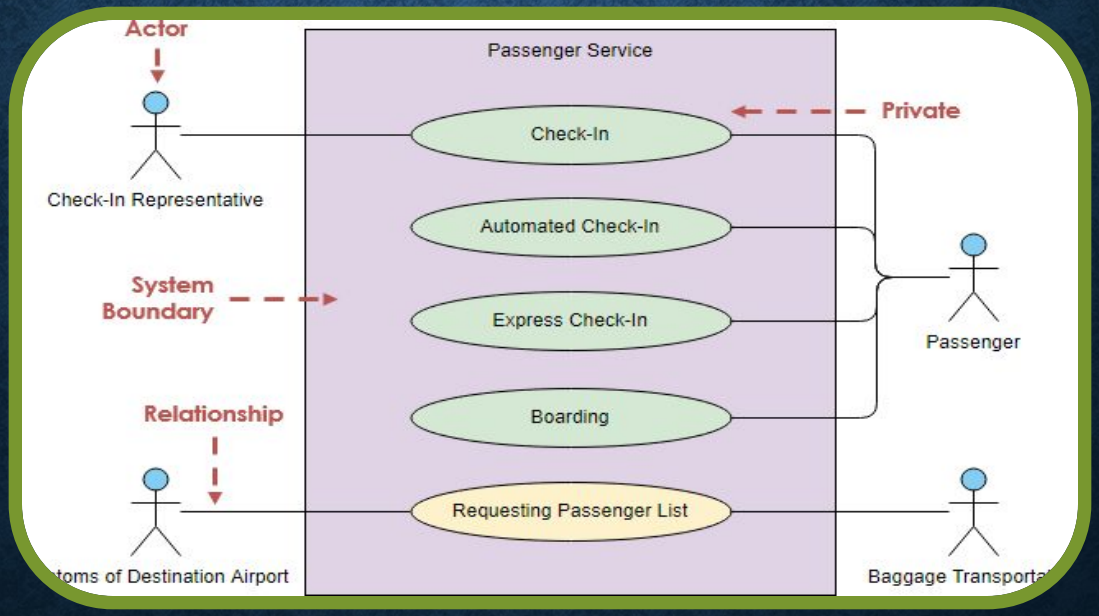
USE CASE MODELING



USE CASE:

- Use case are written stories to fulfill some stakeholders goals in order to discover and record functional requirements
- These are functional requirements
- It must provide an observable user value
- In simple
  - A use case is one usage of the system
    - Actor appears, interacts, goes away
  - It captures the required behaviour of the system
    - What must happen within the system
    - The business rules that must be followed
  - It captures the final system state
    - What's changed

USE CASE:

- **A *use case* defines a** goal-oriented **set of interactions between** external actors **and the** system **under consideration. *Actors* are parties outside the system that interact with the system. (UML 1999)**

- **"A use case is a** narrative document **that describes the** sequence of events **of an** actor **(an external agent) using a** system **to complete a process.“ [Jacobson92]**

- **“A d**escription **of** set of sequences of actions**, including** variants**, that a system performs that yield an** observable result of value **to an** actor**.“ [Booch99]**

WHY USE CASE?

- To develop an increased understanding of the problem.
- Communicate with the end users to make sure how we are perceiving and solving their problems.
- Provide a road map and organizational framework for the actual development process.
- Use cases are used to scope system functionality. They help to produce a more concise, less ambiguous statement of the requirements, with better closure and have replaced low level feature lists of requirements
- They will be used as an aid to vocabulary capture

USE CASE MODELING:

- Use-case modeling is a specialized type of structural modeling concerned with modeling the functionality of a system
- UP defines use case model within requirements discipline, as a set of all use cases, that models the system functionality and enjoinment

ELEMENTS OF USE CASE:

A use case is a collection of related success and failure scenario **that describe** actors **using the** system **to support a** goal**. [Larman]**

- Scenario
- Actors
- System
- Goal

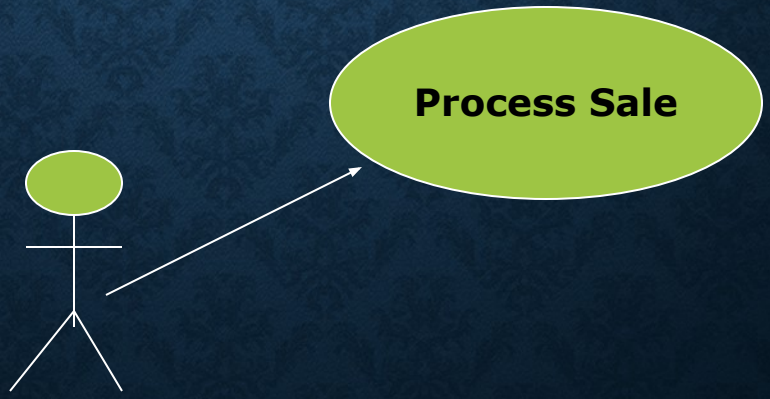
SCENARIO:

- A scenario is a sequence of steps describing an interaction between a user and a system
- It is also called a use case instance, that is a specific sequence of actions and interactions between actors and the system under consideration
- Keep your focus on “how a bank works, not how a computer program that simulates a bank, is used “

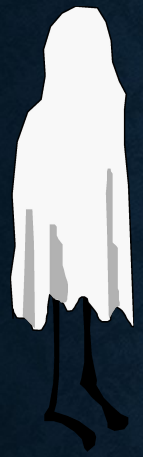
ACTOR:

•An external entity (person or machine) that interacts with or uses the system

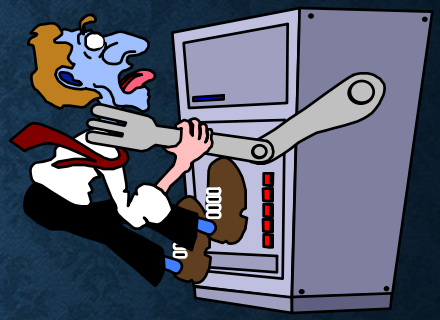
•An actor is external to a system, interacts with the system, may be a human user or another system, and has goals and responsibilities to satisfy in interacting with the system. Actors address the question of who and what interacts with a system. In the UML, an actor is shown as a "stick figure" icon.



Organisation
(Accounts Receivable
Tax Agency
Kitchen)



Role
(Employee
Customer
Guest)



Device
(Drink Dispenser
Cash Register
Door Lock)

System
(Accounting Package
Product Inventory
Bar System)

TYPES OF ACTORS:

- Primary Actor
  - Has user goals, which drive the use cases.
- Secondary Actor
  - External interfaces or protocols that provide a service
- Offstage Actor
  - Has some interest in behavior of the use case

SYSTEM:

- System
  - Everything that the project has control over
- System Boundary
  - System boundary can be a computer system, organization boundary, or department boundary. The system functions and actor may change depending on the system boundary location.
    - Understand the System
    - Understand the limitations of the system
    - Identify the relation of the actor with the system

GOALS:

- **Goal**
  - **Systems are built to meet certain stakeholder needs or goals; these define what the system is supposed to do**
  - **While identifying or capturing Goals focus should be on Elementary Business Processes.**
- **Elementary Business Process (EBP)**
  - **A** task **performed by one person in one place at one time, in response to a** business event**, which adds** measurable business value **and leaves the data in a** consistent state
- **Use case and goals**
  - EBP **level use case is called a** user goal level user case **as it emphasize to fulfill a goal of a user (Actor) of the system**

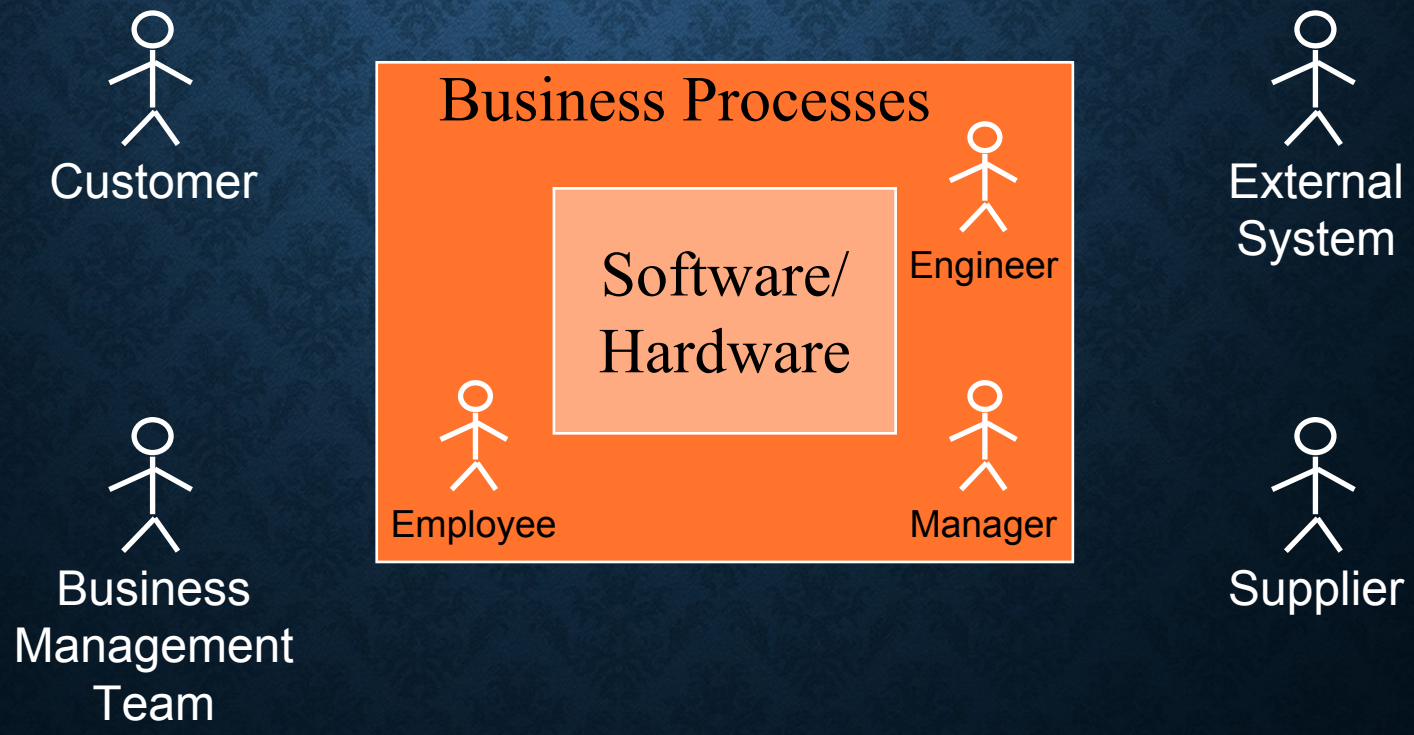
FINDING PRIMARY ACTORS, GOALS, AND USE CASES

PROCEDURE:

- Choose system boundary
- Identify primary actors
- Identify user goals
- Define use case that satisfy each use case

STEP 1: SYSTEM BOUNDARY

- **Organization Perspective:**
  - **Understand the organization structure, behavior, business processes and rules.**
  - **Identify elementary business processes**
- **System Perspective:**
  - **Try to focus on what is in the scope of the system or what is outside the scope of system**
  - **Identify external and supporting actors**



STEP 2 & 3: FIND PRIMARY ACTORS AND GOALS

- Identify who has direct intervention with the system and drives the system
- Ask who need there goals to be fulfilled from our system
- Primary Actors and Goals depend upon system boundary
- The actor goal list

Actor <<Type>>	Goal<<Type>>

STEP 4: DEFINE USE CASES

- **Identify one EBP-level Use case for each user goal.**
- **Name the use case similar to the goal**
- **Try name to be start with a verb**
- **Collapse CRUD (create, read, update, delete) separate goals into one use case**
- **Decide of use case** description level **or type of use case**

TYPES OF USE CASE:

- **Use cases are written in different formats depending on the** need**. Such as**
  - Use case in function context
    - Primary
    - Secondary
  - Use case in Detail level context
    - High level/Black box use cases
    - Expanded/White box use cases
  - Use case in Description level context
    - Essential
    - Concrete/ Real
  - Use case in Formality context
    - Brief
    - Casual
    - Fully dressed
      - One column
      - Two column

Use case in function context

- Primary - These functions are required and are common main processes.
- Secondary - These functions are secondary to the system or rarely occur. Don't need these functions in this iteration. This type of use case is rarely done.

Use case in Detail level context:

- **High Level/Black box use cases**
  - **Brief with no detail, they do not describe the system** internal functionality **or its components.**

- **Expanded/White box use cases**
  - **More detailed with information about every** step in the process**. Don't describe how the system responds.**

Use case in Description level context

- **Essential**
  - **Are expressed in an ideal form that remains relatively** free of technology and implementation detail; design decisions **are deferred and abstracted, especially those related to the user interface.**

- **Concrete/ Real Use Case**
  - **Concretely describes the process in terms of its real current design, committed to specific input and output technologies, and so on. When a** user interface **is involved, they often show screen shots and discuss interaction with the widgets.**

Use case in Formality context:

- Brief
  - One paragraph summary usually of main success scenario
- Casual
  - Multiple paragraphs that covers various scenarios
- Fully dressed
  - All steps and variants are written in detail, along with supporting sections
  - One column: System
  - Two column: Actor system interaction

Use case

Writing Requirements in Context

ELEVATOR PROBLEM

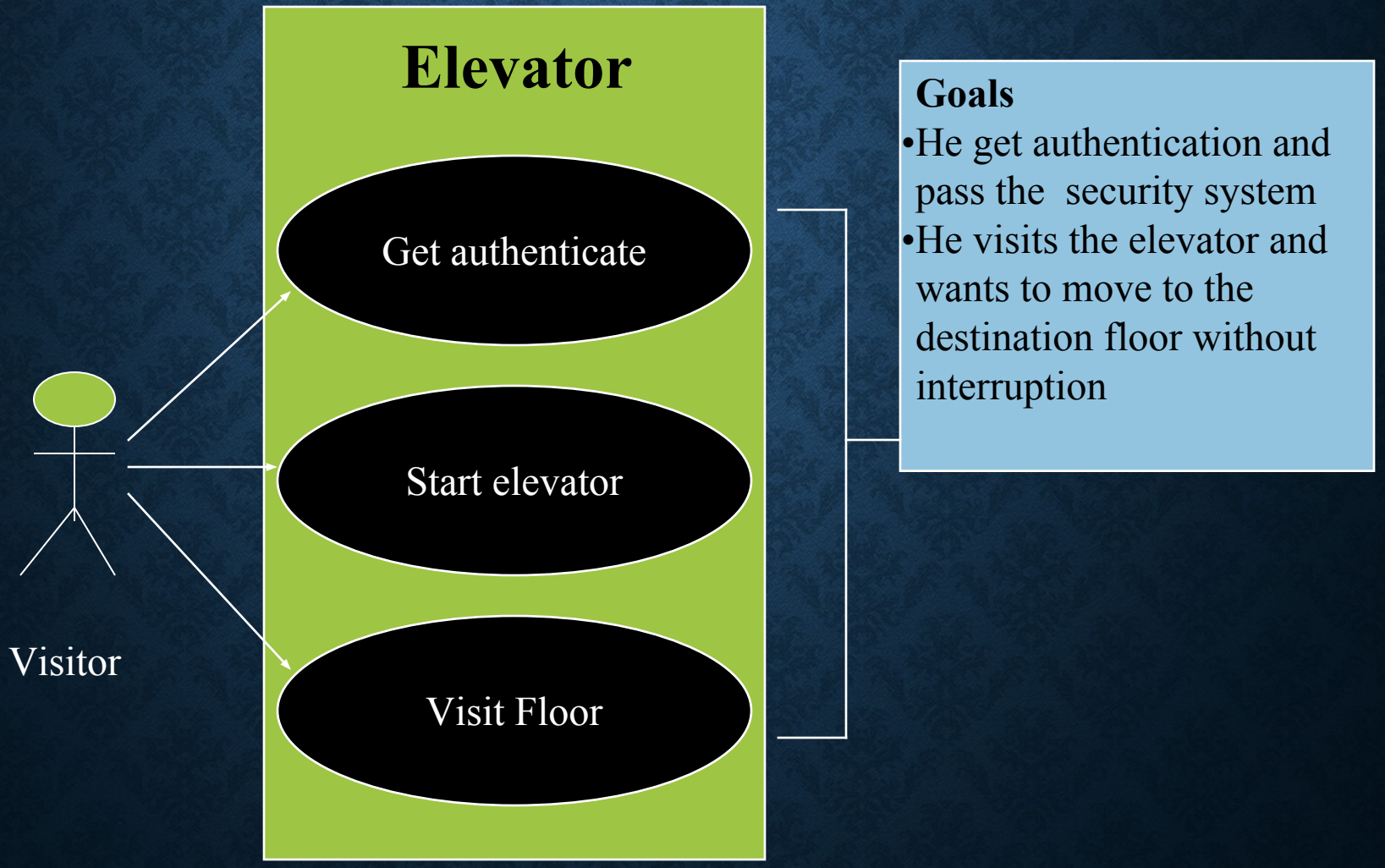
A product is to be installed to control n elevators in a building within m floors. The problem concerns the logic required to move elevators between floors according to the following constraints

1. Each floor , except the first floor and the top floor, has two buttons, one to request an up elevator and one to request a down elevator. These illuminates when pressed. The illumination is cancelled when an elevator visits the floor and then moves in the desired direction
2. Each elevator has a set of m buttons, one for each floor. These illuminates when pressed and cause the elevator to visit the corresponding floor. The illumination is cancelled when corresponding floor is visited by the elevator
3. There is a display window that shows the current floor being visited by the elevator
4. When an elevator has no request, it remains at its current floor with its door closed

PROBLEM CUSTOMIZATION:

- Visitor have to pass through a security system, in which he has to enter a card to get authenticate before entering the elevator
- There is only one elevator in the building
- There are 8 floor in the building

IDENTIFY ACTOR, GOALS, USE CASES:



FULLY DRESSED EXAMPLE:

- Use case name
  - Use case UC1: Start Elevator
- Primary Actor
  - Who calls on system services to fulfill a goal
    - user: visitor
- Stakeholders and Interests
  - Interacting with system, and have specific interests in the system operation
    - Visitor: he visits the elevator and wants to get in side without interruption
    - Elevator boy: he helps people in navigation and elevator operation when asked
    - Electrician: he solve elevator functionality related problems

- **Preconditions**
  - What must always be true before beginning of a scenario, they are not tested in use case but are considered true before hand
    - Visitor is authenticated and is identified
- **Success Guaranties (Post condition)**
  - What must be true on success completion of the use case
    - Visitor presses the button and get inside the elevator and elevator door close's

- Main success Scenario
  - Describes typical success path, it stores three record,
    - interaction between actors
    - A validation
    - A state change

1. Visitor A authenticate himself
2. Visitor A presses the Up floor button at floor 3 to request an elevator, he wishes to go to floor 7
3. The up floor button turned on
4. An elevator arrives at floor 3
5. The up floor button turn off
6. The elevator doors open
7. The timer starts
8. Visitor A enters the elevator
9. The elevators door close after the time out

- Extensions (Alternative Flow)
  - They indicates all the other scenarios or branches, both success and failures
  - An extension has two parts; the condition and the handling
  - At the end of the extension handling the scenario merges back with the main scenario
    - 1a. Unauthorized Visitor
      - System indicates error message

- Special requirements
  - Any non functional requirement specific to use case under consideration
    - Language internationalization on the text display

- Technology and Data Variations list
  - Technology constraints of customer
    - Visitor identification by card system

- Open Issues
  - Any other relevant information
    - Department laws

USE CASE DIAGRAM

Super Market Example

Use case Diagram:

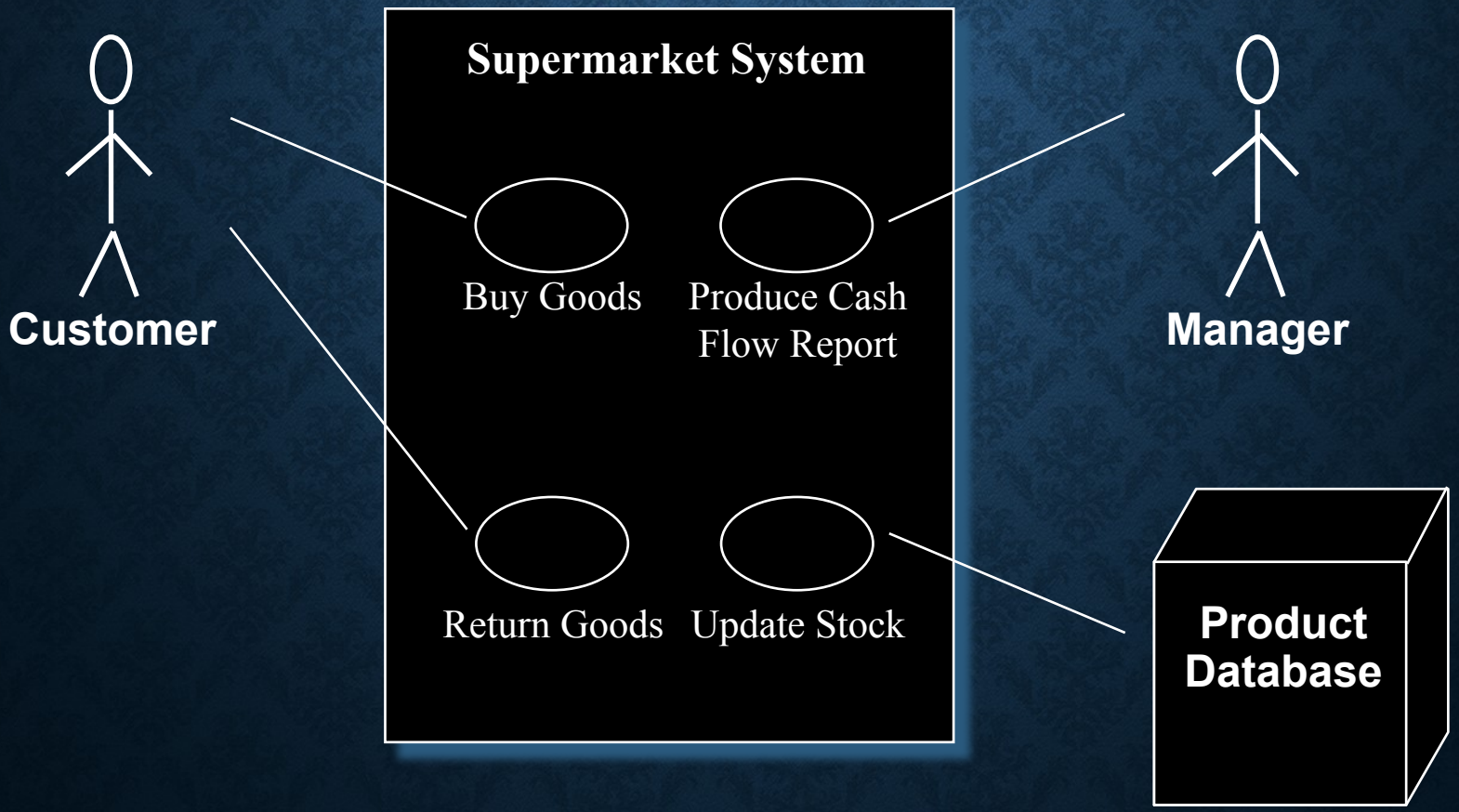
- A pictorial representation of actors, use cases and the system boundary
  - UML allows various rendering of items
  - Draw a simple diagram but in conjunction with the actor goal list
- Captures system functionality as seen by users
- Built in early stages of development
- Purpose
  - Specify the context of a system
  - Capture the requirements of a system
  - Validate a system's architecture
  - Drive implementation and generate test cases
- Developed by analysts and domain experts

USE CASE DIAGRAM VIEWS:

- System/Business level use case diagram
- Analysis level use case diagram

SYSTEM/BUSINESS LEVEL USE CASE DIAGRAM:

- Show system level or black box use cases
- Usually captured in Inception

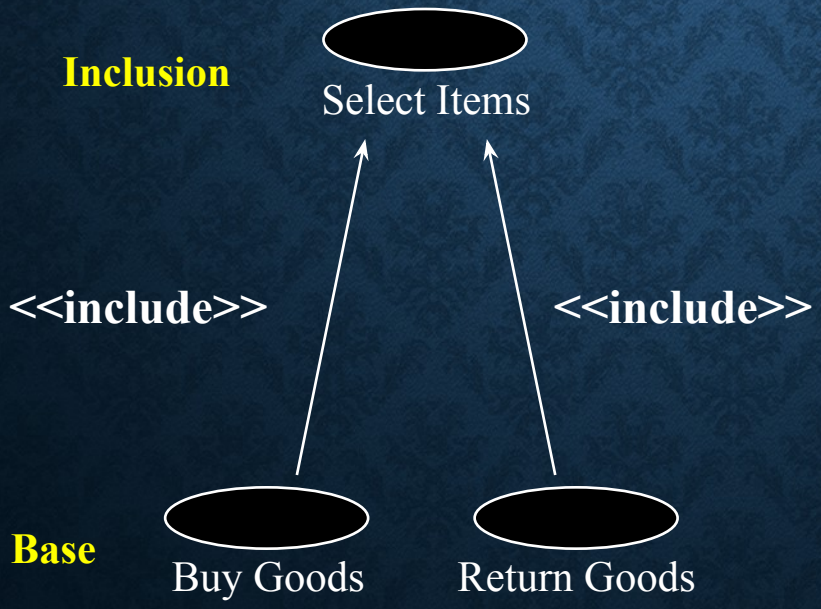


ANALYSIS/ DETAIL LEVEL USE CASE DIAGRAM

- Show detail level or white box use cases
- Concrete, Abstract, Base and Addition Use cases
- Add inclusion, extension and generalization relationships
- Usually captured in Elaboration iteration 2

INCLUDE RELATIONSHIP:

- Extract a common sequence of events or partial behaviour that is common across several use cases
  - From multiple use cases
  - Create a new inclusion use case
  - Modify the base use cases to include the inclusion
  - Avoid repetition



For each product that the customer selects: the product is identified to the system the system finds the product and looks up its price the price is supplied to the customer

The use case starts when the customer requests to buy goods. Include Select Items use case. When there are no more products, . . .

EXTEND RELATIONSHIP:

- Base use case
  - Add extension points - locations where extensions are allowed
- Extension use cases
  - Specify the added behaviour at one or more extension points

The use case starts when the customer requests to buy goods. Include Select Items use case.

When there are no more products, the total amount is produced. Include Pay for Purchases use case.

If a valid payment is made, then the stock is reduced for each item sold,

extension point: <saver card>

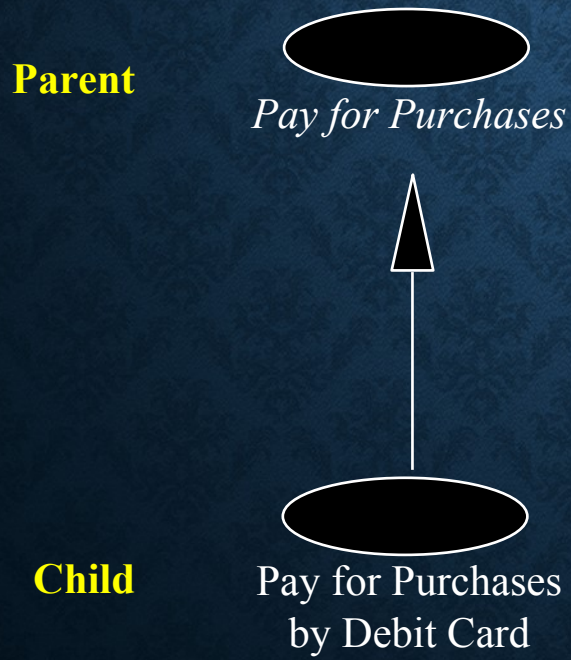
otherwise the goods remain unsold.

segment <saver card>

If the customer provides a saver card, then the number of points on the saver card account is increased by 1 for each pound spent.

GENERALIZATION RELATIONSHIP:

- Parent use case
  - May be concrete or abstract
- Child use case
  - A more specific form of the parent
  - Modifications of the parent behaviour are allowed throughout

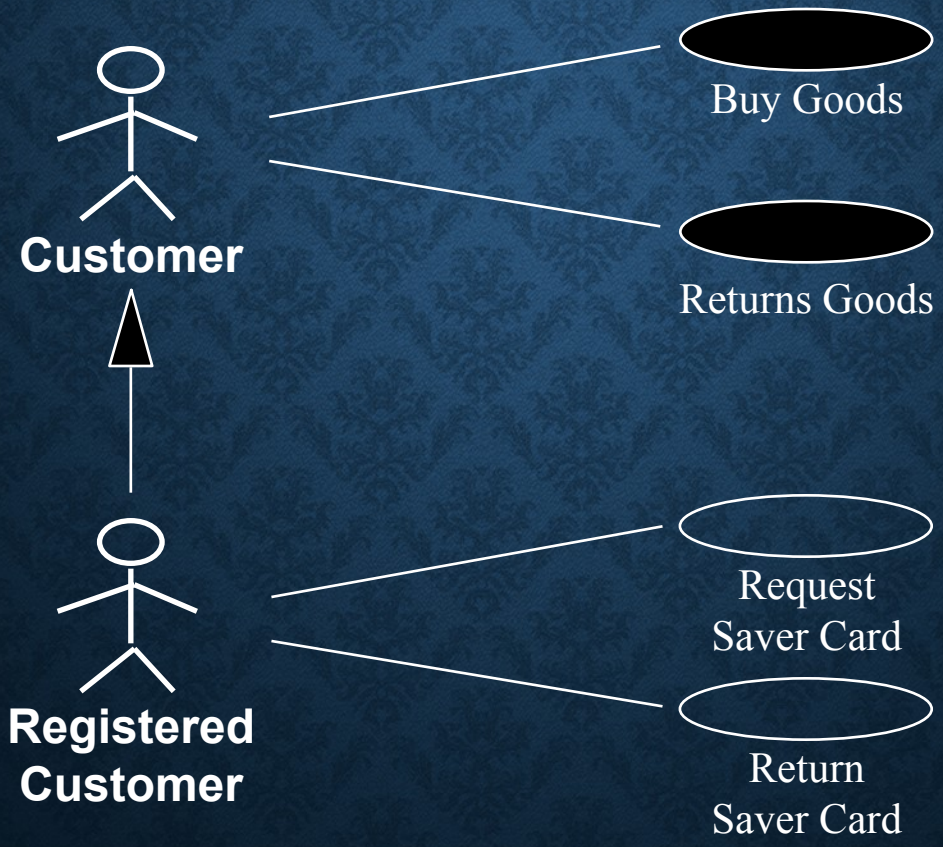


The customer chooses a payment method. The details are checked, and recorded. May be successful or unsuccessful. May be retried.

The customer chooses to pay by debit card. The debit card details are read. The system contacts the debit agency and requests payment of the required amount from the supplied card. If successful the debit transaction is remembered for future reconciliation. If unsuccessful, another debit card may be used.

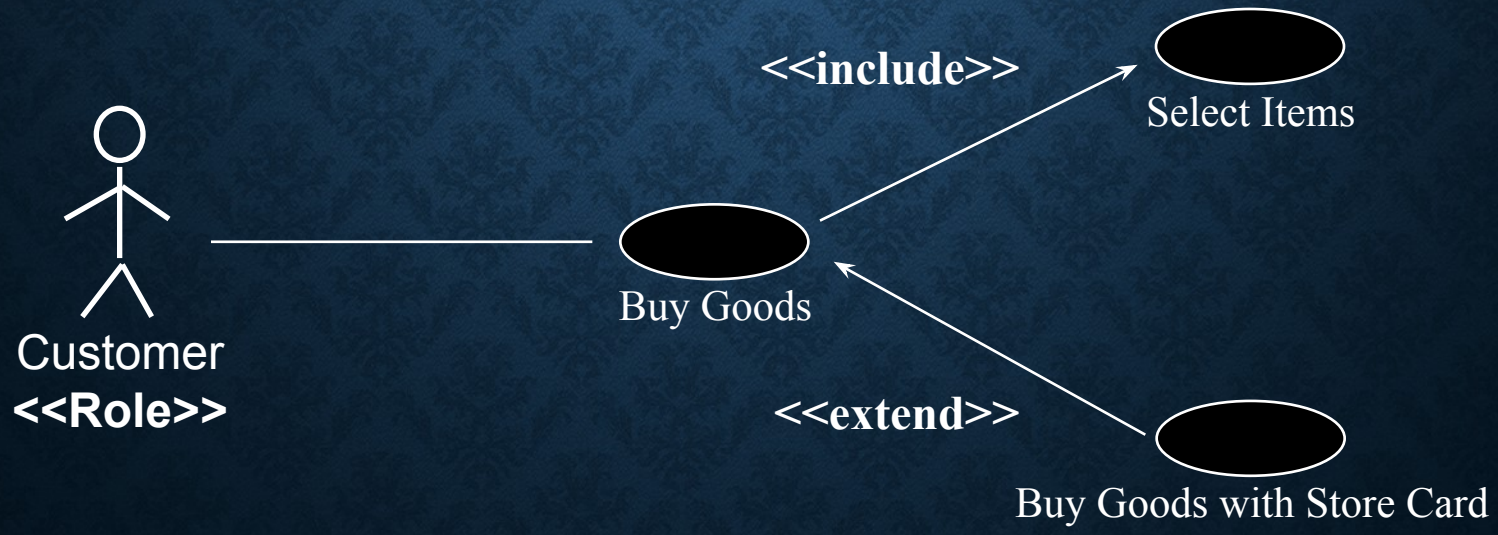
ACTOR GENERALIZATION/SPECIALIZATION

- Generic actors have use cases common to all specialised actors
- Specialised actors have special use cases



Diagramming suggestions

- To categorize an element use stereotypes
  - A stereotype represents the meta-classification of an element
    - New UML element based on standard ones
    - Allows additional semantics to be defined
  - New stereotypes may be added to the meta-model
  - Stereotypes are then used in the domain model



- Show computer system actors with an alternate notation than a stick figure
- Put primary actor on left and supporting actors on right

REFERENCES

- Applying UML and Patterns by Craig Larman
  - Chapter 6, Chapter 25