (Base Month) และชนิดฐานปี (Base Year) ออบพเจ็กต์ของฐานวันนี้ยังเป็นส่วนประกอบภายในออพเจ็กต์ของตาราง 7x3 ด้วย

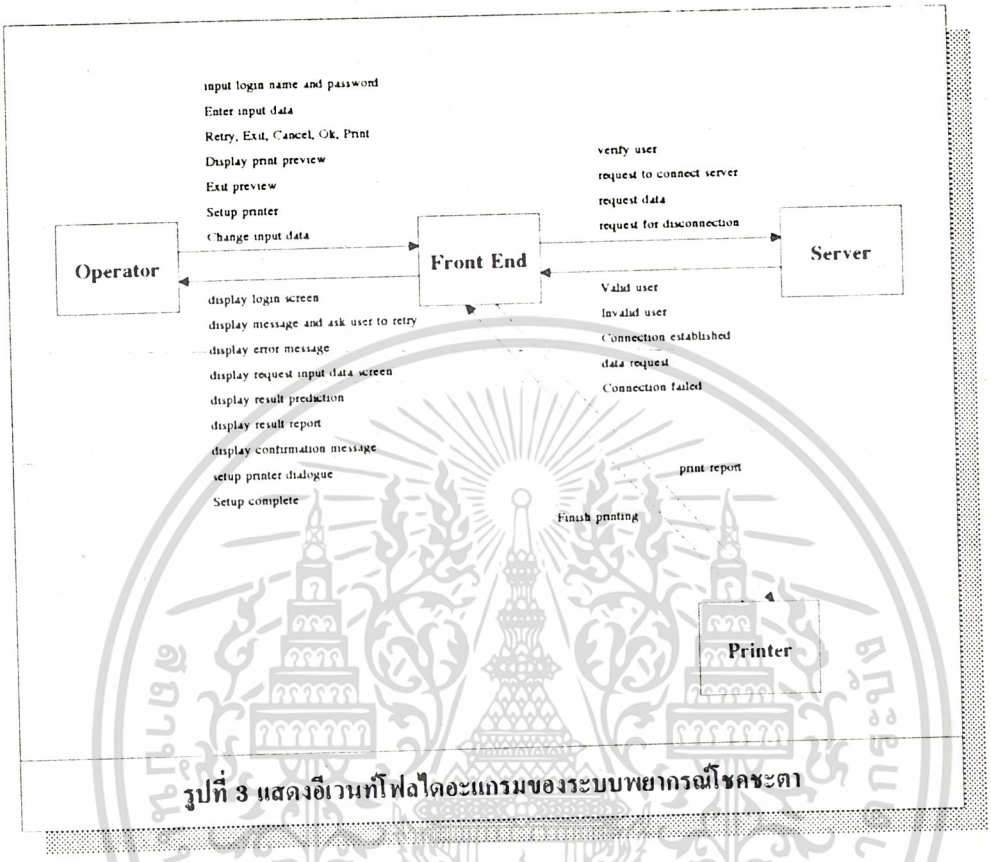
ต่อไปก็จะเป็นแผนภาพที่แสดงไดนามิกส์โมเดลของระบบพยากรณ์ ในรูปที่ 3 เป็นอีเวนท์โฟลไดอะแกรมของระบบพยากรณ์ดวงชะตา แผนภาพนี้แสดงให้เห็นถึงการติดต่อกันระหว่างออพเจ็กต์ต่างในระบบว่ามีอีเวนท์ใดไหลระหว่างออพเจ็กต์ใด ซึ่งอีเวนท์เหล่านี้เปรียบเสมือนเครื่องมือที่ออพเจ็กต์ใช้เป็นเครื่องมือสื่อสารระหว่างกัน อีเวนท์ที่ไหลจากออพเจ็กต์หนึ่งไปยังออพเจ็กต์หนึ่งจะกระตุ้นให้ออพเจ็กต์ที่รับอีเวนท์นั้นมีการเปลี่ยนแปลงสถานะถ้าอีเวนท์นั้นเหมาะสมที่จะทำให้เกิดการเปลี่ยนสถานะจากสถานะปัจจุบันของออพเจ็กต์

จากภาพที่สามนี้อีเวนท์ที่โอเปอร์เรเตอร์จะส่งให้กับฟร้อนท์เอ็นด์ก็จะมีดังนี้ คือ input login name, Enter input data, Retry, Exit, Cancel, Ok, Print, Display print preview, Exit preview, Setup printer, Change input data อีเวนท์ที่ฟร้อนท์เอ็นด์จะตอบกลับมาให้โอเปอร์เรเตอร์ก็ประกอบด้วย display login screen, display message and ask user to retry, display error message, display request input data screen, display result prediction, display result report, display confirmation message, setup printer dialogues, Setup complete

อีเวนท์ที่ฟร้อนท์เอ็นด์ส่งให้กับเครื่องพิมพ์ก็คือ print report และอีเวนท์ที่เครื่องพิมพ์จะส่งกลับมาก็คือ Finish printing อีเวนท์ที่ฟร้อนท์เอ็นด์ส่งให้กับเซอร์ฟเวอร์ก็มี verify user, request to connect server, request

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

to disconnection, request data อีเวนท์ที่เซอร์ฟเวอร์จะส่งกลับให้ฟร้อนท์เอ็นด์ก็มี Valid user, Invalid user,



รูปที่ 3 แสดงอีเวนท์โฟลไดอะแกรมของระบบพยากรณ์โรคชะตา

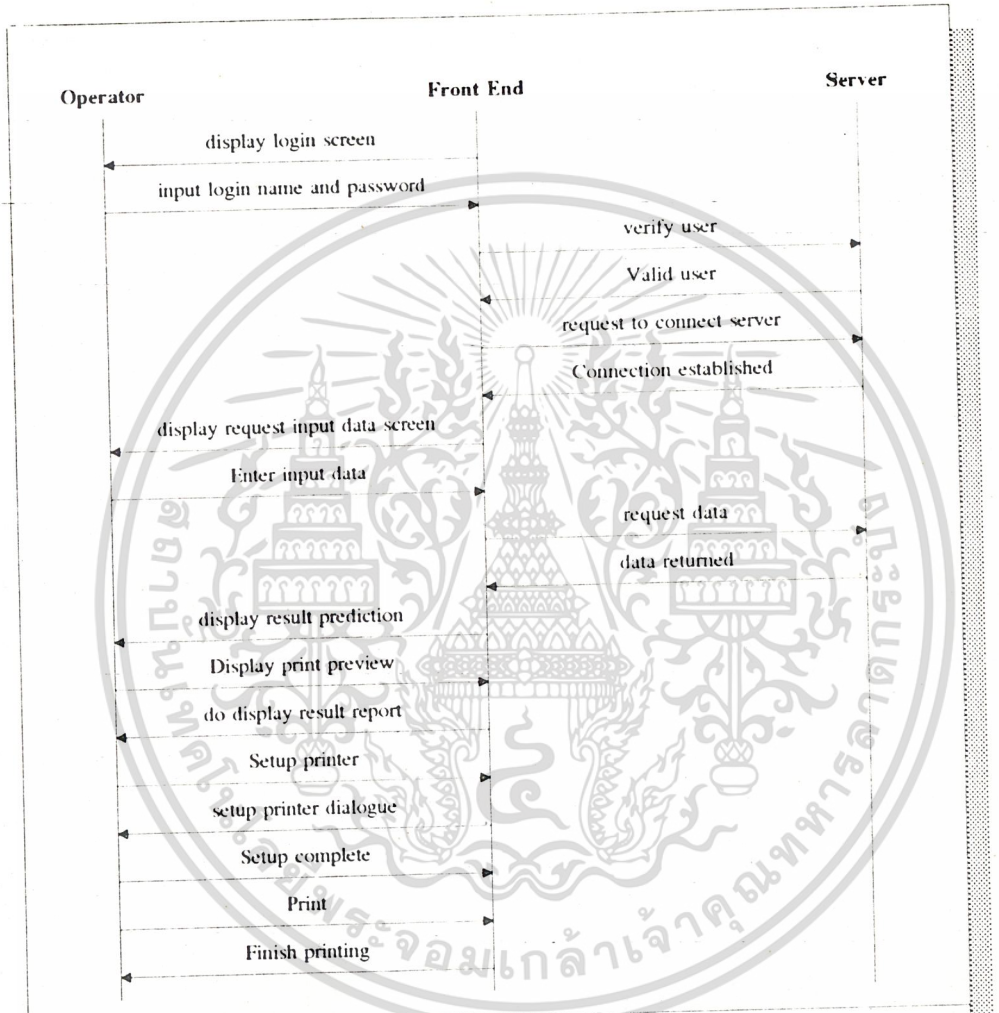
Connection established, Connection failed data return, Connection closed, data returned

สำหรับลำดับของการส่งอีเวนท์กันระหว่างอีเวนท์ต่างๆ เราสามารถดูได้จากอีเวนท์เทรสไดอะแกรมซึ่งได้แสดงไว้ในรูปที่ 4 จากไดอะแกรมนี้เราสามารถดูลำดับของการเกิดอีเวนท์ที่ส่งกันระหว่างออพเจ็กต์ต่างๆในระบบ ในระบบนี้เมื่อเริ่มต้นก็จะมีการส่งอีเวนท์ display login screen จากฟร้อนท์เอ็นด์ไปยังโอเปอร์เรเตอร์เพื่อให้โอเปอร์เรเตอร์ทำการป้อนข้อมูลชื่อและรหัสที่จะติดต่อกับระบบ เมื่อโอเปอร์เรเตอร์ป้อนชื่อและรหัสเสร็จเรียบร้อยก็ทำให้เกิดอีเวนท์ input login name and password ส่งกลับไปยังฟร้อนท์เอ็นด์ เมื่อฟร้อนท์เอ็นด์ได้รับอีเวนท์ก็จะทำการส่งอีเวนท์ verify user ไปยังเซอร์ฟเวอร์เพื่อตรวจสอบว่าเป็นคนที่สามารถเข้าใช้ข้อมูลได้หรือไม่ ถ้ามีสิทธิ์เข้าใช้ข้อมูลเซอร์ฟเวอร์ก็จะทำการส่งอีเวนท์ Valid user กลับมาบอกฟร้อนท์เอ็นด์ เมื่อฟร้อนท์เอ็นด์ได้รับอีเวนท์ Valid user ที่เซอร์ฟเวอร์ตอบกลับมาก็จะส่งอีเวนท์ request to connect server ไปอีกเพื่อขอเปิดการติดต่อกับเซอร์ฟเวอร์ เมื่อเซอร์ฟเวอร์ได้รับอีเวนท์นั้นและไม่มีข้อขัดข้องอันใดก็จะส่งอีเวนท์ Connection established กลับมาให้ฟร้อนท์เอ็นด์

หลักจากที่ติดต่อกับเซอร์ฟเวอร์ได้แล้วนั้น ฟร้อนท์เอ็นด์ก็จะส่งอีเวนท์ display request input data screen มายังโอเปอร์เรเตอร์เพื่อให้โอเปอร์เรเตอร์ทำการป้อนข้อมูลที่เป็นอินพุตของระบบพยากรณ์ดวงชะตา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โอเปอร์เรเตอร์ก็จะทำการป้อนอินพุต ซึ่งเกิดเป็น Enter input data ส่งกลับมากระตุ้นฟร้อนท์เอ็นด์ให้ทำการทำนายผล ซึ่งในการทำนายผลก็จะต้องมีการขอข้อมูลบางจากเซอร์ฟเวอร์ เมื่อฟร้อนท์เอ็นด์ต้องการข้อมูลจากเซอร์ฟเวอร์ก็จะเกิดอีเวนท์ request data ไปให้เซอร์ฟเวอร์ และเซอร์ฟเวอร์ก็จะส่งข้อมูลที่ต้องการกลับมาเกิดเป็นอีเวนท์ data returned เมื่อทำการเตรียมผลการพยากรณ์เสร็จเรียบร้อยแล้ว ฟร้อนท์เอ็นด์ก็จะแสดงผลการทำนาย



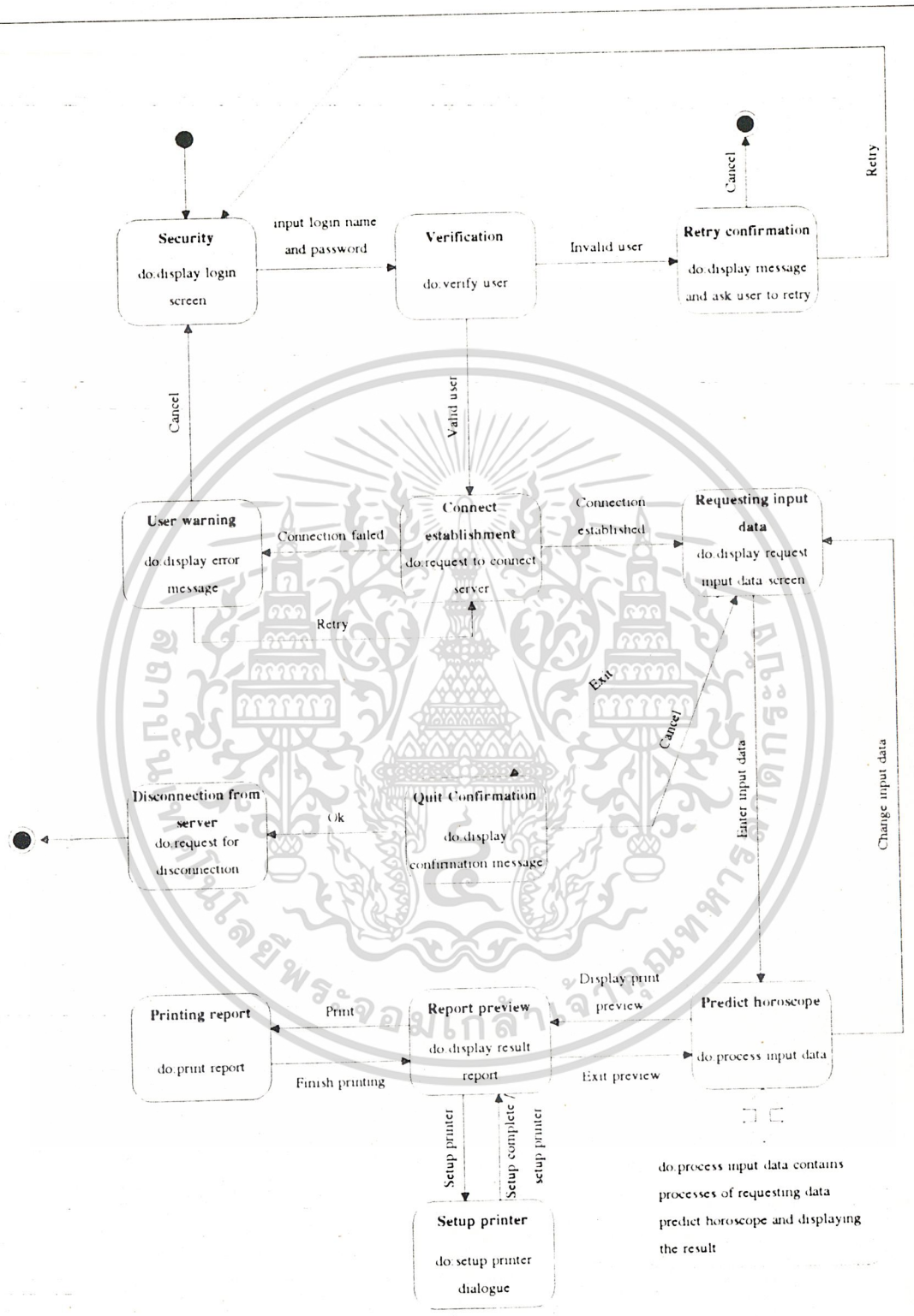
รูปที่ 4 แสดงอีเวนท์เทรสไดอะแกรมของระบบพยากรณ์โชคชะตา

ออกทางจอภาพเกิดเป็นอีเวนท์ display result prediction ถ้าโอเปอร์เรเตอร์ต้องการให้ดูผลลัพธ์ก่อนพิมพ์ออกทางเครื่องพิมพ์ด้วยก็จะสั่งคำสั่งดูเอกสารก่อนพิมพ์ หรือก็คือเกิดอีเวนท์ Display print preview ไปยังฟร้อนท์เอ็นด์ เมื่อฟร้อนท์เอ็นด์ได้รับอีเวนท์นี้ก็จะทำการแสดงเอกสารก่อนพิมพ์ขึ้นบนจอภาพเกิดเป็นอีเวนท์ do: display result report กลับมายังโอเปอร์เรเตอร์ ถ้าไม่มีการแก้ไขการตั้งค่าเกี่ยวกับเครื่องพิมพ์ โอเปอร์เรเตอร์ก็สามารถสั่งให้พิมพ์รายงานออกมาได้เลย ซึ่งการสั่งพิมพ์รายงานก็เป็นอีเวนท์ Print กลับมาสั่งฟร้อนท์เอ็นด์ให้พิมพ์ตัวรายงานออกมาทางเครื่องพิมพ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อมาถึงจุดนี้เราก็ทำการวิเคราะห์ระบบต่อไปเพื่อสร้างเสตทไดอะแกรม เสตทไดอะแกรมของฟร้อนท์เอ็นด์สามารถแสดงได้ดังรูปที่ 5 จากเสตทไดอะแกรมในรูปที่ 5 เมื่อฟร้อนท์เอ็นด์เริ่มต้นก็จะอยู่ที่แสตท Security เพื่อทำ do: display login screen ในการดูแลความปลอดภัยของข้อมูลไม่ให้มีการเข้าถึงข้อมูลจากบุคคลที่ไม่มีสิทธิ์ เมื่อมีอีเวนท์ input login name and password เข้ามาฟร้อนท์เอ็นด์ก็จะเปลี่ยนเสตทไปเป็น Verification เพื่อทำการตรวจสอบผู้ใช้ว่ามีสิทธิ์ในการเข้าถึงข้อมูลหรือไม่ ถ้ามีการตอบรับว่าผู้ใช้มีสิทธิ์ในการเข้าถึงข้อมูลโดยผ่านมาทางอีเวนท์ Valid user เสตทของฟร้อนท์เอ็นด์ก็จะเปลี่ยนไปเป็น Connect establishment เพื่อขอทำการติดต่อกับฐานข้อมูลในเซอร์ฟเวอร์ ถ้าเซอร์ฟเวอร์ตอบรับคำขอโดยส่งอีเวนท์ Connection established กลับมา เสตทของฟร้อนท์เอ็นด์ก็จะเปลี่ยนเป็น Requesting input data เพื่อแสดงหน้าจอรอรับอินพุตจากผู้ใช้ เมื่อผู้ใช้ป้อน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

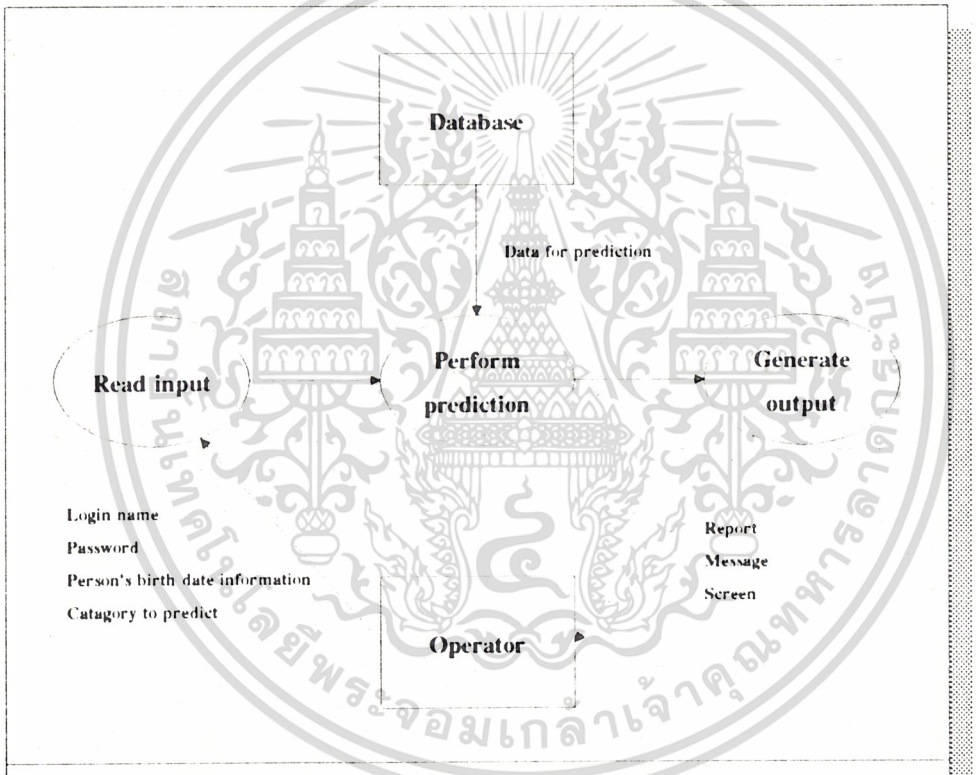


รูปที่ 5 แสดงเสตทไดอะแกรมของฟร้อนเอ็นท์ของระบบพยากรณ์โชคชะตา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อินพุตก็จะเกิดเป็นอีเวนท์ Enter input data มากระตุ้นให้เสตทของฟร้อนท์เอ็นด์เปลี่ยนไปเป็น Predict horoscope ทำการพยากรณ์ดวงชะตาตามวันเดือนปีเกิดนั้น และแสดงผลลัพธ์ออกบาทางจอภาพ ถ้าผู้ใช้ต้องการให้พิมพ์เอกสารก่อนพิมพ์ (print preview) ก็สั่งฟร้อนท์เอ็นด์โดยเกิดเป็นอีเวนท์ Display print preview มากระตุ้นให้ฟร้อนท์เอ็นด์เปลี่ยนเสตทเป็น Report preview และถ้าผู้ใช้สั่งให้มีการพิมพ์รายงานก็จะเกิดเป็นอีเวนท์มากระตุ้นให้เปลี่ยนเสตทเป็น Printing report

