

# Introduction

**Dr. RAJIB MALL**

Professor

Department Of Computer Science & Engineering  
IIT Kharagpur.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# About Myself

- RAJIB MALL
- B.E., M.E., Ph.D from Indian Institute of Science, Bangalore
- Worked with Motorola (India)
- Shifted to IIT, Kharagpur in 1994
  - Currently Professor at CSE department



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What is Software Engineering?

- Engineering approach to develop software.
  - Building Construction Analogy.
- Systematic collection of past experience:
  - Techniques,
  - Methodologies,
  - Guidelines.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# IEEE Definition

- “Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.”



IIT KHARAGPUR

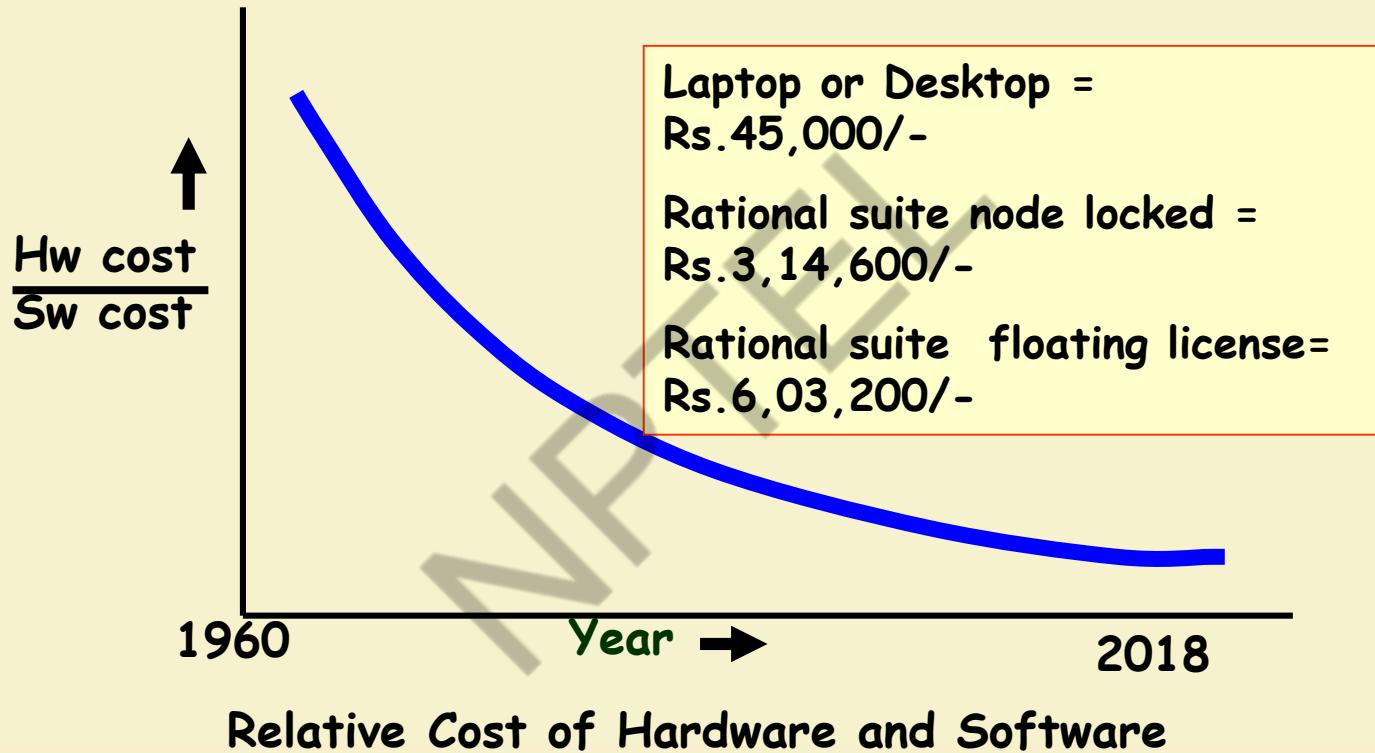


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Software Crisis

- It is often the case that software products:
  - Fail to meet user requirements.
  - Expensive.
  - Difficult to alter, debug, and enhance.
  - Often delivered late.
  - Use resources non-optimally.

# Software Crisis (cont.)



# Then why not have entirely hardware systems?...

- A virtue of software:
  - Relatively easy and faster to develop and to change...
  - Consumes no space, weight, or power...
  - Otherwise all might as well be hardware.
- The more is the complexity of software, the harder it is to change--why?
  - Further, the more the changes made to a program, the greater becomes its complexity.

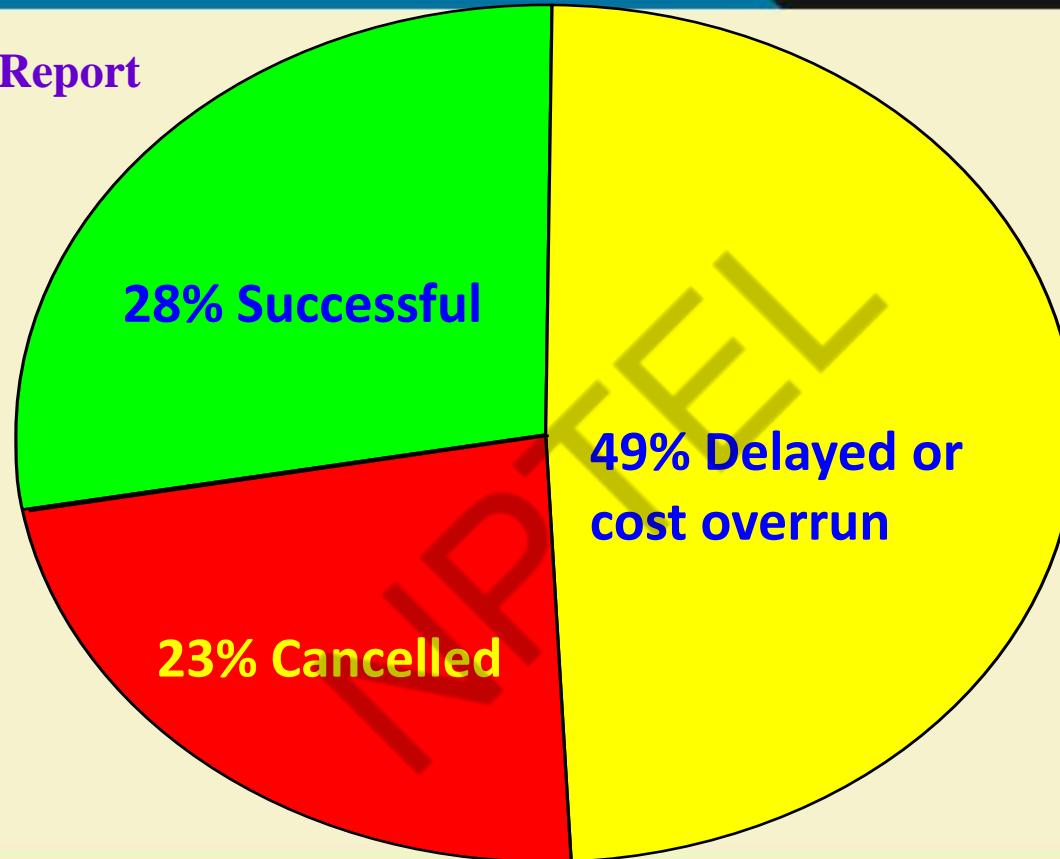


IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Standish Group Report



IIT KHARAGPUR

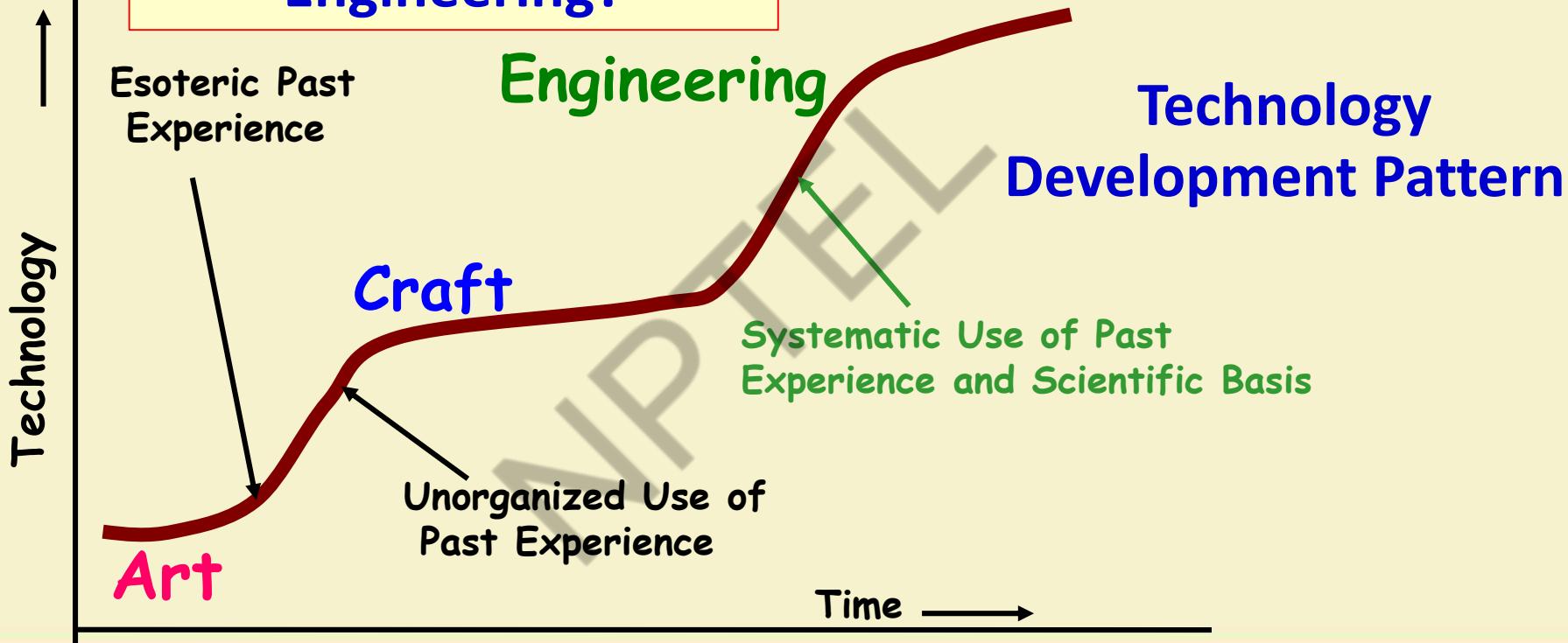


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Which Factors are Contributing to the Software Crisis?

- Larger problems,
- Poor project management
- **Lack of adequate training in software engineering,**
- Increasing skill shortage,
- Low productivity improvements.

# Programming: an Art or Engineering?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Programming an Art or Engineering?

- Heavy use of past experience:
  - Past experience is systematically arranged.
- Theoretical basis and quantitative techniques provided.
- Many are just thumb rules.
- Tradeoff between alternatives.
- Pragmatic approach to cost-effectiveness.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

- Early programmers used **exploratory** (also called **build and fix**) style.
  - A `dirty' program is quickly developed.
  - The bugs are fixed as and when they are noticed.
  - Similar to how a junior student develops programs...

What is  
Exploratory  
Software  
Development?

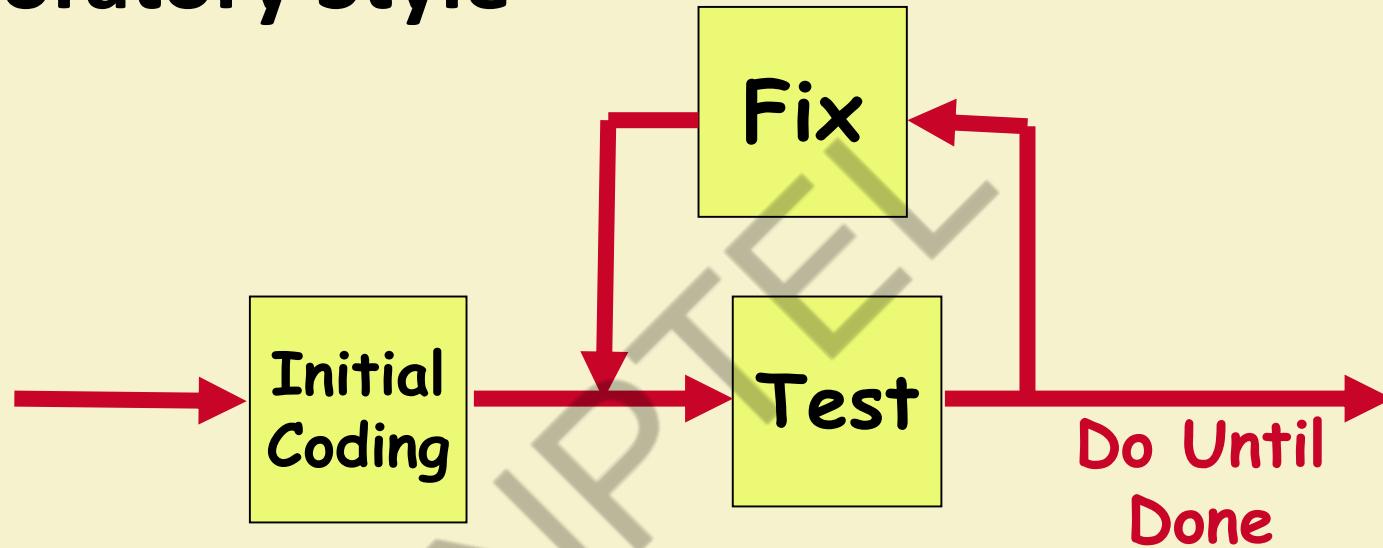


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

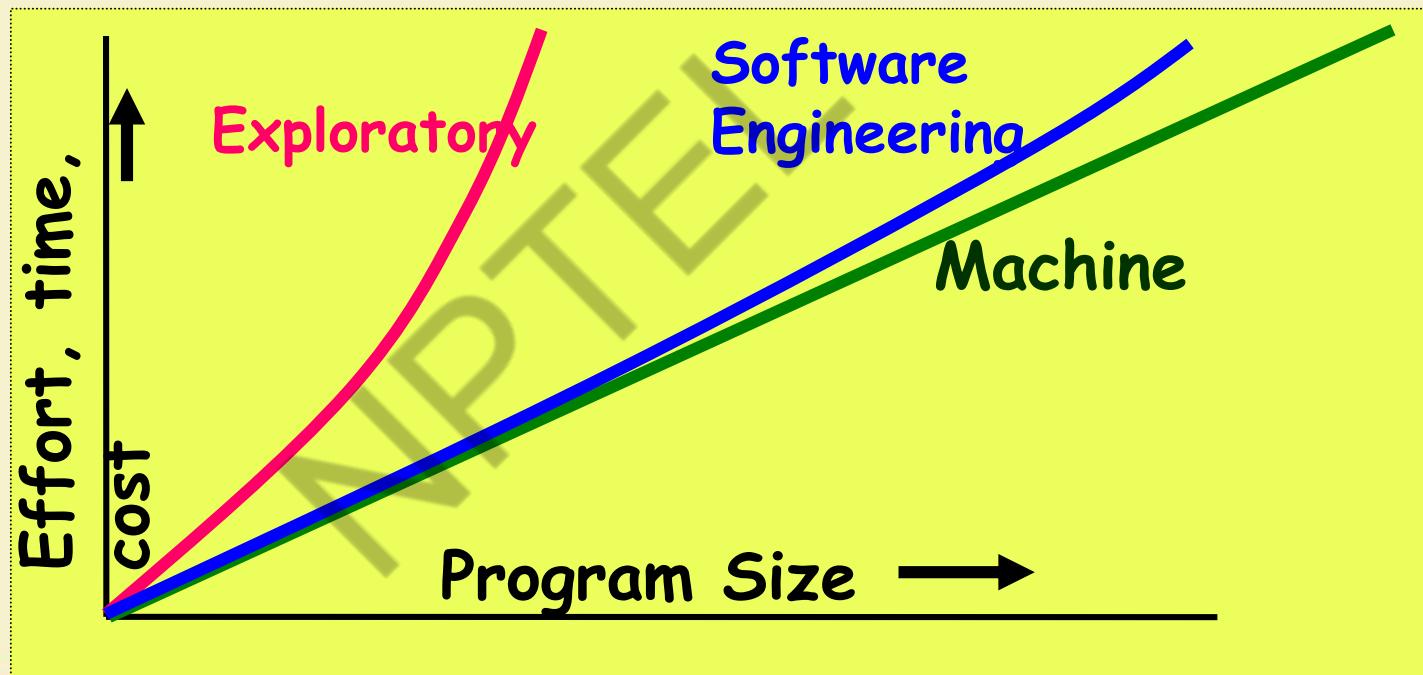
# Exploratory Style



Does not work for nontrivial projects... Why?...

# What is Wrong with the Exploratory Style?

- Can successfully be used for developing only very small (toy) programs.



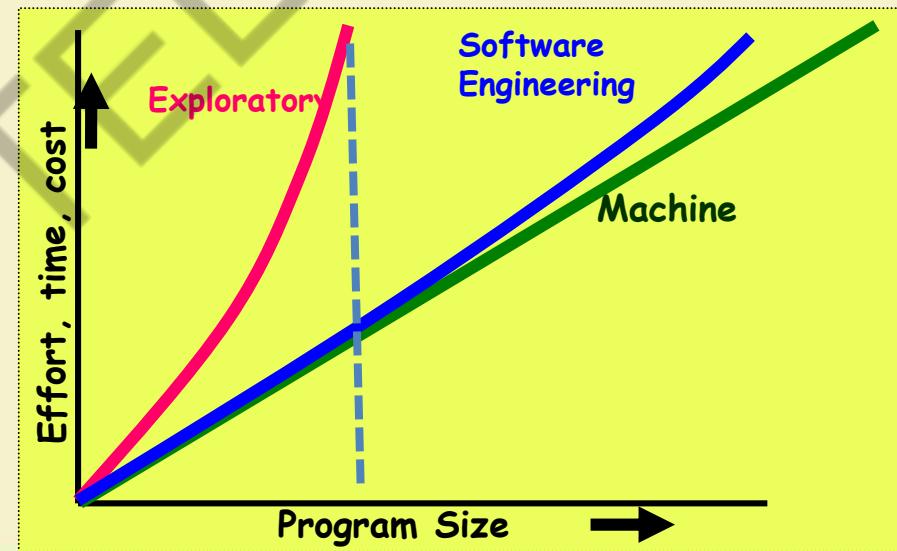
# What is Wrong with the Exploratory Style?

Cont...

- Besides the exponential growth of effort, cost, and time with problem size:
  - Exploratory style usually results in unmaintainable code.
  - **It becomes very difficult to use the exploratory style in team development environments...**

# What is Wrong with the Exploratory Style? Cont...

- Why does the effort required to develop a software grow exponentially with size?
- Why does the approach completely breaks down when the size of software becomes large?



# An Interpretation Based on Human Cognition Mechanism

- Human memory can be thought to be made up of two distinct parts [Miller 56]:
  - **Short term memory and**
  - **Long term memory.**



IIT KHARAGPUR

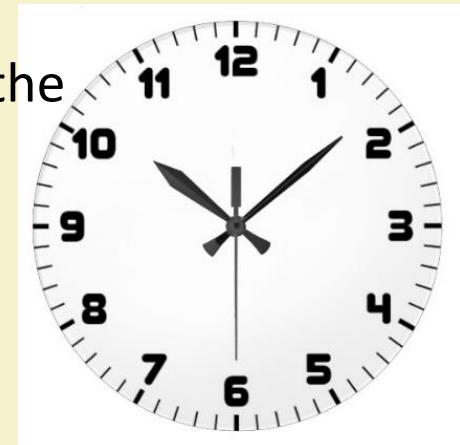


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

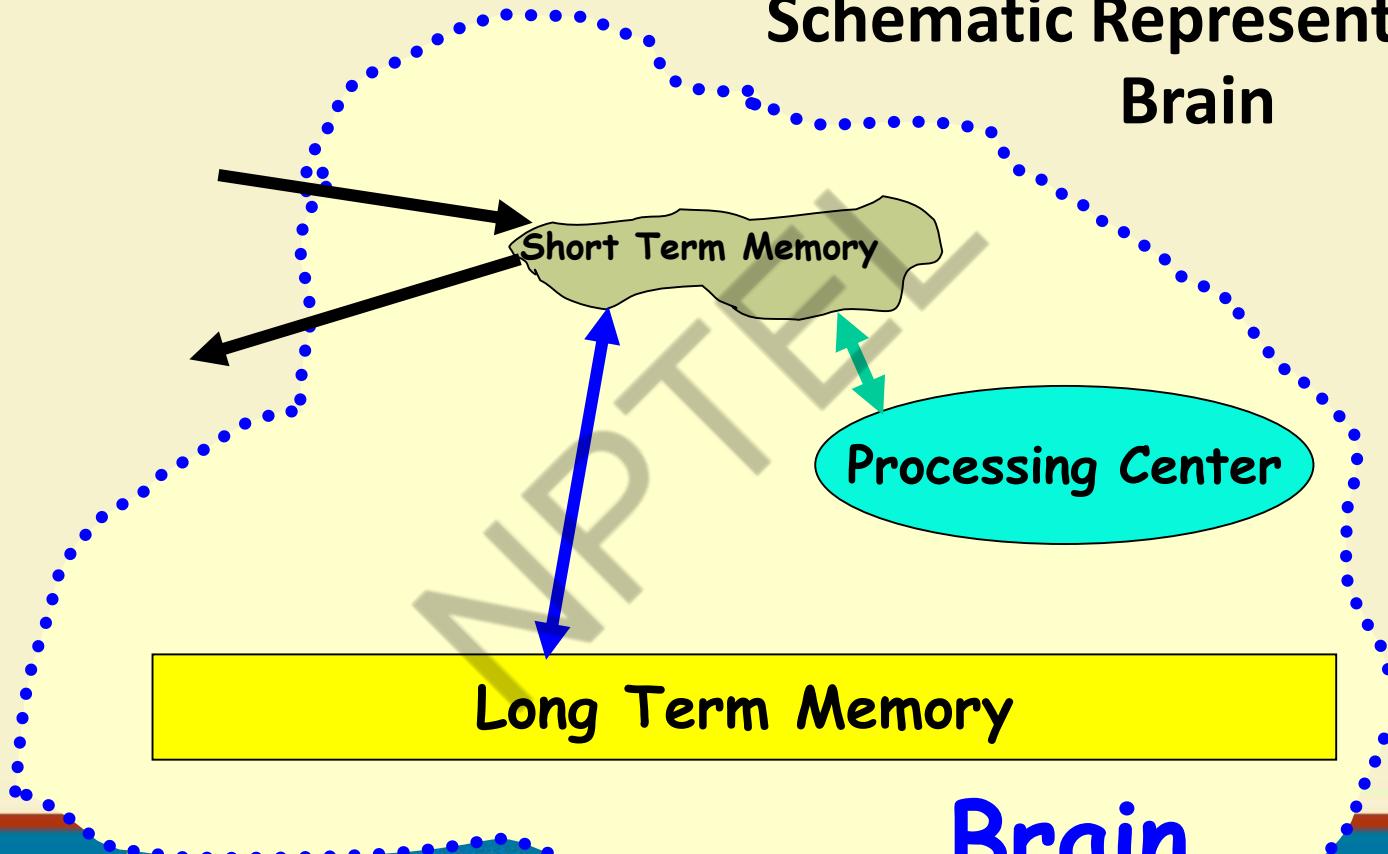
- Suppose I ask: “**It is 10:10AM now, how many hours are remaining today?**”

## Human Cognition Mechanism

- 10AM would be stored in the short-term memory.
- “A day is 24 hours long.” would be fetched from the long term memory into short term memory.
- The mental manipulation unit would compute the difference (24-10).



# Schematic Representation of Brain



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

Brain

## Short Term Memory

- An item stored in the short term memory can get lost:
  - Either due to decay with time or
  - Displacement by newer information.
- This restricts the time for which an item is stored in short term memory:
  - Typically few tens of seconds.
  - However, an item can be retained longer in the short term memory by recycling.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- **An item is any set of related information.**
  - A character such as `a' or a digit such as `5'.
  - A word, a sentence, a story, or even a picture.
- Each item normally occupies one place in memory.
- When you are able to relate several different items together (**chunking**):
  - The information that should normally occupy several places, takes only one place in memory.

## What is an Item?

# Chunking

- If I ask you to remember the number **110010101001**
  - It may prove very hard for you to understand and remember.
  - But, the octal form of 6251 (**110)(010)(101)(001**) would be easier.
  - You have managed to create chunks of three items each.

- In many of our day-to-day experiences:
  - **Short term memory is evident.**
- Suppose, you look up a number from the telephone directory and start dialling it.
  - If you find the number is busy, you can dial the number again after a few seconds without having to look up the number from directory.
- But, after several days:
  - You may not remember the number at all
  - Would need to consult the directory again.

## Evidence of Short Term Memory

- If a person deals with seven or less number of items:
    - These would be accommodated in the short term memory.
    - So, he can easily understand it.
  - As the number of new information increases beyond seven:
    - It becomes exceedingly difficult to understand it.
- The Magical Number 7**

## What is the Implication in Program Development?

- A small program having just a few variables:
  - Is within easy grasp of an individual.
- As the number of independent variables in the program increases:
  - **It quickly exceeds the grasping power of an individual...**
  - **Requires an unduly large effort to master the problem.**

# Implication in Program Development

- Instead of a human, if a machine could be writing (generating) a program,
  - The slope of the curve would be linear.
- But, how does use of software engineering principles help hold down the effort-size curve to be almost linear?
  - **Software engineering principles extensively use techniques specifically targeted to overcome the human cognitive limitations.**

# Which Principles are Deployed by Software Engineering Techniques to Overcome Human Cognitive Limitations?

- Two important principles are profusely used:
  - **Abstraction**
  - **Decomposition**

# **Two Fundamental Techniques to Handle Complexity**



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# What is Abstraction?

- Simplify a problem by omitting unnecessary details.
  - Focus attention on only one aspect of the problem and ignore other aspects and irrelevant details.
  - Also called model building.



IIT KHARAGPUR



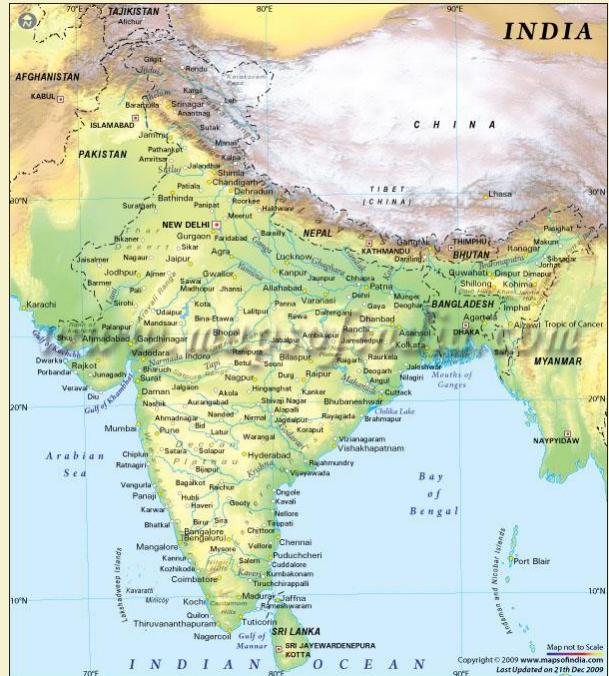
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Suppose you are asked to develop an overall understanding of some country.
  - Would you:
    - Meet all the citizens of the country, visit every house, and examine every tree of the country?
    - You would possibly refer to various types of maps for that country only.

## Abstraction Example

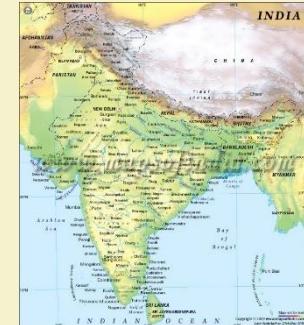
# You would study an Abstraction...

- A map is:
  - An abstract representation of a country.
  - Various types of maps (abstractions) possible.



# Does every Problem have a single Abstraction?

- Several abstractions of the same problem can be created:



- Focus on some specific aspect and ignore the rest.
- Different types of models help understand different aspects of the problem.



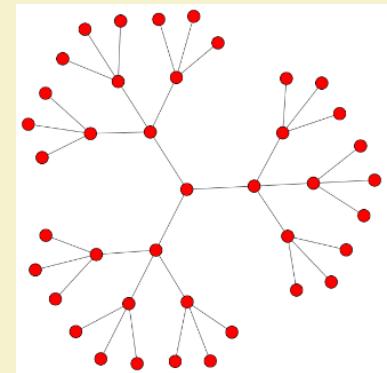
IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Abstractions of Complex Problems

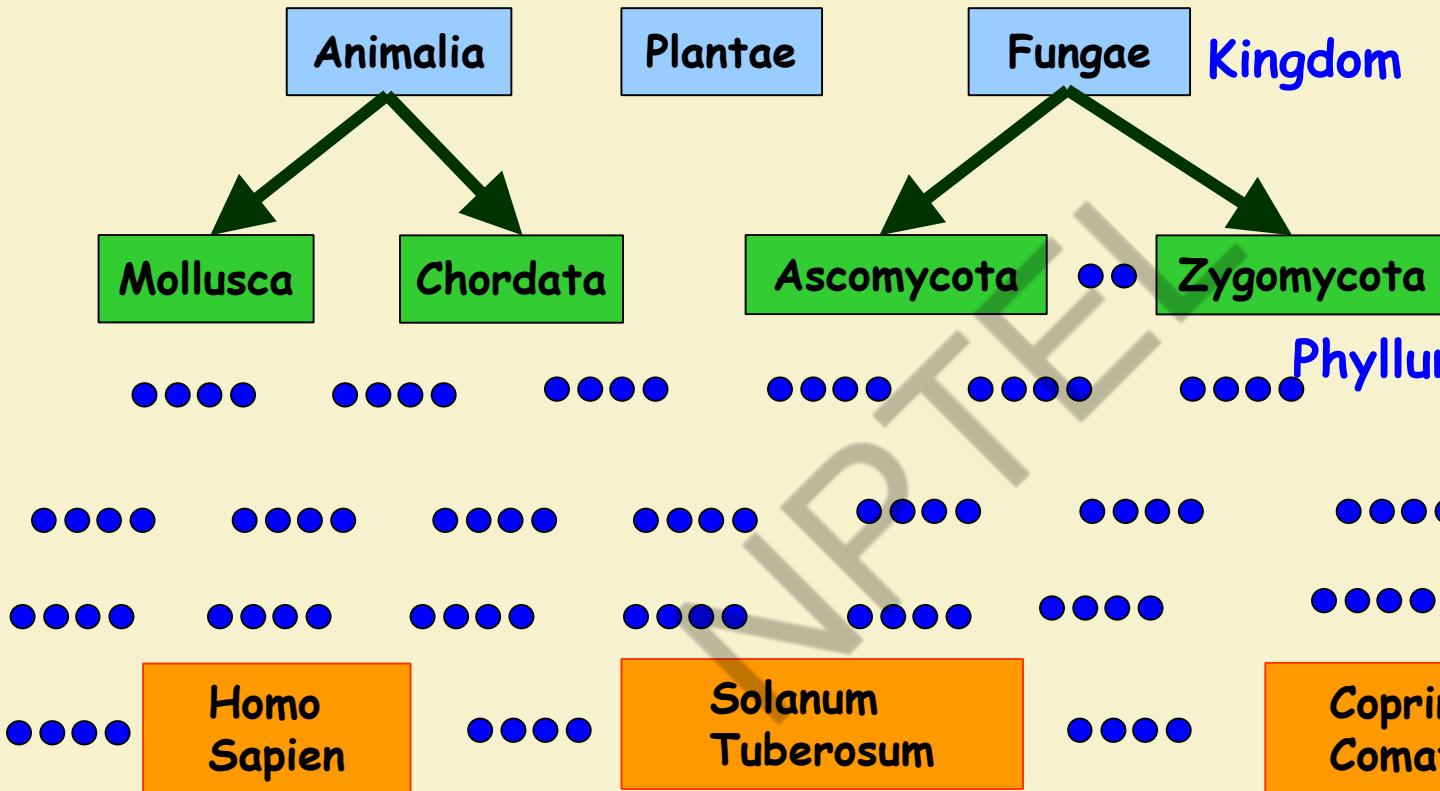
- For complex problems:
  - A single level of abstraction is inadequate.
  - A hierarchy of abstractions may have to be constructed.
- Hierarchy of models:
  - A model in one layer is an abstraction of the lower layer model.
  - An implementation of the model at the higher layer.



# Abstraction of Complex Problems -- An Example

- Suppose you are asked to understand all life forms that inhabit the earth.
- Would you start examining each living organism?
  - You will almost never complete it.
  - Also, get thoroughly confused.
- **Solution: Try to build an abstraction hierarchy.**

# Living Organisms



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Quiz

- What is a model?
- Why develop a model? That is, how does constructing a model help?
- Give some examples of models.

# Decomposition

- Decompose a problem into many small independent parts.
  - The small parts are then taken up one by one and solved separately.
  - **The idea is that each small part would be easy to grasp and therefore can be easily solved.**
  - **The full problem is solved when all the parts are solved.**



# Decomposition

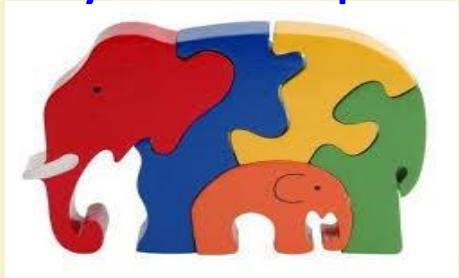
- A popular example of decomposition principle:

- Try to break a bunch of sticks tied together versus breaking them individually.



- Any arbitrary decomposition of a problem may not help.

- The decomposed parts must be more or less independent of each other.

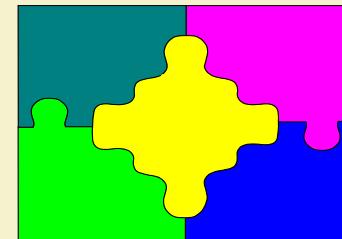


# Decomposition: Another Example

- Example use of decomposition principle:
  - You understand a book better when the contents are organized into independent chapters.
  - Compared to when everything is mixed up.

# Why Study Software Engineering? (1)

- To acquire skills to develop large programs.
  - Handling exponential growth in complexity with size.
  - Systematic techniques based on abstraction (modelling) and decomposition.

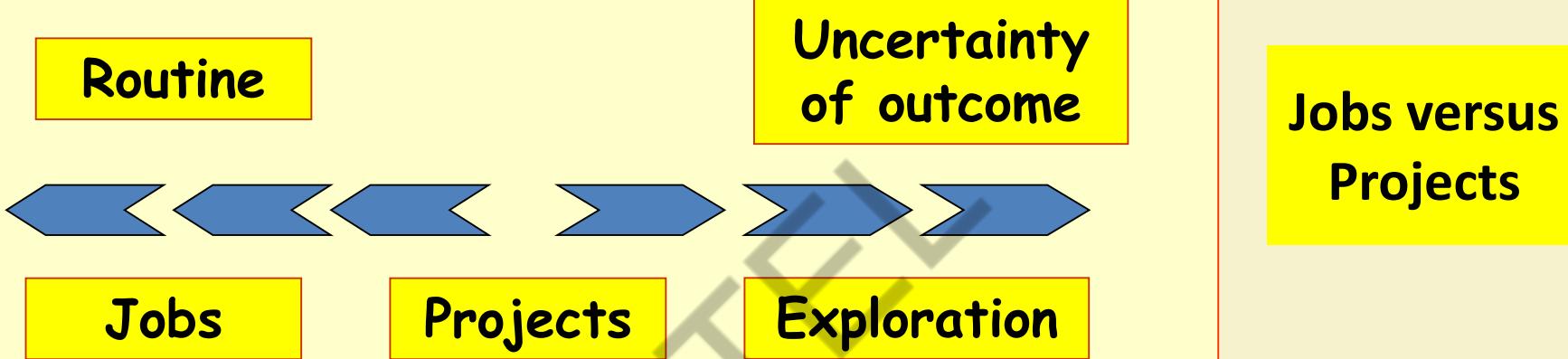


## Why Study Software Engineering? (2)

- Learn systematic techniques of:
  - Specification, design, user interface development, testing, project management, maintenance, etc.
  - Appreciate issues that arise in team development.

## Why Study Software Engineering? (3)

- To acquire skills to be a better programmer:
  - Higher Productivity
  - Better Quality Programs



**Jobs** – repetition of very well-defined and well understood tasks with very little uncertainty

**Exploration** – The outcome is very uncertain, e.g. finding a cure for cancer.

**Projects** – in the middle! Has challenge as well as routine...

## Types of Software Projects

- Two types of software projects:

- Products (Generic software)**
  - Services (custom software)**

- Total business – Several Trillions of US \$
  - Half in products and half services
  - Services segment is growing fast!**

**Packaged software** –  
prewritten software available for purchase

**Custom software** –  
software developed at some user's requests-Usually developer tailors some generic solution

Horizontal market software—meets needs of many companies

Vertical market software—designed for particular industry

## Types of Software



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Types of Software Projects

- Software product development projects
- Software services projects

# Software Services

- Software service is an umbrella term, includes:
  - Software customization
  - Software maintenance
  - Software testing
  - Also contract programmers (CP) carrying out coding or any other assigned activities.



# **Factors responsible for accelerated growth of services...**

- Now lots of code is available in a company:
  - New software can be developed by modifying the closest.
- Speed of Conducting Business has increased tremendously:
  - Requires shortening of project duration



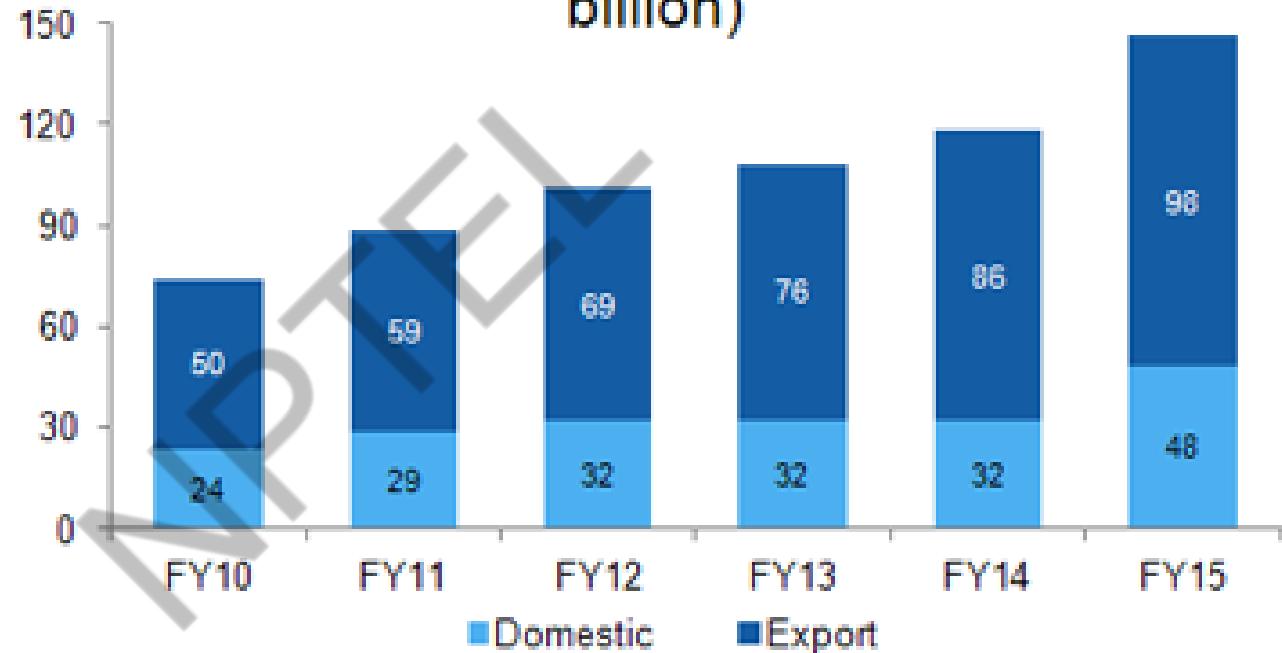
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

**Contribution of the IT sector to India's GDP rose to approximately 9.5% in 2015 from 1.2% in 98**

## Market size of IT industry in India (US\$ billion)



Source: Nasscom, TechSci Research; Note: E - Estimates



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Scenario of Indian Software Companies

- Indian companies have largely focused on the services segment --
  - Why?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# A Few Changes in Software Project Characteristics over Last 40 Years

- 40 years back, very few software existed
  - Every project started from scratch
  - Projects were multi year long
- The programming languages that were used earlier hardly provided any scope for reuse:
  - FORTRAN, PASCAL, COBOL, BASIC
- No application was GUI-based:
  - Mostly command selection from displayed text menu items.

# Traditional versus Modern Projects

- Projects are increasingly becoming services:
  - Either tailor some existing software or reuse pre-built libraries.
- Facilitate and accommodate client feedbacks
- Facilitate customer participation in project development work
- Incremental software delivery with evolving functionalities.
- **No software is being developed from scratch --- Significant reuse is being made...**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Computer Systems Engineering

- Many products require development of software as well as specific hardware to run it:
  - a coffee vending machine,
  - a robotic toy,
  - A new health band product, etc.
- Computer systems engineering:
  - encompasses software engineering.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

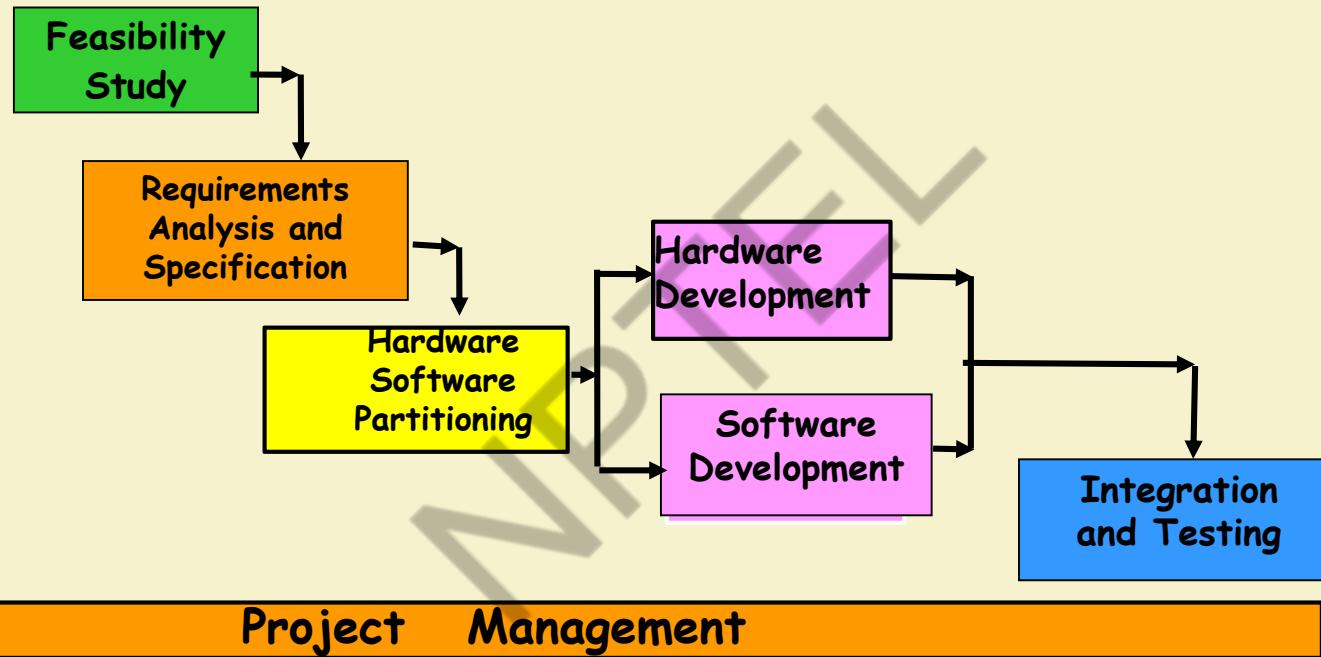
# Computer Systems Engineering

- The high-level problem:
  - Deciding which tasks are to be solved by software.
  - Which ones by hardware.

# Computer Systems Engineering (CONT.)

- Typically, hardware and software are developed together:
  - Hardware simulator is used during software development.
- Integration of hardware and software.
- Final system testing

# Computer Systems Engineering (CONT.)



# Emergence of Software Engineering Techniques



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Emergence of Software Engineering Techniques

- Early Computer Programming (1950s):
  - Programs were being written in assembly language...
  - Sizes limited to about a few hundreds of lines of assembly code...

# Early Computer Programming (50s)

- Every programmer developed his/her own style of writing programs:
  - According to his intuition (called exploratory or build-and-fix programming) .



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# High-Level Language Programming (Early 60s)

- High-level languages such as FORTRAN, ALGOL, and COBOL were introduced:
  - This reduced software development efforts greatly.
  - Why reduces?

# High-Level Language Programming (Early 60s)

- **Software development style was still exploratory.**
  - Typical program sizes were limited to a few thousands of lines of source code.

## Control Flow-Based Design (late 60s)

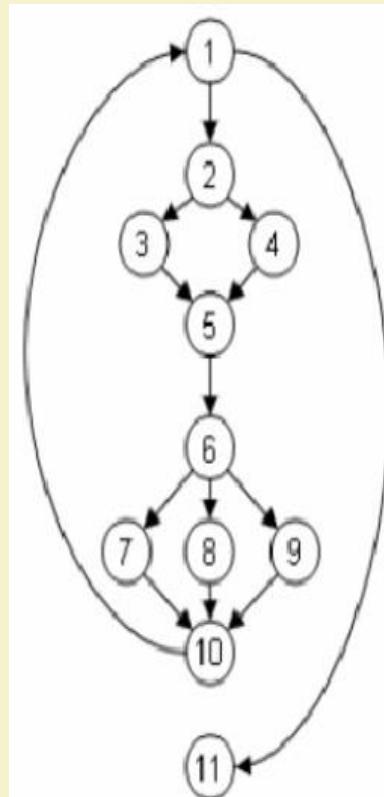
- Size and complexity of programs increased further:
  - Exploratory programming style proved to be insufficient.
- Programmers found:
  - Very difficult to write cost-effective and correct programs.

# Control Flow-Based Design (late 60s)

- Programmers found it very difficult:
  - To understand and maintain programs written by others.
- To cope up with this problem, experienced programmers advised--"**Pay particular attention to the design of the program's control structure.**"

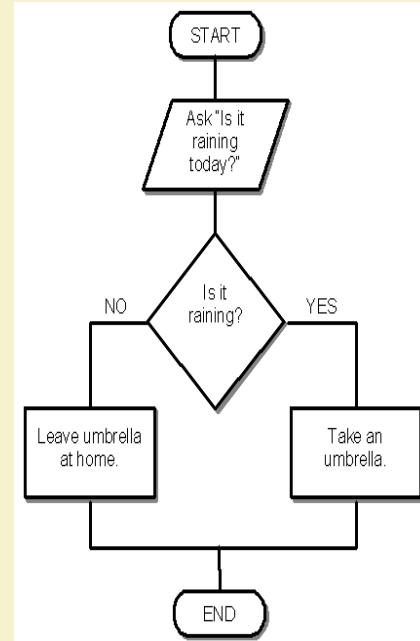
## Control Flow-Based Design (late 60s)

- What is a program's control structure?
  - The sequence in which the program's instructions are executed.
- To help design programs having good control structure:
  - Flow charting technique was developed.



# Control Flow-Based Design (late 60s)

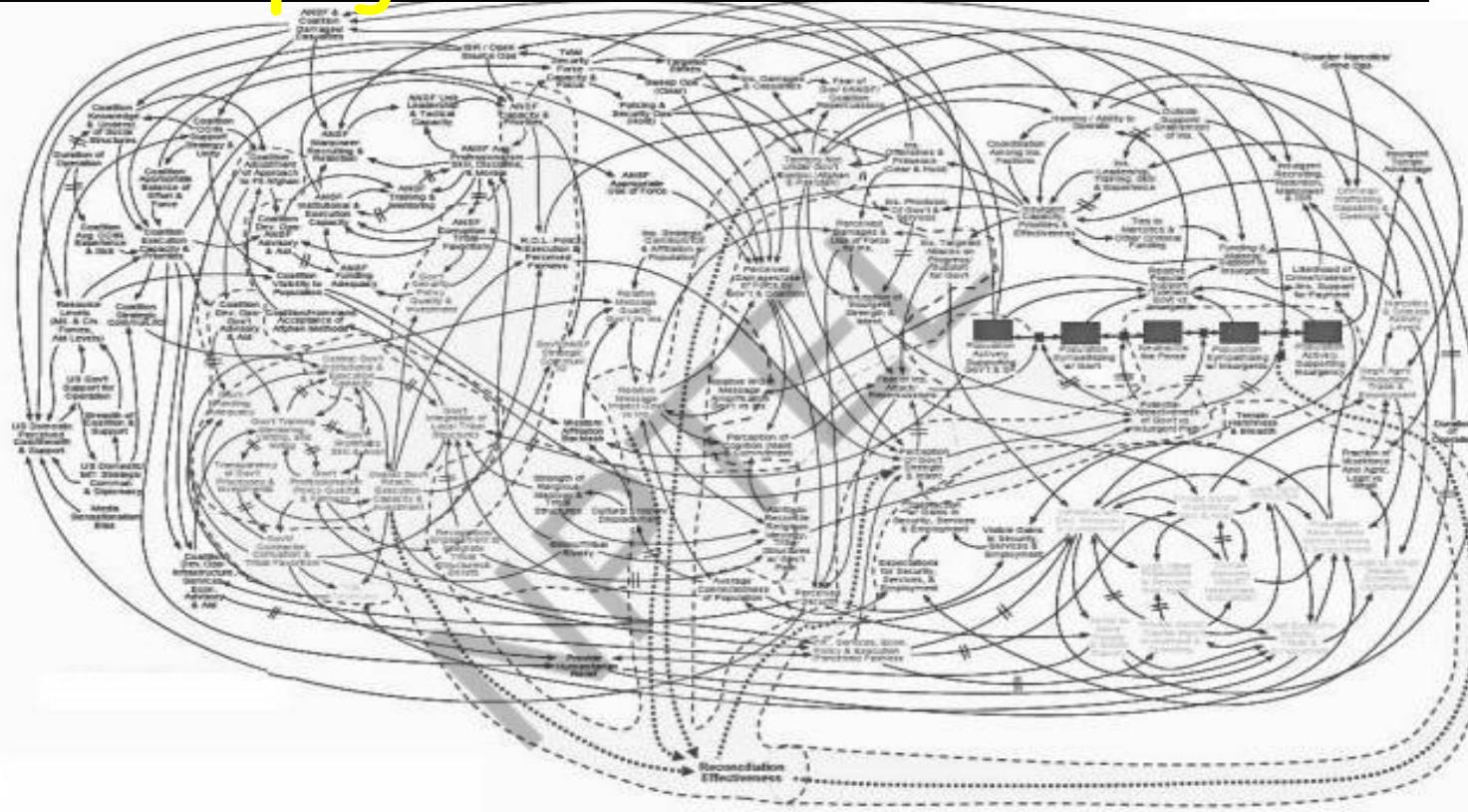
- Using flow charting technique:
  - One can represent and design a program's control structure.
  - When asked to understand a program:
    - One would mentally trace the program's execution sequence.



# Control Flow-Based Design

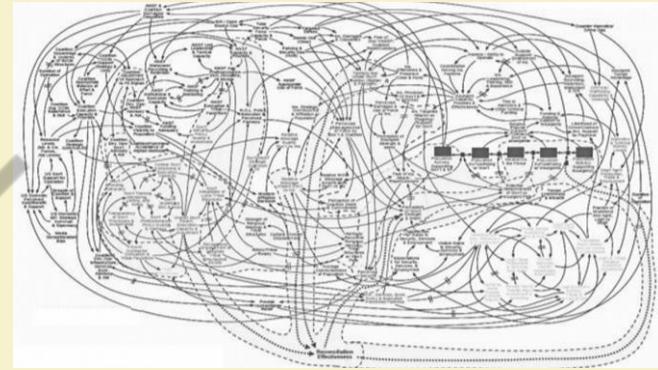
- A program having a messy flow chart representation:
  - Difficult to understand and debug.

# Spaghetti Code Structure



# Control Flow-Based Design (Late 60s)

- What causes program complexity?
  - GO TO statements makes control structure of a program messy.
  - GO TO statements alter the flow of control arbitrarily.
  - The need to restrict use of GO TO statements was recognized.



# Control Flow-Based Design

(Late 60s)

- Many programmers had extensively used assembly languages.
  - JUMP instructions are frequently used for program branching in assembly languages.
  - Programmers considered use of GO TO statements inevitable.

```
addi $a0, $0, 1
j next
next:
j skip1
add $a0, $a0, $a0
skip1:
j skip2
add $a0, $a0, $a0
add $a0, $a0, $a0
skip2:
j skip3
loop:
add $a0, $a0, $a0
add $a0, $a0, $a0
add $a0, $a0, $a0
skip3:
j loop
```



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Control-flow Based Design (Late 60s)

- At that time, Dijkstra published his article:
  - “Goto Statement Considered Harmful” Comm. of ACM, 1969.
- Many programmers were unhappy to read his article.

# Control Flow-Based Design

(Late 60s)

- Some programmers published several counter articles:
  - Highlighted the advantages and inevitability of GO TO statements.

# Control Flow-Based Design

(Late 60s)

- It soon was conclusively proved:
  - Only three programming constructs are sufficient to express any programming logic:
    - **sequence** (`a=0;b=5;`)
    - **selection** (`if(c==true) k=5 else m=5;`)
    - **iteration** (`while(k>0) k=j-k;`)

# Control-flow Based Design (Late 60s)

- Everyone accepted:
  - It is possible to solve any programming problem without using GO TO statements.
  - This formed the basis of **Structured Programming methodology**.

# Structured Programming

- A program is called structured:
  - When it uses only the following types of constructs:
    - **sequence**,
    - **selection**,
    - **iteration**
  - Consists of **modules**.

# Structured Programs

- Sometimes, violations to structured programming are permitted:
  - Due to practical considerations such as:
  - Premature loop exit (break) or for exception handling.

# Advantages of Structured programming

- Structured programs are:
  - Easier to read and understand,
  - Easier to maintain,
  - Require less effort and time for development.
  - Less buggy



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Structured Programming

- Research experience shows:
  - Programmers commit less number of errors:
    - While using structured **if-then-else** and **do-while** statements.
    - Compared to **test-and-branch** (**GOTO**) constructs.

# Data Structure-Oriented Design (Early 70s)

- As program sizes increased further, soon it was discovered:
  - **It is important to pay more attention to the design of data structures of a program**
  - Than to the design of its control structure.

# Data Structure-Oriented Design (Early 70s)

- Techniques which emphasize designing the data structure:
  - Derive program structure from it:
- Are called **data structure-oriented design techniques**.

# Data Structure Oriented Design (Early 70s)

- An example of data structure-oriented design technique:

—Jackson's Structured Programming(JSP) methodology

- Developed by Michael Jackson in 1970s.

# Data Structure Oriented Design (Early 70s)

- **JSP technique:**
  - Program code structure should correspond to the data structure.

## ●JSP methodology:

# A Data Structure Oriented Design

(Early 70s)

- A program's data structures are first designed using notations for
  - sequence, selection, and iteration.
- The data structure design is then used :
  - To derive the program structure.

# Data Structure Oriented Design (Early 70s)

- Several other data structure-oriented Methodologies also exist:
  - e.g., [Warnier-Orr Methodology](#).

# Data Flow-Oriented Design (Late 70s)

- Data flow-oriented techniques advocate:
  - The data items input to a system must first be identified,
  - Processing required on the data items to produce the required outputs should be determined.

