Lecture Zero
Software Engineering
CSE320

Course details

- **LTP – 3 0 0 [Three lectures/week]**

- **Text Book**

  FUNDAMENTALS OF SOFTWARE ENGINEERING by RAJIB MALL, PHI (PRETICE HALL INDIA),

Course Assessment Model

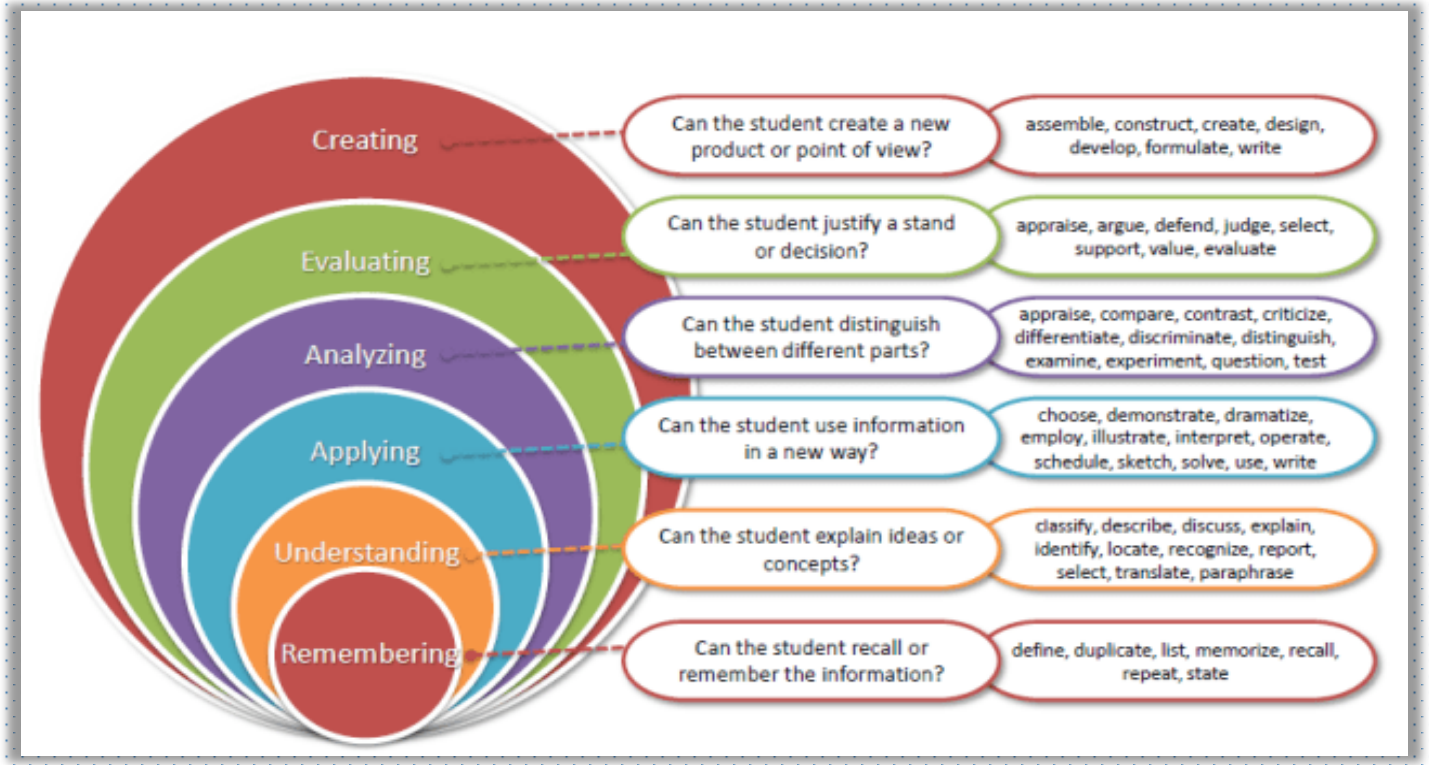
- **Marks break up***
- **Attendance** 5
- **CA (Assignment–Case Based+Test)** 25
- **MTT** 20
- **ETE** 50
- **Total** 100

Detail of Academic Tasks

- **AT1: Assignment- Case based**
- **AT2: Class Test**

(both are compulsory components)

Revised Bloom's Taxonomy

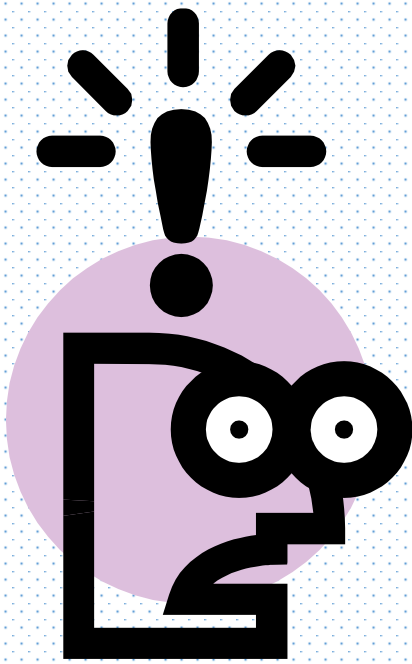


Course Outcomes

- Explain various software development life cycle models and write software requirement Specifications.

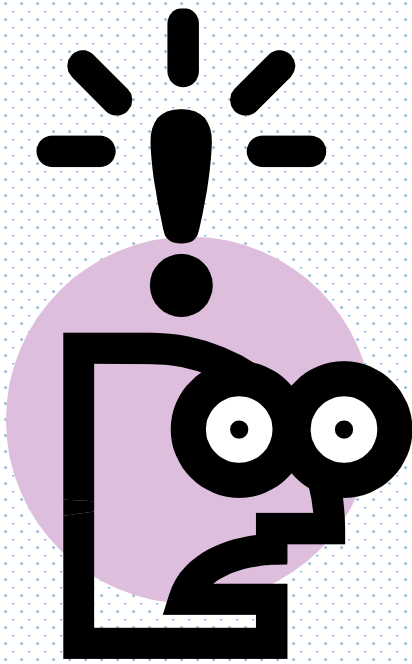
- Use design techniques to develop software design from requirement specifications

- Apply the constructs of unified modelling language for object modelling



Course Outcomes

- Apply testing techniques to develop test cases for testing the software systems
- Apply project management techniques to plan , organize and manage software project development
- Compare various software quality standards and know the current trends in the area of software engineering



Program Outcomes

- **PO1**
  Engineering knowledge:Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

- **PO2**
  Problem analysis::Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

- **PO3**
  Design/development of solutions::Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

- **PO4**
  Conduct investigations of complex problems::Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

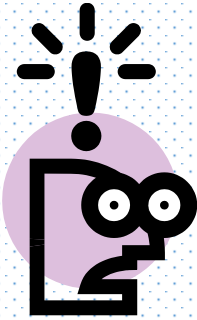
Program Outcomes

- **PO5**
  Modern tool usage::Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

- **PO6**
  The engineer and society::Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

- **PO7**
  Environment and sustainability::Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

- **PO8**
  Ethics::Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.



Program Outcomes

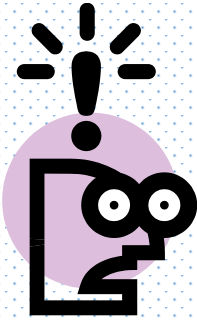
- **PO9**
  Individual and team work::Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

- **PO10**
  Communication::Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

- **PO11**
  Project management and finance::Demonstrate knowledge and understanding of the engineering, management principles and apply the same to one's own work, as a member or a leader in a team, manage projects efficiently in respective disciplines and multidisciplinary environments after consideration of economic and financial factors.

- **PO12**
  Life-long learning::Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

- **PO13**
  Competitive Skills::Ability to compete in national and international technical events and building the competitive spirit alongwith having a good digital footprint.



The course contents

Before MTE

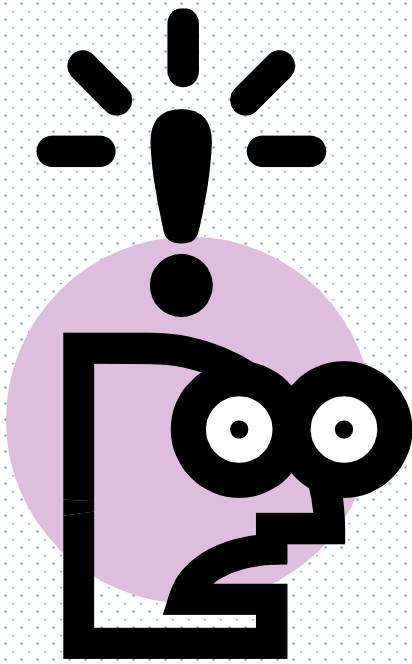
- **Introduction to software engineering** :

  Evolution and impact of software engineering, ***Software life cycle models,*** **Feasibility study,** Functional and non-functional requirements, Requirement gathering, Requirement analysis and specification

- **Issues in software design** : cohesion, coupling, DFDs

- **Object modelling** :

  Object modelling using UML, Object oriented software development, User interface design, Coding standards and code review techniques



The course contents

- **Testing** :

