* + 1. **Modelling Tools and Methodologies**

**Unified Modeling Language (UML)**

* Use case diagrams
* Class diagrams
* Sequential diagrams
* State Diagrams

**USE CASE MODEL**

* Describes the proposed functionality of the new system
* Represent a discrete unit of interaction between a user and the system
* A single unit of meaningful work

Customer

**Use Case Model**

<<extends>>

A **Use Case** may ‘include’ another Use Case’s functionality or ‘extend’ another Use Case with its own behavior.

Example:

**Login to system**

**Register with system**

**Create order**

A Use Case description will generally include:

* General comments and notes describing the Use Case
* Requirements- things that the Use Case must allow the user to do such as

<ability to update order>

<ability to modify order> and others.

A Use Case may include:

* Constraints- rules about what can and cannot be done:
  + Pre-conditions that must be true before the Use Case is run

–e.g. <create order> must precede <modify order>;

* Post-conditions that must be true before once the Use Case is run

–e.g. <order is modified and consistent>;

* Invariants: these are always true

–e.g. an order must always have a customer number;

* Scenarios: Sequential descriptions of the steps taken to carry out the Use Case. May include multiple scenarios, to cater for exceptional circumstances and alternate processing paths;
* Scenario Diagrams: Sequence diagrams to depict the workflow – as above but graphically portrayed;
* Additional attributed such as implementation phase, version number, complexity rating, stereotype and status.

**An ACTOR:**

* Is a user of the system
* This includes both human user and other computer systems
* Uses a Use Case to perform some piece of work which is of value to the business
* The set of Use Cases an actor has access to, defines their overall role in the system and the scope of their action

Person

Student

* Actors can participate in a generalization with other actors

Is a generalization of

Student

* Actors may be connected to Use Cases only by associations

Student

Billing System

Registrar

* Here we have a Student interacting with the Registrar and the Billing System via a “Register for Courses” Use Case

**Requirements, Constraints & Scenarios**

* **Requirements**:

These are the formal functional requirements that a Use Case must provide to the end user. They correspond to the functional specifications found in structured methodologies. A requirement is a contract that the Use Case will perform some actions or provide some value to the system.

* **Constraints**:

These are the formal rules and limitations that a Use Case operator under, and includes pre- post- and invariant conditions.

A **pre-condition** specifies what must have already occurred or be in place before the Use Case may start.

A **post-condition** documents what will be true once the Use Case is complete. An **invariant** specifies what will be true throughout the time the Use Case operates.

* **Scenarios**:

These formal descriptions of the flow of events that occurs during a Use Case instance.

Usually described in text and correspond to a textual representation of the Sequence Diagram.

* **Includes and Extends Relationships Between Use Cases**
  + May include the functionality of another as part of its normal processing
  + It is assumed that the included Use Case will be called every time the basic path is run.

*Example* <list order> Use Case may be included every time the <modify order> Use Case is run.

* + A Use Case may be included by one or more Use Cases. This helps to reduce duplication of functionality by factoring out common behavior into Use Cases that are re-used many times.