88823459
การวิเคราะห์และออกแบบระบบเชิงวัตถุ

พีระศักดิ์ เพียรประสิทธิ์

Outline

- “Things” in the Problem Domain
  - Domain classes
  - Data entities
- The Domain Model Class Diagram
- The Entity-Relationship Diagram

วัตถุประสงค์การเรียนรู้

- เรียนรู้แนวคิด “things” ใน problem domain เพื่อใช้ระบุความต้องการ (requirements)
- สามารถระบุ domain classes ในระบบได้
- เข้าใจวิธีการอ่านและเขียนแผนภาพ domain model class
- เข้าใจวิธีการอ่านและเขียนแผนภาพ entity-relationship
- เรียนรู้ domain model class จากกรณีศึกษา

Overview

- จากบทที่ก่อนหน้า ผู้เรียนจะทราบถึง use case ใช้ระบุ functional requirements โดยที่แผนภาพยูสเคสจะแสดงขอบเขตระบบในภาพรวมแล้ว
- ในบทนี้ กล่าวถึงแนวคิดของที่ใช้ระบุ requirements ซึ่งก็คือ data entities หรือ domain classes
- จาก กรณีของ RMO Tradeshow ในบทที่ 1 มีการกล่าวถึง domain classes คือ Supplier (ผู้ผลิต), Product (สินค้า/ผลิตภัณฑ์) และ Contact (ผู้ติดต่อ)

Things in the Problem Domain

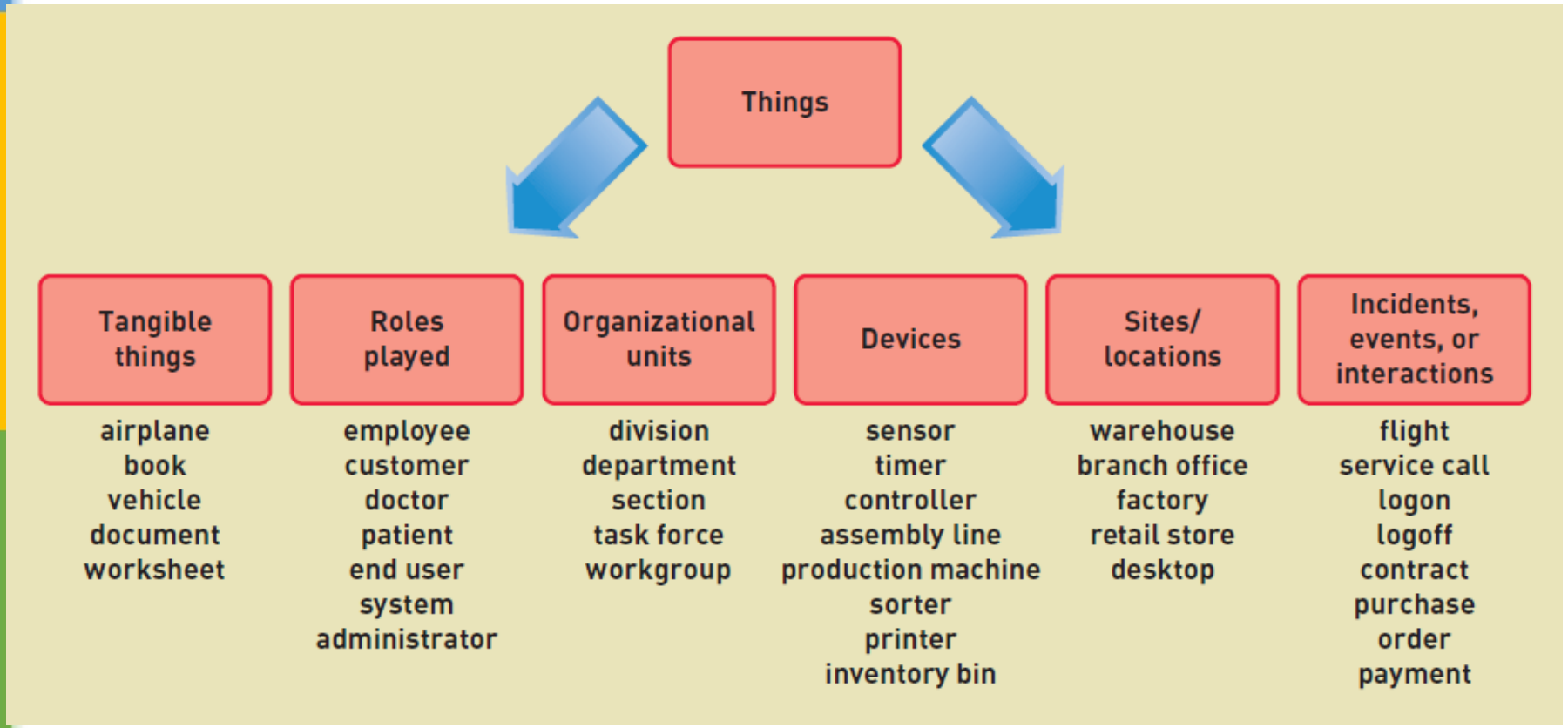
- Problem domain - ขอบเขตของปัญหาหรือขอบเขตความต้องการทางธุรกิจของผู้ใช้งานที่อยู่ภายใต้ขอบเขตของระบบงาน
- “Things” คือ สิ่งต่างๆ ที่เกิดขึ้นเมื่อผู้ใช้ทำงานและต้องการเก็บข้อมูลเหล่านั้นไว้
- ตัวอย่าง “Things” คือ products (สินค้า/ผลิตภัณฑ์), sales (รายการขาย), shippers (ผู้ขนส่ง), customers (ลูกค้า), invoices (ใบแจ้งหนี้), payments (การชำระเงิน)
- ในรายวิชานี้ เรียกสิ่งต่างๆ (things) ว่า domain class และเรียก database class ว่า data entities

Things in the Problem Domain

Two Techniques for Identifying them

- เทคนิคระดมความคิด (Brainstorming Technique)
  - ใช้ checklist สำหรับทุกๆ things ที่พบ และใช้วิธีการระดมความคิดในการระบุ domain classes ของแต่ละประเภท
- เทคนิคค้นหาคำนาม (Noun Technique)
  - ค้นหาคำนามจะคำอธิบายระบบ แล้วทำการกำหนดว่านั้นเป็น domain class, attribute หรือไม่เกี่ยวข้องกับระบบ

Brainstorming Technique



Brainstorming Technique

- แยก things ออกเป็นประเภทต่างๆ เช่น
  - รูปธรรม เช่น เครื่องบิน หนังสือ ยานยนต์ เอกสาร
  - บทบาทหน้าที่ เช่น พนักงาน ลูกค้า หมอ ผู้ป่วย
  - หน่วยงาน เช่น ฝ่าย แผนก
  - อุปกรณ์ เช่น เซ็นเซอร์ เครื่องผลิต เครื่องพิมพ์เอกสาร เครื่องวัดแสง อุปกรณ์จับเวลา
  - สถานที่ เช่น คลังสินค้า โรงงาน สำนักงานย่อย ร้านค้าปลีก
  - เหตุการณ์ที่ต้องการบันทึกข้อมูล เช่น เที่ยวบิน การจัดซื้อ การขาย

Brainstorming Technique:

Steps

- 1. ระบุผู้ใช้งาน (user) และกลุ่มของยูสเคส (use case)
- 2. ระดมความคิดกับผู้ใช้งาน (user) เพื่อระบุ things ที่เกี่ยวข้องใน use case เช่น ในยูสเคสเพิ่มข้อมูลสินค้าใหม่ ระบบ (System) จะต้องเก็บข้อมูล Product (สินค้า)
- 3. ใช้ระบบคำถามกับ things ประเภทต่างๆ เช่น
  - สิ่งนั้นเก็บข้อมูลอะไร
  - มีสถานที่ใดเกี่ยวข้อง
  - ผู้ใช้งานนั้นมีบทบาทใดในระบบ และจำเป็นต้องเก็บข้อมูลหรือไม่

Brainstorming Technique:

Steps

- 4. ทำงานร่วมกับผู้ใช้งานและ stakeholder เพื่อขยายความจากการระดมความคิด
- 5. กำจัดสิ่งที่ซ้ำกัน และรวมรวมผลลัพธ์

The Noun Technique

- เทคนิคในการระบุ problem domain classes (things) จากการค้นหาคำนามจากเอกสารที่ใช้อธิบายระบบ
- เป็นเทคนิคที่ได้รับความนิยม
- อุปสรรค
  - คำนามจำนวนมากถูกค้นพบ
  - คำนามจำนวนมากไม่ได้เป็นสิ่งที่ต้องเก็บในระบบ
  - คำพ้อง คำเหมือน ที่ยากต่อการระบุ attribute จริงๆ
- เป็นวิธีหนึ่งในการเริ่มต้นทำการวิเคราะห์ระบบ เมื่อผู้ใช้งานไม่พร้อมที่จะช่วยทำระดมความคิด

กรณีศึกษาของ RMO

- บริษัท RMO เป็นบริษัทค้าปลีกขนาดใหญ่ มีความต้องการที่จะขยายธุรกิจ ทำการขายสินค้าออนไลน์ โดยจะจัดทำระบบย่อยสมาชิกของร้าน เพื่อจัดเก็บประวัติลูกค้า ที่อยู่ในการจัดส่ง หมายเลขบัตรเครดิต ซึ่งเมื่อลูกค้าทำการสั่งซื้อสินค้าผ่านระบบออนไลน์แล้วก็จะจัดส่งสินค้าไปให้ลูกค้าตามที่ต้องการ หากลูกค้าไม่พอใจสินค้าสามารถส่งคืนสินค้าคืนมาที่บริษัทได้ภายใน 7 วัน และทุกๆ สิ้นเดือนก็แจ้งยอดเงินของแต่ละบัญชีลูกค้าไปยังธนาคารเพื่อทำการโอนเงินเข้าบริษัท รวมถึงรายงานสรุปยอดการขายสินค้าแต่ละรายการ

The Noun Technique:

Steps

- 1. ใช้ use case, actor และข้อมูลอื่นๆ ที่เกี่ยวข้องกับระบบ รวมถึงข้อมูล input และข้อมูล output เพื่อระบุคำนาม
  - จากกรณีศึกษาของ RMO คำนามได้แก่ customer, product item, sale, confirmation, transaction, shipping, bank, change request, summary report, management, transaction report, accounting, back order, back order notification, return, return confirmation
- 2. ใช้ข้อมูลจากระบบที่มีอยู่ ขั้นตอนการทำงานปัจจุบัน แบบฟอร์ม รายงาน เพื่อระบุรายการเพิ่ม
  - จากกรณีของ RMO ได้แก่ price, size, color, style, season, inventory quantity, payment method, and shipping address.

The Noun Technique:

Steps

- 3. จากรายการของคำนาม ให้พิจารณาคำนาม
  - คำถามที่ใช้ในการช่วยตัดสินใจว่าให้คำนามนั้น รวมหรือคงไว้ (include)
    - Things เป็นเอกลักษณ์ (ไม่ซ้ำ) หรือไม่
    - อยู่ในขอบเขตของระบบหรือไม่
    - ระบบต้องจัดเก็บข้อมูลเหล่านั้น มากกว่าหนึ่งรายการ
  - คำถามที่ใช้ในการช่วยตัดสินใจว่าให้คำนามนั้น นำออกไป (exclude)
    - เป็นคำพ้องกับ thing อื่นที่ได้ระบุไปแล้วหรือไม่
    - เป็น output ของระบบอื่น
    - เป็นเพียงข้อมูลที่ใช้ในการป้อนข้อมูลในการบันทึก
  - คำถามที่ใช้ระบุ thing ที่ยังไม่สามารถระบุได้
    - มีโอกาสที่จะเป็นข้อมูลเฉพาะเจาะจง
    - อาจจะเป็นบางสิ่ง หากสมมติฐานเปลี่ยนไป

The Noun Technique:

Steps

- 4. สร้างรายการของคำนามทั้งหมด พร้อมทั้งบันทึกช่วยจำว่าในแต่ละคำนามต้องทำการรวม (include) หรือเอาออก (exclude) หรือต้องทำการพิจารณาต่อไป

- 5. ตรวจทานกับผู้ใช้งาน ผู้มีส่วนได้ส่วนเสีย และสมาชิกในทีม และทำการกำหนดรายการของ things ใน problem domain

รายละเอียดของ Domain Classes

- คุณลักษณะ (Attribute) บรรยายถึงส่วนหนึ่งข้อมูลในแต่ละออบเจ็กต์ เช่น ลูกค้า มีชื่อ นามสกุล เบอร์โทรศัพท์
- คีย์ (Identifier or key)
  - คุณลักษณะหนึ่งที่มีคุณสมบัติเอกลักษณ์ เพื่อใช้ในการระบุถึงออบเจ็กต์ เช่น ในคลาสลูกค้า (Customer) มีคุณลักษณ์ รหัสลูกค้า (CustomerId) เป็นคีย์
  - หากเป็น data entities จะต้องมีคีย์ ส่วน domain class อาจจะมีหรือไม่มีก็ได้
- คีย์ประกอบ (Compound attribute)
  - คุณลักษณะจำนวนตั้งแต่ 2 คุณลักษณะขึ้นไป รวมกันอยู่ในโครงสร้างหนึ่งๆ เช่น ที่อยู่ ประกอบด้วย บ้านเลขที่ ถนน ตำบล อำเภอ จังหวัด รหัสไปรณีย์
  - บางครั้งคีย์ประกอบก็เป็นคีย์ที่ใช้ระบุเช่นกัน

รายละเอียดของ Domain Classes

- Attribute — describes one piece of information about each instance of the class
  - Customer has first name, last name, phone number
- Identifier or key
  - One attribute uniquely identifies an instance of the class. Required for data entities, optional for domain classes. Customer ID identifies a customer
- Compound attribute
  - Two or more attributes combined into one structure to simplify the model. (E.g., address rather than including number, street, city, state, zip separately). Sometimes an identifier or key is a compound attribute.

