Project Management

Massimo Felici

Room 1402, JCMB, KB

0131 650 5899

mfelici@inf.ed.ac.uk

Project Management

- Software project management is an essential part of software engineering

  - Concerned with **activities** involved in ensuring that software is delivered on **time** and on **schedule** and in accordance with the **requirements** of the organizations developing and procuring the software
  - Project management is needed because software development is always subject to **budget** and **schedule** constraints that are set by the organization developing the software

- **The Good, the Bad and the Ugly**

  - **Good** management cannot guarantee project success
  - **Bad** Management usually results in project failures
    - Software is delivered late, costs more than originally estimated and fails to meet its requirements …**The Ugly**

Software Management Distinctions

- The product is **intangible**

- The product is uniquely **flexible**

- **Software engineering** is not recognized as an engineering discipline with the sane status as mechanical, electrical engineering, etc.

- The software **development process** is not standardised

- Many software **projects** are 'one-off' projects

The Role of the Project Manager

Software managers are responsible for **planning** and **managing** project development

- **Estimation** of the project effort, time and cost

- **Planning**. Scheduling deliverables, review points and allocation of staff to activities

- **Replanning**. Re-estimating and rescheduling in the light of unfolding circumstances, e.g., risks and quality assurance results

- **Organization**. Establishing a division of labor which is able to make the most effective use of available skills and maximizes productivity potential in the context of characteristics (e.g., risk factors) of the particular project

- **Quality assurance**. Planning and carrying out actions to ensure that the software product meets required quality targets

Project Planning

- Probably the most **time-consuming** project management activity

- **Continuous activity** from initial concept through to system delivery

  - Plans must be regularly revised as new information becomes available

- Various different types of plan may be developed to support the main software project plan that is concerned with schedule and budget

Types of project plan

- **Quality plan**: describes the quality procedures and standards that will be used in the project

- **Validation plan**: describes the approach, resources and schedule used for system validation

- **Configuration Management plan**: describes the configuration management procedures and structures to be used

- **Maintenance plan**: predicts the maintenance requirements of the system, maintenance costs and effort required

- **Staff Development plan**: describes how the skills and experience of the project team members will be developed

Scoping the Problem

- **Objectives** expressed in general terms and in the language application domain

- **Scope** defines the system boundary, explaining what will be included in the system and what will not be included

- **Identify**: the Customer, the system environment, necessary tools, potential reuse, etc.

  - Ask the Customer: Who is the end user? (often not the customer) Who has the authority to accept the finished product? What problem are we addressing? What documentation will be required? When do they believe they need the product? Where is the work to be dune? Why do they need the product? How will the product be developed/acquired?
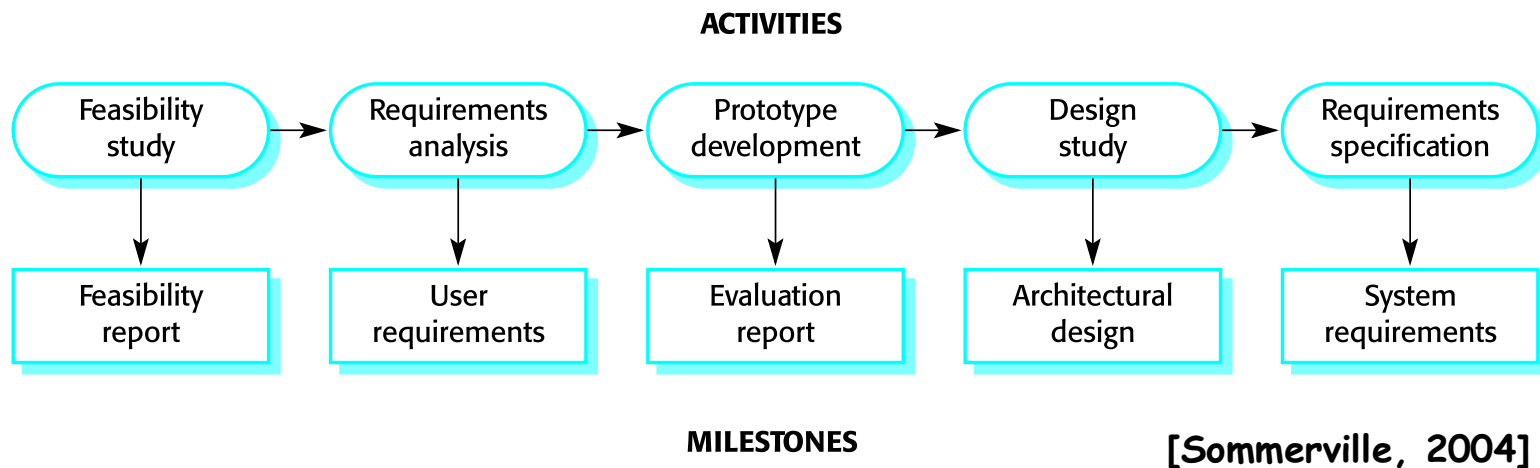
Other Management Activities...

