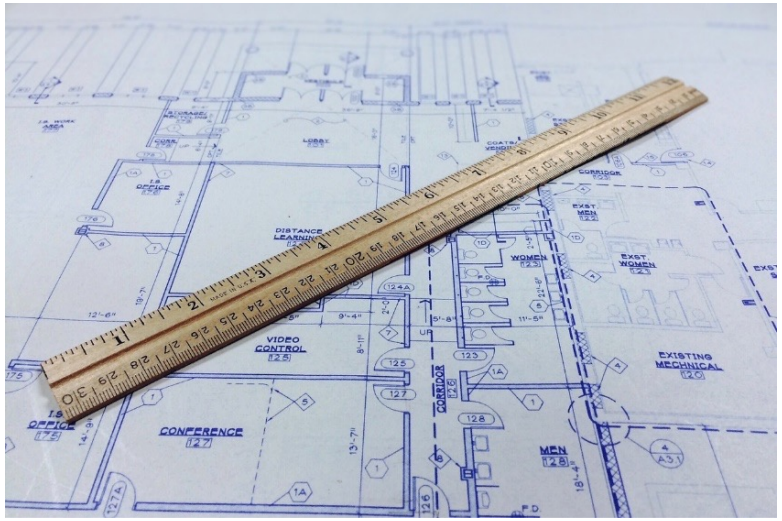
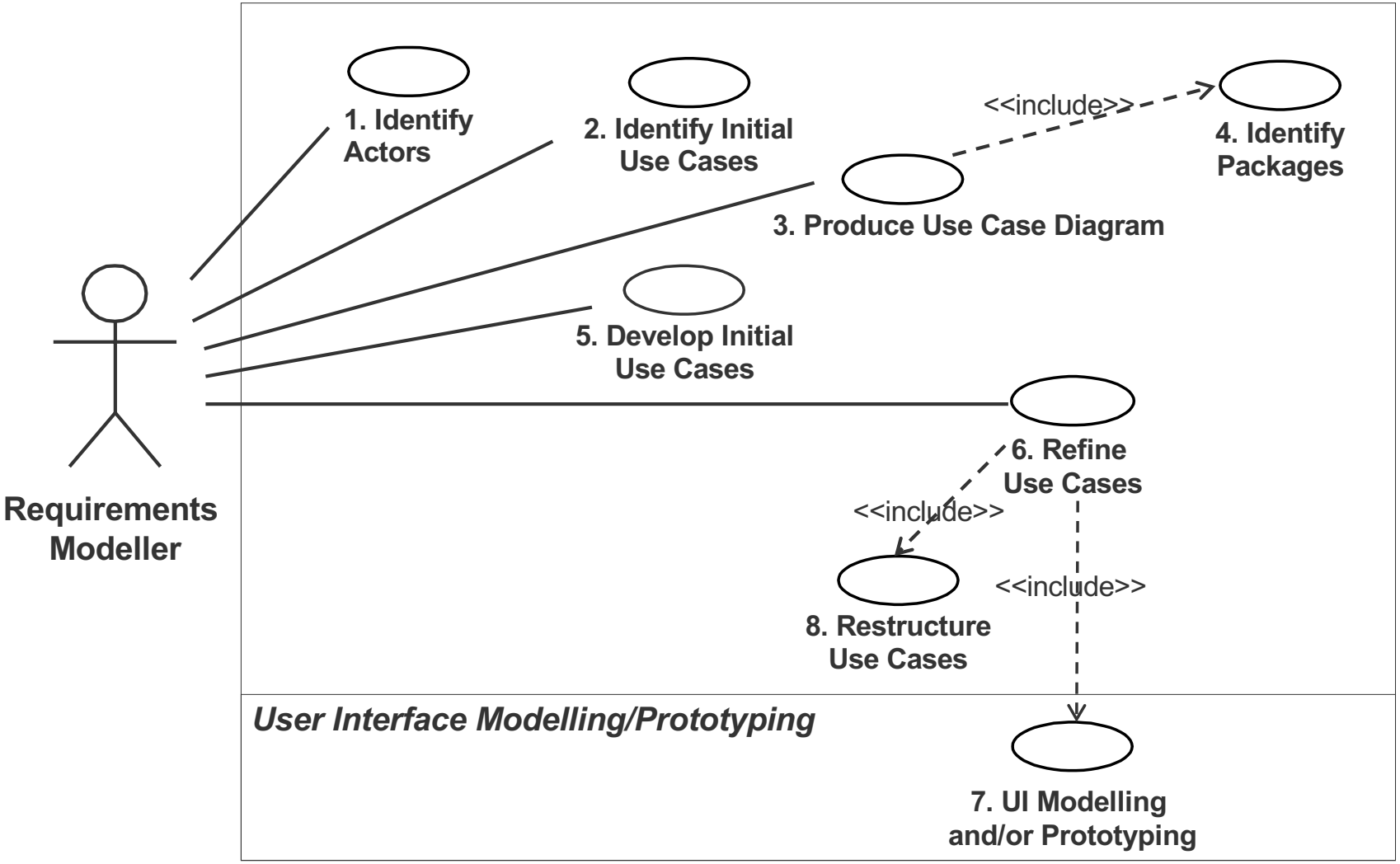
Use Case Modelling Process

The hardest single part of building a software system is deciding precisely what to build.

– Frederick Brooks, *The Mythical Man-Month*



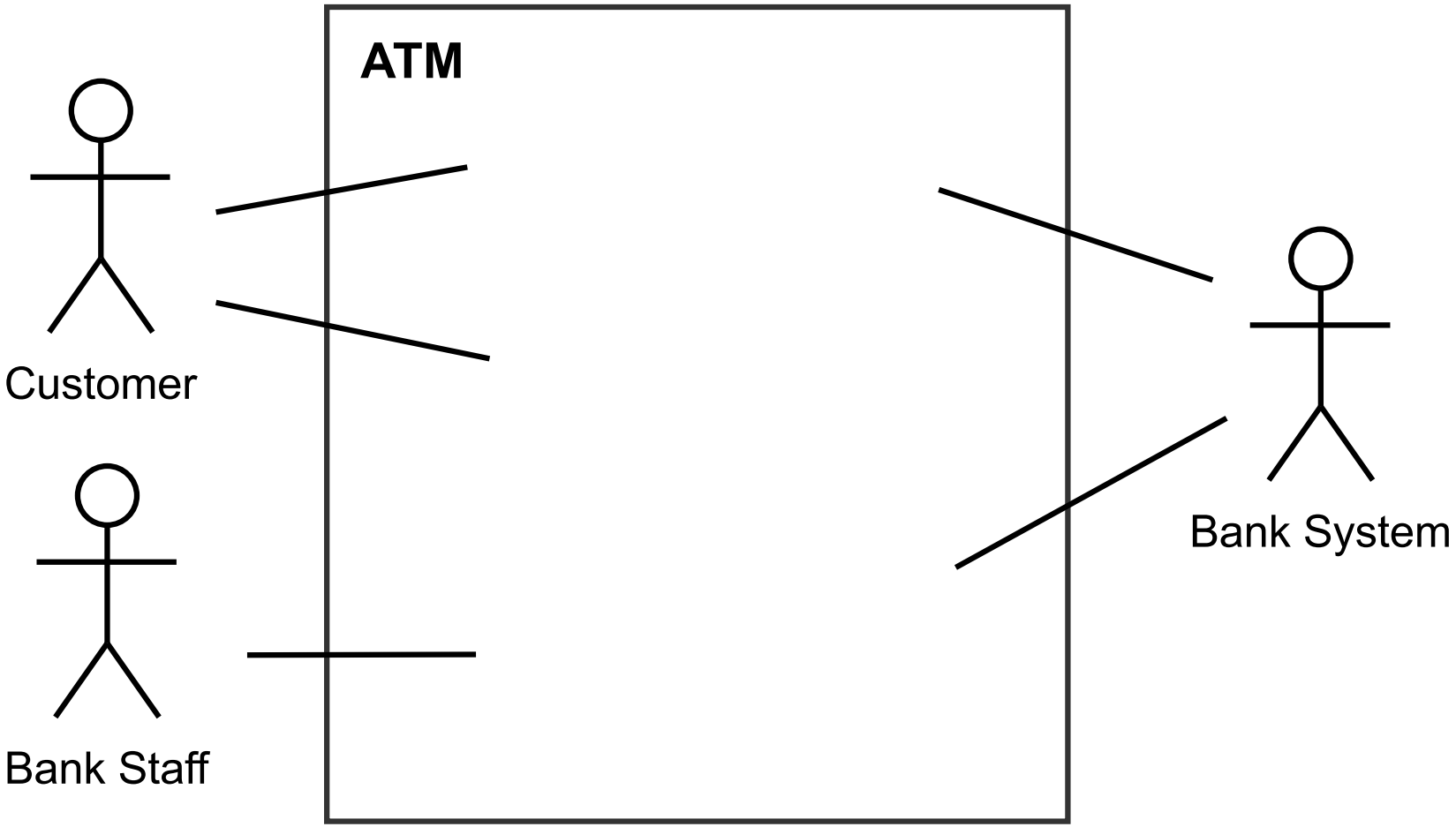
Use Case Modelling Tasks



Task 1: Identify Actors

- **Identifying actors is the starting point**
  - **– "Who is this system supposed to help?"**
- **Finding human actors is relatively straight forward**
- **Other systems are sometimes less obvious**
- **Don't assume that current users will be actors**
  - **– think about potential users**
  - **– i.e. how can the current system be improved**
- **May wish to record current system actors to help think about potential actors**