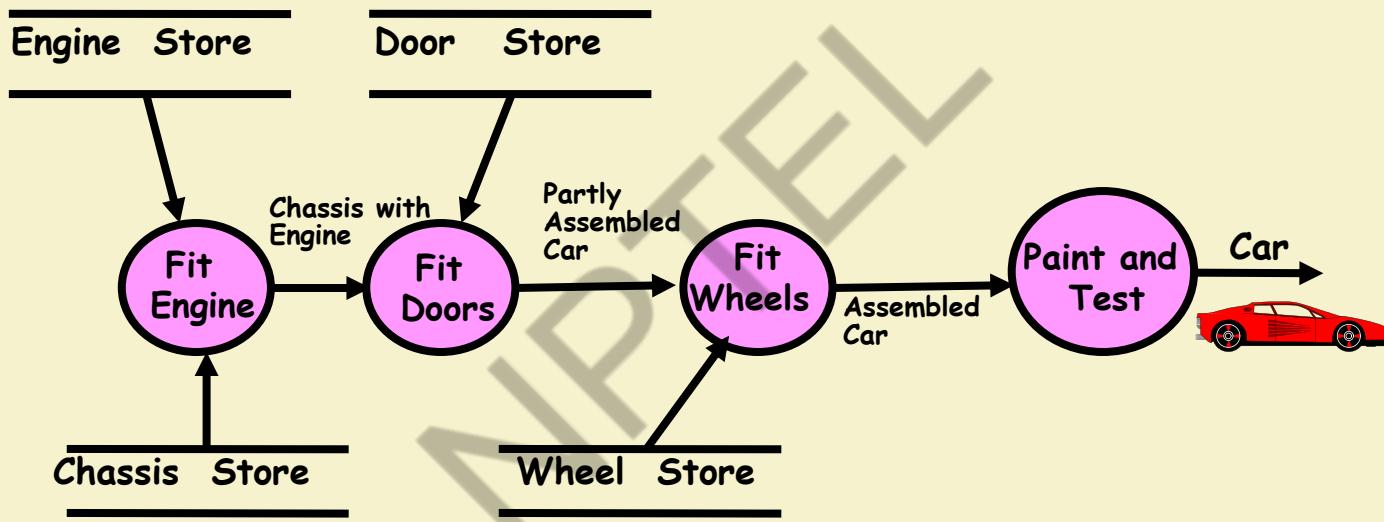
# Data Flow-Oriented Design (Late 70s)

- Data flow technique identifies:
  - Different processing stations (functions) in a system.
  - The items (data) that flow between processing stations.

# Data Flow-Oriented Design (Late 70s)

- Data flow technique is a generic technique:
  - Can be used to model the working of any system.
    - not just software systems.
- A major advantage of the data flow technique is its simplicity.

# Data Flow Model of a Car Assembly Unit



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented Design (80s)

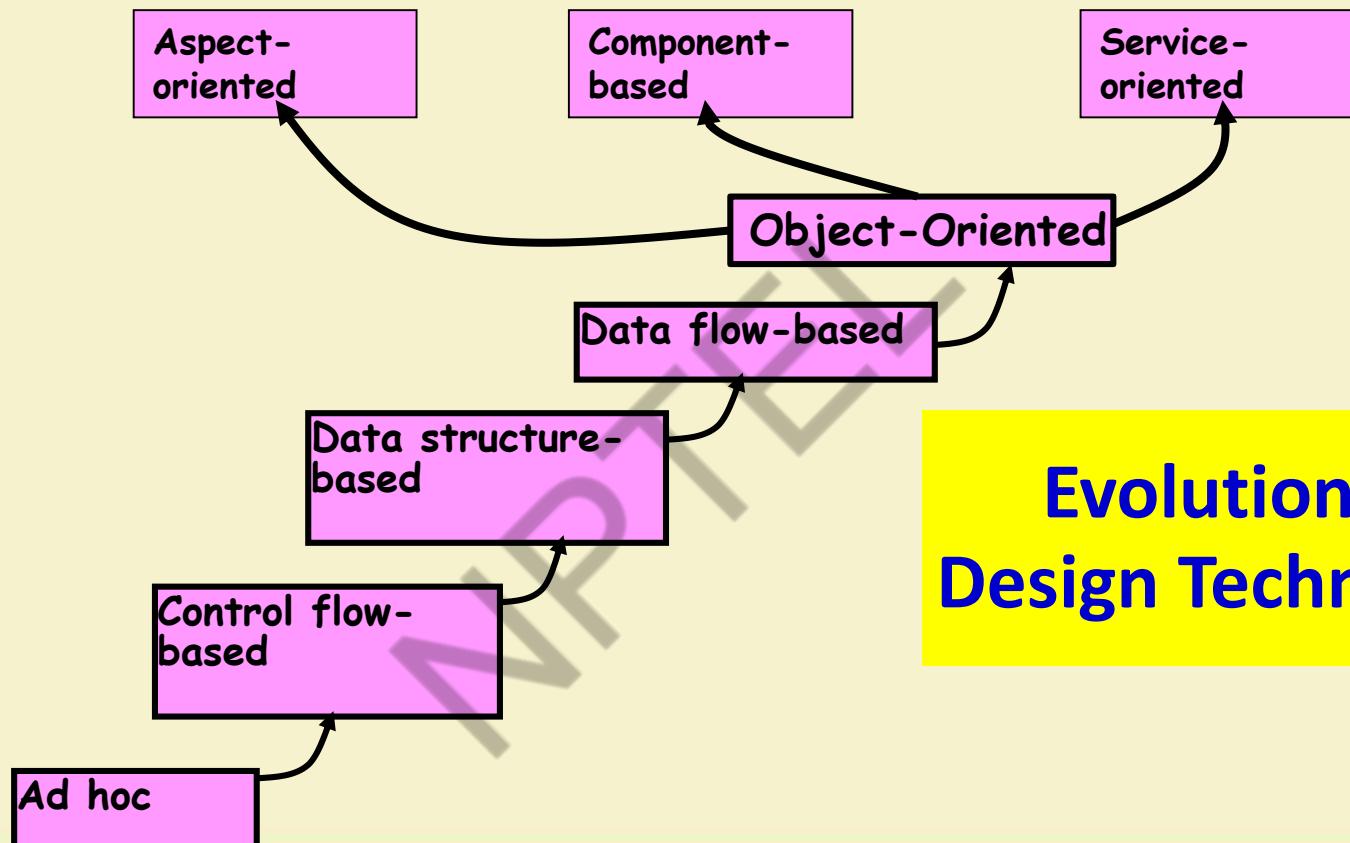
- Object-oriented technique:
  - An intuitively appealing design approach:
  - Natural objects (such as employees, pay-roll-register, etc.) occurring in a problem are first identified.

## Object-Oriented Design (80s)

- Relationships among objects:
  - Such as composition, reference, and inheritance are determined.
- Each object essentially acts as:
  - A data hiding (or data abstraction) entity.

- Object-Oriented Techniques have gained wide acceptance:
  - Simplicity
  - Increased Reuse possibilities
  - Lower development time and cost
  - More robust code
  - Easy maintenance

## Object-Oriented Design (80s)



## Evolution of Design Techniques



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Evolution of Other Software Engineering Techniques

- The improvements to the software design methodologies
  - are indeed very conspicuous.
- In addition to the software design techniques:
  - Several other techniques evolved.

- Life cycle models,
- Specification techniques,
- Project management techniques,
- Testing techniques,
- Debugging techniques,
- Quality assurance techniques,
- Metrics,
- CASE tools, etc.

## Evolution of Other Software Engineering Techniques

- Use of Life Cycle Models
- Software is developed through several well-defined stages:
  - Requirements analysis and specification,
  - Design,
  - Coding,
  - Testing, etc.

**Differences between the exploratory style and modern software development practices**

## Differences between the exploratory style and modern software development practices

- Emphasis has shifted
  - from error correction to error prevention.
- Modern practices emphasize:
  - detection of errors as close to their point of introduction as possible.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Differences between the exploratory style and modern software development practices (CONT.)

- In exploratory style,
  - errors are detected only during testing,
- Now:
  - Focus is on detecting as many errors as possible in each phase of development.

## Differences between the exploratory style and modern software development practices (CONT.)

- In exploratory style:
  - coding is synonymous with program development.
- Now:
  - coding is considered only a small part of program development effort.

## Differences between the exploratory style and modern software development practices (CONT.)

- A lot of effort and attention is now being paid to:
  - Requirements specification.
- Also, now there is a distinct design phase:
  - Standard design techniques are being used.

## **Differences between the exploratory style and modern software development practices (CONT.)**

- During all stages of development process:
  - Periodic reviews are being carried out
- Software testing has become systematic:
  - Standard testing techniques are available.

## Differences between the exploratory style and modern software development practices (CONT.)

- There is better visibility of design and code:
  - **Visibility means production of good quality, consistent and standard documents.**
  - In the past, very little attention was being given to producing good quality and consistent documents.
  - We will see later that increased visibility makes software project management easier.

## Differences between the exploratory style and modern software development practices (CONT.)

- Because of good documentation:
  - fault diagnosis and maintenance are smoother now.
- Several metrics are being used:
  - help in software project management, quality assurance, etc.

## Differences between the exploratory style and modern software development practices (CONT.)

- Projects are being properly planned:
  - estimation,
  - scheduling,
  - monitoring mechanisms.
- Use of CASE tools.

# Review Questions

- What is structured programming?
- What problems may appear if a large program is developed without using structured programming techniques?

# Life Cycle Models



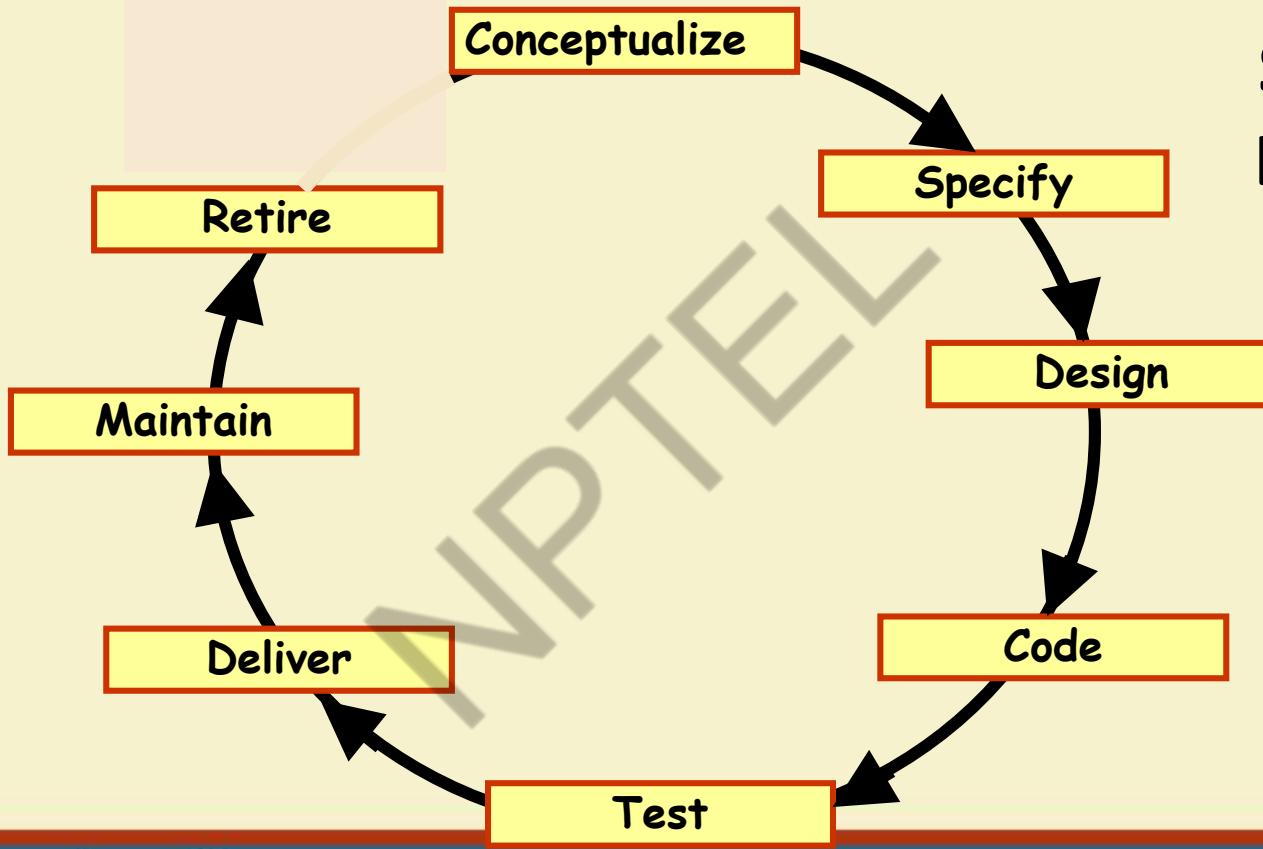
IIT KHARAGPUR



NPTEL

NPTEL ONLINE  
CERTIFICATION COURSES

# Software Life Cycle



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Life Cycle Model

- A software life cycle model (also process model or SDLC):
  - A descriptive and diagrammatic model of software life cycle:
  - Identifies all the activities undertaken during product development,
  - Establishes a precedence ordering among the different activities,
  - Divides life cycle into phases.

# Life Cycle Model (CONT.)

- Each life cycle phase consists of several activities.
  - For example, the design stage might consist of:
    - structured analysis
    - structured design
    - Design review



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Why Model Life Cycle?

- A graphical and written description:
  - Helps common understanding of activities among the software developers.
  - Helps to identify inconsistencies, redundancies, and omissions in the development process.
  - Helps in tailoring a process model for specific projects.



IIT KHARAGPUR



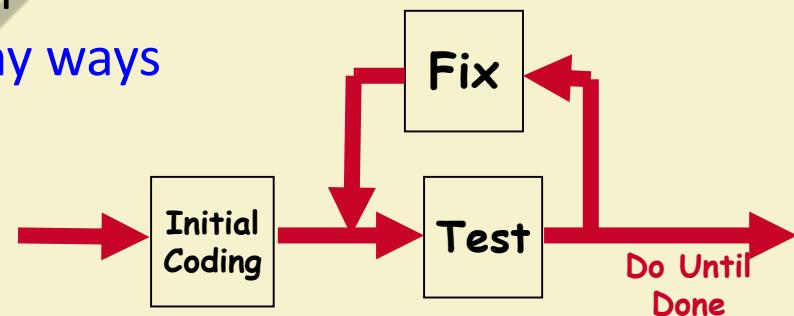
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Life Cycle Model (CONT.)

- The development team must identify a suitable life cycle model:
  - and then adhere to it.
  - Primary advantage of adhering to a life cycle model:
    - Helps development of software in a systematic and disciplined manner.

# Life Cycle Model (CONT.)

- When a program is developed by a single programmer ---
  - The problem is within the grasp of an individual.
  - He has the freedom to decide his exact steps and still succeed --- called Exploratory model--- One can use it in many ways
  - Code → Test → Design
  - Code → Design → Test → Change Code →
  - Specify → code → Design → Test → etc.



# Life Cycle Model (CONT.)

- When software is being developed by a team:
  - There must be a precise understanding among team members as to when to do what,
  - Otherwise, it would lead to chaos and project failure.

## Life Cycle Model (CONT.)

- A software project will never succeed if:
  - one engineer starts writing code,
  - another concentrates on writing the test document first,
  - yet another engineer first defines the file structure
  - another defines the I/O for his portion first.

# Phase Entry and Exit Criteria

- A life cycle model:
  - defines entry and exit criteria for every phase.
  - A phase is considered to be complete:
    - only when all its exit criteria are satisfied.



# Life Cycle Model (CONT.)

- What is the phase exit criteria for the software requirements specification phase?
  - Software Requirements Specification (SRS) document is complete, reviewed, and approved by the customer.
- A phase can start:
  - Only if its phase-entry criteria have been satisfied.

# Life Cycle Model: Milestones

- Milestones help software project managers:
  - Track the progress of the project.
  - Phase entry and exit are important milestones.



# Life Cycle and Project Management

- When a life cycle model is followed:
  - The project manager can at any time fairly accurately tell,
    - At which stage (e.g., design, code, test, etc.) the project is.

## Project Management Without Life Cycle Model

- It becomes very difficult to track the progress of the project.
  - The project manager would have to depend on the guesses of the team members.
- This usually leads to a problem:
  - known as the **99% complete syndrome**.

# Project Deliverables: Myth and Reality

## Myth:

The only deliverable for a successful project is the working program.

## Reality:

Documentation of **all aspects** of software development are needed to help in operation and maintenance.

- Many life cycle models have been proposed.
- We confine our attention to only a few commonly used models.

- Waterfall
- V model,
- Evolutionary,
- Prototyping
- Spiral model,
- Agile models

Traditional models

**Life Cycle Model** (CONT.)

- Software life cycle (or software process):
  - **Series of identifiable stages that a software product undergoes during its life time:**
  - Feasibility study
  - Requirements analysis and specification,
  - Design,
  - Coding,
  - Testing
  - Maintenance.

## Software Life Cycle



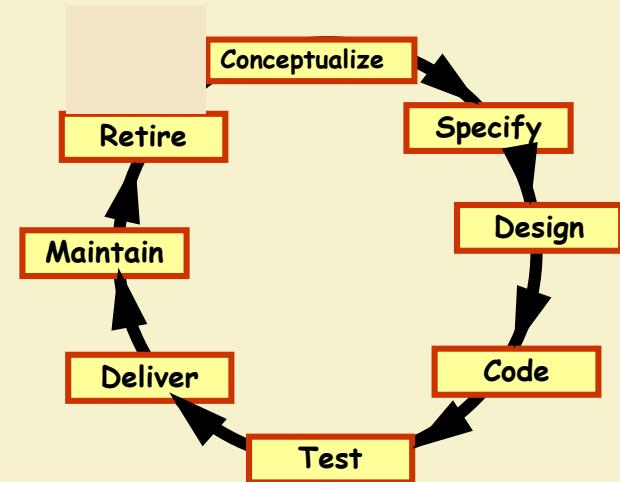
IIT KHARAGPUR



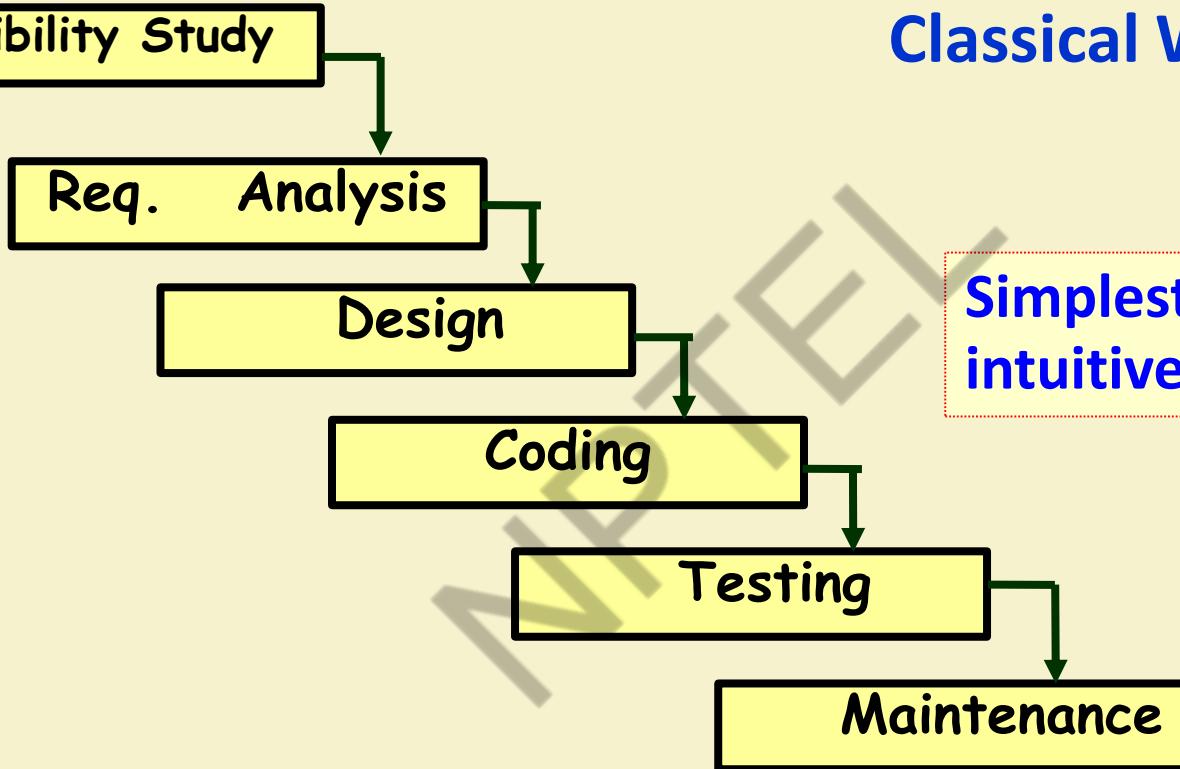
NPTEL  
ONLINE  
CERTIFICATION COURSES

# Classical Waterfall Model

- Classical waterfall model divides life cycle into following phases:
  - Feasibility study,
  - Requirements analysis and specification,
  - Design,
  - Coding and unit testing,
  - Integration and system testing,
  - Maintenance.



# Classical Waterfall Model



Simplest and most intuitive



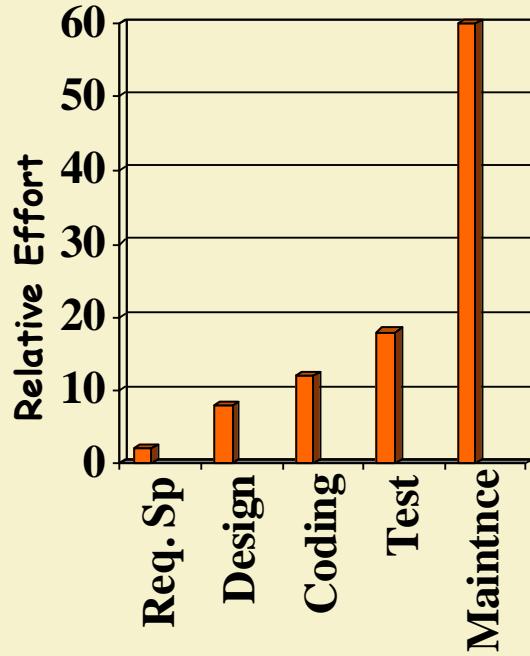
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Relative Effort for Phases

- Phases between feasibility study and testing
  - Called **development phases**.
- Among all life cycle phases
  - Maintenance phase consumes maximum effort.
- Among development phases,
  - Testing phase consumes the maximum effort.



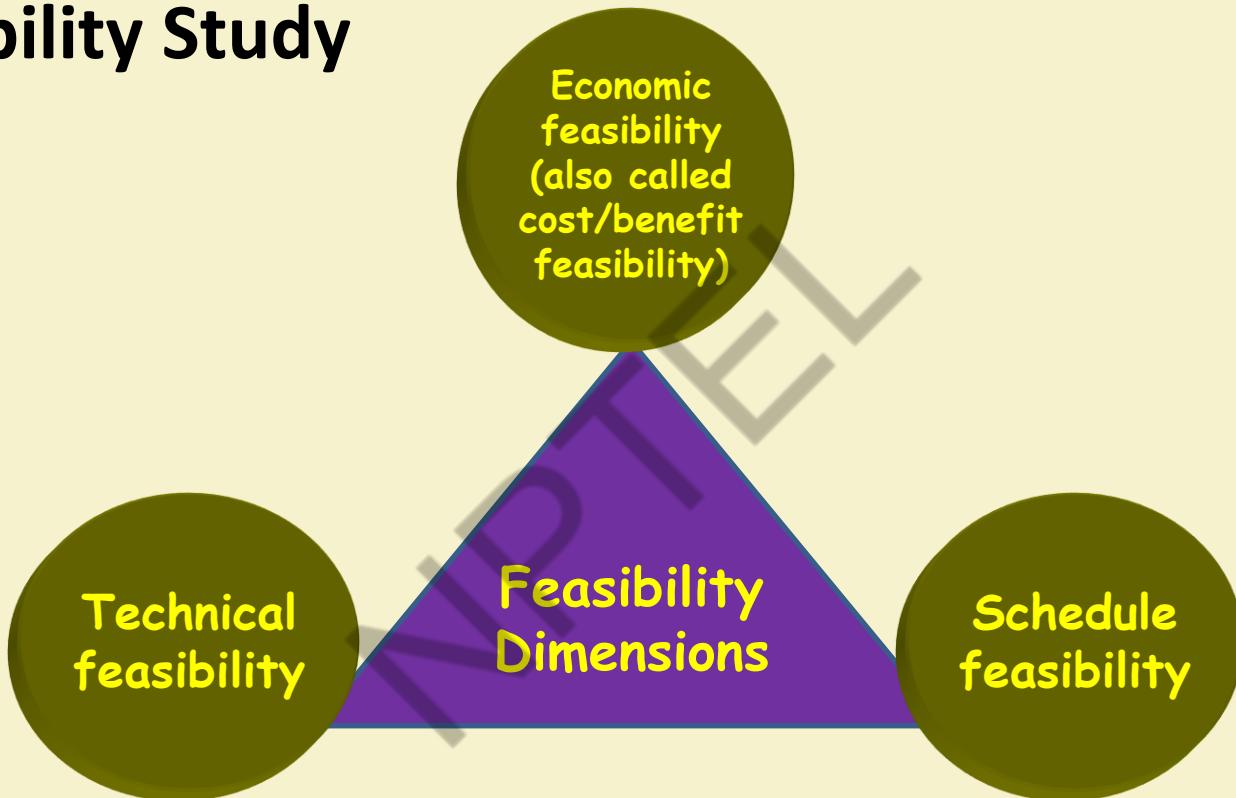
- Most organizations usually define:
  - Standards on the outputs (deliverables) produced at the end of every phase
  - Entry and exit criteria for every phase.
- They also prescribe methodologies for:
  - Specification,
  - Design,
  - Testing,
  - Project management, etc.

## Process Model

# Classical Waterfall Model (CONT.)

- The guidelines and methodologies of an organization:
  - Called the organization's software development methodology.
- Software development organizations:
  - Expect fresh engineers to master the organization's software development methodology.

# Feasibility Study



- Main aim of feasibility study: determine whether developing the software is:
  - Financially worthwhile
  - Technically feasible.
- Roughly understand what customer wants:
  - Data which would be input to the system,
  - Processing needed on these data,
  - Output data to be produced by the system,
  - Various constraints on the behavior of the system.

## Feasibility Study

## First Step

- SPF Scheme for CFL
- CFL has a large number of employees, exceeding 50,000.
- Majority of these are casual labourers
- Mining being a risky profession:
  - Casualties are high
- Though there is a PF:
  - But settlement time is high
- There is a need of SPF:
  - For faster disbursement of benefits

## Case Study

# Feasibility: Case Study

- Manager visits main office, finds out the main functionalities required
- Visits mine site, finds out the data to be input
- Suggests alternate solutions
- Determines the best solution
- Presents to the CFL Officials
- Go/No-Go Decision



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Activities During Feasibility Study

- Work out an overall understanding of the problem.
- Formulate different solution strategies.
- Examine alternate solution strategies in terms of:
  - resources required,
  - cost of development, and
  - development time.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Perform a cost/benefit analysis:
  - Determine which solution is the best.
  - May also find that none of the solutions is feasible due to:
    - high cost,
    - resource constraints,
    - technical reasons.

### Activities during Feasibility Study



IIT KHARAGPUR



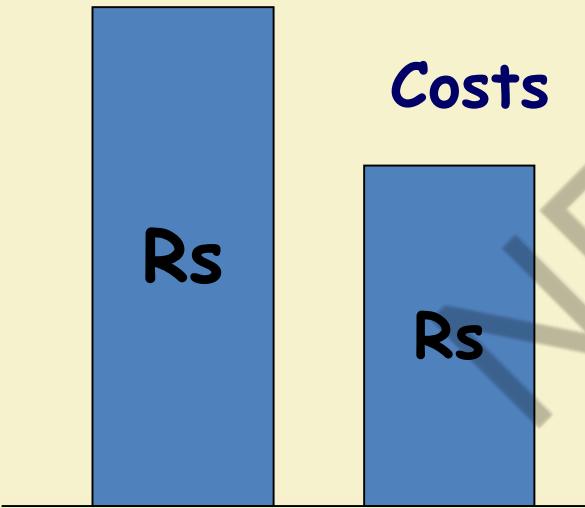
NPTEL ONLINE  
CERTIFICATION COURSES

# Cost benefit analysis (CBA)

- Need to identify all costs --- these could be:
  - Development costs
  - Set-up
  - Operational costs
- Identify the value of benefits
- Check benefits are greater than costs

# The business case

Benefits



- Benefits of delivered project must outweigh costs
- Costs include:
  - Development
  - Operation
- Benefits:
  - Quantifiable
  - Non-quantifiable



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Thank You !!

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

Rajib Mall  
CSE Department

134

# Life Cycle Models

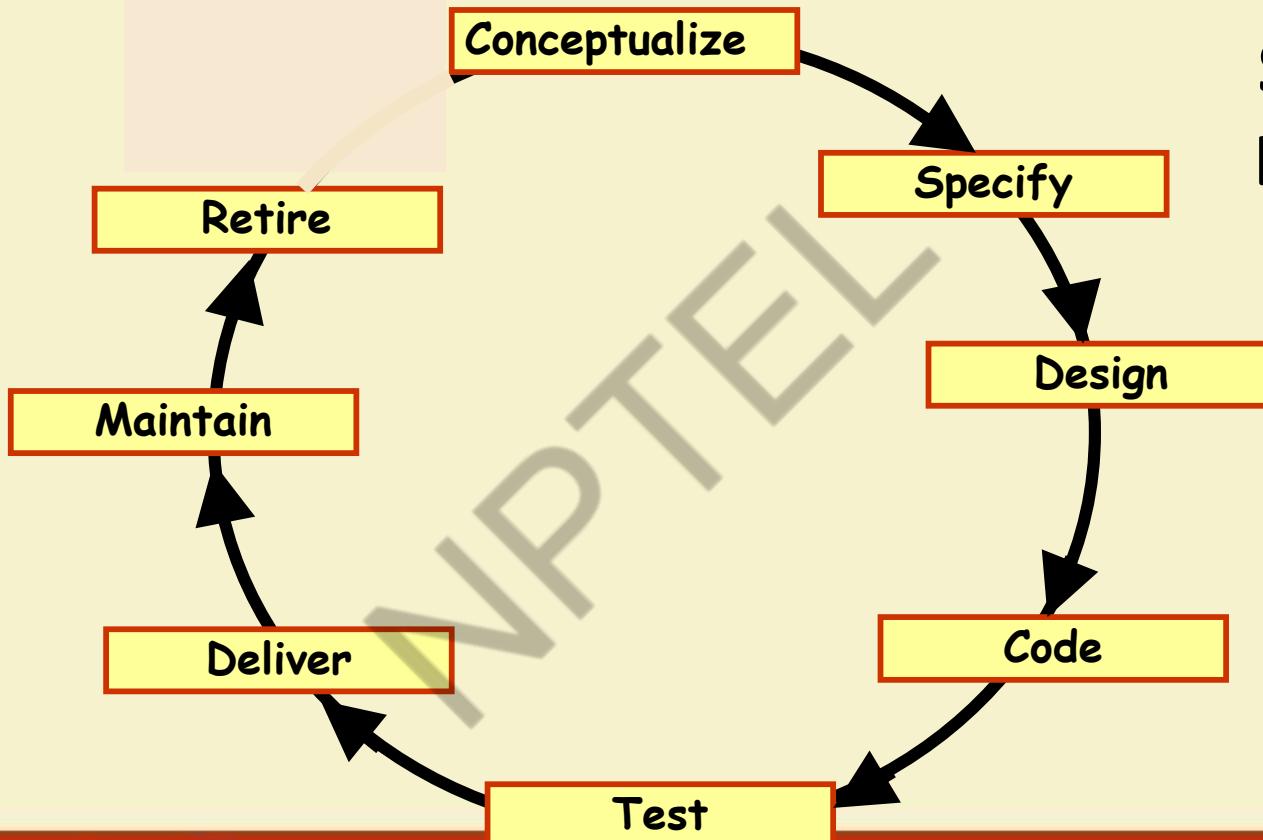


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Software Life Cycle



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Life Cycle Model

- A software life cycle model (also process model or SDLC):
  - A descriptive and diagrammatic model of software life cycle:
  - Identifies all the activities undertaken during product development,
  - Establishes a precedence ordering among the different activities,
  - Divides life cycle into phases.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Life Cycle Model (CONT.)

- Each life cycle phase consists of several activities.
  - For example, the design stage might consist of:
    - structured analysis
    - structured design
    - Design review



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Why Model Life Cycle?

- A graphical and written description:
  - Helps common understanding of activities among the software developers.
  - Helps to identify inconsistencies, redundancies, and omissions in the development process.
  - Helps in tailoring a process model for specific projects.



IIT KHARAGPUR



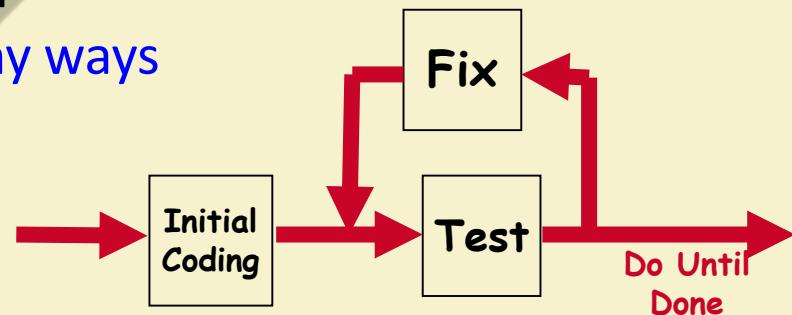
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Life Cycle Model (CONT.)

- The development team must identify a suitable life cycle model:
  - and then adhere to it.
  - Primary advantage of adhering to a life cycle model:
    - Helps development of software in a systematic and disciplined manner.

# Life Cycle Model (CONT.)

- When a program is developed by a single programmer ---
  - The problem is within the grasp of an individual.
  - He has the freedom to decide his exact steps and still succeed --- called **Exploratory model**--- One can use it in many ways
  - Code → Test → Design
  - Code → Design → Test → Change Code →
  - Specify → code → Design → Test → etc.



# Life Cycle Model (CONT.)

- When software is being developed by a team:
  - There must be a precise understanding among team members as to when to do what,
  - Otherwise, it would lead to chaos and project failure.

## Life Cycle Model (CONT.)

- A software project will never succeed if:
  - one engineer starts writing code,
  - another concentrates on writing the test document first,
  - yet another engineer first defines the file structure
  - another defines the I/O for his portion first.

# Phase Entry and Exit Criteria

- A life cycle model:
  - defines entry and exit criteria for every phase.
  - A phase is considered to be complete:
    - only when all its exit criteria are satisfied.



# Life Cycle Model (CONT.)

- What is the phase exit criteria for the software requirements specification phase?
  - Software Requirements Specification (SRS) document is complete, reviewed, and approved by the customer.
- A phase can start:
  - Only if its phase-entry criteria have been satisfied.

# Life Cycle Model: Milestones

- Milestones help software project managers:
  - Track the progress of the project.
  - Phase entry and exit are important milestones.



# Life Cycle and Project Management

- When a life cycle model is followed:
  - The project manager can at any time fairly accurately tell,
    - At which stage (e.g., design, code, test, etc.) the project is.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Project Management Without Life Cycle Model

- It becomes very difficult to track the progress of the project.
  - The project manager would have to depend on the guesses of the team members.
- This usually leads to a problem:
  - known as the **99% complete syndrome**.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Project Deliverables: Myth and Reality

## Myth:

The only deliverable for a successful project is the working program.

## Reality:

Documentation of **all aspects** of software development are needed to help in operation and maintenance.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Many life cycle models have been proposed.
- We confine our attention to only a few commonly used models.

- Waterfall
- V model,
- Evolutionary,
- Prototyping
- Spiral model,
- Agile models

Traditional models

**Life Cycle Model** (CONT.)

- Software life cycle (or software process):
  - **Series of identifiable stages that a software product undergoes during its life time:**
  - Feasibility study
  - Requirements analysis and specification,
  - Design,
  - Coding,
  - Testing
  - Maintenance.

## Software Life Cycle



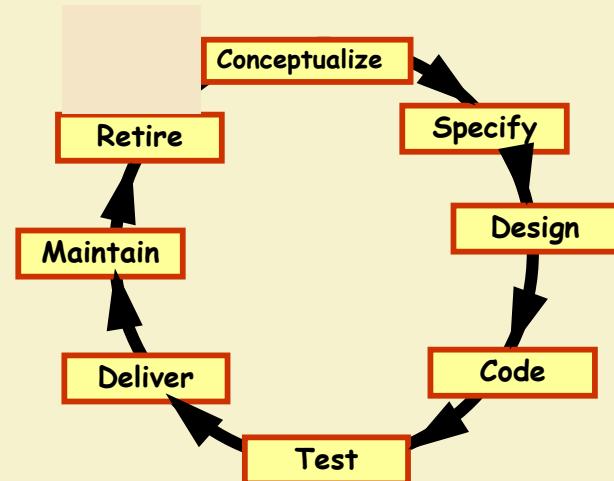
IIT KHARAGPUR



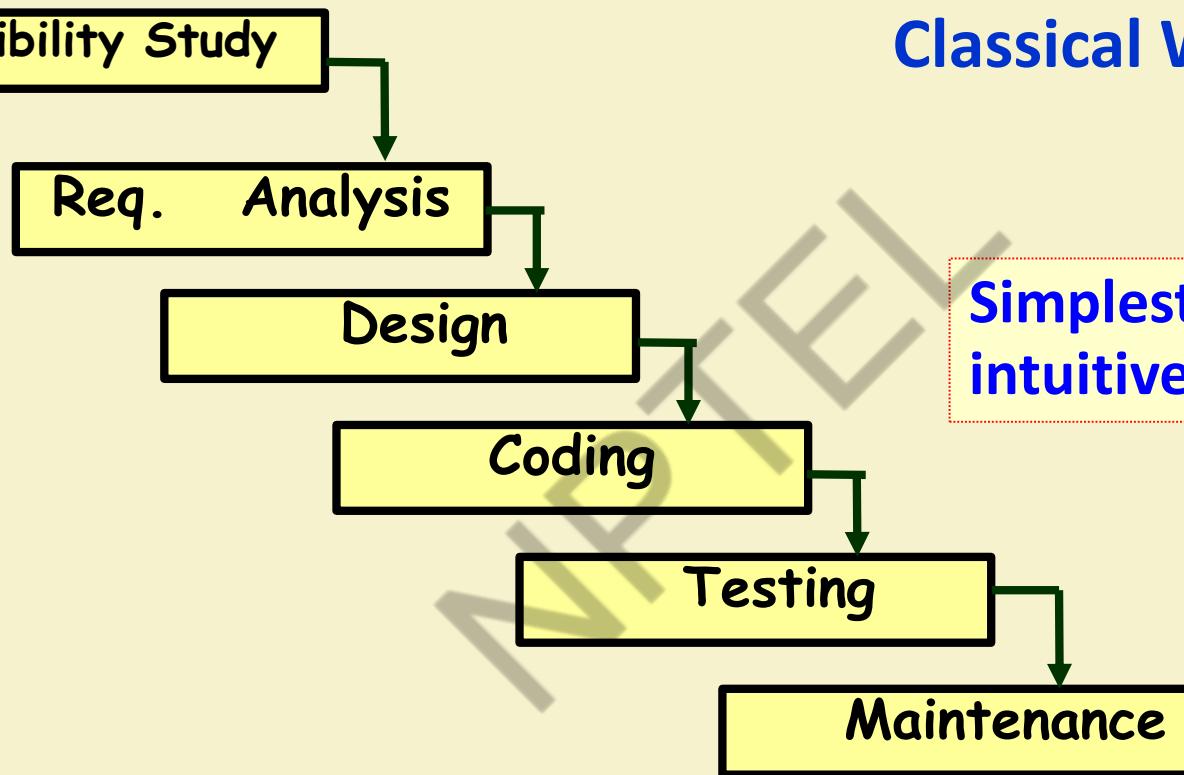
NPTEL  
ONLINE  
CERTIFICATION COURSES

# Classical Waterfall Model

- Classical waterfall model divides life cycle into following phases:
  - Feasibility study,
  - Requirements analysis and specification,
  - Design,
  - Coding and unit testing,
  - Integration and system testing,
  - Maintenance.



# Classical Waterfall Model

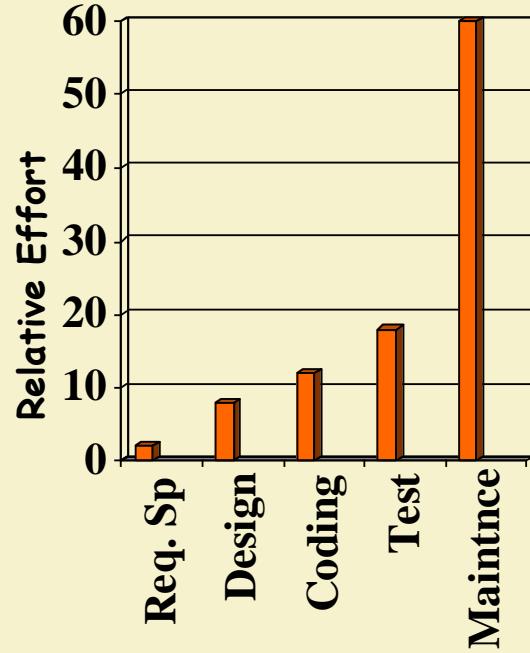


Simplest and most  
intuitive



# Relative Effort for Phases

- Phases between feasibility study and testing
  - Called **development phases**.
- Among all life cycle phases
  - Maintenance phase consumes maximum effort.
- Among development phases,
  - Testing phase consumes the maximum effort.



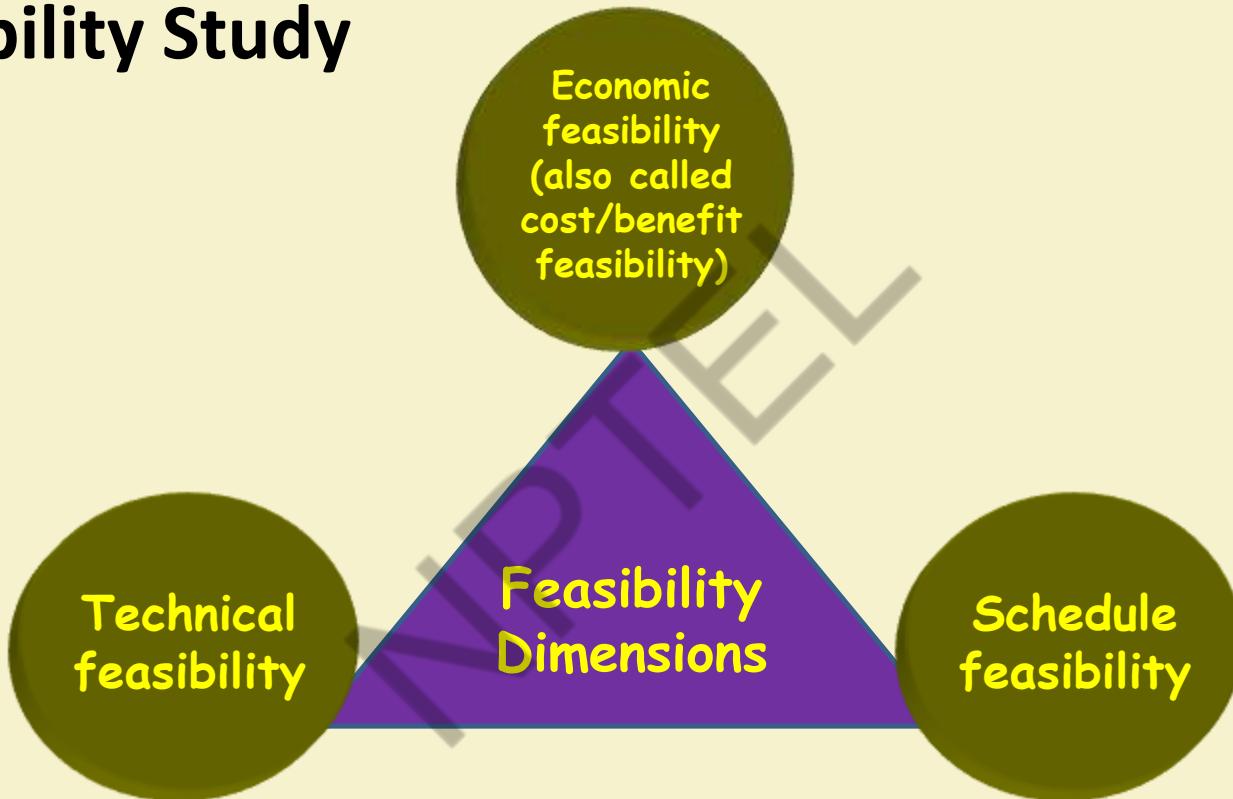
- Most organizations usually define:
  - Standards on the outputs (deliverables) produced at the end of every phase
  - Entry and exit criteria for every phase.
- They also prescribe methodologies for:
  - Specification,
  - Design,
  - Testing,
  - Project management, etc.

## Process Model

## Classical Waterfall Model (CONT.)

- The guidelines and methodologies of an organization:
  - Called the organization's software development methodology.
- Software development organizations:
  - Expect fresh engineers to master the organization's software development methodology.

# Feasibility Study



- Main aim of feasibility study: determine whether developing the software is:
  - Financially worthwhile
  - Technically feasible.
- Roughly understand what customer wants:
  - Data which would be input to the system,
  - Processing needed on these data,
  - Output data to be produced by the system,
  - Various constraints on the behavior of the system.

## Feasibility Study

## First Step

- SPF Scheme for CFL
- CFL has a large number of employees, exceeding 50,000.
- Majority of these are casual labourers
- Mining being a risky profession:
  - Casualties are high
- Though there is a PF:
  - But settlement time is high
- There is a need of SPF:
  - For faster disbursement of benefits

## Case Study

# Feasibility: Case Study

- Manager visits main office, finds out the main functionalities required
- Visits mine site, finds out the data to be input
- Suggests alternate solutions
- Determines the best solution
- Presents to the CFL Officials
- Go/No-Go Decision

# Activities During Feasibility Study

- Work out an overall understanding of the problem.
- Formulate different solution strategies.
- Examine alternate solution strategies in terms of:
  - resources required,
  - cost of development, and
  - development time.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Perform a cost/benefit analysis:
  - Determine which solution is the best.
  - May also find that none of the solutions is feasible due to:
    - high cost,
    - resource constraints,
    - technical reasons.

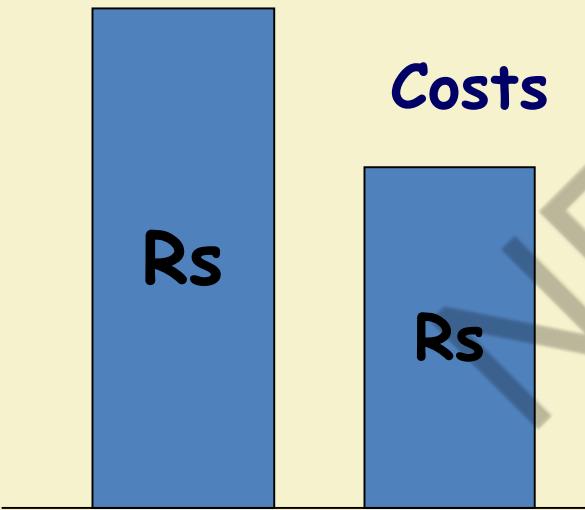
### Activities during Feasibility Study

# Cost benefit analysis (CBA)

- Need to identify all costs --- these could be:
  - Development costs
  - Set-up
  - Operational costs
- Identify the value of benefits
- Check benefits are greater than costs

# The business case

Benefits



- Benefits of delivered project must outweigh costs
- Costs include:
  - Development
  - Operation
- Benefits:
  - Quantifiable
  - Non-quantifiable



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# The business case

- Feasibility studies should help write a ‘business case’
- Should provide a justification for starting the project
- Should show that the benefits of the project will exceed:
  - Various costs
- Needs to take account of business risks

1. Executive summary
2. Project background:

- The focus must be on what, exactly, the project is undertaking, and should not be confused with what might be a bigger picture.

3. Business opportunity

- What difference will it make?
- What if we don't do it?

4. Costs

- Should include the cost of development, implementation, training, change management, and operations.

5. Benefits

- Benefits usually presented in terms of revenue generation and cost reductions.

6. Risks

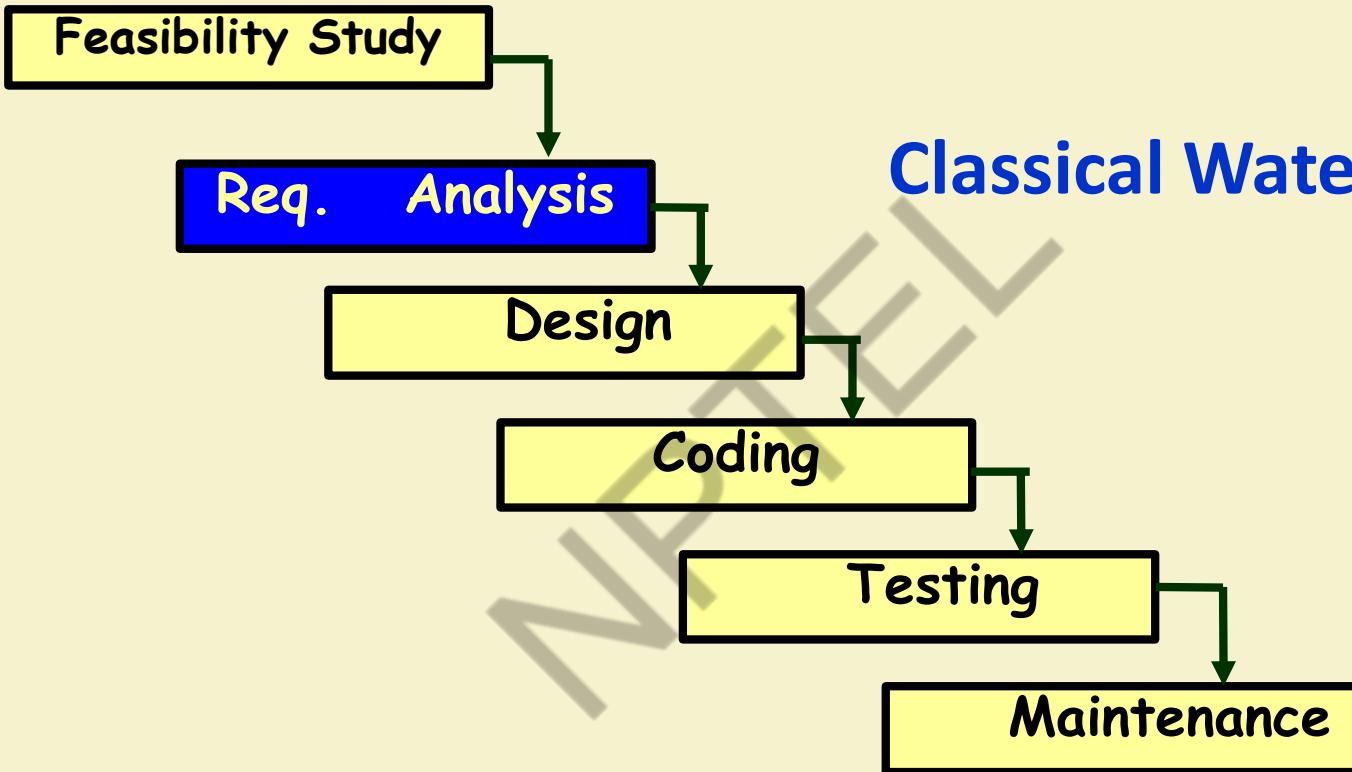
- Identify risks.
- Explain how these will be managed.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



## Classical Waterfall Model



# Requirements Analysis and Specification

- Aim of this phase:
  - Understand the exact requirements of the customer,
  - Document them properly.
- Consists of two distinct activities:
  - Requirements gathering and analysis
  - Requirements specification.

# Requirements Analysis and Specification

- Gather requirements data from the customer:
  - Analyze the collected data to understand what customer wants
- Remove requirements problems:
  - Inconsistencies
  - Anomalies
  - Incompleteness
- Organize into a Software Requirements Specification (SRS) document.

# Requirements Gathering

- Gathering relevant data:
  - Usually collected from the end-users through interviews and discussions.
  - Example: for a business accounting software:
    - Interview all the accountants of the organization to find out their requirements.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

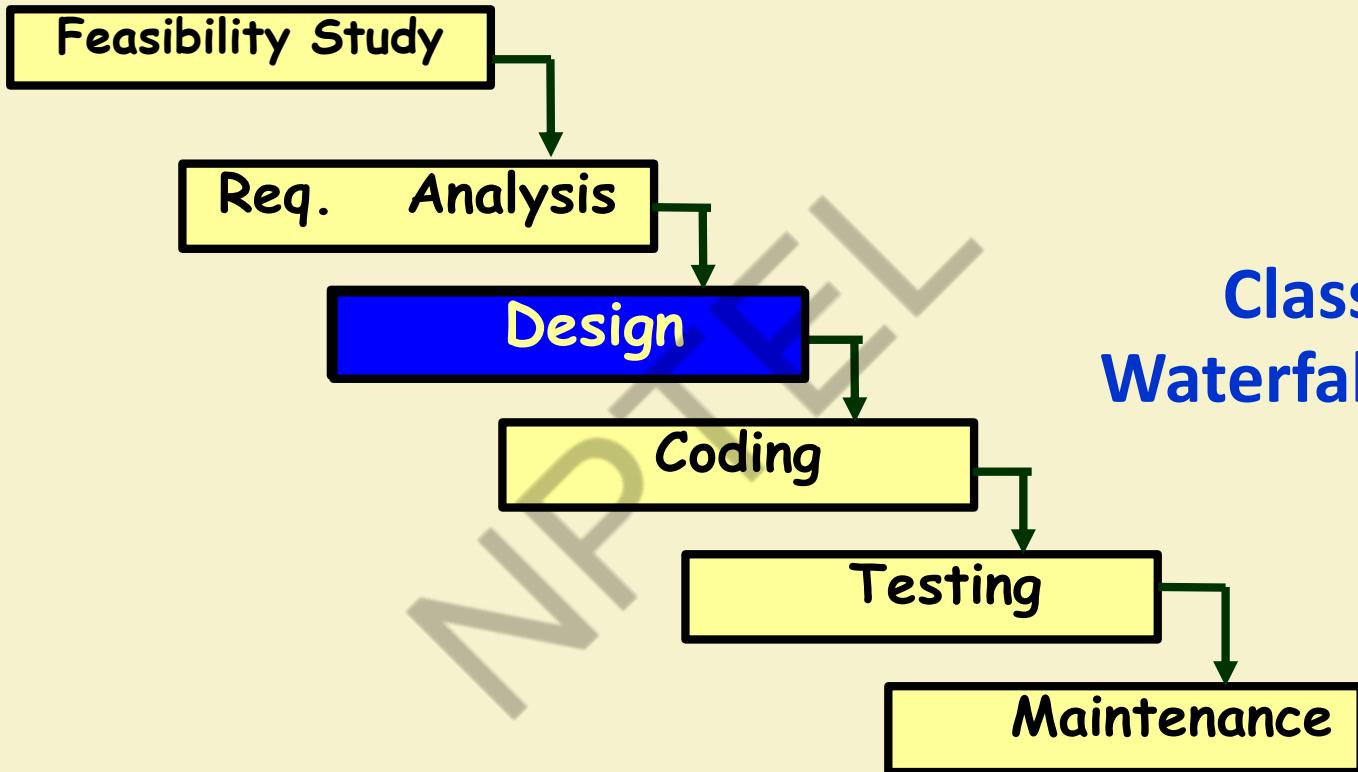
# Requirements Analysis (Cont...)

- The data you initially collect from the users:
  - Usually contain several contradictions and ambiguities.
  - Why?
  - Each user typically has only a partial and incomplete view of the system.**

# Requirements Analysis (Cont...)

- Ambiguities and contradictions:
  - must be identified
  - resolved by discussions with the customers.
- Next, requirements are organized:
  - into a Software Requirements Specification (SRS) document.

## Classical Waterfall Model



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Design

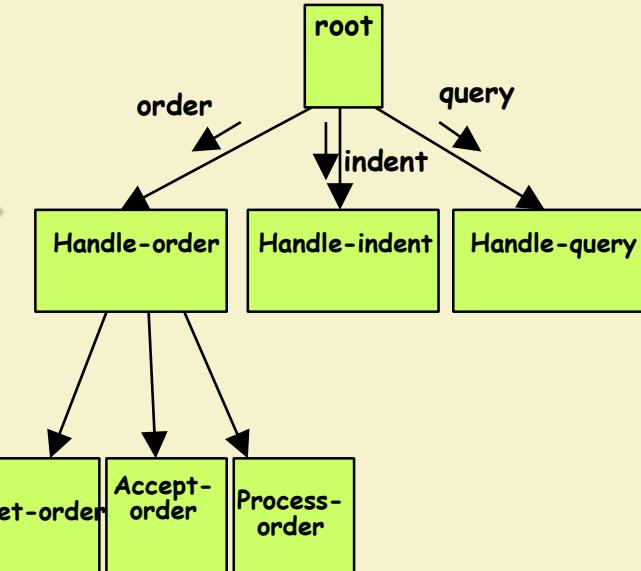
- During design phase requirements specification is transformed into :
  - A form suitable for implementation in some programming language.
- Two commonly used design approaches:
  - Traditional approach,
  - Object oriented approach

# Traditional Design Approach

- Consists of two activities:
  - Structured analysis (typically carried out by using DFD)
  - Structured design

# Structured Design

- High-level design:
  - decompose the system into **modules**,
  - represent invocation relationships among the modules.
- Detailed design:
  - different modules designed in greater detail:
    - data structures and algorithms for each module are designed.



- First identify various objects (real world entities) occurring in the problem:
  - Identify the relationships among the objects.
  - For example, the objects in a pay-roll software may be:
    - employees,
    - managers,
    - pay-roll register,
    - Departments, etc.

## Object-Oriented Design

# Object Oriented Design (CONT.)

- Object structure:
  - Refined to obtain the detailed design.
- OOD has several advantages:
  - Lower development effort,
  - Lower development time,
  - Better maintainability.

**Feasibility Study**

**Req. Analysis**

**Design**

**Coding**

**Testing**

**Maintenance**

**Classical Waterfall Model**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Coding and Unit Testing

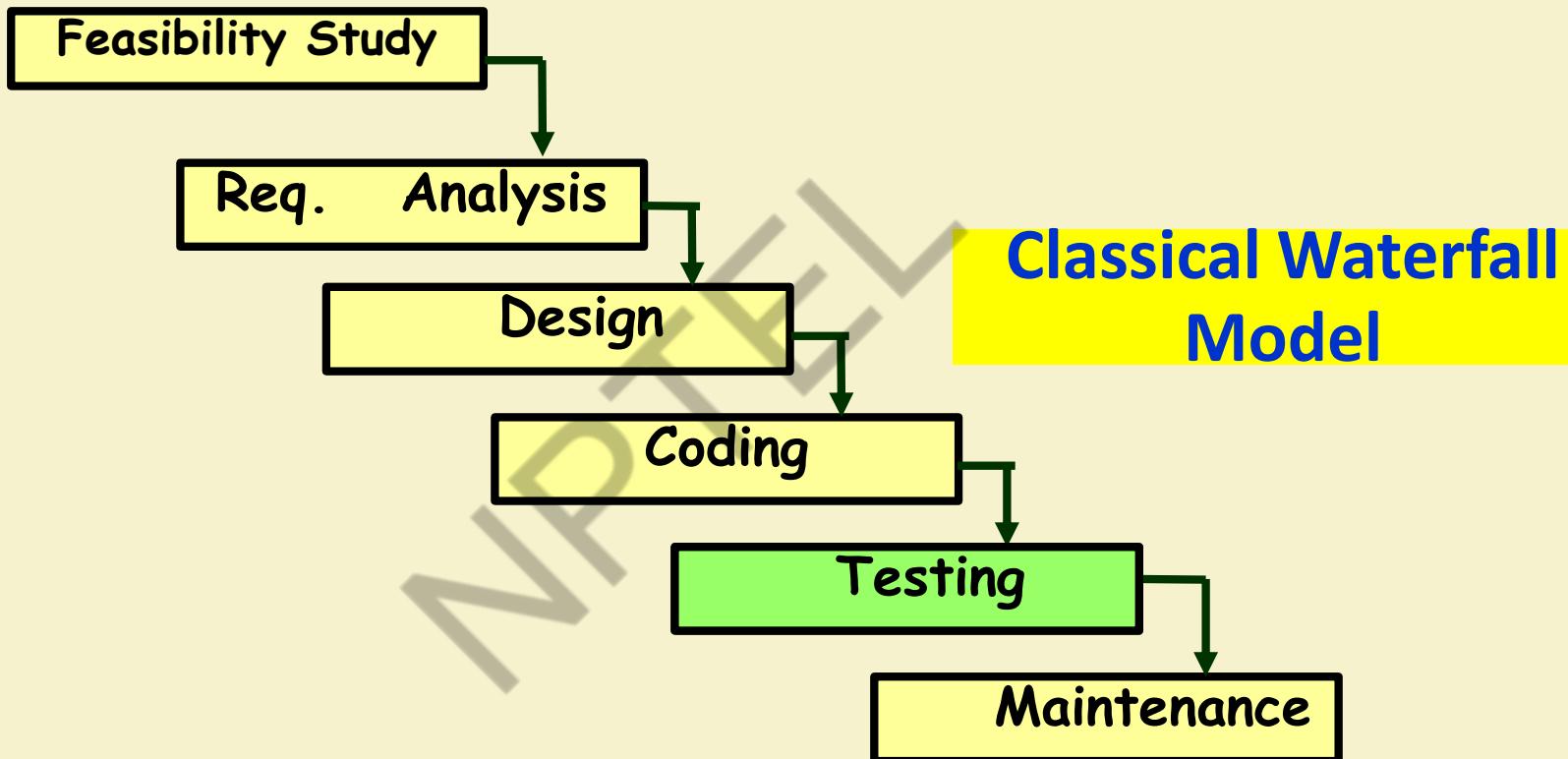
- During this phase:
  - Each module of the design is coded,
  - Each module is unit tested
    - That is, tested independently as a stand alone unit, and debugged.
  - Each module is documented.



IIT KHARAGPUR

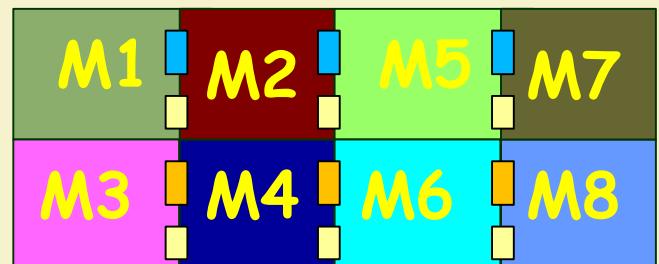


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



# Integration and System Testing

- Different modules are integrated in a planned manner:
  - Modules are usually integrated through a number of steps.
- During each integration step,
  - the partially integrated system is tested.



# System Testing

- After all the modules have been successfully integrated and tested:
  - System testing is carried out.
- Goal of system testing:
  - Ensure that the developed system functions according to its requirements as specified in the SRS document.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

Feasibility Study

Req. Analysis

Design

Coding

Testing

Maintenance

Classical Waterfall Model



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Maintenance

- Maintenance of any software:
  - Requires much more effort than the effort to develop the product itself.
  - Development effort to maintenance effort is typically 40:60.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Types of Maintenance?

- **Corrective maintenance:**

- Correct errors which were not discovered during the product development phases.

- **Perfective maintenance:**

- Improve implementation of the system
- enhance functionalities of the system.

- **Adaptive maintenance:**

- Port software to a new environment,
  - e.g. to a new computer or to a new operating system.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Iterative Waterfall Model

- Classical waterfall model is idealistic:
  - Assumes that no defect is introduced during any development activity.
  - In practice:
    - Defects do get introduced in almost every phase of the life cycle.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

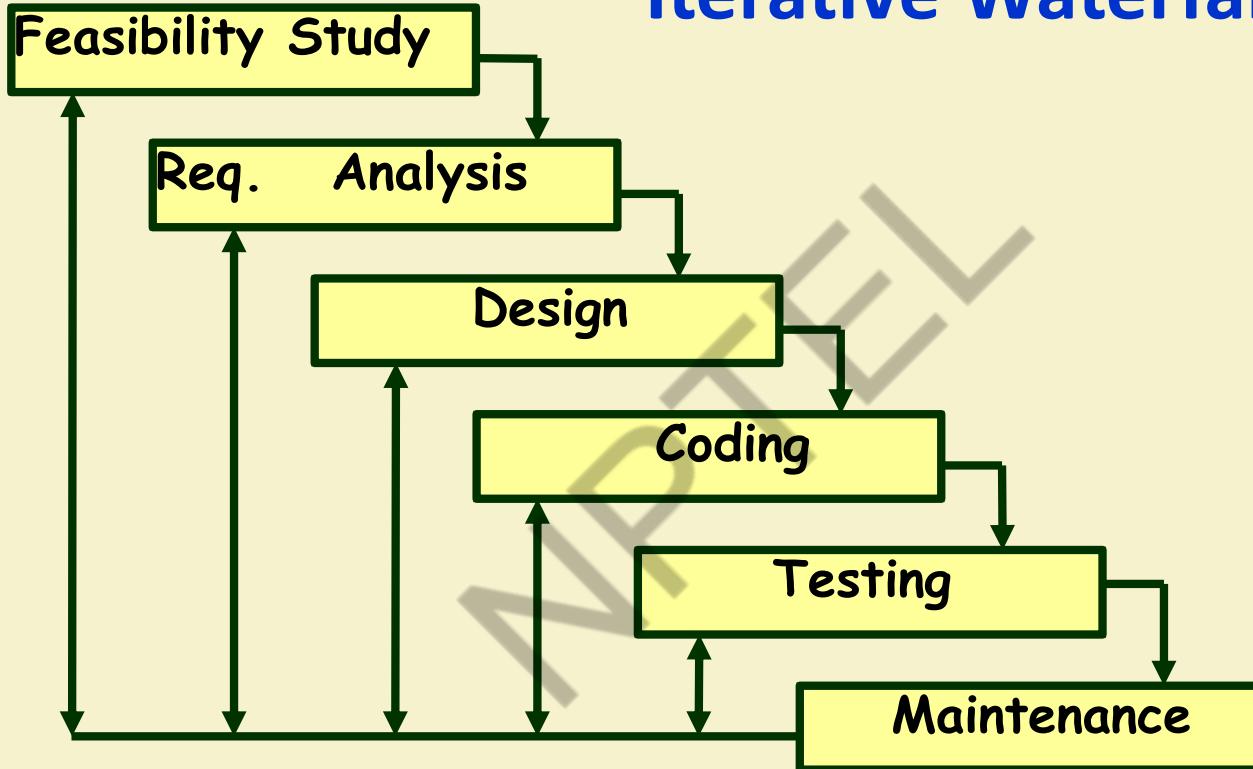
# Iterative Waterfall Model (CONT.)

- Defects usually get detected much later in the life cycle:
  - For example, a design defect might go unnoticed till the coding or testing phase.
  - The later the phase in which the defect gets detected, the more expensive is its removal --- why?**

## Iterative Waterfall Model (CONT.)

- Once a defect is detected:
  - The phase in which it occurred needs to be reworked.
  - Redo some of the work done during that and all subsequent phases.
- Therefore need feedback paths in the classical waterfall model.

# Iterative Waterfall Model (CONT.)



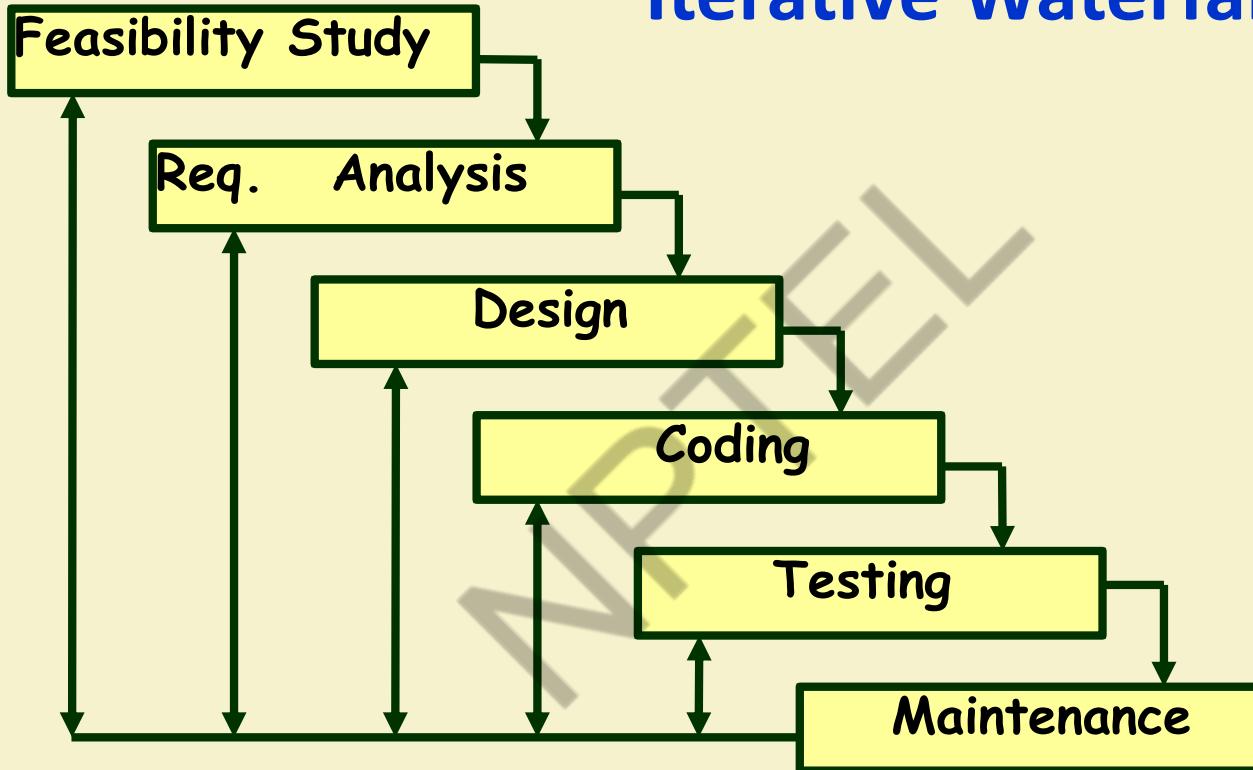
# Phase Containment of Errors (Cont...)

- Errors should be detected:
  - In the same phase in which they are introduced.
- For example:
  - If a design problem is detected in the design phase itself,
    - The problem can be taken care of much more easily
    - Than say if it is identified at the end of the integration and system testing phase.

# Phase Containment of Errors

- Reason: rework must be carried out not only to the design but also to code and test phases.
- The principle of detecting errors as close to its point of introduction as possible:
  - is known as **phase containment of errors**.
- Iterative waterfall model is by far the most widely used model.
  - Almost every other model is derived from the waterfall model.

# Iterative Waterfall Model (CONT.)



# Waterfall Strengths

- Easy to understand, easy to use, especially by inexperienced staff
- Milestones are well understood by the team
- Provides requirements stability during development
- Facilitates strong management control (plan, staff, track)



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Waterfall Deficiencies

- All requirements must be known upfront – **in most projects requirement change occurs after project start**
- Can give a false impression of progress
- Integration is one big bang at the end
- Little opportunity for customer to pre-view the system.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# When to use the Waterfall Model?

- Requirements are well known and stable
- Technology is understood
- Development team have experience with similar projects

## Classical Waterfall Model (CONT.)

- Irrespective of the life cycle model actually followed:
  - The documents should reflect a classical waterfall model of development.
  - Facilitates comprehension of the documents.**

# Classical Waterfall Model (CONT.)

- Metaphor of mathematical theorem proving:
  - A mathematician presents a proof as a single chain of deductions,
    - Even though the proof might have come from a convoluted set of partial attempts, blind alleys and backtracks.

# V Model



IIT KHARAGPUR

7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# V Model

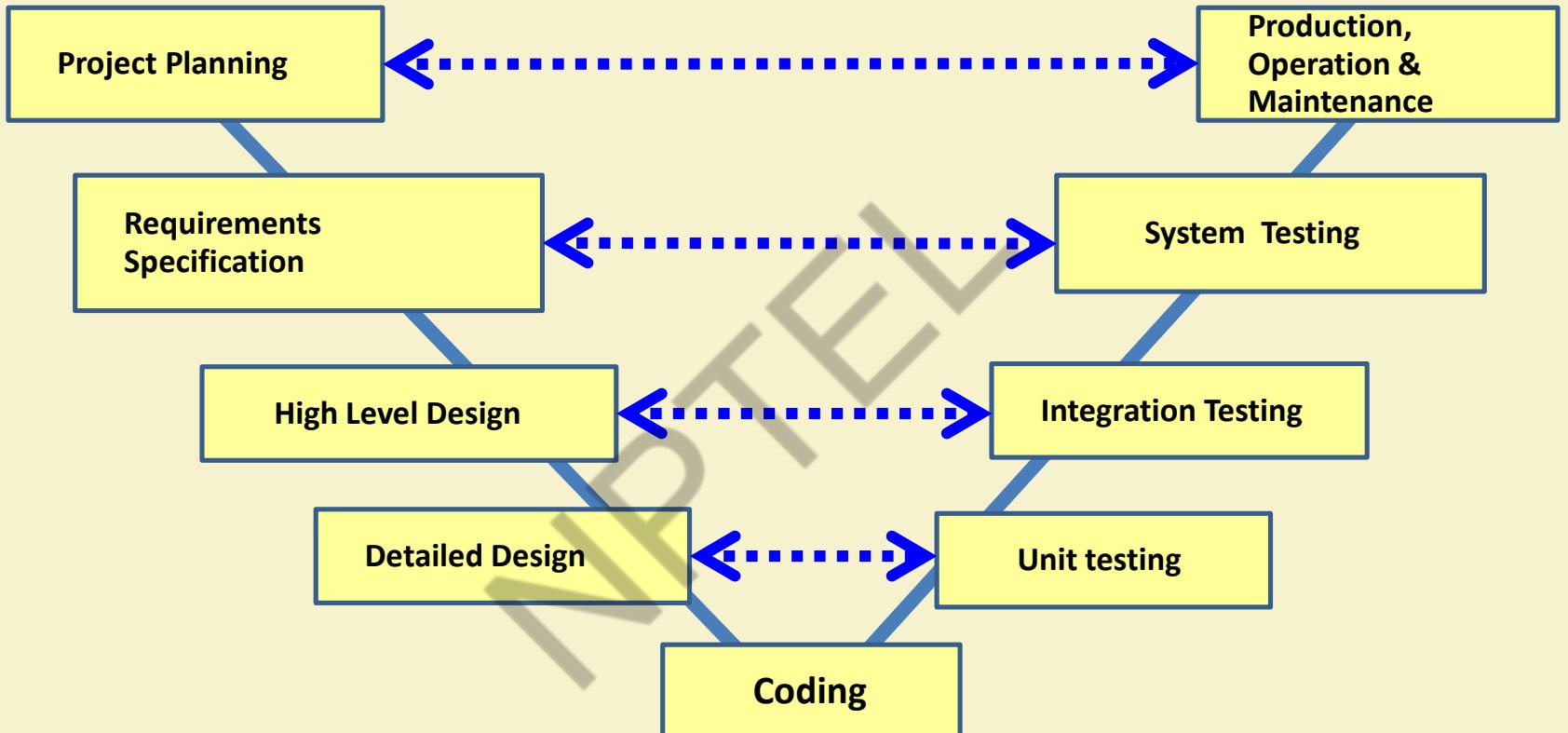
- It is a variant of the Waterfall
  - emphasizes verification and validation
  - V&V activities are spread over the entire life cycle.
- In every phase of development:
  - Testing activities are planned in parallel with development.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# V Model Steps

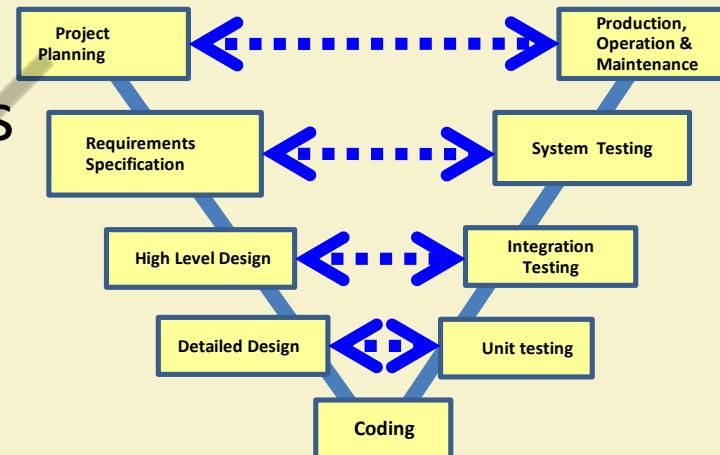
- Planning
- Requirements Analysis and Specification
  - System test design
- High-level Design
  - Integration Test design
- Detailed Design
  - Unit test design

# V Model: Strengths

- Starting from early stages of software development:
  - Emphasizes planning for verification and validation of the software
- Each deliverable is made testable
- Easy to use

# V Model Weaknesses

- Does not support overlapping of phases
- Does not handle iterations or phases
- Does not easily accommodate later changes to requirements
- Does not provide support for effective risk handling



# When to use V Model

- Natural choice for systems requiring high reliability:
  - Embedded control applications, safety-critical software
- All requirements are known up-front
- Solution and technology are known



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Prototyping Model



IIT KHARAGPUR

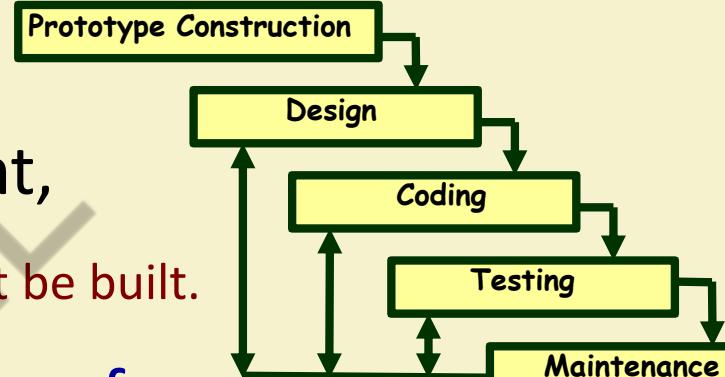
7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

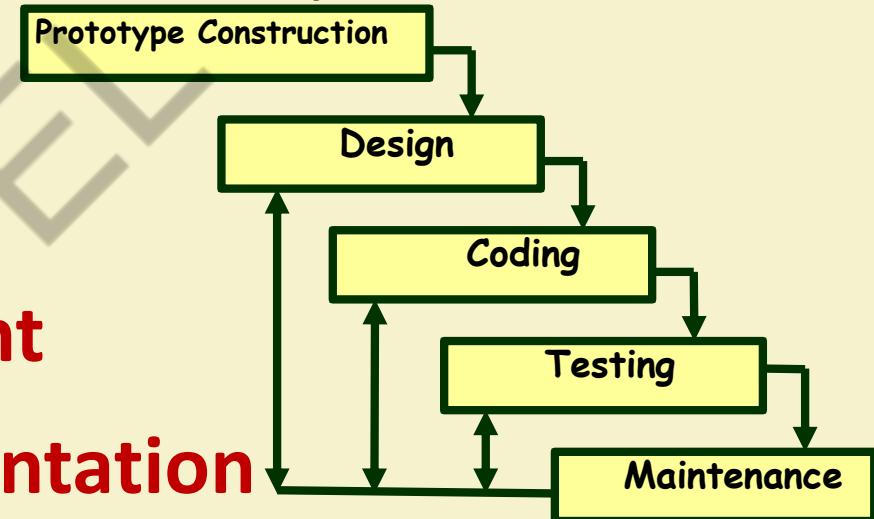
# Prototyping Model

- A derivative of waterfall model.
- Before starting actual development,
  - A working prototype of the system should first be built.
- A prototype is a toy implementation of a system:
  - Limited functional capabilities,
  - Low reliability,
  - Inefficient performance.



# Reasons for prototyping

- **Learning by doing:** useful where requirements are only partially known
- **Improved communication**
- **Improved user involvement**
- **Reduced need for documentation**
- **Reduced maintenance costs**



# Reasons for Developing a Prototype

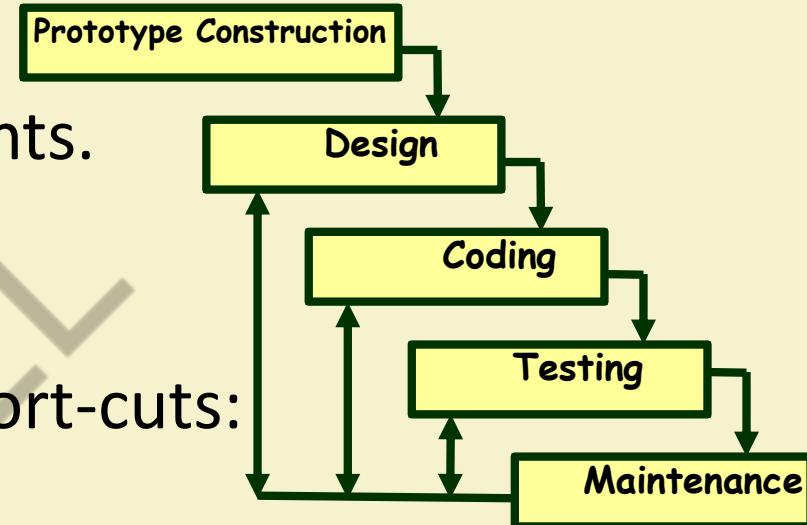
- **Illustrate to the customer:**
  - input data formats, messages, reports, or interactive dialogs.
- **Examine technical issues associated with product development:**
  - Often major design decisions depend on issues like:
    - Response time of a hardware controller,
    - Efficiency of a sorting algorithm, etc.

# Prototyping Model (CONT.)

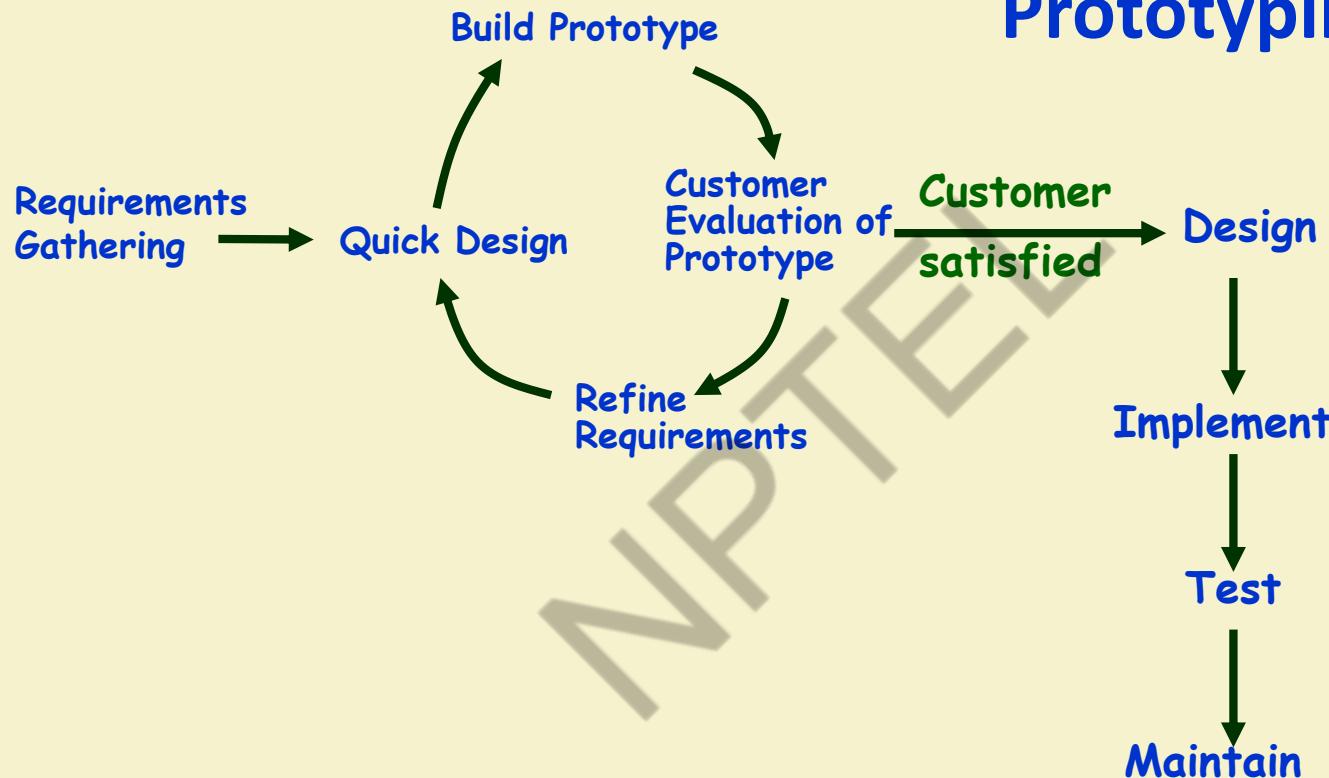
- Another reason for developing a prototype:
  - **It is impossible to “get it right” the first time,**
  - We must plan to throw away the first version:
    - If we want to develop a good software.

# Prototyping Model

- Start with approximate requirements.
- Carry out a quick design.
- Prototype is built using several short-cuts:
  - Short-cuts might involve:
    - Using inefficient, inaccurate, or dummy functions.
    - A table look-up rather than performing the actual computations.

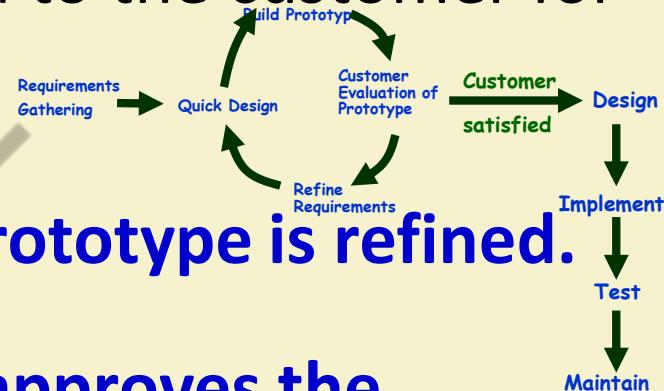


# Prototyping Model (CONT.)



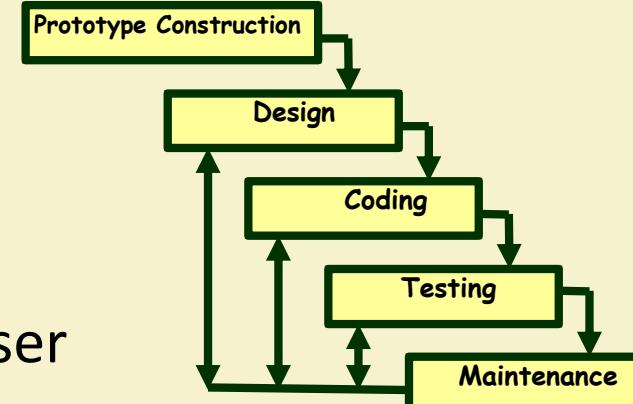
## Prototyping Model (CONT.)

- The developed prototype is submitted to the customer for his evaluation:
  - Based on the user feedback, the prototype is refined.
  - This cycle continues until the user approves the prototype.
- The actual system is developed using the waterfall model.



# Prototyping Model

- Requirements analysis and specification phase becomes redundant:
  - Final working prototype (incorporating all user feedbacks) serves as an **animated requirements specification.**
- **Design and code for the prototype is usually thrown away:**
  - However, experience gathered from developing the prototype helps a great deal while developing the actual software.



# Prototyping Model (CONT.)

- Even though construction of a working prototype model involves additional cost --- **overall development cost usually lower for:**
  - Systems with unclear user requirements,
  - Systems with unresolved technical issues.
- Many user requirements get properly defined and technical issues get resolved:
  - These would have appeared later as change requests and resulted in incurring massive redesign costs.

# Prototyping: advantages

- The resulting software is usually more usable
- User needs are better accommodated
- The design is of higher quality
- The resulting software is easier to maintain
- Overall, the development incurs less cost



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Prototyping: disadvantages

- For some projects, it is expensive
- Susceptible to over-engineering:
  - Designers start to incorporate sophistications that they could not incorporate in the prototype.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Major difficulties of Waterfall-Based Models

1. Difficulty in accommodating change requests during development.
  - **40% of the requirements change during development**
2. High cost incurred in developing custom applications.
3. **“Heavy weight processes.”**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Major difficulties of Waterfall-Based Life Cycle Models

- Requirements for the system are determined at the start:
  - Are assumed to be fixed from that point on.
  - Long term planning is made based on this.

“... the assumption that one can specify a satisfactory system in advance, get bids for its construction, have it built, and install it. ...this assumption is fundamentally wrong and many software acquisition problems spring from this...”

**Frederick Brooks**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Incremental Model



IIT KHARAGPUR

7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- “The basic idea... take advantage of what was being learned during the development of earlier, incremental, deliverable versions of the system. Learning comes from both the development and use of the system... Start with a simple implementation of a subset of the software requirements and iteratively enhance the evolving sequence of versions. At each version design modifications are made along with adding new functional capabilities. **“Victor Basili**



IIT KHARAGPUR

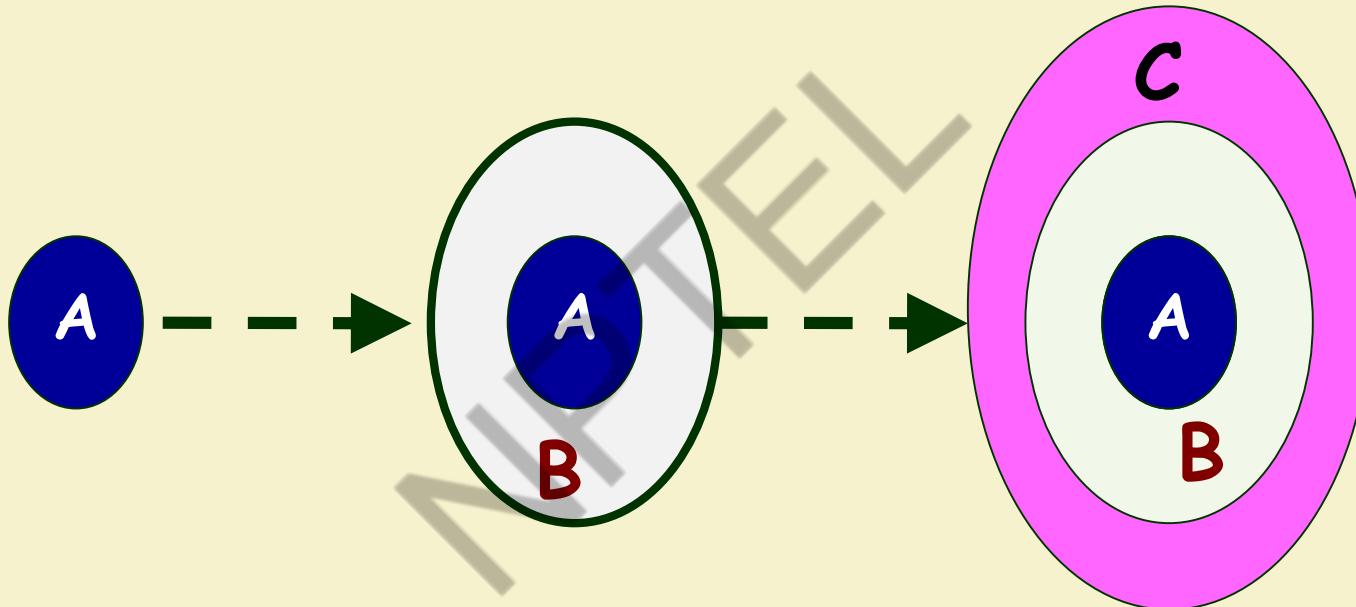


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- **Key characteristics**
  - Builds system incrementally
  - Consists of a planned number of iterations
  - Each iteration produces a working program
- **Benefits**
  - Facilitates and manages changes
- **Foundation of agile techniques and the basis for**
  - Rational Unified Process (RUP)
  - Extreme Programming (XP)

## Incremental and Iterative Development (IID)

# Customer's Perspective

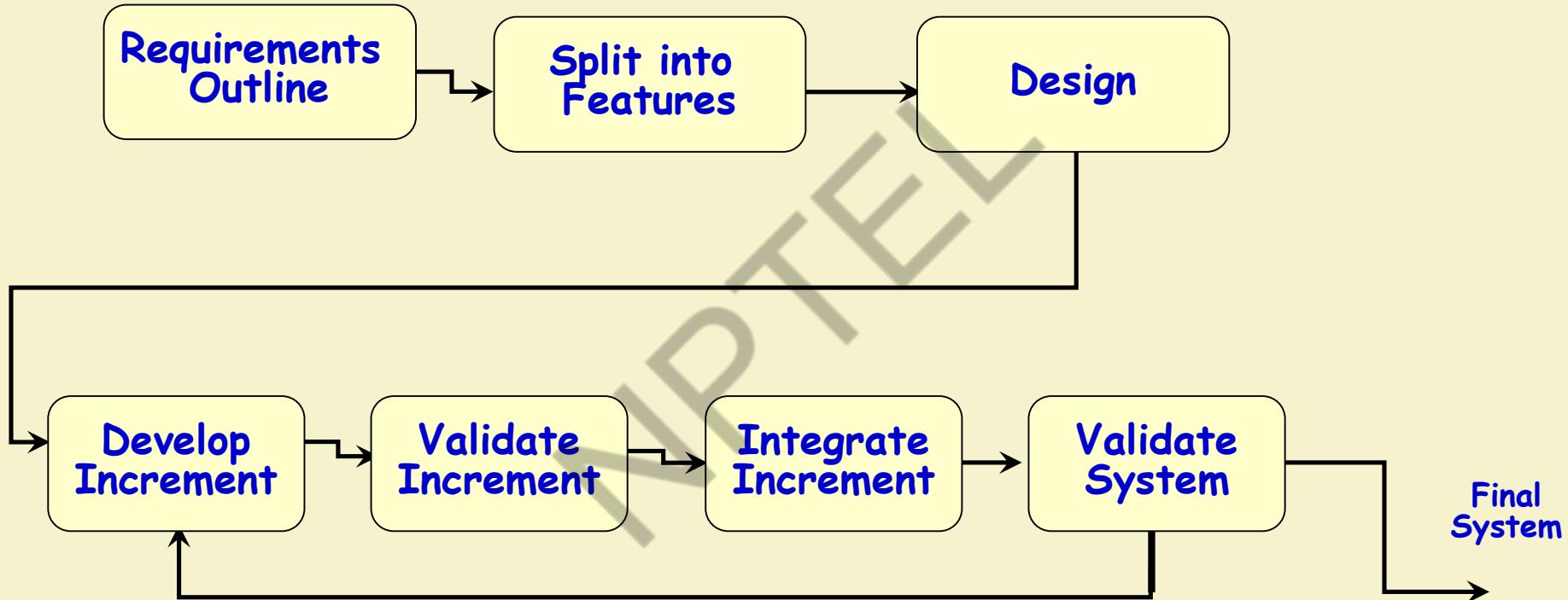


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Incremental Model



# Incremental Model: Requirements

Split into Features

## Requirements: High Level Analysis

Slice



IIT KHARAGPUR

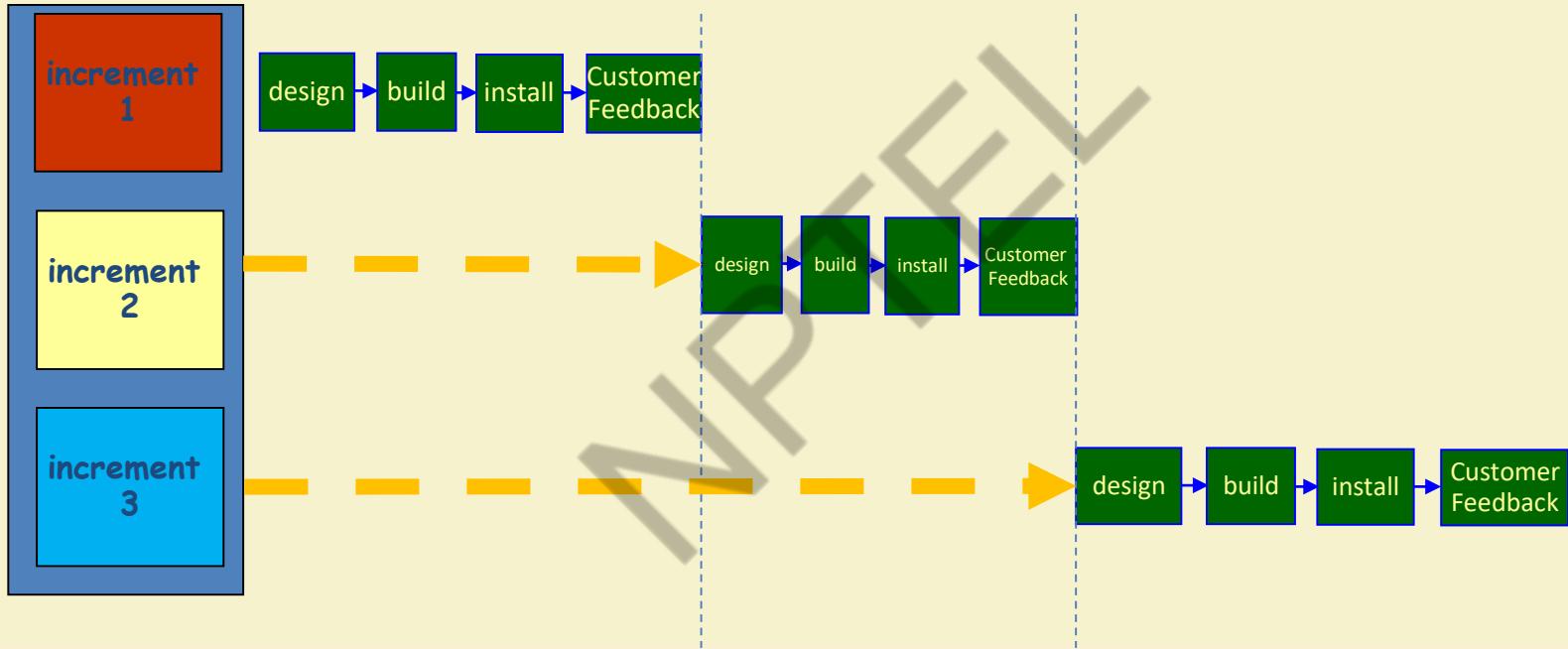


NPTEL ONLINE  
CERTIFICATION COURSES

# Incremental Model

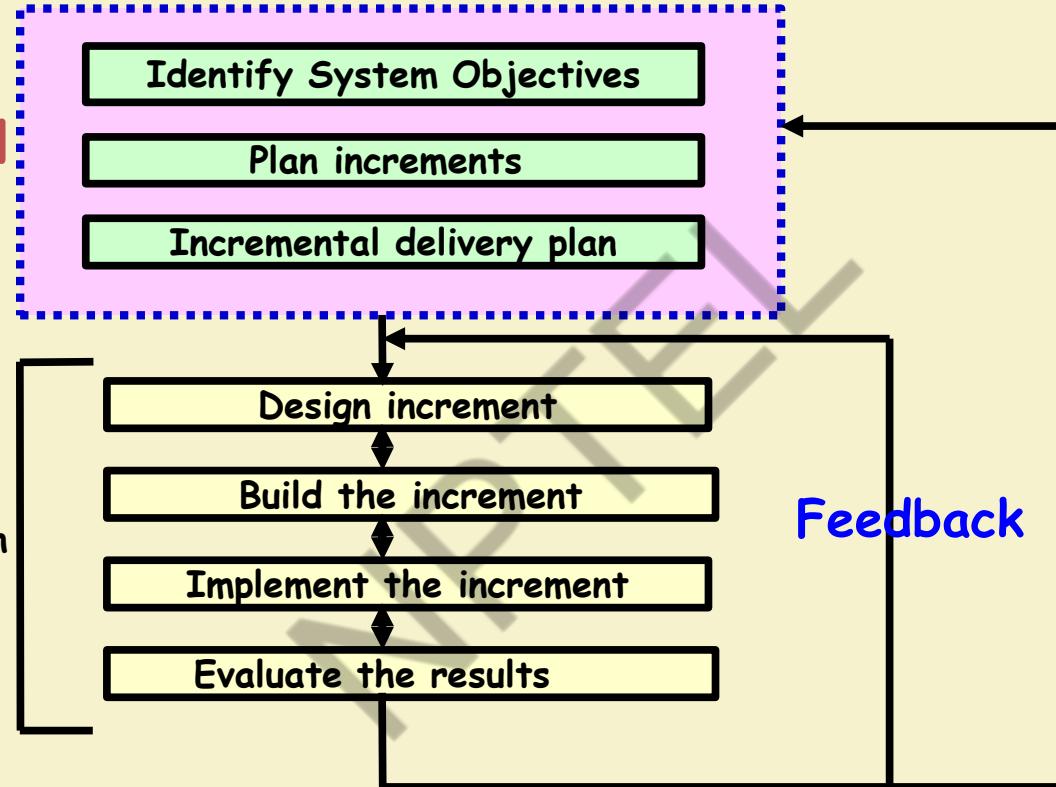
- Waterfall: single release
- Iterative: many releases (increments)
  - **First increment: core functionality**
  - **Successive increments: add/fix functionality**
  - **Final increment: the complete product**
- Each iteration: a short mini-project with a separate lifecycle
  - e.g., waterfall

# Incremental delivery



# The incremental process

Planned  
incremental  
delivery



# Which step first?

- Some steps will be pre-requisite because of physical dependencies
- Others may be in any order
- Value to cost ratios may be used
  - V/C where
  - V is a score 1-10 representing value to customer
  - C is a score 0-10 representing cost to developers



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# V/C ratios: an example

step	value	cost	ratio	
profit reports	9	2	4.5	2nd
online database	1	9	0.11	5th
ad hoc enquiry	5	5	1	4th
purchasing plans	9	4	2.25	3rd
profit-based pay for managers	9	1	9	1st

# **Evolutionary Model with Iterations**



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# An Evolutionary and Iterative Development Process...

- Recognizes the reality of changing requirements
  - Capers Jones's research on 8000 projects: **40% of final requirements arrived after development had already begun**
- Promotes early risk mitigation:
  - Breaks down the system into mini-projects and focuses on the riskier issues first.
  - **“plan a little, design a little, and code a little”**
- Encourages all development participants to be involved earlier on,:
  - End users, Testers, integrators, and technical writers

# Evolutionary Model with Iteration

- “A complex system will be most successful if implemented in small steps... “retreat” to a previous successful step on failure... opportunity to receive some feedback from the real world before throwing in all resources... and you can correct possible errors...” **Tom Glib in Software Metrics**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Evolutionary model with iteration

- Evolutionary iterative development implies that the requirements, plan, estimates, and solution evolve or are refined over the course of the iterations, rather than fully defined and “frozen” in a major up-front specification effort before the development iterations begin. Evolutionary methods are consistent with the pattern of unpredictable discovery and change in new product development.” **Craig Larman**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Evolutionary Model

- First develop the core modules of the software.
- The initial skeletal software is refined into increasing levels of capability: (Iterations)
  - By adding new functionalities in successive versions.

# Activities in an Iteration

- Software developed over several “mini waterfalls”.
- The result of a single iteration:
  - Ends with delivery of some tangible code
  - An incremental improvement to the software --- leads to evolutionary development

# Evolutionary Model with Iteration

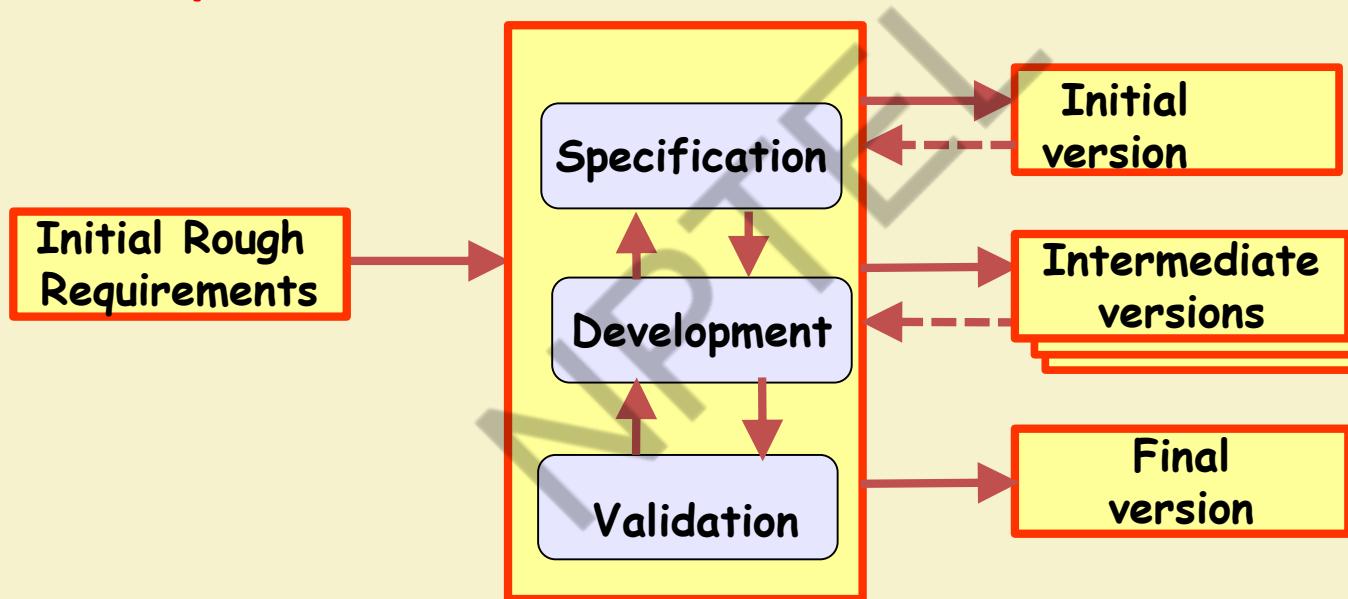
- Outcome of each iteration: tested, integrated, executable system
- Iteration length is short and fixed
  - Usually between 2 and 6 weeks
  - Development takes many iterations (for example: 10-15)
- Does not “freeze” requirements and then conservatively design :
  - Opportunity exists to modify requirements as well as the design...

## Evolutionary Model (CONT.)

- Successive versions:
  - Functioning systems capable of performing some useful work.
  - A new release may include new functionality:
    - Also existing functionality in the current release might have been enhanced.

# Evolutionary Model

- Evolves an initial implementation with user feedback:
  - **Multiple versions until the final version.**



# Advantages of Evolutionary Model

- Users get a chance to experiment with a partially developed system:
  - Much before the full working version is released,
- **Helps finding exact user requirements:**
  - Software more likely to meet exact user requirements.
- **Core modules get tested thoroughly:**
  - Reduces chances of errors in final delivered software.

# Advantages of evolutionary model

- Better management of complexity by developing one increment at a time.
- Better management of changing requirements.
- Can get customer feedback and incorporate them much more efficiently:
  - As compared when customer feedbacks come only after the development work is complete.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Advantages of Evolutionary Model with Iteration

- Training can start on an earlier release
  - customer feedback taken into account
- Frequent releases allow developers to fix unanticipated problems quicker.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Evolutionary Model: Problems

- **The process is intangible:**
  - No regular, well-defined deliverables.
- **The process is unpredictable:**
  - Hard to manage, e.g., scheduling, workforce allocation, etc.
- **Systems are rather poorly structured:**
  - Continual, unpredictable changes tend to degrade the software structure.
- **Systems may not even converge to a final version.**

# RAD Model



IIT KHARAGPUR

7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Rapid Application Development (RAD) Model

- Sometimes referred to as the **rapid prototyping model**.
- Major aims:
  - Decrease the time taken and the cost incurred to develop software systems.
  - Facilitate accommodating change requests as early as possible:
    - Before large investments have been made in development and testing.

# Important Underlying Principle

- A way to reduce development time and cost, and yet have flexibility to incorporate changes:
  - **Make only short term plans and make heavy reuse of existing code.**

# Methodology

- Plans are made for one increment at a time.
  - The time planned for each iteration is called a **time box**.
- Each iteration (increment):
  - Enhances the implemented functionality of the application a little.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Methodology

- During each iteration,
  - A quick-and-dirty prototype-style software for some selected functionality is developed.
  - The customer evaluates the prototype and gives his feedback.
  - The prototype is refined based on the customer feedback.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# How Does RAD Facilitate Faster Development?

- RAD achieves fast creation of working prototypes.
  - Through use of specialized tools.
- These specialized tools usually support the following features:
  - **Visual style of development.**
  - **Use of reusable components.**
  - **Use of standard APIs (Application Program Interfaces).**

## For which Applications is RAD Suitable?

- Customized product developed for one or two customers only
- Performance and reliability are not critical.
- The system can be split into several independent modules.

## For Which Applications RAD is Unsuitable?

- Few plug-in components are available
- High performance or reliability required
- No precedence for similar products exists
- The system cannot be modularized.

# Prototyping versus RAD

- In prototyping model:

- The developed prototype is primarily used to gain insights into the solution
- Choose between alternatives
- Elicit customer feedback.

- The developed prototype:

- Usually thrown away.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Prototyping versus RAD

- In contrast:
  - In RAD the developed prototype evolves into deliverable software.
  - RAD leads to faster development compared to traditional models:
    - However, the quality and reliability would possibly be poorer.

# RAD versus Iterative Waterfall Model

- In the iterative waterfall model,
  - All product functionalities are developed together.
- In the RAD model on the other hand,
  - Product functionalities are developed incrementally through heavy code and design reuse.
  - Customer feedback is obtained on the developed prototype after each iteration:
    - Based on this the prototype is refined.

# RAD versus Iterative Waterfall Model

- The iterative waterfall model:
  - Does not facilitate accommodating requirement change requests.
- Iterative waterfall model does have some important advantages:
  - Use of the iterative waterfall model leads to production of good documentation.
  - Also, the developed software usually has better quality and reliability than that developed using RAD.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## RAD versus Evolutionary Model

- Incremental development:
  - Occurs in both evolutionary and RAD models.
- However, in RAD:
  - Each increment is a quick and dirty prototype,
  - Whereas in the evolutionary model each increment is systematically developed using the iterative waterfall model.
- Also, RAD develops software in shorter increments:
  - The incremental functionalities are fairly large in the evolutionary model.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Unified Process



IIT KHARAGPUR

7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Unified Process

- Developed Ivar Jacobson, Grady Booch and James Rumbaugh
  - Incremental and iterative
- Rational Unified Process (RUP) is version tailored by Rational Software:
  - Acquired by IBM in February 2003.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Four Phases --- and iterative Development at Every phase

- Inception Phase
- Elaboration Phase
- Construction Phase
- Transition Phase



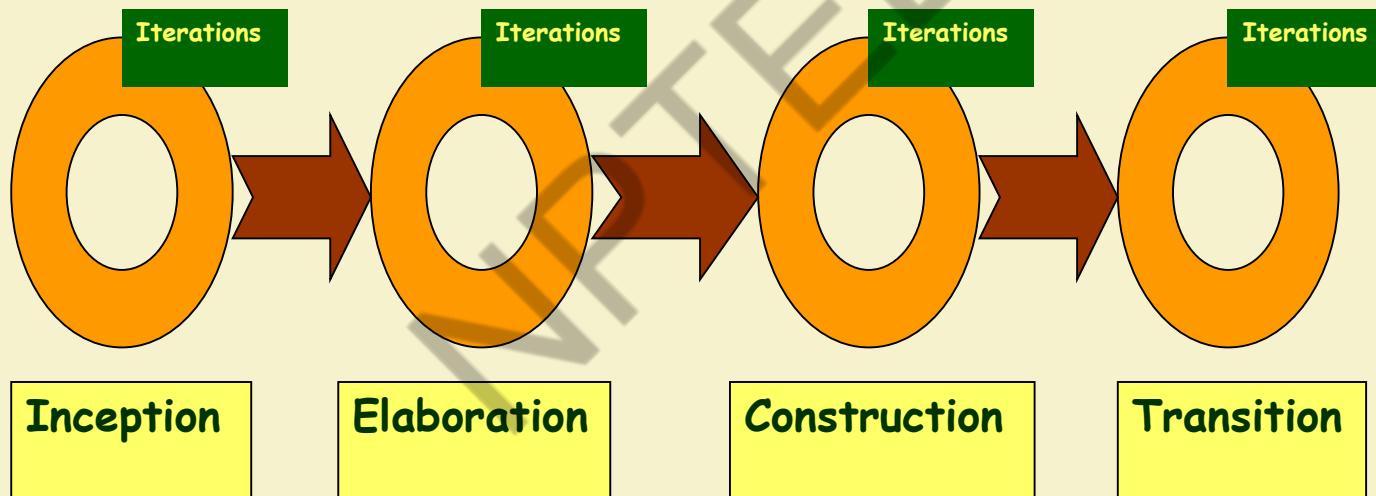
IIT KHARAGPUR



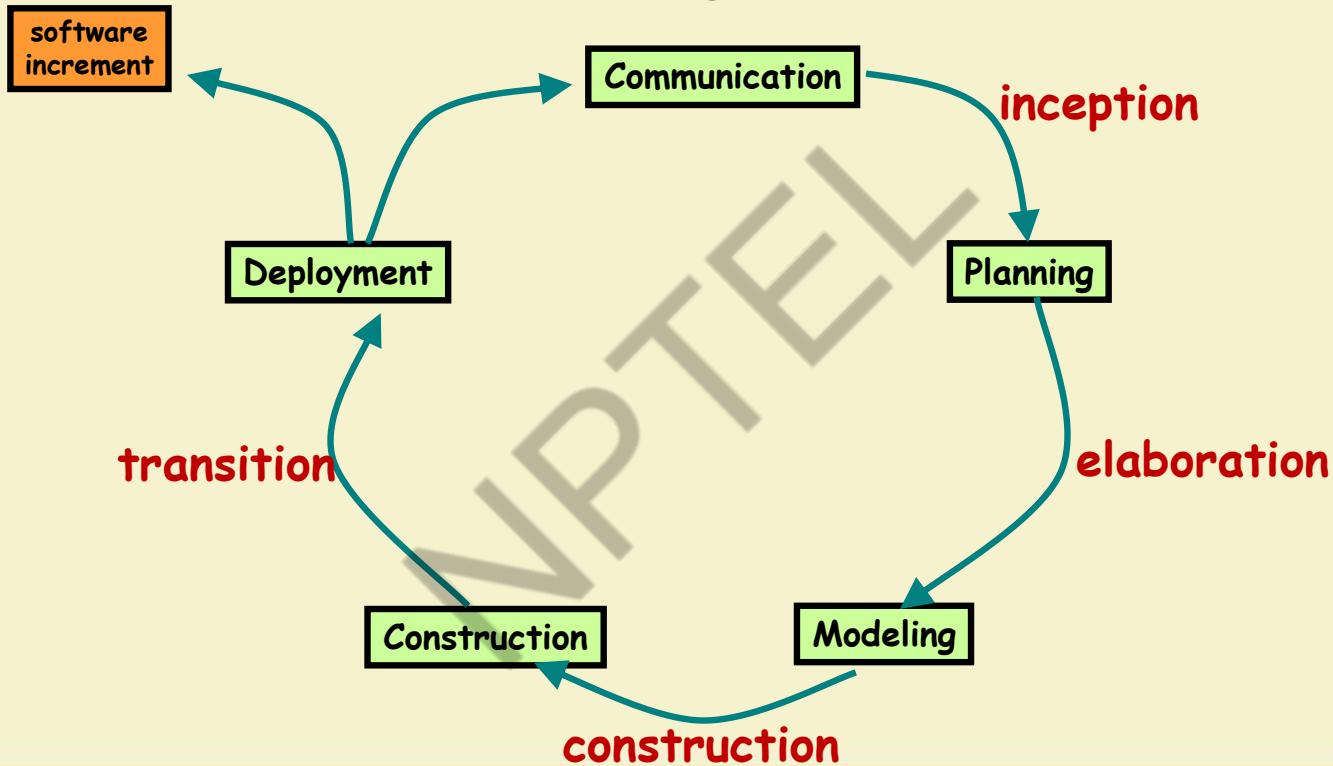
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Unified Process Iterations in Phases

The duration of an iteration may vary from two weeks or less.