- **Measurement Framework** allows the quantitative analysis of project (e.g., productivity, progress, etc.) and product features (e.g., quality, size, etc.)
  - **Software Metrics.** Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real word in such a way as to describe them according to clearly defined rules.
  - **Quality Assurance plan** describes how reviews, inspections, testing, and other techniques will help to evaluate quality and ensure that it meets the customer's needs.

- **Resource management** identifies (and quantifies) the (needed) resources and describes how resources are allocated throughout the project
  - Resources include infrastructure, staff and time.

- **Feasibility study** also explores alternative solutions

Activity Organization and Milestones

- **Activities** in a project should be organised to produce tangible outputs for management to judge progress

- **Milestones** are the end-point of a process activity

- **Deliverables** are project results delivered to customers

- The waterfall **process** allows for the straightforward definition of progress milestones

Milestones in the Requirements Engineering Process

ACTIVITIES

Feasibility study	Requirements analysis	Prototype development	Design study	Requirements specification

Feasibility report	User requirements	Evaluation report	Architectural design	System requirements

MILESTONES
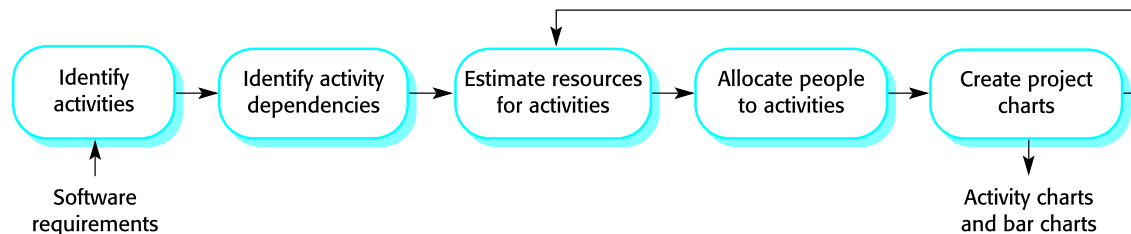
[Sommerville, 2004]

Project Personnel

- Determine the project **schedule** and estimate the associated **effort** and **costs**

  - How many people will be involved in the project
  - What tasks they will perform
  - What abilities and experience they must have so that they can do their job effectively

- The assignment of staff to **tasks** depends on project **size**, staff **expertise** and staff **experience**

- People have different work styles (e.g., preferred styles for interacting with others)

Project Scheduling

- Split project into **tasks** and estimate **time** and **resources** required to complete each task

- Organize tasks concurrently to make optimal use of workforce

- Minimize task **dependencies** to avoid delays caused by one task waiting for another to complete

- Dependent on project managers intuition and **experience**

Problems

- Estimating the difficulty of **problems** and hence the cost of developing a solution is hard

- **Productivity** is not proportional to the number of people working on a task

- Adding people to a late project makes it later because of **communication** overheads

- The **unexpected** always happens. Always allow contingency in planning

Identify activities → Identify activity dependencies → Estimate resources for activities → Allocate people to activities → Create project charts

Software requirements

Activity charts and bar charts

[Sommerville, 2004]

Tracking Progress and Control

- **Scheduling** explores possible ways of allocating (limited) resources across tasks

- **Project scheduling** involves separating the total work involved in a project into separate activities and judging the time required to complete these activities.

- Project can be late with respect to the initial plan. It is important to **track the progress** of the project and compare it to the plan. If significant divergences arise it is necessary to re-plan to take account of the changed circumstances.
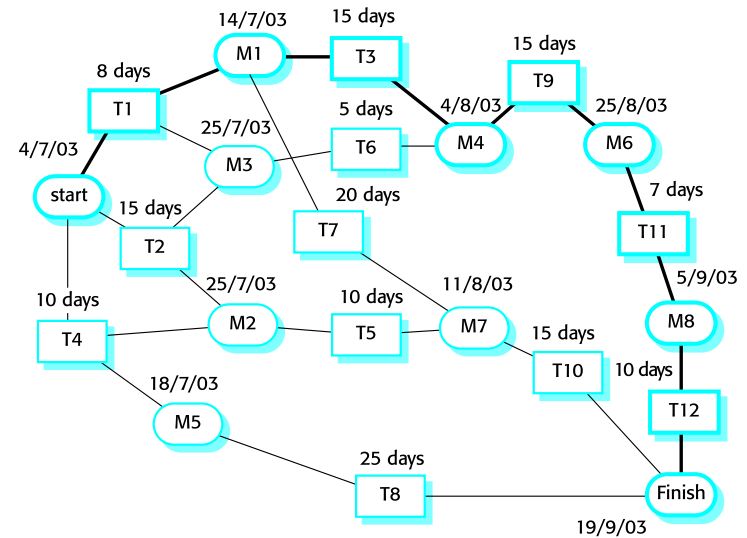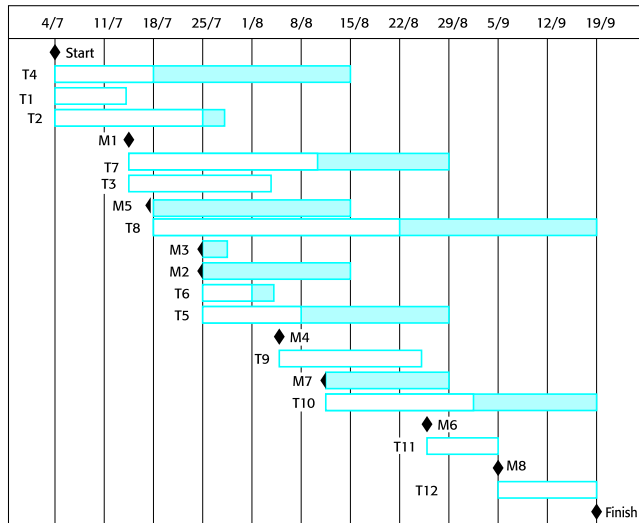
(Graphical) Notations

Task Durations and Dependencies

Activity	Duration (days)	Dependencies
T1	8	
T2	15	
T3	15	T1 (M1)
T4	10	
T5	10	T2, T4 (M2)
T6	5	T1, T2 (M3)
T7	20	T1 (M1)
T8	25	T4 (M5)
T9	15	T3, T6 (M4)
T10	15	T5, T7 (M7)
T11	7	T9 (M6)
T12	10	T11 (M8)

Activity Network

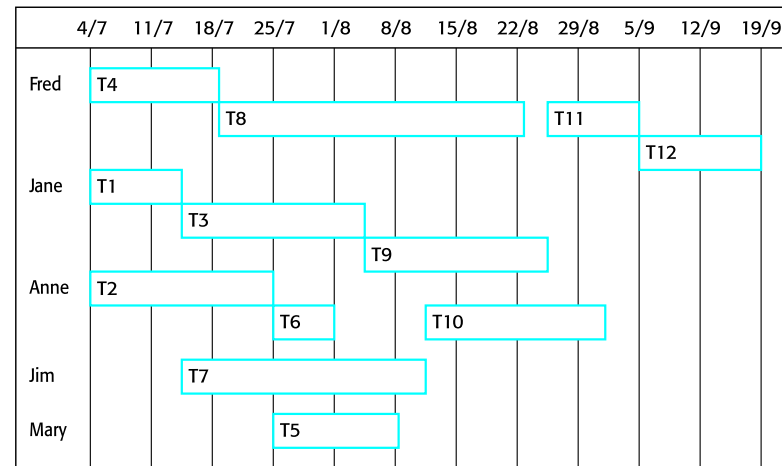## Activity Timeline

Staff Allocation

Risk Management

- Project managers must engage in **risk management** to understand and control the risks on their projects

- A **Risk** is an unwanted event that has negative consequences
  - **Risk impact**: the loss associated with the event
  - **Risk probability**: the likelihood of the risk, measured from 0 (impossible) to 1 (certainty)
  - **Risk control**: the degree to which we can change the outcome
- **Risk Management** involves: Risk **Identification**, Risk **Analysis**, Risk **Planning** and Risk **Monitoring**

  - **Risk Identification** concerns with discovering possible risks to the project
  - **Risk Analysis** considers each identified risk and makes a judgment about the probability and seriousness of it
  - **Risk Planning** considers each identified risk and identifies strategies to manage the risk
  - **Risk Monitoring** involves regularly assessing each identified risk to decide whether that risk is becoming more or less probable and whether the effect of the risk have changed

Software Risks

## Boehm's Top Ten Risk Items (1991)

1. **Personnel** shortfalls

2. Unrealistic **schedules** and **budgets**

3. Developing the wrong software **functions**

4. Developing the wrong **user interface**

5. Gold plating

6. Continuing stream of **requirements changes**

7. Shortfalls in externally performed **tasks**

8. Shortfalls in externally furnished **components**

9. Real-time **performance** shortfalls

10. Straining computer science capabilities

[Sommerville, 2004]

Risk	Affects	Description
Staff turnover	Project	Experienced staff will leave the project before it is finished.
Management change	Project	There will be a change of organisational management with different priorities.
Hardware unavailability	Project	Hardware that is essential for the project will not be delivered on schedule.
Requirements change	Project and product	There will be a larger number of changes to the requirements than anticipated.
Specification delays	Project and product	Specifications of essential interfaces are not available on schedule
Size underestimate	Project and product	The size of the system has been underestimated.
CASE tool under-performance	Product	CASE tools which support the project do not perform as anticipated
Technology change	Business	The underlying technology on which the system is built is superseded by new technology.
Product competition	Business	A competitive product is marketed before the system is completed.

Risk Management Process
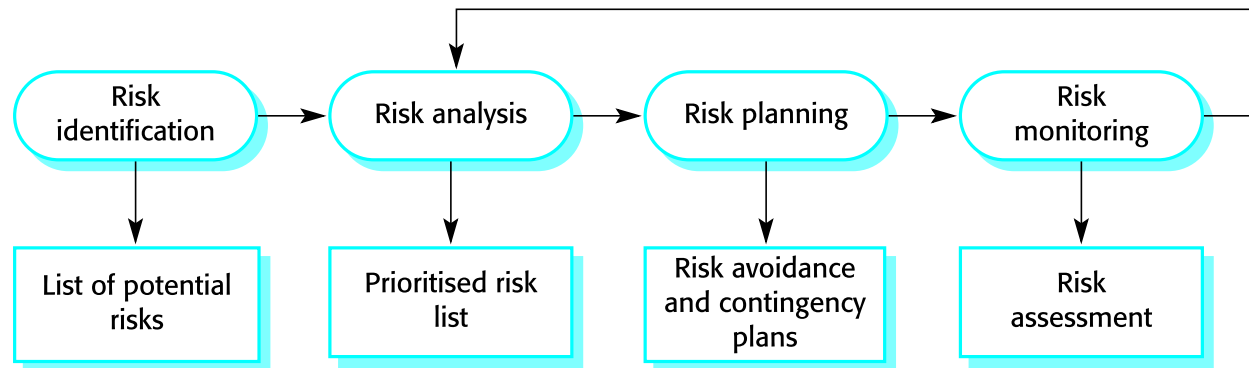
[Sommerville, 2004]

- **Risk identification**: identifies project, product and business risks

- **Risk analysis**: assesses the likelihood and consequences of these risks

- **Risk planning**: draws up plans to avoid or minimise the effects of the risk

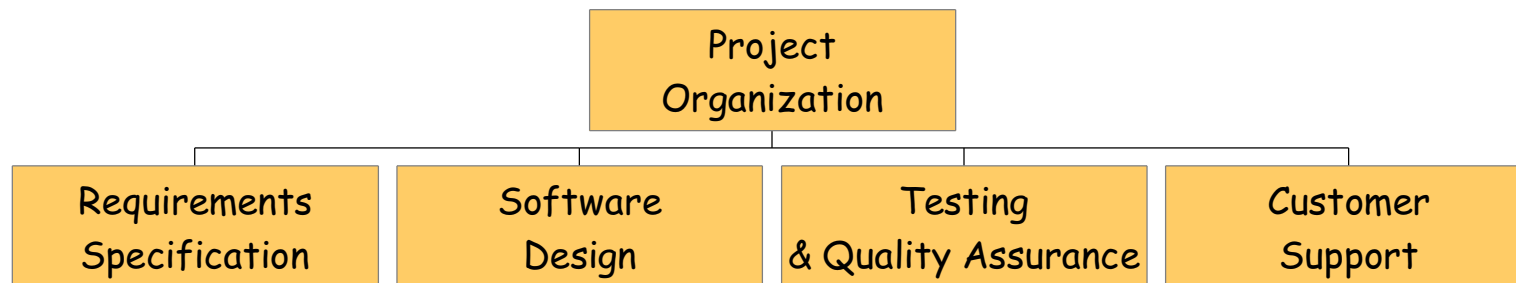- **Risk monitoring**: monitors the risks throughout the project

Project Organization

- Team members are organized in ways that enhance the completion of quality products

- The choice of an appropriate structure for your project depends on several things
  - The backgrounds and work styles of the team members
  - The number of people on the team
  - The management styles of the customers and developers

- Comparison of **Organizational Structures**: Highly or Loosely Structured; High Certainty of Uncertainty; Repetition or New techniques (or Technology); Large or Small Projects

- Examples of Organizations
  - Functional
  - Matrix
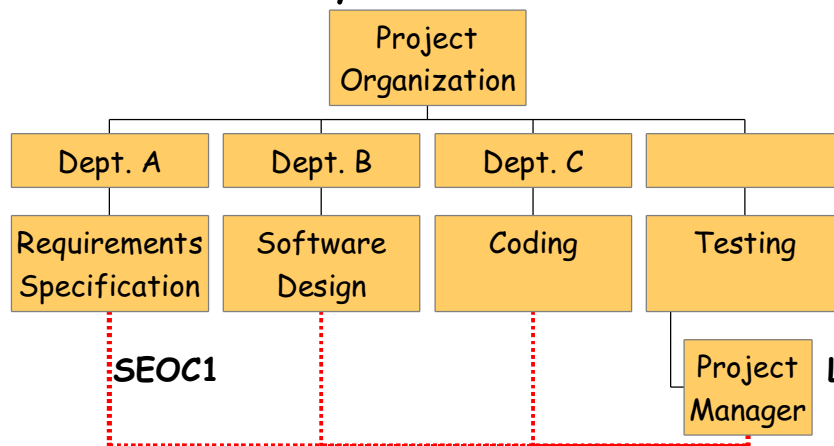 - Integrated Product Development Teams (IPDTs)

Project Organization: Functional

- Basic hierarchical organization

- Project organized by disciplines and functions

- **Characteristics**: Narrow set of work methods, deep technical expertise, Develops skills and morale; Service-oriented, Communication responsibility on group manager

- **Problems**: Elitism within expertise areas, Communication difficult, no project "ownership"

```
              ┌──────────────────┐
              │     Project      │
              │  Organization    │
              └──────────────────┘
```

| Requirements Specification	Software Design	Testing & Quality Assurance	Customer Support

Project Organization: Matrix

- Based on a specific project; Experts are borrowed, but not removed

- **Strong Matrix:** team leader is the principal authority, Control of schedule and budget, Acquire personnel, Perform reviews

- **Weak Matrix:** team leader is only a coordinator, Spokesperson to higher management, Steering committee has ultimate authority

- **Characteristics**: Specialists work on part-time basis for several projects, Top management selects project manager and staff Good for short-lived projects "Task force" Mentality

- **Problems**: Staff attention fractured Conflicting obligations Large amount of communication Strong top management involvement; Reporting to home "base" is difficult

```
                    Project
                  Organization
         ┌──────────┬──────────┬──────────┐
      Dept. A    Dept. B    Dept. C
  Requirements   Software    Coding      Testing
  Specification   Design
                                      Project
                                      Manager
```

Lecture Note 07

Project Organization: IPDT

- Single, long-term project; Organized by component

- Combining individuals from different functional groups into an interdisciplinary work unit

```
┌─────────────────────────┐
│   System Architecture   │
│    Project Manager      │
└─────────────────────────┘
        │
    ┌───┴────────┐
┌──────────────┐ ┌──────────────┐
│ Subsystem A  │ │ Subsystem B  │
│Project Manager│ │Project Manager│
└──────────────┘ └──────────────┘
```

- **Characteristics**: Tightly controlled effort, Complex or large project, Independent authority for sub-managers, Direct contact with customer, Reporting is easy

- **Problems**: Loss of project – what to do with staff?, Difficult to enforce standards, Overspecialization
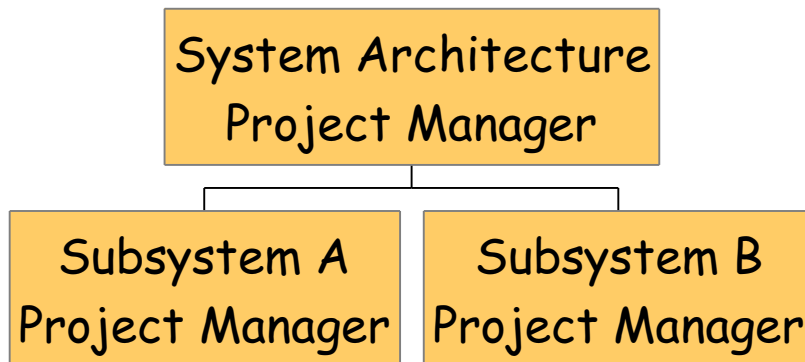
Reading/Activity

- Chapter 8 (Software Engineering Management) and 9 (Software Engineering Process) of the SWEBOK

# Secure Coursework Submission System

Massimo Felici

Room 1402, JCMB, KB

0131 650 5899

mfelici@inf.ed.ac.uk

Requirements Viewpoints

- What are your system requirements?
  - You are looking at the same system from different viewpoints
  - Requirements viewpoints highlight system requirements (i.e., your perspective understanding of the system)

- Requirements Viewpoints:
  - **Student viewpoint (S)**
  - **Lecturer viewpoint (P)**
  - **Admin viewpoint (A)**

- Why viewpoints?
  - Requirements completeness
  - Partial specifications
  - Viewpoints highlight requirements association, hence Traceability

- What is a viewpoint?
  - Encapsulation of partial information about system requirements from a particular perspective

- **Main Issues:** Difficult to Identify a stable set of requirements; Contradicting viewpoints

SCS

S

Student Viewpoint

Admin Viewpoint

Lecturer Viewpoint
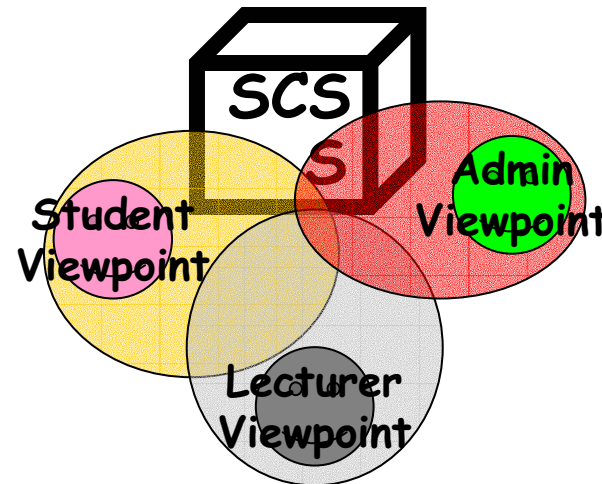
From Requirements Viewpoints to Requirements Specification(s)