  After MTE

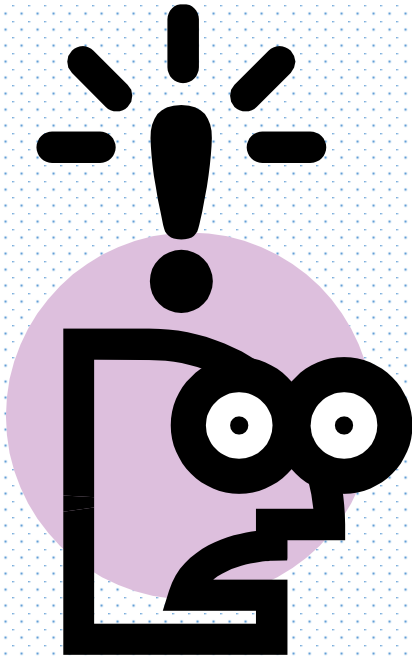
  Fundamentals of testing, White box and black box testing, Test coverage analysis and test case design techniques, Mutation testing, Static and dynamic analysis, Software reliability metrics, Reliability growth modelling.

- **Software project management** :

  Project management, Project planning and control, Cost estimation, Project scheduling using PERT and GANTT charts, Software Configuration Management

- **Quality management** :

  Quality management, ISO and SEI CMMI, PSP and Six sigma, Software Maintenance, reuse, CBSD, CASE, Advance topics of Software Engineering.



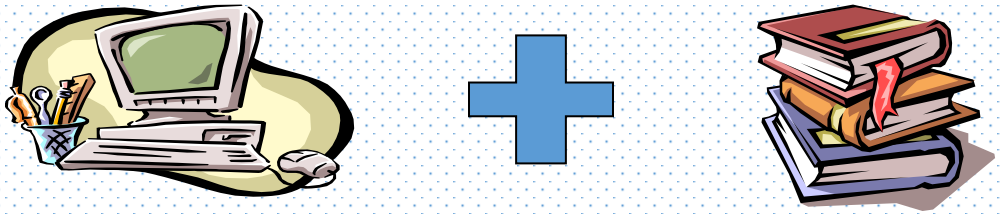
The hitch…

The three BURNING questions in mind…

- **What is software? Is it different from Program?**
- **What is Software Engineering?**
- **Why Software Engineering?**
- **What are learning outcomes?**

What is software?

- **Computer programs** and **associated documentation**



- **Software products** may be developed for a particular customer or may be developed for a general market.
- **Software products** may be

1. **Generic** - developed to be sold to a range of different customers

2. **Bespoke** - developed for a single customer according to their specification

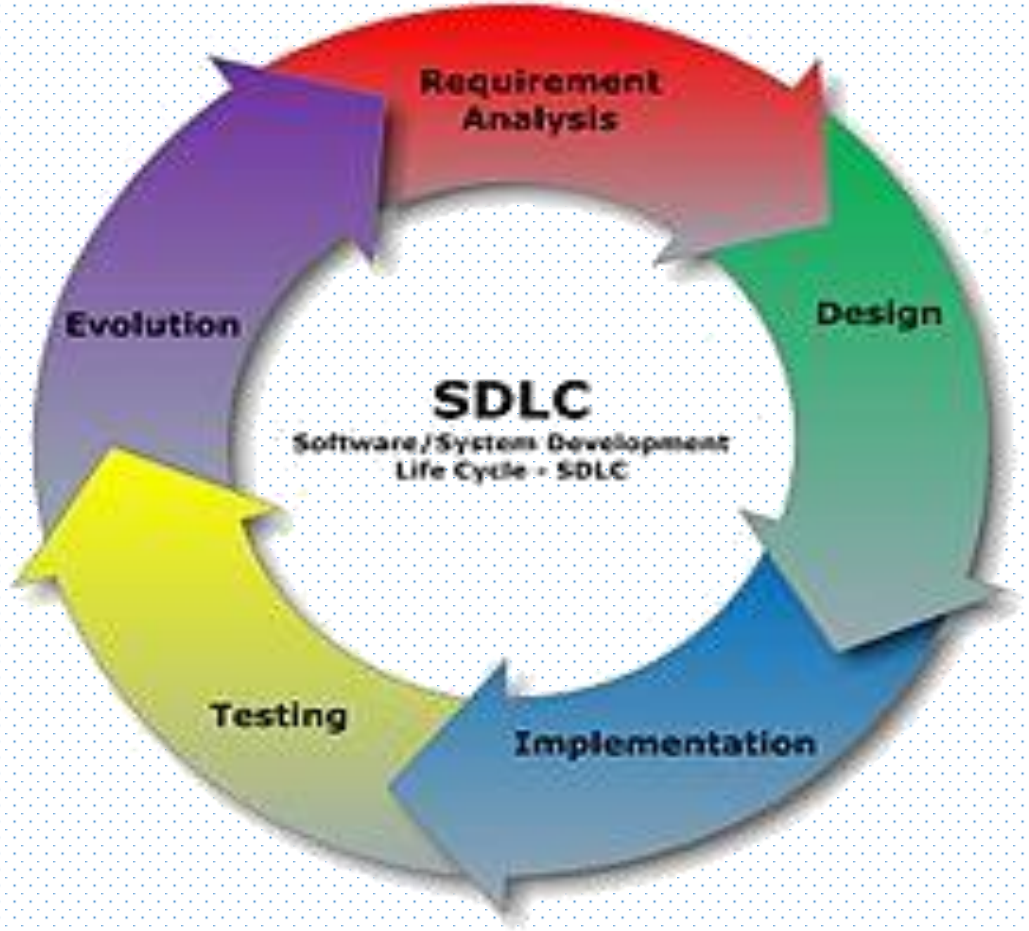
What is software engineering?