# Unified process



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Unified process work products

## Inception phase

vision document  
initial use-case model  
initial business case  
initial risk list  
project plan  
prototype(s)  
...

## Elaboration phase

use-case model  
requirements  
analysis model  
preliminary model  
revised risk list  
preliminary  
manual  
...

## Construction phase

design model  
SW components  
test plan  
test procedure  
test cases  
user manual  
installation manual  
...

## Transition phase

SW increment  
beta test reports  
user feedback  
...



IIT KHARAGPUR

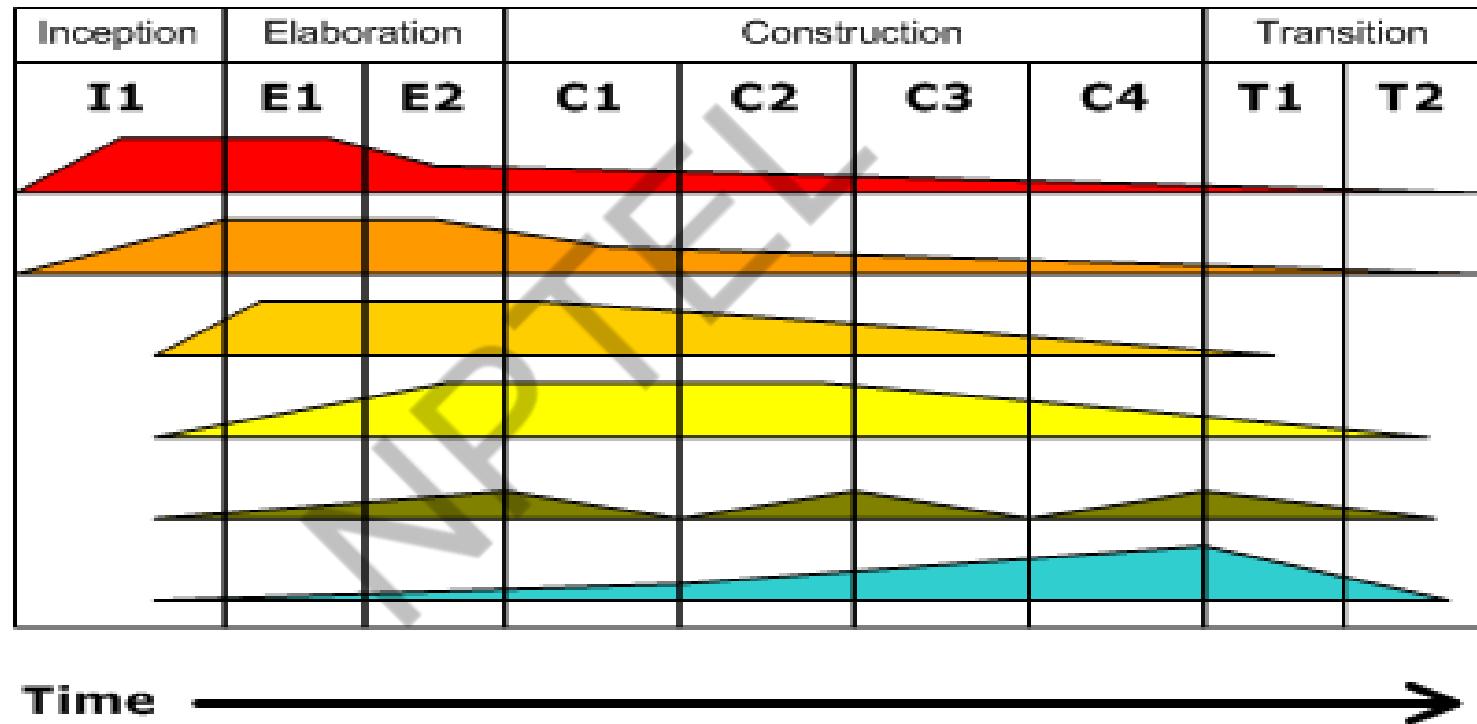


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Structure of RUP Process

- Two dimensions.
- Horizontal axis:
  - Represents time and shows the lifecycle aspects of the process.
- Vertical axis:
  - Represents core process workflows.

# Two dimensions of Unified Process



# Inception activities

- Formulate scope of project
- Risk management, staffing, project plan
- Synthesize a candidate architecture.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Initial requirements capture
- Cost Benefit Analysis
- Initial Risk Analysis
- Project scope definition
- Defining a candidate architecture
- Development of a disposable prototype
- Initial Use Case Model (10% - 20% complete)
- First pass Domain Model

## Outcome of Inception Phase

# Spiral Model

- Proposed by Boehm in 1988.
- Each loop of the spiral represents a phase of the software process:
  - the innermost loop might be concerned with system feasibility,
  - the next loop with system requirements definition,
  - the next one with system design, and so on.
- There are no fixed phases in this model, the phases shown in the figure are just examples.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Spiral Model (CONT.)

- The team must decide:
  - how to structure the project into phases.
- Start work using some generic model:
  - add extra phases
    - for specific projects or when problems are identified during a project.
- Each loop in the spiral is split into four sectors (quadrants).

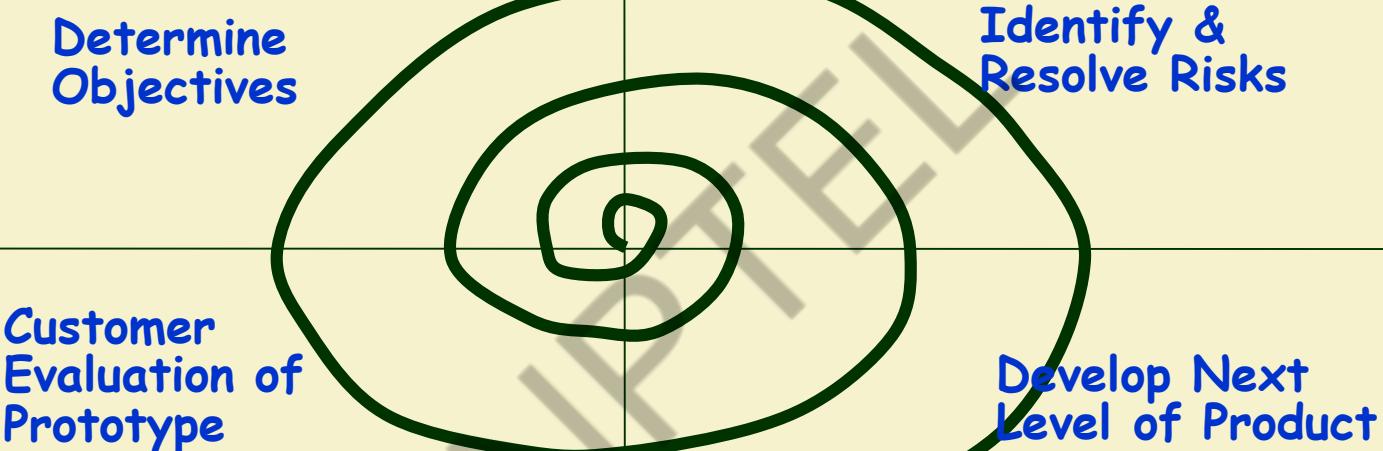


IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Spiral Model



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Objective Setting (First Quadrant)

- Identify objectives of the phase,
- Examine the risks associated with these objectives.
  - Risk:
    - Any adverse circumstance that might hamper successful completion of a software project.
  - Find alternate solutions possible.

# Risk Assessment and Reduction (Second Quadrant)

- For each identified project risk,
  - a detailed analysis is carried out.
- Steps are taken to reduce the risk.
- For example, if there is a risk that requirements are inappropriate:
  - A prototype system may be developed.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Spiral Model (CONT.)

- Development and Validation (**Third quadrant**):
  - develop and validate the next level of the product.
- Review and Planning (**Fourth quadrant**):
  - review the results achieved so far with the customer and plan the next iteration around the spiral.
- With each iteration around the spiral:
  - progressively more complete version of the software gets built.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Subsumes all discussed models:
  - a single loop spiral represents waterfall model.
  - uses an evolutionary approach --
    - iterations over the spiral are evolutionary levels.
  - enables understanding and reacting to risks during each iteration along the spiral.
  - Uses:
    - prototyping as a risk reduction mechanism
    - retains the step-wise approach of the waterfall model.

**Spiral Model as  
a Meta Model**

# Agile Models



IIT KHARAGPUR

7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What is Agile Software Development?

- **Agile:** Easily moved, light, nimble, active software processes
- How agility achieved?
  - Fitting the process to the project
  - Avoidance of things that waste time

# Agile Model

- To overcome the shortcomings of the waterfall model of development.
  - Proposed in mid-1990s
- The agile model was primarily designed:
  - To help projects to adapt to change requests
- In the agile model:
  - The requirements are decomposed into many small incremental parts that can be developed over one to four weeks each.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Ideology: Agile Manifesto

- Individuals and interactions over
  - process and tools
- Working Software over
  - comprehensive documentation
- Customer collaboration over
  - contract negotiation
- Responding to change over
  - following a plan

<http://www.agilemanifesto.org>



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- XP
- Scrum
- Unified process
- Crystal
- DSDM
- Lean

## Agile Methodologies

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Model: Principal Techniques

- **User stories:**
  - Simpler than use cases.
- **Metaphors:**
  - Based on user stories, developers propose a common vision of what is required.
- **Spike:**
  - Simple program to explore potential solutions.
- **Refactor:**
  - Restructure code without affecting behavior, improve efficiency, structure, etc.

- At a time, only one increment is planned, developed, deployed at the customer site.
  - No long-term plans are made.
- An iteration may not add significant functionality,
  - But still a new release is invariably made at the end of each iteration
  - Delivered to the customer for regular use.

## Agile Model: Nitty Gritty

# Methodology

- **Face-to-face communication favoured over written documents.**
- To facilitate face-to-face communication,
  - Development team to share a single office space.
  - Team size is deliberately kept small (5-9 people)
  - This makes the agile model most suited to the development of small projects.



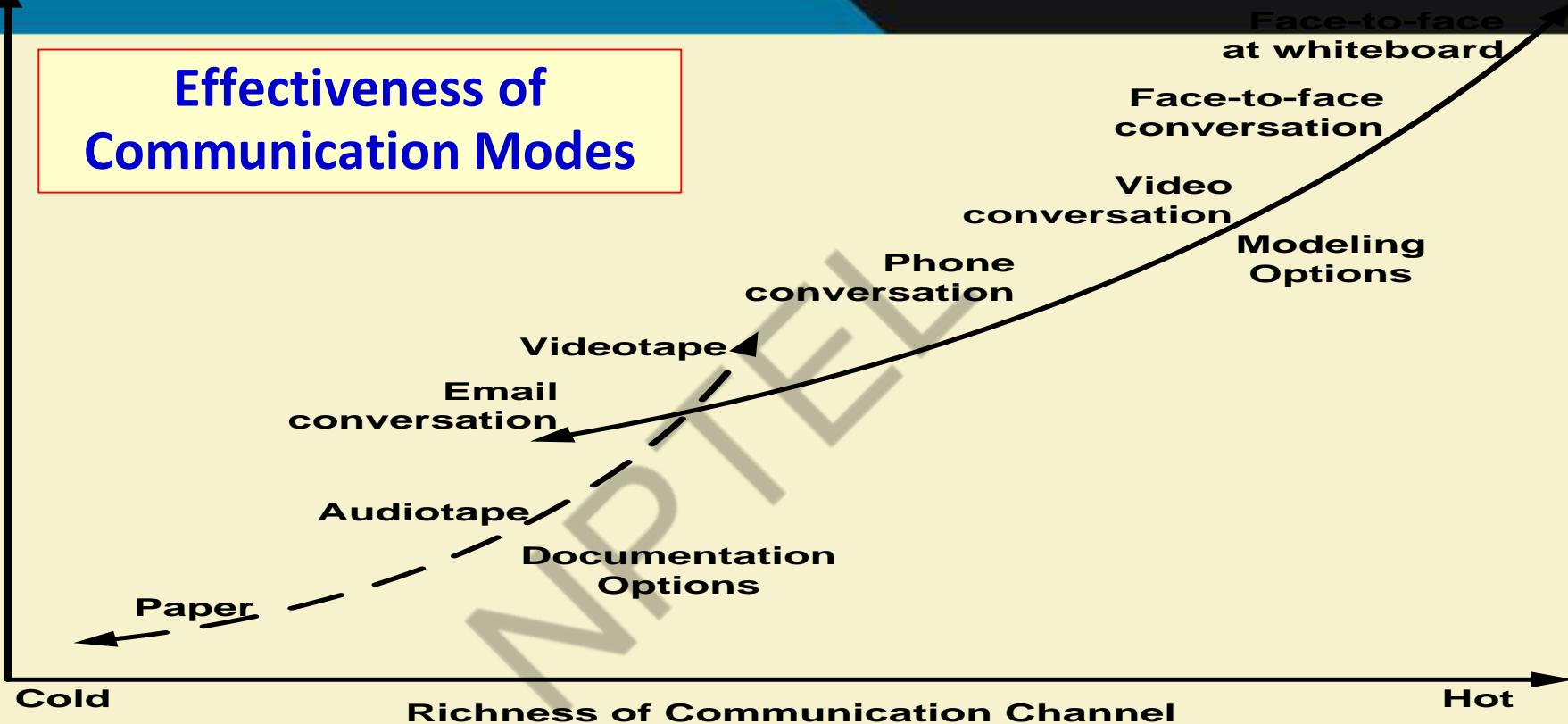
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Effectiveness of Communication Modes

Communication Effectiveness



Copyright 2002-2005 Scott W. Ambler  
Original Diagram Copyright 2002 Alistair Cockburn



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Model: Principles

- The primary measure of progress:
  - Incremental release of working software
- Important principles behind agile model:
  - Frequent delivery of versions --- once every few weeks.
  - Requirements change requests are easily accommodated.
  - Close cooperation between customers and developers.
  - Face-to-face communication among team members.

# Agile Documentation

- Travel light:
  - You need far less documentation than you think.
- Agile documents:
  - Are concise
  - Describe information that is less likely to change
  - Describe “good things to know”
  - Are sufficiently accurate, consistent, and detailed
- Valid reasons to document:
  - Project stakeholders require it
  - To define a contract model
  - To support communication with an external group
  - To think something through

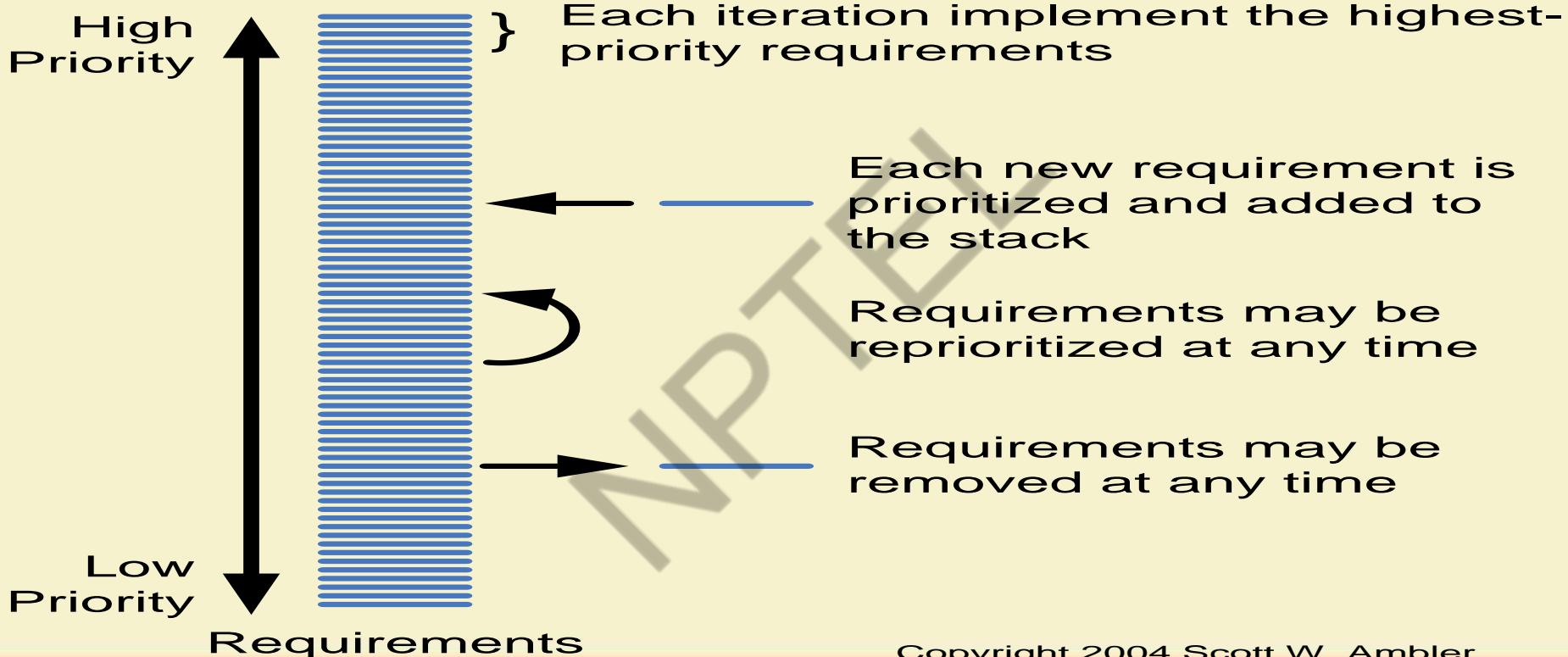


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Software Requirements Management



Copyright 2004 Scott W. Ambler



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Adoption Detractors

- Sketchy definitions, make it possible to have
  - Inconsistent and diverse definitions
- High quality people skills required
- Short iterations inhibit long-term perspective
- Higher risks due to feature creep:
  - Harder to manage feature creep and customer expectations
  - Difficult to quantify cost, time, quality.

# Agile Model Shortcomings

- Derives agility through developing tacit knowledge within the team, rather than any formal document:
  - Can be misinterpreted...
  - External review difficult to get...
  - When project is complete, and team disperses, maintenance becomes difficult...

## Agile Model versus Iterative Waterfall Model

- The waterfall model steps through in a planned sequence:
  - Requirements-capture, analysis, design, coding, and testing .
- Progress is measured in terms of delivered artefacts:
  - Requirement specifications, design documents, test plans, code reviews, etc.
- In contrast agile model sequences:
  - Delivery of working versions of a product in several increments.

# Agile Model versus Iterative Waterfall Model

- As regards to similarity:
  - We can say that Agile teams use the waterfall model on a small scale.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Agile versus RAD Model

- Agile model does not recommend developing prototypes:
  - Systematic development of each incremental feature is emphasized.
- In contrast:
  - RAD is based on designing quick-and-dirty prototypes, which are then refined into production quality code.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Agile versus exploratory programming

- Similarity:
  - Frequent re-evaluation of plans,
  - Emphasis on face-to-face communication,
  - Relatively sparse use of documents.
- Agile teams, however, do follow defined and disciplined processes and carry out rigorous designs:
  - This is in contrast to chaotic coding in exploratory programming.

# Extreme Programming (XP)



IIT KHARAGPUR

7/18/2020



NPTEL ONLINE  
CERTIFICATION COURSES

# Extreme Programming Model

- Extreme programming (XP) was proposed by Kent Beck in 1999.
- The methodology got its name from the fact that:
  - **Recommends taking the best practices to extreme levels.**
  - If something is good, why not do it all the time.

## Taking Good Practices to Extreme

- **If code review is good:**
  - Always review --- **pair programming**
- **If testing is good:**
  - Continually write and execute test cases --- **test-driven development**
- **If incremental development is good:**
  - Come up with new increments every few days
- **If simplicity is good:**
  - Create the simplest design that will support only the currently required functionality.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Taking to Extreme

- **If design is good,**
  - everybody will design daily (refactoring)
- **If architecture is important,**
  - everybody will work at defining and refining the architecture (metaphor)
- **If integration testing is important,**
  - build and integrate test several times a day (continuous integration)



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# 4 Values

- **Communication:**
  - Enhance communication among team members and with the customers.
- **Simplicity:**
  - Build something simple that will work today rather than something that takes time and yet never used
  - May not pay attention for tomorrow
- **Feedback:**
  - System staying out of users is trouble waiting to happen
- **Courage:**
  - Don't hesitate to discard code



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Best Practices

- **Coding:**
  - without code it is not possible to have a working system.
  - Utmost attention needs to be placed on coding.
- **Testing:**
  - Testing is the primary means for developing a fault-free product.
- **Listening:**
  - Careful listening to the customers is essential to develop a good quality product.

# Best Practices

- **Designing:**
  - Without proper design, a system implementation becomes too complex
  - The dependencies within the system become too numerous to comprehend.
- **Feedback:**
  - Feedback is important in learning customer requirements.

# Extreme Programming Activities

- **XP Planning**

- Begins with the creation of “user stories”
- Agile team assesses each story and assigns a cost
- Stories are grouped to form a deliverable increment
- A commitment is made on delivery date

- **XP Design**

- Follows the KIS principle
- Encourage the use of CRC cards
- For difficult design problems, suggests the creation of “spike solutions”—a design prototype
- Encourages “refactoring”—an iterative refinement of the internal program design



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## • XP Coding

- Recommends the construction of unit test cases *before* coding commences (test-driven development)
- Encourages “pair programming”

## • XP Testing

- All unit tests are executed daily
- “Acceptance tests” are defined by the customer and executed to assess customer visible functionalities

**Extreme  
Programming  
Activities**

# Full List of XP Practices

1. **Planning** – determine scope of the next release by combining business priorities and technical estimates
2. **Small releases** – put a simple system into production, then release new versions in very short cycles
3. **Metaphor** – all development is guided by a simple shared story of how the whole system works
4. **Simple design** – system is to be designed as simple as possible
5. **Testing** – programmers continuously write and execute unit tests

# Full List of XP Practices

7. **Refactoring** – programmers continuously restructure the system without changing its behavior to remove duplication and simplify
8. **Pair-programming** -- all production code is written with two programmers at one machine
9. **Collective ownership** – anyone can change any code anywhere in the system at any time.
10. **Continuous integration** – integrate and build the system many times a day – every time a task is completed.

# Full List of XP Practices

11. **40-hour week** – work no more than 40 hours a week as a rule
12. **On-site customer** – a user is a part of the team and available full-time to answer questions
13. **Coding standards** – programmers write all code in accordance with rules emphasizing communication through the code

# Emphasizes Test-Driven Development (TDD)

- Based on user story develop test cases
- Implement a quick and dirty feature every couple of days:
  - Get customer feedback
  - Alter if necessary
  - Refactor
- Take up next feature

# **Project Characteristics that Suggest Suitability of Extreme Programming**

- Projects involving new technology or research projects.
  - In this case, the requirements change rapidly and unforeseen technical problems need to be resolved.
- Small projects:
  - These are easily developed using extreme programming.

# Practice Questions

- What are the stages of iterative waterfall model?
- What are the disadvantages of the iterative waterfall model?
- Why has agile model become so popular?
- What difficulties might be faced if no life cycle model is followed for a certain large project?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Suggest Suitable Life Cycle Model

- A software for an academic institution to automate its:
  - Course registration and grading
  - Fee collection
  - Staff salary
  - Purchase and store inventory
- The software would be developed by tailoring a similar software that was developed for another educational institution:
  - 70% reuse
  - 10% new code and 20% modification

# Practice Questions

- Which types of risks can be better handled using the spiral model compared to the prototyping model?
- Which type of process model is suitable for the following projects:
  - A customization software
  - A payroll software for contract employees that would be add on to an existing payroll software

# Practice Questions

- Which lifecycle model would you select for the following project which has been awarded to us by a mobile phone vendor:
  - A new mobile operating system by upgrading the existing operating system
  - Needs to work well efficiently with 4G systems
  - Power usage minimization
  - Directly upload backup data on a cloud infrastructure maintained by the mobile phone vendor



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Scrum



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Scrum: Characteristics

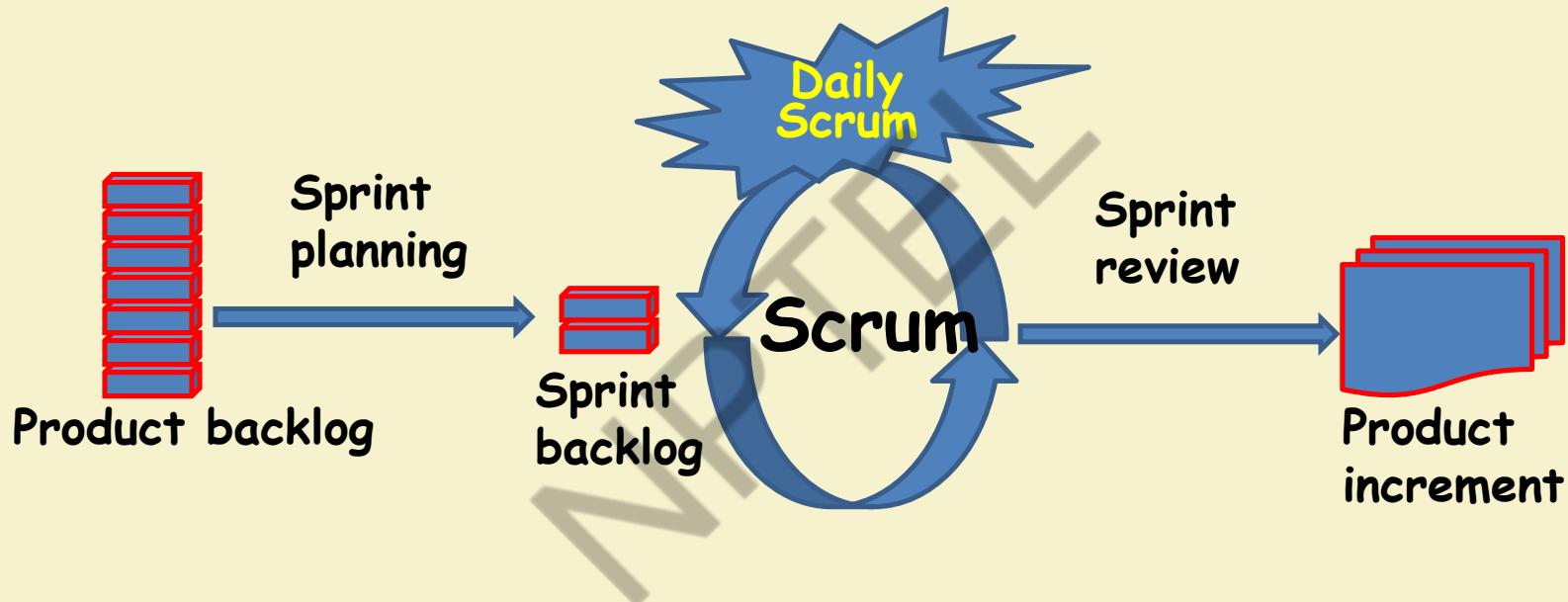
- Self-organizing teams
- Product progresses in a series of month-long **sprints**
- Requirements are captured as items in a list of **product backlog**
- One of the agile processes



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



# Sprint

- Scrum projects progress in a series of “sprints”
  - Analogous to XP iterations or time boxes
  - Target duration is one month
- **Software increment is designed, coded, and tested during the sprint**
- **No changes entertained during a sprint**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Scrum Framework

- **Roles** : Product Owner, ScrumMaster, Team
- **Ceremonies** : Sprint Planning, Sprint Review, Sprint Retrospective, and Daily Scrum Meeting
- **Artifacts** : Product Backlog, Sprint Backlog, and Burndown Chart

## Key Roles and Responsibilities in Scrum Process

- Product Owner
  - Acts on behalf of customers to represent their interests.
- Development Team
  - Team of five-nine people with cross-functional skill sets.
- Scrum Master (aka Project Manager)
  - Facilitates scrum process and resolves impediments at the team and organization level by acting as a buffer between the team and outside interference.

# Product Owner

- Defines the features of the product
- Decide on release date and content
- Prioritize features according to market value
- Adjust features and priority every iteration, as needed
- Accept or reject work results.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# The Scrum Master

- Represents management to the project
- Removes impediments
- Ensure that the team is fully functional and productive
- Enable close cooperation across all roles and functions
- Shield the team from external interferences



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Scrum Team

- Typically 5-10 people
- Cross-functional
  - QA, Programmers, UI Designers, etc.
- Teams are self-organizing
- Membership can change only between sprints

# Ceremonies

- Sprint Planning Meeting
- Sprint
- Daily Scrum
- Sprint Review Meeting

# Sprint Planning

- Goal is to produce Sprint Backlog
- Product owner works with the Team to negotiate what Backlog Items the Team will work on in order to meet Release Goals
- Scrum Master ensures Team agrees to realistic goals



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Sprint

- Fundamental process flow of Scrum
- A month-long iteration, during which an incremental product functionality completed
- NO outside influence can interfere with the Scrum team during the Sprint
- Each day begins with the Daily Scrum Meeting

- Daily
- 15-minutes
- Stand-up meeting
- Not for problem solving
- Three questions:
  1. What did you do yesterday
  2. What will you do today?
  3. What obstacles are in your way?

## Daily Scrum

## Daily Scrum

- Is NOT a problem solving session
- Is NOT a way to collect information about WHO is behind the schedule
- Is a meeting in which team members make commitments to each other and to the Scrum Master
- Is a good way for a Scrum Master to track the progress of the Team

- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features
- Informal
  - 2-hour prep time rule
- Participants
  - Customers
  - Management
  - Product Owner
  - Other engineers

## Sprint Review Meeting



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Product Backlog

- A list of all desired work on the project
  - Usually a combination of
    - story-based work (“let user search and replace”)
    - task-based work (“improve exception handling”)
- List is prioritized by the Product Owner
  - Typically a Product Manager, Marketing, Internal Customer, etc.

# Product Backlog

- Requirements for a system, expressed as a prioritized list of Backlog Items
  - Managed and owned by Product Owner
  - Spreadsheet (typically)
  - Usually is created during the Sprint Planning Meeting



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Sample Product Backlog

	Item #	Description	Est	By
<b>Very High</b>				
	1	<b>Finish database versioning</b>	16	KH
	2	<b>Get rid of unneeded shared Java in database</b>	8	KH
	-	<b>Add licensing</b>	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		<b>Analysis Manager</b>		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
<b>High</b>				
	-	<b>Enforce unique names</b>	-	-
	7	In main application	24	KH
	8	In import	24	AM
	-	<b>Admin Program</b>	-	-
	9	Delete users	4	JM
	-	<b>Analysis Manager</b>	-	-
	10	When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
	-	<b>Query</b>	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
	-	<b>Population Genetics</b>	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
	19	<b>Add icons for v1.1 or 2.0</b>	-	-
	-	<b>Pedigree Manager</b>	-	-
	20	Validate Derived kindred	4	KH
<b>Medium</b>				
	-	<b>Explorer</b>	-	-
	21	Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	22	Delete settings (?)	4	T&A



# Sprint Backlog

- A subset of Product Backlog Items, which define the work for a Sprint
  - Created by Team members
  - Each Item has it's own status
  - Updated daily



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Sprint Backlog during the Sprint

- Changes occur:
  - Team adds new tasks whenever they need to in order to meet the Sprint Goal
  - Team can remove unnecessary tasks
  - But: Sprint Backlog can only be updated by the team
- Estimates are updated whenever there's new information

# Burn down Charts

- Are used to represent “work done”.
- Are remarkably simple but effective Information disseminators
- 3 Types:
  - Sprint Burn down Chart (progress of the Sprint)
  - Release Burn down Chart (progress of release)
  - Product Burn down chart (progress of the Product)



IIT KHARAGPUR

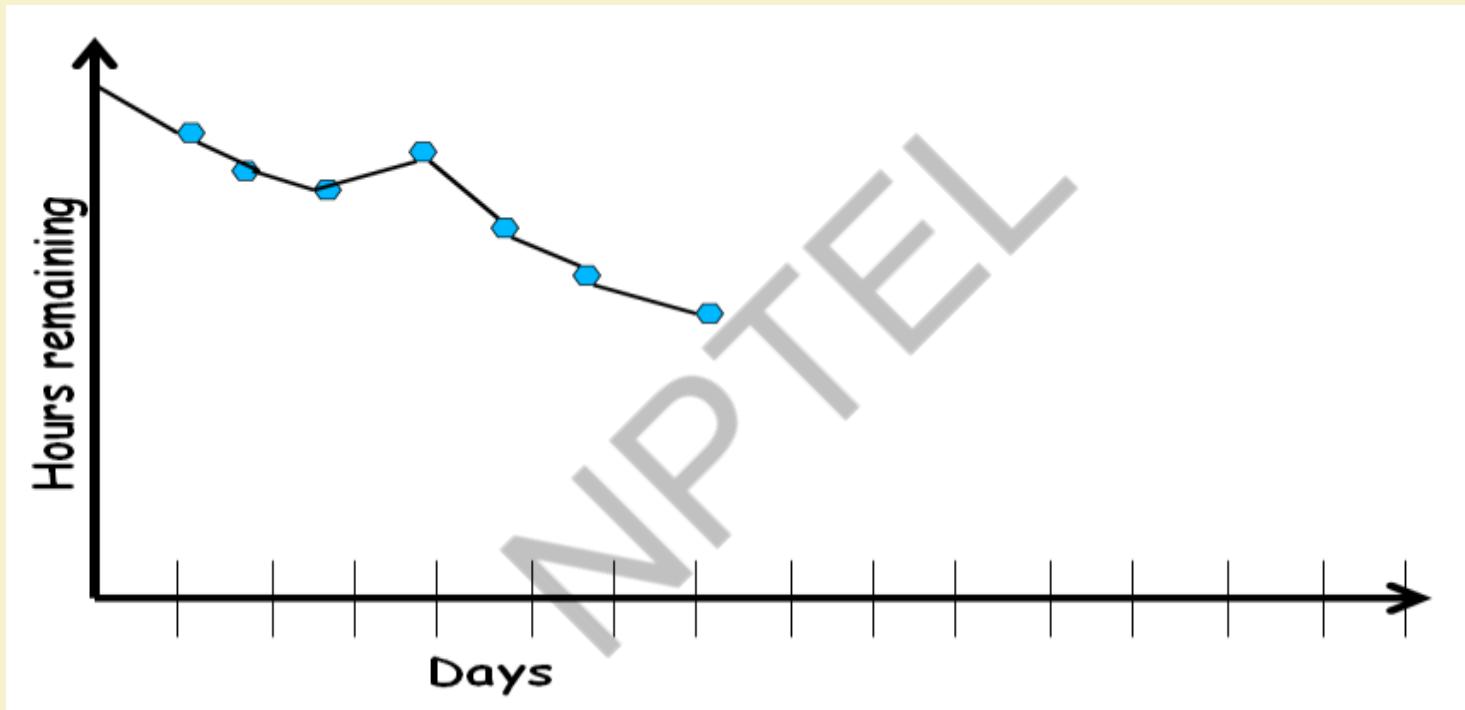


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Sprint Burn down Chart

- Depicts the total Sprint Backlog hours remaining per day
- Shows the estimated amount of time to release
- Ideally should burn down to zero to the end of the Sprint
- Actually is not a straight line

# Sprint Burndown Chart



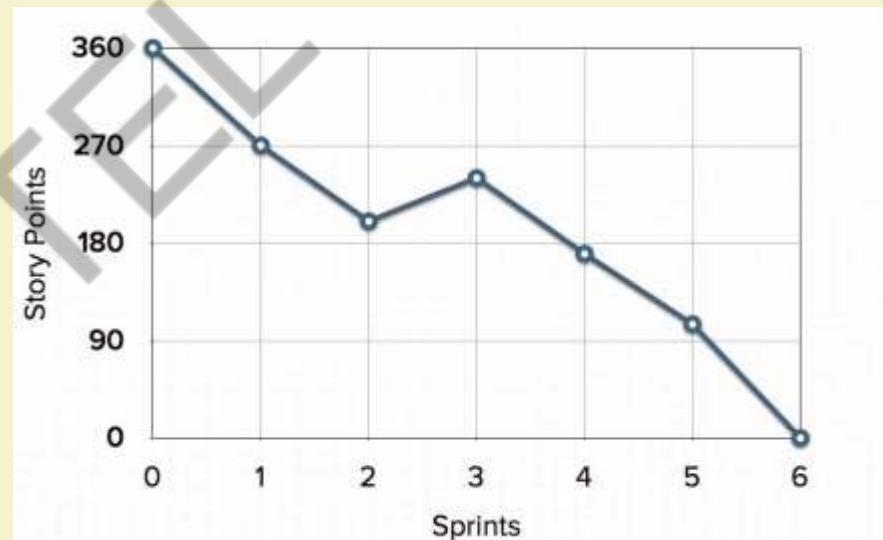
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

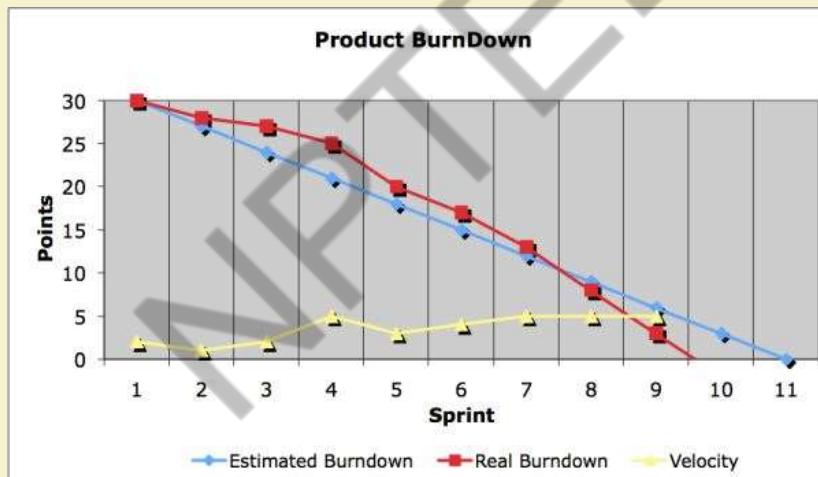
# Release Burndown Chart

- Will the final release be done on right time?
- How many more sprints?
- X-axis: sprints
- Y-axis: amount of story points remaining



# Product Burndown Chart

- Is a “big picture” view of project’s progress (all the releases)



# Scalability of Scrum

- A typical Scrum team is 6-10 people
- Jeff Sutherland - up to over 800 people
- "Scrum of Scrums" or what called "Meta-Scrum"
- Frequency of meetings is based on the degree of coupling between packets

# Thank You!!

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

Faculty Name  
Department Name

# Agile vs. Plan-Driven Processes

- 1. Small products and teams:
    - Scalability limited
  - 2. Untested on safety-critical products
  - 3. Good for dynamic, but expensive for stable environments.
  - 4. Require experienced personnel throughout
  - 5. Personnel thrive on freedom and chaos
- 
- 1. Large products and teams;
    - hard to scale down
  - 2. Proven for highly critical products;
  - 3. Good for stable:
    - But expensive for dynamic environments
  - 4. Require only few experienced personnel
  - 5. Personnel thrive on structure and order

# Agile Models



IIT KHARAGPUR

7/18/2020



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What is Agile Software Development?

- **Agile:** Easily moved, light, nimble, active software processes
- How agility achieved?
  - Fitting the process to the project
  - Avoidance of things that waste time



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Model

- To overcome the shortcomings of the waterfall model of development.
  - Proposed in mid-1990s
- The agile model was primarily designed:
  - To help projects to adapt to change requests
- In the agile model:
  - The requirements are decomposed into many small incremental parts that can be developed over one to four weeks each.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Ideology: Agile Manifesto

- Individuals and interactions over
  - process and tools
- Working Software over
  - comprehensive documentation
- Customer collaboration over
  - contract negotiation
- Responding to change over
  - following a plan

<http://www.agilemanifesto.org>



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- XP
- Scrum
- Unified process
- Crystal
- DSDM
- Lean

# Agile Methodologies

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Model: Principal Techniques

- **User stories:**
  - Simpler than use cases.
- **Metaphors:**
  - Based on user stories, developers propose a common vision of what is required.
- **Spike:**
  - Simple program to explore potential solutions.
- **Refactor:**
  - Restructure code without affecting behavior, improve efficiency, structure, etc.

- At a time, only one increment is planned, developed, deployed at the customer site.
  - No long-term plans are made.
- An iteration may not add significant functionality,
  - But still a new release is invariably made at the end of each iteration
  - Delivered to the customer for regular use.

## Agile Model: Nitty Gritty

# Methodology

- **Face-to-face communication favoured over written documents.**
- To facilitate face-to-face communication,
  - Development team to share a single office space.
  - Team size is deliberately kept small (5-9 people)
  - This makes the agile model most suited to the development of small projects.



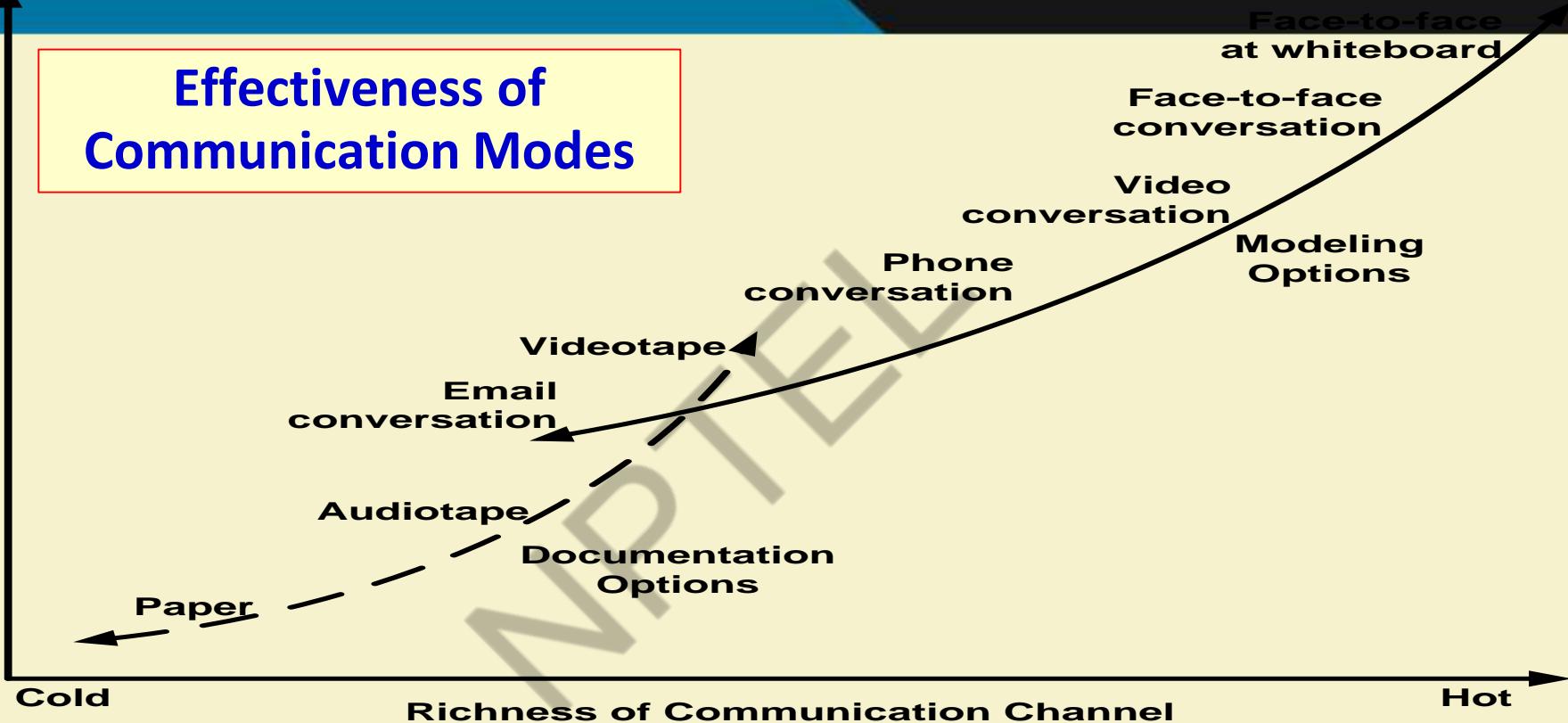
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Effectiveness of Communication Modes

Communication Effectiveness



Copyright 2002-2005 Scott W. Ambler  
Original Diagram Copyright 2002 Alistair Cockburn



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Model: Principles

- The primary measure of progress:
  - Incremental release of working software
- Important principles behind agile model:
  - Frequent delivery of versions --- once every few weeks.
  - Requirements change requests are easily accommodated.
  - Close cooperation between customers and developers.
  - Face-to-face communication among team members.

# Agile Documentation

- Travel light:
  - You need far less documentation than you think.
- Agile documents:
  - Are concise
  - Describe information that is less likely to change
  - Describe “good things to know”
  - Are sufficiently accurate, consistent, and detailed
- Valid reasons to document:
  - Project stakeholders require it
  - To define a contract model
  - To support communication with an external group
  - To think something through

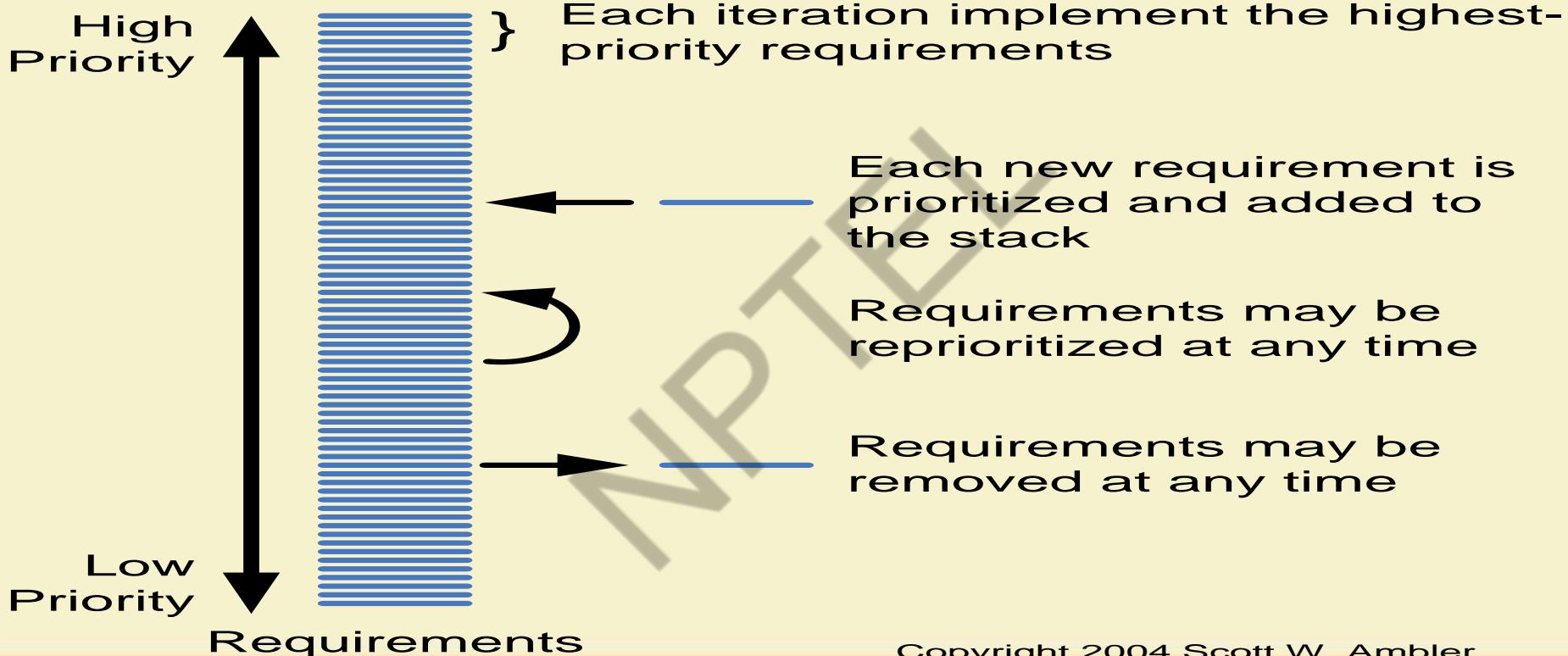


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Agile Software Requirements Management



Copyright 2004 Scott W. Ambler



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Adoption Detractors

- Sketchy definitions, make it possible to have
  - Inconsistent and diverse definitions
- High quality people skills required
- Short iterations inhibit long-term perspective
- Higher risks due to feature creep:
  - Harder to manage feature creep and customer expectations
  - Difficult to quantify cost, time, quality.

# Agile Model Shortcomings

- Derives agility through developing tacit knowledge within the team, rather than any formal document:
  - Can be misinterpreted...
  - External review difficult to get...
  - When project is complete, and team disperses, maintenance becomes difficult...

# Agile Model versus Iterative Waterfall Model

- The waterfall model steps through in a planned sequence:
  - Requirements-capture, analysis, design, coding, and testing .
- Progress is measured in terms of delivered artefacts:
  - Requirement specifications, design documents, test plans, code reviews, etc.
- In contrast agile model sequences:
  - Delivery of working versions of a product in several increments.

# Agile Model versus Iterative Waterfall Model

- As regards to similarity:
  - We can say that Agile teams use the waterfall model on a small scale.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Agile versus RAD Model

- Agile model does not recommend developing prototypes:
  - Systematic development of each incremental feature is emphasized.
- In contrast:
  - RAD is based on designing quick-and-dirty prototypes, which are then refined into production quality code.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Agile versus exploratory programming

- Similarity:
  - Frequent re-evaluation of plans,
  - Emphasis on face-to-face communication,
  - Relatively sparse use of documents.
- Agile teams, however, do follow defined and disciplined processes and carry out rigorous designs:
  - This is in contrast to chaotic coding in exploratory programming.

# Extreme Programming (XP)



IIT KHARAGPUR

7/18/2020



NPTEL ONLINE  
CERTIFICATION COURSES

# Extreme Programming Model

- Extreme programming (XP) was proposed by Kent Beck in 1999.
- The methodology got its name from the fact that:
  - **Recommends taking the best practices to extreme levels.**
  - If something is good, why not do it all the time.

## Taking Good Practices to Extreme

- **If code review is good:**
  - Always review --- **pair programming**
- **If testing is good:**
  - Continually write and execute test cases --- **test-driven development**
- **If incremental development is good:**
  - Come up with new increments every few days
- **If simplicity is good:**
  - Create the simplest design that will support only the currently required functionality.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Taking to Extreme

- **If design is good,**
  - everybody will design daily (refactoring)
- **If architecture is important,**
  - everybody will work at defining and refining the architecture (metaphor)
- **If integration testing is important,**
  - build and integrate test several times a day (continuous integration)



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# 4 Values

- **Communication:**
  - Enhance communication among team members and with the customers.
- **Simplicity:**
  - Build something simple that will work today rather than something that takes time and yet never used
  - May not pay attention for tomorrow
- **Feedback:**
  - System staying out of users is trouble waiting to happen
- **Courage:**
  - Don't hesitate to discard code



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Best Practices

- **Coding:**
  - without code it is not possible to have a working system.
  - Utmost attention needs to be placed on coding.
- **Testing:**
  - Testing is the primary means for developing a fault-free product.
- **Listening:**
  - Careful listening to the customers is essential to develop a good quality product.

# Extreme Development Activities

- **XP Planning an increment**

- Begins by creating “user stories”
- Agile team assesses each story and assigns a cost
- Few stories are grouped into a deliverable increment
- Delivery date planned

- **XP Design**

- Follows the KIS principle
- Encourage the use of CRC cards
- For difficult design problems, suggests the creation of “spike solutions”—a design prototype
- Encourages “refactoring”—refinement of the internal program design



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## • XP Coding

- Recommends the construction of unit test cases *before* coding commences (test-driven development)
- Encourages “pair programming”

## • XP Testing

- All unit tests are executed daily
- “Acceptance tests” are defined by the customer and executed to assess customer visible functionalities

Extreme  
Program  
Development  
Activities



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Full List of XP Practices

1. **Planning** – determine scope of the next release by combining business priorities and technical estimates
2. **Small releases** – put a simple system into production, then release new versions in very short cycles
3. **Metaphor** – all development is guided by a simple shared story of how the whole system works
4. **Simple design** – system is to be designed as simple as possible
5. **Testing** – programmers continuously write and execute unit tests

## Full List of XP Practices

7. **Refactoring** – programmers continuously restructure the system without changing its behavior to remove duplication and simplify
8. **Pair-programming** -- all production code is written with two programmers at one machine
9. **Collective ownership** – anyone can change any code anywhere in the system at any time.
10. **Continuous integration** – integrate and build the system many times a day – every time a task is completed.

