

UML (Unified Modeling Language) **Activity / State / Sequence Diagram**

01418321 System Analysis and Design
Chalothon Chootong (Ph.D.)

Department of Computer Science and Information, Faculty of
Science at Sriracha, Kasetsart University Sriracha Campus

chootong.c@ku.th



ไคอะแกรมต่างๆ

- ☐ ยูสเคสไคอะแกรม (Use Case Diagram)
- ☐ คลาสไคอะแกรม (Class Diagram)
- ☐ ออบเจ็คไคอะแกรม (Object Diagram)
- ☐ แอ็กทิวิตีไคอะแกรม (Activity Diagram)
- ☐ สเตทชาร์ตไคอะแกรม (State chart Diagram)
- ☐ คอลแลบอเรชันไคอะแกรม (Collaboration Diagram)
- ☐ ซีเควนซ์ไคอะแกรม (Sequence Diagram)
- ☐ คอมโพเนนต์ไคอะแกรม (Component Diagram)
- ☐ ดีพลอยเมนต์ไคอะแกรม (Deployment Diagram)

▷ มุมมองเชิงโครงสร้าง
Class Diagram
Object Diagram
Component Diagram
Deployment Diagram

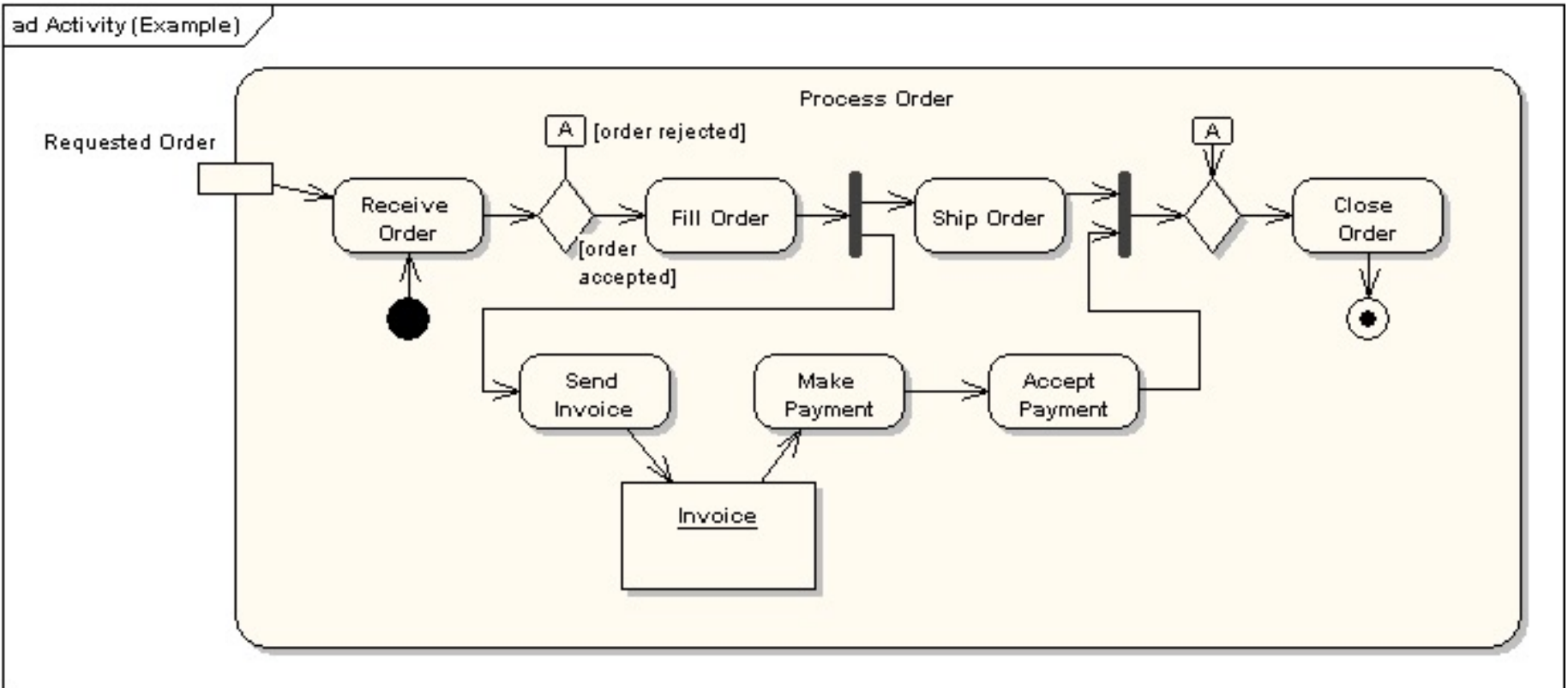
▷ มุมมองเชิงพฤติกรรม
Use case Diagram
Sequence Diagram
Collaborative Diagram
State Diagram
Activity Diagram

} **Interaction Diagrams**

What is a UML Activity Diagram?

- ▷ An activity is the execution of a task whether it be a **physical activity or the execution** of code.
- ▷ The activity diagram **shows the sequence of activities**, Like the simple flow chart.
- ▷ Activity diagrams have **support for conditional behaviour**, but has added support for parallel execution as well.

Activity Diagram



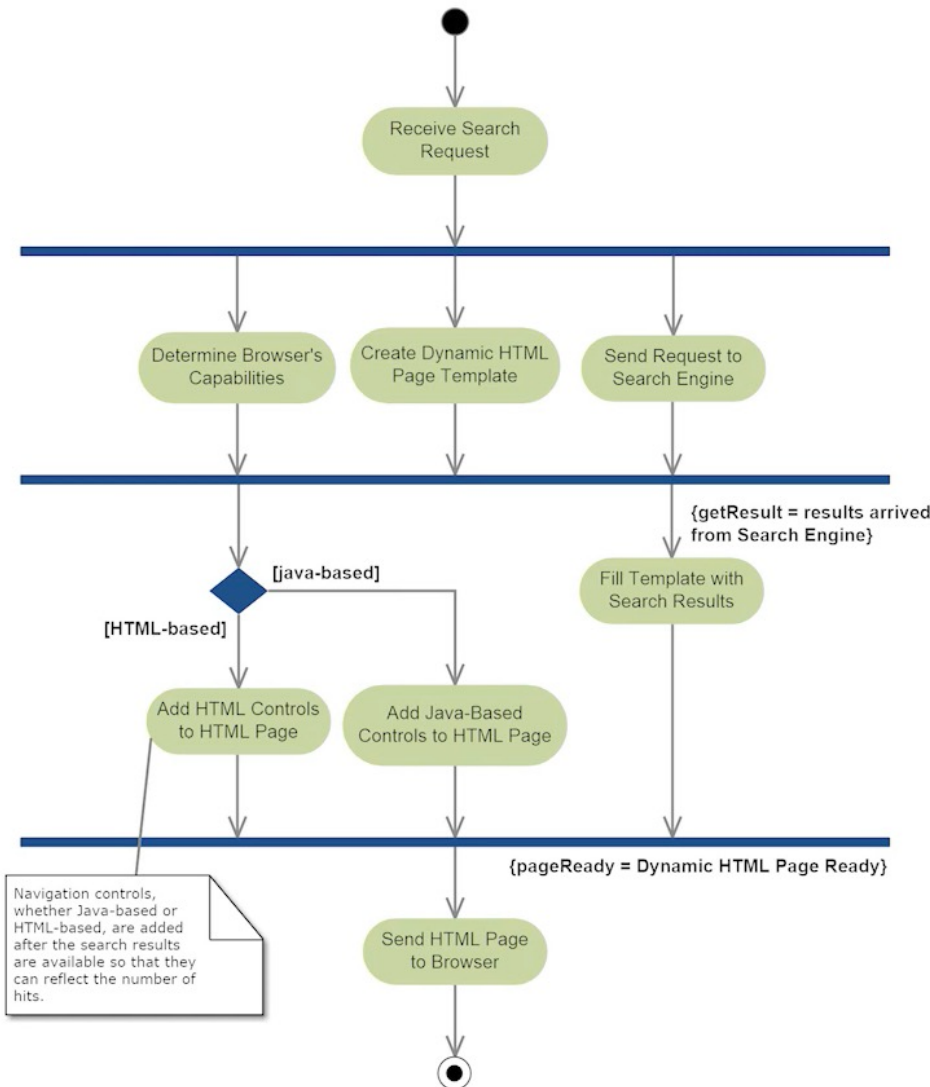
- ▷ อธิบายกิจกรรมที่เกิดขึ้นในลักษณะกระแสการไหลของการทำงาน (workflow)
- ▷ จะมีลักษณะเดียวกับ Flowchart

When should use the Activity Diagram

- ▷ ต้องการเน้นกิจกรรม (Activity) / หน้าที่การทำงาน (Functionality) ไม่ใช่วัตถุ (Object) ที่ทำให้เกิดกิจกรรม
- ▷ มีขั้นตอนการทำงานเป็นลำดับ (Step) จนกระทั่งสิ้นสุดการทำงานโดยไม่ถูก Interrupt จากเหตุการณ์ภายนอก
- ▷ ต้องการแสดงการไหล (Flow) ของข้อมูลหรือวัตถุระหว่างแต่ละขั้นตอน

Activity Diagram

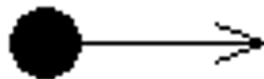
- > Use for Business Process Modelling
- > Part of the Unified Modelling Language (UML)
- > Can model both computational and organization workflows



Basic Activity Diagram Symbols and Notations

Initial State or Start Point

- A filled circle followed by an arrow represents the initial action state.
- Start: each activity diagram has one start (above) at which the sequence of actions begins.
- For activity diagram using swimlanes, make sure the start point is placed in the top left corner of the first column.



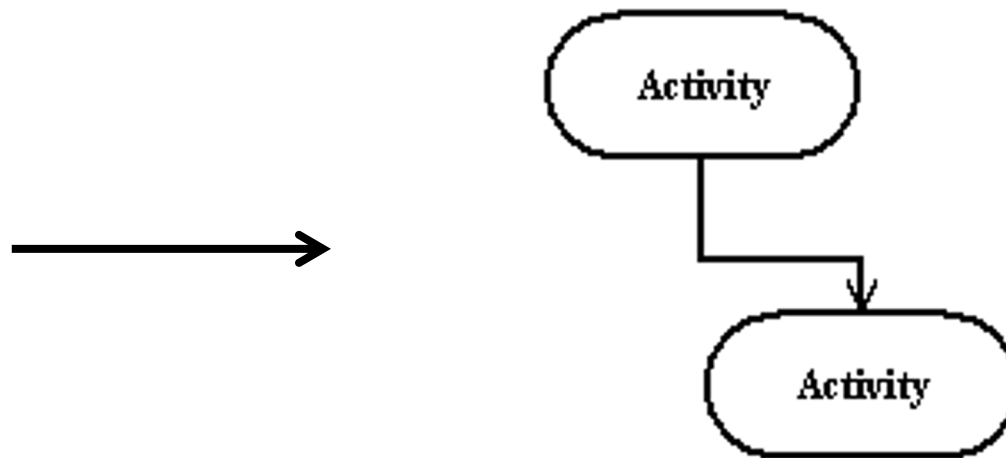
▷ Action states

- Action states represent the non interruptible actions of objects.
- You can draw an action state using a rectangle with rounded corners.



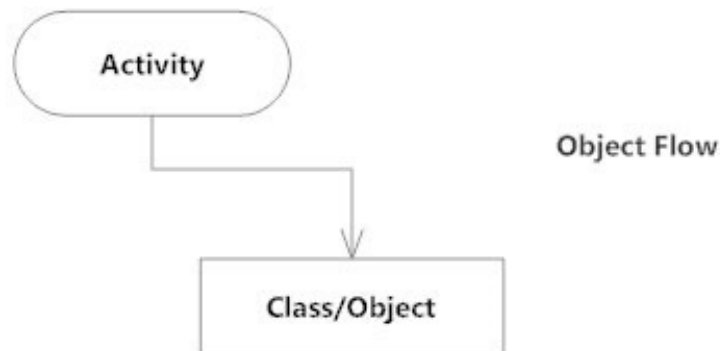
Action Flow

- Action flow arrows illustrate the relationships among action states.
- Also called edges and paths, illustrate the transitions from one action state to another.



Object Flow

- ▷ Object flow refers to the creation and modification of objects by activities.
- ▷ An object flow arrow from an action to an object means that the action creates or influences the object.
- ▷ An object flow arrow from an action to an object means that the action creates or influences the object.



Branching

- A diamond represents a decision with **alternate paths**. The outgoing alternates should be labeled with a condition or guard expression.
- You can also label one of the paths "else."

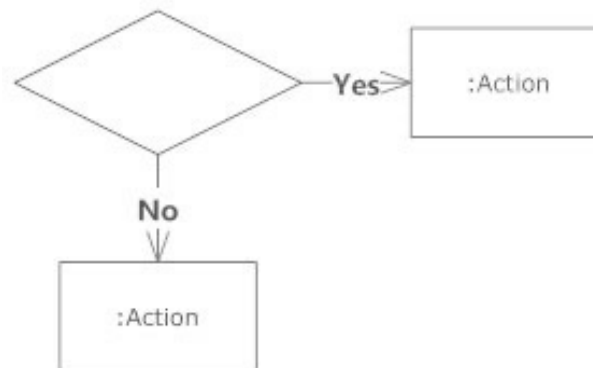
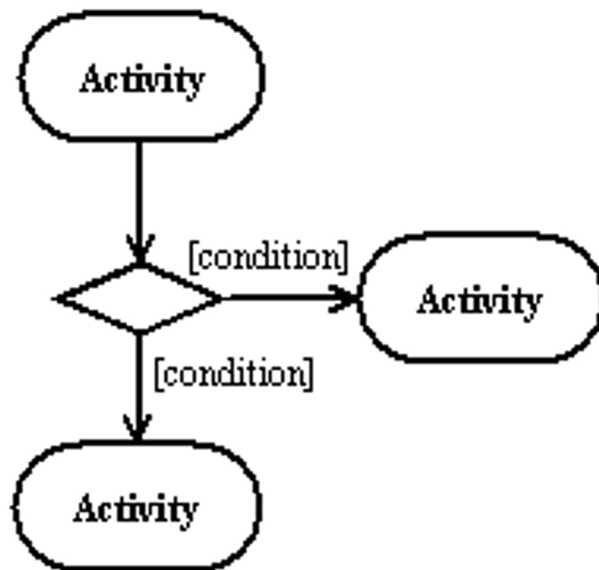


Decision Symbol

Guards

- ▷ In UML, guards are a statement written next to a decision diamond that must be true before moving next to the next activity.
- ▷ These are not essential, but are useful when a specific answer, such as "Yes, three labels are printed," is needed before moving forward.
- ▷ The guard is a conditional expression that when true indicates that the transition is taken.
- ▷ The label is also optional and is freeform.

Guards

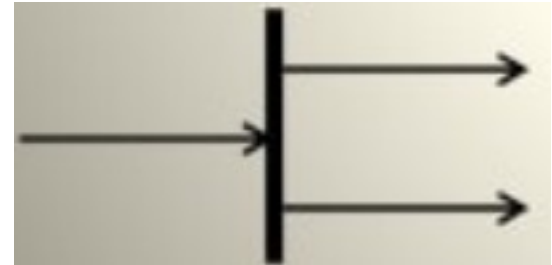
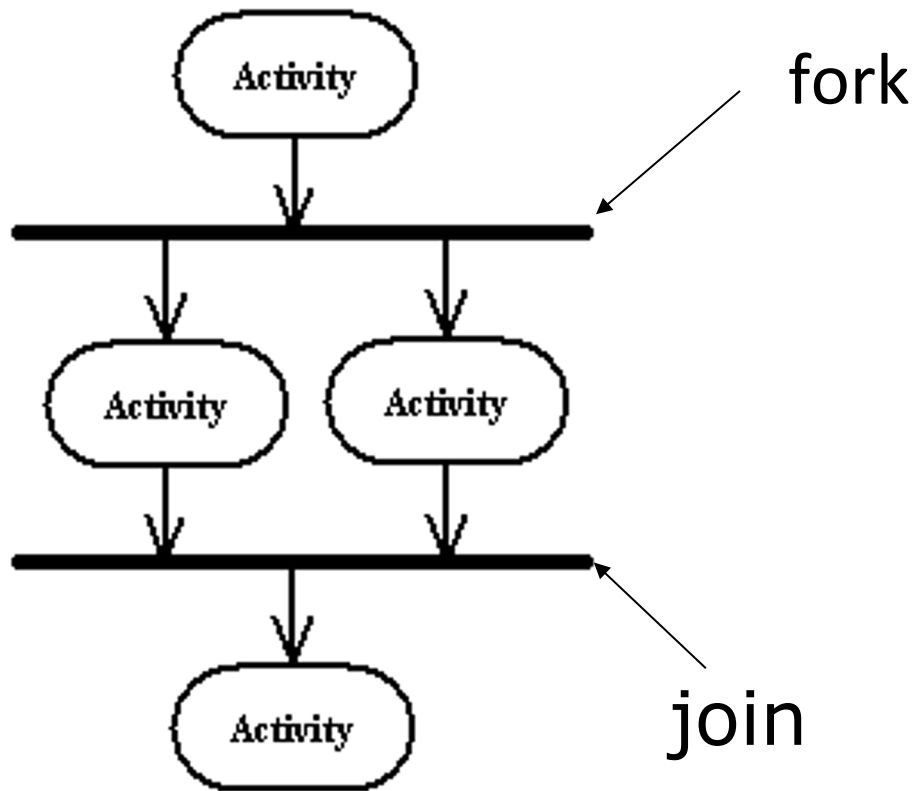


Guard Symbols

Synchronization

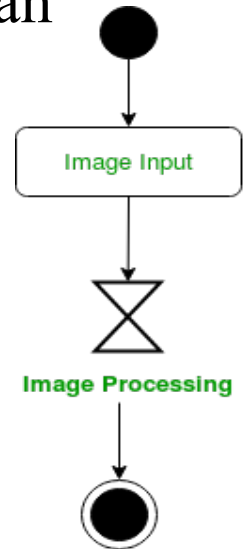
- ▷ A synchronization bar helps illustrate parallel transitions. Synchronization is also called **forking** and **joining**.
- ▷ To show parallel behaviour use a fork and a join.
 - # The fork (top) has **one transition** entering and **any number of transitions exiting**, all of which will be taken.
 - # The join (bottom) represents the end of the parallel behaviour and **has any number of transitions entering, and only one leaving**.

Synchronization



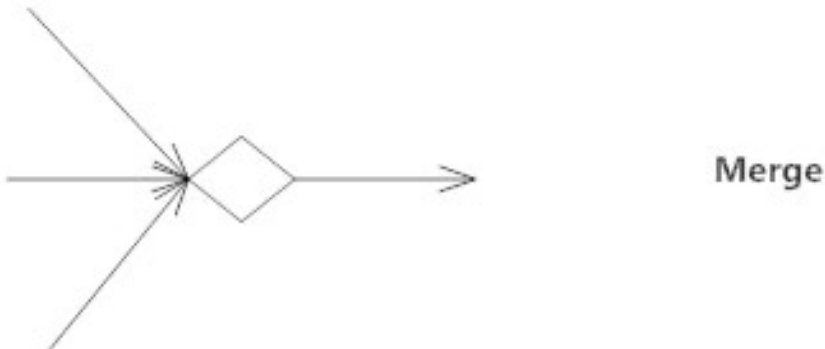
Time Event

- This refers to an event that stops the flow for a time; an hourglass depicts it.



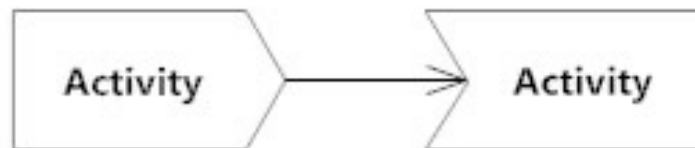
Merge Event

- A merge event brings together multiple flows that are not concurrent.



Sent and Received Signals

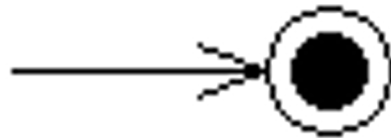
- ▷ Signals represent how activities can be modified from outside the system.
- ▷ They usually appear in pairs of sent and received signals, because the state can't change until a response is received, much like synchronous messages in a sequence diagram.
- ▷ For example, an authorization of payment is needed before an order can be completed.



Signal sent and
received

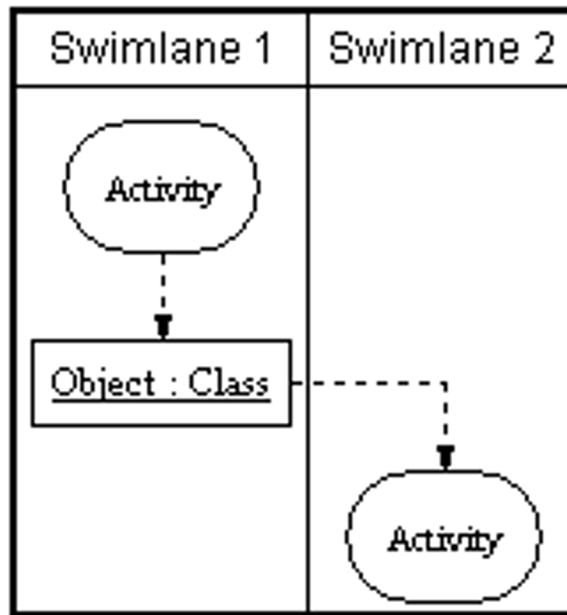
Final State

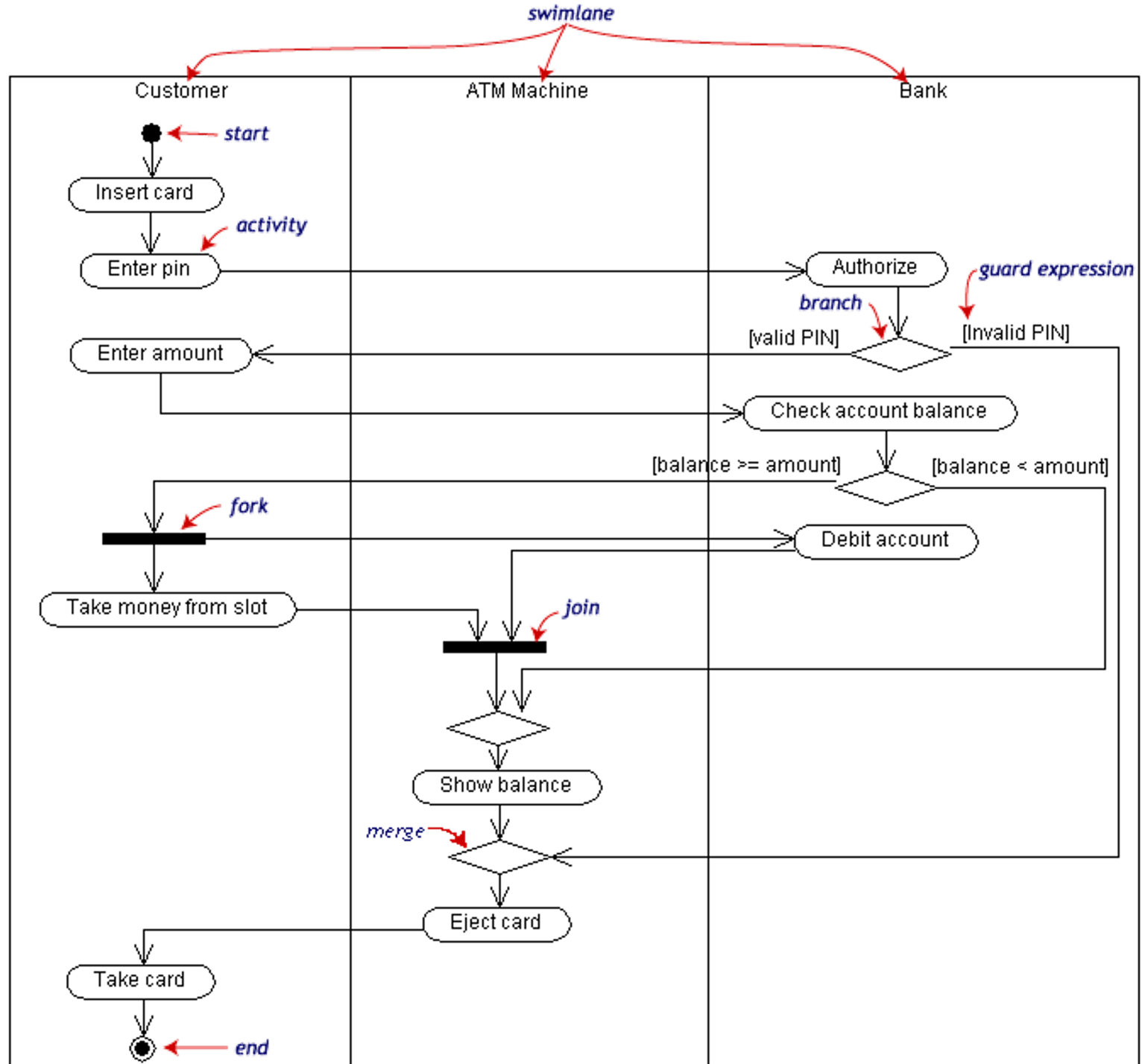
- An arrow pointing to a filled circle nested inside another circle represents the final action state.
- End: each activity diagram has one finish at which the sequence of actions ends



Swimlanes

- ▷ Swim lanes group related activities into one column.

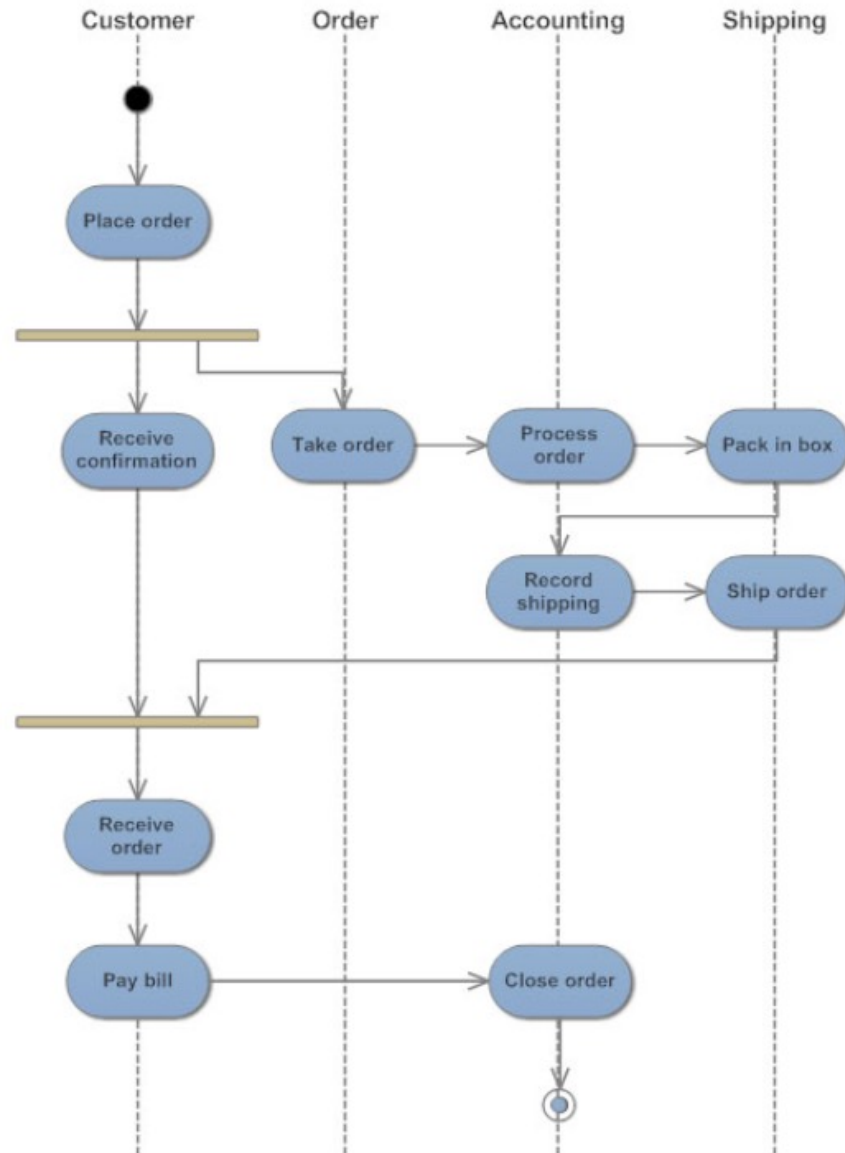


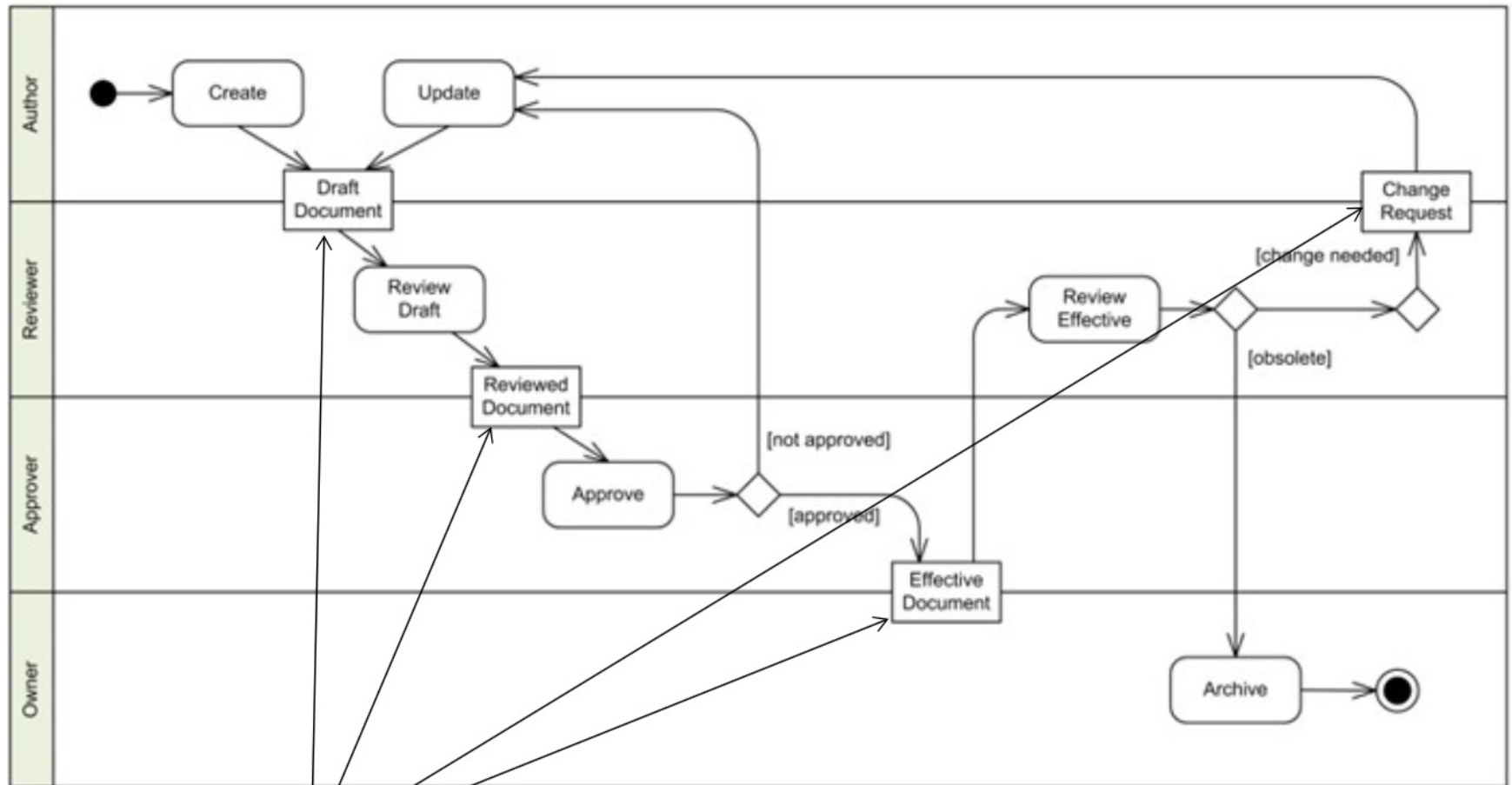


How to Draw an activity Diagram

- ▷ Identify **the initial state** and **the final states**.
- ▷ Identify **the intermediate activities** needed to reach the final state from the initial state.
- ▷ Identify **the conditions** or **constraints** which cause the system to change control flow.
- ▷ Draw the diagram with appropriate notations.

UML Activity Diagram: Order Processing





Objects! (should really use [] for state)

Activity Diagram Summary

▷ Pros:

- Map use case scenarios directly on to actions
- Most intuitive for most procedural programmers
- Includes constructs for top down decomposition (activity frames)

▷ Cons:

- Some confusion of the relationship between activity diagrams and statecharts

Recommendation: useful early in analysis, after use case.



Statechart Diagram

What is a UML Statechart Diagram?

- ▷ A statechart diagram is a dynamic model that shows the different states that a single object passes through during its life in response to events, along with its responses and action.

State transition diagram เป็นแผนภาพใช้แสดงสถานะ(state) ต่างๆ ของ Object ที่เป็นไปได้ในระหว่างช่วงชีวิต ในการตอบสนองต่อ เหตุการณ์ (Event) ที่เกิดขึ้น โดยทั่วไปแล้ว StateDiagram จะไม่ถูกใช้กับ Class ทั้งหมด แต่จะใช้อธิบายเฉพาะ Class ที่มีความซับซ้อนสูงเท่านั้น เพื่อที่จะช่วยให้การออกแบบ Algorithm ง่ายขึ้น

สถานะนิติ

☐ ปกติ

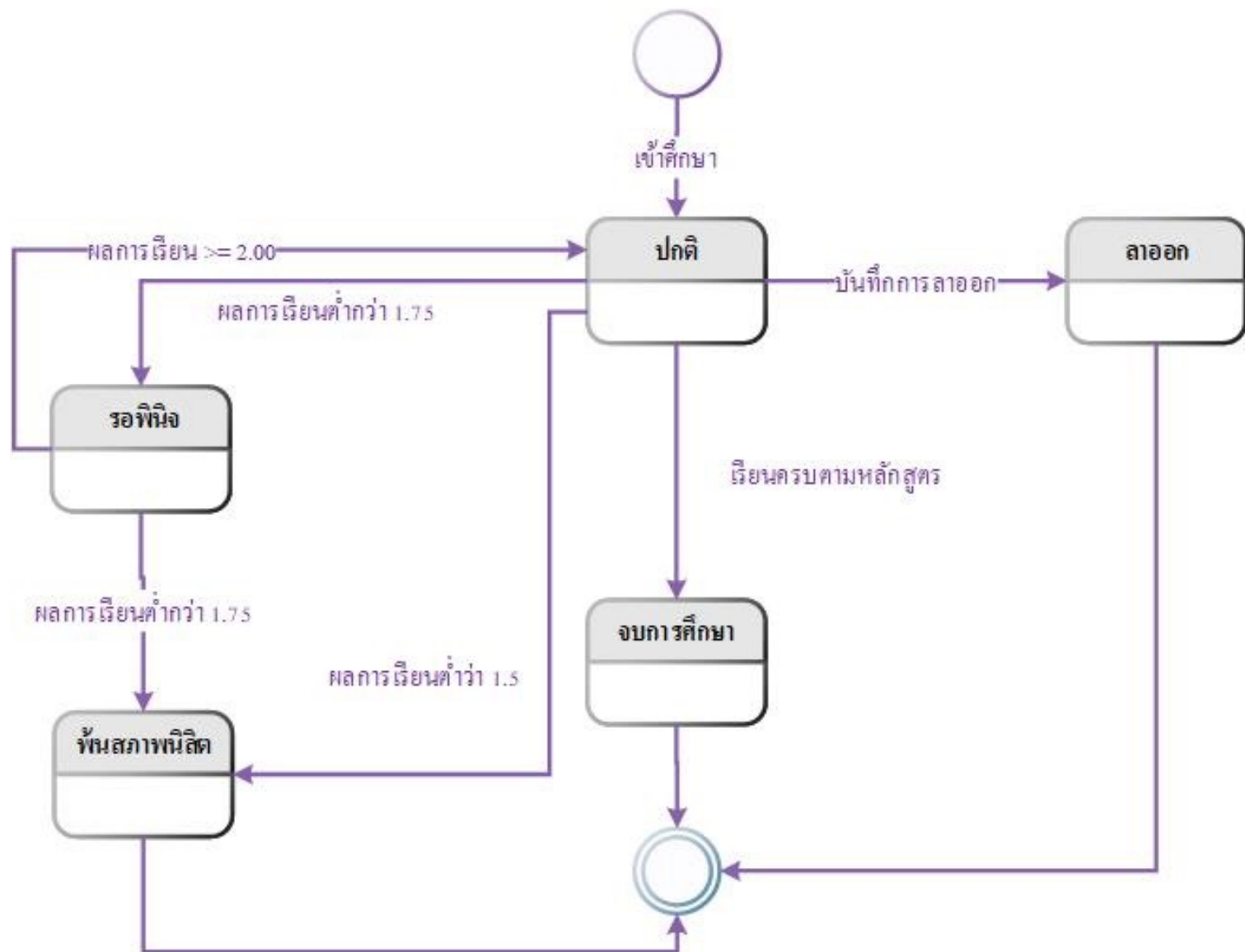
☐ รอพินิจ

☐ พันสภาพ

☐ จบการศึกษาแล้ว

☐ พักการเรียน

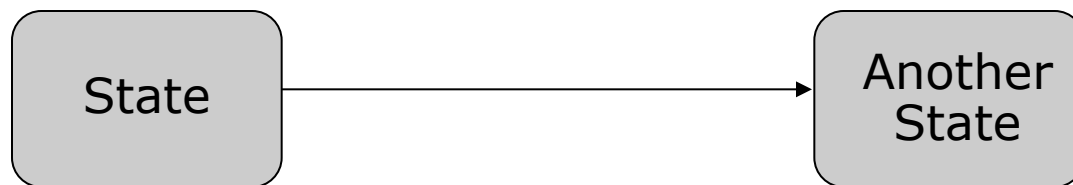
☐ ลาออก



Basic Statechart Diagram Symbols and Notations

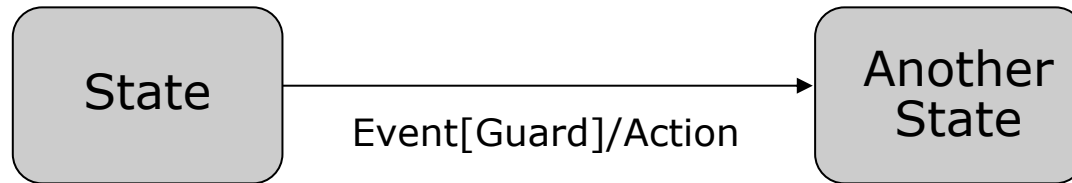
▷ States

- State of an object is defined by the value of its attributes and its relationships with other objects at a particular point in time



Transition

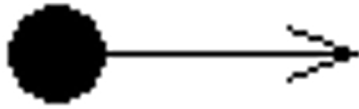
- ▷ A transition is a relationship that represents the movement of an object from one state to another state



Transition

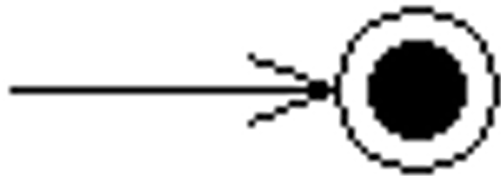
- ▷ An event is something that takes place at a certain point in time and changes a value that describes an object, which in turn changes the object's state.
- ▷ Some transitions will have a guard condition
 - A guard condition is a boolean expression that includes attribute values, which allows a transition to occur only if the condition is true

Initial State



○ A filled circle followed by an arrow represents the object's initial state.

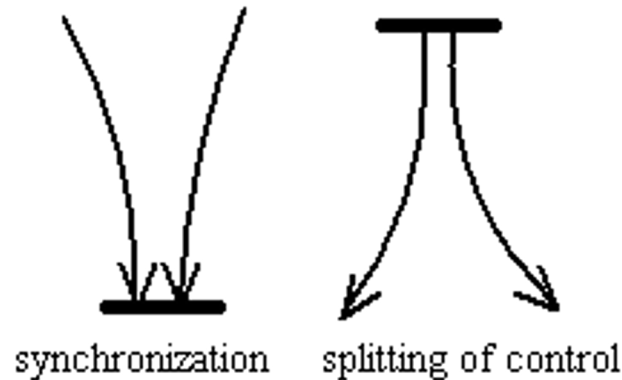
Final State

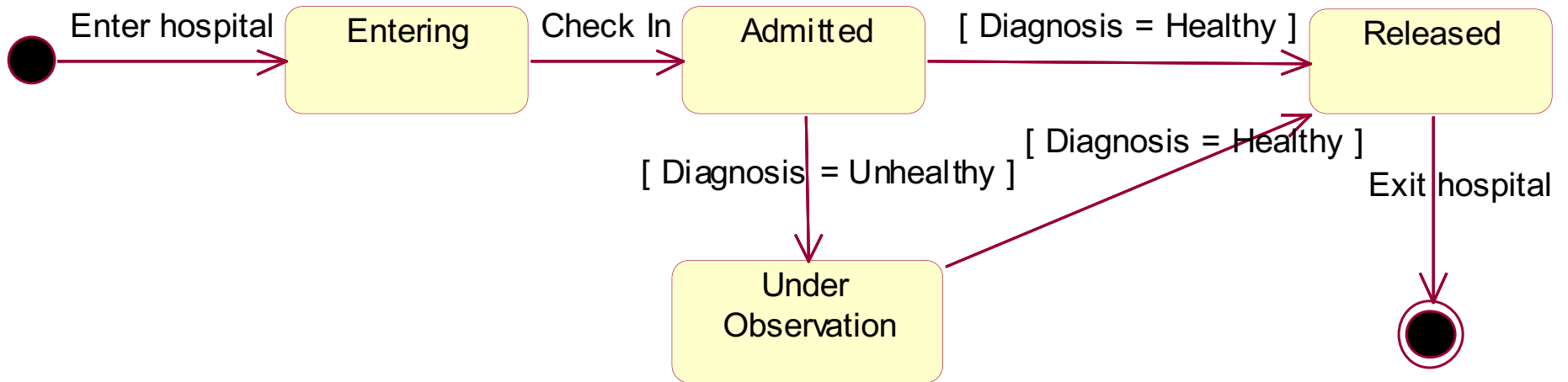


○ An arrow pointing to a filled circle nested inside another circle represents the object's final state.

Synchronization and Splitting of Control

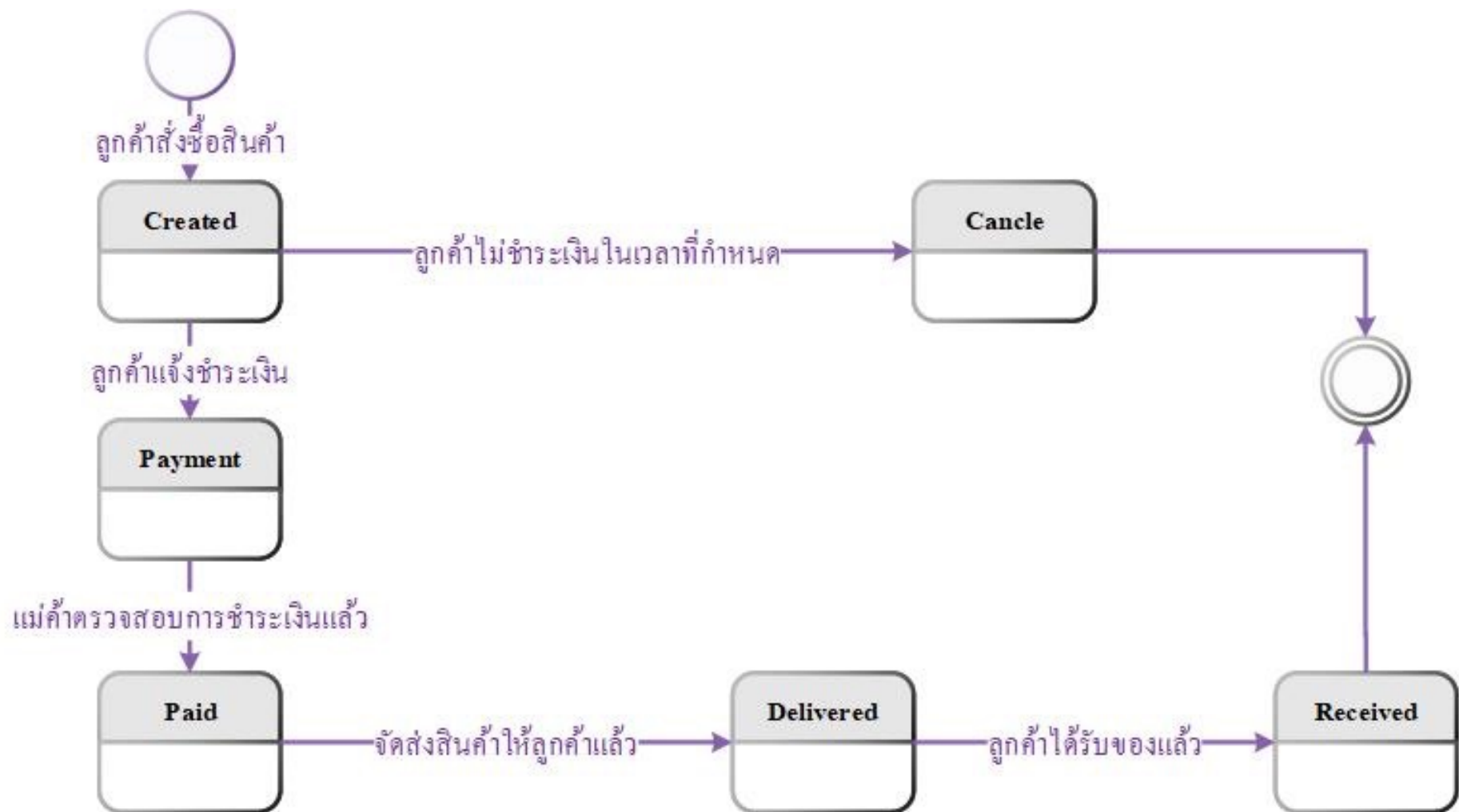
- A short heavy bar with two transitions entering it represents a synchronization of control.
- A short heavy bar with two transitions leaving it represents a splitting of control that creates multiple states.





สถานะของใบสั่งซื้อสินค้า

- ▷ ใบสั่งซื้อสินค้าถูกสร้างแล้ว (Created)
- ▷ ใบสั่งซื้อสินค้าถูกยกเลิก (Cancel)
- ▷ แจ้งชำระเงินแล้ว (Payment)
- ▷ ชำระเงินแล้ว (Paid)
- ▷ จัดส่งแล้ว (Delivered)
- ▷ ได้รับของแล้ว (Received)



ใบสั่งซื้อสินค้า

หมายเลขใบสั่งซื้อ	ราคารวม	วันที่สร้าง	สถานะ	หมายเลขสมาชิก
001	1000	1/1/2020	Created	A001
002	2000	1/1/2020	Created	A002
003	4000	2/1/2020	Payment	A001
004	200	3/1/2020	Paid	A001
005	500	4/1/2020	Delivered	A002
006	400	5/1/2020	Payment	A002

รายการสั่งซื้อของ A001

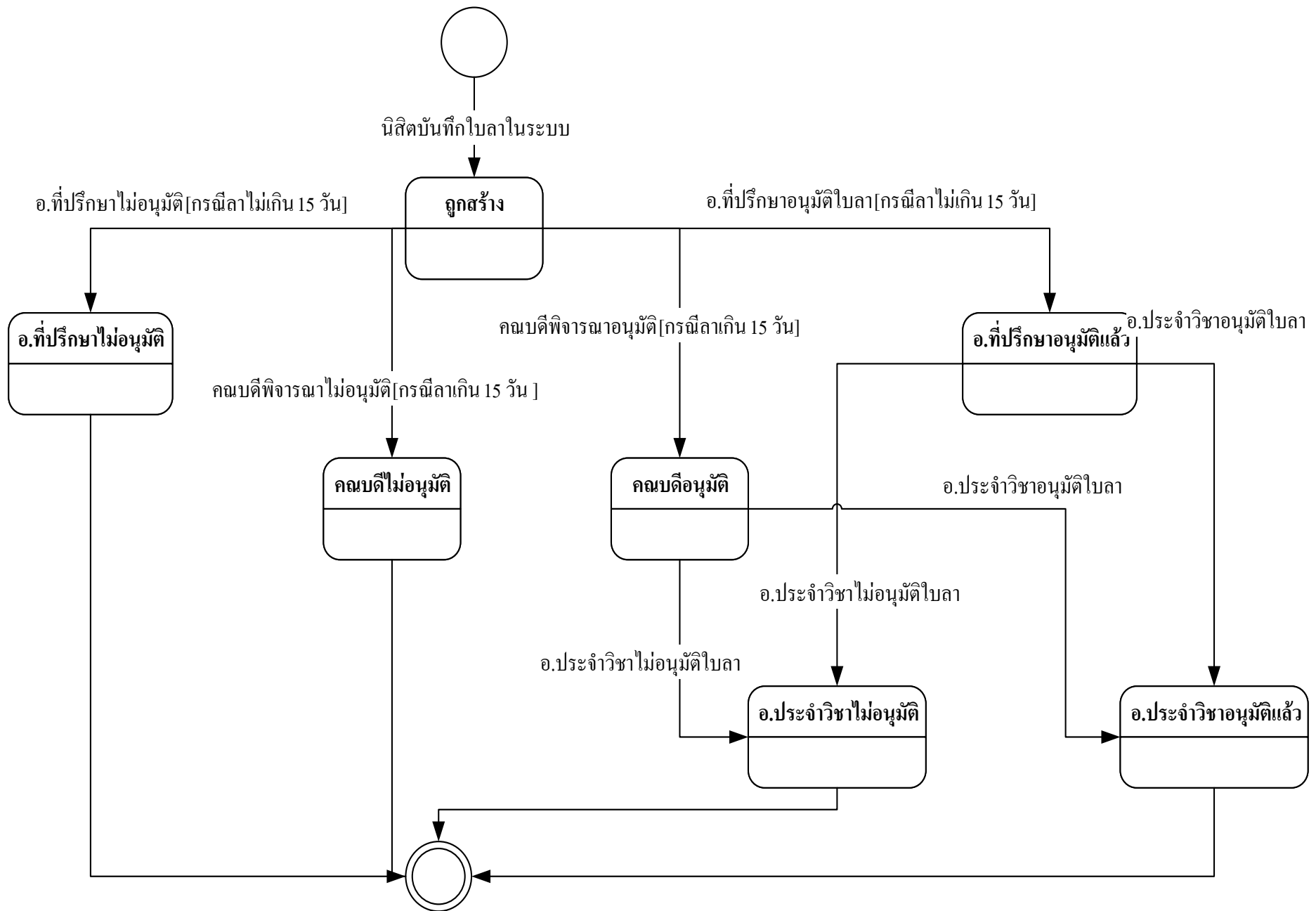
หมายเลขใบสั่งซื้อ	ราคารวม	วันที่สร้าง	สถานะ	หมายเลขสมาชิก
001	1000	1/1/2020	Created	A001
003	4000	2/1/2020	Payment	A001
004	200	3/1/2020	Paid	A001

ต้องการแจ้งชำระเงินของ A001

หมายเลขใบสั่ง ซื้อ	ราคารวม	วันที่สร้าง	สถานะ	หมายเลข สมาชิก
001	1000	1/1/2020	Created	A001

พนักงานเข้ามาดูว่ามีใครแจ้งชำระเงินบ้างเพื่อปรับสถานะเป็นชำระเงิน
แล้ว

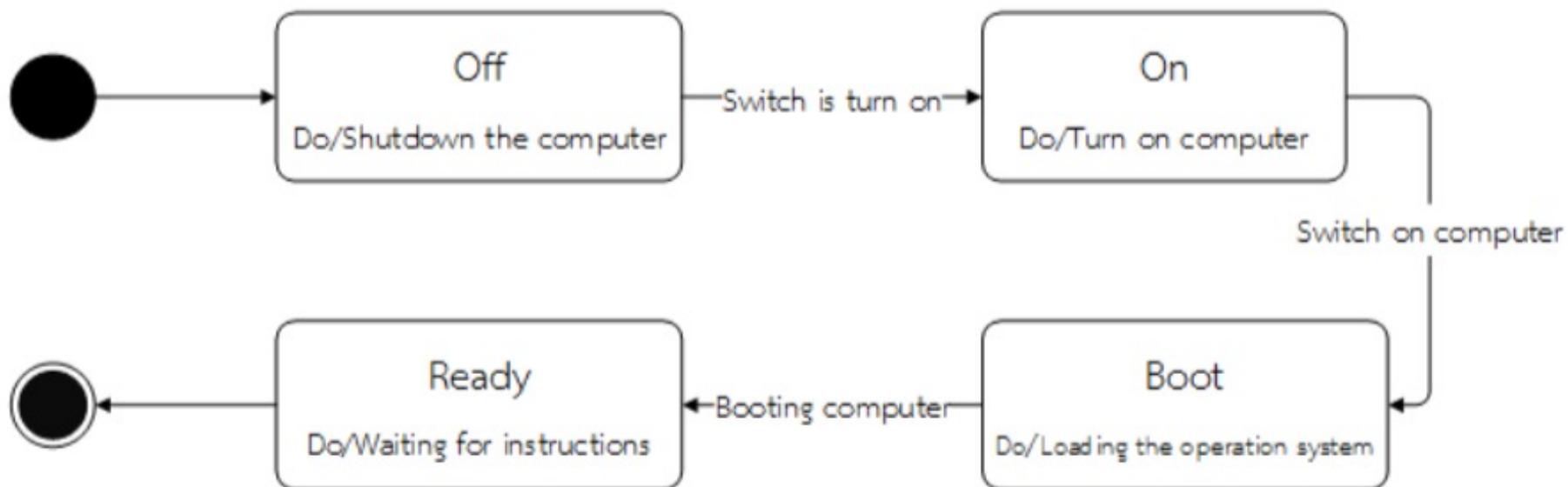
หมายเลขใบสั่งซื้อ	ราคารวม	วันที่สร้าง	สถานะ	หมายเลขสมาชิก
003	4000	2/1/2020	Payment	A001
006	400	5/1/2020	Payment	A002



การทำงานของ *State Diagram* ของแต่ละ *Function*

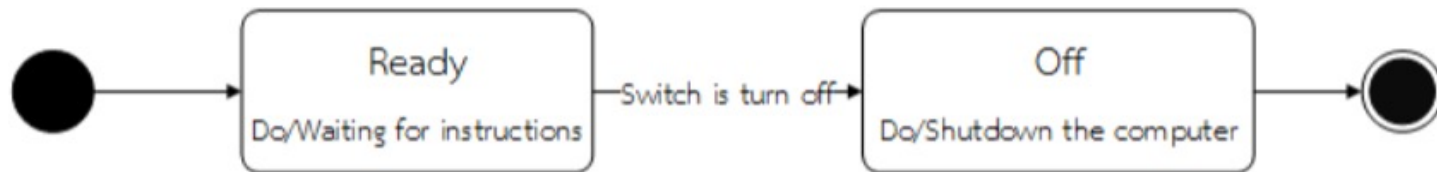
▷ *Turn On*

Turn On (เปิดเครื่อง)

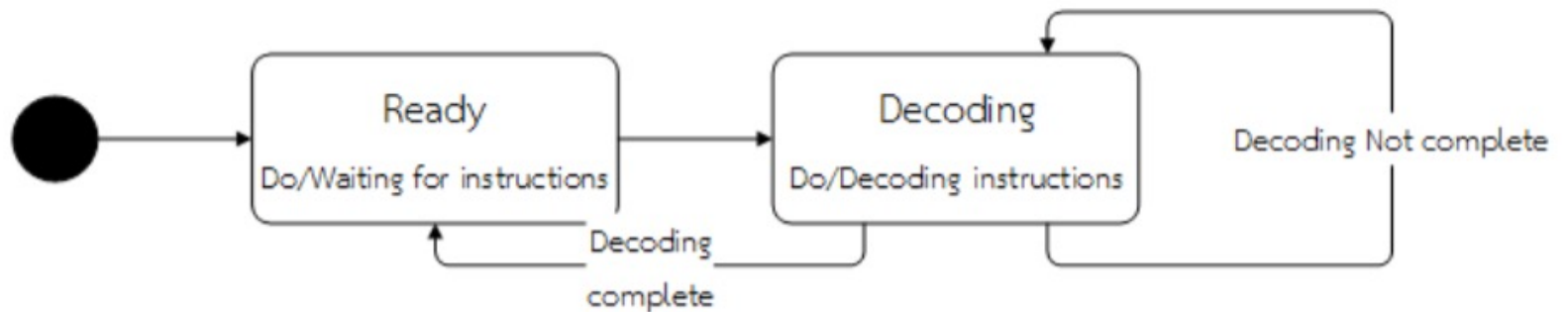


▷ *Shut Down*

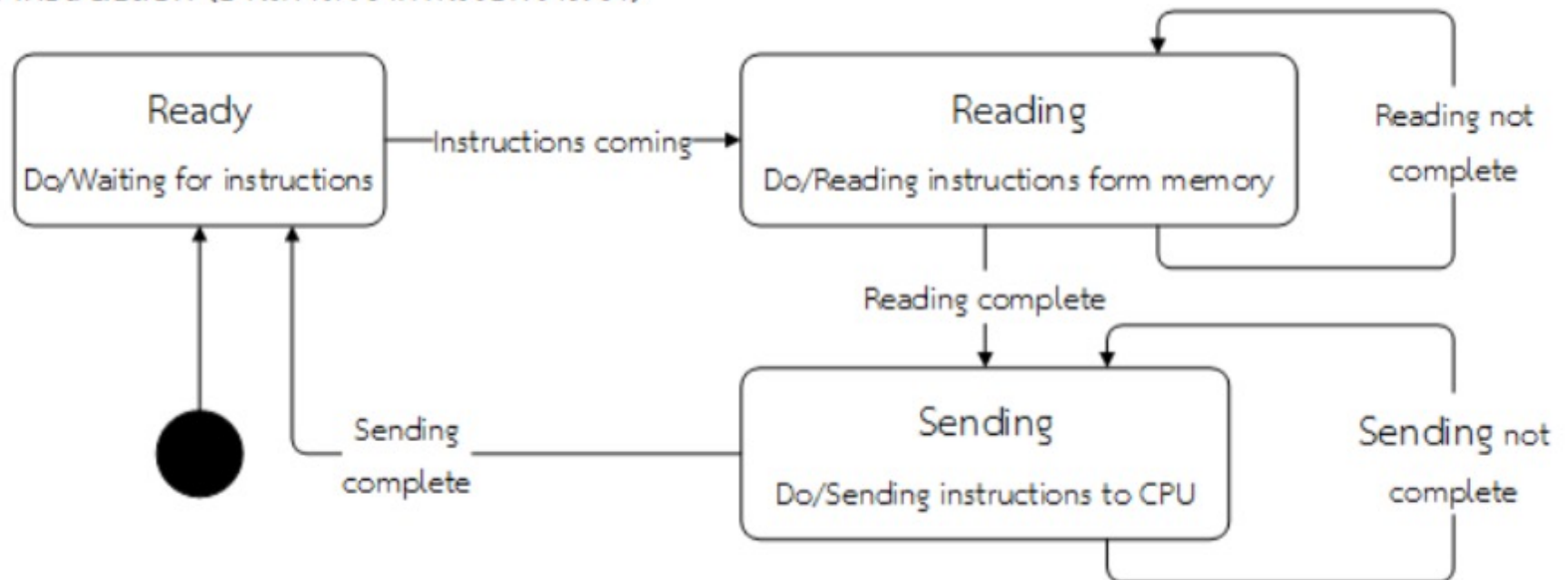
Shut Down (ปิดเครื่อง)



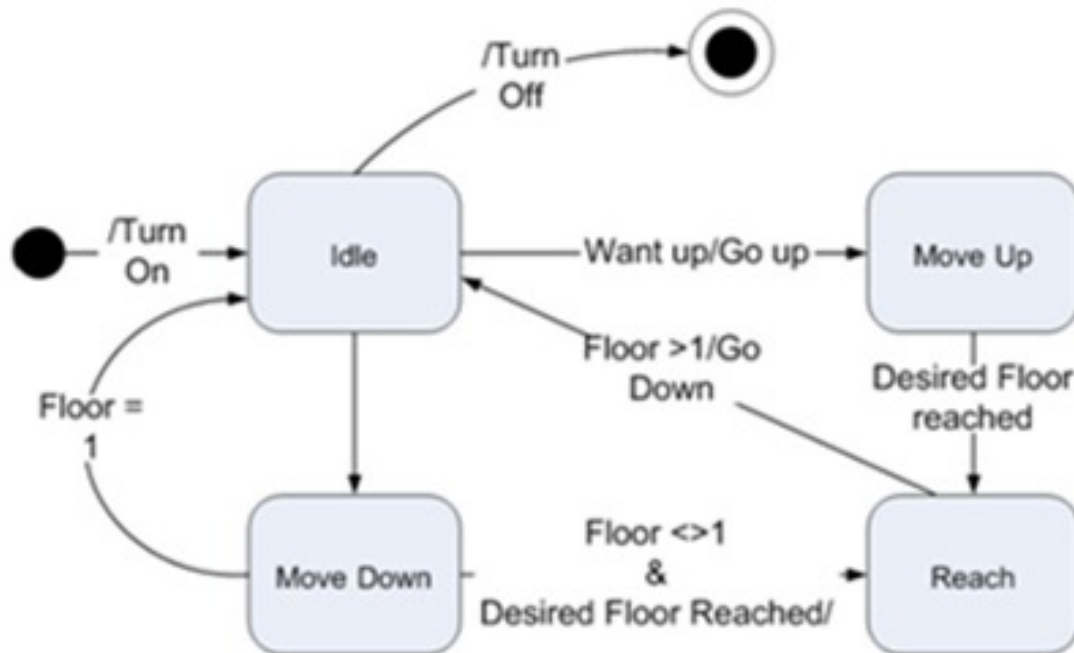
▷ *Decode*

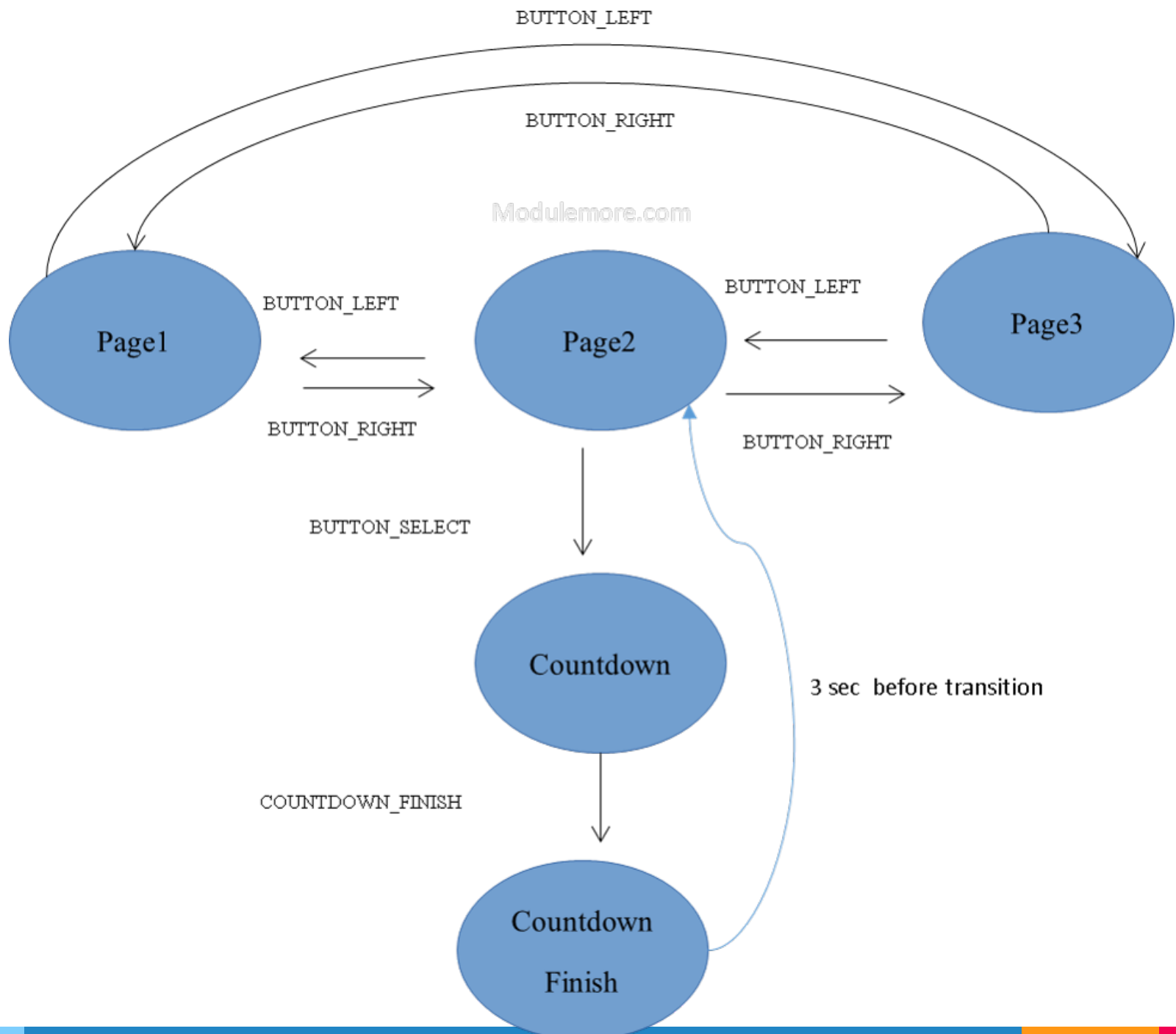


Read Instruction (อ่านคำสั่งจากหน่วยความจำ)



State Diagram การทำงานของลิฟต์



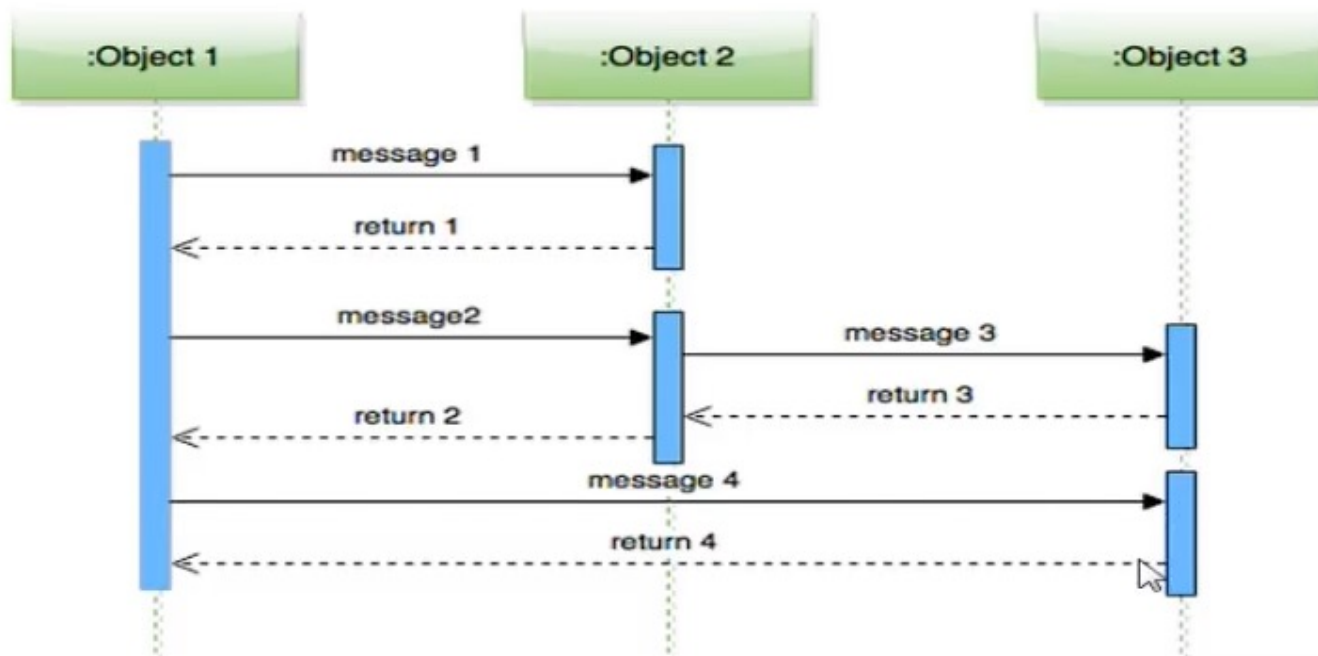






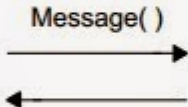



Sequence Diagram

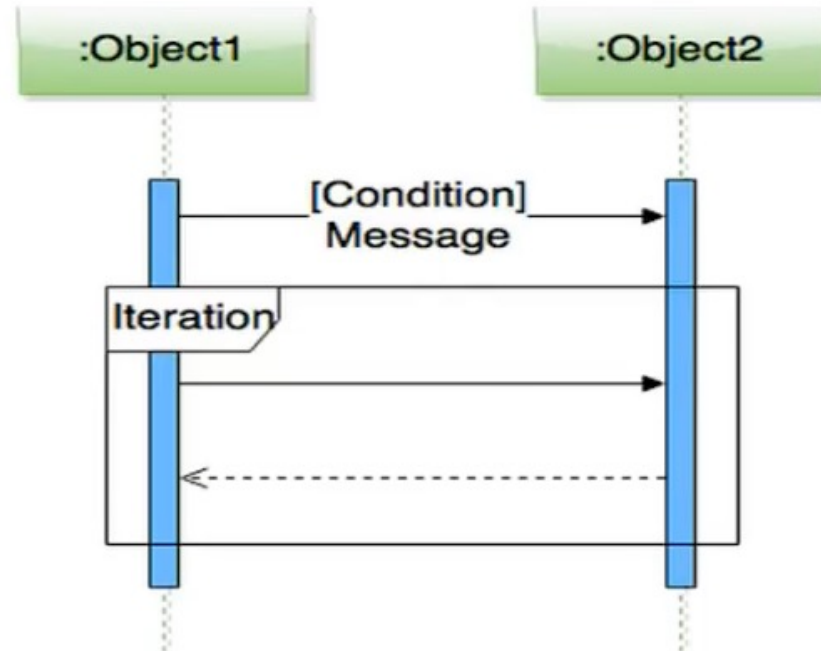
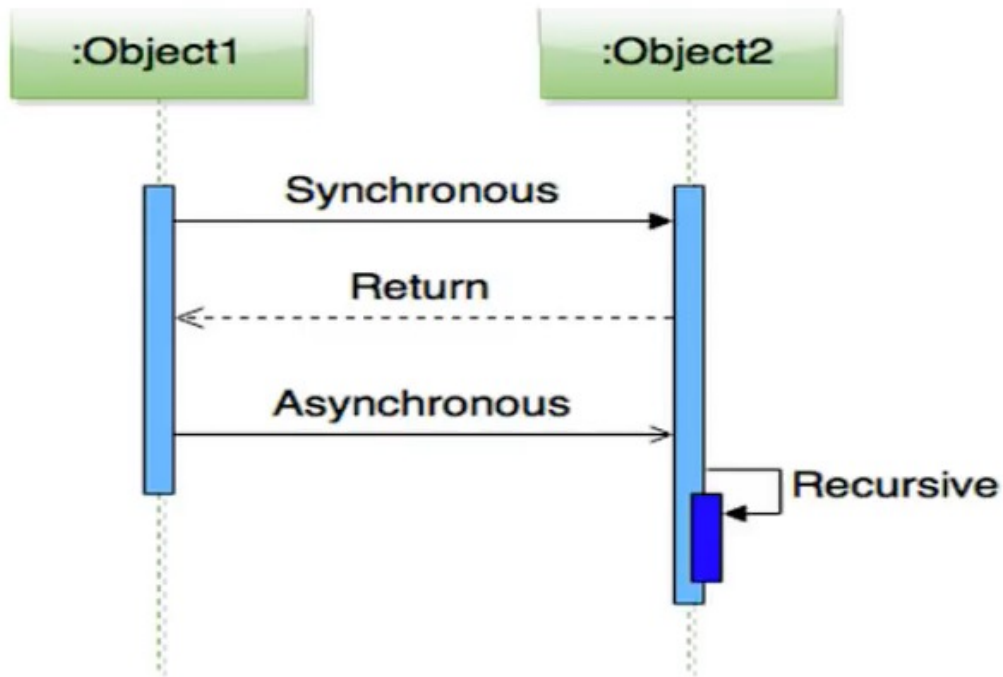
Sequence Diagram

- ▷ แสดงลำดับการทำงานของระบบ แสดงปฏิสัมพันธ์ (Interaction) ระหว่าง Object ตามลำดับ ของเหตุการณ์ที่เกิดขึ้น ณ เวลาที่กำหนด
- ▷ Message ที่เกิดขึ้นระหว่าง class จะสามารถนำไปสู่การสร้าง method ใน class ที่เกี่ยวข้องได้

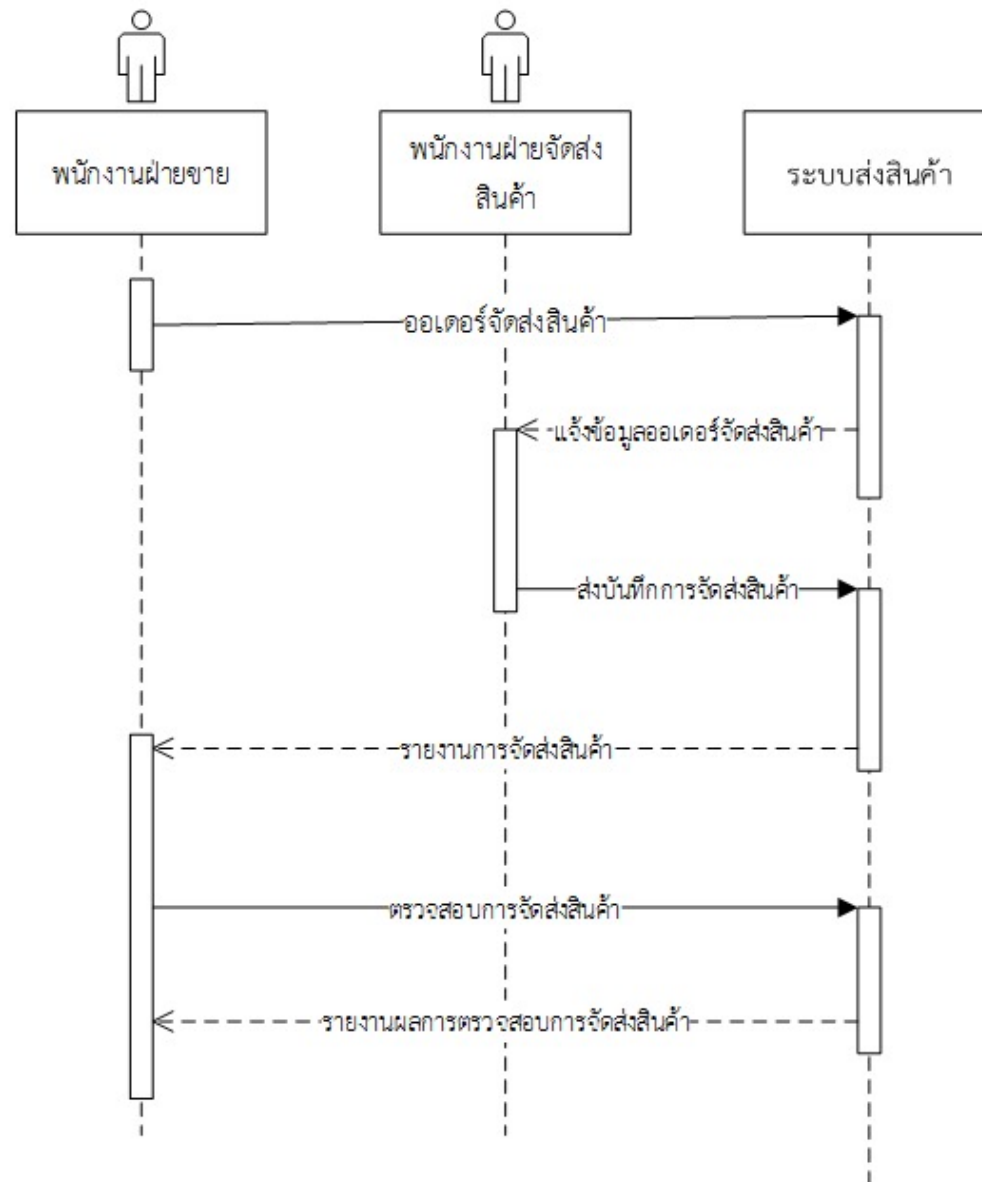


สัญลักษณ์	ชื่อ	ความหมาย
	Actor	ผู้ที่เกี่ยวข้องกับระบบ
	Object	อ็อบเจกต์ที่ต้องทำหน้าที่ ตอบสนองต่อ Actor
	Lifeline	เส้นแสดงชีวิตของอ็อบเจกต์หรือ คลาส
	Focus of Control / Activation	จุดเริ่มต้นและจุดสิ้นสุดของแต่ละ กิจกรรมในระหว่างที่มีชีวิตอยู่
	Message	คำสั่งหรือฟังก์ชันที่อ็อบเจกต์หนึ่ง ส่งให้อ็อบเจกต์หนึ่ง ซึ่ง สามารถส่งกลับได้ด้วย
	Callback / Self Delegation	การประมวลผลและคืนค่าที่ได้ ภายในอ็อบเจกต์เดียวกัน

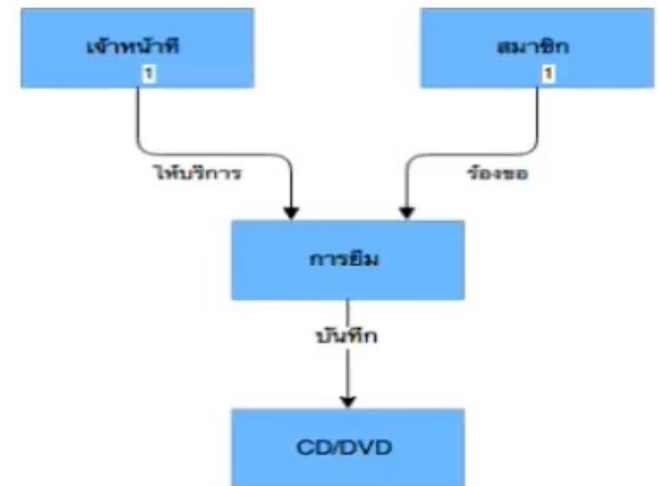
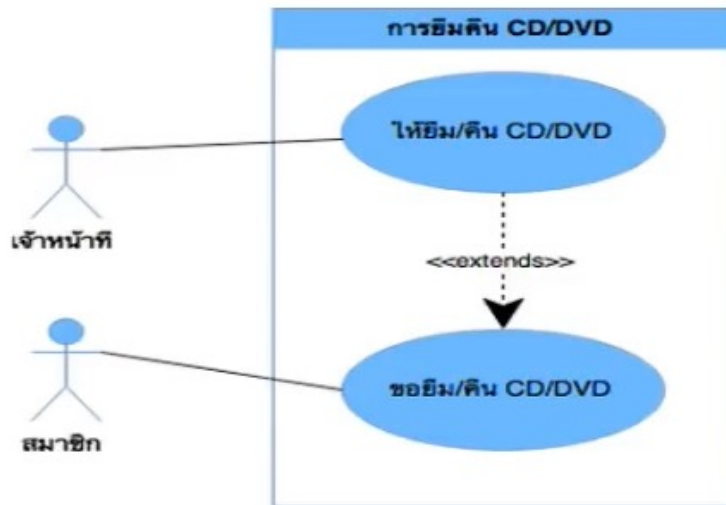
Message Type



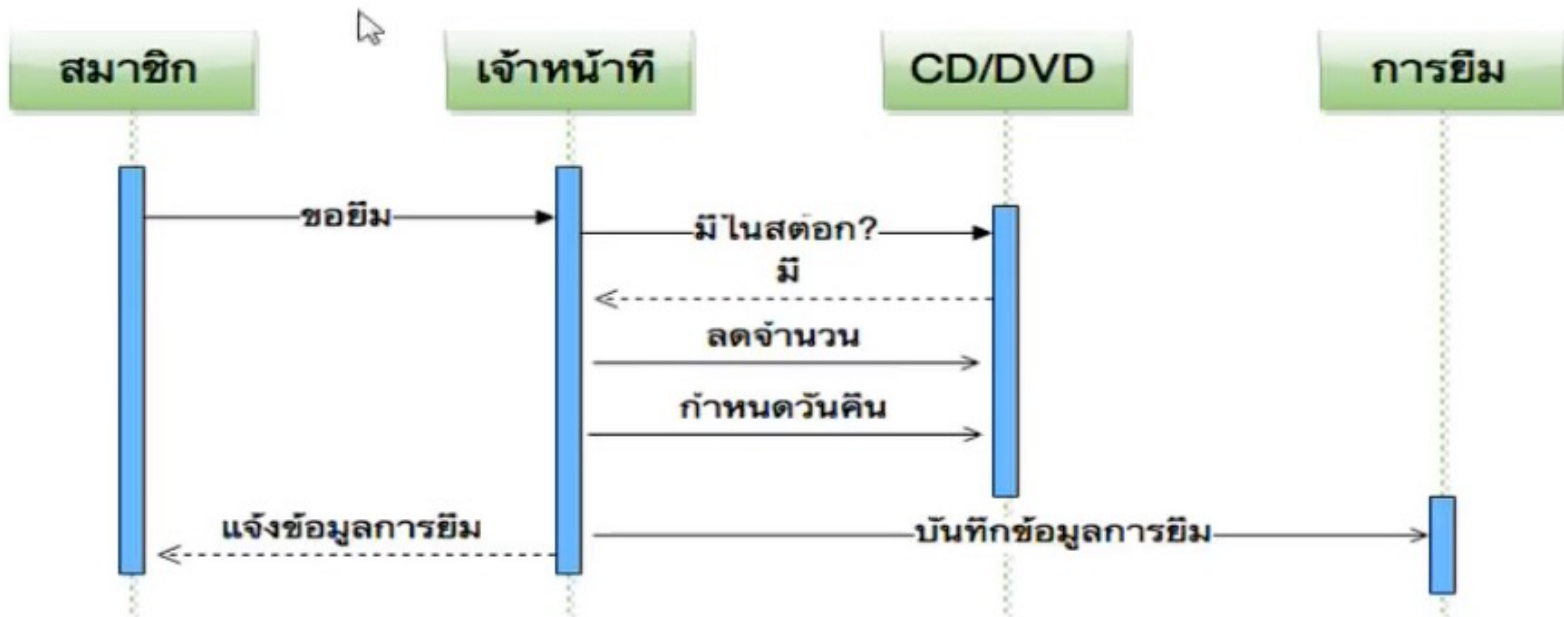
ระบบส่งสินค้า

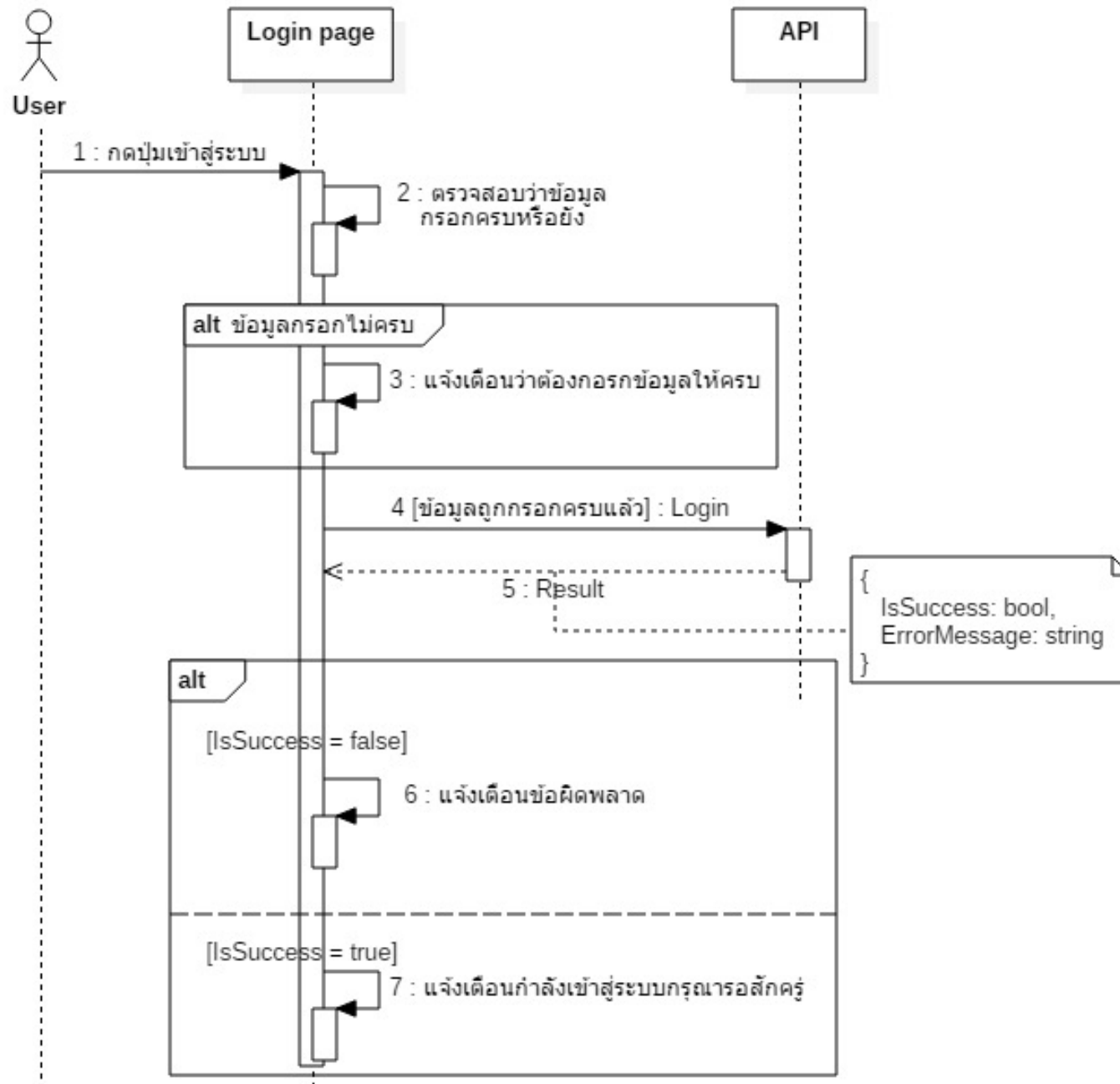


How to create Sequence Diagram?



How to create Sequence Diagram?





Thanks!

Any questions?