ระบบสารสนเทศแผนที่เชิงเวลา

TEMPORAL MAP INFORMATION SYSTEM

โดย

นางสาว สัจจพร ชื่นพิทยาธร

นาย สุวิทย์ สิทธิธรรมชารี

อาจารย์ที่ปรึกษา

อาจารย์ บัณฑิต พัสยา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2544

ปริญญานิพนธ์ปีการศึกษา 2544

ภาควิชา วิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบสารสนเทศแผนที่เชิงเวลา

TEMPORAL MAP INFORMATION SYSTEM

ผู้จัดทำ

1. นางสาว สัจจพร ชื่นพิทยาธร รหัสประจำตัว 41014450
2. นาย สุวิทย์ สิทธิธรรมชารี รหัสประจำตัว 41014493

_____ อาจารย์ที่ปรึกษา

(อาจารย์ บัณฑิต พัสยา)

ระบบสารสนเทศแผนที่เชิงเวลา

นางสาวสัจจพร ชื่นพิทยาธร
นายสุวิทย์ สิทธิธรรมชารี
อาจารย์บัณฑิต พัสยา อาจารย์ที่ปรึกษา
ปีการศึกษา 2544

บทคัดย่อ

ข้อมูลแผนที่เป็นข้อมูลที่สามารถเปลี่ยนแปลงได้ตลอดเวลา แต่การจัดเก็บข้อมูลโดยทั่วไปจะเก็บข้อมูลเฉพาะที่เป็นจริงเพียงสถานเดียว ทำให้เกิดความต้องการจัดเก็บข้อมูลแผนที่ในสถานต่างๆที่สัมพันธ์กับช่วงเวลาที่ข้อมูลนั้นเป็นจริงจึงทำให้ต้องจัดเก็บข้อมูลแผนที่แบบเชิงเวลา

การจัดเก็บข้อมูลเชิงเวลาด้วยฐานข้อมูลเชิงวัตถุจะช่วยลดความยุ่งยากที่เกิดการจัดเก็บข้อมูลเชิงเวลาด้วยฐานข้อมูลเชิงสัมพันธ์ เนื่องจากฐานข้อมูลเชิงวัตถุมีข้อดีที่เกิดจากแนวความคิดเชิงวัตถุ รวมทั้งยังสนับสนุนการทำงานกับระบบ (Application) ซึ่งออกแบบโดยใช้แนวคิดเชิงวัตถุ และสามารถนำคลาสไดอะแกรมของข้อมูลไปจัดเก็บลงในระบบจัดการฐานข้อมูลได้โดยตรง

ปริญญานิพนธ์ฉบับนี้จึงทำการศึกษาและสร้างต้นแบบคลาสไดอะแกรมของข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุด้วย UML โดยจะจัดเก็บข้อมูลแผนที่ทั้งที่เป็นข้อมูลเชิงพื้นที่และข้อมูลเชิงบรรยาย และนำคลาสไดอะแกรมต้นแบบไปจัดเก็บลงในระบบจัดการฐานข้อมูล Caché หลังจากนั้นจึงสร้างระบบแผนที่ขึ้นเพื่อทดสอบคลาสไดอะแกรมต้นแบบว่าสามารถทำการนำเข้าและสืบค้นข้อมูลแผนที่ได้อย่างถูกต้อง

TEMPORAL MAP INFORMATION SYSTEM

Sajjaporn Chenpittayatorn

Suwit Sittitamashree

Mr. Bandit Passaya Advisor

ABSTRACT

In natural way, map data has changed along the time but current database can store only the current state of data. Hence, we can't call back the former state. It may cause some query problems.

One way to solve this manner is to extend database to support temporal data. With advantages of object-oriented database, adding data structure with temporal semantics is not so hard to implement.

We handled this by created class diagram of temporalmap data model on OODB using UML. Class diagram contain geographic data and non- geographic data. We use OODBMS (Object Oriented Database Management System) Caché and use JAVA to manipulate data.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้คงไม่อาจเสร็จได้ด้วยดี หากไม่ได้รับความช่วยเหลือ และร่วมมือจากหลายๆฝ่ายด้วยกัน บุคคลแรกที่ต้องกล่าวถึงเพราะเป็นส่วนสำคัญที่ทำให้วิทยานิพนธ์นี้เสร็จลงได้ก็คือ อาจารย์ บัณฑิต พัสยา อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่ให้ความเอาใจใส่ แนะนำและช่วยเหลือเสมอมา ซึ่งต้องขอขอบพระคุณเป็นอย่างมาก

ขอบคุณบุคคลใกล้ชิดทุกคนที่มีส่วนช่วยให้คำปรึกษา ให้กำลังใจ และเป็นแรงผลักดันให้สามารถฝ่าฟันอุปสรรค์ต่างๆ จนสามารถประสบความสำเร็จในวันนี้ได้

นางสาว สัจจพร ชื่นพิทยาธร

นาย สุวิทย์ สิทธิธรรมชารี

สารบัญ

สารบัญตาราง

สารบัญรูปภาพ

บทที่ 1
บทนำ

1.1 ความสำคัญและที่มา

ตามธรรมชาติของข้อมูลแผนที่นั้นจะมีการเปลี่ยนแปลงไปตามเวลา จึงมีความจำเป็นในการจัดเก็บข้อมูลแผนที่เหล่านั้นด้วยฐานข้อมูลเชิงเวลา เพื่อรองรับการเรียกค้นและสามารถตอบปัญหาที่ต้องอ้างอิงกันข้อมูลในอดีต เช่น อัตราการเปลี่ยนแปลงขององค์ประกอบต่างๆในสภาพแวดล้อมทางแผนที่ที่ทำการศึกษา รวมทั้งยังสามารถวิเคราะห์เพื่อคาดการณ์ถึงสิ่งที่อาจเกิดขึ้นในอนาคตได้ ซึ่งสิ่งเหล่านี้ถือเป็นคุณสมบัติที่สำคัญของระบบสารสนเทศแผนที่เชิงเวลา เนื่องจากฐานข้อมูลของระบบสารสนเทศในปัจจุบันมักจัดเก็บบนฐานข้อมูลเชิงสัมพันธ์ ซึ่งการจัดเก็บข้อมูลเชิงเวลาบนฐานข้อมูลเชิงสัมพันธ์นั้น อาจเกิดปัญหาความซ้ำซ้อนของข้อมูล และมีความยุ่งยากในการจัดการมาก แต่หากใช้ฐานข้อมูลเชิงวัตถุในการจัดเก็บจะสามารถขจัดปัญหาและความยุ่งยากเหล่านั้นได้ รวมทั้งยังมีข้อดีตามแนวความคิดเชิงวัตถุและสามารถสนับสนุนการออกแบบโปรแกรมเชิงวัตถุได้ดีกว่าฐานข้อมูลเชิงสัมพันธ์เป็นอย่างมาก ด้วยคุณสมบัติตามแนวคิดเชิงวัตถุการเพิ่มโครงสร้างข้อมูลและโอเปอร์เรชัน สำหรับจัดการกับการข้อมูลแผนที่เชิงเวลาทำได้ไม่ยาก

งานวิจัยนี้จึงทำการศึกษาและออกแบบคลาสไดอะแกรมต้นแบบฐานข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุด้วยUML แล้วนำคลาสไดอะแกรมที่ได้ไปสร้าง ฐานข้อมูลบนระบบจัดการฐานข้อมูลเชิงวัตถุ Caché หลังจากนั้นจึงสร้างโปรแกมสำหรับตรวจสอบการทำงานของฐานข้อมูลต้นแบบว่าสามารถตอบคำถามข้อมูลแผนที่เชิงเวลาได้ตามที่ต้องการหรือไม่

1.2 วัตถุประสงค์ของงานวิจัย

1) ศึกษาวิธีการจัดเก็บข้อมูลเชิงพื้นที่
2) ศึกษาระบบฐานข้อมูลเชิงวัตถุ
3) ศึกษาระบบฐานข้อมูลเชิงเวลา
4) ศึกษาวิธีการออกแบบฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุ
5) ออกแบบฐานข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุโดยใช้ UML คลาสไดอะแกรม
6) ศึกษาซอฟท์แวร์จัดการระบบฐานข้อมูลเชิงออบเจ็กต์ Caché และศึกษาถึงวิธีการใช้ภาษาจาวาติดต่อ
7) นำต้นแบบของฐานข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุโดยใช้ UML (Unify Modeling Language) คลาสไดอะแกรมที่ได้ออกแบบ ไว้มาจัดเก็บลงใน Caché
8) ทำการสร้างแอพพลิเคชั่นขึ้นมาทดสอบต้นแบบที่ได้ออกแบบไว้ว่าสามารถตอบคำถามข้อมูล

แผนที่เป็นเชิงเวลาตามที่ต้องการได้หรือไม่

9) สรุปผลข้อดีข้อเสียของการออกนำต้นแบบที่ได้จากการออกแบบไว้เมื่อนำมาพัฒนาโดยใช้ Caché ร่วมกับภาษาจาวา

10) เสนอแนวทางในการพัฒนาต่อไป

1.3 ขอบเขตของงานวิจัย

โครงงานนี้ได้ทำการศึกษาวิธีการจัดเก็บข้อมูลเชิงพื้นที่ของระบบแผนที่สารสนเทศภูมิศาสตร์ โดยอ้างอิงจากโครงสร้างตารางที่ใช้ใน Spatial Cartridge ของโอราเคิล และนำความรู้ที่ได้จากการศึกษาวิธีการออกแบบฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุ มาทำการออกแบบฐานข้อมูลที่ใช้จัดเก็บข้อมูลแผนที่ของระบบสารสนเทศแผนที่เชิงเวลา

ซึ่งวิธีที่ใช้ในการออกแบบฐานข้อมูลนี้ได้ใช้คลาสไดอะแกรมของ UML เนื่องจากเป็นวิธีที่เหมาะกับการออกแบบระบบฐานข้อมูลเชิงออบเจ็กต์ และ OODBMS (Object-Oriented Database Management System) ในปัจจุบันสนับสนุนการแปลงคลาสไดอะแกรมเป็น Object Schema ที่ใช้ในการเก็บข้อมูลของระบบฐานข้อมูลเชิงออบเจ็กต์ได้ทันที

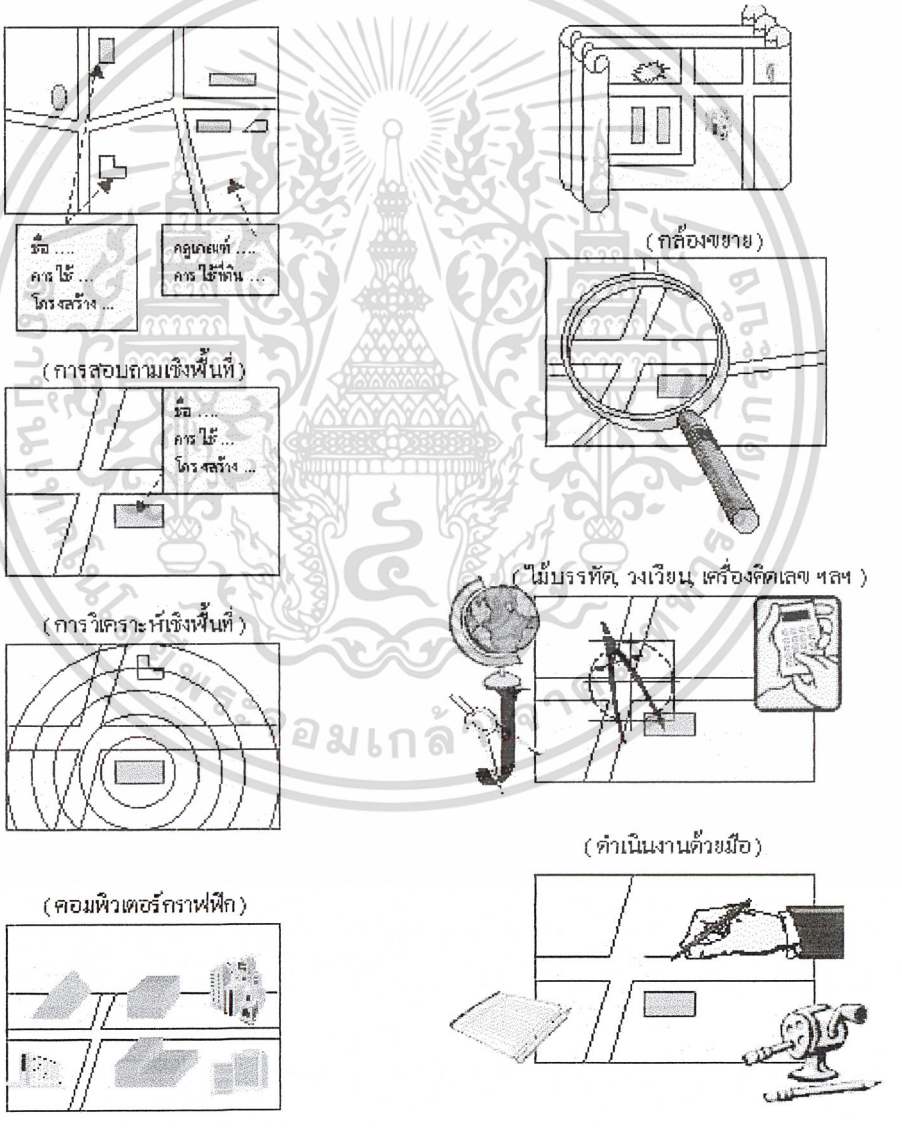
นอกจากนี้ยังต้องทดสอบฐานข้อมูลที่ได้ออกแบบไว้ โดยทำสิ่งที่ได้ออกแบบไว้มาใช้ในการจัดเก็บข้อมูลเชิงพื้นที่ซึ่งจะจัดเก็บใน Caché OODBMS และทำการเขียนแอพพลิเคชั่นโดยใช้ภาษาจาวาขึ้นมาทดสอบอีกด้วย

1.4 วิธีการดำเนินงาน

1) ศึกษาทฤษฎีของระบบแผนที่สารสนเทศภูมิศาสตร์ โดยศึกษาว่าการจัดเก็บข้อมูลเชิงพื้นที่ประกอบไปด้วยอะไรบ้าง และจะมีวิธีการเชื่อมต่อกับข้อมูลเชิงบรรยายได้อย่างไร
2) ศึกษาระบบฐานข้อมูลเชิงเวลา โดยศึกษาว่ามีหลักการและแนวความคิด ในการจัดเก็บข้อมูลธรรมดาให้เป็นข้อมูลเชิงเวลาได้อย่างไร
3) ศึกษาระบบฐานข้อมูลเชิงวัตถุ โดยศึกษาถึงหลักการและแนวความคิด ข้อได้เปรียบที่มีมากกว่าฐานข้อมูลเชิงสัมพันธ์ในการออกแบบฐานข้อมูลที่มีข้อมูลเชิงเวลาอยู่ด้วย และวิธีการออกแบบฐานข้อมูลเชิงวัตถุที่มีข้อมูลเชิงเวลา
4) ออกแบบฐานข้อมูลของระบบสารสนเทศแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุ
5) ทำการจัดเก็บและพัฒนาสิ่งได้ออกแบบมาบนฐานข้อมูล Caché
6) ทำการสร้างแอพพลิเคชั่นขึ้นมาทดสอบ
7) สรุปผลสิ่งที่ได้ทำมา และเสนอแนวทางในการพัฒนาต่อไป

บทที่ 2
ระบบสารสนเทศแผนที่เชิงเวลา (Temporal Map)

ระบบสารสนเทศแผนที่ทางภูมิศาสตร์ เป็นระบบเครื่องมือที่มีประสิทธิภาพในการรวบรวมข้อมูลเชิงพื้นที่ และเชื่อมโยงผสมผสานข้อมูลเชิงพื้นที่ และข้อมูลเชิงบรรยายที่เก็บไว้ในฐานข้อมูล อีกทั้งยังสามารถดัดแปลงแก้ไข,วิเคราะห์,แสดงผลการวิเคราะห์ และการนำเสนอข้อมูล เพื่อให้เห็นมิติและความสัมพันธ์ด้านพื้นที่ของข้อมูล ซึ่งมีส่วนช่วยให้เกิดความเข้าใจปัญหา และประกอบการตัดสินใจในการปัญหาเกี่ยวกับการวางแผนการใช้ทรัพยากรเชิงพื้นที่



รูปภาพ 2-1 การจัดการสารสนเทศภูมิศาสตร์ด้วย GIS เปรียบเทียบกับบนกระดาษ

2.1 ประเภทของข้อมูลในระบบสารสนเทศแผนที่ทางภูมิศาสตร์

ในระบบสารสนเทศแผนที่ทางภูมิศาสตร์ประกอบไปด้วยข้อมูลสองส่วนที่สำคัญคือ ข้อมูลที่ไม่อยู่ในเชิงพื้นที่ (Non-Spatial Data) หรือ ข้อมูลเชิงบรรยาย และข้อมูลเชิงพื้นที่ (Spatial Data) ซึ่งระบบสารสนเทศแผนที่จะต้องทำการเชื่อมต่อข้อมูลทั้งสองอย่างนี้เข้าด้วยกัน เพื่อทำให้ข้อมูลเชิงบรรยายนั้นสามารถแสดงข้อมูลรูปของข้อมูลเชิงพื้นที่ซึ่งแสดงถึงลักษณะของข้อมูลเชิงบรรยายนั้นได้

2.1.1 ข้อมูลที่ไม่อยู่ในเชิงพื้นที่

เป็นข้อมูลที่เกี่ยวกับคุณลักษณะต่าง ๆ ที่เกี่ยวข้องกับพื้นที่นั้น ๆ (Associated Attributes) ข้อมูลเหล่านี้ได้แก่ ข้อมูลประชากร และลักษณะของพื้นที่ เช่นเขตบึงกุ่ม มี 3 ตำบลมีพื้นที่ทั้งหมด 69.903 ตารางกิโลเมตร

2.1.2 ข้อมูลเชิงพื้นที่

ข้อมูลเชิงพื้นที่เป็นข้อมูลที่ระบุตำแหน่งพิกัดที่ตั้ง ข้อมูลประเภทนี้เป็นสิ่งที่จำเป็นอย่างยิ่งเพราะระบบสารสนเทศข้อมูลแผนที่ทางภูมิศาสตร์เป็นระบบข้อมูลที่ต้องอ้างอิงภูมิศาสตร์ (Geo-Referenced) ข้อมูลเหล่านี้ได้แก่ แผนที่ต่าง ๆ เป็นต้น

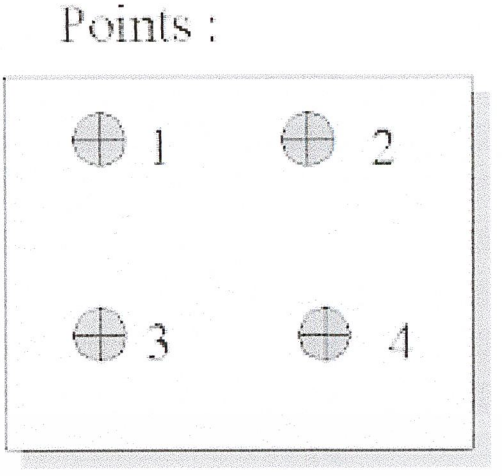
2.1.2.1 ข้อมูลแบบเวคเตอร์ (Vector Model)

ข้อมูลแบบเวคเตอร์ คือข้อมูลเชิงพื้นที่ถูกจัดเก็บในลักษณะเชิงเส้นที่มีโครงสร้างในการกำกับก่อน-หลัง หรือ ซ้าย-ขวา โดยการใช้เส้น และจุดเป็นองค์ประกอบพื้นฐานในการจัดเก็บข้อมูลเชิงพื้นที่ โครงสร้างของแฟ้มข้อมูลระบบสารสนเทศแผนที่ทางภูมิศาสตร์ในระบบนี้จะประกอบด้วยเครื่องหมายประจำตัว (ID) ตำแหน่งพิกัด X ,Y และ ตัวชี้ลำดับก่อน-หลัง หรือ ซ้าย-ขวา ของข้อมูลข้างเคียง โครงสร้างของข้อมูลระบบนี้จะใช้เนื้อที่ในการจัดเก็บน้อย แต่การปรับปรุงแก้ไขจะทำได้ยาก และไม่สะดวกเท่าที่ควร ข้อมูลเชิงพื้นที่ในรูปเวคเตอร์จะเก็บอยู่ใน 4 รูปแบบดังต่อไปนี้

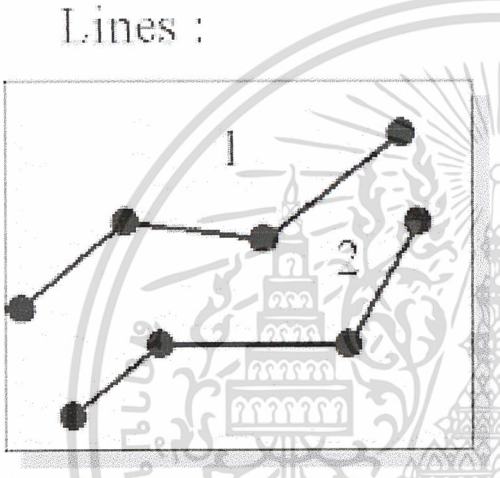
- จุด (Point) เป็นค่าพิกัดในตำแหน่งแนวนอน (X axis) และแนวตั้ง (Y axis) ไม่มีความยาวหรือพื้นที่จะใช้แสดงข้อมูลบนพื้น โลกที่เป็นลักษณะของตำแหน่งที่ตั้งเช่น ที่ตั้งของมหาวิทยาลัย เป็นต้น

- เส้น (Line) เป็นชุดของค่าพิกัดตำแหน่งที่ต่อเนื่องกัน โดยมีจุดเริ่มต้นและจุดปลายมีเฉพาะความยาวไม่มีพื้นที่ จะใช้แสดงข้อมูลบนพื้นโลกที่เป็นลักษณะของเส้นเช่น เส้นทางคมนาคม , ขอบเขตการปกครอง

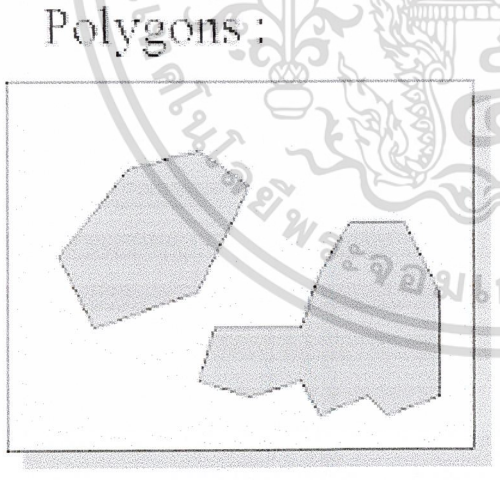
- พื้นที่รูปปิด (Polygon) เป็นชุดของค่าพิกัดตำแหน่งที่ต่อเนื่องกัน โดยมีจุดเริ่มต้นและจุดปลายอยู่ที่ตำแหน่งเดียวกันมีทั้งความยาว (เส้นรอบรูป) และพื้นที่จะใช้แสดงข้อมูลที่เป็นลักษณะของพื้นที่ เช่น พื้นที่ของบริเวณแหล่งปลูกข้าว เป็นต้น



รูปภาพ 2-2แบบจำลองโดยใช้จุด เช่น ที่ตั้งหมู่บ้าน บ่อน้ำ โรงเรียน ฯลฯ



รูปภาพ 2-3แบบจำลองโดยใช้ลายเส้น เช่น ถนน ทางน้ำ สายไฟฟ้า ฯลฯ



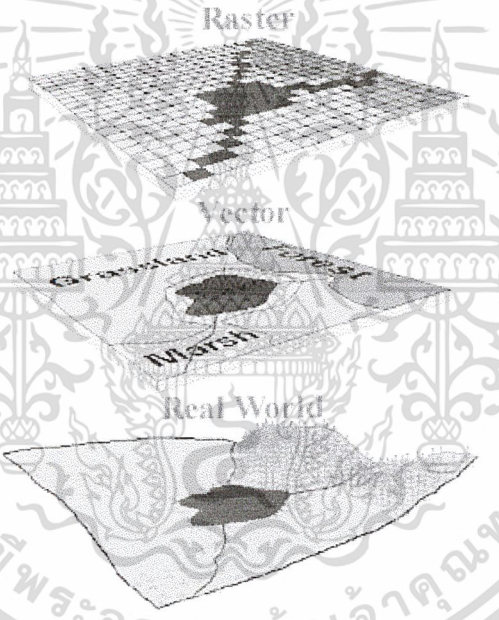
รูปภาพ 2-4แบบจำลองโดยใช้เส้นขอบเขตรูปปิด เช่น ชุดดิน สภาพการใช้ที่ดิน ขอบเขตการปกครอง ฯลฯ

2.1.2.2 ข้อมูลแบบตารางกริด (Raster Model)

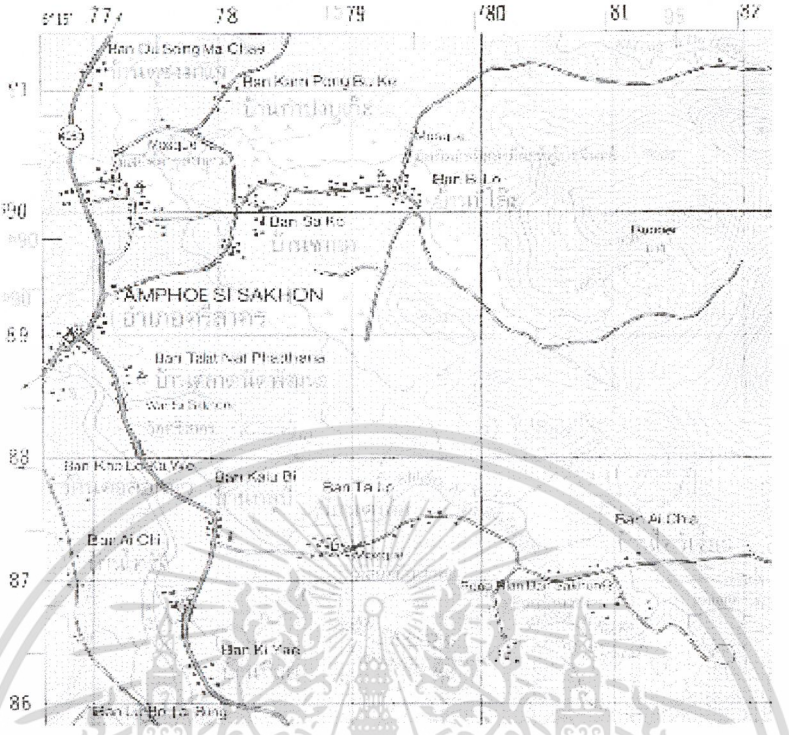
ระบบสารสนเทศทางแผนที่ที่จัดเก็บข้อมูลในลักษณะตารางกริดจะแบ่งพื้นที่ออกเป็นตารางกริดที่มีรูปเป็นสี่เหลี่ยมจัตุรัสเล็ก ๆ จำนวนมาก โดยในรูปสี่เหลี่ยมจัตุรัสเล็ก ๆ เหล่านี้มีศัพท์เรียกเฉพาะว่าหน่วยภาพย่อย (Picture Element) หรือนิยมเรียกย่อๆ ว่า Pixel โดยที่แต่ละ Pixel จะเป็นหน่วยที่เล็กที่สุด

ของข้อมูล ถ้าข้อมูลที่มีความละเอียดสูง ขนาดของ Pixel ก็จะมีขนาดเล็ก แต่ถ้าข้อมูลที่ใช้ในงานสารสนเทศค่อนข้างหยาบ ขนาดของ Pixel จะมีขนาดใหญ่ ข้อดีของระบบข้อมูลแบบตารางกริดนี้ก็คือ ภายหลังจากการจัดเก็บแล้ว สามารถแก้ไขข้อมูลได้ง่าย สะดวก รวดเร็ว และมีประสิทธิภาพ แต่ข้อเสียของข้อมูลระบบนี้ก็คือ ต้องการแฟ้มข้อมูลขนาดใหญ่ เพื่อการจัดเก็บ หน่วยภาพย่อย ทั้งหมดในพื้นที่ ตัวอย่างของข้อมูลในระบบตารางกริด ได้แก่ ข้อมูลจากภาพถ่ายดาวเทียม

- ตารางกริดย่อย (cell) เรียงลำดับตามแถวจากมุมบนซ้ายไปขวา
- แต่ละตารางกริดย่อยบรรจุข้อมูลได้เพียงค่าเดียว
- ตารางกริดย่อยแต่ละช่องจะมีค่าตำแหน่งพิกัด
- ตารางกริดหนึ่งชุดเรียกว่า ชั้น (Layer)
- ฐานข้อมูลหนึ่งอาจจะประกอบด้วยชั้นหลายๆ ชั้น เช่น ชั้นของการใช้ที่ดิน (Landuse Layer) ชั้นของเส้นค่าความสูง (Contour Layer)



รูปภาพ 2-5ระบบสารสนเทศแผนที่แบบเชิงเส้น และเชิงตารางกริด



รูปภาพ 2-6ข้อมูลเชิงพื้นที่

File Edit Options Compute Window Help

J:12

*	A	B	C	D	E	
1	พิกัด x	พิกัด y	ความลึก (ม.)	จังหวัด	อำเภอ	ตำบ
2	748300	711300	-35	ยะลา	เมืองยะลา	ปุโร
3	748300	721100	-36	ยะลา	เมืองยะลา	ท่าส
4	748600	722000	-35	ยะลา	เมืองยะลา	ท่าส
5	748600	727900	-35	ยะลา	เมืองยะลา	ยุโป
6	749600	692100	-29	ยะลา	บันนังสตา	บันนั
7	750600	724500	-21	ยะลา	เมืองยะลา	ท่าส
8	751000	728000	-15	ยะลา	เมืองยะลา	สะเต
9	751500	709000	-26	ยะลา	เมืองยะลา	สะเอ
10	751600	748700	-21	ปัตตานี	ยะรัง	ยะรั

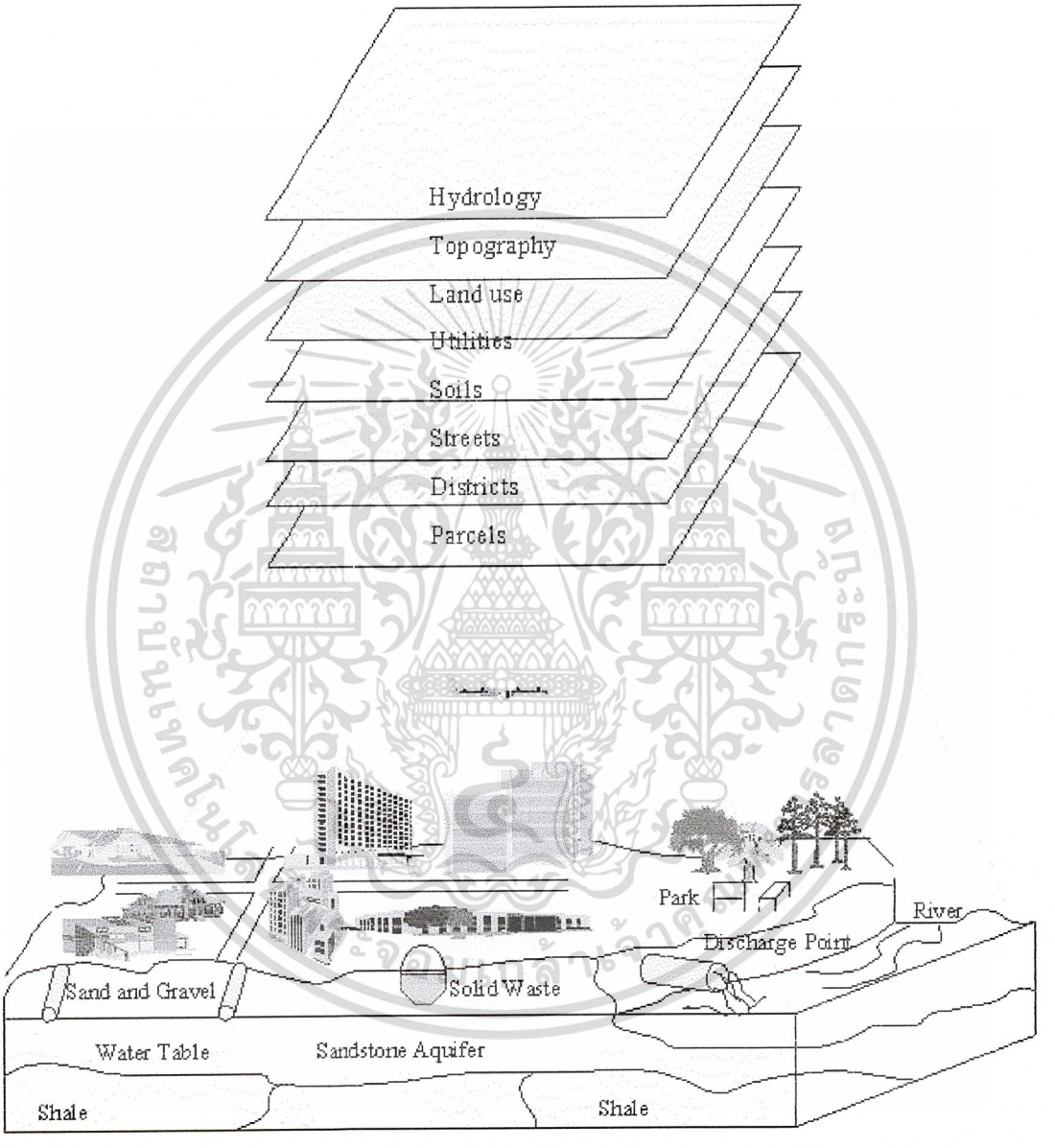
รูปภาพ 2-7ข้อมูลที่ไม่อยู่ในเชิงพื้นที่

	แบบเวคเตอร์	แบบตารางกริด
ข้อดี	- แสดงโครงสร้างข้อมูลเชิงปรากฏการณ์ได้ดี - โครงสร้างข้อมูลกะทัดรัด - ความเชื่อมโยงทางโทโพโลยีสามารถทำได้ครบถ้วน ด้วยการเชื่อมโยงแบบเครือข่าย - มีความถูกต้องในเชิงการฟิก - สามารถทำการสืบค้น การแก้ไข และการวางนัยทั่วไปกับข้อมูลการฟิกและลักษณะประจำได้	- มีโครงสร้างข้อมูลง่ายๆ - การวางซ้อนและการรวมข้อมูลแผนที่กับข้อมูลที่รับรู้จากระยะไกลทำได้ง่าย - การวิเคราะห์ทางพื้นที่ในแบบต่างๆ ทำได้ง่าย - การทดสอบด้วยการจำลองสถานการณ์ทำได้ง่าย เพราะหน่วยพื้นที่แต่ละหน่วยมีรูปร่างและขนาดเท่ากัน - เทคโนโลยีราคาถูกและกำลังมีการพัฒนาอย่างจริงจัง
ข้อเสีย	- โครงสร้างข้อมูลซับซ้อน - การรวมแผนที่แบบเวคเตอร์หลายๆ แผ่นหรือรวมแผนที่เวคเตอร์กับตารางกริดด้วยวิธีวางซ้อนมีความยุ่งยาก - การทดสอบด้วยการจำลองสถานการณ์ทำได้ยาก เพราะแต่ละหน่วยของแผนที่มีโครงสร้างโทโพโลยี - การแสดงและการเขียนเป็นแผนที่เสียค่าใช้จ่ายสูง โดยเฉพาะเมื่อต้องการแสดงสีและสัญลักษณ์ที่มีคุณภาพสูง - เทคโนโลยีชนิดนี้ที่ราคาแพง โดยเฉพาะถ้าต้องใช้ฮาร์ดแวร์ และซอฟต์แวร์ที่มีความซับซ้อน - การวิเคราะห์พื้นที่ และการกรองรายละเอียดภายในรูปหลายเหลี่ยมเกือบจะเป็นไปไม่ได้	- ข้อมูลการฟิกมีขนาดใหญ่ - การใช้ช่องกริดขนาดใหญ่เพื่อลดปริมาตรข้อมูล ทำให้สูญเสียโครงสร้างข้อมูลเชิงพื้นที่และเป็นการสูญเสียข้อมูลสารสนเทศอย่างมาก - แผนที่ตารางกริดที่หยาบจะไม่สวยเท่าแผนที่ซึ่งเขียนด้วยเส้น - การสร้างเครือข่ายเชื่อมโยงทำได้ยาก - การแปลงเส้นโครงแผนที่ต้องใช้เวลามากเว้นแต่จะใช้ขั้นตอนวิธีหรือฮาร์ดแวร์พิเศษ

ตาราง 2-1เปรียบเทียบข้อดีข้อเสียของการจัดเก็บข้อมูลแบบเวคเตอร์และแบบตารางกริด

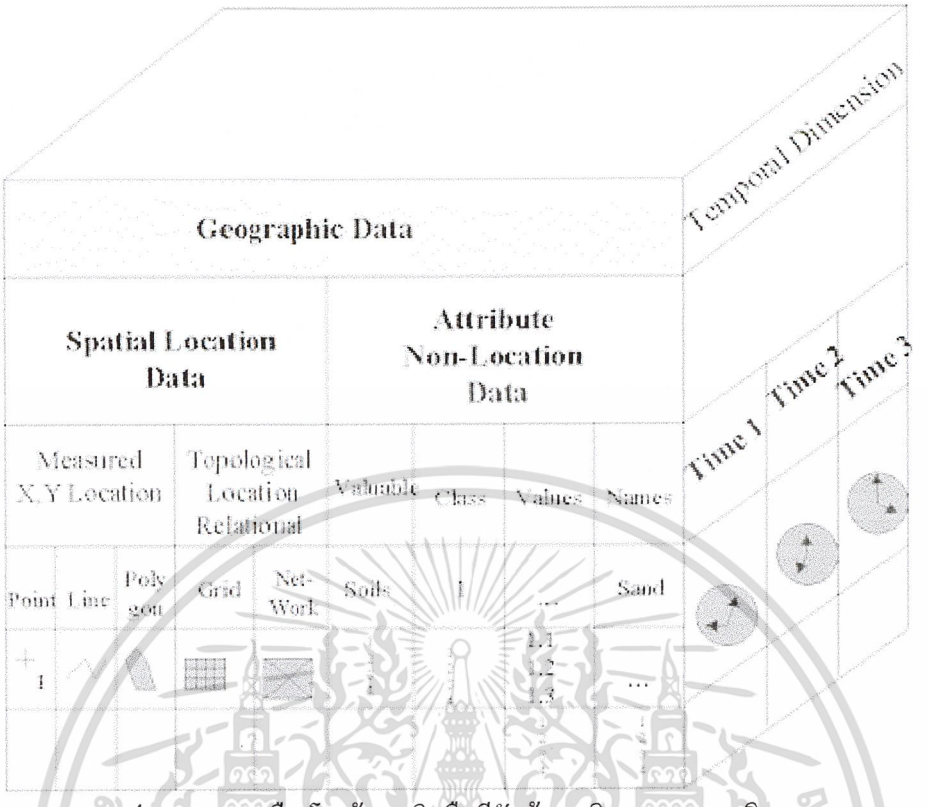
2.2 แบบจำลองข้อมูลของระบบสารสนเทศ

ข้อมูลเชิงพื้นที่ต่าง ๆ ที่ต้องการศึกษาจะถูกจัดเก็บ/นำเข้าเป็นชั้น ๆ ตามประเภทของสัญลักษณ์หรือแยกตามเนื้อหา (Aspect) ของข้อมูล เช่น ถนน แม่น้ำ แม้จะใช้สัญลักษณ์ที่เป็นลายเส้นเหมือนกัน แต่มักจะแยกเก็บในคนละชั้นข้อมูล เป็นต้น



รูปภาพ 2-8แบบจำลองลักษณะทางภูมิศาสตร์ในลักษณะ 2 มิติ

การจำลองลักษณะทางภูมิศาสตร์ที่ปรากฏบนพื้นผิวโลกของแผนที่/หลาย ๆ ชั้นในลักษณะ 2 มิติ เมื่อมีการจัดเก็บในฐานข้อมูลแผนที่จะสามารถเชื่อมโยงกับข้อมูลบรรยาย (Attribute Data) และข้อมูลในมิติของเวลาที่แตกต่างกัน ดังภาพการเชื่อมโยงข้อมูลเชิงพื้นที่ และข้อมูลบรรยายที่เกี่ยวข้องโดยการใช้ความสัมพันธ์และการจัดเก็บข้อมูลดังนี้



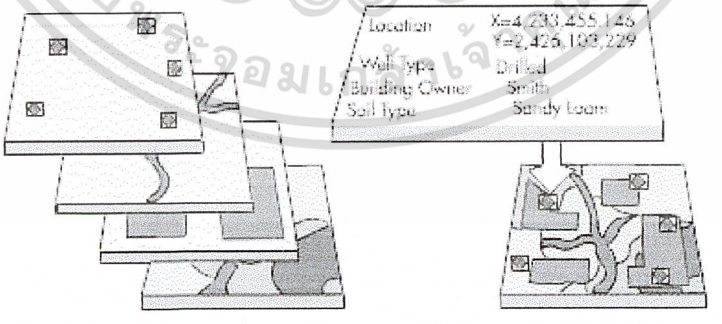
รูปภาพ 2-9การเชื่อมโยงข้อมูลเชิงพื้นที่กับข้อมูลเชิงบรรยายและเชิงเวลา

2.3 คำถามที่สามารถตอบได้ด้วยระบบสารสนเทศแผนที่ทางภูมิศาสตร์

คำตอบที่สามารถตอบได้ด้วยระบบสารสนเทศแผนที่ทางภูมิศาสตร์มีดังต่อไปนี้

2.3.1 ทำเลที่ตั้ง (Location)

questions - Location - What is at...?