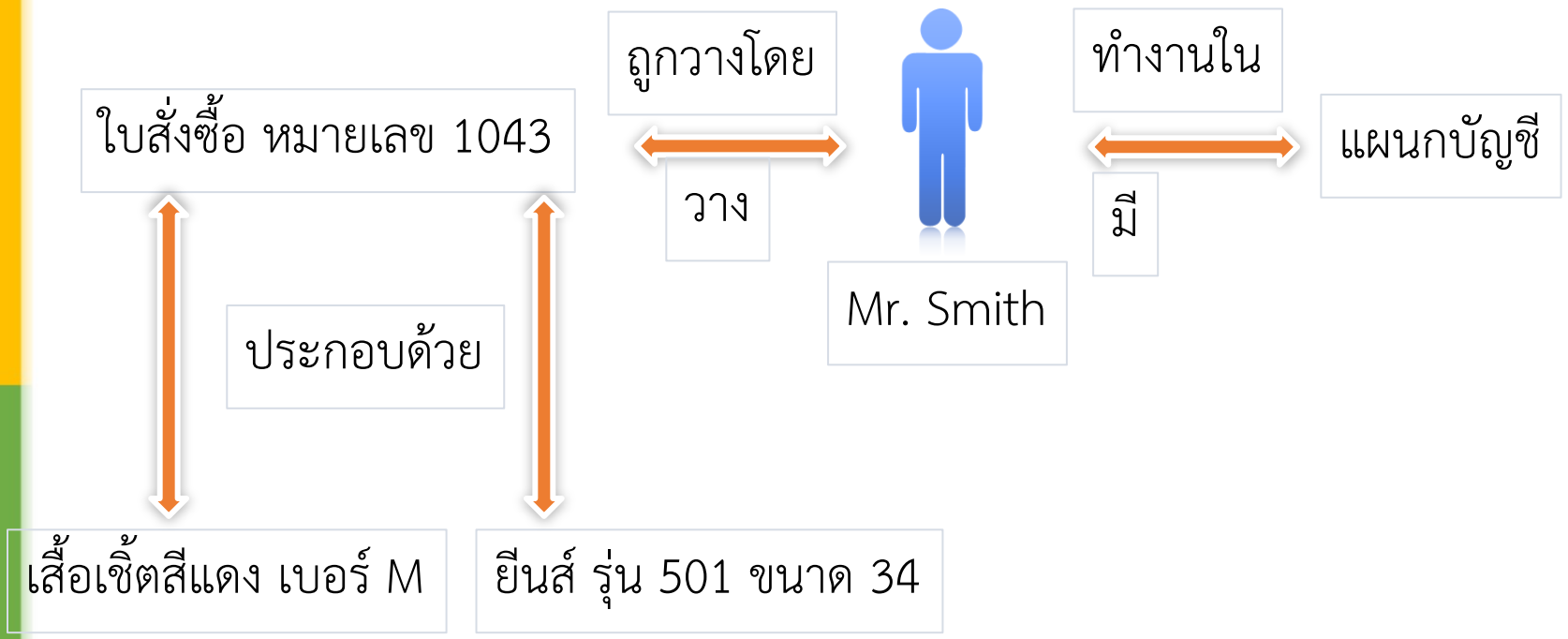
Attributes and Values

- Class คือ ประเภทของสิ่งต่างๆ (type of thing)
- Object คือ instance of the class ซึ่งในแต่ละ instance จะมีค่าของ attribute เป็นของตนเอง

ลูกค้าทุกคนจะต้องมี attribute	ลูกค้าแต่ละคนมี attribute		
รหัสลูกค้า (Customer ID)	101	102	103
ชื่อ (First name)	John	Mary	Bill
สกุล (Last name)	Smith	Jones	Casper
เบอร์โทรบ้าน (Home phone)	555-9182	423-1298	874-1297
เบอร์โทรที่ทำงาน (Work phone)	555-3425	423-3419	874-8566

Associations Among Things

- Association — ความสัมพันธ์ระหว่างคลาส



Just to Clarify...

- class diagram ใน UML เรียกความสัมพันธ์ว่า **association**
  - Multiplicity is term for the number of associations between classes: 1 to 1 or 1 to many
- ERD in database เรียกความสัมพันธ์ว่า **relationship**
  - Cardinality is term for number of relationships in entity relationship diagrams: 1 to 1 or 1 to many
- Associations and Relationships สามารถทำความเข้าใจได้ 2 รูปแบบ โดยการอ่านจากคลาสหนึ่งไปยังอีกคลาสหนึ่ง เช่น
  - A customer places an order (ลูกค้าวางใบสั่งซื้อ)
  - An order is placed by a customer (ใบสั่งซื้อของลูกค้า)

Minimum and Maximum Multiplicity

- ในความสัมพันธ์ระหว่างคลาสใดๆ จะต้องระบุถึงจำนวนต่ำสุด (Minimum) และจำนวนสูงสุด (Maximum)
  - จำนวนต่ำสุด (Minimum) มีค่าเป็น 0 กล่าวได้ว่าความสัมพันธ์นั้นเป็น optional
  - จำนวนต่ำสุด (Minimum) มีค่าอย่างน้อยเป็น 1 กล่าวได้ว่าความสัมพันธ์นั้นเป็น mandatory
- ตัวอย่าง
- Optional relationship เช่น ลูกค้ายังไม่มีการวางใบสั่งซื้อเลย หรือวางใบสั่งซื้อหลายใบสั่งซื้อ
- Mandatory relationship เช่น ใบสั่งซื้อ 1 ใบ มีรายการสั่งซื้อ 1 รายการหรือหลายรายการสั่งซื้อ

Types of Associations

- Binary Association
  - ความสัมพันธ์ระหว่าง 2 คลาส
    - กลุ่มเรียน (Course Section) มีนักเรียน (Students)
    - สมาชิก (Members) เข้าร่วม ชมรม (Club)
- Unary Association (recursive)
  - ความสัมพันธ์ระหว่าง 2 คลาส (2 คลาสนั้นเป็นคลาสเดียวกัน)
    - บุคคล (Person) แต่งงานกับ บุคคล (Person)
    - ส่วนประกอบ (Part) ถูกสร้างโดยใช้ส่วนประกอบ (parts)
- Ternary Association (three)
  - ความสัมพันธ์ระหว่างคลาสตั้งแต่ 3 คลาสขึ้นไป (N-ary Association)

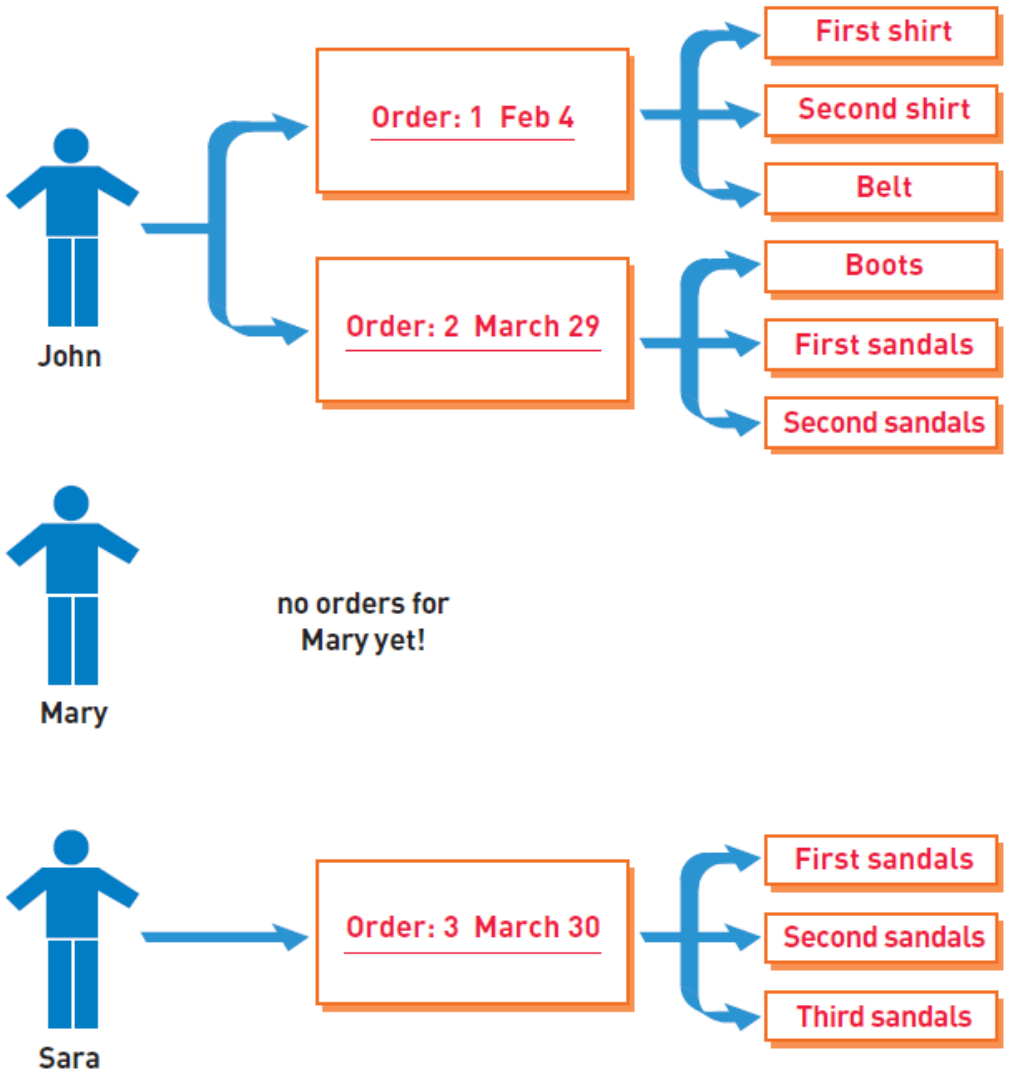
ตัวอย่าง

จากภาพแสดงถึงอ็อบเจกต์จำนวน 3 คลาส

Quick quiz:

How many associations are there?

1. มีคลาสใดบ้าง
2. มีความสัมพันธ์จำนวนเท่าใด
3. ค่าต่ำสุด ค่าสูงสุด ของแต่ละความสัมพันธ์ คืออะไร