Examples of Actors – ATM



Actor Description Template

Actor Name	
Description	***A description of the role this actor plays.***
Aliases	***Other names that may be used in the problem domain to describe this actor.***
Inherits	***The name of any other actor from which behaviour is inherited.***
Contact	***Details of the person to contact to clarify information about this actor.***
Protocol	***Only included if this actor represents an external device or system.*** Receives: ***Messages that this actor receives from the system.*** Sends: ***Messages that this actor sends to the system.***

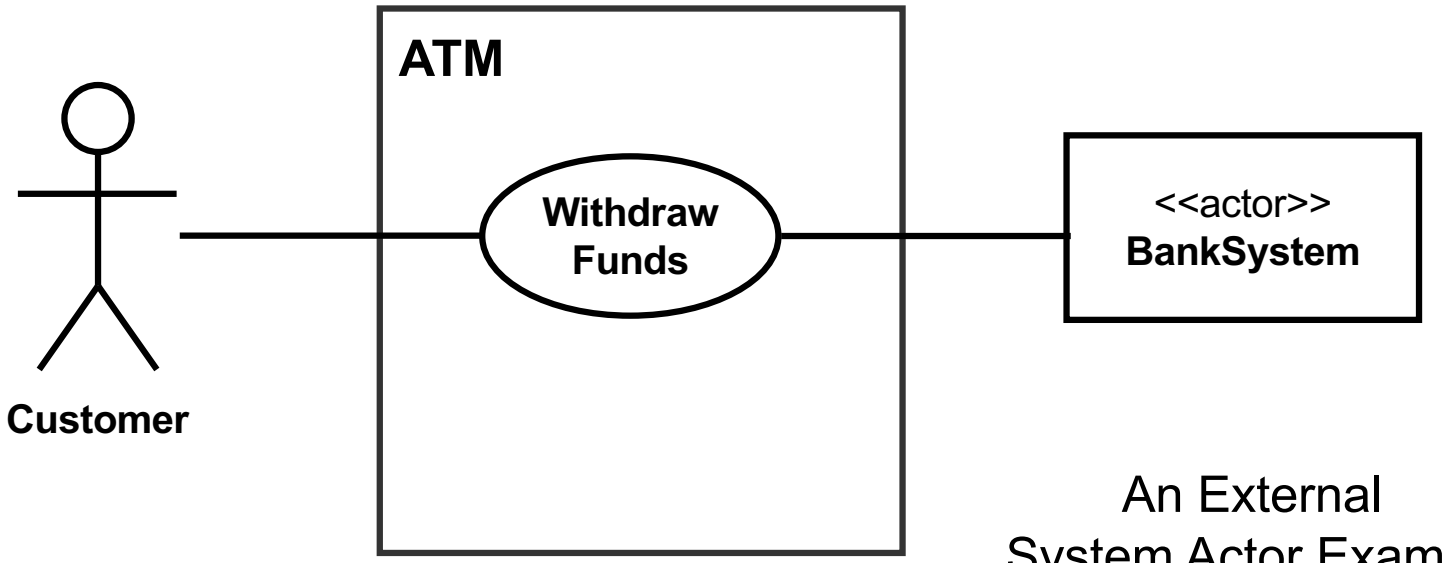
Example Actor Description

Actor Name: Customer
Description: Uses the ATM to conduct bank transactions.
Alias: User
Inherits: none
Contact: Jane Simmonds
Contact Details: 3333 1234

Actor Name: Bank System
Description: External system used to verify customers and transactions.
Alias: none
Inherits: none
Protocol: **Receives:** Validate PIN, Transaction Request … …
Sends: PIN Validation Result, Transaction Approval … …
Contact: Michael Clarke
Contact Details: 3333 5678

External Systems as Actors

- **Entities that interact with a domain are *actors***
- **Actors that are other systems or devices are *External System Actors***



An External
System Actor Example

Importance of Actors

- **Without actors**
  - **We miss requirements**
  - **We miss the "why" of requirements**
- **By confusing actors with users**
  - **We specify a rigid solution**
  - **Users can play multiple roles**

Task 2: Identify Initial Use Cases

- **For each actor note the activities/tasks they do as part of their role**
  - **each activity/task should be listed as a verb phrase**
- **Examine the tasks, or potential tasks, of each actor in turn**
  - **an initial use case can be identified from each activity/task**
- **Interview business users**
  - **what they want the system to do**
  - **how they want to interact with it**

Sources of Use Cases

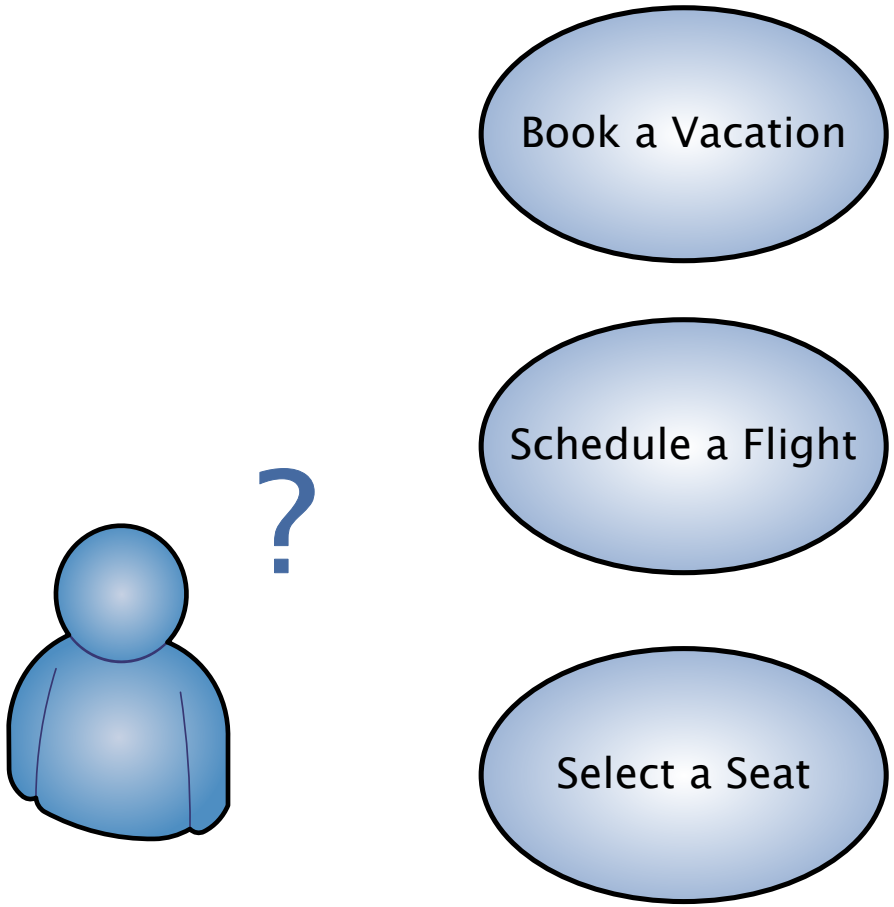
- **Possible sources for use cases (to name a few)**
  - **– domain material**
  - **– interviews with users**
  - **– personal experience**
  - **– workshops with users**
  - **– requirements documents**
- **Identify the logical functioning of the business process – *do not* include technical details**
  - **– State "what" is required and "why"**
  - **– Not "how"**

Tips for Identifying Use Cases

- **The following questions are useful starting points**
  - **What are the main tasks of each actor?**
  - **Will the actor have to read/write/change any of the system information?**
  - **Will the actor have to inform the system about outside changes?**
  - **Does the actor wish to be informed about unexpected changes?**

