University of Westminster

7SENG001 Enterprise Development

Week 1

This semester

- We will learn the C# language and introduce the .NET platform
- We will concrete our knowledge of UML and OOD
- At the end of this module you will:
  - Be proficient in C#
  - Have an understanding of development frameworks.
  - Confident in essential OOD design and concepts
  - Explore enterprise software considerations

Module Outline

- Lecture 1 – the .NET Framework introduction
- Lecture 2 – C# programming language Part 1
  - Working with objects and Object Oriented Design
- Lecture 3 – C# programming language Part 2
- Lecture 4 – Modelling Design and UML
- Lecture 5 – C# programming language Part 3
- Engagement Week
- Lecture 6 – Advanced Features: Collections, custom events, Threads
- Lecture 7 – Persistent data and data handling Part 1 XML
- Lecture 8 – Persistent data and data handling Part 2 Databases
- Lecture 9 – Programming language features
- Lecture 10 – Enterprise considerations
- Lecture 11 – Agile methodologies and TDD

Learning Outcomes

- Proficient in C# programming
- Knowledge of OO patterns and techniques
- Good understanding of the .NET framework
- Know how to create and use databases
- Know when to use alternatives (Java, C, C++ etc)
- Understand the main considerations for enterprise application development

Assessment

- Module is coursework based
- Consists of

- 40% Coursework 1 - due 15th March 2021, 1pm
- 60% Coursework 2 - due 20th April 2021, 1pm

- Assessment 2 will have a peer reviewed report and a viva

Our expectations of you

- You attend **all** lectures and tutorials
  - Lectures are on Blackboard just before lecture
- You do all tutorial exercises
- Submit all coursework on time
- Flag any problems you have early