โมเดลสุดท้ายที่เรากำลังจะกล่าวกันนี้ก็จะเป็นฟังก์ชั่นนอลโมเดล ฟังก์ชันนอลโมเดลนี้แสดงอยู่ในรูปของดาต้าโฟลไดอะแกรมของระบบพยากรณ์ดวงชะตา รูปที่ 6 และรูปที่ 7 ฟังก์ชันนอลโมเดลนี้จะอธิบายถึงการเปลี่ยนแปลงข้อมูลนับตั้งแต่เริ่มไหลเข้ามาสู่ระบบจนกระทั่งออกเป็นผลลัพธ์ในตอนท้าย ซึ่งการแสดงการแปลงนี้ไม่ได้ขึ้นกับเวลาแต่อย่างใดเนื่องจากดาต้าโฟลไดอะแกรมไม่ได้ใช้ในการแสดงความสัมพันธ์ระหว่างเวลากับระบบ



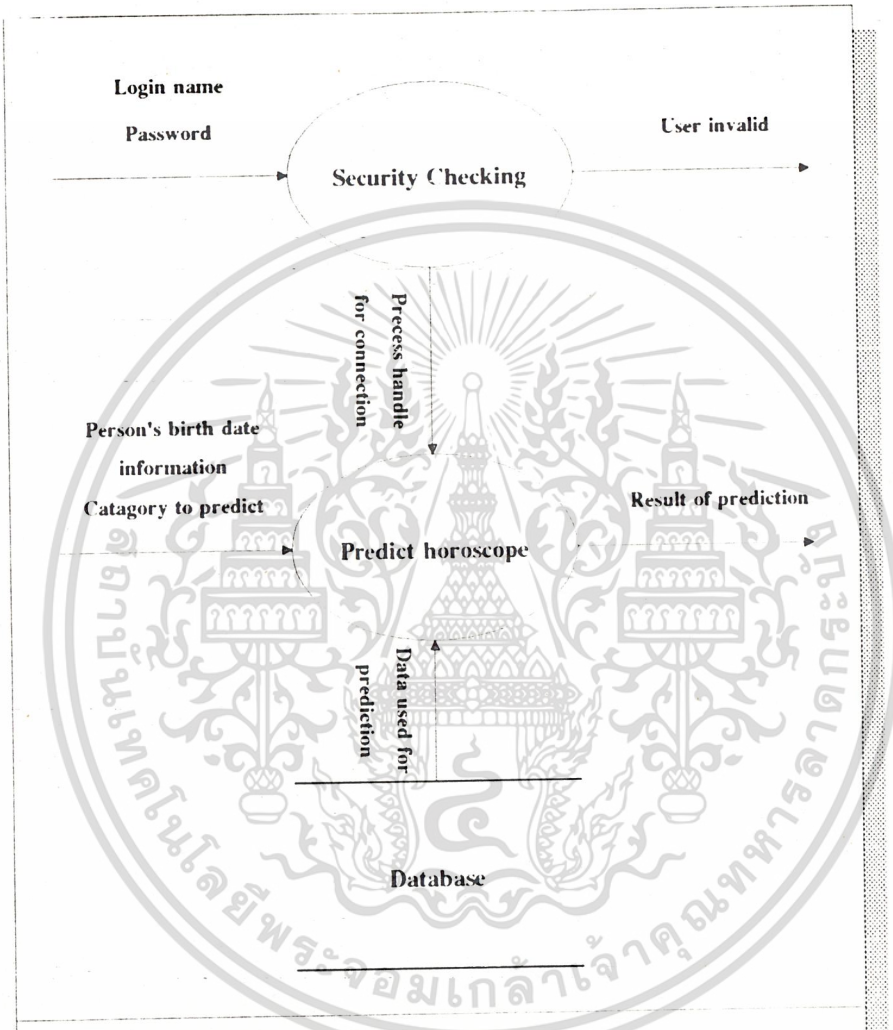
รูปที่ 6 แสดงดาต้าโฟลไดอะแกรมของระบบพยากรณ์โชคชะตา

เหมือนกับเสตทไดอะแกรมนั่นเอง

จากรูปที่ 6 นี้จะเห็นได้ว่าโอเปอร์เรเตอร์จะส่ง Login name, Password, Person's birth date information, และ Catagory to predict มาผ่านกระบวนการรับข้อมูล (Read input) ซึ่งหลังจากนั้นข้อมูลก็จะถูกส่งผ่านไปยังกระบวนการทำนาย (Perform prediction) กระบวนการนี้จะมีการดึงข้อมูลที่จะใช้ในการทำนาย จากฐานข้อมูล เมื่อสามารถหาผลการทำนายได้แล้วก็จะส่งผลการทำนายต่อไปให้กระบวนการแสดงผล (Generate output) ซึ่งจะเป็นกระบวนการที่ทำ Report, Message และ Screen ต่างๆให้ติดต่อกับโอเปอร์เรเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนในรูปที่ 7 ก็จะอธิบายถึงดาต้าโฟลไดอะแกรมที่แยกย่อยกระบวนการทำนาย (Perform prediction) ออกเป็นส่วนย่อยเพื่อที่จะกำหนดในรายละเอียดของการทำงานของระบบ จากภาพจะเป็นได้ว่าเมื่อข้อมูลที่เป็นอินพุตได้รับเข้ามาก็จะแบ่ง Login name และ Password ไปผ่านกระบวนการที่ป้องกันความปลอด ภัยและติดต่อกับฐานข้อมูล และข้อมูลอีกส่วนหนึ่ง (วันเดือนปีเกิด,และเรื่องที่ต้องการให้ทำนาย) ก็จะถูกใช้ในการพยากรณ์



รูปที่ 7 แสดงดาต้าโฟลวไดอะแกรมของโพรเซส "perform prediction"

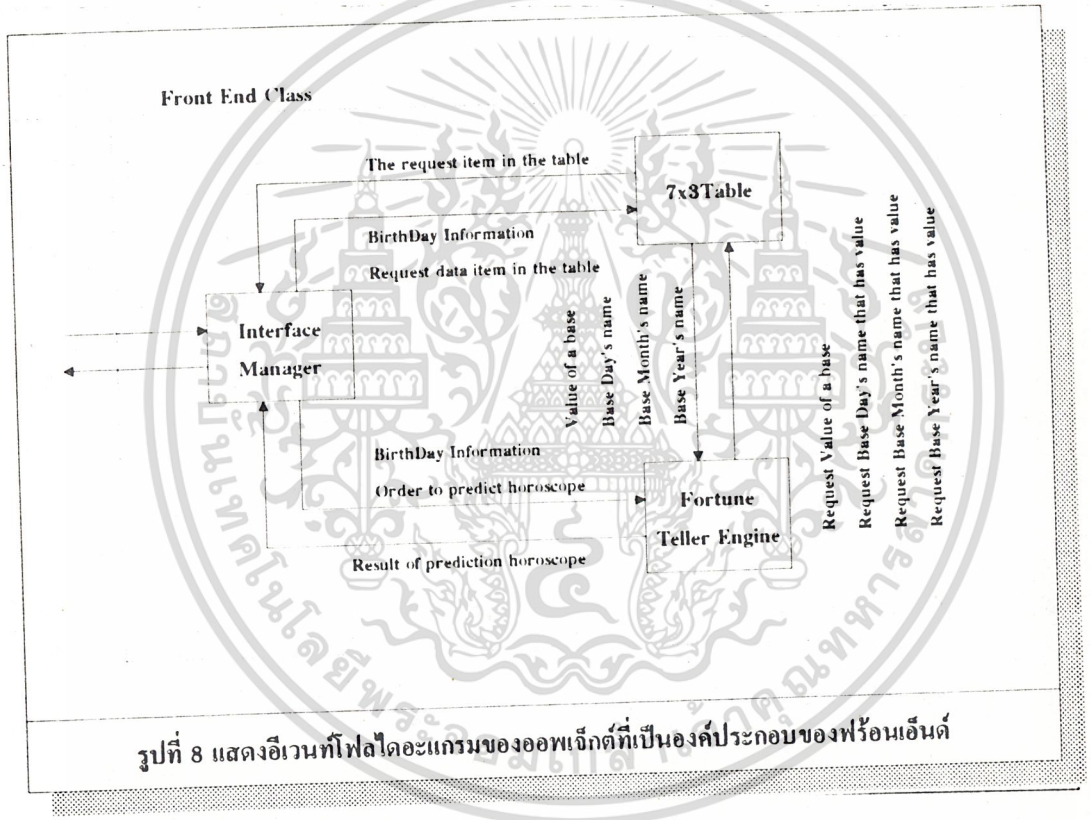
ควงชะตา ซึ่งผลลัพธ์ต่างๆก็จะถูกส่งออกไปให้กระบวนการแสดงผลต่อไป

เมื่อเราได้แบบของระบบที่แสดงได้ด้วยโมเดลต่างๆดังที่แสดงไปแล้วนี้ เราก็จะมาทำการออกแบบตัวฟร้อนท์เอ็นด์ในรายละเอียดว่าภายในตัวฟร้อนท์เอ็นด์นี้จะต้องแบ่งออกเป็นระบบย่อยอย่างไรบ้าง ในที่นี้จากออฟเจ็กต์โมเดลรูปที่ 2 ได้อธิบายว่าภายใต้ตัวฟร้อนท์เอ็นด์จะประกอบไปด้วยระบบย่อยอีก 3 ส่วนคือ Interface manager, 7x3 Table และ Fortune Teller Engine ดังนั้นเราจึงมาทำการวิเคราห์และออกแบบต่อว่าระบบย่อยๆเหล่านี้มีการติดต่อกับอย่างไรบ้าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับการติดต่อระหว่างกันของระบบย่อยนี้เราสามารถแสดงได้ในรูปที่ 8 จากภาพเราจะเห็นว่า Interface manager จะเป็นตัวที่ทำหน้าที่ติดต่อกับภายนอกของฟร้อนท์เอ็นด์ และทำการกระตุ้นกลไกตัวอื่นภายในฟร้อนท์เอ็นด์ให้ทำงานตามที่ต้องการ อีเวนท์ที่ Interface manager ส่งให้กับ 7x3 Table จะมีดังนี้คือ BirthDay Information ซึ่งจะไปกระตุ้นให้ตารางทำการคำนวณค่าอื่นภายในตารางให้สมบูรณ์ และ Request data item in the table เพื่อเป็นการกระตุ้นให้ตารางส่งข้อมูลตัวที่ต้องการกลับมา ซึ่งตารางก็จะตอบสนองด้วยการส่งข้อมูลกลับมาแล้วเกิดเป็นอีเวนท์ The request item in the table

อีเวนท์ที่ Interface manager ส่งให้กับ Fortune Teller Engine ก็จะประกอบด้วย BirthDay Information ซึ่งเป็นข้อมูลที่เป็นอินพุตที่จะต้องใช้ในการคำนวณ และ Order to predict horoscope เพื่อเป็นการสั่งให้ทำการพยากรณ์ดวงชะตา ซึ่ง Fortune Teller Engine ก็จะทำการพยากรณ์และส่งผลลัพธ์กลับมาเกิดเป็นอีเวนท์ Result



รูปที่ 8 แสดงอีเวนท์โฟลไดอะแกรมของออพเจ็กต์ที่เป็นองค์ประกอบของฟร้อนเอ็นด์

of prediction horoscope

ระหว่าง Fortune Teller Engine กับ 7x3 Table ก็จะมีการส่งอีเวนท์ไปมาระหว่างกัน โดย Fortune Teller Engine จะส่ง Request Value of a base เพื่อขอข้อมูลว่าฐานหนึ่งๆมีค่าประจำฐานเป็นเท่าใด, Request Base Day's name that has value เพื่อขอทราบชื่อของฐานวันที่มีค่าเท่ากับค่าหนึ่งๆ, Request Base Month's name that has value เพื่อขอทราบชื่อของฐานเดือนที่มีค่าเท่ากับค่าหนึ่งๆ, Request Base Year's name that has value เพื่อขอทราบชื่อของฐานปีที่มีค่าเท่ากับค่าหนึ่งๆ ส่วนอีเวนท์ที่ตาราง 7x3 จะตอบกลับมาให้กับ Fortune Teller Engine ก็จะมี Value of a base เพื่อบอกค่าของฐานหนึ่งๆ, Base Day's name เพื่อบอกชื่อของฐานวันที่ถามเข้ามา, Base Month's name เพื่อบอกชื่อของฐานเดือนที่ถามเข้ามา, Base Year's name เพื่อบอกชื่อของฐานปีที่ถามเข้ามา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทสรุป

จากการใช้วิธีการแบบโอเอ็มทีมาใช้ในการออกแบบระบบพยากรณ์นี้ เราก็สามารถนำเอาไดอะแกรมต่างที่ได้จากการวิเคราะห์และออกแบบ นำไปสร้างเป็นระบบพยากรณ์ดวงชะตาได้ ในช่วงของการนำมาสร้างนั้นเรายังต้องทำการตัดสินใจอีกว่าระบบของเราจะใช้แฟลทฟอร์มของเครือข่ายแบบใด มีการใช้โอเปอร์เรตติ้งซิสเต็มแบบใด และอื่นๆในรายละเอียด เป็นต้น ในโครงงานนี้เราจะใช้เทคโนโลยีของระบบฐานข้อมูลแบบไคลเอ็นท์เซอร์ฟเวอร์มาทำการสร้าง และใช้ระบบจัดการฐานข้อมูล SQLBase ซึ่งเราจะได้กล่าวถึงเอ็นไวรอนเมนต์ในการพัฒนานี้ต่อไปในบทที่ 6

สิ่งที่เราอยากจะเน้นอีกครั้งก็คือ ระบบที่เราทำการออกแบบนี้ใช้วิธีการโอเอ็มที ซึ่งเป็นระเบียบวิธีการหนึ่งที่ใช้แนวความคิดแบบออพเจ็กต์โอเรียนเต็ด ดังนั้นระบบที่สร้างได้ จะมีลักษณะของการเป็นออพเจ็กต์โอเรียนเต็ดในระดับโลจิคอล ซึ่งจะง่ายต่อการแก้ไขและเพิ่มเติม นอกจากนี้ยังช่วยทำให้ระบบที่ทำการสร้างนี้มีความน่าเชื่อถือมากกว่าระบบที่ใช้การออกแบบแบบดั้งเดิม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6

การสร้างระบบ (System Implementation)

6.1 กล่าวนำ

ในบทนี้ต่อเนื่องมาจากบทก่อนหน้านี้ที่เราได้ทำการศึกษาระเบียบวิธีการที่จะนำมาใช้ในการออกแบบระบบแบบไคลเอ็นท์เซอร์ฟเวอร์ได้อย่างมีประสิทธิภาพ และได้นำเอาระเบียบวิธีการเหล่านั้นมาทำการวิเคราะห์ออกแบบระบบการลงทะเบียนที่เป็นตัวอย่างไปแล้วในบทที่ 3และระบบพยากรณ์ดวงชะตาในบทที่ 5 พอมาถึงในบทนี้เราจะได้กล่าวถึงการนำแบบที่ได้สร้างไว้แล้วมาสร้างให้เป็นแอพพลิเคชั่นจริงๆ โดยในบทนี้เราได้ทดลองใช้ซอฟต์แวร์ทูลในการพัฒนา 2 แบบคือ เอสคิวแอลวินโดวส์ (SQLWindows) ของบริษัทกุปต้า คอร์ปเปอร์เรชั่น (Gupta Corporation) และ วิชวลเบสิก (Visual Basic) ของบริษัทไมโครซอฟต์คอร์ปเปอร์เรชั่น ทั้งนี้ก็เพื่อศึกษาถึงว่าทูลแบบใดจะมีความเหมาะสมกว่ากันในการพัฒนาระบบ และสนับสนุนกับระเบียบวิธีที่เราเลือกมาใช้ในการวิเคราะห์ออกแบบระบบ

ในส่วนแรกนี้เราจะแนะนำเอสคิวแอลวินโดวส์ของบริษัทกุปต้าก่อน เนื่องจากบริษัทที่เป็นเจ้าของทูลตัวนี้กล่าวว่าได้พัฒนาทูลนี้ขึ้นมาเพื่อการพัฒนาคาต้าเบสแอพพลิเคชั่นแบบไคลเอ็นท์เซอร์ฟเวอร์โดยเฉพาะ ดังนั้น เราจึงคิดว่าทูลตัวนี้น่าจะมีดีกว่าในการพัฒนาระบบไคลเอ็นท์เซอร์ฟเวอร์ เมื่อเทียบกับทูลวิชวลที่ออกแบบมาเพื่อพัฒนาแอพพลิเคชั่นโดยทั่วไป อย่างไรก็ตามทูลแต่ละตัวต่างมีจุดดีจุดด้อยแตกต่างกันดังที่เราจะได้กล่าว ถึงต่อไป

6.2 เอสคิวแอลวินโดวส์

เอสคิวแอลวินโดวส์เป็นซอฟต์แวร์ทูลของบริษัทกุปต้าคอร์ปเปอร์เรชั่นที่ใช้ในการพัฒนาโปรแกรมฐานข้อมูลแบบไคลเอ็นท์เซอร์ฟเวอร์บนระบบไมโครซอฟต์วินโดวส์ ซอฟต์แวร์ที่สร้างขึ้นโดยเอสคิวแอลวินโดวส์จะมีคุณสมบัติครบถ้วนตามคุณสมบัติที่แอพพลิเคชั่นทั่วไปบนวินโดวส์สามารถจะมีได้ นอกจากนี้โปรแกรมที่สร้างด้วยเอสคิวแอลวินโดวส์ยังสามารถติดต่อกับเซอร์ฟเวอร์ต่างๆจากหลายผู้ผลิตได้ ซึ่งรวมถึงเอสคิวแอลเบสของกุปต้า (Gupta SQLBase), ออราเคิล (Oracle), เอสคิวแอลเซอร์ฟเวอร์ของไมโครซอฟต์/ไซเบส (Microsoft/Sybase SQLServer), ดีบีทูของไอบีเอ็ม (IBM DB2), เอเอสสี่ร้อย (AS/400), อินฟอร์มิกส์ (Informix) และอื่นๆ

เอสคิวแอลวินโดวส์รวมเอาวิธีการใช้ที่ง่ายแบบชี้และกด (point-and-click ease of use) เข้ากับคุณสมบัติในการพัฒนาซอฟต์แวร์ที่มีประสิทธิภาพ เป็นต้นว่าเอสคิวแอลวินโดวส์ช่วยให้ทีมโปรแกรมเมอร์ร่วมกันพัฒนาแอพพลิเคชั่น โดยที่ส่วนต่างๆของโครงงานสามารถถูกแบ่งกระจายไปให้กับโปรแกรมเมอร์แต่ละคนได้ และ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สามารถนำกลับมารวมกันได้เป็นแอพพลิเคชั่นที่สมบูรณ์อย่างไม่มีปัญหา นอกจากนี้ส่วนของแอพพลิเคชั่นหนึ่งๆ ยังสามารถนำกลับมาใช้ใหม่ในแอพพลิเคชั่นอื่นได้อีกด้วย การที่เป็นเช่นนี้ทำให้เป็นไปได้ที่จะทำการสร้างไลบรารี่ของออพเจ็กต์ในแอพพลิเคชั่นเก็บเอาไว้เพื่อการนำกลับมาประกอบใช้เป็นแอพพลิเคชั่นตัวใหม่ได้อย่างรวดเร็ว

เอสคิวแอลวินโดวส์ยังสนับสนุนออพเจ็กต์โอเรียนเต็ดอย่างสมบูรณ์ การนำเอาวิธีการแบบออพเจ็กต์โอเรียนเต็ดมาใช้ในการสร้างแอพพลิเคชั่นจะทำให้มีโอกาสในการนำส่วนต่างๆกลับมาใช้ได้มากขึ้น และยังช่วยลดขนาดของแอพพลิเคชั่นและระยะเวลาที่ต้องใช้ในการพัฒนาลงอีกด้วย คุณสมบัติในการโปรแกรมแบบออพเจ็กต์โอเรียนเต็ดทั้งหมดจะมีใช้โปรแกรมเมอร์เลือกใช้ได้ ทั้งนี้รวมถึงความสามารถในการกำหนดคลาสส์ไฮราคี่ส์ที่มีทั้งซิงเกิลอินเฮียริแทนส์ (single inheritance) และมัลติเพิลอินเฮียริแทนส์ (multiple inheritance) ด้วย

