

Chapter 6

System Model



ดร.อานทิฏฐ์ นรบิน

เรียบเรียง

อ.วไลลักษณ์ วงษ์ริน



Content

- 1) Model
- 2) Structured Model
- 3) Object Model

1 - Model



Model

- ❑ แบบจำลอง คือ สัญลักษณ์ที่ใช้จำลองข้อเท็จจริงต่าง ๆ ที่เกิดขึ้นในระบบ
- ❑ ประกอบด้วยแผนภาพชนิดต่าง ๆ เพื่อแสดงให้เห็นแต่ละมุมมองของระบบ

ความสำคัญของ Model

- ❑ ใช้ในการสื่อสารกันระหว่างโครงสร้างกับวิธีการทำงานของระบบ
- ❑ ให้มองเห็นภาพรวมของระบบ
- ❑ ใช้ในการควบคุมระบบงาน
- ❑ ใช้เป็น Template ในการสร้างระบบงานจริงได้ง่าย

Reasons for Modeling



หลักการของ Model

- ❑ Model ต้องแก้ปัญหาที่ต้องการได้เป็นอย่างดี
- ❑ Model ต้องให้มุมมองตรงตามความต้องการของผู้ใช้
- ❑ Model ต้องสามารถนำไปพัฒนาระบบงานได้จริง
- ❑ ต้องใช้หลาย Model ร่วมกันในการอธิบายการทำงานต่าง ๆ

Model ตามความต้องการของอุตสาหกรรมซอฟต์แวร์

- ❑ ลดค่าใช้จ่าย (Cost)
- ❑ ลดเวลาการพัฒนาก่อนออกสู่ตลาด (time-to-Market)
- ❑ สามารถจัดการกับความซับซ้อนของปัญหาได้เสมอ ถึงแม้ว่าขอบเขตและขนาดของปัญหาจะขยายใหญ่ขึ้นก็ตาม (Scalability)

Models

Created by

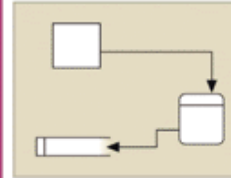
Analysis

Activities

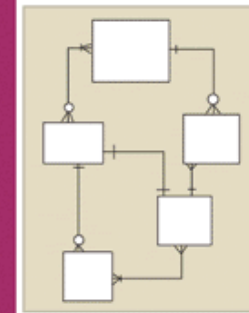
ISA-6-System Model

1 buy new car
2 sell car
3 get car serviced
4 make payment
5 trade in car

Event list



Data flow diagram (DFD)



Entity-relationship diagram (ERD)

dataflow 1 =
element 1+
element 2+
element 3

Data flow definition

element 1 =
description
data type
validation rules

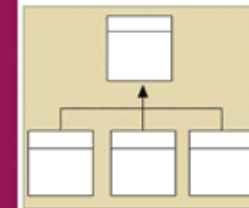
Data element definition

do this
If ...
else ...
while x
do that
do the other

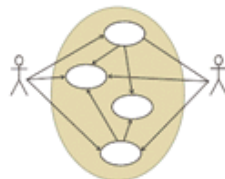
Process description/
structured English/
action diagram



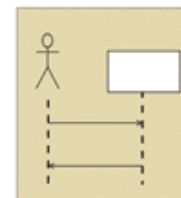
Location diagram



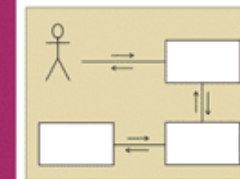
Class diagram



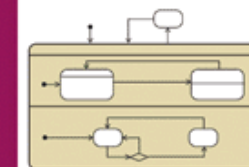
Use case diagram



Sequence diagram

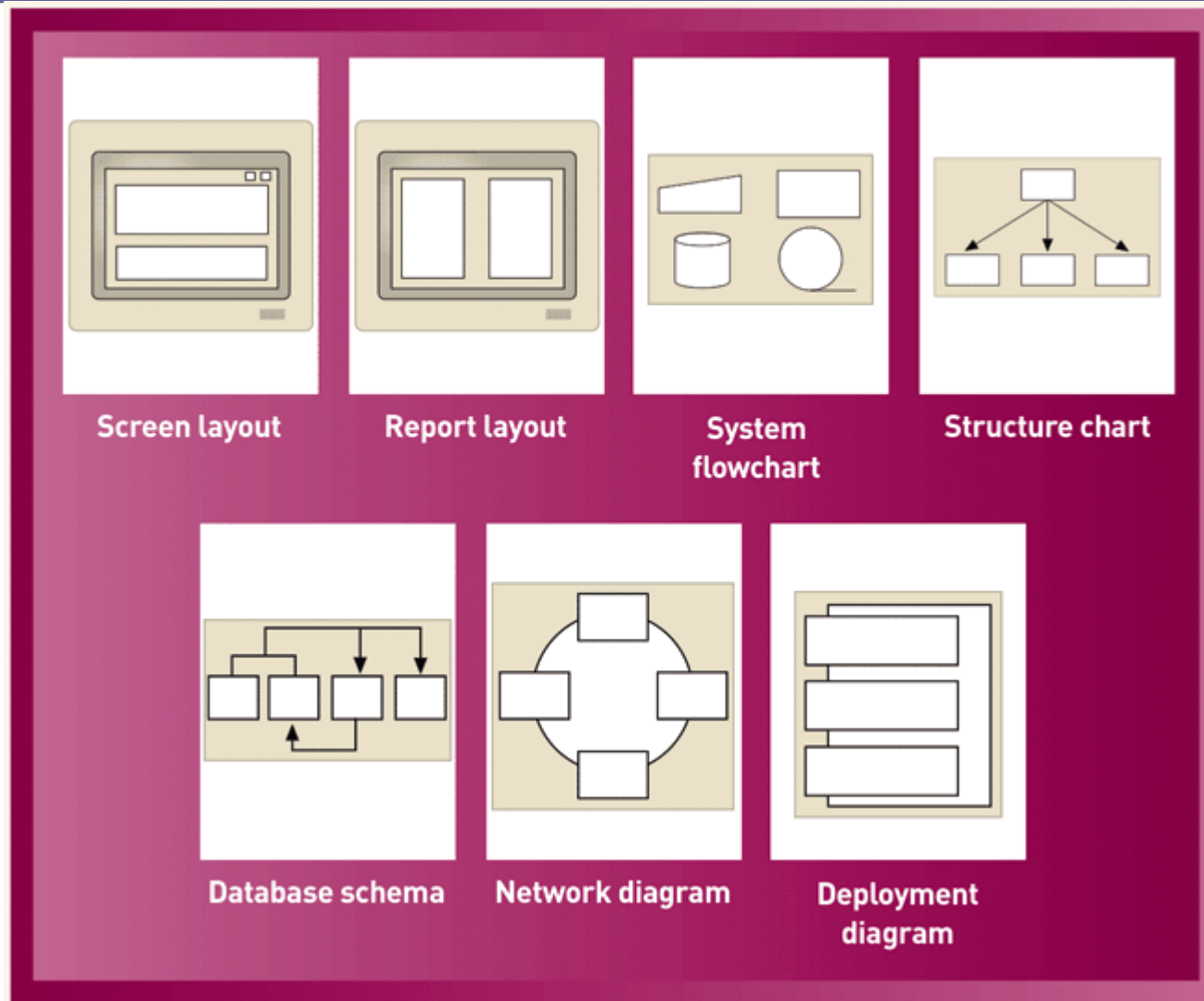


Communication diagram

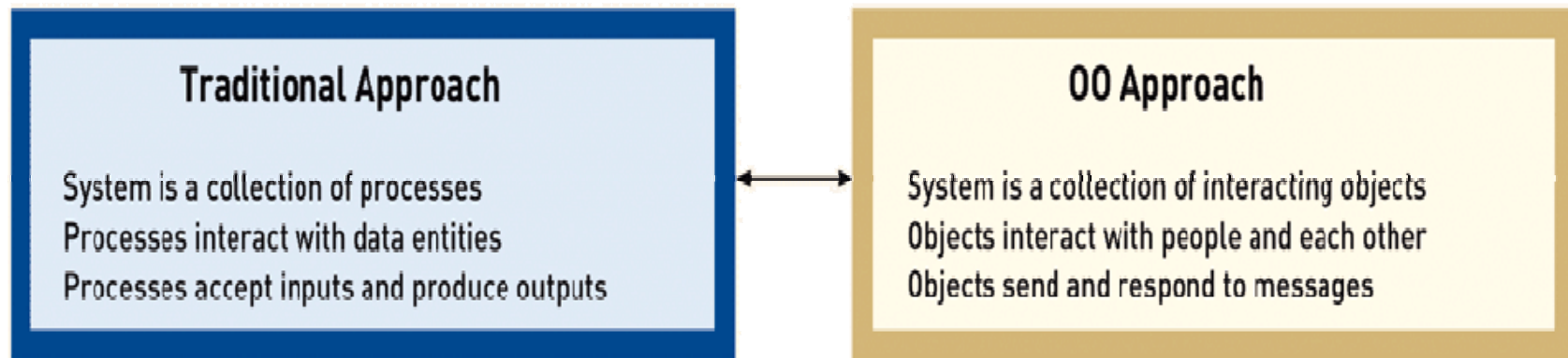


State machine diagram

Models Used in Design



Traditional versus Object-Oriented Approaches



2 – Structured Model



แบบจำลองตามแนวทางเชิงโครงสร้าง

2.1 แบบจำลองกระบวนการทำงานของระบบ (Process Model)

ใช้จำลองกระบวนการทำงานของระบบโดยใช้ Data Flow Diagram (DFD)

2.2 แบบจำลองข้อมูล (Data Model)

ใช้จำลองข้อมูลทั้งหมดในระบบโดยใช้ Entity Relationship Diagram (ERD)

Data Flow Diagram (DFD)

- ❑ เป็นแผนภาพแสดงให้เห็นทิศทางการไหลของข้อมูลภายในระบบ จากกระบวนการ (Process) ทำงานหนึ่ง ไปยังอีกกระบวนการทำงานหนึ่ง หรือไปยังส่วนที่เกี่ยวข้อง
- ❑ สามารถนำ DFD ไปออกแบบฐานข้อมูลได้