Software engineering is an engineering discipline which is concerned with all aspects of software production

Software engineers should

- adopt a systematic and organised approach to their work
- use appropriate tools and techniques depending on
  - the problem to be solved,
  - the development constraints and
  - the resources available

Phases of Development



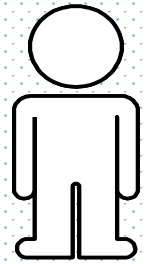
The Role of Software Engineering-1

A bridge from customer needs to programming implementation

First law of software engineering

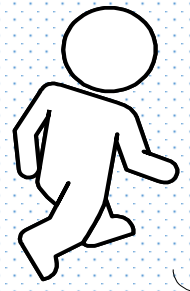
Software engineer must learn the problem domain (problem cannot be solved without understanding it first)

The Role of Software Engineering-2

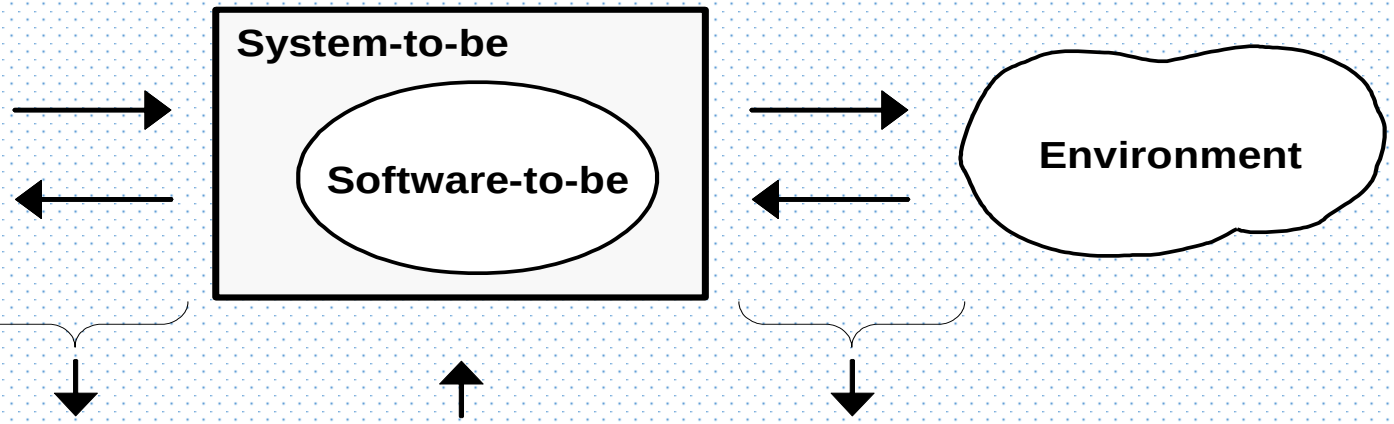


Customer:

Requires a computer system to *achieve some business goals* by user interaction or interaction with the environment in a specified manner



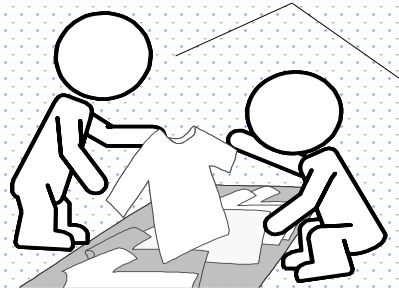
User



Software Engineer's task:

To ***understand how*** the system-to-be needs to interact with the user or the environment so that customer's requirement is met and ***design*** the software-to-be