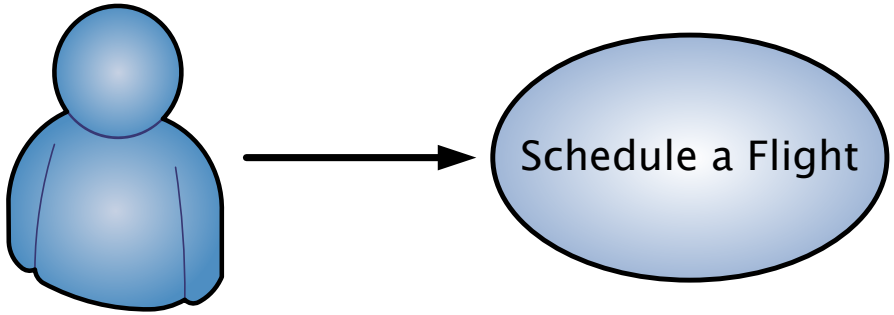
Appropriate Size

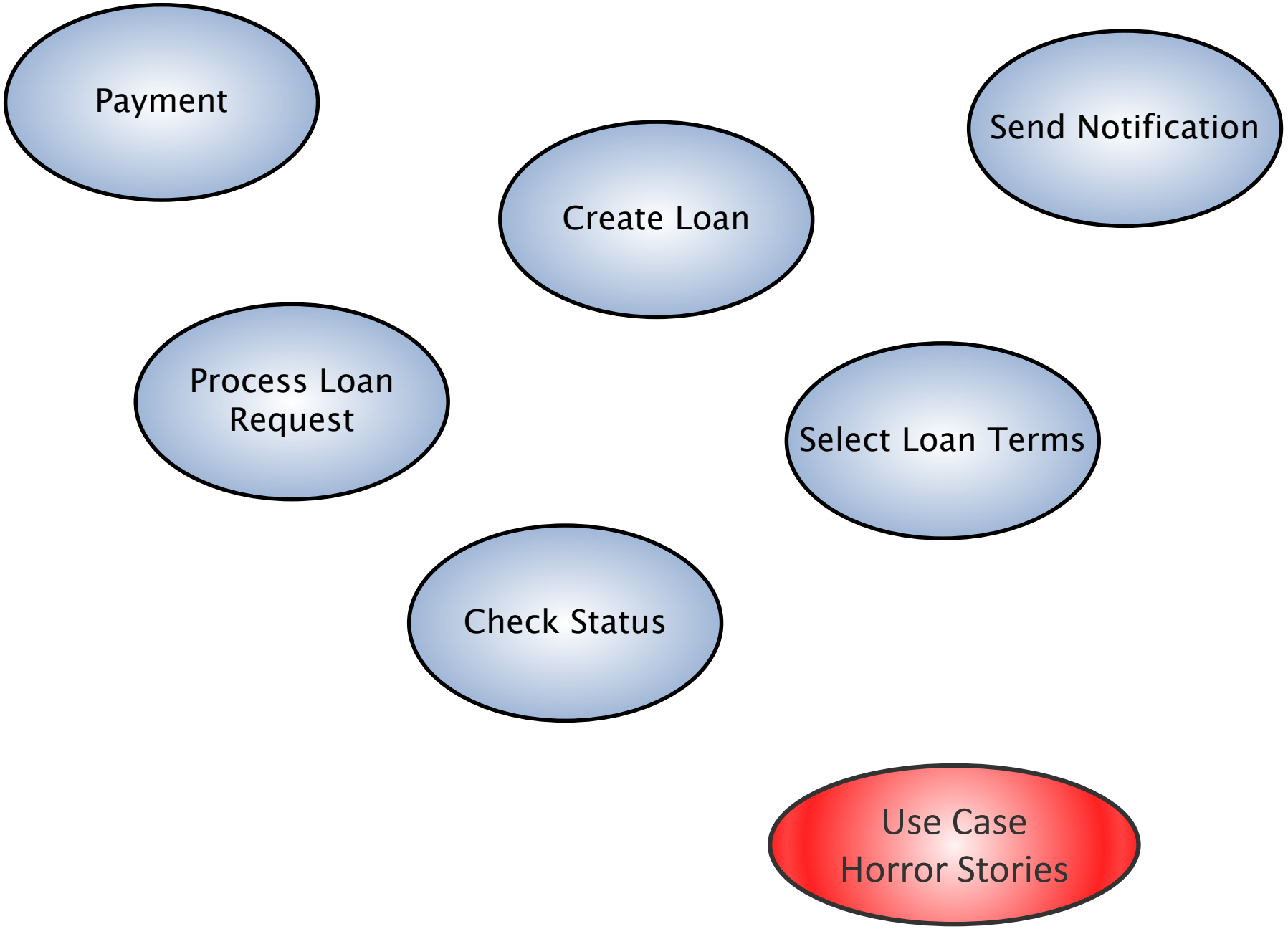
- **How big?**
- **Where does it start?**
- **When does it end?**
- **How much detail?**

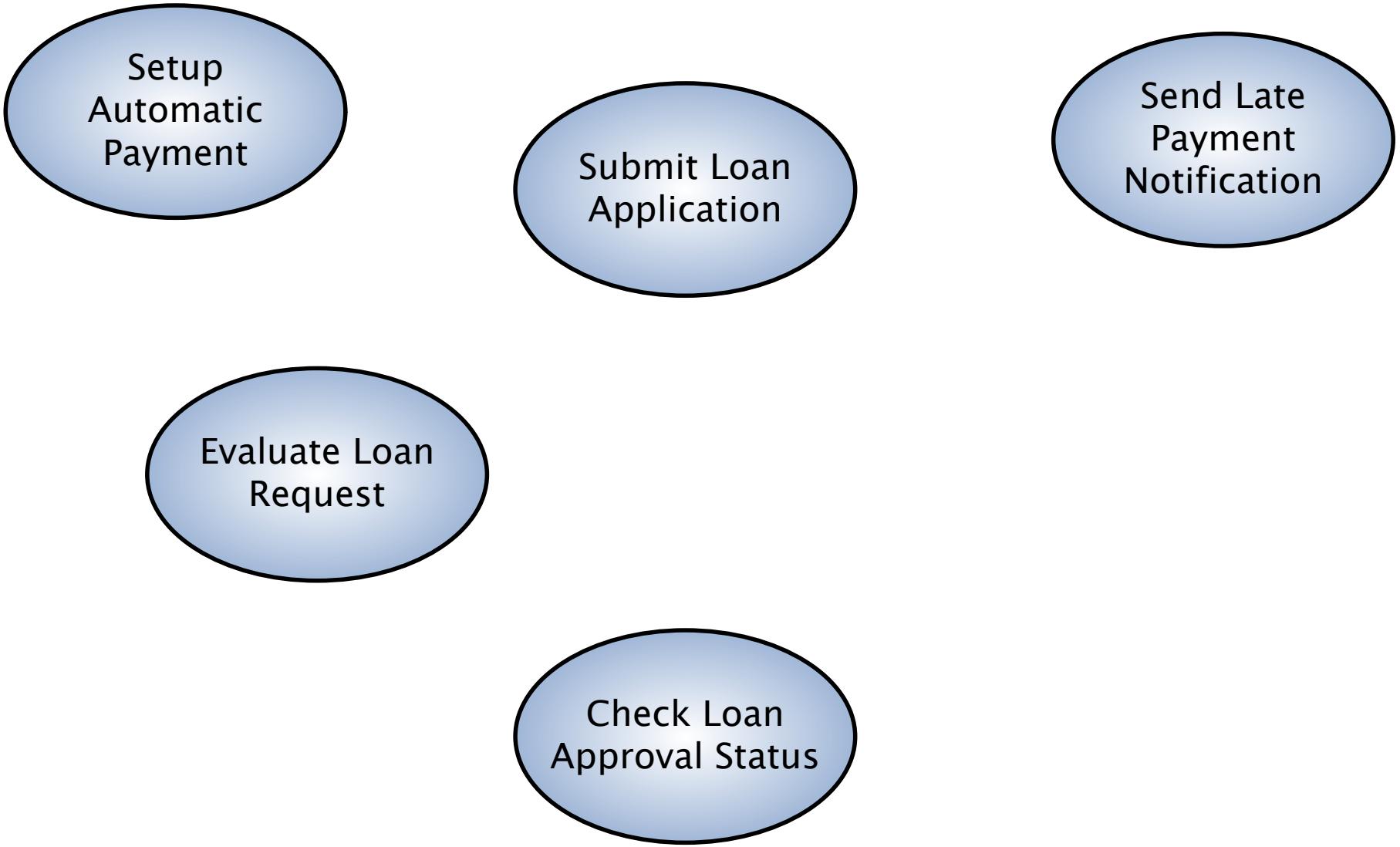


Size: Completable Useful Task

- **Either 100% complete or 0% complete**
- **Named after a goal**
- **Verb Phrase (e.g. Schedule Flight)**
- **Six to ten steps**
- **One actor, one place, one time**




Payment
Send Notification
Create Loan
Process Loan Request
Select Loan Terms
Check Status
Use Case Horror Stories


Setup Automatic Payment
Submit Loan Application
Send Late Payment Notification
Evaluate Loan Request
Check Loan Approval Status

ATM: Example Use Cases

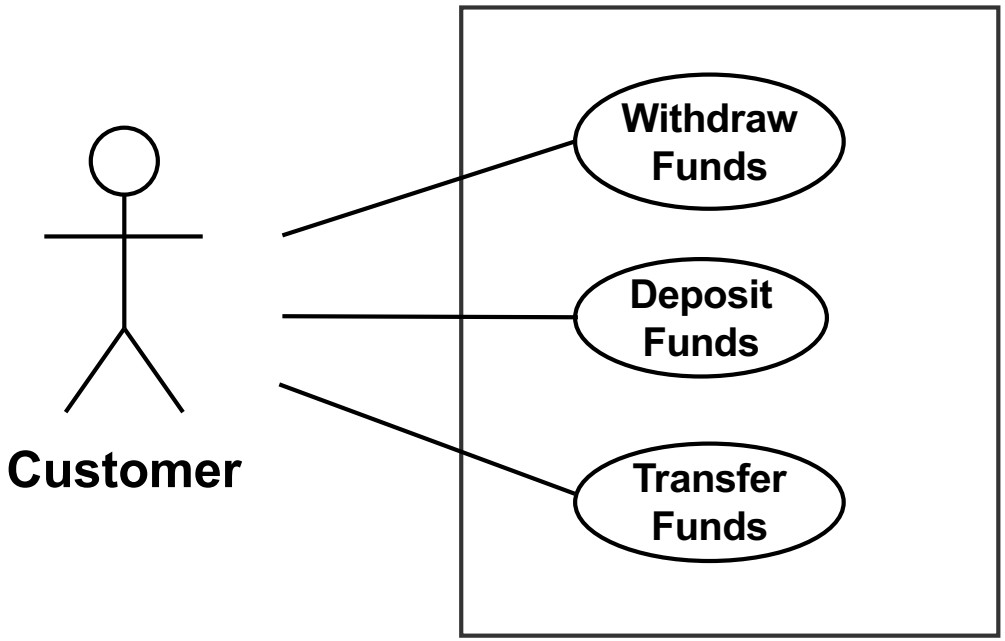
- **Withdraw Funds**
  - **Customer withdraws funds from an account**
- **Deposit Funds**
  - **Customer deposits funds into an account**
- **Transfer Funds**
  - **Customer transfers funds between accounts**
- **Close ATM**
  - **Bank Staff closes ATM**
- **Open ATM**
  - **Bank Staff opens ATM**

Task 3: Draw Use Case Diagram

- **Once actors and their use cases have been identified, draw a use case diagram**
- **Use case diagram is a high-level overview of system**
  - **– actors**
  - **– use cases**
  - **– boundary of the system**
- **Useful for presentation and communication purposes**

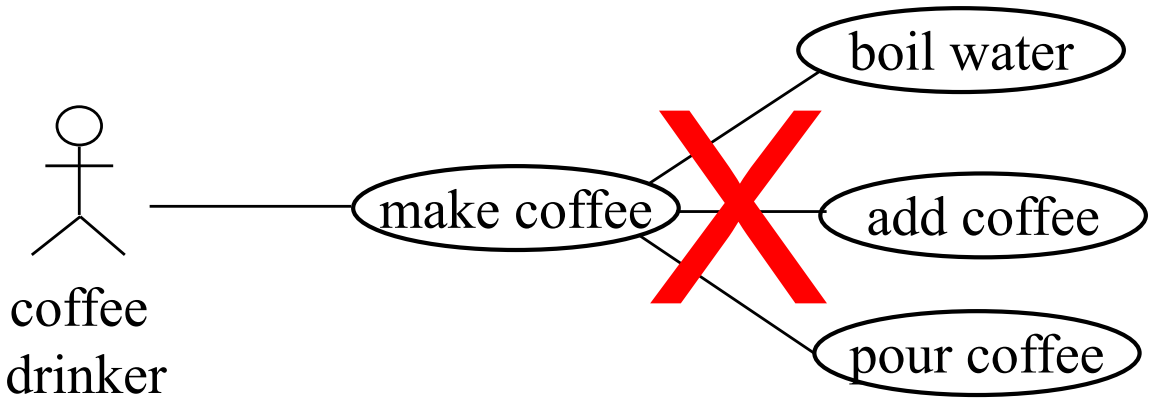
Task 3: Draw Use Case Diagram (cont.)

- **Provides a high-level overview of the system requirements**

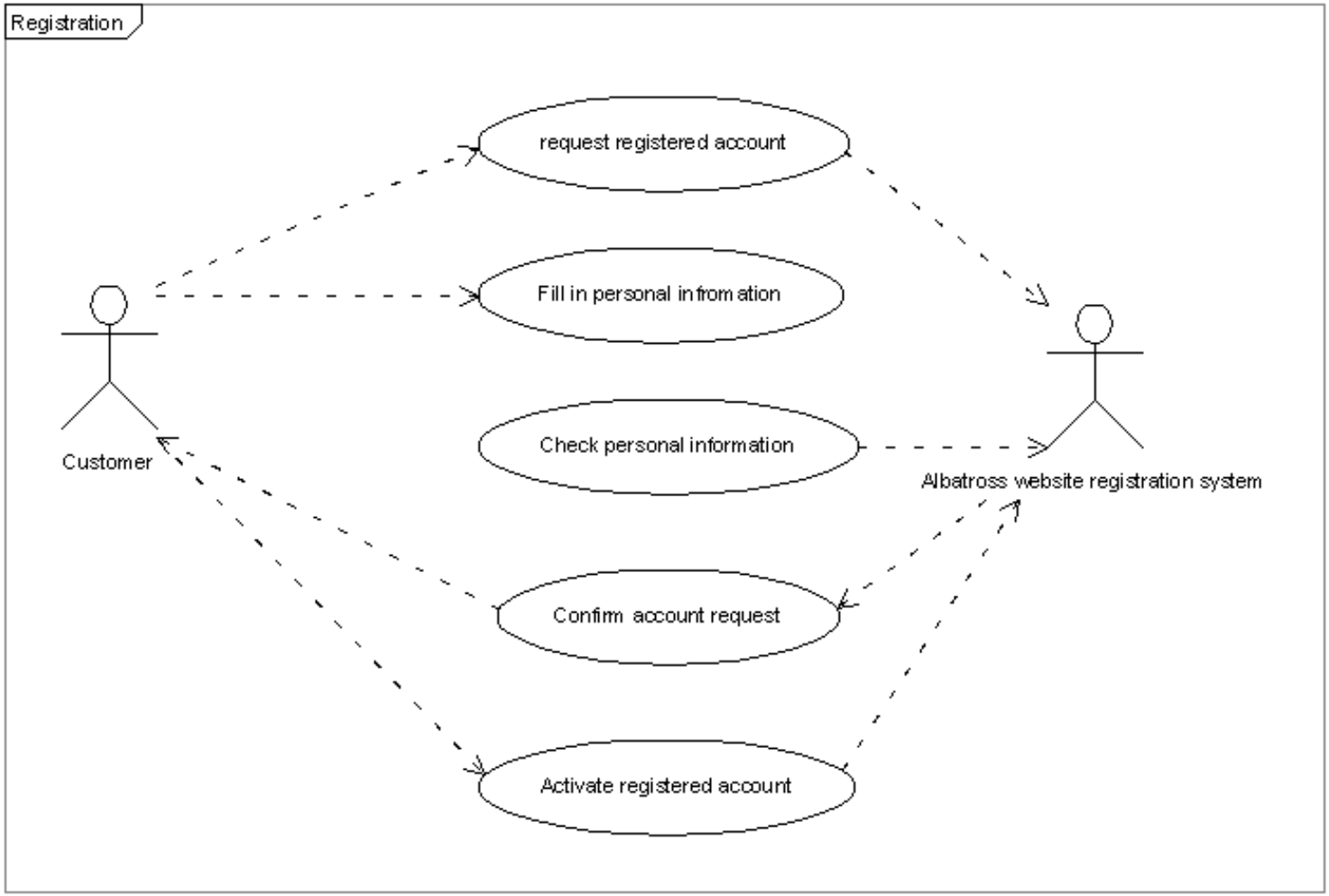


Mis-Use Cases

- **Associations in use case diagrams connect actors to use cases**
  - **never actors to actors, or use cases to use cases**
- **Most importantly, do *not* show a functional decomposition of the system**
  - **the following diagram is *not* a valid use case diagram**



Use Cases are Not a Flowchart

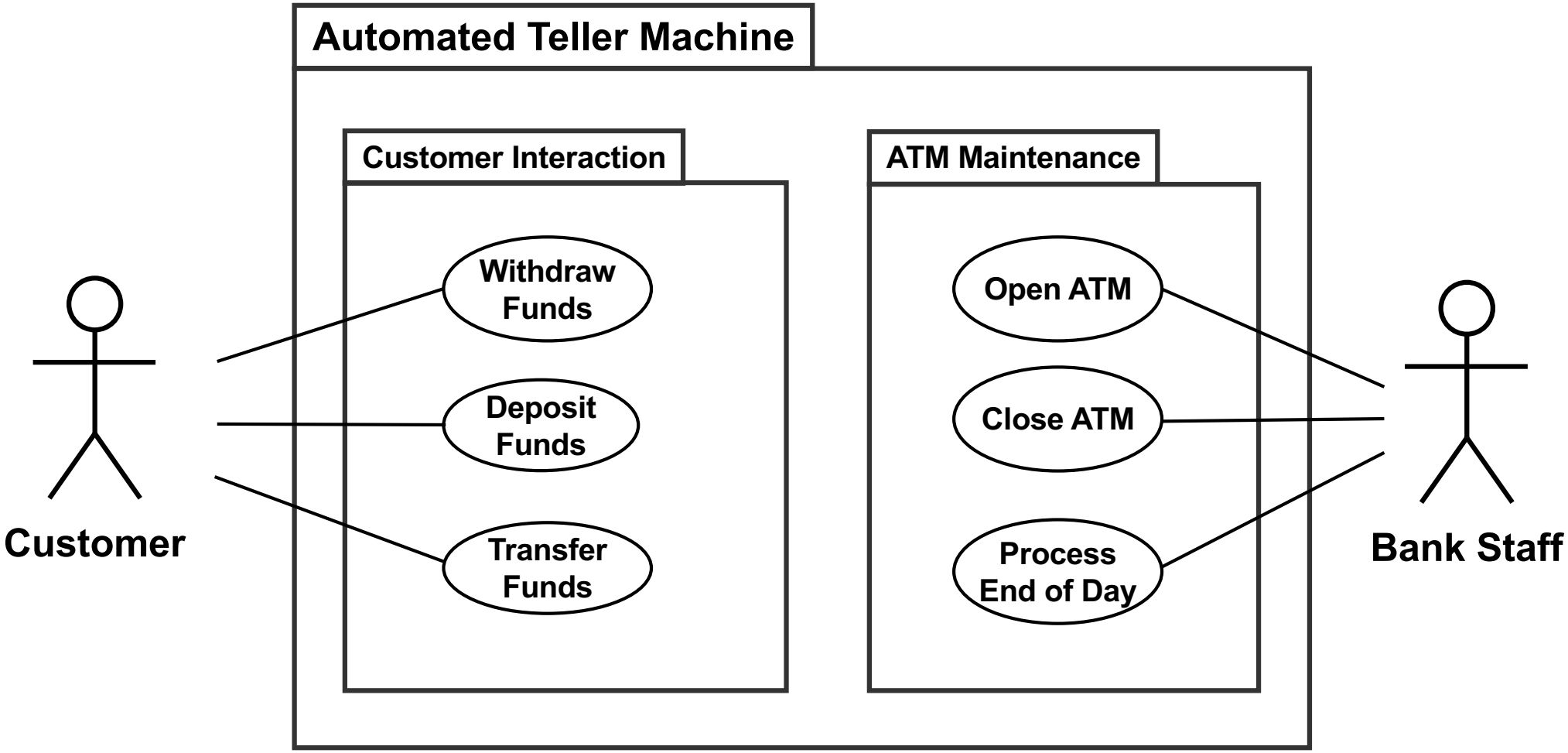


- *"A sequence of transactions offered by the system that yield a useful result for an actor." [Jacobson et al, 1992]*

Task 4: Identify Packages

- **Group use cases into packages that represent features / subject areas / themes**
  - **logical aspect of the problem space where a group of use cases work towards a common goal**

Packages – UML Notation



Task 5: Develop Initial Use Cases

- **Workshop in conjunction with client representatives**
  - **– Requirements Modeller acts as facilitator**
  - **– Prototype Developer may be observer**
- **Initially keep use cases simple and abstract**
  - **– describe basic idea of what needs to be done**
  - **– detailed use cases will be hard to change at this stage**
- **Use cases may vary considerably at this stage as ideas are brainstormed**

Breadth-First Development

- **We are too eager for detail**
- **It's a process of discovery**
- **Early detail leads to wasted effort**
  - **requirements "churn"**
  - **prolonged requirements phase**
  - **loss of patience**

Prioritise Use Cases

- **Complete listing of use cases should be prioritised based on organisational needs**
- **Allows important features to be delivered first**
  - **– system roll-out driven by business value (ROI)**

Identify Typical and Alternative Scenarios

- **Typical Scenario – normal sequence of events**
  - **– what is intended to happen**
- **Alternative Scenarios – abnormal courses of operation**
  - **– represent errors, or conditions not often encountered**
    - **» only conditions system can detect**
    - **» combine conditions with the same result**
- **Focus on understanding the typical scenario first**
  - **– initially just list the alternatives**

Documenting Use Cases Guidelines

- **Initially just specify the events of the use case as a series of steps**
- **During refinement ensure that a statement of "why" a step is being executed is included**
  - **easy for domain experts to forget to document why**
    - » **because it seems obvious**
  - **remember the model will be read by people unfamiliar with the domain**

Documenting Use Cases (cont.)