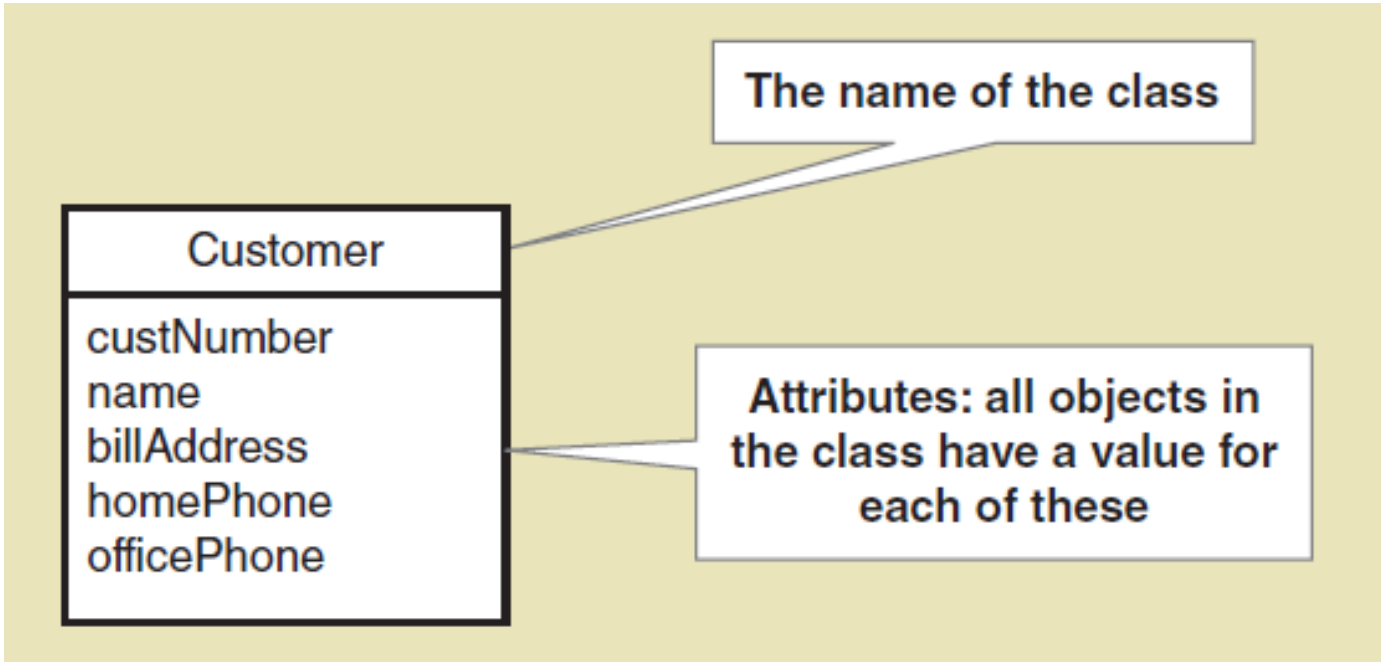


The Domain Model Class Diagram

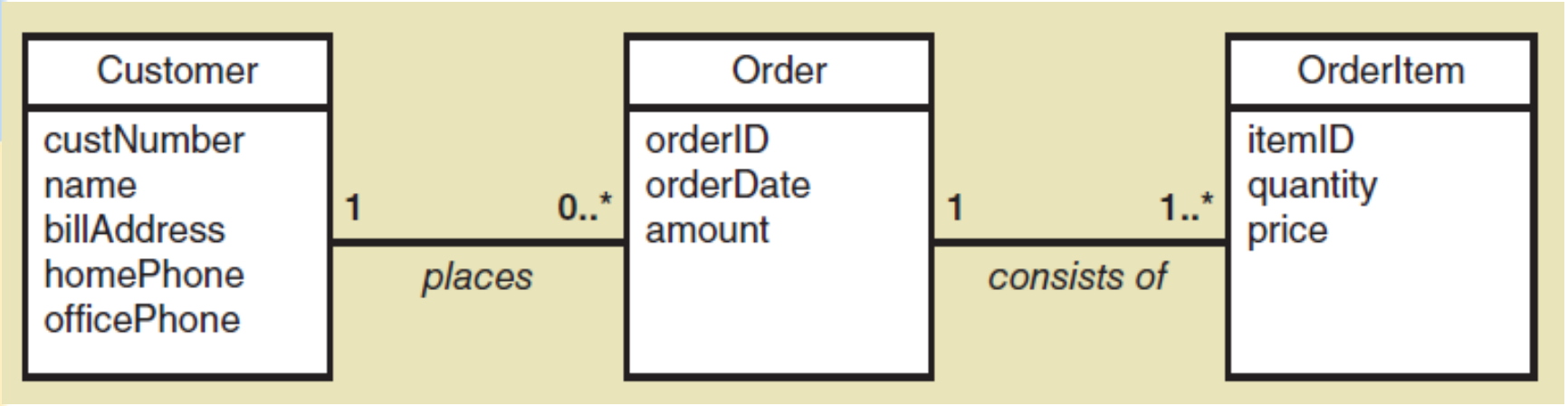
- Class
  - A category of classification used to describe a collection of objects
- Domain Class
  - Classes that describe objects in the problem domain
- Class Diagram
  - A UML diagram that shows classes with attributes and associations (plus methods if it models software classes)
- Domain Model Class Diagram
  - A class diagram that only includes classes from the problem domain, not software classes so no methods

Domain Class Notation

- Domain class has no methods
- Class name is always capitalized
- Attribute names are not capitalized and use camelback notation (words run together and second word is capitalized)

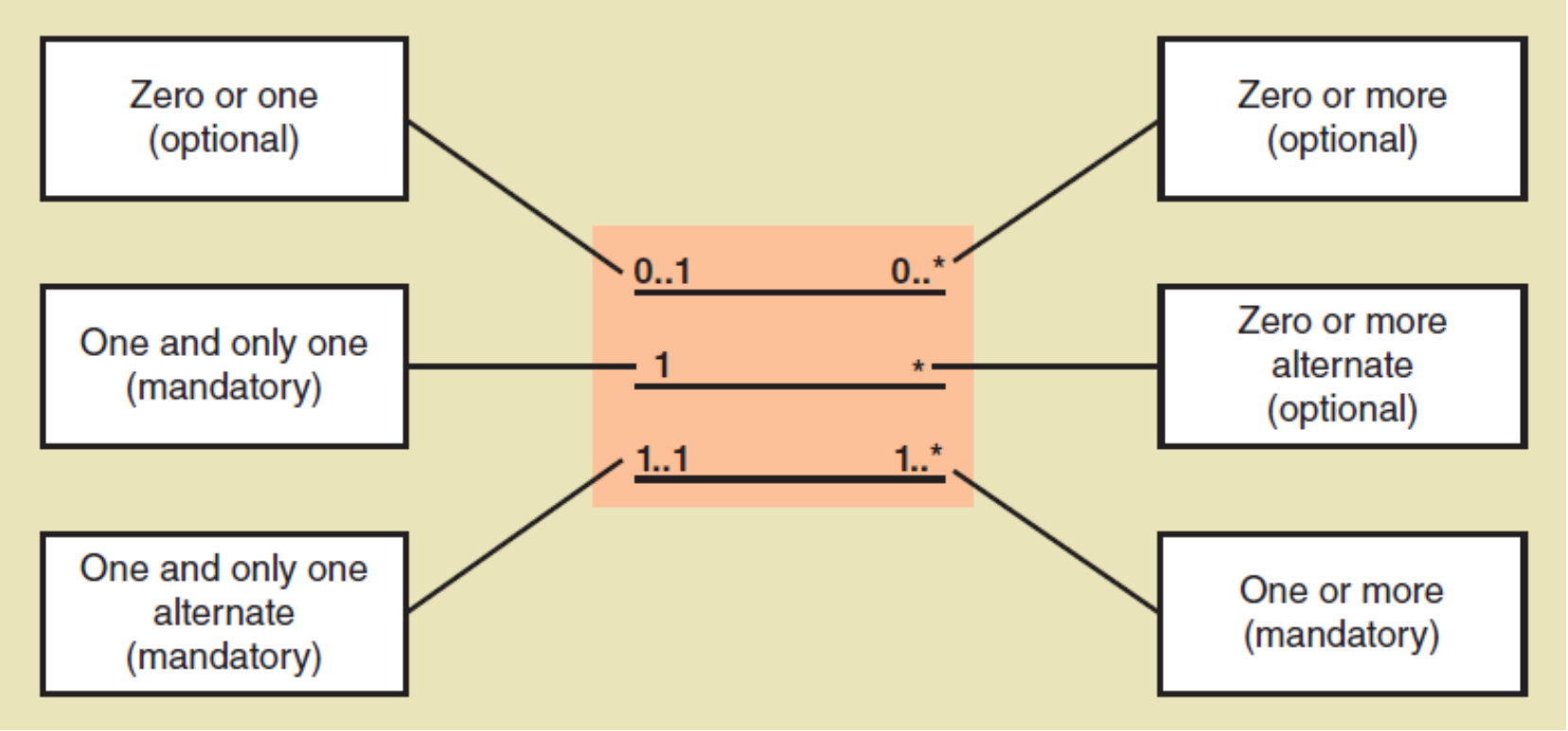


A Simple Domain Model Class Diagram



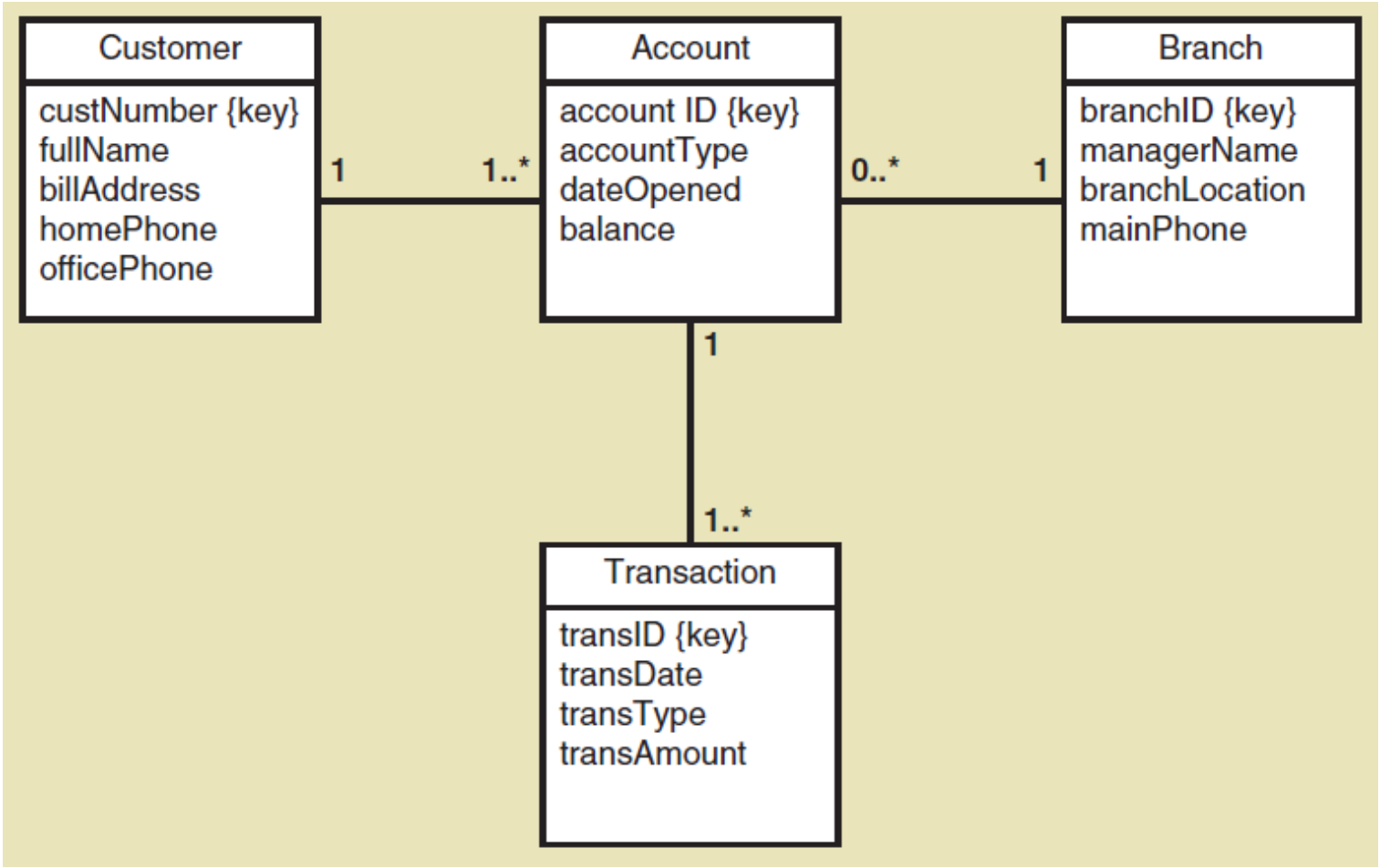
- Note: This diagram matches the semantic net shown previously
  - A customer places zero or more orders
  - An order is placed by exactly one customer
  - An order consists of one or more order items
  - An order item is part of exactly one order