# Full List of XP Practices

- 11. 40-hour week** – work no more than 40 hours a week as a rule
- 12. On-site customer** – a user is a part of the team and available full-time to answer questions
- 13. Coding standards** – programmers write all code in accordance with rules emphasizing communication through the code

# Emphasizes Test-Driven Development (TDD)

- Based on user story develop test cases
- Implement a quick and dirty feature every couple of days:
  - Get customer feedback
  - Alter if necessary
  - Refactor
- Take up next feature

# Project Characteristics that Suggest Suitability of Extreme Programming

- Projects involving new technology or research projects.
  - In this case, the requirements change rapidly and unforeseen technical problems need to be resolved.
- Small projects:
  - These are easily developed using extreme programming.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Practice Questions

- What are the stages of iterative waterfall model?
- What are the disadvantages of the iterative waterfall model?
- Why has agile model become so popular?
- What difficulties might be faced if no life cycle model is followed for a certain large project?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Suggest Suitable Life Cycle Model

- A software for an academic institution to automate its:
  - Course registration and grading
  - Fee collection
  - Staff salary
  - Purchase and store inventory
- The software would be developed by tailoring a similar software that was developed for another educational institution:
  - 70% reuse
  - 10% new code and 20% modification

# Practice Questions

- Which types of risks can be better handled using the spiral model compared to the prototyping model?
- Which type of process model is suitable for the following projects:
  - A customization software
  - A payroll software for contract employees that would be add on to an existing payroll software

# Practice Questions

- Which lifecycle model would you select for the following project which has been awarded to us by a mobile phone vendor:
  - A new mobile operating system by upgrading the existing operating system
  - Needs to work well efficiently with 4G systems
  - Power usage minimization
  - Directly upload backup data on a cloud infrastructure maintained by the mobile phone vendor



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Scrum



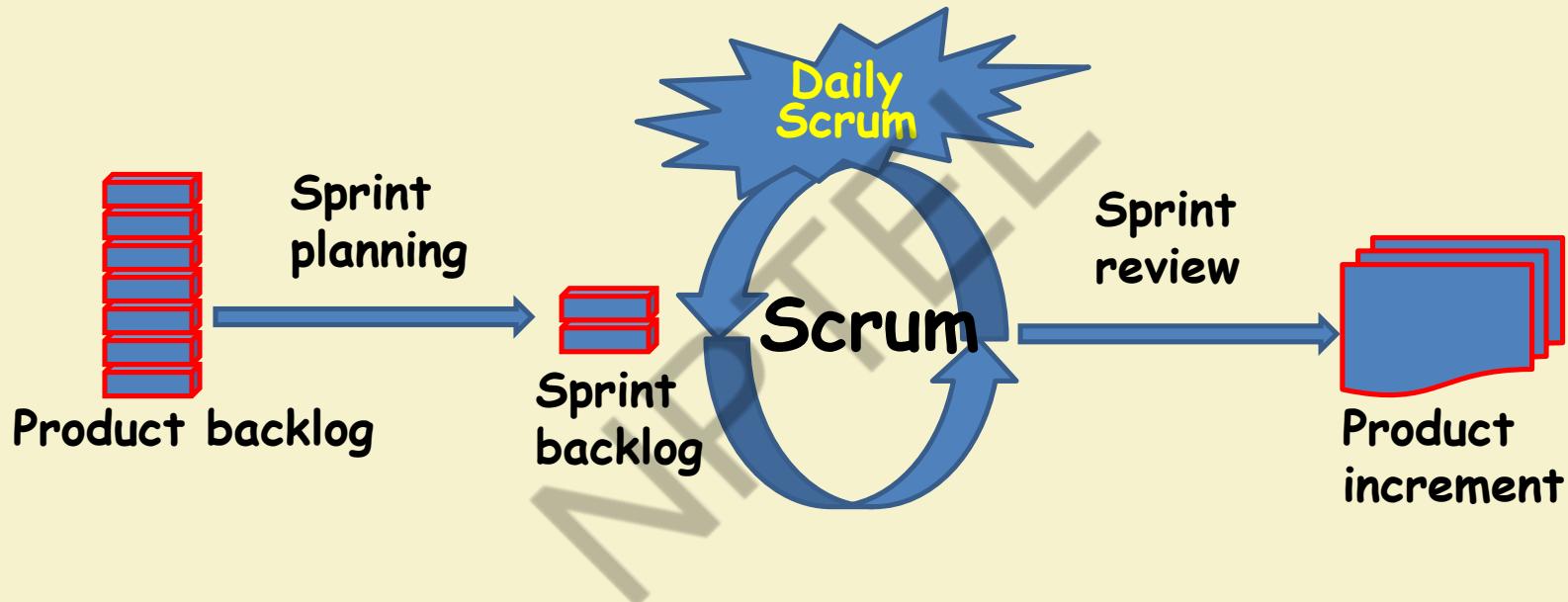
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Scrum: Characteristics

- One of the agile processes
- Self-organizing teams
- Product development progresses in a series of month-long **sprints**
- Requirements are listed in a **product backlog**



# Sprint

- Scrum projects progress in a series of “sprints”
  - Analogous to XP iterations or time boxes
  - Target duration is one month
- **Software increment is designed, coded, and tested during a sprint**
- **No changes entertained during a sprint**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Sprint

- Fundamental process flow of Scrum
- A month-long iteration, during which an incremental product functionality completed
- NO outside influence can interfere with the Scrum team during the Sprint
- Each day begins with the Daily Scrum Meeting

# Scrum Framework

- **Roles :** Product Owner, ScrumMaster, Team
- **Ceremonies :** Sprint Planning, Sprint Review, Sprint Retrospective, and Daily Scrum Meeting
- **Artifacts :** Product Backlog, Sprint Backlog, and Burndown Chart

# **Key Roles and Responsibilities in Scrum Process**

- **Product Owner**
  - Acts on behalf of customers to represent their interests.
- **Development Team**
  - Team of five to nine people with cross-functional skill sets.
- **Scrum Master (aka Project Manager)**
  - Facilitates scrum process and resolves impediments and acts as a buffer between the team and outside interference.

# Product Owner

- Defines the features of the product
- Decides on release date and content
- Prioritizes new features
- Adjusts features and priority every iteration, as needed
- Accepts or rejects work results.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# The Scrum Master

- Represents management
- Removes impediments
- Ensures that the team is fully functional and productive
- Shield the team from external interferences



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Scrum Team

- Typically 5-10 people
- Cross-functional
  - QA, Programmers, UI Designers, etc.
- Teams are self-organizing
- Membership can change only between sprints



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Ceremonies

- Sprint Planning Meeting
- Daily Scrum
- Sprint Review Meeting



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Sprint Planning

- Goal is to produce Sprint Backlog
- Product owner works with the Team to negotiate what Backlog Items the Team will work on in order to meet Release Goals
- Scrum Master ensures Team agrees to realistic goals



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Daily
- 15-minutes
- Stand-up meeting
- Not for problem solving
- Three questions:

1. What did you do yesterday
2. What will you do today?
3. What obstacles are in your way?

## Daily Scrum

## Daily Scrum

- Is NOT a problem solving session
- Is NOT a way to collect information about WHO is behind the schedule
- Is a meeting in which team members make informal commitments to each other and to the Scrum Master
- Is a good way for a Scrum Master to track the progress of the Team

- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features
- Informal
  - 2-hour prep time rule
- Participants
  - Customers
  - Management
  - Product Owner
  - Other teammates

## Sprint Review Meeting

# Product Backlog

- A list of all desired work on the project
  - Usually a combination of
    - **story-based work (“allow user to search and replace”)**
    - **task-based work (“improve exception handling”)**
- List is prioritized by the Product Owner.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Product Backlog

- Requirements for a system, expressed as a prioritized list of Backlog Items
  - **Managed and owned by Product Owner**
  - **Spreadsheet (typically)**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Sample Product Backlog

	Item #	Description	Est	By
<b>Very High</b>				
	1	<b>Finish database versioning</b>	16	KH
	2	<b>Get rid of unneeded shared Java in database</b>	8	KH
	-	<b>Add licensing</b>	-	-
	3	Concurrent user licensing	16	TG
	4	Demo / Eval licensing	16	TG
		<b>Analysis Manager</b>		
	5	File formats we support are out of date	160	TG
	6	Round-trip Analyses	250	MC
<b>High</b>				
	-	<b>Enforce unique names</b>	-	-
	7	In main application	24	KH
	8	In import	24	AM
	-	<b>Admin Program</b>	-	-
	9	Delete users	4	JM
	-	<b>Analysis Manager</b>	-	-
	10	When items are removed from an analysis, they should show up again in the pick list in lower 1/2 of the analysis tab	8	TG
	-	<b>Query</b>	-	-
	11	Support for wildcards when searching	16	T&A
	12	Sorting of number attributes to handle negative numbers	16	T&A
	13	Horizontal scrolling	12	T&A
	-	<b>Population Genetics</b>	-	-
	14	Frequency Manager	400	T&M
	15	Query Tool	400	T&M
	16	Additional Editors (which ones)	240	T&M
	17	Study Variable Manager	240	T&M
	18	Haplotypes	320	T&M
	19	<b>Add icons for v1.1 or 2.0</b>	-	-
	-	<b>Pedigree Manager</b>	-	-
	20	Validate Derived kindred	4	KH
<b>Medium</b>				
	-	<b>Explorer</b>	-	-
	21	Launch tab synchronization (only show queries/analyses for logged in users)	8	T&A
	22	Delete settings (?)	4	T&A



# Sprint Backlog

- A subset of Product Backlog Items, which define the work for a Sprint
  - **Created by Team members**
  - **Each Item has its own status**
  - **Updated daily**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Sprint Backlog during the Sprint

- Changes occur:
  - Team adds new tasks whenever they need to in order to meet the Sprint Goal
  - Team can remove unnecessary tasks
  - But: Sprint Backlog is only updated by the team
- Estimates are updated whenever there's new information

# Burn down Charts

- Are used to represent “work done”.
- Are remarkably simple but effective Information disseminators
- Three Types:
  - Sprint Burn down Chart (progress of the Sprint)
  - Release Burn down Chart (progress of release)
  - Product Burn down chart (progress of the Product)



IIT KHARAGPUR

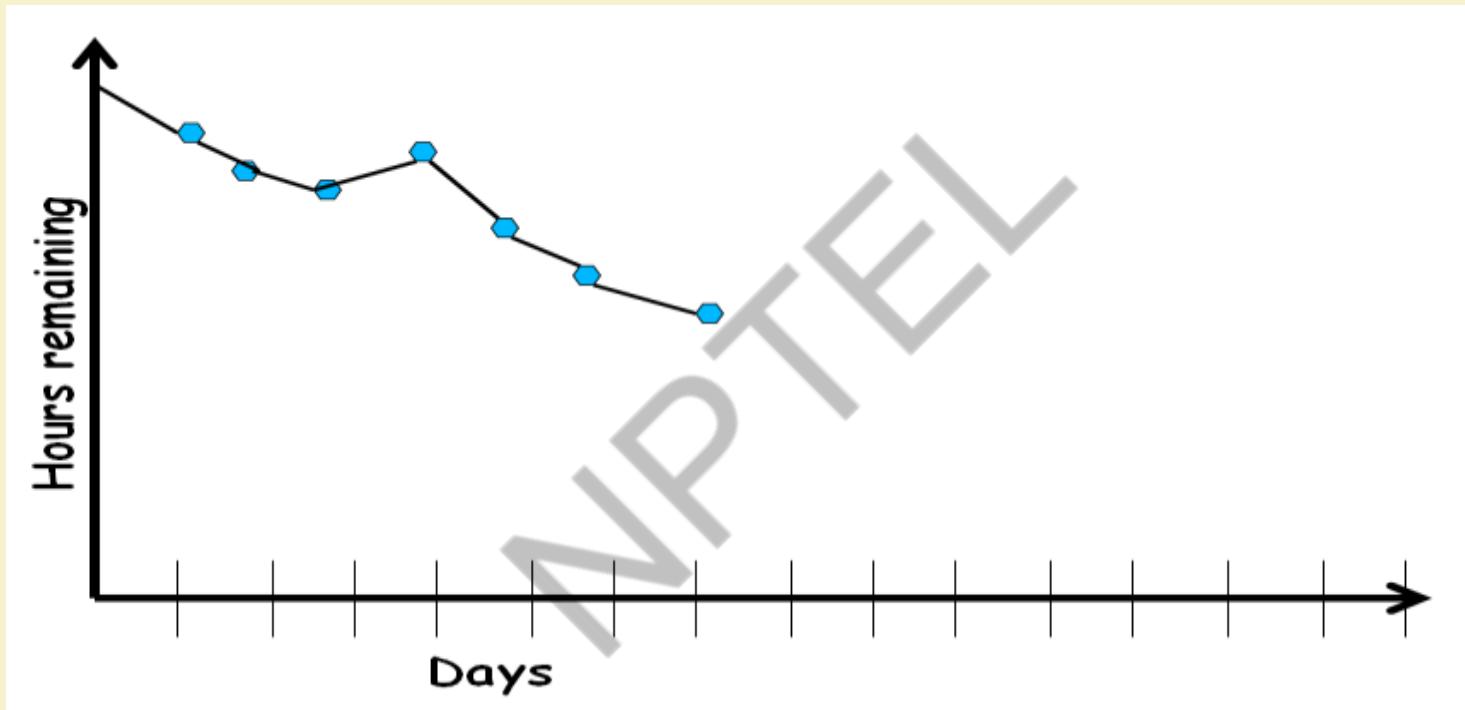


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Sprint Burn down Chart

- Depicts the total Sprint Backlog hours remaining per day
- Shows the estimated amount of time to complete
- Ideally should burn down to zero to the end of the Sprint
- Usually is not a straight line

# Sprint Burndown Chart



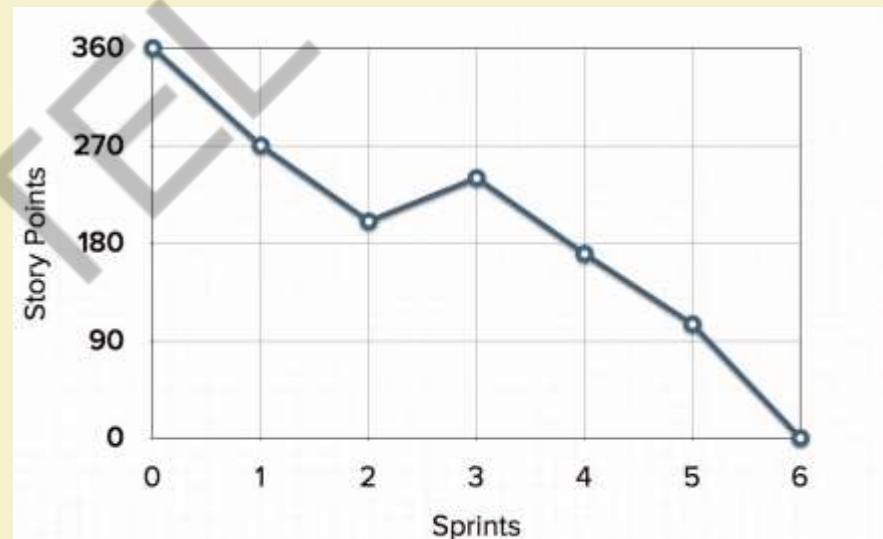
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

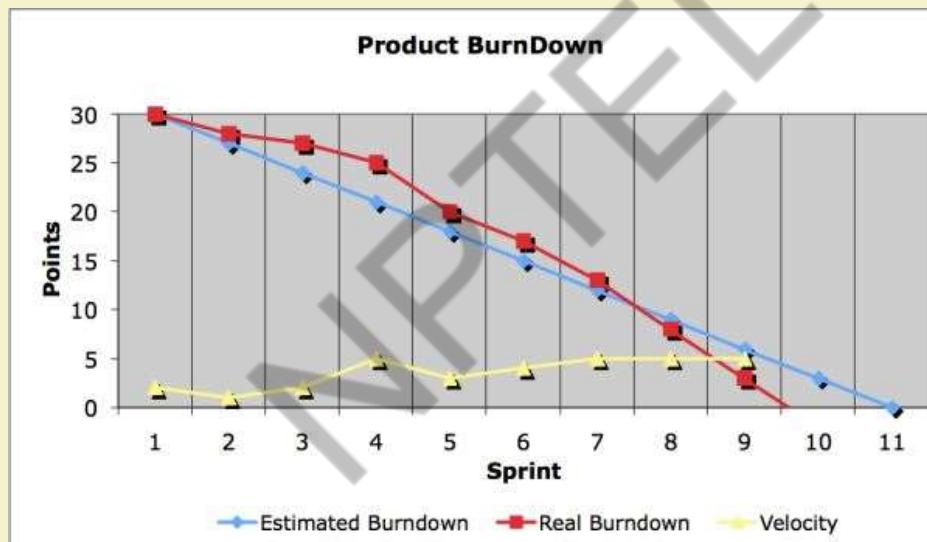
# Release Burndown Chart

- Will the next release be done on right time?
- How many more sprints?
- X-axis: sprints
- Y-axis: amount of story points remaining



# Product Burndown Chart

- Is a “big picture” view of project’s progress (all the releases)



# Scalability of Scrum

- A typical Scrum team is 6-10 people
- Jeff Sutherland proposed and experimented with up to over 800 people
- "Scrum of Scrums" also called "Meta-Scrum"

# **Requirements Analysis and Specification**

**RAJIB MALL**

**Computer Science and Engineering Department  
IIT KHARAGPUR**



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# What are Requirements?

- A Requirement is:
  - **A capability or condition required from the system.**
- What is involved in requirements analysis and specification?
  - **Determine what is expected by the client from the system. (Gather and Analyze)**
  - **Document those in a form that is clear to the client as well as to the development team members. (Document)**

# **Understanding and specifying requirements**

- **For toy problems:** understanding and specifying requirements is rather easy...
- **For industry-standard problems:** Probably the hardest, most problematic and error prone among development tasks...
- The task of requirements specification :
  - **Input:** User needs that are hopefully fully understood by the users.
  - **Output:** Precise statement of what the software will do.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Requirements for Products

- When a company plans to develop a generic product:
  - Who gives the requirements?
- **The sales personnel!**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Activities in Requirements Analysis and Specification

Requirements Gathering



Requirements Analysis



Requirements Specification



SRS Document

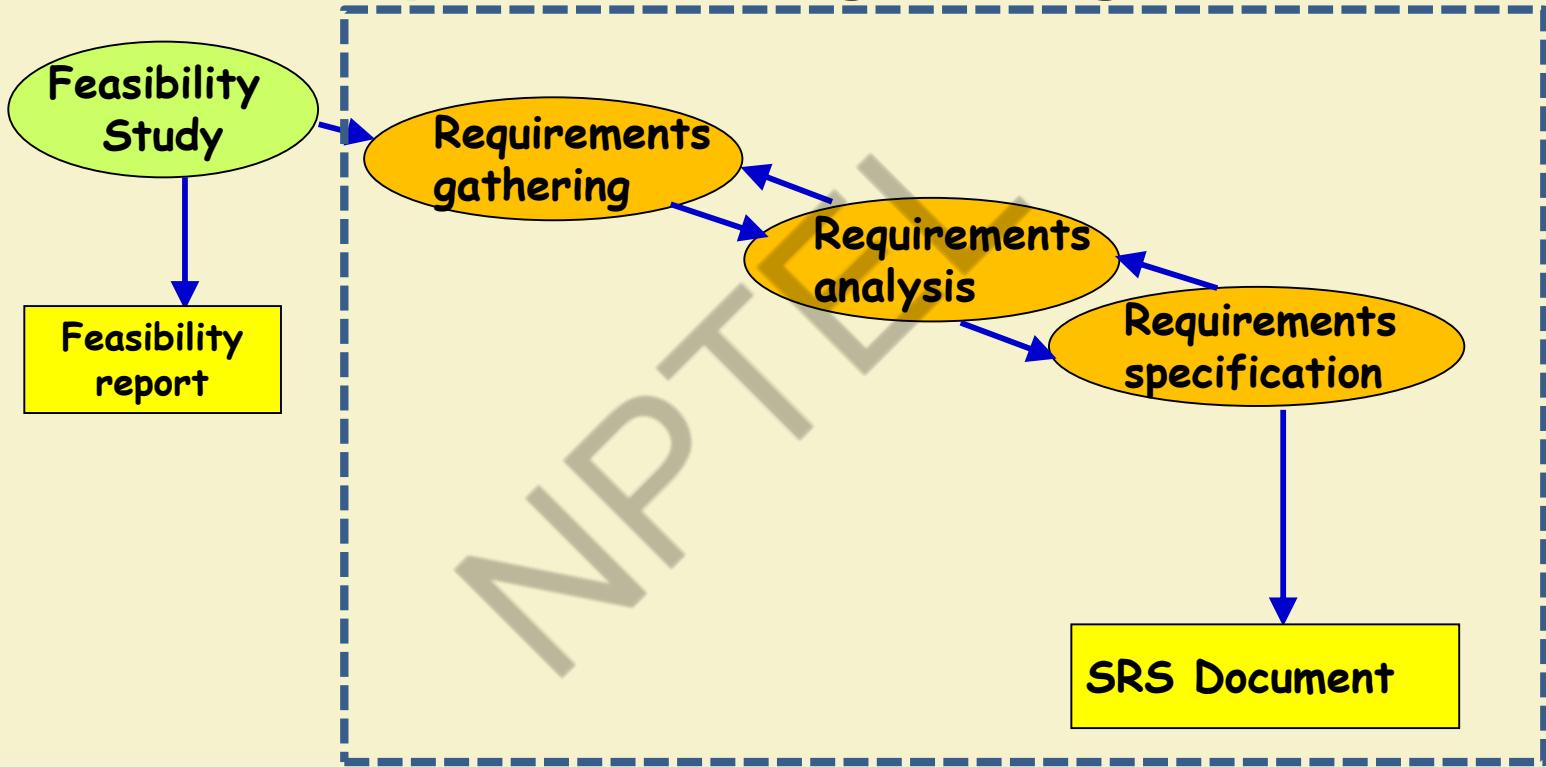


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Requirements Engineering Process



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Requirements Analysis and Specification

- Requirements Gathering:
  - Fully understand the user requirements.
- Requirements Analysis:
  - Remove inconsistencies, anomalies, etc. from requirements.
- Requirements Specification:
  - Document requirements properly in an SRS document.

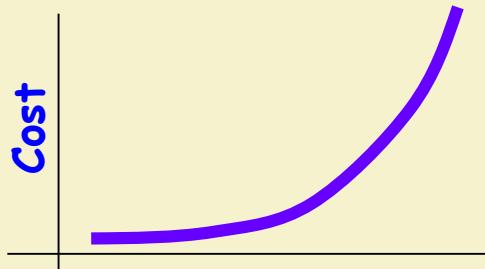
- **Good SRS reduces development cost:**

- Req. errors are expensive to fix later
- Req. changes cost a lot (typically 40% of requirements change later)
- Good SRS can minimize changes and errors
- **Substantial savings --- effort spent during req. saves multiple times that effort**

Need for SRS...

- **An Example:**

- Cost of fixing errors in req. , design , coding , acceptance testing and operation increases exponentially



- Establishes the basis for agreement between the customers and the suppliers
- Forms the starting point for development.
- Provide a basis for estimating costs and schedules.
- Provide a basis for validation and verification.
- Provide a basis for user manual preparation.
- Serves as a basis for later enhancements.

**What are the Uses of an SRS Document?**

# Forms A Basis for User Manual

- The SRS serves as the basis for writing User Manual for the software:
  - **User Manual: Describes the functionality from the perspective of a user --- An important document for users.**
  - Typically also describes how to carry out the required tasks with examples.



IIT KHARAGPUR

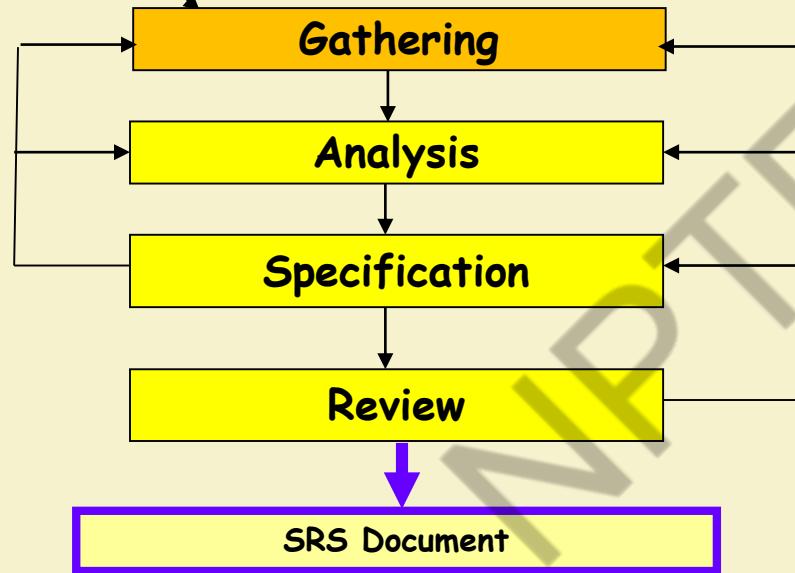


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# SRS Document: Stakeholders

- SRS intended for a diverse audience:
  - **Customers and users use it for validation, contract, ...**
  - **Systems (requirements) analysts**
  - **Developers, programmers to implement the system**
  - **Testers use it to check whether requirements have been met**
  - **Project Managers to measure and control the project**
- Different levels of detail and formality is needed for each audience
- Different templates for requirements specifications used by companies:
  - Often variations of **IEEE 830**

# Requirement process..



- Specification and review may lead to further gathering and analysis.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Thank You!!

NPTEL



IIT KHARAGPUR

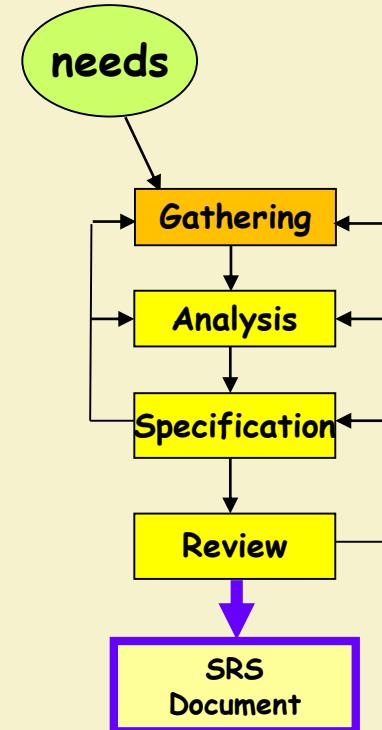


NPTEL ONLINE  
CERTIFICATION COURSES

Faculty Name  
Department Name

# How to Gather Requirements?

- Observe existing (manual) systems,
- Study existing procedures,
- Discuss with customer and end-users,
- Input and Output analysis
- Analyze what needs to be done



# Requirements Gathering Activities

- 1. Study existing documentation
- 2. Interview
- 3. Task analysis
- 4. Scenario analysis
- 5. Form analysis



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Requirements Gathering (CONT.)

- In the absence of a working system,
  - Lot of imagination and creativity are required.
- Interacting with the customer to gather relevant data:
  - Requires a lot of experience.

# Requirements Gathering (CONT.)

- Some desirable attributes of a good requirements analyst:
  - Good interaction skills,
  - Imagination and creativity,
  - Experience...

## **Case Study: Automation of Office Work at CSE Dept.**

- The academic, inventory, and financial information at the CSE department:
  - At present carried through manual processing by two office clerks, a store keeper, and two attendants.
- Considering the low budget he had at his disposal:
  - The HoD entrusted the work to a team of student volunteers.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Case Study: Automation of Office Work at CSE Dept.

- The team was first briefed by the HoD:
  - Concerning the specific activities to be automated.
- The analysts first discussed with the two office clerks:
  - Regarding their specific responsibilities (tasks) that were to be automated.
- The analyst also interviewed student and faculty representatives who would also use the software.

**Interview**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Case Study: Automation of Office Work at CSE Dept.

- For each task that a user needs the software to perform, they asked:
  - The steps through which these are to be performed.
  - The various scenarios that might arise for each task.
- Also collected the different types of forms that were being used.

**Task and Scenario Analysis**

**Form Analysis**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Case Study: Automation of Office Work at CSE Dept.

- The analysts understood the requirements for the system from various user groups:  
**Requirements Analysis**
  - Identified inconsistencies, ambiguities, incompleteness.
- Resolved the requirements problems through discussions with users:  
  - Resolved a few issues which the users were unable to resolve through discussion with the HoD.  
**Requirements Specification**
- Documented the requirements in the form of an SRS document.



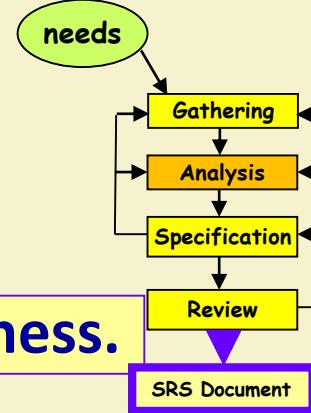
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Analysis of Gathered Requirements

- Main purpose of req. analysis:
  - Clearly understand user requirements,
  - Detect inconsistencies, ambiguities, and incompleteness.**
- Incompleteness and inconsistencies:
  - Resolved through further discussions with the end-users and the customers.



# Ambiguity

**"All customers must have the same control field."**

Do you notice any problems?

Multiple interpretations possible

1. All control fields have the same format.
2. One control field is issued for all customers.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Inconsistent Requirement

- Some part of the requirement:
  - contradicts some other requirement.
- **Example:**
  - One customer says turn off heater and open water shower when temperature  $> 100^{\circ}\text{C}$
  - Another customer says turn off heater and turn ON cooler when temperature  $> 100^{\circ}\text{C}$



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Some requirements not included:
  - Possibly due to oversight.
- Example:
  - The analyst has not recorded that when temperature falls below 90°C :
    - heater should be turned ON
    - water shower turned OFF.

Incomplete  
Requirement

# Analysis of the Gathered Requirements

- Requirements analysis involves:
  - Obtaining a clear, in-depth understanding of the software to be developed
  - Remove all ambiguities and inconsistencies from the initial customer perception of the problem.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Analysis of the Gathered Requirements (CONT.)

- It is quite difficult to obtain:
  - A clear, in-depth understanding of the problem:
    - Especially if there is no working model of the problem.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Analysis of the Gathered Requirements

(CONT.)

- Experienced analysts take considerable time:
  - Clearly understand the exact requirements the customer has in his mind.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Analysis of the Gathered Requirements

(CONT.)

- Experienced systems analysts know - often as a result of painful experiences ---
  - **“Without a clear understanding of the problem, it is impossible to develop a satisfactory system.”**

## Analysis of the Gathered Requirements

- Several things about the project should be clearly understood:
  - What is the problem?
  - What are the possible solutions to the problem?
  - What complexities might arise while solving the problem?



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Analysis of the Gathered Requirements

- Some anomalies and inconsistencies can be very subtle:
  - Escape even most experienced eyes.
  - If a formal specification of the system is constructed,
    - Many of the subtle anomalies and inconsistencies get detected.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Analysis of the Gathered Requirements<sub>(CONT.)</sub>

- After collecting all data regarding the system to be developed,
  - Remove all inconsistencies and anomalies from the requirements,
  - Systematically organize requirements into a Software Requirements Specification (SRS) document.



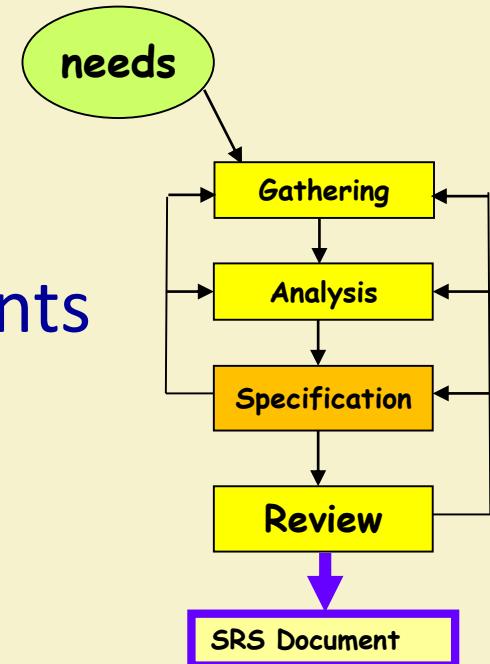
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Software Requirements Specification

- Main aim:
  - Systematically organize the requirements arrived during requirements analysis.
  - Document requirements properly.



# SRS Document

- As already pointed out--- useful in various contexts:
  - Statement of user needs
  - Contract document
  - Reference document
  - Definition for implementation

# SRS Document (CONT.)

- SRS document is known as **black-box specification**:
  - The system is considered as a black box whose internal details are not known.
  - Only its visible external (i.e. input/output) behavior is documented.



# SRS Document (CONT.)

- SRS document concentrates on:
  - What needs to be done in terms of input-output behaviour
  - Carefully avoids the solution (“how to do”) aspects.

# SRS Document (CONT.)

- The requirements at this stage:
  - Written using end-user terminology.
- If necessary:
  - Later a formal requirement specification may be developed from it.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

- **It should be concise**
  - and at the same time should not be ambiguous.
- **It should specify what the system must do**
  - and not say how to do it.
- **Easy to change.,**
  - i.e. it should be well-structured.
- **It should be consistent.**
- **It should be complete.**

Properties of a Good SRS  
Document

# Properties of a Good SRS Document (cont...)

- **It should be traceable**

- You should be able to trace which part of the specification corresponds to which part of the design, code, etc and vice versa.

- **It should be verifiable**

- e.g. “system should be user friendly” is not verifiable



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

**SRS should not include...**

- **Project development plans**

- E.g. cost, staffing, schedules, methods, tools, etc
  - Lifetime of SRS is until the software is made obsolete
  - Lifetime of development plans is much shorter

- **Product assurance plans**

- Configuration Management, Verification & Validation, test plans, Quality Assurance, etc
  - Different audiences
  - Different lifetimes

- **Designs**

- Requirements and designs have different audiences
- Analysis and design are different areas of expertise



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# SRS Document (CONT.)

- Four important parts:
  - Functional requirements,
  - External Interfaces
  - Non-functional requirements,
  - Constraints



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Functional Requirements

- Specifies all the functionality that the system should support
  - Heart of the SRS document:
  - Forms the bulk of the Document
- Outputs for the given inputs and the relationship between them
- Must specify behavior for invalid inputs too!



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Functional Requirement Documentation

- **Overview**
  - describe purpose of the function and the approaches and techniques employed
- **Inputs and Outputs**
  - sources of inputs and destination of outputs
  - quantities, units of measure, ranges of valid inputs and outputs
  - timing

# Functional Requirement Documentation

- **Processing**

- validation of input data
- exact sequence of operations
- responses to abnormal situations
- any methods (eg. equations, algorithms) to be used to transform inputs to outputs



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Nonfunctional Requirements

- Characteristics of the system which can not be expressed as functions:
  - **Maintainability,**
  - **Portability,**
  - **Usability,**
  - **Security,**
  - **Safety, etc.**

NPTEL



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Nonfunctional Requirements

- Reliability issues
- Performance issues:
  - **Example:** How fast can the system produce results?
    - At a rate that does not overload another system to which it supplies data, etc.
    - Response time should be less than 1sec 90% of the time
    - Needs to be measurable (verifiability)

# Constraints

- Hardware to be used,
- Operating system
  - or DBMS to be used
- Capabilities of I/O devices
- Standards compliance
- Data representations by the interfaced system



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- User interfaces
- Hardware interfaces
- Software interfaces
- Communications interfaces with other systems
- File export formats

## External Interface Requirements



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Goals of Implementation

- Goals describe things that are desirable of the system:
  - But, would not be checked for compliance.
  - For example,
    - Reusability issues
    - Functionalities to be developed in future



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# IEEE 830-1998 Standard for SRS

- Title
  - Table of Contents
  - 1. Introduction
    - 1.1 Purpose
    - 1.2 Scope
    - 1.3 Definitions, Acronyms, and Abbreviations
    - 1.4 References
    - 1.5 Overview
  - 2. Overall Description
  - 3. Specific Requirements
  - Appendices
  - Index
- 
- Describe purpose of the system
  - Describe intended audience
- What the system will and will not do
- Define the vocabulary of the SRS (may also be in appendix)
- List all referenced documents and their sources SRS (may also be in appendix)
- Describe how the SRS is organized



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# IEEE 830-1998 Standard – Section 2 of SRS

- Title
  - Table of Contents
  - 1. Introduction
  - 2. Overall Description
    - 2.1 Product Perspective
    - 2.2 Product Functions
    - 2.3 User Characteristics
    - 2.4 Constraints
    - 2.5 Assumptions and Dependencies
  - 3. Specific Requirements
  - 4. Appendices
  - 5. Index
- Present the business case and operational concept of the system
  - Describe external interfaces: system, user, hardware, software, communication
  - Describe constraints: memory, operational, site adaptation
- Summarize the major functional capabilities
- Describe technical skills of each user class
- Describe other constraints that will limit developer's options; e.g., regulatory policies; target platform, database, network, development standards requirements



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- ...
- 1. Introduction
- 2. Overall Description
- 3. Specific Requirements
  - 3.1 External Interfaces
  - 3.2 Functions
  - 3.3 Performance Requirements
  - 3.4 Logical Database Requirements
  - 3.5 Design Constraints
  - 3.6 Software System Quality Attributes
  - 3.7 Object Oriented Models
- 4. Appendices
- 5. Index

## IEEE 830-1998 Standard – Section 3 of SRS (1)

Specify software requirements in sufficient detail so that designers can design the system and testers can verify whether requirements met.

State requirements that are externally perceivable by users, operators, or externally connected systems

Requirements should include, at the least, a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

...

- 1. Introduction

- 2. Overall Description

- 3. Specific Requirements

- 3.1 External Interfaces

- 3.2 Functions

- 3.3 Performance Requirements

- 3.4 Logical Database Requirements

- 3.5 Design Constraints

- 3.6 Software System Quality Attributes

- 3.7 Object Oriented Models

## IEEE 830-1998 Standard – Section 3 of SRS (2)

**• Detail all inputs and outputs  
(complement, not duplicate, information presented in section 2)  
• Examples: GUI screens, file formats**

**• Include detailed specifications of each use case, including  
collaboration and other diagrams useful for this purpose**

**Types of Data entities and their relationships**

**Standards compliance and specific software/hardware to be used**

**• Class Diagram, State and Collaboration Diagram, Activity Diagrams etc.**

- 4. Appendices

- 5. Index



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## IEEE 830-1998 Standard – Templates

- Section 3 (Specific Requirements) can be organized in several different ways based on
  - Modes
  - User classes
  - Concepts (object/class)
  - Features
  - Stimuli



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- **SPECIFIC REQUIREMENTS**

### **3.1 Functional Requirements**

#### **3.1.1 Subject Registration**

- The subject registration requirements are concerned with functions regarding subject registration which includes students selecting, adding, dropping, and changing a subject.

- **F-001:**

- The system shall allow a student to register a subject.

- **F-002:**

- It shall allow a student to drop a course.

- **F-003:**

- It shall support checking how many students have already registered for a course.

## **Example Section 3 of SRS of Academic Administration Software**

## Design Constraints (3.2)

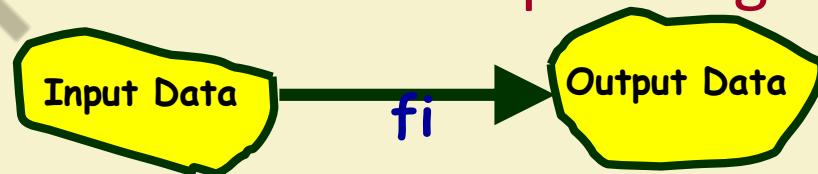
- **3.2 Design Constraints**
- **C-001:**
  - AAS shall provide user interface through standard web browsers.
- **C-002:**
  - AAS shall use an open source RDBMS such as Postgres SQL.
- **C-003:**
  - AAS shall be developed using the JAVA programming language

## Non-functional requirements

- **3.3 Non-Functional Requirements**
- **N-001:**
  - AAS shall respond to query in less than 5 seconds.
- **N-002:**
  - AAS shall operate with zero down time.
- **N-003:**
  - AAS shall allow upto 100 users to remotely connect to the system.
- **N-004:**
  - The system will be accompanied by a well-written user manual.

# Functional Requirements

- It is desirable to consider every system as:
  - Performing a set of functions  $\{f_i\}$ .
- Each function  $f_i$  considered as:
  - Transforming a set of input data to corresponding output data.



# Example: Functional Requirement

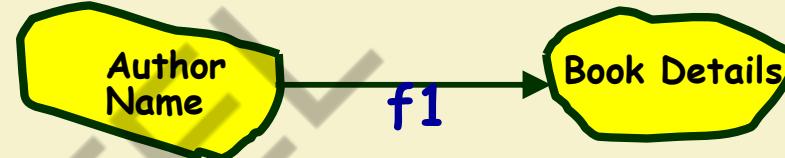
- F1: Search Book

- Input:

- an author's name:

- Output:

- details of the author's books and the locations of these books in the library.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Functional requirements describe:

- A set of high-level requirements
- Each high-level requirement:
  - takes in some data from the user
  - outputs some data to the user
- Each high-level requirement:
  - might consist of a set of identifiable sub-functions

Functional Requirements



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- For each high-level requirement:
  - A function is described in terms of:
    - Input data set
    - Output data set
    - Processing required to obtain the output data set from the input data set.

Functional Requirements



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- A high-level function is one: **Is it a Functional Requirement?**
  - Using which the user can get some useful piece of work done.
- Can the receipt printing work during withdrawal of money from an ATM:
  - Be called a functional requirement?
- A high-level requirement typically involves:
  - Accepting some data from the user,
  - Transforming it to the required response, and then
  - Outputting the system response to the user.

# Use Cases

- A use case is a term in UML:
  - Represents a high level functional requirement.
- Use case representation is more well-defined and has agreed documentation:
  - Compared to a high-level functional requirement and its documentation
  - Therefore many organizations document the functional requirements in terms of use cases



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example Functional Requirements

- Req. 1:
  - Once user selects the “search” option,
    - he is asked to enter the key words.
  - The system should output details of all books
    - whose title or author name matches any of the key words entered.
    - Details include: Title, Author Name, Publisher name, Year of Publication, ISBN Number, Catalog Number, Location in the Library.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example Functional Requirements

- Req. 2:
  - When the “renew” option is selected,
    - The user is asked to enter his membership number and password.
  - After password validation,
    - The list of the books borrowed by him are displayed.
  - The user can renew any of the books:
    - By clicking in the corresponding renew box.

# High-Level Function: A Closer View

- A high-level function:
  - Usually involves a series of interactions between the system and one or more users.
- Even for the same high-level function,
  - There can be different interaction sequences (or scenarios)
  - Due to users selecting different options or entering different data items.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Examples of Bad SRS Documents

- Unstructured Specifications:

- **Narrative essay --- one of the worst types of specification document:**

- Difficult to change,
    - Difficult to be precise,
    - Difficult to be unambiguous,
    - Scope for contradictions, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Examples of Bad SRS Documents

- Noise:
  - Presence of text containing information irrelevant to the problem.
- Silence:
  - Aspects important to proper solution of the problem are omitted.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Examples of Bad SRS Documents

- Overspecification:

- Addressing “how to” aspects
- For example, “Library member names should be stored in a sorted descending order”
- Overspecification restricts the solution space for the designer.

- Contradictions:

- Contradictions might arise
  - if the same thing described at several places in different ways.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Examples of Bad SRS Documents

- **Ambiguity:**

- Literary expressions
- Unquantifiable aspects, e.g. “good user interface”

- **Forward References:**

- References to aspects of problem
  - defined only later on in the text.

- **Wishful Thinking:**

- Descriptions of aspects
  - for which realistic solutions will be hard to find.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Suggestions for Writing Good Quality Requirements

- Keep sentences and paragraphs short.
- Use active voice.
- Use proper grammar, spelling, and punctuation.
- Use terms consistently and define them in a glossary.
- To see if a requirement statement is sufficiently well defined,
  - Read it from the developer's perspective



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Suggestions for Writing Good Quality Requirements

- Split a requirement into multiple sub-requirements:
  - Because each will require separate test cases and because each should be separately traceable.
  - If several requirements are strung together in a paragraph, it is easy to overlook one during construction or testing.



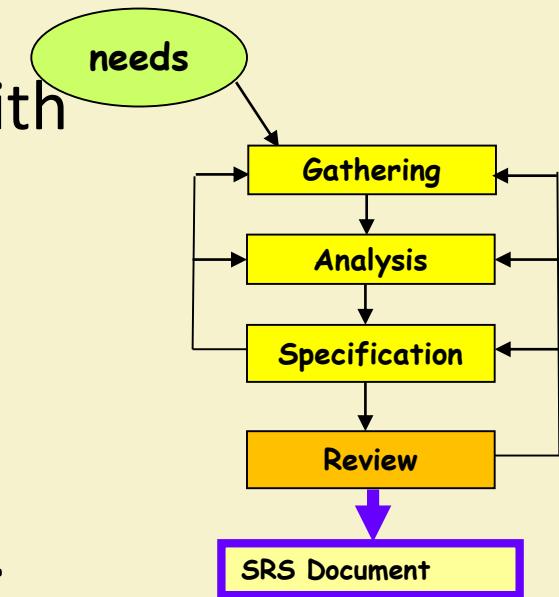
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# SRS Review

- Review done by the Developers along with the user representatives.
- To verify that SRS confirms to the actual user requirements
- To detect defects early and correct them.
- Review typically done using standard inspection process:
  - Checklists.



# Representation of complex processing logic

- Decision trees
- Decision tables

NPTEL



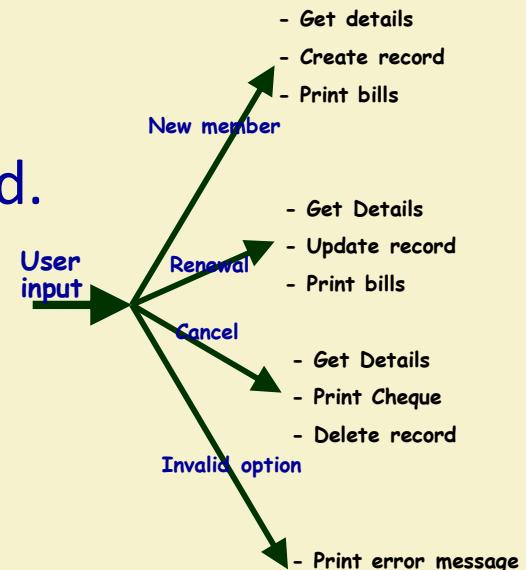
IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

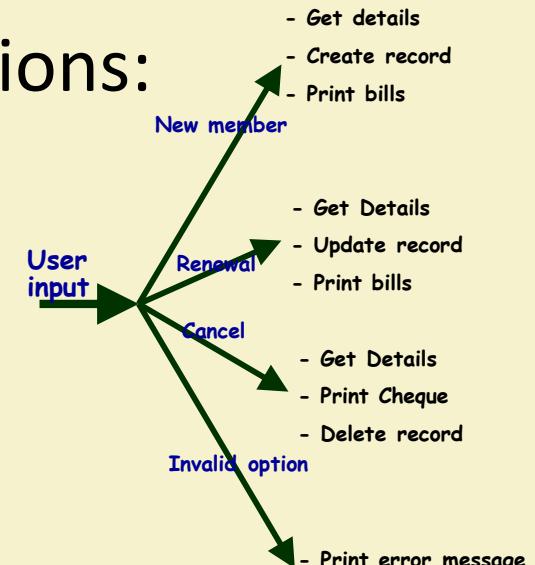
# Decision Trees

- Decision trees:
  - Edges of a decision tree represent conditions
  - Leaf nodes represent actions to be performed.
- A decision tree gives a graphic view of:
  - Logic involved in decision making
  - Corresponding actions taken.



# Example: LMS

- A Library Membership automation Software (LMS) should support the following three options:
  - New member,
  - Renewal,
  - Cancel membership.

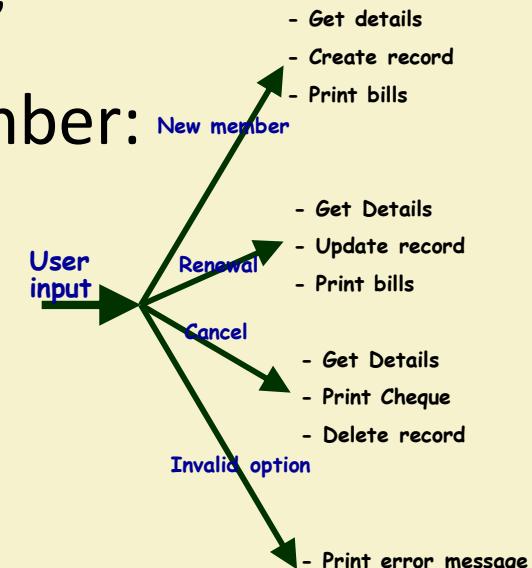


# Example: LMS

- When the new member option is selected,

– The software asks details about the member:

- name,
- address,
- phone number, etc.



# Example<sub>(cont.)</sub>

- If proper information is entered,
  - A membership record for the member is created
  - A bill is printed for the annual membership charge plus the security deposit payable.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example<sub>(cont.)</sub>

- If the renewal option is chosen,
  - LMS asks the member's name and his membership number
    - checks whether he is a valid member.
  - If the name represents a valid member,
    - the membership expiry date is updated and the annual membership bill is printed,
    - otherwise an error message is displayed.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example<sub>(cont.)</sub>

- If the cancel membership option is selected and the name of a valid member is entered,
  - The membership is cancelled,
  - A cheque for the balance amount due to the member is printed
  - The membership record is deleted.

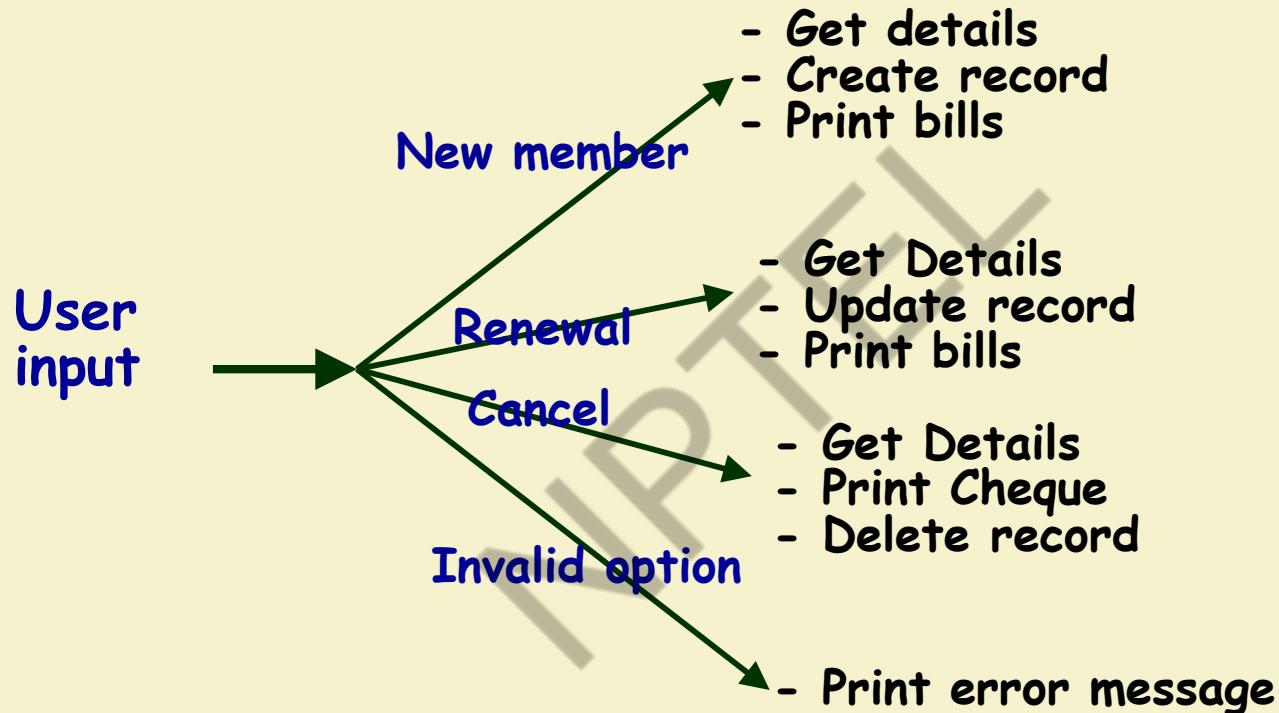


IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Decision Tree



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Decision Table

- Decision tables specify:
  - Which variables are to be tested
  - What actions are to be taken if the conditions are true,
  - The order in which decision making is performed.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Decision Table

- A decision table shows in a tabular form:
  - Processing logic and corresponding actions
- Upper rows of the table specify:
  - The variables or conditions to be evaluated
- Lower rows specify:
  - The actions to be taken when the corresponding conditions are satisfied.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Decision Table

- In technical terminology,
  - A column of the table is called a rule.
  - A rule implies:
    - If a condition is true, then execute the corresponding action.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- **Conditions**

Valid selection	NO	YES	YES	YES
New member	--	YES	NO	NO
Renewal	--	NO	YES	NO
Cancellation	--	NO	NO	YES

**Example**

- **Actions**

Display error message	-	-	-	-
Ask member's name etc.	-	-	-	-
Build customer record	--	--	--	--
Generate bill	--	-	-	-
Ask membership details	--	-	-	-
Update expiry date	--	--	--	-
Print cheque	-	-	-	-
Delete record	--	--	--	-



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Exercise

- Develop decision table for the following:
  - If the flight is more than half-full and ticket cost is more than Rs. 3000, free meals are served unless it is a domestic flight. The meals are charged on all domestic flights.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Thank You!!

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

RAJIB MALL  
CSE



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

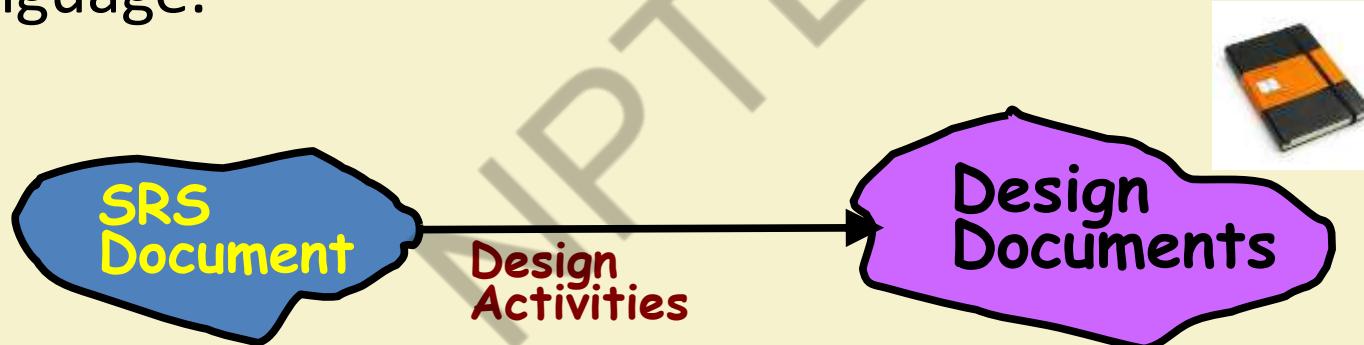
# Software Design

Rajib Mall

CSE Department  
IIT KHARAGPUR

# What is Achieved during design phase?

- Transformation of SRS document to Design document:
  - A form easily implementable in some programming language.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Items Designed During Design Phase

- Module structure,
- Control relationship among the modules
  - call relationship or invocation relationship
- Interface among different modules,
  - data items exchanged among different modules,
- Data structures of individual modules,
- algorithms for individual modules.



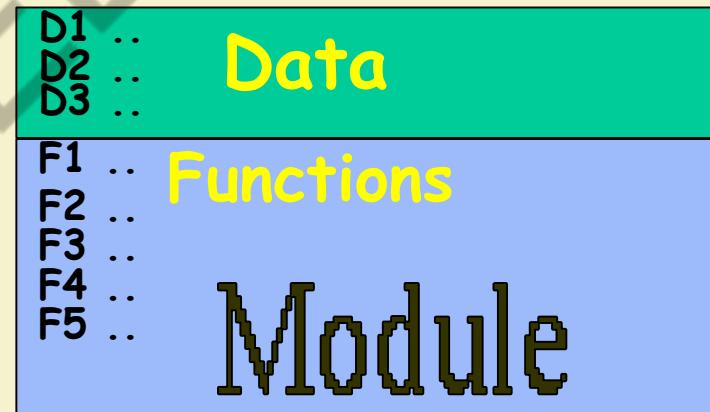
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Module

- A module consists of:
  - several functions
  - associated data structures.

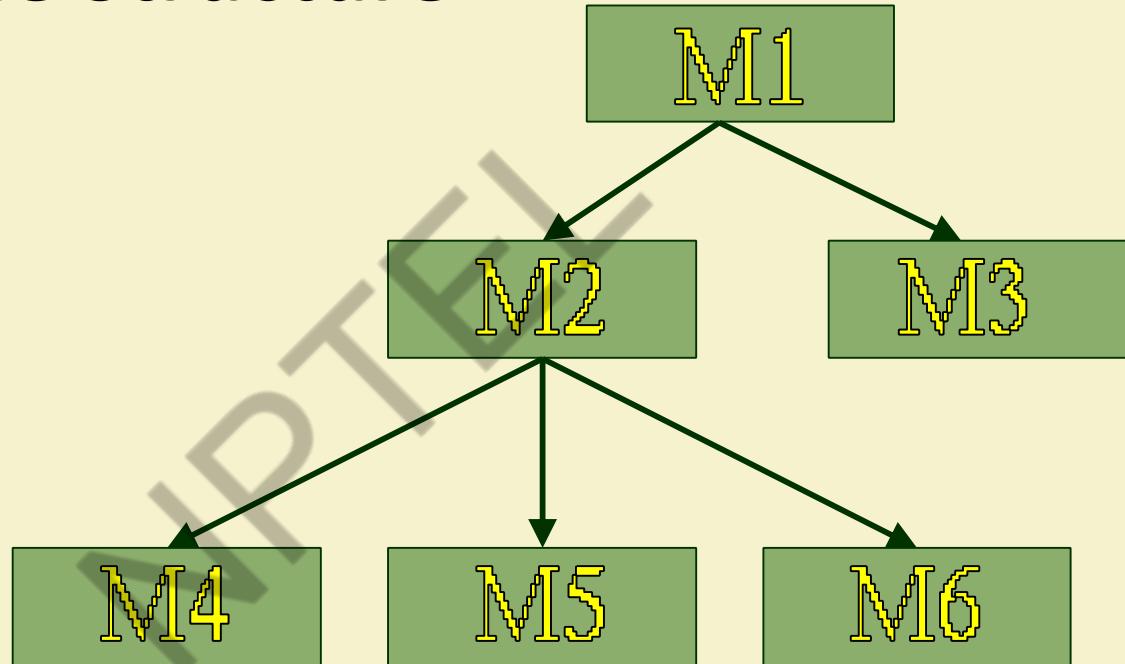


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Module Structure



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Iterative Nature of Design

- Good software designs:
  - Seldom arrived through a single step procedure:
  - But through a series of steps and iterations.