เราสามารถเลือกที่จะใช้ความสามารถในการโปแกรมแบบออพเจ็กต์โอเรียนเต็ดของเอสคิวแอลวินโดวส์หรือไม่ก็ได้ ขึ้นอยู่กับเราเอง หรือเราสามารถเลือกใช้บางส่วนโดยไม่ใช้บางส่วนก็ได้ เอสคิวแอลวินโดวส์มีความยืดหยุ่นที่ให้เราสามารถเลือกใส่ความสามารถทางออพเจ็กต์โอเรียนเต็ดลงไปในส่วนใดก็ได้ตามแต่เราจะเลือก

นอกจากเอสคิวแอลวินโดวส์แล้วภายในทูลชุดนี้ยังประกอบไปด้วยทูลตัวอื่นที่มาช่วยประกอบในการพัฒนาโดยที่เราสามารถสรุปได้ดังนี้

- **เอสคิวแอลวินโดวส์ (SQLWindows)** : เป็นทูลที่ให้สภาวะแวดล้อมในการสร้างโปรแกรมประยุกต์ (Application program) ที่ง่ายต่อการเรียนรู้และใช้งาน โดยจะใช้หลักการชี้และกดมาเป็นกลไกในการสร้างโปรแกรม. โปรแกรมที่สร้างโดยเอสคิวแอลวินโดวส์นี้จะสามารถติดต่อกับคาต้าเบสเซอร์ฟเวอร์ที่แตกต่างกันได้ตามความต้องการของผู้เขียนโปรแกรม.
- **เควสท์ (Quest)** : เป็นทูลในการเข้าถึงข้อมูลที่ช่วยผู้ใช้ในการสร้างเอสคิวแอลคิวรี่ (SQL queries), แสดงและแก้ไขตารางเอสคิวแอล (SQL tables), สร้างรายงานโดยไม่จำเป็นต้องมีความรู้ภาษาเอสคิวแอล (SQL) เลย. นอกจากนี้เควสท์ยังสามารถติดต่อกับคาต้าเบสเซอร์ฟเวอร์ที่มีชื่อเสียงอื่นๆได้อีกหลายตัวด้วย.
- **รีพอร์ทวินโดวส์ (ReportWindows)** : เป็นทูลในการออกแบบรายงานเพื่อสร้างเป็นรีพอร์ทเท็มเพลท (report template) ที่จะนำไปใช้กับโปรแกรมที่พัฒนาด้วยเอสคิวแอลวินโดวส์
- **ทีมวินโดวส์ (TeamWindows)** :
  เป็นทูลที่ใช้ในการจัดการการพัฒนาโปรแกรมโดยใช้เอสคิวแอลวินโดวส์แบบที่มีโปรแกรมเมอร์หลายคน. ทีมวินโดวส์จะมีคาต้าดิกชั่นนารี่ครีเวนท์แอพพลิเคชั่นเจนเนอร์เรเตอร์ (data dictionary-driven application generator), เช็คอินแอนด์เช็คเอาท์เฟซิลิตี้ (check-in and check-out facilities), ระบบรักษาความปลอดภัย (security), ความสามารถในการจัดการควบคุมโพรเจ็กต์ (project management feature), และการควบคุมเวอร์ชั่นของซอร์ทโค๊ด (source-code and version control) เพื่อช่วยทีมพัฒนาในการจัดการการทำงานร่วมกัน.
- **อีดิทท์วินโดวส์ (EditWindows)** : เป็นทูลที่ใช้ในการปรับปรุงโปรแกรมที่พัฒนาโดยใช้เอสคิวแอลวินโดวส์ให้สามารถนำไปใช้ในประเทศอื่นๆได้โดยไม่ต้องทำการเปลี่ยนแปลงซอร์ทโค๊ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

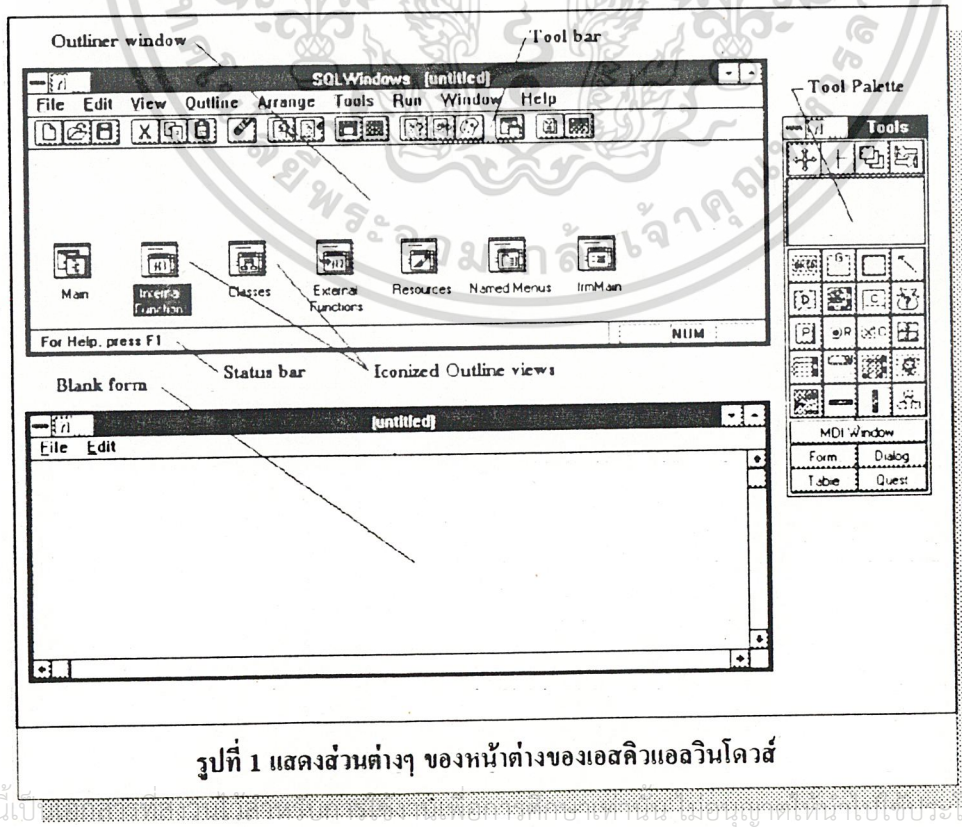
- **เอสคิวแอลทอร์คฟอร์วินโดวส์ (SQLTalk for Windows)** : เป็นทูลในการจัดการฐานข้อมูล เช่น สร้างฐานข้อมูล, ทำคิวรี่ค้นหาข้อมูล และดูแลฐานข้อมูล
  โดยที่ตัวเอสคิวแอลทอร์คนี้จะมีลักษณะการทำงานในแบบอินเตอร์แอ็คทิฟ (interactive)
- **เอสคิวแอลเบสฟอร์วินโดวส์ (SQLBase Server for Windows)** :
  เป็นส่วนที่ทำหน้าที่เป็นเซอร์ฟเวอร์ของฐานข้อมูลทั้งแบบผู้ใช้คนเดียวและผู้ใช้หลายคน.

6.2.1 การพัฒนาแอพพลิเคชั่นโดยใช้เอสคิวแอลวินโดวส์

เอสคิวแอลวินโดวส์เป็นทูลที่ใช้ในการพัฒนาแอพพลิเคชั่นที่ใช้หลักการในการสร้างแอพพลิเคชั่นที่ง่าย ๆ กล่าวคือเป็นการลากออพเจ็กต์มาตรฐานที่มีให้บนทูลแพลเล็ท (Tool Palette) นำมาวางบนหน้าต่างของแอพพลิเคชั่นที่จะสร้างขึ้น แล้วจากนั้นก็ทำการใส่ตัวโปรแกรมที่ควบคุมการทำงานของระบบ โดยที่หน้าต่างของแอพพลิเคชั่นหนึ่งๆอาจจะมีได้หลายๆหน้าต่างก็ได้.

บนเอสคิวแอลวินโดวส์จะมีภาษาที่ใช้ในการพัฒนาโปรแกรมเฉพาะของมันเอง ภาษานี้มีชื่อว่าภาษาเอสเอแอล (SAL, SQLWindows Application Language) ภาษาเอสเอแอลถูกออกแบบเพื่อการพัฒนาโปรแกรมด้วยเอสคิวแอลวินโดวส์โดยเฉพาะ และมีฟังก์ชันระดับสูงให้ในงานเกี่ยวกับฐานข้อมูลมากมาย. นอกจากฟังก์ชันที่เกี่ยวกับฐานข้อมูลแล้วเอสเอแอลยังมีฟังก์ชันที่ใช้สนับสนุนคุณสมบัติของวินโดวส์ เช่น ดีดีอี (DDE, Dynamic Data Exchange), โอแอลอี (OLE, Object Linking and Embedding), เอ็มดีไอ (MDI, Multiple Document Interface) และ ดรักแอนด์ดรอพ (drag-and-drop) เป็นต้น.

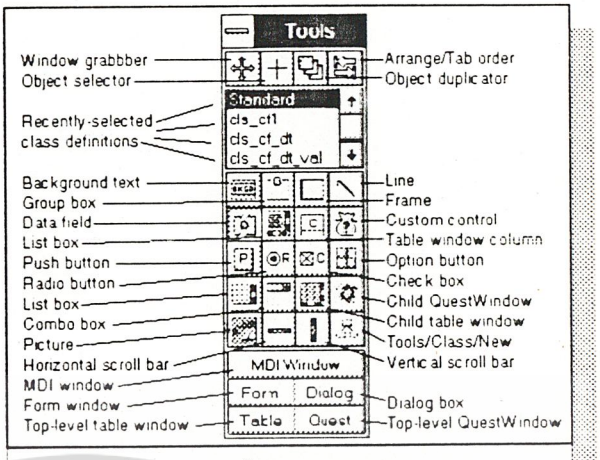
ก่อนอื่นผมขออธิบายถึงเอ็นไวรอนเมนต์หรือสิ่งแวดล้อมในการพัฒนาเสียก่อนว่าบนเอสคิวแอลวินโดว



รูปที่ 1 แสดงส่วนต่างๆ ของหน้าต่างของเอสคิวแอลวินโดวส์

เอกสารนี้เป็น... ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส์มีเอ็นไวรอนเมนต์เป็นอย่างไร เมื่อเริ่มต้นเข้าสู่เอสคิวแอลวินโดวส์ โปรแกรมก็จะแสดงส่วนที่ใช้ในการ พัฒนาโปรแกรมดังรูปที่ 1 นี้ ส่วนต่างๆที่เห็นภาพในรูปประกอบด้วย 3 ส่วนหลักๆคือ



รูปที่ 2 แสดงเครื่องมือต่างๆในทูลแพลเล็ท

1. **เอาร์ทไลน์เนอร์วินโดวส์** (Outliner windows)

ซึ่งเป็นใจกลางของเอ็นไวรอนเมนต์ในการพัฒนาของเอสคิวแอลวินโดวส์ และเป็นส่วนที่แสดงซ้อร์ทโค๊ดทั้ง หมดของแอพพลิเคชั่นที่เรากำลังพัฒนา

2. **ทูลแพลเล็ท** (Tool Palette)

ซึ่งเป็นส่วนที่ให้เครื่องมือที่ใช้ในการวาดยูสเซอร์อินเตอร์เฟสออพเจ็กต์ของแอพพลิเคชั่น รายละเอียดของทูลที่มีบนแพลเล็ทแสดงไว้ในรูปที่ 2

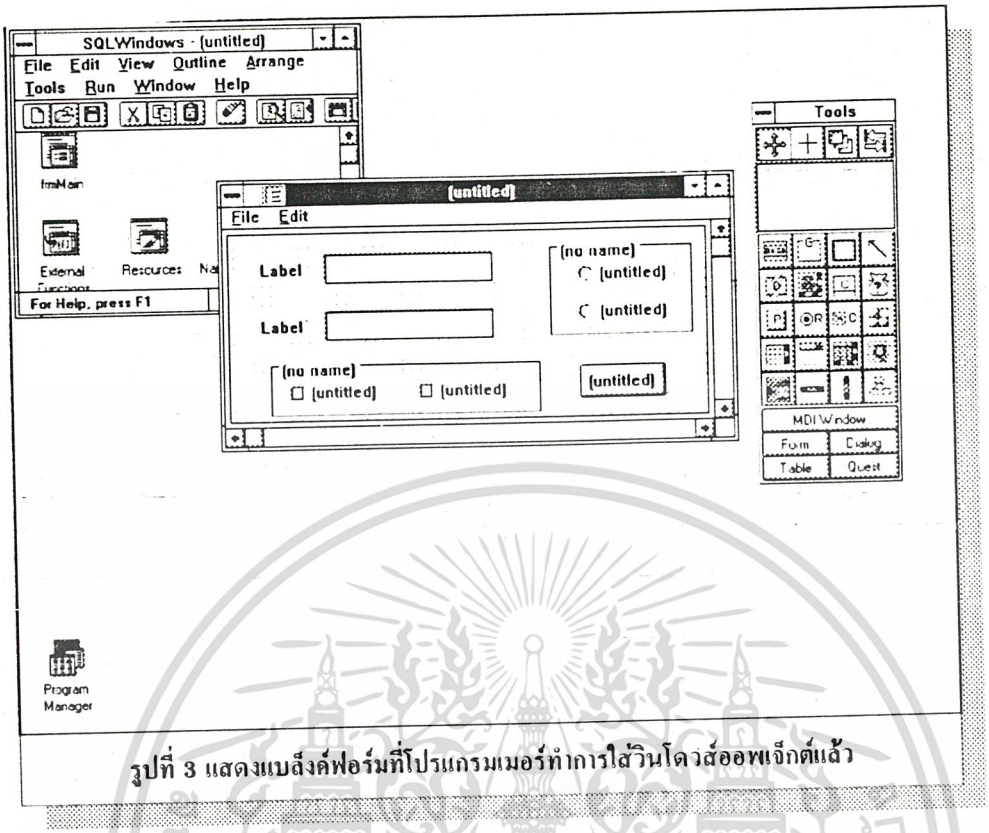
3. **แบล็งค์ฟอร์ม** (Blank Form) เป็นหน้าต่างที่จะถูกสร้างเป็นแอพพลิเคชั่น เราสามารถปรับขนาด, ตั้งคุณสมบัติ (Properties) ต่างๆ ใส่เมนูไอเท็ม (Menu Item) หรือเมนูบาร์ (Menu Bar) และอื่นๆได้ตามต้องการ

เมื่อเริ่มต้นที่จะสร้างโปรแกรมด้วยเอสคิวแอลวินโดวส์โปรแกรมเมอร์ก็จะทำการวาดยูสเซอร์อินเตอร์เฟสของแอพพลิเคชั่นตามความต้องการโดยเลือกใช้เครื่องมือในการวาดที่เหมาะสมจากทูลแพลเล็ท เมื่อเลือกเครื่องมือได้แล้วก็นำมาวาดบนแบล็งค์ฟอร์มในตำแหน่งที่ต้องการ ถ้ายังขาดส่วนใดที่ต้องการใช้มีบนหน้าต่างของแอพพลิเคชั่นก็สามารถวนเลือกเครื่องมืออื่นๆตามความต้องการมาวาดต่อไปได้

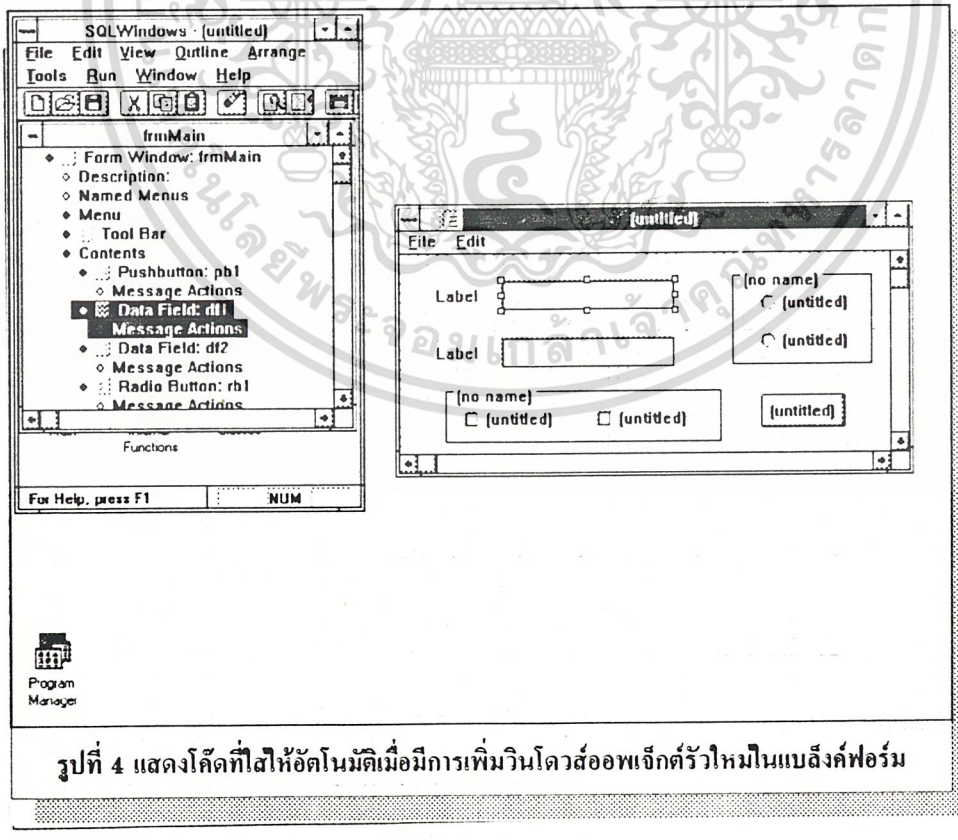
เมื่อโปรแกรมเมอร์ทำการวางส่วนประกอบต่างของยูสเซอร์อินเตอร์เฟส ซึ่งในเอสคิวแอลวินโดวส์จะเรียกว่าวินโดวส์ออพเจ็กต์ (Windows Object) ไปตัวหนึ่งเอสคิวแอลวินโดวส์ก็จะทำการใส่โค๊ดของวินโดวส์ออพเจ็กต์นั้นลงในซอร์ทโค๊ดของแอพพลิเคชั่นของเราให้อย่างอัตโนมัติ อย่างไรก็ตามโค๊ดที่เอสคิวแอลวินโดวส์ใส่ให้นั้นเป็นเพียงโครงสร้างที่แสดงการมีอยู่ของวินโดวส์ออพเจ็กต์นี้บนฟอร์มเท่านั้น จะยังไม่มีการใส่โค๊ดที่ทำหน้าที่ควบคุมการทำงานของวินโดวส์ออพเจ็กต์นั้น การใส่โค๊ดควบคุมจะเป็นหน้าที่ของโปรแกรมเมอร์เอง

ต่อไปนี้ก็จะเป็นรูปอธิบายตัวอย่างหน้าต่างที่ได้วางออพเจ็กต์ต่างๆลงไปเรียบร้อยแล้ว ในรูปที่ 3 สังเกตว่าชื่อ (title) ที่ปรากฏบนแต่ละออพเจ็กต์จะเป็น untitled, Label, noname เป็นต้น ชื่อเหล่านี้โปรแกรมเมอร์สามารถเปลี่ยนเป็นชื่ออื่นที่มีความหมายตามในแอพพลิเคชั่นที่จะสร้างได้ ส่วนในรูปที่ 4 จะแสดงการใส่โค๊ดให้อัตโนมัติในเอาร์ทไลน์เนอร์วินโดวส์เมื่อมีการวางออพเจ็กต์ลงบนฟอร์ม (โค๊ดในแถบสีในเอาร์ทไลเนอร์) ส่วนของโค๊ดตรงนี้เองที่โปรแกรมเมอร์จะทำการใส่โค๊ดเพิ่มเข้าไปเพื่อการควบคุม สำหรับเรื่องการใส่โค๊ดควบคุมนั้นจำเป็นต้องเข้าใจถึงภาษาเอสเอแอลเสียก่อนดังนั้นจึงยกไปกล่าวในช่วงหลังจากที่เราได้ทำความเข้าใจกับภาษาเอสเอแอลแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3 แสดงแบล็งค์ฟอร์มที่โปรแกรมเมอร์ทำการใส่วินโดวส์ออพเจ็กต์แล้ว



รูปที่ 4 แสดงโค๊ดที่ใส่ให้อัตโนมัติเมื่อมีการเพิ่มวินโดวส์ออพเจ็กต์รัวใหม่ในแบล็งค์ฟอร์ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2.2 ภาษาเอสเอแอล