Layers of DFD

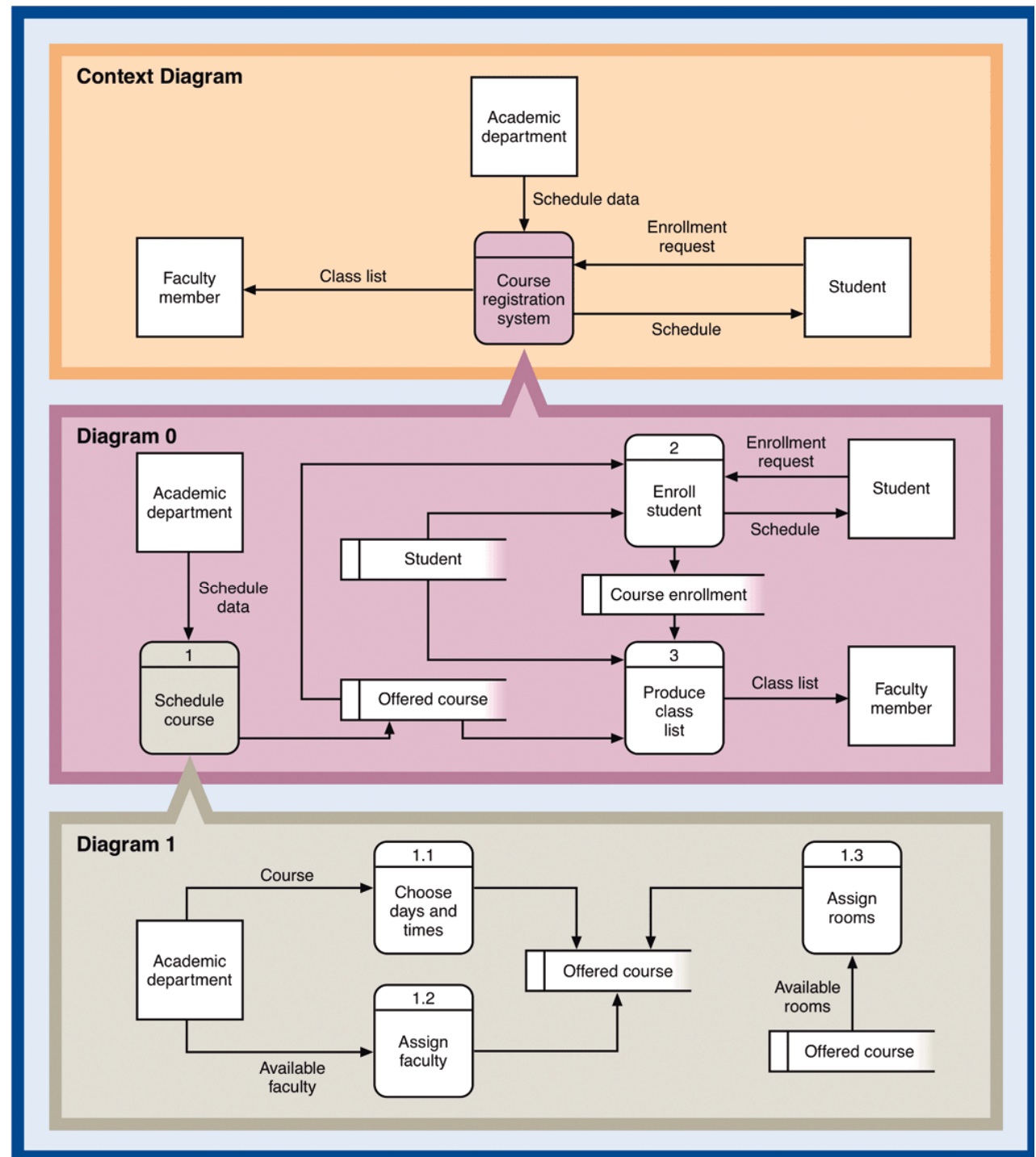
Abstraction for

Course

Registration

System

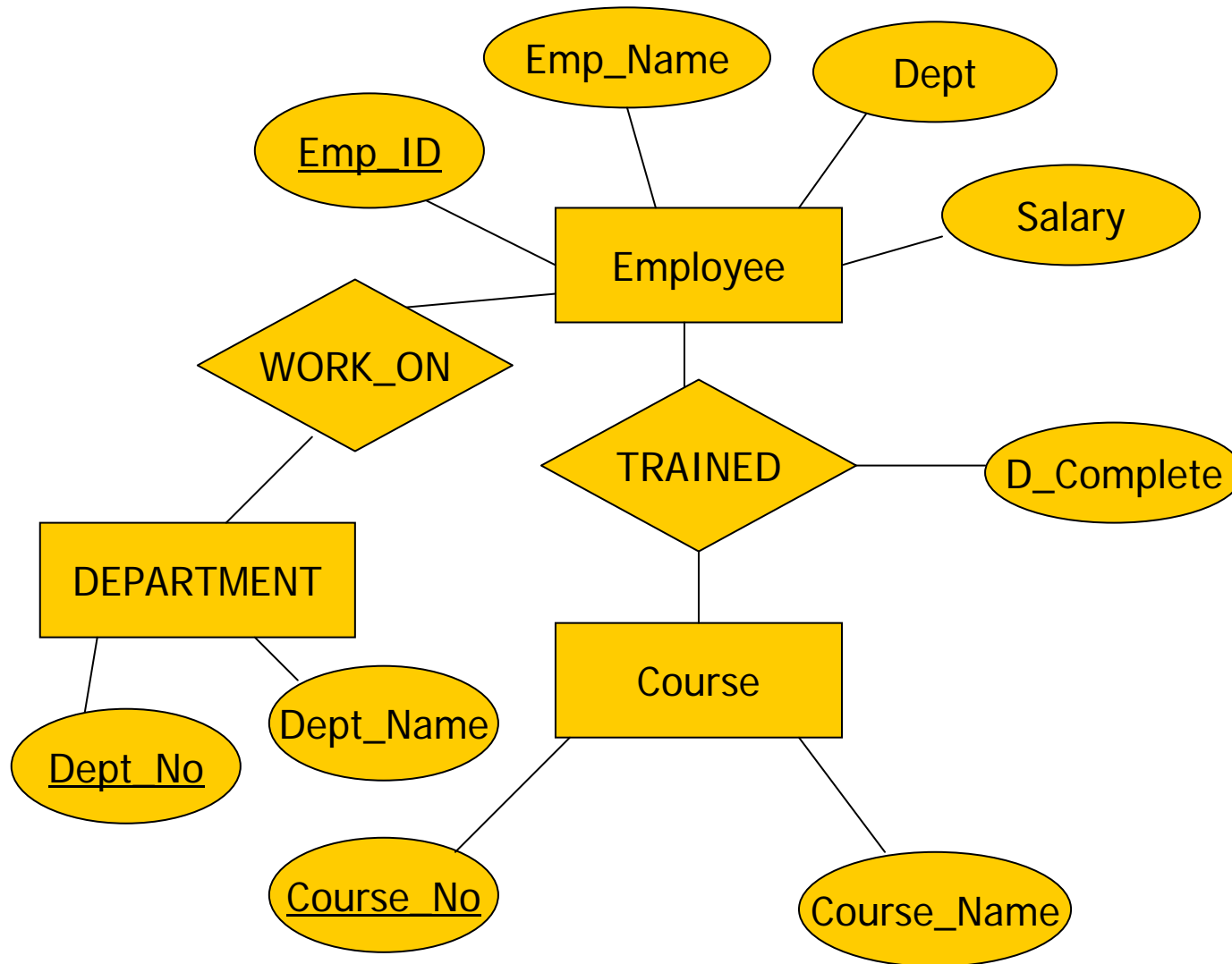
ISA-6-System Model



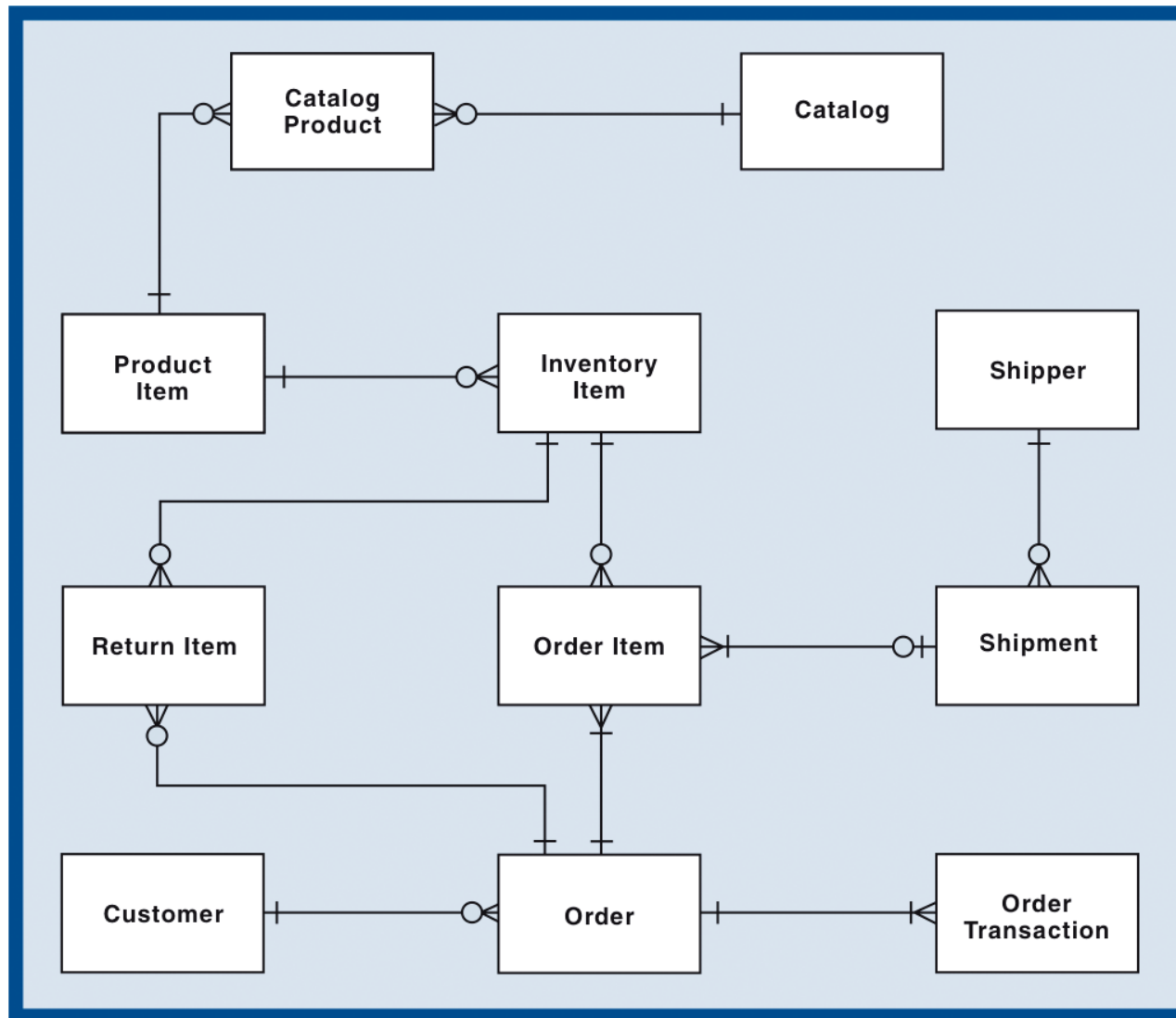
Entity Relationship Diagram (ERD)

- ❑ เป็นแผนภาพที่เป็นเครื่องมือสำหรับจำลองข้อมูล
- ❑ ประกอบด้วย Entity และความสัมพันธ์ระหว่างข้อมูล (Relationship) ที่เกิดขึ้นทั้งหมดในระบบ
- ❑ ทุก ๆ Entity จะมี Attribute เป็นสิ่งที่บ่งบอกคุณสมบัติของ Entity นั้น

Example : Entity Relationship Diagram (Chen Model)



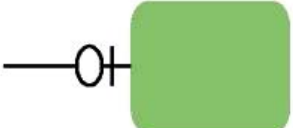

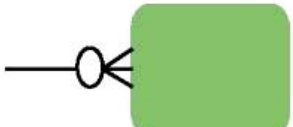



Example : Entity Relationship Diagram (Crow's Foot Model)

