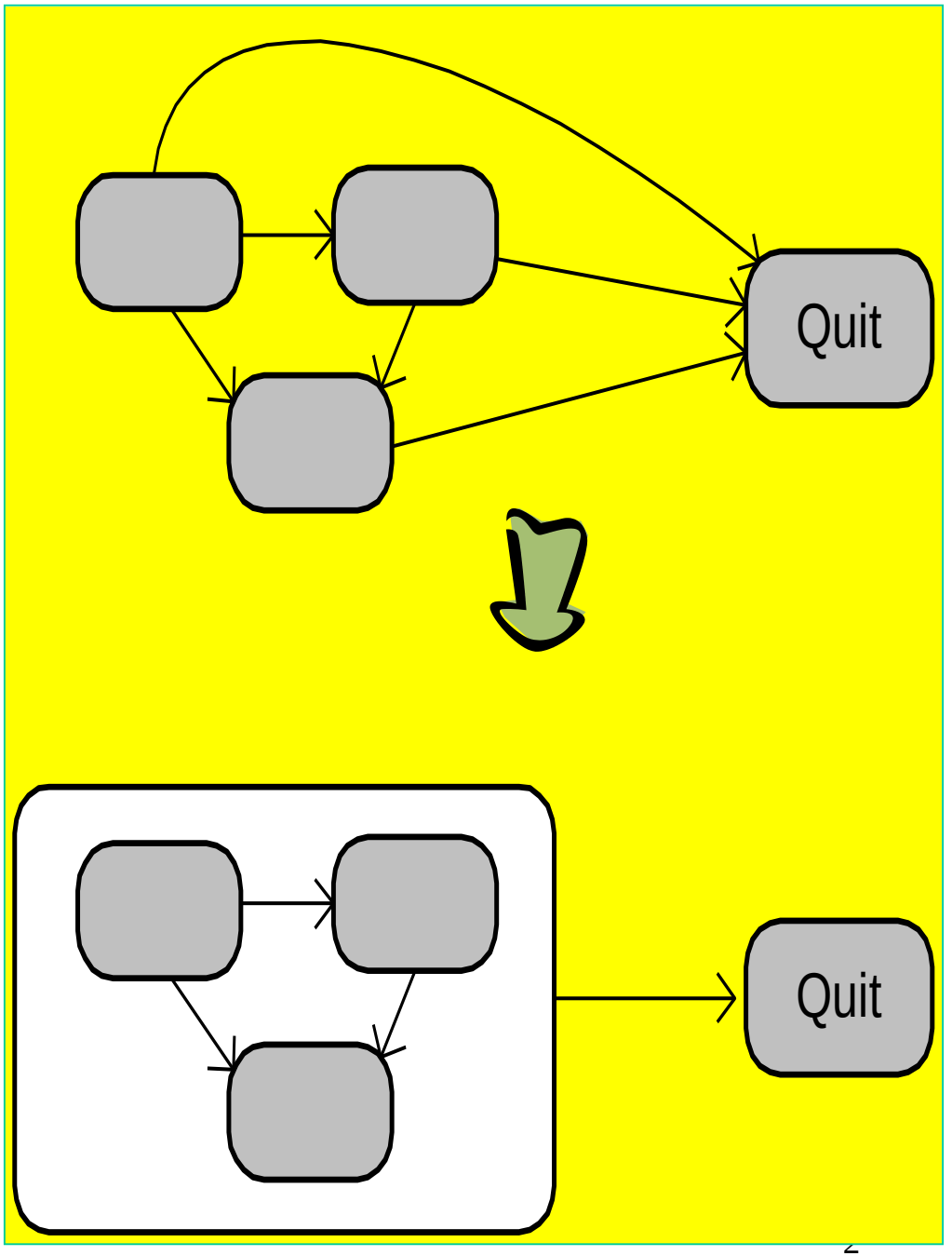
UML State Machine Model

Lect 7--8
21-08-2023

State Machine: Eliminating Duplicated Transitions

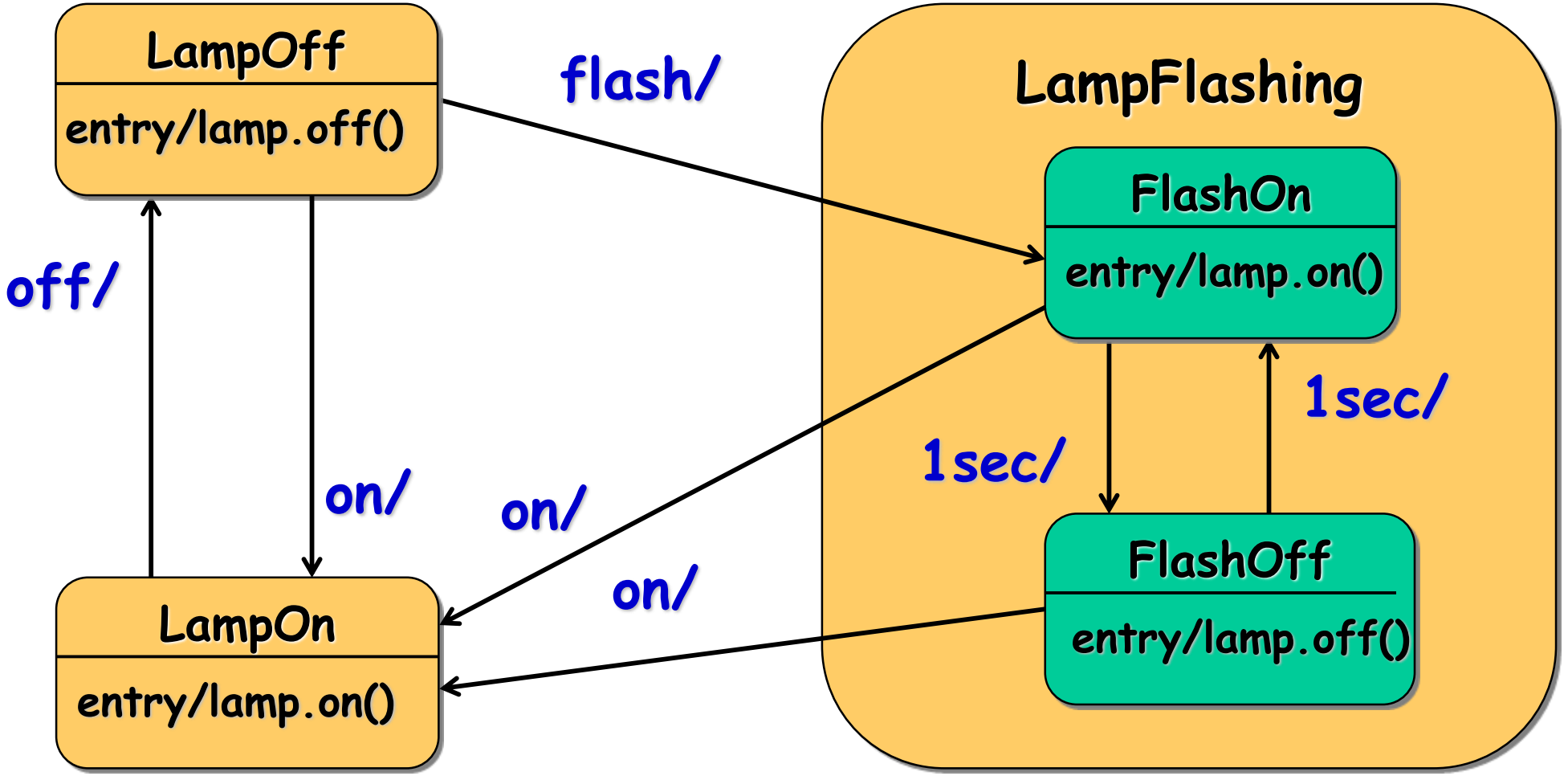
- Duplicate transitions usually exist when some transition can take place from every state:
  - "error"
  - "quit"
  - "abort"

- These duplicates can be combined into a single transition:
  - **A transition from/to a superstate holds for all its substates!**



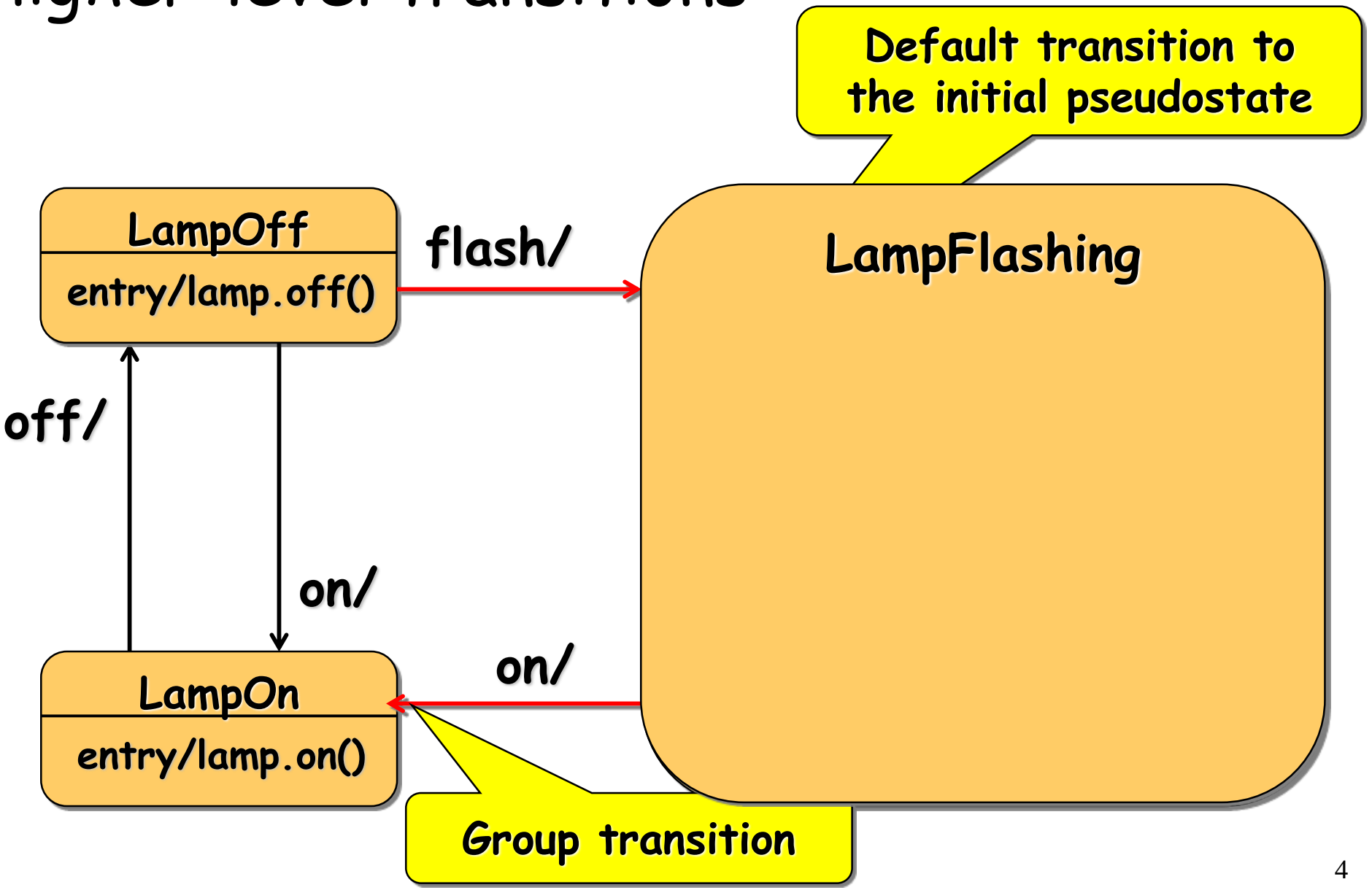
Hierarchical State Machines

- Obtained by replacing transition from every substate by a single transition from superstate…



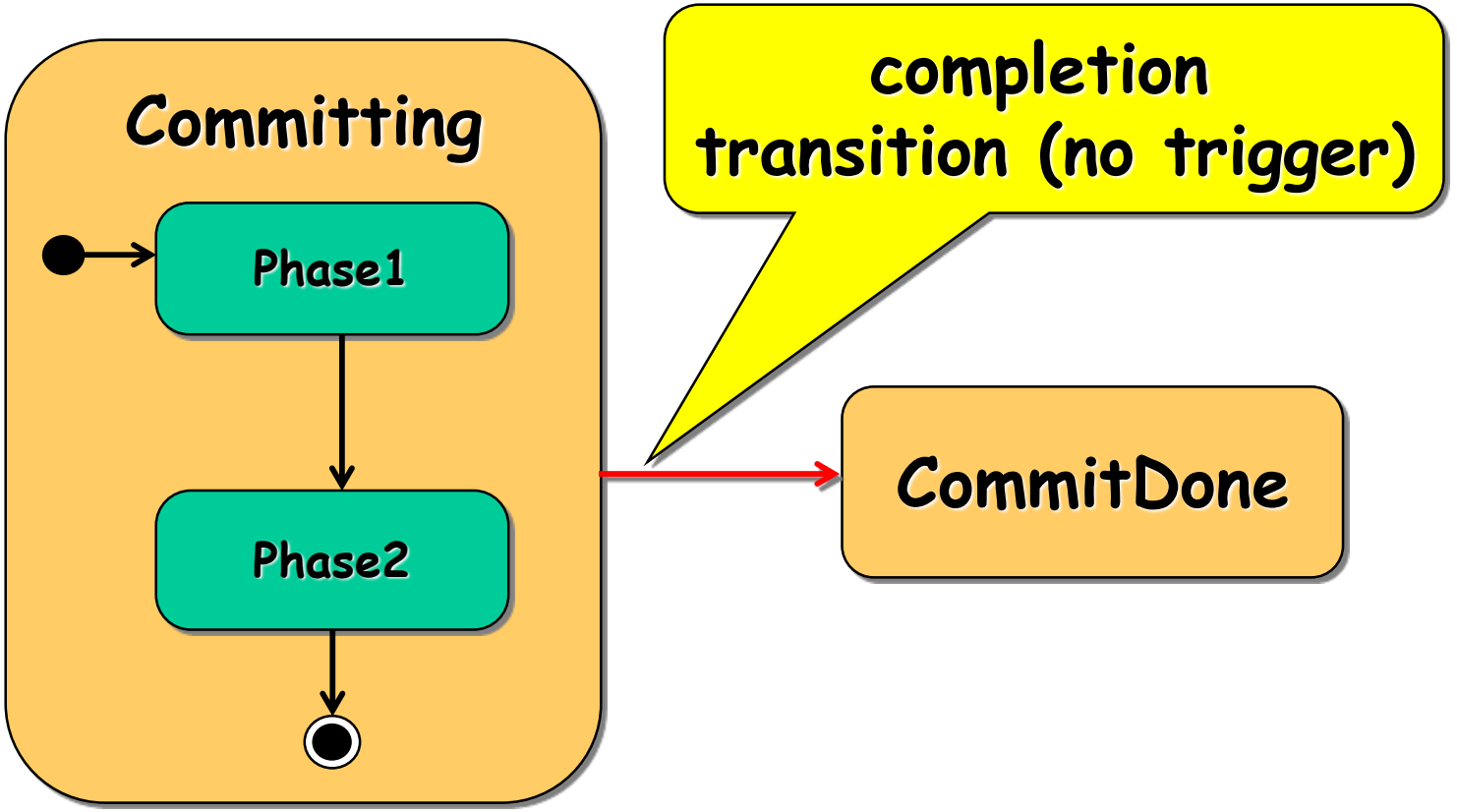
Group Transitions

- Higher-level transitions

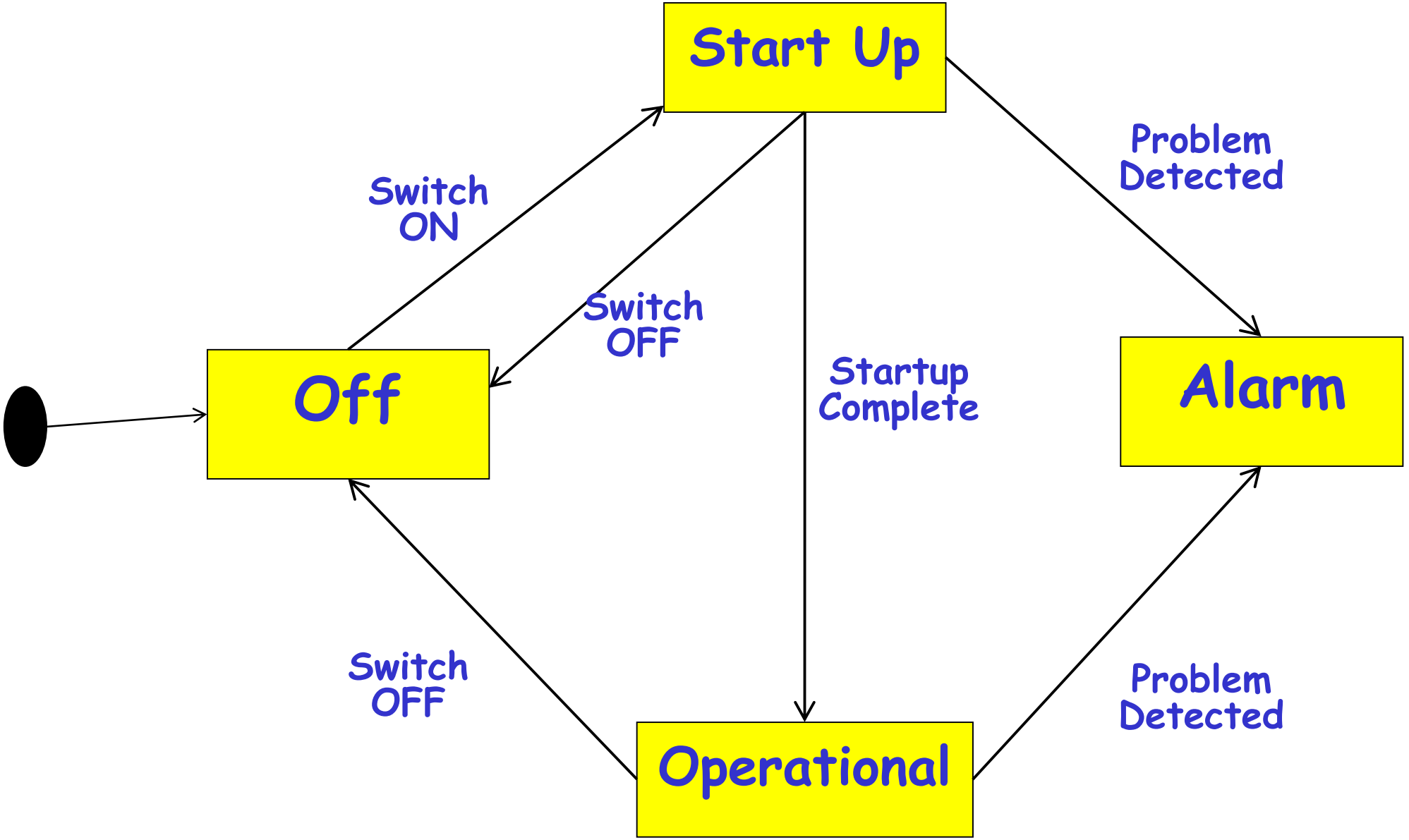


Completion Transitions

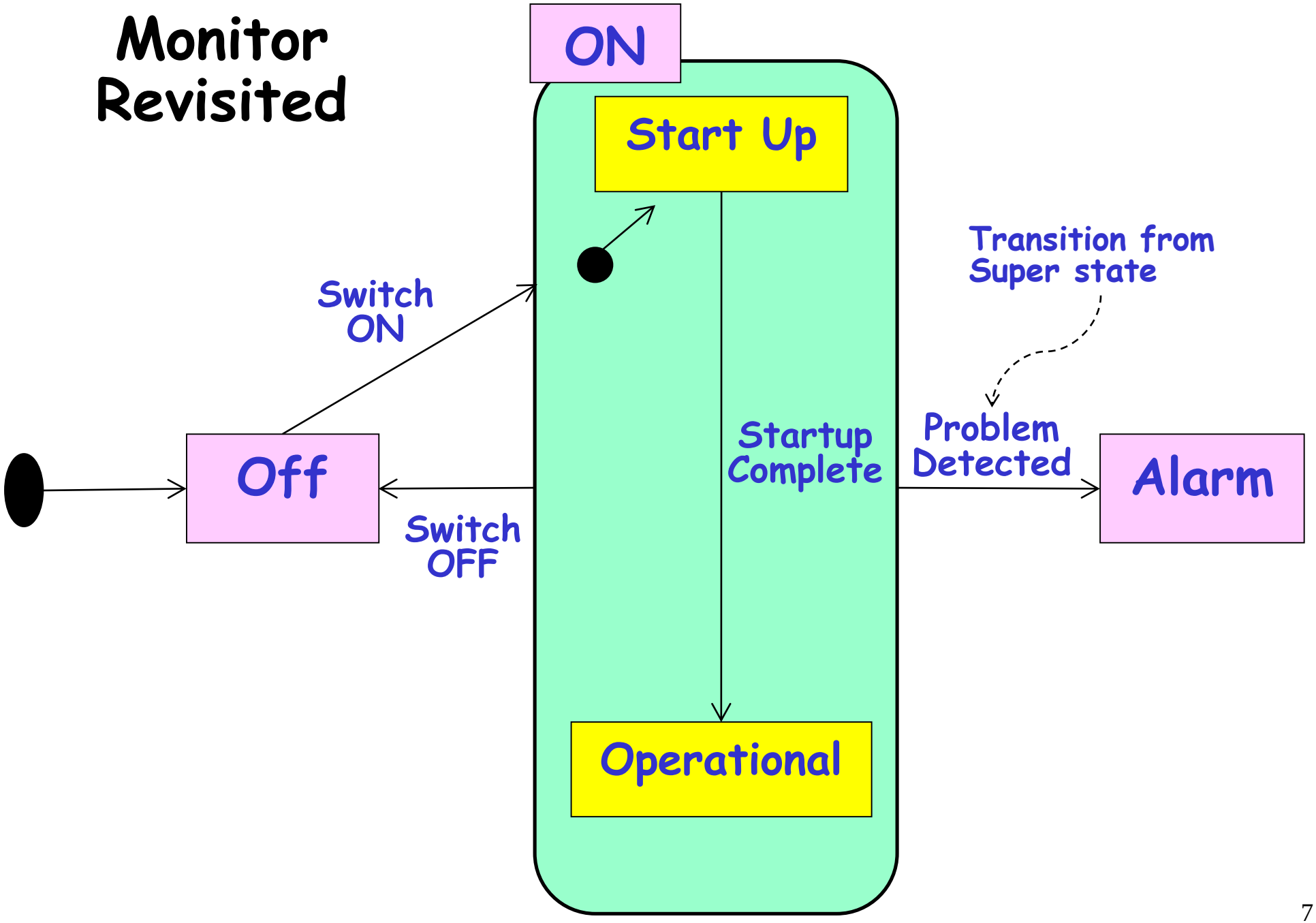
- Triggered by a completion event:
  - Generated automatically when an immediately nested state machine terminates.