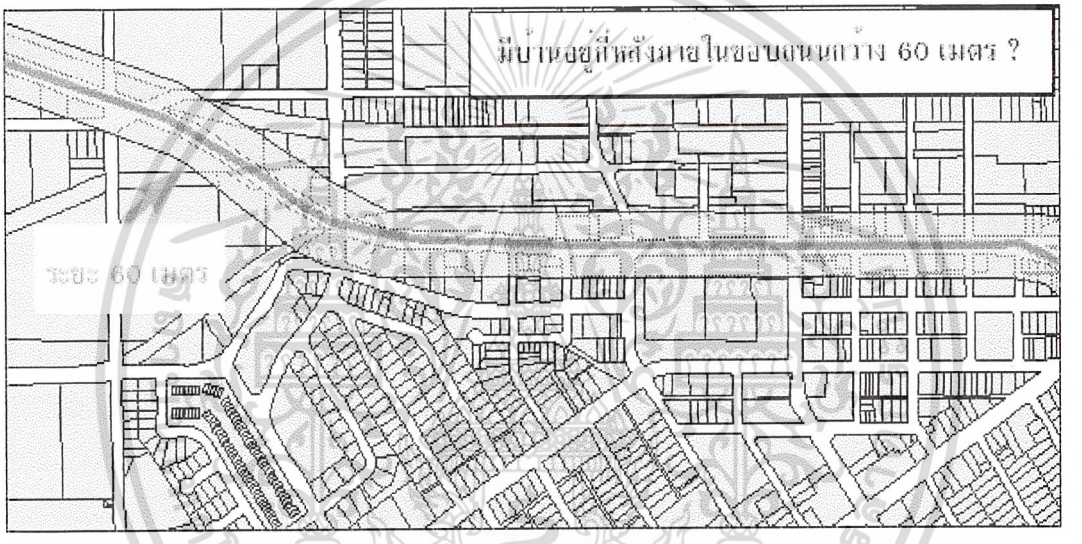
รูปภาพ 2-10ทำเลที่ตั้ง

เช่นอะไรอยู่ที่ไหน (ตำแหน่งที่ระบุ) ได้แก่การค้นหาว่ามีอะไรอยู่ที่ตำแหน่งที่ตั้งแห่งใดแห่งหนึ่ง โดยเฉพาะการระบุบรรยายตำแหน่งที่ตั้งทำได้หลายอย่าง เช่น ชื่อสถานที่ รหัสไปรษณีย์ หรือพิกัดทางภูมิศาสตร์

2.3.2 สภาพการณ์หรือเงื่อนไข (Condition)

questions - Conditions - Where is it...?

เป็นคำถามที่ตรงข้ามกับคำถามแรก และต้องอาศัยการวิเคราะห์เชิงพื้นที่ในการตอบคำถามดังกล่าว เช่น แทนที่จะค้นหาว่ามีอะไรอยู่ในตำแหน่งที่กำหนด แต่ต้องการจะทราบว่าตำแหน่งใดที่ตรงตามเงื่อนไขที่กำหนด เช่น เขตที่อยู่อาศัยบริเวณใดที่ราคาประเมินต่ำกว่าไร่ละ 2 ล้านบาท และอยู่ห่างจากถนนไม่เกิน 200 เมตร

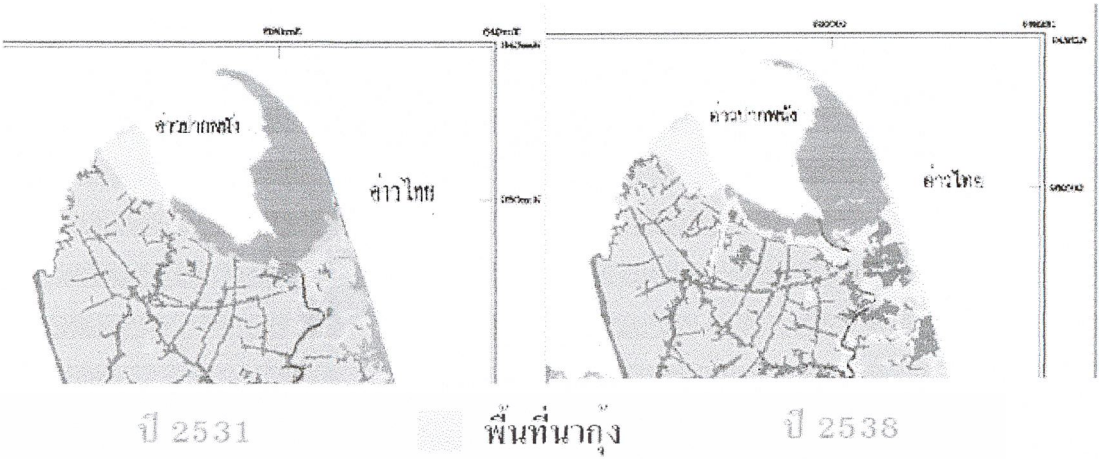


รูปภาพ 2-11สภาพการณ์หรือเงื่อนไข

2.3.3 แนวโน้มการเปลี่ยนแปลงอะไรบ้าง นับตั้งแต่...? (Trends)

เป็นการค้นหาลักษณะของการเปลี่ยนแปลงของเหตุการณ์ทางภูมิศาสตร์ในช่วงเวลาหนึ่ง เช่น วิเคราะห์หาจำนวนพื้นที่ป่าไม้ที่ถูกทำลายและในอัตราที่เพิ่มขึ้นหรือลดลงในช่วงเวลาที่กำหนด

questions - Trends - What has changed since...?



รูปภาพ 2-12แนวโน้มการเปลี่ยนแปลง

2.3.4 รูปแบบของสถานการณ์ (Pattern)

เป็นการวิเคราะห์ข้อมูลปัจจัยที่อยู่รอบๆ เหตุการณ์ที่เกิดขึ้นเพื่อให้ได้รายละเอียด และเงื่อนไขที่เกี่ยวข้องเช่นเกิดอุบัติเหตุบ่อยครั้งที่ทางแยกใด และพบเงื่อนไขอะไรที่ทางแยกนั้นหรือไม่ หรือคำถามที่ว่าโรคมะเร็งนั้นเป็นสาเหตุหลักของการตายของผู้คนที่อาศัยอยู่ในบริเวณโรงไฟฟ้านิวเคลียร์หรือไม่

GIS questions - Patterns - Which data are related...?

รูปภาพ 2-13รูปแบบของสถานการณ์

2.3.5 แบบจำลอง หรือเกิดอะไรขึ้น ถ้า...? (Modelling - What if)

เป็นการสร้างความสัมพันธ์ของข้อมูลเพื่อหาสิ่งที่เกิดขึ้น เช่น ถ้ามีถนนสายใหม่เข้ามาในโครงข่าย หรือหาข้อสรุปทางเลือกในเงื่อนไขที่กำหนด เช่น การหาสถานที่เป้าหมายในการทำธุรกิจที่มีเงื่อนไขต่าง ๆ เช่น สามารถเดินทางถึงได้ในเวลา 10, 15 และ 20 นาที รายได้ของประชากรในบริเวณนั้นอยู่ในเกณฑ์ที่กำหนด และมีการแข่งขันในธุรกิจประเภทเดียวกันในระดับไม่รุนแรงนัก เป็นต้น

GIS questions - Modelling - What if ...?

รูปภาพ 2-14แบบจำลอง หรือเกิดอะไรขึ้น ถ้า...?

2.4 การนำระบบสารสนเทศแผนที่ทางภูมิศาสตร์

เราสามารถนำระบบสารสนเทศแผนที่ทางภูมิศาสตร์มาประยุกต์ใช้งานในด้านต่างๆ ได้ดังต่อไปนี้

2.4.1 การประยุกต์ใช้งานด้านการคมนาคมขนส่ง

เป็นการนำเอาระบบสารสนเทศแผนที่ทางภูมิศาสตร์ไปใช้ในการดูแล และจัดการซ่อมแซมถนน โดยที่จะต้องเก็บข้อมูลเส้นถนนต่างๆ ลงไปรวมทั้งแอตทริบิวต์เกี่ยวกับการซ่อมบำรุงถนนแต่ละสายโดยเมื่อถึงเวลาที่จะทำการเลือกถนนที่จะต้องซ่อมบำรุงก็สามารถที่จะเลือก และแสดงข้อมูลของเส้นถนนที่ต้องการซ่อมบำรุงแล้วสร้างแผนที่หรือส่งรายงานให้ฝ่ายซ่อมบำรุงทราบว่าต้องซ่อมถนนเส้นใดบ้างเมื่อทำการซ่อมเสร็จก็จะต้องปรับแก้ข้อมูลถนนให้ถูกต้อง อีกตัวอย่างของการนำเอาระบบสารสนเทศแผนที่ทางภูมิศาสตร์มาใช้ได้แก่ การกำหนดเส้นทางการเดินรถประจำทางใหม่ซึ่งจะต้องให้บริการประชาชนได้มากที่สุด โดยจะต้องมีการเลือกเส้นทางที่ได้มีการวางแผนไว้แล้ว ซึ่งจะมีมาตรฐานกำหนดไว้ว่าผู้ที่มาใช้บริการจะต้องอยู่ห่างจากเส้นทางไม่เกิน 1 กิโลเมตร ดังนั้นจึงต้องนำเอาข้อมูลจำนวนประชากร จากนั้นทำการเปรียบเทียบจำนวนประชากรที่เส้นทางแต่ละเส้นให้บริการได้แล้วเลือกเส้นทางที่ให้บริการมากที่สุด

2.4.2 การประยุกต์ใช้งานเกี่ยวกับการวางแผนการขยายพื้นที่ของสนามบิน

เป็นการนำเอาระบบสารสนเทศแผนที่ทางภูมิศาสตร์ไปวิเคราะห์หาว่าจะเลือกแผนผังใดที่เหมาะสมต่อการขยายตัวของสนามบินโดยมีเงื่อนไขว่าจะต้องมีเสียงดังรบกวนประชาชนให้น้อยที่สุด และมีผลกระทบกับจำนวนประชากรที่ต้องย้ายออกไปให้น้อยที่สุดเมื่อนำเอาแบบแปลนต่างๆ มาวิเคราะห์กับข้อมูลประชากรแล้วก็จะเห็นได้ว่าแบบแปลนใดที่เหมาะสมที่สุดโดยพิจารณาจากผลกระทบข้างต้น

2.4.3 การประยุกต์ใช้งานเกี่ยวกับการประเมินอาชญากรรม

จะเป็นการจัดเก็บข้อมูลของอาชญากรรมประเภทต่างๆ ที่เกิดขึ้น เพื่อพิจารณาถึงความสัมพันธ์ของประเภทอาชญากรรมและช่วงเวลาที่เกิด เมื่อเห็นความสัมพันธ์ของทั้งสองสิ่งแล้วเราก็จะทราบว่าในบริเวณใดมีอาชญากรรมเกิดขึ้นบ่อยในช่วงเวลาที่กำหนดทำให้ทางตำรวจสามารถที่จะกระจายกำลังออกไปดูแลความสงบได้ถูกต้องมากยิ่งขึ้น

2.4.4 การประยุกต์ใช้งานทางด้านการว่างแผนการใช้ที่ดิน

ทำการนำข้อมูลทางด้านการใช้ที่ดินเข้าสู่ระบบ แล้วก็ทำการเปรียบเทียบดูกับพื้นที่และข้อมูลอื่นๆ ดูว่ามีพื้นที่ใดบ้างที่ควรจะต้องเปลี่ยนแปลงการใช้ที่ดินเพื่อให้ผลประโยชน์มากที่สุด

2.4.5 การประยุกต์ใช้ทางด้านสิ่งแวดล้อม

การดูผลกระทบของฝนกรด (Acid rain) ที่จะมีต่อพื้นที่ป่าซึ่งเราทราบว่าฝนกรดมีผลกระทบต่อพันธ์ไม้แต่ละประเภทแตกต่างกันไป ทำให้เราสามารถที่จะว่างแผนที่จะป้องกันฝนกรดได้หรือการตรวจการย้ายถิ่นฐานของหมีขั้วโลกในช่วงฤดูหนาว ซึ่งจะไม่มีแสงแดดเลยทำให้นักชีววิทยาไม่สามารถที่จะตรวจสอบการย้ายถิ่นฐานของหมี ดังนั้นนักชีววิทยาจึงใช้เครื่อง GPS ผูกติดกับคอของหมีเพื่อให้ระบบ GPS ส่งค่าตำแหน่งพิกัดของหมี

ออกมาทุกๆ ช่วงเวลาที่กำหนดแล้วนำข้อมูลที่ได้จากระบบ GPS เข้าสู่ระบบสารสนเทศแผนที่ทางภูมิศาสตร์ก็จะได้ข้อมูลการเคลื่อนย้ายถิ่นฐานของหมีได้

2.4.6 การประยุกต์ใช้งานทางด้านการแบ่งขอบเขต

ในประเทศอเมริกาจะมีการแบ่งขอบเขต สำหรับการเลือกตั้งเพื่อที่จะทำให้จำนวนประชากรของแต่ละเขตมีจำนวนเท่าๆ กัน โดยที่ทางอเมริกาจะมีการสำรวจสัมมโนประชากรทุกๆ 10 ปี ทำให้เราทราบจำนวนของประชากรที่ถูกต้องได้

2.4.7 การประยุกต์ใช้ทางด้านเกี่ยวกับการหาพื้นที่

ใช้หาพื้นที่ที่น่าจะมีผลผลิตสูงแล้วทำการสร้างระบบชลประทานจะเป็นการนำเอาข้อมูลต่างๆ ที่มีผลต่อการเจริญเติบโตของพืช ซึ่งเป็นวัตถุดิบในการผลิตมาทำการวิเคราะห์หาพื้นที่ที่น่าจะมีผลผลิตสูงเพื่อใช้ในการตัดสินใจในการสร้างระบบชลประทานเพื่อส่งเสริมผลผลิตของพื้นที่นี้ให้มากขึ้น

บทที่ 3
ฐานข้อมูลเชิงวัตถุและฐานข้อมูลเชิงเวลา

เนื้อหาในบทนี้จะกล่าวถึงความรู้พื้นฐาน ของระบบฐานข้อมูลเชิงวัตถุ และฐานข้อมูลที่จำเป็นต้องใช้ในการออกแบบสร้างระบบฐานข้อมูลแผนที่เชิงเวลา โดยจะกล่าวถึงแนวความคิดเชิงวัตถุพื้นฐาน ระบบจัดการฐานข้อมูลเชิงวัตถุ ระบบฐานข้อมูลเชิงเวลา และวิธีการออกแบบฐานข้อมูลเชิงวัตถุบนฐานข้อมูลเชิงเวลา

3.1 แนวความคิดเชิงวัตถุ

3.1.1 ออบเจ็กต์ (Object)

ออบเจ็กต์คือ Abstract Representation ของ Real World Entity ที่มี

- Unique Identity: ออบเจ็กต์ต้องมีสิ่งที่ระบุถึงความแตกต่างจากออบเจ็กต์อื่น
- Embedded Properties
- ความสามารถในการติดต่อสื่อสารกับออบเจ็กต์อื่นและตัวเอง

3.1.2 OID

Object Identity หรือ Object Identifier (OID) เป็นสิ่งที่ใช้ระบุหรืออ้างอิงออบเจ็กต์ ซึ่งแต่ละออบเจ็กต์จะมี OID ไม่ซ้ำกัน OID จะถูกกำหนดให้โดยระบบตั้งแต่เมื่อออบเจ็กต์ถูกสร้างขึ้นและจะไม่สามารถเปลี่ยนได้

OID จะแตกต่างจาก Primary key (PK) ของระบบฐานข้อมูลเชิงสัมพันธ์ เพราะ PK จะขึ้นอยู่กับค่าของแอททริบิวต์ (Attribute) ที่ผู้ใช้กำหนด (user-given value) ซึ่งสามารถเปลี่ยนแปลงในภายหลังได้ แต่ OID ถูกกำหนดโดยระบบซึ่งจะไม่ขึ้นกับค่าแอททริบิวต์ของออบเจ็กต์และเปลี่ยนแปลงไม่ได้ เมื่อออบเจ็กต์ถูกลบออกไปจากระบบ

OID ของออบเจ็กต์นั้นก็จะถูกลบตามไปด้วยและจะไม่มีการนำ OID ที่ถูกลบไปแล้วกลับมาใช้ใหม่ ค่าของ OID จะไม่ผูกติดกับตำแหน่งทางกายภาพ (Physical Address) ของหน่วยความจำถาวร (Permanent Memory) ทำให้ระบบเชิงวัตถุมีความเป็นอิสระของข้อมูลทางกายภาพ (Physical Data Independence)

3.1.3 แอททริบิวต์ (Attributes หรือ Instance Variables)

แอททริบิวต์ (Attribute) หรือ Instance Variable ก็คือ ข้อมูลของออบเจ็กต์ อาจเป็นได้ทั้งข้อมูลชนิดพื้นฐานหรือเป็นออบเจ็กต์ก็ได้

3.1.4 สถานะของออบเจ็กต์ (Object State)

สถานะของออบเจ็กต์ขึ้นกับค่าของแอททริบิวต์ของออบเจ็กต์ ณ เวลาที่กำหนด ถึงแม้ว่าสถานะของออบเจ็กต์จะเปลี่ยนแปลงไปแต่ OID ยังคงเดิม ถ้าต้องการเปลี่ยนสถานะของออบเจ็กต์ต้องเปลี่ยนค่าของแอททริบิวต์โดยจะต้องส่งเมสเซจไปยังออบเจ็กต์ แล้วเมสเซจที่ส่งไปนี้จะเรียกเมทธอดที่เกี่ยวข้องให้ทำงาน

3.1.5 เมสเซจและเมทธอด (Message and Method)

ตามคุณสมบัติของเอนแคปซูเลชั่นการที่จะกระทำการใดๆ กับออบเจ็กต์ต้องกระทำผ่านเมทธอดเท่านั้น เมทธอดจะถูกใช้เพื่อเรียกดูหรือเปลี่ยนค่าแอททริบิวต์ของออบเจ็กต์ ในการเรียกใช้เมทธอดนั้นจะต้องส่งเมสเซจไปยังออบเจ็กต์ เมสเซจที่ส่งจะต้องระบุออบเจ็กต์ที่จะรับเมสเซจ ชื่อเมทธอด และพารามิเตอร์ที่เกี่ยวข้อง

3.1.6 เอนแคปซูเลชัน (Encapsulation)

ออบเจ็กต์จะเป็นการรวมเอาแอททริบิวต์ และเมทธอดไว้ด้วยกัน การจะเข้าถึงแอททริบิวต์ต้องทำผ่านเมทธอดเท่านั้น ซึ่งจะช่วยปกป้องข้อมูลของออบเจ็กต์

3.1.7 คลาส (Class)

ในระบบเชิงวัตถุจะนำออบเจ็กต์ที่มีคุณสมบัติเหมือนกันเข้าไว้ด้วยกันเป็นคลาส หรืออาจพูดได้อีกอย่างว่า คลาสก็คือ Collection ของออบเจ็กต์ที่มีคุณสมบัติเหมือนกันและใช้แอททริบิวต์และเมทธอดร่วมกัน

3.1.8 การสืบทอดคุณสมบัติ (Inheritance)

การสืบทอดคุณสมบัติเป็นการสร้างคลาสใหม่ (Subclass) โดยสืบทอดคุณสมบัติ (ทั้งแอททริบิวต์และเมทธอด) จากคลาสที่มีอยู่แล้ว (Superclass) นอกจากนี้ยังสามารถเพิ่มรายละเอียดให้ Subclass ได้อีกด้วย การสืบทอดคุณสมบัตินี้มีประโยชน์ในเรื่องการนำกลับมาใช้ใหม่

Single Inheritance เป็นการสืบทอดคุณสมบัติมาจากคลาสเดียว

Multiple Inheritance เป็นการสืบทอดคุณสมบัติจากหลายๆ คลาส

3.1.9 Method Overriding and Polymorphism

Subclass สามารถสร้างเมทธอดที่ชื่อซ้ำกับเมทธอดที่สืบทอดมาจาก Superclass ได้ เพื่อทำให้ Subclass นั้นมีความเฉพาะเจาะจงมากขึ้น เมทธอดที่ Subclass สร้างขึ้นนี้จะถูกเรียกใช้แทน (Override) เมทธอดของ Superclass

พอลิมอร์ฟิซึม (Polymorphism) เป็นการทำให้ออบเจ็กต์ที่ต่างกันสามารถตอบสนองต่อเมสเซจเดียวกันได้ในหลายๆ วิธีการพอลิมอร์ฟิซึมเป็นคุณสมบัติที่สำคัญที่สำคัญของระบบเชิงวัตถุเพราะช่วยให้แต่ละออบเจ็กต์มีความเฉพาะเจาะจงมากขึ้น ในระบบเชิงวัตถุคำว่าพอลิมอร์ฟิซึมจะหมายถึง

1. เมทธอดสามารถใช้ชื่อเดียวกันได้ในหลายๆ คลาส

2. ผู้ใช้ส่งเมสเซจเดียวกันไปยังออบเจ็กต์จากคลาสต่างๆ กันก็ยังคงได้ผลลัพธ์ที่ถูกต้อง

3.1.10 Abstract Data Type

Abstract Data Type (ADT) เป็นคุณสมบัติที่ใช้สร้างชนิดของข้อมูลใหม่ขึ้นมา โดยกำหนดโครงสร้างข้อมูลและโอเปอร์เรชันที่ใช้จัดการข้อมูลขึ้นมาจากข้อมูลพื้นฐาน

จะสังเกตได้ว่า Abstract Data Type และคลาสมีความหมายใกล้เคียงกัน แต่ในระบบเชิงวัตถุสองคำนี้มีความหมายต่างกัน โดย Type จะหมายถึงโครงสร้างข้อมูลและเมทธอดของคลาส และคลาสหมายถึง Collection ของ Object Instance เมื่อกำหนดคลาสใหม่นี้ขึ้นมาก็เป็นการกำหนด Type ใหม่ด้วย Type ที่กำหนดขึ้นจะถูกใช้เป็นต้นแบบในการสร้างออบเจ็กต์ใหม่ ซึ่งจะถูกจัดการโดยคลาสขณะ run-time

ด้วยคุณสมบัติของ ADT และ Inheritance จะสนับสนุนให้มี Complex Object โดย Complex Object ถูกสร้างขึ้นโดยนำออบเจ็กต์อื่นเข้ามาร่วมไว้ด้วยในรูปของ Set ของ Complex Relation

3.2 คุณลักษณะของ Object-Oriented Data Model (OODM)

OODM อย่างน้อยที่สุดควรมีลักษณะดังนี้

1. ต้องสนับสนุนการทำ Complex Object
2. Extensible: ต้องมีความสามารถในการกำหนด Data Type และ Operation ที่เกี่ยวข้องขึ้นมาใหม่ได้
3. ต้องสนับสนุน Encapsulation: รูปแบบของข้อมูลและการจัดการของเมทธอดต้องถูกซ่อนจากภายนอก
4. ต้องมีคุณสมบัติ Inheritance: ออบเจ็กต์ต้องสามารถสืบทอดคุณสมบัติ (ข้อมูลและเมทธอด) จากออบเจ็กต์อื่นได้
5. ต้องสนับสนุน Object Identity (OID)

OODM	**ER Model**
Type	Entity Definition
Object	Entity
Class	Entity Set
Instance Variable	Attribute
N/A	Primary Key
OID	N/A
Method	N/A
Class Hierarchy	ER Diagram (Database Schema)

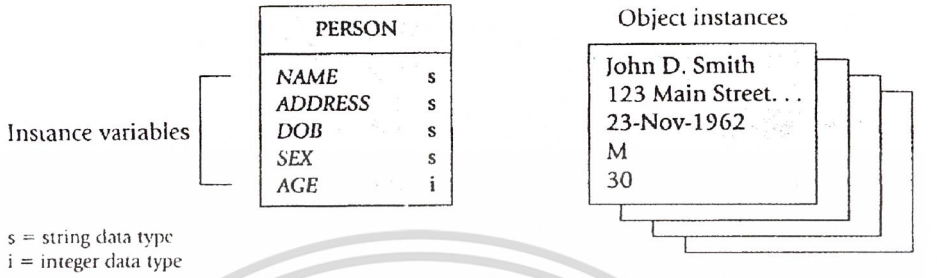
ตารางที่ 3-1 เปรียบเทียบ Object-oriented และ ER-Model Component

ที่มา : Peter Rob "Database systems : design, implementation, and management"

3.3 The Graphics Representation of Objects: Object Diagram

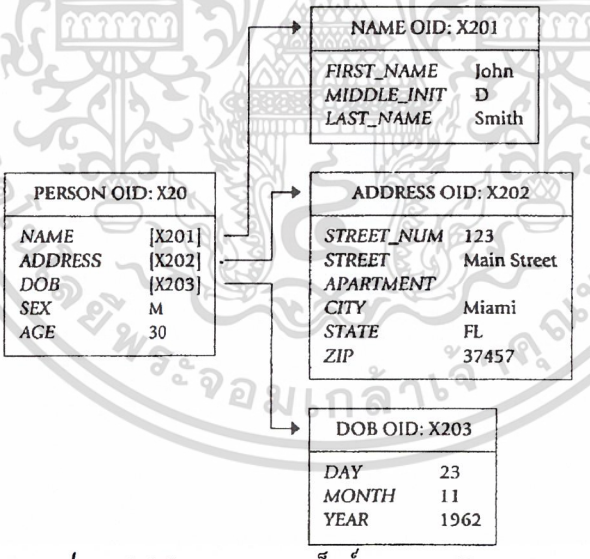
ใน Object Diagram จะแทนออบเจ็กต์ด้วยรูปสี่เหลี่ยมมุมฉาก มี Instance Variable อยู่ภายใน

รูปที่ 3.1 แสดงคลาส Person มี Instance Variable คือ NAME, ADDRESS, DOB (Date Of Birth) และ SEX ซึ่งมีชนิดของข้อมูลเป็นสตริง และ AGE มีชนิดข้อมูลเป็นเลขจำนวนเต็ม (Integer)

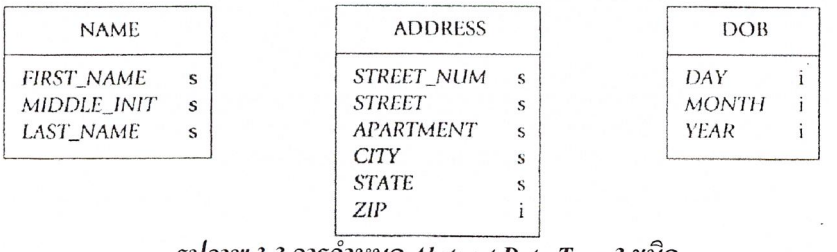


รูปภาพ 3-1 คลาส Person

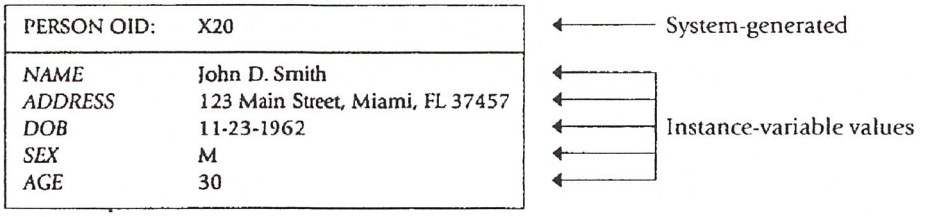
พิจารณาสถานะของออบเจ็กต์ Person ได้ในรูปที่ 3.2 ในระบบเชิงวัตถุสามารถที่จะสร้าง ADT ใหม่ได้จากข้อมูลชนิดพื้นฐานที่มีมาให้ เช่น ถ้าให้ NAME, ADDRESS และ DOB เปลี่ยนเป็น Composite Attribute ซึ่งสามารถทำได้โดยจัดการผ่านคลาสหรือ ADT ดังรูปที่ 3.3 เป็นผลให้คลาส Person มี Attribute ที่ชี้ไปยังออบเจ็กต์ของคลาสหรือ ADT อื่น ดังรูปที่ 3.4 แทนที่จะเป็นข้อมูลชนิดพื้นฐาน



รูปภาพ 3-2 State ของออบเจ็กต์จากคลาส Person

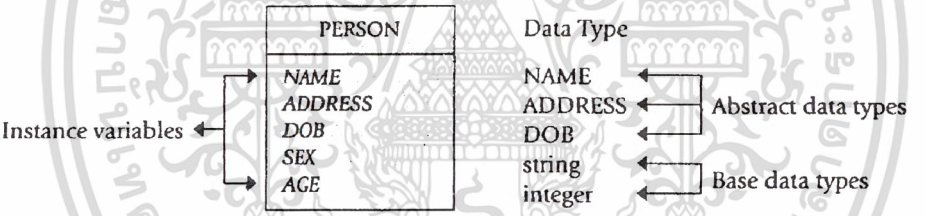


รูปภาพ 3-3 การกำหนด Abstract Data Type 3 ชนิด



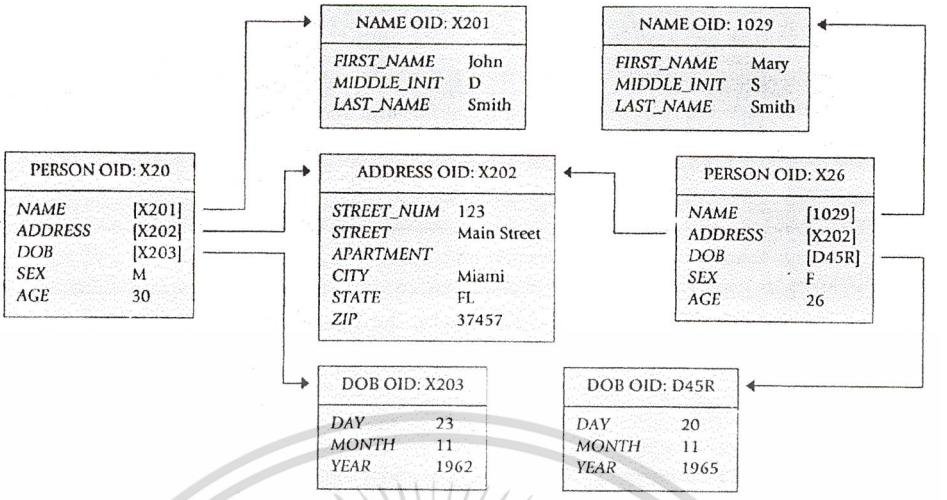
รูปภาพ 3-4ออบเจ็กต์ของคลาส Person กับ ADT

คำว่า Object Space หรือ Object Schema ก็คือ Database Schema ณ เวลาที่กำหนดใช้ Object Space เพื่อแสดงให้เห็นสถานะต่างๆ ของออบเจ็กต์ ณ เวลาหนึ่ง Object Space ของคลาส Person เป็นดังรูปที่ 3.5 สังเกตได้ว่าใช้ OID ในการอ้างอิงไปยังออบเจ็กต์อื่น โดยแอททริบิวต์ NAME, ADDRESS และ DOB ขณะนี้มีค่าเป็น OID ของออบเจ็กต์จากคลาสหรือ ADT ที่เกี่ยวข้อง แทนที่จะเป็นค่าของข้อมูลชนิดพื้นฐาน การใช้ OID ในการอ้างอิงถึงออบเจ็กต์อื่นทำให้สามารถหลีกเลี่ยงปัญหาความขัดแย้งของข้อมูล (Data Inconsistency) ได้ เพราะ OID ไม่ได้ขึ้นอยู่กับสถานะของออบเจ็กต์ ซึ่งปัญหานี้จะปรากฏอยู่ในระบบฐานข้อมูลเชิงสัมพันธ์ เมื่อ Primary Key ถูกเปลี่ยนค่าโดยผู้ใช้



รูปภาพ 3-5 สถานะของออบเจ็กต์ที่เป็น Instance ของคลาส Person ที่เป็น ADT

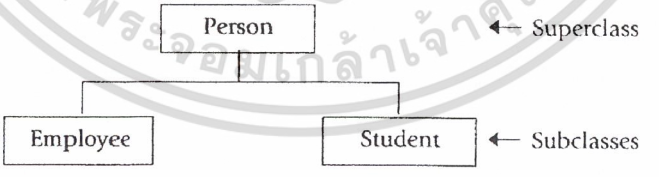
ตัวอย่างเช่น ถ้ามีคนสองคนอาศัยอยู่ในบ้านเดียวกันควรอ้างอิงไปยัง Object Instance ของคลาส ADDRESS ตัวเดียวกัน แทนที่จะเป็น 2 Object Instances ที่มีสถานะเดียวกัน สภาวะเช่นนี้ถูกเรียกว่า Referential Object Sharing ซึ่งการเปลี่ยนแปลงออบเจ็กต์จากคลาส Address ที่ใช้ร่วมกันอยู่ จะส่งผลกระทบต่อออบเจ็กต์จากคลาส Person ทั้งสองออบเจ็กต์ ดังรูปที่ 3.6 ซึ่งมี 4 คลาสประกอบด้วย คลาส Person (2 Instance) คลาส Name (2 Instance) คลาส Address และคลาส DOB (2 Instance)



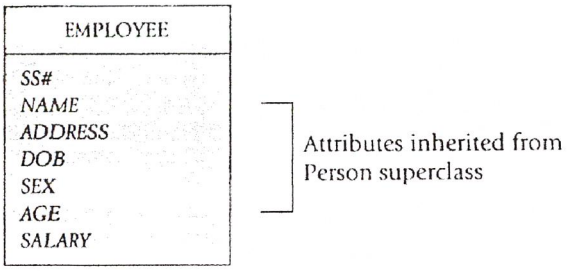
รูปภาพ 3-6Referential Object Sharing

3.4 ความสัมพันธ์ระหว่างคลาสและ Subclass (Class-Subclass Relationship)

ในการสืบทอดคุณสมบัติของคลาสมาจาก Superclass จะเรียกความสัมพันธ์แบบนี้ว่า 'is-a relationship' เช่น an employee is a person, a student is a person ตามรูปที่ 3.7 ซึ่งออบเจ็กต์ Employee ในรูปที่ 3.8 ประกอบด้วยแอททริบิวต์ต่างๆ ดังนี้ หมายเลขประกันสังคม (SS#) เก็บเป็นสตริง เงินเดือน (SALARY) เก็บเป็นตัวเลข และมี NAME, ADDRESS, DOB และ AGE ที่สืบทอดมาจาก Supercalss ความสัมพันธ์ระหว่างคลาสและ Subclass นี้จะเป็นแบบ 1:1 ถ้าเป็น Single Inheritance



รูปภาพ 3-7 Class Hierarchy

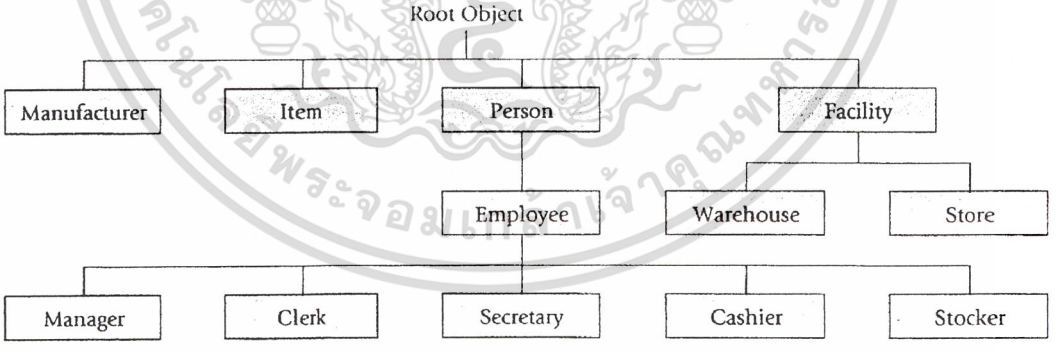


รูปภาพ 3-8 แสดงออบเจ็กต์ของคลาส Employee

3.5 ความสัมพันธ์ระหว่างคลาส (Interclass Relationships: Attribute-Class Links)

นอกจากความสัมพันธ์ระหว่างคลาสและ Subclass แล้ว OODM ยังสนับสนุนความสัมพันธ์ระหว่างคลาส (Interclass Relationships) ด้วยความสัมพันธ์ระหว่างคลาสจะเกิดขึ้นเมื่อชนิดข้อมูลของแอททริบิวต์มีการอ้างอิงไปยังชนิดข้อมูลของแอททริบิวต์อื่น ความสัมพันธ์ระหว่างคลาสมีความแตกต่างจากความสัมพันธ์ระหว่างคลาส และ Subclass ในหัวข้อก่อนหน้านี้ ลองพิจารณา Class Hierarchy ของ EDLP (Every Day Low Price) Retail Corporation ดังรูปที่ 3.9

ตามรูปที่ 3.9 ทุกๆ คลาสสืบทอดคุณสมบัติมาจากคลาส Root Object จากรูป Class Hierarchy ประกอบไปด้วยคลาส Manufacturer, Item, Person และ Facility คลาส Facility ประกอบไปด้วย Subclass Warehouse และ Store คลาส Person ประกอบไปด้วย Subclass Employee ที่มี Subclass Manager, Clerk, Secretary, Cashier และ Stoker จากตัวอย่างนี้จะแสดงให้เห็นถึงความสัมพันธ์แบบ 1:M และ M:N และจะ Implement ความสัมพันธ์แบบ M:N



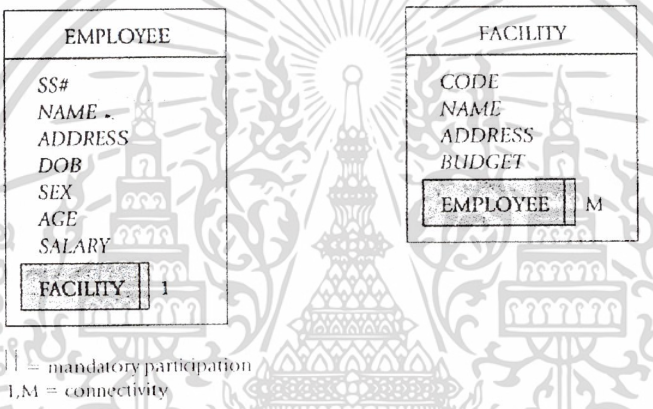
รูปภาพ 3-9 Class Hierarchy ของ EDLP Retail Corp.