# Stages in Design

- Design activities are usually classified into two stages:
  - **Preliminary (or high-level) design**
  - **Detailed design.**
- Meaning and scope of the two stages:
  - vary considerably from one methodology to another.



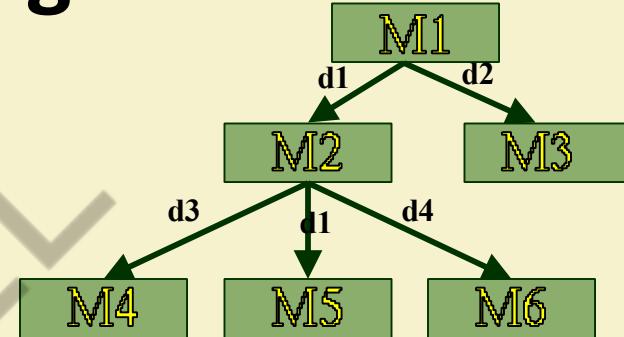
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# High-level design

- Identify:
  - modules
  - control relationships among modules
  - interfaces among modules.



# High-level design

- The outcome of high-level design:
  - program structure, also called software architecture.



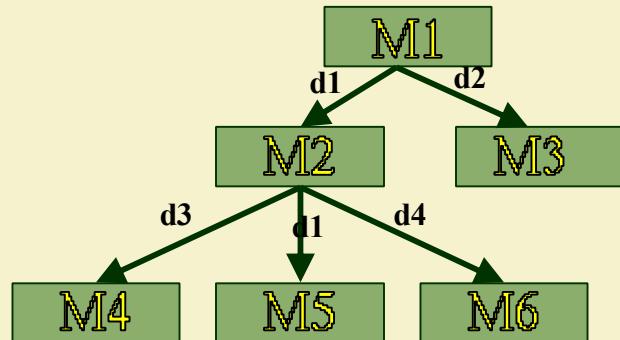
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# High-level Design

- Several notations are available to represent high-level design:
  - Usually a tree-like diagram called **structure chart** is used.
  - Other notations:
    - Jackson diagram or Warnier-Orr diagram can also be used.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Detailed design

- For each module, design for it:
  - data structure
  - algorithms
- Outcome of detailed design:
  - module specification.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# A fundamental question

- How to distinguish between good and bad designs?
  - Unless we know what a good software design is:
    - we can not possibly design one.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Good and bad designs

- There is no unique way to design a software.
- Even while using the same design methodology:
  - different engineers can arrive at very different designs.
- Need to determine which is a better design.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What Is a Good Software Design?

- Should implement all functionalities of the system correctly.
- **Should be easily understandable.**
- Should be efficient.
- Should be easily amenable to change,
  - i.e. easily maintainable.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What Is Good Software Design?

- Understandability of a design is a major issue:
  - Largely determines goodness of a design:
  - a design that is easy to understand:
    - also easy to maintain and change.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What Is a Good Software Design?

- Unless a design is easy to understand,
  - Tremendous effort needed to maintain it
  - We already know that about 60% effort is spent in maintenance.
- If the software is not easy to understand:
  - maintenance effort would increase many times.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# How to Improve Understandability?

- Use consistent and meaningful names
  - for various design components,
- Design solution should consist of:
  - A set of well decomposed modules (**modularity**),
- Different modules should be neatly arranged in a hierarchy:
  - A tree-like diagram.
  - Called Layering

# Modularity

- Modularity is a fundamental attributes of any good design.
  - Decomposition of a problem into a clean set of modules:
  - Modules are almost independent of each other
  - Based on **divide and conquer principle.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Modularity

- If modules are independent:
  - Each module can be understood separately,
    - reduces complexity greatly.
  - To understand why this is so,
    - remember that it is very difficult to break a bunch of sticks but very easy to break the sticks individually.

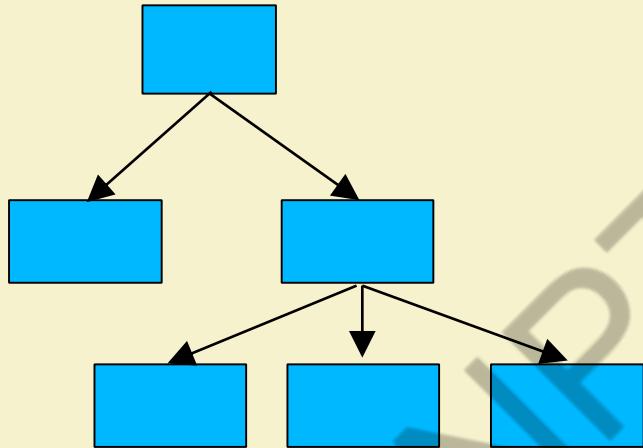


IIT KHARAGPUR

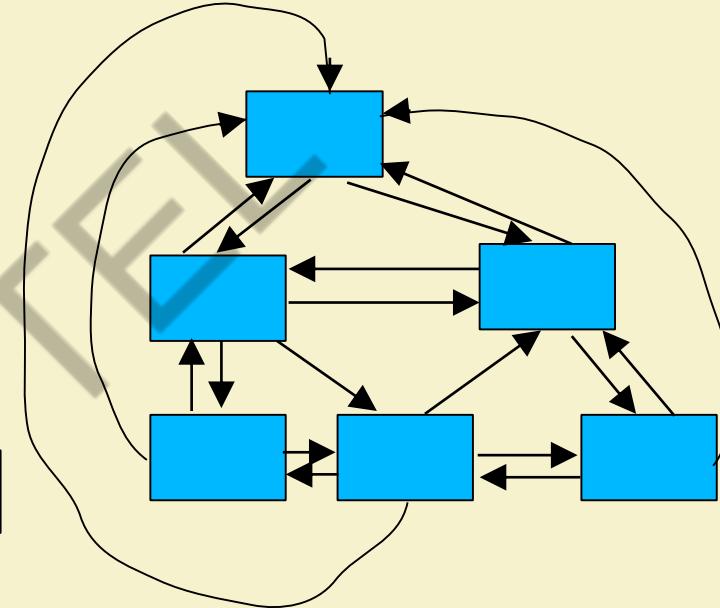


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Layering



Superior



Inferior



IIT KHARAGPUR

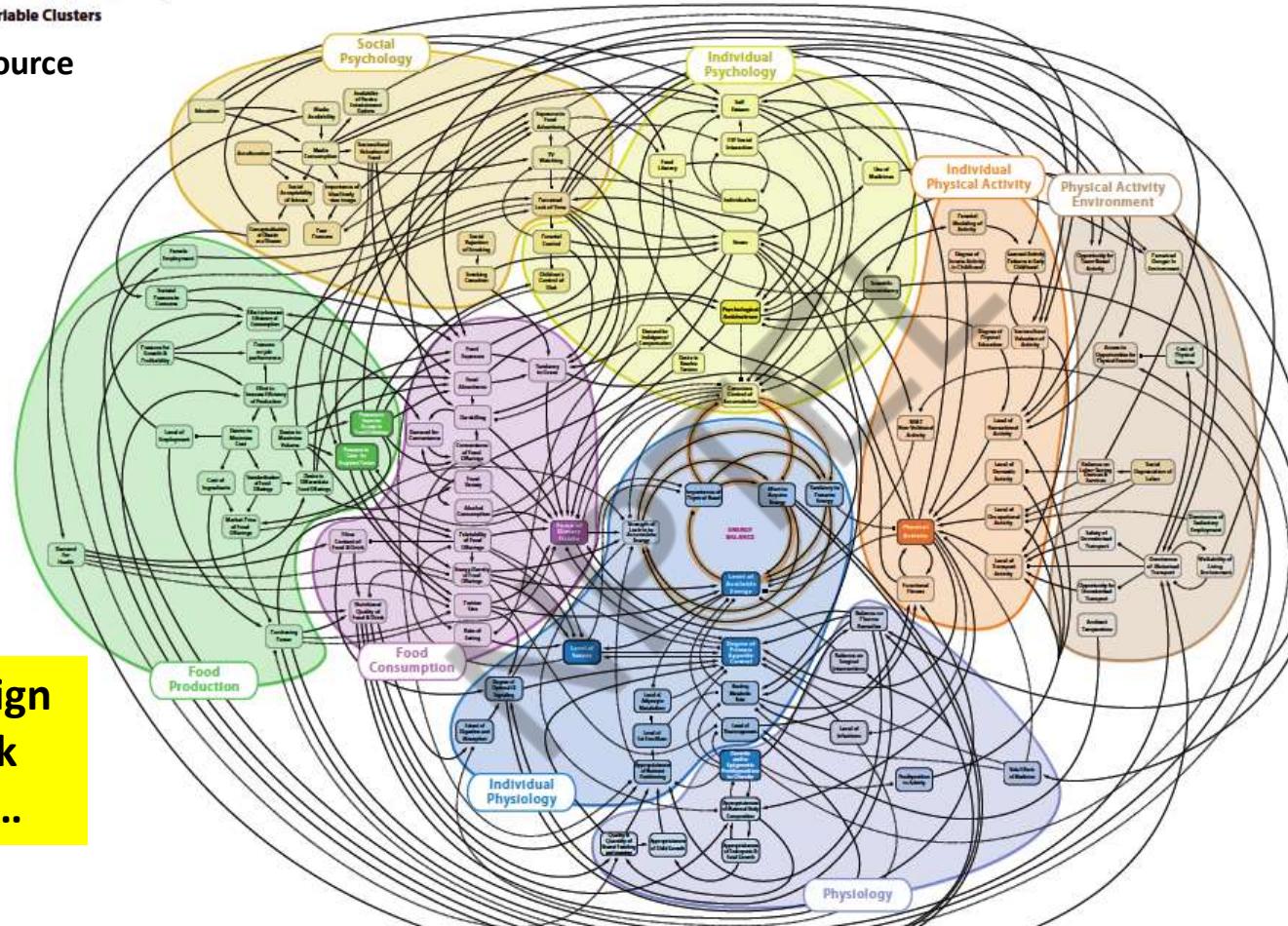


NPTEL  
ONLINE  
CERTIFICATION COURSES

## Obesity System Map

### Variable Clusters

:Source



Bad design  
may look  
like this...



# Modularity

- In technical terms, modules should display:
  - **high cohesion**
  - **low coupling.**
- We next discuss:
  - cohesion and coupling.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Modularity

- Arrangement of modules in a hierarchy ensures:
  - **Low fan-out**
  - **Abstraction**

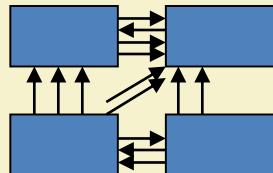
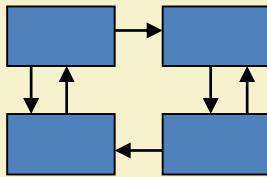
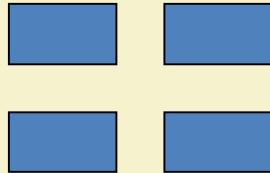


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Coupling: Degree of dependence among components



High coupling makes modifying parts of the system difficult, e.g., modifying a component affects all the components to which the component is connected.

Source:

Pfleeger, S., *Software Engineering Theory and Practice*. Prentice Hall, 2001.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Cohesion and Coupling

- Cohesion is a measure of:
  - functional strength of a module.
  - **A cohesive module performs a single task or function.**
- Coupling between two modules:
  - **A measure of the degree of interdependence or interaction between the two modules.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Cohesion and Coupling

- A module having **high cohesion and low coupling**:
  - **Called functionally independent** of other modules:
    - A functionally independent module needs very little help from other modules and therefore has minimal interaction with other modules.



IIT KHARAGPUR



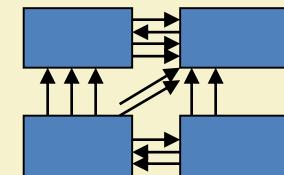
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Advantages of Functional Independence

- Better understandability
- Complexity of design is reduced,
- Different modules easily understood in isolation:
  - Modules are independent



No dependencies



Highly coupled-many dependencies



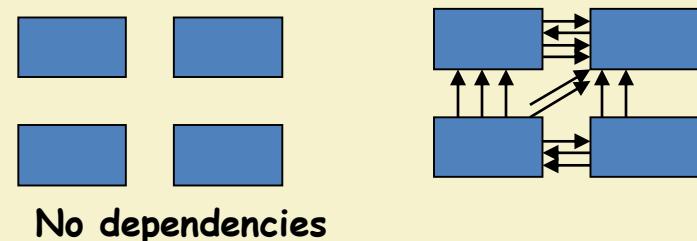
IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Why Functional Independence is Advantageous?

- Functional independence **reduces error propagation.**
  - degree of interaction between modules is low.
  - an error existing in one module does not directly affect other modules.
- **Also: Reuse of modules is possible.**



# Reuse: An Advantage of Functional Independence

- A functionally independent module:
  - can be easily taken out and reused in a different program.
    - each module does some well-defined and precise function
    - the interfaces of a module with other modules is simple and minimal.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Measuring Functional Independence

- Unfortunately, there are no ways:
  - to quantitatively measure the degree of cohesion and coupling:
  - At least classification of different kinds of cohesion and coupling:
    - will give us some idea regarding the degree of cohesiveness of a module.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Classification of Cohesiveness

- Classification can have scope for ambiguity:
  - yet gives us some idea about cohesiveness of a module.
- By examining the type of cohesion exhibited by a module:
  - we can roughly tell whether it displays high cohesion or low cohesion.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Classification of Cohesiveness

functional  
sequential  
communicational  
procedural  
temporal  
logical  
coincidental

Degree of cohesion



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Coincidental cohesion

- The module performs a set of tasks:
  - which relate to each other very loosely, if at all.
    - That is, the module contains a random collection of functions.
    - **functions have been put in the module out of pure coincidence without any thought or design.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Coincidental Cohesion - example

Module AAA{

    Print-inventory();

    Register-Student();

    Issue-Book();

};



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Logical cohesion

- All elements of the module perform similar operations:
  - e.g. error handling, data input, data output, etc.
- An example of logical cohesion:
  - a set of print functions to generate an output report arranged into a single module.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Logical Cohesion

```
module print{
    void print-grades(student-file){ ... }

    void print-certificates(student-file){...}

    void print-salary(teacher-file){...}
}
```



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Temporal cohesion

- The module contains functions so that:
  - **all the functions must be executed in the same time span.**
- Example:
  - The set of functions responsible for
    - initialization,
    - start-up, shut-down of some process, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

```
init() {  
    Check-memory();  
    Check-Hard-disk();  
    Initialize-Ports();  
    Display-Login-Screen();  
}
```

Temporal  
Cohesion -  
Example



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Procedural cohesion

- The set of functions of the module:
  - all part of a procedure (algorithm)
  - certain sequence of steps have to be carried out in a certain order for achieving an objective,
    - e.g. the algorithm for decoding a message.



IIT KHARAGPUR



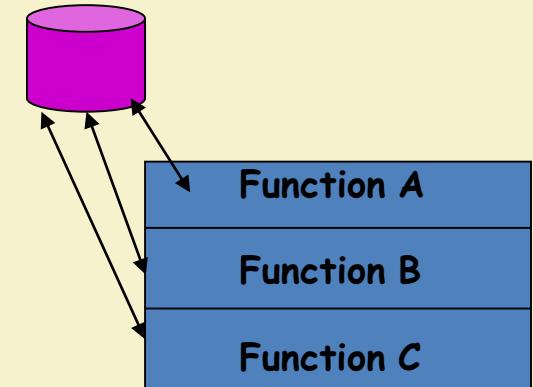
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Communicational cohesion

- All functions of the module:
  - Reference or update the same data structure,
- **Example:**
  - The set of functions defined on an array or a stack.

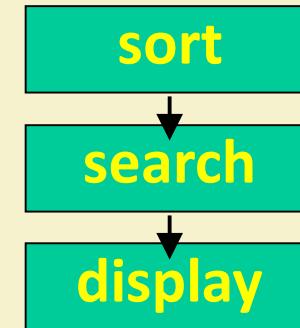
# Communicational Cohesion

```
handle-Student- Data() {  
    Static Struct Student-data[10000];  
    Store-student-data();  
    Search-Student-data();  
    Print-all-students();  
};
```



# Sequential cohesion

- Elements of a module form different parts of a sequence,
  - output from one element of the sequence is input to the next.
  - Example:



# Functional cohesion

- Different elements of a module cooperate:
  - to achieve a single function,
  - e.g. managing an employee's pay-roll.
- When a module displays functional cohesion,
  - **we can describe the function using a single sentence.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

Write down a sentence to describe the function of the module

## Determining Cohesiveness

- If the sentence is compound,
  - it has a sequential or communicational cohesion.
- If it has words like “first”, “next”, “after”, “then”, etc.
  - it has sequential or temporal cohesion.
- If it has words like initialize,
  - it probably has temporal cohesion.

# Coupling

- Coupling indicates:
  - how closely two modules interact or how interdependent they are.
  - **The degree of coupling between two modules depends on their interface complexity.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Coupling

- There are no ways to precisely measure coupling between two modules:
  - classification of different types of coupling will help us to approximately estimate the degree of coupling between two modules.
- Five types of coupling can exist between any two modules.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Classes of coupling

data

stamp

control

common

content

Degree of  
coupling



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Data coupling

- Two modules are data coupled,
  - if they communicate via a parameter:
    - an elementary data item,
    - e.g an integer, a float, a character, etc.
  - The data item should be problem related:
    - not used for control purpose.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Stamp coupling

- Two modules are stamp coupled,
  - if they communicate via a composite data item
    - or an array or structure in C.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Control coupling

- Data from one module is used to direct
  - order of instruction execution in another.
- Example of control coupling:
  - a flag set in one module and tested in another module.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Common Coupling

- Two modules are common coupled,
  - if they share some global data.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Content coupling

- Content coupling exists between two modules:
  - if they share code,
  - e.g, branching from one module into another module.
- The degree of coupling increases
  - from data coupling to content coupling.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Hierarchical Design

- Control hierarchy represents:
  - organization of modules.
  - control hierarchy is also called program structure.
- Most common notation:
  - a tree-like diagram called structure chart.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Good Hierarchical Arrangement of modules

- Essentially means:
  - low fan-out
  - abstraction



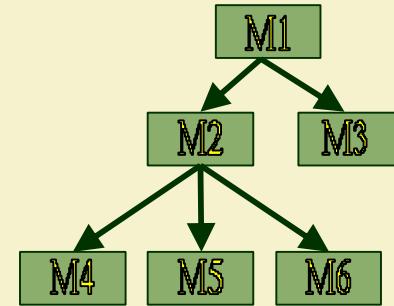
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Characteristics of Module Structure

- **Depth:**
  - number of levels of control
- **Width:**
  - overall span of control.
- **Fan-out:**
  - a measure of the number of modules directly controlled by given module.



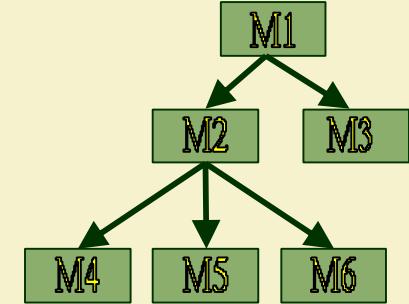
IIT KHARAGPUR



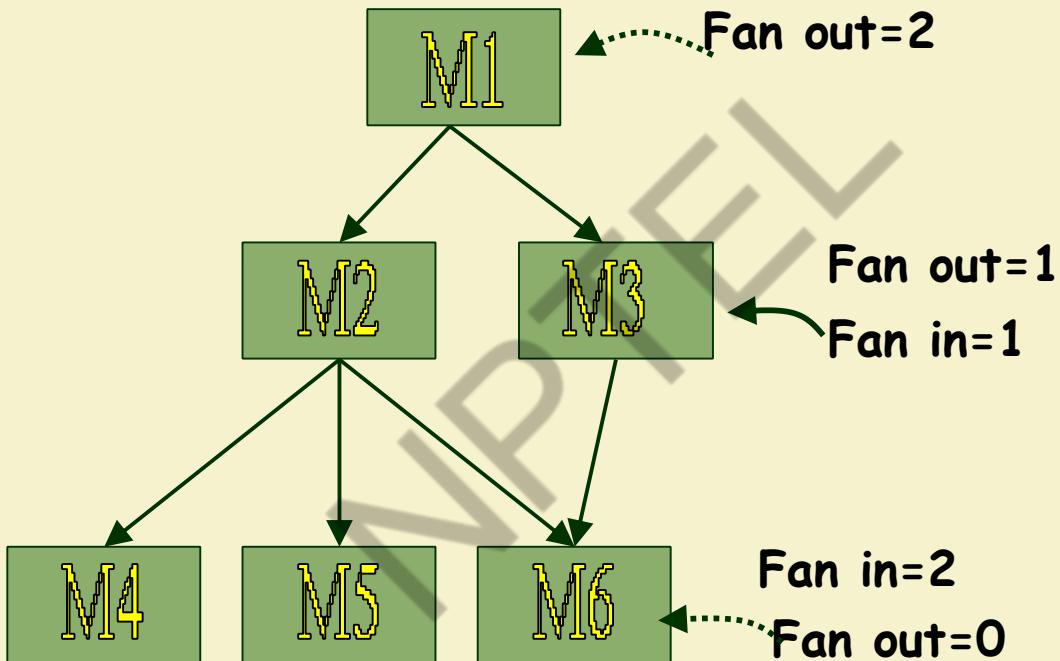
NPTEL  
ONLINE  
CERTIFICATION COURSES

# Characteristics of Module Structure

- Fan-in:
  - indicates how many modules directly invoke a given module.
  - High fan-in represents code reuse and is in general encouraged.



# Module Structure



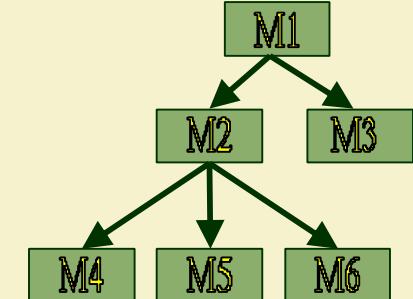
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

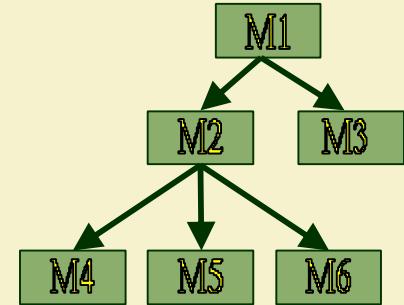
# Goodness of Design

- A design having modules:
  - with high fan-out numbers is not a good design.
  - a module having high fan-out lacks cohesion.



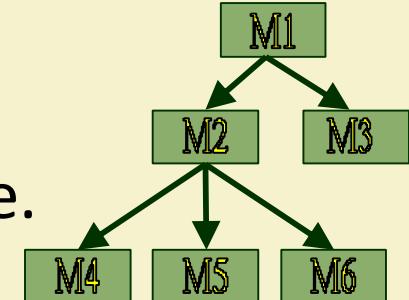
# Large Fan Out

- A module that invokes a large number of other modules:
  - likely to implement several different functions:
  - not likely to perform a single cohesive function.



# Control Relationships

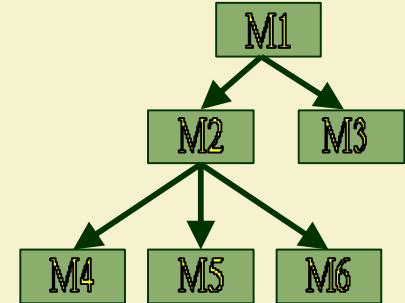
- A module that controls another module:
  - said to be superordinate to the later module.



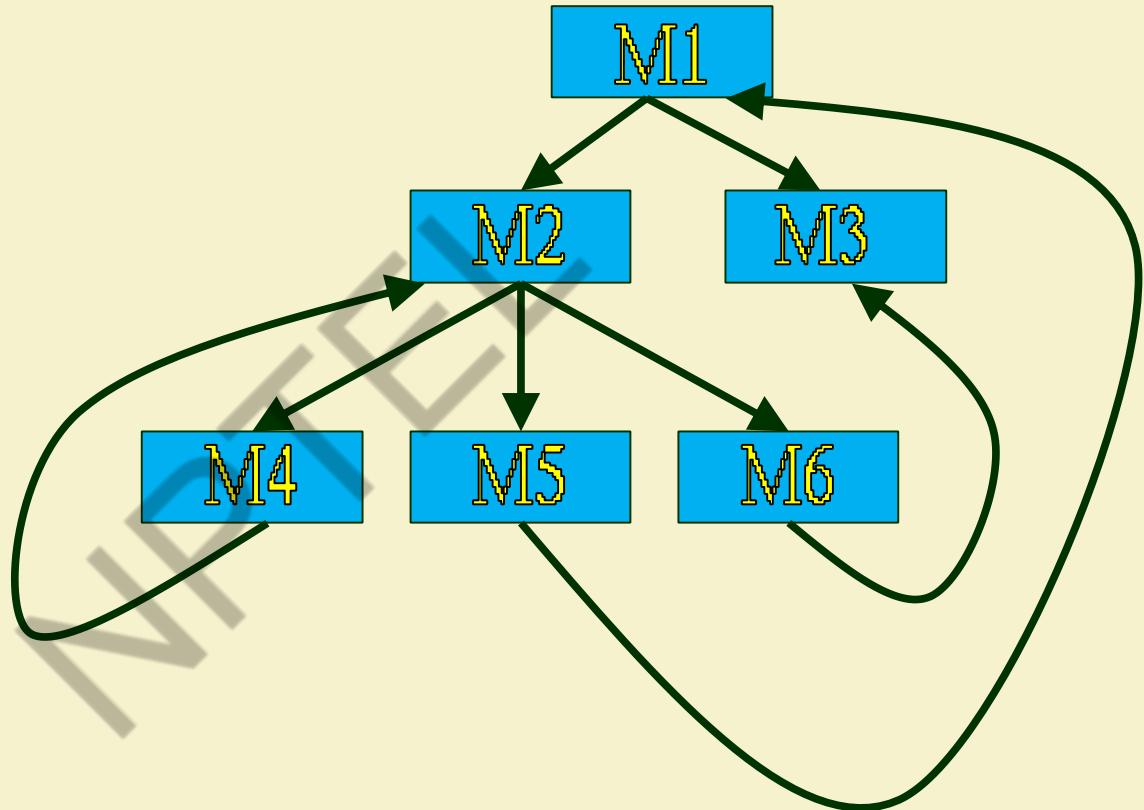
- Conversely, a module controlled by another module:
  - said to be subordinate to the later module.

# Visibility and Layering

- A module A is said to be visible by another module B,
  - if A directly or indirectly calls B.
- The layering principle requires:
  - modules at a layer can call only the modules immediately below it.



# Bad Design



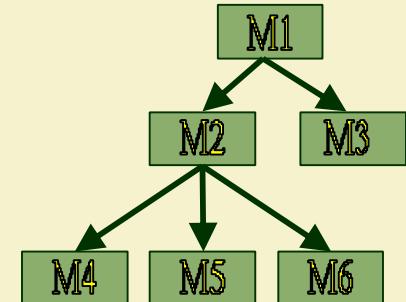
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

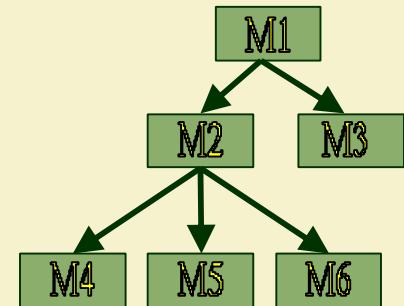
# Abstraction

- **Lower-level modules:**
  - Perform input/output and other low-level functions.
- **Upper-level modules:**
  - Perform more managerial functions.

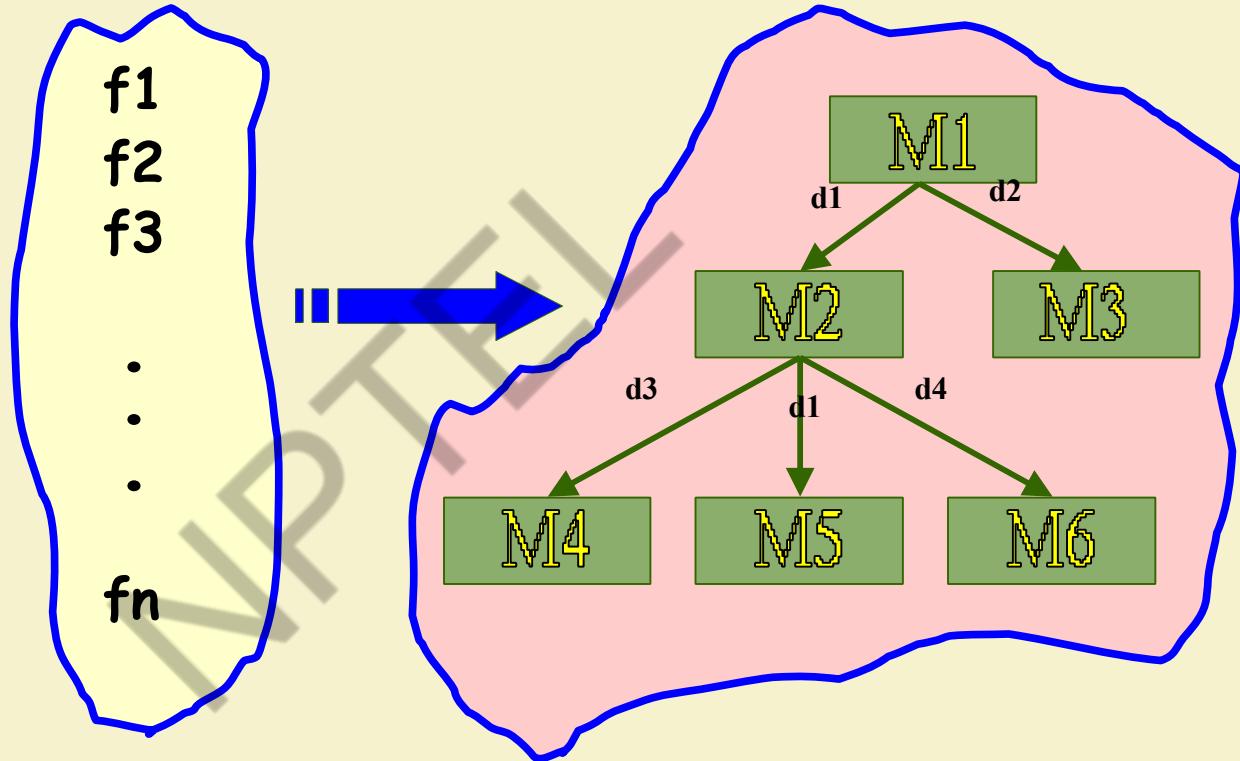


# Abstraction

- The principle of abstraction requires:
  - lower-level modules do not invoke functions of higher level modules.
  - Also known as layered design.



# High-level Design



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Design Approaches

- Two fundamentally different software design approaches:
  - **Function-oriented design**
  - **Object-oriented design**



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Design Approaches

- These two design approaches are radically different.
  - However, are complementary
    - rather than competing techniques.
  - Each technique is applicable at
    - different stages of the design process.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Function-Oriented Design

- A system is looked upon as something
  - that performs a set of functions.
- Starting at this high-level view of the system:
  - each function is successively refined into more detailed functions (top-down decomposition).
  - Functions are mapped to a module structure.

# Example

- The function **create-new-library-member**:
  - creates the record for a new member,
  - assigns a unique membership number
  - prints a bill towards the membership



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Function-Oriented Design

- The system state is centralized:
  - accessible to different functions,
  - member-records:
    - available for reference and updation to several functions:
      - create-new-member
      - delete-member
      - update-member-record



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Function-Oriented Design

- Several function-oriented design approaches have been developed:
  - Structured design (Constantine and Yourdon, 1979)
  - Jackson's structured design (Jackson, 1975)
  - Warnier-Orr methodology
  - Wirth's step-wise refinement
  - Hatley and Pirbhai's Methodology



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example

- Create-library-member function consists of the following sub-functions:
  - assign-membership-number
  - create-member-record
  - print-bill
- Split these into further subfunctions, etc.

# Object-Oriented Design

- System is viewed as a collection of objects (i.e. entities).
- System state is decentralized among the objects:
  - each object manages its own state information.

# Object-Oriented Design Example

- Library Automation Software:
  - each library member is a separate object
    - with its own data and functions.
  - Functions defined for one object:
    - cannot directly refer to or change data of other objects.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented Design

- Objects have their own internal data:
  - defines their state.
- Similar objects constitute a class.
  - each object is a member of some class.
- Classes may inherit features
  - from a super class.
- Conceptually, objects communicate by message passing.

# Object-Oriented versus Function-Oriented Design

- Unlike function-oriented design,
  - in OOD the basic abstraction is not functions such as “sort”, “display”, “track”, etc.,
  - but real-world entities such as “employee”, “picture”, “machine”, “radar system”, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- In OOD:
  - software is not developed by designing functions such as:
    - update-employee-record,
    - get-employee-address, etc.
  - but by designing objects such as:
    - employees,
    - departments, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- Grady Booch sums up this fundamental difference saying:

– “Identify verbs if you are after procedural design and nouns if you are after object-oriented design.”



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- In OOD:
  - state information is not shared in a centralized data.
  - but is distributed among the objects of the system.

# Example

- In an employee pay-roll system, the following can be global data:
  - names of the employees,
  - their code numbers,
  - basic salaries, etc.
- In contrast, in object oriented systems:
  - data is distributed among different employee objects of the system.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- Objects communicate by message passing.
  - one object may discover the state information of another object by interrogating it.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- Of course, somewhere or other the functions must be implemented:
  - the functions are usually associated with specific real-world entities (objects)
  - directly access only part of the system state information.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- Function-oriented techniques group functions together if:
  - as a group, they constitute a higher level function.
- On the other hand, object-oriented techniques group functions together:
  - on the basis of the data they operate on.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- To illustrate the differences between object-oriented and function-oriented design approaches,
  - let us consider an example ---
  - **An automated fire-alarm system for a large building.**

# Fire-Alarm System

- We need to develop a computerized fire alarm system for a large multi-storied building:
  - There are 80 floors and 2000 rooms in the building.



# Fire-Alarm System

- Different rooms of the building:
  - fitted with smoke detectors and fire alarms.
- The fire alarm system would monitor:
  - status of the smoke detectors.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Fire-Alarm System

- Whenever a fire condition is reported by any smoke detector:
  - the fire alarm system should:
    - determine the location from which the fire condition was reported
    - sound the alarms in the neighbouring locations.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Fire-Alarm System

- The fire alarm system should:
  - flash an alarm message on the computer console:
    - fire fighting personnel man the console round the clock.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Fire-Alarm System

- After a fire condition has been successfully handled,
  - the fire alarm system should let fire fighting personnel reset the alarms.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

```
/* Global data (system state) accessible by various functions */
BOOL          detector_status[2000];
int           detector_locs[2000];
BOOL          alarm_status[2000]; /* alarm activated when set */
int           alarm_locs[2000]; /* room number where alarm is located */
int           neighbor_alarms[2000][10];/*each detector has at most*/
                                         /* 10 neighboring alarm locations */

interrogate_detectors();
get_detector_location();
determine_neighbor();
ring_alarm();
reset_alarm();
report_fire_location();
```

Function-Oriented  
Approach



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented Approach:

## class detector

attributes: status, location, neighbors

operations: create, sense-status, get-location, find-neighbors

## class alarm

attributes: location, status

operations: create, ring-alarm, get\_location, reset-alarm

- Appropriate number of instances of the class detector and alarm are created.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- In a function-oriented program :
  - the system state is centralized
  - several functions accessing these data are defined.
- In the object oriented program,
  - the state information is distributed among various sensor and alarm objects.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- Use OOD to design the classes:
  - then applies top-down function oriented techniques
    - to design the internal methods of classes.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented versus Function-Oriented Design

- Though outwardly a system may appear to have been developed in an object oriented fashion,
  - but inside each class there is a small hierarchy of functions designed in a top-down manner.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Summary

- We started with an overview of:
  - activities undertaken during the software design phase.
- We identified:
  - the information need to be produced at the end of design:
    - so that the design can be easily implemented using a programming language.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Summary

- We characterized the features of a good software design by introducing the concepts of:
  - fan-in, fan-out,
  - cohesion, coupling,
  - abstraction, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Summary

- We classified different types of cohesion and coupling:
  - enables us to approximately determine the cohesion and coupling existing in a design.

# Summary

- Two fundamentally different approaches to software design:
  - function-oriented approach
  - object-oriented approach

# Summary

- We examined the essential philosophy behind these two approaches
  - these two approaches are not really competing but complementary approaches.

# Thank You!!

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

Faculty Name  
Department Name



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Structured Analysis and Design

Rajib Mall

CSE Department

IIT KHARAGPUR

# Introduction

- Structured analysis is a top-down decomposition technique:
  - DFD (Data Flow Diagram) is the modelling technique used
  - Functional requirements are modelled and decomposed.
- Why model functionalities?
  - **Functional requirements exploration and validation**
  - **Serves as the starting point for design.**



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Function-oriented vs. Object-oriented Design

- Two distinct style of design:

- **Function-oriented or Procedural**

- Top-down approach
    - Carried out using Structured analysis and structured design
    - Coded using languages such as C

- **Object-oriented**

- Bottom-up approach
    - Carried out using UML
    - Coded using languages such as Java, C++, C#



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Structured analysis and Structured Design

- During Structured analysis:
  - High-level functions are successively decomposed:
    - Into more detailed functions.
- During Structured design:
  - The detailed functions are mapped to a module structure.



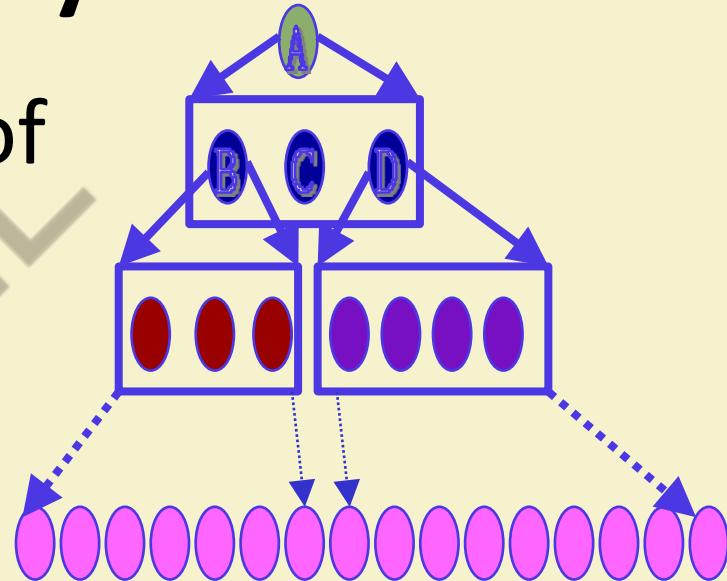
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Structured Analysis

- Successive decomposition of high-level functions:
  - Into more detailed functions.
  - Technically known as **top-down decomposition**.



# SA/SD (Structured Analysis/Structured Design)

- SA/SD technique draws heavily from the following methodologies:
  - Constantine and Yourdon's methodology
  - Hatley and Pirbhai's methodology
  - Gane and Sarson's methodology
  - DeMarco and Yourdon's methodology
- SA/SD technique results in:
  - high-level design.

We largely use



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Functional Decomposition

- Each function is analyzed:
  - Hierarchically decomposed into more detailed functions.
  - Simultaneous decomposition of high-level data
    - Into more detailed data.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Structured Analysis

- Textual problem description converted into a graphic model.
  - Done using **data flow diagrams (DFDs)**.
  - DFD graphically represents the results of structured analysis.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Structured Analysis

- The results of structured analysis can be easily understood even by ordinary customers:
  - Does not require computer knowledge.
  - Directly represents customer's perception of the problem.
  - Uses customer's terminology for naming different functions and data.
- Results of structured analysis:
  - Can be reviewed by customers to check whether it captures all their requirements.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Structured Design

- The functions represented in the DFD:
  - Mapped to a **module structure**.
- Module structure:
  - Also called **software architecture**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Structured Analysis vs. Structured Design

- Purpose of structured analysis:
  - Capture the detailed structure of the system as the user views it.
- Purpose of structured design:
  - Arrive at a form that is suitable for implementation in some programming language.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Structured Analysis: Recap

- Based on principles of:
  - **Top-down decomposition approach.**
  - **Divide and conquer principle:**
    - Each function is considered individually (i.e. isolated from other functions).
    - Decompose functions totally disregarding what happens in other functions.
  - Graphical representation of results using
    - **Data flow diagrams (or bubble charts).**



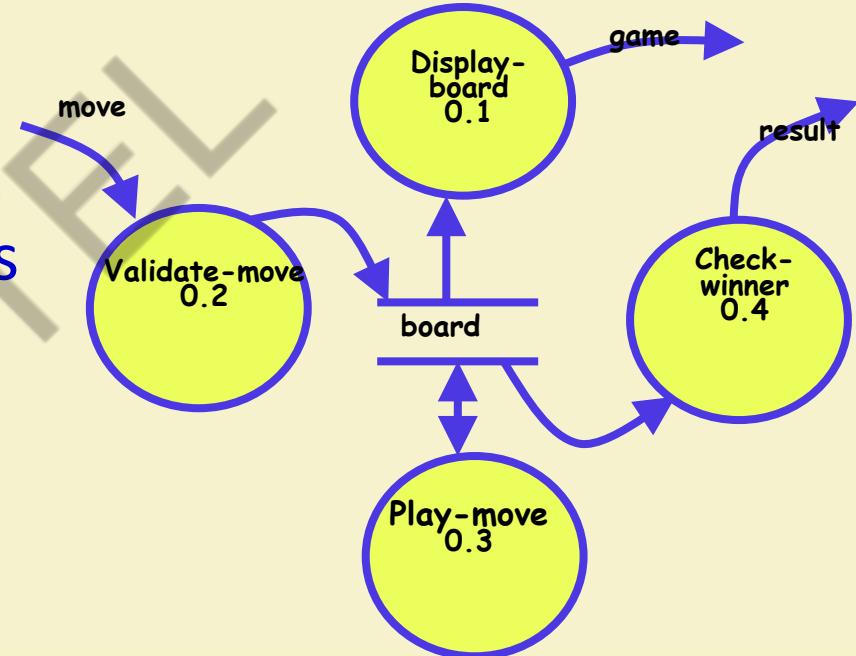
IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

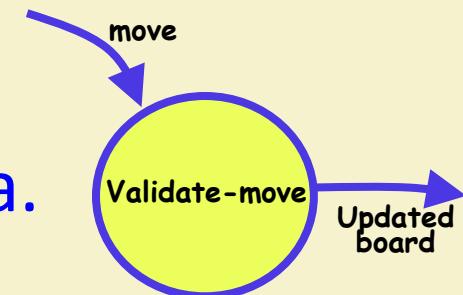
# Data Flow Diagram

- DFD is a hierarchical graphical model:
  - Shows the different functions (or processes) of the system
  - Data interchange among the processes.

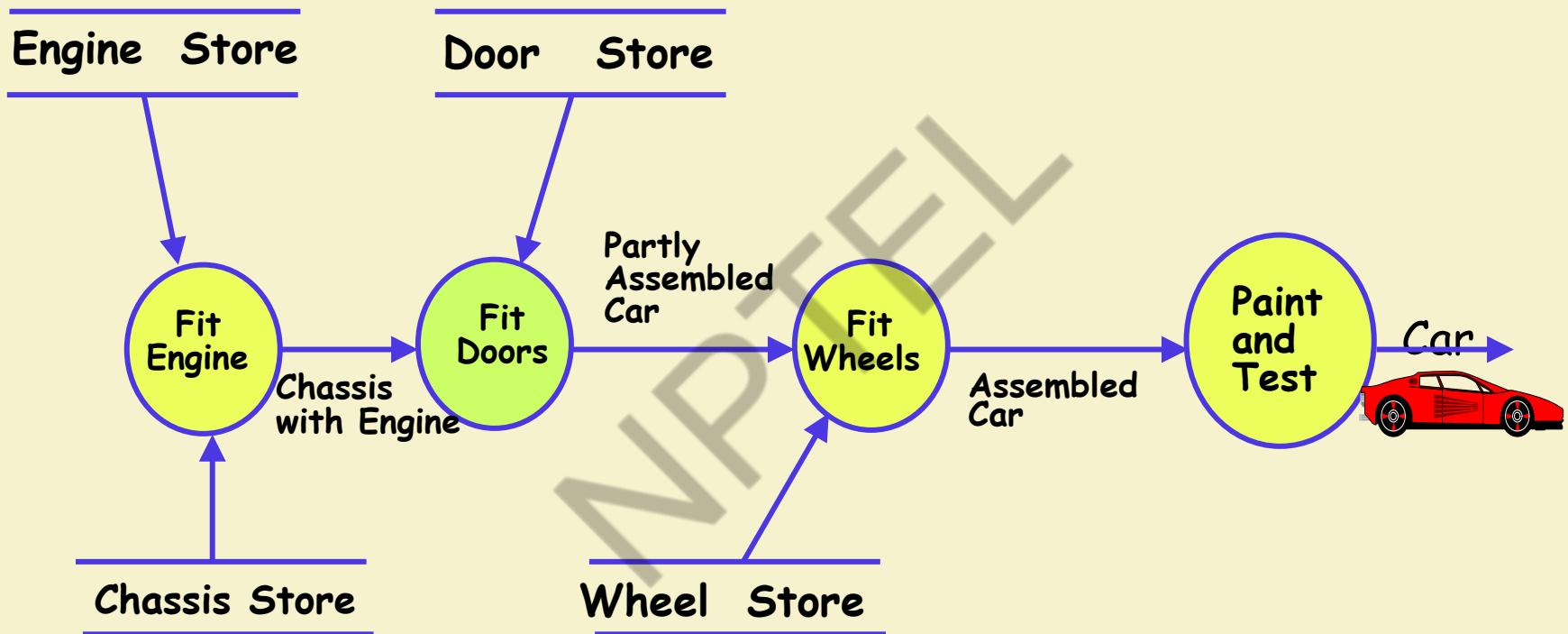


# DFD Concepts

- It is useful to consider each function as a processing station:
  - Each function consumes some input data.
  - Produces some output data.



# Data Flow Model of a Car Assembly Unit



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Pros of Data Flow Diagrams (DFDs)

- A DFD model:
  - Uses limited types of symbols.
  - Simple set of rules
  - Easy to understand --- a hierarchical model.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Hierarchical Model

- As pointed out earlier:
  - Human cognitive restrictions are overcome through use of a hierarchical model:
  - In a hierarchical model:
    - We start with a very simple and abstract model of a system,
    - Details are slowly introduced through the hierarchies.

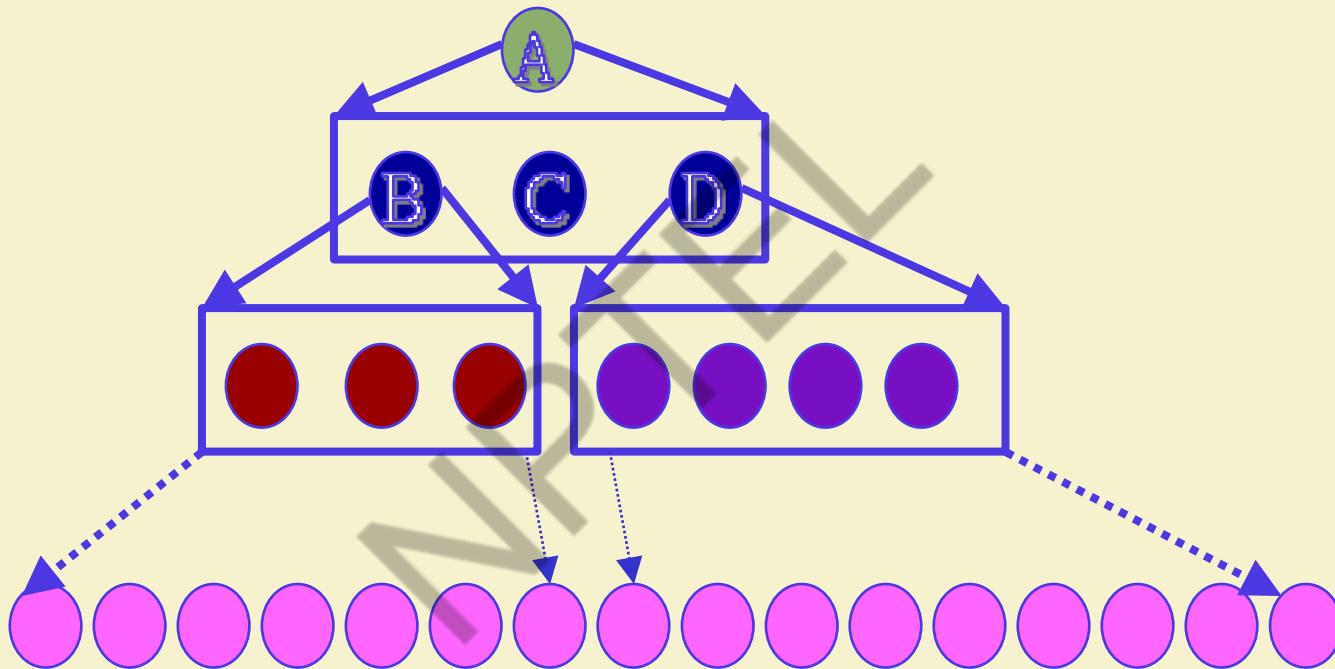


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# A Hierarchical Model



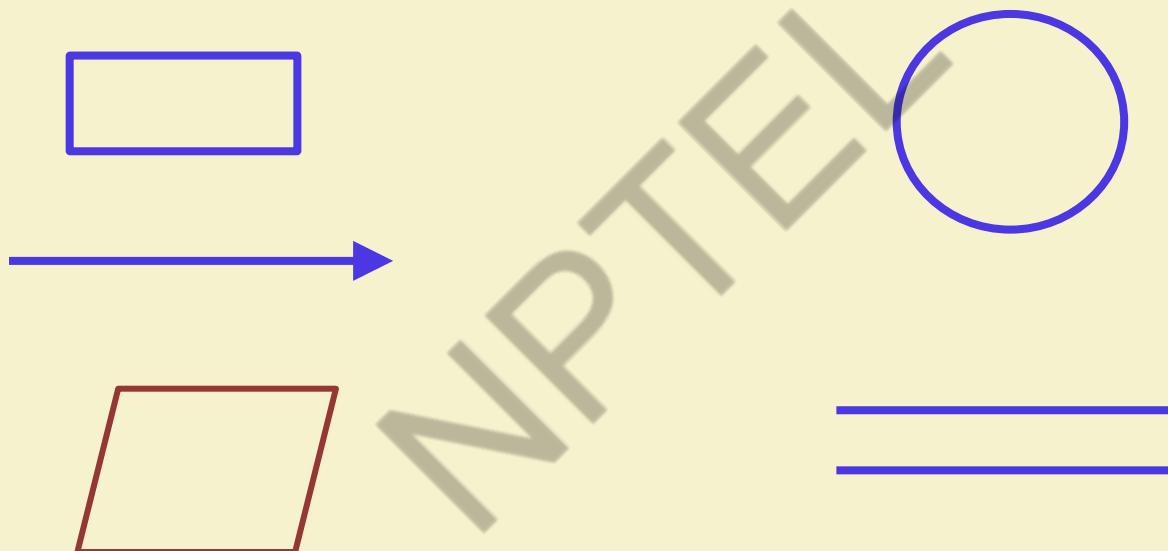
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Data Flow Diagrams (DFDs)

- Basic Symbols Used for Constructing DFDs:



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# External Entity Symbol

- Represented by a rectangle
- External entities are either users or external systems:
  - input data to the system or
  - consume data produced by the system.
  - Sometimes external entities are called **terminator, source, or sink.**

Librarian



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Function Symbol

- A function such as “search-book” is represented using a circle:
  - This symbol is called a **process** or **bubble** or **transform**.
  - Bubbles are annotated with corresponding function names.
  - A function represents some activity:
    - **Function names should be verbs.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data Flow Symbol

- A directed arc or line.
  - Represents data flow in the direction of the arrow.
  - Data flow symbols are annotated with names of data they carry.

book-name 



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data Store Symbol

- Represents a logical file:
  - A logical file can be:
    - **a data structure**
    - **a physical file on disk.**
  - Each data store is connected to a process:
    - By means of a data flow symbol.

book-details



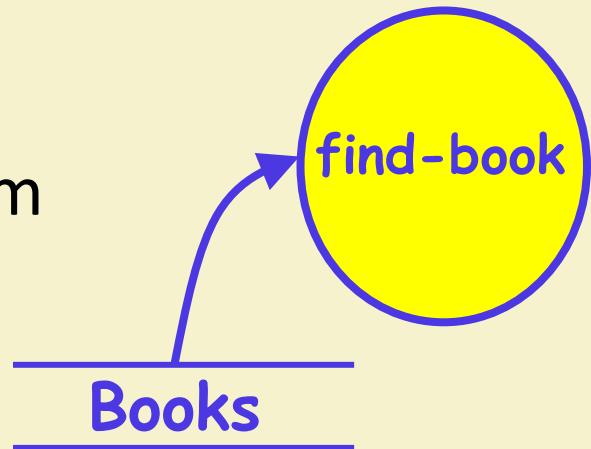
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

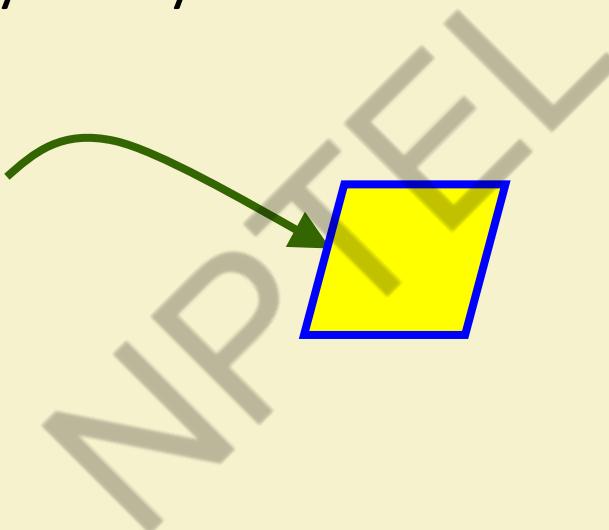
# Data Store Symbol

- Direction of data flow arrow:
  - Shows whether data is being read from or written into it.
- An arrow into or out of a **data store**:
  - Implicitly represents the entire data of the data store
  - Arrows connecting to a data store need not be annotated with any data name.