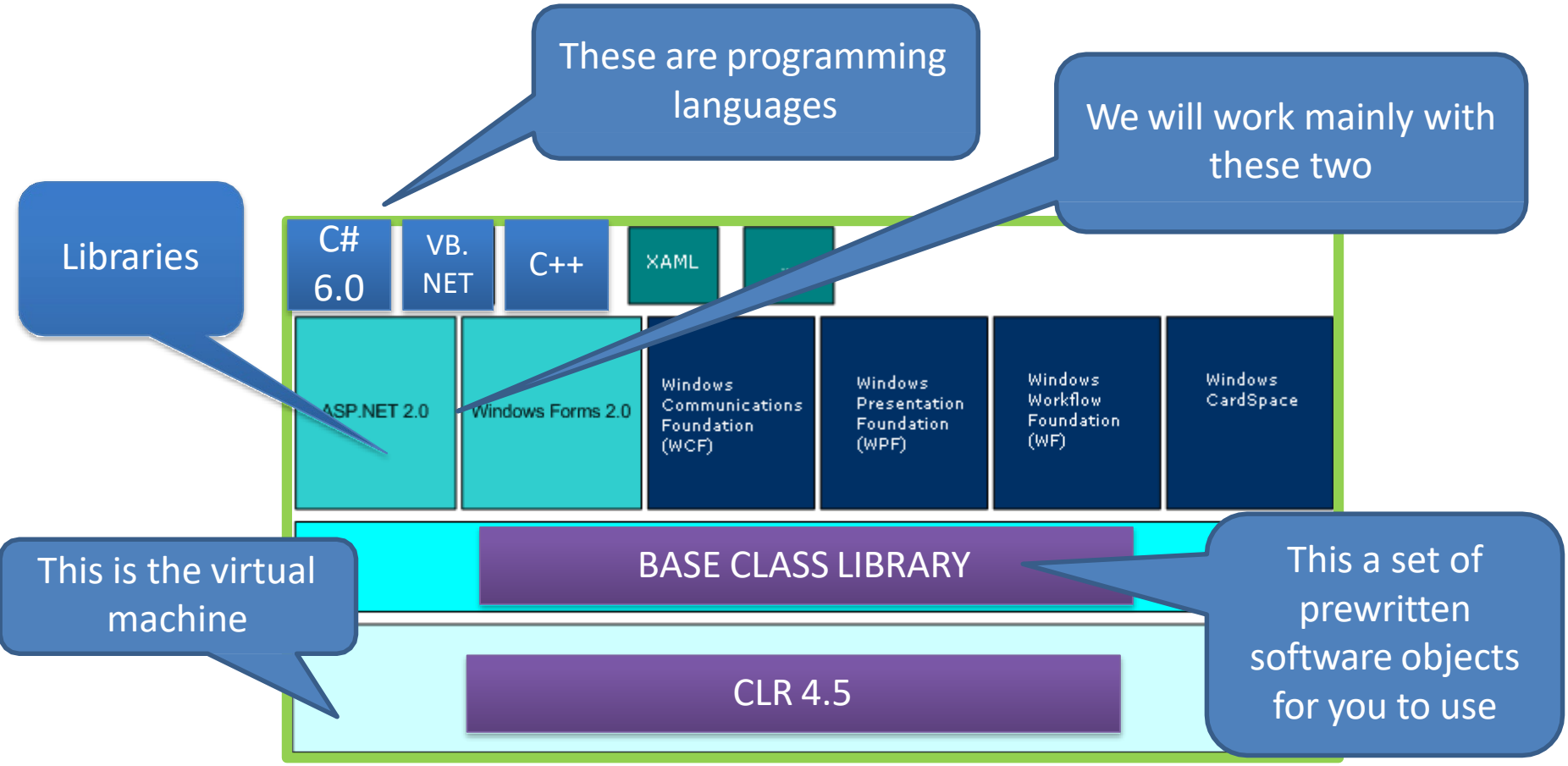
Why learn .NET and C#

- The main purpose of the module is to teach you OOP and OOD development
- Also to teach you the significant OO frameworks for enterprise development
- Like Java, .NET offers a complete architecture for developing complex apps
- It is a popular development framework for software developers and software companies

Is .NET/C# good for everything?

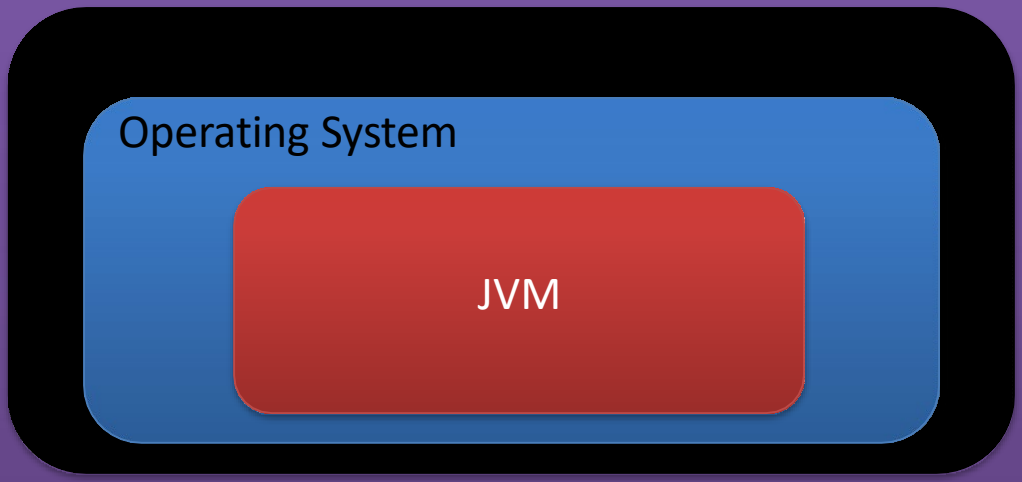
- No, you cannot use it effectively on any other platform that isn't a windows based OS. But this is changing fast.
- No use for Android or Apple iOS for example. (But can with Xamarin)
- No good for native apps but is a good IDE for generic C++. (also changing fast)
- Leads to vendor lock-in and tools are expensive
- Java is a perfectly good alternative with free or much cheaper tools