3.5.1 Representing 1:M Relationship

จากตัวอย่างในรูปที่ 3.9 ระหว่างคลาส Employee และ Facility มีความสัมพันธ์แบบ one to many หรือ 1:M เกิดขึ้น โดย Employee ทุกคนจะทำงานเพียงหนึ่ง Facility ในขณะเดียวกันหนึ่ง Facility จะมีหลายๆ Employee ทำงานอยู่

ตามรูปที่ 3.10 แสดงความสัมพันธ์ระหว่าง Employee และ Facility โดยออบเจ็กต์ Facility จะถูกรวมไว้ในออบเจ็กต์ Employee และออบเจ็กต์ Employee ก็ถูกรวมไว้ในออบเจ็กต์ Facility โดยมีรายละเอียดดังนี้

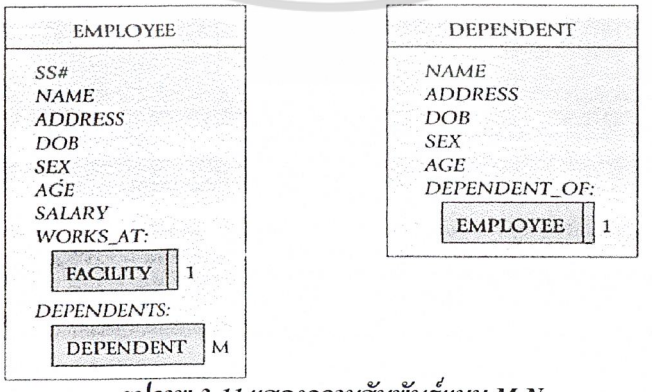
1) คลาสที่เกี่ยวข้องต้องถูกล้อมรอบโดยสี่เหลี่ยมเพื่อที่จะได้เห็นเด่นชัด
2) เส้นคู่ทางด้านขวาของรูปสี่เหลี่ยมแสดงว่าความสัมพันธ์นั้นเป็นแบบ Mandatory (ต้องมีค่าเสมอ)
3) Connectivity จะระบุโดยเขียนกำกับไว้ที่ข้างรูปสี่เหลี่ยมนั้น ในกรณีที่เขียน 1 ถัดจาก Facility ในออบเจ็กต์ Employee หมายถึง Employee ทุกคนจะทำงานเพียงหนึ่ง Facility และ M ที่อยู่ข้าง Employee ใน ออบเจ็กต์ Facility หมายถึงหนึ่ง Facility จะประกอบไปด้วยหลายๆ Employee



รูปภาพ 3-10 แสดงความสัมพันธ์แบบ 1:M

3.5.2 Representing M:N Relationship :

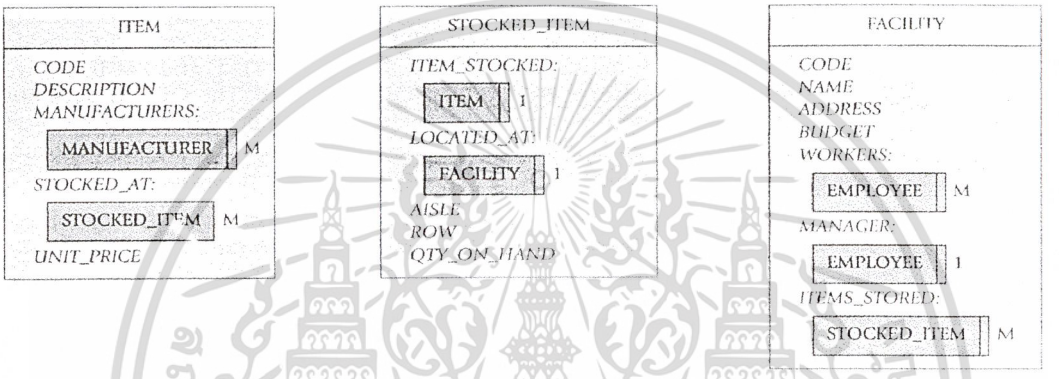
ตามตัวอย่าง EDLP Retail Corp. ความสัมพันธ์แบบ many to many (M:N) เกิดขึ้นระหว่างคลาส Manufacturer และ Item ตามรูปที่ 3.11 ซึ่งแสดง Conceptual view โดยที่แต่ละ Item สามารถผลิตได้จากหลายๆ Manufacturer และแต่ละ Manufacturer ก็สามารถผลิตได้หลายๆ Item



รูปภาพ 3-11 แสดงความสัมพันธ์แบบ M:N

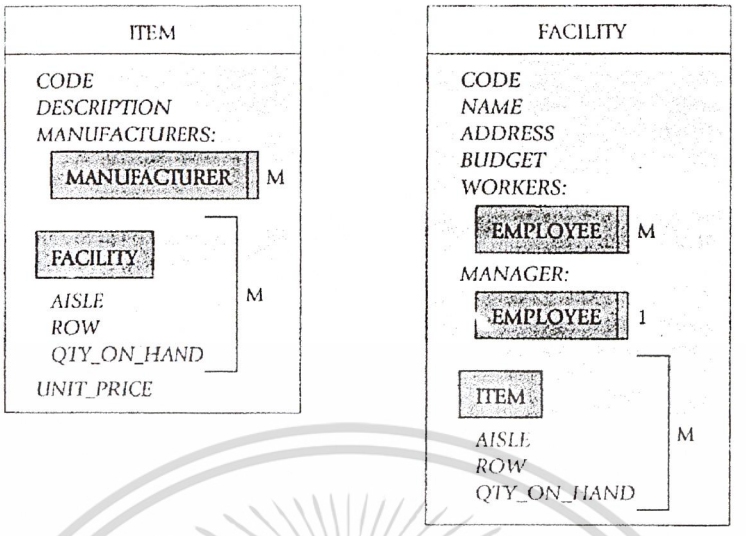
3.5.3 Representing M:N Relationships with an Intersection Class

ถ้าเพิ่มเงื่อนไขให้กับความสัมพันธ์แบบ M:N เพื่อให้สามารถติดตาม (Keep Track) ข้อมูลที่จะเพิ่มเข้าไปได้ เช่น ความสัมพันธ์ระหว่างคลาส Item และ Facility โดยที่แต่ละ Facility จะมีหลาย Item เก็บอยู่ และแต่ละ Item ก็จะเก็บได้หลายๆ Facility ถ้าต้องการรู้เพิ่มว่าแต่ละ Item ในแต่ละ Facility มีปริมาณเท่าไรและเก็บที่ตำแหน่ง (aisle และ row) ไหน โดยแสดงตัวอย่างที่รูป 3.12 วงเล็บใหญ่ ( ]) ที่ใช้ในรูปหมายถึงรวมแอททริบิวต์ทั้งหมดในวงเล็บถือเป็นหนึ่ง Logical Unit เพราะฉะนั้นแต่ละ Item Instance จะประกอบไปด้วยหลายๆ Facility ซึ่งแต่ละ Facility จะประกอบไปด้วยแอททริบิวต์ 3 ตัว ได้แก่ AISLE, ROW และ QTY_ON_HAND ซึ่งในกรณีกลับกันก็เป็นจริงสำหรับทุกๆ Facility



รูปภาพ 3-12 ความสัมพันธ์แบบ M:N และ attribute ที่เกี่ยวข้อง

ถ้ามองในมุมมองของฐานข้อมูลเชิงสัมพันธ์ สิ่งที่ควรจะเพิ่มขึ้นสำหรับความสัมพันธ์แบบ M:N นี้ ก็คือ Intersection (bridge) class เพื่อที่จะเชื่อมทั้ง Facility กับ Item และแอททริบิวต์ที่เกี่ยวข้อง โดยสร้างคลาส Stocked_Item ซึ่งประกอบไปด้วย Facility และ Item Instance และแอททริบิวต์ AISLE, ROW และ QTY_ON_HAND ตามรูปที่ 3.13

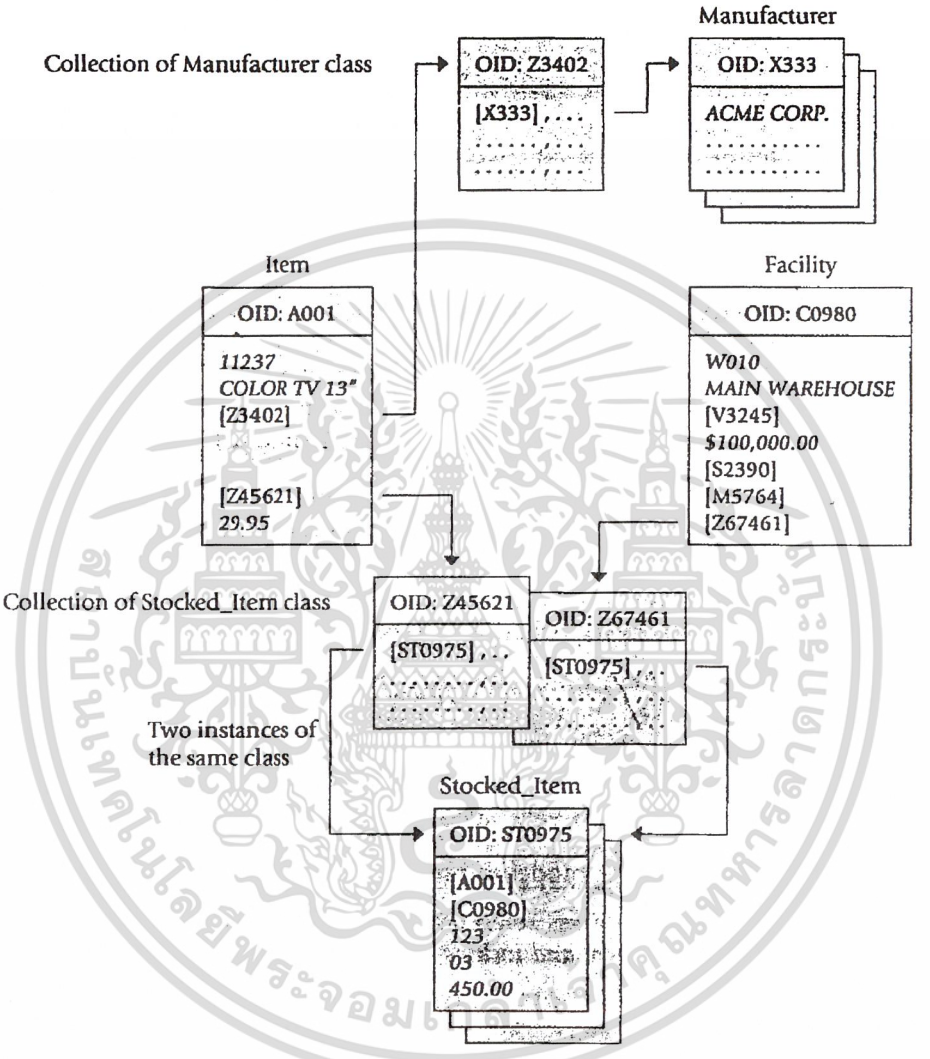


รูปภาพ 3-13 ความสัมพันธ์แบบ M:N กับ Intersection class

รูปที่ 3.14 แสดง Object Space ตามแนวคิดข้างต้น อย่างไรก็ดีการออกแบบเช่นนี้ก็เป็นการออกแบบที่ต้องระมัดระวังเป็นพิเศษ โดยมีจุดที่ต้องสนใจคือ

- Stocked_Item Object Instance จะมีการอ้างอิงไปยัง Instance ของคลาสอื่นที่เกี่ยวข้อง ซึ่งการทำ Intersection Class เช่นนี้ก็เพื่อต้องการติดตามข้อมูลที่เพิ่มเข้ามาเท่านั้น
- Item Object Instance ใน Object Schema นี้ประกอบไปด้วย Collection of Stocked_Item Object Instance ซึ่งแต่ละอันก็ประกอบไปด้วย 1 Facility Object Instance ความสัมพันธ์ในทางกลับกันก็ยังคงเป็นจริง
- ใน Object Space การอ้างอิงระหว่างคลาสจะใช้ OID ของออบเจ็กต์ที่ถูกอ้าง
- ค่าที่อยู่ในวงเล็บ [ ] จะแทน OID ของ Object Instance ของบางคลาส เช่น ค่า Z3402 และ Z45621 เป็นต้น เป็น OID ของออบเจ็กต์ที่ถูกอ้างถึง ซึ่งก็คือ Collection ของ Manufacturer และ Stocked_Item ตามลำดับ

ใน Relational Model ตาราง ITEM จะไม่มีข้อมูลที่เกี่ยวข้องกับ MANUFACTURER หรือ STOCKED_ITEM เลย ซึ่งถ้าต้องการใช้ข้อมูลทีเกี่ยวพันกันของทั้ง 3 ตาราง จะต้อง Join ตารางเข้าด้วยกัน แต่ OODM ไม่ต้องการการ Join เพราะออบเจ็กต์ Item จะมีการอ้างอิงไปยังออบเจ็กต์อื่นที่เกี่ยวข้องอยู่แล้ว การอ้างอิงนี้จะนำออบเจ็กต์ Item เข้าสู่ Object Space โดยอัตโนมัติเมื่อถูกเรียกใช้



รูปภาพ 3-14 Object Space

3.6 ระบบจัดการฐานข้อมูลเชิงวัตถุ

OODBMS จะทำหน้าที่คอยจัดการกับฐานข้อมูลเชิงวัตถุ โดยจะใช้คุณสมบัติบางส่วนตามแนวคิดเชิงวัตถุ (OO Concept) และ OODM ที่เป็นเพียงบางส่วน ทั้งนี้เพราะยังไม่มีมาตรฐานที่ใช้กำหนดถึงคุณสมบัติของ OODBMS ที่ต้องสามารถทำได้ ทำให้ OODBMS ที่มีอยู่มีคุณสมบัติแตกต่างกัน อย่างไรก็ดี OODBMS ก็มีคุณสมบัติที่จำเป็นต้องมีอยู่ 14 ข้อด้วยกัน ซึ่งสามารถแบ่งได้เป็น 2 ส่วน ดังนี้

3.6.1 ส่วนที่ทำให้เป็น OO System

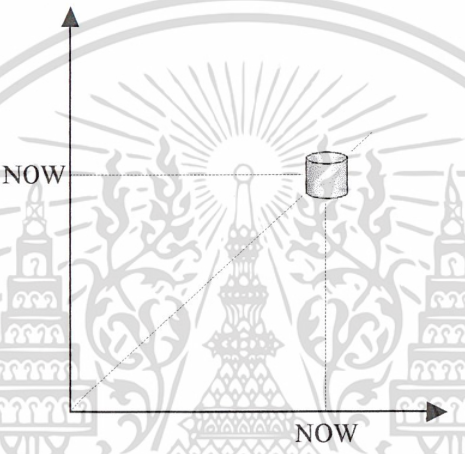
1. ระบบต้องสนับสนุน Complex Object
2. สนับสนุน Object Identity (OID)
3. ออบเจ็กต์ต้องถูก Encapsulation
4. ระบบต้องสนับสนุน Type และ Class
5. ระบบต้องสนับสนุนการสืบทอดคุณสมบัติ (Inheritance)
6. ระบบต้องมีคุณสมบัติ Overriding, Overloading และ late binding
7. ระบบต้องเป็น Computational Complete
8. ระบบต้อง Extensible
9. ระบบต้องจัดการ persistence ให้

3.6.2 ส่วนที่ทำให้เป็น DBMS

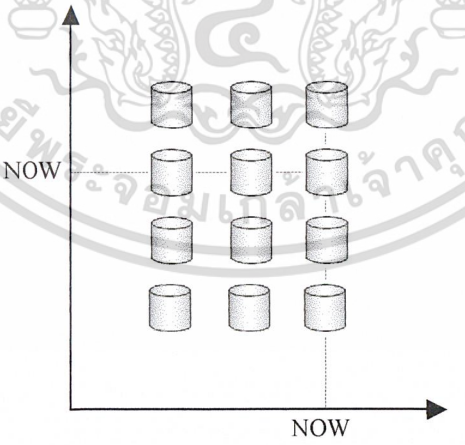
10. ระบบต้องมีการจัดเก็บข้อมูล (Secondary storage management)
11. ต้องสามารถจัดการกับฐานข้อมูลที่มีขนาดใหญ่ได้
12. สนับสนุนการทำงานแบบ Concurrency
13. ต้องสามารถกู้ระบบ (Recovery) จากการทำงานที่ผิดพลาดของ Hardware และ Software ได้
14. ต้องสามารถทำการค้นหาข้อมูลได้ง่าย

3.7 ฐานข้อมูลเชิงเวลา (Temporal Databases)

ระบบฐานข้อมูลที่ใช้งานกันอยู่ทั่วไปมักจะเป็นระบบฐานข้อมูลเชิงสัมพันธ์ ซึ่งมีการจัดเก็บ ข้อมูลเพียงสถานะเดียว เมื่อมีการแก้ไขข้อมูลข้อมูลเก่าจะถูกลบออกจากฐานข้อมูลและแทนที่ด้วย ข้อมูลใหม่ แม้ว่าระบบฐานข้อมูลแบบเดิมนี้จะรองรับโปรแกรมประยุกต์บางโปรแกรมได้ดี แต่ก็ยังไม่เพียงพอที่จะใช้งานกับโปรแกรมประยุกต์ที่ต้องการเรียกค้นข้อมูลหลาย ๆ สถานะ อาทิ โปรแกรม ทางการแพทย์ โปรแกรมทางด้านธุรกิจ ดังนั้นเราจึงต้องการระบบฐานข้อมูลที่สามารเรียกค้นข้อมูล ที่แปรผันตามเวลาได้ ซึ่งก็คือระบบฐานข้อมูลที่เก็บข้อมูลหลาย ๆ สถานะหรือเรียกว่า "ระบบฐานข้อมูลเชิงเวลา" (Temporal Database) นั่นเอง



รูปภาพ 3-15การจัดเก็บข้อมูลของระบบฐานข้อมูลปกติ



รูปภาพ 3-16 การจัดเก็บข้อมูลของระบบฐานข้อมูลเชิงเวลา

3.7.1 หลักการพื้นฐานของฐานข้อมูลเชิงเวลา

ฐานข้อมูลจะเก็บข้อมูลที่เป็นจริงบางส่วน (เฉพาะที่จำเป็น) เอาไว้ ซึ่งจะเรียกข้อมูลที่เก็บนี้ว่า mini-world โดยอาจเก็บไว้ในรูปแบบต่างๆ กัน เรียกว่า database entities และ คำว่า fact จะหมายความถึง

ข้อมูลใด ๆ ที่เราสามารถหาค่าได้ว่าเป็นจริงหรือไม่ โดยทั่วไปเวลา (Time) จะสัมพันธ์กับ database entities เวลาที่เก็บใน Temporal Database มี 3 แบบคือ

1) **Valid Time** คือ เวลาที่ข้อมูล (fact) นั้นเป็นจริง ซึ่งจะแบ่งแยกย่อยออกเป็น
   - ValidTimeStart คือ เวลาเริ่มต้นที่ข้อมูลนั้นเป็นจริง
   - ValidTimeStop คือ เวลาสุดท้ายที่ข้อมูลนั้นเป็นจริง
2) **Transaction Time** คือ เวลาที่ข้อมูลถูกเก็บใน Database โดยสามารถแบ่งออกเป็น
   - Transaction Time Start คือ เวลาที่เริ่มจัดเก็บข้อมูลลงในฐานข้อมูล
   - Transaction Time End คือ เวลาที่แก้ไขหรือลบข้อมูล
3) **User Defined Time** คือ เวลาที่ผู้ใช้งานสร้างขึ้นเพื่อใช้งาน ซึ่งผู้ใช้งานต้องจัดการและเรียกค้นเอง ระบบฐานข้อมูลจะไม่จัดการให้อย่างอัตโนมัติ หรือกล่าวอีกนัยหนึ่งว่า เวลาชนิดนี้จะเป็นเวลาที่เป็นจริงเสมอ เช่น วันเกิด วันเข้าทำงาน และวันเกษียร เป็นต้น

เวลาที่สำคัญที่สุด คือ valid time ของ fact (อาจมีความหมายได้ทั้งอดีต ปัจจุบัน และอนาคต) ซึ่งจะเก็บช่วงเวลาที่ fact นั้นเป็นจริงใน mini-world และจะเก็บสถานะที่เปลี่ยนไปตามเวลา (Time-varying States) ในทางทฤษฎีทุก ๆ fact จะมี valid time แต่ในบางครั้ง valid time ก็ไม่จำเป็นจะต้องเก็บลงบนฐานข้อมูล อาจเป็นเพราะ ไม่ทราบ valid time หรือไม่มีความจำเป็นต้องใช้ในงานนั้น ๆ

transaction time ของ database fact ต่างกับ valid time เพราะ transaction time อาจสัมพันธ์กันกับ database entities ใด ๆ ก็ได้ไม่เฉพาะ fact เช่น transaction time อาจสัมพันธ์กับออบเจ็กต์ (Object) หรือ value ใด ๆ ที่ไม่เป็น fact เพราะไม่สามารถเป็นจริงหรือเท็จได้โดยอิสระ (in isolation) ดังนั้นทุก ๆ database entity จะมี transaction time แต่อาจเลือกที่จะเก็บ transaction time ลงในฐานข้อมูลหรือไม่ก็ได้

ตัวอย่างของ Valid Time และ Transaction Time

ถ้านายสุวิทย์เป็นนักศึกษาตั้งแต่ วันที่ 1 มิถุนายน ค.ศ.1994 ถึงวันที่ 30 พฤษภาคม ค.ศ.1998 แต่ข้อมูลนี้ถูกป้อนเข้าฐานข้อมูลใน วันที่ 20 มิถุนายน ค.ศ.1994 โดยที่ข้อมูลนี้ถูกลบออกจากฐานข้อมูลใน วันที่ 25 มิถุนายน ค.ศ.1998 เราสามารถแจกแจงเวลาได้ ดังนี้

- ValidTimeStart คือ วันที่ 1 มิถุนายน ค.ศ.1994
- ValidTimeStop คือ วันที่ 30 พฤษภาคม ค.ศ.1998
- Transaction Time Start คือ วันที่ 20 มิถุนายน ค.ศ.1994
- Transaction Time End คือ วันที่ 25 มิถุนายน ค.ศ.1998

ช่วงเวลา 3 ช่วงนี้ไม่จำเป็นต้องเป็นช่วงเวลาเดียวกัน ตัวอย่างเช่น สมมติว่ามี Temporal Database ที่เก็บข้อมูลของศตวรรษที่18 Valid Time ของข้อมูล คือ ช่วงเวลาใด ๆ ระหว่างปี 1700 และ 1799 ส่วน Transaction Time จะเป็นเวลาที่เราเก็บข้อมูลเข้าไปในฐานข้อมูล ถ้า ValidTimeStart เป็นอดีตแต่ข้อมูลที่ใส่เข้าไปยังไม่ทราบ ValidTimeEnd เราจึงกำหนดให้ ValidTimeEnd เป็น NOW และให้ ValidTimeEnd เป็น NULL เมื่อ ValidTimeStart เป็นอนาคต

$$T_E := NOW, \text{ if } T_S \leq NOW$$

$$T_E := NOW, \text{ if } T_S < NOW$$

จะเห็นได้ว่าช่วงเวลา ValidTime และช่วงเวลา Transaction Time ไม่จำเป็นต้องเป็นช่วงเวลาเดียวกันสำหรับข้อมูลตัวเดียวกัน ตัวอย่างเช่น ฐานข้อมูลเก็บชื่อของพนักงานคนกนึ่งไว้ตั้งแต่ปี 2540 หากพนักงานคนนี้มีการเปลี่ยนแปลงชื่อในปี 2543 แต่มาตรวจพบว่าชื่อมีการเปลี่ยนแปลงในปี 2544 จะทำให้ ValidTime และ Transaction Time เป็น 2543 และ 2544 ตามลำดับ ซึ่งมีค่าไม่เท่ากัน

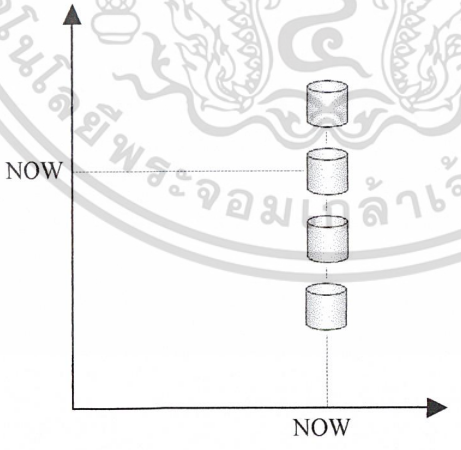
ทั้ง valid time และ transaction time เป็นแนวคิดหลักที่สำคัญของฐานข้อมูลทุกชนิด และมีความจำเป็นที่จะต้องใช้เวลาทั้งสองชนิดนี้ในหลาย ๆ application และยิ่งไปกว่านั้น transaction time นั้นมีลักษณะพิเศษที่อาจให้ระบบจัดการฐานข้อมูล (Database Management System ; DBMS) จัดการอย่างอัตโนมัติ โดยเฉพาะ transaction time ของ fact ที่เก็บในฐานข้อมูลที่มีทิศทางไปข้างหน้าอย่างเดียว (marches monotonically forward) และผูกพันอยู่กับเวลาเพียง 2 ชนิด คือ เวลาที่สร้างฐานข้อมูลขึ้นมาและเวลาปัจจุบัน (current time)

3.7.2 ประเภทของฐานข้อมูลเชิงเวลา

ช่วงเวลา valid time และ transaction time จะทำให้เกิดฐานข้อมูลในหลายประเภท ดังนี้

3.7.2.1 ฐานข้อมูลวาลิดไทม์ (Valid-Time Database)

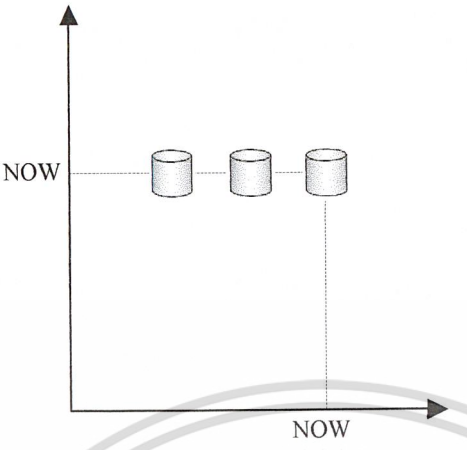
ฐานข้อมูลประวัติ (Historical Database) จะเก็บเฉพาะวาลิดไทม์ ซึ่งจะเก็บข้อมูลในอดีตกับปัจจุบัน



รูปภาพ 3-17ฐานข้อมูลประวัติ

3.7.2.2 ฐานข้อมูลทรานส์แอกชันไทม์ (Transaction-Time Database)

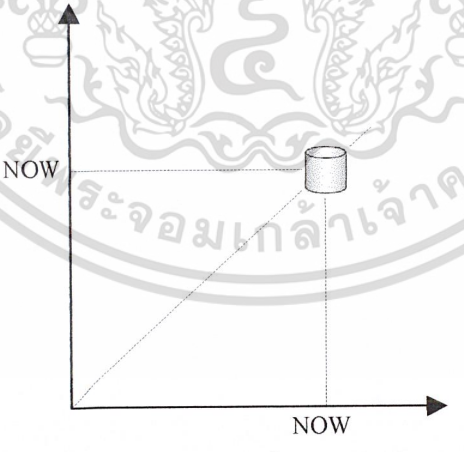
ฐานข้อมูลย้อนกลับ (Rollback Database) ซึ่งจะเก็บเฉพาะมูลทรานส์แอกชันไทม์



รูปภาพ 3-18ฐานข้อมูลย้อนกลับ

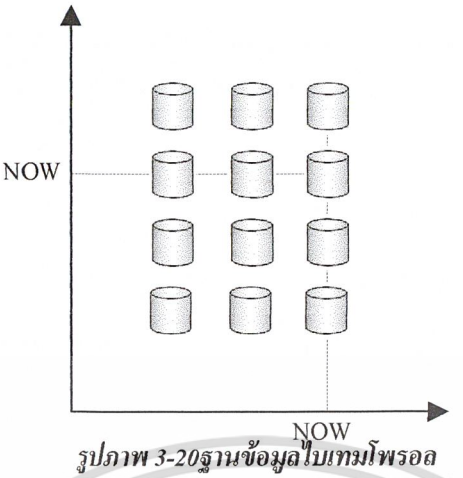
3.7.2.3 ฐานข้อมูลไบเทมโพรอล (Bitemporal Database)

เก็บทั้งวาลิดไทม์ และทรานส์แอกชันไทม์ จะทำให้เก็บข้อมูลในอดีต อนาคต และปัจจุบันได้ในระบบจัดการฐานข้อมูลเชิงพาณิชย์จะเก็บเฉพาะข้อมูลที่เป็นจริงในปัจจุบันเท่านั้น ฐานข้อมูลชนิดนี้เรียกว่า ฐานข้อมูลสแนปช็อท (Snapshot Database) โดยฐานข้อมูลสแนปช็อทในแบบของวาลิดไทม์ และทรานส์แอกชันไทม์



รูปภาพ 3-19ฐานข้อมูลสแน็ปช็อท

ฐานข้อมูลแบบไบเทมโพรอลอาจจะเป็นได้ทั้งตารางสแนปช็อท โดยเก็บตารางวาลิดไทม์ คือ เก็บว่าข้อมูลเป็นจริงเมื่อใด หรือเก็บตารางทรานส์แอกชันไทม์ คือเก็บว่าข้อมูลถูกบันทึกลงในฐานข้อมูลเมื่อใด หรืออาจจะเป็นตารางไบเทมโพรอลที่เก็บทั้งวาลิดไทม์และทรานส์แอกชันไทม์ก็ได้



รูปภาพ 3-20ฐานข้อมูลไบเทมโพรอล

3.8 ฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงสัมพันธ์

ฐานข้อมูลเชิงสัมพันธ์ โดยปกติจะเก็บเฉพาะข้อมูลที่เป็นจริงในปัจจุบันเท่านั้น และมีคีย์หลัก (Primary Key) ของตาราง ดังเช่นตารางที่ 3.21

ID	Name	Surname	SALARY
1	Jame	SMITH	20000
2	ANNA	Counicova	25000

รูปภาพ 3-21การจัดเก็บฐานข้อมูลเชิงสัมพันธ์ทั่วไป

หากมีการแก้ไขค่าของแอตทริบิวต์ใดแอตทริบิวต์หนึ่ง จำเป็นต้องมีการอัพเดท (Update) ทับข้อมูลเดิม ถ้ายังคงต้องการใช้ ID ตัวเดิมในการอ้างถึงอยู่

ID	Name	Surname	SALARY
~~1~~	~~Jame~~	~~SMITH~~	~~20000~~
2	ANNA	Counicova	25000
1	John	Smith	20000

รูปภาพ 3-22การอัพเดทข้อมูลในฐานข้อมูลเชิงสัมพันธ์ทั่วไป

จากตารางที่ 3.22 จะเป็นการอัพเดทแอตทริบิวต์ Name จาก JAME เปลี่ยนชื่อเป็น JOHN ทำให้ต้องทำการอัพเดททับ โรว์เดิม ซึ่งทำให้ไม่สามารถทราบได้ว่า JOHN เคยมีชื่อเป็น JAME มาก่อน และไม่สามารถทราบได้ว่าแอตทริบิวต์ใดเปลี่ยน

หากต้องการให้ตารางฐานข้อมูลเชิงสัมพันธ์จัดเก็บฐานข้อมูลเชิงเวลาได้ จำเป็นต้องเพิ่ม 2 คอลัมน์ คือ FROM_DATE และ TO_DATE ซึ่งเป็นตัวบอกว่าข้อมูลมีการเปลี่ยนแปลงเมื่อใด และต้องคอมไบน์คีย์หลัก ให้เป็น ID,FORM_DATE และ TO_DATE ดังตารางที่ 3.23

ID	NAME	SURNAME	SALARY	FROM_DATE	TO_DATE
1	JAME	SMITH	20000	1/1/1990	1/1/2001
2	ANNA	CUNICOVA	25000	1/1/1990	31/12/9999
1	JOHN	SMITH	20000	1/1/2001	31/12/9999