Student Viewpoint

Admin Viewpoint

SCSS

Lecturer Viewpoint

SCSS
Requirements Specification

SCSS
Requirements Specification

SCSS
Requirements Specification
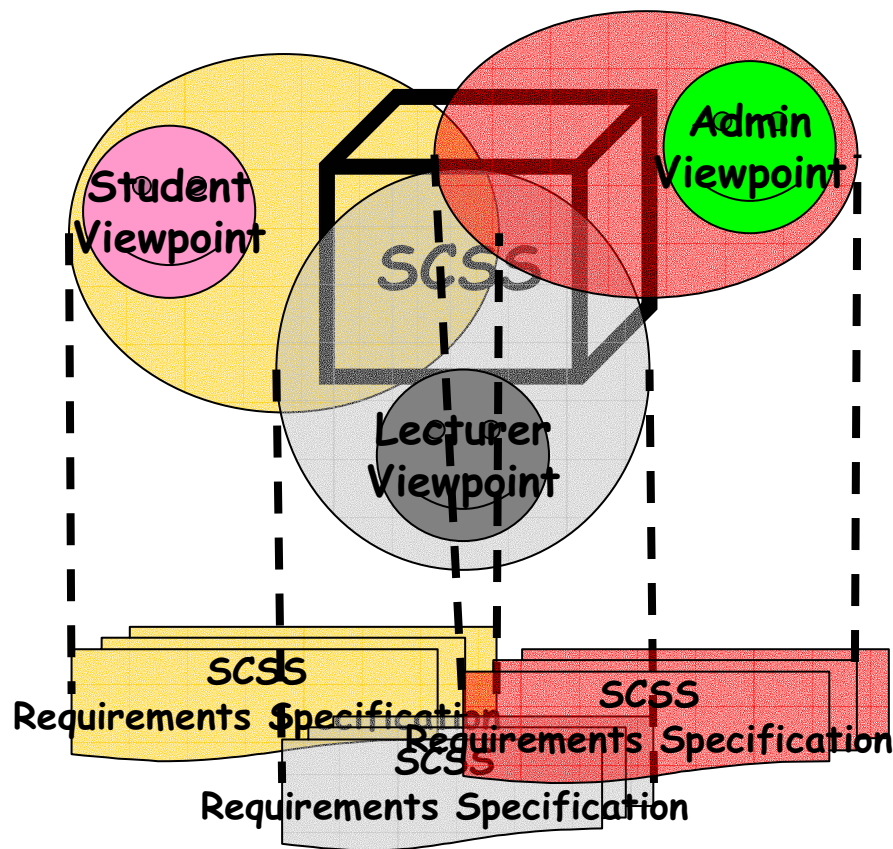
- ■ Identify
  - high level requirements
  - the System Stakeholders
  - The scope of the system
- ■ Capture requirements from your viewpoint (i.e., S, P or A)

Document your Requirements

1. Document your Requirements

- **Use a Requirements Specification template**

o Overall Objectives

"The system has to support teaching and administrative activities related to the courses' assessments. The system should support the secure submission of course exams and solutions."

o Functional Requirements

1. "The system should be integrated with DICE."
2. "The system should allow the submission of practicals and exams.
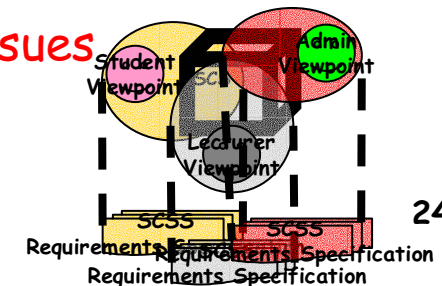3. "The system supports logins from different locations."

…

o Non-Functional Requirements

1. "The System should guarantee secure submissions"

…

o Others

1. Open Issues

UML Modeling

2. Use Cases

- Capture your "functional" requirements" graphically
- Define the system boundaries
- Generalizing and structuring let you simplify use cases
- **Use a Template to write each use case**

3. Class Diagrams

- Some class may have missing information, because you don't yet know all the "responsibilities" and the "collaborators"

4. Validation by CRC Cards

- Run few use cases (point 2) to validate your class diagrams (and use cases too)
- Pick up significant use cases
- Pick up also use cases involving different Actors (e.g., S and P, S and A, P and A)
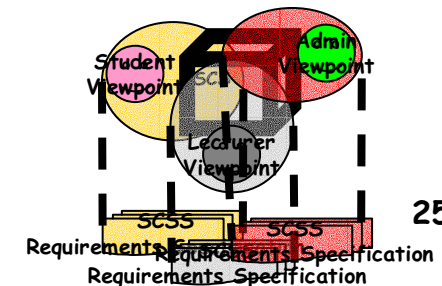
Deliverable 1 Assessment

- **Completeness**

  Deliverable 1 consists of **1-4**

- **Coverage (Requirements Completeness)**

  - The "basic functionalities" should be covered

- **Quality (of the design)**

  - UML and OO Design Proficiency
  - Software Engineering practice (e.g., use of templates)

- **Productivity**

  - Taking into account team size