# Output Symbol: Parallelogram

- Output produced by the system



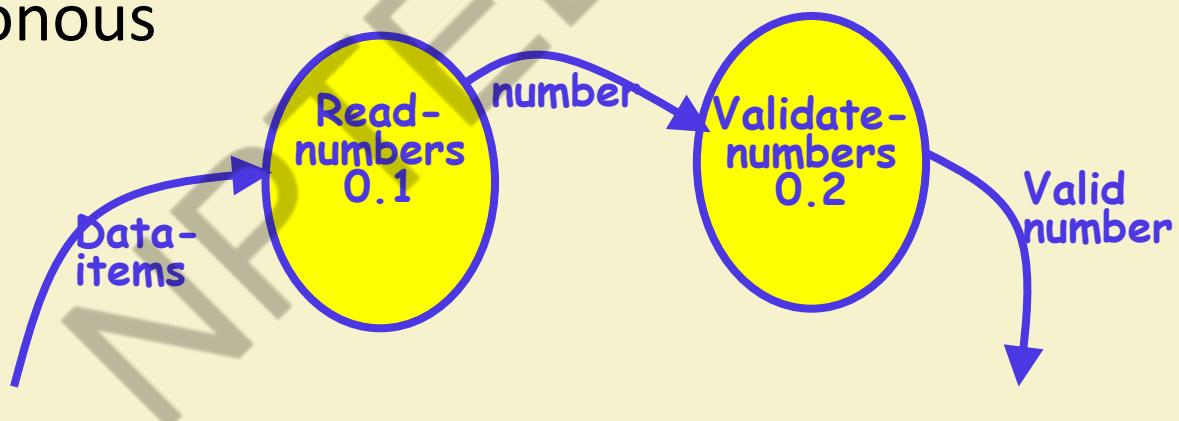
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

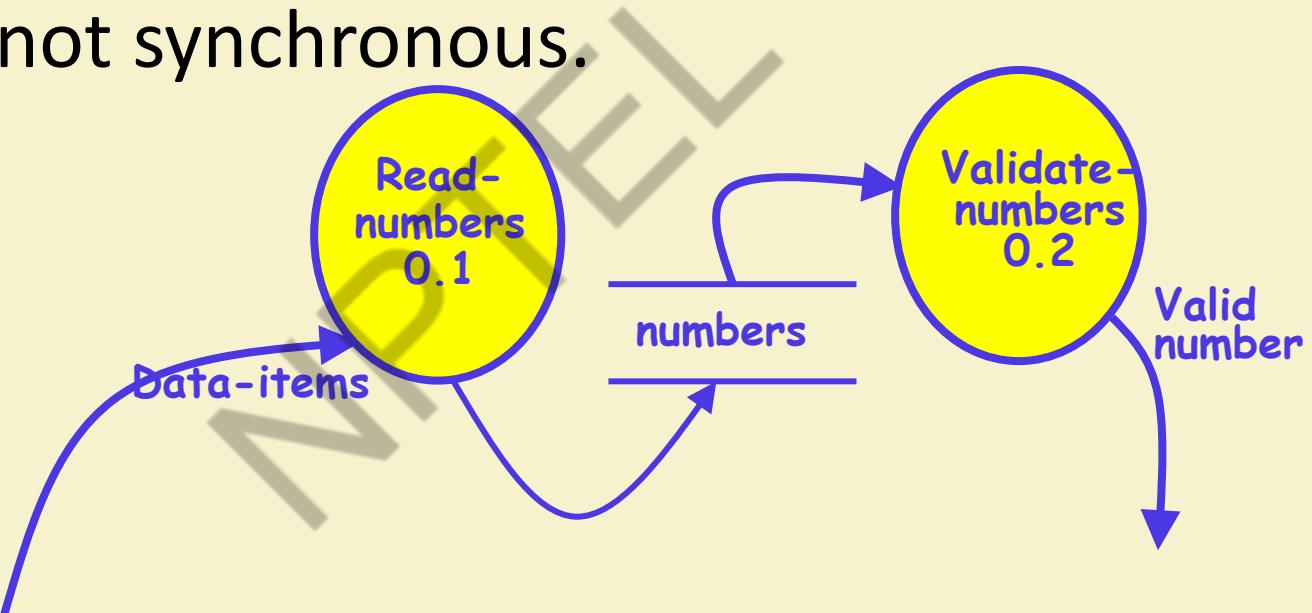
# Synchronous Operation

- If two bubbles are directly connected by a data flow arrow:
  - They are synchronous



# Asynchronous Operation

- If two bubbles are connected via a data store:
  - They are not synchronous.



# Yourdon's vs. Gane Sarson Notations

- The notations that we are following:
  - Are closer to the Yourdon's notations
- You may sometimes find notations in books and used in some tools that are slightly different:
  - For example, the data store may look like a box with one end closed



IIT KHARAGPUR



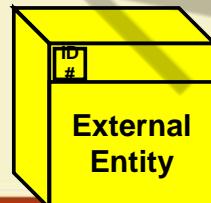
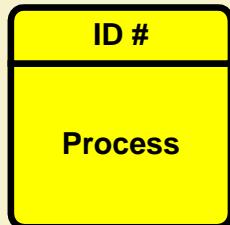
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Visio 5.x

From Flow Chart /  
Data Flow Diagram

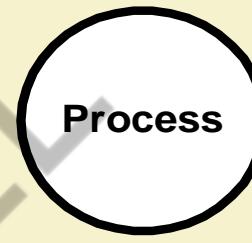


From Software Diagram /  
Gane-Sarson DFD



## Visio 2000

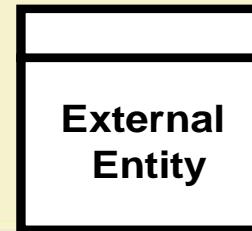
Data Flow Diagram



---

Data Store

---



DFD  
Shapes  
from Visio



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# How is Structured Analysis Performed?

- Initially represent the software at the most abstract level:
  - Called the **context diagram**.
  - The entire system is represented as a single bubble,
  - This bubble is labelled according to the main function of the system.

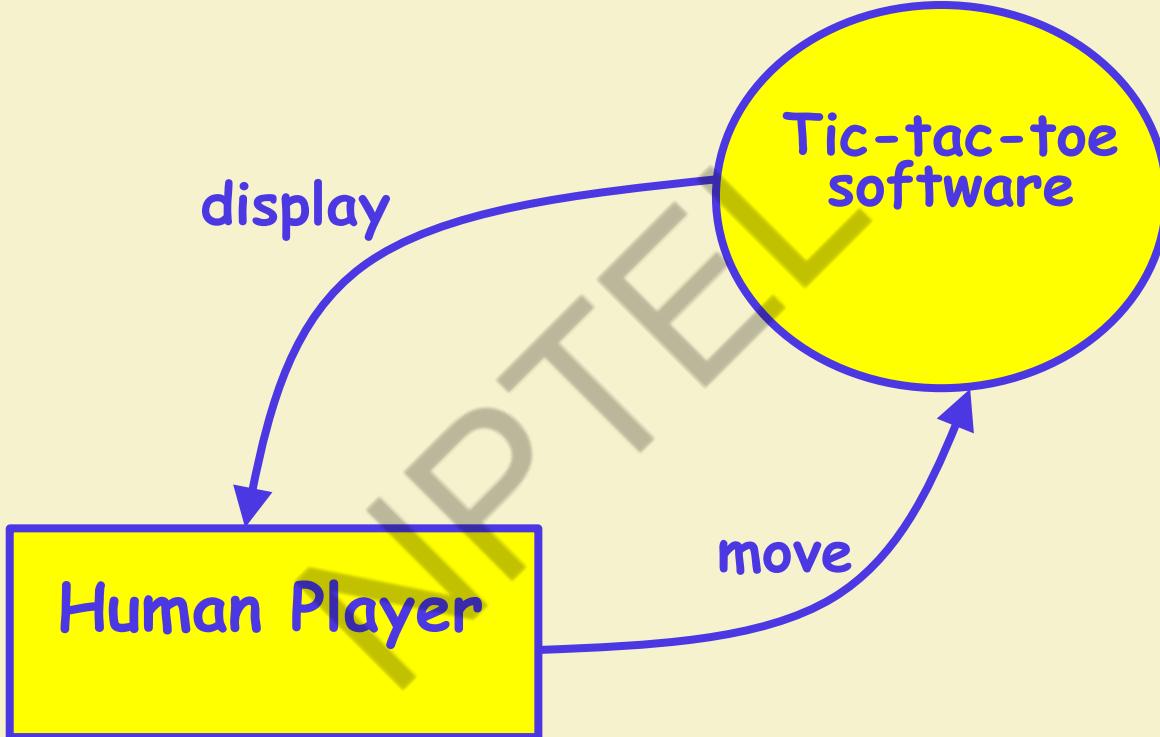


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Tic-tac-toe: Context Diagram



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Context Diagram

- A context diagram shows:
  - External entities.
  - Data input to the system by the external entities,
  - Output data generated by the system.
- The context diagram is also called the **level 0 DFD**.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Context Diagram

- Establishes the context of the system, i.e.
  - Represents the system level
    - **Data sources**
    - **Data sinks.**



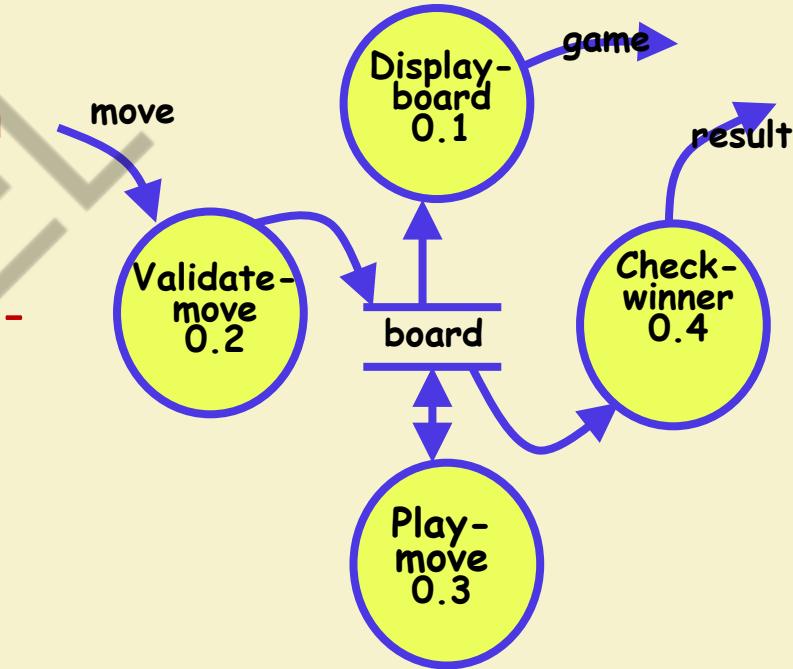
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

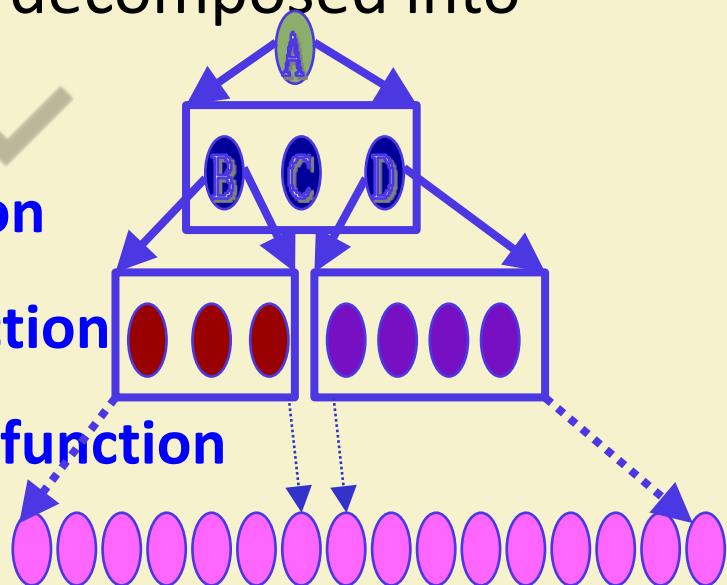
# Level 1 DFD Construction

- Examine the SRS document:
  - Represent each high-level function as a bubble.
  - Represent data input to every high-level function.
  - Represent data output from every high-level function.



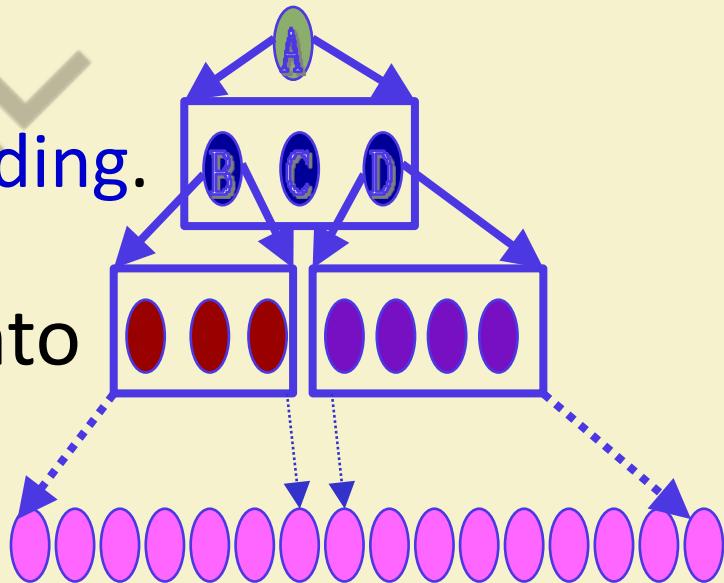
# Higher Level DFDs

- Each high-level function is separately decomposed into subfunctions:
  - **Identify the subfunctions of the function**
  - **Identify the data input to each subfunction**
  - **Identify the data output from each subfunction**
- These are represented as DFDs.



# Decomposition

- Decomposition of a bubble:
  - Also called factoring or exploding.
- Each bubble is decomposed into
  - Between 3 to 7 bubbles.



# Decomposition

- Too few bubbles make decomposition superfluous:
  - If a bubble is decomposed to just one or two bubbles:
    - Then this decomposition is redundant.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Decomposition Pitfall

- Too many bubbles at a level, a sign of poor modelling:
  - More than 7 bubbles at any level of a DFD.
  - Make the DFD model hard to understand.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Decompose How Long?

- Decomposition of a bubble should be carried on until:
  - A level at which the function of the bubble can be described using a simple algorithm.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example 1: RMS Calculating Software

- Consider a software called RMS calculating software:
  - Reads three integers in the range of -1000 and +1000
  - Finds out the root mean square (rms) of the three input numbers
  - Displays the result.



IIT KHARAGPUR

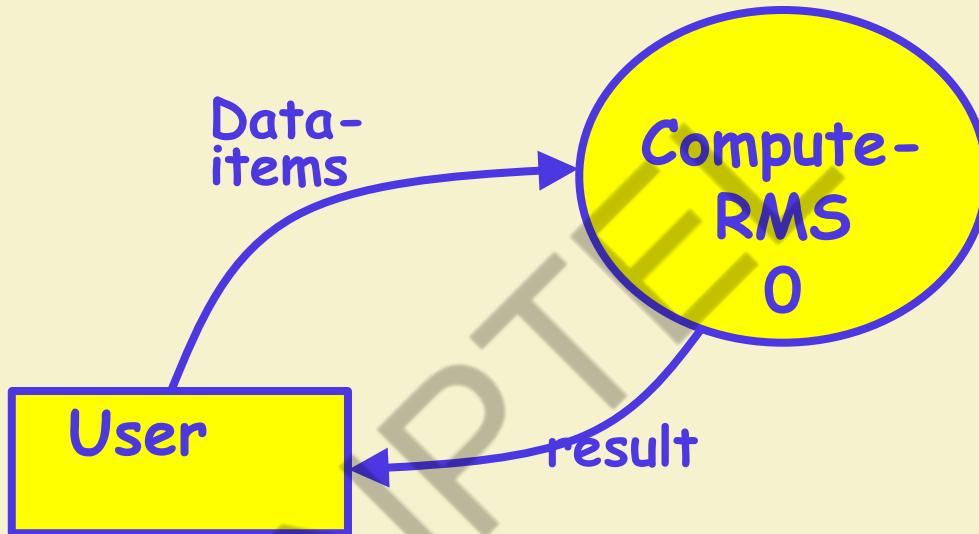


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example 1: RMS Calculating Software

- The context diagram is simple to develop:
  - The system accepts 3 integers from the user
  - Returns the result to him.

# Example 1: RMS Calculating Software



Context Diagram (Level 0 DFD)



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 1: RMS Calculating Software

- From a cursory analysis of the problem description:
  - We can see that the system needs to perform several things.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example 1: RMS Calculating Software

- Accept input numbers from the user:
- Validate the numbers,
- Calculate the root mean square of the input numbers
- Display the result.

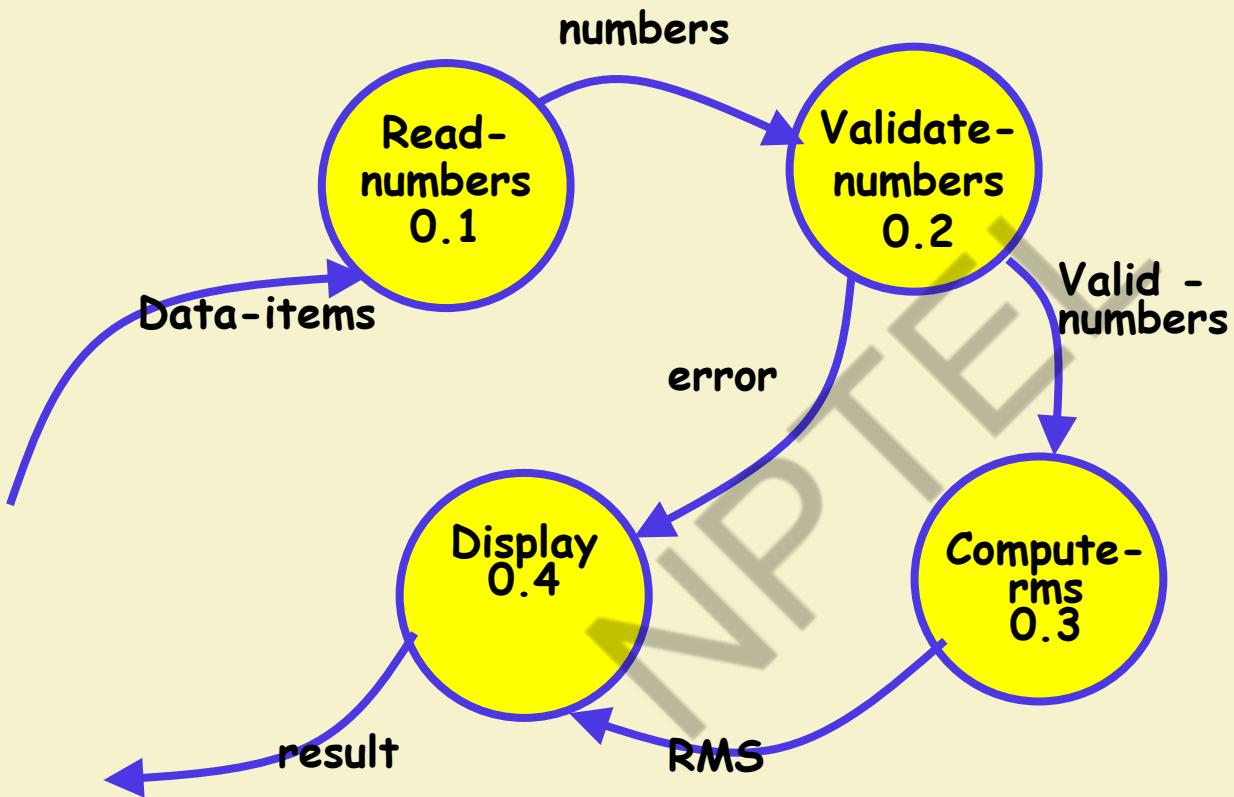


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example 1: Level 1 DFD RMS Calculating Software



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example: RMS Calculating Software

- Decomposition is never carried on up to basic instruction level:
  - A bubble is not decomposed any further:
    - If it can be represented by a simple set of instructions.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- A DFD is always accompanied by a data dictionary.
- A data dictionary lists all data items appearing in a DFD:
  - Definition of all composite data items in terms of their component data items.
  - All data names along with the purpose of the data items.
- For example, a data dictionary entry may be:
  - **grossPay = regularPay+overtimePay**

## Data Dictionary

# Importance of Data Dictionary

- Provides the team of developers with standard terminology for all data:
  - A consistent vocabulary for data is very important
- In the absence of a data dictionary, different developers tend to use different terms to refer to the same data,
  - Causes unnecessary confusion.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Importance of Data Dictionary

- Data dictionary provides the definition of different data:
  - In terms of their component elements.
- For large systems,
  - The data dictionary grows rapidly in size and complexity.
  - Typical projects can have thousands of data dictionary entries.
  - It is extremely difficult to maintain such a dictionary manually.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Data Dictionary

- CASE (Computer Aided Software Engineering) tools come handy:
  - CASE tools capture the data items appearing in a DFD automatically to generate the data dictionary.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data Dictionary

- CASE tools support queries:
  - About definition and usage of data items.
- For example, queries may be made to find:
  - Which data item affects which processes,
  - A process affects which data items,
  - The definition and usage of specific data items, etc.
- Query handling is facilitated:
  - If data dictionary is stored in a relational database management system (RDBMS).



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Data Definition

- Composite data are defined in terms of primitive data items using simple operators:
- **+**: denotes composition of data items, e.g
  - **a+b represents data a together with b.**
- **[,,,]:** represents selection,
  - Any one of the data items listed inside the square bracket can occur.
  - For example, **[a,b] represents either a occurs or b**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Data Definition

- **( )**: contents inside the bracket represent optional data
  - which may or may not appear.
  - **a+(b)** represents either a or a+b
- **{ }**: represents iterative data definition,
  - **{name}5** represents five name data.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data Definition

- $\{name\}^*$  represents
  - zero or more instances of name data.
- $=$  represents equivalence,
  - e.g.  $a=b+c$  means that a represents b and c.
- $* \ *:$  Anything appearing within \* \* is considered as comment.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

- numbers=valid-numbers=a+b+c
- a:integer \* input number \*
- b:integer \* input number \*
- c:integer \* input number \*
- asq:integer
- bsq:integer
- csq:integer
- squared-sum: integer
- Result=[RMS,error]
- RMS: integer \* root mean square value\*
- error:string \* error message\*

## Data Dictionary for RMS Software

# Balancing a DFD

- **Data flowing into or out of a bubble:**
  - Must match the data flows at the next level of DFD.
- In the level 1 of the DFD,
  - Data item c flows into the bubble P3 and the data item d and e flow out.
- In the next level, bubble P3 is decomposed.
  - The decomposition is balanced as data item c flows into the level 2 diagram and d and e flow out.

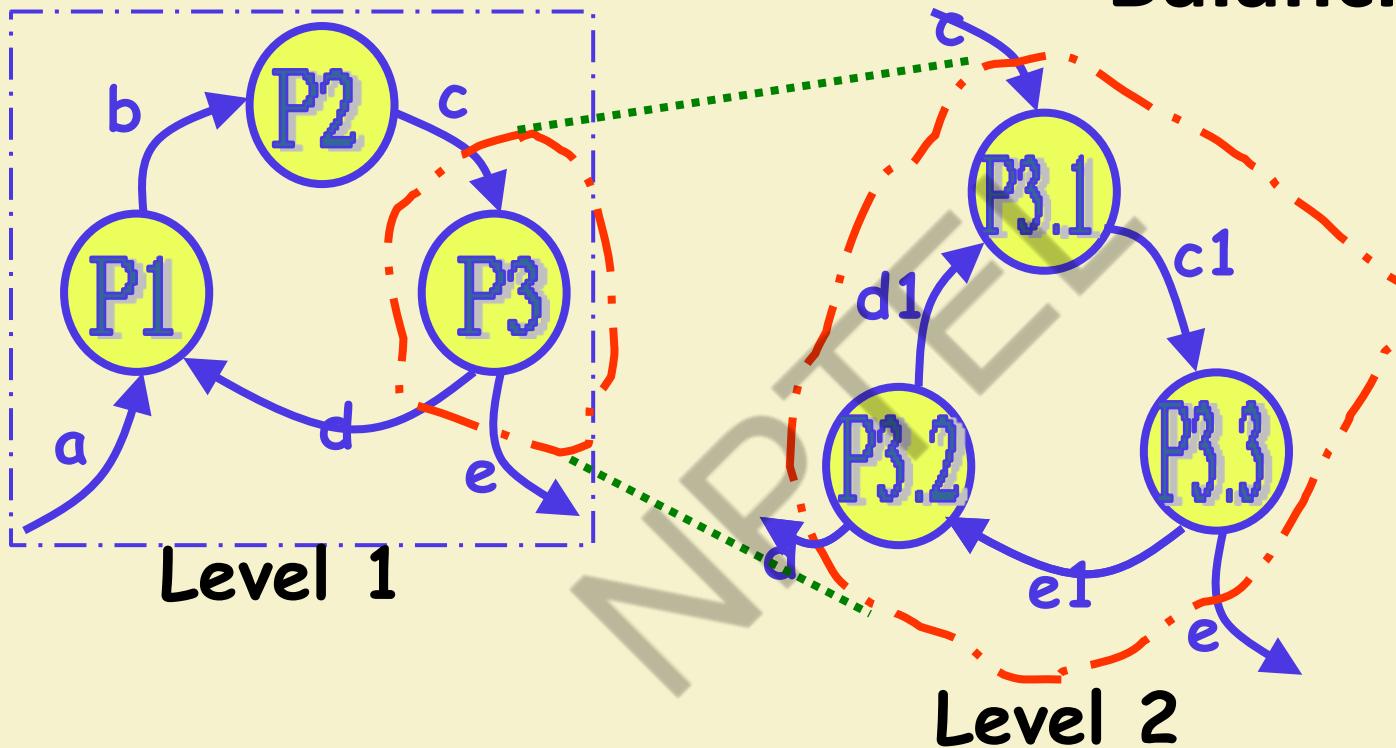


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Balancing a DFD



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Numbering of Bubbles

- Number the bubbles in a DFD:
  - **Numbers help in uniquely identifying any bubble from its bubble number.**
- The bubble at context level:
  - Assigned number 0.
- Bubbles at level 1:
  - Numbered 0.1, 0.2, 0.3, etc
- When a bubble numbered x is decomposed,
  - Its children bubble are numbered x.1, x.2, x.3, etc.



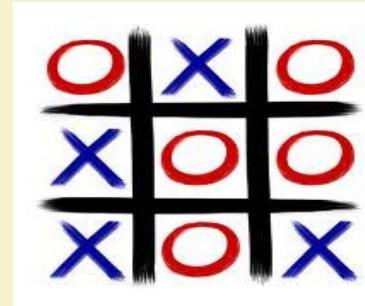
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 2: Tic-Tac-Toe Computer Game

- A human player and the computer make alternate moves on a 3 X 3 square.
- A move consists of marking a previously unmarked square.
- The user inputs a number between 1 and 9 to mark a square
- Whoever is first to place three consecutive marks along a straight line (i.e., along a row, column, or diagonal) on the square wins.



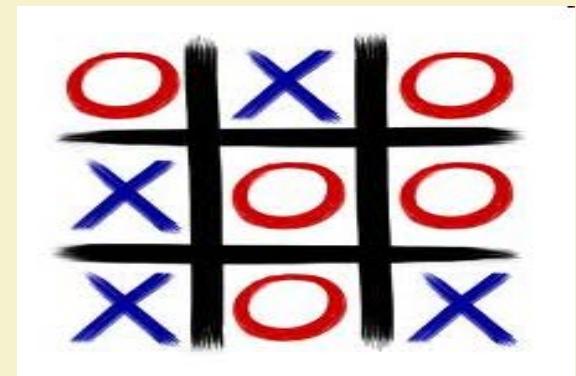
IIT KHARAGPUR



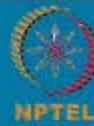
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example: Tic-Tac-Toe Computer Game

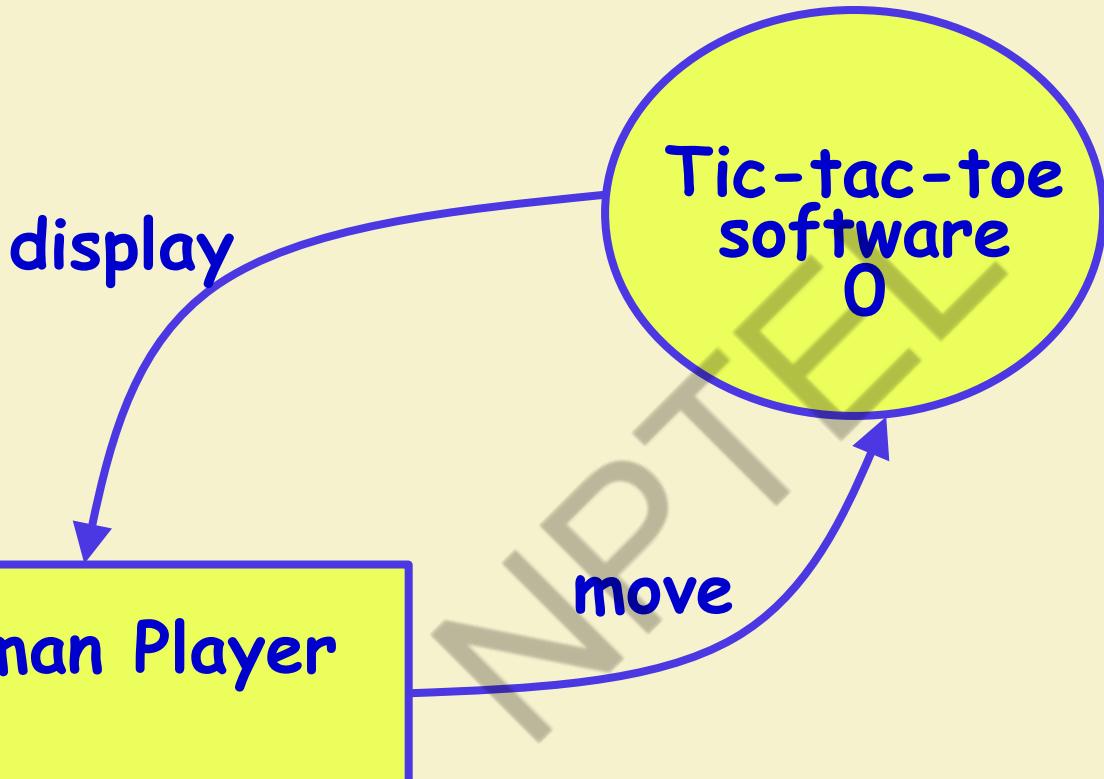
- As soon as either of the human player or the computer wins,
  - A message announcing the winner should be displayed.
- If neither player manages to get three consecutive marks along a straight line,
  - And all the squares on the board are filled up,
  - Then the game is drawn.
- The computer always tries to win a game.



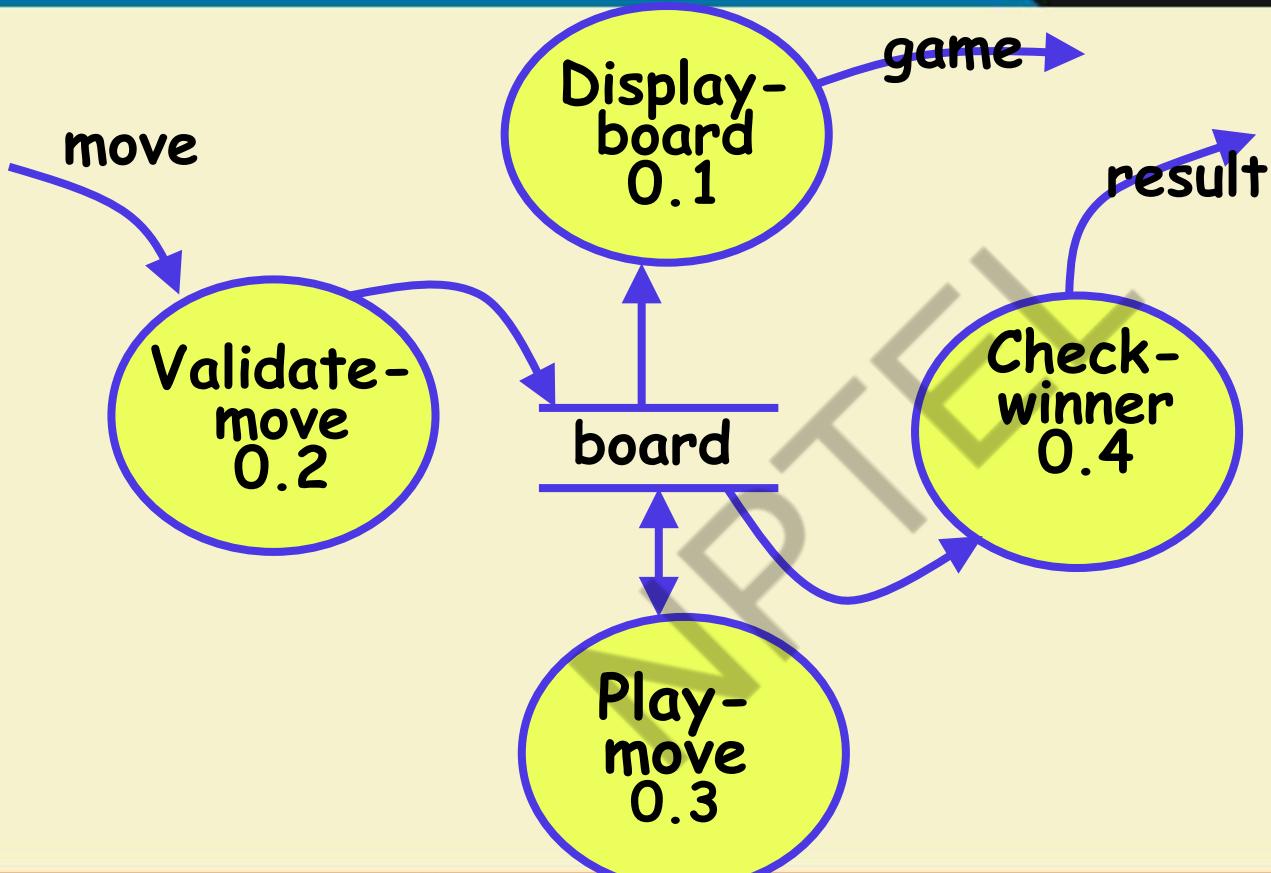
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



Context  
Diagram for  
Example



Level 1 DFD



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Data Dictionary

Display=game + result

move = integer

board = {integer}9

game = {integer}9

result=string



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example 3: Trading-House Automation System (TAS)

- A large trading house wants us to develop a software:
  - To automate book keeping activities associated with its business.
- It has many regular customers:
  - They place orders for various kinds of commodities.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## **Example 3: Trading-House Automation System (TAS)**

- The trading house maintains names and addresses of its regular customers.
- Each customer is assigned a unique customer identification number (CIN).
- As per current practice when a customer places order:
  - The accounts department first checks the credit-worthiness of the customer.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example: Trading-House Automation System (TAS)

- The credit worthiness of a customer is determined:
  - By analyzing the history of his payments to the bills sent to him in the past.
- If a customer is not credit-worthy:
  - His orders are not processed any further
  - An appropriate order rejection message is generated for the customer.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example: Trading-House Automation System (TAS)

- If a customer is credit-worthy:
  - Items he/she has ordered are checked against the list of items the trading house deals with.
- **The items that the trading house does not deal with:**
  - Are not processed any further
  - An appropriate message for the customer for these items is generated.

# Example: Trading-House Automation System (TAS)

- The items in a customer's order that the trading house deals with:
  - Are checked for availability in inventory.
- If the items are available in the inventory in desired quantities:
  - A bill with the forwarding address of the customer is printed.
  - A material issue slip is printed.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example: Trading-House Automation System (TAS)

- The customer can produce the material issue slip at the store house:
  - Take delivery of the items.
  - Inventory data adjusted to reflect the sale to the customer.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example: Trading-House Automation System (TAS)

- If an ordered item is not available in the inventory in sufficient quantity:
  - To be able to fulfil pending orders store details in a "pending-order" file :
    - out-of-stock items along with quantity ordered.
    - customer identification number



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example: Trading-House Automation System (TAS)

- The purchase department:
  - would periodically issue commands to generate indents.
- When **generate indents** command is issued:
  - The system should examine the "pending-order" file
  - Determine the orders that are pending
  - Total quantity required for each of the items.

## Example: Trading-House Automation System (TAS)

- TAS should find out the addresses of the vendors who supply the required items:
  - Examine the file containing vendor details (their address, items they supply etc.)
  - Print out indents to those vendors.



IIT KHARAGPUR



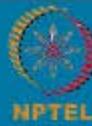
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example: Trading-House Automation System (TAS)

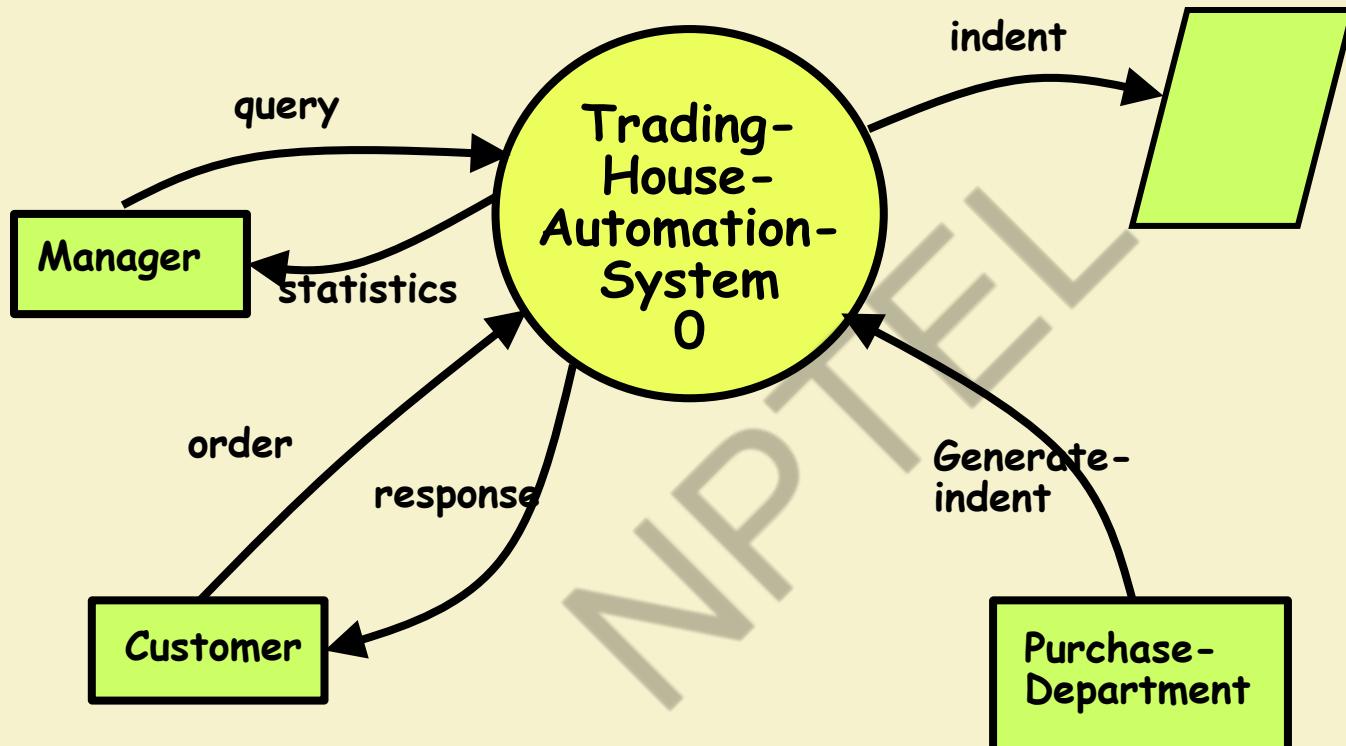
- TAS should also answers managerial queries:
  - Statistics of different items sold over any given period of time
  - Corresponding quantity sold and the price realized.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



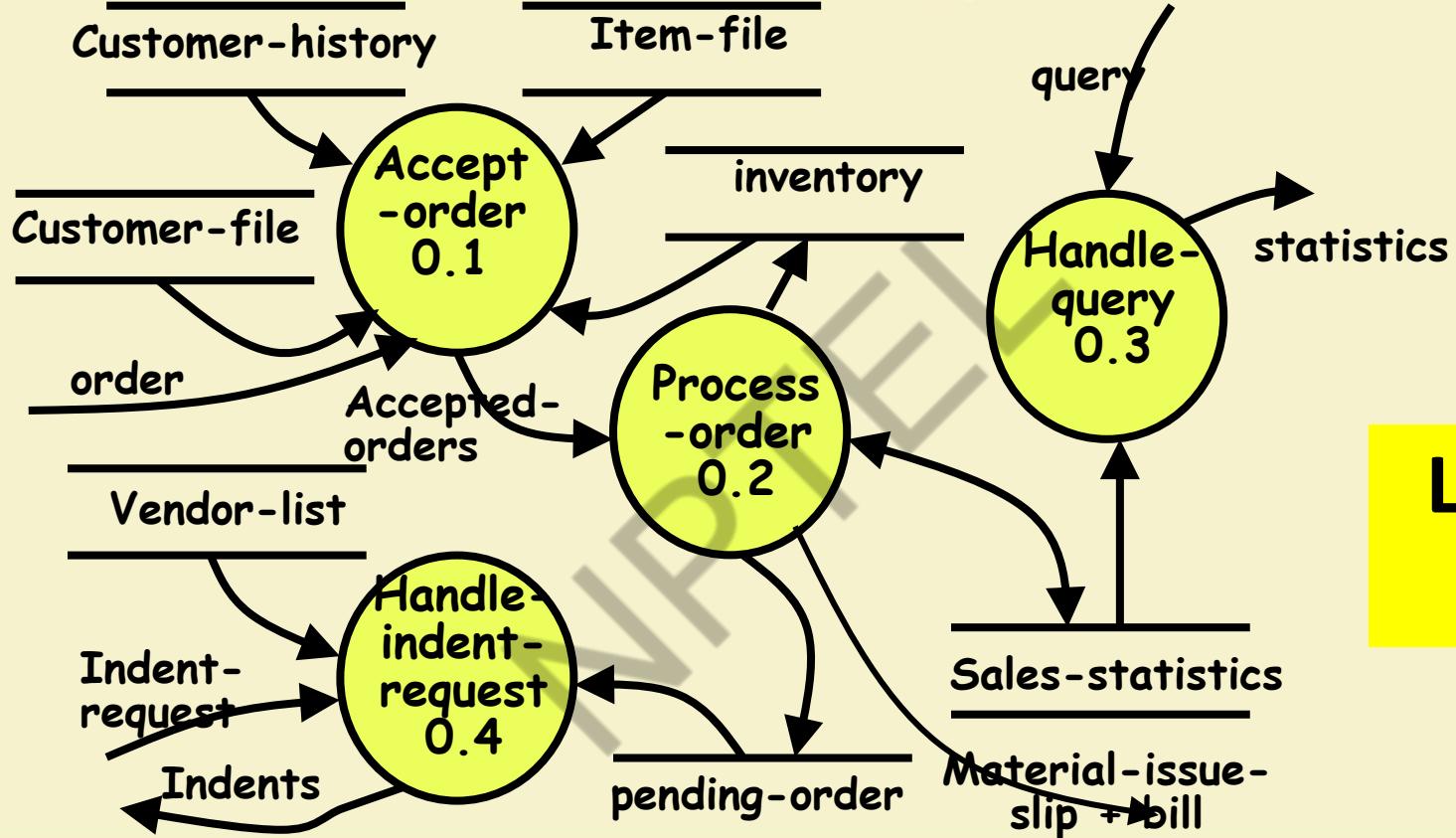
# Context Diagram



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



**Level 1  
DFD**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- response: [bill + material-issue-slip, reject-message]
- query: period /\* query from manager regarding sales statistics \*/
- period: [date+date,month,year,day]
- date: year + month + day
- year: integer
- month: integer
- day: integer
- order: customer-id + {items + quantity}\*  
• accepted-order: order /\* ordered items available in inventory \*/
- reject-message: order + message /\* rejection message \*/
- pending-orders: customer-id + {items+quantity}\*  
• customer-address: name+house#+street#+city+pin

## Example: Data Dictionary

## Example: Data Dictionary

- item-name: string
- house#: string
- street#: string
- city: string
- pin: integer
- customer-id: integer
- bill: {item + quantity + price}\* + total-amount + customer-address
- material-issue-slip: message + item + quantity + customer-address
- message: string
- statistics: {item + quantity + price }\*
- sales-statistics: {statistics}\*  
• quantity: integer

# Observation

- From the discussed examples,
  - Observe that DFDs help create:
    - **Data model**
    - **Function model**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Observation

- As a DFD is refined into greater levels of detail:
  - The analyst performs an **implicit functional decomposition**.
  - At the same time, refinements of data takes place.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Guidelines For Constructing DFDs

- Context diagram should represent the system as a single bubble:
  - Many beginners commit the mistake of drawing more than one bubble in the context diagram.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Guidelines For Constructing DFDs

- All external entities should be represented in the context diagram:
  - External entities should not appear at any other level DFD.
- Only 3 to 7 bubbles per diagram should be allowed:
  - Each bubble should be decomposed to between 3 and 7 bubbles.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Guidelines For Constructing DFDs

- A common mistake committed by many beginners:
  - Attempting to represent control information in a DFD.
  - e.g. trying to represent the order in which different functions are executed.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Guidelines For Constructing DFDs

- A DFD model does not represent control information:
  - When or in what order different functions (processes) are invoked
  - The conditions under which different functions are invoked are not represented.
  - For example, a function might invoke one function or another depending on some condition.
  - **Many beginners try to represent this aspect by drawing an arrow between the corresponding bubbles.**

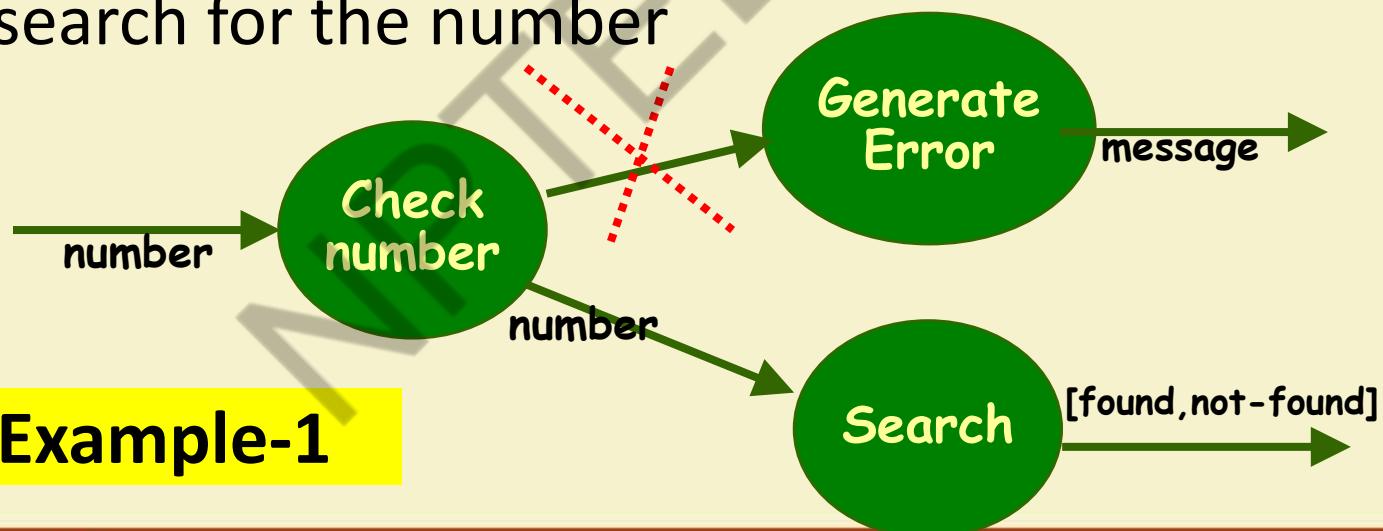


IIT KHARAGPUR

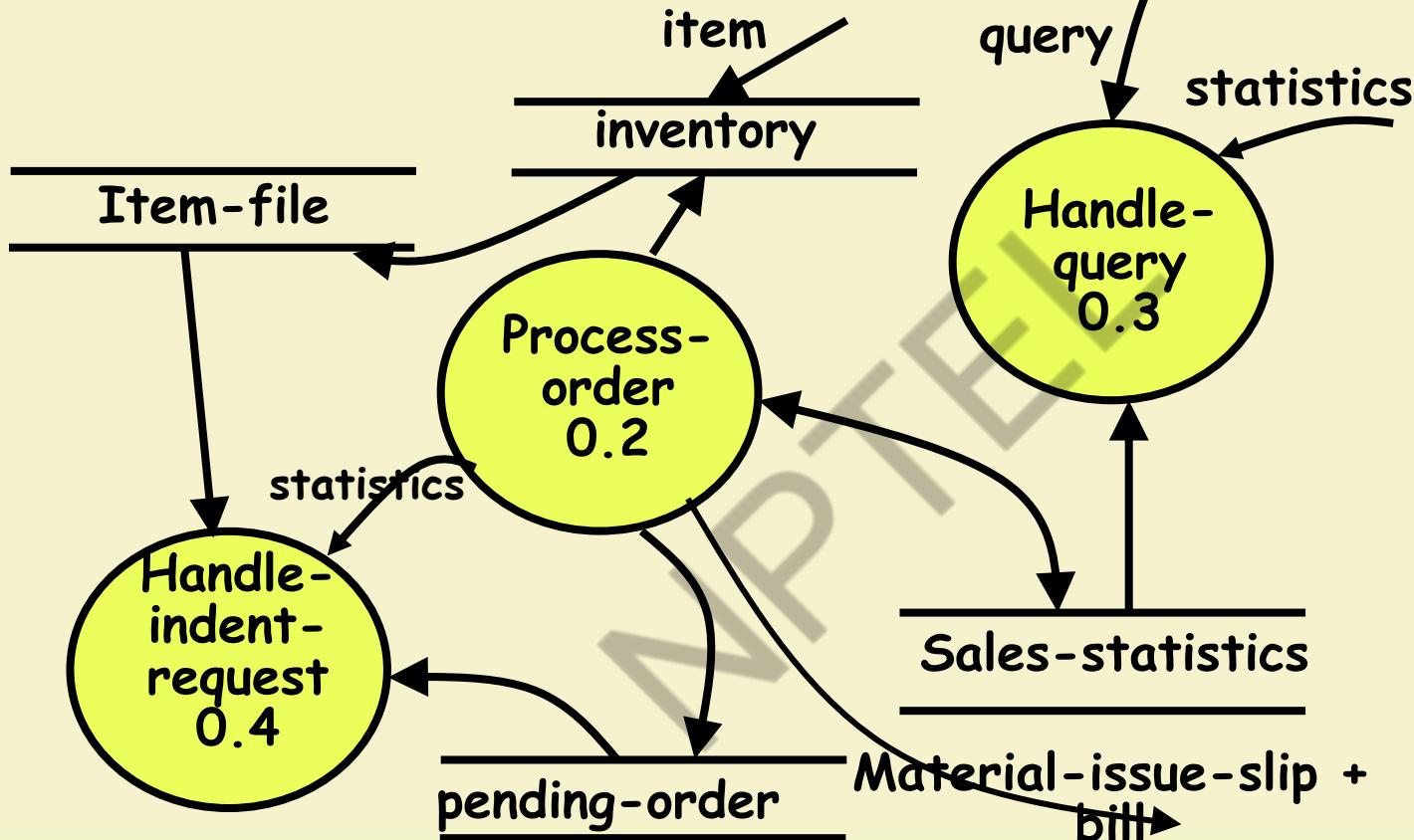


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Functionality: Check the input value:
  - If the input value is less than -1000 or greater than +1000 generate an error message
  - otherwise search for the number



# Find 4 Errors



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Common Mistakes in Constructing DFDs

- If a bubble A invokes either bubble B or bubble C depending on some conditions:
  - Represent the data that flows from bubble A to bubble B and bubbles A to C
  - Not the conditions depending on which a process is invoked.



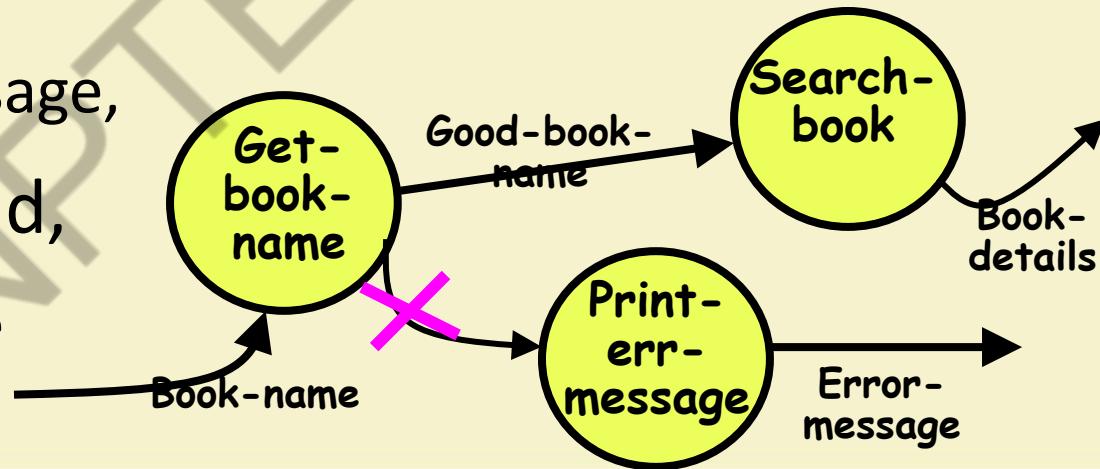
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Find Error Example-2

- A function accepts the book name to be searched from the user
- If the entered book name is not a valid book name
  - Generates an error message,
- If the book name is valid,
  - Searches the book name in database.



# Guidelines For Constructing DFDs

- All functions of the system must be captured in the DFD model:
  - **No function specified in the SRS document should be overlooked.**
- Only those functions specified in the SRS document should be represented:
  - **Do not assume extra functionality of the system not specified by the SRS document.**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Commonly Made Errors

- Unbalanced DFDs
- Forgetting to name the data flows
- Unrepresented functions or data
- External entities appearing at higher level DFDs
- Trying to represent control aspects
- Context diagram having more than one bubble
- A bubble decomposed into too many bubbles at next level
- Terminating decomposition too early
- Nouns used in naming bubbles



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Shortcomings of the DFD Model

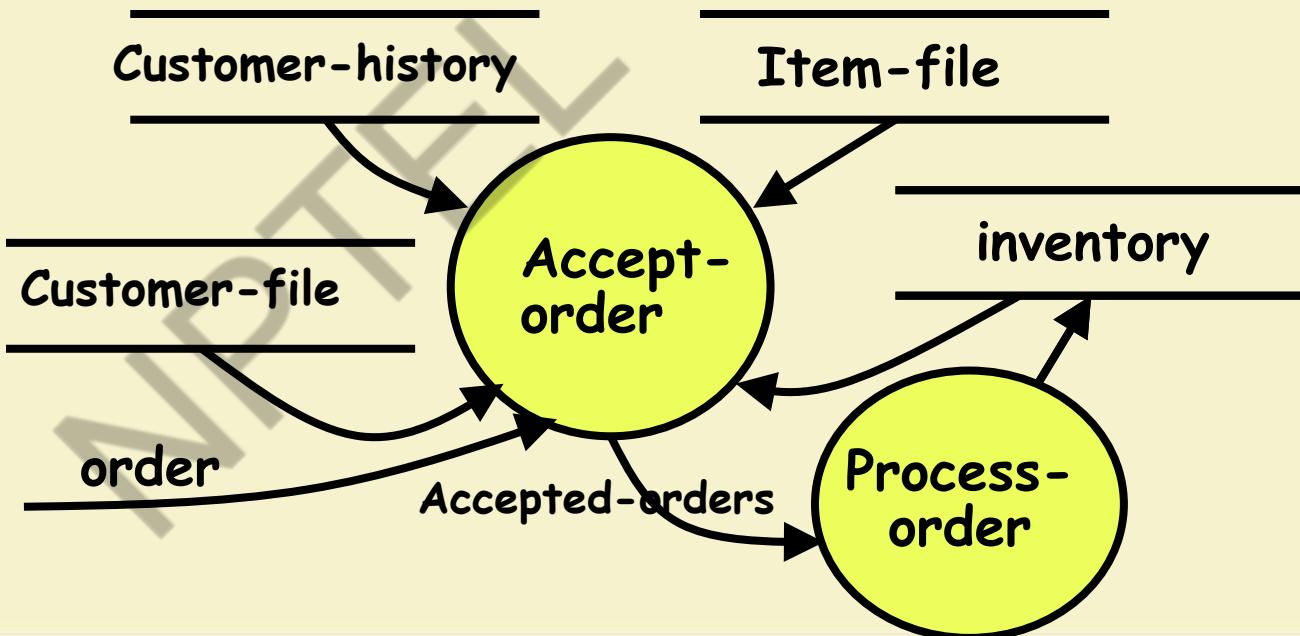
- DFD models suffer from several shortcomings:
- DFDs leave ample scope to be imprecise.
  - In a DFD model, we infer about the function performed by a bubble from its label.
  - A label may not capture all the functionality of a bubble.

# Shortcomings of the DFD Model

- For example, a bubble named **find-book-position** has only intuitive meaning:
  - Does not specify several things:
    - What happens when some input information is missing or is incorrect.
    - Does not convey anything regarding what happens when book is not found
    - What happens if there are books by different authors with the same book title.

# Shortcomings of the DFD Model

- Control information is not represented:
  - For instance, order in which inputs are consumed and outputs are produced is not specified.



# Shortcomings of the DFD Model

- Decomposition is carried out to arrive at the successive levels of a DFD is subjective.
- **The ultimate level to which decomposition is carried out is subjective:**
  - Depends on the judgement of the analyst.
- **Even for the same problem,**
  - **Several alternative DFD representations are possible:**
  - **Many times it is not possible to say which DFD representation is superior or preferable.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Shortcomings of the DFD Model

- DFD technique does not provide:
  - Any clear guidance as to how exactly one should go about decomposing a function:
  - One has to use subjective judgement to carry out decomposition.
- Structured analysis techniques do not specify when to stop a decomposition process:
  - To what length decomposition needs to be carried out.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Several commercial and free tools available.
- **Commercial:**
  - Visio
  - Smartdraw (30 day free trial)
  - Edraw
  - Creately
  - Visual analyst
- **Free:**
  - Dia (GNU open source)

**DFD  
Tools**

# Word of Caution

- Tools can be learnt and used with some effort.
- **But, too much focus on SA/SD case tools does not make you any more a good designer:**
  - Than an expert knowledge of the Word Package making you a famous writer of thriller stories.