รูปภาพ 3-23การจัดเก็บฐานข้อมูลเชิงเวลาโดยใช้ฐานข้อมูลเชิงสัมพันธ์

การจัดเก็บข้อมูลดังตารางที่ 3.23 จะทำให้ฐานข้อมูลเชิงสัมพันธ์สามารถจัดเก็บข้อมูลเชิงเวลาได้ ซึ่งจากตาราง JAME จะมีการเปลี่ยนชื่อ JOHN เมื่อวันที่ 1/1/2001 ซึ่งจะทำให้ทราบว่าชื่อ JAME และ JOHN มีค่าช่วงใดโดยดูจากคอลัมน์ FROM_DATE และ TO_DATE และสามารถที่จะอ้างได้ว่า JAME และ JOHN เป็นคนคนเดียวกัน

อย่างไรก็ตามการจัดเก็บฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงสัมพันธ์จะเกิดปัญหา ดังนี้

1) การเอาช่วงเวลามาเกาะกับทั้งทัพเปิล ทำให้ไม่ทราบว่าแอตทริบิวต์ใดที่มีการเปลี่ยนแปลงค่าตามเวลา
2) การอ้างอิงคีย์หลัก (Primary Key) และคีย์อ้างอิง (Foreign Key) ทำได้ยุ่งยาก จากตารางที่ 4.3 จะเห็นได้ว่าคีย์หลักเพิ่มเป็น 3 แอตทริบิวต์ ซึ่งถ้านำไปเป็นคีย์อ้างอิงกับตารางอื่นจะเกิดความยุ่งยาก

ID	Name	Surname	SALARY
~~1~~	~~Jame~~	~~SMITH~~	~~20000~~
2	ANNA	Counicova	25000
1	John	Smith	20000

รูปภาพ 3-22การอัพเดทข้อมูลในฐานข้อมูลเชิงสัมพันธ์ทั่วไป

จากตารางที่ 3.22 จะเป็นการอัพเดทแอตทริบิวต์ Name จาก JAME เปลี่ยนชื่อเป็น JOHN ทำให้ต้องทำการอัพเดททับ โรว์เดิม ซึ่งทำให้ไม่สามารถทราบได้ว่า JOHN เคยมีชื่อเป็น JAME มาก่อน และไม่สามารถทราบได้ว่าแอตทริบิวต์ใดเปลี่ยน

หากต้องการให้ตารางฐานข้อมูลเชิงสัมพันธ์จัดเก็บฐานข้อมูลเชิงเวลาได้ จำเป็นต้องเพิ่ม 2 คอลัมน์ คือ FROM_DATE และ TO_DATE ซึ่งเป็นตัวบอกว่าข้อมูลมีการเปลี่ยนแปลงเมื่อใด และต้องคอมไบน์คีย์หลัก ให้เป็น ID,FORM_DATE และ TO_DATE ดังตารางที่ 3.23

ID	NAME	SURNAME	SALARY	FROM_DATE	TO_DATE
1	JAME	SMITH	20000	1/1/1990	1/1/2001
2	ANNA	CUNICOVA	25000	1/1/1990	31/12/9999
1	JOHN	SMITH	20000	1/1/2001	31/12/9999

รูปภาพ 3-23การจัดเก็บฐานข้อมูลเชิงเวลาโดยใช้ฐานข้อมูลเชิงสัมพันธ์

การจัดเก็บข้อมูลดังตารางที่ 3.23 จะทำให้ฐานข้อมูลเชิงสัมพันธ์สามารถจัดเก็บข้อมูลเชิงเวลาได้ ซึ่งจากตาราง JAME จะมีการเปลี่ยนชื่อ JOHN เมื่อวันที่ 1/1/2001 ซึ่งจะทำให้ทราบว่าชื่อ JAME และ JOHN มีค่าช่วงใดโดยดูจากคอลัมน์ FROM_DATE และ TO_DATE และสามารถที่จะอ้างได้ว่า JAME และ JOHN เป็นคนคนเดียวกัน

อย่างไรก็ตามการจัดเก็บฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงสัมพันธ์จะเกิดปัญหา ดังนี้

1) การเอาช่วงเวลามาเกาะกับทั้งทัพเปิล ทำให้ไม่ทราบว่าแอตทริบิวต์ใดที่มีการเปลี่ยนแปลงค่าตามเวลา
2) การอ้างอิงคีย์หลัก (Primary Key) และคีย์อ้างอิง (Foreign Key) ทำได้ยุ่งยาก จากตารางที่ 4.3 จะเห็นได้ว่าคีย์หลักเพิ่มเป็น 3 แอตทริบิวต์ ซึ่งถ้านำไปเป็นคีย์อ้างอิงกับตารางอื่นจะเกิดความยุ่งยาก

3.9 การจัดเก็บฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุ

ข้อดีของฐานข้อมูลเชิงวัตถุนั้นไม่เพียงแต่มีความสามารถตามแนวคิดเชิงวัตถุซึ่งได้กล่าวไว้ในบทที่ 3 แล้ว ในการจัดเก็บฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุยังสามารถแก้ปัญหาที่เกิดกับฐานข้อมูลเชิงสัมพันธ์ได้ ดังนี้

1) การจัดเก็บฐานข้อมูลเชิงเวลาโดยใช้ฐานข้อมูลเชิงสัมพันธ์ จะนำช่วงเวลามาเกาะกับทัพเปิล ทำให้ไม่ทราบว่าแอตทริบิวต์ใดที่มีการเปลี่ยนแปลงค่าตามเวลา สำหรับในฐานข้อมูลเชิงวัตถุเราสามารถใส่ timestamp ให้กับแอตทริบิวต์ของออบเจ็กต์ โดยใช้วิธียกแอตทริบิวต์ที่ต้องการใส่ time stamp ขึ้นมาเป็น class ใหม่

2) ในฐานข้อมูลเชิงสัมพันธ์จะใช้ External Identifier คือ แอตทริบิวต์ (หรือกลุ่มของ แอตทริบิวต์) มาเป็นตัวระบุหรืออ้างอิง (Identifier) หรือเรียกว่าคีย์หลัก (Primary Key) ในแต่ละทัพเปิลในตารางเดียวกันค่าของคีย์หลักจะต้องไม่ซ้ำกัน ในการจัดเก็บฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงสัมพันธ์ หากต้องการอัพเดทแอตทริบิวต์ที่มีการเปลี่ยนแปลงค่าตามเวลา จึงมีความจำเป็นที่จะต้องอัพเดทโรว์ใหม่ซึ่งส่งผลให้คีย์หลักซ้ำ แม้ว่าจะสามารถแก้ปัญหาได้โดยการคอมไบน์คีย์หลักแต่จะสร้างความยุ่งยากในการอ้างอิงคีย์หลักและคีย์อ้างอิง (Foreign Key)

สำหรับฐานข้อมูลเชิงวัตถุจะไม่เกิดปัญหานี้ เนื่องจากใช้ Object Identity (OID) ในการระบุหรืออ้างอิงออบเจ็กต์ ซึ่งแต่ละออบเจ็กต์จะมี OID ไม่ซ้ำกัน ซึ่ง OID จะถูกกำหนดให้โดยระบบตั้งแต่เมื่อออบเจ็กต์ถูกสร้างขึ้น ไม่สามารถเปลี่ยนได้และไม่ซ้ำกันอย่างแน่นอน เรียกว่า Internal Identifier

นอกจากนี้ฐานข้อมูลเชิงวัตถุยังสนับสนุนการทำงานกับระบบ (Application) ซึ่งออกแบบโดยใช้แนวคิดเชิงวัตถุ และทำการออกแบบด้วยคลาสไดอะแกรมของ UML โดยสำหรับฐานข้อมูลเชิงสัมพันธ์ผู้ออกแบบจะต้องทำการแปลงจากคลาสไดอะแกรมให้เป็น Relational Schema และจะต้องสร้างโปรแกรมสำหรับจัดการกับข้อมูลที่ได้จากฐานข้อมูลเชิงสัมพันธ์ ให้อยู่ในรูปออบเจ็กต์สำหรับใช้งานจริงในระบบ ทำให้เกิดความยุ่งยากถือได้ว่าเป็นข้อเสียอีกประการหนึ่งของฐานข้อมูลเชิงสัมพันธ์

3.10 Extension for Time in Database

ในการสร้างระบบฐานข้อมูลเชิงเวลานั้น สิ่งที่ non-temporal database system ต้องสนับสนุนมี 3 ประการดังต่อไปนี้คือ

3.10.1 data structure จะต้องสามารถเก็บข้อมูลเกี่ยวกับเวลาได้

การเพิ่มแอตทริบิวต์เกี่ยวกับเวลาเข้าในโครงสร้างข้อมูลนั้นสามารถทำได้ง่ายมาก เราอาจเก็บ valid time โดยเพิ่มแอตทริบิวต์ VTS (Valid Time Start) และ VTE (Valid Time End) ให้กับคลาส Date ส่วน transaction time ก็สามารถทำได้เช่นเดียวกัน

แต่สำหรับโอเปอเรชั่น temporal จะซับซ้อนกว่า เช่น ผลต่างของสองช่วงเวลาอาจได้ผลออกมาเป็นช่วงเวลาหลายๆช่วง (set of interval) ซึ่งโดยทั่วไปจะใช้ temporal element ซึ่งเป็น set of interval เข้ามาช่วยเรื่อง timestamp

สำหรับ relational data model นั้นส่วนที่ซับซ้อนที่สุดคือคือ timestamp ของส่วนของ data structure อย่างเช่น tuple หรือ attribute timestamp ซึ่งใน object data model อาจทำได้ 3 แนวทาง คือ

1. timestamp ที่แอตทริบิวต์หรือ timestamp ในระดับออบเจ็กต์
2. timestamp ในระดับ type
3. timestamp ที่ระดับ identifier

3.10.2 โอเปอเรชั่นต่างๆสำหรับ temporal เพื่อการค้นหาและแก้ไขข้อมูล

การเพิ่มโอเปอเรชั่นเกี่ยวกับ temporal เป็นเรื่องที่มีปัญหามาก จาก temporal algebra ที่ได้มีการนิยามไว้แล้ว เราอาจนำไปทำได้โดยตรงกับระบบฐานข้อมูลเชิงเวลาหรือทำเป็นเลเยอร์อยู่บน non-temporal database system ซึ่งก็จำเป็นต้องมีส่วนขยายของภาษาสืบค้นข้อมูลให้สามารถใช้ โอเปอเรชั่นเกี่ยวกับ temporal operation ร่วมกับส่วนที่เป็น non-temporal อื่นๆได้

แต่ non-temporal database system อย่างเช่น Cache เราสามารถเพิ่มคลาสเกี่ยวกับ temporal และฟังก์ชั่นเกี่ยวกับเวลา ซึ่งจะทำงานเข้ากันกับ non-temporal ที่มีอยู่แล้วได้ง่าย

3.10.2.1 temporal constraint

ใน temporal relational database ตัวระบบจะหน้าที่จัดการ temporal constraint อย่างเช่น referential integrity ให้ แต่สำหรับ non-temporal database system รวมทั้ง Caché จะให้ programmer จัดการกับ temporal constraint ที่เพิ่มขึ้นมาเอง

3.11 การออกแบบฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุด้วย UML

การออกแบบฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุ เราสามารถใช้ภาษามาตรฐาน TODL (Temporal Object Definition Language) ซึ่งเป็นภาษานิยามเชิงเวลาที่สร้างขึ้นเพื่อเพิ่มคุณสมบัติเชิงเวลาเข้าไปในภาษานิยามเชิงวัตถุ ODL (Object Definition Language) ซึ่งเป็นภาษามาตรฐานนิยามฐานข้อมูลเชิงวัตถุ

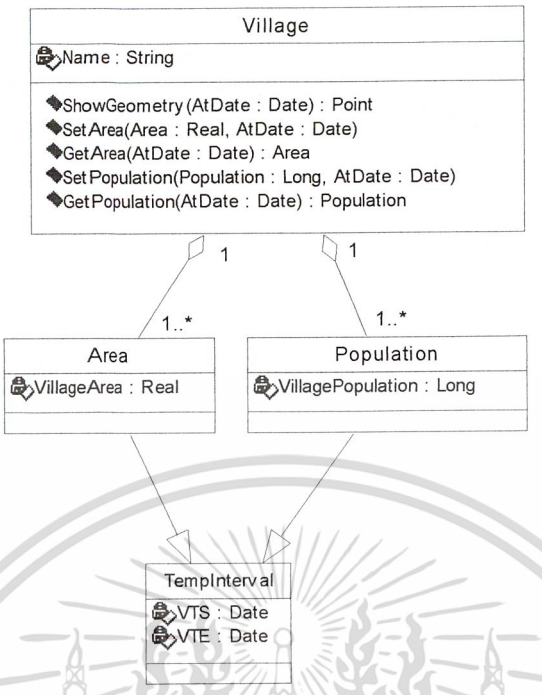
เนื่องจากในปัจจุบันยังไม่ค่อยมี OODBMS ที่สนับสนุนการทำงานร่วมกับ TODL รวมทั้งบาง OODBMS ยังไม่สามารถสนับสนุน ODL ได้อย่างสมบูรณ์ เช่น รูปแบบไวยากรณ์ของภาษาที่ใช้สร้างคลาสของใน OODBMS ไม่ตรงกับ ODL แต่ทว่าก็ยังมีบาง OODBMS บางตัวเช่น Caché สนับสนุนการใช้งานคลาสไดอะแกรมใน UML

ในการออกแบบ Object Schema ซึ่งใช้ในการจัดเก็บข้อมูลได้อีกด้วย

ดังนั้นจึงต้องใช้ Class Diagram ในการออกแบบแทนซึ่ง แอตทริบิวต์ของออบเจ็กต์ที่ต้องการให้เป็นเชิงเวลาเราสามารถใส่ timestamp ให้กับแอตทริบิวต์ของออบเจ็กต์นั้น โดยใช้วิธียกแอตทริบิวต์ขึ้นมาเป็นคลาสใหม่ ซึ่งสืบทอดมาจากคลาส TempInterval ดังรูป 3.24 ข้างล่าง

รูปภาพ 3-24รูป Class TempInterval

ตัวอย่างการออกแบบฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุด้วย UML เป็นดังรูปข้างล่าง ซึ่งคลาส Village จะเก็บข้อมูลจำนวนประชากร (Population) และขนาดของหมู่บ้าน (Area) ข้อมูลแต่ละชนิดสามารถเปลี่ยนแปลงตามเวลาได้แต่ใน แต่ละช่วงเวลาสามารถมีค่าของแอตทริบิวต์นั้นๆได้เพียงค่าเดียว เช่น ในวันที่ 1/1/2000 ถึง 1/1/2001 มีประชากร 500 คน และในวันที่ 1/1/2001 ถึง 31/12/9999 มีประชากร 534 คน



รูปภาพ 3-25 ตัวอย่างการออกแบบฐานข้อมูลเชิงเวลาบนฐานข้อมูลเชิงวัตถุด้วย UML

แอตทริบิวต์ Area และ Population ที่เราต้องการจัดเก็บเชิงเวลานั้นจะต้องแยกออกมาเป็นอีกคลาสหนึ่งซึ่งจะต้องสืบทอดมาจากคลาส TempInterval โดยที่คลาส Village จะมีเมดถอดที่ใช้ method ที่ใช้ในการแก้ไข และดึงค่าแอตทริบิวต์ที่เป็นเชิงเวลา Area และ Population ซึ่งสามารถกำหนดเวลาที่ต้องการแก้ไขและดึงค่าได้ด้วย

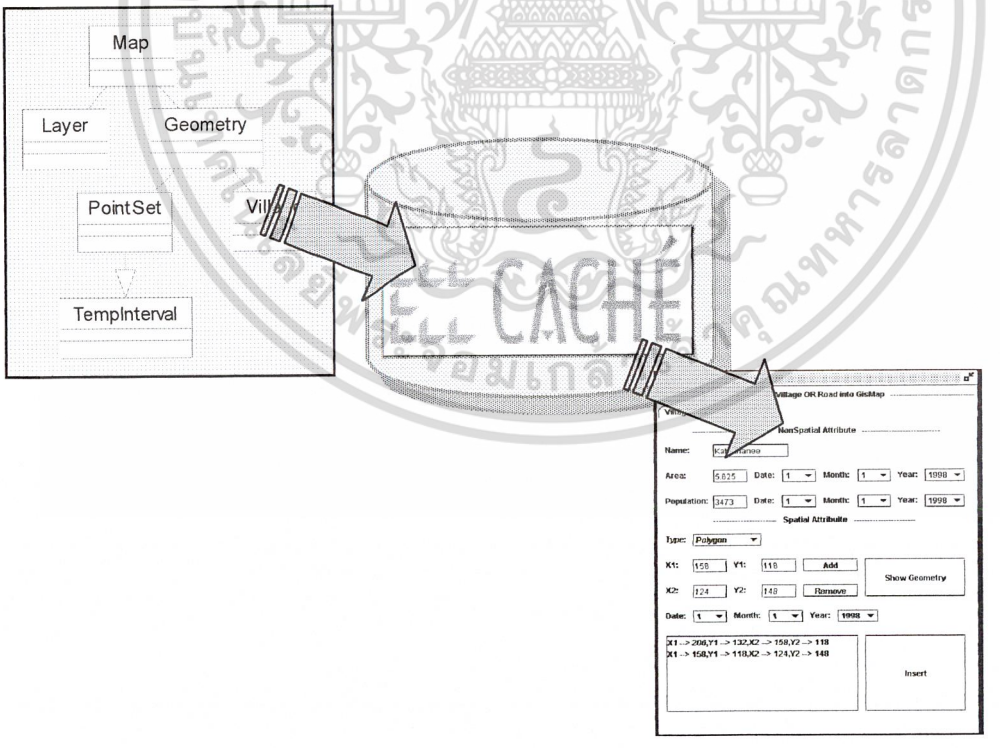
บทที่ 4
การพัฒนาระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่

เนื้อหาในบทนี้เราจะกล่าวถึงการพัฒนาระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่ หลังจากที่เราได้ทำการศึกษาความรู้พื้นฐานต่างๆที่จำเป็นไปแล้วในบทที่ 2 และบทที่ 3 โดยเนื้อหาจะกล่าวถึงภาพรวมของสถาปัตยกรรมของระบบว่าประกอบไปด้วยอะไรบ้าง แล้วจากนั้นจะทำการพัฒนาที่ละส่วนเพื่อให้ได้ระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่

4.1 สถาปัตยกรรมระบบ

ในหัวข้อนี้จะกล่าวถึงภาพรวมของการพัฒนาระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่ว่าต้องมีส่วนประกอบอะไรบ้างในระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่ และจะได้กล่าวถึงโปรแกรม Caché ในหัวข้อต่อไป ซึ่งเป็นส่วนสำคัญในการพัฒนาระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่

ส่วนประกอบต่างๆ ที่สำคัญของสถาปัตยกรรมของระบบฐานข้อมูลออบเจ็กต์เชิงเวลาสำหรับแผนที่ที่สำคัญๆ สามารถแสดงได้ดังรูปข้างล่าง



รูปภาพ 4-1ภาพรวมสถาปัตยกรรมของระบบ

ในภาพข้างต้นเราจะเห็นว่าสถาปัตยกรรมรวมของระบบประกอบไปด้วย 3 ส่วนคือ

- ส่วนที่เป็นคลาสไดอะแกรมที่เราได้ออกแบบไว้สำหรับข้อมูลแผนที่ที่เป็นเชิงเวลา

- ส่วนที่เป็นระบบฐานข้อมูล Caché ซึ่งจะทำการจัดเก็บคลาสต่างๆที่เราได้ออกแบบเอาไว้ในส่วนแรกซึ่งส่วนนี้เป็นส่วนที่สำคัญที่สุด เนื่องจากเป็นส่วนที่ใช้ในการเก็บข้อมูลแผนที่ของระบบ และเมทถอดที่ใช้ในการจัดการข้อมูลเป็นเชิงเวลา
- เป็นส่วนของแอพพลิเคชั่นซึ่งจะนำเอาข้อมูลแผนที่เชิงเวลานี้เอาไปใช้งานซึ่งจะต้องทำการติดต่อกับฐานข้อมูล Caché

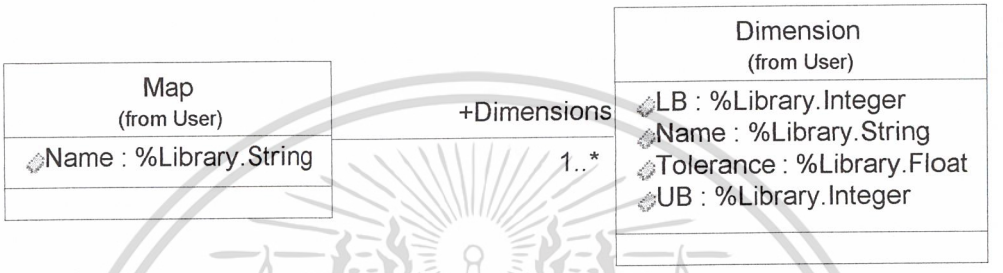
เนื้อหาในบทนี้เราจะกล่าววิธีการพัฒนาในแต่ละส่วนดังกล่าวข้างต้นอย่างละเอียด โดยก่อนที่จะกล่าวถึงวิธีในการพัฒนานั้นเราจำเป็นต้องมีความรู้ในส่วนของโปรแกรม Caché ซึ่งเป็นส่วนสำคัญต้องใช้ในการพัฒนาระบบฐานข้อมูลแผนที่เชิงเวลา โดยจะกล่าวถึงในหัวข้อต่อไป

4.2 ระบบฐานข้อมูลเชิงออบเจ็กต์ Caché

ในหัวข้อนี้เราจะกล่าวถึงระบบฐานข้อมูลเชิงออบเจ็กต์ Caché โดยกล่าวถึงเฉพาะส่วนที่จำเป็นต้องใช้ในการพัฒนาระบบฐานข้อมูลแผนที่เชิงเวลา โดยมีรายละเอียดดังต่อไปนี้

4.2.1 การจัดเก็บข้อมูลใน Caché

เมื่อนำคลาสที่ออกแบบมาสร้างในระบบจัดการฐานข้อมูลเชิงวัตถุ (Object-Oriented Database Management System ; OODBMS) Caché Caché จะเพิ่มแอตทริบิวต์ลงในแต่ละคลาสเพื่อเก็บ OID ของออบเจ็กต์ของคลาสที่มีความสัมพันธ์ด้วย

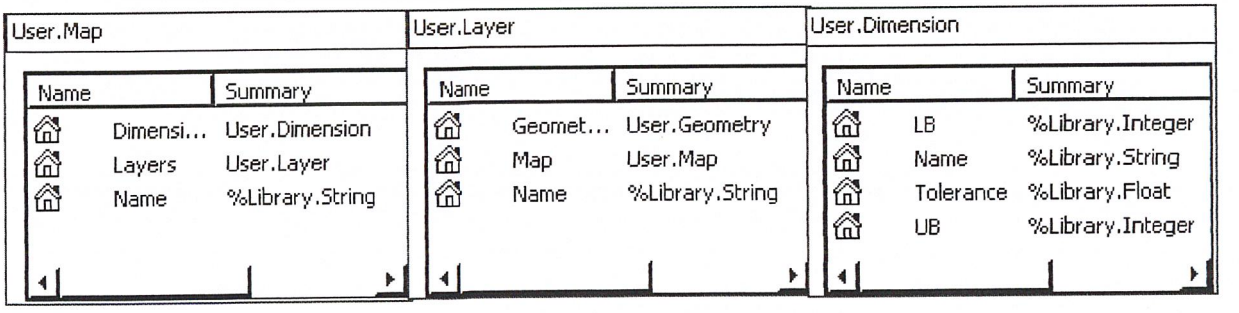


รูปภาพ 4-2ความสัมพันธ์ระหว่างคลาส Map และคลาส Dimension แบบทางเดียว

คลาส Map มีความสัมพันธ์แบบทางเดียวกับคลาส Dimension ดังนั้น Caché จะเพิ่มแอตทริบิวต์ Dimensions ลงในคลาส Map เพื่อทำให้สามารถเข้าถึงออบเจ็กต์ของคลาส Dimension ได้จากออบเจ็กต์ของคลาส Map แต่ออบเจ็กต์ของคลาส Dimension จะไม่สามารถสืบค้น และเข้าถึงออบเจ็กต์ของคลาส Map ได้

รูปภาพ 4-3ความสัมพันธ์ระหว่างคลาส Map และคลาส Layer แบบสองทาง

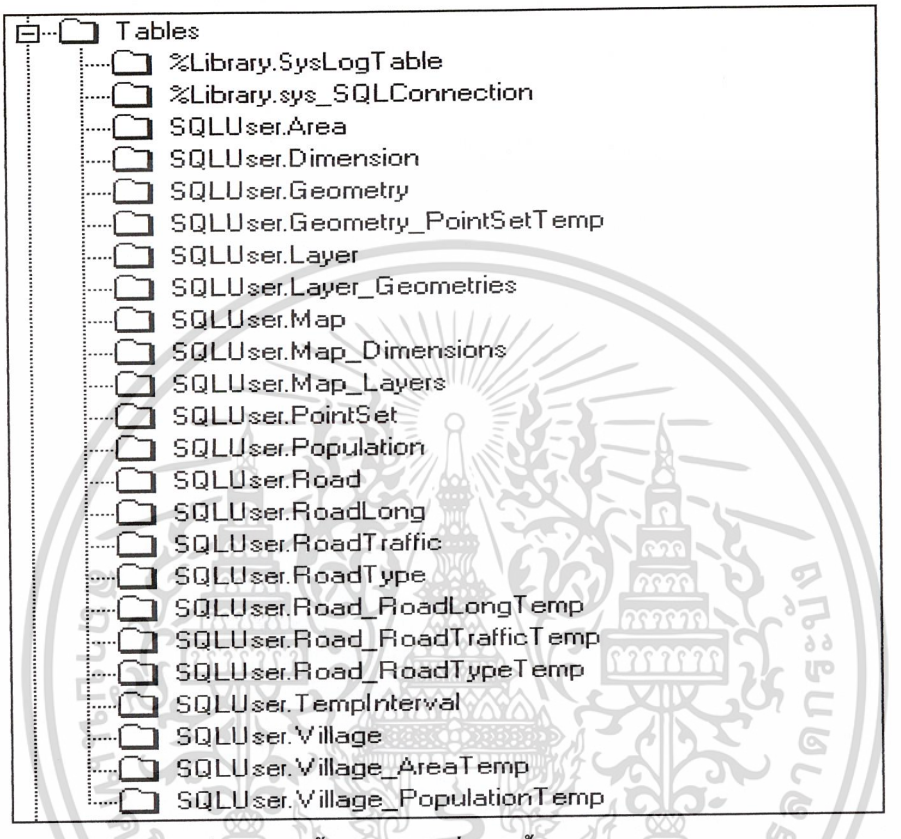
ความสัมพันธ์ระหว่างคลาส Map และคลาส Layer เป็นความสัมพันธ์แบบสองทาง คือออบเจ็กต์ของทั้งสองคลาสจะเก็บแอตทริบิวต์ สำหรับเก็บ OID ของคลาสที่มีความสัมพันธ์ด้วย ทำให้สามารถสืบค้นไปยังอีกคลาสหนึ่งได้



รูปภาพ 4-4 แอตทริบิวต์ของคลาส Map, Layer และ Dimension จาก Caché ObjectArchitect

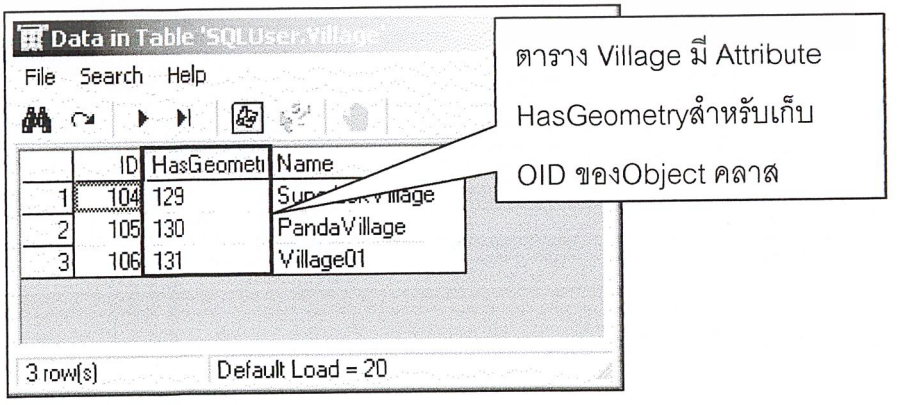
4.2.2 Unified Data Architecture

จากคลาสที่สร้างขึ้นบนระบบจัดการฐานข้อมูล Caché สามารถมองได้ทั้งในรูปแบบ Class และในรูปแบบตาราง ในระบบจัดการฐานข้อมูล Caché ไม่ว่าจะเข้าถึงข้อมูลผ่านทางออบเจ็กต์หรือทางตาราง ผลลัพธ์ที่ได้จะไม่แตกต่างกัน

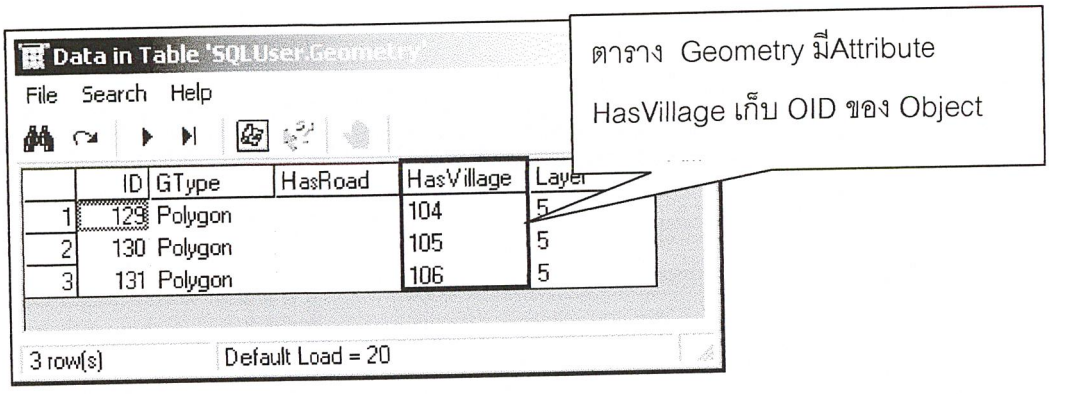


รูปภาพ 4-5ตารางที่ถูกผนวกขึ้นจากคลาสที่สร้างขึ้นบนระบบจัดการฐานข้อมูล Caché

สำหรับคลาสที่มีความสัมพันธ์แบบ 1:1 เช่น คลาส Geometry กับคลาส Road หรือ คลาส Geometry กับคลาส Village Caché จะสร้างแอตทริบิวต์สำหรับเก็บ OID ของคลาสที่มีความสัมพันธ์ด้วย เมื่อนำมาผนวกเป็นตาราง จะไม่มีตารางเก็บความสัมพันธ์ของทั้ง 2 คลาส

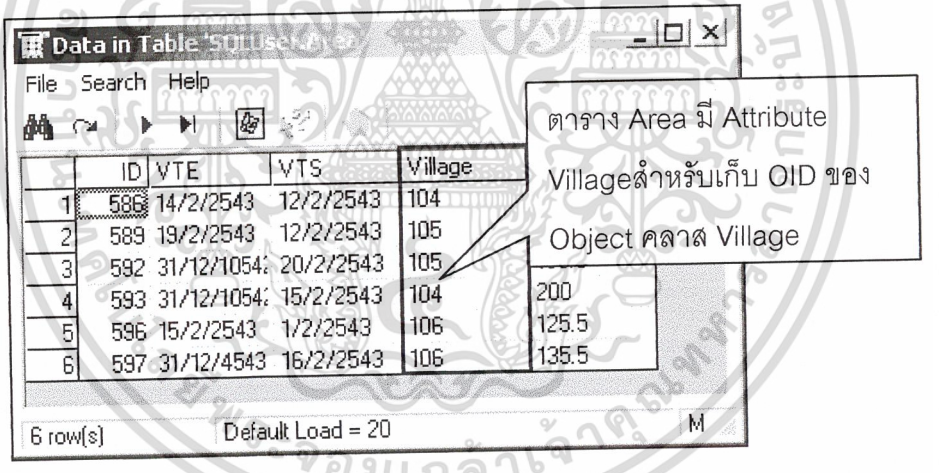


รูปภาพ 4-6 Attribute ของตาราง Village



รูปภาพ 4-7 **Attribute ของตาราง Geometry**

สำหรับคลาสที่มีความสัมพันธ์แบบ 1:N เช่น คลาส Village สัมพันธ์กับ คลาส Area คลาส Area จะมีแอตทริบิวต์ Village สำหรับเก็บ OID ของออบเจ็กต์ของคลาส Village แต่สำหรับออบเจ็กต์ของคลาส Village หนึ่งออบเจ็กต์จะสัมพันธ์กับออบเจ็กต์ของคลาส Area หลายๆออบเจ็กต์เนื่องจากข้อมูลขนาดของหมู่บ้านเป็นข้อมูลเชิงเวลา Caché จึงสร้างตาราง Village_AreaTemp เพื่อเก็บ OID ของออบเจ็กต์ของคลาส Village และ OID ของ Object คลาส Area ที่มีความสัมพันธ์กัน



รูปภาพ 4-8 Attribute ของตาราง Geometry

4.3 การออกแบบฐานข้อมูลแผนที่เชิงเวลา

สำหรับหัวข้อนี้เราจะกล่าวถึงการออกแบบสร้างต้นแบบของฐานข้อมูลแผนที่เชิงเวลาโดยใช้หลักการของเชิงวัตถุซึ่งเราได้กล่าวถึงไปในบทที่แล้ว ซึ่งการออกแบบข้อมูลแผนที่ให้เป็นเชิงเวลานั้นจะทำทั้งสองส่วนคือทั้งส่วนที่เป็นข้อมูลเชิงบรรยาย และข้อมูลเชิงพื้นที่ โดยข้อมูลเชิงบรรยายนั้นเราสามารถออกแบบได้ตามความต้องการของผู้ใช้ว่าต้องการให้มีการจัดเก็บข้อมูลอะไรบ้าง ส่วนข้อมูลเชิงพื้นที่ ซึ่งจะต้องทำการเชื่อมต่อกับข้อมูลเชิงบรรยายเพื่อทำการแสดงรูปของข้อมูลเชิงบรรยายนั้น ได้อ้างอิงจาก Spatial Cartridge ของโอราเคิล ซึ่งเราจะกล่าวถึงรายละเอียดในหัวข้อต่อไป

4.4 การออกแบบข้อมูลเชิงพื้นที่แบบเชิงเวลา

ในส่วนนี้เราจะทำการออกแบบข้อมุลแผนที่ในส่วนที่เป็นข้อมูลเชิงพื้นที่ โดยจะทำการออกแบบให้เป็นเชิงเวลาด้วย ซึ่งส่วนที่เป็นข้อมุลเชิงพื้นที่ จะทำการออกแบบข้อมูลเชิงพื้นที่ โดยอ้างอิงจาก Spatial Cartridge ของโอราเคิล ซึ่งใช้ตาราง 4 ตารางสำหรับเก็บข้อมูลเชิงพื้นที่ และเก็บอินเด็กซ์ของข้อมูล โดยในแต่ละเลเยอร์ของ Spatial Cartridge นี้ประกอบด้วย 4 ตารางดังกล่าวนี้

SDO_ORDCNT	SDO_LEVEL	SDO_NUMTILES	SDO_COORDSYS
Number	Number	Number	Varchar

ตาราง 4-1 ตาราง <layername>_SDOLAYER

SDO_DIMNUM	SDO_LB	SDO_UB	SDO_TOLERANCE	SDO_DIMNAME
Number	Number	Number	Varchar	Varchar

ตาราง 4-2 ตาราง <layername>_SDODIM

SDO_GID	SDO_ESEQ	SDO_ETYGE	SDO_SEQ	SDO_X1	SDO_Y1	...	SDO_XN	SDO_YN
Number	Number	Number	Number	Number	Number	...	Number	Number

ตาราง 4-3 ตาราง <layername>_SDOGEOM

SDO_GID	SDO_CODE	SDO_MAXCODE	SDO_GROUPCODE	SDO_META
Number	Raw	Raw	Raw	Raw

ตาราง 4-4 ตาราง <layername>_SDOINDEX

คอลัมน์ของแต่ละตารางถูกกำหนดไว้ดังนี้

ตาราง <layername>_SDOLAYER

- **SDO_ORDCNT** คอลัมน์ SDO_ORDCNT เป็นจำนวนของจุดพิกัดที่เก็บใน 1 แถว (row) ขนาดตาราง <layername>_SDOGEOM ซึ่งค่าของคอลัมน์นี้จะเป็นจำนวนของ คอลัมน์ที่เก็บข้อมูลเกี่ยวกับจุดพิกัดของตาราง <layername>_SDOGEOM ตัวอย่างเช่น ถ้าในตาราง <layername>_SDOGEOM เก็บข้อมูลจุดพิกัดได้ 4 จุดค่าของคอลัมน์นี้ก็ จะเป็น 8 เนื่องจากใน 1 จุดจะมีการเก็บข้อมูลเป็น 2 คอลัมน์สำหรับเก็บค่า X และ Y
- **SDO_LEVEL** คอลัมน์ SDO_LEVEL เป็นคอลัมมน์ที่ใช้เก็บจำนวนครั้งของการแบ่งชั้นข้อมูลออกเป็นช่องสี่เหลี่ยมย่อยๆ ในระหว่างการทำอินเด็กซ์ ของข้อมูลเชิงพื้นที่