The .NET Framework



Virtual Machines

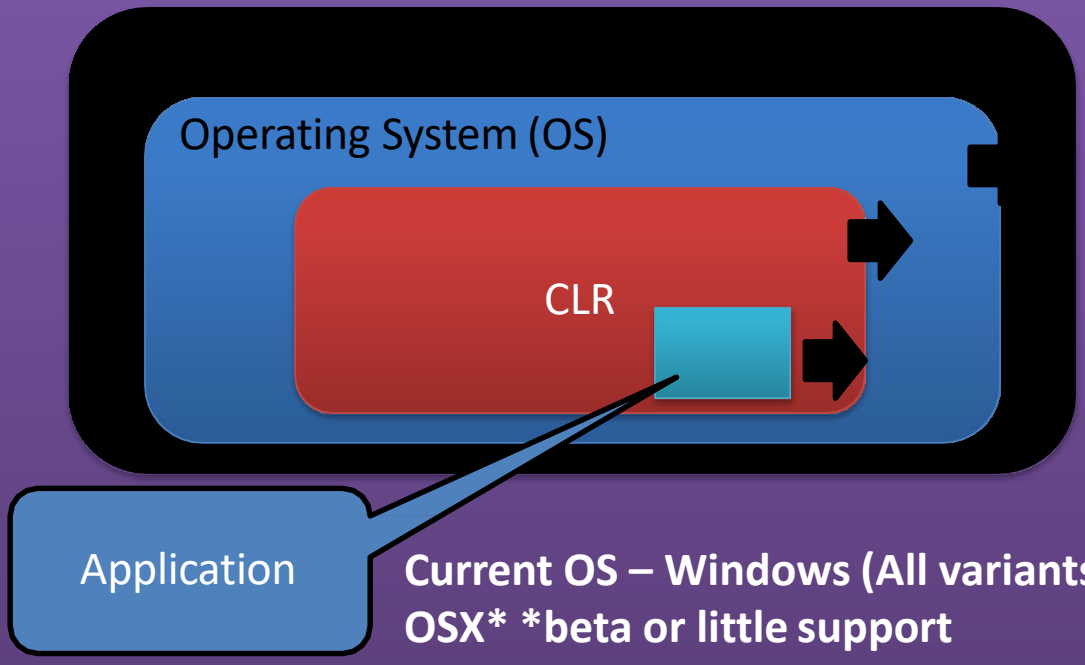
- Common Language Runtime
- **Java Virtual Machine**



Current OS supported – Windows (All variants), Linux , Mac OSX (not iOS) , UNIX, many mobile and embedded devices all fully supported
Also Java for mobile phones and especially Android

Virtual Machines

- **Common Language Runtime**

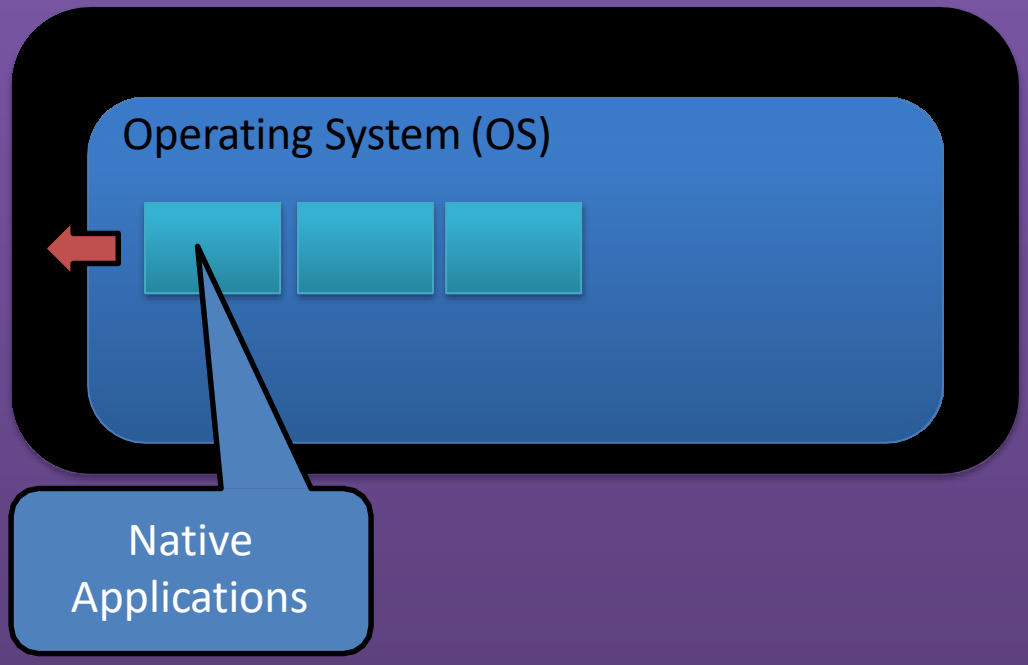


Current OS – Windows (All variants), Linux *, Mac OSX* *beta or little support

.NET is really intended for Windows based OS only so forget other platforms

Native Applications

- Application run directly on the OS with no other intervening layer



What is .NET?