A Heart Monitor Application: Convert to Hierarchical State Machine



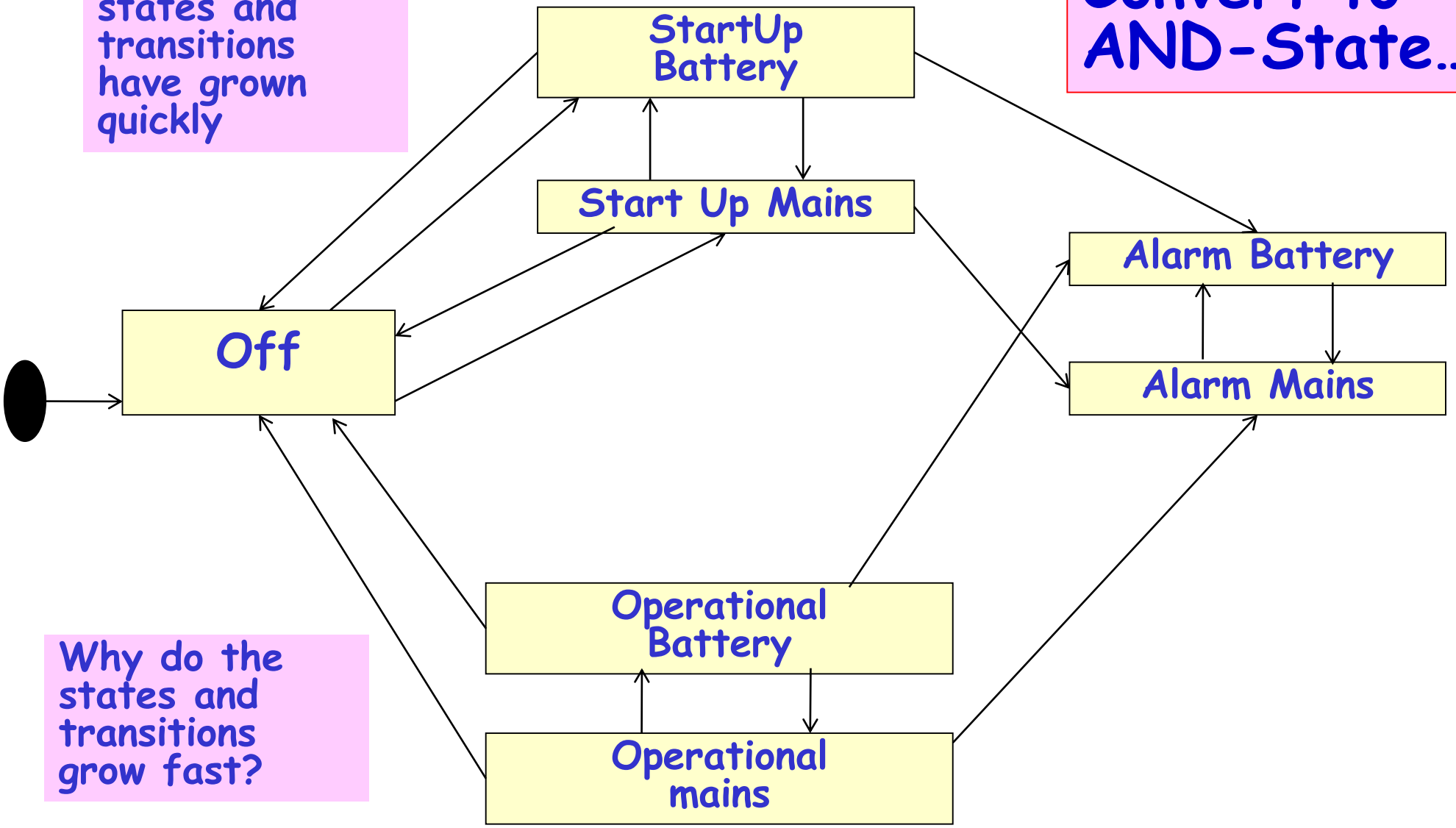
Heart Monitor Revisited



Complete Diagram for Heart Monitor Application

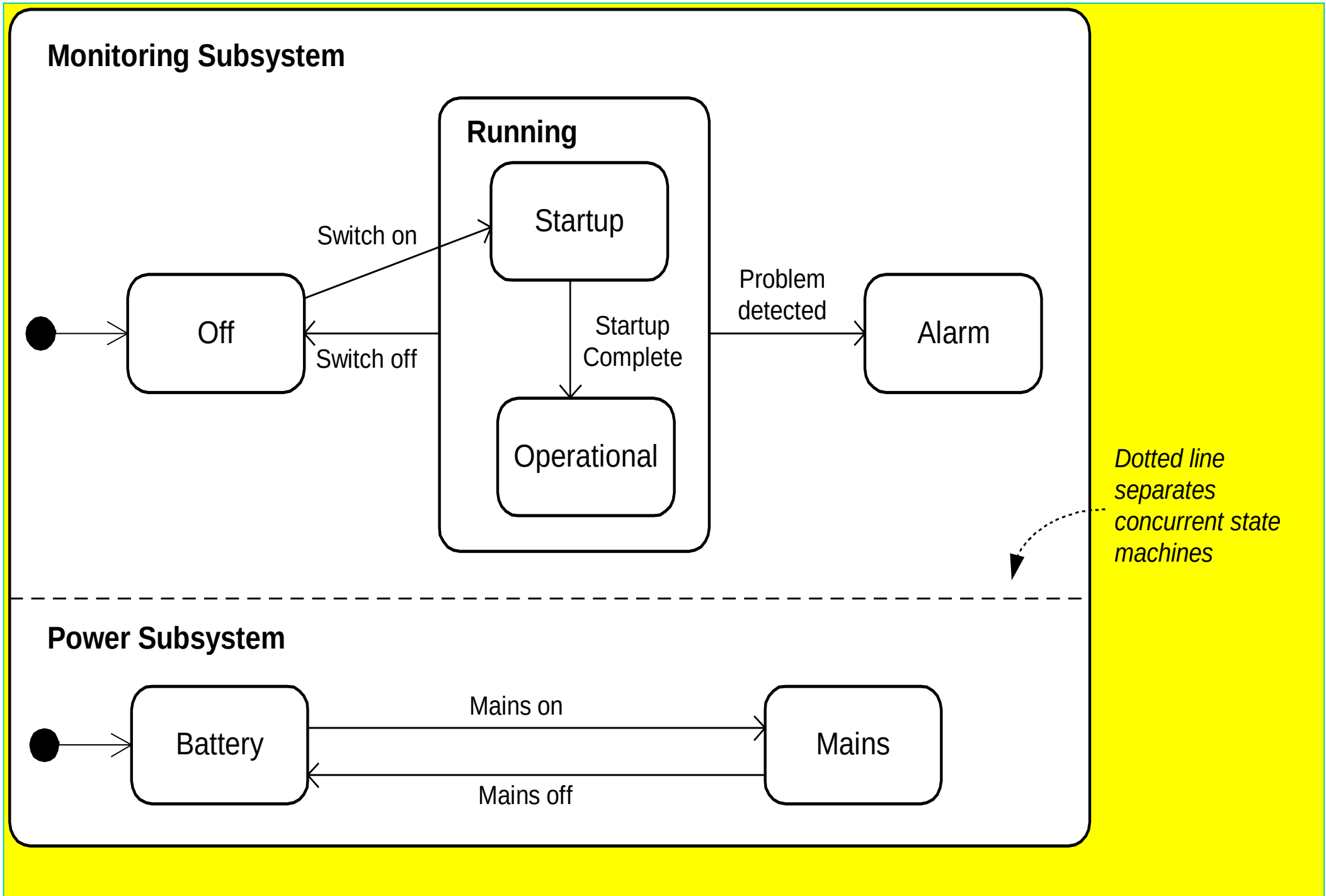
The number of states and transitions have grown quickly

Quiz: Convert to AND-State…

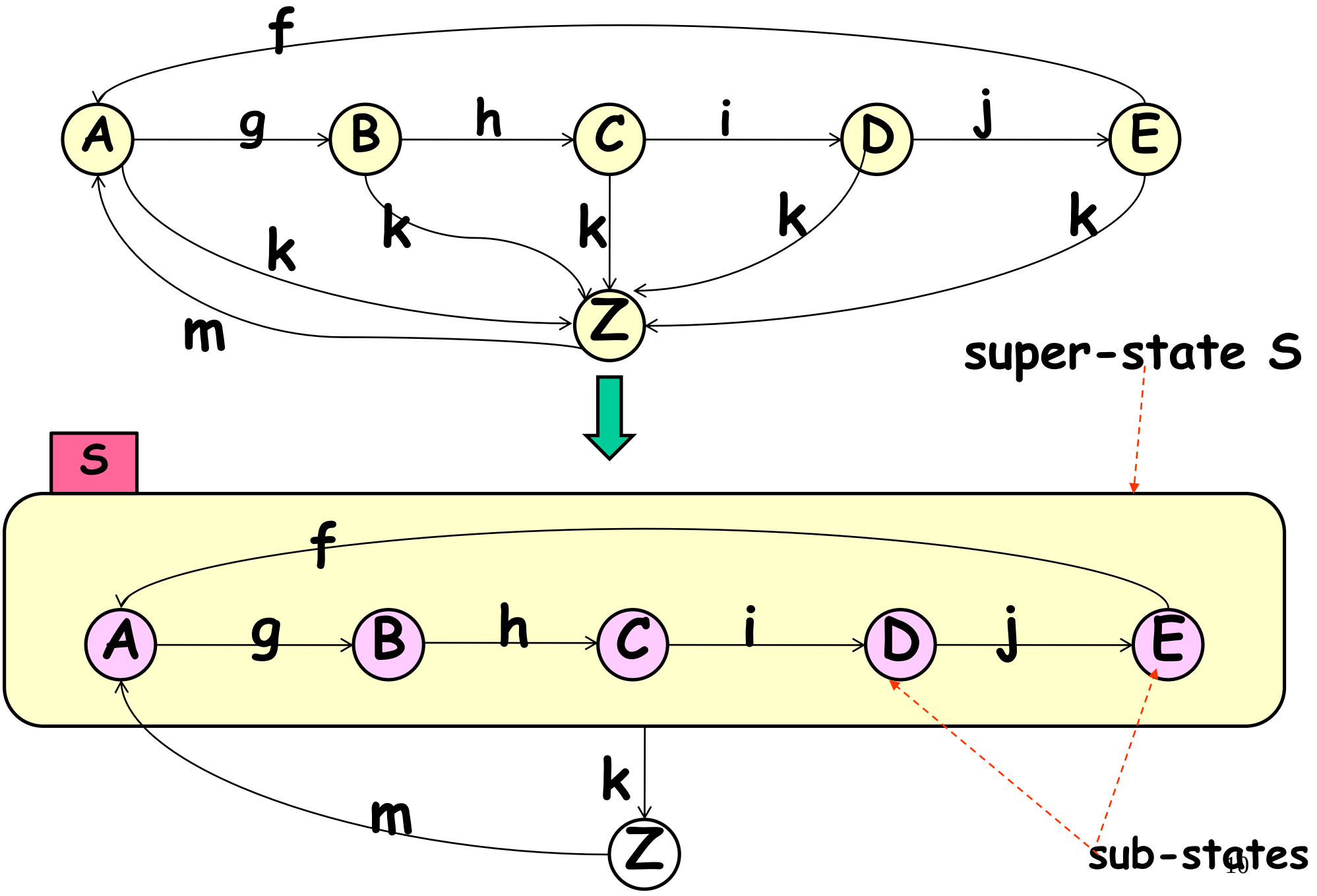


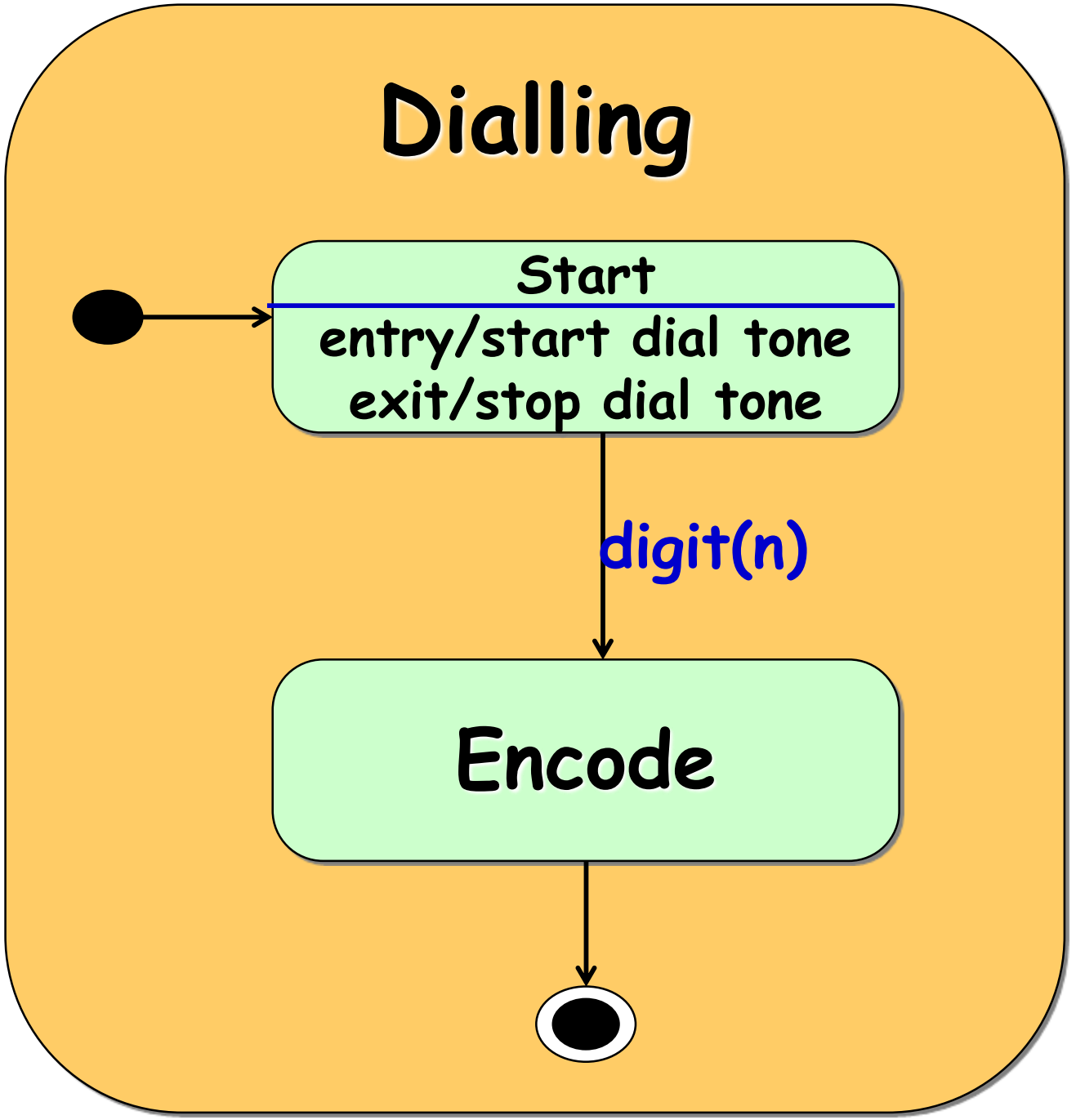
Why do the states and transitions grow fast?

Heart Monitor as Concurrent State Machine



Introducing Hierarchy

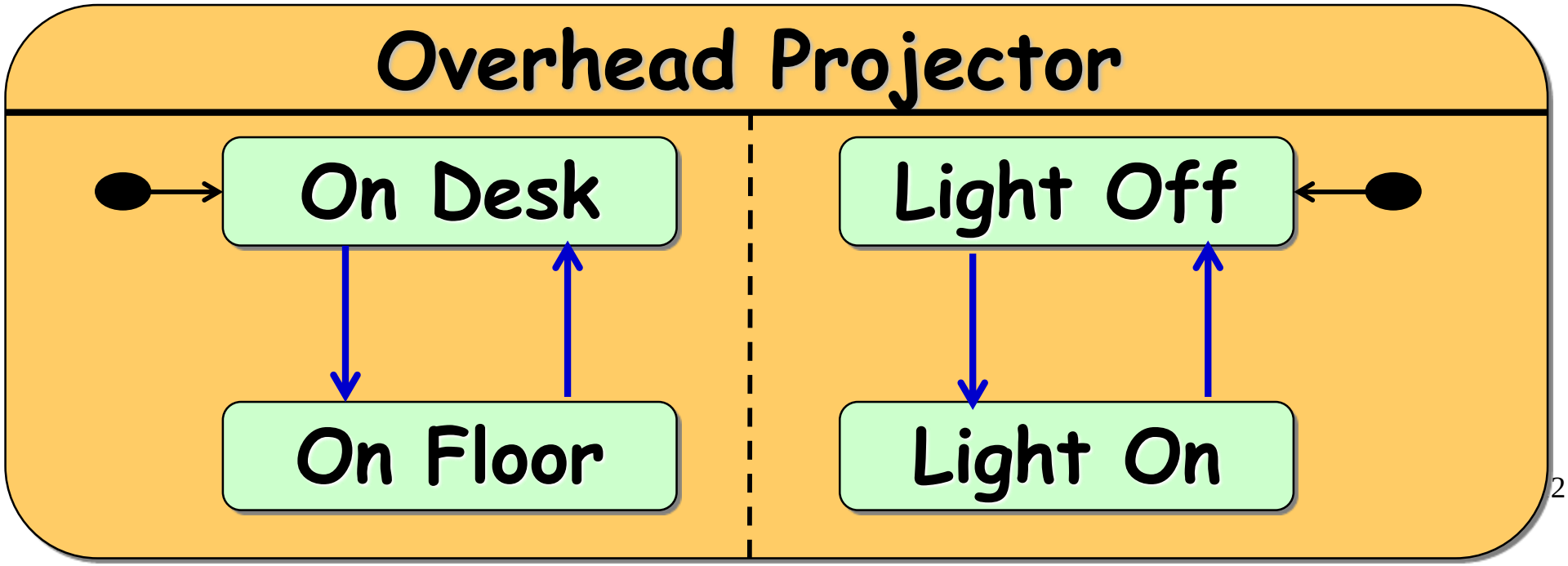




Basic Composite State: OR state

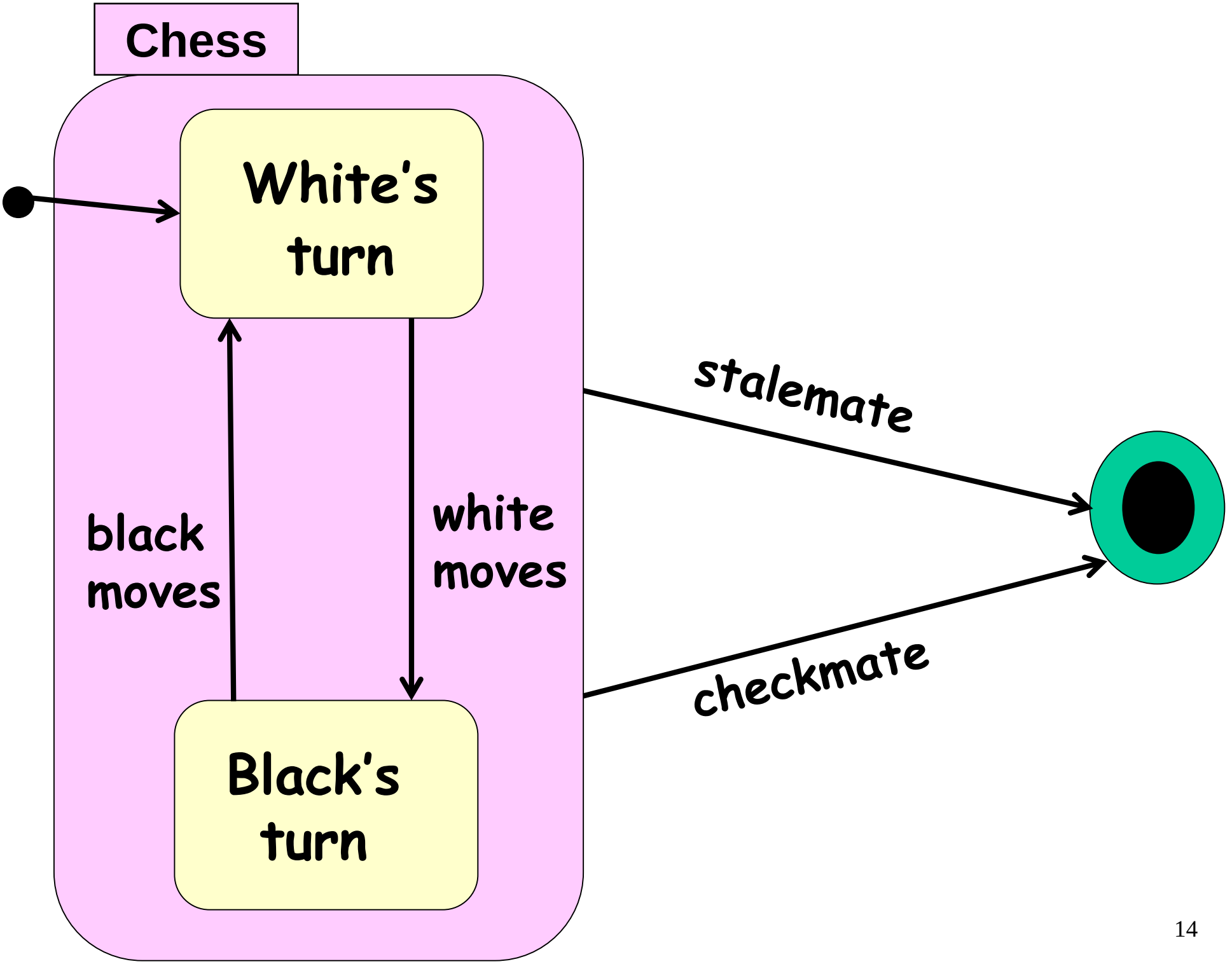
Composite State: AND State

- Composite states can have:
  - **concurrent substates** -**and- relationship**
    - substate separated from others by dotted line
  - **disjoint substates** -**or- relationship**
    - transitions between substates



Exercise 1: Develop State Machine Model

- In a chess game:
  - Black and white sides take turn to play.
- The game ends anytime when:
  - Either there is a checkmate, or
  - There is a stalemate.

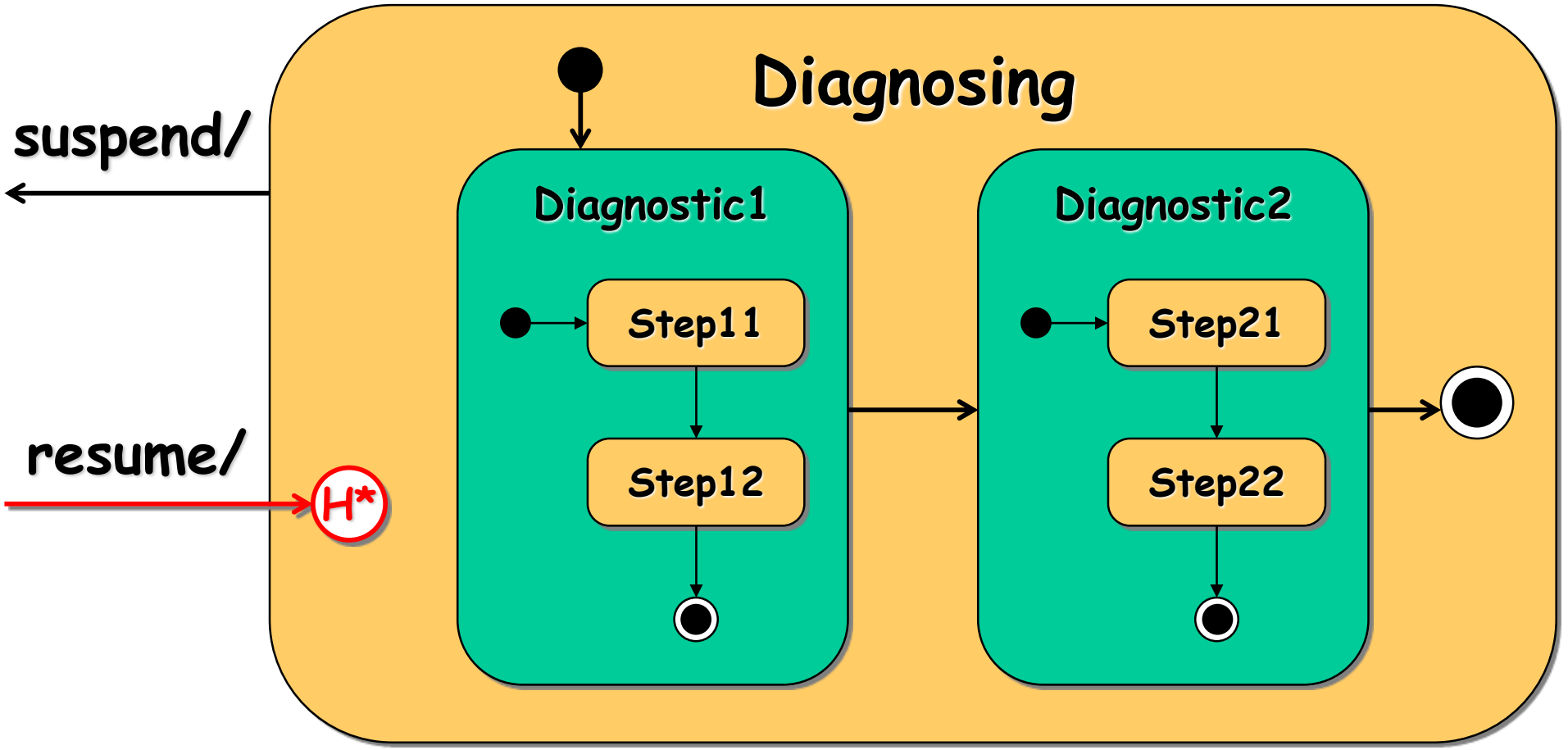

Chess
White's turn
Black's turn
black moves
white moves
stalemate
checkmate

Handling Interrupted States

- Sometimes it is necessary to interrupt a complicated procedure:
  - And then resume exactly from where you stopped
- Consider installing a program on your computer
  - "Out of memory: please delete some files and hit continue"
- In many situations a normal course of processing is **interrupted** and then **resumed**
  - How can this be handled in a nested state diagram?

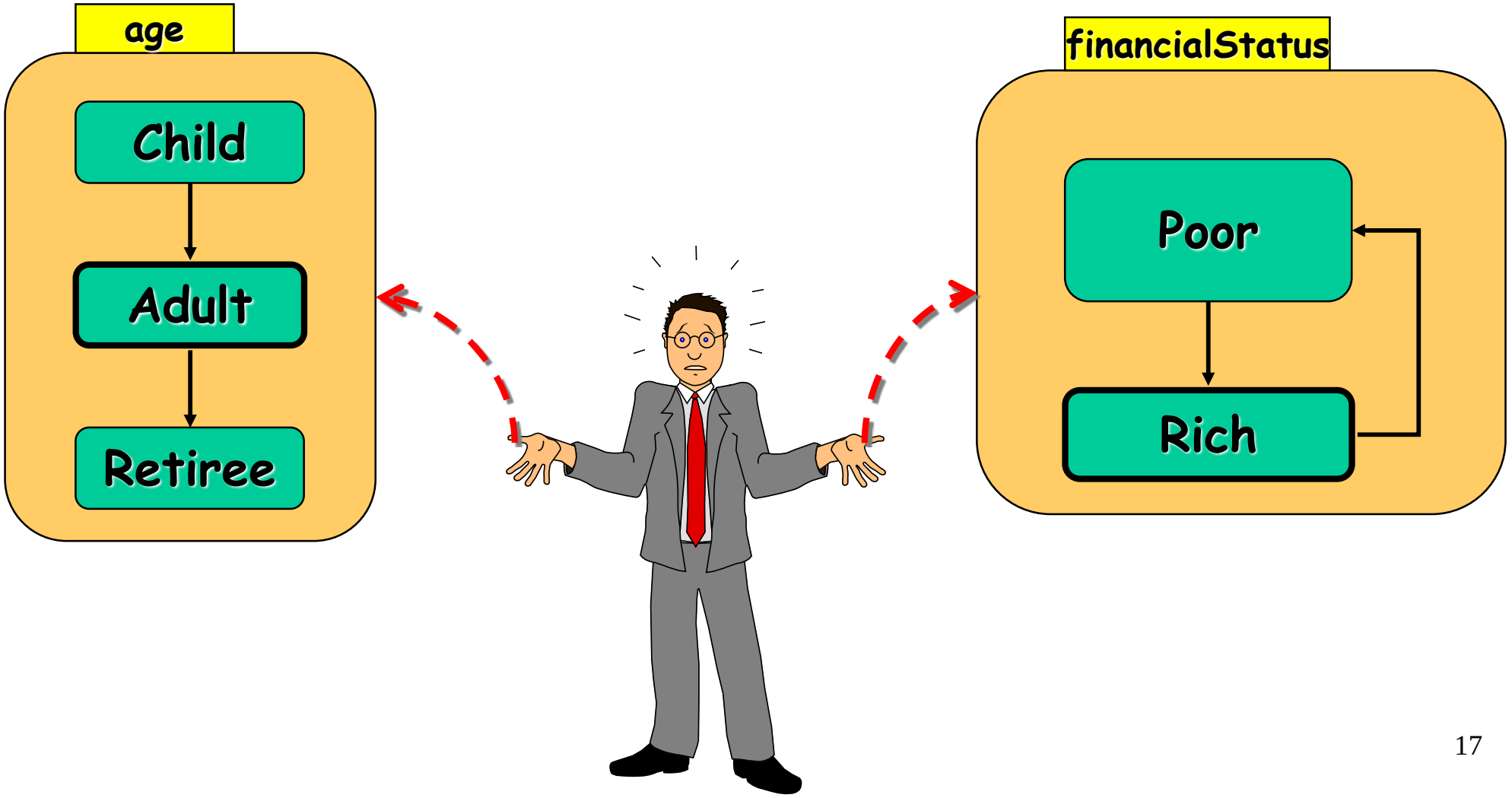
History

- Transit to a higher level state and then return to resume the activity at which transited.



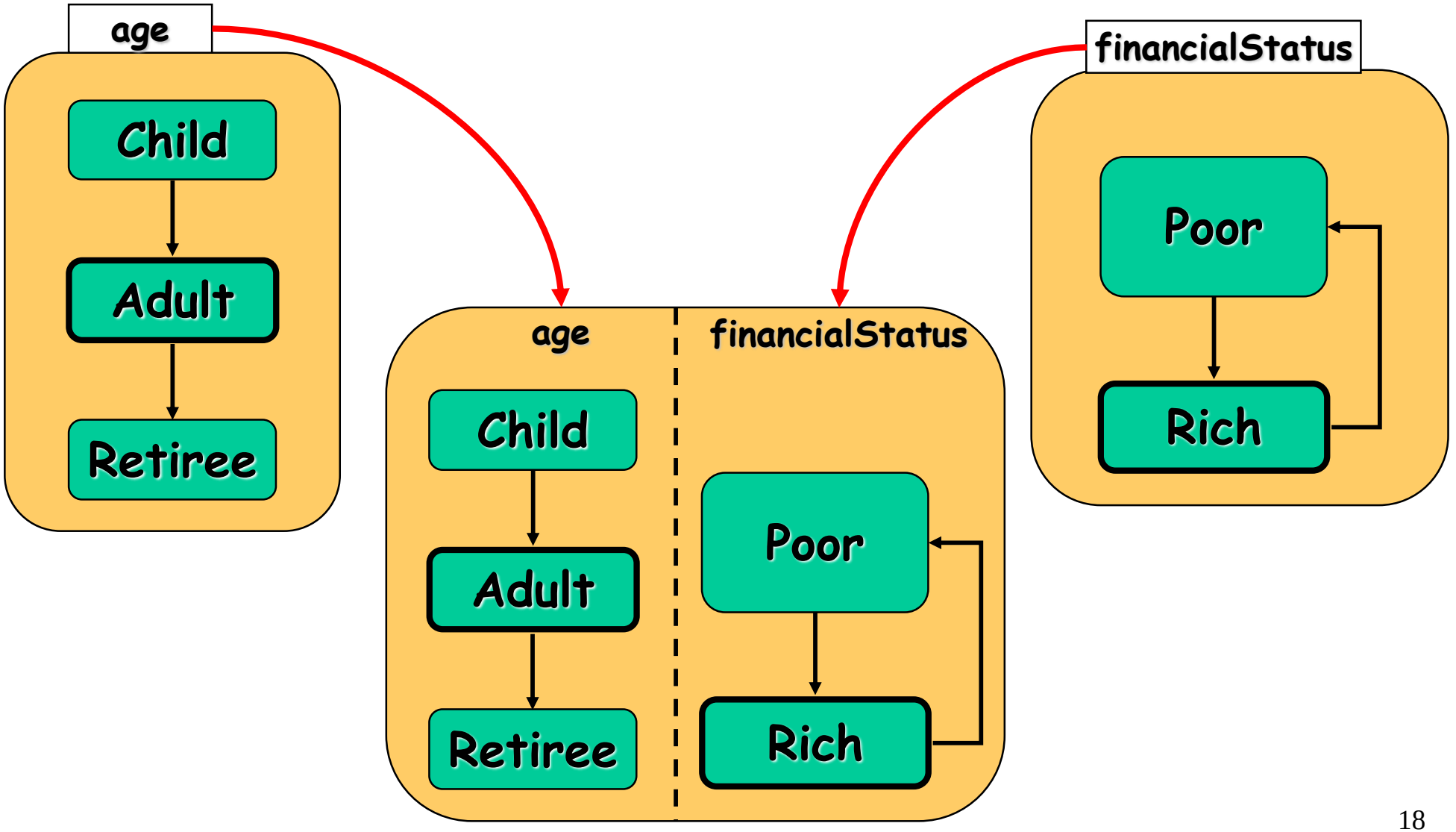
Orthogonality

- Multiple simultaneous perspectives on the same entity.



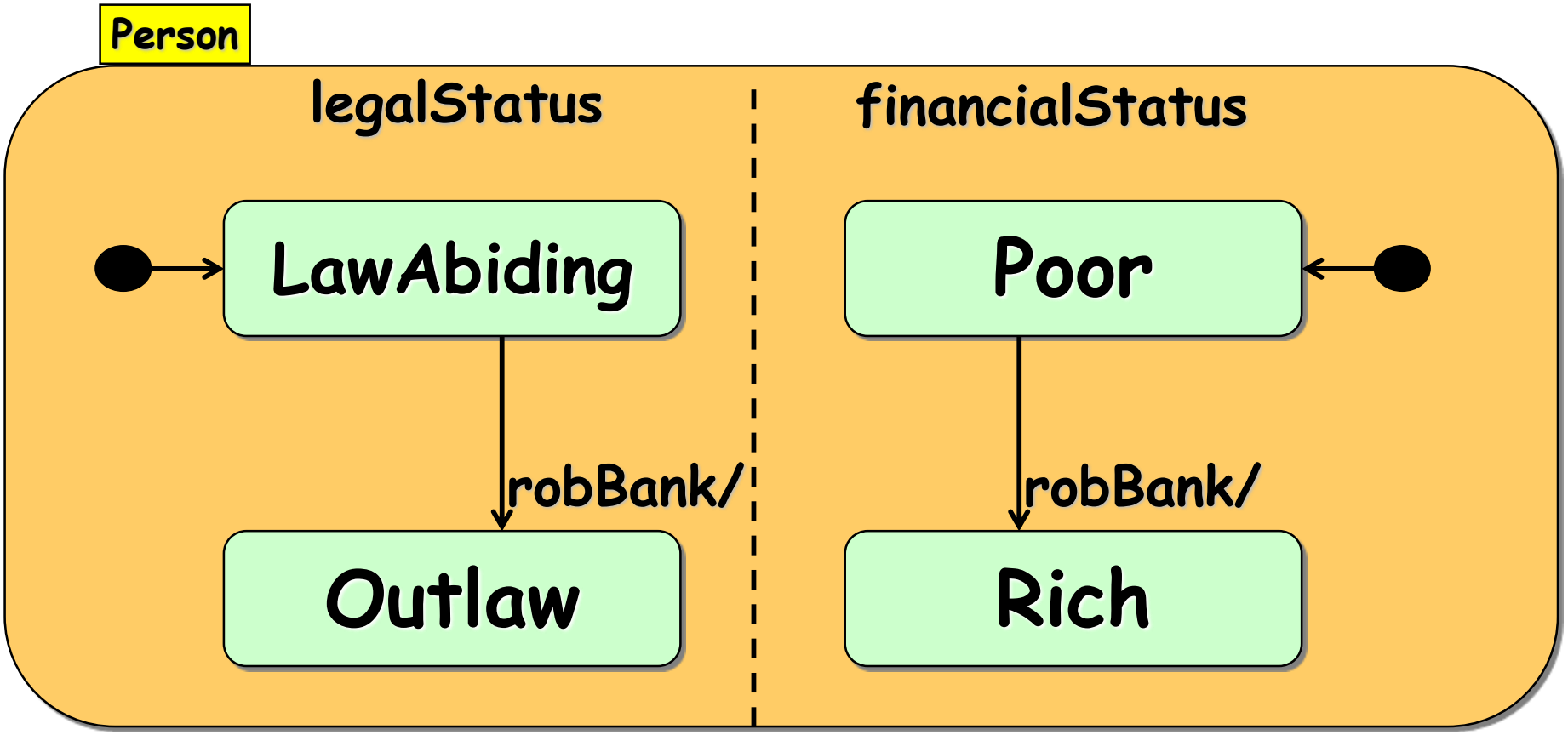
Orthogonal Regions

- Combine multiple simultaneous descriptions



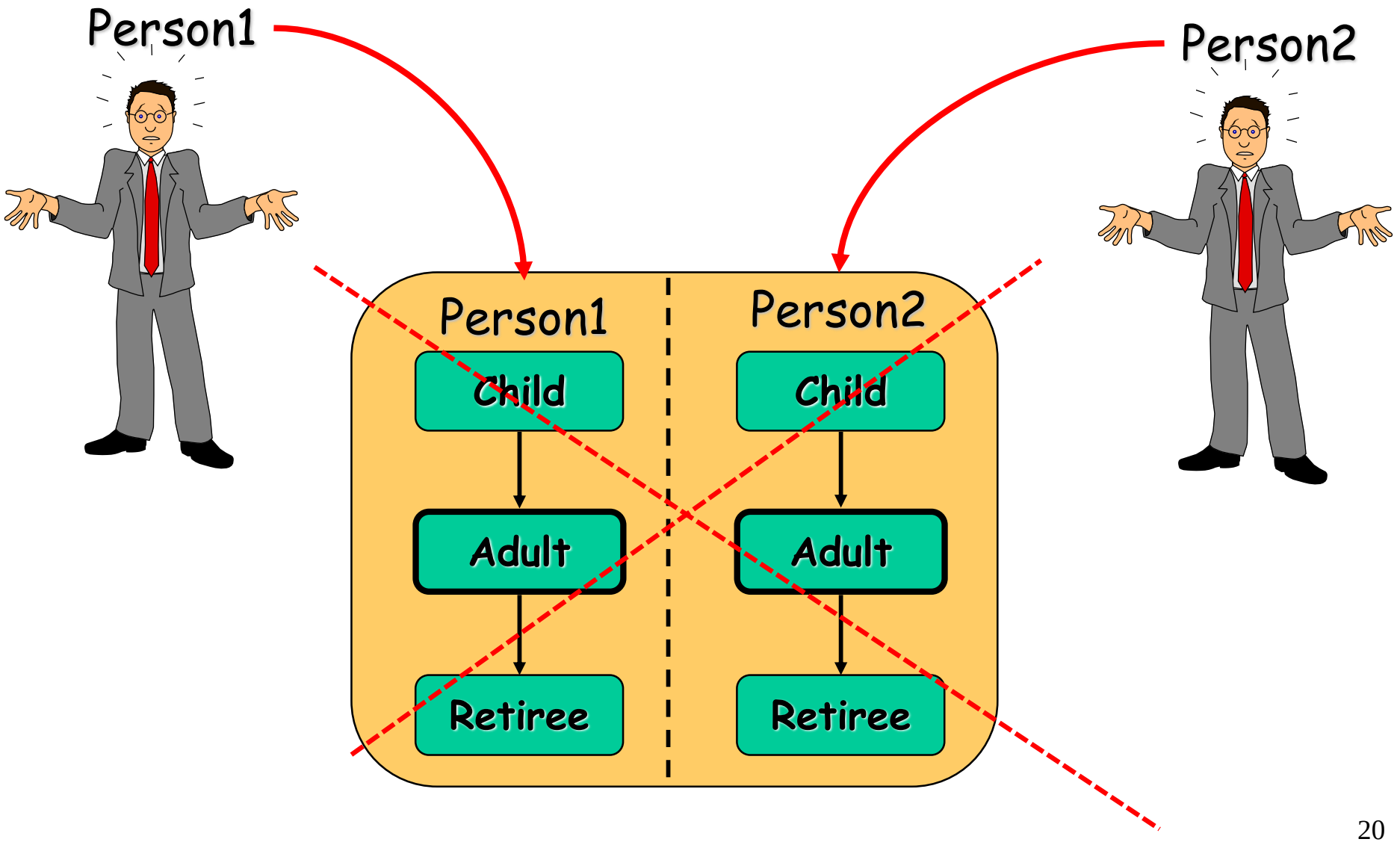
Orthogonal Regions - Semantics

- All mutually orthogonal regions detect the same events and respond to them "simultaneously"
  - Usually reduces to interleaving of some kind.



Common Misuse of Orthogonality

- Using regions to model independent objects



Exercise : Simple Elevator System (Partial)

- The elevator is by default at the ground floor.
- When idle and a button at the same or a higher floor is pressed:
  - It moves up when button at any upper floor is pressed and halts when the requesting floor is reached.
- When idle and a button at the same or a lower floor is pressed:
  - It moves down. Halts when requesting floor is reached.
- If inactive at a floor for more than 10 minutes:
  - It moves down to the ground floor.

Simple Elevator System

Lift Call

Lift Call

Lift Call

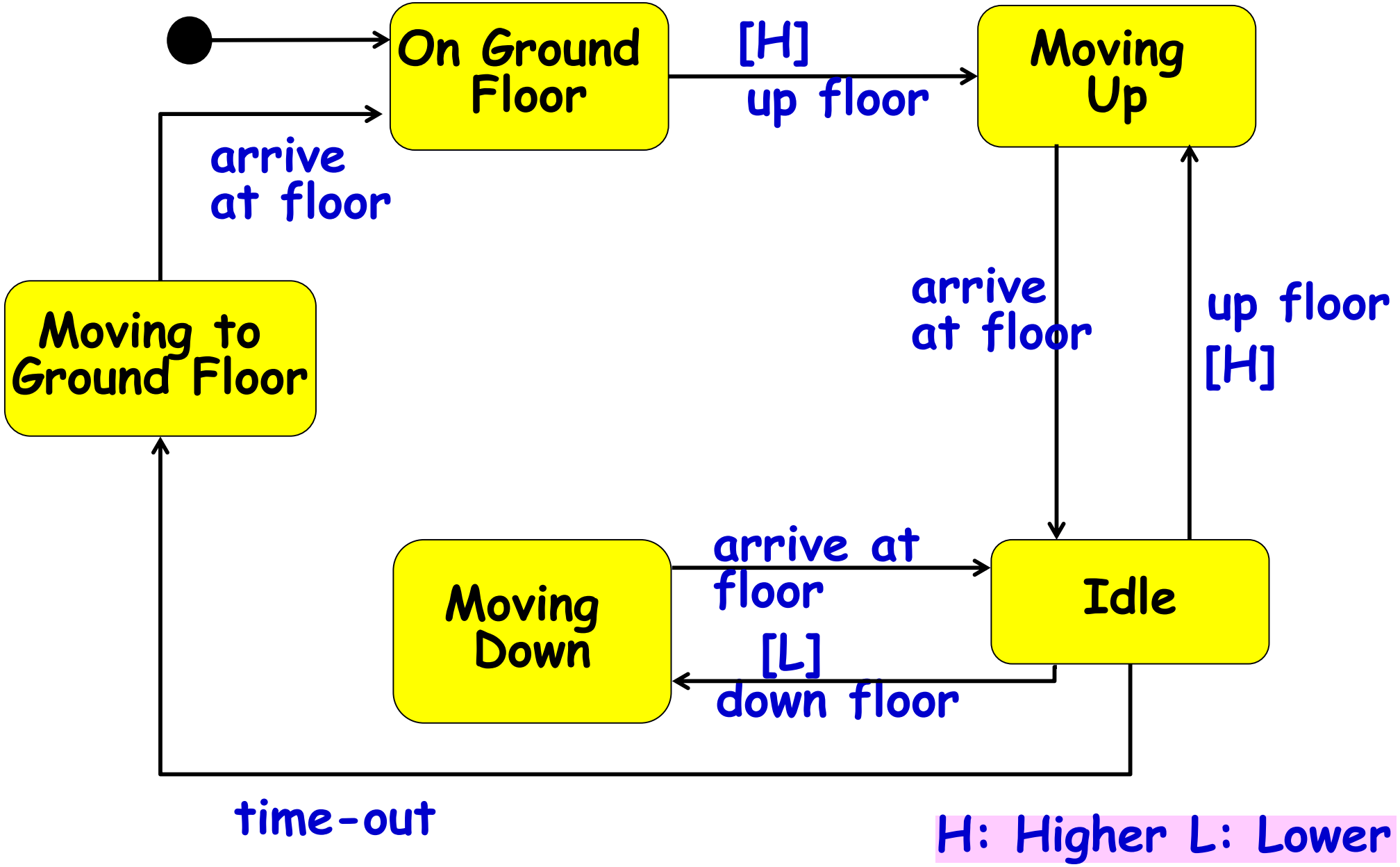
Lift Call

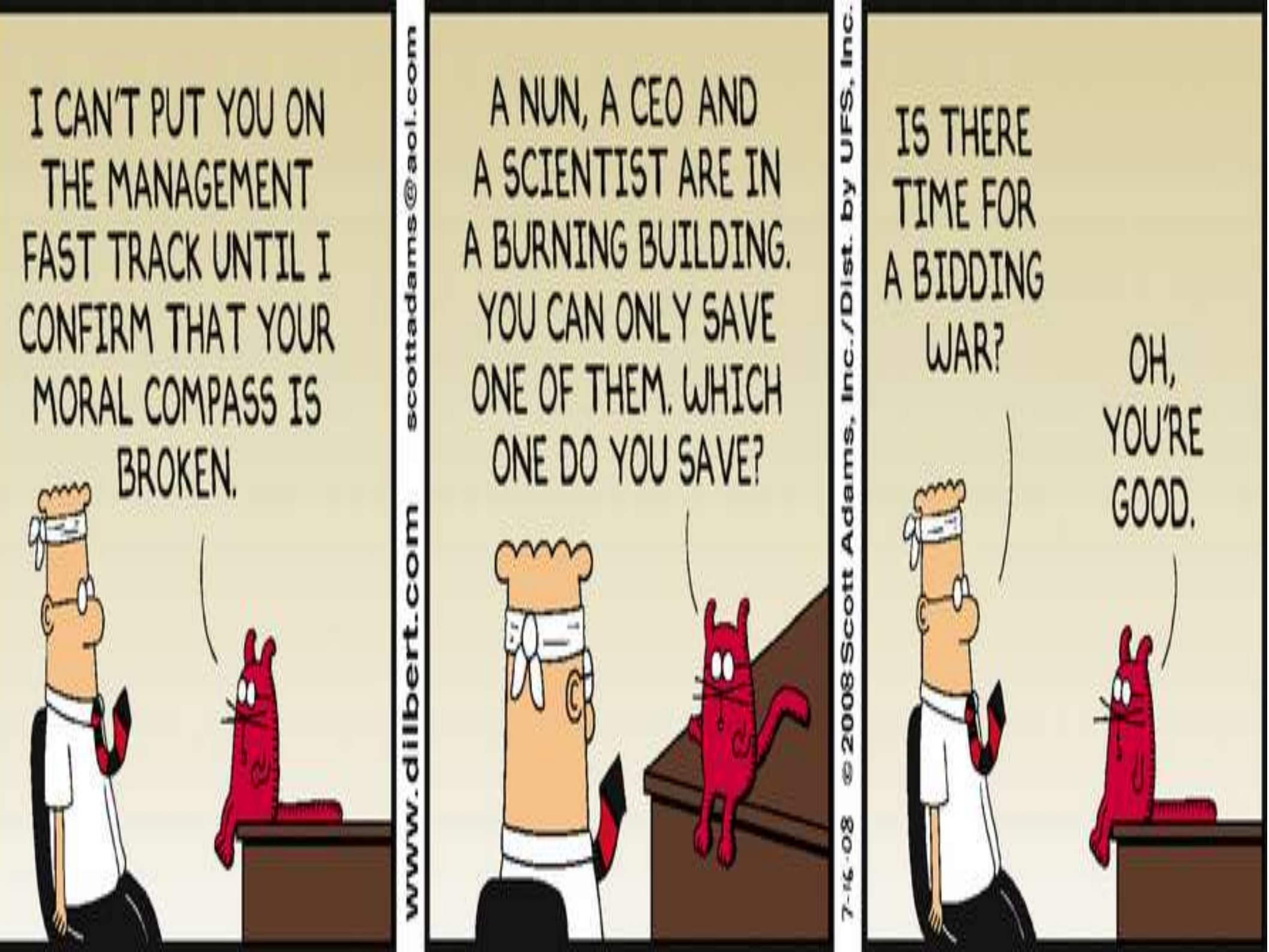
Lift Call

Lift Call

Lift Call

Elevator: State Machine Representation




I CAN'T PUT YOU ON THE MANAGEMENT FAST TRACK UNTIL I CONFIRM THAT YOUR MORAL COMPASS IS BROKEN.
www.dilbert.com
scottadams@aol.com
A NUN, A CEO AND A SCIENTIST ARE IN A BURNING BUILDING. YOU CAN ONLY SAVE ONE OF THEM. WHICH ONE DO YOU SAVE?
7-16-08 © 2008 Scott Adams, Inc./Dist. by UFS, Inc.
IS THERE TIME FOR A BIDDING WAR?
OH, YOU'RE GOOD.

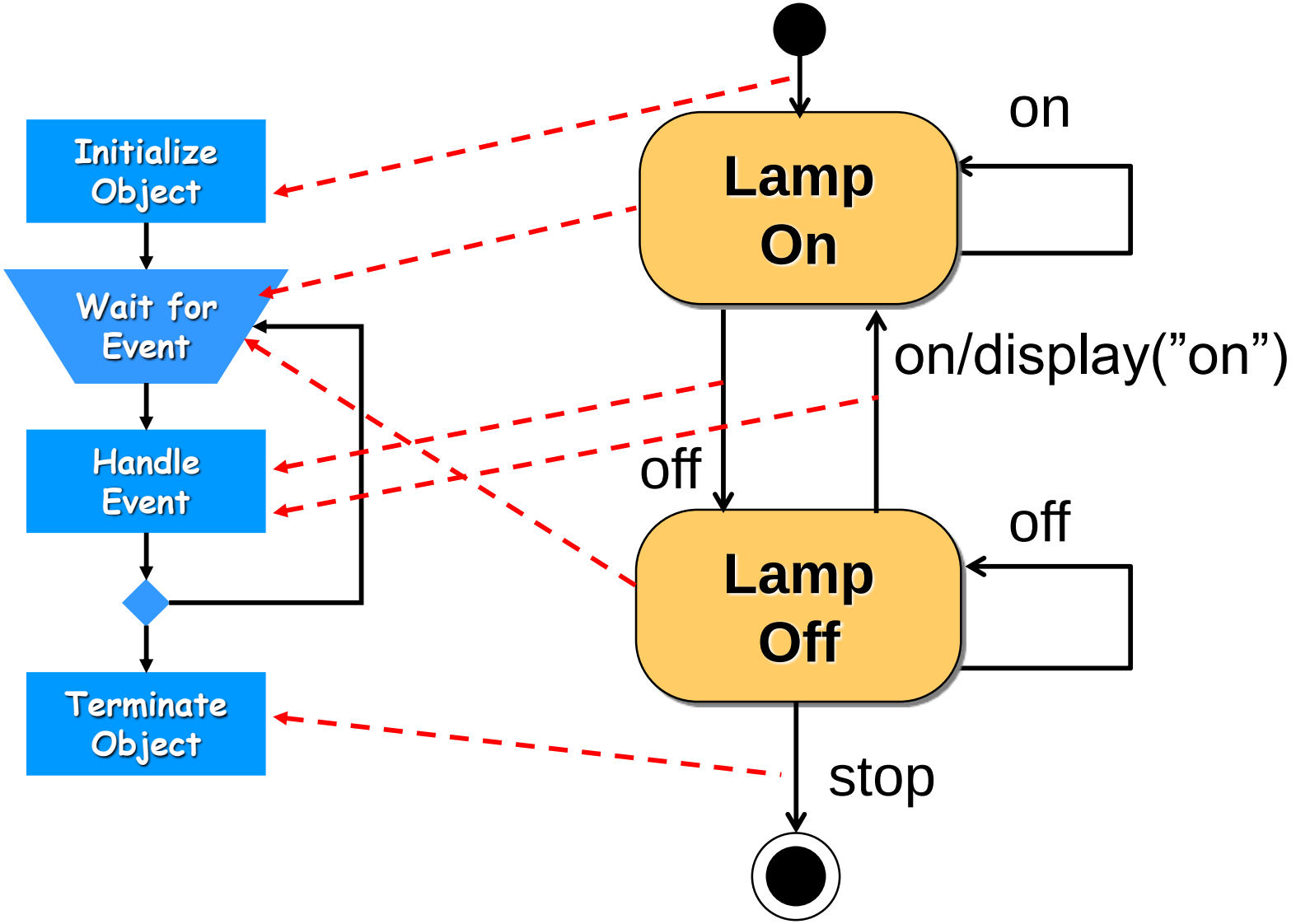
Encoding State Machine Diagrams

Translating State Machine to Code

- **Complicacy:**
  - State machines have many concepts that are not directly supported in programming languages ---- neither procedural nor OO languages, for example:
    - **States, events, transitions, composite states, concurrent states, history, etc**

General Idea: How to Generate Code?