- **SDO_NUMTILES** คอลัมน์เก็บจำนวนของช่องสี่เหลี่ยมที่เปลี่ยนแปลงขนาดได้ (variable size tile) ที่ถูกใช้เพื่อแบ่งแต่ละวัตถุในตาราง <layername>_SDOGEOM ค่าใน คอลัมน์นี้จะมีค่าเป็น NULL เมื่อใช้ช่องสี่เหลี่ยมในการแบ่งที่มีขนาดคงที่
- **SDO_CORRDSYS** คอลัมน์นี้เป็นออปชั่น ใช้บอกชื่อของระบบโคออร์ดิเนต โดยใช้มาตรฐานตัวอย่างเช่น GOSC หรือ OGIS

ตาราง <layername>_SDODIM

- **SDO_DIMNUM** คอลัมน์นี้เก็บหมายเลขมิติเริ่มต้นที่หมายเลข 1 และเพิ่มขึ้นเรื่อยๆ
- **SDO_LB** คอลัมน์นี้เก็บของเขตล่างของพิกัดในมิตินี้ ตัวอย่างเช่น มิติเป็นละติจูดขอบเขตล่างควรเป็น -90
- **SDO_UB** คอลัมน์นี้เก็บของเขตบนของพิกัดในมิตินี้ ตัวอย่างเช่น มิติเป็นละติจูดขอบเขตบนควรเป็น 180
- **SDO_TOLERANCE** คอลัมน์นี้เก็บระยะห่างระหว่างจุดสองจุดที่สามารถพิจารณาเป็นจุดเดียวกันได้ เนื่องจากการปัดเศษ ค่านี้ต้องมากกว่า 0 ตัวอย่างเช่นอาจจะมีค่าเป็น 0.5 เป็นต้น
- **SDO_DIMNAME** คอลัมน์นี้เก็บชื่อของมิติ เช่น ลองจิจูด,ละติจูด , x หรือ y

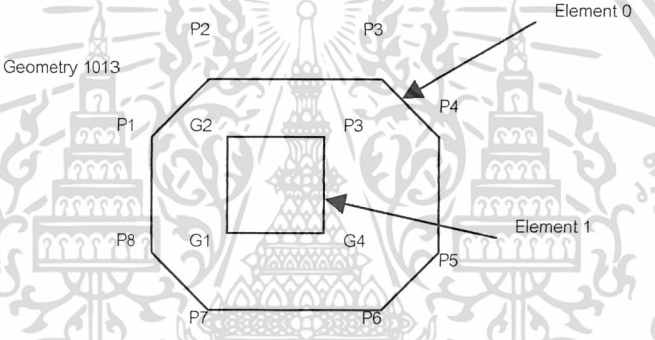
ตาราง <layername>_SDOGEOM

- **SDO_GID** คอลัมน์นี้เก็บหมายเลขที่ไม่ซ้ำซึ่งเป็นตัวบ่งชี้รูปทรงเรขาคณิตในเลเยอร์
- **SDO_ESEQ** คอลัมน์นี้เก็บลำดับหมายเลขของแต่ละอีเลเมนต์ (Element Sequence Number) ในรูปทรงเรขาคณิต
- **SDO_ETYGE** คอลัมน์นี้เก็บชนิดข้อมูลของรูปทรงเรขาคณิตพื้นฐานของอีลีเมนต์ซึ่งจะมี 3 ชนิดคือ SDO_GEOM.GOINT_TYGE (ชนิดจุด) , SDO_GEOM.LINESTRING_TYGE (ชนิดเส้น) , SDO_GEOM.GOLYGON_TYGE (ชนิดพื้นที่รูปปิด) ซึ่งอาจแทนค่าได้ด้วย 1,2,3 ตามลำดับ ซึ่งหากมีค่าเป็นศูนย์จะถูกเพิกเฉย
- **SDO_SEQ** คอลัมน์นี้เก็บหมายเลขลำดับของแต่ละแถวของข้อมูลที่ทำให้เกิดอีลีเมนต์
- **SDO_X1** คอลัมน์นี้เก็บค่าของคู่ลำดับที่ 1
- **SDO_Y1** คอลัมน์นี้เก็บค่าของคู่ลำดับที่ 1
- **SDO_Xn** คอลัมน์นี้เก็บค่าของคู่ลำดับที่ n
- **SDO_Yn** คอลัมน์นี้เก็บค่าของคู่ลำดับที่ n

ตาราง <layername>_SDOINDEX

- **SDO_GID** คอลัมน์นี้เก็บหมายเลขที่ไม่ซ้ำซึ่งเป็นตัวบ่งชี้รูปทรงเรขาคณิตในเลเยอร์เป็น FOREIGN KEY กลับไปยังตาราง <layername>_SDOGEOM

- **SDO_CODE** คอลัมน์นี้เก็บหมายเลขของช่องสี่เหลี่ยมย่อย (tile) ที่ปกคลุม SDO_GID จำนวนของไบต์ที่ต้องใช้สำหรับคอลัมน์ SDO_CODE และ SDO_MAXCODE ขึ้นอยู่กับระดับของการแบ่งช่องสี่เหลี่ยมย่อยซึ่งเราสามารถใช้ฟังก์ชั่น
- **SDO_ADMIN.SD_CODE_SIZE** เพื่อหาขนาดที่ต้องการในแต่ละเลเยอร์ได้ จำนวนไบต์สูงสุดที่เป็นไปได้คือขนาด 255 ไบต์
- **SDO_MAXCODE** คอลัมน์นี้เก็บค่าของช่องสี่เหลี่ยมที่สามารถเปลี่ยนแปลงขนาดได้ซึ่งเป็นช่องสี่เหลี่ยมย่อยที่เล็กที่สุดในควอทแดรท์ปัจจุบัน คอลัมน์นี้จะไม่ถูกใช้สำหรับ fixed size tile
- **SDO_GROUPCODE** คอลัมน์นี้เก็บส่วนหน้า (Grefix) ของ SDO_CODE แทนค่า variable-size tile ที่ระดับ <layername>_SDOLAYER ,SDO_LAYER ซึ่งบรรจุหรือเท่า กับช่องสี่เหลี่ยมที่ถูกแทนโดย SDO_CODE คอลัมน์นี้ไม่ถูกใช้สำหรับ fixed-size tile
- **SDO_META** คอลัมน์นี้ไม่จำเป็นสำหรับการเรียกข้อมูล เชิงพื้นที่ แต่จะจัดเตรียมข้อมูลที่จำเป็นในการหาขอบเขตของช่องสี่เหลี่ยม (tile) Geometry : 1013



รูปภาพ 4-9 ตัวอย่างรูปทรงเรขาคณิต

SDO_ORDCNT (number)
4

ตาราง 4-5 ตัวอย่างข้อมูลในตาราง <layername>_sdolayer

SDO_DIMNUM (number)	SDO_LB (number)	SDO_UB (number)	SDO_TOLERANCE (number)	SDO_DIMNAME (number)
1	0	100	.05	X axis
2	0	100	.05	Y axis

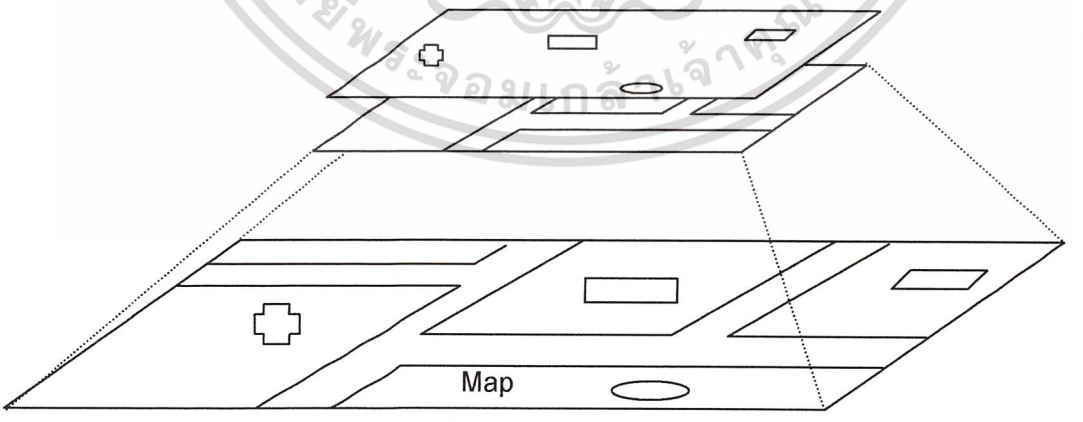
ตาราง 4-6 ตัวอย่างข้อมูลในตาราง <layername>_sdodim

SDO_GID (number)	SDO_ESEQ (number)	SDO_ETYGE (number)	SDO_SEQ (number)	SDO_X1 (number)	SDO_Y1 (number)	SDO (num	SDO (num
1013	0	3	0	P1(X)	P1(Y)	P2(X)	P2(Y)

SDO_GID (number)	SDO_ESEQ (number)	SDO_ETYGE (number)	SDO_SEQ (number)	SDO_X1 (number)	SDO_Y1 (number)	SDO (num	SDO (num
1013	0	3	1	P2(X)	P2(Y)	P3(X)	P3(Y)
1013	0	3	2	P3(X)	P3(Y)	P4(X)	P4(Y)
1013	0	3	3	P4(X)	P4(Y)	P5(X)	P5(Y)
1013	0	3	4	P5(X)	P5(Y)	P6(X)	P6(Y)
1013	0	3	5	P6(X)	P6(Y)	P7(X)	P7(Y)
1013	0	3	6	P7(X)	P7(Y)	P8(X)	P8(Y)
1013	0	3	7	P8(X)	P8(Y)	P1(X)	P1(Y)
1013	1	3	0	G1(X)	G1(Y)	G2(X)	G2(Y)
1013	1	3	1	G2(X)	G2(Y)	G3(X)	G3(Y)
1013	1	3	2	G3(X)	G3(Y)	G4(X)	G4(Y)
1013	1	0	3	G4(X)	G4(Y)	G1(X)	G1(Y)

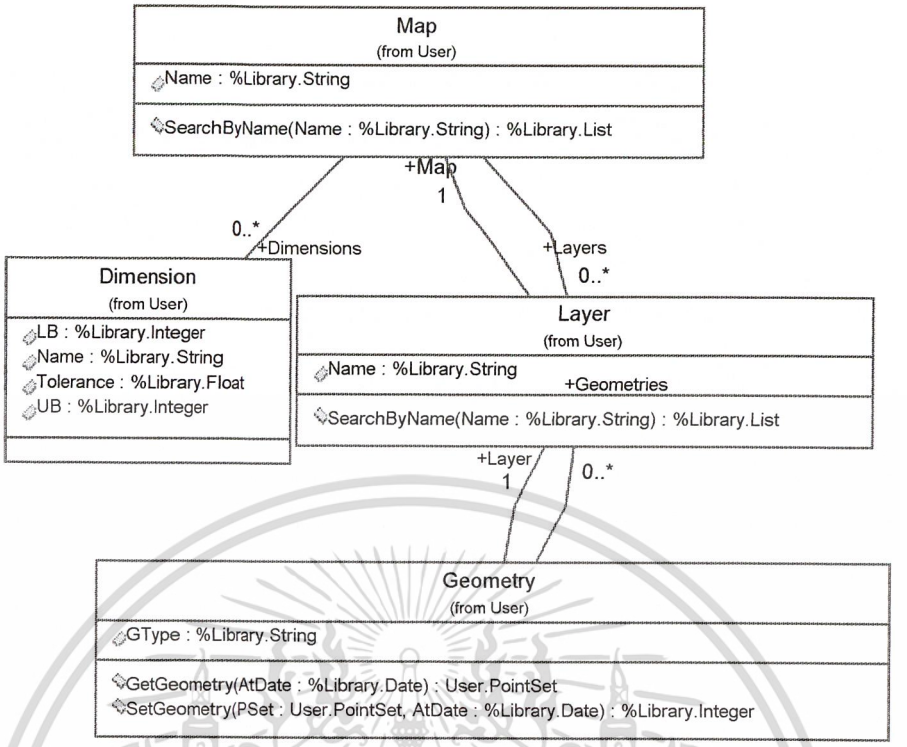
ตาราง 4-7 ตัวอย่างข้อมูลในตาราง <layername>_sdogeom

จากตาราง Spatial Cartridge 4 ตารางข้างต้น สามารถมองการจัดเก็บข้อมูลเชิงพื้นที่เป็นตามลำดับชั้นข้อมูล โดยที่ในแผนที่ Map (แผนที่) หนึ่งสามารถมีได้หลายๆ Layer (เลเยอร์) ซึ่งใช้เก็บข้อมูลส่วนที่สนใจเฉพาะด้านเช่น เลเยอร์ถนน ก็จะใช้เก็บข้อมูลเชิงพื้นที่ของถนนเท่านั้น ดังนั้นในแผนที่หนึ่งจึงสามารถแสดงได้ โดยนำเลเยอร์มาซ้อนทับกันหลายๆชั้น นอกจากนี้ในแต่ละเลเยอร์ยังมีการเก็บรูปที่แสดงข้อมูลเชิงพื้นที่ (Geometry) เอาไว้หลายๆรูปอีกด้วย



รูปภาพ 4-10 แสดงการจัดเก็บข้อมูลตามลำดับชั้นของข้อมูล

จากรูปแสดงการจัดเก็บข้อมูลตามลำดับชั้นของข้อมูลกับตารางทั้งสี่ของ Spatial Cartridge เราสามารถนำมาดัดแปลงออกแบบเป็นคลาสไดอะแกรมสำหรับข้อมูลเชิงพื้นที่ให้เป็นเชิงเวลาได้ดังรูปข้างล่าง



รูปภาพ 4-11ความสัมพันธ์ระหว่างคลาส Map, คลาส Village, คลาส Layer และคลาส Geometry

ในคลาสไดอะแกรมดังรูปข้างต้น มีคลาสที่เกี่ยวกับข้อมูลเชิงพื้นที่ คือ Map,Dimension,Layer, GeometryPointSet และ TempInterval ซึ่งมีรายละเอียดดังต่อไปนี้

Class Map : เมื่อนำข้อมูลที่สนใจทั้งหมดมารวมเข้าด้วยกันเป็นแผนที่ ซึ่งถูกแทนด้วย คลาส Map

- Attribute::Name ใช้เก็บชื่อของแต่ละแผนที่
- Method::SearchByName รับพารามิเตอร์เป็นชื่อของแผนที่เพื่อใช้ในการค้นหาแล้วคืนค่าเป็นลิสต์ของ OID ของ ออบเจ็กต์ Map ที่มีแอตทริบิวต์ Name ตรงกับชื่อแผนที่ที่ส่งเข้ามาในเมดถอด

Class Dimension : ใช้สำหรับเก็บคุณสมบัติของมิติ (X, Y) เช่น ขอบเขตบนและขอบเขตล่าง

- Attribute::Name ชื่อมิติ เช่น X, Y, Z
- Attribute::LB ขอบเขตล่างของมิตินั้น
- Attribute::UB ขอบเขตบนของมิตินั้น
- Attribute::Tolerance ระยะห่างระหว่างจุดสองจุดที่สามารถพิจารณาเป็นจุดเดียวกันได้ เนื่องจากค่าที่ได้จากการคำนวณอาจเป็นจุดทศนิยม แต่เมื่อนำมาแสดงบนจอภาพจะต้องเป็นค่าจำนวนเต็ม

Class Layer : ข้อมูลแผนที่จะจัดเก็บไว้ในคลาสเลเยอร์ข้อมูลที่สนใจแต่ละเรื่องจะถูกแยกเป็นเลเยอร์ในแต่ละแผนที่สามารถมีเลเยอร์ได้หลายเลเยอร์จากในระบบตัวอย่างจะสนใจข้อมูล 2 ชนิด คือ หมู่บ้านและถนน จึงนำมาสร้างเป็นคลาสเลเยอร์ได้สองคลาสคือ คลาสเลเยอร์ที่มีแอตทริบิวต์ Name เป็น Village และ Road

- Attribute::Name คือ ชื่อของข้อมูลที่สนใจที่ถูกจัดเก็บในเลเยอร์นั้น
- Method::SearchByName รับพารามิเตอร์เป็นชื่อเลเยอร์เพื่อค้นหาลิสต์ของ OID ของเลเยอร์ที่มีแอตทริบิวต์ Name ตรงกับที่กำหนด

Class PointSet : เป็นคลาสสำหรับใช้จัดเก็บข้อมูลเชิงพื้นที่ และเก็บช่วงเวลาที่ข้อมูลนี้เป็นจริงเอาไว้ด้วย ดังนั้นจึงต้องมีการสืบทอดมาจากคลาส TempInterval ด้วยโดยคลาสนี้จะจัดเก็บส่วนของเส้นตรงที่สามารถนำมาประกอบกันเป็นรูป polygon และ line ที่มีหลายเหลี่ยมมุมได้

- Attrubute::X1 เก็บพิกัด X ของจุดเริ่มต้นส่วนของเส้นตรง
- Attrubute::Y1 เก็บพิกัด Y ของจุดเริ่มต้นส่วนของเส้นตรง
- Attrubute::X2 เก็บพิกัด X ของจุดสิ้นสุดส่วนของเส้นตรง
- Attrubute::Y2 เก็บพิกัด Y ของสิ้นสุดส่วนของเส้นตรง

Class TempInterval : จัดเก็บช่วงเวลาที่ข้อมูลนั้นเป็นจริง

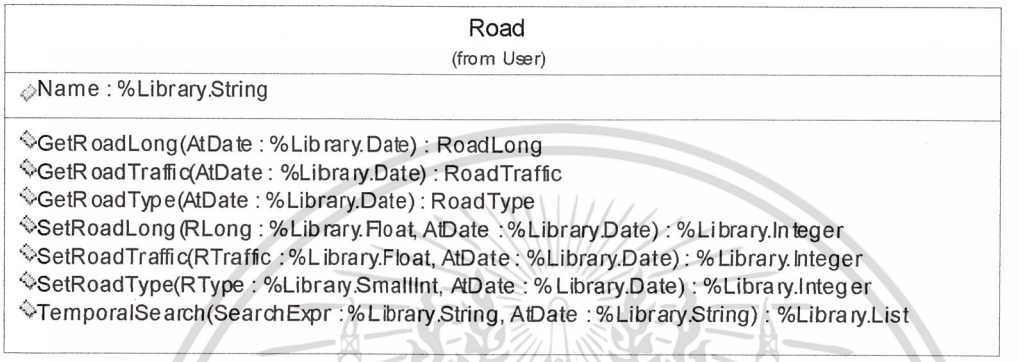
- Attribute::VTS จัดเก็บ ValidTimeStart คือ เวลาเริ่มต้นที่ข้อมูลนั้นเป็นจริง
- Attribute::VTE จัดเก็บ ValidTimeStop คือ เวลาสุดท้ายที่ข้อมูลนั้นเป็นจริง

Class Geometry : จัดเก็บข้อมูลที่เป็นทั้งเชิงพื้นที่ที่เป็นเชิงเวลา ซึ่งในแต่ละออบเจ็กต์ Geometryจะมีออบเจ็กต์ PointSet ได้หลายๆออบเจ็กต์ตามช่วงเวลาที่แตกต่างกัน

- Attribute::GType เป็นแอตทริบิวต์ของข้อมูลเชิงพื้นที่ สำหรับระบุว่าข้อมูลเชิงพื้นที่นั้นมี Geometry แบบไหน เช่น เป็นเส้น (Line) หรือ วงปิด (Polygon) เป็นต้น
- Method::GetGeometry ใช้สำหรับนำข้อมูลที่อยู่ในออบเจ็กต์ PointSet ของคลาส Geometry นั้นออกมานั้นตามพารามิเตอร์ AtDate ซึ่งแราได้ส่งไป
- Method::SetGeometry ใช้สำหรับกำหนดค่าให้แก่ออบเจ็กต์ PointSet ของคลาส Geometry นั้นโดยพารามิเตอร์ AtDate คือเวลาล่าสุดที่ข้อมูลนั้นเป็นจริง โดยเมดถอดนี้จะทำการเก็บข้อมูล State เก่าเอาไว้อีกด้วย

4.5 การออกแบบข้อมูลเชิงเวลา

ในการออกแบบข้อมูลเชิงบรรยายแบบเชิงเวลานั้น เราสามารถออกแบบให้มีกี่คลาสก็ได้แล้วแต่ความต้องการของแอพพลิเคชั่น ซึ่งในโครงงานนี้ได้ทำการสร้างคลาสข้อมูลเชิงบรรยายตัวอย่างสองคลาสด้วยกันคือคลาส Village และคลาส Road ซึ่งมีคลาสไดอะแกรมของคลาส Village เมื่อทำการออกแบบให้เป็นเชิงเวลาได้ดังรูปข้างล่าง



รูปภาพ 4-12 Attribute และ Method ของคลาส Village

ซึ่งรายละเอียดของคลาส Village มีดังต่อไปนี้

NonTemporal Attribute :: Name จัดเก็บชื่อของหมู่บ้าน

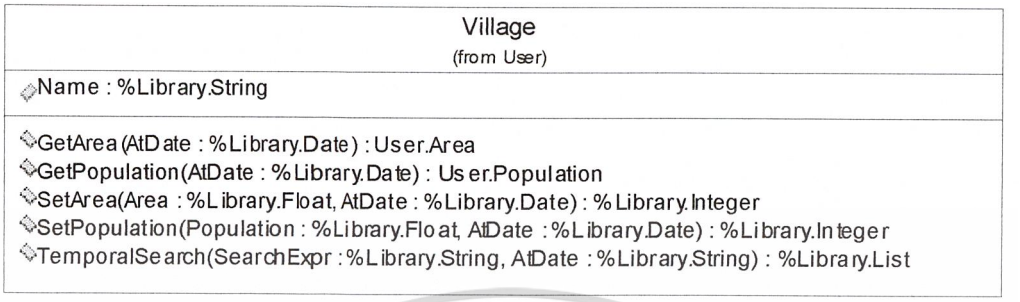
Temporal Attribute :: Area จัดเก็บพื้นที่ของหมู่บ้านซึ่งสามารถเปลี่ยนแปลงได้ตามเวลา

Temporal Attribute :: Population จัดเก็บจำนวนประชากรซึ่งสามารถเปลี่ยนแปลงได้ตามเวลา

เนื่องจากคุณสมบัติของหมู่บ้านที่เราสนใจ เช่น พื้นที่ และ จำนวนประชากร สามารถเปลี่ยนแปลงได้ตามเวลาที่เปลี่ยนไปจึงต้องสืบทอดมาจากคลาส TempInterval และสร้างเมดถอดสำหรับจัดการแอตทริบิวต์เหล่านี้โดยต้องรับค่าเวลา (AtDate) เข้ามาเป็นพารามิเตอร์

- Method::GetArea ใช้สำหรับเรียกใช้ค่าขนาดพื้นที่ของหมู่บ้านเป็นเชิงเวลา
- Method::SetArea ใช้สำหรับกำหนดค่าขนาดพื้นที่ของหมู่บ้านเป็นเชิงเวลา
- Method::GetPopulation ใช้สำหรับเรียกใช้ค่าจำนวนประชากรในหมู่บ้านเป็นเชิงเวลา
- Method::SetPopulation ใช้สำหรับกำหนดค่าจำนวนประชากรในหมู่บ้านเป็นเชิงเวลา
- Method::TemporalSearch ใช้สำหรับค้นหาแอตทริบิวต์ของหมู่บ้านตามเวลาที่กำหนด

ส่วนคลาสไดอะแกรมของคลาส Road เมื่อทำการออกแบบให้เป็นเชิงเวลาได้ดังรูปข้างล่าง ซึ่งรายละเอียดของคลาส Road มีดังต่อไปนี้



รูปภาพ 4-13 Attribute และ Method ของคลาส Village

NonTemporal Attribute::Name จัดเก็บชื่อของถนน

Temporal Attribute :: RoadLong จัดเก็บความยาวของถนนซึ่งสามารถเปลี่ยนแปลงได้ตามเวลา

Temporal Attribute :: RoadTraffic จัดเก็บความหนาแน่นของการจราจรซึ่งสามารถเปลี่ยนแปลงได้ตามเวลา

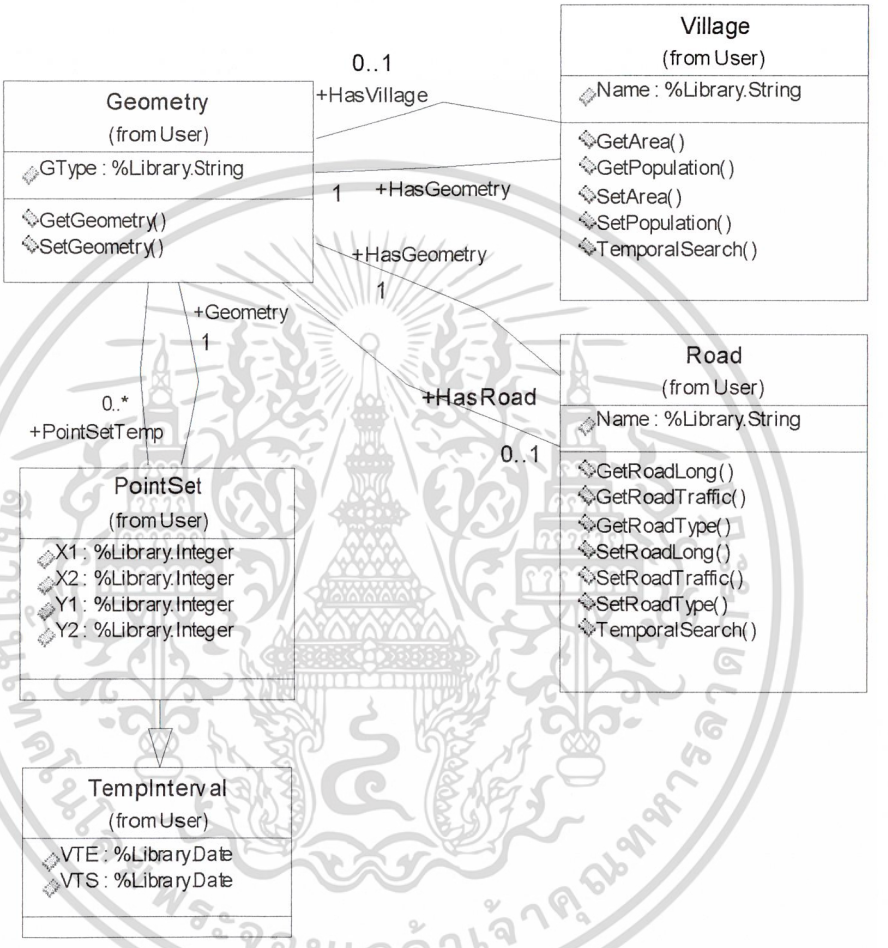
Temporal Attribute :: RoadType จัดเก็บชนิดของถนนซึ่งสามารถเปลี่ยนแปลงได้ตามเวลา

เนื่องจากคุณสมบัติของถนนที่เราสนใจ เช่น ความยาว(roadlong), ระดับขนาดของถนน (roadtype) และความคับคั่งของการจราจร (roadtraffic) สามารถเปลี่ยนแปลงได้ตามเวลาที่เปลี่ยนไปจึงต้องสืบทอดมาจากคลาส TempInterval และสร้างเมดถอดสำหรับจัดการแอตทริบิวต์เหล่านี้โดยต้องรับค่าเวลาเข้ามาเป็นพารามิเตอร์

- Method::GetRoadType ใช้สำหรับเรียกใช้ค่าชนิดของถนนเป็นเชิงเวลา
- Method::SetRoadType ใช้สำหรับกำหนดค่าชนิดของถนนเป็นเชิงเวลา
- Method::GetRoadLong ใช้สำหรับเรียกใช้ค่าความยาวของถนนเป็นเชิงเวลา
- Method::SetRoadLong ใช้สำหรับกำหนดค่าความยาวของถนนเป็นเชิงเวลา
- Method::GetRoadTraffic ใช้สำหรับเรียกใช้ค่าความคับคั่งของการจราจรเป็นเชิงเวลา
- Method::SetRoadTraffic ใช้สำหรับกำหนดค่าความคับคั่งของการจราจรเป็นเชิงเวลา
- Method::TemporalSearch ใช้สำหรับค้นหาแอตทริบิวต์ของถนนตามเวลาที่กำหนด

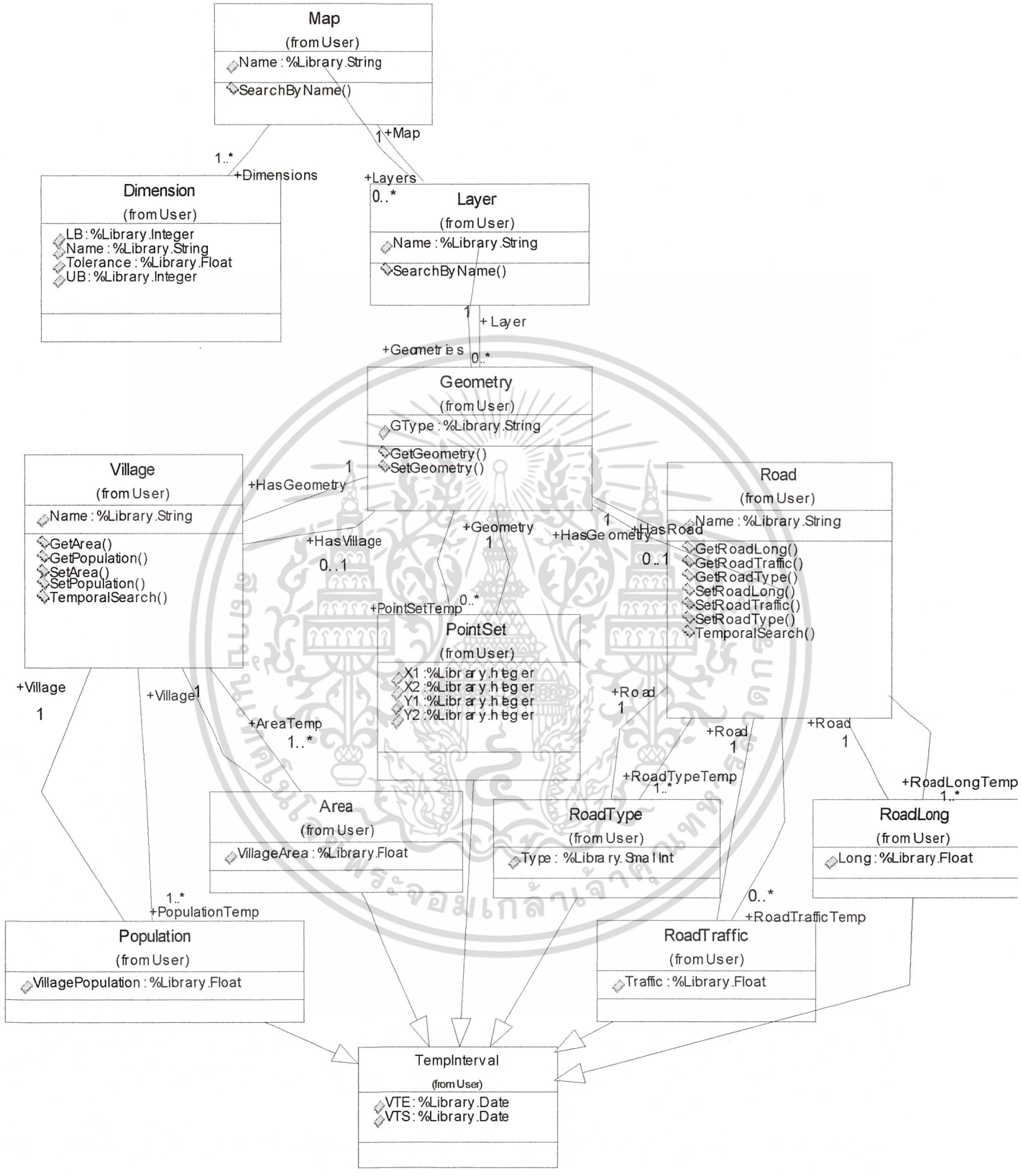
4.6 การออกแบบการเชื่อมต่อข้อมูลเชิงพื้นที่กับเชิงบรรยาย

ส่วนของการออกแบบเชื่อมค่อระหว่างข้อมูลเชิงพื้นที่ และข้อมูลเชิงบรรยายนั้นทำได้โดยนำคลาสของข้อมูลเชิงพื้นที่และเชิงบรรยายมาเชื่อมต่อกันโดยใช้ความสัมพันธ์แบบ association ดังรูปข้างล่างเป็นตัวอย่างการเชื่อมต่อข้อมูลเชิงบรรยายกับข้อมูลเชิงพื้นที่ของคลาส Village กับคลาส Geometry



รูปภาพ 4-14 คลาสแสดงความสัมพันธ์ระหว่างข้อมูลเชิงพื้นที่และเชิงบรรยาย

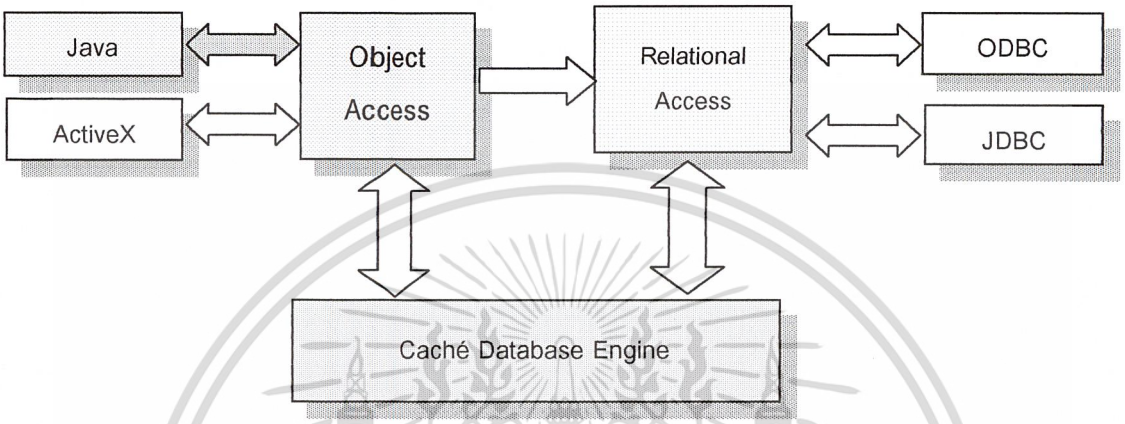
จากรูปจะเห็นว่าความสัมพันธ์แบบ association ระหว่างข้อมูลเชิงพื้นที่ Geometry กับข้อมูลเชิงบรรยาย Village เป็นแบบ 1:1 โดยจะเป็นความสัมพันธ์แบบสองทางเพื่อให้เราสามารถค้นหาข้อมูลจากเชิงบรรยายแล้วเชื่อมต่อไปค้นหาข้อมูลเชิงพื้นที่ที่ตรงกับข้อมูลเชิงบรรยายได้ หรืออาจจะค้นหาข้อมูลจากเชิงพื้นที่แล้วเชื่อมต่อไปค้นหาข้อมูลเชิงบรรยายที่ตรงกับข้อมูลเชิงพื้นที่ได้



รูปภาพ 4-15 คลาส Diagram ของระบบแผนที่

4.7 การติดต่อกับฐานข้อมูล Caché

ในหัวข้อนี้เราจะกล่าวถึงการติดต่อกับฐานข้อมูล Caché เพื่อทำการจัดการกับข้อมูลเชิงพื้นทีและข้อมูลเชิงบรรยายที่ได้ทำการออกแบบไว้ในหัวข้อที่แล้ว เช่น การสอบถามข้อมูลเชิงบรรยาย และพื้นที่ที่เป็นเชิงเวลา,การเพิ่มข้อมูลเชิงบรรยายและพื้นที่ เป็นต้น โดยการติดต่อกับฐานข้อมูลใน Caché สามารถทำได้หลายๆวิธีดังรูปข้างล่าง



รูปภาพ 4-16 แสดงวิธีการติดต่อกับฐานข้อมูล Caché

จากรูปข้างต้นเราสามารถทำการติดต่อกับฐานข้อมูล Caché ได้ทั้งหมด 2 รูปแบบ คือ

- **Relational Access** เป็นการติดต่อกับฐานข้อมูลโดยมองออบเจ็กต์ต่างๆที่เก็บในฐานข้อมูลในรูปแบบของตาราง ซึ่งต้องใช้คำสั่ง SQL ในการติดต่อ
- **Object Access** เป็นการติดต่อกับฐานข้อมูลใน Caché โดยผ่านทางออบเจ็กต์ตัวแทน ซึ่งเมื่อมีกระทำการเปลี่ยนแปลงค่าใดๆในออบเจ็กต์ตัวนี้แล้วจะมีผลต่อข้อมูลในฐานข้อมูลในโครงงานนี้เราได้เลือกแบบการติดต่อกับฐานข้อมูล Caché โดยใช้รูปแบบ Object Access และใช้จาวาในการติดต่อซึ่งการใช้วิธีนี้มีข้อดีดังต่อไปนี้
  - ❑ ง่ายต่อแอพพลิเคชั่นที่ต้องการใช้งานข้อมูลในฐานข้อมูลเนื่องจากถ้ามีการเปลี่ยนแปลงแอตทริบิวต์ของออบเจ็กต์ก็จะมีผลต่อข้อมูลของออบเจ็กต์นั้นในฐานข้อมูลทันที ทำให้ผู้สร้างแอพพลิเคชั่นเห็นภาพพจน์การเขียนโปรแกรมเป็นออบเจ็กต์ทั้งหมด
  - ❑ ไม่ต้องใช้คำสั่ง SQL ซึ่งมีความยุ่งยากซับซ้อน เนื่องจากเราสามารถเปลี่ยนแปลง,แก้ไข,ค้นหาข้อมูลได้โดยผ่านทางออบเจ็กต์
  - ❑ เห็นภาพพจน์การทำงานร่วมกับข้อมูลเชิงพื้นที่และเชิงบรรยายที่ได้ออกแบบไว้ เนื่องจากคลาสที่อยู่ในคลาสไดอะแกรมที่เราออกแบบไว้ เป็นคลาสที่เราต้องทำงานร่วมด้วยในแอพพลิเคชั่นที่เราสร้างขึ้น