To answer this we need to consider what it is we need in a framework designed for building rich distributed software applications

- Programming language(s)
- Libraries of code to support applications
  - So we don't have to do everything from scratch
    - Network support, Maths functions, Database support etc.
- Compilers/Linkers/Debuggers
- A runtime (Native or Java or CLR virtual machine)
- A development environment such as Visual Studio
  - These may enable visual designs of GUI or even databases
  - May also generate code automatically to save us the trouble, for example when building user interfaces

.NET variants

- Both .NET and JAVA frameworks have more than one subset of frameworks designed for different tasks
  - Desktop computer applications
  - Database, server applications
  - Mobile applications
  - Embedded Applications
    - Real-time (hard and soft)
    - Non deterministic

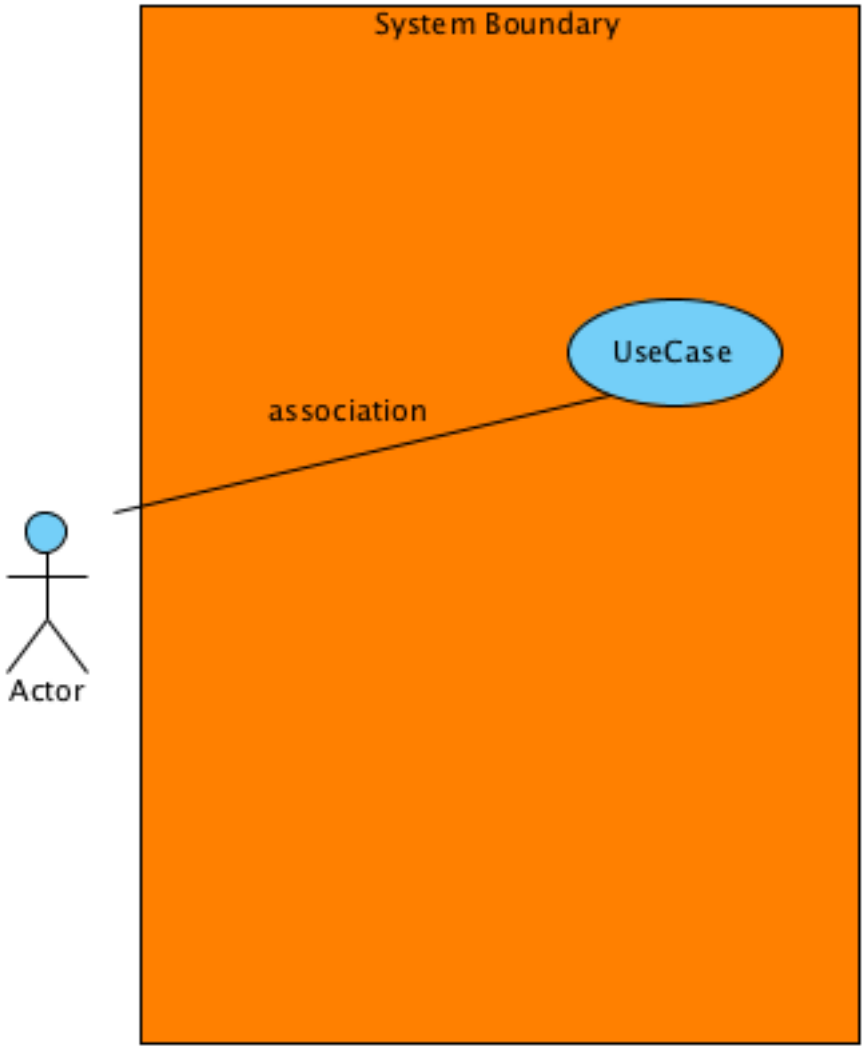
OOD Section

- We shall make use of UML and OOD patterns
- We shall use the MVC and other OOD patterns