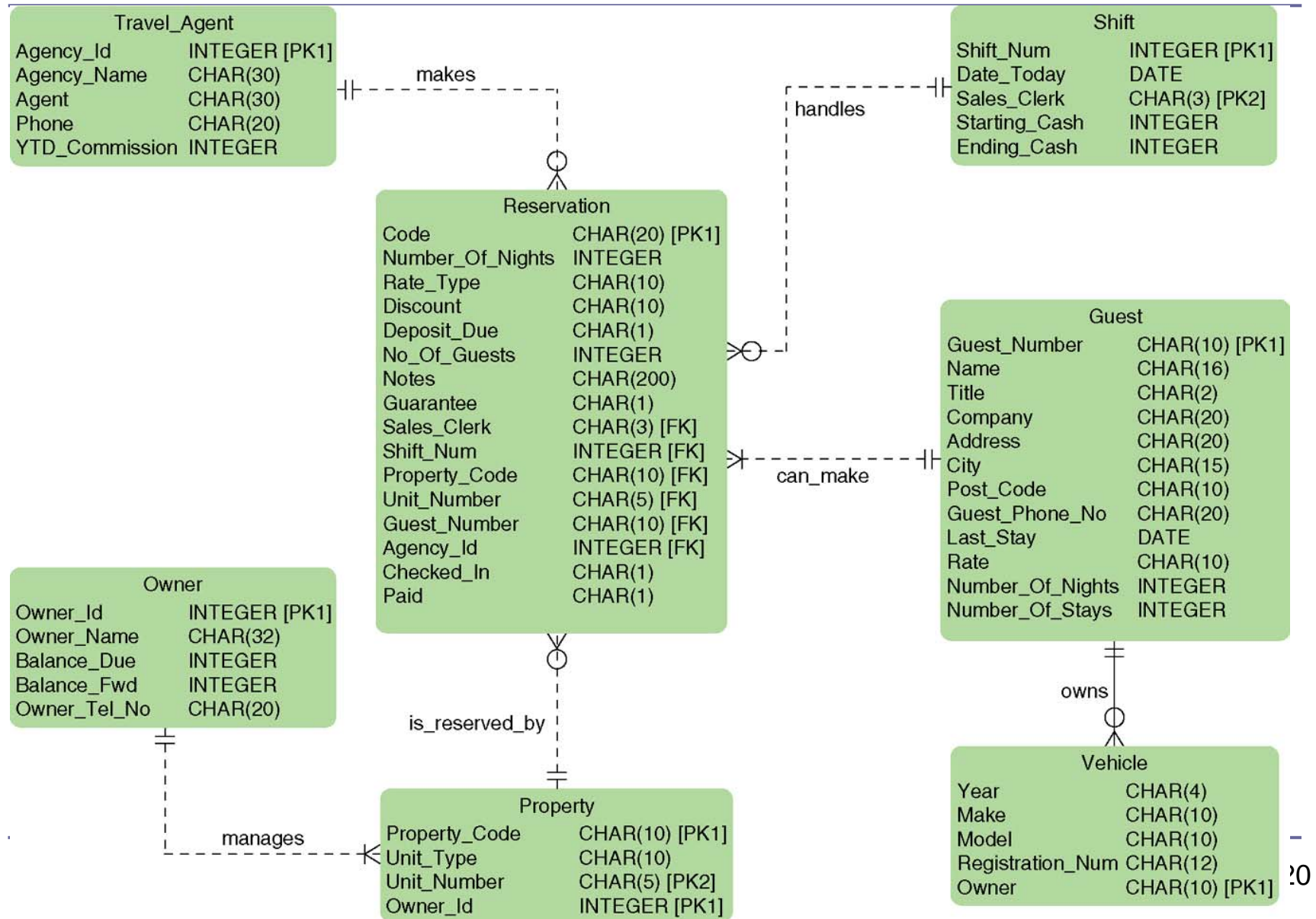


Cardinality Notations

ISA-6-System Model

CARDINALITY INTERPRETATION	MINIMUM INSTANCES	MAXIMUM INSTANCES	GRAPHIC NOTATION
Exactly one (one and only one)	1	1	 — or — 
Zero or one	0	1	
One or more	1	many (>1)	
Zero, one, or more	0	many (>1)	
More than one	>1	>1	

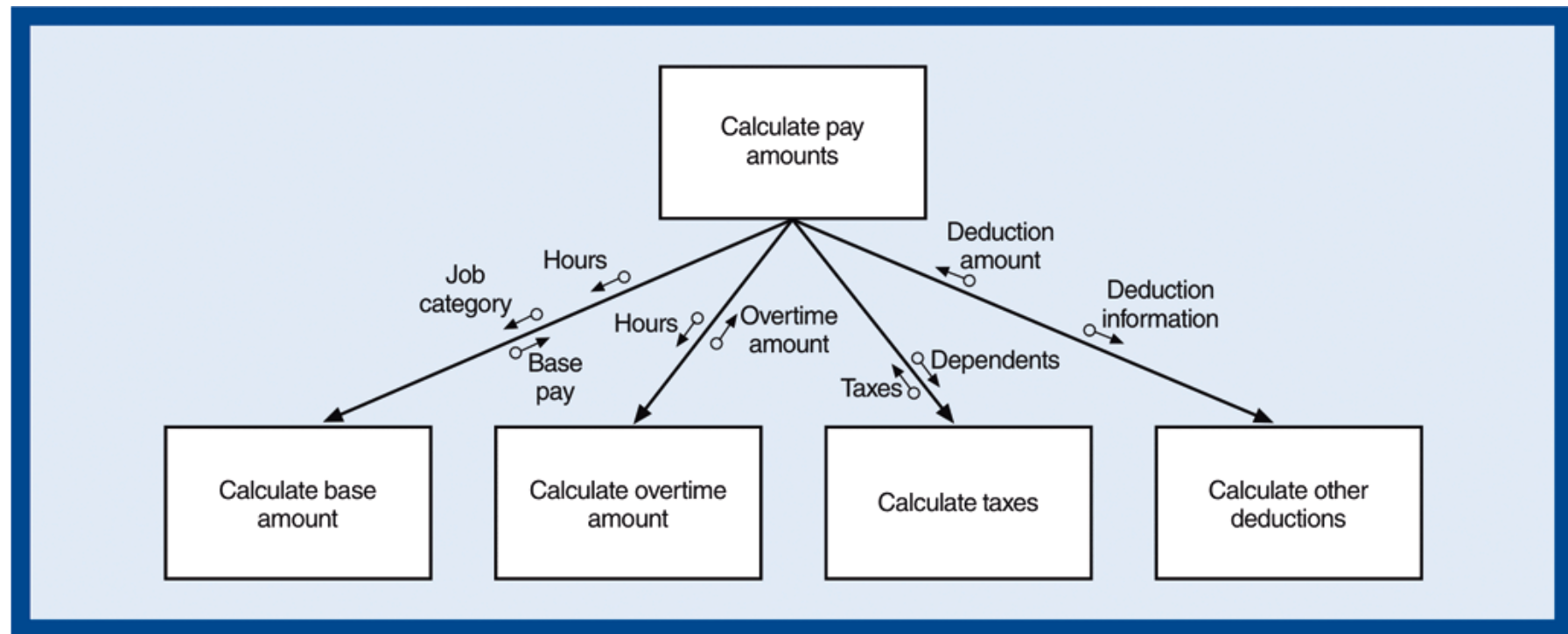
Example : Physical Entity Relationship Diagram



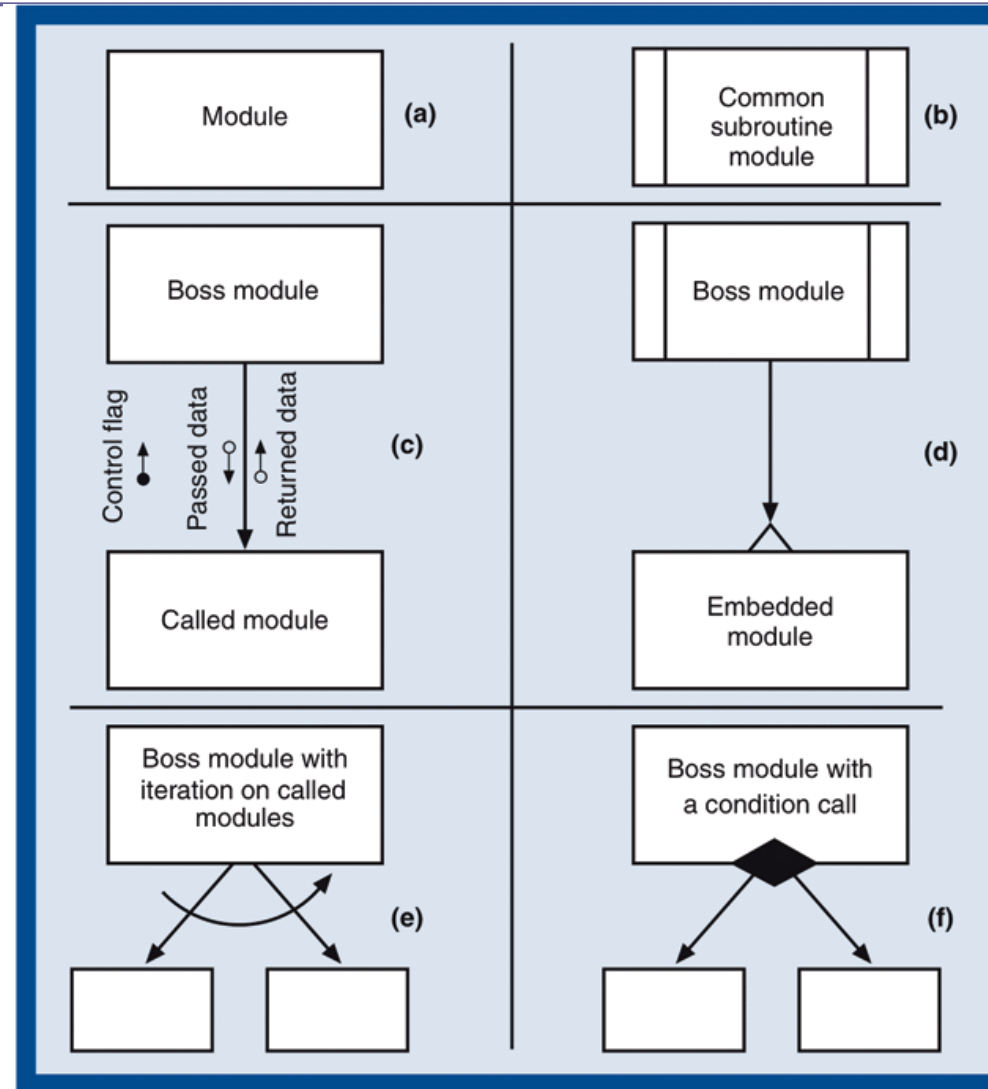
Structure Chart

- ❑ เป็นแผนผังแบบลำดับชั้นที่แสดงให้เห็นความสัมพันธ์ระหว่างฟังก์ชันของโปรแกรม (Module)
- ❑ แต่ละโมดูลจะมีการเรียกใช้ข้อมูลตามลำดับชั้น โมดูลที่อยู่ระดับบนสุดจะเรียกใช้โมดูลที่อยู่ระดับล่าง
- ❑ มีโมดูลหลักอยู่ด้านบนเพียงโมดูลเดียว
- ❑ โดยทั่วไป โมดูลที่อยู่ระดับล่าง ๆ จะประกอบด้วยอัลกอริธึมและลอจิกของโปรแกรมจำนวนมาก เพื่อใช้ประกอบการทำงานของโมดูลในระดับบนให้สมบูรณ์