- ซ่อนความยุ่งยากในการใช้เขียนโปรแกรมที่ฝั่งแอพพลิเคชั่น เนื่องจากใช้คุณสมบัติ Encapsulation ของออบเจ็กต์ ผู้ใช้แค่ใช้เมทธอดที่ผู้ออกแบบระบบฐานข้อมูลได้จัดเตรียมไว้ให้ก็สามารถสร้างแอพพลิเคชั่นได้แล้ว โดยไม่ต้องทราบว่าข้างในออบเจ็กต์นั้นเป็นยังไง

ซึ่งวิธีนี้เราจะต้องทำการออกแบบคลาส โดยข้อมูลที่ต้องการเก็บลงในฐานข้อมูลก็ให้กำหนดเป็นแอตทริบิวต์ของคลาสจากนั้น Caché จะทำการสร้างโค้ดภาษาจาวา ซึ่งเมื่อคอมไพล์เราจะได้คลาส ซึ่งเหมือนเป็นตัวแทนที่ใช้ในการติดต่อกับฐานข้อมูล โดยเมื่อเราการเปลี่ยนแปลงแอตทริบิวต์ใดๆของออบเจ็กต์ ก็จะมีผลกับข้อมูลในฐานข้อมูล

เราจะลองยกตัวอย่างการออกแบบคลาส Village ซึ่งจะทำการเก็บข้อมูลของหมู่บ้านว่ามีชื่ออะไรและมีพื้นที่เป็นเท่าไปไหร่ ซึ่งคลาสไดอะแกรมเป็น ดังรูป

รูปภาพ 4-17 คลาส Village

โปรแกรม Rational Rose จะมีเครื่องมือที่ใช้ในการส่งออกคลาสไดอะแกรมเป็นคลาสที่ใช้ในการเก็บข้อมูลของฐานข้อมูล Caché ให้เราทำการคอมไพล์คลาส Village และให้ Caché ทำการสร้างโค้ดจาวาดังรูปข้างล่างซึ่งเมื่อคอมไพล์แล้วจะได้คลาส Village.class ที่ใช้ในการติดต่อฐานข้อมูลใน Caché

```
    |----+----1----+----2----+----3----+----4----+----5----+----6----+
117     /**
118      * Get Accessor method for Area
119     **/
120     public double getArea() throws CacheException {
121         return m_Area.get();
122     }
123     /**
124      * Set Accessor method for Area
125     **/
126     public void setArea(double value) throws CacheException {
127         m_Area.set( value );
128     }
129     /**
130      * Get Accessor method for Name
131     **/
132     public String getName() throws CacheException {
133         return m_Name.get();
134     }
135     /**
136      * Set Accessor method for Name
137     **/
138     public void setName(String value) throws CacheException {
139         m_Name.set( value );
140     }
141
```

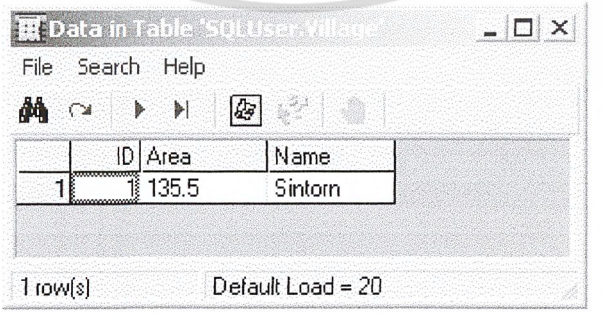
รูปภาพ 4-18 แสดงโค้ด Village.java ที่ Caché ได้สร้างขึ้น

เราจะเห็นว่าในคลาสข้างต้นจะมีเมถอด getName,setName,getArea,setArea ซึ่งเป็นเมทธอดที่ใช้อ่านค่า และกำหนดค่า Name และ Area ตามลำดับ ดังนั้นเราสามารถสร้างโปรแกรมที่สามารถแก้ไขหรือเพิ่มข้อมูล Village ในฐานข้อมูล ได้โดยใช้โค้ดตัวอย่างดังต่อไปนี้

```
    ├---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8
  1 import COM.intersys.objects.*;
  2 //import  Village
  3 public class VillageTest
  4 {
  5     public static void main( String[] args )
  6     {
  7         ObjectFactory  factory = null;
  8         Village  village = null;
  9         try
 10         {
 11             factory = new ObjectFactory( "cn_iptcp:127.0.0.1[1972]:SAMPLES" );
 12             village = new Village(factory);
 13             village.setName("Sintorn Village");
 14             village.setArea(135.5);
 15             village._save();
 16             village._close();
 17             factory.close();
 18         }
 19         catch (Exception ex)
 20         {
 21             System.out.println( "Caught exception: " + ex.getClass().getName()
 22                 + ": "+ex.getMessage() );
 23         }
 24     }
 25 }
```

รูปภาพ 4-19 VillageTest.java ซึ่งใช้ทดสอบเมทธอดของคลาส Village

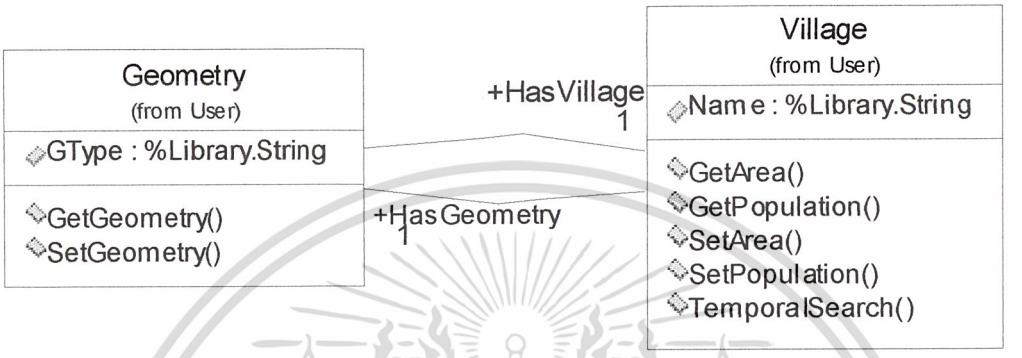
จากโค้ดตัวอย่างข้างต้นเป็นการเก็บข้อมูลของหมู่บ้านใหม่ที่มีชื่อว่า Sintorn Village และมีพื้นที่ 135.5 ได้โดยทำการสร้างออบเจ็กต์ village จากคลาส Village ที่ได้จากการคอมไพล์ไฟล์ Vilage.java ซึ่งได้มาจาก Caché และเราสามารถแก้ไขค่า Name และ Area ได้โดยผ่านเมทธอด setName() และ setArea() ตามลำดับ หลังจากนั้นเมื่อทำการเรียก village._save() ก็จะมีการทำการเก็บค่า Name และ Area ลงตาราง Village ในฐานข้อมูล Caché ดังรูปข้างล่าง



รูปภาพ 4-20 ข้อมูลในตาราง Village ซึ่งใช้เก็บข้อมูลของออบเจ็กต์ Village

4.8 การเชื่อมต่อข้อมูลเชิงพื้นที่กับเชิงบรรยาย

ในโครงงานนี้เราได้ทำการเชื่อมต่อข้อมูลเชิงบรรยายกับข้อมูลเชิงพื้นที่เข้าไว้ด้วยกันเพื่อให้แอพพลิเคชั่นที่สร้างขึ้นมาสามารถสอบถามข้อมูลเชิงบรรยายที่ต้องการขึ้นมาแล้วได้ข้อมูลเชิงพื้นที่ขึ้นมาแสดงผลด้วยทันที โดยข้อมูลเชิงบรรยายหนึ่งตัวจะมีข้อมูลเชิงพื้นที่หนึ่งตัวเช่นกัน ดังคลาสไดอะแกรมข้างล่างเป็นการเชื่อมต่อข้อมูลเชิงบรรยายคลาส Village ข้อมูลเชิงพื้นที่ Geometry



รูปภาพ 4-21 การเชื่อมต่อข้อมูลเชิงบรรยายและเชิงพื้นที่

คลาสไดอะแกรมข้างต้นเมื่อจัดเก็บเป็นคลาสใน Caché ในคลาส Village จะมีแอตทริบิวต์ HasGeometry ซึ่งจะทำการเก็บเป็นค่า OID ของออบเจ็กต์ Geometry ที่แสดงข้อมูลเชิงพื้นที่ของออบเจ็กต์ Village ตัวนี้ โดยการอ้างถึงออบเจ็กต์ Geometry ที่เชื่อมต่อกับออบเจ็กต์ Village สามารถทำได้โดยการใช้คำสั่งดังต่อไปนี้

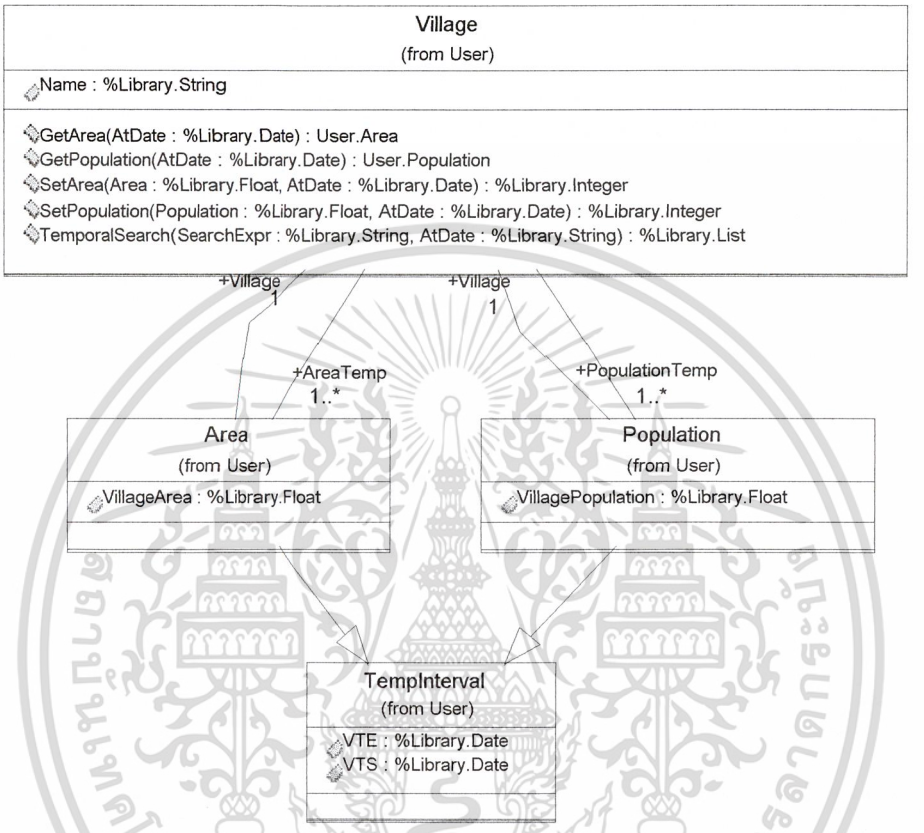
```
geomety = village.getHasGeometry();
```

คำสั่งข้างต้นเราจะได้ออบเจ็กต์ geometry ที่แสดงถึงข้อมูลเชิงพื้นที่ของข้อมูลเชิงบรรยายที่เก็บอยู่ในออบเจ็กต์ village ส่วนกรณีที่เราต้องการเชื่อมต่อข้อมูลเชิงบรรยายเข้ากับข้อมูลเชิงพื้นที่สามารถทำได้โดยใช้คำสั่งดังต่อไปนี้

```
village.setHasGeometry(geometry);
```

4.9 การจัดการกับแอตทริบิวต์ที่สามารถเปลี่ยนแปลงตามเวลาได้

โครงงานนี้ ในส่วนของข้อมูลเชิงพื้นที่และเชิงบรรยายเราได้มีการจัดเก็บทั้งที่เป็นแบบเชิงเวลาและไม่เป็นเชิงเวลา โดยหัวข้อนี้เราจะกล่าวถึงการจัดการเรื่องข้อมูลที่เป็นเชิงเวลาว่ามีวิธีอย่างไร โดยยกตัวอย่างเป็นคลาส Village ดังรูปข้างล่าง



รูปภาพ 4-22คลาส Village

จากรูปข้างต้นคลาส Villageจะทำการจัดเก็บแอตทริบิวต์ Name แบบไม่เป็นเชิงเวลา ซึ่งส่วนนี้เมื่อคอมไพล์คลาส Village Caché จะทำการสร้างเมทธอด getName(),setName() ที่ใช้ในการจัดการกับแอตทริบิวต์เหล่านี้ให้โดยอัตโนมัติ แต่สำหรับแอตทริบิวต์ที่เราต้องการจัดเก็บ และค้นหาเป็นเชิงเวลา เช่น area,population นั้น Caché ไม่สามารถทำการสร้างเมทธอดที่จัดการแอตทริบิวต์ที่เป็นเชิงเวลาได้ ซึ่งเราจะต้องทำการสร้างเมทธอดต่างๆเหล่านี้ ซึ่งเราสามารถสร้างเมทธอดต่างๆเหล่านี้ได้โดยใช้ภาษา Caché Object Script โดยในโครงงานนี้เราได้ทำการสร้างเมทธอดพื้นฐานต่างๆที่ใช้ในการจัดการแอตทริบิวต์ที่เป็นเชิงเวลาดังต่อไปนี้

- **GetAttributeName(AtDate)** เป็นเมทธอดที่ใช้ในการอ่านค่าแอตทริบิวต์ที่เป็นเชิงเวลาโดยต้องกำหนดเวลาเป็นพารามิเตอร์ AtDate ซึ่งเป็นเวลาที่ค่าในแอตทริบิวต์นั้นเป็นจริงด้วย
- **SetAttributeName(Value,AtDate)** เป็นเมทธอดที่ใช้ในการกำหนดค่าแอตทริบิวต์ที่เป็นเชิงเวลาโดยต้องกำหนดเวลาเป็นพารามิเตอร์ AtDate ซึ่งเป็นเวลาล่าสุดที่ค่าแอตทริบิวต์ที่กำหนดนั้นเป็นจริง

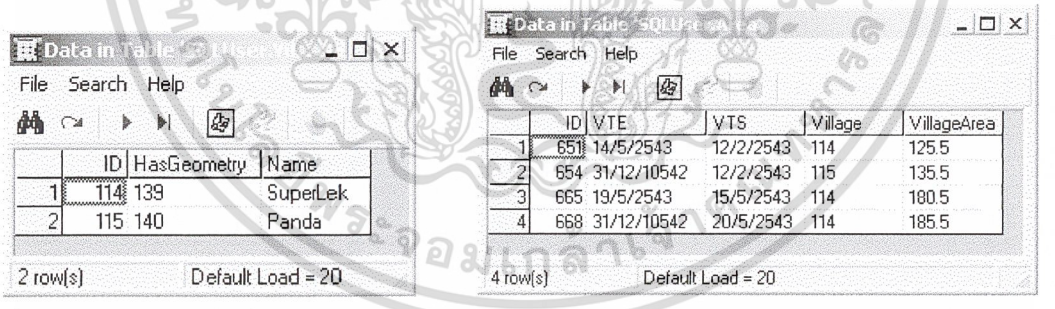
- **TemporalSearch(SearchExpr,AtDate)** เป็นเมทธอดที่ใช้ในการค้นหาแอตทริบิวต์ที่เป็นเชิงเวลา โดยต้องกำหนดเวลาเป็นพารามิเตอร์ AtDate ซึ่งเป็นเวลาที่ค่าในแอตทริบิวต์ที่ค้นหาได้นั้นเป็นจริงด้วย

ในหัวข้อต่อไปเราจะกล่าวถึงการสร้างเมทธอดพื้นฐานที่ใช้ในการจัดการกับข้อมูลที่เป็นเชิงเวลาเหล่านี้ โดยใช้ภาษา Caché Object Script ซึ่งในเมทธอดต่างๆเหล่านี้ได้มีการแทรกคำสั่ง SQL ที่ใช้ในการค้นหาข้อมูลเป็นเชิงเวลาลงไปด้วย

4.10 การอ่านค่าแอตทริบิวต์ที่สามารถเปลี่ยนแปลงตามเวลาได้

ในส่วนของการออกแบบเราได้ทำการออกแบบแอตทริบิวต์ให้เป็นเชิงเวลาได้โดยทำการแยกออกมาเป็นคลาสใหม่และทำการใส่เวลา ValidTimeStart และ ValidTimeStop ลงไปด้วย ดังรูป 20 แอตทริบิวต์ Area และ Population เราต้องการจัดเก็บเป็นเชิงเวลาดังนั้นจึงทำการแยกออกมาเป็นคลาส Area และ Population และทำการสืบทอดเวลา ValidTimeStart และ ValidTimeStop มาจากคลาส TempInterval

ยกตัวอย่างคลาส Village ข้างต้นเมื่อเก็บอยู่ใน Caché จะทำการเก็บคลาส Area,Population เป็นสมาชิกของอาเรย์ในแอตทริบิวต์ของคลาส Village ซึ่งมีชื่อเป็น AreaTemp,PopulationTemp ตามลำดับ ซึ่งหมายความว่า ในคลาส Village จะสามารถมีออบเจ็กต์ Area และ Population ได้หลายๆตัวตามช่วงเวลาที่ข้อมูล Area และ Population เป็นจริงเมื่อเราลองดูตารางที่ใช้เก็บข้อมูลคลาส Village และ Area ที่เราได้ทำการออกแบบไว้ตามรูป ความสัมพันธ์จะเป็นดังรูปข้างล่าง



รูปภาพ 4-23 ตาราง Village และ Area

จากรูปข้างต้นเป็นรูปของตารางที่ใช้ในการเก็บข้อมูลของคลาส Village และ Area ซึ่งจะเห็นได้ว่าออบเจ็กต์ Village ที่มี ID เป็น 114 จะมีข้อมูลพื้นที่ของหมู่บ้าน (VillageArea) อยู่ 3 ช่วงเวลาคือ 12/2/2543 ถึง 14/2/2543 , 15/5/2543 ถึง 19/5/2543 และ 20/5/2543 ถึง 31/12/10542 ดังนั้นเมทธอด **GetAttributeName(AtDate)** สำหรับข้อมูล Area จะใช้เป็น **GetArea(AtDate)** ซึ่งเป็นเมทธอดของคลาส Village นั้นจะต้องมีคุณสมบัติดังต่อไปนี้

- เนื่องจากตาราง Area นั้นได้เก็บค่าออบเจ็กต์ของ Area ที่เป็นของหลายๆ Village เช่นจากตาราง

Area ดังรูปข้างต้น มีออบเจ็กต์ Area ที่เป็นของ Village ที่มี ID เป็น 113 และ 114 ดังนั้นออบเจ็กต์ Area ที่ค้นหาได้จากตาราง Area นั้น โดยใช้เมทธอดนี้ต้องเป็น Area ที่เป็นของออบเจ็กต์ Village ที่ทำการเรียกเมทธอดตัวนี้

- ต้องค้นหาออบเจ็กต์ Area จากตาราง Area ที่พารามิเตอร์ AtDate นั้นอยู่ในช่วง ValidTimeStart และ ValidTimeStop ซึ่งหมายความว่า เมทธอดนี้ใช้ค้นหาออบเจ็กต์ Area ณ เวลาที่ต้องการ (AtDate) ได้

ยกตัวอย่างการอ่านค่าเช่นให้ออบเจ็กต์ Village ที่มี ID เป็น 114 GetArea(14/2/2543) จะต้องได้ออบเจ็กต์ Area ที่มี ID เป็น 651 เป็นต้น สำหรับการสร้างเมทธอดที่ใช้ในการอ่านค่าแอตทริบิวต์ที่เป็นเชิงเวลา ในที่ขอยกตัวอย่างแอตทริบิวต์ Area ซึ่งใช้เมทธอด GetArea(AtDate) โดยใช้ Caché Object Script สามารถทำได้ดังรูปข้างล่าง

```
NEW id,VillageID,ListID,area
// หา ID ของออบเจ็กต์ Village ที่ทำการเรียกเมตถอดนี้เก็บค่าไว้ในตัวแปร VillageID

Set ListID = ##this.%Oid()
Set VillageID = $LIst(ListID,1)

// หา ID ของออบเจ็กต์ Area ซึ่งเป็นของ Village ที่ทำการเรียกเมตถอดนี้และค่า AtDate อยู่ในช่วง VTS และ VTE

&sql(SELECT ID into :id FROM SQLUser.Area WHERE (Village = :VillageID) AND
    (CAST(:AtDate AS INTEGER) BETWEEN CAST(VTS AS INTEGER) AND CAST(VTE AS INTEGER)))

// โหลดออบเจ็กต์ Area จาก ID ที่หาได้และทำการคืนค่ากลับไปเป็นออบเจ็กต์ Area
SET area=##class(Area).%OpenId(id)
QUIT area
```

รูปภาพ 4-24 Object Script ในเมทธอด GetArea(AtDate)

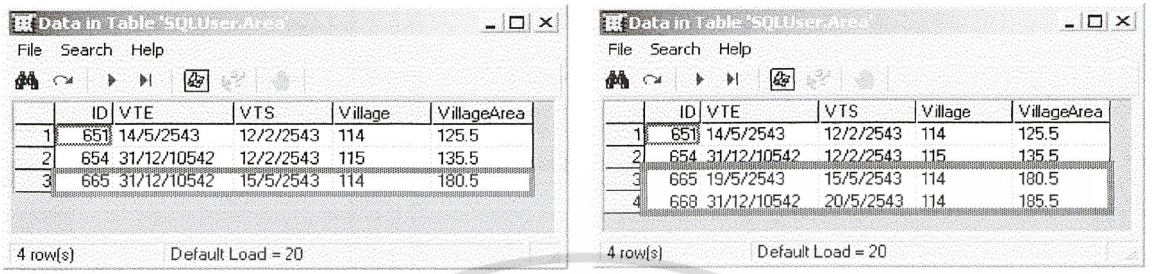
4.11 การกำหนดค่าแอตทริบิวต์ที่สามารถเปลี่ยนแปลงตามเวลาได้

สำหรับการกำหนดค่าของแอตทริบิวต์ที่เป็นเชิงเวลานั้นจะแตกต่างกับการกำหนดค่าแอตทริบิวต์ที่ไม่เป็นเชิงเวลา ตรงที่แอตทริบิวต์ที่เป็นเชิงเวลานั้นจะต้องเก็บค่าเก่าเอาหลังจากที่มีการกำหนดค่าแอตทริบิวต์นั้นใหม่อีกด้วย ตัวอย่างเช่น แอตทริบิวต์ Area ของคลาส Village ซึ่งเป็นแอตทริบิวต์เชิงเวลาซึ่งมีเมทธอด **SetArea(Value,AtDate)** นั้นจะต้องมีคุณสมบัติดังต่อไปนี้

- เนื่องจากการเก็บข้อมูลที่เป็นเชิงเวลานั้นจะต้องทำการเก็บข้อมูลหลายๆสเตท ดังนั้นเมื่อมีการกำหนดค่าแอตทริบิวต์ Area ใหม่ นั้นค่า AtDate นั้นจะต้องมีค่ามากกว่า ValidTimeStart ของข้อมูลสเตทล่าสุดซึ่งจะมี ValidTimeStop เป็นค่าอินฟินิตี้ (31/12/10542) ตัวอย่างจากรูป 21 เช่น Village ที่มี ID เป็น 114 นั้นถ้าต้องการกำหนดค่า Area ใหม่จะต้องส่งพารามิเตอร์ AtDate จะต้องมีค่ามากกว่า 20/5/2543 ดังนั้นเมทธอดนี้ต้องทำการตรวจสอบตรงจุดนี้ก่อนที่จะทำการกำหนดค่า Area ด้วย

สำหรับการสร้างเมทธอดที่ใช้ในการกำหนดค่าแอตทริบิวต์ที่เป็นเชิงเวลา ในที่ขอยกตัวอย่างแอตทริบิวต์ Area ซึ่งใช้เมทธอด SetArea(Value,AtDate) โดยใช้ Caché Object Script สามารถทำได้ดังรูป

ที่ 24 ยกตัวอย่างการกำหนดค่าเช่นให้ออบเจ็กต์ Village ที่มี ID เป็น 114 สมมติว่าก่อนที่จะมีการกำหนดค่า Area โดยใช้เมทธอด SetArea(Value,AtDate) ตาราง Area มีลักษณะเป็นดังรูปที่ 23 ด้านซ้าย เมื่อมีการเรียกใช้เมทธอด SetArea(185.5,20/5/2543) จากออบเจ็กต์ Village ที่มี ID เป็น 114 เราได้ผลเป็นตาราง Area ใหม่ดังรูปที่ 23 ด้านขวา



Data in Table 'SQLUser.Area'

File Search Help

	ID	VTE	VTS	Village	VillageArea
1	651	14/5/2543	12/2/2543	114	125.5
2	654	31/12/10542	12/2/2543	115	135.5
3	665	31/12/10542	15/5/2543	114	180.5

4 row(s) Default Load = 20

Data in Table 'SQLUser.Area'

File Search Help

	ID	VTE	VTS	Village	VillageArea
1	651	14/5/2543	12/2/2543	114	125.5
2	654	31/12/10542	12/2/2543	115	135.5
3	665	19/5/2543	15/5/2543	114	180.5
4	668	31/12/10542	20/5/2543	114	185.5

4 row(s) Default Load = 20

รูปภาพ 4-25ตาราง Area ก่อนและหลังใช้เมทธอด SetArea(Value,AtDate)

```
NEW SQLCODE,VillageID,ListID,village,areaarry,area,id
// ทำการหา ID ของออบเจ็กต์ Village ที่ทำการเรียกเมดถอดนี้เก็บค่าไว้ในตัวแปร VillageID
Set ListID = ##this.%Oid()
Set VillageID = $LIst(ListID,1)

// ทำการตรวจสอบว่าพารามิเตอร์ AtDate นั้นมีค่ามากกว่า ValidTimeStart ของข้อมูลสเตทล่าสุดหรือไม่
&sql(SELECT ID FROM SQLUser.Area WHERE
     (VTS=(SELECT VTS FROM SQLUser.Area WHERE (VTE=(SELECT MAX(VTE) FROM SQLUser.Area)))
      AND (Village = :VillageID)) AND (CAST(:AtDate As Integer) > CAST(VTS As Integer)))

// ถ้าพารามิเตอร์ AtDate นั้นมีค่ามากกว่า ValidTimeStart ของข้อมูลสเตทล่าสุด
If (SQLCODE=0) {
// เปลี่ยนค่า ValidTime Stop ของข้อมูล Area สเตทล่าสุดให้มีค่าน้อยกว่า ValidTime Start ของข้อมูล Area สเตทใหม่ 1 วัน
&sql(UPDATE SQLUser.Area SET VTE=CAST(DATEADD('dd',-1,:AtDate) As DATE) WHERE
     (ID =(SELECT ID FROM SQLUser.Area WHERE VTE = (SELECT MAX(VTE) FROM SQLUser.Area)) AND
     (Village = :VillageID)))

// เพิ่มข้อมูล Area สเตทใหม่ลงในฐานข้อมูล
&sql(INSERT INTO SQLUser.Area VALUES(CAST('9999-12-31 00:00:00' As Date),
      CAST(CAST(:AtDate As Integer) As Date),:VillageID,:Area))

// ทำการเพิ่มข้อมูล Area สเตทใหม่ลงในอาเรย์ AreaTemp ของออบเจ็กต์ Village
&sql(SELECT MAX(ID) into :id FROM SQLUser.Area)
Set area=##class(Area).%OpenId(id)
Set village = ##class(Village).%OpenId(VillageID)
Set areaarry = village.AreaTempGet()
DO areaarry.SetAt(area,areaarry.Count()+1)
DO village.AreaTempSet(areaarry)
DO village.%Save()
DO areaarry.%Close()
DO village.%Close()
Quit 1
}
// ถ้าพารามิเตอร์ AtDate นั้นมีค่าน้อยกว่า ValidTimeStart ของข้อมูลสเตทล่าสุด
Else
{
Quit 0
}
```

รูปภาพ 4-26 เมทธอด SetArea(Value,AtDate) โดยใช้ Caché Object Script

4.12 การค้นหาค่าแอตทริบิวต์ที่สามารถเปลี่ยนแปลงตามเวลาได้

เมทธอดนี้จะคล้ายๆกับ การอ่านค่า แอตทริบิวต์ที่เป็นเชิงเวลา แต่เราสามารถกำหนดเงื่อนไขเองได้โดยผ่านพารามิเตอร์ SearchExpr ซึ่งเมทธอดนี้จะทำการคืนค่าลิสต์ OID ของออบเจ็กต์ที่ตรงตามเงื่อนไข

สำหรับการสร้างเมทธอดที่ใช้ในการค้นหาค่าแอตทริบิวต์ที่เป็นเชิงเวลา ในที่ขอยกตัวอย่างคลาส Village ซึ่งใช้เมทธอด TemporalSerch(SearchExpr,AtDate) โดยใช้ Caché Object Script สามารถทำได้ดังรูป

```
Set populationchk = $Find(SearchExpr,"VillagePopulation")
Set areaexpr = "("_datenum_
        " BETWEEN (CAST(Area.VTS AS INTEGER)) AND (CAST(Area.VTE AS INTEGER)))"
Set populationexpr = "("_datenum_
        " BETWEEN (CAST(Population.VTS AS INTEGER)) AND (CAST(Population.VTE AS INTEGER)))"
Set tempstr = ""
if (areachk '= 0) //found VillageArea Expr
{
   Set tempstr = areaexpr
}
if (populationchk '= 0) //found VillagePopulation Expr
{
   if (tempstr '= "")
   {
        Set tempstr = tempstr_" AND "
   }
   Set tempstr = tempstr_populationexpr
}

// ส่งคำสั่ง SQL เพื่อทำการค้นหาข้อมูลที่ต้องการ
Set sqlcmd = "SELECT DISTINCT Area.Village FROM Area,Population WHERE (Area.Village ="_
       " Population.Village) AND ("_SearchExpr_") AND ("_tempstr_")"
Set ok = dquery.Prepare(sqlcmd)
Set ok = dquery.Execute()

// อ่านค่า OID จากเรคอร์ดข้อมูลที่ค้นหาได้
Set colcount=dquery.GetColumnCount()
Set idlist = $ListBuild()
Set rowcount = 0
for
{
    quit:'dquery.Next()
    //write !
    Set rowcount=rowcount+1
    for col=1:1:colcount
    {
        Set $LIst(idlist,rowcount) = dquery.GetData(col)
    }
}
Set ok=dquery.%Close()
if rowcount>0
{
    quit idlist
}
```

รูปภาพ 4-27 เมทธอด TemporalSerch(SearchExpr,AtDate) โดยใช้ Caché Object Script

บทที่ 5
สรุปผลการทดลอง

5.1 ตัวอย่างโปรแกรมระบบแผนที่ที่ใช้ในการทดสอบคลาสไดอะแกรมต้นแบบ

5.1.1 ภาพรวมของระบบ

หลังจากทำการศึกษาและออกแบบต้นแบบฐานข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุและนำไปจัดเก็บลงในระบบจัดการฐานข้อมูลเชิงวัตถุ Caché เรียบร้อยแล้ว จึงทำการสร้างต้นแบบระบบแผนที่ขึ้นเพื่อตรวจสอบต้นแบบคลาสไดอะแกรมของฐานข้อมูลที่ได้ออกแบบไว้ว่า สามารถนำไปใช้จัดเก็บข้อมูลแผนที่เชิงเวลาทั้งที่เป็นข้อมูลเชิงพื้นที่และข้อมูลเชิงบรรยายและสามารถทำการสืบค้นเพื่อตอบปัญหาได้อย่างถูกต้องหรือไม่

แอปพลิเคชั่นต้นแบบที่จัดทำขึ้นเป็นระบบที่พัฒนาขึ้นโดยใช้ภาษาจาวาและใช้ Forte For Java เป็นเครื่องมือในการพัฒนา

5.1.2 ข้อกำหนดความต้องการของระบบ

ระบบต้นแบบจะจัดเก็บข้อมูลแผนที่เชิงเวลาทั้งข้อมูลเชิงพื้นที่และเชิงบรรยายลงในฐานข้อมูลที่สร้างจากคลาสไดอะแกรมต้นแบบที่ได้ออกแบบไว้ โดยในระบบต้นแบบจะจัดเก็บข้อมูลเชิงบรรยาย 2 ชนิด คือ ข้อมูลหมู่บ้านและข้อมูลถนน

5.1.3 ฟังก์ชันการทางานของระบบ

- ผู้ใช้สามารถนำเข้าข้อมูลแผนที่ทั้งเชิงพื้นที่และเชิงบรรยาย ผ่านทางหน้าจอสำหรับนำเข้าข้อมูลในระบบต้นแบบ ด้วยการพิมพ์ผ่านทางคีย์บอร์ดและเมาส์
- ผู้ใช้สามารถสืบค้นข้อมูลทั้งเชิงพื้นที่และเชิงข้อมูลที่ต้องการได้ โดยกำหนดวันที่ข้อมูลนั้นเป็นจริง รวมทั้งสามารถสืบค้นการข้อมูลเชิงบรรยายไปสู่ข้อมูลเชิงพื้นที่ได้ โดยกำหนดเงื่อนไขที่ต้องการเป็นประโยคเพื่อใช้ในการสืบค้น
- แสดงผลข้อมูลแผนที่เชิงพื้นที่และเชิงบรรยายที่ได้จากการค้นหา
- นำข้อมูลแผนที่ที่ได้จากการค้นหาไปแสดงบนหน้าจอสำหรับแก้ไขข้อมูล โดยผู้ใช้สามารถเลือก Geometry ที่ต้องการแก้ไขได้

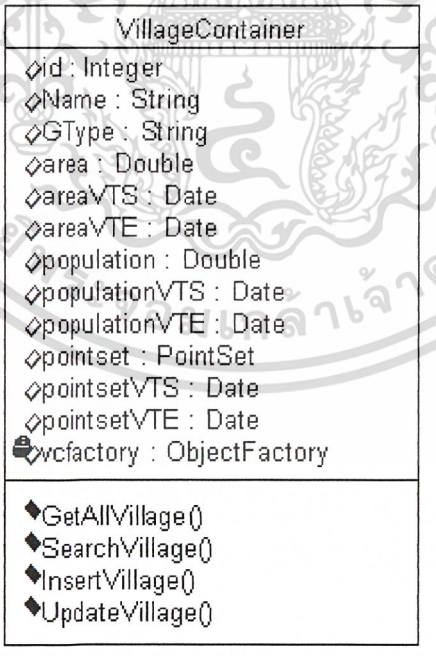
5.1.4 การสร้างระบบต้นแบบ

1) คลาส Container

ถึงแม้ว่าในโครงงานนี้เราจะเลือกใช้วิธีติดต่อกับฐานข้อมูล Caché โดยใช้ Object Access โดย Java ซึ่งมีความง่ายในการสร้างแอพพลิเคชั่นในระดับหนึ่งแล้ว แต่ในการที่จะต้องทำการ เพิ่ม,แก้ไข,ค้นหา ข้อมูลที่เป็นเชิงบรรยายแล้วสามารถทำการเชื่อมต่อกับข้อมูลเชิงพื้นที่ได้ทันทีนั้นยังต้องเขียนโค้ดโปรแกรมภาษา Java ขึ้นมาจัดการหลายบรรทัด

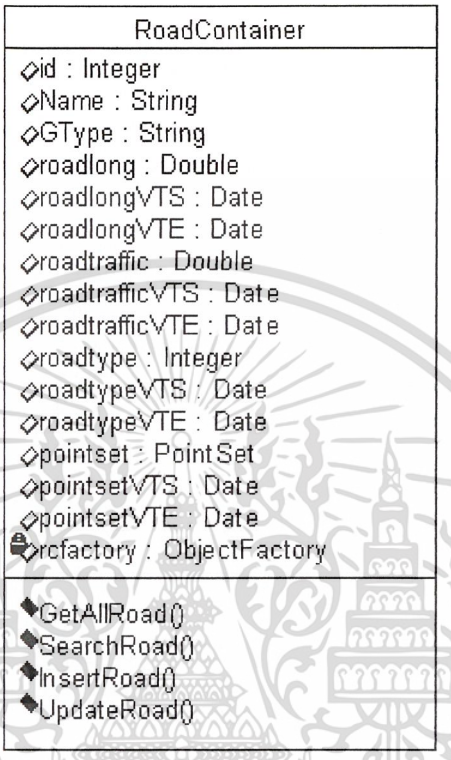
ดังนั้นในโครงงานนี้จึงได้อาศัยข้อดีของหลักการ Encapsulation ของ Object-Oriented Programming ทำการสร้างคลาส Container ที่บรรจุทั้งข้อมูลเชิงบรรยายและข้อมูลเชิงพื้นที่ที่เป็นเชิงเวลาเอาไว้ด้วยกัน อีกทั้งยังได้สร้างเมทธอดพื้นฐานที่ใช้ในการ เพิ่ม,แก้ไข,ค้นหา เอาไว้อีกด้วยโดยครอบส่วนเมทธอดพื้นฐาน GetAttributeName (Atdate), SetAttributeName (Value,AtDate), TemporalSearch (SearchExpr,AtDate) เพื่อให้ใช้งานได้ง่ายขึ้นอีกด้วย โดยคลาส Container จะมีอยู่ทั้งหมดสองคลาสด้วยกันคือ