OOD Exercise Use Case Diagrams

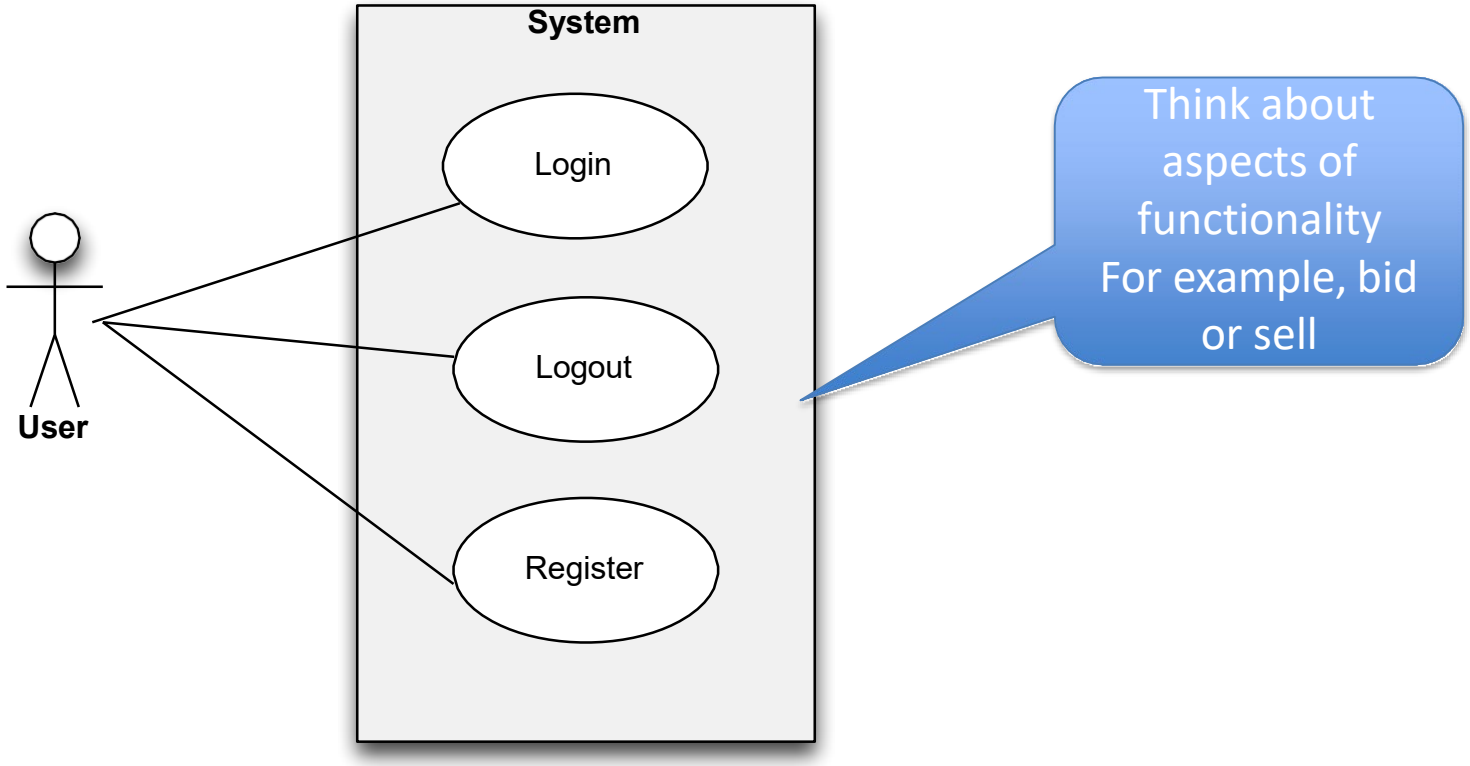
- Use cases are simple verb statements
- Actors are users of the system and can be systems themselves
- Actor can be generalised (like classes) so we could have **Member** 'is a' **User** for example
- First stage of analytical design and are strongly bound to software requirements
- Can also offer a simple view of user interaction with a system
- A use case permeate all design stages so; sequence, activity, class diagram (use case space)
  - Here is the sequence diagram for the use case …
  - Here are the classes involved in satisfying the use case…
  - Here is the activity diagram for the use case…
  - Here are tests for the use case…
  - etc.