Mapping State Model to Program Code



How to Encode an FSM?

- Three main approaches:
  - **Doubly nested switch in loop**
  - **State table**
  - **State Design Pattern**

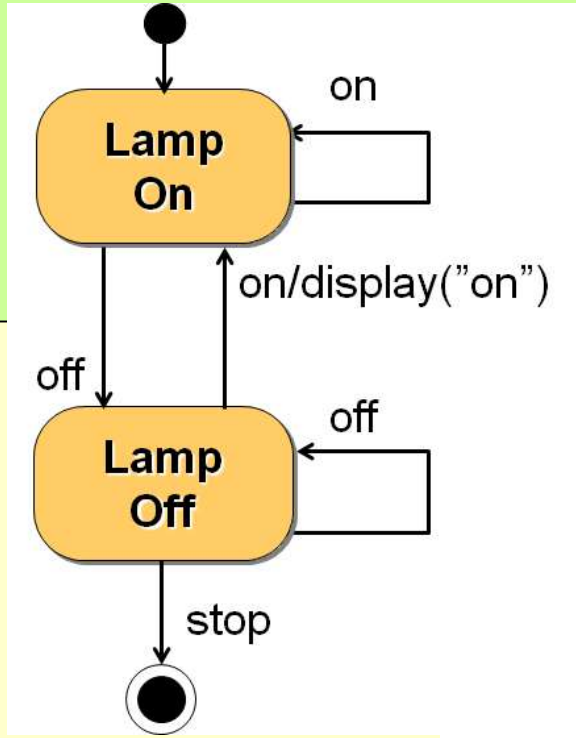
3 Principal Ways

- **Doubly nested switch in loop:**
  - Scalar variables store state --- Used as switch discriminator for first level switch
  - Event type is discriminator in second switch
  - Hard to handle concurrent states, composite state, history, etc.
- **State table:** Straightforward table lookup
- **Design Pattern:**
  - States are represented by classes
  - Transitions handled by methods in classes

Doubly Nested Switch Approach

```
int state, event; /* state and event are variables */

while(TRUE){
   switch (state){          /* Wait for event */
     Case state1: switch(event){
                  case event1:  state=state2; etc…; break;
                  case event2:…
                  default:
                  }
     Case state2: switch(event){…
                  }
     ….
   }}
```



State Table Approach

- From the state machine, we can set up a state transition table with a column for the actions associated with each transition

Present state	Event	Next state	Actions
BTN_off	e1	BTN_off	none
	e2	BTN_on	set red LED flashing
BTN_on	e1	BTN_on	none
	e2	BTN_off	reset red LED flashing


ASOK, THE BEST WAY TO ADVANCE YOUR CAREER IS BY LEARNING AS MUCH AS YOU CAN FROM CO-WORKERS.
Dilbert.com DilbertCartoonist@gmail.com
CAN YOU SHOW ME WHAT YOU ARE DOING?
COME BACK IN TEN YEARS WHEN I'M NOT BUSY.
9-12-14 ©2014 Scott Adams, Inc./Dist. by Universal Uclick
NO ONE WILL HELP ME LEARN ANYTHING.
YOU LEARNED THAT!

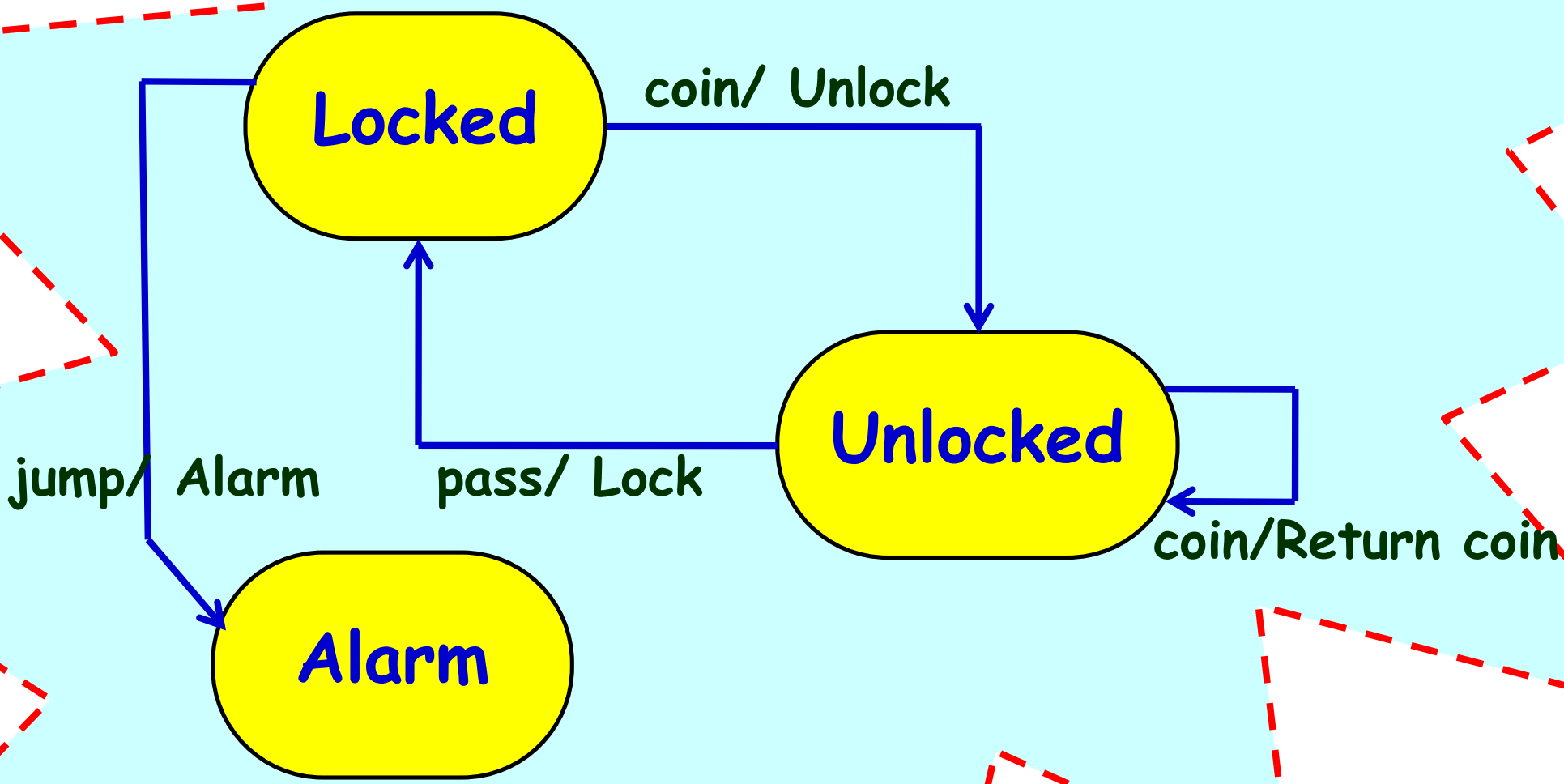
Turnstile: Exercise 2

- A turnstile check gate would be installed at a railway station.
- When the turnstile is powered on:
  - It starts in the locked state.
- When a commutator drops a one rupee coin:
  - The gate gets unlocked and a message "Gate Open" appears.
  - Exactly one person is allowed to pass through before the gate gets locked.
  - If more than one person try to pass through, an alarm is sounded and gate gets locked.
- Also when any one tries to jump over the gate:
  - An alarm is started and gate is locked.

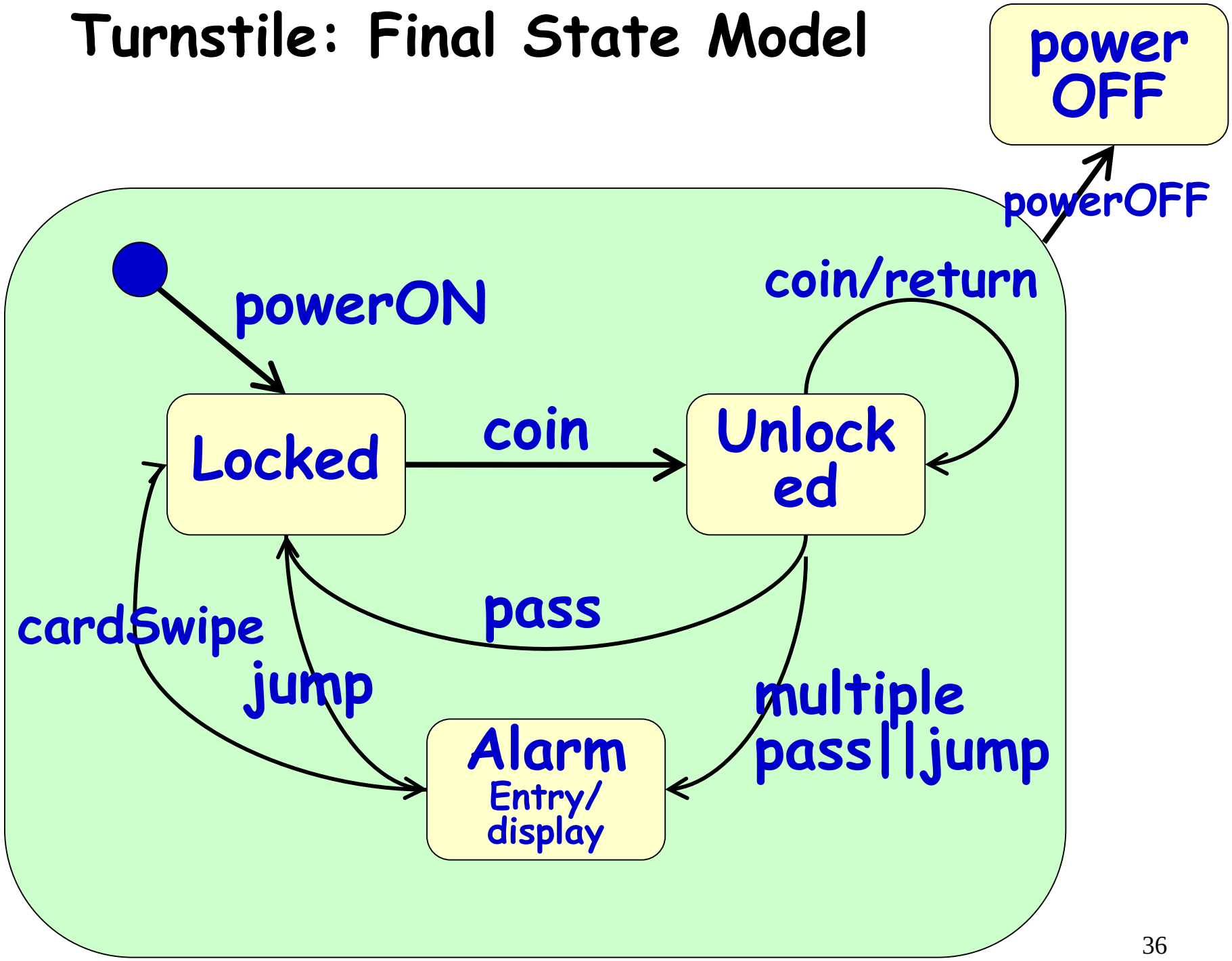
Turnstile Exercise cont…

- When the alarm is ON:
  - A message **"Please wait: Gate temporarily blocked"** is displayed.
  - If any one still inserts a coin, it return the coin without unlocking the gate.
  - The alarm is reset when an attendant swipes a card and the gate starts at the locked state.
- When any one inserts a coin when the gate is already open:
  - The coin is returned.
- If there is a power failure any time:
  - The gate gets locked and **"Power fail: inoperative"** message is displayed and coins are not accepted.

Turnstile: First-Cut State Model



Turnstile: Final State Model



```
enum State {Locked, Unlocked, Alarm, PowerOFF};
enum Event {Pass, Coin, multiplePass, jump, cardSwipe};
static State s = Locked;
void Transition(Event e){
        switch(s){
                case Locked:
                                switch(e){
                                        case Coin:
                                                s = Unlocked;
                                                Unlock();
                                                 break;
                                        case Jump:
                                                  s=Alarm;
                                                 Alarm();
                                                  break;
                                        }
                 break;
```

C Code

cont…

```
case Unlocked:
   switch(e){
     case Coin: returnCoin();
                break;
     case Pass: s = Locked;
                Lock();
                 break;
     case jump:
     case multiplePass: s=Alarm;
                 Alarm();
                 break;}
    break;
    case Alarm: …. … ….
```

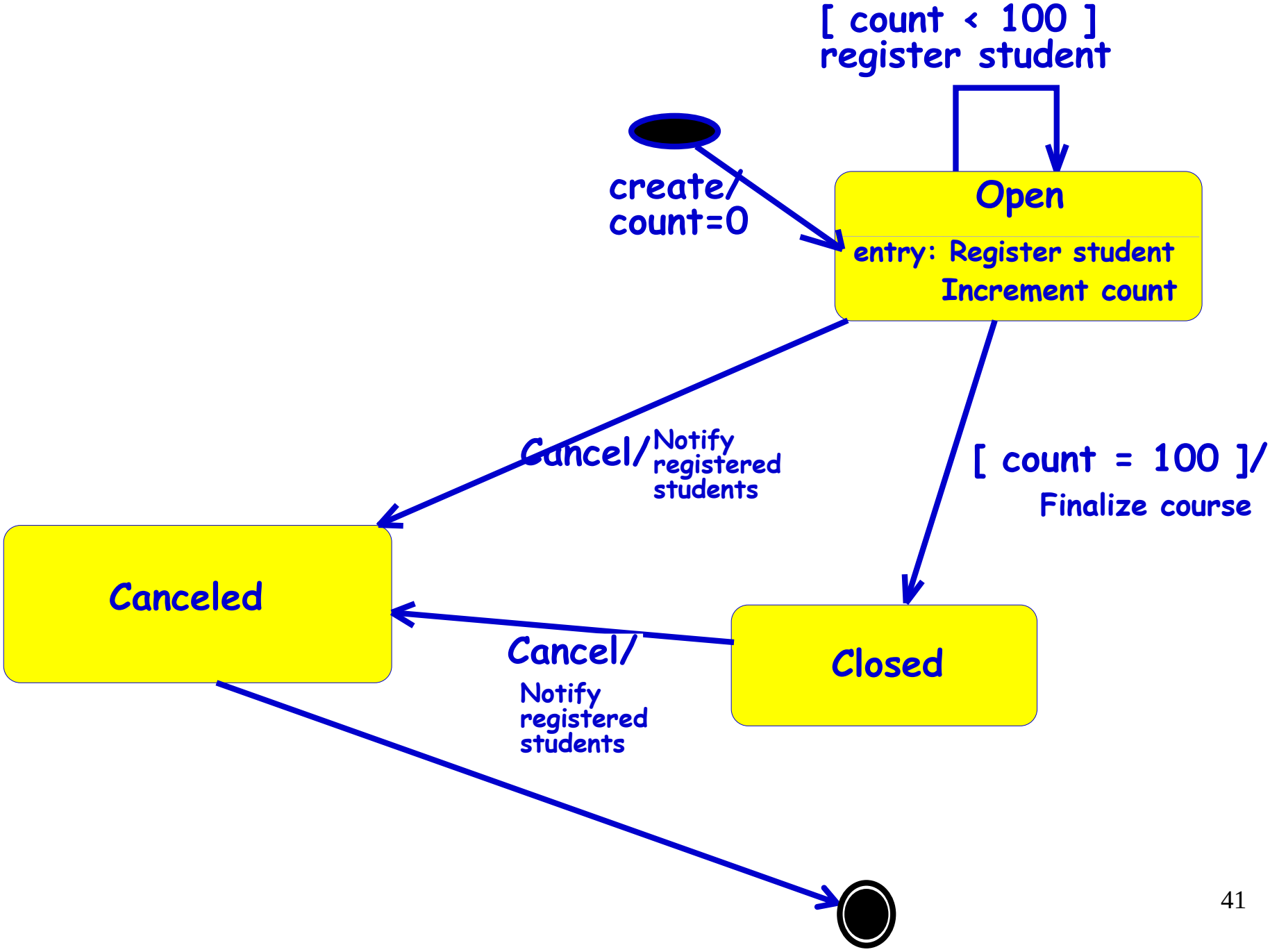
```
public class Turnstile{
enum state{ Locked, Unlocked, Alarm}
public Turnstile(){ state=Unlocked;}
public pass(){
  if(state== Unlocked) state=Locked;}
public coin(){
        switch(state){
              case Locked: state=Unlocked; break;
              case  Unlocked: returnCoin(); break;
              }}
public jump(){…}
…}
```

Java Code

Ex. 3: Course Registration Software

- When a course is created, it is open for registration, and number of students is made 0.
- If the course is open and a student registers, the student count incremented.
- A course can be cancelled anytime:
  - The registered students are notified and their registration cancelled.
- If registered student number reaches 100:
  - The course is closed for registration, and it is allocated a room and a time slot.

Exercise 3: State Transition Diagram



Home Work: ATM

- To withdraw money:
  - customer inserts the ATM card.
- Authentication is done by checking an entered PIN.
- If PIN is incorrect:
  - Customer asked to enter PIN again
  - On three unsuccessful attempts, the card is seized and account is locked.
- On correct entry amount requested is dispensed and card ejected.

Home Work

- How to handle code generation for composite AND and OR states?
- How to generate code for history state?

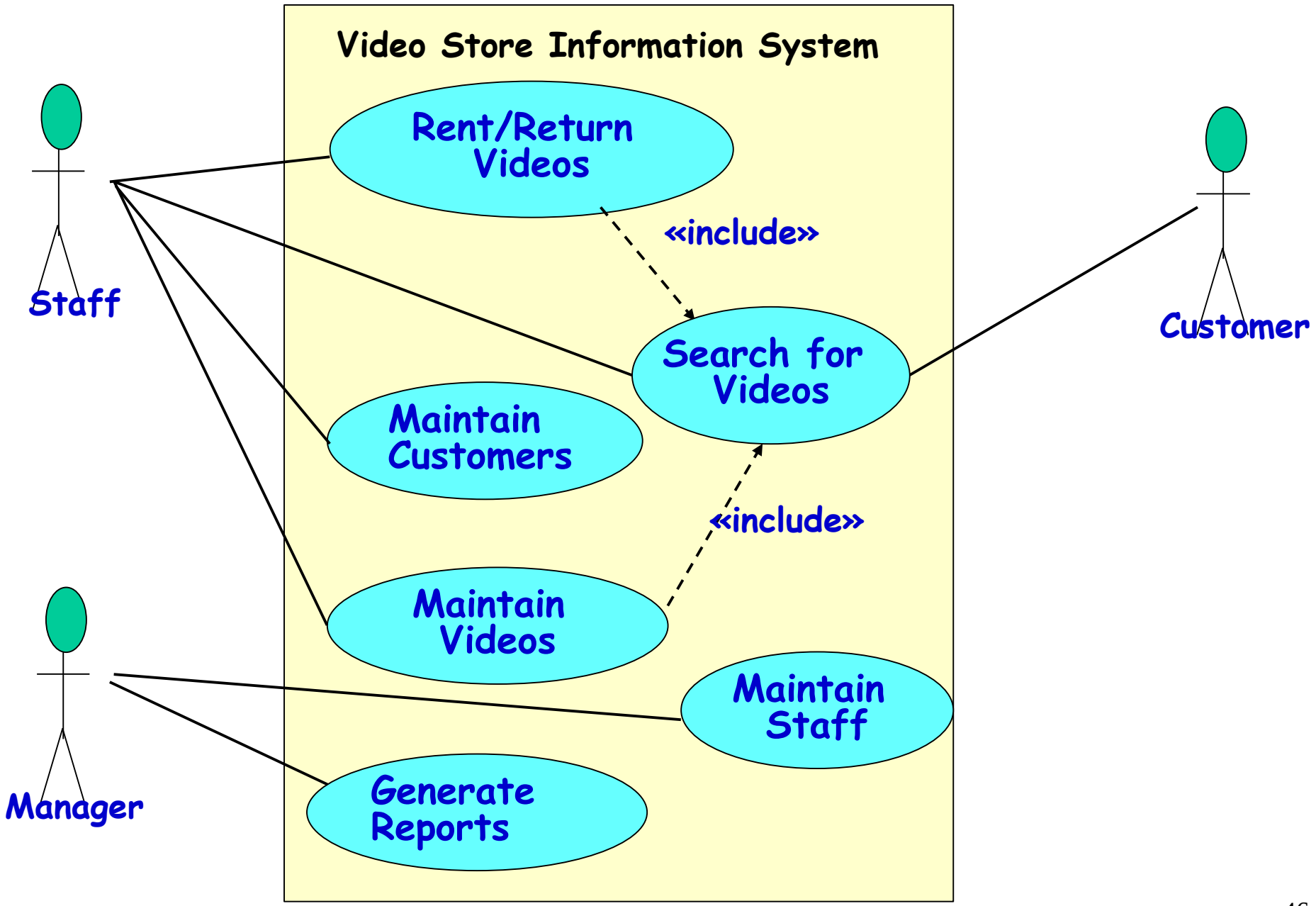
Use Case Modelling

Diagnostic: Video Store Information System

- A Video Store Information System needs to support the following business functions:
  - Recording information about videos the store owns
    - This database is searchable by staff and all customers
  - Find the videos borrowed by a customer
    - Access by staff and also the customer. It involves video database searching.
  - Staff can record video rentals and returns by customers. It involves video database searching.
  - Staff can maintain customer, video and staff information.
  - Managers of the store can generate various reports.

Draw Use Case Diagram

Diagnostic Problem: Solution



Use Case Model of a Problem

- Consists of a set of "use cases"
- It is the central model:
  - Other models must conform to this model…
  - Not really an object-oriented model, it is functional model of a system.

What is a Use Case?

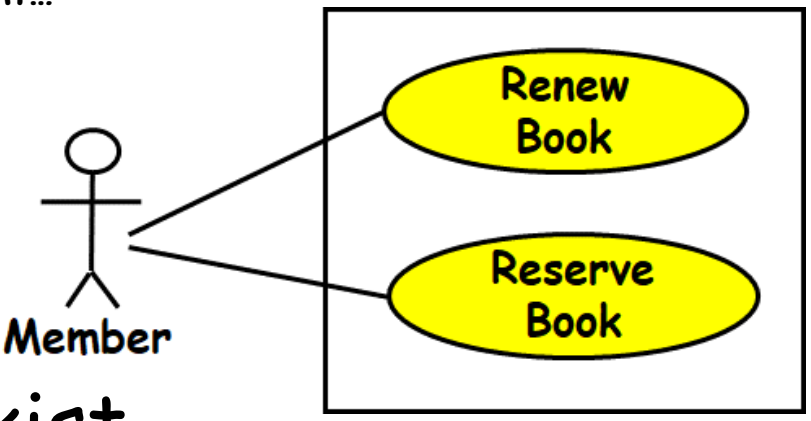
- **A way in which a system can be used by users to achieve some specific goal.** Ex: Withdraw money
- Corresponds to a high-level requirement.
- Defines external behavior without revealing internal structure of system
- **Set of related scenarios tied together by a common goal.**

Example Use Cases

- Library information system:
  - issue-book
  - query-book
  - return-book
  - create-member
  - add-book, etc.

Use Cases Cont…

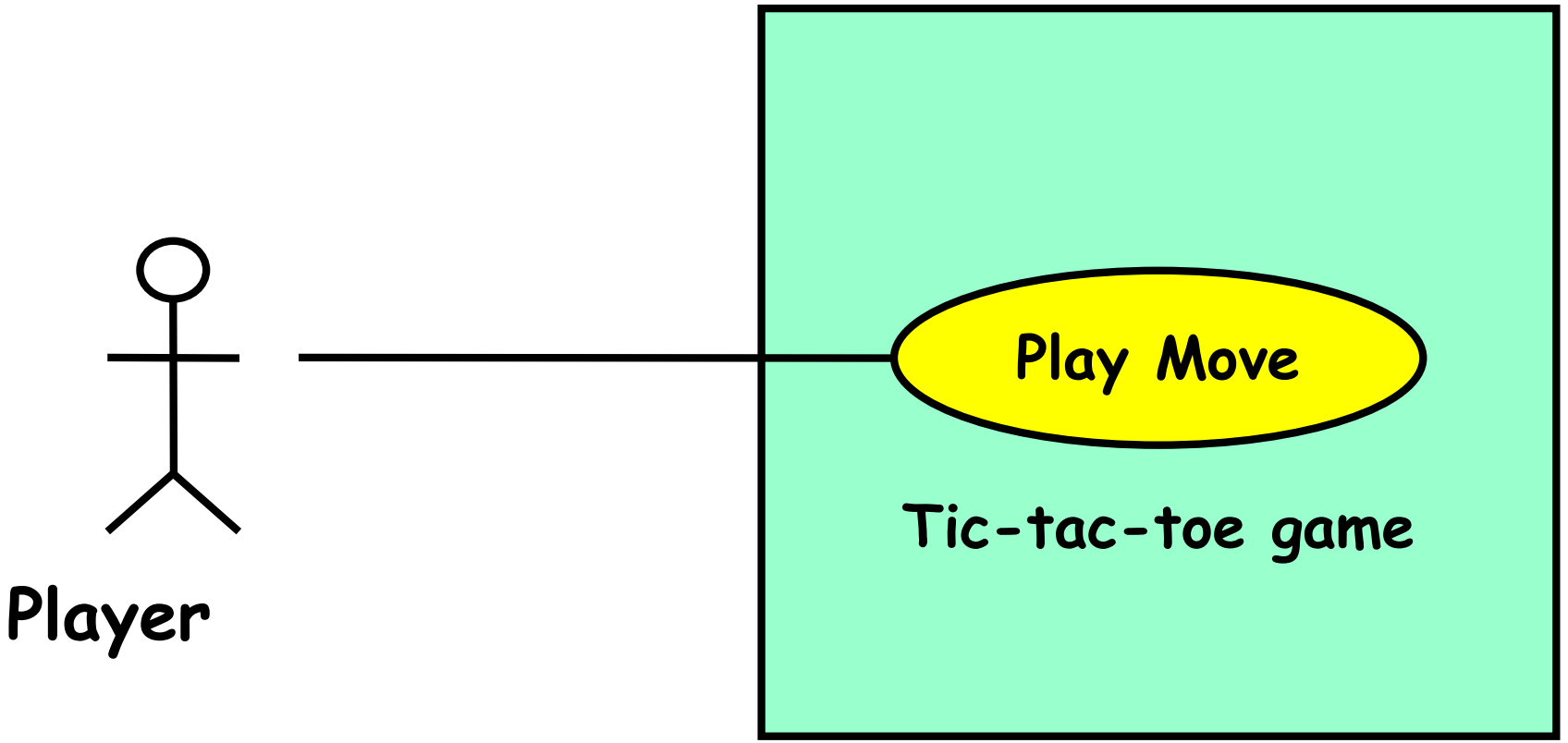
- Normally, use cases are independent of each other



- Implicit dependencies may exist

- **Example**: In a Library Automation System, renew-book and reserve-book are independent use cases.
  - But in actual implementation of renew-book--- *A check is made to see if the book has been reserved using reserve-book.*

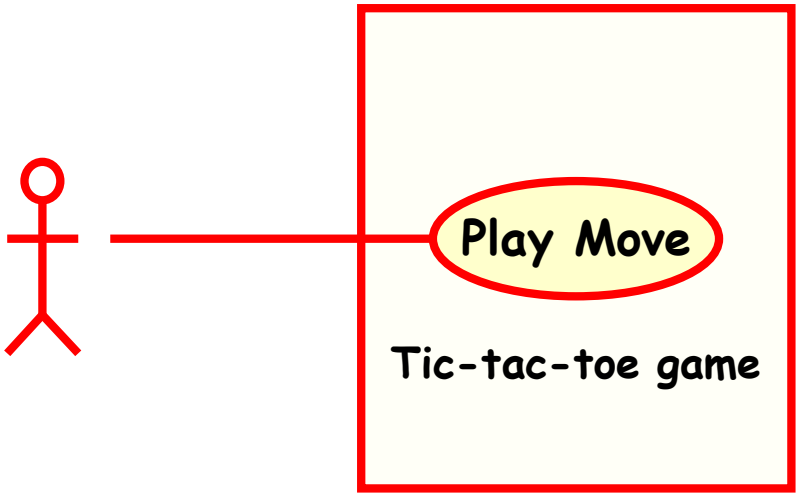
An Example Use Case Diagram



Use case model

Representation of Use Cases

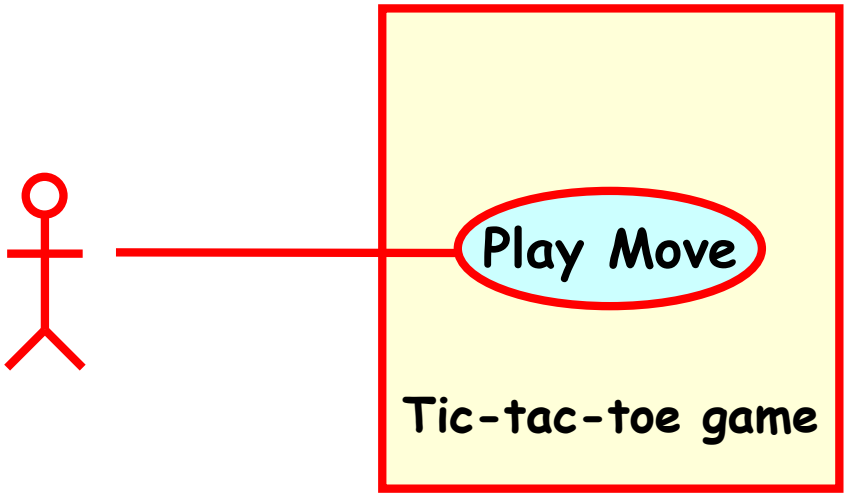
- A use case is represented by an ellipse
- System boundary is represented by a rectangle
- Users are represented by stick person icons (actor)



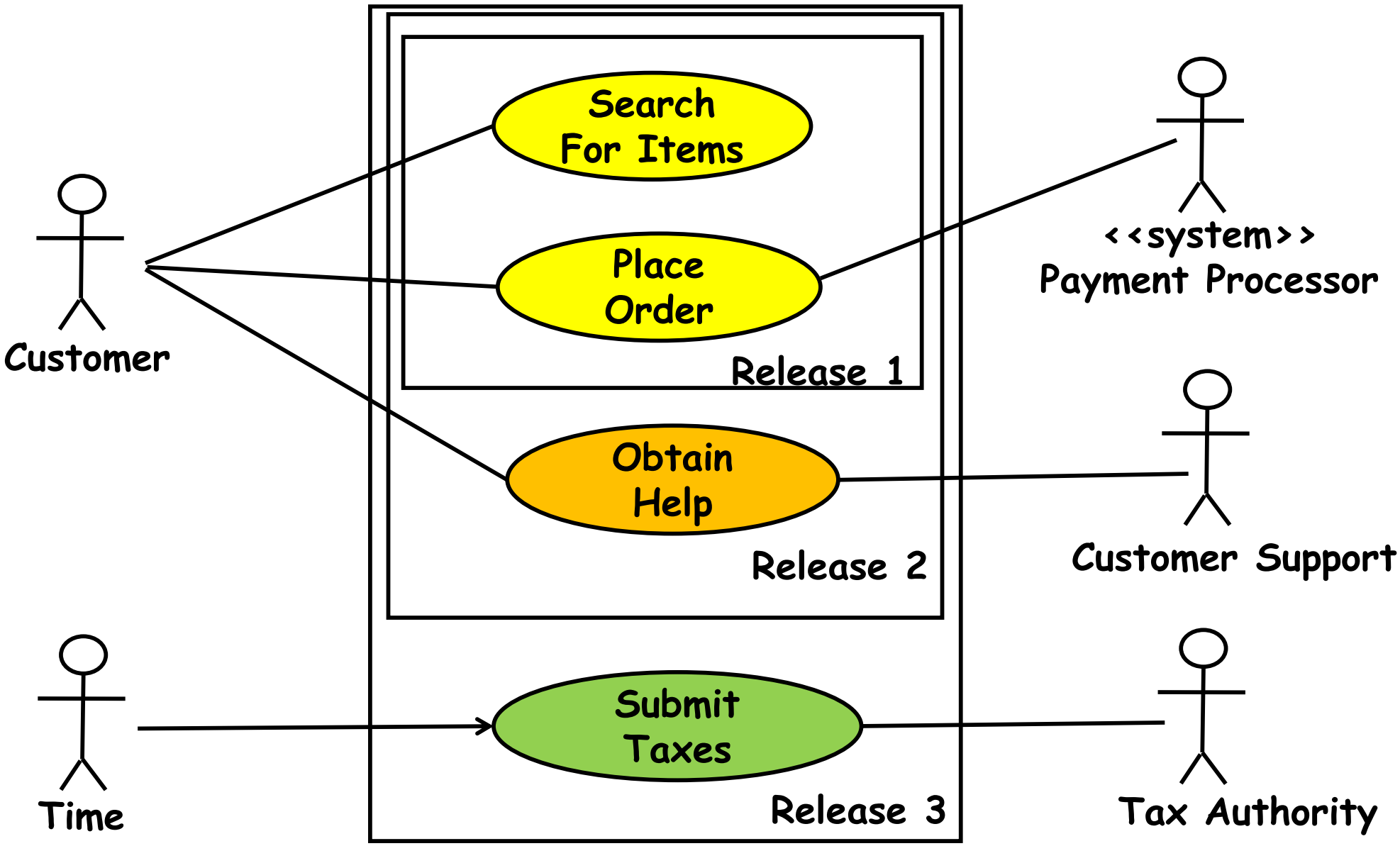
- Communication relationship between actor and use case by a line
- External system by a stereotype

What is a Connection?

- A connection is an association between an actor and a use case.
- **Depicts a usage relationship…**
- Connection does not indicate data flow

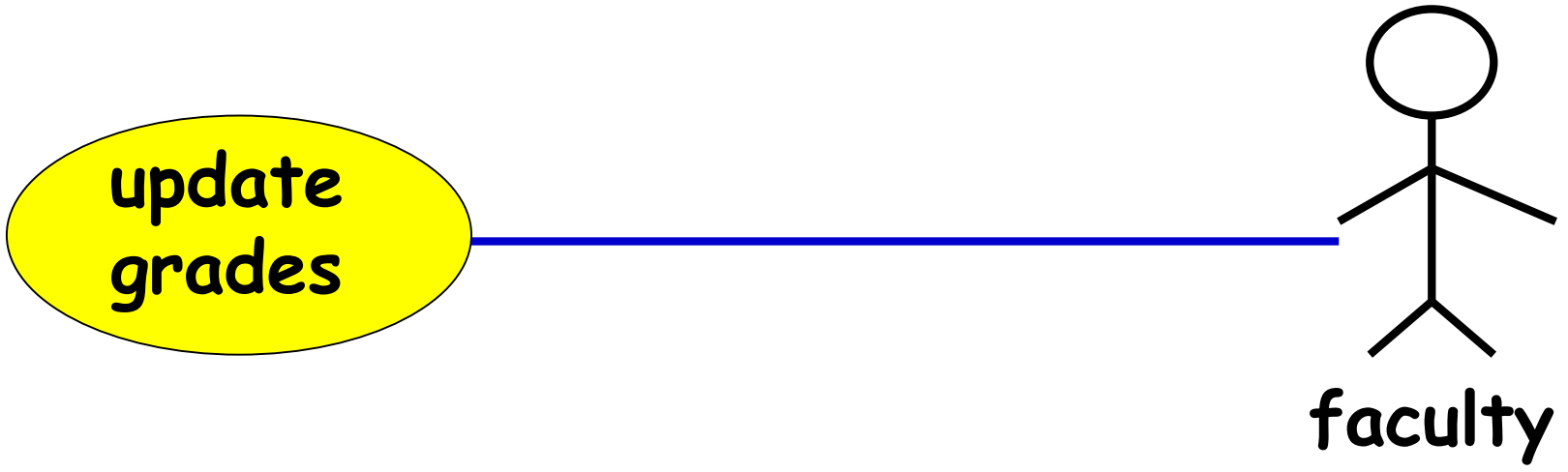


System Boundary Box

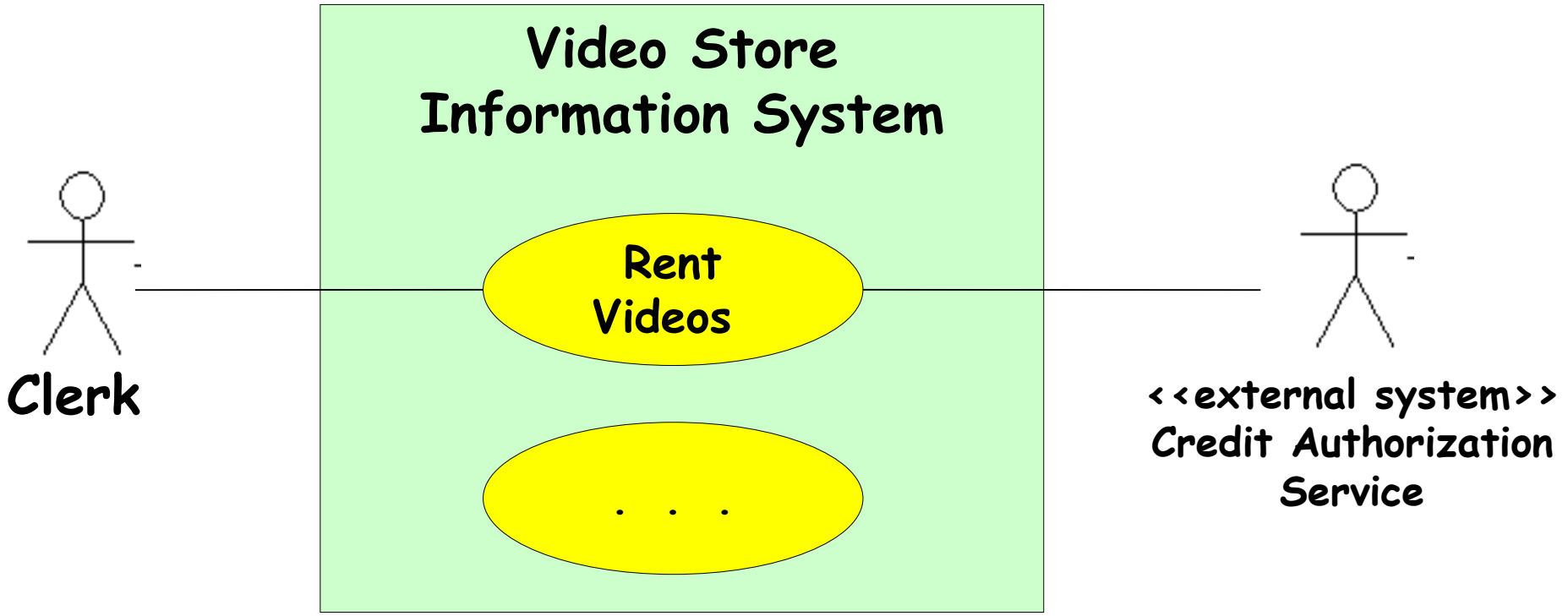


Relationships between Use Cases and Actors

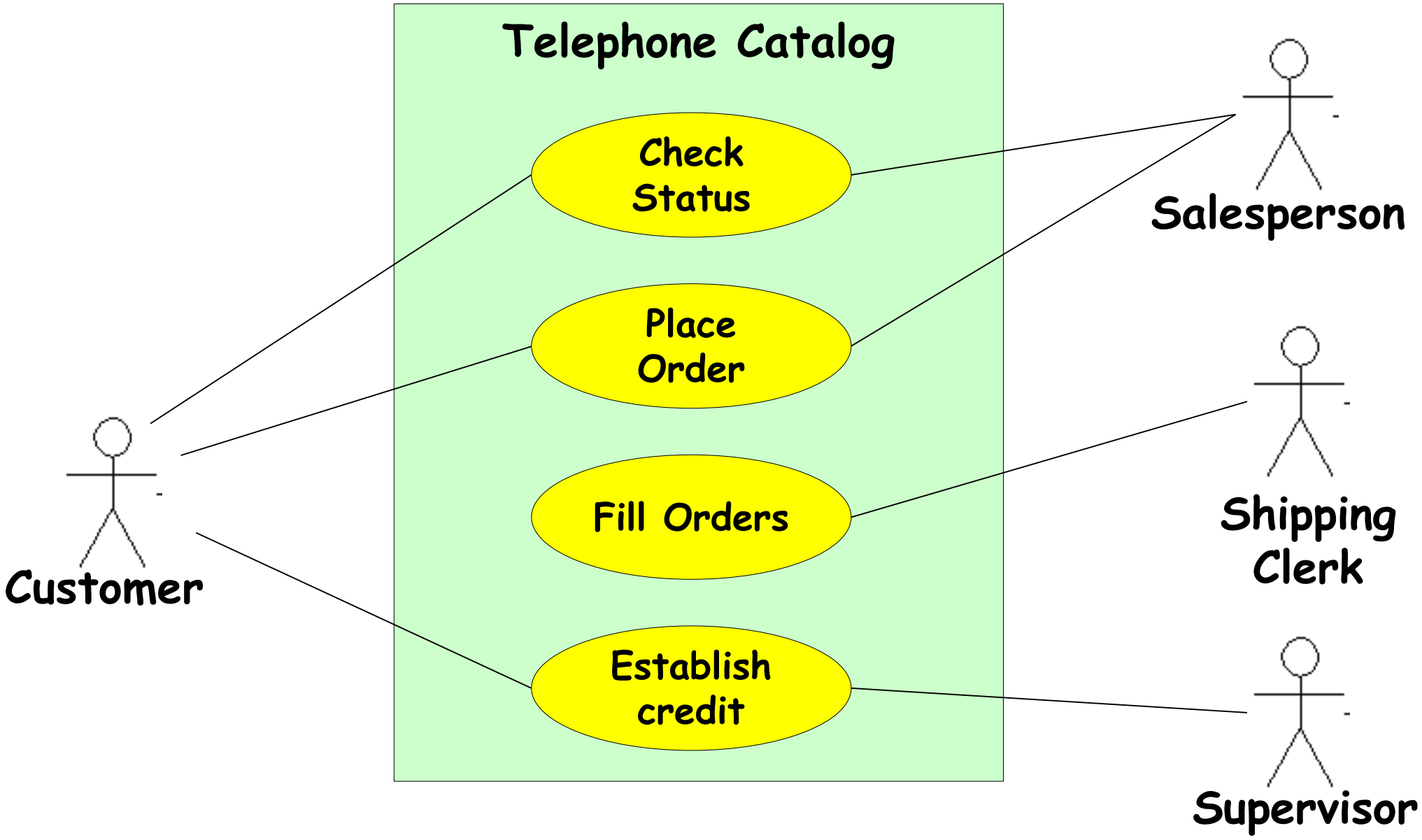
- An actor is connected to a use case by an association relation:
  - Indicates that the actor and the use case communicate with one another.



Another Example Use Case Diagram



Yet Another Use Case Example



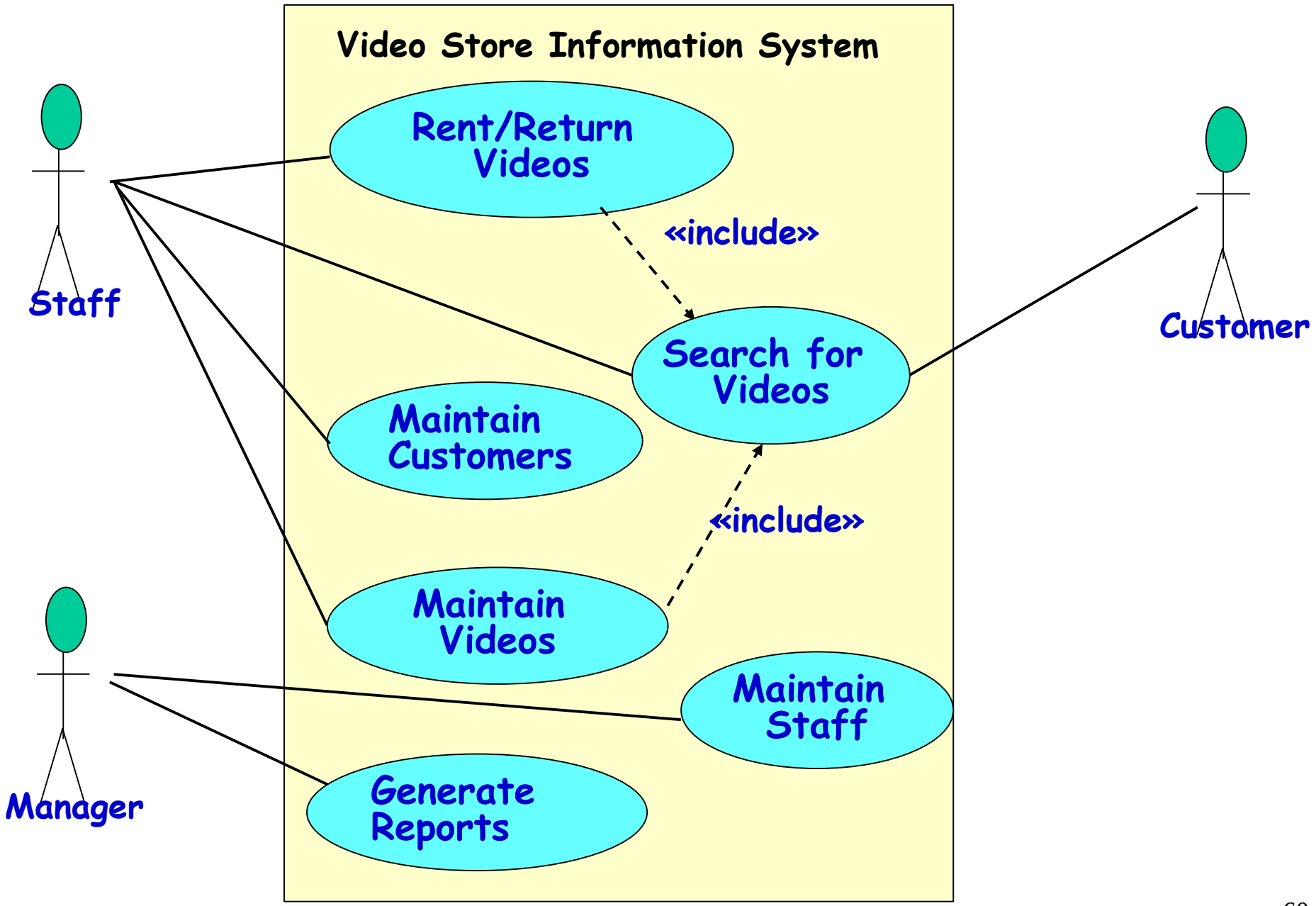
Why Develop A Use Case Diagram?

- Serves as requirements specification
- How is identification of actors useful in software development?
  - Identifies different categories of users:
    - Helps in implementing appropriate interfaces for each category of users.
  - Helps in preparing appropriate documents (e.g. **user's manual**).

Exercise1: Video Store Information System

- Video Store Information System supports the following business functions:
  - Recording information about videos the store owns
    - This database is searchable by staff and all customers
  - Information about a customer's borrowed videos
    - Access by staff and also the customer. It involves video database searching.
  - Staff can record video rentals and returns by customers. It involves video database searching.
  - Staff can maintain customer, video and manager can maintain the staff information.
  - Manager of the store can generate various reports

Exercise 1: Solution



Use Case Description

- Name
- Actors
- Trigger
- Preconditions
- Post conditions
- Success Scenario
- Alternatives flows

Alistair Cockburn "Writing Effective Use Cases"

Triggers

- What starts a use-case?
- **Examples:**
  - Customer initiates a claim report by clicking a button on a web page
  - Customer inserts ATM card
  - System clock is 10:00
  - etc.

Preconditions

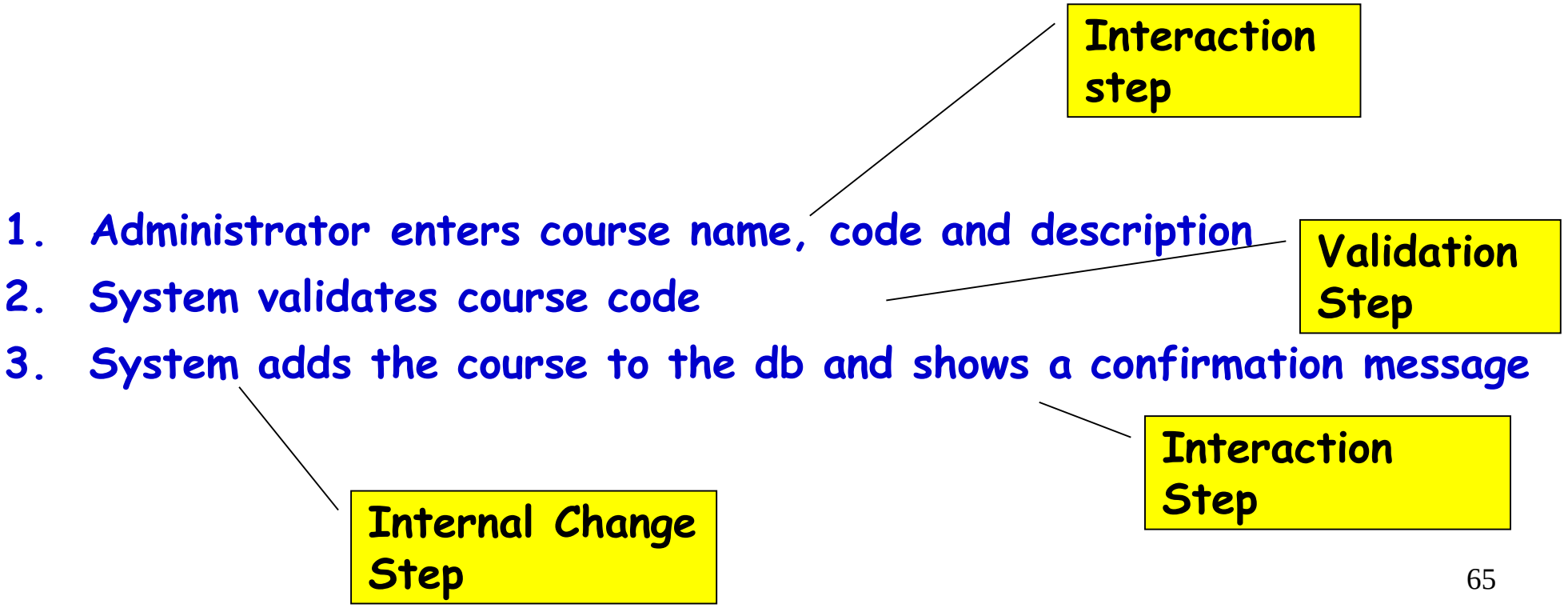
- What the system needs to be true before running the use-case.
- **Examples:**
  - The use has logged on
  - Requested Book is issued out
  - User account has been created
  - The membership has expired

Post-Conditions

- **A post-condition is the outcome of a use-case.**
- **Examples:**
  - Money is transferred to the user account
  - User is logged in
  - The file is saved to the hard-disk
- **Minimal guarantee:**
  - The things that a system can promise, holding even when the use case execution ended in failure
  - **Example:** Money is not transferred unless authorization is granted by the user
- **Success guarantee:**
  - What happens after a successful conclusion of the use-case.
  - **Example:** Transaction logged; Money is transferred

Success Scenario

- **The success scenario is the main story-line of the use-case:**
  - Under the assumption that everything is okay, no errors or problems occur
- Typically, it is composed of a sequence of action steps



Member Services System

Author (s): K. Dittman

Date: 11/01/02
Version: 1.00

Use Case Name:	Place New Order	**Use Case Type** **Business Requirements:** ☐ **System Analysis:** ☑
Use Case ID:	MSS-BUC002.00	
Priority:	High	
Source:	Requirement — MSS-R1.00 Requirements Use Case — MSS-BUC002.00	
Primary Business Actor:	Club Member (Alias — Active Member, Member)	
Primary System Actor:	Club Member (Alias — Active Member, Member)	
Other Participating Actors:	• Warehouse (Alias — Distribution Center) (external receiver) • Accounts Receivable (external server)	
Other Interested Stakeholders:	• Marketing — Interested in sales activity in order to plan new promotions. • Procurement — Interested in sales activity in order to replenish inventory. • Management — Interested in order activity in order to evaluate company performance and customer (member) satisfaction.	
Description:	This use case describes the event of a Club member submitting a new order for SoundStage products via the World Wide Web. The member selects the items he or she wishes to purchase. Once the member has completed shopping, the member's demographic information as well as account standing will be validated. Once the products are verified as being in stock, a packing order is sent to the warehouse for it to prepare the shipment. For any product not in stock, a back order is created. On completion, the member will be sent an order confirmation.	
Precondition:	The individual submitting the order must be an active club member. The member must log in to the system (provide identification) to enter an order.	
Trigger:	This use case is initiated when the member selects the option to enter a new order.	

Typical Course of Events:	**Actor Action**	**System Response**
	Step 1: The member requests the option to enter a new order.	**Step 2:** The system responds by displaying the catalog of the SoundStage products.
	Step 3: The member browses the available items and selects the ones he or she wishes to purchase, along with the quantity.	**Step 4:** Once the member has completed the selections, the system retrieves from file and presents the member's demographic information (shipping and billing addresses).
	Step 5: The member verifies demographic information (shipping and billing addresses). If no changes are necessary, the member responds accordingly (to continue).	**Step 6:** For each product ordered, the system verifies the product availability and determines an expected ship date, determines the price to be charged to the member, and determines the cost of the total order. If an item is not immediately available, it indicates the product is back ordered or that it has not been released for shipping (for preorders). If an item is no longer available, that is indicated also. The system then displays a summary of the order to the member for verification.
	Step 7: The member verifies the order. If no changes are necessary, the member responds accordingly (to continue).	**Step 8:** The system checks the status of the member's account. If satisfactory, the system prompts the member to select the desired payment option (to be billed later or pay immediately with a credit card).
	Step 9: The member responds by selecting the desired payment option.	**Step 10:** The system displays a summary of the order, including the desired payment option, to the member for verification.
	Step 11: The member verifies the order. If no changes are necessary, the member responds accordingly (to continue).	**Step 12:** The system records the order information (including back orders if necessary). **Step 13:** Invoke abstract use case *MSS-AUC001.00, Determine Appropriate Distribution Center and Release Order to Be Filled.* **Step 14:** Once the order is processed, the system generates an order confirmation and displays it to the member as well as sending it to the member via e-mail.

Alternate Courses:	**Alt-Step 3:** The member enters search criteria to retrieve a specific item or to display a reduced list of items to browse and order from. **Alt-Step 5:** If changes are required, the member updates the appropriate shipping, billing, or e-mail addresses and tells the system to store them accordingly. The system will validate the changes and, if successful, will store the new information to file. **Alt-Step 7:** If the order requires changes, the member can delete any item no longer wanted or change the order quantity. Once the member has completed the order changes, the system reprocesses the order (**go to step 6**). If the member requests to do additional shopping, **go to step 3**. If the member needs to change the demographic information, **go to step 5**. **Alt-Step 8:** If the member's account is not in good standing, display to the member the account status, the reason the order is being held, and what actions are necessary to resolve the problem. In addition, an e-mail is sent to the member with the same information. The system prompts the member to hold the order for later processing or cancel the order. If the member wishes to hold the order, the system records the order information and places it in hold status and then displays the SoundStage main page. If the member chooses to cancel the order, the system clears the inputted information and then displays the SoundStage main page. Terminate the use case. **Alt-Step 10:** If the member selects the option to pay by credit card, the system prompts the member to enter the credit card information (number and expiration date) and reminds the member that the billing address on file must match the billing address of the credit card provided. The member enters the required information and requests that the system continue. The system validates the credit card account provided. If the account cannot be validated, the system notifies the member and requests an alternative means of payment. If the member cannot provide an alternative means at this time, he or she can choose either to hold or to cancel the order. If the member wishes to hold the order, the system records the order information and places it in hold status and then displays the SoundStage main page. If the member chooses to cancel the order, the system clears the inputted information and then displays the SoundStage main page. Terminate the use case. **Alt-Step 11:** If the order requires changes, the member can delete any item no longer wanted or change the order quantity. Once the member has completed the order changes, the system reprocesses the order (**go to step 6**). If the member requests to do additional shopping, **go to step 3**. If the member needs to change the demographic information, **go to step 5**. **Alt-Step 12:** If all items ordered are on back order, the order is not released to the distribution center.
Conclusion:	This use case concludes when the member receives a confirmation of the order.
Postcondition:	The order has been recorded, and if the ordered products were available, released to the distribution center. For any product not available a back order has been created.
Business Rules:	• Member must have a valid e-mail address to submit online orders. • Member is billed for products only when they are shipped.
Implementation Constraints and Specifications:	• Use case must be available to the member 24 × 7. • Frequency — It is estimated that this use case will be executed 3,500 times per day. It should support up to 50 concurrent members.
Assumptions:	• Product can be transferred among distribution centers to fill orders. • Procurement will be notified of back orders by a daily report (separate use case). • The member responding to a promotion or using credits may affect the price of each ordered item. • The member can cancel the order at any time.
Open Issues:	None

Member Services System

Author (s): K. Dittman **Date:** 11/01/02

Version: 1.00

Use Case Name:	Determine Appropriate Distribution Center and Release Order to Be Filled.	**Use Case Type** **Business Requirements:** ☐ **System Analysis:** ☑
Use Case ID:	MSS-AUC001.00	
Priority:	High	
Source:	MSS-SUC002.00 MSS-SUC003.00	
Participating Actors:	• Warehouse (Alias — Distribution Center) (external receiver)	
Description:	This use case describes the event of selecting the distribution center that services the shipping address provided by the club member for a particular order. The order information (packing order) is then sent (released) to that distribution center to be filled.	
Precondition:	The order is ready to be released to the appropriate distribution center.	
Typical Course of Events:	**Step 1:** The system selects the appropriate distribution center based on the state and zip code of the shipping address. **Step 2:** Once the distribution center has been selected, a packing order containing the items to ship is formatted. **Step 3:** The packing order is transmitted to the distribution center (shipping and receiving system) to be used to prepare the shipment.	
Alternate Courses:	**Alt-Step 1:** If the shipping address is an international address, route the packing order to the Indianapolis, IN, location.	
Postcondition:	The packing slip has been transmitted (released) to the appropriate distribution center.	

Use Case Description: Issue Video

- **Actors:** Staff
- **Preconditions:**
  - Staff has logged on to the system and selected 'Issue Video' option
- **Basic course**
  1. Staff scans the video bar code and the member's card
  2. System retrieves member's account and video details
  3. Staff presses OK
  4. System generates transaction summary
- **Alternative courses**

  3. If membership dues pending then …

Use Case Description: Change Flight

- **Actors:** traveler
- **Preconditions:**
  - Traveler has logged on to the system and selected 'change flight itinerary' option
- **Basic course**
  1. System retrieves traveler's account and flight itinerary from client account database
  2. System asks traveler to select itinerary segment she wants to change; traveler selects itinerary segment.
  3. System asks traveler for new departure and destination information; traveler provides information.
  4. If flights are available then
  5. ...
  6. System displays transaction summary.
- **Alternative courses**
  4. If no flights are available then ...

ATM Money Withdraw Example

- **Actors:** Customer
- **Pre Condition:**
  - ATM must be in ready state
- **Post Condition:**
  - The current amount of cash in the user account is the amount before withdraw minus withdraw amount
  - A receipt is printed for the withdrawn amount

ATM Money Withdraw - Mainline Scenario

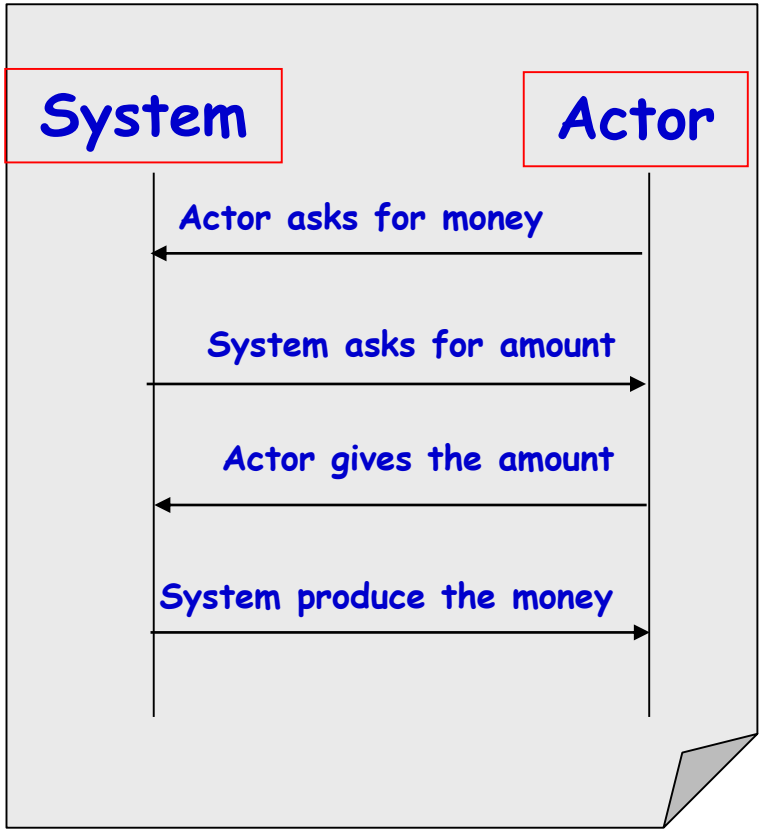
Actor Actions	System Actions
1. Begins when a Customer arrives at ATM	
2. Customer inserts a Credit card into ATM	3. System verifies the customer ID and status
5. Customer chooses "Withdraw" operation	4. System asks to enter operation type
7. Customer enters the cash amount	6. System asks for the amount to be withdrawn
	8. System checks if withdraw amount is legal
	9. System dispenses the cash
	10. System deduces the withdraw amount from account
	11. System prints a receipt
13. Customer takes the cash and the receipt	12. System ejects the cash card

ATM Money Withdraw (cont.)

- **Alternative flow of events:**
  - **Step 3:** Customer authorization failed. Display an error message, cancel the transaction and eject the card.
  - **Step 8:** Customer has insufficient fun in his account. Display an error message, and go to step 6.
  - **Step 8:** Customer exceeds his legal amount. Display an error message, and go to step 6.
- **Exceptional flow of events:**
  - Power failure in the process of the transaction before step 9, cancel the transaction and eject the card.

Guidelines for Effective Use Case Writing

- Use simple sentence…
- Do not have both system and actor doing something in a single step…
  - Bad: "Get the amount from the user and give him the receipt."
- Any step should lead to some progress:
  - Bad: "User clicks a key"



Identification of Use Cases

1. Actor-based:

- Identify the actors related to a system or organization.
- For each actor, identify the processes they initiate or participate in.

2. Event-based

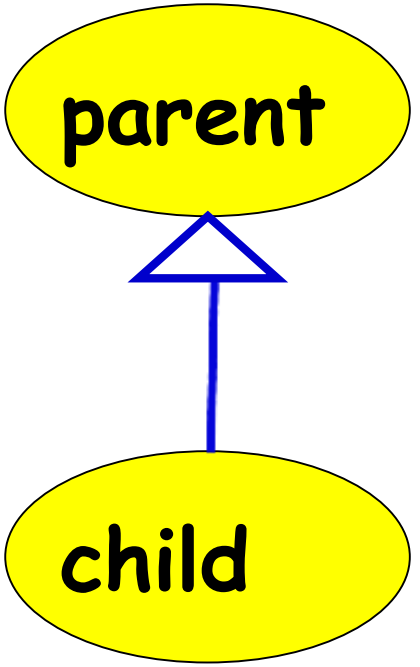
- Identify the external events that the system must respond to.
- Relate the events to actors and use cases.

Factoring Use Cases

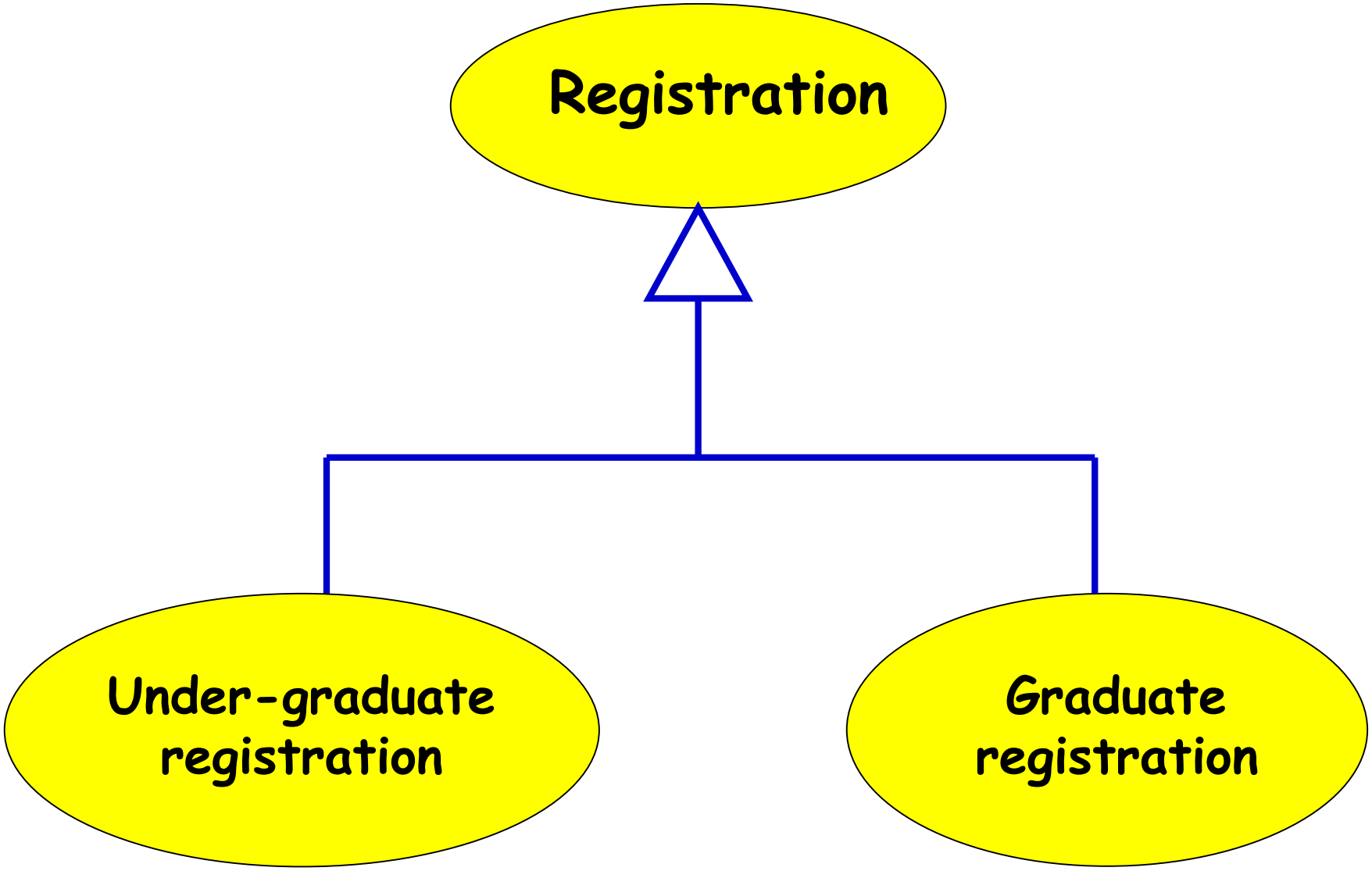
- Two main reasons for factoring:
  - Complex use cases need to be factored into simpler use cases for better design
  - Helps represent common behavior across different use cases
- Three ways of factoring:
  - Generalization
  - Include
  - Extend

Generalization

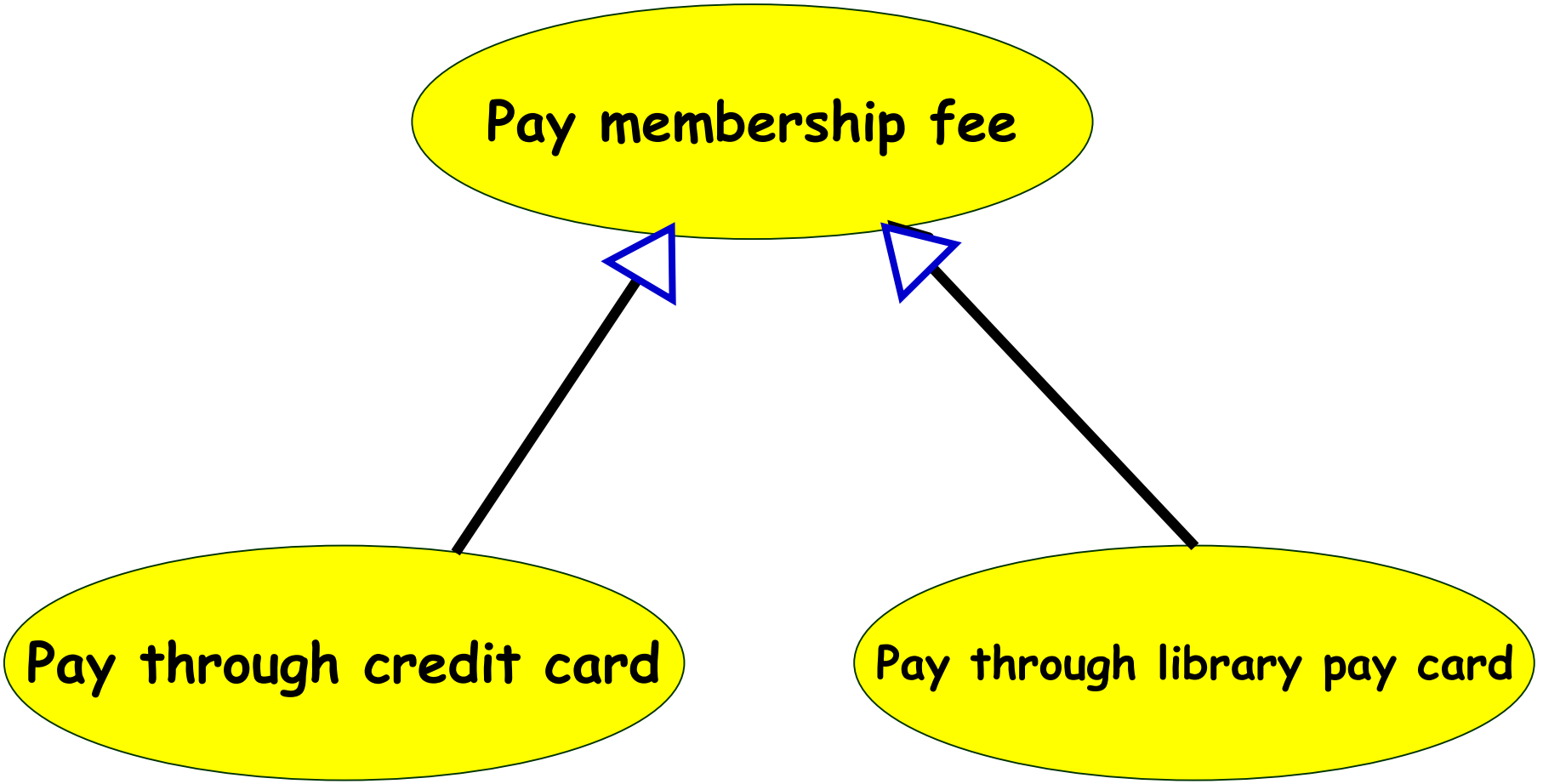
- The child use case inherits the behavior of the parent use case.
  - The child may add to or override some of the behavior of its parent.



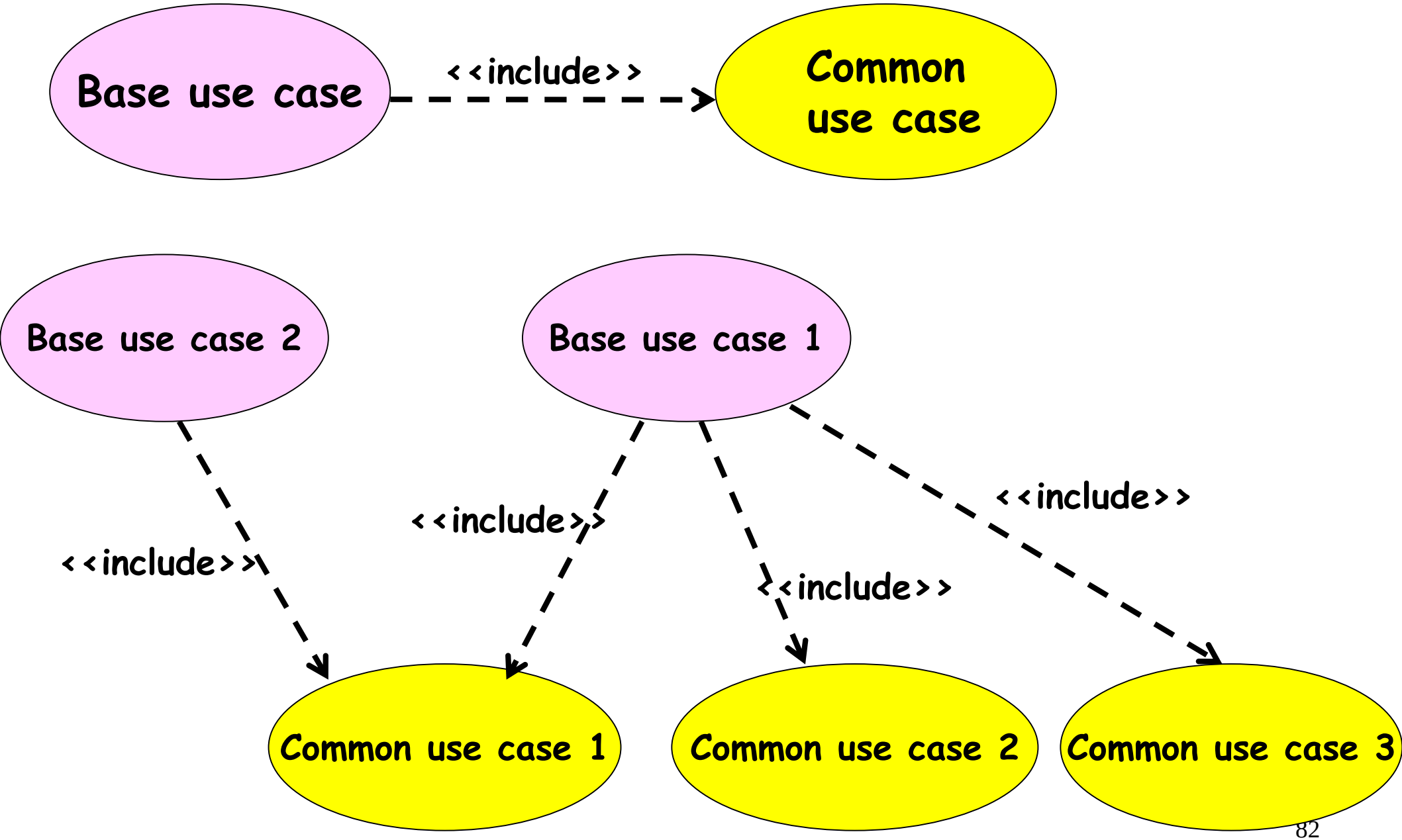
Generalization Example 1



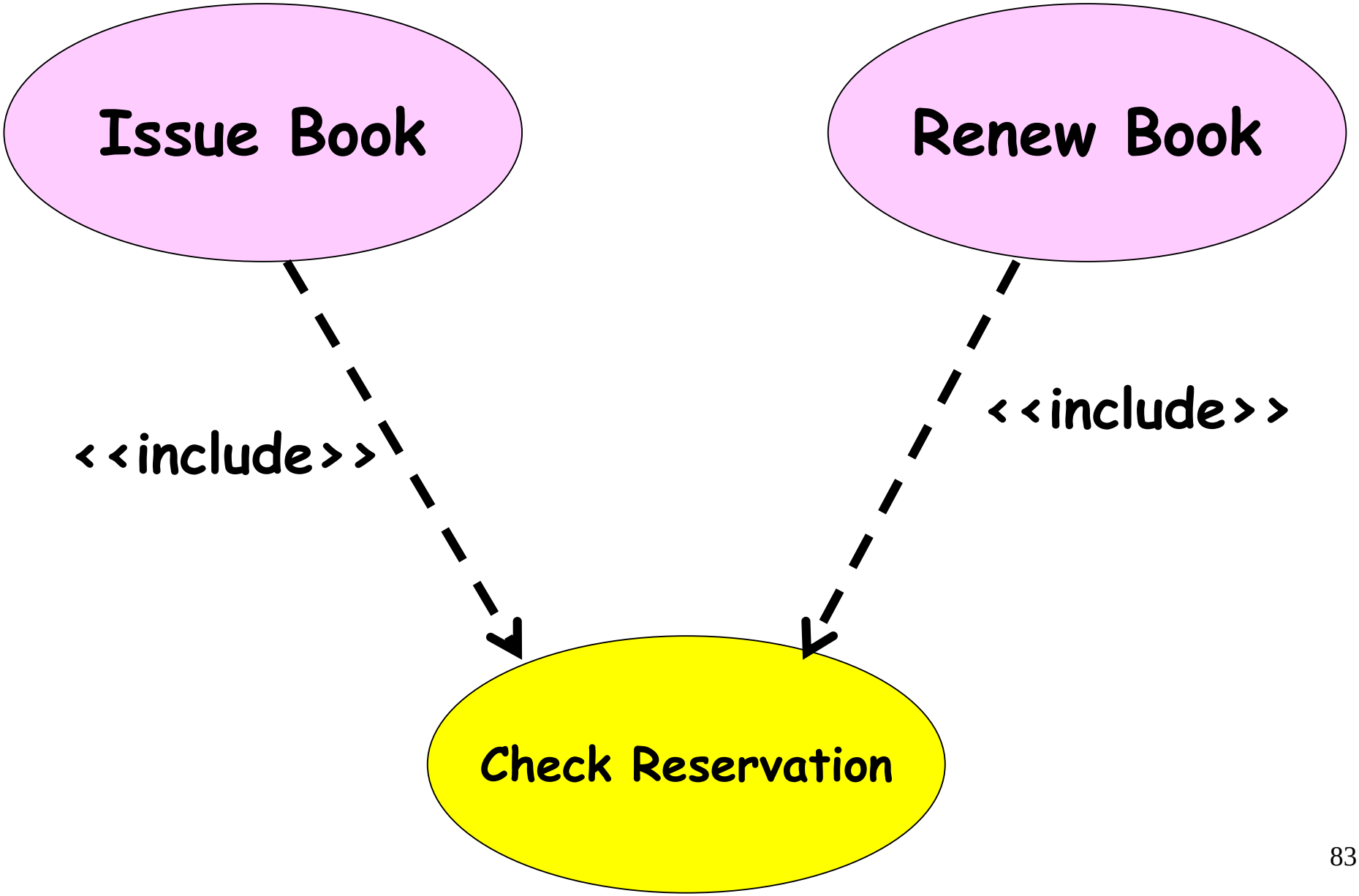
Factoring Use Cases Using Generalization



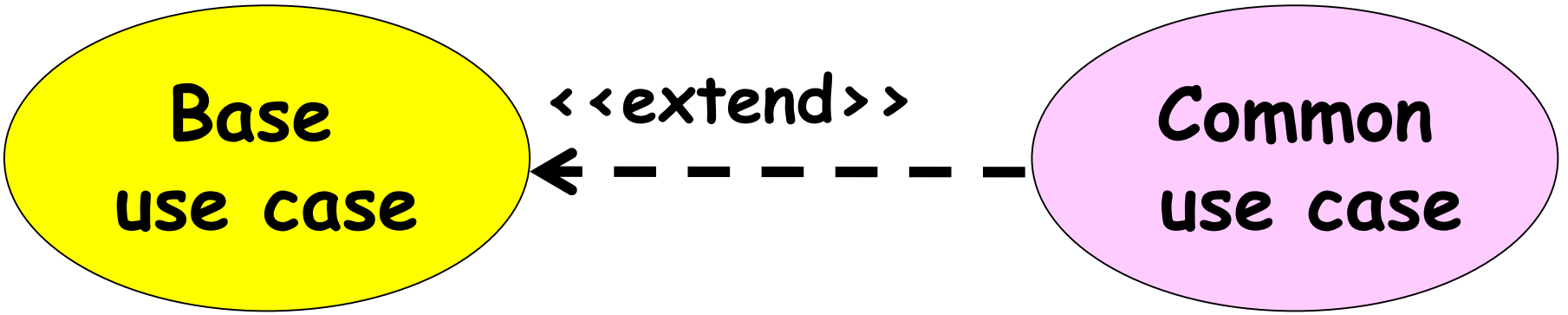
Factoring Use Cases Using Include



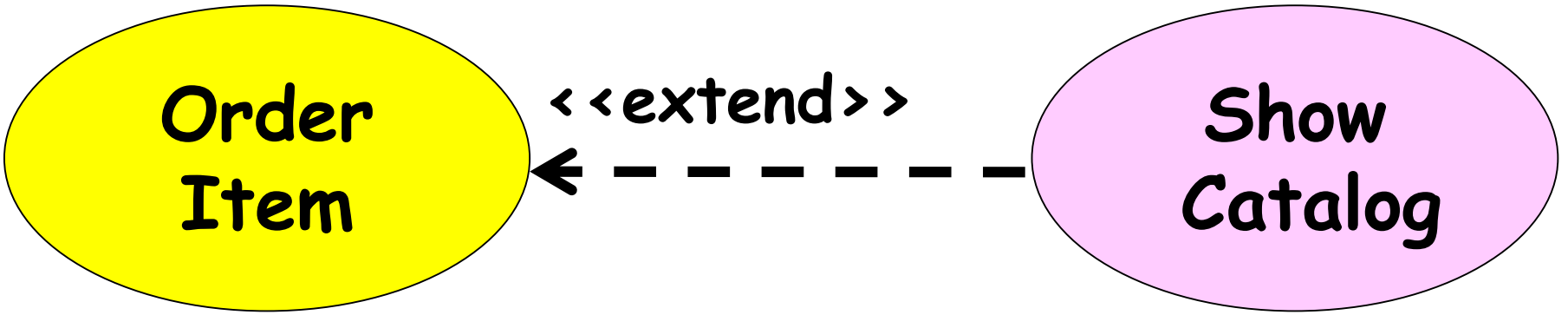
Example of Factoring Use Cases Using Include



Factoring Use Cases Using Extend

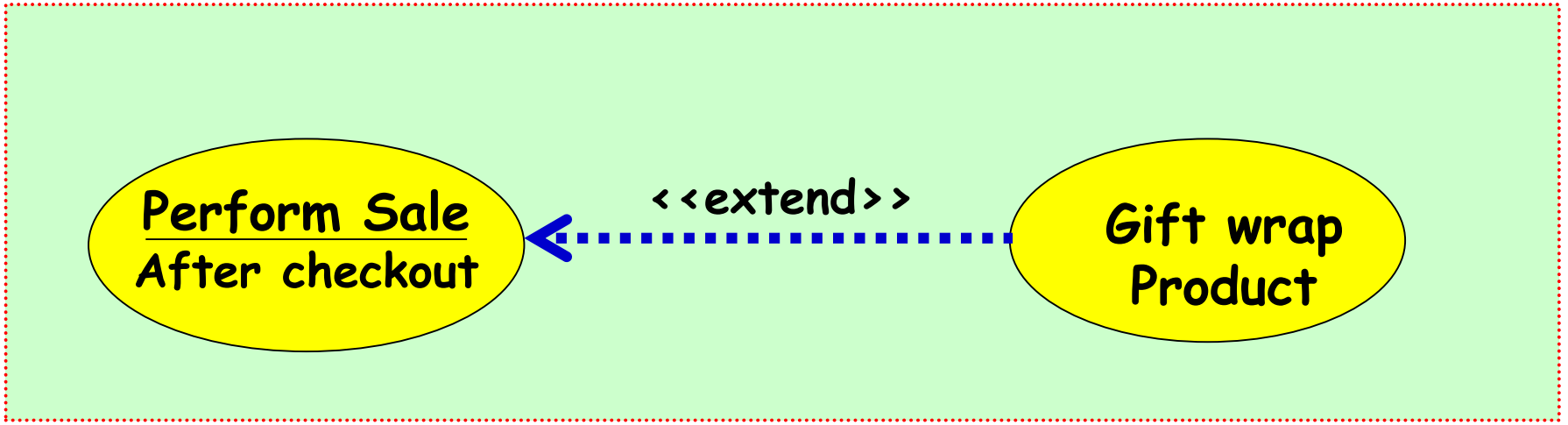


Factoring Use Cases Using Extend

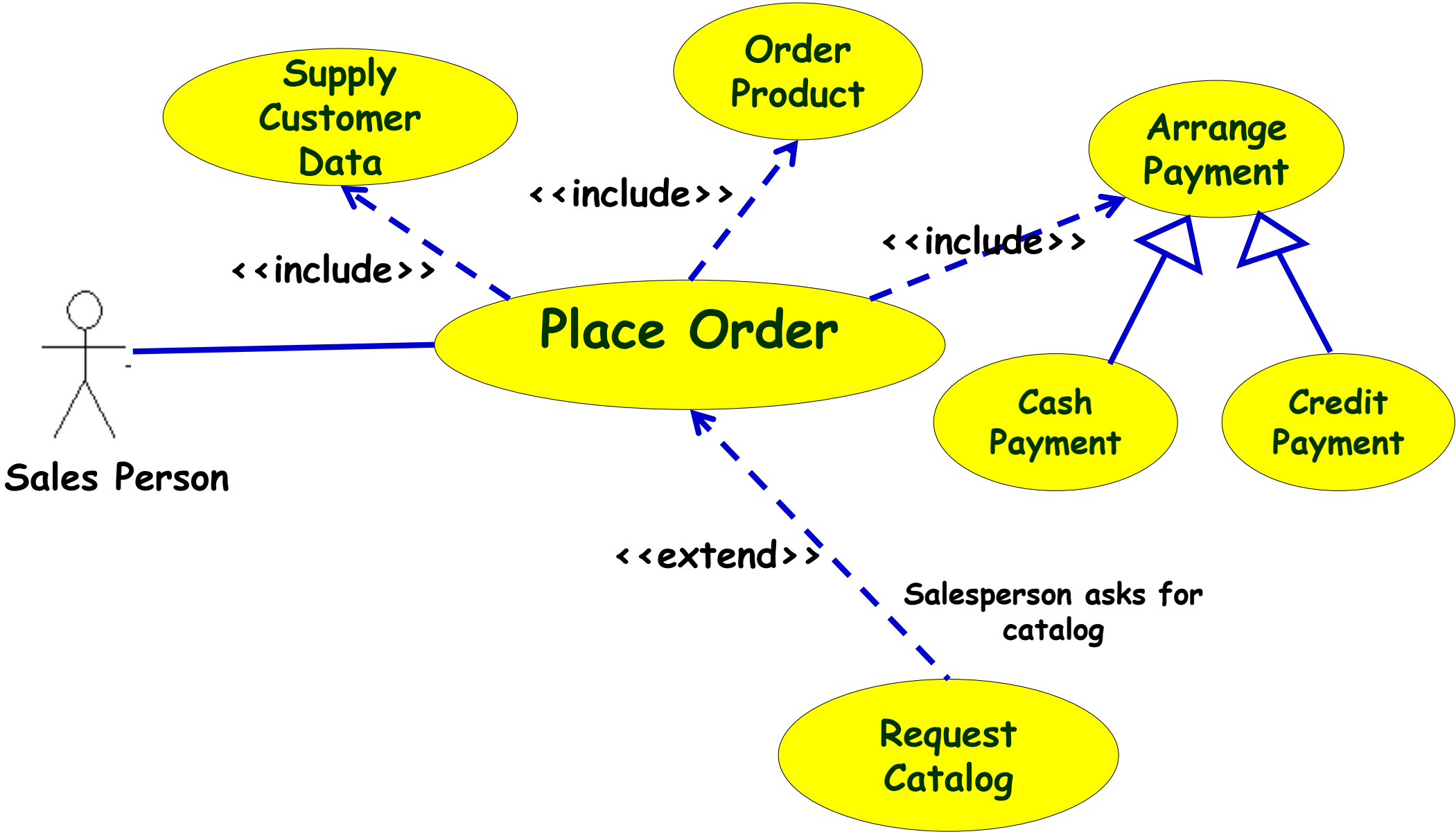


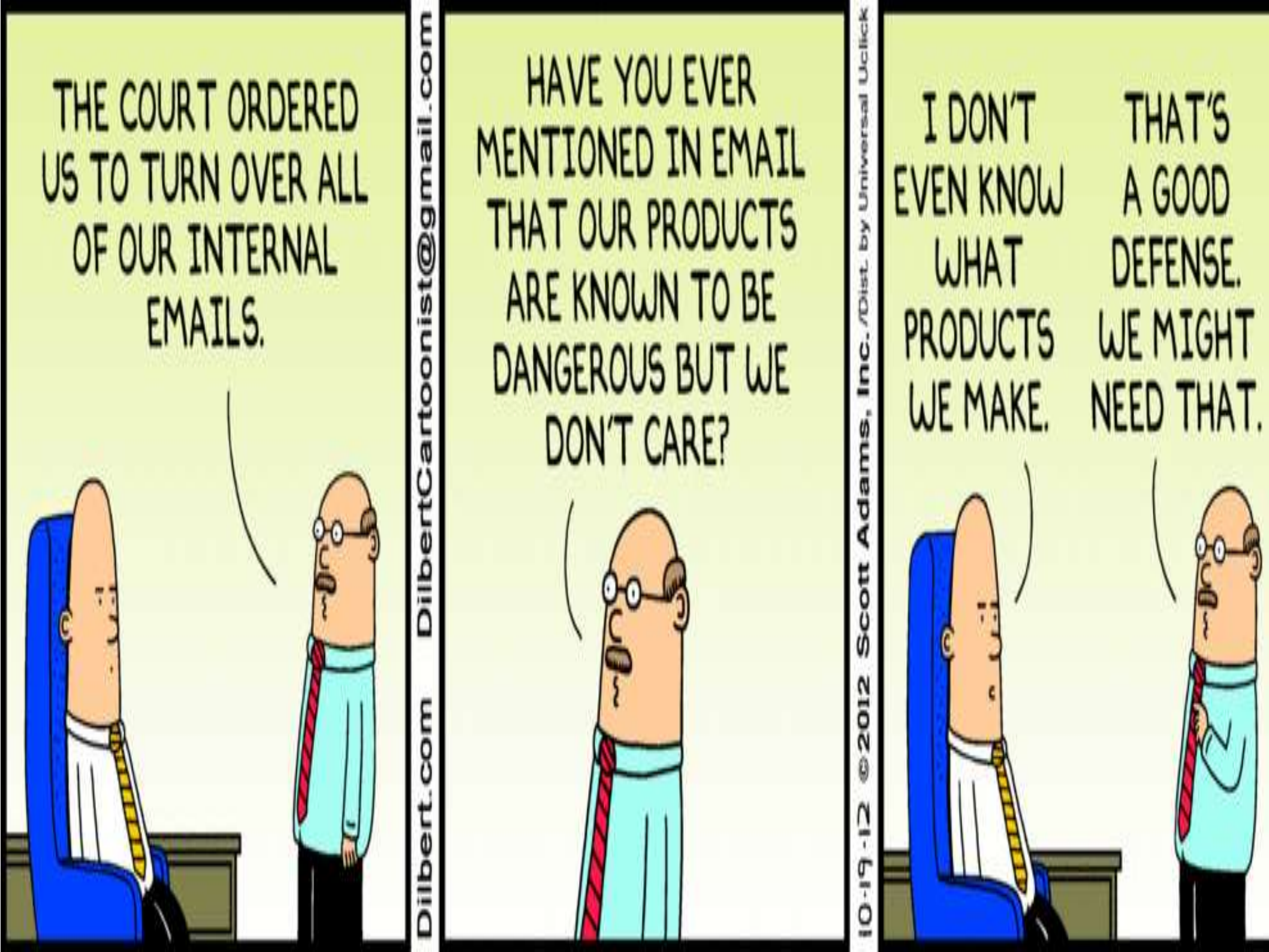
Extension Point

- The base use case may include/extend other use cases:
  - **At certain points, called extension points.**
- Note the direction of the arrow



Use Case Relationships

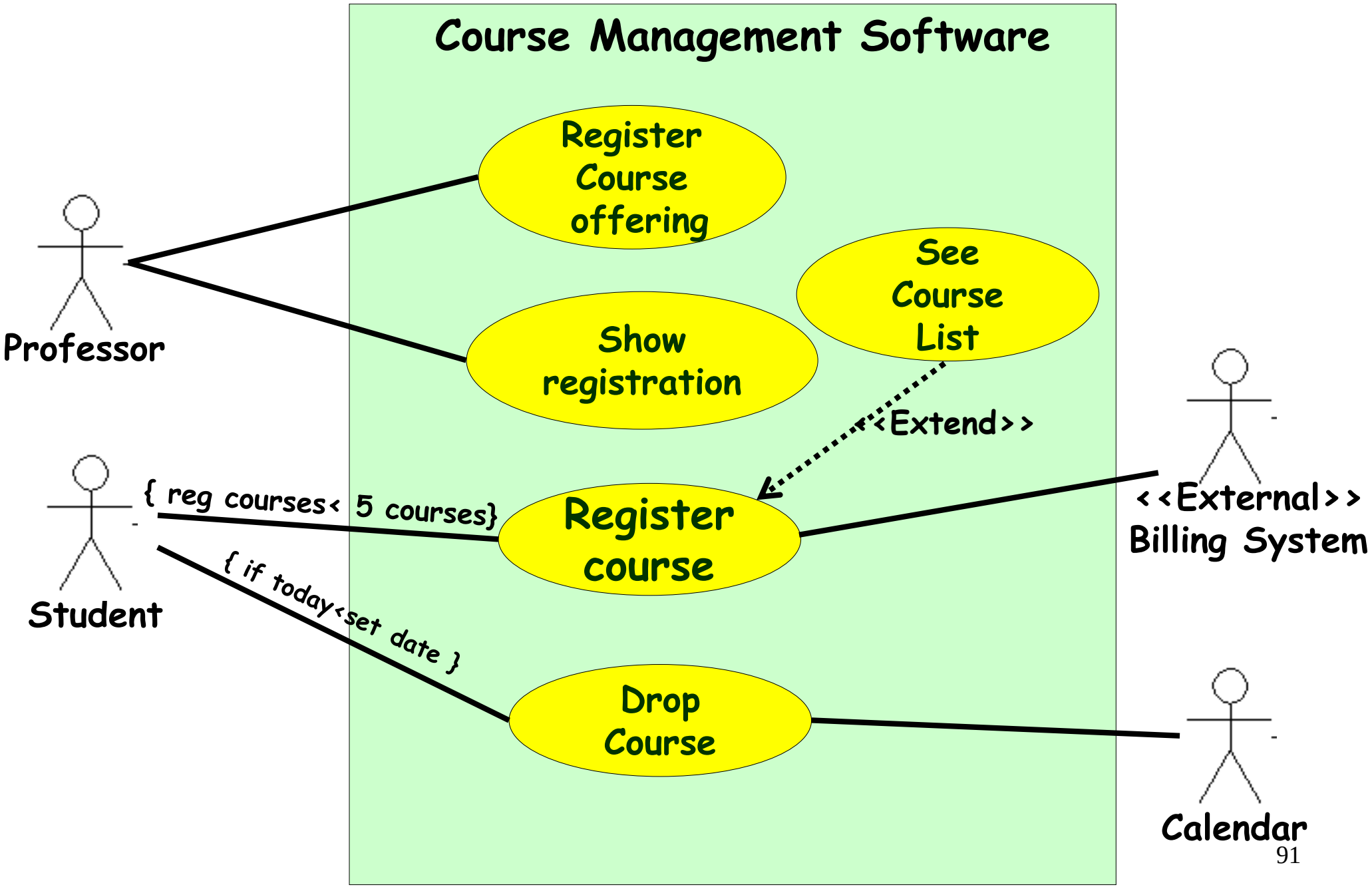



THE COURT ORDERED US TO TURN OVER ALL OF OUR INTERNAL EMAILS.
HAVE YOU EVER MENTIONED IN EMAIL THAT OUR PRODUCTS ARE KNOWN TO BE DANGEROUS BUT WE DON'T CARE?
I DON'T EVEN KNOW WHAT PRODUCTS WE MAKE.
THAT'S A GOOD DEFENSE. WE MIGHT NEED THAT.
Dilbert.com DilbertCartoonist@gmail.com
10-19-12 ©2012 Scott Adams, Inc./Dist. by Universal Uclick

Exercise 2: Use Case Model for Course Management Software

- At the beginning of each semester,
  - Each professor shall register the courses that he is going to teach.
- A student can select up to four-course offerings.
  - During registration a students can at any time request a course catalogue showing course offerings for the semester.
  - Information about each course such as professor, department and prerequisites would be displayed.
  - The registration system sends information to the billing system so the students can be billed for the semester.
- For each semester, there is a period of time during which dropping of courses is permitted.
- Professors must be able to access the system to see which students signed up for each of their course offerings.

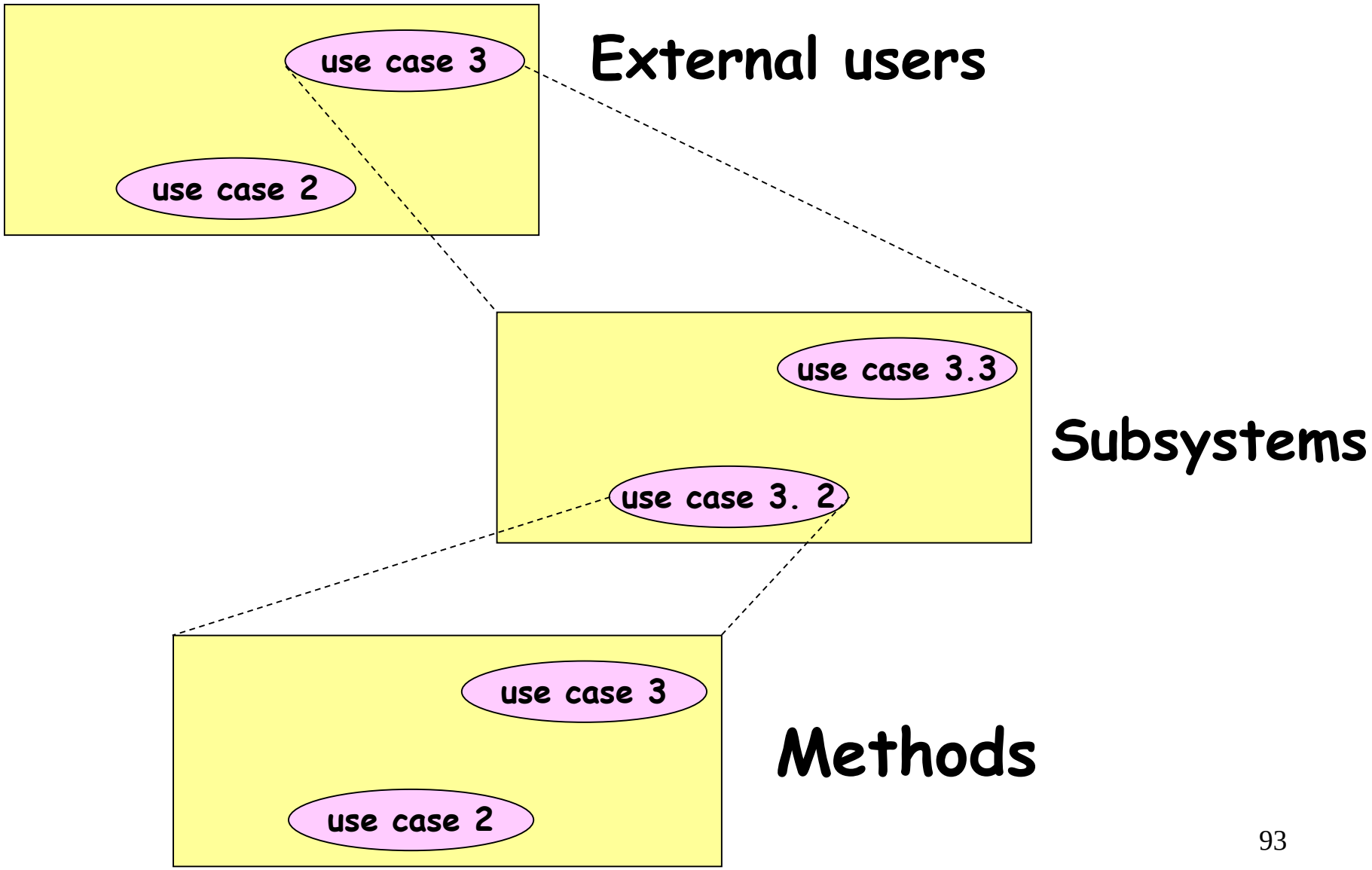
Exercise 2: Model Solution



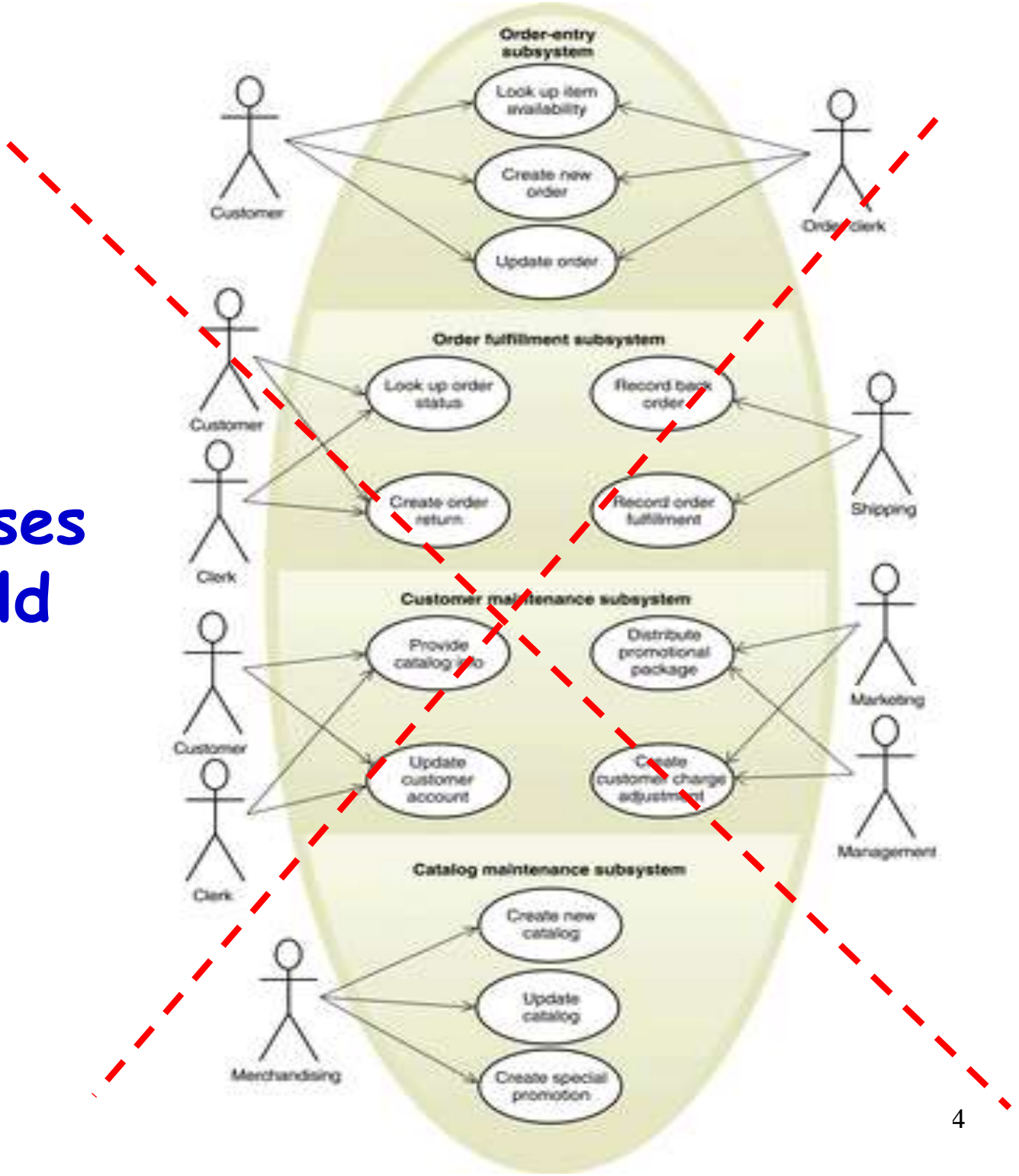
Effective Use Case Modelling

- Use cases should be named and organized from the perspective of the actor.
- Use cases should start off simple and at as much high view as possible.
  - Can be refined and detailed further.
- Use case diagrams represent functionality:
  - Should focus on the "what" and not the "how".