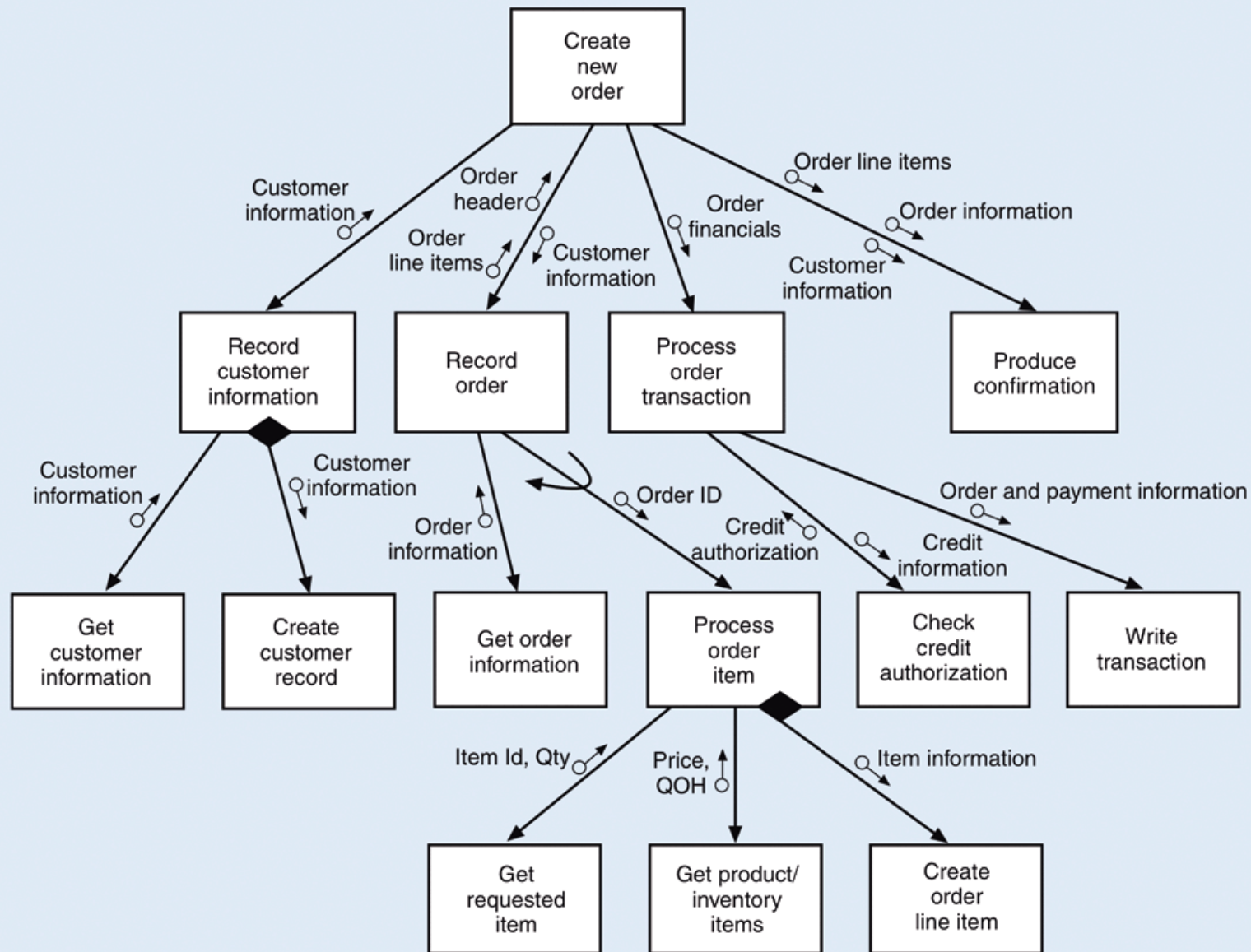
A Simple Structure Chart for the Calculate Pay Amounts Module



Structure Chart Symbols



The Structure Chart for the Create New Order Program



3 – Object Model



แบบจำลองเชิงวัตถุ

- ❑ เป็นการอธิบายการใช้ระบบสารสนเทศในการกำหนดสิ่งต่าง ๆ โดยเรียกลักษณะเหล่านั้นว่า วัตถุหรือออบเจกต์ (Objects)
- ❑ ตัวอย่างของออบเจกต์ ได้แก่ คน สถานที่ เหตุการณ์ รายการเปลี่ยนแปลงหรือทรานแซคชัน (Transaction)
- ❑ ยกตัวอย่างเช่น เมื่อคนไข้มีนัดหมายเพื่อไปพบแพทย์ คนไข้เป็นออบเจกต์ แพทย์เป็นออบเจกต์ และการนัดหมายก็เป็นออบเจกต์

แบบจำลองเชิงวัตถุ (ต่อ)

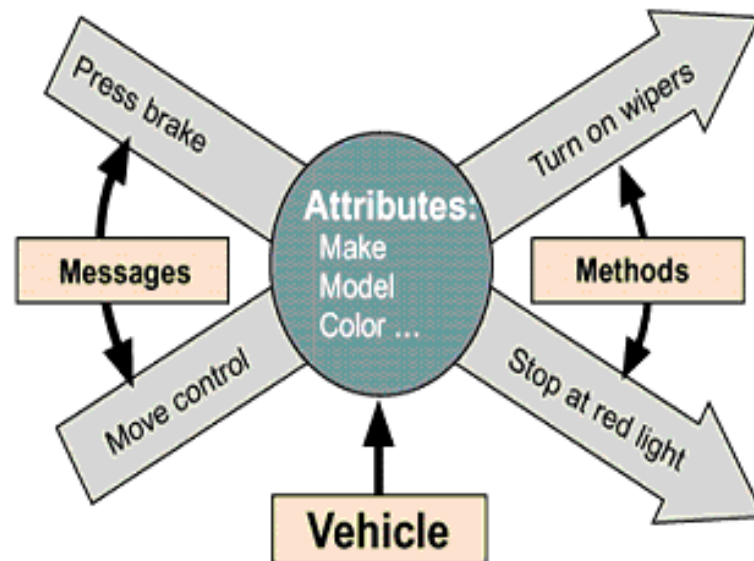
- ❑ เมื่อถึงระยะ Implementation นั้น SA และโปรแกรมเมอร์จะทำการแปลงออกแบบเจ็กต์ให้เป็นส่วนจำเพาะของรหัสชุดคำสั่ง
- ❑ การใช้วิธีการแยกเป็นส่วนจำเพาะหรือโมดูลาร์ (Modular) จะช่วยประหยัดเงินและเวลา เนื่องจากสามารถถูกใช้อย่างเต็มที่ สามารถถูกตรวจสอบ และสามารถนำเอากลับมาใช้ใหม่ได้อีก

แบบจำลองเชิงวัตถุ (ต่อ)

- ❑ ออบเจกต์จะมีแอตทริบิวต์ (Attribute) ซึ่งแสดงคุณสมบัติที่อธิบายถึงลักษณะของออบเจกต์นั้น ๆ เช่น แอตทริบิวต์ของรถยนต์ คือ ยี่ห้อ แบบ และสี
- ❑ ออบเจกต์ยังมี เมธอด (Method) ซึ่งเป็นวิธีการปฏิบัติตามที่ได้รับเมสเสจ (Message) เช่น รถยนต์แสดงเมธอดที่เรียกว่า เปิดที่ปิดน้ำฝนเพื่อส่งเมสเสจให้มีการเคลื่อนไหวของที่ปิดน้ำฝน

ตัวอย่าง : รถยนต์

- ❑ รถยนต์ ซึ่งเป็นออบเจกต์ มีแอตทริบิวต์ เช่น ยี่ห้อ แบบ สี สามารถแสดงเมทอดตามเมสเสจ ที่ได้รับ เช่น เมื่อได้รับเมสเสจ ให้เหยียบเบรค เมทอดคือ การหยุดที่ไฟแดง หรือการให้หมวกกันปัดน้ำฝนเพื่อเปิดที่ปัดน้ำฝน



UML

- ❑ เกิดขึ้นจากแรงผลักดันของ Grady Booch, James Rumbaugh และ Ivar Jacobson
- ❑ ทั้ง 3 คนได้นำโมเดลของตนมารวมกันและมีการพัฒนาปรับปรุงจนกลายเป็นภาษาที่ใช้ในการสร้างโมเดลภาษาใหม่ เรียกว่า UML (The Unified Modeling Language)
- ❑ ใช้ได้กับทุกแนวทางในการพัฒนาระบบเชิงวัตถุ

UML (ต่อ)

- ❑ เป็นภาษาเพื่อใช้อธิบายโมเดลต่าง ๆ
- ❑ เป็นภาษาที่มีลักษณะของ map language (ภาษาที่ใช้กราฟิกเป็นสัญลักษณ์ ซึ่งเป็นภาษาที่เหมาะสมสำหรับผู้ทำงานบางกลุ่ม เช่น นักออกแบบ (Designer) หรือนักพัฒนาระบบคอมพิวเตอร์ (Developer) เป็นต้น)
- ❑ มีผู้เข้าใจสับสนว่า UML เป็นเพียงการใช้สัญลักษณ์สร้างไดอะแกรมเพื่ออธิบายระบบงาน (ซึ่งเป็นความเข้าใจที่ไม่ถูกต้อง)

UML (Cont.)

- ❑ แท้จริงแล้ว UML มีลักษณะของ metamodel (เป็นโมเดลที่ใช้อธิบายโมเดลอื่นๆ)
- ❑ เป็นภาษามาตรฐานสำหรับสร้างพิมพ์เขียวให้กับระบบงานเช่น สร้างมุมมอง กำหนดรายละเอียด สร้างระบบงาน และจัดทำเอกสารอ้างอิงในระบบงาน
- ❑ เป็นภาษาเหมาะสำหรับระบบงานระดับกิจการ, web-based application และ real time system

Class

- ❑ คือ กลุ่มของออบเจกต์ที่มีลักษณะเหมือนกัน
- ❑ เช่น Cruisers อยู่ในคลาสที่เรียกว่า CAR โดยมีอินสแตนส์ (Instance) ที่มีลักษณะเฉพาะของคลาสนั้น ๆ เช่น รถสีแดง 2001 Cruiser เป็นอินสแตนส์ของคลาสที่เรียกว่า CAR ในร้านจำหน่ายรถยนต์
- ❑ จะพบอินสแตนส์หลาย ๆ แบบในแต่ละคลาส เช่น คลาสรถยนต์นั่งส่วนบุคคล คลาสรถปิกอัพ หรือคลาสรถบรรทุก เป็นต้น

Class & Object

□ Class

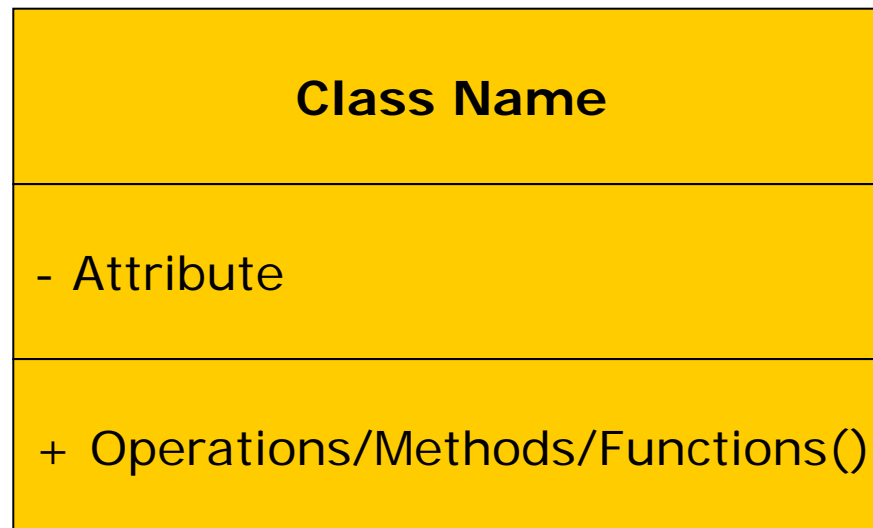
คือกลุ่มของวัตถุที่มีคุณสมบัติ (Properties or Attributes) การกระทำ (Behavior or Operation) ร่วมกัน

□ Object

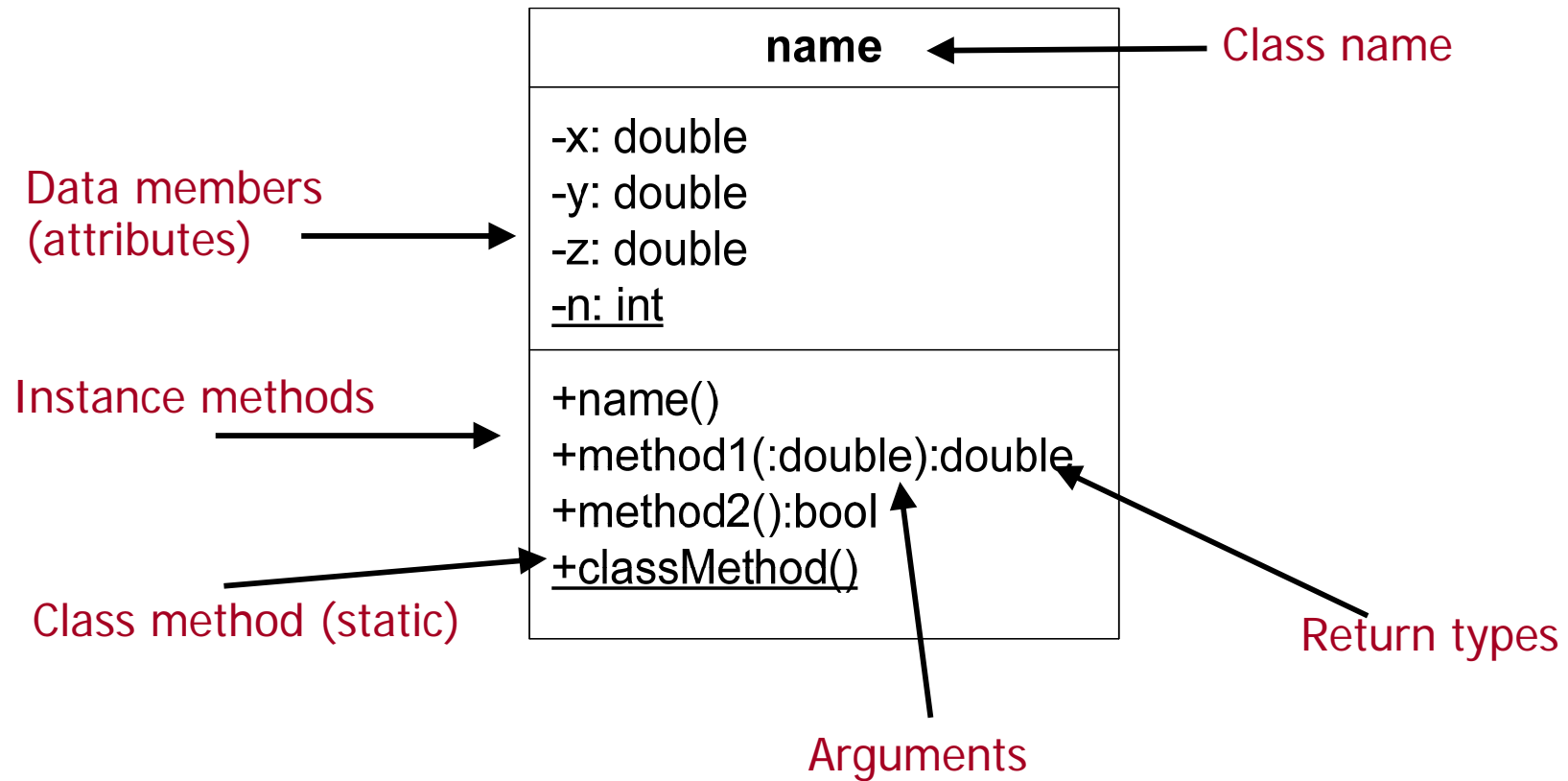
คือ ผลผลิตของคลาส เป็นกลุ่มของคุณสมบัติที่บอกขอบเขตชัดเจน โดยมีเอกลักษณ์ สถานะ (State) และพฤติกรรม (Behavior) ซ่อนอยู่

องค์ประกอบของ Class

- ❑ Class Name
- ❑ Attributes
- ❑ Operations/Methods/Functions



UML Class



Views of Stored Data

Object-Oriented	Entity-Relationship	Relational Database
Class	Entity Type	Table
Object	Entity Instance	Row
Attribute	Attribute	Column

แบบจำลองตามแนวทางเชิงวัตถุ

3.1 Structure Diagram

3.2 Behavioral Diagram

3.1 Structure Diagram

- ❑ เป็นกลุ่มแผนภาพที่แสดงให้เห็นโครงสร้างแบบ Static ของระบบ (คือส่วนที่ไม่มีการเปลี่ยนแปลงหรือเคลื่อนไหวแม้ว่าจะมีเหตุการณ์ใด ๆ เกิดขึ้น)
- ❑ ได้แก่
 - Class Diagram
 - Object Diagram
 - Component Diagram
 - Deployment Diagram

3.2 Behavioral Diagram

- ❑ เป็นกลุ่มแผนภาพที่แสดงให้เห็นกิจกรรมของระบบ (Dynamic)
- ❑ เมื่อมีเหตุการณ์ใด ๆ เกิดขึ้นกับระบบแผนภาพจะแสดงให้เห็นพฤติกรรมต่าง ๆ
- ❑ แสดงให้เห็นถึงความสามารถของระบบที่ดำเนินการในหน้าที่บางอย่างได้

Behavioral Diagram (ต่อ)

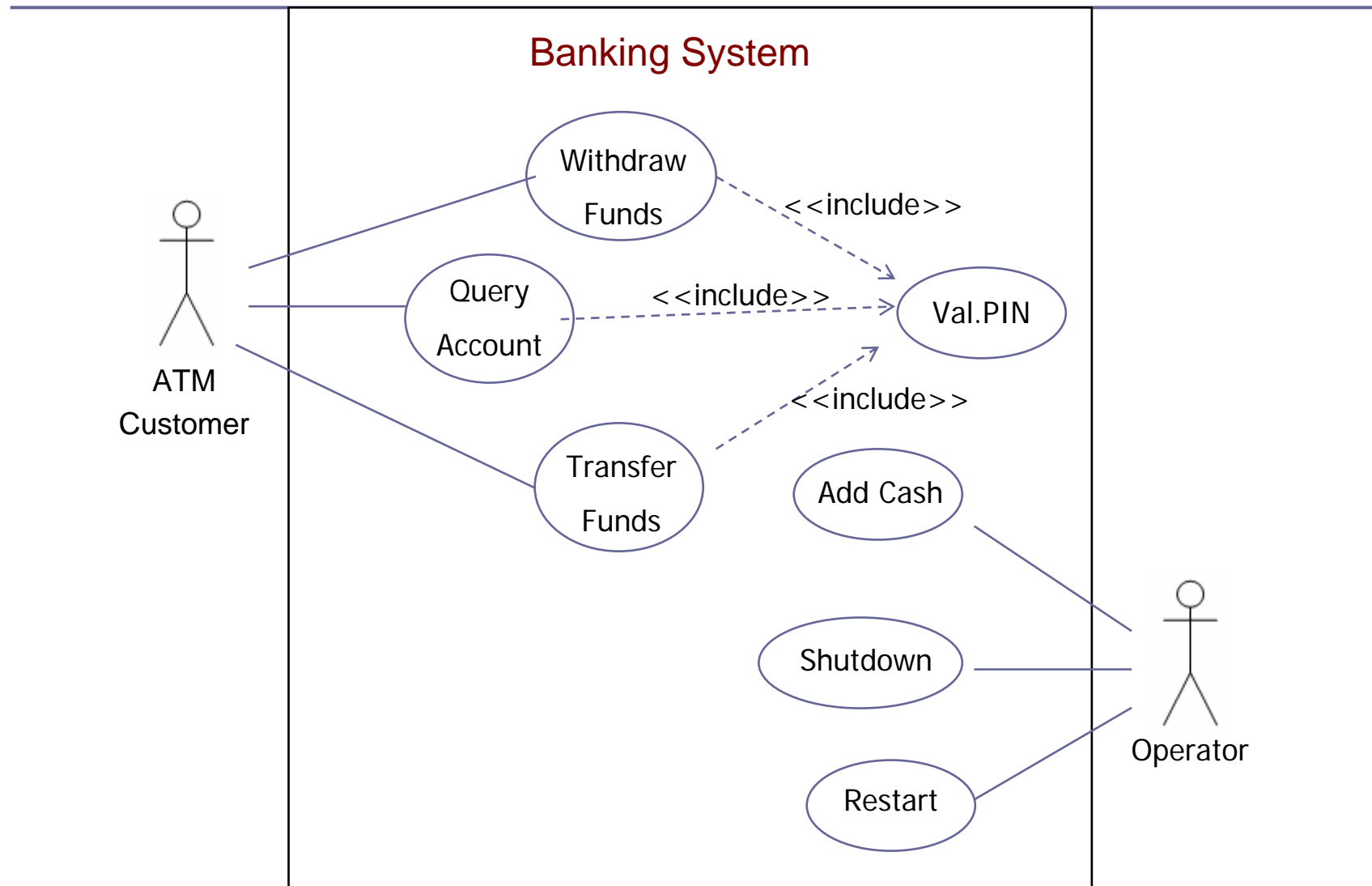
□ ได้แก่

- Use Case Diagram
- Sequence Diagram
- Collaboration Diagram
- Statechart Diagram
- Activity Diagram

5.1 Use Case Diagram

- ❑ แสดงขั้นตอนการทำงานที่สำคัญของระบบ (Use Case) หรือแสดงหน้าที่และงานที่ระบบจะต้องปฏิบัติ เพื่อตอบสนองต่อผู้กระทำ (Actors) ต่อระบบ
- ❑ แสดงถึงบทบาท (roles) ต่าง ๆ ของ user และบทบาทเหล่านั้นเกี่ยวข้องกับระบบ (system) อย่างไร
- ❑ ทราบถึงผู้ใช้งานในแต่ละส่วนของระบบ