OOD Exercise Use Case Diagrams

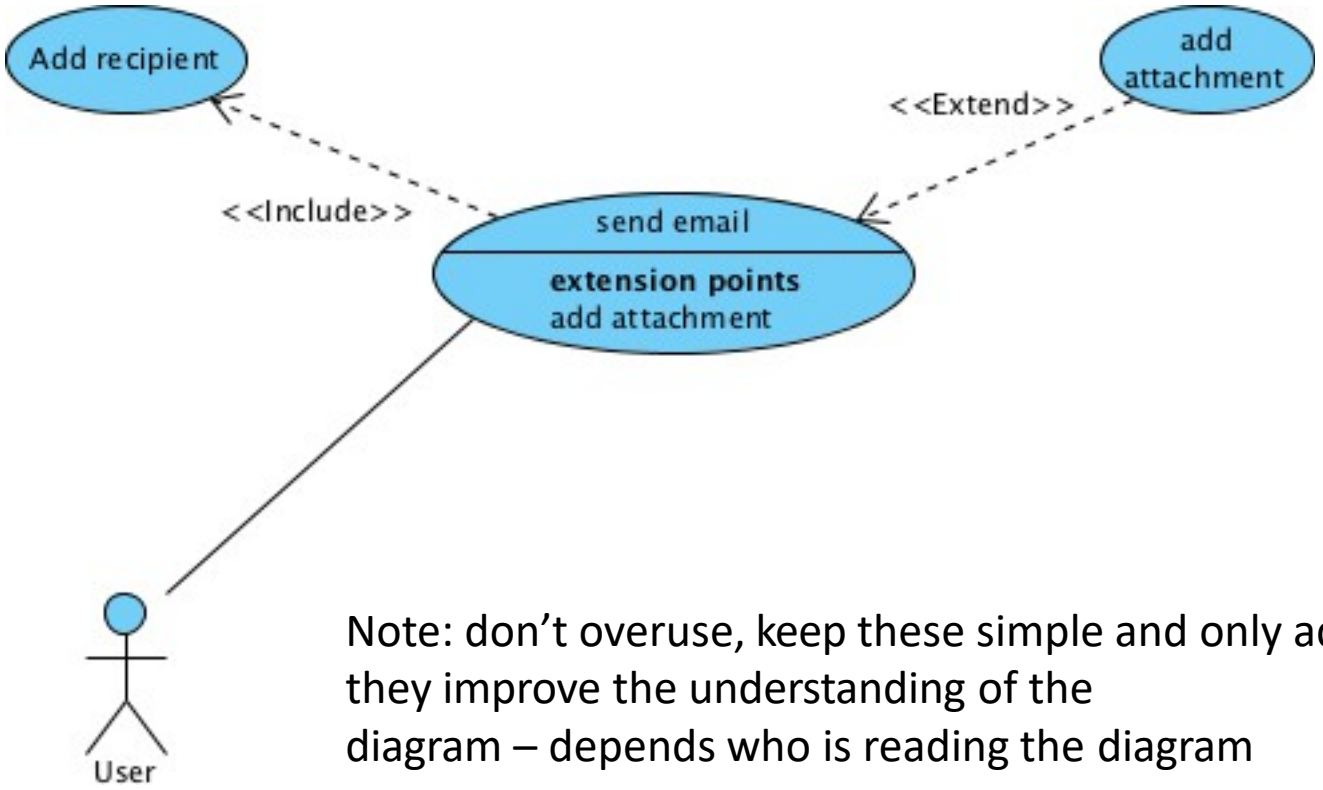


Use Case Diagram Example

- Draw a use-case diagram of the eBay system with the **user** as the actor



Use Case Example Includes and Extends



Note: don't overuse, keep these simple and only add if they improve the understanding of the diagram – depends who is reading the diagram

Books on OOD and UML

- UML Distilled and other short books on UML are essential reading, for example:
  - **UML Distilled: Applying the Standard Object Modeling Language, 2nd Edition**
  - **by Martin Fowler, Kendall Scott, Ivar Jacobson**
- Design patterns
- **Design Patterns: Elements of Reusable Object-Oriented Software**
  - **Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides**