Grand Valley State University
www.gvsu.edu

Software Engineering Requirements Use-Cases

Erik Fredericks // frederer@gvsu.edu

Adapted from materials provided by Byron DeVries, Jagadeesh Nandigam

Outline

What is a use-case?

Use-case diagrams

Capturing the use-case

First of all, what is a use-case?

A use-case captures some user visible function

This may be a large or small function
- Depends on the level of detail in your modeling

A use-case achieves a discrete goal for the user

Examples:
- Format a document
- Request an elevator

How are the use-cases found (e.g., captured or elicited)?

How do we capture use cases from users or requirements?

User Goals vs User Interactions

Consider the following when formatting a document

- Define a style
- Change a style
- Copy a style from one document to the next

Compared to:

- Format a document
- Ensure consistent formatting of two documents

The **latter** is a **user goal**: Something the user wants to achieve

The **former** are **user interactions**: Something the user does to the system to achieve the goal.

Goals and Interactions

There is a place for both goals and interactions

Understand what the system shall do:

- Capture the user goals

Understand how the user will achieve these goals:

- Capture the user interactions, and
- Sequences of user interactions

Therefore:

- Start with the user goals and then **refine** the user goals into several (many) user interactions

What comprises a use-case diagram?

Use-Case Diagrams include:

Actors: A role (not necessarily a person), with three essential traits:
- **External** to the system
- **Interactions** with the system
- Tries to achieve a **goal**

Use-Cases: One way to use the system:
- Centered around a **goal**
- Yields a **result** to an actor

Connections: Between use-cases and actors (and sometimes use-cases)

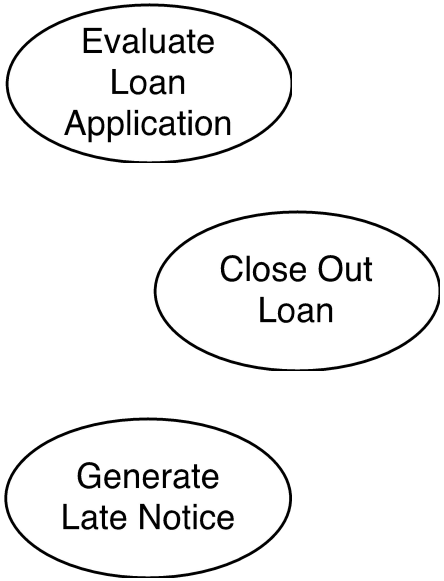
Identifying use-cases

A use case is an event:

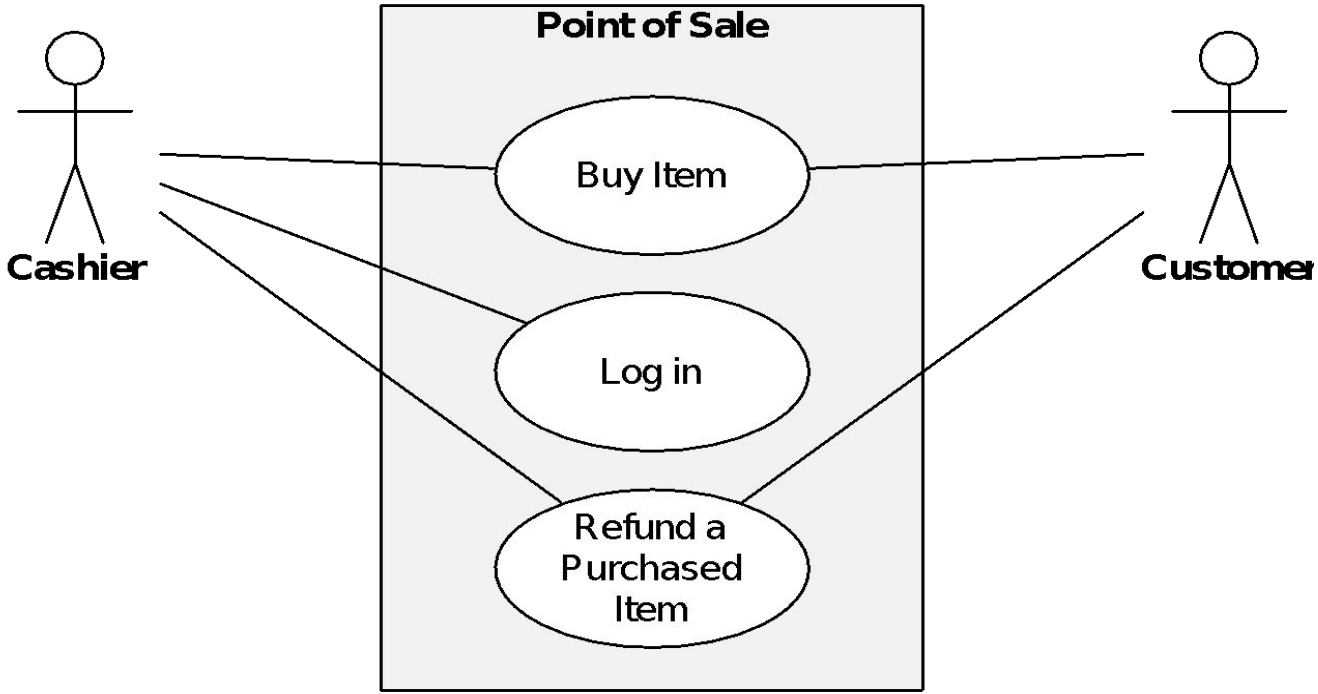
- Not an information flow
- Not a process flow

A use case should:

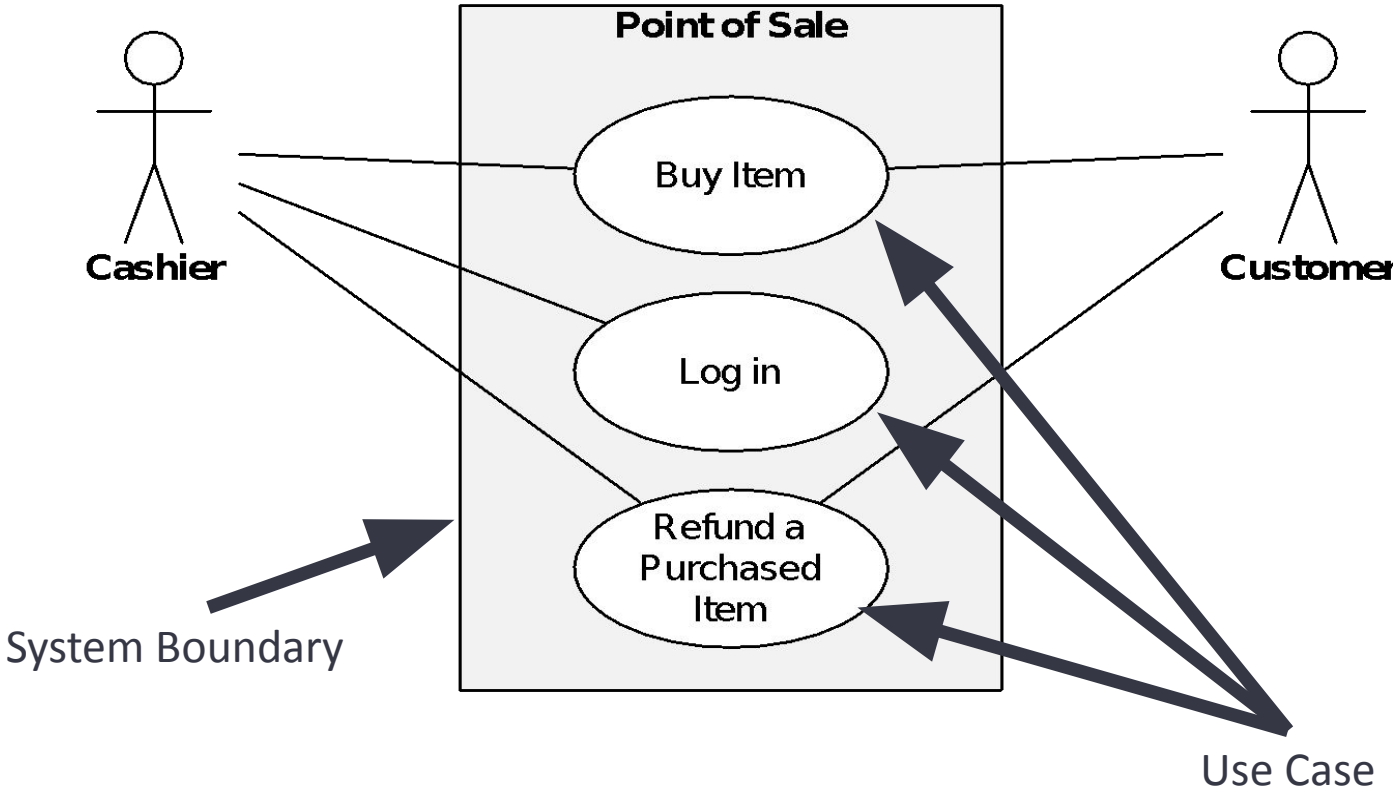
- Use a short, **verb-noun** combination
- Describe the **interaction**
- Reflect the actor's **goal**



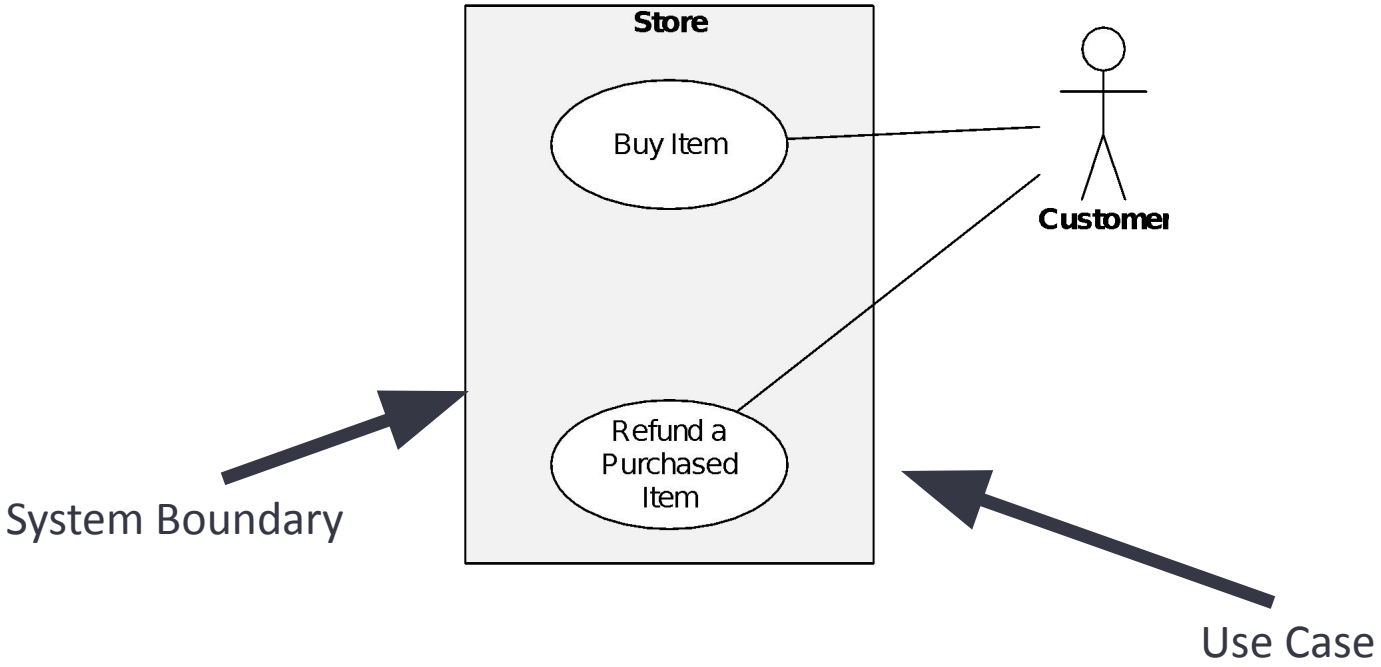
Use-Case Diagram: Point of Sale (PoS)

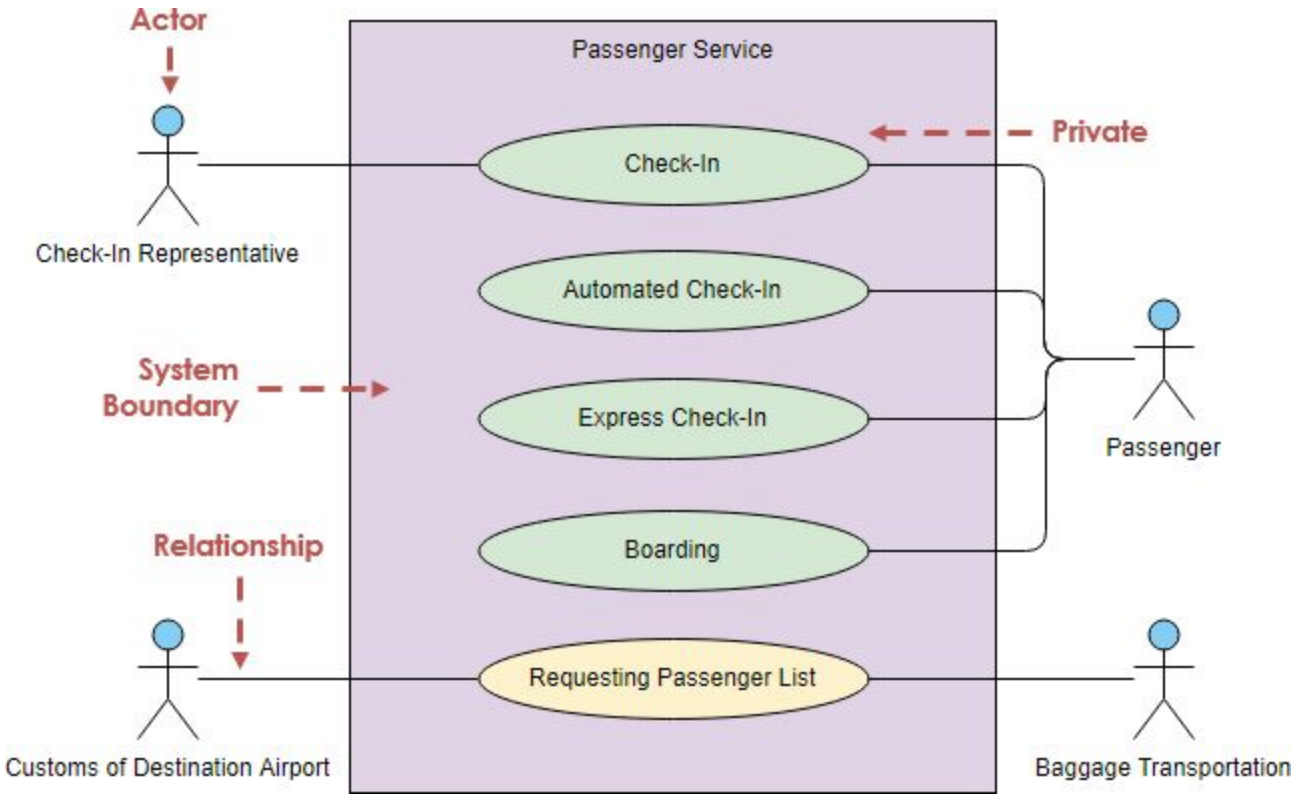


Setting the system boundary



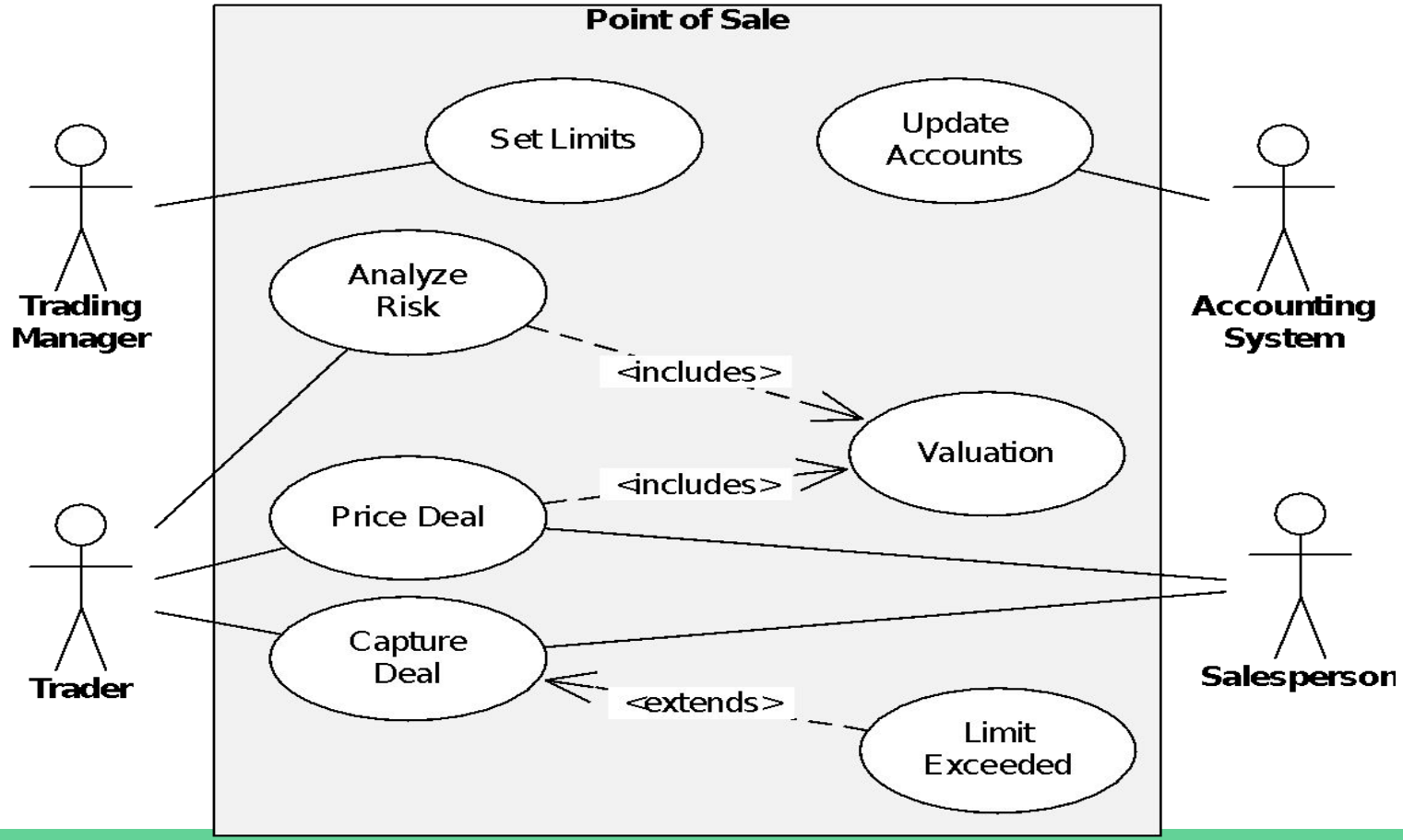
Setting the system boundary





https://medium.com/@warren2lynch/all-you-need-to-know-about-use-case-modeling-828756da3215

Use-Case Diagram: Financial Trading System



Includes and Extends

Includes

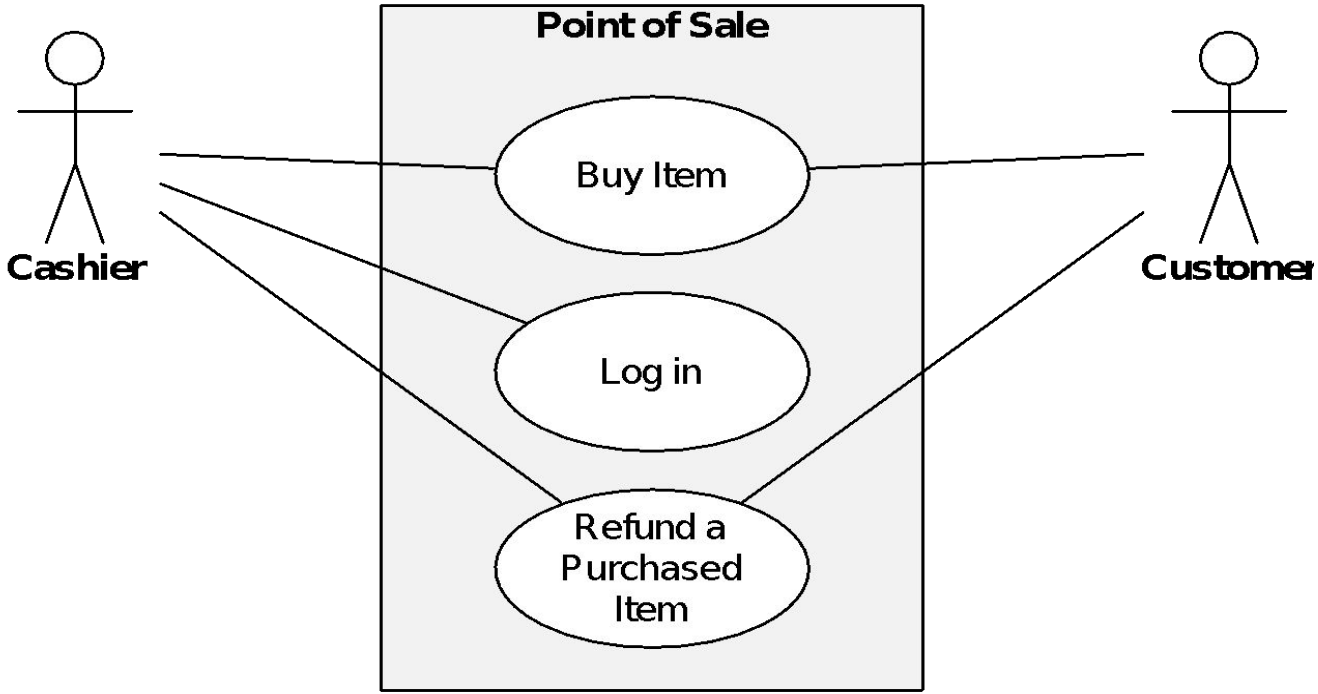
- You have a piece of behavior that is similar across many use cases
- Break this out as a separate use-case and let the other ones “includes” it
- Sometimes called “uses”

- Examples include:
  - Valuation
  - Validate user interaction
  - Sanity check on sensor inputs
  - Check for proper authorization

Extends

- A use-case is similar to another but does a little bit more
- Put the normal behavior in the use-case and the exceptional behavior somewhere else
  - Capture the normal behavior
  - Try to figure out what can go wrong in each step
  - Capture the exceptional cases in separate use cases
- Significantly improves understanding

Use-Case Diagram: Point of Sale (PoS)



PoS use-case description

Use Case: **Buy Item**

Actors: Customer (initiator), Cashier

Description: The Customer arrives at the checkout with items to purchase. The Cashier records the purchase items and collects a payment. On completion the Customer leaves with the items.

PoS **expanded** use-case description

Use Case: **Buy Item**

Actors: Customer (initiator), Cashier

Description: The Customer arrives at the checkout with items to purchase. The Cashier records the purchase items and collects a payment. On completion the Customer leaves with the items.

Cross Ref.: Requirements XX, YY, and ZZ

Use-Cases: Cashier must have completed the Log In use-case

PoS **expanded** use-case description

Use Case: **Buy Item**

Actors: Customer (initiator), Cashier

Description: (Ordered)

1.) The Customer arrives at the checkout with items to purchase.
2.) The Cashier records the purchase items and collects a payment.
3.) On completion the Customer leaves with the items.

Cross Ref.: Requirements XX, YY, and ZZ

Use-Cases: Cashier must have completed the Log In use-case

When to use use-cases

In short, it is **always good** to use use-cases.

Requirements definition is the toughest part of software development and can have the largest impact when done incorrectly:

- Use-Cases is a powerful tool to understand:
  - Who your users are, including interacting systems
  - What functions the system shall provide
  - How these functions work at a high level

It is important to spend adequate time on requirements and in the elaboration and elicitation phases.

How might you use these in your projects?