UML Notation for Multiplicity



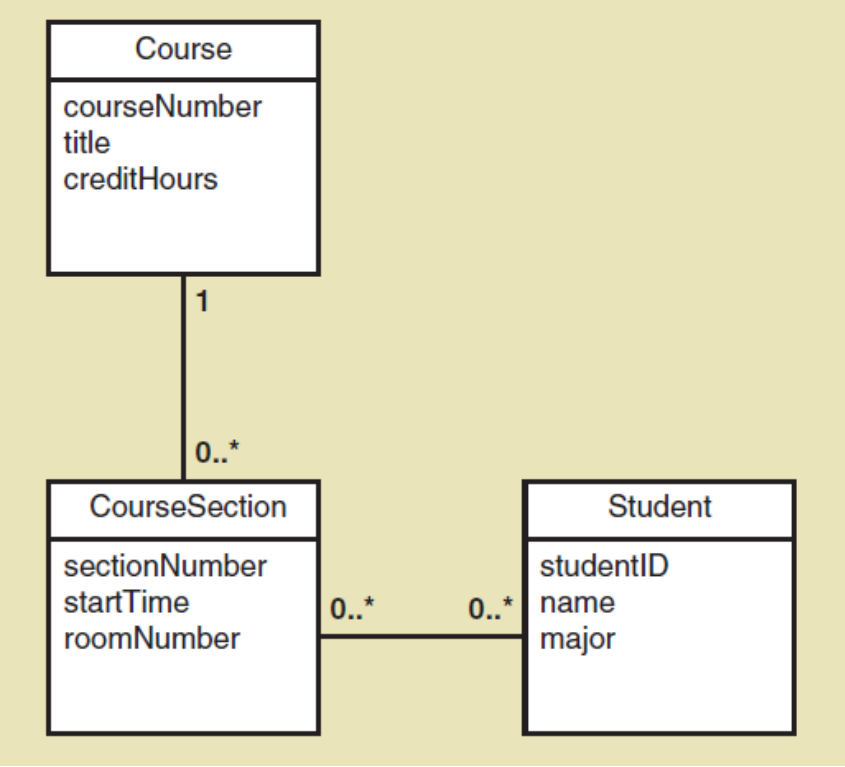
Domain Model Class Diagram

for a bank with many branches



Domain Model Class Diagram

for course enrollment at a university

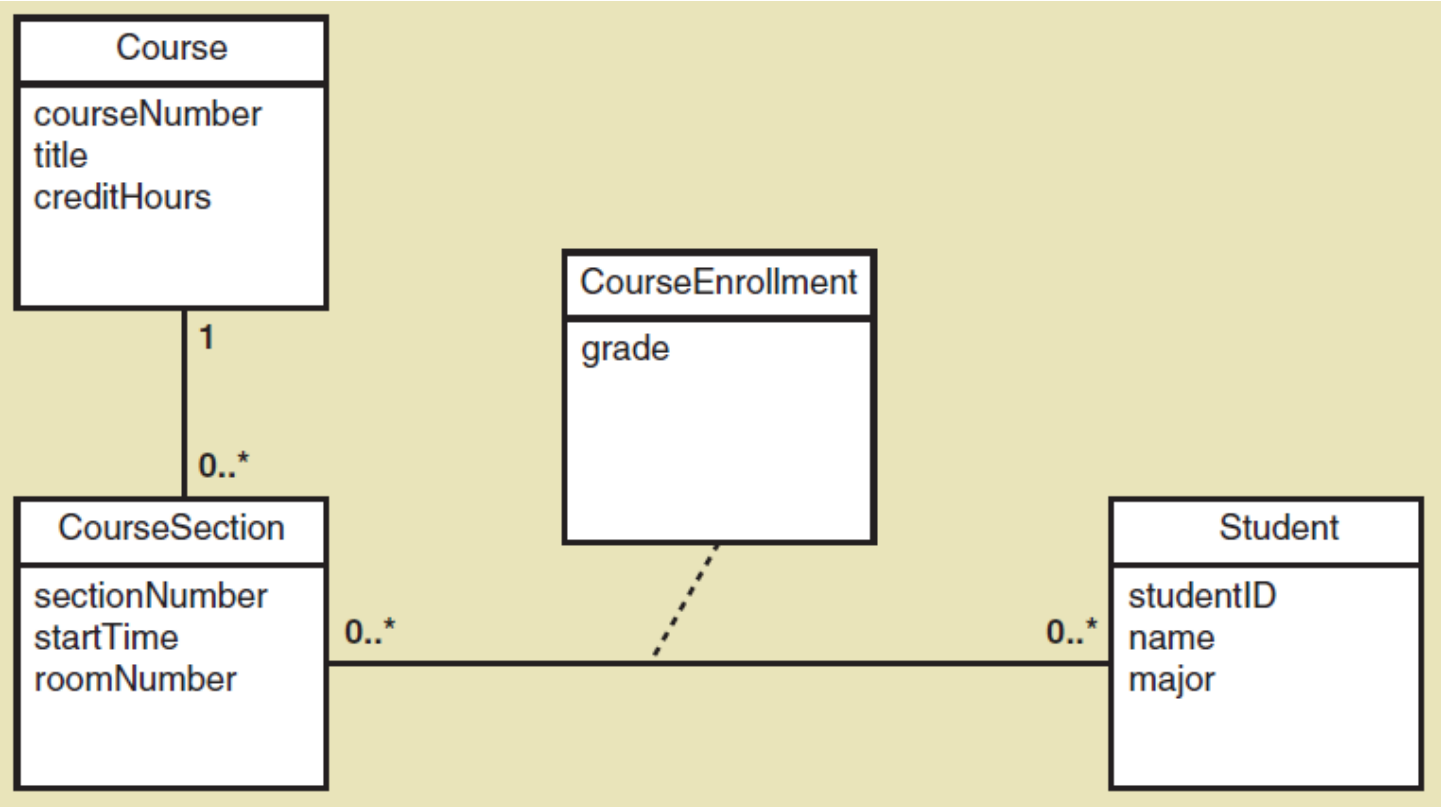


- Where is each student's grade remembered in this model?
  - Each section has many grades and each grade is association with a student
  - Each student has many grades and each grade is association with a section

Refined Course Enrollment Model

with an Association Class CourseEnrollment

- Association class— an association that is treated as a class in a many to many association because it has attributes that need to be remembered, such as



Note : คลาสที่มีความสัมพันธ์แบบ many to many แล้วต้องการจัดเก็บข้อมูล ให้สร้าง association class

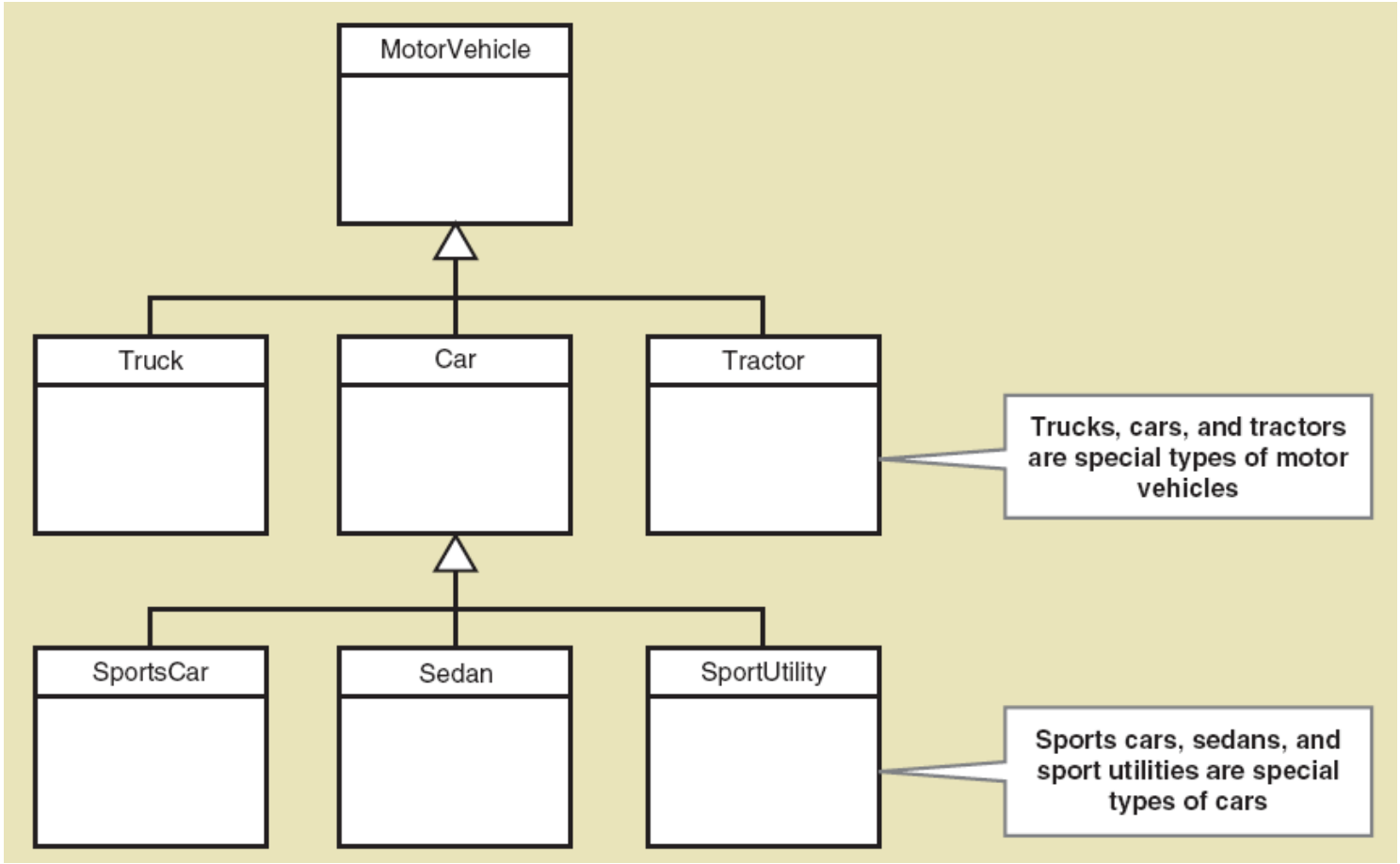
More Complex Issues about Classes:

Generalization/Specialization Relationships

- Generalization/Specialization
  - A hierarchical relationship where subordinate classes are special types of the superior classes. Often called an Inheritance Hierarchy
- Superclass
  - the superior or more general class in a generalization/specialization hierarchy
- Subclass
  - the subordinate or more specialized class in a generalization/specialization hierarchy
- Inheritance
  - the concept that subclasses classes inherit characteristics of the more general superclass

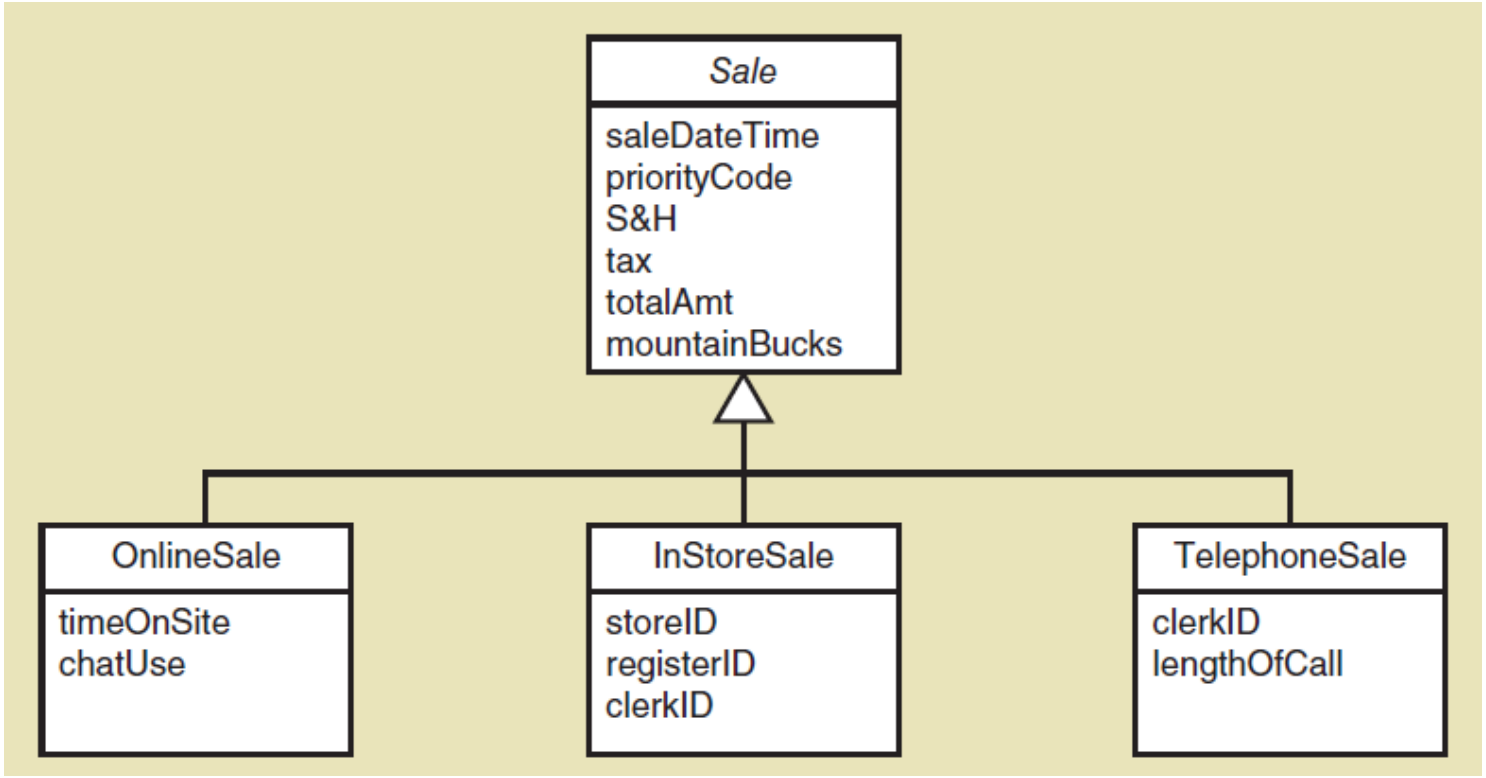
Generalization/Specialization

Inheritance



Generalization/Specialization

Inheritance for RMO Three Types of Sales

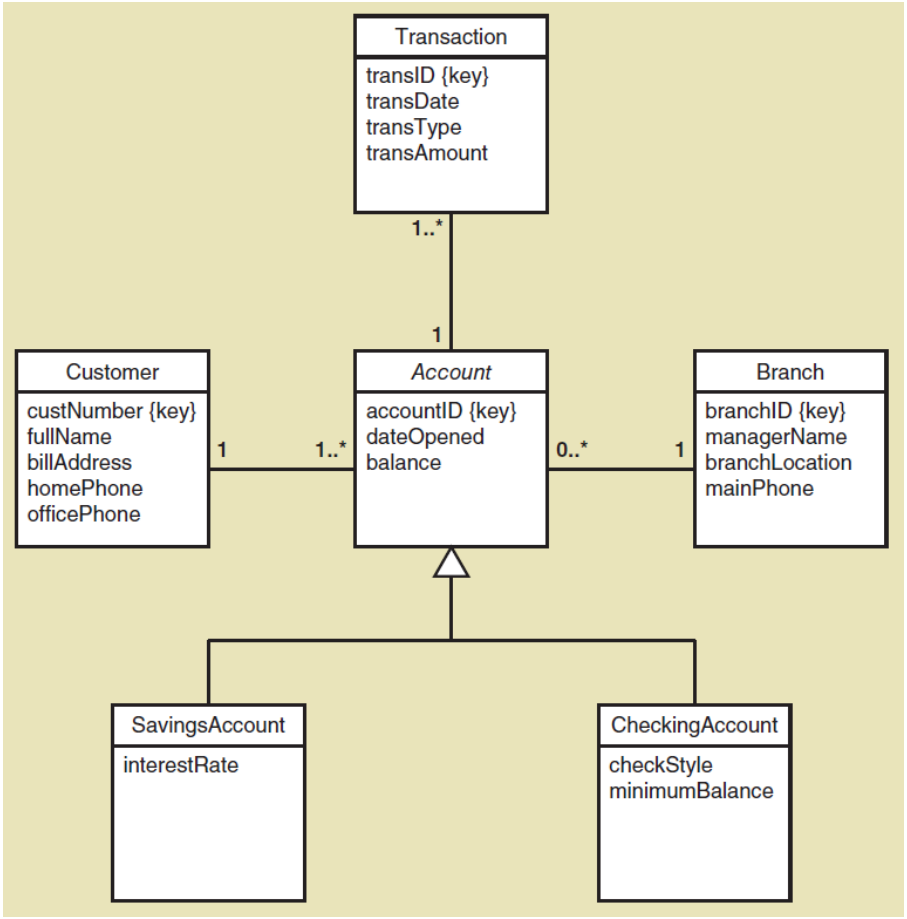


- Abstract class— a class that allow subclasses to inherit characteristics but never gets instantiated. In Italics (Sale above)
- Concrete class— a class that can have instances