May be the same person

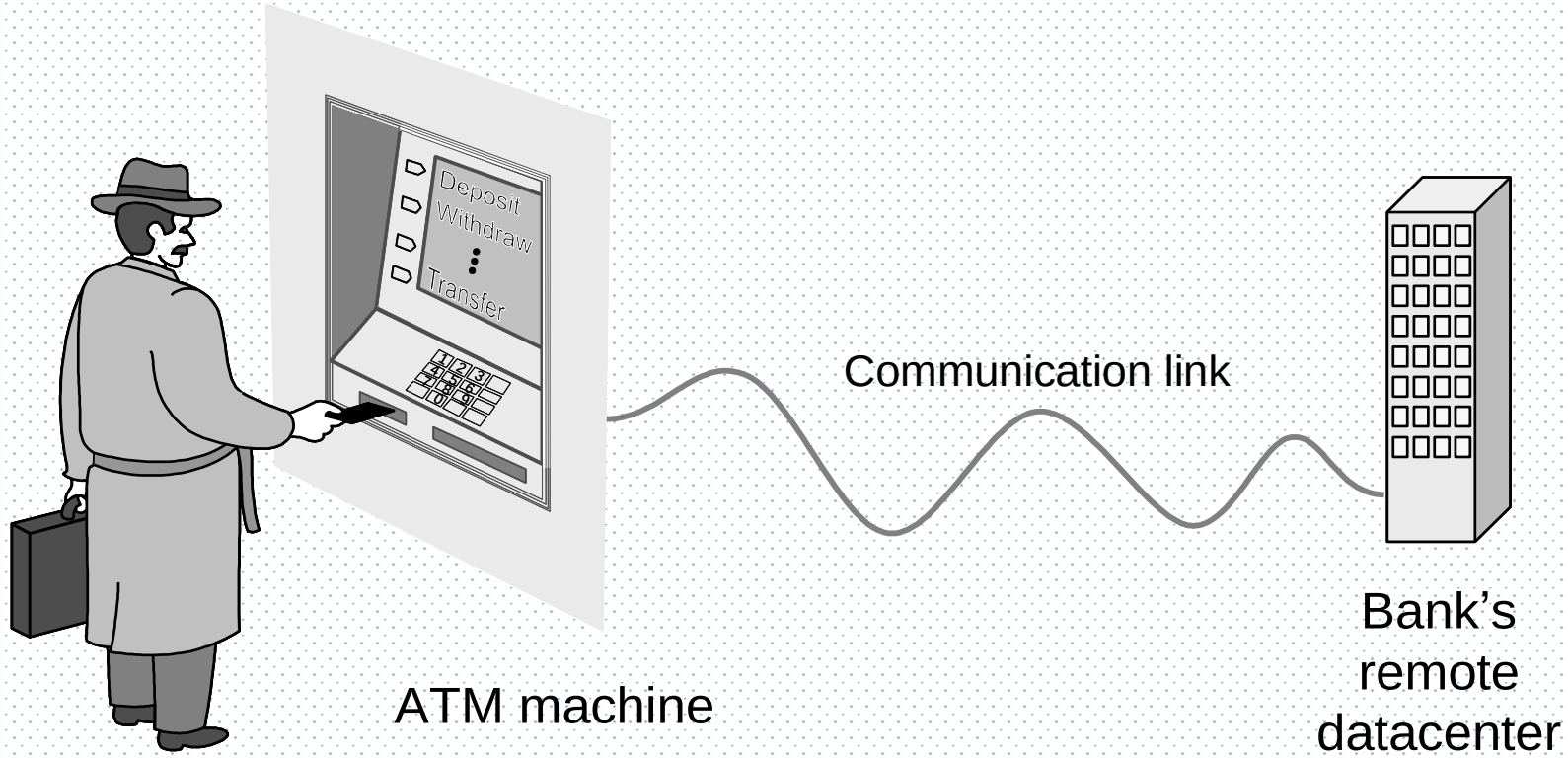


Programmer's task:

To ***implement*** the software-to-be designed by the software engineer

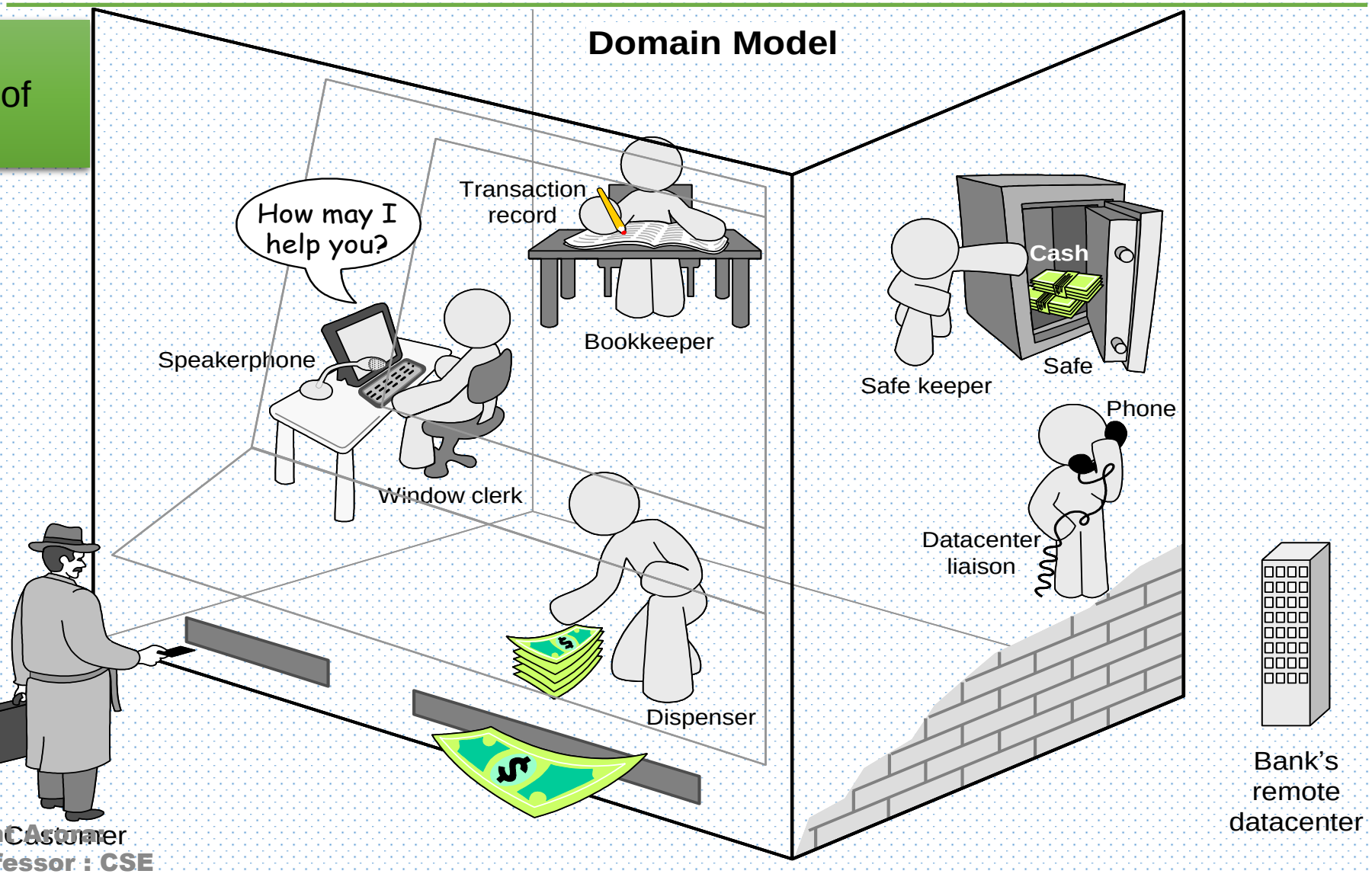
Example: ATM Machine

Understanding the money-machine problem:

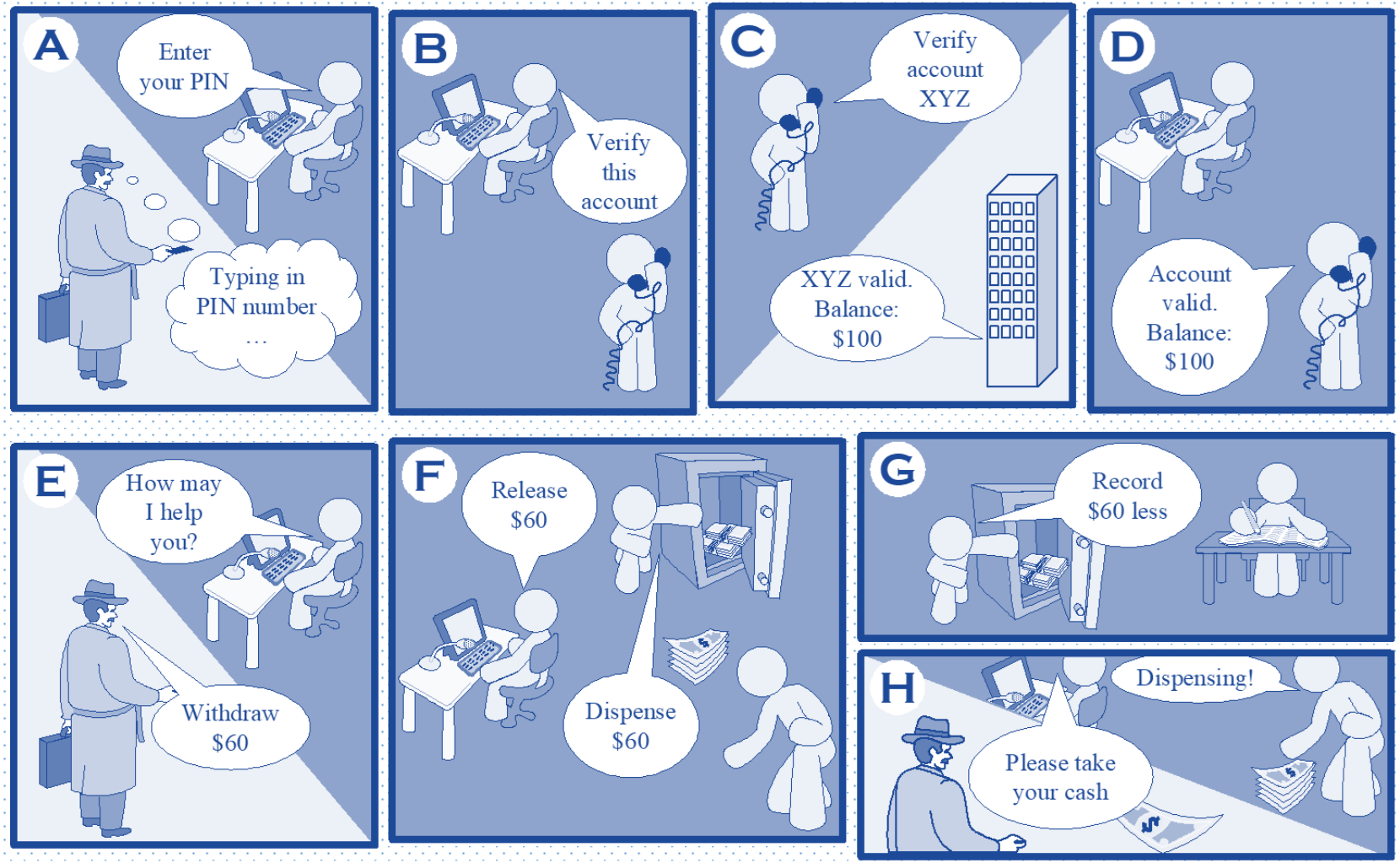


How ATM Machine Might Work

Domain model created with help of domain expert



Cartoon Strip: How ATM Machine Works?



Software Engineering Blueprints

- Specifying software problems and solutions is like cartoon strip writing
- Unfortunately, most of us are not artists, so we will use something less exciting:
  Designing symbols
- However …