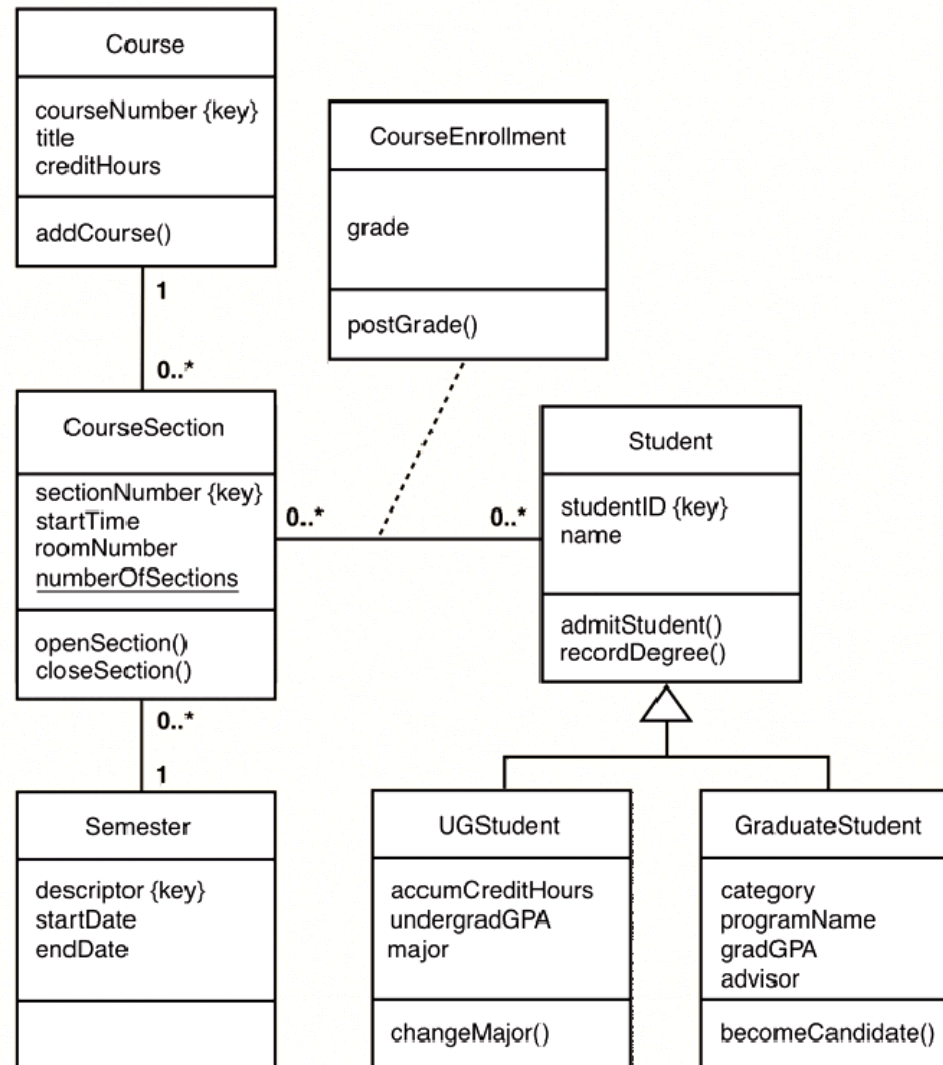
Example : Banking System



5.2 Class Diagram

- ❑ แสดงโครงสร้างที่เป็น static view ของระบบ
- ❑ แสดงกลุ่มของคลาส โครงสร้างของคลาส และความสัมพันธ์ระหว่างคลาส
- ❑ จะสามารถสร้างคลาสไดอะแกรมได้ จะต้องค้นหาออบเจกต์ใน Use Case ก่อน

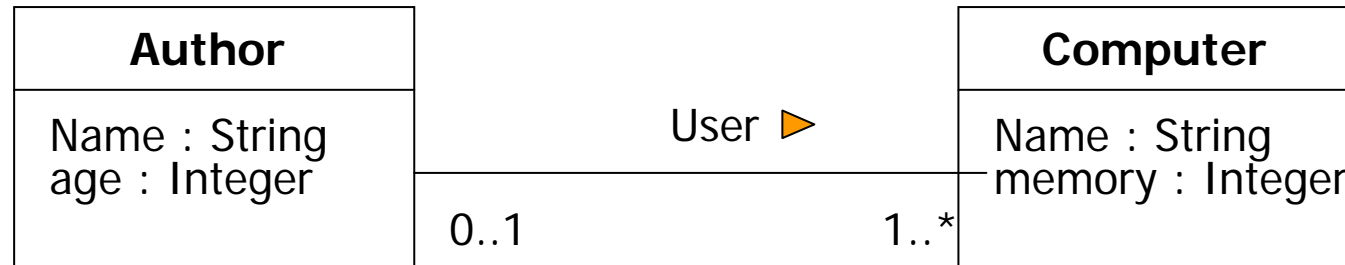
Example : Course Enrollment System



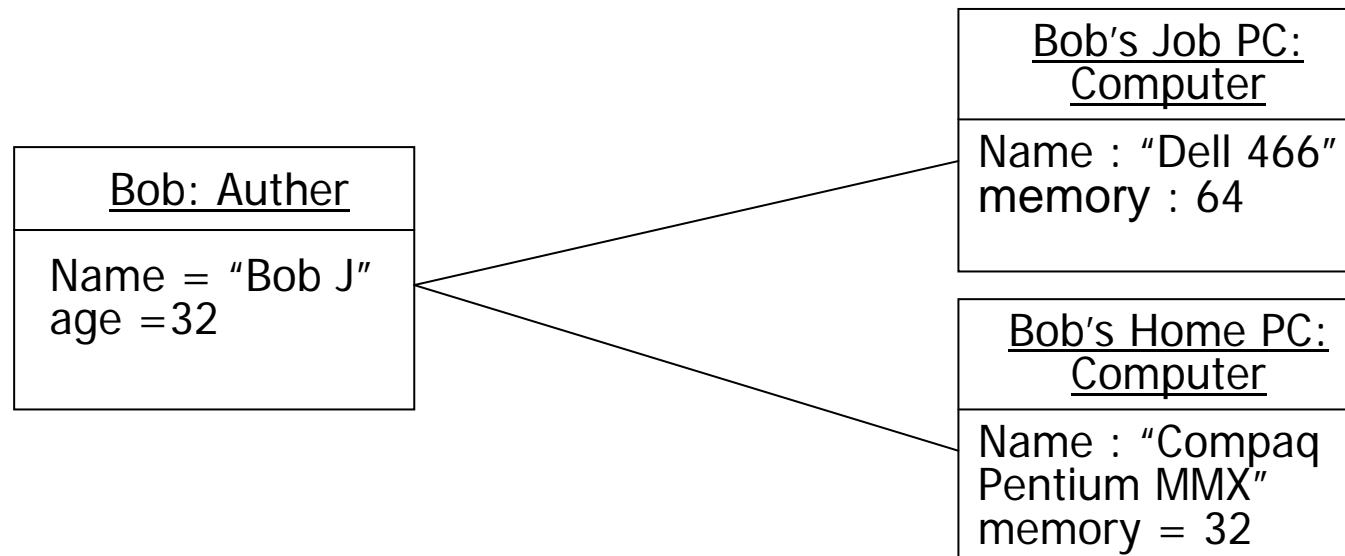
5.3 Object Diagram

- ❑ ใช้ในการแสดงกลุ่มของออบเจกต์และความสัมพันธ์ระหว่างออบเจกต์ที่เกิดขึ้นในคลาสต่าง ๆ ของ Class Diagram
- ❑ แสดง objects ที่เกี่ยวข้อง ณ เวลาใดเวลาหนึ่งของการ execute program

Example : Object Diagram



Class Diagram

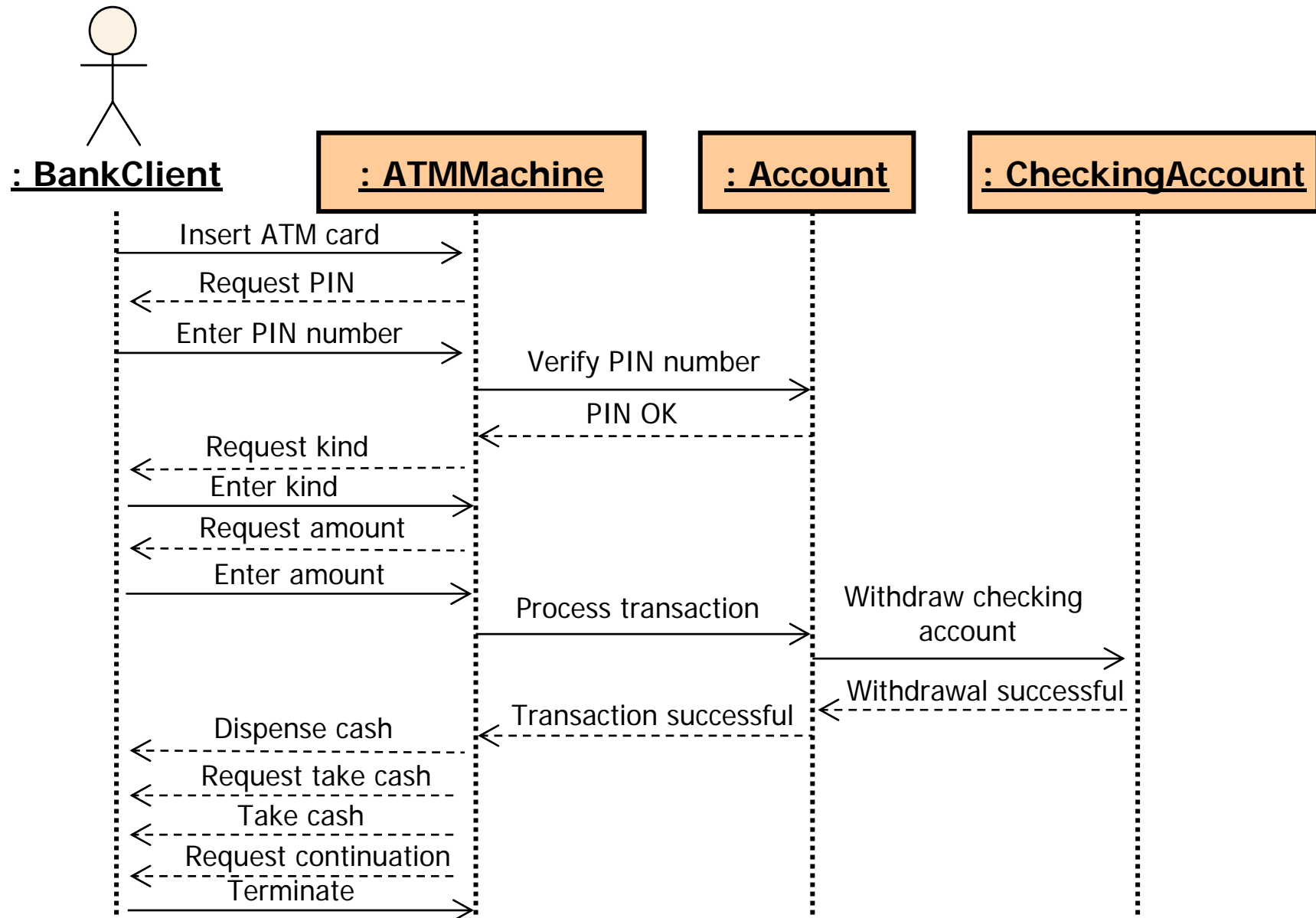


Object Diagram

5.4 Sequence diagram

- ❑ แสดงการปฏิสัมพันธ์ระหว่างออบเจกต์
- ❑ แสดงให้เห็นการส่ง Message ระหว่างออบเจกต์ตามลำดับเวลาที่เกิดเหตุการณ์ขึ้น