- **Number the steps**
- **If steps are completed within the context of another step, indent the text and the numbers**

1. The First Step

1.1 This step must be completed to accomplish step 1

1.2 This is the next step to accomplish step 1

2. The Second Step

Documenting Use Cases (cont.)

- **When documenting the typical scenario, identify any alternative scenarios**
  - **they do not have to be fully documented until the final release point**
- **Document the steps of a use case sequentially**
  - **step the actor performs**
  - **followed by step the system does in response**
  - **next step the actor performs**
  - **…**
- **Show what the actor *does* or *gets***

Use Case Template
– see Req Spec Template

Description Detail

- **Subject to some debate**
- **Previous example and template are based on what I consider best practice**
  - **particularly when there is limited interaction between developers and user representatives**
  - **based on well-known published methods**
    - » **e.g. RUP, Thomsen-Dué, Cockburn**
- **Provide enough detail so the Business Owner can verify it performs the actions they require**
- **Functionality of the system needs to be described in enough detail to allow object decomposition**

Use Cases and the User Interface

- **Keep use cases general**
  - **use cases should describe what the actor logically wants to achieve**
    - » ***not*** **the means by which they achieve those goals**
- **UI/UX design is a separate activity**
  - **after use cases have been validated**
- **See Larry Constantine's *Essential Use Cases***
  - **refines use cases down to the core essentials and user intentions, nothing at all related to even the style of UI**
  - **UX design is a completely separate (*and large*) step**

Task 6: Refine the Typical and Alternative Scenarios

- **Once the typical and alternative scenarios have been identified and agreed upon they need to be refined**
- **Prototyping may be used during the refinement process to model interaction with the system**
- **Refining use cases involves adding detail**
  - **– may be done through workshops and/or prototyping**
    - **» depending upon the complexity of the use case**

Task 6: Refine the Typical and Alternative Scenarios (cont.)

- **Requires attention to detail**
  - **good understanding of problem domain and system goal**
- **Use cases can be developed in parallel**
  - **different primary actors by different teams**
- **Use cases can be added to each other to create more complex use cases**

Example Refined Use Case

- **Withdraw Funds – first iteration**

1. Customer identifies themselves to ATM	2.1 ATM successfully identifies Customer 2.2 ATM requests Customer perform transaction
3. Customer withdraws funds from account	4.1 ATM verifies transaction 4.2 ATM provides customer with requested funds and receipt

- **Alternative Scenarios**

1. Customer attempts to identify themselves to ATM	2.1 ATM can't identify Customer 2.2 Display error message

3. Customer attempts to withdraw funds from account	4.1 Funds not available 4.2 Display error message

Withdraw Funds Use Case

Task 8: Restructure Use Cases

- **Restructure use cases to maximise reuse and minimise redundancy**
- **Use cases can be restructured in three ways**
  - **«include» relationship**
  - **«extend» relationship**
  - **generalisation**
- **Functional decomposition**
  - **identify repeatedly used or specialised portions and factor them out**

«include» Relationship

- **Factor out common behaviour in use cases**
  - **– sequence of steps appearing in *multiple* use cases**
    - **» needs to be more than *one or two* steps**
  - **– avoids inconsistency in requirements**
- **Scenario *always* uses the included steps**
  - **– included steps do not have to be a complete use case**
    - **» i.e. initiated by an actor**
- **Too much factoring**
  - **– behaviour is scattered across multiple use cases**

Refactoring Example

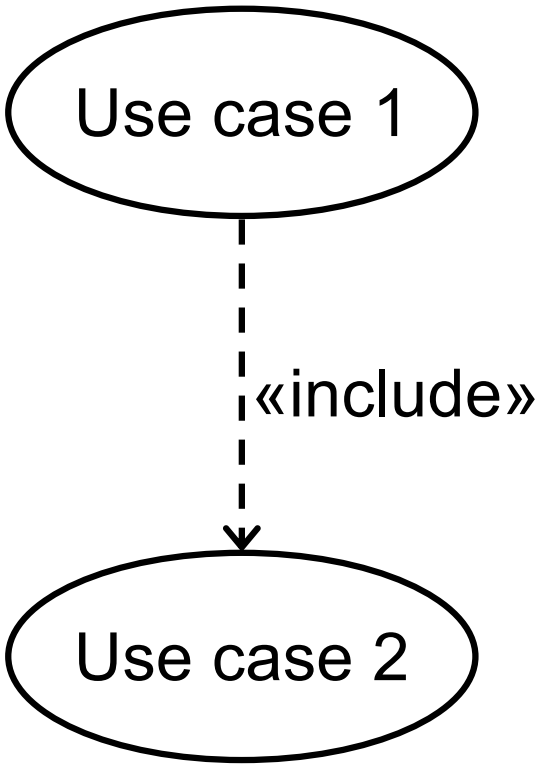
- **Withdraw, Deposit, Transfer – all same initial steps**

1. Customer inserts their card	2. ATM reads card data and prompts for PIN
3. Customer enters their PIN	4. ATM sends card number and PIN to Bank System to confirm customer identity
5. Bank System confirms card number and PIN are a valid match	6. ATM requests customer to select a transaction

- **Create Identify Customer use case**
  - **included by Withdraw, Deposit, Transfer, …**

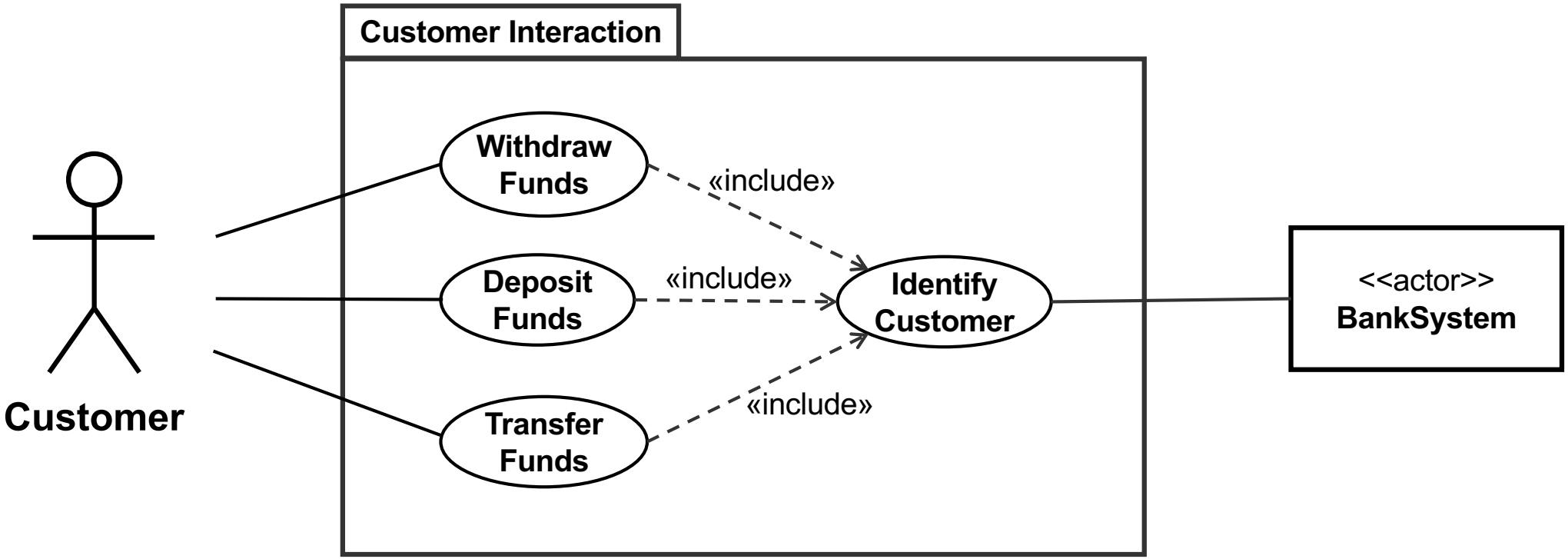
Illustrating the «include» Relationship

- **In original use case descriptions, the phrase «include» <*use case name*> is inserted where the included use case begins**
  - **steps are removed from the original use cases**
- **«include» relationship is shown in the use case diagram**
  - **Use case 1 «include» Use case 2**