มีอยู่ 2 อย่างที่ภาษาเอสเอแอลต่างจากภาษาแบบโพรซีเดอร์รอลโดยทั่วไป อย่างแรกก็คือภาษาเอสเอแอลเป็นภาษาที่สนับสนุนการโปรแกรมแบบออพเจ็กต์โอเรียนเต็ด อย่างที่สองคือภาษาเอเอสแอลใช้แนว ความคิดแบบอีเวนท์ดริเวนท์โปรแกรมมิ่ง คือเอสเอแอลจะมองโอเปอร์เรชั่นทุกตัวในแอพพลิเคชั่นเป็น อีเวนท์ที่เกิดกับออพเจ็กต์ ออพเจ็กต์ในความหมายของภาษาเอสเอแอลจะเป็นโมดูลของโปรแกรมที่อาจจะ บรรจุข้อมูลและสามารถรับแมสเสจและปฏิบัติโพรซีเดอร์ที่โปรแกรมไว้ได้

ตัวอย่างเช่นท็อปแลเวิลฟอร์มวินโดวส์ (Top-Level Form Window) เป็นออพเจ็กต์ มันสามารถบรรจุเซ็กชั่น (Section) วินโดวส์วาริเอเบิ้ล (Window Variables) สำหรับใช้ประกาศตัวแปรของวินโดวส์มันได้ และยังมีเซ็กชั่นคอนเทินท์ (Contents) ที่ใช้ในการประกาศชายด์ออพเจ็กต์ (Child Object) ที่ประกอบอยู่ในตัวมันได้ นอกจากนี้ท็อปแลเวิลฟอร์มวินโดวส์ยังมีเซ็กชั่นฟังก์ชั่น (Function) ที่ใช้ในการประกาศฟังก์ชั่นที่ใช้ภายในวินโดวส์ได้ สุดท้ายมันก็ยังมีแมสเสจแฮ็คชั่นเซ็กชั่น ในเซ็กชั่นนี้เราสามารถใส่โพรซีเดอร์ที่เราต้องการให้ปฏิบัติเมื่อวินโดวส์นี้รับแมสเสจบางตัวได้

คาต้าฟิลด์ก็เป็นออพเจ็กต์ มันบรรจุคาต้า -- ค่าที่ถูกให้กับมันโดยมีชนิดที่เหมาะสม -- และมันก็ยังมีแมสเสจแฮ็คชั่นเซ็กชั่นไว้สำหรับดักแมสเสจเหมือนท็อปแลเวิลวินโดวส์เช่นกัน

แต่แบล็กกราวน์เท็กท์ (Background Test), เส้น (Line), กรอบ (Frame), และเมนูไอเทม (Menu Item)ไม่เป็นออพเจ็กต์เพราะพวกมันต่างก็ขาดคุณสมบัติของออพเจ็กต์ในความหมายของภาษาเอสเอแอล ที่ออพเจ็กต์ต้องสามารถรับแมสเสจและปฏิบัติโพรซีเดอร์ โดยที่อาจจะบรรจุคาต้าไว้ด้วยก็ได้

สัญญาณที่ไปทริกโพรซีเดอร์ในเอสคิวแอลวินโดวส์ออพเจ็กต์จะถูกเรียกว่า แมสเสจ (message) เอสคิวแอลวินโดวส์ได้กำหนดแมสเสจของระบบที่แตกต่างกันไว้ประมาณ 40 แมสเสจ แต่ละแมสเสจจะสัมพันธ์กับเหตุการณ์หรืออีเวนท์บางเหตุการณ์ ตัวอย่างเช่น "ผู้ใช้กดปุ่มเมาส์ที่พุชส์บัททอนน์ (push button)" ก็เป็นอีเวนท์หนึ่ง เมื่ออีเวนท์หนึ่งซึ่งเอสคิวแอลวินโดวส์ได้กำหนดแมสเสจที่สัมพันธ์กับมันเกิดขึ้น เอสคิวแอลวินโดวส์ ก็จะส่งแมสเสจที่สัมพันธ์กับอีเวนท์นั้นไปยังแมสเสจแฮ็คชั่นเซ็กชั่นของออพเจ็กต์ที่ได้รับผลกระทบ

ในกรณีการพุชส์บัททอนน์ แมสเสจชื่อ เอสเอเอ็มคลิกค์ (SAM_Click, SAM stands *for SQLWindows Application Message*) จะถูกส่งไปยังแมสเสจแฮ็คชั่นเซ็กชั่นของพุชส์บัททอนน์นั้น (หรือกล่าวอย่างสั้นๆว่าแมสเสจถูกส่งไปยังพุชส์บัททอนส์นั้น) ถ้าเราใส่โค๊ดอะไรไว้ที่ที่รับแมสเสจอันนั้น โค๊ดดังกล่าวก็จะถูกปฏิบัติ ถ้าเราไม่ได้ใส่อะไรไว้ ก็จะไม่มีอะไรเกิดขึ้น

สมมุติว่ามีไดอะล็อกค์บ็อกส์ที่บรรจุพุชส์บัททอนน์โอเค (OK push button) ไว้ แล้วเมื่อไดอะล็อกค์นี้ถูกเปิดขึ้นและพุชส์บัททอนน์ถูกสร้าง (นี้เป็นอีเวนท์หนึ่ง) แล้วเราต้องการทำให้มันเป็นสีเทาหรือดิสเอเบิส (disable) มันไว้เพื่อไม่ให้ผู้ใช้สามารถกดปุ่มมันได้ก่อนที่เขาจะป้อนข้อมูลลงในฟิลด์บางฟิลด์ ซึ่งถ้าป้อนแล้วก็จะมีโค๊ดบางตัวทำให้ปุ่มนั้นสามารถกลับมาให้กดได้อีกครั้ง เมื่อบัททอนน์ถูกกด (นี่ก็เป็นอีเวนท์หนึ่ง) เราต้องการใช้ทำการให้ค่า (assign) เท็กส์ (Text) บางตัวให้กับคาต้าฟิลด์ตัวหนึ่งและจากนั้นก็ทำการปิด (close) หรือทำลาย (destroy) ไดอะล็อกส์นั้น

โค๊ดในแมสเสจแฮ็คชั่นเซ็กชั่นของพุชส์บัททอนส์อาจจะเป็นดังโค๊ดตัวอย่างต่อไปนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

◆ On SAM_Create

◊ Call SalDisableWindows(pbOK)

◆ On SAM_Click

◊ Set dfData = sString

◊ Call SalEndDialog(dbDialog, 0)

โค้ดชุดนี้จะตอบสนองกับแมสเสจสองอันคือ เอสเอเอ็มครีเอท (SAM_Create) และเอสเอเอ็มคลิกค์(SAM_Click), และจะทำงานแตกต่างกันตามแต่ละแมสเสจ

6.2.2.1 ชนิดของข้อมูลในภาษาเอสเอแอล

เอสเอแอลมีชนิดของข้อมูลจำนวนไม่มากให้ใช้ไม่มาก และมีบางตัวดูไม่ค่อยจะเป็นชนิดข้อมูลที่คุ้นเคยสักเท่าใด ซึ่งเราจะสรุปชนิดข้อมูลของภาษาเอสเอแอลไว้ดังนี้

บูอลีน (Boolean)	เป็นชนิดข้อมูลที่สามารถมีค่าได้เพียงสองค่าเท่านั้นคือ 1 (ถูก (TRUE)) หรือ 0 (ผิด (FALSE)) ค่าคงที่เหล่านี้ได้ถูกกำหนดไว้แล้วโดยเอสคิวแอลวินโดวส์
เดท/ไทม์ (Date/time)	สำหรับเดท/ไทม์ ฟิลด์ใดที่ถูกกำหนดให้มีชนิดเป็นเดท/ไทม์จะสามารถรับวันที่และเวลาที่ถูกป้อนในแบบที่เป็นมาตรฐานได้เช่นแบบ ไอเอสโอ (ISO), ยูโรเปี้ยน และ แจแปนนิสอินดัสเตรียลไทม์ (European, and Japanese Industrial Time)
ลองสตริง (Long string)	สำหรับข้อมูลแบบไบนารี่หรือข้อมูลชนิดลองวาชาร์ไทป์ (LONG VARCHAR type) ของเอสคิวแอลเบส ใช้ชนิดข้อมูลแบบนี้สำหรับบิทแมพ (bitmap) และเอสคิวแอลวินโดวส์ออพเจ็กต์หรือสตริงวาริเอเบิล (string variable) ที่สัมพันธ์กับคอลัมน์ที่เป็นชนิดลอง (LONG type)
นัมเบอร์ (Number)	สำหรับข้อมูลตัวเลขที่มีจำนวนหลักสูงได้ถึง 18 หลักรวมจุดทศนิยม
สตริง (String)	สำหรับข้อมูลตัวอักษรหรือข้อมูลบิทแมพที่มีความยาวไม่แน่นอน

นอกจากชนิดของข้อมูลทั่วไปแล้ว เอสเอแอลยังมีข้อมูลประเภทที่ใช้กับวินโดว์, ไฟล์ และเอสคิวแอลคาต้าเบสคอนเน็กชั่น (SQL database connection) ด้วยเพื่อใช้ในการจัดการกับสิ่งเหล่านี้ ตัวชี้หรือไอเดนท์ติไฟเออร์ (identifier) ที่มีชนิดของข้อมูลเป็นชนิดพิเศษเหล่านี้จะถูกเรียกว่า แฮนด์เดิล (handle) คาต้าไทพ์พิเศษที่กล่าวนี้มีดังต่อไปนี้

วินโดว์แฮนด์เดิล (Window handle)	ใช้ในการบ่งชี้ถึงวินโดวส์หรือท๊อปแลเวิลหรือชายด์
ไฟล์แฮนด์เดิล (File handle)	ใช้ในการบ่งชี้ถึงไฟล์ในการเปิดใช้ไฟล์ของคอสส์
เอสคิวแอลแฮนด์เดิล (SQL handle)	ใช้ในการบ่งชี้ถึงการติดต่อกับเอสคิวแอลคาต้าเบส (SQL database connection)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชนิดข้อมูลทั้งหมดยกเว้นลองสตริง สามารถประกาศให้เป็นแบบรีซีฟ (receive) ได้ ซึ่งถ้าเราประกาศให้เป็นชนิดรีซีฟด้วยแล้ว ก็จะทำให้ตัวแปรนั้นมีลักษณะเป็นตัวชี้ (pointer) ใช้ในการส่งผ่านพารามิเตอร์ระหว่างฟังก์ชันแบบคอลบายเรฟเฟอร์เรนส์ (call by reference)

6.2.2.2 เสตทเมนท์ (Statement) ของภาษาเอสเอแอล

ในช่วงนี้เราจะมากล่าวถึงเสตทเมนท์ของภาษาเอสเอแอลซึ่งมีดังต่อไปนี้

On

เสตทเมนท์ออน (On) นี้ใช้ในการดักจัดแมสเสจ ดังนั้นก็จะเป็นการทริกเกอร์ (trigger) โพรซีเดอร์ที่อยู่ภายใต้หรือเป็นซับเซ็กชั่นของมัน รูปแบบการใช้ประโยคจะเป็น

On *message*

เช่นโค๊ดดังต่อไปนี้

ที่คอยดักจัดแมสเสจเอสเอ็มคลิกเพื่อให้ปฏิบัติฟังก์ชันในการทำลายหรือปิดวินโดวส์นั้น

◆ On SAM_Click

◊ Call SalDestroyWindow(hWndForm)

Set

เสตทเมนท์เซ็ท (Set) นี้ใช้เป็นเสตทเมนท์ให้ค่าหรือแอสไซน์เมนท์เสตทเมนท์ (Assignment statement) มีรูปแบบการใช้ดังนี้

Set *variable* = *expression*

โดยที่ชนิดข้อมูลของวาริเอเบิ้ลกับเอ็กซ์เพรสชั่นจะต้องเหมือนกัน

ตัวอย่างเซ็ทเสตทเมนท์สามารถแสดงได้ดังนี้

◊ Set hWndTemp = hWndNULL

◊ Set nIndex = 0

◊ Set saTreates[nIndex] = ''

◊ Set hWndNew = SalCreateWindow(frmNewForm, hWndNULL)

Call

เสตทเมนท์คอล (Call) นี้ใช้ในการเรียกฟังก์ชัน ดังตัวอย่างต่อไปนี้

◊ Call SalCreateWindow(frmNewForm, hWndNULL)

◊ Call SqlImmediate('rollback')

If, Else and Else If

เสตทเมนท์อีฟ (If) เอวส (Else) และ เอวสอีฟ (Else If) ทำให้เราสามารถทำการเลือกการทำงานตามเงื่อนไขที่เราตั้งไว้ได้ รูปแบบการใช้จะเป็นดังนี้คือ

If *expressionA*

expressionI

และ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

If *expressionA*

expression1

Else *expression2*

และ

If *expressionA*

expression1

Else If *expressionB*

expression2

Else If *expression C*

expression3

Else *expression4*

Loop and Break

เสตทเมนท์ลูป (Loop) และเบลกค์ (Break) ใช้ในการปฏิบัติตามกลุ่มของเสตทเมนท์ที่อยู่ภายใต้มันจนกระทั่งพบเสตทเมนท์เบลกค์หรือรีเทอนน์ (Return) จึงจะหยุดการวนซ้ำ รูปแบบการใช้จะเป็นดังนี้

Loop [*loopname*]

ตัวอย่างในการใช้ลูปเสตทเมนท์มีดังนี้

◊ Set nIndex = 0

◆ Loop

◆ If nIndex > 6

◊ Break

◊ Call SalStrUpper(saWeekdays[nIndex], saWeekdays[nIndex])

◊ Set nIndex = nIndex + 1

Select Case

เสตทเมนท์ซีเล็กท์เคส (Select Case) เป็นเสตทเมนท์ที่ทดสอบจำนวนเพื่อหาค่าที่ตรงกันกับค่าคงที่ที่กำหนดไว้แล้วทำการปฏิบัติตามโค๊ดที่เหมาะสมกับการมีค่าตรงกันนั้น ตัวอย่างเสตทเมนท์แบบนี้มีดังนี้

◆ Select Case nPartNumber

◊ Case 456

◊ Case 475

◆ Case 386

◊ Set dfPrice = 55

◊ Break

◆ Default

◊ Set dfPrice = 60

◊ Break

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Return

เสตทเมนท์รีเทอนน์ใช้ในการหนดการประมวลผลโค้ดภายใต้เสตทเมนท์ออน หรือเมนูไอเทม และจะส่งค่าและการควบคุมกลับไปยังฟังก์ชันที่เรียกมันหรือเอสคิวแอลวินโดวส์ เสตทเมนท์รีเทอนน์มีรูปแบบดังนี้

Return *expression*

ตัวอย่างของเสตทเมนท์แบบนี้มีดังนี้

```
◆ On SAM_Validate
   ◆ If Not SalIsValidNumber( hWndItem )
      ◊ Call SalMessageBox( 'Not a valid check-in weight', 'Spa',
         MB_Ok | MB_IconStop )
      ◊ Return VALIDATE_Cancel
```

While

เสตทเมนท์ไวล์ (While) นี้คล้ายกันและบางทีสามารถใช้แทนกันได้กับเสตทเมนท์ลูป เพราะต่างก็ใช้ในการวนซ้ำทำงานตามกลุ่มของเสตทเมนท์เช่นกัน เสตทเมนท์ไวล์มีรูปแบบดังนี้

While *expressionA*

expression1

ตัวอย่างการใช้เสตทเมนท์ไวล์มีดังนี้

```
◊ Set nIndex = 0
◆ While nIndex < 7
   ◊ Call SalStrUpper( saWeekdays[ nIndex ], saWeekdays[ nIndex ] )
   ◊ Set nIndex = nIndex + 1
```

จากที่ได้กล่าวมาแล้วก็ทำให้เราเข้าใจถึงตัวภาษาและเอ็นไวรอนเมนท์ของเอสคิวแอลวินโดวส์มาระดับหนึ่งแล้ว แต่สำหรับการพัฒนาแอพพลิเคชั่นจริงๆ โปรแกรมเมอร์ยังจำเป็นที่จะต้องรู้ถึงฟังก์ชั่นที่จะใช้งานต่างๆ อีกหลายตัว ซึ่งฟังก์ชันเหล่านี้เอสคิวแอลวินโดวส์ได้มีไว้ให้เรียบร้อยพร้อมที่จะใช้งานแล้ว ดังนั้นจึงจำเป็นต้องอ้างอิงกับหนังสืออ้างอิงของทูลชุดนี้ซึ่งมีรายชื่อไว้ตอนท้ายของรายงานชุดนี้แล้ว ถ้าผู้อ่านต้องการรายละเอียดทั้งหมดก็สามารถหาอ่านได้ตามรายชื่อหนังสือดังกล่าวเช่นกัน

6.2.3 ระบบการลงทะเบียนที่สร้างด้วยแอสคิวแอลวินโดวส์

จากที่เราได้ทำการออกแบบระบบการลงทะเบียนไปแล้วในบทที่ 3 มาถึงตอนนี้เราจะนำแบบที่เราได้ออกแบบไว้แล้วนั้นมาสร้างเป็นโปรแกรมที่วิ่งบนเครื่องกันจริงๆ เรามาเริ่มที่ขั้นตอนในการสร้างกันเลยดังต่อไปนี้

การสร้างระบบลงทะเบียนนี้เราเริ่มต้นที่การสร้างดาต้าเบสที่จะใช้ทดลองกับโปรแกรมที่จะทำขึ้นโดยการสร้างสคริปไฟล์ที่บรรจุภาษาเอสคิวแอลที่สร้างฐานข้อมูลและใส่ข้อมูลทดสอบบางส่วนให้กับเรา สำหรับสคริปไฟล์ที่ว่านี้มีดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
CREATE TABLE SUBJECT
(SUB_NO INT NOT NULL,
 SUB_NAME CHAR(30),
 SUB_CREDIT INT,
 PRIMARY KEY (SUB_NO))
/

CREATE UNIQUE INDEX SUB_IDX ON SUBJECT (SUB_NO)
/

CREATE TABLE REGISTER
(SUB_NO INT NOT NULL,
 STU_NO INT NOT NULL,
 REG_DATE DATE,
 PRIMARY KEY (SUB_NO,STU_NO),
 FOREIGN KEY (SUB_NO) REFERENCES SUBJECT,
 FOREIGN KEY (STU_NO) REFERENCES STUDENT)
/

CREATE UNIQUE INDEX REG_IDX ON REGISTER (SUB_NO,STU_NO)
/

CREATE TABLE TEACH
(SUB_NO INT NOT NULL,
 TEA_NO INT NOT NULL,
 PRIMARY KEY (SUB_NO,TEA_NO),
 FOREIGN KEY (SUB_NO) REFERENCES SUBJECT,
 FOREIGN KEY (TEA_NO) REFERENCES TEACHER)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/

CREATE UNIQUE INDEX TEAC_IDX ON TEACH (SUB_NO, TEA_NO)
/

INSERT INTO STUDENT (STU_NO, STU_NAME, STU_SURNAME,
STU_MAJOR)
VALUES (:1, :2, :3, :4)
\
$DATATYPES NUMERIC,CHARACTER,CHARACTER,CHARACTER
34101036,Kletikul,Jeerinaitanakit,Computer Engineering
34101039,Kraisit,Vittinanon,Computer Engineering
34102115,Natapong,Monkolnavin,Computer Engineering
34102098,Somchai,Srirungtong,Electronic Engineering
35104321,Sirichai,Vongvitaya,Control Engineering
35134523,Kanokvan,Puranon,Telecommunication Engineering
/

INSERT INTO TEACHER (TEA_NO, TEA_NAME, TEA_SURNAME)
VALUES (:1, :2, :3)
\
$DATATYPES NUMERIC,CHARACTER,CHARACTER
1,Somkiet,Supradej
2,Tavil,Gingtong
3,Supramit,Jittayasotorn
4,Manus,Sungvolsilp
5,Vorawat,Limpoka
6,Kopchai,Dejharn
/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
INSERT INTO SYSADM.SUBJECT VALUES(:1,:2,:3)
\
$datatypes NUMERIC,CHARACTER,NUMERIC
1001002,"Mathematics I",6
1021101,"Engineering Laboratory I",1
1041101,"Electrical Circuit Analysis",3
1050001,"Mechanics",3
3010020,"English for Engineering I",3
3020001,"Introduction to Japanese I",3
3150015,"General Psychology",2
/
CREATE TABLE SYSADM.TEACH (SUB_NO INTEGER NOT NULL,
TEA_NO INTEGER NOT NULL) PCTFREE 10
/
INSERT INTO SYSADM.TEACH VALUES(:1,:2)
\
$datatypes NUMERIC,NUMERIC
1001002,1
1001002,2
1001002,6
1001002,19
1001002,20
1001002,29
1021101,2
1021101,8
1021101,13
1021101,18
1021101,21
1041101,4
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

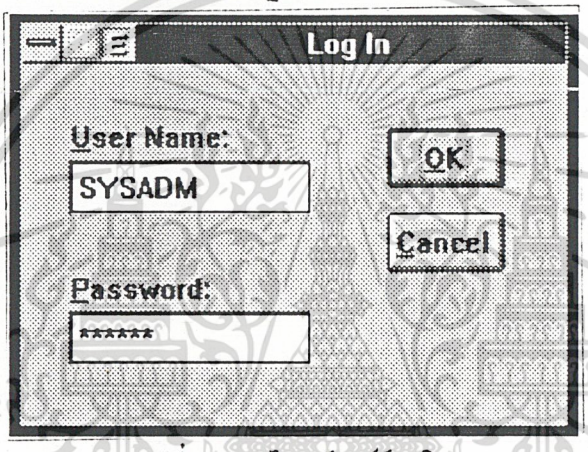
```
1041101,9
1041101,11
1041101,14
1041101,22
1050001,3
1050001,4
1050001,23
1050001,26
1050001,29
3010020,5
3010020,12
3010020,28
3020001,7
3020001,17
3020001,25
3020001,27
3020001,30
3150015,10
3150015,15
3150015,16
3150015,17
3150015,24
/
COMMIT
/
```