- **VillageContainer** เป็นคลาสที่ทำการรวมข้อมูลเชิงบรรยายของคลาส Village กับข้อมูลเชิงพื้นที่ที่เป็นเชิงเวลาเข้าไว้ด้วยกัน และมีเมทธอดพื้นฐานในการ เพิ่ม,แก้ไข,ค้นหาข้อมูล โดยลักษณะของคลาสไดอะแกรมเป็นดังรูปข้างล่าง



รูปภาพ 5-1 คลาสไดอะแกรมของคลาส VillageContainer

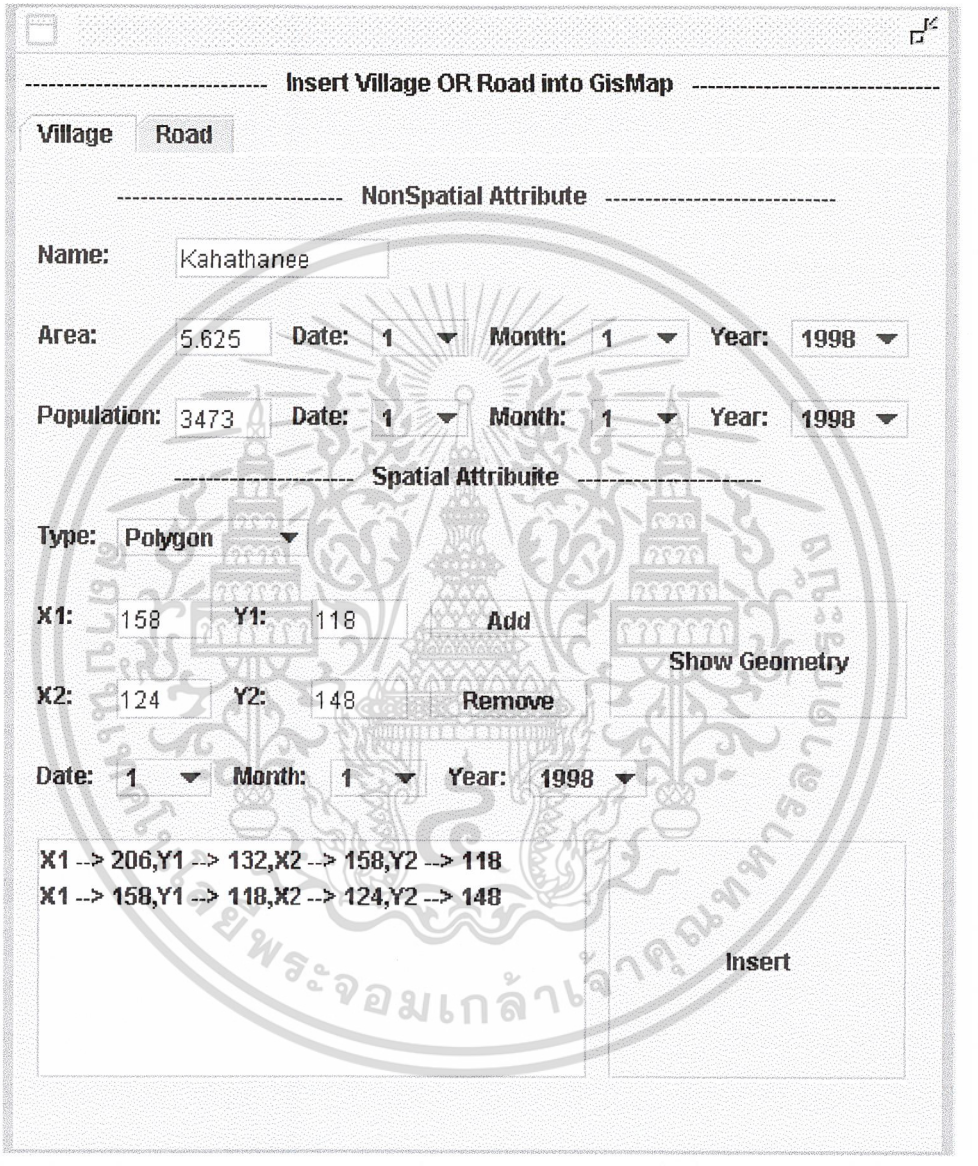
- **RoadContainer** เป็นคลาสที่ทำการรวมข้อมูลเชิงบรรยายของคลาส Road กับข้อมูลเชิงพื้นที่เข้าไว้ด้วยกัน และมีเมทธอดพื้นฐานในการ เพิ่ม,แก้ไข,ค้นหาข้อมูล โดยลักษณะของคลาสไดอะแกรมเป็นดังรูปข้างล่าง



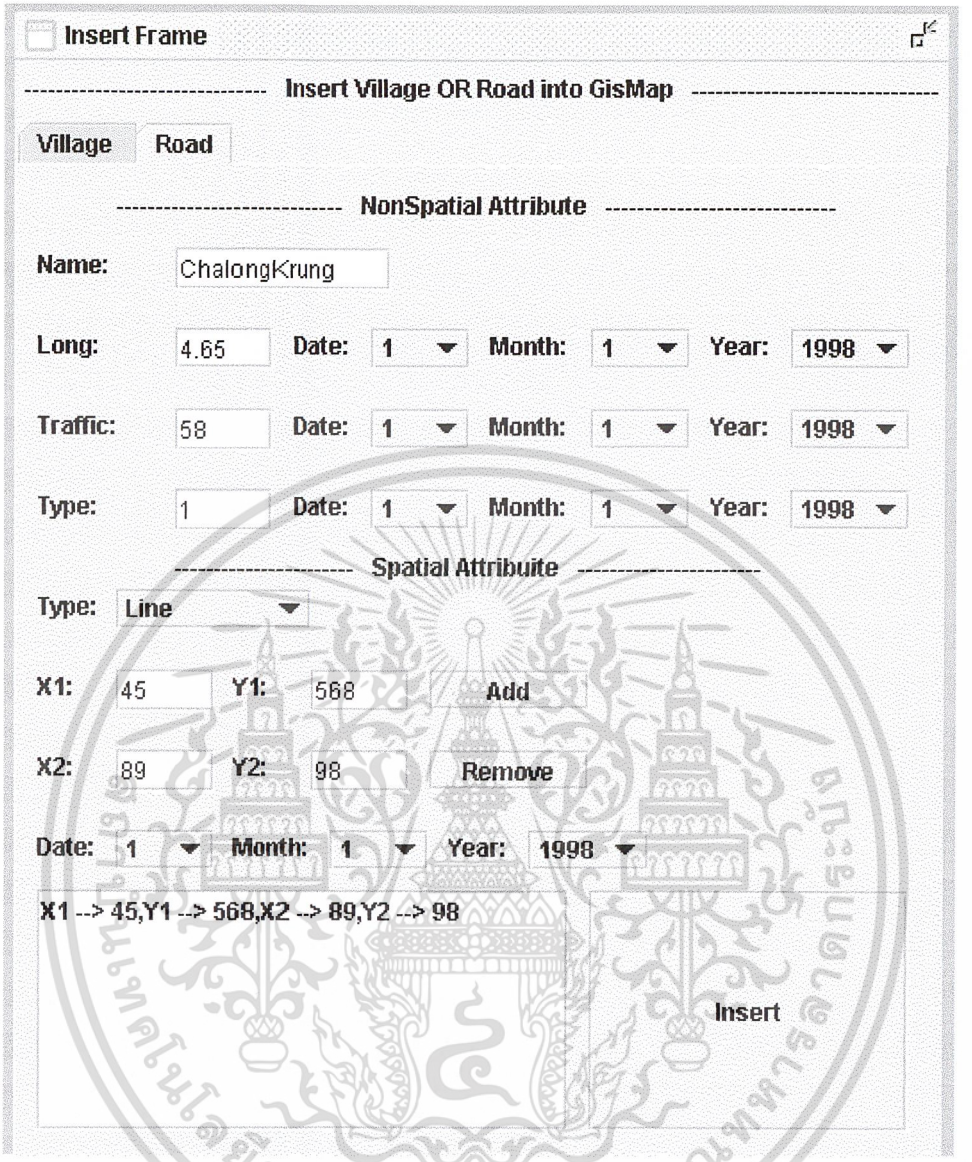
รูปภาพ 5-2 คลาสไดอะแกรมของคลาส RoadContainer

2) การนำเข้าข้อมูล

หน้าจอสำหรับนำเข้าข้อมูลหมู่บ้านและถนนลงในฐานข้อมูลโดยการกรอกข้อมูลลงในแบบฟรอมและกำหนดวันที่ข้อมูลนั้นเป็นจริง(Valid Tim Start) โปรแกรมจะเก็บข้อมูลลงในฐานข้อมูล Caché โดยจะโปรแกรมจะกำหนดวันสิ้นสุดที่ข้อมูลนั้นเป็นจริง (Valid Time Stop) ให้กับข้อมูลโดยอัตโนมัติ



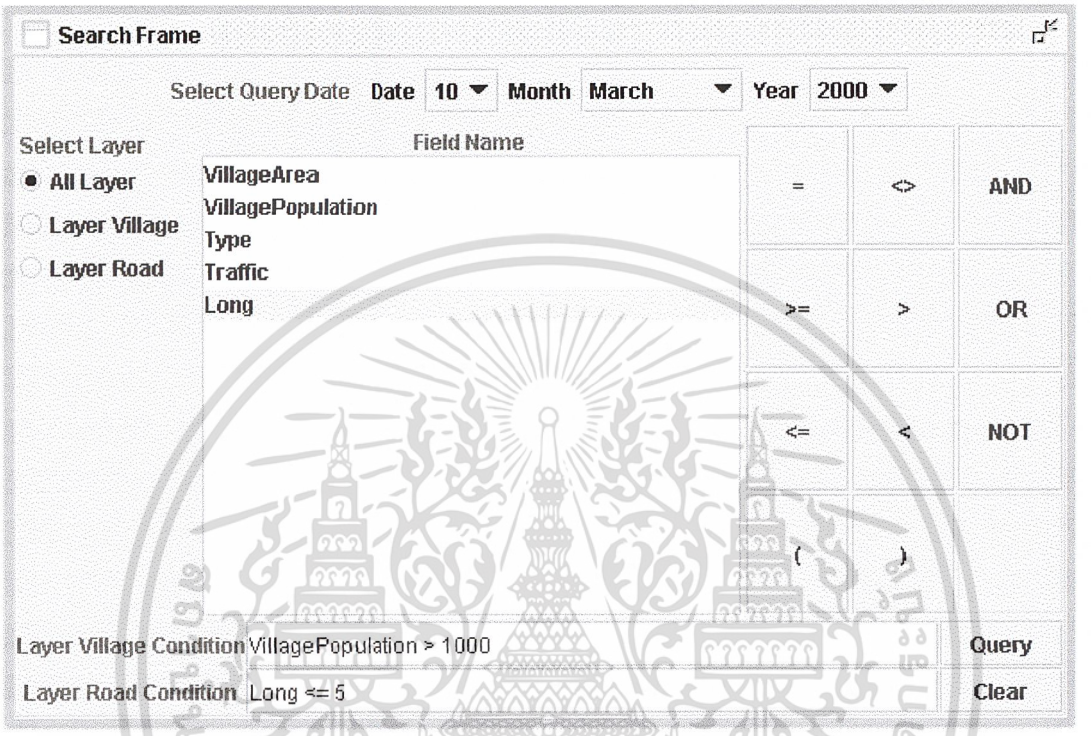
รูปภาพ 5-3 แบบฟอรมการนำเข้าข้อมูลหมู่บ้าน



รูปภาพ 5-4 แบบฟอรมการนำเข้าข้อมูลหมู่บ้าน

3) การค้นหาข้อมูล

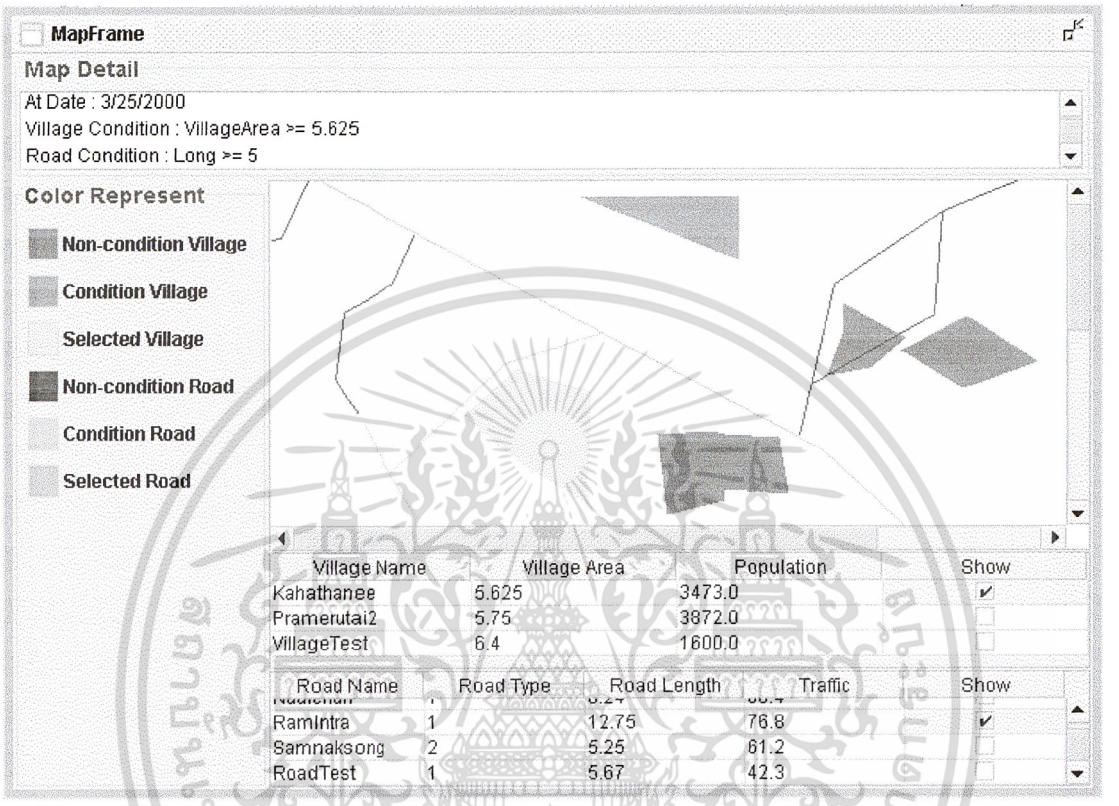
ผู้ใช้สามารถเรียกดูข้อมูลที่มีอยู่รวมทั้งค้นหาข้อมูลหมู่บ้านและถนนที่ต้องการ ในวันที่ข้อมูลนั้นเป็นจริง โดยใส่เงื่อนไขของข้อมูลได้ตามต้องการ



รูปภาพ 5-5 แบบฟอร์มสำหรับสืบค้นข้อมูลตามที่ต้องการ

4) การแสดงผล

โปรแกมจะทำการสืบค้นข้อมูลจากฐานข้อมูลและแสดงข้อมูลแผนที่ที่ได้จากการค้นหาลงในหน้าจอสำหรับแสดงข้อมูล



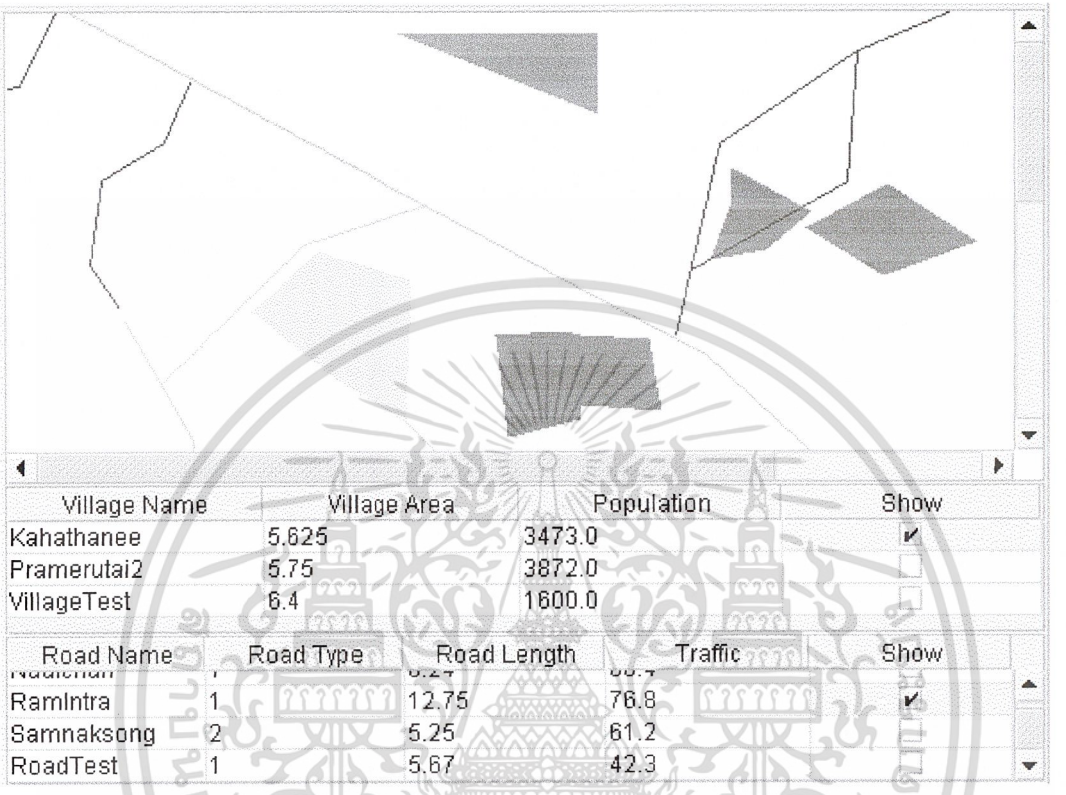
รูปภาพ 5-6 ข้อมูลภาพแผนที่ที่ได้จากการสืบค้น

บนหน้าจอแสดงผลประกอบด้วย 3 ส่วน ดังนี้

- ส่วนแรกแสดงเงื่อนไขของข้อมูลที่สืบค้นได้ ประกอบด้วยวันที่ที่ข้อมูลแผนที่นั้นเป็นจริงและเงื่อนไขข้องข้อมูลเชิงพื้นที่ที่ใช้ในการสืบค้นทั้งข้อมูลหมู่บ้านและข้อมูลถนน

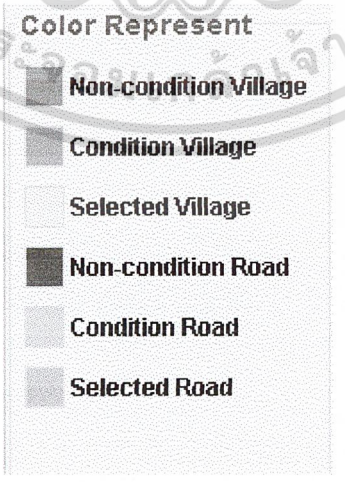
รูปภาพ 5-7 เงื่อนไปของข้อมูลที่ได้จากการสืบค้น

- ส่วนที่สองแสดงข้อมูลแผนที่ที่ได้จากการสืบค้นทั้งข้อมูลเชิงพื้นที่และข้อมูลเชิงบรรยาย โดยข้อมูลเชิงพื้นที่จะแสดงด้วยภาพและข้อมูลเชิงบรรยายจะแสดงข้อมูลหมู่บ้านแยกกับข้อมูลถนนอยู่ต่างตารางกัน รวมทั้งสามารถเลือกข้อมูลเชิงบรรยายที่สนใจเพื่อดูภาพของข้อมูลเชิงบรรยายนั้นได้



รูปภาพ 5-8 การแสดงข้อมูลแผนที่เชิงพื้นที่และเชิงบรรยายที่ได้จากการสืบค้น

- ส่วนสุดท้ายแสดงความหมายของสัญลักษณ์สีที่ใช้ในการแสดงข้อมูลเชิงพื้นที่



รูปภาพ 5-9 สัญลักษณ์สี

สีแดง : แสดงพื้นที่ของหมู่บ้านที่ไม่ตรงตามเงื่อนไขแต่เป็นจริงในวันที่กำหนด

สีม่วง : แสดงพื้นที่ของหมู่บ้านที่ตรงตามเงื่อนไขและเป็นจริงในวันที่กำหนด

สีชมพู : แสดงพื้นที่ของหมู่บ้านที่ถูกผู้ใช้เลือก

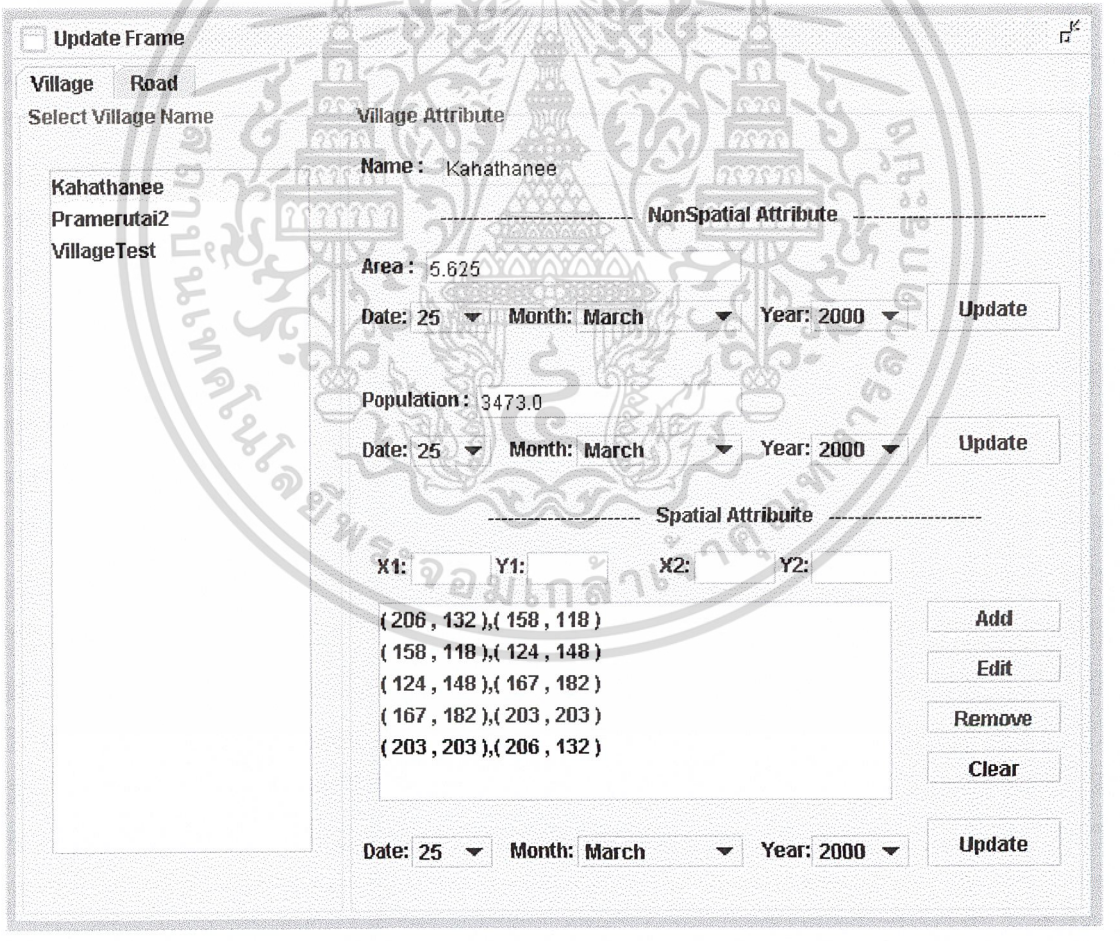
สีน้ำเงิน : แสดงรูปของถนนที่ไม่ตรงตามเงื่อนไขแต่เป็นจริงในวันที่กำหนด

สีฟ้า : แสดงรูปของถนนที่ตรงตามเงื่อนไขและเป็นจริงในวันที่กำหนด

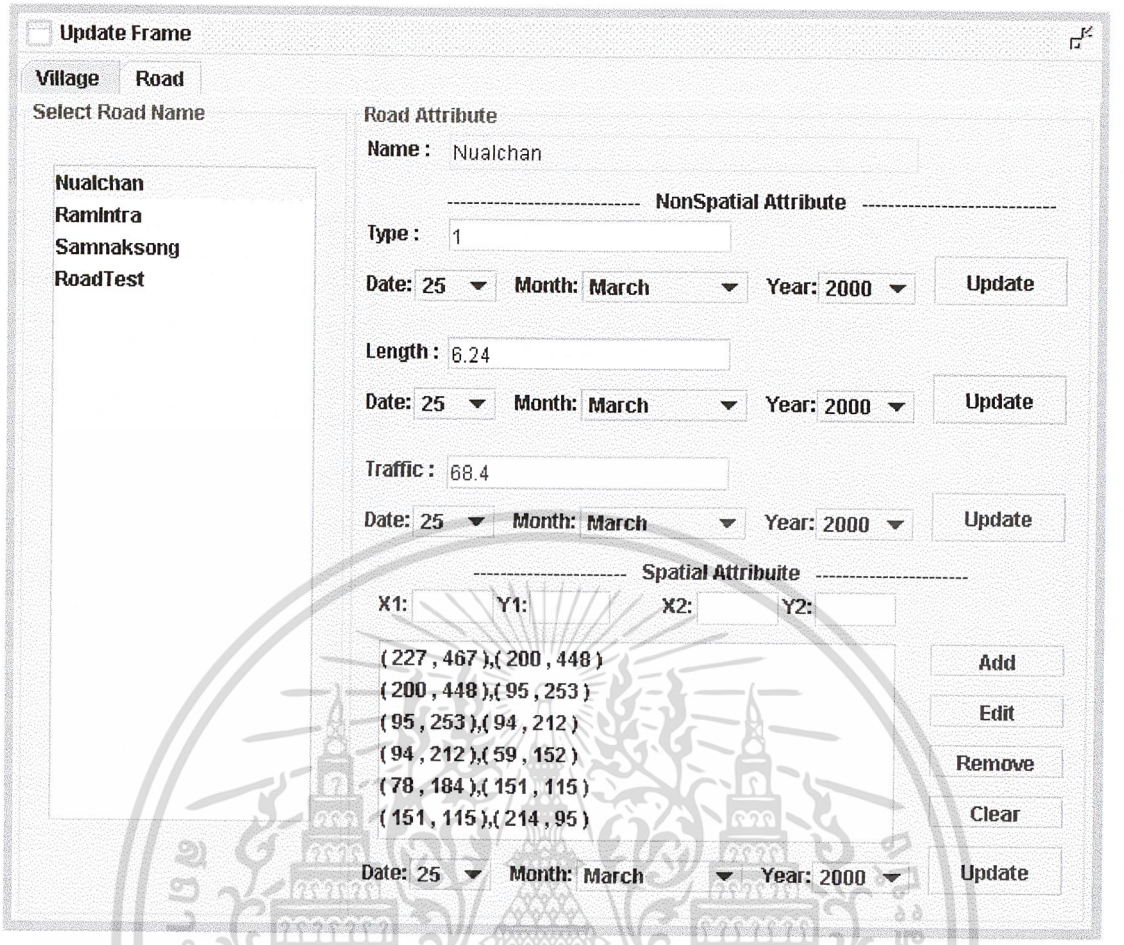
สีเขียว : แสดงรูปของถนนที่ถูกผู้ใช้เลือก

5) การแก้ไขข้อมูล

ข้อมูลที่ได้จากการค้นหาจะถูกนำมาแสดงบนแบบฟอร์มสำหรับการแก้ไขข้อมูลเพื่อให้ผู้ใช้สามารถแก้ไขข้อมูลได้ตามที่ต้องการ โดยโปรแกรมจะจัดการกำหนดเวลาสิ้นสุดของข้อมูลให้แก่ข้อมูลเดิมพร้อมทั้งนำเข้าข้อมูลใหม่



รูปภาพ 5-10 แสดงแบบฟอร์มสำหรับแก้ไขข้อมูลหมู่บ้าน



รูปภาพ 5-11 แสดงแบบฟรอมสำหรับแก้ไขข้อมูลหมู่บ้าน

5.2 สรุปผลการทำโครงการ

1) การใช้หลักการของ Object Oriented ออกแบบฐานข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงวัตถุ ช่วยแก้ปัญหาความซ้ำซ้อนของวิธีการออกแบบฐานข้อมูลแผนที่เชิงเวลาบนฐานข้อมูลเชิงสัมพันธ์ ได้ เนื่องจากฐานข้อมูลเชิงวัตถุมีคุณสมบัติดังต่อไปนี้

- มี OID เป็น External Identifier
- มีสนับสนุนชนิดข้อมูล อาเรย์,ลิสต์

2) ข้อมูลเชิงพื้นที่ที่ได้ออกแบบเอาไว้เป็นคลาส Map, Dimension, Layer, Geometry และ PointSet แต่ยังไม่มีเมทธอดสำหรับวิเคราะห์ข้อมูลเชิงพื้นที่ด้วยวิธีการทางคณิตศาสตร์ เช่น การตรวจสอบการทับกัน

3) การใช้วิธี Object Access ติดต่อกับฐานข้อมูล Caché ทำให้แอพพลิเคชั่นที่เราทำการสร้าง ขึ้นมาเพื่อติดต่อกับ Caché ทำได้ง่ายมากขึ้นกว่าวิธีอื่น เนื่องจากหลักการของออบเจ็กต์ใน เรื่อง Encapsulation ทำให้เราสามารถซ่อนรายละเอียดของออบเจ็กต์ ได้โดยรายละเอียดตรง ส่วนนี้ก็คือคำสั่ง SQL ที่ใช้ใน

การติดต่อกับฐานข้อมูล ซึ่งเราได้เขียนใส่ไว้ในเมทธอดโดยใช้ Caché Object Script ตรงจุดนี้ผู้สร้างแอพพลิเคชั่นไม่จำเป็นต้องรู้คำสั่ง SQL เพียงแต่รู้รายละเอียดของเมทธอดที่ใช้ก็เพียงพอแล้ว

4) ถึงแม้ว่าในโครงงานนี้เราได้ใช้หลักการของ Object Oriented ในออกแบบฐานข้อมูลเชิง เวลาซึ่งได้แก้ปัญหาของวิธีการแบบเชิงสัมพันธ์ได้ก็ตาม แต่ส่วนของเรื่องการจัดการกับข้อมูลที่เป็นเชิงเวลานั้น เรายังต้องผู้เขียนเมทธอดขึ้นมาจัดการเอง และวิธีการที่ใช้ในโครงงานนี้ทำให้แอตทริบิวต์ที่เป็นเชิงเวลานั้นได้แยกออกมาเป็นคลาส ถ้ายิ่งมีจำนวนมากก็จะต้องเขียนเมทธอดขึ้นมาเพื่อจัดการกับข้อมูลเชิงเวลามากขึ้นเท่านั้น ซึ่งปัญหาข้อนี้ได้เสนอวิธีแก้ไว้ในหัวข้อแนวทางการพัฒนาต่อ

5) แอพพลิเคชั่นซึ่งสร้างขึ้นมาทดสอบสิ่งได้ออกแบบเอาไว้นั้น สามารถตอบคำถามข้อมูลเชิงบรรยายแล้วทำการเชื่อมต่อไปยังข้อมูลเชิงพื้นที่แบบเชิงเวลาได้โดยอยู่กับการเขียนเมทธอดทางคลาสของข้อมูลเชิงบรรยายขึ้นมาจัดการ แต่การตอบคำถามข้อมูลเชิงพื้นที่แล้วทำการเชื่อมต่อไปยังข้อมูลเชิงบรรยายแบบเชิงเวลาได้นั้นยังติดข้อจำกันในหัวข้อที่ 2 อยู่

5.3 แนวทางการพัฒนาต่อ

1) ทำการสร้างเครื่องมือที่สามารถสร้างคลาสข้อมูลเชิงบรรยายได้โดยอัตโนมัติ ซึ่งเครื่องมือนี้จะสร้างคลาสของข้อมูลเชิงบรรยายที่ต้องการเก็บไว้ให้พร้อมใช้งานในฐานข้อมูล Caché ทันที รวมถึงยังช่วยสร้างเมทธอดที่จำเป็นสำหรับการจัดการข้อมูลเชิงเวลาและการเชื่อมต่อกับข้อมูลเชิงพื้นที่ ซึ่งตรงจุดนี้ Caché ได้มีไลบรารีไว้สนับสนุนอยู่แล้ว

2) ทำการพัฒนาข้อมูลเชิงพื้นที่ที่ได้ออกแบบไว้ให้ดีขึ้น โดยอาจจะเพิ่มเมทธอดที่ใช้จัดการกับข้อมูลเชิงพื้นที่ เช่น การตรวจสอบการทับกันของรูป เป็นต้น เพื่อให้เราสามารถสอบถามข้อมูลเชิงพื้นที่แล้วทำการเชื่อมต่อกับข้อมูลเชิงบรรยายได้ดีขึ้น

3) หาวิธีการเชื่อมต่อข้อมูลเชิงบรรยายกับข้อมูลเชิงพื้นที่ ซึ่งข้อมูลเชิงพื้นที่ให้ใช้ซอฟท์แวร์ที่มีอยู่ในท้องตลาด เช่น ArcInfo,ArcView,Autocad Map เป็นต้น เนื่องจากซอฟท์แวร์เหล่านี้มีประสิทธิภาพในการจัดการข้อมูลเชิงพื้นที่ได้มีประสิทธิภาพ

ภาคผนวก ก
ภาษา Java เบื้องต้น

1. ความหมายของ Object และการเขียนโปรแกรมด้วยภาษาจาวาเบื้องต้น

ความหมายเชิงวัตถุ

วัตถุคือสิ่งใดๆก็ตามซึ่งมีคุณลักษณะ(State)บ่งบอกถึงความเป็นตัวของมันเองในขณะนั้น และสามารถแสดงพฤติกรรม(Behavior)ของตัวเองออกมาได้ เช่นรถยนต์สีแดง มีความหมายคือวัตถุประเภทรถยนต์มีคุณลักษณะของสีเป็นสีแดง และมีพฤติกรรมที่แสดงถึงการเคลื่อนที่ หรือหยุดได้ อีกตัวอย่างเช่น สุนัขพันธ์พุดเดิลสีขาวก็บ่งบอกตัววัตถุว่ามีคุณลักษณะสีขาว และสามารถแสดงพฤติกรรมในการวิ่ง หรือเห่า(ส่งเสียง)ได้ ตัวอย่างวัตถุที่กล่าวมาถูกพิจารณาจากสิ่งที่เป็นจริงในโลก

เราสามารถมองภาพวัตถุได้ง่ายๆด้วยการพิจารณาว่า เมื่อเวลาเรียกของบางอย่างแล้วมีหน่วยเรียก เช่น รถยนต์มีหน่วยเป็นคัน ก้อนหินมีหน่วยเป็นก้อน กระดาษมีหน่วยเป็นแผ่น เด็กมีหน่วยเป็นคน นาฬิกามีหน่วยเป็นเรือน ดังนั้นสามารถมองหลายๆสิ่งหลายๆอย่างรอบตัวแล้วพิจารณาสิ่งเหล่านั้นเป็นวัตถุได้ทุกอย่าง เมื่อสังเกตุลงไปอีกอาจพบว่าวัตถุหนึ่งอย่างประกอบมาจากวัตถุหลายๆประเภท เช่น รถจักรยานประกอบขึ้นมาจาก ล้อรถ แฮนด์บังคับ เบาะนั่ง เบรก(ห้ามล้อ)รถ เป็นต้น หรือถ้าพิจารณาให้ถี่ถ้วนอีกทีจะเห็นว่าล้อรถก็ยังประกอบขึ้นมาจาก โครงล้อ(ซี่ล้อ) และยางล้อรถ

ลักษณะของภาษา

ภาษาจาวาถูกพัฒนามาจากบริษัทซัน (Sun Microsystems) ซึ่งจัดให้เป็นภาษาคอมพิวเตอร์ภาษาหนึ่งที่ใช้หลักการออกแบบตัวภาษาด้วยวิธีเชิงวัตถุ และตัวภาษาถูกใช้เป็นเครื่องมือสำหรับพัฒนาโปรแกรมด้วยแนวคิดเชิงวัตถุ โดยตัวภาษามีลักษณะพิเศษดังนี้

- Portability สามารถในการใช้งานในสภาพวะแวดล้อมที่แตกต่าง โดยไม่ต้องมีการปรับแต่ง
- Simple ความง่ายในการเขียนโปรแกรม
- Robust ความคงสภาพในการทำงาน มีโอกาสเกิดข้อผิดพลาดที่ไม่พึงประสงค์ได้น้อย
- Secure การรองรับมาตรฐานความปลอดภัยในการใช้งานรูปแบบต่างๆ
- Distributed มีความสามารถในการประมวลผลแบบกระจาย
- Object-Oriented มีหลักการของแนวคิดเชิงวัตถุ ในการสร้างโปรแกรม

