# Requirements and Use Cases

## ENGI 5895: Software Design

Andrew Vardy

Faculty of Engineering & Applied Science
Memorial University of Newfoundland

January 24, 2018

# Requirements

**Requirements** describe the capabilities that your system must exhibit and any constraints on its performance.
e.g. You are building a realistic aircraft simulator:

- How should crashes be handled (end simulation or allow some control of a partially damaged craft)?
- Should your simulator model other aircraft? What are the range of interactions (e.g. collisions, prop wash, ...).
- How will the software be packaged (Download or stack of Blu-Rays)?

Such requirements may not be fully appreciated at the start of the project. Also, requirements change. On average 25% of the requirements change [Larman(2005)] on software projects.

# Types of Requirements

The **FURPS+** model of requirements [Larman(2005)]:

- **F**unctional: features, capabilities, security
- **U**sability: human factors, help, documentation
- **R**eliability: frequency of failure, recoverability, predictability
- **P**erformance: response time, throughput, accuracy,...
- **S**upportability: adaptability, maintainability,...
- +: language/tool constraints, packaging, legal, ...

# Requirements Elicitation

The customer's requirements may not be clearly spelled out. They must be **elicited**. The following are techniques for requirements elicitation:

- Direct conversation between developers and customers
- Focus groups with proxy customers
- Demo of working prototype to customer
- Writing use cases...

# Use Cases

Use cases are **text stories** of some **actor** using a system. Consider the following example from ch. 6 of [Larman(2005)] for a Point-Of-Sale (POS) system:

> *Process Sale: A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.*

Here the actor is the customer. Actors are external to the system and could represent users or other software systems.

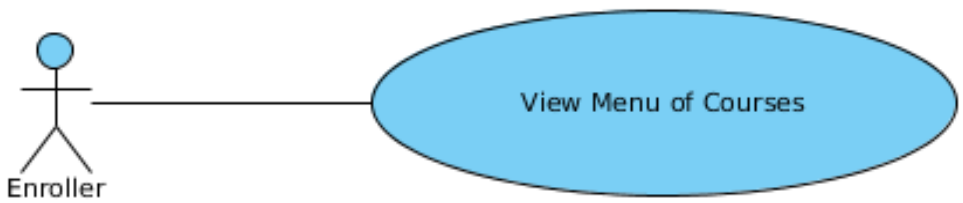
# Example: Course Enrollment System

This example comes from [Martin(2003)]: An educational company needs a system to keep track of courses taught and the students enrolled in those courses. Users should be able to see course offerings, register, and pay for courses. The following are the relavent actors:

- Enroller: Enrolls a student in a course (the student or someone acting on behalf of the student)
- Enrollment Clerk: Receives notifications from the system
- Student: Receives e-mail confirmations of enrollments. Attends courses.

# Use Case #1: View Menu of Courses

*The Enroller requests a list of courses that are currently available in the course catalog. The system displays the list of courses. Included in that list are the name, time, place, and cost of the course. The list will also show the number of students allowed in the course and whether or not the course is currently sold out.*

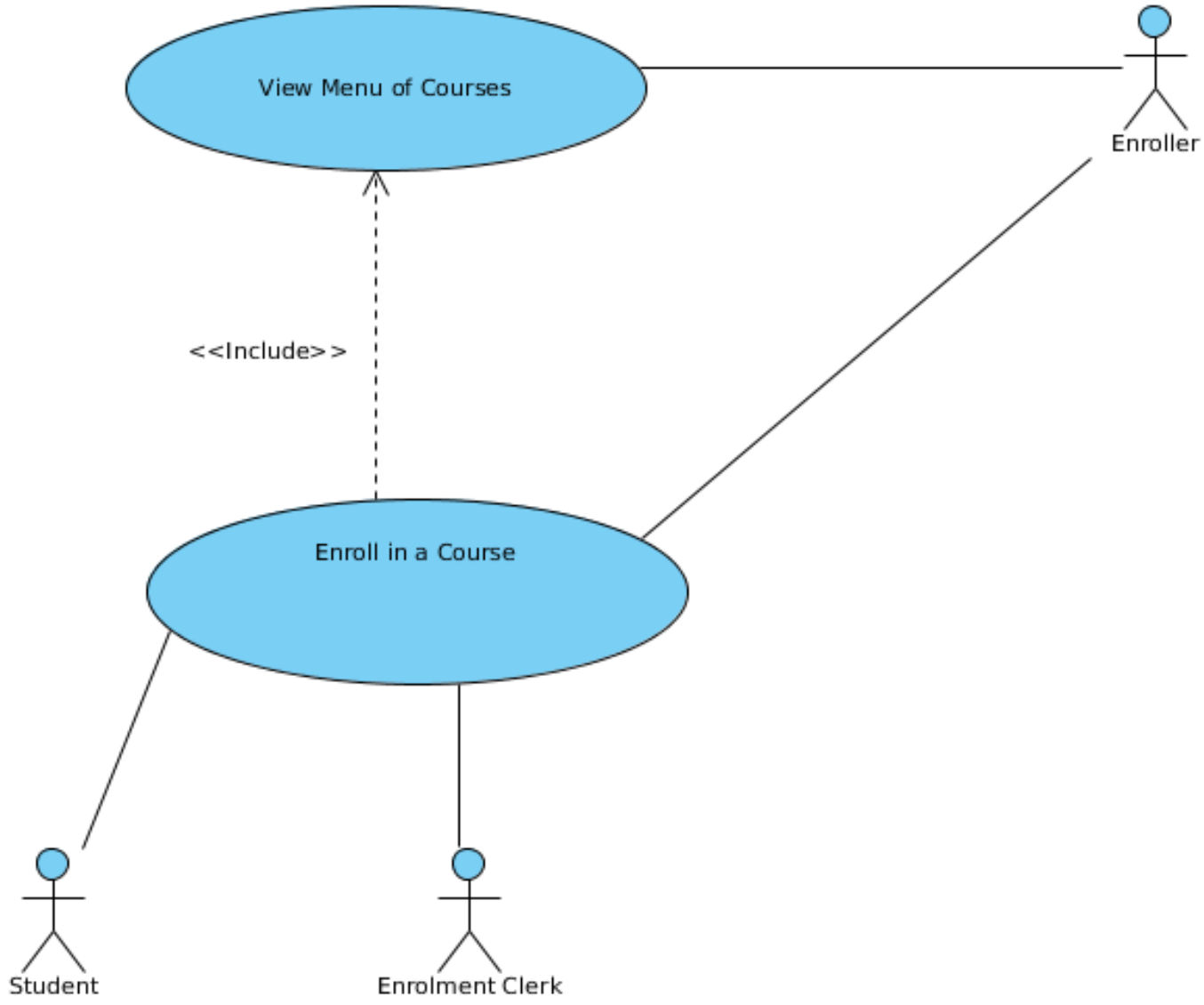



**Notes:**

- A use case includes no user interface or implementation details
- The text of the use case is the main point, the diagram is secondary

# Use Case #2 Enroll in a Course

*The Enroller first views the menu of courses (Use Case #1). The Enroller selects a course for enrollment from the menu. The system prompts the Enroller for the student's information. The system also prompts the Enroller for the preferred payment method (Extension point: fill out payment method). The Enroller submits the form. The Student and Enrollment Clerk are sent emails confirming the enrollment.*

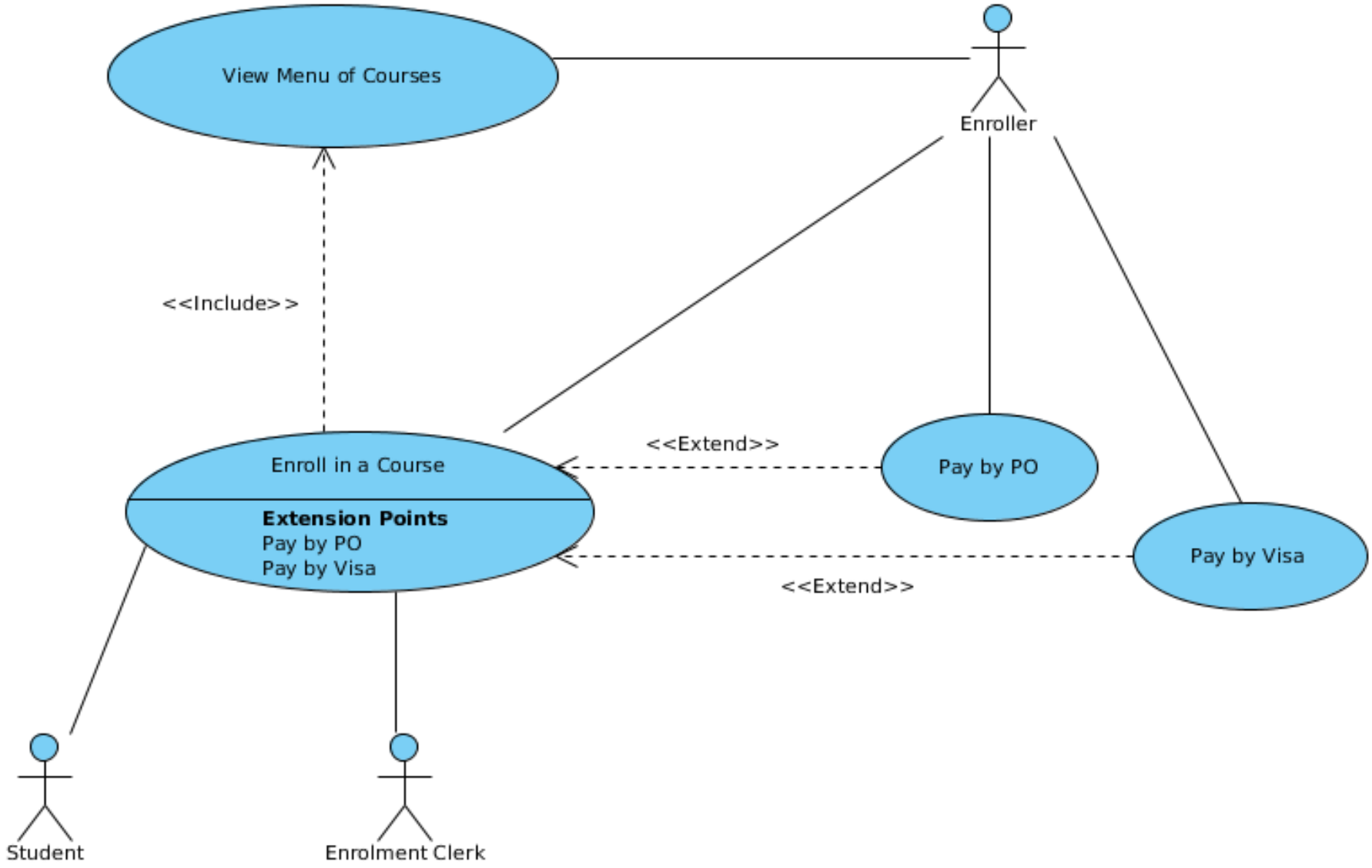



**Note:** To enroll in a course, you must first view the menu of courses. Hence, the <<include>> relationship shown above.

# Extension Points

A use case may have extension points, which are use cases that fill in parts of the interaction with the actor.

> ***Use Case #2.1 Pay by Purchase Order:*** *The Enroller is prompted for the PO#, company name, and contact details for someone in accounts-payable.*
>
> ***Use Case #2.1 Pay by Visa:*** *The Enroller is prompted for the credit card #, expiry date, etc...*


View Menu of Courses
Enroller
<<Include>>
Enroll in a Course
Extension Points
Pay by PO
Pay by Visa
<<Extend>>
Pay by PO
Pay by Visa
<<Extend>>
Student
Enrolment Clerk

# Summary

- Use cases provide a tool for describing a system's functional requirements
- They are stories of a system's usage by different actors and include no implementation details
- The text of the use case is more important than the diagram

# References


Craig Larman.
*Applying UML and Patterns*.
Prentice Hall, third edition, 2005.

Robert C. Martin.
*Agile Software Development: Principles, Patterns, and Practices*.
Prentice Hall, 2003.