UML Notation

Example Diagram



Example Scenario Text

1. «include» *Identify Customer*	
3. Customer selects withdraw	4. ATM requests customer to select an account
5. Customer selects their account	6. ATM requests customer to enter amount to withdraw
7. Customer enters amount	8. ATM sends transaction details to Bank System for processing
9. Bank System confirms transaction is valid	10. ATM 10.1 ejects the card 10.2 prints receipt 10.3 provides Customer with requested amount of money

«extend» Relationship

- **Factors out optional behaviour in use cases**
  - **– when a scenario produces a different result in certain situations**
    - **» alternative condition affects several steps**
      - **> 2 or 3**
    - **» alternative flow becomes complex**
      - **> 3 or 4 steps**
- **Understanding complete behaviour requires reading more than one use case**

«extend» Relationship

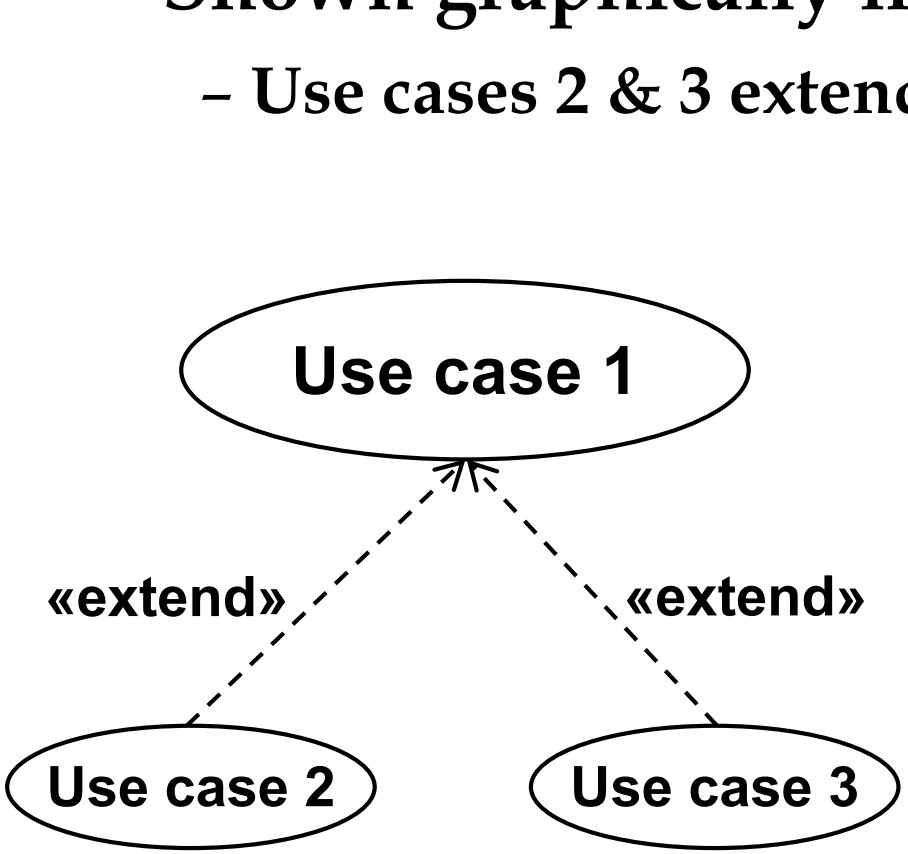
- **Extended scenario may be completed *without* including the steps from the extending use case**
  - **removes need to have multiple typical scenarios to capture the optional paths of a use case**
- **Delivery of extending use cases can occur in later phases of development**

Extension Point

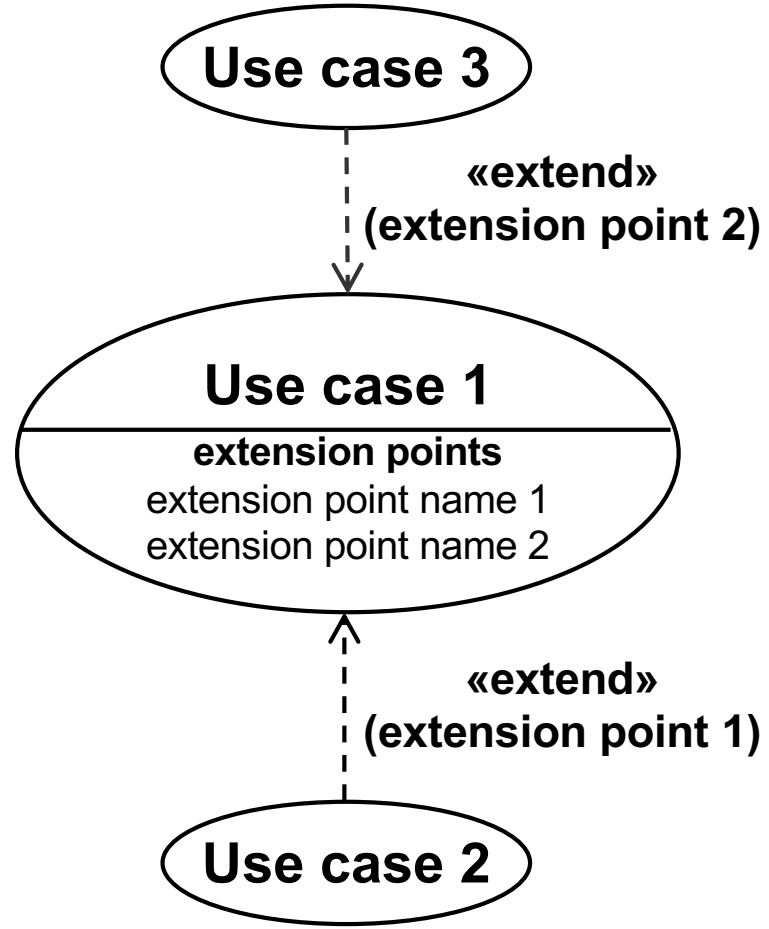
- **In the extended use case description include the phrase (*extension point: <extending use case>*)**
  - **– indicating point at which the extending use case begins**
    - **» condition that causes the extension is highlighted**
- **Scenario for the base use case runs as normal until the extension point**
- **Under certain conditions the extending use case then begins and runs to completion**
- **Base use case then resumes**

Illustrating the «extend» Relationship

- **Shown graphically in two ways**
  - **Use cases 2 & 3 extend Use case 1**

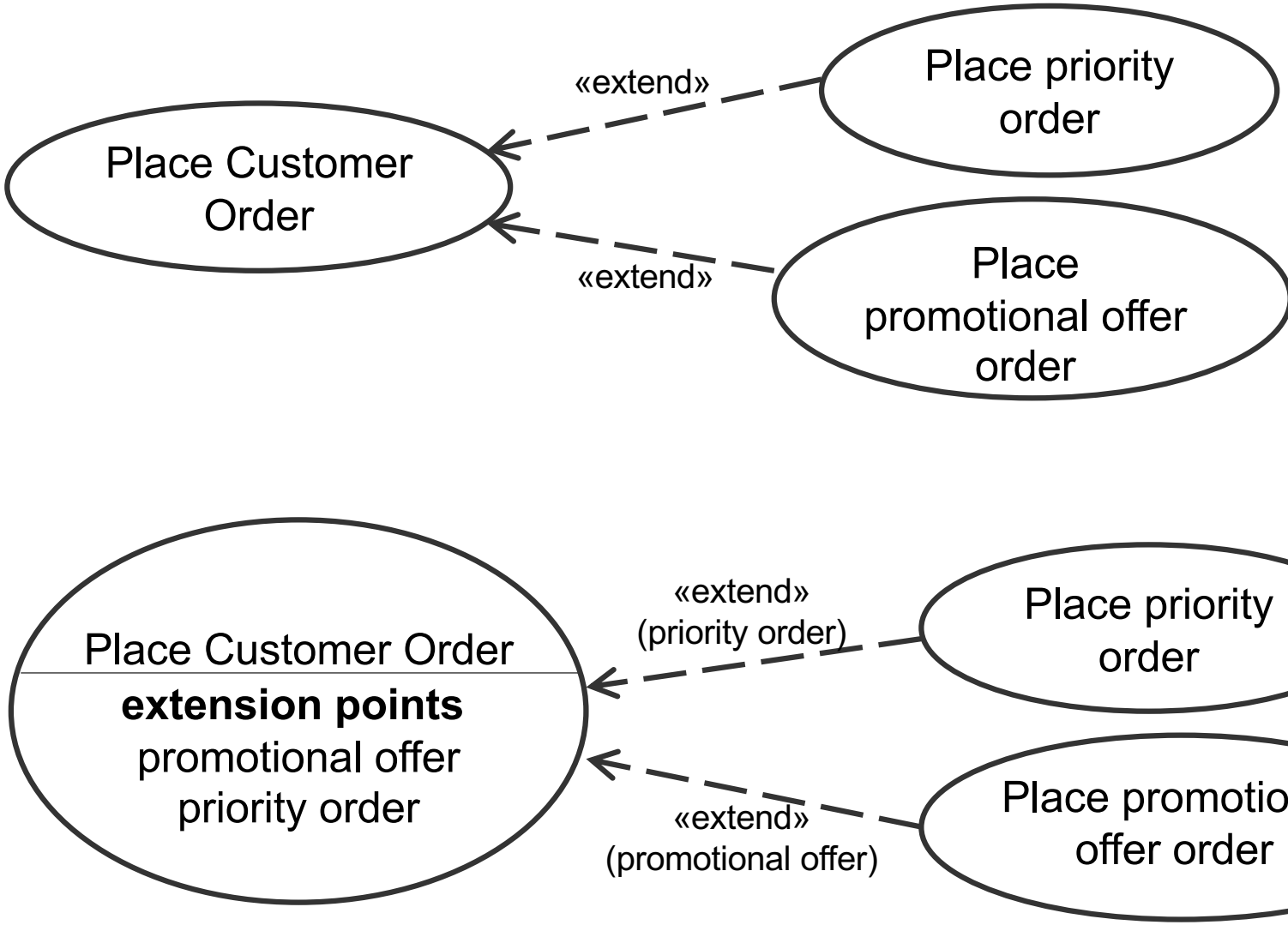


OR



UML Notation

Examples


«extend»
Place priority order
Place Customer Order
«extend»
Place promotional offer order
Place Customer Order
extension points
promotional offer
priority order
«extend»
(priority order)
Place priority order
«extend»
(promotional offer)
Place promotional offer order

«extend» Relationship in the Scenario Text

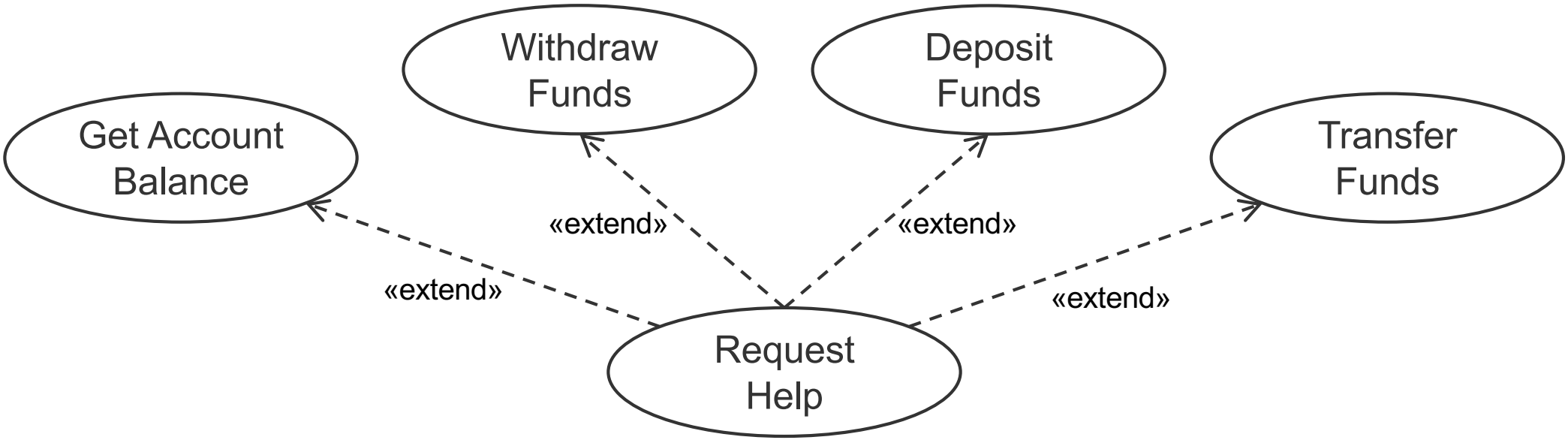
Use Case Title: Place Customer Order
Ref. Number: UC005
Summary: A salesperson selects to place a customer order
Primary Actor: Salesperson
Inherits: None
Includes: None
Extension Points: Promotional Offer, Priority Order

Typical Sequence of Events:
1. The Salesperson selects to place an order
3. The Salesperson completes the order form by:
    3.1 entering the Customer details
    3.2 entering the products and quantities ordered, **if any products qualify for special offer (extension point: promotional offer)**
    3.3 entering the delivery method, if priority order is selected **(extension point: priority order)**
5. … … …

Use Case Title: Promotional Offer
Ref. Number: UC008
Summary: If the item is part of a promotional offer, then calculated discount for the item
Primary Actor: Salesperson
Inherits: None
Includes: None
Extension Points: None

Typical Sequence of Events:
2.1 System calculates the discount applicable to the item
2.2 System displays the discount applied on the order form
2.3 Discount is applied to the order

ATM Example

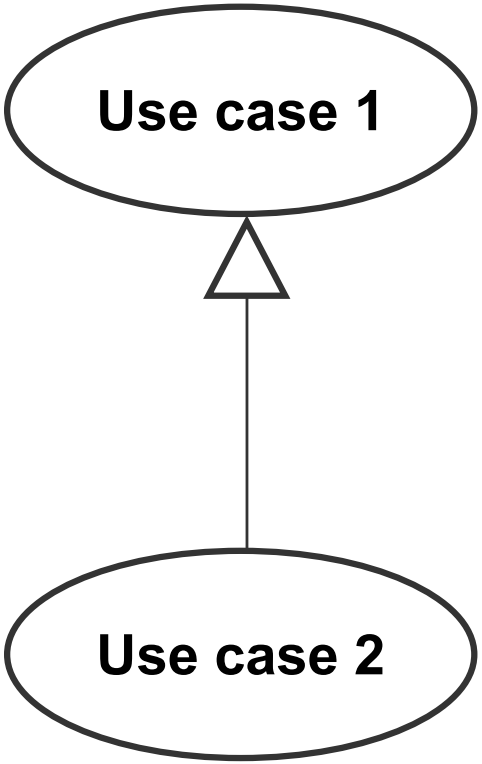

Withdraw Funds
Deposit Funds
Get Account Balance
Transfer Funds
«extend»
«extend»
«extend»
«extend»
Request Help

Use Case Generalisation

- **General behaviour can be factored out using generalisation**
  - **– like inheritance in class design**
- **Use when the behaviour of the child use case is more specific than that described by the parent use case**
  - **– general – specialised relationship**

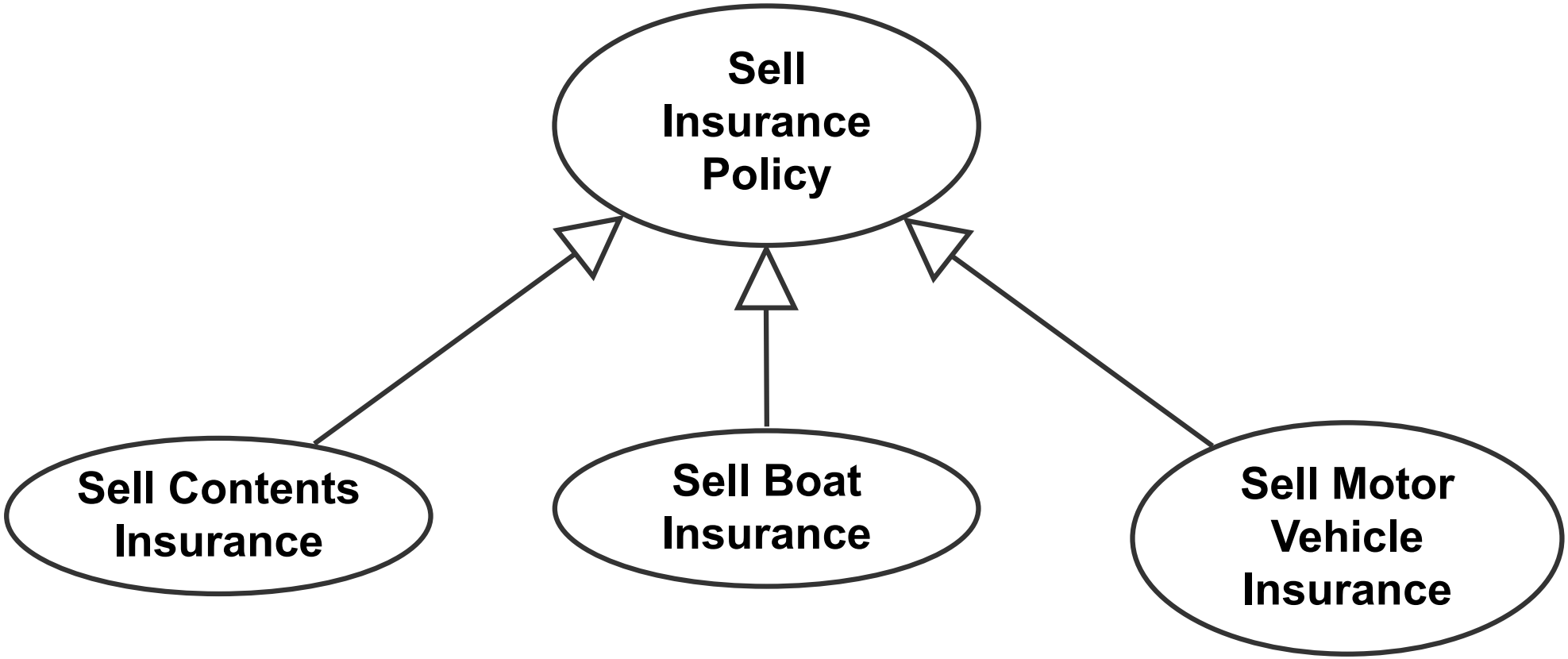
Illustrating Use Case Generalisation

- **In the specialised use case description, include the phrase *inherits: <parent use case>***
- **Scenario description of the child indicates the behaviour of the parent that is replaced or refined**
  - **– *refine* <step number in parent use case>**
  - **– *replace* <step number in parent use case>**
- **Generalisation relationship can be shown in the use case diagram**
  - **– Use case 2 inherits Use case 1**



UML Notation

Example


Sell Insurance Policy
Sell Contents Insurance
Sell Boat Insurance
Sell Motor Vehicle Insurance

Generalisation in the Scenario Text

Use Case Title: Sell Insurance Policy
Ref. Number: UC025
Summary: A Customer Service Officer sells a new insurance policy
Primary Actor: CSO
Inherits: None
Extension Points: None
Includes: None
Typical Sequence of Events:

1. CSO selects to issue a new policy
3. CSO completes the policy form by
   - 3.1 entering the Customer details
   - 3.2 entering the Address details
5. CSO selects the type of insurance policy to create
7. CSO enters the specific information required for the type of policy (refer to child use cases for specifics)
8. System calculates the premium payable

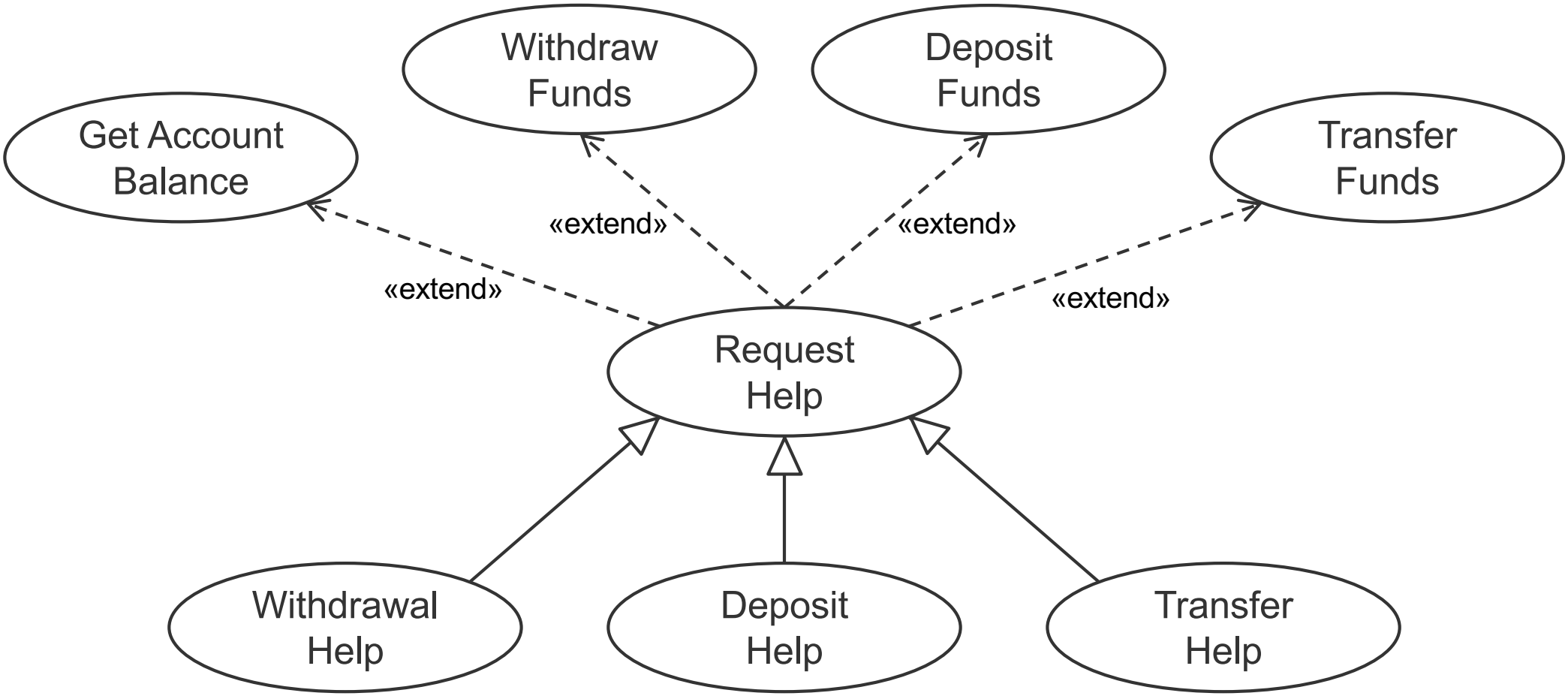
… … …

Use Case Title: Sell Contents Insurance
Ref. Number: UC026
Summary: A Customer Service officer sells a new Contents Insurance Policy
Primary Actor: CSO
Inherits: Sell Insurance Policy
Extension Points: None
Includes: None
Typical Sequence of Events:
Refine Step 7 in *Sell Insurance Policy*

7.1 CSO enters the details of the contents to be covered by the new Policy
7.2 CSO enters any specified items to be covered by the new Policy
7.3 CSO enters any items to be excluded from the Policy

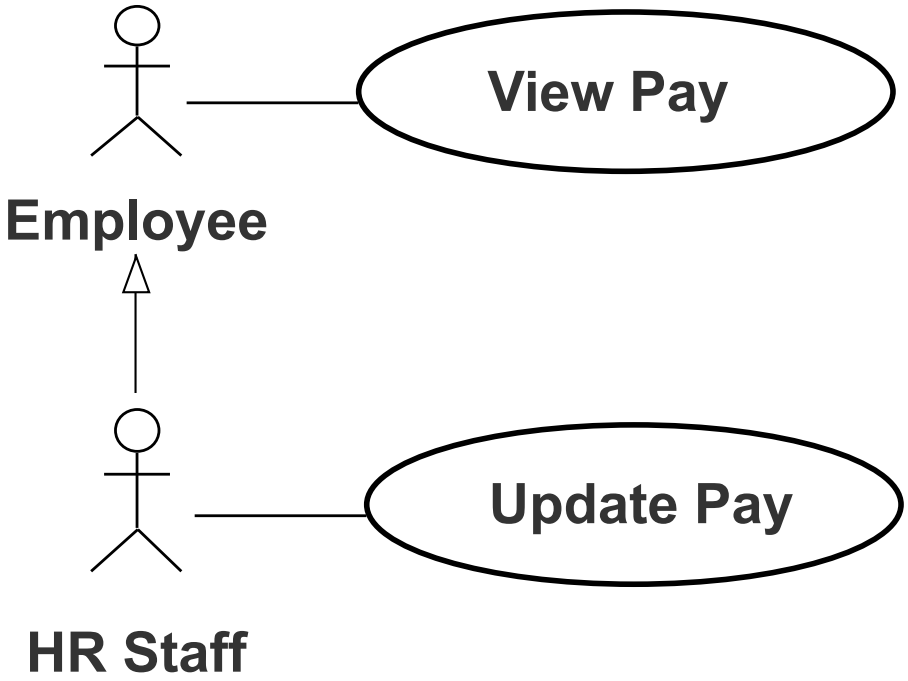
… …

ATM Example


Withdraw Funds
Deposit Funds
Get Account Balance
Transfer Funds
«extend»
«extend»
«extend»
«extend»
Request Help
Withdrawal Help
Deposit Help
Transfer Help

General (Abstract) Actors

- **Actors that share use cases could be abstracted such that a new general actor is created**
  - **don't go overboard**
- **Ensure the general and specialised actors *both* represent useful concepts or users**



Summary

1. **Start with actors and goals** 2. **Work breadth-first**	*Use Case Model*
3. **Get the scope right** 4. **Include and extend judiciously**	*Use Cases*
5. **Use alternatives effectively** 6. **Extract business rules and requirements** 7. **Show what the actor does or gets**	*Use Case Steps*

Use Cases vs. User Stories

- **Both are user-centred**
  - **what is to be accomplished**
- **Difference is in granularity**
- **User Stories**
  - **statement of user needs**
    - » **details discovered during development**
      - **fleshed out by acceptance tests**
- **Use Cases**
  - **description of interaction with system**
    - » **allows**
      - **analysis**
      - **verification**
      - **validation**
      - **more detailed planning**

Use Cases Limitations

- **Interaction focus**
  - **usage scenarios**
- **Not suitable**
  - **batch processing**
    - » **e.g. interest calculation**
  - **complex business rules**
    - » **e.g. airfare calculation**
  - **computationally intensive**
    - » **e.g. environmental monitoring & analysis**
      - **http://www.oceannetworks.ca/**
  - **real-time systems**
    - » **e.g. railway signalling**
  - **embedded systems**
    - » **e.g. air condition control for a building**

Use Case Alternatives

- **Process or target domain dependent**
- **UX Focus or COTS**
  - **Archetypes – behavioural perspective**
  - **Personas and Storylines**
- **Agile or Lean**
  - **User Stories – informal**
  - **Behaviour Driven Development (BDD)**
    - » **Domain-Specific Language (DSL) for formalism**
- **Real-Time or Embedded**
  - **Event – Response Tables**

Event – Response Table

Event	System State	Response
Road Sensor detects vehicle entering left-turn lane.	Left-turn signal is red. Cross-traffic signal is green.	Start green to amber countdown timer for cross-traffic signal.
Green to amber count-down timer reaches zero.	Cross-traffic signal is green.	1. Turn cross-traffic signal amber. 2. Start amber-to-red countdown timer.
Amber-to-red countdown timer reaches zero.	Cross-traffic signal is amber.	1. Turn cross-traffic signal red. 2. Wait 1 second. 3. Turn left-turn signal green. 4. Start left-turn-signal countdown timer.

Reading

- **BlackBoard**
  - **Week 6**
- **Wiegers**
  - **Chapter 8**
- **Larman**
  - **Chapters 6 & 7**
- **Sommerville**
  - **Sections 5.1 & 5.2**

Further Reading

- **Maciaszek, Leszek A. *Requirements Analysis and System Design: Developing Information Systems with UML.***
  - **Chapter 2: " Underpinnings of Requirements Analysis"**
  - **Chapter 4: "Requirements Specification"**

Constantine, L., & L. Lockwood. *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design.* Addison-Wesley, 1999.

- **Essential Use Cases**

- **Cockburn, Alistair. *Writing Effective User Cases*.**
  - **pre-publication draft: https://alistair.cockburn.us/get/2465**
    - » **maybe – site is being updated – slowly!**
    - » **https://www.academia.edu/32227372/Alistair_Cockburn_Writing_Effective_Use_Cases**

Next Steps

- **Tutorial**
  - **Presentation preparation**