Second Law of Software Engineering

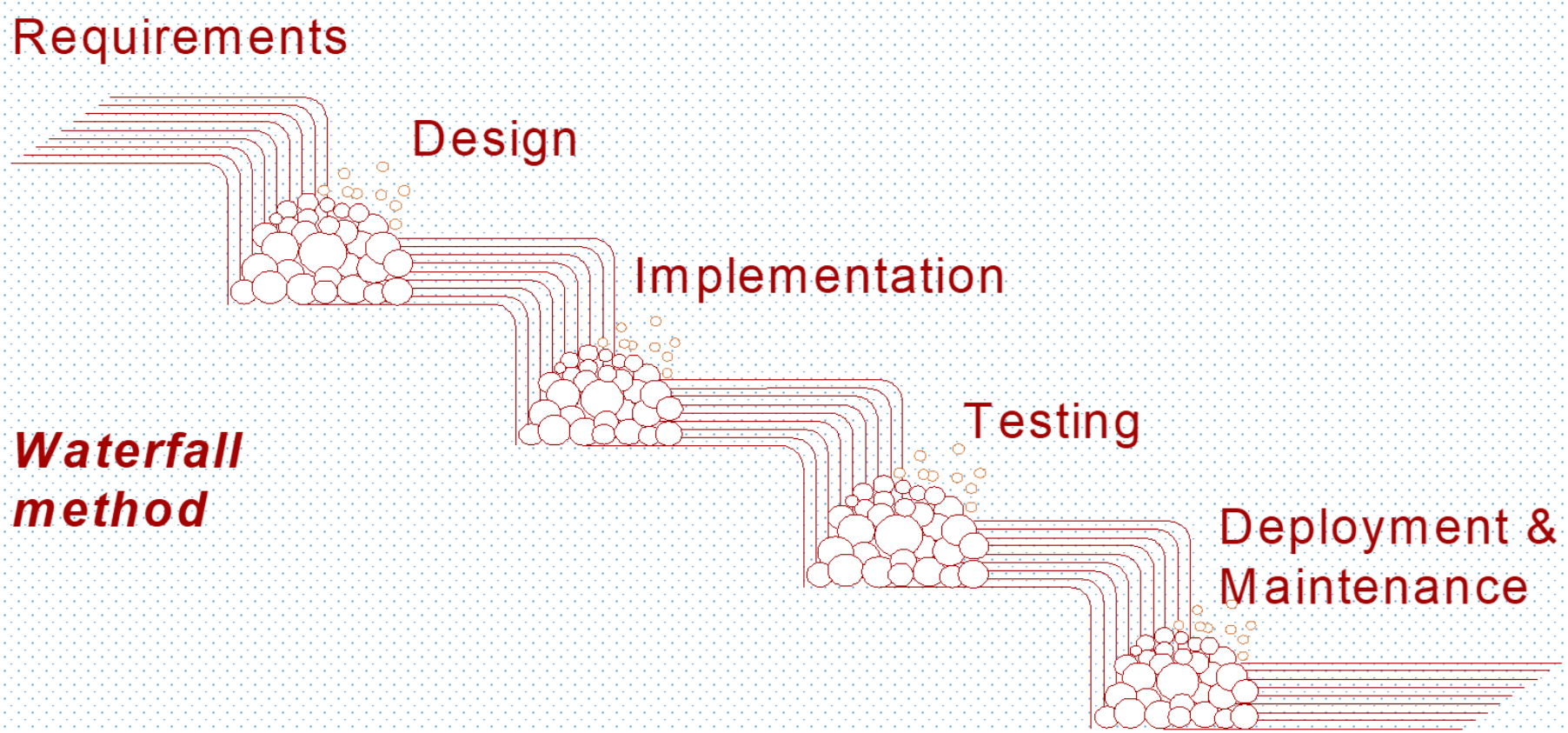
- **Software should be written for people first**
  - (Computers run software, but hardware quickly becomes outdated)
  - Useful + good software lives long
  - To nurture software, people must be able to understand it

Software Development Methods

- Method = work strategy
  - The Feynman Problem-Solving Algorithm:
    (i) Write down the problem (ii) think very hard, and (iii) write down the answer.
- Waterfall
  - Unidirectional, finish this step before moving to the next
- Iterative + Incremental
  - Develop increment of functionality, repeat in a feedback loop
- Agile
  - User feedback essential; feedback loops on several levels of granularity

Software Development Methodologies

Waterfall Method



Unidirectional, no way back finish this step before moving to the next

Software myths

1. "If we get behind schedule, we can just add more people"

- Fact: Adding people to a late project makes it even later.
- Someone has to teach the new people.

2. "A general statement of objectives is enough to start programming".

- Fact: Incomplete requirements are a major cause for project failures.

3. "Changes in requirements are easy to deal with because software is flexible".

- Fact: Changes are hard and expensive.
- Especially during coding and after software deployment.

Software myths

4. "Once we get the program running, we are done"

- Fact: Most effort comes after the software is delivered for the first time.
- Bug fixes, feature enhancements, etc

5. "The only product is the running program"

- Fact: Need the entire configuration
- Documentation of system requirements, design, programming, and usage

Software crises

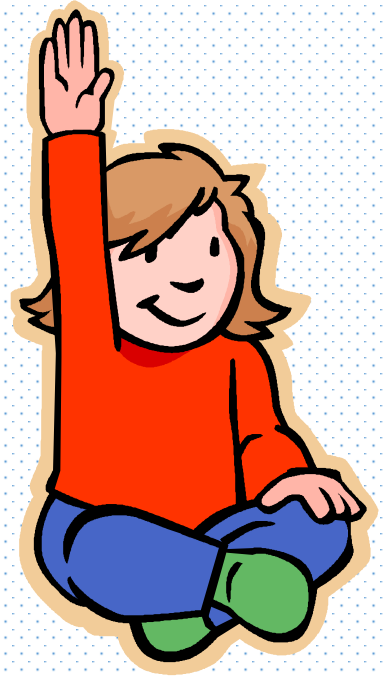
- The various software crises are:

1. Over-budget.
2. Not delivering product on time.
3. Product is of poor quality.
4. Software product is not meeting the customer requirements.

What are the attributes of good software?

The software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable

- **Maintainability**
  - Software must evolve to meet changing needs
- **Dependability**
  - Software must be trustworthy
- **Efficiency**
  - Software should not make wasteful use of system resources
- **Usability**
  - Software must be usable by the users for which it was designed



Next Class: Software Life Cycle Models

www.lpu.in