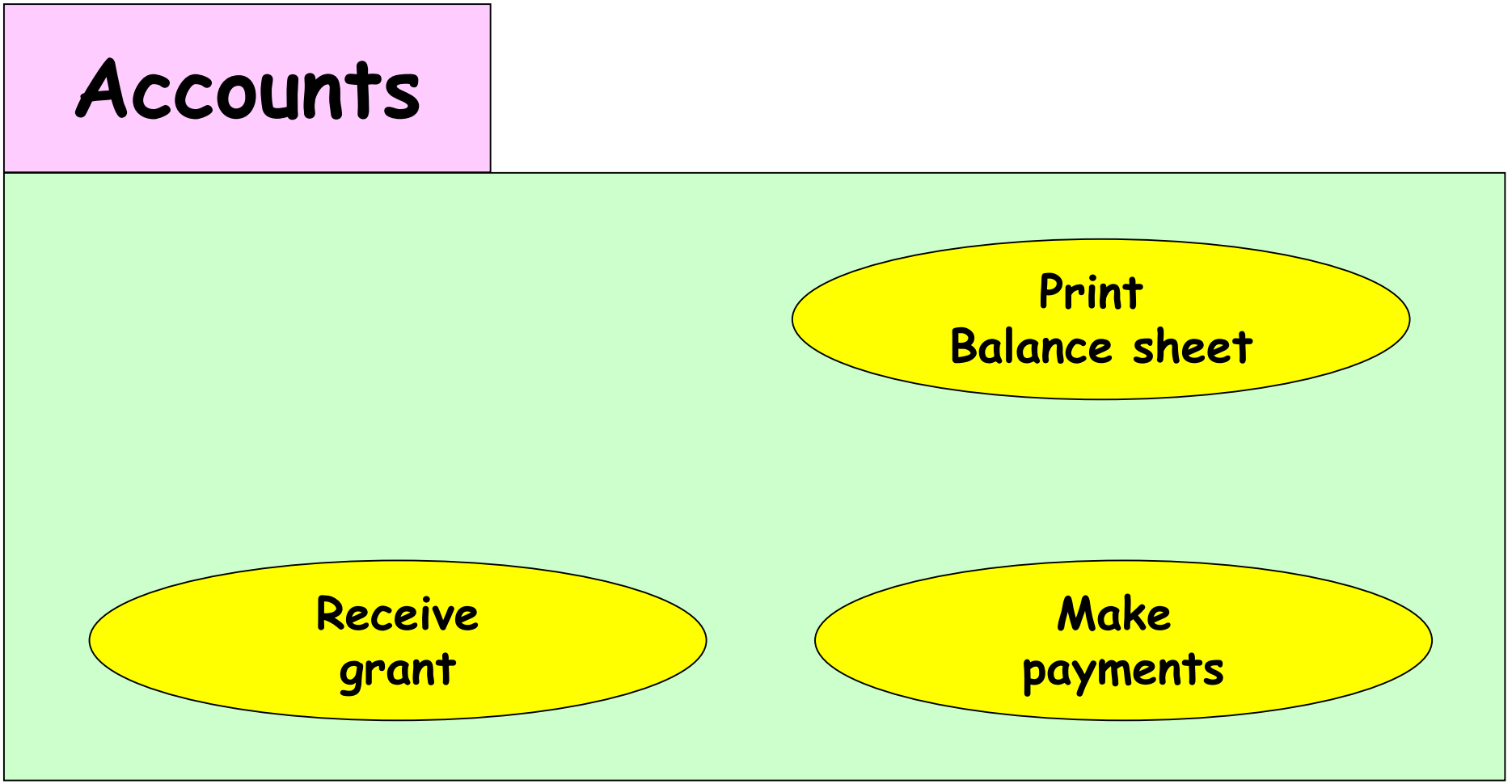
Hierarchical Organization of Use Cases



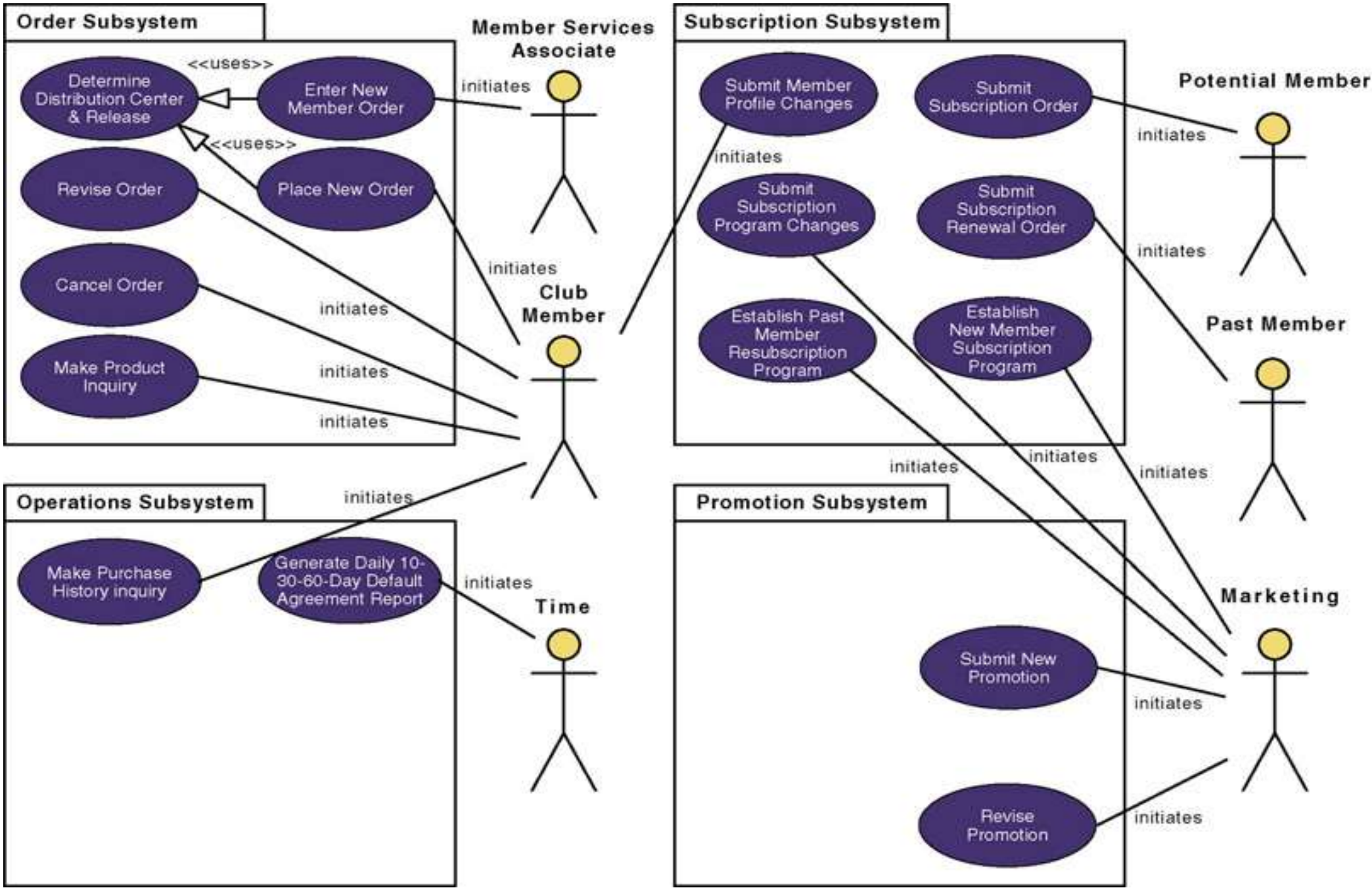
Too many use cases at any level should be avoided!



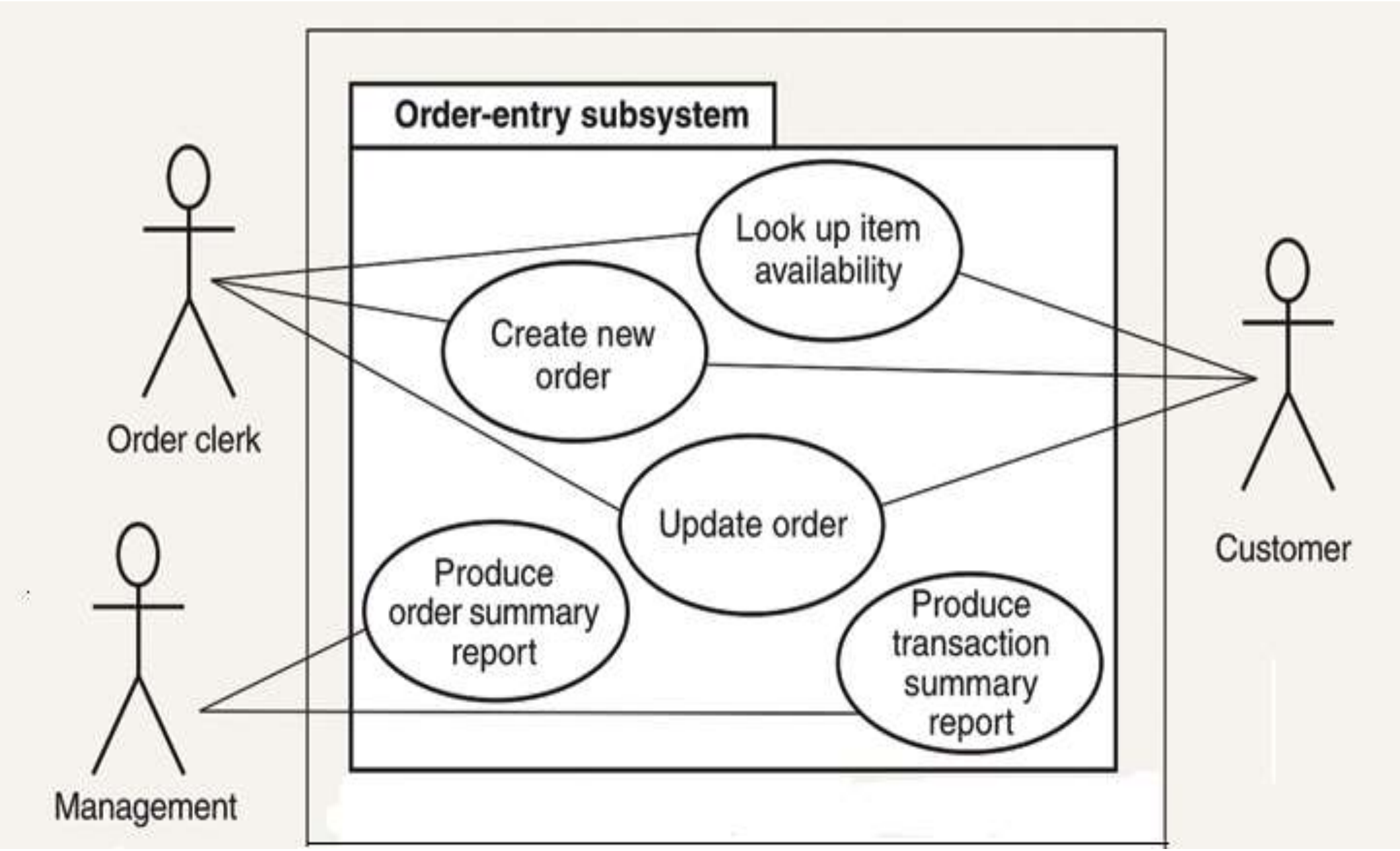
Use Case Packaging



Use-Case Model using Packaging



Use Cases of *Order Entry* Subsystem




WE NEED A VOLUNTEER FOR OUR BRAIN CHIP EXPERIMENT.
NO WAY WOULD I WANT...
DILBERT.COM @SCOTTADAMSSAYS
DON'T STRUGGLE. BREATHE IN.
8-17-22 2022 Scott Adams, Inc./Dist. by Andrews McMeel
HE'S GOING TO BE MAD WHEN HE WAKES UP.
NAH. THE CHIP WILL TAKE CARE OF THAT.

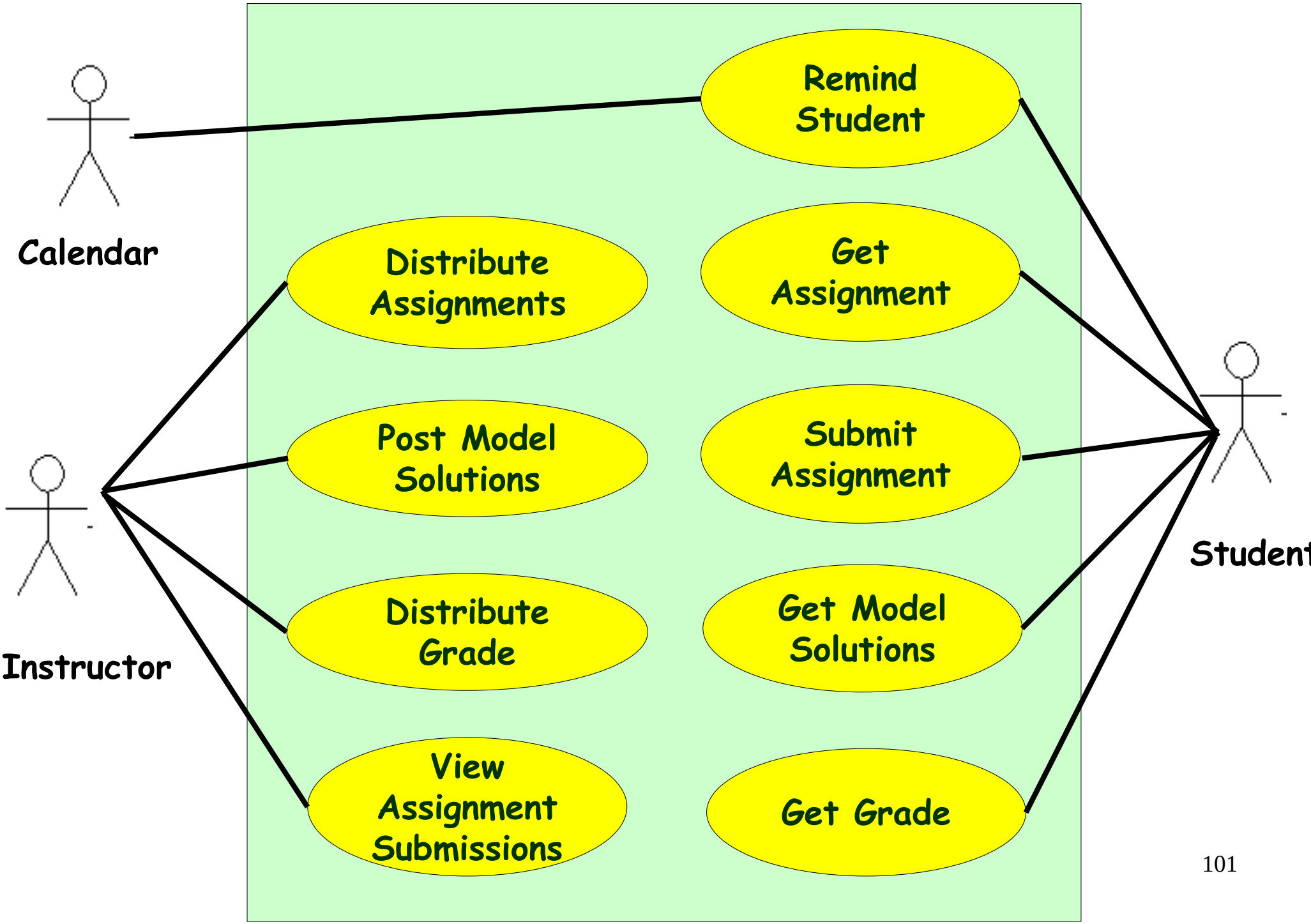
Use Case Style Notes (Ambler, 2005)

- A Use case name should begin with a verb.
- While use cases do not explicitly imply timing:
  - Order use cases from top to bottom to imply timing -- it improves readability.
- **The primary actors should appear in the left.**
- Do not use arrows on the actor-use case relationship.
- **To initiate scheduled events include an actor called "time"**
- **Do not show actors interacting with each other.**
- <<include>> should rarely nest more than 2 deep.

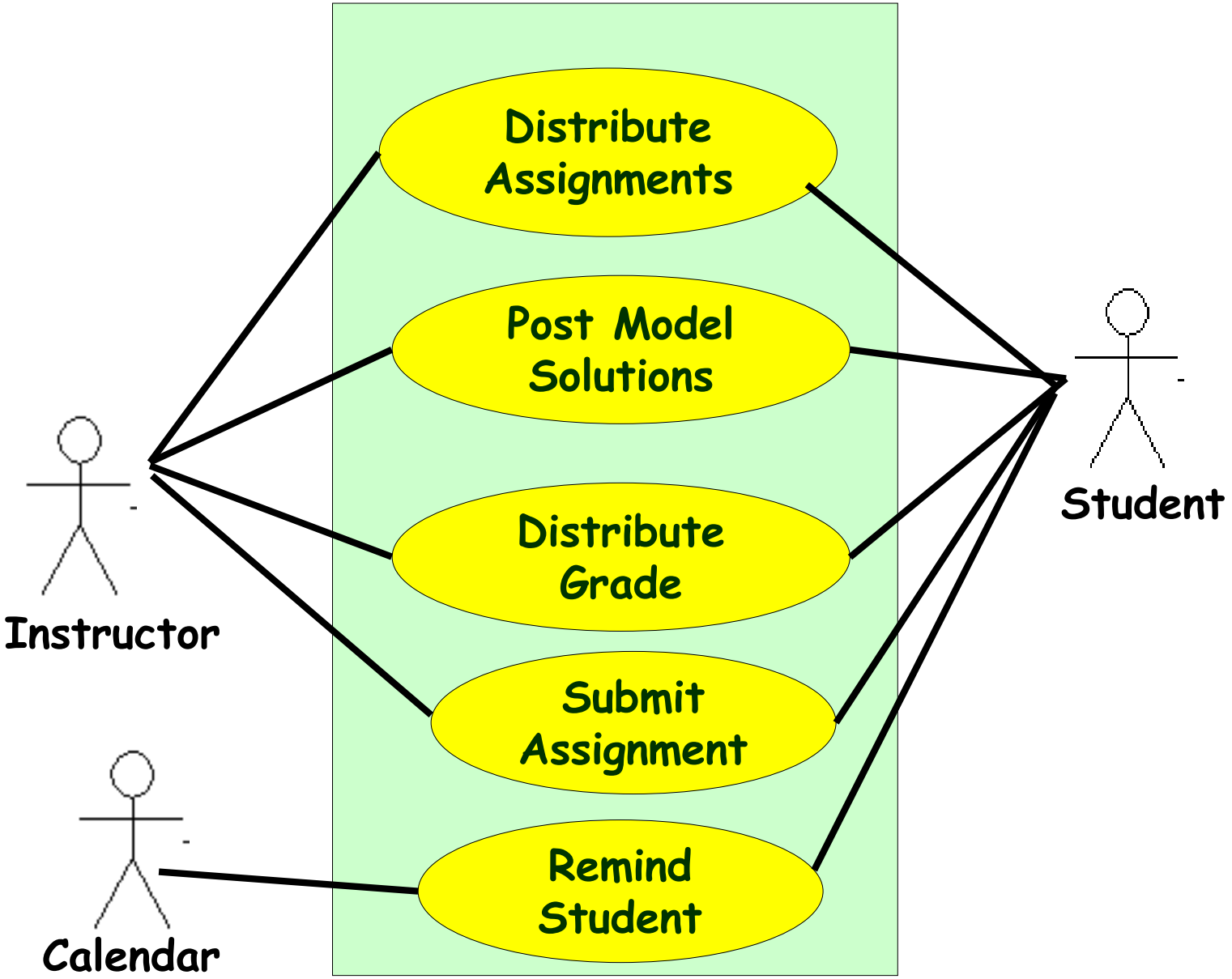
Exercise: Home Assignment System - Use Case Model

- HAS will be used by the instructor to:
  - Distribute the homework assignments,
  - Review the students' solutions,
  - Distribute suggested solution,
  - Assign grade to each assignment.
- Students can:
  - Download assignments and model solutions
  - Submit their assignment solutions and view grade
- System:
  - Automatically reminds the students a day before an assignment is due.

Quiz: Naïve Solution (Inferior)



Quiz: Alternate (Better) Solution



HW: Video Rental Store Software

- A video rental store has a large collection of video CDs and DVDs in VHS and MP4 format as well as music CDs.
- A person can become member a member by depositing Rs. 1000 and filling up details such as name, address, and telephone number. A member can cancel his membership and take back his deposit, if he has no dues outstanding against him.
- Whenever the store purchases a new item, details such as price and date of procurement are entered. The daily rental charge is also entered by the manager. After passage of a year, the daily rental charge is automatically halved.
- A member can, at a time, take on loan at most two video CDs and one music CD. The details are entered by a store clerk and a receipt indicating the daily rental charge should be printed by the software.
- Whenever a member returns his loaned item(s), the due amount to be paid is displayed. After the amount is paid, the items are marked returned.
- If a customer loses or damages any item, the full price of the item is charged to him and the item is removed from the inventory.
- If an item is not lent out by anyone for even once over a year, the item is sold at 10% of the purchase price and is removed from the inventory.
- The manager can, at any time, check the profit/loss account.

HW: IIT Security Software

- The staff and students of IIT register their vehicle by filling up a form which a security staff would enter into the system.
- Each time a vehicle enters or leaves the campus, a camera reads the registration number of the car, and the model of the car. If the car is registered, the check gate should lift automatically to let in (or out) the vehicle. The details regarding the entry and exit of the vehicle are registered in the database. For outside vehicles, the driver needs to enter the purpose of entry, the model number, the registration number, and a photograph of the vehicle is stored in the database.
- When an outside vehicle leaves the campus, the details are automatically registered in the database. For any external vehicle that is inside the campus for more than 8 hours, the driver is stopped by the security personnel manning the gate, queried to satisfaction, and the response are recorded. Considering that there have been several incidence of speeding and rough driving, the security personnel are empowered to telephone the registration number of errant vehicles to the main gate. For outside vehicles, the driver is quizzed at the check gate and his future entry is barred if the response is not satisfactory. For inside vehicles, a letter is issued to the dean to deal with the employee or student as the case might be.
- The security officer can check the data regarding the number of vehicles going in and coming out of the campus (over a day, month or year), the total number of vehicles currently inside the campus.

HW: Personal Library System

- The software is required to manage the collection of books by individuals.

- A person can have a few hundreds of books. The details of all the books such as name of the book, year of publication, date of purchase, price, and publisher must be entered. A book is to be given a unique serial number by the computer which would be written by the owner using a pen on the book. Before a friend can be lent a book, he must be registered. The registration data would include name of the friend, address, land line number, and mobile number. Before a friend is issued a book, the various books outstanding against him also with the date borrowed are displayed. The date of issue and the title of the book are stored. When a friend returns a book, the date of return is stored and the book is removed from his borrowing list. Up on query, the software should display the name, address, and telephone numbers of each friend against whom books are outstanding along with the titles of the outstanding books and the date on which those were issued.

- The owner of the library software, when he borrows books from his friends, would enter the details regarding the title of books borrowed, and the date borrowed. Similarly, the return details of books would be entered. The software should be able display all the books borrowed from various friends. The owner should be able to query about the availability of a particular book, the total number of books in the personal library,