Generalization/Specialization

Inheritance for the Bank with Special Types of Accounts

- A SavingsAccount has 4 attributes
- A CheckingAccount Has 5 attributes
- Note: the subclasses inherit the associations, too



More Complex Issues about Classes:

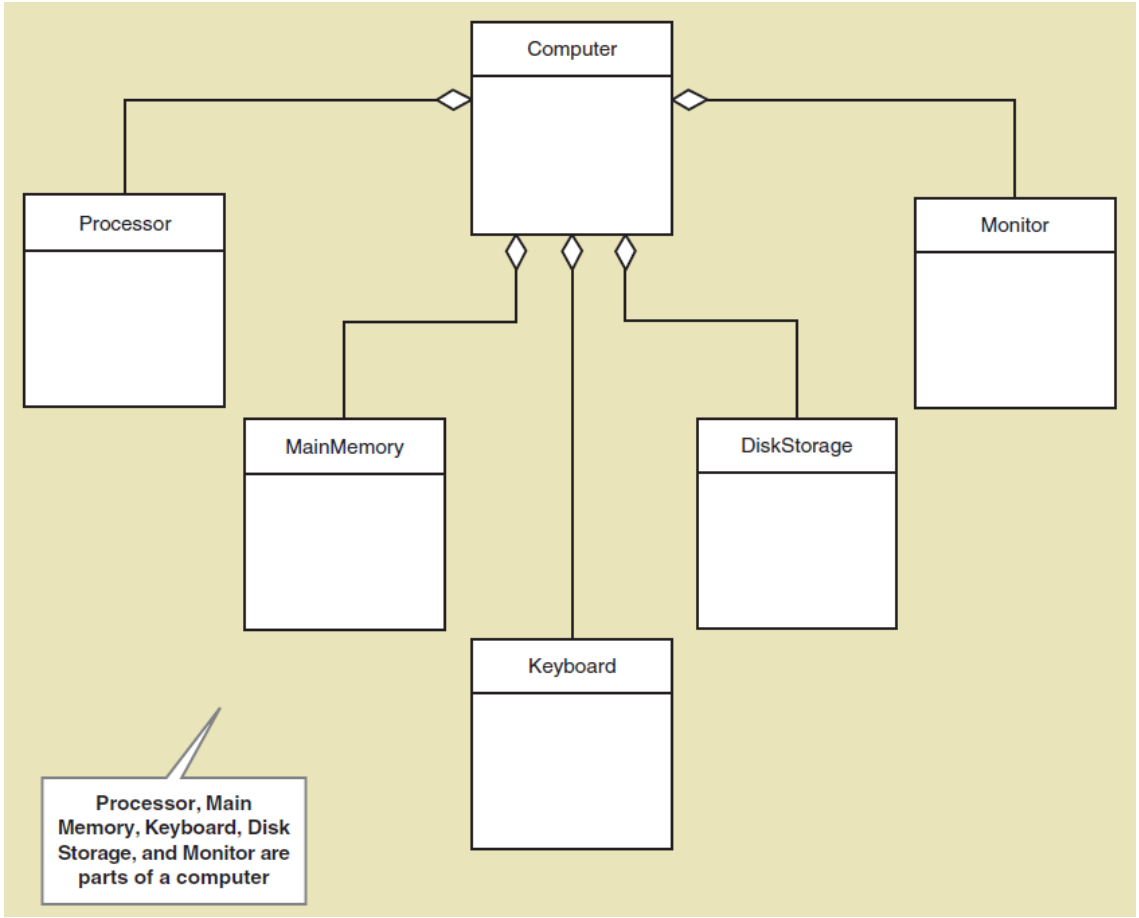
Whole Part Relationships

- Whole-part relationship— a relationship between classes where one class is part of or a component portion of another class
- Aggregation— a whole part relationship where the component part exists separately and can be removed and replaced (UML diamond symbol, next slide)
  - Computer has disk storage devices
  - Car has wheels
- Composition— a whole part relationship where the parts can no longer be removed (filled in diamond symbol)
  - Hand has fingers
  - Chip has circuits

Whole Part Relationships

Computer and its Parts

- Note: this is composition, with diamond symbol.
- Whole part can have multiplicity symbols, too (not shown)



More on UML Relationships

- There are actually three types of **relationships** in class diagrams
  - Association Relationships
    - These are associations discussed previously, just like ERD relationships
  - Whole Part Relationships
    - One class is a component or part of another class
  - Generalizations/Specialization Relationships
    - Inheritance
- So, try not to confuse relationship with association

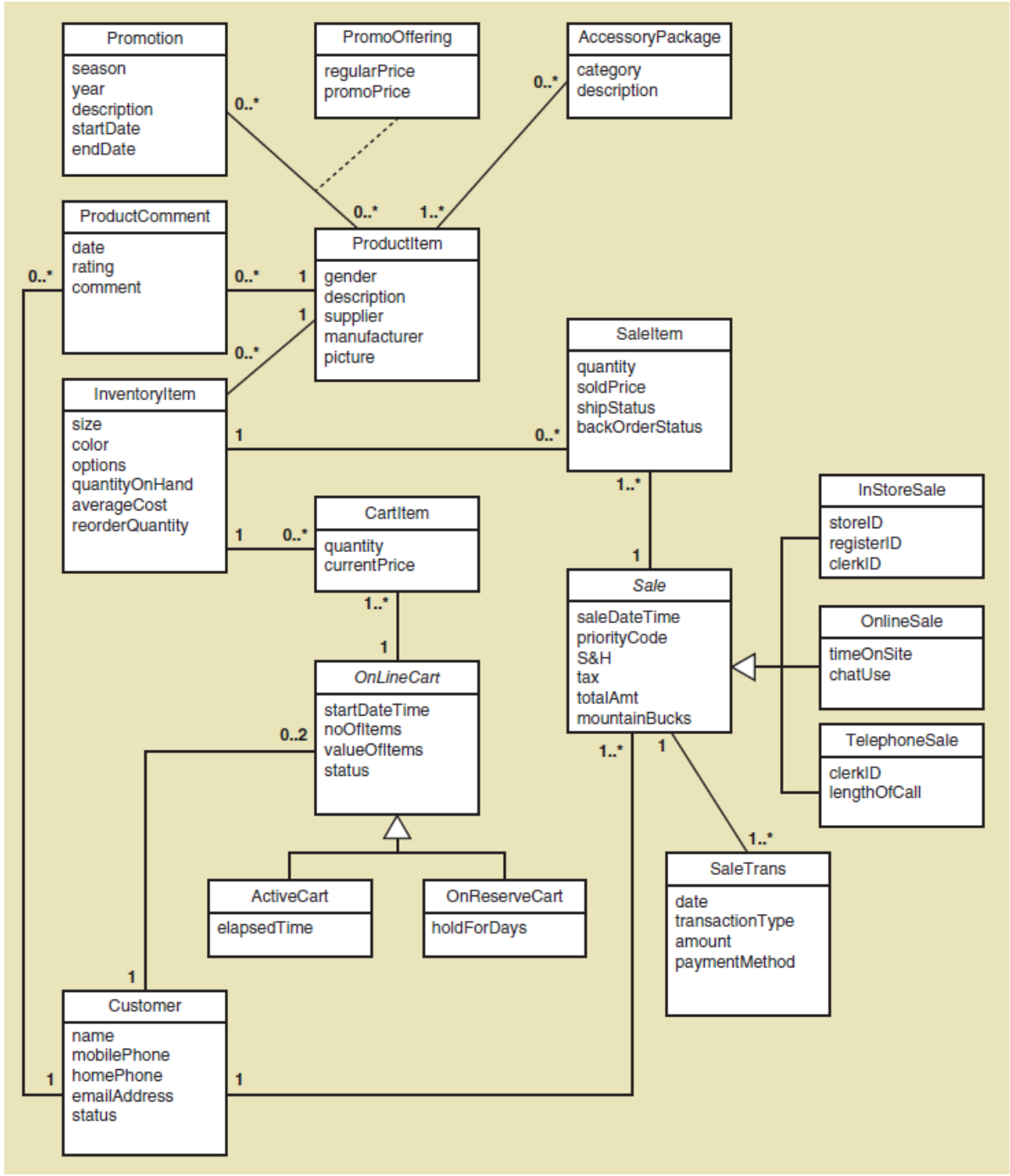
RMO CSMS Project

Domain Model Class Diagrams

- There are several ways to create the domain model class diagram for a project
- RMO CSMS has 27 domain classes overall
- Can create one domain model class diagram per subsystem for those working on a subsystem
- Can create one overall domain model class diagram to provide an overview of the whole system
- Usually in early iterations, an initial draft of the domain model class diagram is completed to guide development and kept up to date