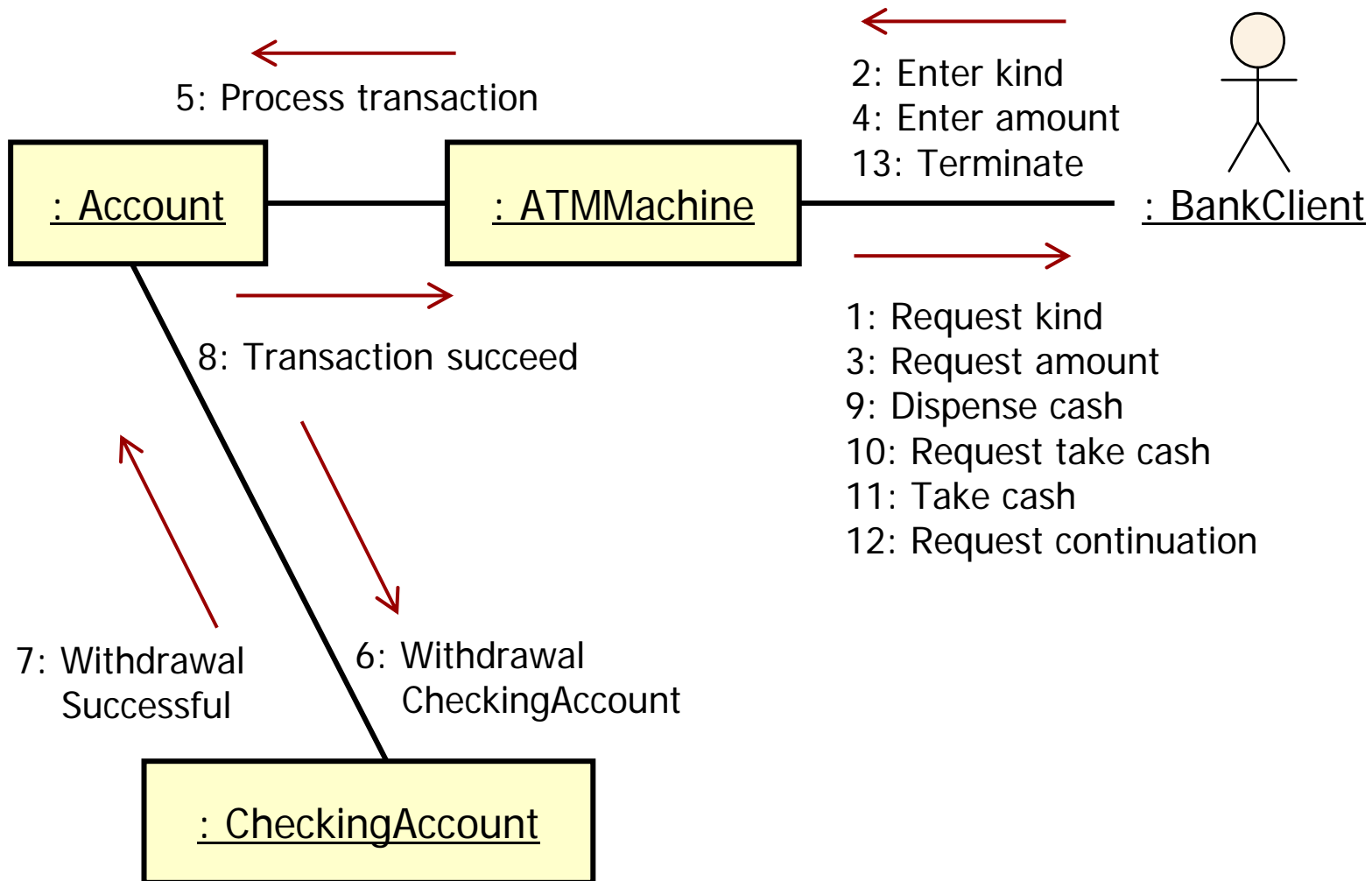
Example : ATM Withdraw Checking



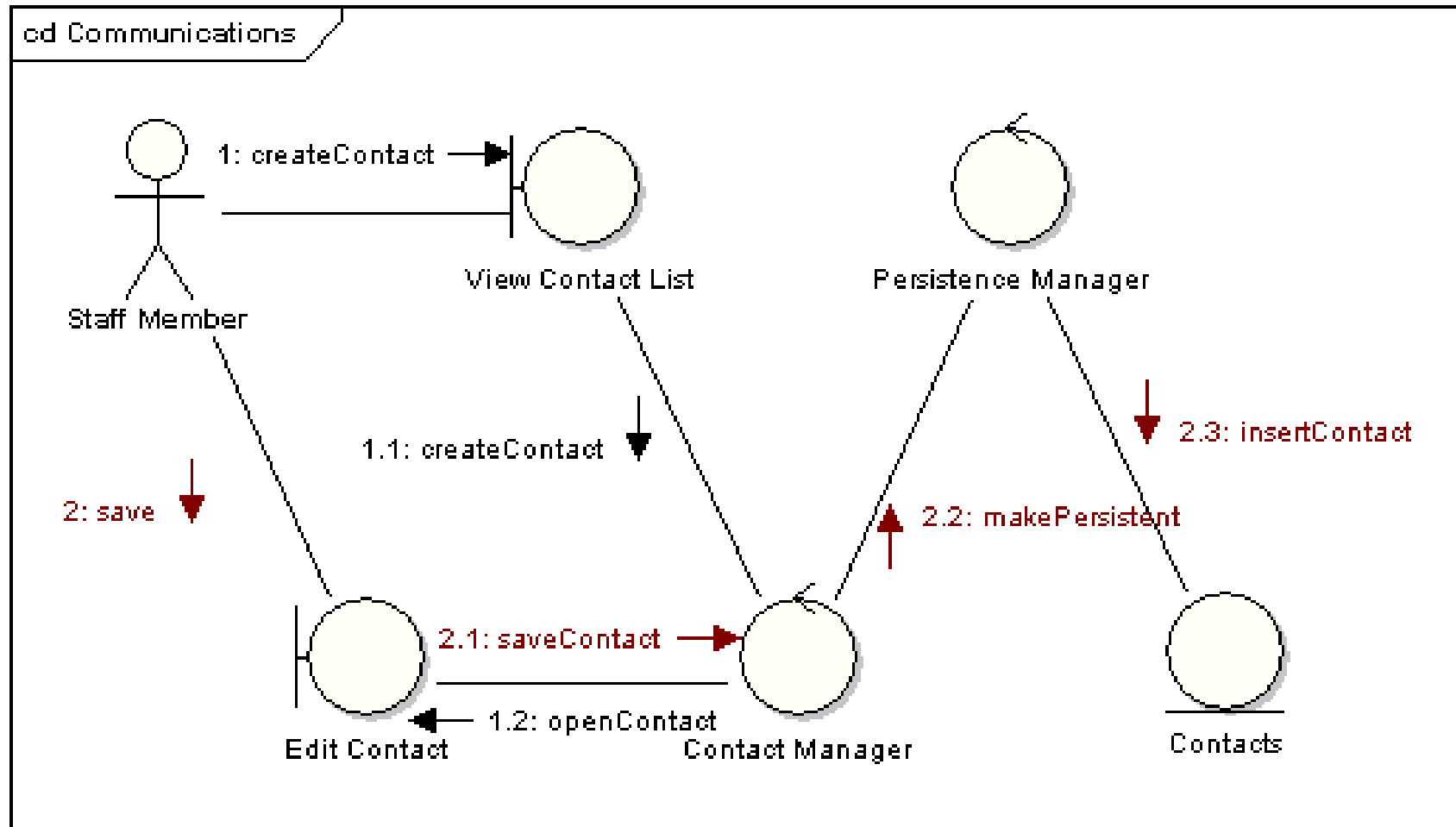
5.5 Collaboration Diagram

- ❑ ใน UML 1.4 หรือต่ำกว่าเรียกว่า Collaboration diagram
- ❑ ใน UML 2.0 เรียกว่า Communication diagram
- ❑ แสดงการปฏิสัมพันธ์ระหว่างออบเจกต์เช่นเดียวกับ Sequence Diagram
- ❑ ไม่แสดงแกนเวลาอย่างชัดเจน ยกเว้นการโต้ตอบกันระหว่างออบเจกต์
- ❑ เน้นการอธิบายโครงสร้างของออบเจกต์ว่ามีการรับส่งข้อความอย่างไร

Example : ATM withdraw checking



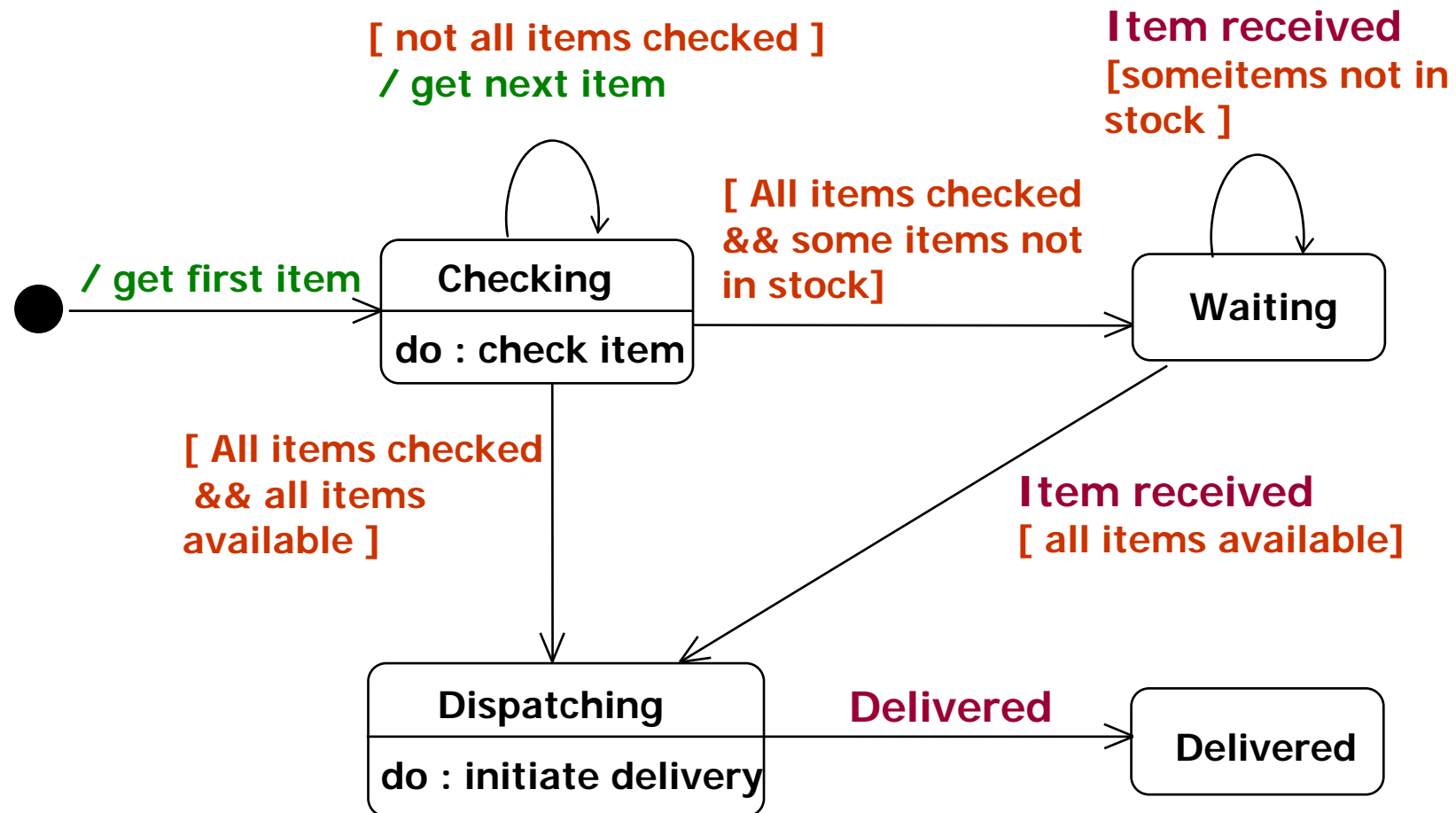
Example : Communication diagram



5.6 State Diagram (Statechart Diagram)

- ❑ เป็นไดอะแกรมที่ใช้แสดงการเปลี่ยนสถานภาพ (State) ของออบเจกต์ตั้งแต่เริ่มต้นจนถึงสิ้นสุดการเปลี่ยนแปลงในรอบ ๆ หนึ่ง (1 sequence)
- ❑ ไดอะแกรมจะแสดงสถานะของแต่ละออบเจกต์จะประกอบด้วยสถานะต่าง ๆ ที่สามารถเกิดขึ้นได้
- ❑ เมื่อเวลาผ่านไปหรือมีเหตุการณ์บางอย่างเกิดขึ้น ย่อมทำให้เกิดการเปลี่ยนสถานะหรือเปลี่ยนพฤติกรรมหรือเปลี่ยนการปฏิบัติของออบเจกต์