เมื่อเราได้ทำการสร้างฐานข้อมูลทดสอบขึ้นมาแล้วเราก็ทำการสร้างระบบการลงทะเบียนตามออพเจ็กต์โมเดล, เสตทไดอะแกรม, และคาต้าโฟลว์ไดอะแกรมที่ได้ออกแบบ ต่อไปนี้จะได้อธิบายระบบลงทะเบียนที่ได้ทำการสร้างจากโมเดลที่ได้ออกแบบไว้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ก่อนอื่นเราอยากจะกล่าวขออภัยที่จะบอกว่า บางส่วนของระบบที่ได้ทำการสร้างขึ้นจริงได้ถูกเพิ่มเติมในภายหลังเพื่อเพิ่มความสามารถบางประการให้แก่ระบบ ดังนั้นจึงไม่ได้ปรากฏบนโมเดล ส่วนที่เราเพิ่มเข้าไปนี้เป็นส่วนแรกที่เป็นส่วนตรวจเช็คหรือล็อกอิน (Login) เข้าสู่ระบบในรูปที่ 5 และส่วนที่ใช้เป็นตัวกลางในการเรียกโ ปรแกรมที่ประกอบอื่นๆ ดังในรูปที่ 6 และส่วนอื่นๆในรูปที่ 7 และ 8 ส่วนระบบลงทะเบียนที่เป็นส่วนของคาค้าเอ็นทรี่จริงๆที่ได้ออกแบบไว้ตามแบบในบทที่ 3 จะเห็นเป็นหน้าต่างในรูปที่ 9 ส่วนคาค้าเอ็นทรี่จริงๆ นี้จะเป็นไ ปตามแบบที่ออกไว้ในบทที่ 3 เราจะขอเริ่มต้นอธิบายจากรูปประกอบเลยก่อน แล้วเราจะไปดูที่ซอร์ทโค๊ดในภายหลัง

รูปแรกนี้จะเป็นไดอะล็อกซ์ที่ให้ผู้ใช้ทำการป้อนชื่อและรหัสผ่านเพื่อติดต่อเข้าสู่ระบบ เมื่อผู้ใช้ทำการป้อนชื่อและรหัสผ่านที่ถูกต้องและกดปุ่ม OK แล้วก็จะสามารถเข้าไปทำงานต่อไปได้ แต่เข้าชื่อหรือรหัสผ่านผิดไปก็จะมีข้อความเตือนปรากฏขึ้นมา

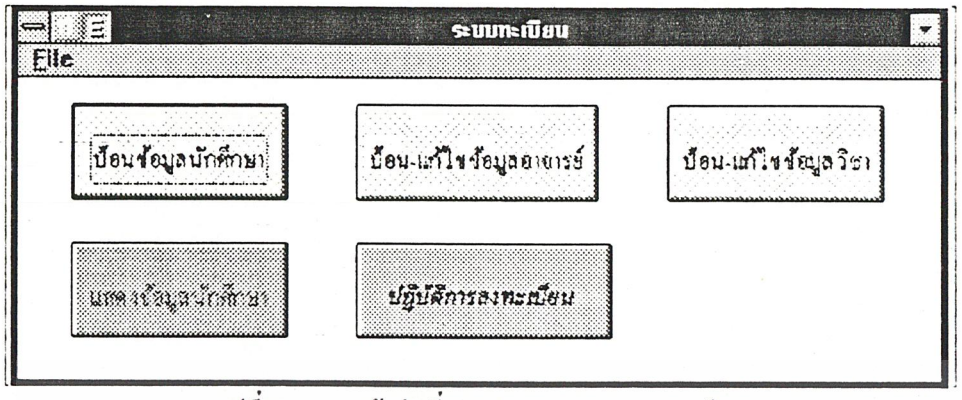


รูปที่ 5 แสดงไดอะล็อกซ์ล็อกอิน

เมื่อผ่านจากขั้นตอนที่แล้วมาได้ระบบก็จะขึ้นหน้าต่างที่เป็นเสมือนเมนูในรูปที่ 6 ให้เลือกเข้าไปในส่วนการทำงานย่อยต่อไป ซึ่งการทำงานในระดับย่อยนั้นมีดังนี้

1. ป้อนข้อมูลนักศึกษา (ไม่ได้อยู่ในโมเดลแต่ทำเพิ่มขึ้นเพื่อใช้เข้าไปป้อนข้อมูลนักศึกษาเพิ่มเติม) แสดงหน้าต่างเมื่อเข้าสู่การทำงานนี้ในรูปที่ 7
2. ป้อนแก้ไขข้อมูลอาจารย์ (ไม่ได้อยู่ในโมเดล แต่ทำเพิ่มขึ้นเพื่อใช้เข้าไปป้อนและแก้ไขข้อมูลเพิ่มเติม)
3. ป้อนแก้ไขข้อมูลวิชา (ไม่ได้อยู่ในโมเดล แต่ทำเพิ่มขึ้นเพื่อใช้เข้าไปป้อนและแก้ไขข้อมูลเพิ่มเติม)
4. แสดข้อมูลนักศึกษา (ไม่ได้อยู่ในโมเดล แต่ทำเพิ่มขึ้นเพื่อใช้เข้าไปค้นหาข้อมูลนักศึกษา) แสดงหน้าต่างเมื่อเข้าสู่การทำงานนี้ในรูปที่ 8
5. ปฏิบัติการลงทะเบียน (เป็นส่วนที่ตรงกับแบบที่ออกไว้ในบทที่ 3) แสดงหน้าต่างเมื่อเข้าสู่การทำงานนี้ในรูปที่ 9

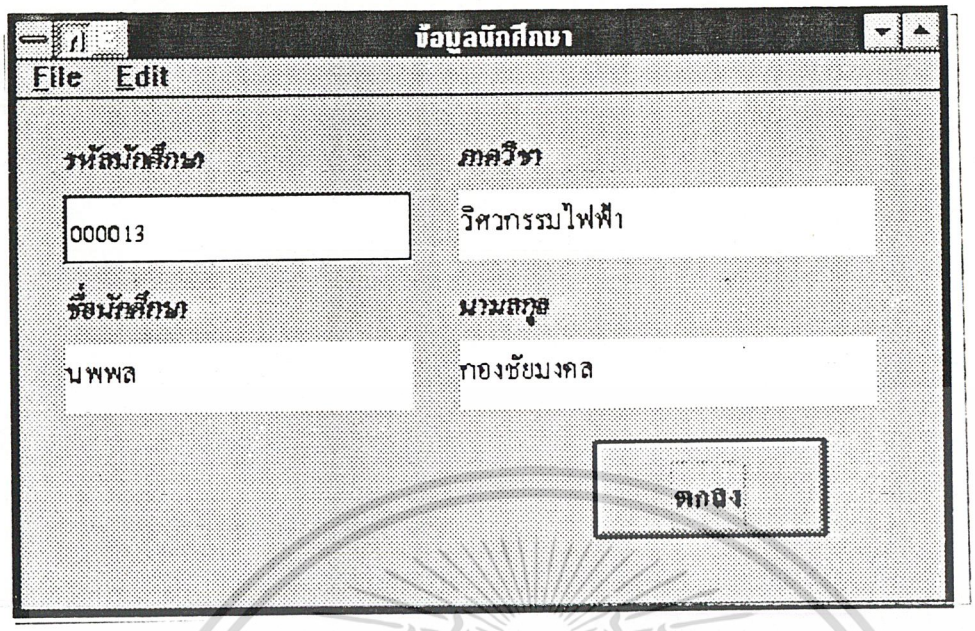
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6 แสดงหน้าต่างที่กลางของระบบกลางลงทะเบียน

รูปที่ 7 แสดงหน้าต่างป้อนข้อมูลนักศึกษา

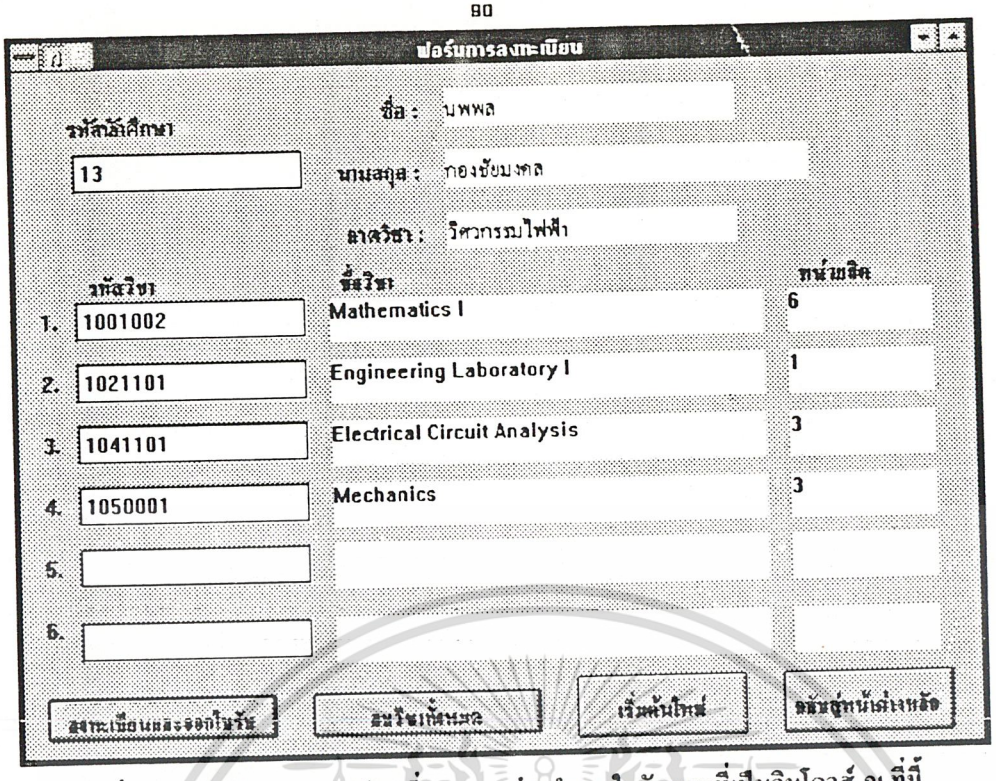
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8 แสดงหน้าต่างค้นหาข้อมูลนักศึกษา

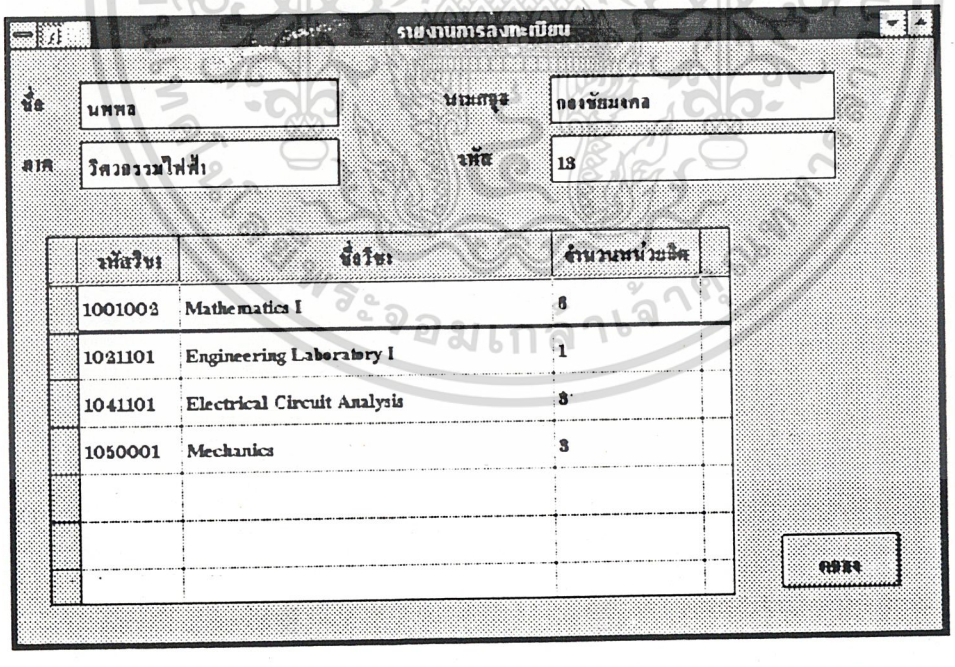
ในรูปที่ 9 นี้แสดงหน้าต่างที่ใช้ในการรับการลงทะเบียน เปรียบเทียบกับสเตทไดอะแกรมของคาต้าเอ็นทรี่ในบทที่ 3 ตอนแรกคาต้าเอ็นทรี่จะอยู่ที่เสตท clear Screen และทำโอเปอร์เรชั่น do: display new register window ซึ่งก็คือหน้าจอในรูปที่ 9 แบบที่ยังไม่มีการใส่ข้อมูลใดๆเลย เมื่อผู้ใช้ใส่รหัสของนักศึกษาลงในช่องรหัสนักศึกษาเรียบร้อย โปรแกรมก็จะเปลี่ยนมาที่เสตท Student Validation เพื่อตรวจสอบข้อมูลของนักศึกษาตามรหัสนั้นถ้านักศึกษานั้นถูกต้องก็จะเปลี่ยนเสตทไปเป็น Input Subject เพื่อรับข้อมูลวิชาที่นักศึกษาคนนั้นต้องการจะลงทะเบียน เมื่อมีการป้อนรหัสวิชาเมื่อใดก็ระบบก็จะทำการตรวจสอบข้อมูลวิชานั้นๆแล้วแสดง ขึ้นมาให้ผู้ใช้ทราบด้วย พอเสร็จแล้วก็จะการสั่งให้ออกใบรับโดยผู้ใช้จะกดปุ่มที่เขียนว่า "ลงทะเบียนและออกใบรับ" ระบบก็จะเปลี่ยนเสตทไปเป็น Update DB เพื่อทำการอัพเดทฐานข้อมูลและไปทำการพิมพ์ต่อไป สำหรับรูปแบบใบรับการพิมพ์จะเป็นในลักษณะที่คล้ายกับหน้าต่างในรูปที่ 9 ซึ่งเราทำการจำลองข้อมูลในรายงานให้อยู่บนหน้าต่างบนวินโดวส์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 9 แสดงใบรับการลงทะเบียนที่จะออก แต่ถูกจำลองในลักษณะที่เป็นวินโดวส์ ณ.ที่นี้

หลังจากที่เราได้เห็นยูสเซอร์อินเตอร์เฟสของระบบลงทะเบียนแล้ว ต่อไปนี้เราก็จะแสดงซอร์ทโค๊ดของโปรแกรมระบบการลงทะเบียนที่ได้ทำขึ้นมา



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2.4 ฟีเจอร์ (Feature) ของเอสคิวแอลวินโดวส์

ภายใต้หัวข้อนี้เราจะมาพูดกันถึงเรื่องฟีเจอร์ที่เอสคิวแอลวินโดวส์มี เพื่อจะใช้ในการเปรียบเทียบกับทูลวิชวลในภายหลังต่อไป สำหรับฟีเจอร์ที่เราจะนำมาของเอสคิวแอลวินโดวส์ตัวนี้เป็นฟีเจอร์ของเอสคิวแอลวินโดวส์เวอร์ชัน 5 ซึ่งเป็นเวอร์ชันล่าสุดในช่วงนี้

6.2.4.1 ในด้านความง่ายในการพัฒนา

เอสคิวแอลวินโดวส์สามารถที่จะสร้างคาต้าเบสแอพพลิเคชั่นทั่วๆไปได้โดยไม่ต้องมีการเขียนตัวโปรแกรมเองเลย เอสคิวแอลวินโดว์จะมีกลไกที่ออโตเมท (automate) การสร้างแอพพลิเคชั่นเหล่านั้นแล้วผู้สร้างเพียงแต่เลือกและตัดสินใจตัวเลือกต่างๆที่เอสคิวแอลวินโดวส์จะจัดการให้ อย่างไรก็ตามไม่ใช่ว่าผู้สร้างจะไม่สามารถเข้าไปแก้ไขในส่วนที่ตนเองต้องการเฉพาะได้ ในทางตรงกันข้ามเอสคิวแอลวินโดวส์ยิ่งสนับสนุนในการแก้ไขของผู้สร้างให้เป็นไปได้อย่างสะดวกและง่ายด้วย

6.2.4.2 ในด้านเอ็นไวรอนเมนท์ในการโปรแกรม

เอสคิวแอลวินโดวส์ใช้เอ็นไวรอนเมนท์ในการพัฒนาที่ทำให้เราสามารถรวมกันระหว่างตัวดีไซน์และโค๊ดได้อย่างมีประสิทธิภาพไม่ว่าวิธีการดีไซน์จะเป็นแบบออพเจ็กต์โอเรียนเต็ดหรือไม่ก็ตาม เพราะเอสคิวแอลวินโดวส์ยืดหยุ่นพอที่จะให้เราเลือกใช้แนวความคิดในการโปรแกรมแบบออพเจ็กต์โอเรียนเต็ดหรือไม่ก็ได้ นอกจากนี้เอสคิวแอลวินโดวส์ยังช่วยใช้โปรแกรมเมอร์เลือกระดับในการดูโค๊ดได้ว่าต้องการเลือกดูทั้งหมดหรือเลือกดูที่ละส่วนก็ได้ เวลาโปรแกรมเมอร์ทำการโค๊ดดิ่งเอสคิวแอลวินโดวส์จะช่วยสนับสนุนในการโค๊ดพร้อมทั้งยังมีคอนเท็กส์เซนซิทิฟเฮวป์ (Context Sensitive Help) ช่วยให้คำอธิบายต่างๆเมื่อโปรแกรมเมอร์ต้องการ ภายหลังจากที่ทำแอพพลิเคชั่นออกมาได้แล้วเมื่อต้องการเปลี่ยนแปลงภาษาที่ใช้บนแอพพลิเคชั่นก็ยังสามารถแก้ไขได้โดยใช้ทูลมาทำการแปลดังนั้นการพัฒนาแอพพลิเคชั่นเพื่อนำไปขายต่อให้ประเทศอื่นๆจึงทำได้โดยสะดวก

6.2.4.3 ในด้านของออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่ง

ในด้านออพเจ็กต์โอเรียนเต็ดโปรแกรมมิ่ง เอสคิวแอลวินโดว์ทำได้ดีมาก โดยมีออพเจ็กต์ที่สร้างไว้แล้วเก็บไว้ให้โปรแกรมเมอร์นำมาใช้ได้เลย และยังเปิดโอกาสให้โปรแกรมเมอร์ทำการแก้ไขปรับให้ตรงตามความต้องการเฉพาะงานได้ และยังสนับสนุนออพเจ็กต์และคลาสส์ทั้งที่เป็นกราฟฟิกบนจอภาพและที่ไม่ใช้กราฟฟิกด้วย -- เช่นพากฟังก์ชันนอลคลาสส์ออพเจ็กต์และคลาสส์ เป็นต้น (ทูลอื่นๆบางตัวเช่นวิชวลเบสิกเวอร์ชั่น 3 สนับสนุนเพียงกราฟฟิกออพเจ็กต์) เอสคิวแอลวินโดวส์สนับสนุนการอินเฮียริแทนส์ทั้งแบบซิงเกิลและมัลติเพิลทำให้มีการรียูสองค์ประกอบต่างๆได้ดีมีประสิทธิภาพ และยังสามารถสร้างคลาสส์ที่ซ้อนไม่ใช้ผู้อื่นเข้าถึงได้อีกด้วย ออพเจ็กต์โอเรียนเต็ดของเอสคิวแอลวินโดวส์สนับสนุนการเลทไบด์ดิ้ง (late binding) และโพลีมอร์ฟิซึม (Polymorphism)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.2.4.4 ในด้านการเข้าถึงข้อมูล

เอสคิวแอลวินโดวส์เป็นทูลที่สามารถพัฒนาให้แอพพลิเคชั่นสามารถวิ่งกับเซอร์ฟเวอร์จากหลายผู้ผลิตได้ และในเวอร์ชันนี้ก็สนับสนุนการติดต่อผ่านโอดีบีซี (ODBC) ของไมโครซอฟท์ นอกจากนี้แอพพลิเคชั่นที่พัฒนาด้วยเอสคิวแอลวินโดวส์ยังสามารถติดต่อกับดาต้าเบสเซอร์ฟเวอร์ของหลายผู้ผลิตในเวลาเดียวกันก็ได้ เอสคิวแอลวินโดวส์มีไอโซเลชั่นแลเวิล (isolation level) หลายระดับที่ใช้จัดการล็อกกิ้งโมเดลแบบต่างๆได้ และเอสคิวแอลวินโดวส์สนับสนุนสตอร์โพรซีเดอร์ (stored procedures) และไดนามิกส์เอสคิวแอล (dynamic SQL)

6.2.5 สรุป

จากการที่เราได้นำเอสคิวแอลวินโดวส์มาใช้เป็นเครื่องมือในการพัฒนาดาต้าเบสแอพพลิเคชั่นแบบไคลเอ็นท์เซอร์ฟเวอร์ เราคิดว่าเอสคิวแอลวินโดวส์เป็นทูลที่เหมาะสมมากเนื่องจากมันให้เอ็นไวรอนเมนท์ที่ง่ายต่อการพัฒนาและยังมีทูลประกอบช่วยในการพัฒนาดาต้าเบสแอพพลิเคชั่นอีกหลายตัว และในแง่ของความเหมาะสมที่นำมาประยุกต์ใช้กับระเบียบวิธีการแบบออพเจ็กต์โอเรียนเต็ดที่ศึกษามานั้น ก็สามารถนำมาประยุกต์ใช้ได้อย่างมีประสิทธิภาพเนื่องจากตัวเอสคิวแอลวินโดวส์เองสนับสนุนออพเจ็กต์โอเรียนเต็ดอย่างสมบูรณ์ ดังนั้นขั้นตอนการแปลงตัวแบบที่ออกแบบมาเป็นโค๊ดของเอสคิวแอลวินโดวส์นั้นก็สามารถที่จะทำได้ง่าย

นอกจากนี้เอสคิวแอลวินโดวส์ยังมีทีมวินโดวส์ซึ่งเป็นทูลที่ช่วยในการจัดการโครงงาน (Project) ในการพัฒนาแอพพลิเคชั่นโดยใช้โปรแกรมเมอร์หลายๆคน ดังนั้นเอสคิวแอลวินโดวส์จึงสามารถพัฒนาแอพพลิเคชั่นที่มีขนาดใหญ่และซับซ้อนได้ดีกว่าทูลแบบที่ไม่มีส่วนช่วยในการจัดการ และยังส่งผลให้ซอฟต์แวร์ที่ได้พัฒนาขึ้นมีความน่าเชื่อถือ (Reliablity) มากกว่าด้วย

อย่างไรก็ตามจากการที่เราได้ใช้เอสคิวแอลวินโดวส์ในการพัฒนาเราคิดว่าเอสคิวแอลวินโดวส์ยังมีความยืดหยุ่นในการเขียนโค๊ดไม่มากเท่าที่ควร คือมีชนิดของข้อมูลที่จำกัดเกินไป คอนโทรลสตักเจอร์ (Control structure) ของภาษาเอสเอแอลยังไม่ยืดหยุ่นสักเท่าใด ส่งผลให้การโปรแกรมบางช่วงเป็นไปได้ลำบากกว่าที่ควรจะเป็น ในส่วนของการพิมพ์ซอร์ทโค๊ดออกก็ยังจำกัดโปรแกรมเมอร์ไม่สามารถเลือกจุดที่เข้าต้องการจะพิมพ์ได้ในครั้งเดียวถ้าเขาไม่ได้ต้องการพิมพ์ซอร์ทโค๊ดทั้งหมด และสิ่งที่ยังคิดว่าเอสคิวแอลวินโดวส์ยังทำได้ไม่ดีก็อยู่ในเรื่องของระบบเฮาป์ที่มีข้อมูลไม่มากเพียงพอ คือผู้โปรแกรมยังจำเป็นต้องมีหนังสือคู่มืออยู่ด้วยในการโปรแกรม ไม่สามารถพึ่งพากับข้อมูลในเฮาป์ได้สักเท่าใด สำหรับเรื่องสุดท้ายก็คือการดีบักกิ้งเอสคิวแอลวินโดวส์ให้เอ็นไวรอนเมนท์ในการดีบักที่ไม่ค่อยจะสะดวกเท่าไหร่เพราะจำกัดในเรื่องการอีแวรูเอท (evaluate) ฟังก์ชันไม่สามารถทำได้ในต้นที่ทำการดีบักโปรแกรม ที่ให้โปรแกรมเมอร์ต้องใส่โค๊ดดักให้คำนวณลงในโปรแกรมเอง ซึ่งเป็นสิ่งที่ทำให้เสียเวลามากพอควรเลย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3 การพัฒนาแอพพลิเคชั่นโดยใช้ภาษาวิชวลเบสิก

วิชวลเบสิกเป็นคอมไพเลอร์(Compiler)ของบริษัทไมโครซอฟท์จำกัด มีจุดประสงค์หลักเพื่อการพัฒนาแอพพลิเคชั่นบนระบบการใช้งานที่เป็นวินโดวส์ สาเหตุที่เรียการโปรแกรมแบบนี้ว่าวิชวลเบสิกก็เพราะผู้พัฒนาสามารถมองเห็นตัวโปรแกรม และ รูปแบบการติดต่อกับผู้ใช้ได้ตั้งแต่ระยะเวลาของการออกแบบ ซึ่งข้อดีนี้ไม่ค่อยจะได้พบในภาษาอื่นๆบ่อยนัก ส่วนการประยุกต์ใช้งานของภาษาวิชวลเบสิกกับระบบงานฐานข้อมูล ทางบริษัทไมโครซอฟท์ได้ออกแบบให้ วิชวลเบสิกใช้ฐานข้อมูลของ ไมโครซอฟท์ แอ็คเซส (MicroSoft Access) เป็นฐานข้อมูลแบบมาตรฐานในการติดต่อ อย่างไรก็ตามวิชวลเบสิกยังสามารถติดต่อกับฐานข้อมูลได้อีกหลายชนิดซึ่งจะได้กล่าวต่อไปในบทนี้

หัวข้อการพัฒนาแอพพลิเคชั่นโดยใช้ภาษาวิชวลเบสิกนี้ ผู้จัดทำวิทยานิพนธ์จะขอสมมติว่าผู้อ่านได้มีความรู้ในการใช้งานวิชวลเบสิกเป็นพื้นฐานพอประมาณแล้ว ดังนั้นรายละเอียดที่จะกล่าวต่อไปนี้จะอธิบายเกี่ยวกับวิธีการติดต่อระหว่างตัวภาษาวิชวลเบสิก และ ส่วนของฐานข้อมูล โดยจะแบ่งออกเป็นหัวข้อดังนี้

- ODBC (Open Database Connectivity) คืออะไร
- แนะนำการใช้งานวิชวลเบสิก กับ ฐานข้อมูล
- การพัฒนาระบบลงทะเบียนโดยใช้ภาษาวิชวลเบสิก
- บทสรุปและวิจารณ์ความสามารถของวิชวลเบสิก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3.1 ODBC คืออะไร

OpenDatabase Connectivity คือวิธีการติดต่อและเข้าถึงจากแอพพลิเคชั่นสู่ระบบบริหารฐานข้อมูล (DBMS) โดยใช้ภาษา SQL เป็นมาตรฐานการเข้าถึงฐานข้อมูล ความสามารถในการต่อเชื่อมแบบนี้ทำให้แอพพลิเคชั่นสามารถเข้าถึงฐานข้อมูลได้หลายรูปแบบ ซึ่งทำให้ผู้พัฒนาโปรแกรมสามารถพัฒนาโปรแกรมไปได้โดยไม่ต้องทำการระบุชนิดของ DBMS

แต่เดิมนั้นการพัฒนาโปรแกรมประยุกต์ที่ใช้งานเกี่ยวกับฐานข้อมูล การเข้าใช้ฐานข้อมูลของโปรแกรมเหล่านี้จะทำผ่านการเรียกใช้ Embedded SQL ซึ่งในขณะนั้นวิถีทางแบบนี้ก็ดูจะไปได้ดีทีเดียว เพราะว่าตัวโปรแกรมสามารถทำการเปลี่ยนรูปแบบของระบบไม่ว่าจะเป็นทางด้านฮาร์ดแวร์ หรือ ซอฟท์แวร์ได้หลายรูปแบบ รวมทั้งระบบปฏิบัติการด้วย (โดยการคอมไพล์ใหม่ทุกครั้งที่มีการย้ายระบบ)

อย่างไรก็ตามในการพัฒนาโปรแกรมในระบบที่มีความแตกต่างกัน เช่น การเรียกใช้ข้อมูลของ ออราเคิล จาก ไมโครซอฟท์ เอ็กซ์เซล (Microsoft Excel) วิธีการเข้าถึงข้อมูลแบบเดิมนั้น จะต้องทำการ พรีคอมไพล์โค้ดของ เอ็กซ์เซล และ ออราเคิลโดยใช้ IBM Precompiler และ Oracle Precompilor ตามลำดับ ซึ่งจะเห็นว่าเป็นการยุ่งยากมากทีเดียว

วิธีการต่อเชื่อมแบบ ODBC จะให้ความสะดวกในการติดต่อข้อมูลมากกว่าวิธีการดั้งเดิม โดยการกำหนดมาตรฐานการต่อเชื่อมของข้อมูล (Data protocol, DBMS capability) และ แนวทางนี้ได้ทำให้เกิดความคิดที่จะสร้างไดร์เวอร์การติดต่อกับของฐานข้อมูลขึ้นมา (DLL)

6.3.1.1 ข้อดีของการติดต่อโดยใช้ ODBC

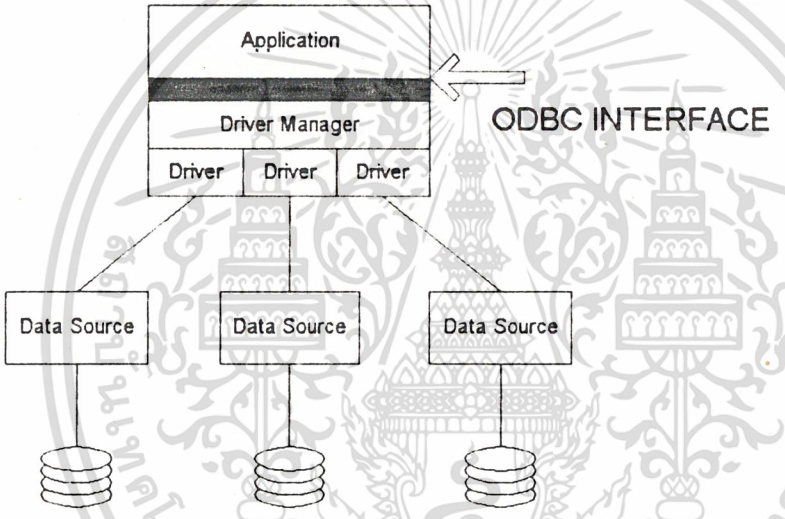
- ฟังก์ชั่นของ ODBC อนุญาตให้ แอพพลิเคชั่น ติดต่อกับ DBMS ได้โดยสะดวก (การทำคำสั่ง SQL และการรับผลลัพท์)
- ใช้ภาษา SQL ตามมาตรฐาน SQL CAE, X/Open และ SQL Access Group (SAG)
- มีการกำหนด การส่งกลับรหัสความผิดพลาด (Error Code) เป็นมาตรฐานเดียวกัน
- เป็นวิธีการมาตรฐานในการติดต่อกับ DBMS
- มีการกำหนด Data Type เป็นมาตรฐาน
- ชุดคำสั่ง SQL สามารถกำหนดได้แม้ในขณะ Runtime
- สามารถเขียนโปรแกรม ชุดเดียว แต่สามารถเข้าใช้ DMBS ได้หลายตัว
- ตัวโปรแกรมไม่ต้องรับผิดชอบในการดูแลการติดต่อข้อมูลกับ DBMS
- ค่าข้อมูลสามารถถูกส่งหรือรับได้ในรูปแบบที่สะดวกขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3.1.2 องค์ประกอบของ ODBC

สถาปัตยกรรมของ ODBC ประกอบด้วย 4 ส่วนสำคัญ

1. แอพพลิเคชั่น ทำหน้าที่ประมวลผลและเรียกใช้ ฟังก์ชั่นของ ODBC ตาม คำสั่งภาษา SQL พร้อมทั้งทำการรับผลลัพธ์ด้วย
2. ตัวจัดการไดร์เวอร์ (Driver Manager) ทำหน้าที่ load driver เชื่อมต่อกับแหล่งข้อมูล
3. ไดร์เวอร์ (Driver) ทำหน้าที่ประมวลผลการเรียกใช้ ฟังก์ชั่นของ ODBC ,ส่ง คำสั่ง SQL ไปสู่แหล่งข้อมูลที่ถูกต้อง และทำการส่งผลลัพธ์กลับให้แอพพลิเคชั่น และในบางครั้งไดร์เวอร์ จะทำหน้าที่แปลงคำสั่งที่ส่งมาให้อยู่ในรูปที่สนับสนุนโดย DBMS แต่ละชนิดอีกด้วย
4. แหล่งข้อมูล (Data Source) เป็นแหล่งข้อมูลที่ผู้ใช้ต้องการเข้าถึง



1.แอพพลิเคชั่น

ตัวโปรแกรมจะเรียกใช้การต่อเชื่อม ODBC ในการทำงานต่อไปนี้

1. ร้องขอการต่อเชื่อมกับแหล่งข้อมูล
2. ส่งคำสั่ง SQL สู่แหล่งข้อมูล
3. กำหนด พื้นที่การจัดเก็บและรูปแบบของข้อมูล ที่เป็นผลลัพธ์จาก SQL request
4. ร้องขอผลลัพธ์
5. ประมวลผลและจัดการกับข้อผิดพลาด
6. รายงานผลให้กับผู้ใช้ (ถ้าจำเป็น)
7. ร้องขอการ Commit หรือ Roll Back สำหรับควบคุมการประมวลผล Transaction
8. ยกเลิกการติดต่อกับแหล่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.ตัวจัดการไดร์เวอร์

ตัวจัดการไดร์เวอร์ คือ DLL และไลบรารี่อื่นๆ หน้าที่หลักของก็คือ การ Load Driver ส่วนหน้าที่อื่นๆ ก็มีดังนี้

2.1 เรียกใช้ไฟล์ODBC.INI เพื่อกำหนดชื่อของแหล่งข้อมูล(data source name) ให้กับไดร์เวอร์ DLL

2.2 ทำการประมวลผลการเริ่มต้นการเชื่อมต่อของ ODBC

2.3 เป็นจุดต่อเชื่อมระหว่าง ฟังก์ชั่นของ ODBC กับ driver แต่ละตัว

2.4 ทำการตรวจสอบพารามิเตอร์และลำดับการเรียกใช้ ODBC

3.ไดร์เวอร์

ไดร์เวอร์คือ DLL ที่สร้างฟังก์ชั่นของ ODBC และทำการโต้ตอบกับแหล่งข้อมูล ไดร์เวอร์ทำการตอบสนองการเรียกใช้ฟังก์ชั่นของ ODBC โดยจะทำงานต่อไปนี้

3.1 สร้างการต่อเชื่อมกับแหล่งข้อมูล

3.2 ส่งคำขอร้องให้กับแหล่งข้อมูล

3.3 แปลงข้อมูลจากรูปแบบหนึ่งสู่อีกรูปแบบหนึ่ง

3.4 ส่งผลลัพธ์กลับให้ แอพพลิเคชั่น

3.5 จัดการส่งข้อมูลความผิดพลาดให้อยู่ในรูปแบบรหัสมาตรฐานแล้วส่งกลับไปให้ แอพพลิเคชั่น

3.6 ทำหน้าที่จัดการและดูแล Cursor

4.แหล่งข้อมูล (Data Source)

แหล่งข้อมูลหมายถึงการรวมกันของ DMBS, ระบบปฏิบัติการ และ ระบบเครือข่าย โดยมีการชี้เฉพาะชนิด และ ประเภท ลงไป หรือ อีกนัยหนึ่งก็หมายความว่า การที่แอพพลิเคชั่นทำการติดต่อกับ DBMS ยี่ห้อหนึ่งบนระบบปฏิบัติการหนึ่ง และเข้าถึงโดยระบบเครือข่ายชนิดหนึ่ง เช่น

- ออราเคิล ที่วิ่งบนระบบปฏิบัติการ OS/2 โดยใช้ ระบบเครือข่ายของ Novell Netwave

6.3.2 การใช้วิชวลเบสิกในการติดต่อฐานข้อมูล

วิชวลเบสิกสามารถที่จะเรียกใช้ข้อมูลจากฐานข้อมูลได้หลายรูปแบบ การเรียกใช้ข้อมูลของ วิชวลเบสิก สามารถแบ่งออกเป็น 3 หัวข้อใหญ่ดังนี้

1. ฐานข้อมูลแบบไมโครซอฟท์ เอ็กซ์เซสฐานข้อมูลแบบนี้สามารถทำการสร้างและบริหารได้โดยใช้ไมโครซอฟท์ แอ็กซ์เซส หรือวิชวลเบสิก ก็ได้ ฐานข้อมูลชนิดนี้จะเป็นรูปแบบพื้นฐานของ วิชวลเบสิก ซึ่งให้ทั้งความคล่องตัวและความเร็วในการใช้งาน

2.ฐานข้อมูลภายนอก เช่น Btrieve, dbaseIII, dbaseIV, Microsoft FoxPro Version 2.0,2.5 และ Paradox, วิชวลเบสิก สามารถที่จะสร้างหรือเข้า ใช้ฐานข้อมูลเหล่านี้ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.ฐานข้อมูลODBC ภายนอก ในหัวข้อนี้หมายถึงฐานข้อมูลแบบไคลเอ็นท์เซอร์ฟเวอร์ เช่น Microsoft SQL Server, Oracle DBMS วิชวลเบสิก สามารถส่งคำสั่ง SQL ไปทำการประมวลผลที่ DBMS เหล่านี้ได้

คุณสมบัติการเข้าใช้ฐานข้อมูลของ วิชวลเบสิก นั้น คุณสมบัติหรือความสามารถบางชนิดนั้นทำได้ใน วิชวลเบสิก รุ่น Professional เท่านั้น ซึ่งต่อไปนี้จะเป็นการแสดงตารางเปรียบเทียบความสามารถระหว่าง วิชวลเบสิก ในรุ่น Standard กับ วิชวลเบสิก ในรุ่น Professional

ความสามารถ	รุ่น Standard	รุ่น Professional
Bound Controls	ได้	ได้
Data Control	ได้	ได้
สร้าง Database Object	เฉพาะใน Data Control	ได้
สร้าง Dynaset Object	เฉพาะใน Data Control	ได้
สร้าง Index Object	ไม่ได้	ได้
สร้าง QueryDef Object	ไม่ได้	ได้
สร้าง Snapshot Object	ไม่ได้	ได้
สร้าง Table Object	ไม่ได้	ได้
สร้าง Data access Object	ไม่ได้	ได้
สร้าง ฐานข้อมูลใหม่	ไม่ได้	ได้
เปลี่ยนโครงสร้างฐานข้อมูล	ไม่ได้	ได้

ตารางแสดงความแตกต่างระหว่าง Visual รุ่น Standard และ รุ่น Professional

6.3.2.1 การเปิดฐานข้อมูล (Opening Database)

การติดต่อกับฐานข้อมูลของวิชวลเบสิกทำได้สองทางด้วยกันคือ

1. การใช้ Data Control
2. การเรียกใช้ ฟังก์ชั่น OpenDatabase

Database object ที่ถูกสร้างขึ้นโดยฟังก์ชั่น OpenDatabase นี้ก็เหมือนกับการสร้างจาก การใช้ Data Control นั่นเอง หน้าที่ของฟังก์ชั่นสามารถเรียงลำดับได้ดังนี้

- กำหนดชื่อแหล่งข้อมูล
- อธิบายวิธีการเข้าถึงฐานข้อมูล
- กำหนดการใช้ฐานข้อมูลแบบ Single-User หรือ Multiple-User
- กำหนดสถานะ Read-only หรือ Read/Write
- สร้างตัวแปรฐานข้อมูลให้กับแหล่งข้อมูล
- สร้างฐานข้อมูลใหม่ ในรูปแบบของ ฐานข้อมูลภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเรียกใช้งาน ฟังก์ชั่น OpenDatabase มีรูปแบบดังนี้

```
OpendataBase (database name [, exclusive [,readonly [, connect ]]])
```

รูปแบบของฐานข้อมูลที่ วิชวลเบสิก สามารถทำการเชื่อมต่อได้ จะอยู่ในรูปของแฟ้มข้อมูล ที่มีนามสกุลเป็น .MDB (ฐานข้อมูลของ ไมโครซอฟท์ แอ็กซ์เซส) ,ไคเรคทอรี่ของฐานข้อมูล เช่นของ dBaseIII หรือแหล่งข้อมูลในรูปแบบของ ODBC

- Databasename เป็นตัวกำหนดแหล่งข้อมูลที่ วิชวลเบสิก จะใช้ รูปแบบของ databasename จะขึ้นอยู่กับชนิดของ database ที่จะเปิด
- Connect ใช้แสดงถึงชนิดของฐานข้อมูลที่จะติดต่อ และอาจใช้ส่งพารามิเตอร์เพิ่มเติมในการติดต่อกับฐานข้อมูล

ตารางแสดงพารามิเตอร์และชนิดของฐานข้อมูลที่สามารถเชื่อมต่อกับวิชวลเบสิกได้

รูปแบบของฐานข้อมูล	Databasename	Connect
Microsoft Access	drive:\path\file.MDB	(ไม่มี)
Btrieve	drive:\path\file.DDF	"Btrieve"
dBASEIII	drive:\path	"dBASEIII"
dBASEIV	drive:\path	"dBASEIV"
FoxPro Version 2.0	drive:\path	"FoxPro 2.0"
FoxPro Version 2.5	drive:\path	"FoxPro 2.5"
ODBC(SQL Server, Oracle)	Registered Data source name(โดยปกติแล้วก็คือ ชื่อของ Server	"Ddbc;Dsn=Server;Uid=User;Pwd=Password"
Paradox	drive:\path	"Paradox"

- Exclusive เป็นพารามิเตอร์ที่ใช้บอกถึงการเข้าใช้ฐานข้อมูลร่วมกับผู้ใช้คนอื่น เมื่อใดที่ต้องการจะใช้ฐานข้อมูลร่วมกันกับคนอื่นให้ทำการ ตั้งค่า Exclusive ให้เป็น False ถ้าฐานข้อมูลถูกเปิดโดยผู้ใช้คนอื่นแล้วเราจะไม่สามารถเปิดฐานข้อมูลโดยตั้งค่า Exclusive ให้เป็น True ได้ แต่ถ้าเมื่อใดก็ตามที่เราเปิดฐานข้อมูลโดยตั้งค่า Exclusive เป็น True แล้ว ผู้ใช้คนอื่นจะไม่สามารถเปิดฐานข้อมูลเราได้
- Readonly ถ้าต้องการเปิดฐานข้อมูลให้เป็นการอ่านอย่างเดียวให้ตั้งค่านี้เป็น True

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างการเปิดฐานข้อมูลหลายๆรูปแบบโดยใช้ วิชวลเบสิก

ชนิดของฐานข้อมูล	ตัวอย่างการเรียกใช้ฟังก์ชั่น OpenDatabase
Mircrosoft Access	OpenDatabase("MYDB.MDB",False,False,"") - หรือ - OpenDatabase("MYDB.MDB")
Btrieve	OpenDatabase("C:\MYBEE.DDF", False,False,"Betrieve;")
dBASEIII	OpenDatabase("C:\MYDBX",False,False, "dBASEIII;")
FoxPro version 2.5	OpenDatabase("C:\MYFOX",False,False, "FoxPro 2.5;")
ODBC (แบบมี Login Dialog)	OpenDatabase("",False,False,"ODBC;")
Paradox	OpenDatabase("C:\MYPDX",False,False, "Paradox;")
ODBC (แบบมีการกำหนด ชื่อแหล่งข้อมูล)	OpenDatabase("Accounts",False,False,"ODBC;")
ODBC (แบบมีการกำหนด ชื่อของแหล่งข้อมูลไว้ในไฟล์ ODBC.INI และใช้ตัวแปร String เก็บข้อมูลพารามิเตอร์ Connect	C$="ODBC;uid=sa;pwd=zzx;DSN=Accounts; OpenDatabase("",False,False,C$)

หลังจากที่ได้รู้วิธีการเปิดฐานข้อมูลเพื่อใช้งานแล้ว งานทางด้านฐานข้อมูลจะ ต้องมีการจัดการทางด้านข้อมูล เช่นการเพิ่มข้อมูลเข้า, จัดเรียงข้อมูล, และค้นหา ข้อมูล ซึ่งการทำงานเหล่านี้สามารถเรียกใช้ผ่านทาง method และ property ของ object ใน วิชวลเบสิก ได้

6.3.2.2 การอ่านข้อมูลโดยใช้ตัวแปรเรคคอร์ดเซท (Recordset)

วิธีที่ง่ายที่สุดในการจัดการกับฐานข้อมูลก็คือการใช้ตัวแปร เรคคอร์ดเซท กลุ่มของตัวแปรพวกนี้ได้แก่ ตัวแปร Table, Dynaset, Snapshot ตัวอย่างของการใช้งานตัวแปรเหล่านี้เช่น การจัดลำดับข้อมูลของฐานข้อมูลและส่งกลับมายัง แอพพลิเคชั่น , การเลือกเรคคอร์ด (Select) และ การเปลี่ยนแปลงข้อมูลในฐานข้อมูล เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวแปร เรคคอร์ดเซท	สมาชิก	เรคคอร์ด	ผลลัพธ์ของ Query
Table	สามารถเปลี่ยนได้	สามารถ เพิ่ม,เปลี่ยน,ลบ ได้	ไม่มีการส่งผลกลับ
Dynaset	จำนวนสมาชิกคงที่	สามารถ เพิ่ม,เปลี่ยน,ลบ ได้	สามารถส่งผลกลับ
Snapshot	จำนวนสมาชิกคงที่	จำนวนสมาชิกคงที่	สามารถส่งผลกลับ

ตารางแสดงความสามารถของตัวแปรเรคคอร์ดเซทแต่ละชนิด

คำว่า สมาชิก ในที่นี้หมายถึงกลุ่มของเรคคอร์ดซึ่งอยู่ในตัวแปร เรคคอร์ดเซท เมื่อถูกสร้างขึ้นมา ในกรณีของตัวแปร Table การเปลี่ยนแปลงใดๆ ก็ตามกับฐานข้อมูล โดยผู้ใช้คนอื่นๆ จะมีผลต่อการเข้าใช้ฐานข้อมูลของตัวแปรชนิดนี้ทันที แต่ในกรณีของตัวแปร Dynaset และ Snapshot จะไม่เป็นเปลี่ยนแปลง

ต่อไปนี้เป็นตัวอย่างการกำหนดตัวแปร Table , Dynaset และ Snapshot ก่อนทำการกำหนดตัวแปรในตัวโปรแกรมจะต้องมีการประกาศตัวแปร Database ไว้ก่อน เพื่อใช้ในการอ้างอิงข้อมูล จากนั้นจึงสามารถใช้เมทอดส์ OpenTable หรือ CreateDynaset หรือ CreateSnapshot ในการกำหนดตัวแปร Table , Dynaset และ Snapshot ได้ตามลำดับ

```
Dim Db as Database, Table Var as Table
Set Db = Open database ("BIBLIO.MDB")
Set Table Var = Db.OpenTable ("Titles")
```

ตัวอย่างการประกาศตัวแปร Table

```
Dim Db as Database, dsSomeData as Dynaset
Set Db = OpenDatabase ("BIBLIO.MDB")
Set dsSomeData = Db.CreateDynaset ("Titles")
```

ตัวอย่างการประกาศตัวแปร Dynaset

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Dim Db as Database, SnapData as Snapshot
Set Db = OpenDatabase ("BIBLIO.MDB")
Set SnapData = Db.CreateSnapshot ("Titles")
```

ตัวอย่างการประการตัวแปร Snapshot

รูปแบบของคำสั่งทั้งสามมีรูปแบบดังนี้

```
OpenTable (tablename [, options ])
```

พารามิเตอร์ tablename หมายถึงชื่อตารางในฐานข้อมูล ในส่วนของ Options หมายถึง ตัวเลือกในการกำหนดชนิดการเข้าใช้ข้อมูล เช่น การอ่าน, เขียน และ การประมวลผลคำสั่งภาษา SQL

```
CreateDynaset (source [, options ])
```

พารามิเตอร์ source อาจเป็นได้ทั้ง ชื่อตาราง, ชื่อของ QueryDef หรือคำสั่งภาษา SQL ในรูปของ ค่าตัวแปรไทป์ แบบ String ตารางของฐานข้อมูลเป็นได้ทั้งตารางที่อยู่ในระบบของตัวเอง (Local) หรืออยู่ในระบบอื่น (Remote) ในส่วนของพารามิเตอร์ options ก็จะเหมือนกับของ OpenTable

```
CreateSnapshot (source [options ])
```

พารามิเตอร์และการใช้งานของคำสั่งนี้จะเหมือนกับคำสั่ง CreateDynaset แต่จะมีข้อยกเว้นที่ตัวแปรประเภทนี้จะไม่สามารถทำการแก้ไข, เปลี่ยนแปลงข้อมูลในฐานข้อมูลได้

```
Dim Db As Database , dsSomeData As Dynaset, SQLQ  As String
Set Db = OpenDatabase("BIBLIO.MDB")
SQLQ = "SELECT Titles.Title, Publishers,Name "
SQLQ = SQLQ  & " FROM Titles.Publishers "
SQLQ = SQLQ & " WHERE Titles.PubId = Publishers.PubId "
Set dsSomeData = Db.CreateDynaset(SQLQ)
```

ตัวอย่างการส่งคำสั่ง SQL โดยใช้ตัวแปร Dynaset

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3.2.3 การเรียงลำดับข้อมูลในตัวแปร Dynaset และ Snapshot

โดยทั่วไปแล้วลำดับการเรียงข้อมูลของตัวแปรทั้งสองนี้จะเป็นไปตามไพร์มารี่คีย์ (Primary Key) ที่กำหนดไว้ในตารางที่อ้างอิงมานั้น อย่างไรก็ตามถ้าไม่ได้มีการกำหนด ไพร์มารี่ คีย์ก็จะเรียงตามลำดับของข้อมูลที่ถูกเพิ่มเข้ามาในตารางหรือไม่ก็เรียงตามลำดับของชุดคำสั่ง ORDER BY ในคำสั่ง SQL

```
Dim Db As Database , SomeData As Dynaset , SortedDataAs Dynaset
Set Db = OpenDatabase("BIBLIO.MDB")
SetSomeData = Db.CreateDynaset("Titles")
SomeData.Sort =   "[Company  Name]"
Set SortedData = SomeData.CreateDynaset()
```

ตัวอย่างการเรียงข้อมูล

6.3.2.4 การเลือกข้อมูล (Selecting)

วิชวลเบสิก มีความสามารถในการเลือกชนิดของข้อมูลและข้อมูลที่มีความสัมพันธ์เหมือนๆ กันให้ออกมาได้โดยใช้ตัวแปร Dynaset และ Snapshot ในส่วนของวิธีการในการเลือกนั้น ก็เพียงแต่กำหนด Property Filter ของตัวแปรทั้ง 2 ให้อยู่ในลักษณะ WHERE-clause ของภาษา SQL แต่มีข้อยกเว้นในการอ้างชื่อ Field ซึ่งเราจะต้องอ้างโดยใช้เครื่องหมาย '[' และ ']' ครอบระหว่างชื่อ Field หรือให้อยู่ในรูปแบบดังนี้

```
Tablename![Field Name]
```

```
' ตัวอย่างการเลือกข้อมูลจากตาราง Publisher โดยเลือกเอาเฉพาะของบริษัท ' Microsoft Press'
Dim Db As Database, SnapData As Snapshot
Set Db = OpenDatabase("BIBLIO.MDB")
Set SnapData = Db.CreateSnapshot("Publishers")
SnapData.Filter = "[Company Name] = 'Microsoft Press' "
Set SnapData = SnapData.CreateSnapshot()
```

ตัวอย่างการเลือกข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อต้องการเลิกใช้ตัวแปร เรคคอร์ดเซท ให้ทำการใช้ เมทตอดส์ Close

```
TableVar.Close
dsSamedata.Close
Snapdata.Close
```

การยกเลิกการใช้งานตัวแปร เรคคอร์ดเซท

6.3.2.5 การเลื่อนตำแหน่งของเรคคอร์ดในตัวแปรเรคคอร์ดเซท

เมทตอดส์ Move

วิชาลเบสิก มีวิธีการเลื่อนตำแหน่งของ เรคคอร์ด ไว้ด้วยกัน 4 วิธีการ

เมทตอดส์	จุดประสงค์
MoveFirst	เลื่อนตำแหน่งไปยัง เรคคอร์ด แรกสุด
MoveLast	เลื่อนตำแหน่งไปยัง เรคคอร์ด ท้ายสุด
MoveNext	เลื่อนตำแหน่งเดินหน้าไปหนึ่ง เรคคอร์ด
MovePrevious	เลื่อนตำแหน่งถอยหลังไปหนึ่ง เรคคอร์ด

```
Sub MoveForward ()
Dim Db As Database, Ds As Dynaset
Set Db = OpenDatabase("BIBLIO.MDB")  ' เปิดฐานข้อมูล
Set Ds = Db.CreateDynaset("Titles")        ' อ่านข้อมูลจากตาราง Titles
Do Until Ds.EOF                            ' ทำซ้ำจนอ่านข้อมูลหมด
   Debug.Print Ds("Year Published"),Ds("Title")
   Ds.MoveNext                             ' เลื่อนข้อมูลเดินหน้าไปหนึ่ง เรคคอร์ด
Loop
Ds.Close
Db.Close
End Sub
```

ตัวอย่างการเลื่อนข้อมูลไปข้างหน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub MoveBack()
Dim Db As Database, Ds As Dynaset
Set Db = OpenDatabase("BIBLIO.MDB")  ' เปิดฐานข้อมูล
Set Ds = Db.CreateDynaset("Titles")        ' อ่านข้อมูลจากตาราง Titles
Ds.MoveLast                                ' เลื่อนไปยังตำแหน่งสุดท้ายของ เรคคอร์ด
Do Until Ds.BOF                            ' ทำซ้ำจนถึงตำแหน่งเริ่มต้น
   Debug.Print Ds("Year Published"),Ds("Title")
   Ds.MovePrevious                         ' เลื่อนข้อมูลถอยหลังไปหนึ่ง เรคคอร์ด
Loop
Ds.Close
Db.Close
End Sub
```

ตัวอย่างการเลื่อนข้อมูลถอยหลัง

6.3.2.6 การค้นหาข้อมูลจากตัวแปร เรคคอร์ดเซท

แทนที่จะให้ วิชวลเบสิก วน loop เพื่อที่จะหาข้อมูลให้สอดคล้องตามเงื่อนไข(Criteria) ที่ต้องการ วิชวลเบสิก ยังมีชุดคำสั่งอีก 4 แบบให้ใช้ในการค้นหาข้อมูล ในตัวแปรแบบ Dynaset และ Snapshot

เมทตอดส์	เงื่อนไขการค้นหา	จุดเริ่มต้นการค้นหา
FindFirst เงื่อนไข	หา เรคคอร์ด แรกที่เข้ากับเงื่อนไข	จาก เรคคอร์ด แรก
FinLast เงื่อนไข	หา เรคคอร์ด สุดท้ายที่เข้ากับเงื่อนไข	จาก เรคคอร์ด สุดท้าย
FindNext เงื่อนไข	หา เรคคอร์ด ต่อไปที่เข้ากับเงื่อนไข	จาก เรคคอร์ด ปัจจุบัน
FindPrevious เงื่อนไข	หา เรคคอร์ด ก่อนที่เข้ากับเงื่อนไข	จาก เรคคอร์ด ปัจจุบัน

เงื่อนไขในคำสั่งจะอยู่ในรูปของ WHERE-clause ของชุดคำสั่งแบบภาษา SQL เพียงแต่ตัดคำว่า WHERE ออก เช่น

"[Lase Name] > 'M'"

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
'ตัวอย่างการค้นหาข้อมูลหนังสือที่เกี่ยวกับ SQL และนำข้อมูลที่ได้มาเก็บใน ListBox
Sub SQLBooks(L As ListBox)
Dim Db As Database, Ds As Dynaset,  Criteria
Set Db = OpenDatabase("BIBLIO.MDB")              ' เปิดฐานข้อมูล
Set Ds =Db.CreateDynaset("Titles")               ' อ่านข้อมูลจากตาราง Titles
Criteria = "[Title] LIKE '*SQL*'"                ' เลือกเฉพาะข้อมูลที่มีข้อความ 'SQL'
Ds.FindFirst Criteria                            ' ค้นหา เรคคอร์ด แรกที่พบ
Do Until Ds.NoMatch                              ' ทำจนกระทั่งไม่พบข้อมูล
   L.AddItem Ds("Title") & " " &Ds("Date  Published")
   Ds.FindNext Criteria
Loop
Ds.Close
Db.Close
End Sub
```

ตัวอย่างการใช้งานคำสั่งค้นหาข้อมูล

6.2.3.7 การแก้ไขข้อมูลในเรคคอร์ด

ในการเปลี่ยนแปลงข้อมูลใน เรคคอร์ด ใดๆ นั้น มีขั้นตอนที่จำเป็นอยู่ 3 ขั้นด้วยกันดังนี้

1. เรียกใช้ Edit เมทตอดส์ เพื่อเป็นการเตรียมการแก้ไขข้อมูล
2. กำหนดค่าข้อมูลใหม่ให้กับแต่ละ field ที่ต้องการจะแก้ไขในแต่ละ เรคคอร์ด
3. เรียกใช้ Update เมทตอดส์ เพื่อบันทึกการแก้ไขที่เกิดขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
'ตัวอย่างการแก้ไขข้อมูลในตาราง  Publishers
Sub ReplaceCity (OldValue, NewValue)
Dim Db As Database , T As Table
   Set Db = OpenDatabase("BIBLIO.MDB")
   Set T  = Db.OpenTable("Publishers")
   Do Until T.EOF
      If T("City") = OldValue   Then
         T.Edit
         T("City") = NewValue
         T.Update
      EndIf
      T.MoveNext
   Loop
   T.Close
End Sub
```

ตัวอย่างการแก้ไขข้อมูล

ในระบบที่มีผู้ใช้หลายคน การแก้ไขข้อมูลในฐานข้อมูลจำเป็นจะต้องมีการป้องกันการใช้ข้อมูลที่กำลังถูกแก้ไขอยู่ซึ่งวิชวลเบสิก มีวิธีป้องกันการใช้ข้อมูลด้วยกัน 2 วิธี

- *Pessimistic Locking* วิชวลเบสิก จะทำการ ล็อค เพจ ที่มีข้อมูลที่กำลังจะถูกแก้ไขหลังจากใช้ เมทตอดส์ Edit โดยอัตโนมัติ ฉะนั้นผู้ใช้คนอื่นๆ ก็จะไม่สามารถเข้ามาใช้ข้อมูลในส่วนนี้ได้ จนกระทั่งมีการใช้ เมทตอดส์ Update อีกที
- *Optimistic Locking* วิชวลเบสิก จะไม่ทำการ ล็อค เพจ ที่กำลังทำการแก้ไขอยู่ จนกระทั่ง เมทตอดส์ Update ถูกเรียกใช้ วิชวลเบสิก จึงจะทำการ ล็อคเพราะเหตุนี้ผู้ใช้คนอื่นในระบบจึงสามารถแก้ไขและเปลี่ยนแปลงข้อมูลในขณะที่เรากำลังทำงานอยู่ได้

6.3.2.8 การลบเรคคอร์ด

การลบเรคคอร์ดออกจากตารางสามารถทำได้โดยการเรียกใช้เมทตอดส์ Delete และหลังจากการเรียกใช้ เมทตอดส์ Delete แล้วเรคคอร์ดนั้นจะยังไม่ถูกลบออกไปทันทีจนกว่าจะมีการเลื่อนตำแหน่งไปยังตำแหน่งใหม่ในตัวแปร Dynaset หรือ Table

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub EmptyTable (tAny As Table)
    tAny.MoveFirst
    Do Until  tAny.EOF
      tAny.Delete
      tAny.MoveNext
    Loop
End Sub
```

ตัวอย่างการลบ เรคคอร์ด

6.3.2.9 การเพิ่มเรคคอร์ดใหม่

วิธีการเพิ่มเรคคอร์ดเข้าสู่ฐานข้อมูลมีขั้นตอน 3 ขั้นตอนดังนี้

1. ใช้ เมทตอดส์ AddNew เพื่อสร้าง เรคคอร์ด ใหม่
2. กำหนดค่าให้กับ field ในแต่ละ field ใน เรคคอร์ด ใหม่
3. ใช้ เมทตอดส์ Update เพื่อบันทึกข้อมูล เรคคอร์ด ที่เพิ่มเข้าไปลงสู่ฐานข้อมูล