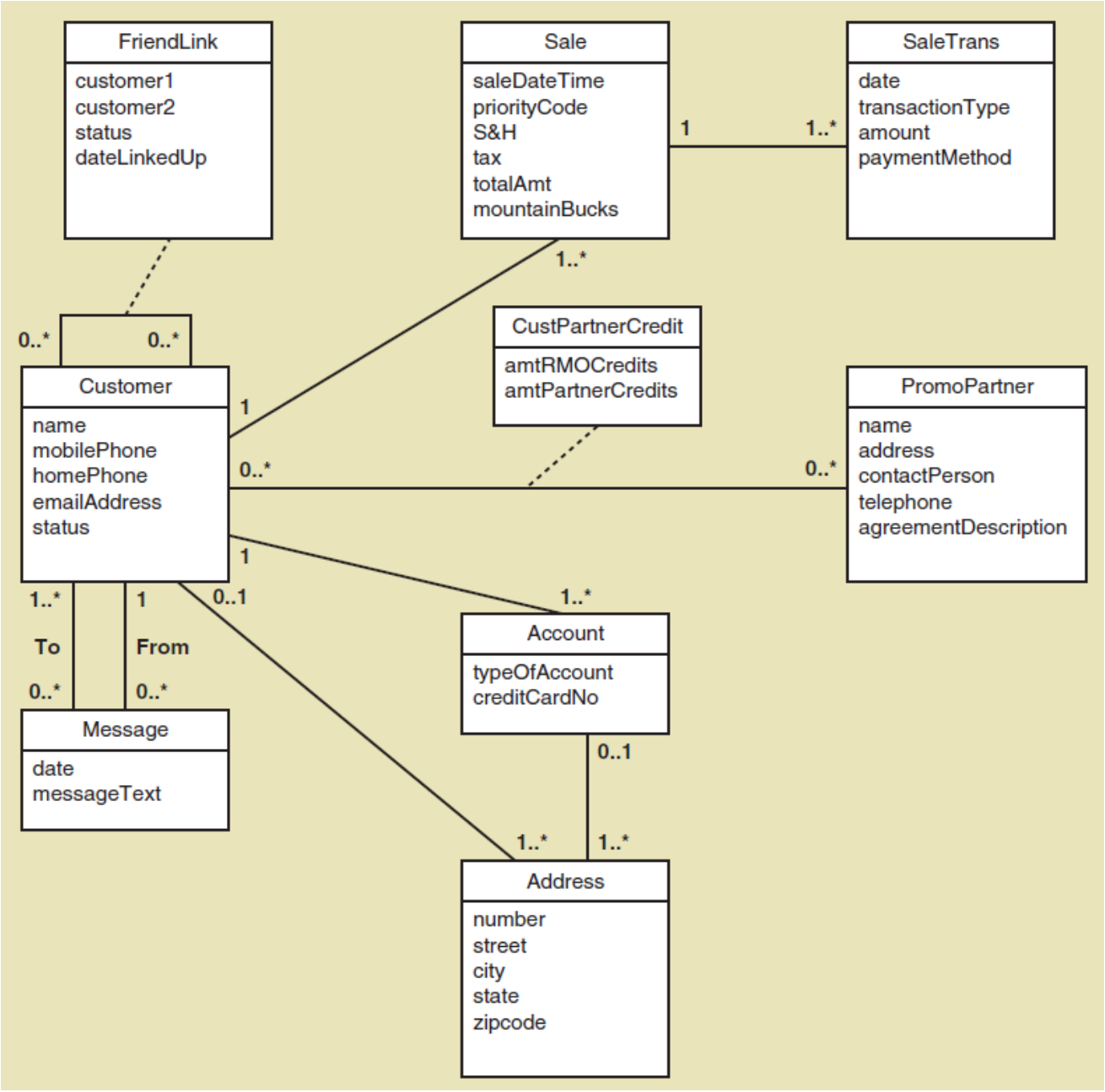
RMO CSMS Project

Sales Subsystem Domain Model Class Diagrams



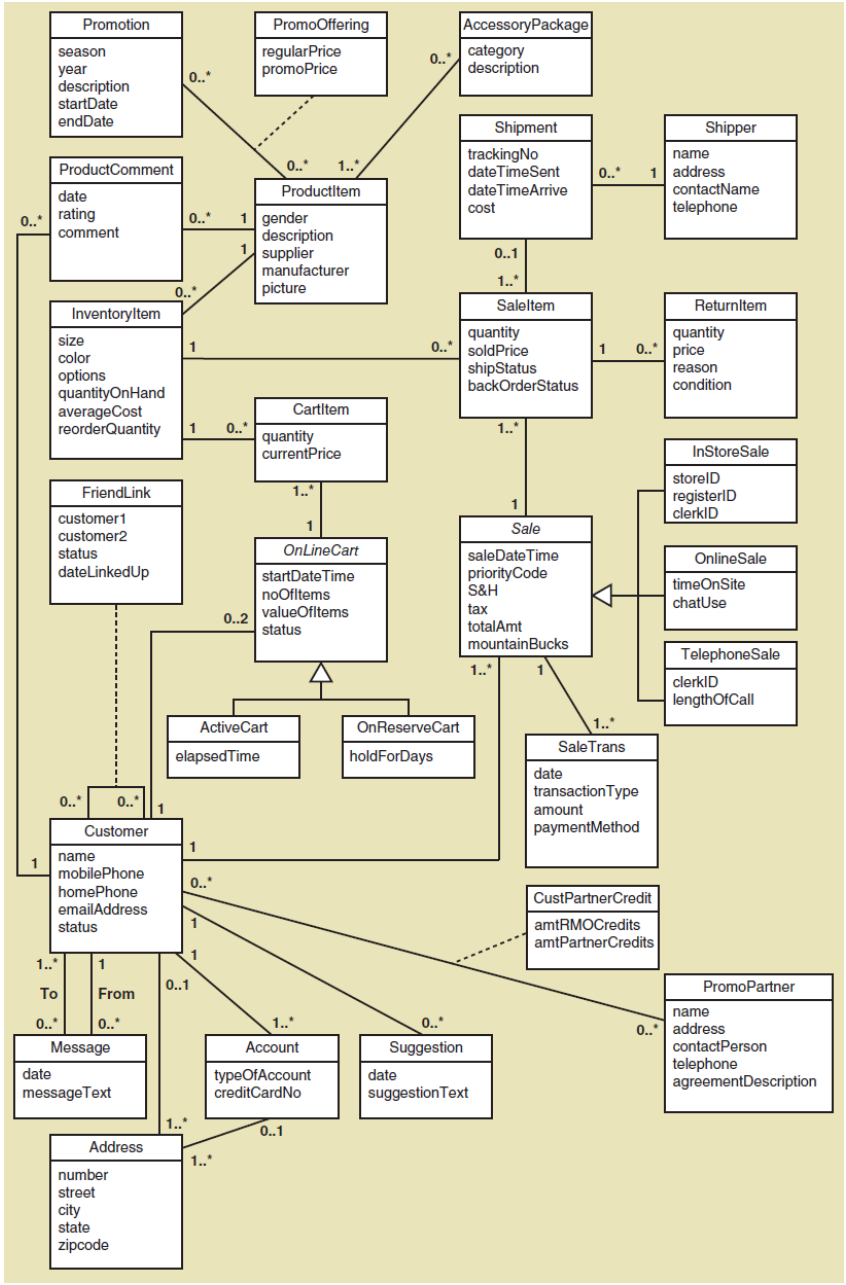
RMO CSMS Project

Customer Account Subsystem Domain Model Class Diagram



RMO CSMS Project

Complete Domain Model Class Diagram



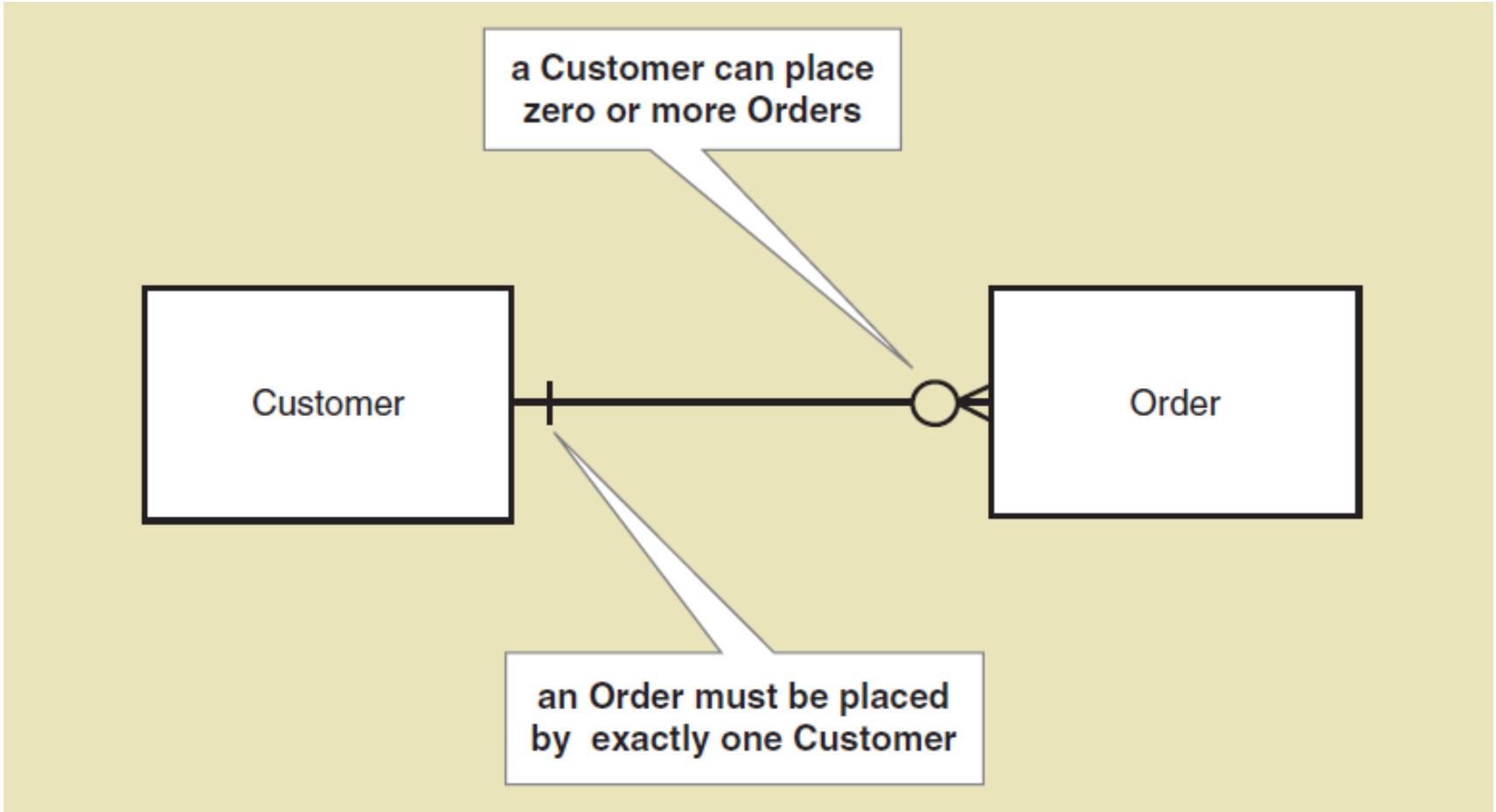
Entity-Relationship Diagrams

ERD

- An ERD shows basically the same information as a domain model class diagram
- It is not a UML diagram, but it is widely used by data analysts in database management
- There really is no standard notation, but most developers use the entity and crows feet notation shown in this text
- An ERD is not good for showing generalization/specialization relationships and whole part relationships