รูปแบบการพัฒนา

- ซอร์สโค้ด(Source Code) คือ ประกอบด้วยคำสั่งในภาษาจาวา โดยทั่วไปต้องมีนามสกุล java ซอร์สโค้ดสามารถสร้างด้วยโปรแกรมประเภทพิมพ์ตัวอักษร (Text Editor) ใดๆก็ได้
- ไฟล์คลาส(Class File หรือ Bytecodes) คือชุดคำสั่งกลางที่ได้จากการคอมไพล์ซอร์สโค้ด ปกติไฟล์ประเภทนี้จะมีนามสกุล class
- การคอมไฟล์(Compiling) เป็นขบวนการแปลซอร์สโค้ดให้เป็นไฟล์คลาส โปรแกรมชื่อ javac เป็นโปรแกรมที่ใช้คอมไพล์ซอร์สโค้ดเพื่อให้ได้ไฟล์คลาสโดยมีรูปแบบดังนี้

javac ชื่อไฟล์ซอร์สโค้ด.java

เช่น เมื่อเขียนโปรแกรมภาษาจาวาแล้วบันทึกด้วยชื่อไฟล์ HiJavaWorld.java ให้ใช้คำสั่งดังนี้

c:>javac HiJavaWorld.java

และผลลัพธ์ที่ได้ก็คือไฟล์คลาสที่ชื่อว่า HiJavaWorld.class

- การดีบัก(Debugging) การแก้ไขซอร์สโค้ดให้ถูกต้อง ถูกใช้ในกรณีที่ซอร์สโค้ดถูกเขียนผิด หรือกรณีปรับปรุงซอร์สโค้ดให้ทำงานดีขึ้น
- การประมวลผลหรือการอินเทอร์พลีเตอร์(Interpreting) เป็นขบวนการแปลชุดคำสั่งกลางในไฟล์คลาสให้ทำงานบนเวอร์ชวลแม็ชชีน โปรแกรมชื่อ java เป็นโปรแกรมที่จะแปลชุดคำสั่งกลางในไฟล์คลาสเพื่อทำงานบนเครื่องคอมพิวเตอร์ มีรูปแบบการทำงานดังนี้

java ชื่อไฟล์คลาส

ตัวอย่างเช่น เมื่อต้องการใช้งานไฟล์คลาสที่ชื่อว่า HiJavaWorld.class ให้พิมพ์คำสั่งดังนี้

java HiJavaWorld

2. ลักษณะโครงสร้าง และรูปแบบการเขียนโปรแกรมภาษาจาวา

โครงสร้างของคลาส

โปรแกรมที่สร้างจากภาษาจาวา ต้องสามารถสร้างออปเจ็คหรือคลาสให้ได้อย่างน้อยหนึ่งตัว โดยมีรูปแบบโครงสร้างดังนี้

```
class Class_Name
{  Data_Member
Mothod_Member
}
```

class คือคีย์เวิร์ดสำหรับกำหนดคลาส

Class_Name คือชื่อคลาส

Data_Member คือดาต้าในคลาส

Method_Member คือเมธรอดในคลาส

สัญลักษณ์ { } คือขอบเขตของคลาส

กฎการตั้งชื่อ(Identify)

ใช้ตั้งชื่อคลาส ชื่อดาต้า ชื่อเมธรอด และชื่อตัวแปร

- ประกอบด้วยตัวอักษร และหรือตัวเลข โดยตัวอักษรให้ใช้ตัวอักษรภาษาอังกฤษไม่ว่าตัวเลขหรือตัวใหญ่ รวมถึงสัญลักษณ์พิเศษ _ หรือ $ เช่น age, name2, int2float, _name, Currency$ เป็นต้น
- ความยาวตัวอักษรไม่ควรเกิน 65535 ตัวอักษร
- ไม่ควรมีตัวเลขเป็นตัวแรก เช่น 101database, 2name ถือว่าไม่สามารถใช้ตั้งชื่อได้
- ตัวอักษรตัวเล็กและตัวใหญ่มีความแตกต่างกัน ดังนั้น Count, count และ CoUnT ทั้งสามตัวอ่านเหมือนกัน แต่ถือว่าเป็นคนละตัวกัน ต้องไม่ตรงกับคีย์เวิร์ดใดในภาษาจาวาดังต่อไปนี้

abstract	double	int	strictfp **	boolean
else	interface	super	break	extends
long	switch	byte	final	native
synchronized	case	finally	new	this
catch	float	package	throw	char
for	private	throws	class	goto *

2. ลักษณะโครงสร้าง และรูปแบบการเขียนโปรแกรมภาษาจาวา

โครงสร้างของคลาส

โปรแกรมที่สร้างจากภาษาจาวา ต้องสามารถสร้างออปเจ็คหรือคลาสให้ได้อย่างน้อยหนึ่งตัว โดยมีรูปแบบโครงสร้างดังนี้

```
class Class_Name
{  Data_Member
Mothod_Member
}
```

class คือคีย์เวิร์ดสำหรับกำหนดคลาส

Class_Name คือชื่อคลาส

Data_Member คือดาต้าในคลาส

Method_Member คือเมธรอดในคลาส

สัญลักษณ์ { } คือขอบเขตของคลาส

กฎการตั้งชื่อ(Identify)

ใช้ตั้งชื่อคลาส ชื่อดาต้า ชื่อเมธรอด และชื่อตัวแปร

- ประกอบด้วยตัวอักษร และหรือตัวเลข โดยตัวอักษรให้ใช้ตัวอักษรภาษาอังกฤษไม่ว่าตัวเลขหรือตัวใหญ่ รวมถึงสัญลักษณ์พิเศษ _ หรือ $ เช่น age, name2, int2float, _name, Currency$ เป็นต้น
- ความยาวตัวอักษรไม่ควรเกิน 65535 ตัวอักษร
- ไม่ควรมีตัวเลขเป็นตัวแรก เช่น 101database, 2name ถือว่าไม่สามารถใช้ตั้งชื่อได้
- ตัวอักษรตัวเล็กและตัวใหญ่มีความแตกต่างกัน ดังนั้น Count, count และ CoUnT ทั้งสามตัวอ่านเหมือนกัน แต่ถือว่าเป็นคนละตัวกัน ต้องไม่ตรงกับคีย์เวิร์ดใดในภาษาจาวาดังต่อไปนี้

abstract	double	int	strictfp **	boolean
else	interface	super	break	extends
long	switch	byte	final	native
synchronized	case	finally	new	this
catch	float	package	throw	char
for	private	throws	class	goto *

2. ลักษณะโครงสร้าง และรูปแบบการเขียนโปรแกรมภาษาจาวา

โครงสร้างของคลาส

โปรแกรมที่สร้างจากภาษาจาวา ต้องสามารถสร้างออปเจ็คหรือคลาสให้ได้อย่างน้อยหนึ่งตัว โดยมีรูปแบบโครงสร้างดังนี้

```
class Class_Name
{  Data_Member
Mothod_Member
}
```

class คือคีย์เวิร์ดสำหรับกำหนดคลาส

Class_Name คือชื่อคลาส

Data_Member คือดาต้าในคลาส

Method_Member คือเมธรอดในคลาส

สัญลักษณ์ { } คือขอบเขตของคลาส

กฎการตั้งชื่อ(Identify)

ใช้ตั้งชื่อคลาส ชื่อดาต้า ชื่อเมธรอด และชื่อตัวแปร

- ประกอบด้วยตัวอักษร และหรือตัวเลข โดยตัวอักษรให้ใช้ตัวอักษรภาษาอังกฤษไม่ว่าตัวเลขหรือตัวใหญ่ รวมถึงสัญลักษณ์พิเศษ _ หรือ $ เช่น age, name2, int2float, _name, Currency$ เป็นต้น
- ความยาวตัวอักษรไม่ควรเกิน 65535 ตัวอักษร
- ไม่ควรมีตัวเลขเป็นตัวแรก เช่น 101database, 2name ถือว่าไม่สามารถใช้ตั้งชื่อได้
- ตัวอักษรตัวเล็กและตัวใหญ่มีความแตกต่างกัน ดังนั้น Count, count และ CoUnT ทั้งสามตัวอ่านเหมือนกัน แต่ถือว่าเป็นคนละตัวกัน ต้องไม่ตรงกับคีย์เวิร์ดใดในภาษาจาวาดังต่อไปนี้

abstract	double	int	strictfp **	boolean
else	interface	super	break	extends
long	switch	byte	final	native
synchronized	case	finally	new	this
catch	float	package	throw	char
for	private	throws	class	goto *

protected	transient	const *	if	public
try	continue	implements	return	void
default	import	short	volatile	do
instanceof	static	while		

ตาราง ก-1 คีย์เวิร์ดในภาษาจาวา

* แสดงคีย์เวิร์ดที่ไม่มีใช้ใน JDK เวอร์ชัน 1.2 ขึ้นไป

** แสดงคีย์เวิร์ดที่เพิ่มเข้ามาตั้งแต่ JDK เวอร์ชัน 1.2 ขึ้นไป

ดาต้า(Data Member)

รูปแบบ

[Access_Level] [final] [static] Data_Type Data_Name

ดาต้าคือส่วนประกอบส่วนหนึ่งของคลาส ถูกกำหนดเพื่อใช้สำหรับเก็บข้อมูล เช่นคลาส Pen มีดาต้า Color ไว้เก็บข้อมูลสี

1) **Access_Level** ระดับการเข้าถึง ประกอบด้วยคีย์เวิร์ด 3 ตัวคือ public, private และ protected
   - คีย์เวิร์ด public เป็นระดับการเข้าถึงข้อมูลที่ไม่มีข้อจำกัดใดๆ
   - คีย์เวิร์ด private เป็นระดับการเข้าถึงข้อมูล สำหรับการใช้งานภายในคลาสเท่านั้น
   - คีย์เวิร์ด protected เป็นระดับการเข้าถึงข้อมูลภายในคลาส และสำหรับคลาสที่สืบทอดมา (Inherit) แต่ต้องอยู่ในแพ็กเกจ (package) เดียวกัน
   - ถ้าไม่ระบุคีย์เวิร์ด เป็นระดับการเข้าถึงข้อมูลภายในคลาส และอยู่แพ็กเกจเดียวกัน

2) **final** เป็นคีย์เวิร์ดตัวหนึ่งซึ่งใช้บอกว่าดาต้าตัวนั้นใช้สำหรับเก็บข้อมูล โดยที่ข้อมูลจะไม่สามารถทำการเปลี่ยนแปลงได้ไม่ว่าในกรณีใดๆ ปกติจะใส่คีย์เวิร์ดนี้ไว้เมื่อต้องการให้ดาต้าเก็บข้อมูลที่เป็นข้อมูลที่คงที่ (Constant) ตลอดการทำงานของโปรแกรม

3) **static** เป็นคีย์เวิร์ด สำหรับใช้บอกถึงคุณลักษณะพิเศษในการใช้งาน เมื่อมีการกำหนดให้ดาต้าใดๆนำหน้าด้วยคีย์เวิร์ด static แล้ว ดาต้าตัวนั้นจะมีคุณลักษณะดังนี้
   - ดาต้า จะถูกโหลดลงในหน่วยความจำและพร้อมที่จะถูกใช้งานในทันทีเมื่อมีการอ้างถึง ตามข้อกำหนดของระดับการเข้าถึง (Access Level)

- ดาต้า จะอยู่ในหน่วยความจำเพียงตัวเดียว ไม่ว่าคลาสจะถูกสร้างเพื่อเป็นออปเจ็คกี่ตัวก็ตาม ดังนั้นจึงสามารถใช้ดาต้าเป็นที่เก็บข้อมูลรวมของกลุ่มคลาสเดียวกันได

```
class SimpleClass
{
    int data1;
}
```

เมธรอด(Method Member)

รูปแบบ

```
[Access_Level] [final] [static] Return_Type Method_Name ( Argument_List )
{   Statement    }
```

- Argument_List คือช่องทางสำหรับการผ่านข้อมูลเพื่อส่งให้กับเมธรอดใช้ในการทำงาน
- Statement คือคำสั่ง คีย์เวิร์ดควบคุมการทำงาน เอ็กเพลสชั่นใดๆ เพื่อกำหนดหลักการและวิธีการประมวลผลภายในเมธรอด
- สัญลักษณ์ { และ } เป็นเครื่องหมายบ่งบองขอบเขตของเมธรอด

```
class SimpleClass
{  void method1( )
    {        }
}
```

3. Swing

Swing คือชุดคอมโพเน็นต์ในการติดต่อกับผู้ใช้ โดยจัดเป็นคอมโพเน็นต์จำพวก Lightweight ที่ทำให้รูปร่างการแสดงผลบนจอภาพมีความเหมือนกันไม่ว่าจะทำงานในแพลตฟอร์มที่แตกต่างกัน สวิงซ์ใช้สำหรับสร้างส่วนติดต่อกับผู้ใช้ในรูปแบบของฟอร์ม (Form-Based) รูปแบบของฟอร์มนี้เป็นพื้นฐานของเครื่องมือสร้างโปรแกรมประเภทวีชวล (Visual Tools) ต่างๆ

แพ็กเกจของสวิงซ์

- javax.swing คือชุดแพ็กเกจซึ่งถือเป็น Root ของทุกแพ็กเกจ
- javax.swing.border คือแพ็กเกจซึ่งรวบรวมคลาส ที่เกี่ยวข้องกับการจัดการลักษณะขอบของส่วนชิ้นคอมโพเน็นต์
- javax.swing.colorchooser คือแพ็กเกจสำหรับการเลือกสี
- javax.swing.event คือแพ็กเกจที่บรรจุคลาส ที่ถูกใช้ในส่วนของการระบุอีเว็นต์(Events) ที่เกิดขึ้น
- javax.swing.filechooser คือแพ็กเกจที่สนับสนุนส่วนติดต่อกับผู้ใช้ในการใช้งานระบบไฟล์
- javax.swing.plaf คือแพ็กเกจที่สนับสนุนการแสดงคอมโพเน็นต์แบบ Plugable Look-and-Feel (PLAF)
- javax.swing.table คือแพ็กเกจที่บรรจุคลาส ที่สามารถแสดงและจัดการข้อมูลในรูปแบบตาราง
- javax.swing.text คือแพ็กเกจที่บรรจุคลาสที่สนับสนุนการจัดการเอกสาร (Document)
- java.swing.tree คือแพ็กเกจที่รวบรวมคลาสที่สนับสนุนการแสดงข้อมูลในลักษณะทรี (Tree)
- javax.swing.undo คือแพ็กเกจที่รวบรวมคลาสที่สนับสนุนรูปแบบกิจกรรมการใช้งานของผู้ใช้ในแบบ Undo และ Redo

ทำความเข้าใจกับการใช้สวิงซ์

สวิงซ์ประกอบด้วยคลาสหลากหลายคลาส ที่จะนำมาประกอบกันขึ้นเพื่อให้เกิดเป็นส่วนติดต่อกับผู้ใช้ ดังนั้นจึงควรทำความเข้าใจก่อนว่าคลาสอะไร เป็นส่วนไหนของหน้าจอโปรแกรม

1) คอนเทนเนอร์หลัก (Top-Level Containner) คือส่วนควบคุมชั้นบนสุด ซึ่งใช้ควบคุมขอบเขตหรือกำหนดขนาดในการใช้งาน เช่นความกว้าง และความยาม นอกจากนั้นแล้วยังสามารถสั่งให้แสดงหรือไม่แสดงตัวโปรแกรมนั่นเอง คลาสในกลุ่มนี้ประกอบไปด้วย JFrame, JWindow, JDialog, JInternalFrame สำหรับสร้างจาวาแอปพลิเคชั่น และ JApplet สำหรับสร้างจาวาแอปเพล็ต

2) เมนูบาร์ (Menu Bar) คือส่วนประกอบย่อยที่บรรจุอยู่ภายในคอนเทนเนอร์หลัก บริเวณส่วนบน ซึ่งเป็นพื้นที่สำหรับสร้างเมนูบาร์ของโปรแกรม โดยเมนูบาร์สามารถสร้างได้จากคลาส JMenuBar, JMenu, JMenuItem, JSeperator, JCheckBoxMenuItem, JRadioButtonMenuItem หลังจากที่สร้างเมนูจาก

คลาสดังกล่าวแล้ว จึงนำเมนูไปประกอบในคอนเทนเนอร์หลัก คอนเทนเนอร์หลักจะจัดเมนูให้อยู่บริเวณส่วนบนของพื้นที่โปรแกรม

3) คอนเทนเนอร์พื้นที่ใช้งาน (Content Pane) คือส่วนประกอบย่อยที่บรรจุอยู่ภายในคอนเทนเนอร์หลัก เป็นพื้นที่แสดงคอมโพนเน็นต์ต่างๆเพื่อให้ผู้ใช้ได้ใช้งานโปรแกรม ดังนั้นพื้นที่ส่วนนี้จึงเกี่ยวข้องกับการจัดวางตำแหน่งคอมโพนเน็นต์ต่างๆในตำแหน่งที่เหมาะสม คลาสที่จัดอยู่ในส่วนพื้นที่ใช้งานคือ JPanel, JScrollPane, JSplitPane, JTabbedPane, JEditorPane (ส่วนใหญ่ลงท้ายด้วย Pane) ส่วนคลาสที่ใช้จัดตำแหน่งคอมโพนเน็นต์ต่างๆ คือ FlowLayout, BorderLayout, GridLayout, CardLayout (ส่วนใหญ่ลงท้ายด้วย Layout)

4) คอมโพนเน็นต์ (Swing Component) เป็นชิ้นส่วนที่วางไว้ในคอนเทนเนอร์พื้นที่ใช้งาน ซึ่งมีคลาสอยู่หลายคลาสใช้งาน การนำมาใช้งานก็ขึ้นอยู่กับรูปแบบ ที่ผู้เขียนโปรแกรมต้องการ เช่น JLabel, JTextField, JTextArea, JRadioButton, JButton, JToggleButton, JCheckBox และอื่นๆอีกมาก

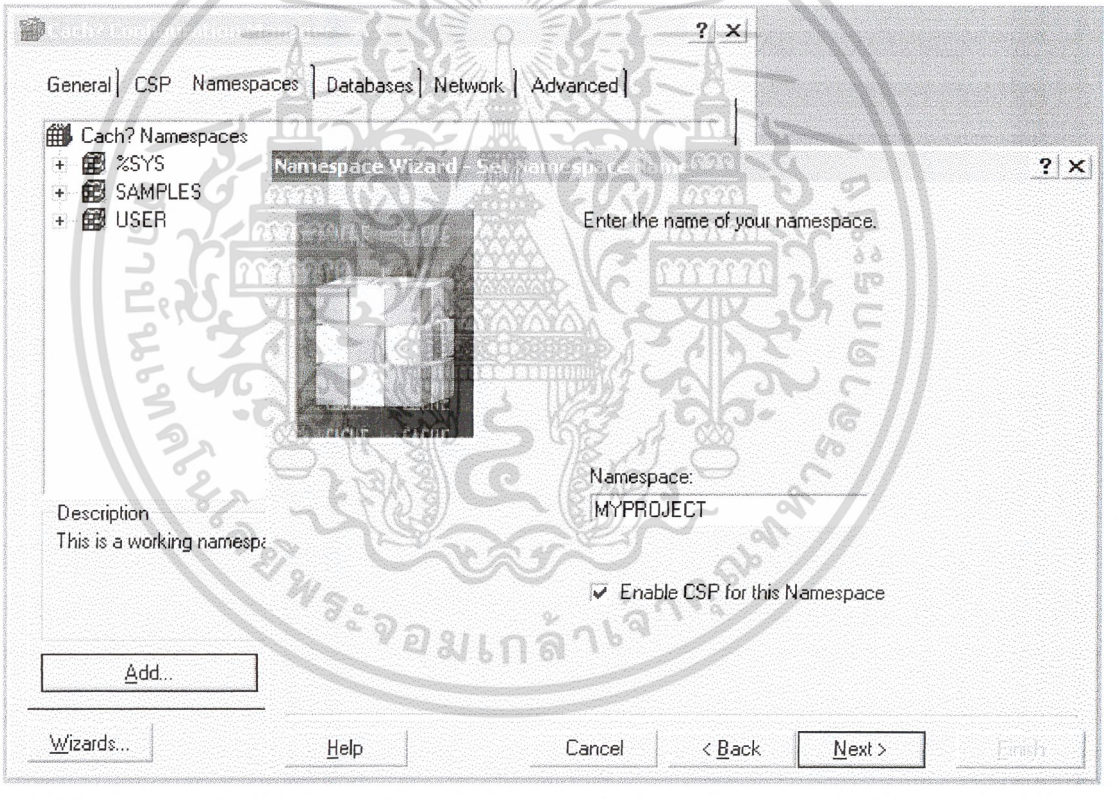
5) การดักฟังอีเว็นต์ (Event Listener) เป็นรูปแบบการรองรับการใช้งานของผู้ใช้งาน เนื่องจากการใช้งานของผู้ใช้ทำให้เกิดอีเว็นต์ขึ้นในตัวโปรแกรม ไม่ว่าจะเป็นการ กดคีย์บอร์ด การเลื่อนเมาส์ เป็นต้น การดักฟังจะวิเคราะห์ว่าอีเว็นต์ที่เกิดขึ้น เกิดขึ้นกับอะไรบนตัวโปรแกรม และควรจะให้ชุดคำสั่งไหนทำงานเพื่อรองรับอีเว็นต์นั้นๆ การดักฟังสามารถดักฟังอีเว็นต์ที่เกิดขึ้นกับคอนเทนเนอร์หลัก เช่นการย่อ/ขยาย การปิดโปรแกรม สามารถ ใช้ดักฟังเมนูบาร์คือการเลือกใช้งานเมนู สามารถใช้ดักฟังพื้นที่ใช้งาน เช่นการกดเมาส์ หรือย้ายเมาส์ สามามารถดักฟังคอมโพนเน็นต์ ใช้งากรกดคีย์บอร์ด การกดเมาส์เป็นต้น

ภาคผนวก ข
การติดตั้งระบบแผนที่บนฐานข้อมูล Caché

1. การจัดการฐานข้อมูล Caché เพื่อใช้งานระบบแผนที่

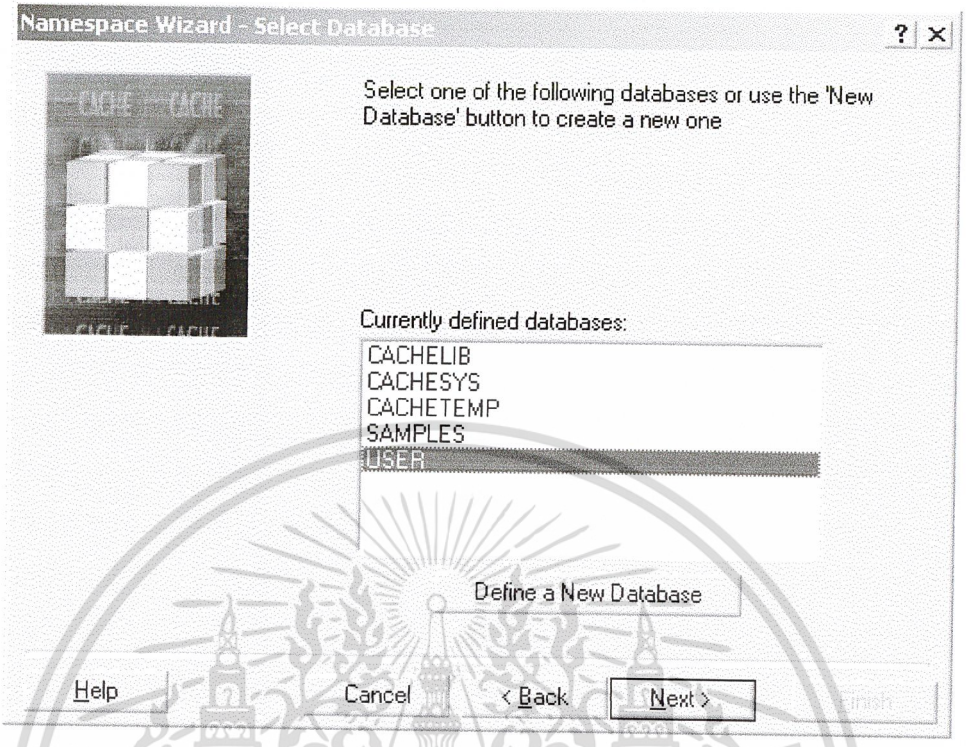
หลังการตัดตั้งระบบจัดการฐานข้อมูลเชิงวัตถุ Caché เรียบร้อยแล้ว จะต้องสร้าง Namespaces "MYPROJECT" สำหรับใช้งานกับระบบแผนที่ เนื่องจากระบบจะติดต่อกับฐานข้อมูลผ่านทาง Namespaces นี้ ขั้นตอนการสร้าง Namespaces มี ดังนี้

1) หลังจาก Start Caché แล้ว เลือก Configuration Manager
2) เลือกแท็ป Namespace แล้วกดปุ่ม Add เพื่อสร้างNamespace ใหม่ และตั้งชื่อว่า "MYPROJECT"



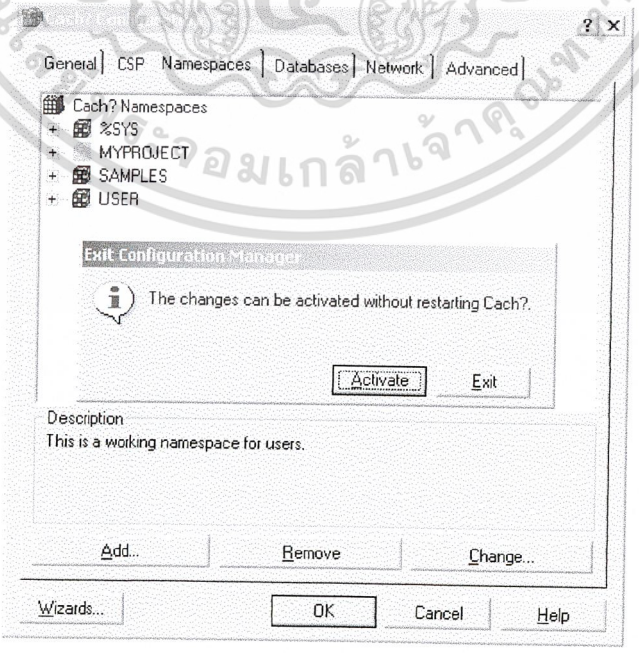
รูปภาพ ข-1 การสร้าง Namespace ใหม่และตั้งชื่อ

3) หลังจากนั้นจึงเลือกฐานข้อมูลที่จะทำการติดต่อด้วยให้เลือกฐานข้อมูล “USER”



รูปภาพ ข-2 การเลือ Database “USER”

4) หลังจากเสร็จสิ้นการสร้าง Namespace แล้ว จะต้อง Restart Caché เพื่อให้สามารถใช้งาน Namespace ที่สร้างใหม่ได้

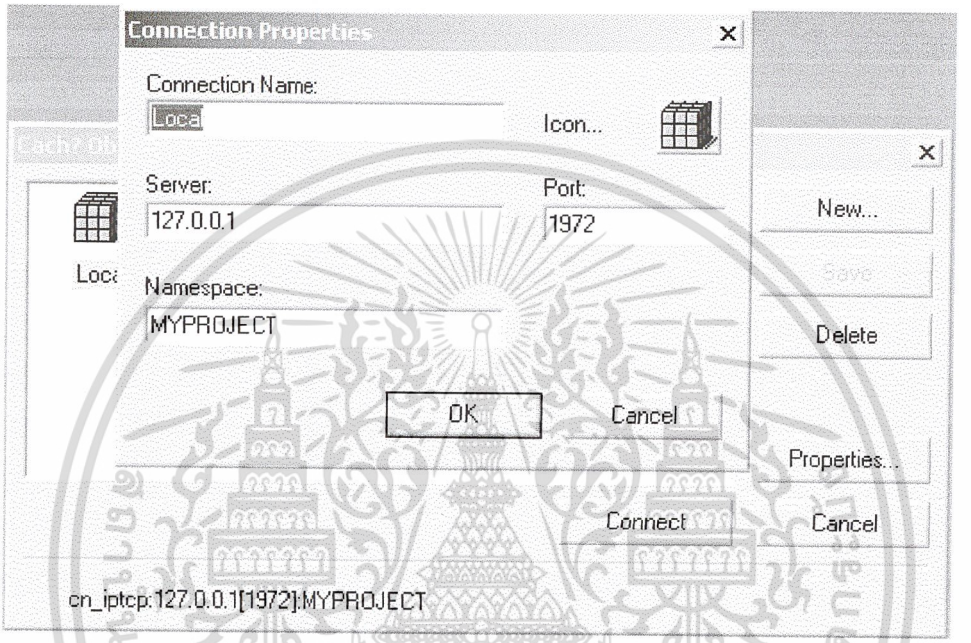


รูปภาพ ข-3 เลือก Activate เพื่อให้ Namespace ใหม่สามารถใช้งานได้

2. การนำเข้าไฟล์ CDL

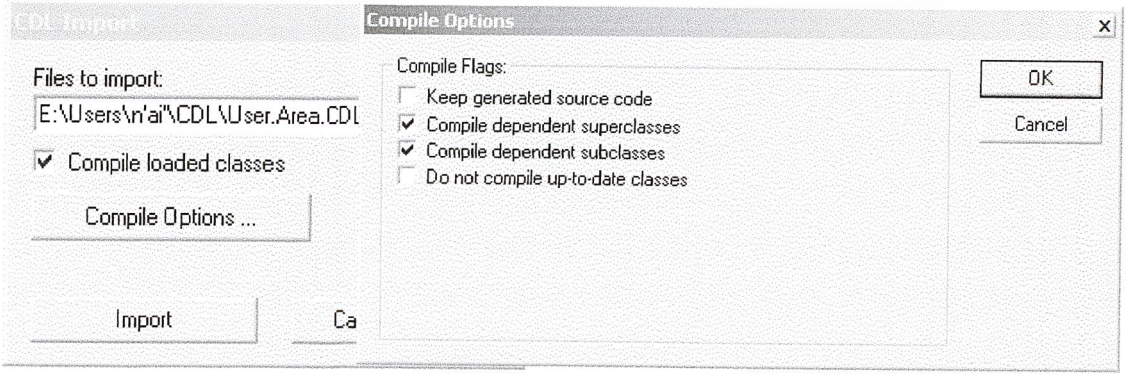
คลาสของข้อมูลของระบบแผนที่ถูกจัดเก็บอยู่ในไฟล์ CDL (ไฟล์.CDL) เมื่อต้องการใช้งานจะต้องนำเข้า ไฟล์CDL เพื่อสร้าง คลาสลงในฐานข้อมูล Caché ใน Namespace ที่ได้สร้างขึ้น

1) เลือก Object Architect จาก เมนู กดปุ่ม Property และกำหนด Namespace ใน Connection Property ให้เป็น “MYPROJECT” แล้วกดปุ่ม OK และ Connect



รูปภาพ ข-4 การกำหนด Connection Property

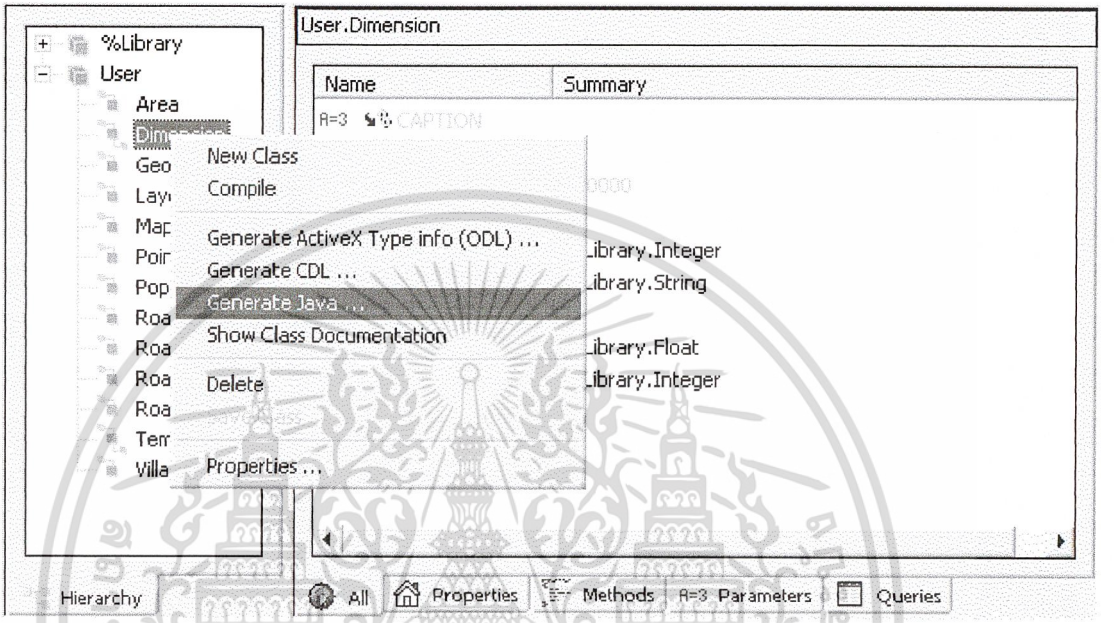
2) เลือกเมนู File/Import/Import CDL แล้วเลือกเปิดไฟล์นามสกุล CDL เลือก คอมไพล์ loaded classes และกำหนด Compile Options โดยเลือก Compile dependent superclasses และ Compile dependent subclasses หลังจากนั้นจึงนำเข้า



รูปภาพ ข-5 การเลือกไฟล์และกำหนดคุณสมบัติการ คอมไพล์

3. การสร้างไฟล์ภาษาจาวาจากคลาสในฐานข้อมูล Caché

เมื่อนำเข้าคลาสสำหรับเก็บข้อมูลแผนที่ในระบบจัดการฐานข้อมูลเชิงวัตถุ Caché แล้ว เมื่อต้องการติดต่อกับ Caché ด้วยภาษาจาวาจะต้องสร้างคลาสภาษาจาวาขึ้นจากคลาสในฐานข้อมูล โดยเลือกชื่อคลาสที่ต้องการใน Hierarchy และเลือก Generate Java จาก popup menu แล้วนำคลาสที่ได้ไปเก็บไว้ที่เดียวกับไฟล์โปรแกรมระบบแผนที่



รูปภาพ ข-6 การสร้างไฟล์ภาษาจาวาจากคลาสในฐานข้อมูลCaché

เมื่อต้องการใช้งานจะต้อง คอมไพล์ ไฟล์ภาษาจาวาที่สร้างขึ้น โดยจะต้องคอมไพล์เรียงลำดับจากคลาสลูกจนหมดแล้วจึงคอมไพล์คลาสพ่อแม่ โดยสามารถดูได้จากคลาสไดอะแกรมต้นแบบที่ออกแบบไว้ในบทที่ 4

4. การติดต่อฐานข้อมูล Caché จากโปรแกรมภาษา Java

1) กำหนด Classpath ไปที่ \CacheSys\Java\CacheJava.jar

2) การติดต่อกับฐานข้อมูลจะต้อง สร้างออบเจ็กต์จากคลาส " ObjectFactory " เพื่อเก็บผลที่ได้จากการเชื่อมต่อและนำไปใช้ในการเขียนโปรแกรมต่อไป การเชื่อมต่อทำได้ดังนี้

```
import COM.intersys.objects.*;
...
private ObjectFactory  factory;
try
{
    factory = new ObjectFactory( "cn_iptcp:127.0.0.1[1972]:MYPROJECT" );
```

```
}
catch (Exception ex)
{
        System.out.println( "Main::Caught exception: " +
        ex.getClass().getName() + ": " + ex.getMessage() );
}
```

สำหรับการเรียกใช้คลาสต่างๆ สามารถทำได้ตามปกติเช่นเดียวกับการเรียกใช้คลาสในไลบลารีภาษาจาวาทั่วไปหรือคลาสที่เราเขียนขึ้น

บรรณานุกรม

[1] ปีเตอร์ เอ. เบอร์รอฟ เขียน, ศรีสอาด ตั้งประเสริฐ แปล : "ระบบสารสนเทศภูมิศาสตร์เพื่อการประเมินค่าทรัพยากรที่ดิน = Principles of geographical information systems for land resources assessment", กรุงเทพฯ : ศูนย์พัฒนาหนังสือ กรมวิชาการ กระทรวงศึกษาธิการ, 2537

[2] W.Kirsten, M.Ihringer, B. Röhrig, P.Schulte : "Object-Oriented Application Development Using the Caché Postrelational Database", Springer-Verlag 1999

[3] กิตติ ภักดีวัฒนะกุล : "JAVA ฉบับพื้นฐาน" บริษัท เคทีพี คอมพ์ แอนด์ คอนซัลท์ จำกัด

[4] กิตติ ภักดีวัฒนะกุล : "JAVA ฉบับโปรแกรมเมอร์" บริษัท เคทีพี คอมพ์ แอนด์ คอนซัลท์ จำกัด

[5] ดร.วีระยักดิ์ ซึงถาวร : "Java Programming Volume I", กรุงเทพฯ : ซีเอ็ดยูเคชั่นม 2543

[6] ดร.วีระยักดิ์ ซึงถาวร : "Java Programming Volume II", กรุงเทพฯ : ซีเอ็ดยูเคชั่นม 2543