```
Dim  Db As Database, T As Table
Set Db = OpenDatabase("BIBLIO.MDB")
Set T = Db.OpenTable("Titles")
T.AddNew
T("Name")   = "A Hitchhiker' s Guide to   VBSQL"
T("Year Published") = "1993"
T.Update
T.Close
Db.Close
```

ตัวอย่างการเพิ่มข้อมูล

การเพิ่มข้อมูลใหม่โดยผ่านทางตัวแปร Table ข้อมูลที่เพิ่มเข้าใหม่นั้นจะจัดลำดับตาม Index ที่กำหนดไว้ สำหรับกรณีที่ไม่มีการกำหนด Index ข้อมูลจะถูกใส่เข้าไปที่ท้ายตาราง แต่ถ้าเป็นการเพิ่มข้อมูลผ่านทางตัวแปร Dynaset ข้อมูลที่เพิ่มเข้าไปใหม่จะปรากฏอยู่ที่ด้านท้ายของตัวแปร เพราะฉะนั้นถ้าต้องการจัดเรียงลำดับใหม่ให้ถูกต้องเราจะต้องทำการจัดเรียงลำดับตัวแปร Dynaset ใหม่ และสร้างตัวแปรขึ้นมาอีกทีหนึ่งหรือไม่ก็ยกเลิกการใช้งานตัวแปรนั้น แล้วก็สร้างตัวแปรโดยให้ทำการประมวลผล Query ใหม่อีกครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3.2.10 การป้องกันการเข้าใช้ข้อมูล (Locking Data)

ในระบบที่มีผู้ใช้หลายคน บางครั้งอาจมีการใช้และเปลี่ยนแปลงข้อมูลในที่เดียวกัน ซึ่งก่อให้เกิดปัญหาในการใช้งานขึ้น วิธีการที่จะแก้ปัญหาที่เกิดขึ้นมานี้ที่ระบบบริหารฐานข้อมูลทั่วไปที่นิยมใช้กันก็คือการ ล็อค (Lock) การป้องกันการใช้ข้อมูลของ วิชวลเบสิก มีไว้ด้วยกัน 3 ระดับคือ

- Database locking
- Table and Dynaset locking
- Page locking

การล็อค Database (Database Locking)

เป็นวิธีการที่ง่ายที่สุดในการป้องกันการใช้ข้อมูล แต่ในขณะเดียวกันก็มีข้อจำกัดในการใช้งานมากที่สุดเช่นกัน การ ล็อค วิธีนี้จะมีเพียงแต่ผู้ที่ทำการ ล็อค เท่านั้นที่สามารถใช้ข้อมูลจากฐานข้อมูลได้ วิธีการ ล็อคฐานข้อมูลทั้งหมดสามารถทำได้โดยการกำหนดค่าพารามิเตอร์ของฟังก์ชั่น OpenDatabase ให้เท่ากับ True

```
Set Db = OpenDatabase ("BIBLIO.MDB"True)
```

การล็อค Table และ Dynaset (Table and Dynaset Locking)

การล็อคชนิดนี้มีข้อจำกัดในการใช้งานลดลง หมายความว่าผู้ใช้คนอื่นสามารถที่จะเข้ามาร่วมใช้ฐานข้อมูลด้วยได้ วิธีการล็อคสามารถกำหนดได้จากพารามิเตอร์ options ของฟังก์ชั่น

```
Option% = DB_DENYWRITE + DB_DENYREAD
Set DataRecs = DB.CreateDynaset ("Titles", Option%)
```

จากตัวอย่างด้านบน DB_DENYWRITEบ่งบอกว่าไม่อนุญาตให้ผู้ใช้คนอื่นสามารถเขียนข้อมูลลงในเรคคอร์ด ที่อยู่ใน Dynasetได้ ส่วน DB_DENYREAD หมายถึงการไม่อนุญาตให้ผู้อื่นอ่านข้อมูลจาก Dynaset ได้ การ ล็อคข้อมูลของ Data Control สามารถทำได้เช่นเดียวกัน โดยการกำหนดค่า Property ของ Data Control ให้มีค่าตามต้องการ เช่น

```
Data1.Options = DB_DENYWRITE  + DB_DENYREAD
Data1.DatabaseName = "BIBLIO.MDB"
Data1.RecordSource = "TITLES"
Data1.Refresh
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การล็อคเพจ (Locking Pages)

ในการใช้งานทั่วไปถ้าไม่ได้มีการกำหนดวิธีการ ล็อค ข้อมูลไว้ วิชวลเบสิก จะทำการล็อคข้อมูลเอง โดยอัตโนมัติการล็อคในที่นี้จะล็อคเป็นเพจ โดยมีขนาดเพจละ 2048 ไบต์ (2K)

การล็อคในระดับนี้ แบ่งออกเป็น 2 ชนิดคือ

1. Pessimistic Locking เมื่อมีการแก้ไขหรือเพิ่มข้อมูลในตัวแปร Dynaset และ Table วิชวลเบสิก จะใช้การล็อค เป็นแบบนี้โดยอัตโนมัติ การ ล็อค จะเริ่มตั้งแต่การเริ่มใช้ เมทธอดส์ Edit และสิ้นสุดเมื่อ เมทธอดส์ Update ทำงานเสร็จสิ้น

- ข้อดี ของการล็อคชนิดนี้คือสามารถป้องกันไม่ให้ผู้ใช้คนอื่นเข้ามาใช้ข้อมูลในขณะที่ทำการเปลี่ยนแปลงได้อย่างแน่นอน
- ข้อเสีย ก็คือถ้าช่วงเวลาระหว่างการเรียกใช้ เมทธอดส์ Edit และUpdate มีช่วงเวลาที่ห่างกันมาก ซึ่งก็จะทำให้ข้อมูลในเพจนั้นไม่สามารถเข้าใช้ได้ จึงเป็นสาเหตุสำคัญที่ทำให้เสียเวลามาก

```
Dim Db As Data,  DataRecs As  Dynaset
Set  Db  = OpenDatabase("BIBLIO.MDB")
Set DataRecs  =  Db.CreateDynaset("Titles")
DataRecs.Edit                ' เพจ เริ่มถูก ล็อค
● ● ●
DataRecs.Update              ' เพจ ถูกปล่อยจากการ ล็อค
```

ตัวอย่างการใช้การ ล็อค แบบ Pessimistic

2. Optimistic Locking การ ล็อค จะไม่เกิดขึ้นจนกว่าจะมีการเรียกใช้เมทธอดส์ Update วิธีการ ล็อค แบบนี้สามารถทำได้โดยการกำหนดค่า Property Lock Edit ของตัวแปร Dynaset ให้เท่ากับ False

- ข้อดีของวิธีการนี้คือ ช่วงเวลาที่ใช้ในการ ล็อค ข้อมูลจะกินเวลาไม่มาก
- ข้อเสียคือ เราไม่สามารถทราบได้ว่าการแก้ไขของเราต่อฐานข้อมูลสำเร็จหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
On Error Resume Next
Dim Db As Database, Database, DataRecs As Dynaset
Set Db = OpenDatabase("BIBLIO.MDB")
Set DataRecs = Db.CreateDynaset("Titles")
DataRecs.LockEdits  = False
DataRecs.Edit                  ' เพจ ยังคงไม่ถูก ล็อค
● ● ●
DataRecs.Update                ' เพจ เริ่มถูก ล็อค ในช่วงระหว่างการ Update
If  Err Then  MsgBox "Update Failed."
```

ตัวอย่างการใช้การ ล็อค แบบ Optimistic

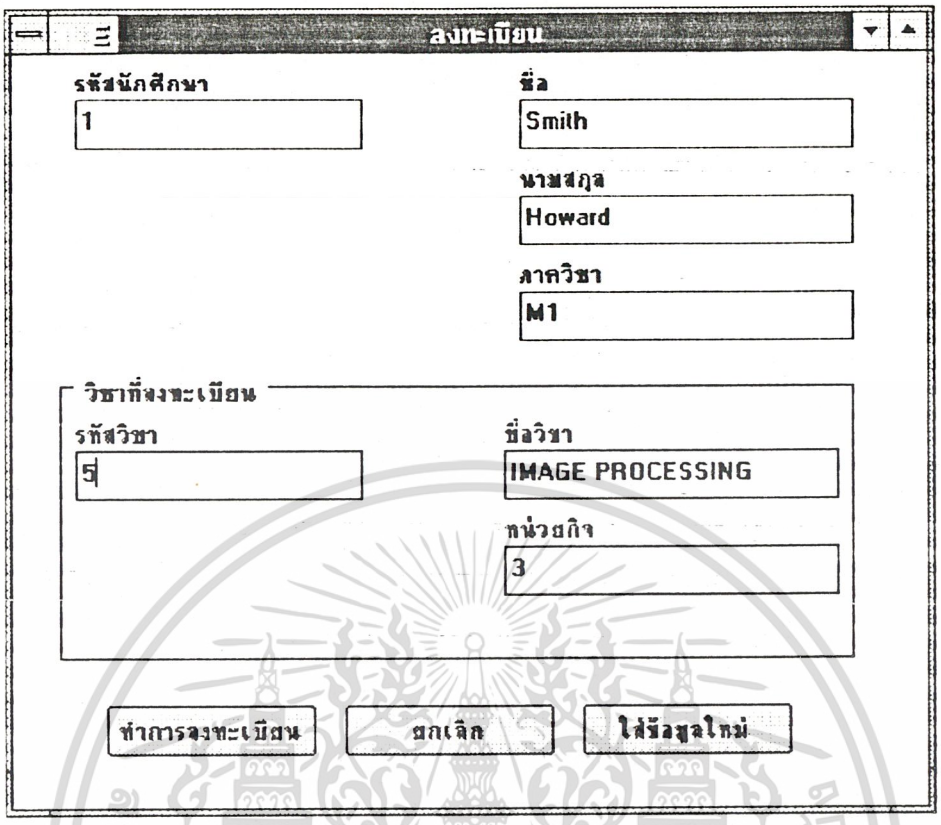
หมายเหตุ ถ้าตัว DBMS ของระบบเป็น DBMS แบบ ODBC แล้วการ ล็อค ฐานข้อมูลจะกระทำโดยตัว DBMS โดยอัตโนมัติทำให้ไม่ต้องจัดการเอง

ที่ได้กล่าวมาทั้งหมดนั้นคือการประยุกต์ใช้ วิชวลเบสิก กับระบบฐานข้อมูลของ แอ็กซ์เซส และฐานข้อมูลแบบ ODBC ซึ่งจะเห็นว่ามีความสะดวกในการใช้งานมากมาย เมื่อเทียบกับการโปรแกรมในแบบเก่าๆ ที่ใช้การ Embeded SQL มาเป็นส่วนเชื่อมต่อระหว่างฐานข้อมูล เพราะเหตุที่ว่า วิชวลเบสิก สามารถส่งคำสั่ง SQL ให้ไปประมวลผลแบบ ไดนามิค(Dynamic) ได้ จึงทำให้ แอพพลิเคชั่นที่สร้างขึ้น มีความยืดหยุ่นในการใช้งานอย่างมาก และด้วยความสามารถที่จะใช้ ฐานข้อมูลเป็นแบบ ODBC ได้ จึงทำให้ วิชวลเบสิก สามารถติดต่อกับระบบบริหารฐานข้อมูลได้หลายชนิด

6.3.3 การพัฒนาระบบลงทะเบียนโดยใช้ภาษาวิชวลเบสิก

ในสองหัวข้อที่แล้วได้แนะนำการใช้งานและการติดต่อกับฐานข้อมูลไปแล้ว ในหัวข้อนี้จะนำเสนอวิธีการใช้งานวิชวลเบสิก กับ ฐานข้อมูล ในระบบงานลงทะเบียน โดยจะนำเสนอการใช้งานเป็นสองรูปแบบ แบบแรกจะใช้ฐานข้อมูลเป็น ไมโครซอฟท์ แอ็กซ์เซส ส่วนในระบบที่สองจะใช้ฐานข้อมูลเป็นฐานข้อมูลแบบ ODBC ของ SQL Server

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแบบหน้าจอการลงทะเบียน

```
'All variable must be declared
Option Explicit
Dim flag1 As Integer
Dim flag2 As Integer

Sub Command3_Click ()
    Dim Regs_Db As database
    Dim Regs_Tb As dynaset
    Dim Msg As String
    On Error GoTo ErrorHanddle
    Set Regs_Db = OpenDatabase("a:\register.mdb")
    Set Regs_Tb = Regs_Db.CreateDynaset("register")
    Regs_Tb.AddNew
    Regs_Tb("stu_no") = text1.Text
    Regs_Tb("sub_no") = text5.Text
    Regs_Tb.Update
    Regs_Tb.Close
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    Regs_Db.Close
    flag1 = 0
    flag2 = 0
    command3.Enabled = False
    Exit Sub
ErrorHanddle:
    command3.Enabled = False
    flag1 = 0
    flag2 = 0
    Msg = "Can't insert!"
    MsgBox Msg
    Regs_Tb.Close
    Regs_Db.Close
    Exit Sub
    Resume Next
End Sub

Sub Command4_Click ()
    End
End Sub

Sub Command5_Click ()
    text1.Text = ""
    text2.Text = ""
    text3.Text = ""
    text4.Text = ""
    text5.Text = ""
    text6.Text = ""
    text7.Text = ""
End Sub

Sub Form_Load ()
    flag1 = 0
    flag2 = 0
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Sub Text1_KeyPress (KeyAscii As Integer)
    If KeyAscii <> 13 Then
        Exit Sub
    End If
    Dim stu_Db As database
    Dim stu_Ds As dynaset
    Dim sql As String
    Dim Msg As String
    On Error Resume Next
    Set stu_Db = OpenDatabase("a:\register.mdb")
    If text1.Text = "" Then
        flag1 = 0
        Msg = "Please enter student number"
        MsgBox Msg
        stu_Ds.Close
        stu_Db.Close
        Exit Sub
    Else
        sql = "select * from student where"
        sql = sql & " stu_no=" & text1.Text
        Set stu_Ds = stu_Db.CreateDynaset(sql)
    End If
    If stu_Ds("stu_name") = "" Then
        flag1 = 0
        Msg = "Having no data!"
        MsgBox Msg
        stu_Ds.Close
        stu_Db.Close
        Exit Sub
    End If
    flag1 = 1
    text2.Text = stu_Ds("stu_name")
    text3.Text = stu_Ds("stu_surname")
    text4.Text = stu_Ds("stu_major")
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    stu_Ds.Close
    stu_Db.Close
End Sub

Sub Text5_GotFocus ()
    Dim Msg As String
    If text1.Text = "" Then
        Msg = "Please enter student number"
        MsgBox Msg
        text1.SetFocus
    End If
End Sub

Sub Text5_KeyPress (KeyAscii As Integer)
    If KeyAscii <> 13 Then
        Exit Sub
    End If
    Dim sub_Db As database
    Dim sub_Ds As dynaset
    Dim sql As String
    Dim Msg As String
    On Error Resume Next
    Set sub_Db = OpenDatabase("a:\register.mdb")
    If text1.Text = "" Then
        flag1 = 0
        Msg = "Please enter subject number"
        MsgBox Msg
        sub_Ds.Close
        sub_Db.Close
        Exit Sub
    Else
        sql = "select * from subject where"
        sql = sql & " sub_no=" & text5.Text
        Set sub_Ds = sub_Db.CreateDynaset(sql)
    End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    If sub_Ds("sub_name") = "" Then
        flag2 = 0
        Msg = "Having no data!"
        MsgBox Msg
        sub_Ds.Close
        sub_Db.Close
        Exit Sub
    End If
    flag2 = 1
    If flag1 And flag2 Then
        command3.Enabled = True
    End If
    text6.Text = sub_Ds("sub_name")
    text7.Text = sub_Ds("sub_credit")
    sub_Ds.Close
    sub_Db.Close
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6.3.4 บทสรุปและวิจารณ์ความสามารถของวิชวลเบสิก

ในหัวข้อนี้จะเป็นการเปรียบเทียบความสามารถต่างๆ ระหว่างการพัฒนาระบบด้วย วิชวลเบสิก และ กุปต้า โดยจะใช้หัวข้อเดียวกันกับในหัวข้อที่ 6.2.4

6.3.4.1 ในด้านความง่ายของการพัฒนา

ในการพัฒนาแอพพลิเคชั่นโดยใช้วิชวลเบสิกนั้น ผู้พัฒนาระบบจะต้องมีพื้นฐานความรู้เกี่ยวกับตัวภาษาพอสมควรทีเดียว ซึ่งเมื่อเทียบกับการพัฒนาโดยใช้ กุปต้า แล้วจะดูเหมือนว่า กุปต้าจะพัฒนาได้ง่ายกว่า เพราะถ้าเป็นการพัฒนาระบบฐานข้อมูลที่ไม่ซับซ้อนแล้ว การพัฒนาโดยใช้ กุปต้า นั้นแทบที่จะไม่จำเป็นต้องเขียนตัว โค้ด เลย ซึ่งทำให้ผู้ที่ไม่มีพื้นฐานการโปรแกรมมาก่อนก็สามารถทำความเข้าใจและใช้งานได้อย่างรวดเร็วทีเดียว อย่างไรก็ตาม ถ้าผู้พัฒนาเป็นผู้ที่มีประสพการณ์ในการโปรแกรมมาบ้างแล้ว ผู้พัฒนาแบบนี้อาจจะชอบการใช้งานแบบวิชวลเบสิกมากกว่าเพราะยังคงรักษาระบบการโปรแกรมไว้ ซึ่งทำให้สามารถใส่การทำงานได้จาก โค้ด และยังให้ความยืดหยุ่นในการโปรแกรมมากกว่า กุปต้า

6.3.4.2 ในด้านเอ็นไวนรอนเมนท์ในการโปรแกรม

การพัฒนาจากการออกแบบและนำมาใช้กับวิชวลเบสิกสามารถประยุกต์เข้ากันได้ดีพอสมควรทีเดียว เราสามารถที่จะ แม็พ (Map) สเตทไดอะแกรม เข้ากับระบบติดต่อผู้ใช้ได้อย่างเหมาะเจาะ นอกจากนี้แล้วระบบช่วยเหลือต่างๆในการโปรแกรมก็ถูกจัดไว้อย่างดีทีเดียว เช่นการแบ่ง โค้ดการโปรแกรมออกเป็นส่วนๆ ตามเหตุการณ์ (Event) และ ระบบขอความช่วยเหลือ (Help) ของวิชวลเบสิคได้จัดทำไว้ได้ดีมาก

6.3.4.3 ในด้านของออพเจกต์โอเรียนเต็ดโปรแกรมมิ่ง

หัวข้อนี้คูจะเป็นจุดอ่อนของวิชวลเบสิก เพราะเหตุที่ว่า วิชวลเบสิกไม่ได้เป็นภาษาที่สนับสนุนการโปรแกรมแบบ ออพเจกต์โอเรียนเต็ด อย่างแท้จริง เพราะไม่สามารถทำการประกาศ คลาส โดยผู้พัฒนาเองได้ แต่อย่างไรก็ตามตัวภาษาก็ได้มีการบังคับให้การโปรแกรมของเรามีลักษณะเป็น ออพเจกต์โอเรียนเต็ดภายในตัว วิชวลเบสิกไม่สามารถทำการอินเฮียริแทนส์ได้

6.3.4.4 ในด้านการเข้าถึงของข้อมูล

การเข้าถึงข้อมูลวิชวลเบสิกดูจะมีภาษีกว่าเพราะว่า สามารถเข้าถึงฐานข้อมูลที่ไม่ใช่ ODBC ได้โดยไม่ต้องใช้ Router ในขณะที่กุปต้าจะต้องมี Router เพิ่มเข้ามา นอกจากนี้แล้วการติดต่อกับฐานข้อมูลแบบ ODBC วิชวลเบสิกจะไม่ต้องทำการ ล็อค ข้อมูลเอง โดยจะโอนหน้าที่เหล่านี้ให้กับฐานข้อมูลที่ทำการติดต่อด้วย ในขณะที่ กุปต้า ติดต่อกับฐานข้อมูล SQLBase การล็อคข้อมูลผู้พัฒนาจะต้องเป็นผู้จัดการเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สรุป

ในการพัฒนาระบบที่ต้องการ การติดต่อกับผู้ใช้ที่สวยงามแล้วการใช้ กุปต้า เป็นเครื่องมือในการพัฒนาดูที่จะเป็นหนทางเลือกที่ไม่เลวทีเดียว เพราะนอกจากจะมีความสวยงาม แล้ว ความง่ายในการพัฒนาระบบ และเครื่องมือต่างที่สนับสนุนการพัฒนาโดยมีผู้พัฒนาหลายคน กุปต้าได้สนับสนุนเต็มที่ อย่างไรก็ตามถึงแม้ว่ากุปต้าจะให้ความสะดวกสบายมากก็จริงแต่ถ้าผู้พัฒนาชอบการโปรแกรมแบบที่เป็นการโปรแกรมจริงๆ แล้วผู้พัฒนาคนนั้นอาจจะมาใช้วิชวลเบสิกเป็นเครื่องมือในการพัฒนาก็ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

วิทยานิพนธ์เล่มนี้สามารถสำเร็จลุล่วงไปได้เนื่องมาจากได้รับการช่วยเหลือจากบุคคลต่อไปนี้ ซึ่งผู้จัดทำวิทยานิพนธ์ต้องขอขอบพระคุณเป็นอย่างสูง

1. ผศ. ดร.ศุภมิตร จิตตะยโศธร ผู้ให้คำปรึกษา และ แนะนำแนวทาง
2. พี่ๆห้องรองคณบดีฝ่ายระบบสารสนเทศ
3. บริษัทจักรวาล คอมมิวนิเคชั่น จำกัด ผู้ให้การสนับสนุนซอฟแวร์
4. บุรัช ภัทรโกศล ผู้ให้ความช่วยสนับนนุนการใช้งานฐานข้อมูล SQLSever

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

๑. พิษณุเพทางค์ , "ตำราพรหมชาติ ฉบับชาวบ้าน" , สำนักหอสมุดกลาง 09

๒. Bruce Ring , "SQLWindows The Client/Server Application Development System" , Gupta Corporation Febury , 1993

๓. David Maher , "Team Windows User's Guide" , Gupta Corporation , March , 1993

๔. James Rumbaugh , Michael Blaha , William Premerlani , Frederick Eddy , William Lorensen , "Object-Oriented Modeling and Design" , PRENTICE HALL , 1991

๕. William Gietz , "SQLTalk/Windows User's Guide" , Gupta Technologies , September , 1989

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้