Example : An Order System

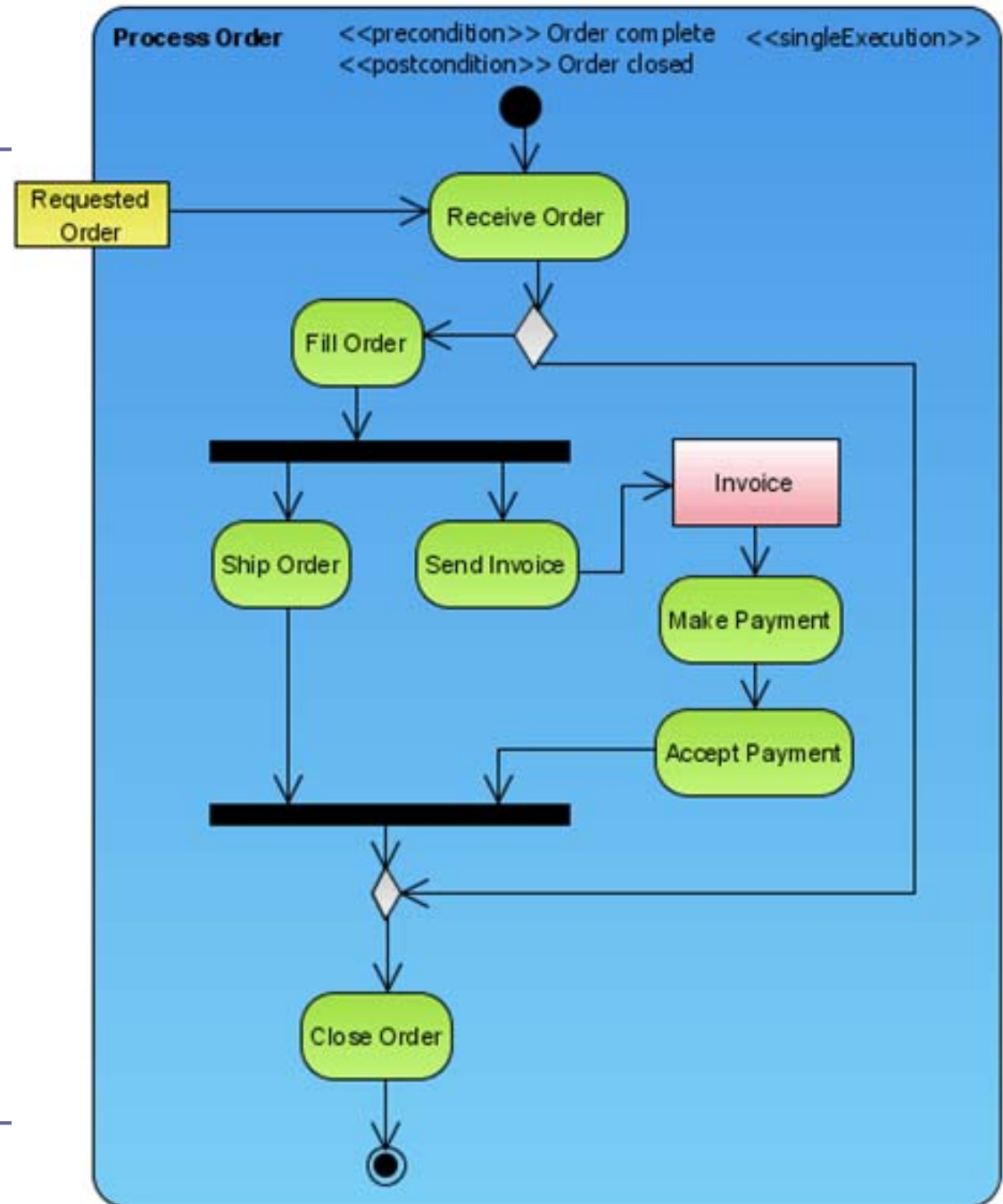


5.7 Activity Diagram

- ❑ แสดงลำดับกระแสนะของกิจกรรมของการทำงานใด ๆ เช่น ขั้นตอนการทำ operation ขั้นตอนการลงทะเบียน
- ❑ แสดงขั้นตอนการทำงานของระบบ เช่นเดียวกับ State Diagram, Sequence Diagram และ Collaboration Diagram
- ❑ เน้นที่กิจกรรมย่อยของออบเจกต์
- ❑ ต่างจาก State Diagram ตรงที่ Activity Diagram จะแสดงการเปลี่ยนกิจกรรม โดยมีคลาสตั้งแต่ 1 คลาสขึ้นไปที่ทำกิจกรรม

Example :

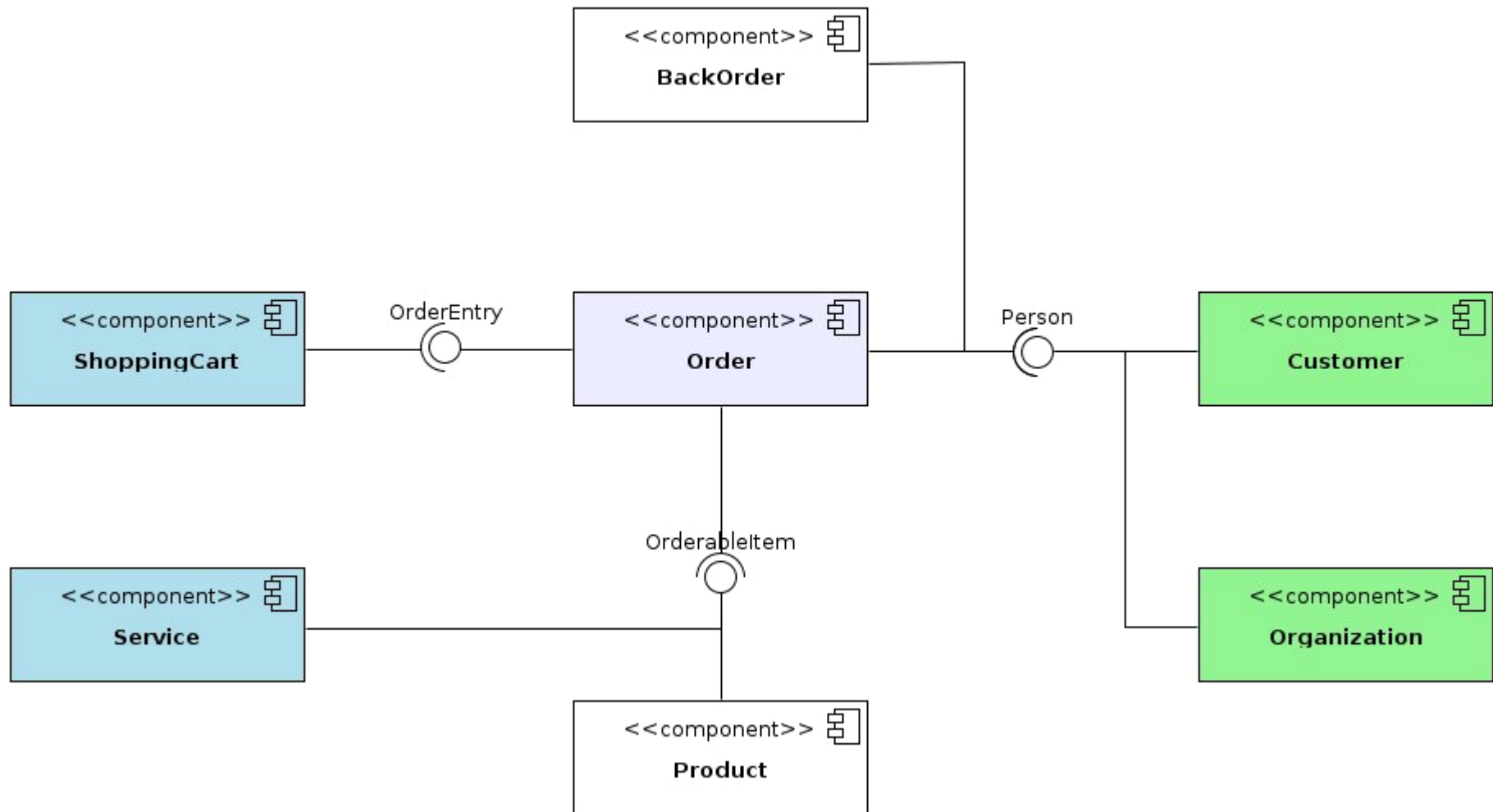
Process Order



5.8 Component Diagram

- ❑ เป็นไดอะแกรมที่อธิบายองค์ประกอบของซอฟต์แวร์และความสัมพันธ์ในแต่ละส่วน
 - คอมโพเนนต์เป็นหน่วยที่แบ่งเป็นส่วนที่ทำงานอย่างเป็นอิสระภายในระบบงาน
 - คอมโพเนนต์สามารถใช้กำหนดโครงสร้างระบบงาน
 - UML component diagrams สามารถนำไปสร้างโมเดลของซอฟต์แวร์ชั้นสูงและสามารถแสดงการเชื่อมต่อของแต่ละคอมโพเนนต์

Example : e-Commerce system



5.9 Deployment Diagram

- ❑ เป็นการแสดงให้เห็นการเชื่อมโยงระหว่าง components diagrams และ deployment diagrams
- ❑ แสดงความสัมพันธ์ทางกายภาพระหว่างฮาร์ดแวร์และซอฟต์แวร์ในระบบ
- ❑ แสดงองค์ประกอบของฮาร์ดแวร์ภายในระบบ เช่น
 - Computers (clients, servers)
 - Embedded processors
 - Devices (sensors, peripherals)
- ❑ แสดงโหนดที่ซอฟต์แวร์คอมโพเนนต์ต่าง ๆ ทำงานภายในระบบ

Example : A Java Online Web Application