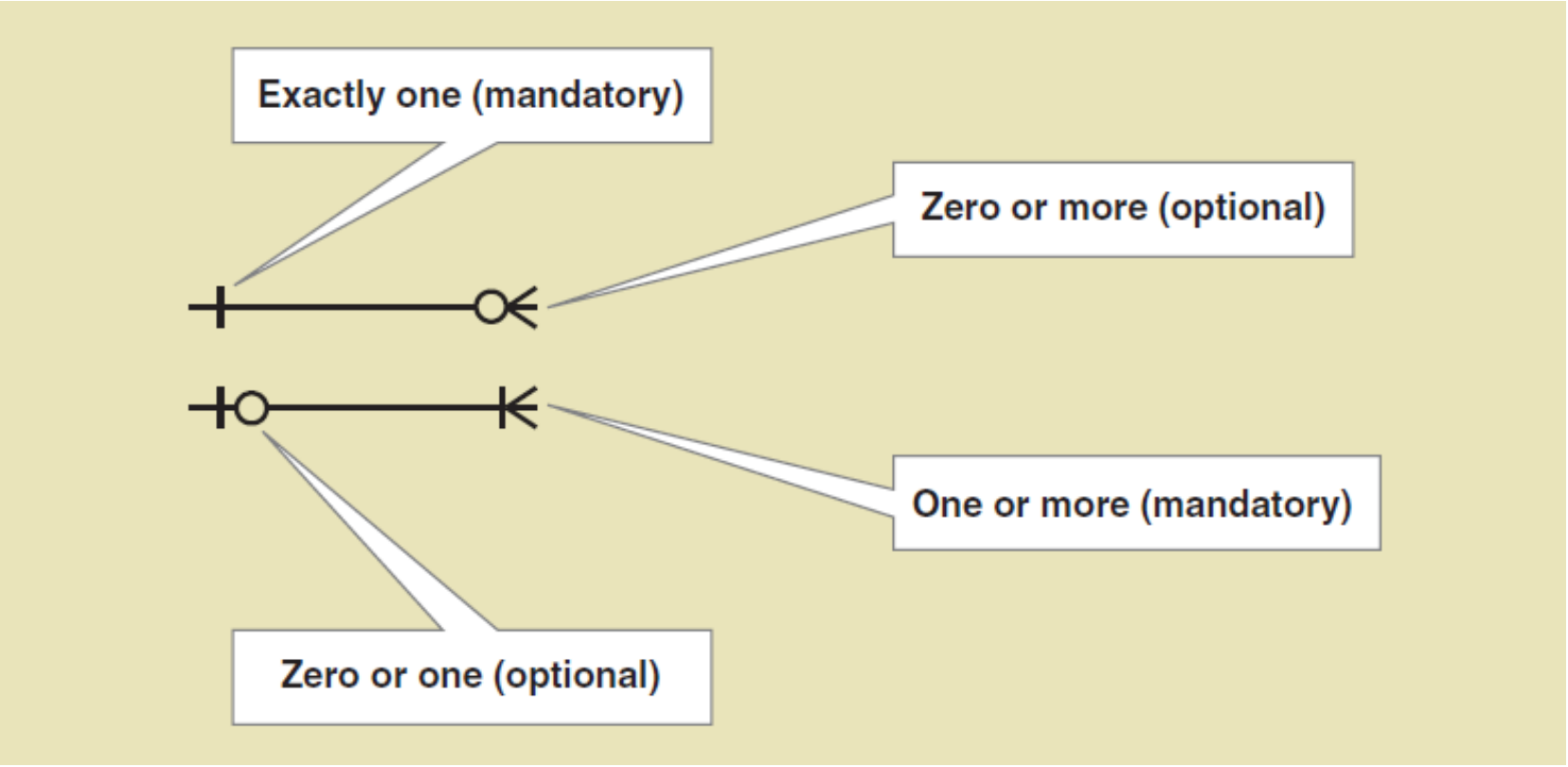
Example of ERD Notation

- A simple ERD without showing attributes

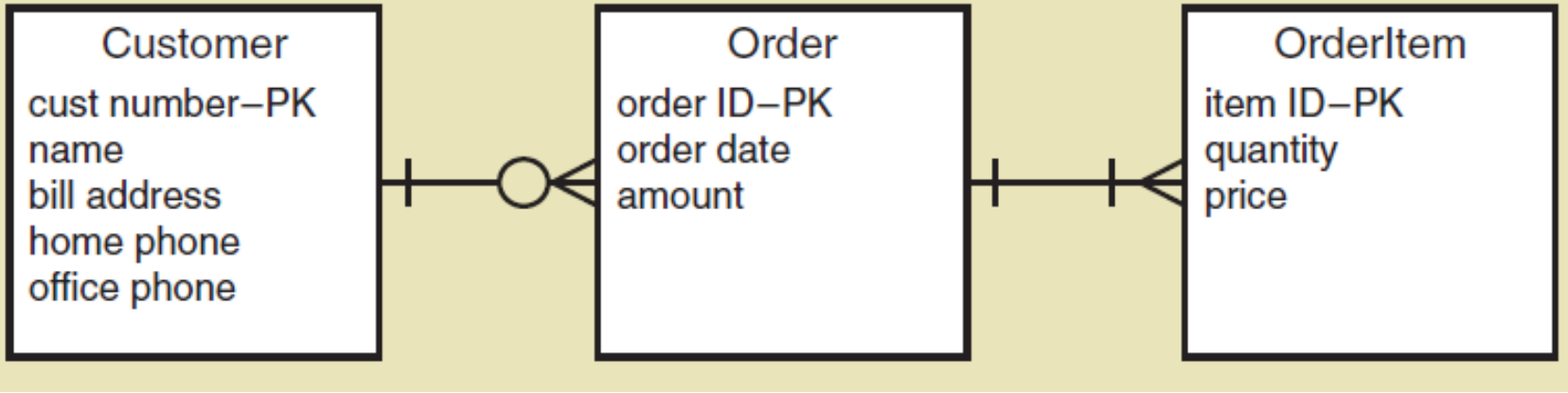


ERD Cardinality Symbols

often called the crows feet notation

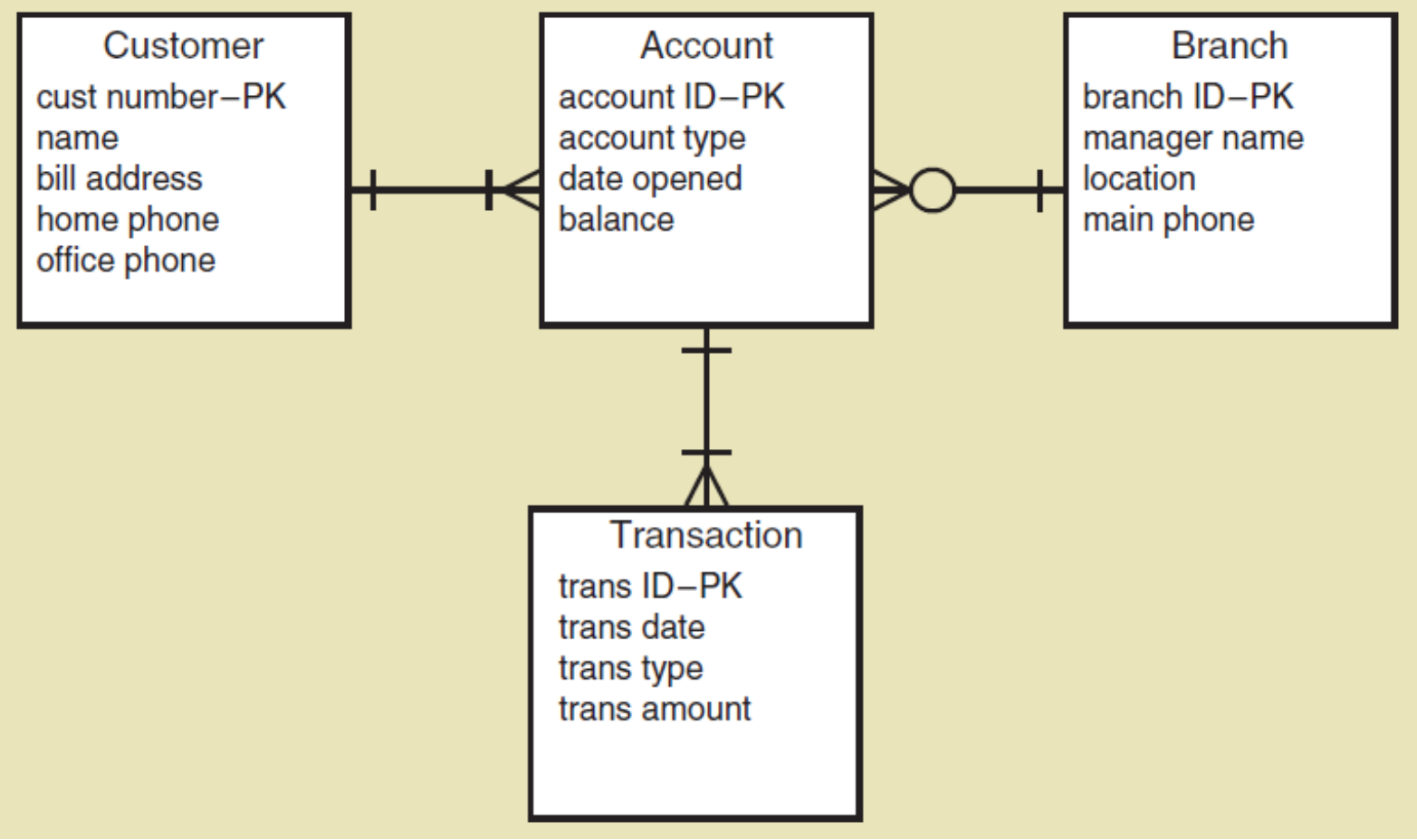


Expanded ERD with Attributes



- Note: This diagram matches the semantic net shown previously
- Also matches a domain model class diagram shown previously

An ERD for a Bank



Summary

- This chapter is the second of three that focuses on modeling functional requirements as a part of systems analysis
- “Things” in the problem domain are identified and modeled, called domain classes or data entities
- Two techniques for identifying domain classes/data entities are the brainstorming technique and the noun technique
- Domain classes have attributes and associations
- Associations are naturally occurring relationships among classes, and associations have minimum and maximum multiplicity

Summary

- The UML class diagram notation is used to create a domain model class diagram for a system. The domain model classes do not have methods because they are not yet software classes.
- There are actually three UML class diagram relationships: association relationships, generalization/specialization (inheritance) relationships, and whole part relationships
- Other class diagram concepts are abstract versus concrete classes, compound attributes, composition and aggregation, association classes, super classes and subclasses

Summary

- Entity-relationship diagrams (ERDs) show the same information as a domain model class diagram
- ERDs are preferred by database analysts and are widely used
- ERDs are not UML diagrams, and an association is called a relationship, multiplicity is called cardinality, and generalization/specialization (inheritance) and whole part relationships are usually not shown

กิจกรรม

- ให้นิสิตวิเคราะห์และออกแบบระบบ
  - ค้นหา class จาก problem domain
  - เขียน Domain Model Class Diagram

Exercise

- คอมพิวเตอร์ยี่ห้อ Dell เชื่อมต่อกับ Printer และ Scanner ยี่ห้อ Hewlett Packard ที่ถูกใช้งานโดยนาย ก ซึ่งเป็น Programmer
- นาย ข เป็น programmer เช่นเดียวกัน แต่เครื่องคอมพิวเตอร์ที่ใช้เป็นยี่ห้อ IBM และเชื่อมต่อกับ Printer
- นาย ค เป็นนักวิเคราะห์ระบบ รับผิดชอบการวิเคราะห์ระบบให้กับฝ่ายการบัญชี และฝ่ายสั่งซื้อของบริษัทนำเข้าและส่งออกแห่งหนึ่ง

Exercise

- ในบริษัทจัดหางานแห่งหนึ่ง จะให้บริการเป็นสื่อกลางในการติดต่อกันระหว่าง นายจ้างที่ต้องการลูกจ้างและผู้สมัครงานที่ต้องการหางานทำ โดยจะรับจัดเก็บข้อมูลเกี่ยวกับคุณสมบัติของผู้สมัครและคุณสมบัติของงานแต่ละตำแหน่งที่นายจ้างต้องการ และเมื่อพบว่ามีตำแหน่งงานที่เหมาะแก่ผู้สมัคร บริษัทจะจัดทำจดหมายเพื่อติดต่อให้นายจ้างและผู้สมัครให้มาสัมภาษณ์กันที่บริษัท

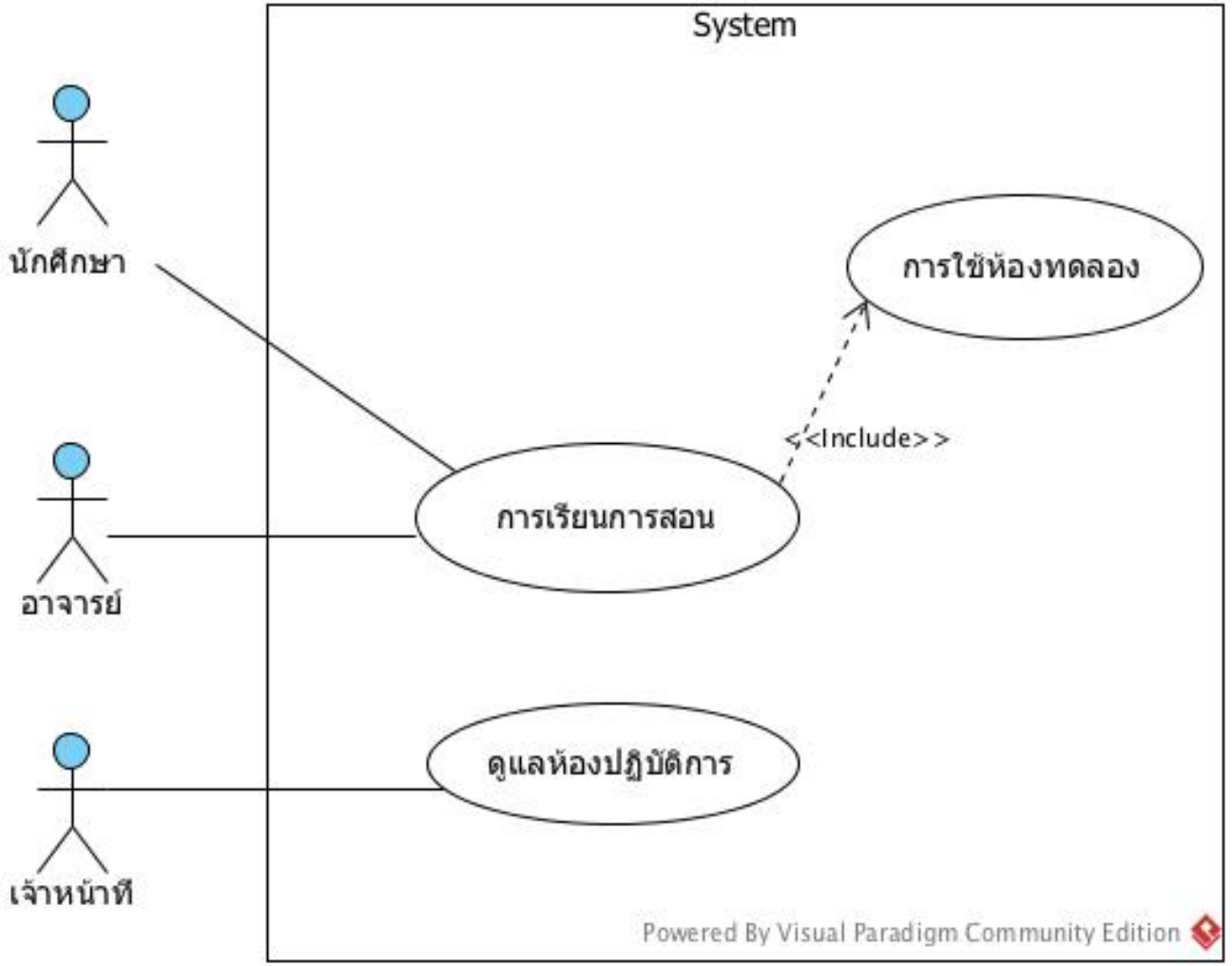
Exercise

- ใน 1 ปี ประกอบไปด้วยเดือนทั้งสิ้น 12 เดือน ได้แก่ มกราคม , กุมภาพันธ์, มีนาคม, ..., ธันวาคม อนึ่ง ในแต่ละเดือนจะประกอบไปด้วยวันจำนวนหนึ่ง ซึ่งวันที่มีในเดือนนั้น จำแนกได้เป็น วันธรรมดา (Working Days) วันหยุดปลายสัปดาห์ (Weekends) และวันหยุดนักขัตฤกษ์ (Holidays) ซึ่งวันหยุดนักขัตฤกษ์นั้นสามารถจำแนกได้เป็น
  - 1. วันหยุดสากล เช่น วันแรงงาน วันปีใหม่ เป็นต้น
  - 2. วันหยุดทางศาสนา เช่น วันมาฆบูชา
  - 3. วันหยุดที่ประกาศโดยรัฐ เช่น วันจักรี วันเฉลิมพระชนมพรรษาฯ เป็นต้น

ตัวอย่าง จงสร้าง Class Diagram จาก Problem Domain ที่กำหนดให้ต่อไปนี้

- ในคณะวิทยาศาสตร์ของสถาบันแห่งหนึ่งมีบุคลากรหลายประเภทด้วยกัน ได้แก่ อาจารย์ นักศึกษา และเจ้าหน้าที่ โดยที่อาจารย์แต่ละท่านมีหน้าที่ในการสอนวิชาใดวิชาหนึ่ง หรือมากกว่า 1 วิชาก็ได้ และนักศึกษาก็มีหน้าที่ในการศึกษาวิชาใดวิชาหนึ่ง หรือมากกว่า 1 วิชาก็ได้ในเวลาเดียวกัน เจ้าหน้าที่ของภาควิชา คือเจ้าหน้าที่ที่ประจำห้องทดลองต่างๆ โดยกำหนดว่าใน 1 ห้องทดลองจะต้องมีเจ้าหน้าที่ 1 คนเสมอ

Use Case Diagram ที่ได้

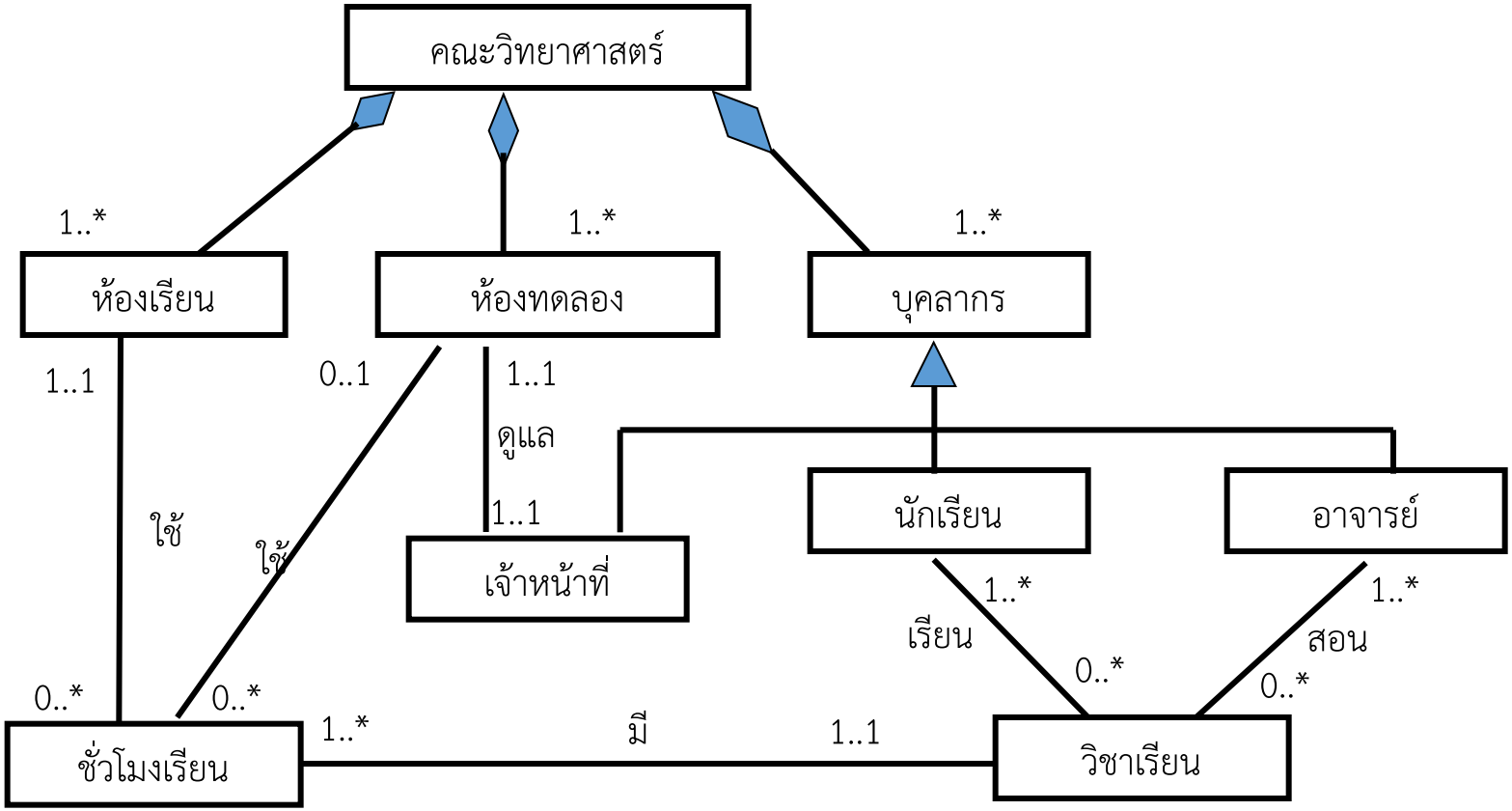


Use Case	*Object หรือ Class ที่หาได้จาก Use Case*
การเรียนการสอน	*นักเรียน อาจารย์* *ห้องเรียน วิชาเรียน ชั่งโมงเรียน*
การใช้ห้องทดลอง	*นักเรียน อาจารย์* *ห้องทดลอง*
การดูแลห้องทดลอง	*เจ้าหน้าที่* *ห้องทดลอง*

จากตาราง จะเห็นว่าในช่อง Object หรือ Class ที่หาได้ จะมี 2 บรรทัด โดยบรรทัดบนจะเป็น Object หรือ Class ที่เป็น Actor และบรรทัดที่ 2 จะหมายถึง Object อื่นๆ

เมื่อรวม Object หรือ Class ที่ได้จากทุกๆ Use Case จะได้ว่า Class ที่มีทั้งหมดของระบบได้แก่ อาจารย์ นักศึกษา เจ้าหน้าที่ ห้องเรียน วิชาเรียน ชั่วโมงเรียน และห้องทดลอง

- สามารถสร้าง Class Diagram เบื้องต้น (ยังไม่มี Attributes และ Functions) ดังรูป โดยในรูปได้เพิ่ม Class บาง Class ที่จำเป็น โดย Class ที่เพิ่มขึ้นได้แก่
  - คณะวิทยาศาสตร์ซึ่งเกิดจาก Aggregation Abstraction (คณะวิทยาศาสตร์ เกิดจาก Aggregation ของห้องเรียน ห้องทดลอง และบุคลากร)
  - Class บุคลากรที่เกิดจาก Generalization (บุคลากร จำแนกเป็นเจ้าหน้าที่ นักศึกษาและอาจารย์)

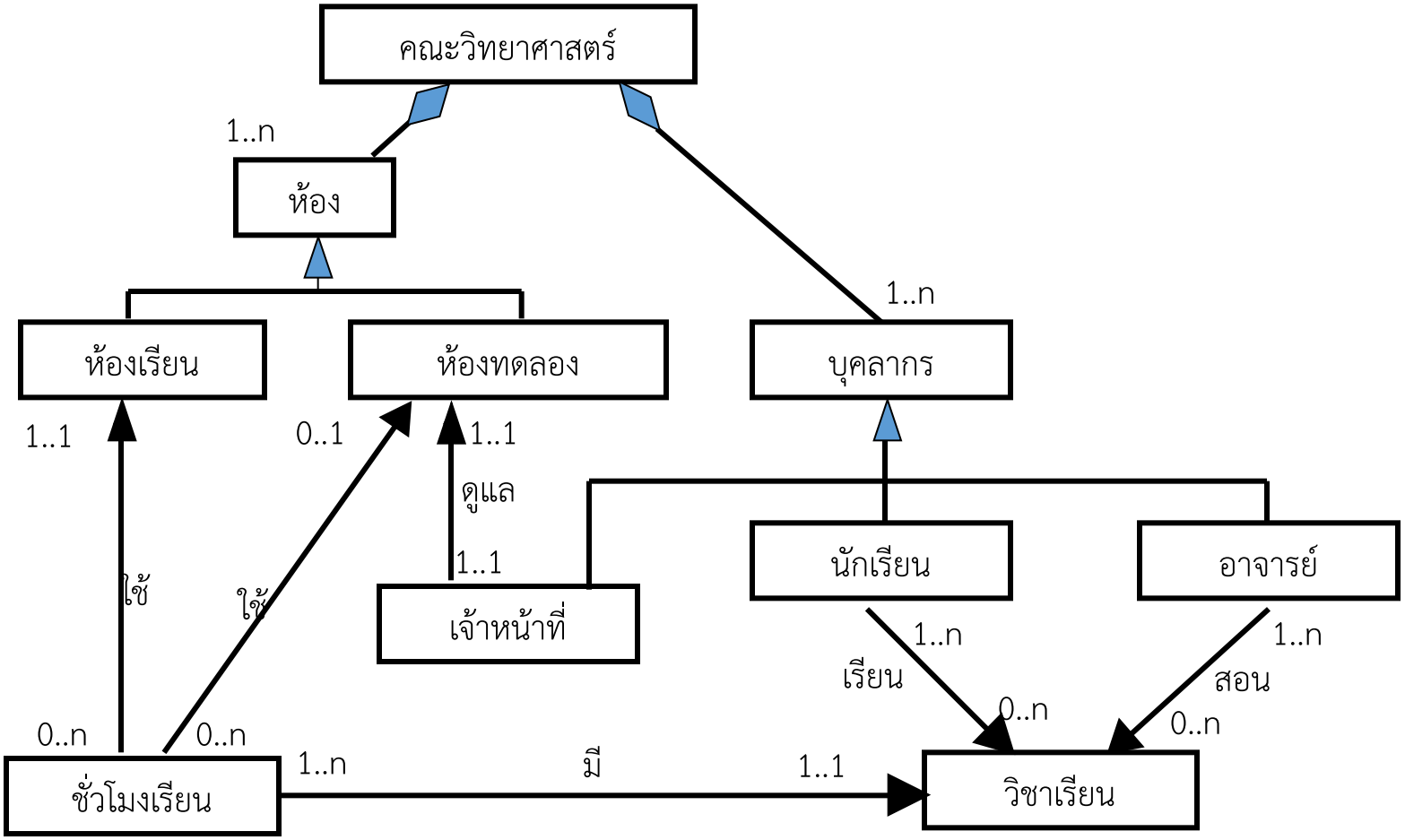


รูป Class Diagram เบื้องต้นจาก Scenario ที่กำหนดให้

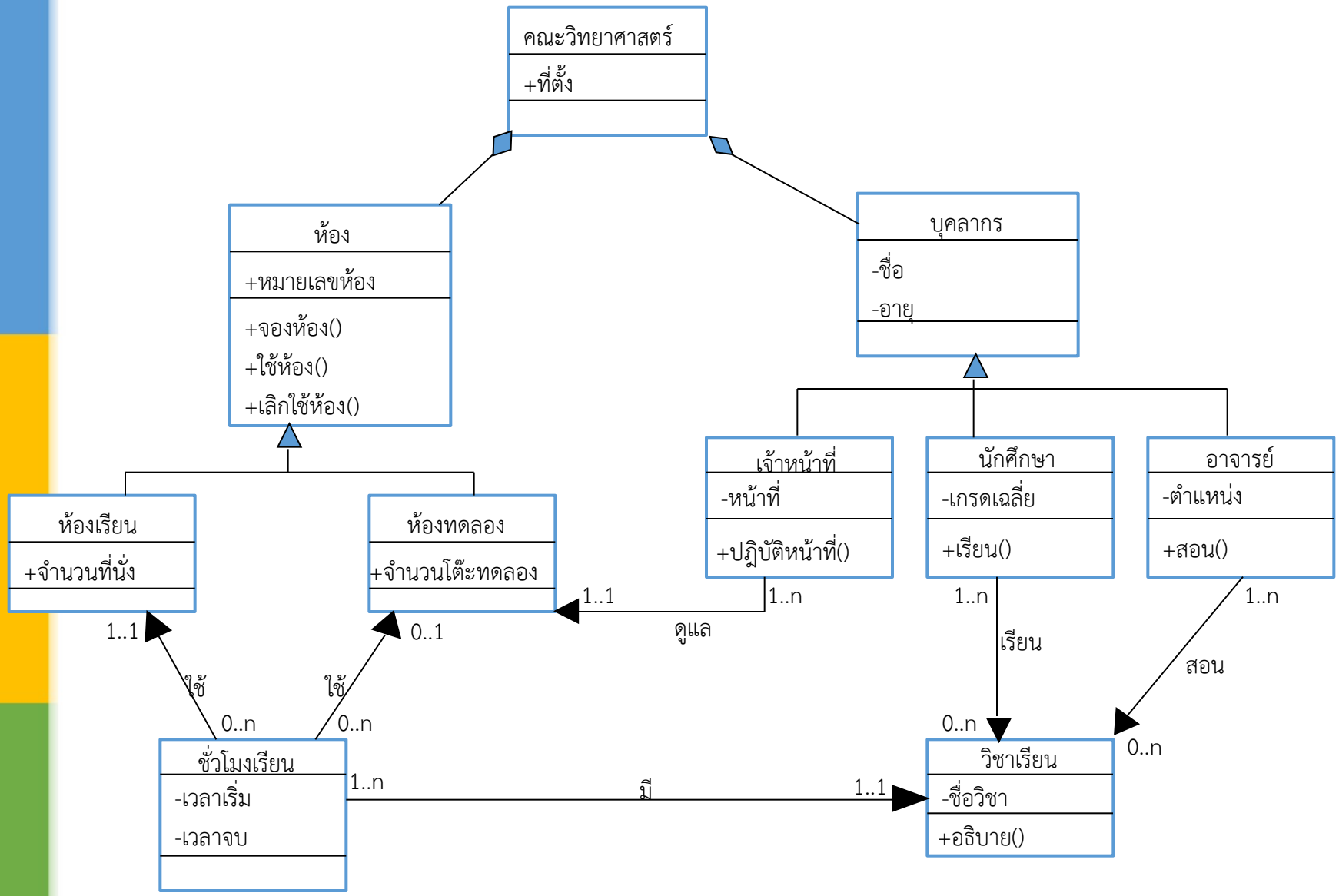
จากรูปจะเห็นว่าใน Class Diagram จะประกอบด้วย Class ที่จับต้องได้ (Tangible) และจับต้องไม่ได้ (Intangible) ซึ่ง Class ที่เป็น Tangible ได้แก่ บุคลากร(จำแนกเป็นเจ้าหน้าที่ นักศึกษาและอาจารย์) ห้องเรียนและห้องทดลอง ส่วน Class ที่เป็น Intangible ได้แก่ คณะวิทยาศาสตร์ วิชาเรียนและชั่วโมงเรียน

นอกจากนี้จาก Problem Domain ที่กำหนดให้ จะพบว่า Class Diagram จะมีความสัมพันธ์กันอันเกิดจากการใช้ Abstraction แบบต่างๆ และมีการสร้าง Class เพิ่มขึ้นด้วย เพื่อเพิ่มความสมบูรณ์ให้กับ Class Diagram ได้แก่ Class ชั่วโมงเรียน เพื่อช่วยในการโยงความสัมพันธ์ระหว่างห้องเรียนกับวิชาเรียน

สามารถปรับเปลี่ยน Class Diagram เบื้องต้นให้สมบูรณ์ขึ้นได้อีกดังรูปต่อไปนี้



จากรูปเป็นภาพแสดงการปรับปรุง Class Diagram โดยการเพิ่ม Class ห้อง ซึ่งเป็น Generalization Class หรือ Supper class ของห้องเรียนกับห้องทดลองนั่นเอง



รูป ที่ทำให้สมบูรณ์ขึ้น โดยการเพิ่ม Attributes และ Functions

Class Diagram ที่ได้ จะถูกทำ Refinement ในขั้นตอนของ Object Oriented Design (OOD) เพื่อเพิ่มความสมบูรณ์ จนกระทั่งสามารถนำมาใช้เป็นต้นแบบในการพัฒนาระบบงานในคอมพิวเตอร์ได้ในที่สุด

มาถึงจุดนี้ได้เรียนรู้วิธีการจำลองภาพของ Problem Domain ที่เป็นภาพเชิงสถิต (Static View) โดยใช้ Use Case Diagram และ Class Diagram แล้ว ในบทต่อไปจะมามองภาพของ Problem Domain ที่เป็นภาพเชิงกิจกรรม (Dynamic View) ซึ่งได้แก่ Sequence Diagram และ State Diagram