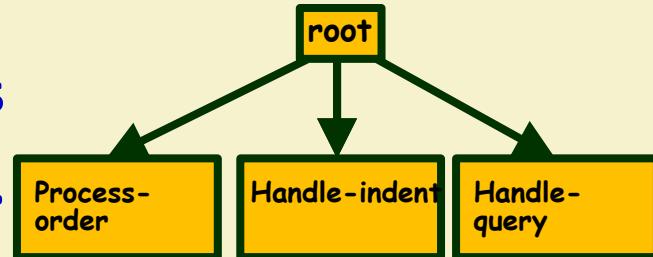
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

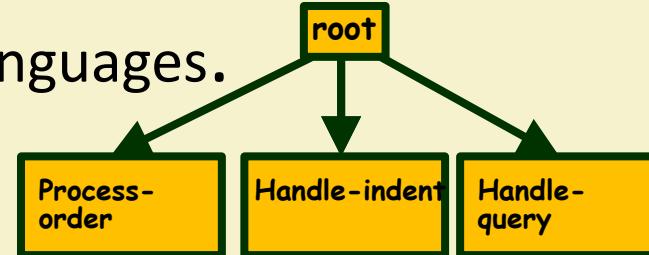
# Structured Design

- The aim of structured design
  - Transform the results of structured analysis (DFD representation) into a structure chart.
- A structure chart represents the software architecture:
  - Various modules making up the system,
  - Module dependency (i.e. which module calls which other modules),
  - Parameters passed among different modules.



# Structure Chart

- Structure chart representation
  - Easily implementable using programming languages.
- Main focus of a structure chart:
  - Define the module structure of a software,
  - Interaction among different modules,
  - **Procedural aspects (e.g, how a particular functionality is achieved) are not represented.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Basic Building Blocks of Structure Chart

- Rectangular box:
  - A rectangular box represents a module.
  - Annotated with the name of the module it represents.

Process-order



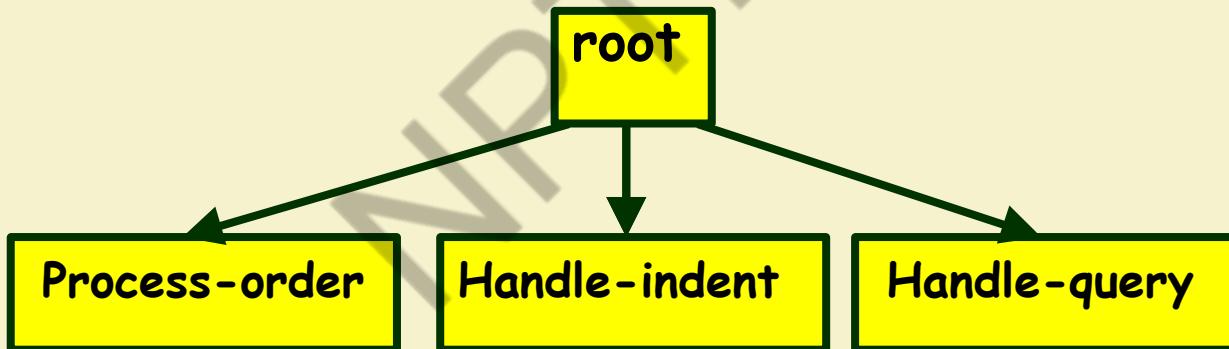
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Arrows

- An arrow between two modules implies:
  - During execution control is passed from one module to the other in the direction of the arrow.



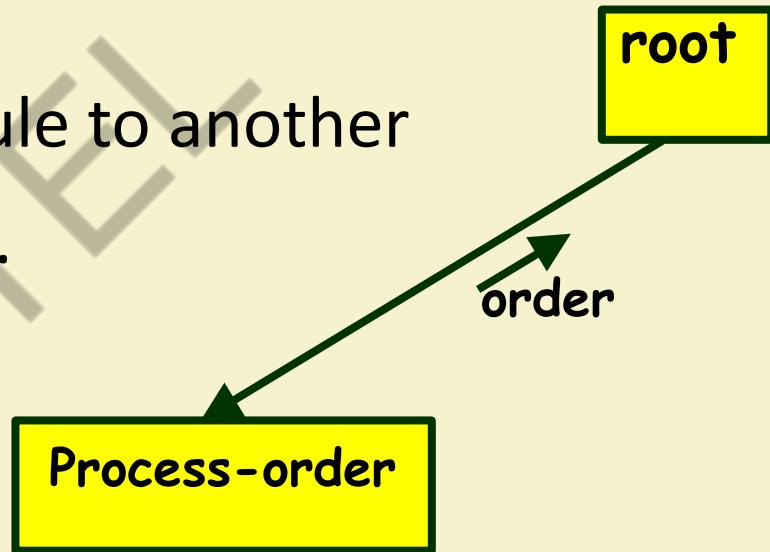
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data Flow Arrows

- Data flow arrows represent:
  - Data passing from one module to another in the direction of the arrow.



# Library Modules

- Library modules represent frequently called modules:
  - A rectangle with double side edges.
  - Simplifies drawing when a module is called by several modules.

Quick-sort



IIT KHARAGPUR

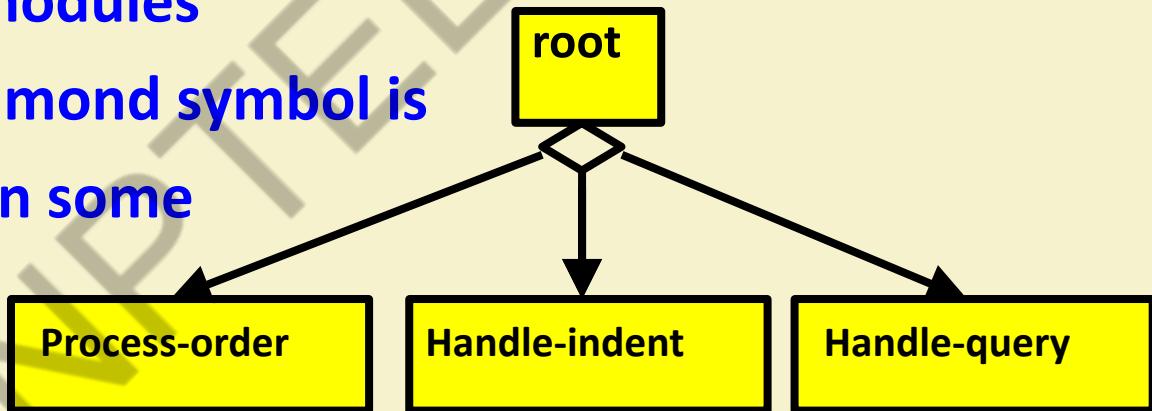


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Selection

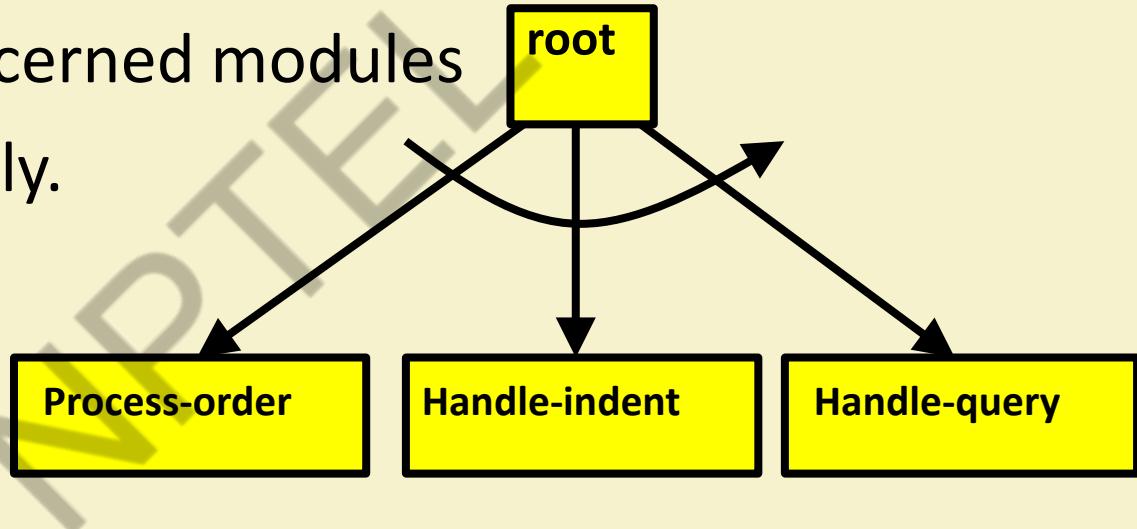
- The diamond symbol represents:

— Each one of several modules connected to the diamond symbol is invoked depending on some condition.



# Repetition

- A loop around control flow arrows denotes that the concerned modules are invoked repeatedly.



- There is only one module at the top:
  - the **root module**.
- There is at most one control relationship between any two modules:
  - if module A invokes module B,
  - Module B cannot invoke module A.
- The main reason behind this restriction:
  - **Modules in a structure chart should be arranged in layers or levels.**

## Structure Chart

# Structure Chart

- Makes use of principle of abstraction:
  - does not allow lower-level modules to invoke higher-level modules:
  - But, two higher-level modules can invoke the same lower-level module.

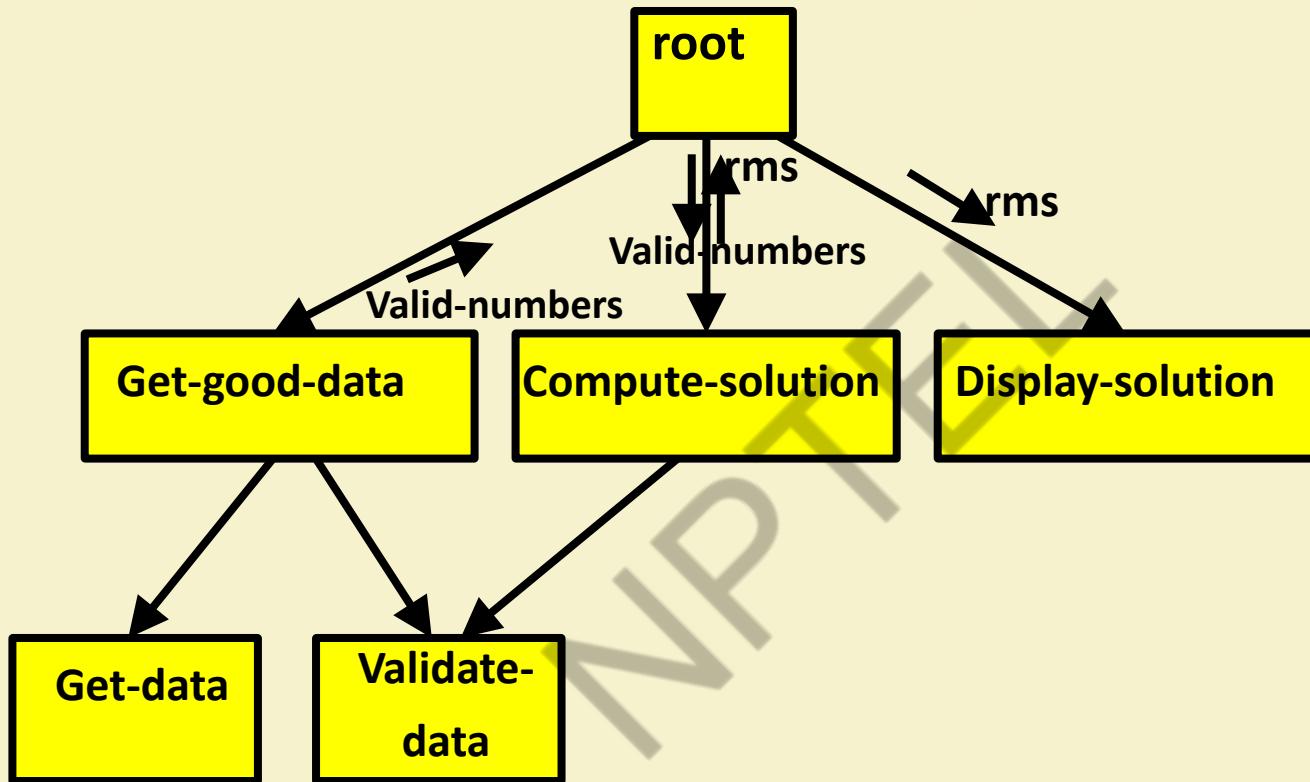


IIT KHARAGPUR

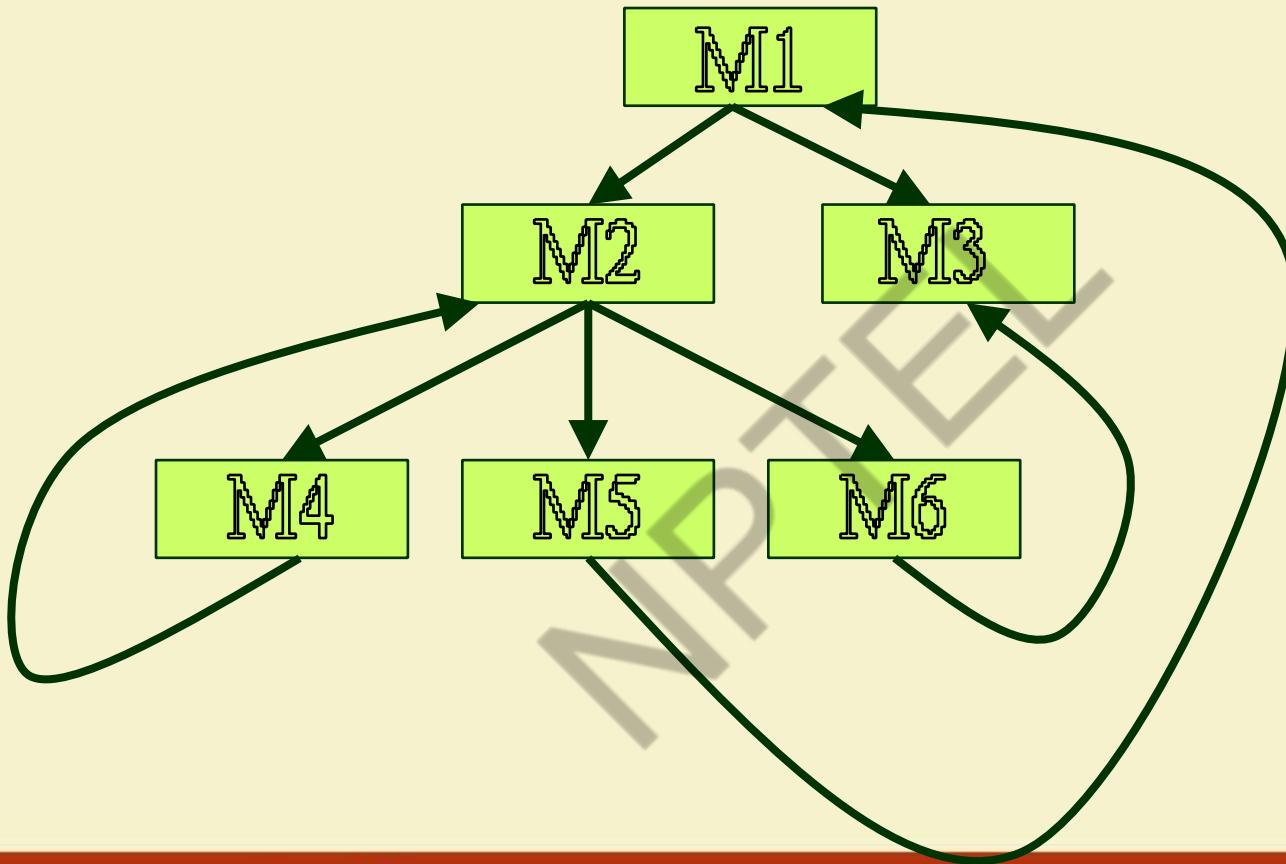


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example: Good Design



## Example: Bad Design



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Shortcomings of Structure Chart

- By examining a structure chart:
  - we can not say whether a module calls another module just once or many times.
- Also, by looking at a structure chart:
  - we can not tell the order in which the different modules are invoked.



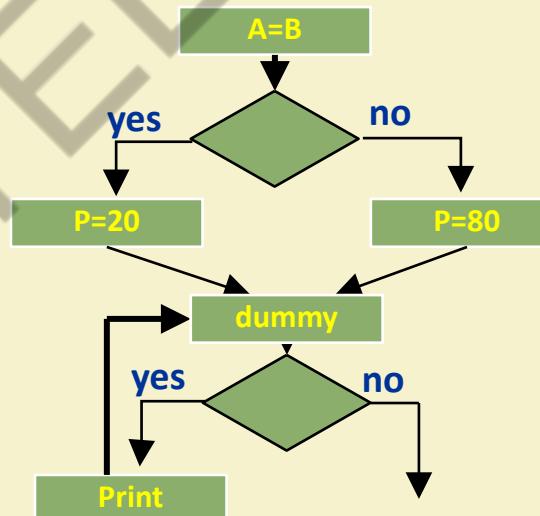
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Flow Chart (Aside)

- We are all familiar with the flow chart representations:
  - Flow chart is a convenient technique to represent the flow of control in a system.
- A=B
- if(c == 100)
- P=20
- else p= 80
- while(p>20)
- print(student mark)



# Flow Chart versus Structure Chart

1. It is difficult to identify modules of a software from its flow chart representation.
2. Data interchange among the modules is not represented in a flow chart.
- 3. Sequential ordering of tasks inherent in a flow chart is suppressed in a structure chart.**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Transformation of a DFD Model into Structure Chart

- Two strategies exist to guide transformation of a DFD into a structure chart:
  - Transform Analysis
  - Transaction Analysis



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Transform Analysis

- The first step in transform analysis:
  - Divide the DFD into 3 parts:
    - **input,**
    - **logical processing,**
    - **output.**



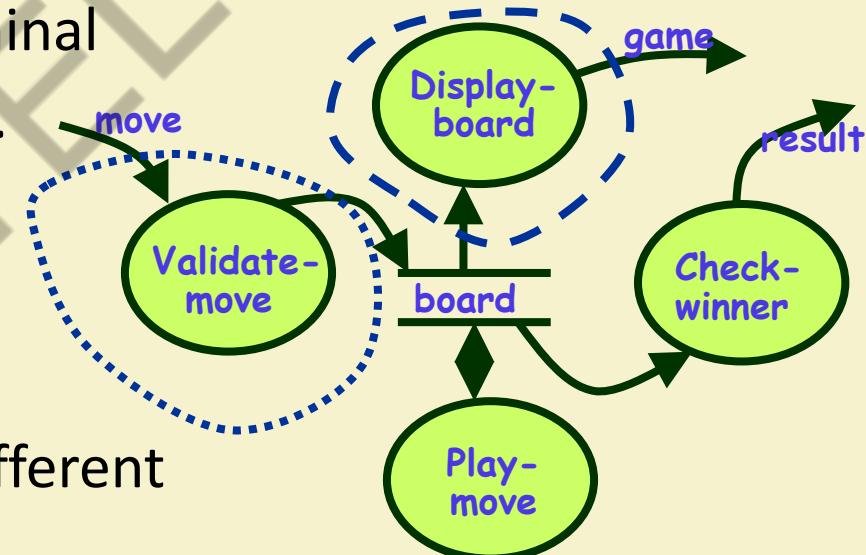
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Transform Analysis

- Input portion in the DFD:
  - processes which convert input data from physical to logical form.
  - e.g. read characters from the terminal and store in internal tables or lists.
- Each input portion:
  - called an **afferent branch**.
  - Possible to have more than one afferent branch in a DFD.



IIT KHARAGPUR

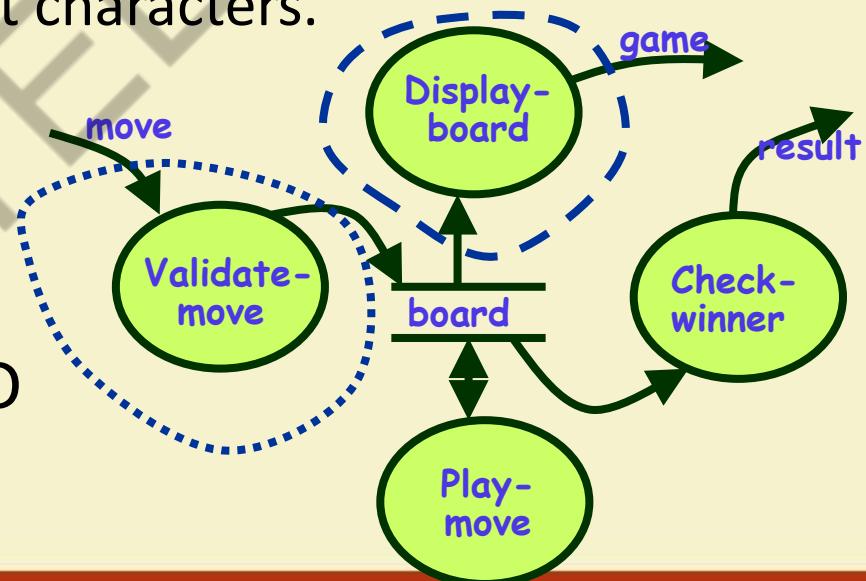


NPTEL  
ONLINE  
CERTIFICATION COURSES

- Output portion of a DFD:

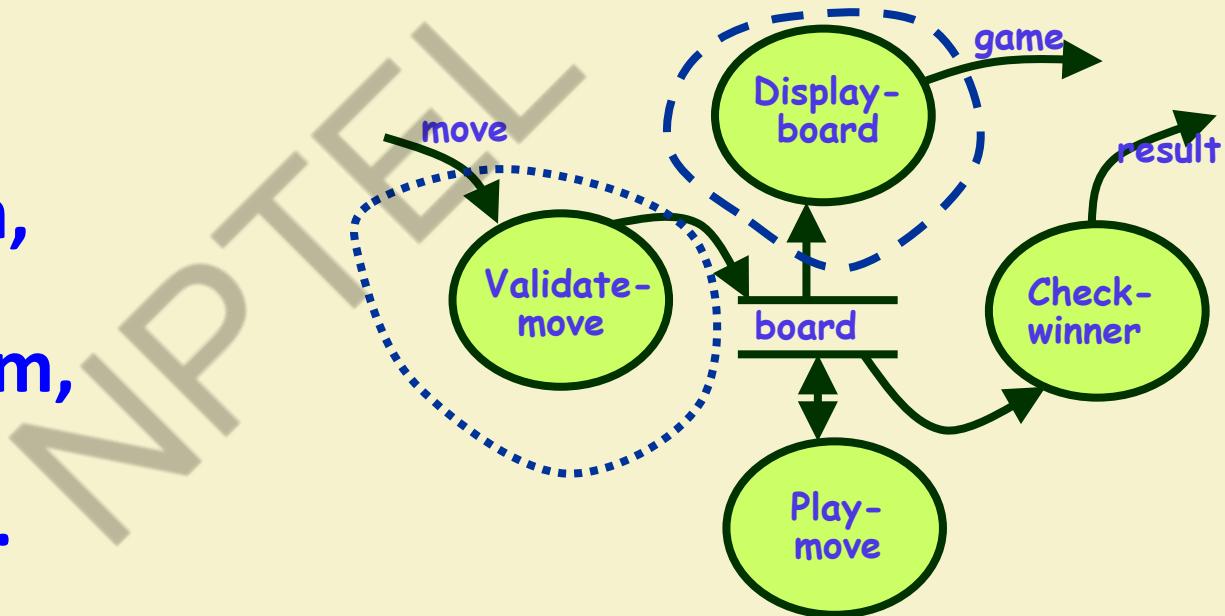
## Transform Analysis

  - transforms output data from logical form to physical form.
    - e.g., from list or array into output characters.
  - Each output portion:
    - called an **efferent branch**.
- The remaining portions of a DFD
  - called **central transform**



# Transform Analysis

- Derive structure chart by drawing one functional component for:
  - **afferent branch,**
  - **central transform,**
  - **efferent branch.**



- Identifying input and output transforms:
  - requires experience and skill.
- Some guidelines for identifying central transforms:
  - Trace inputs until a bubble is found whose output cannot be deduced from the inputs alone.
  - Processes which validate input are not central transforms.
  - Processes which sort input or filter data from it are.

## Transform Analysis

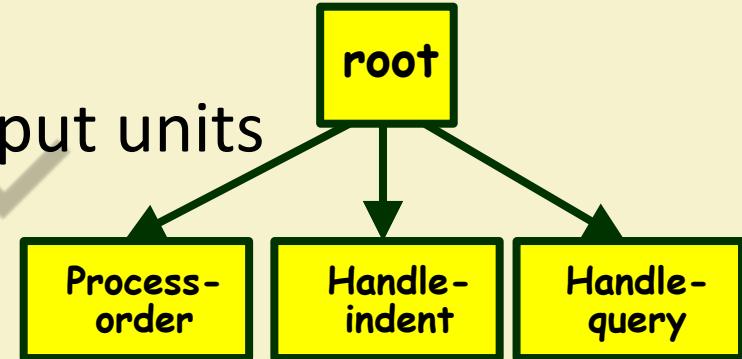
# Transform Analysis

- First level of structure chart:

- Draw a box for each input and output units
  - A box for the central transform.

- Next, refine the structure chart:

- Add subfunctions required by each high-level module.
  - Many levels of modules may required to be added.



## Factoring

- The process of breaking functional components into subcomponents.
- Factoring includes adding:
  - **Read and write modules,**
  - **Error-handling modules,**
  - **Initialization and termination modules,etc.**
- Finally check:
  - Whether all bubbles have been mapped to modules.

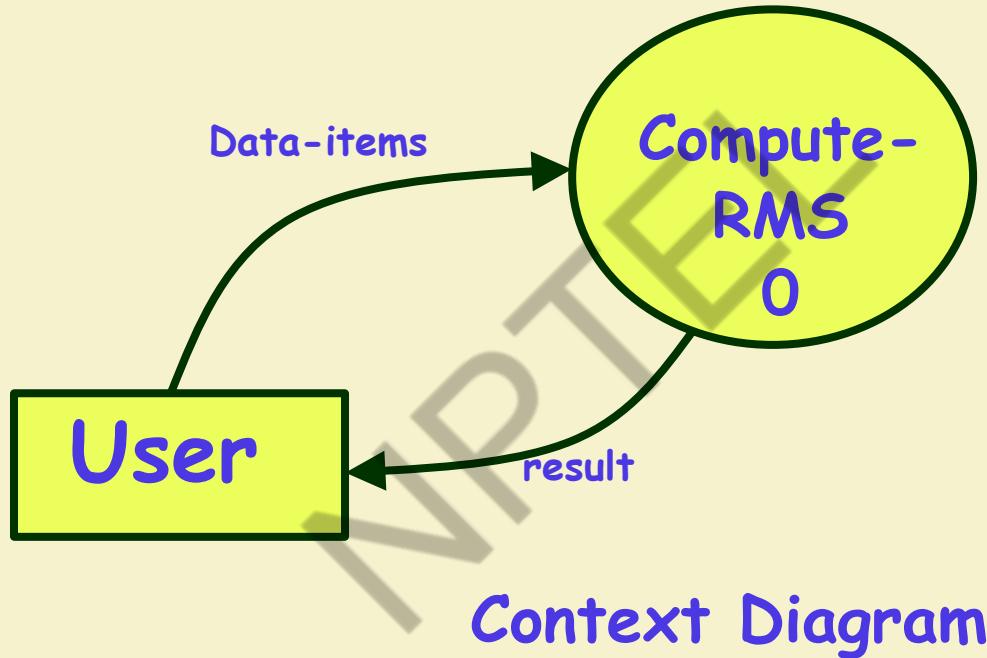


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Example 1: RMS Calculating Software



## Example 1: RMS Calculating Software

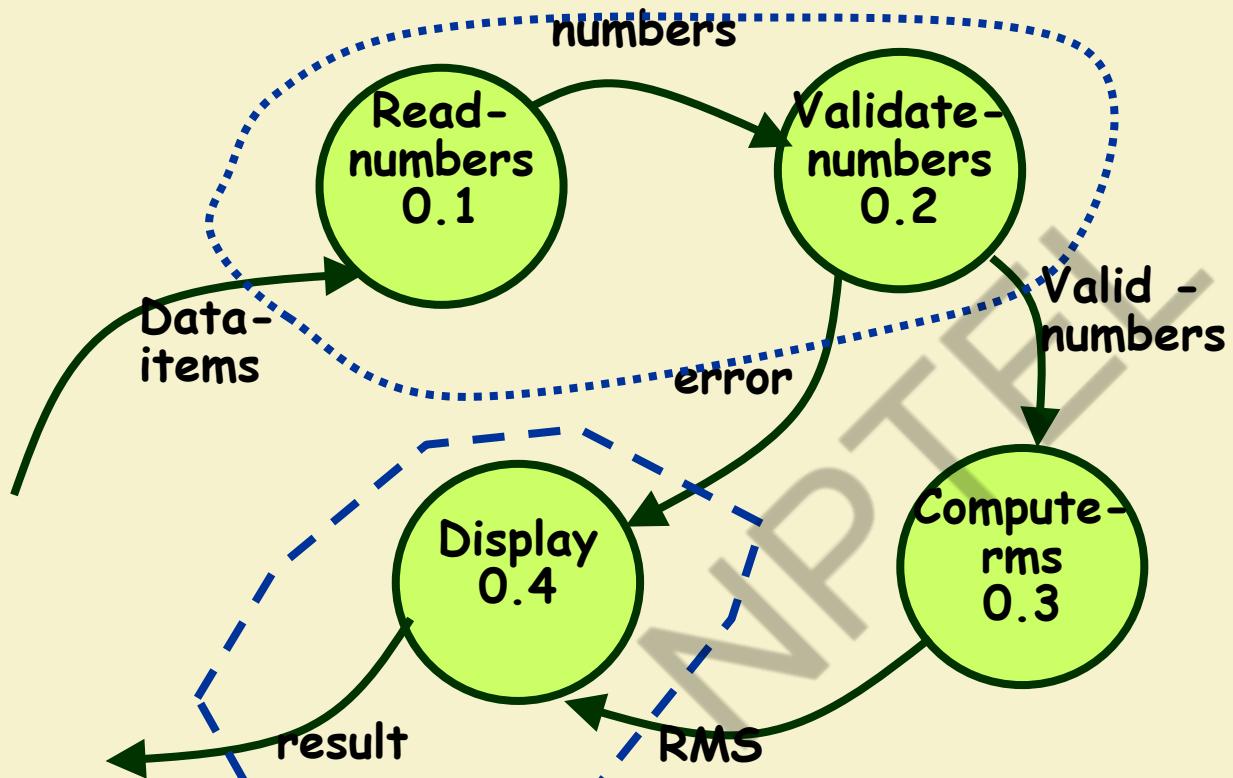
- From a cursory analysis of the problem description,
  - easy to see that the system needs to perform:
    - accept the input numbers from the user,
    - validate the numbers,
    - calculate the root mean square of the input numbers,
    - display the result.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



## Example 1: RMS Calculating Software

# Example 1: RMS Calculating Software

- By observing the level 1 DFD:
  - Identify read-number and validate-number bubbles as the afferent branch
  - Display as the efferent branch.

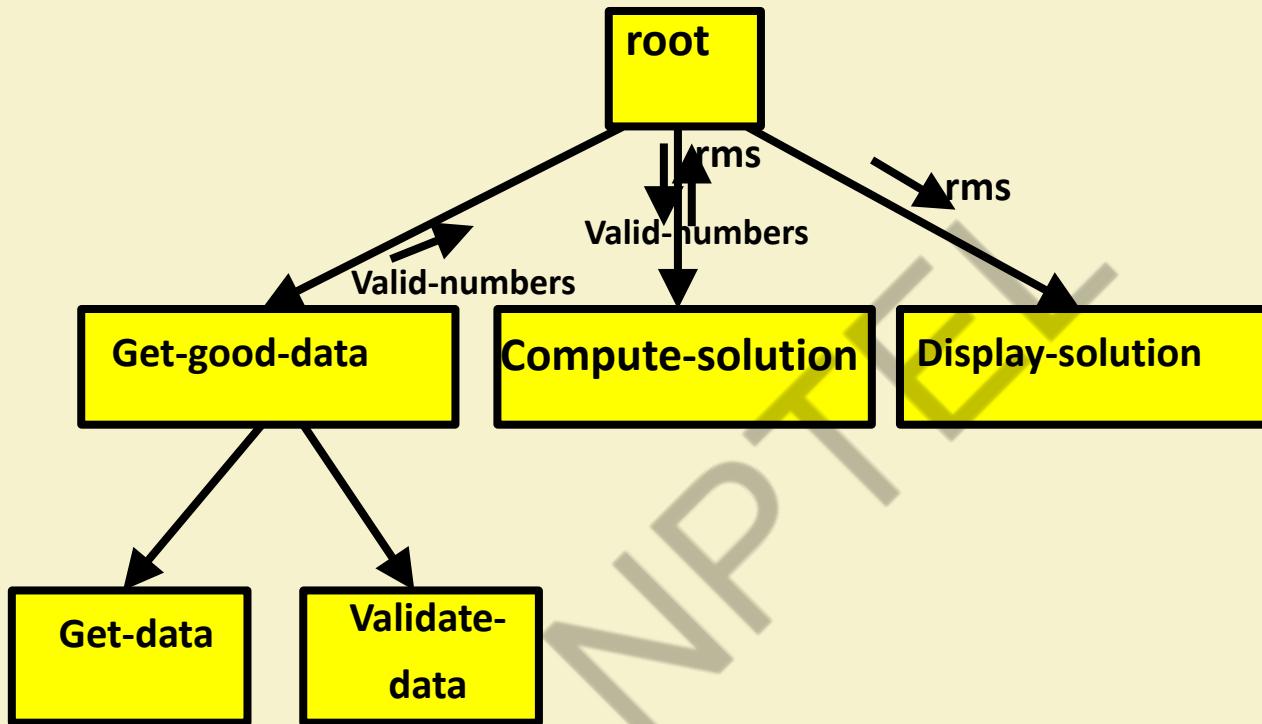


IIT KHARAGPUR

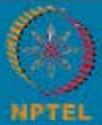


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example 1: RMS Calculating Software



IIT KHARAGPUR

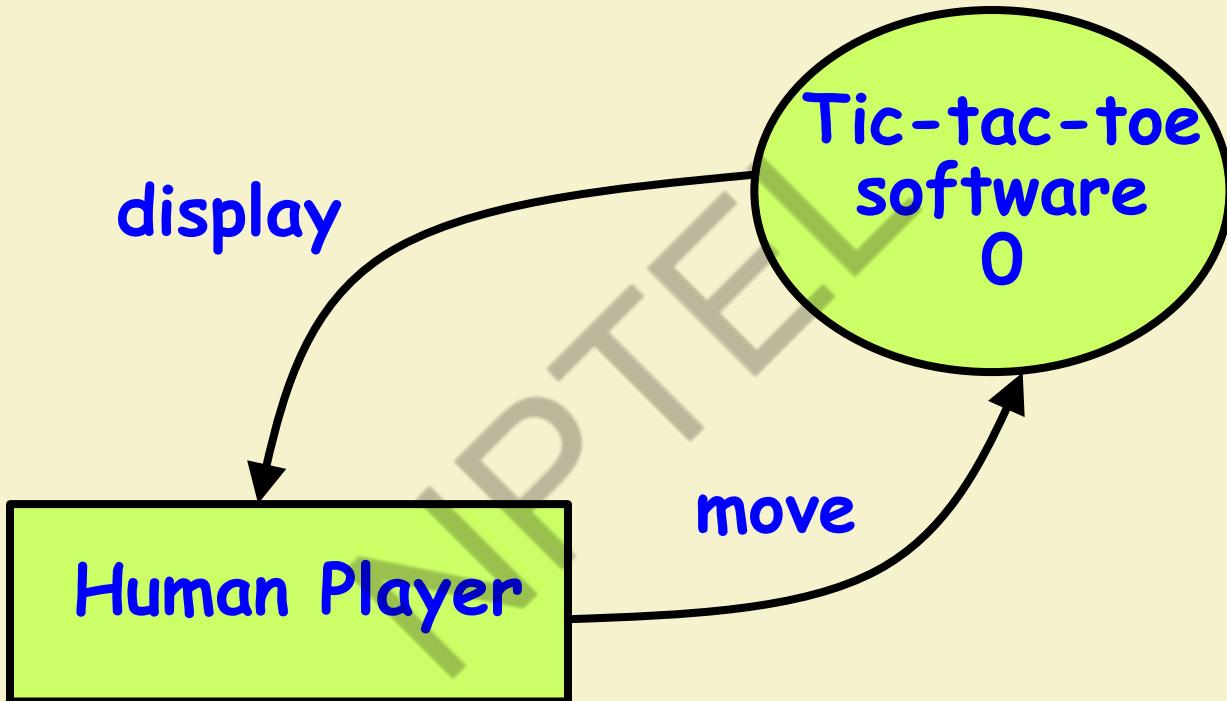


NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 2: Tic-Tac-Toe Computer Game

- As soon as either of the human player or the computer wins,
  - A message congratulating the winner should be displayed.
- If neither player manages to get three consecutive marks along a straight line,
  - And all the squares on the board are filled up,
  - Then the game is drawn.
- The computer always tries to win a game.

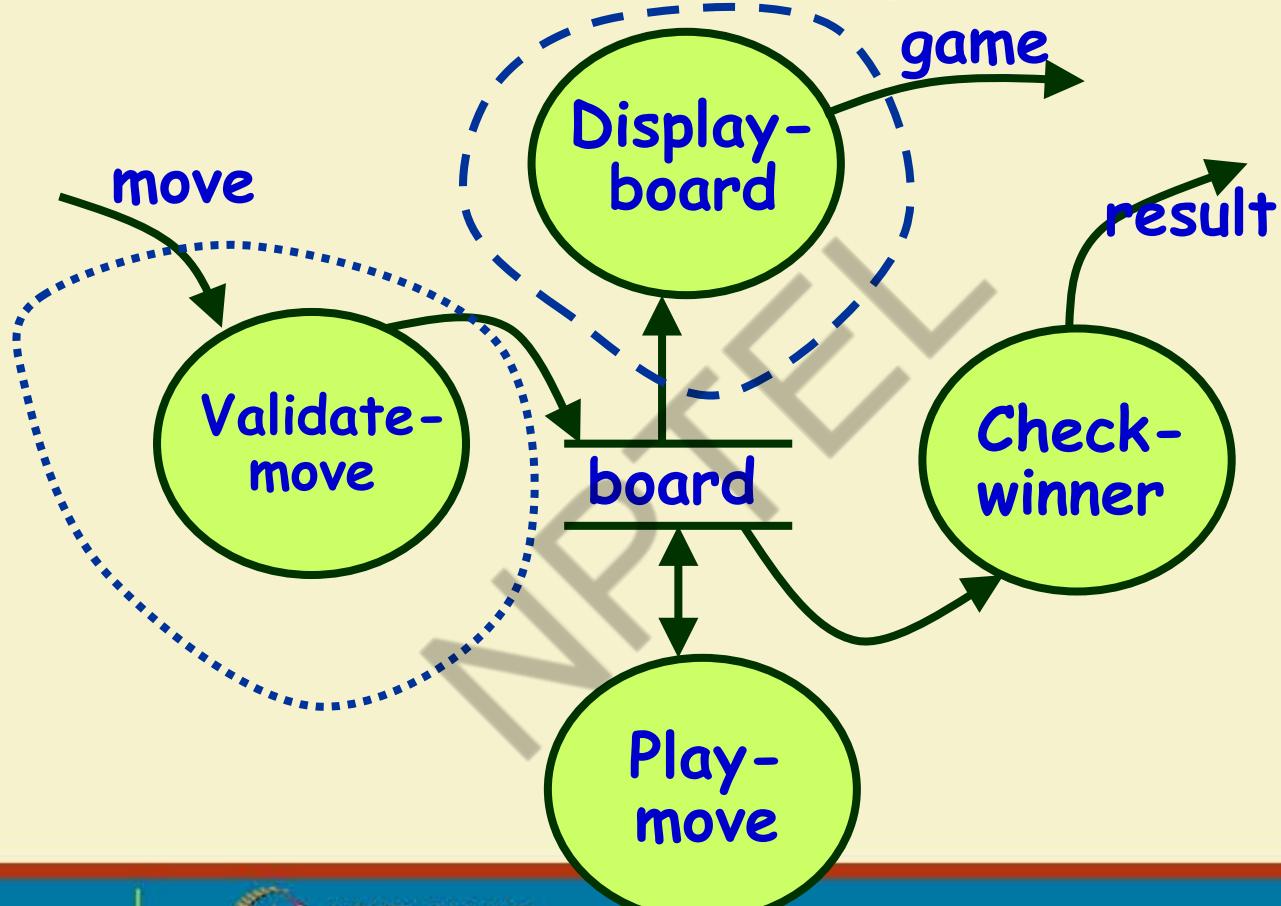
# Context Diagram for Example 2



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



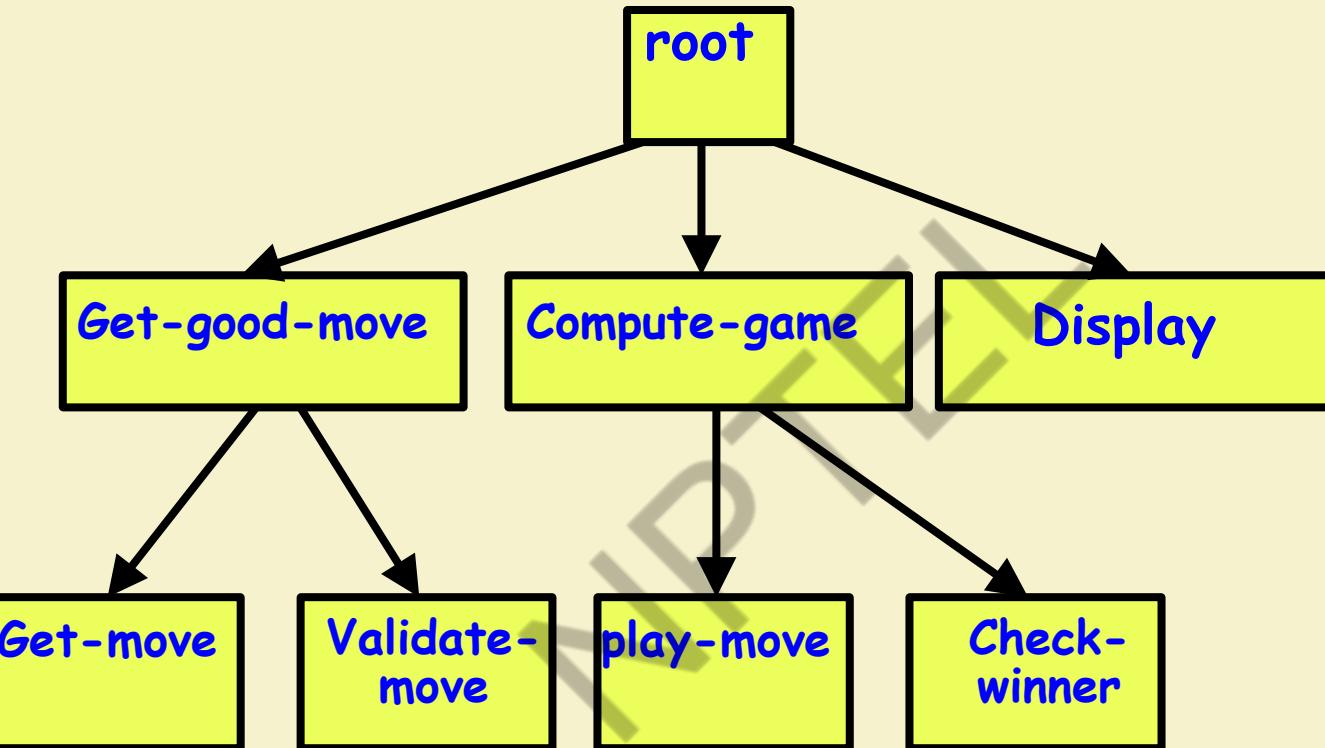
**Level 1  
DFD**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



## Structure Chart

# Transaction Analysis

- Useful for designing transaction processing programs.
  - **Transform-centered systems:**
    - Characterized **by similar processing steps for every data item** processed by input, process, and output bubbles.
  - **Transaction-driven systems,**
    - **One of several possible paths** through the DFD is traversed depending upon the input data value.



IIT KHARAGPUR

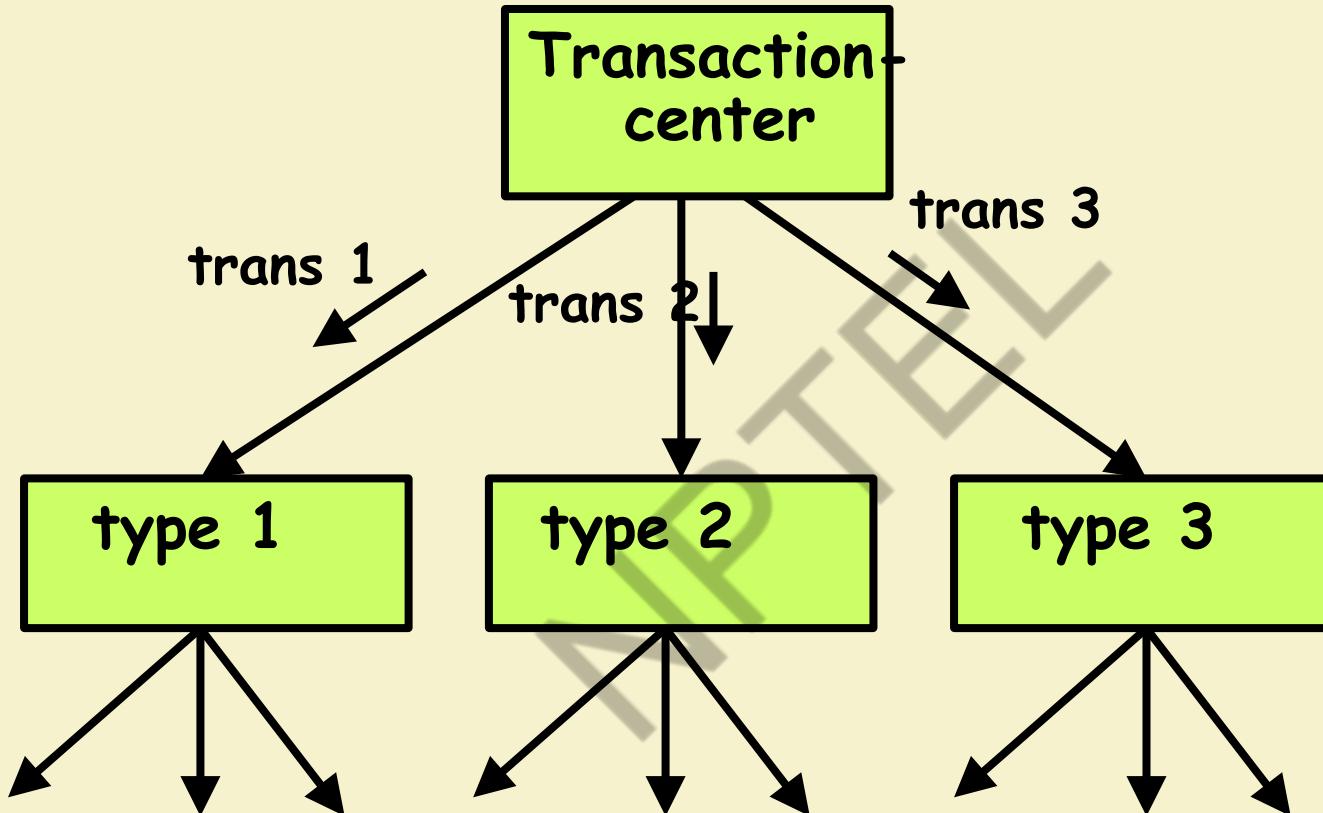


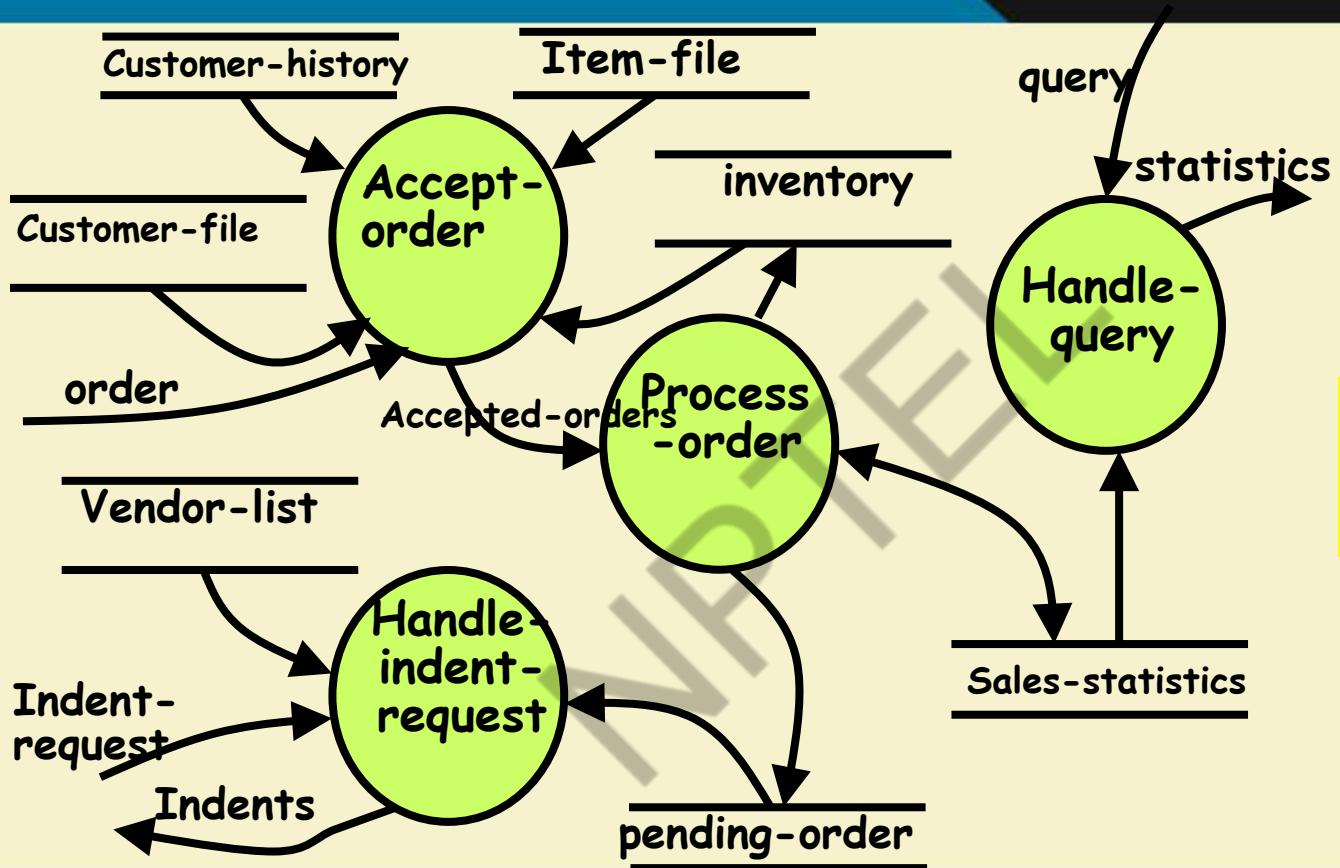
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Transaction Analysis

- **Transaction: Any input data value that triggers an action:**
  - For example, different selected menu options might trigger different functions.
  - Represented by a tag identifying its type.
- Transaction analysis uses this tag to divide the system into:
  - **Several transaction modules**
  - **One transaction-center module.**

# Transaction analysis





## Level 1 DFD for TAS

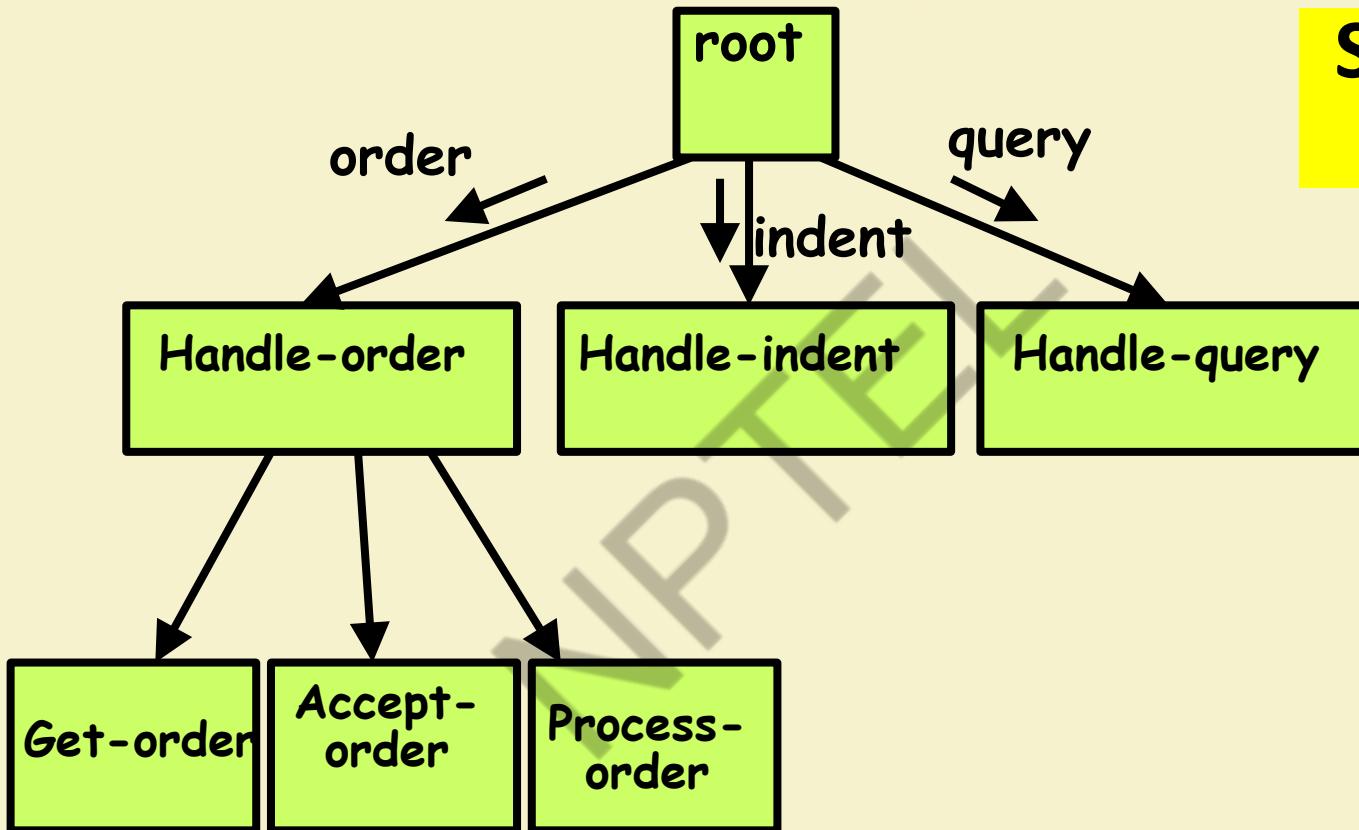


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Structure Chart



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Summary

- We discussed a sample function-oriented software design methodology:
  - Structured Analysis/Structured Design(SA/SD)
  - Incorporates features from some important design methodologies.
- SA/SD consists of two parts:
  - Structured analysis
  - Structured design.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Thank You!!

NPTEL



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

Faculty Name  
Department Name



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Object-Oriented Design using UML

Rajib Mall

CSE Department

IIT KHARAGPUR

# Introduction

- Object-oriented design (OOD) techniques are now extremely popular:
  - Inception in early 1980's and nearing maturity.
  - Widespread acceptance in industry and academics.
  - **Unified Modelling Language (UML) became an ISO standard (ISO/IEC 19501) in 2004.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Object Modelling Using UML

- **UML** is a modelling language.
  - Not a system design or development methodology
- Used to document object-oriented analysis and design results.
- Independent of any specific design methodology.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# UML Origin

- OOD in late 1980s and early 1990s:
  - Different software development houses were using different notations.
  - **Methodologies were tied to notations.**
- UML developed in early 1990s:
  - To standardize the large number of object-oriented modelling notations that existed.

# UML Lineology

- Based Principally on:
  - OMT [Rumbaugh 1991]
  - Booch's methodology [Booch 1991]
  - OOSE [Jacobson 1992]
  - Odell's methodology [Odell 1992]
  - Shlaer and Mellor [Shlaer 1992]

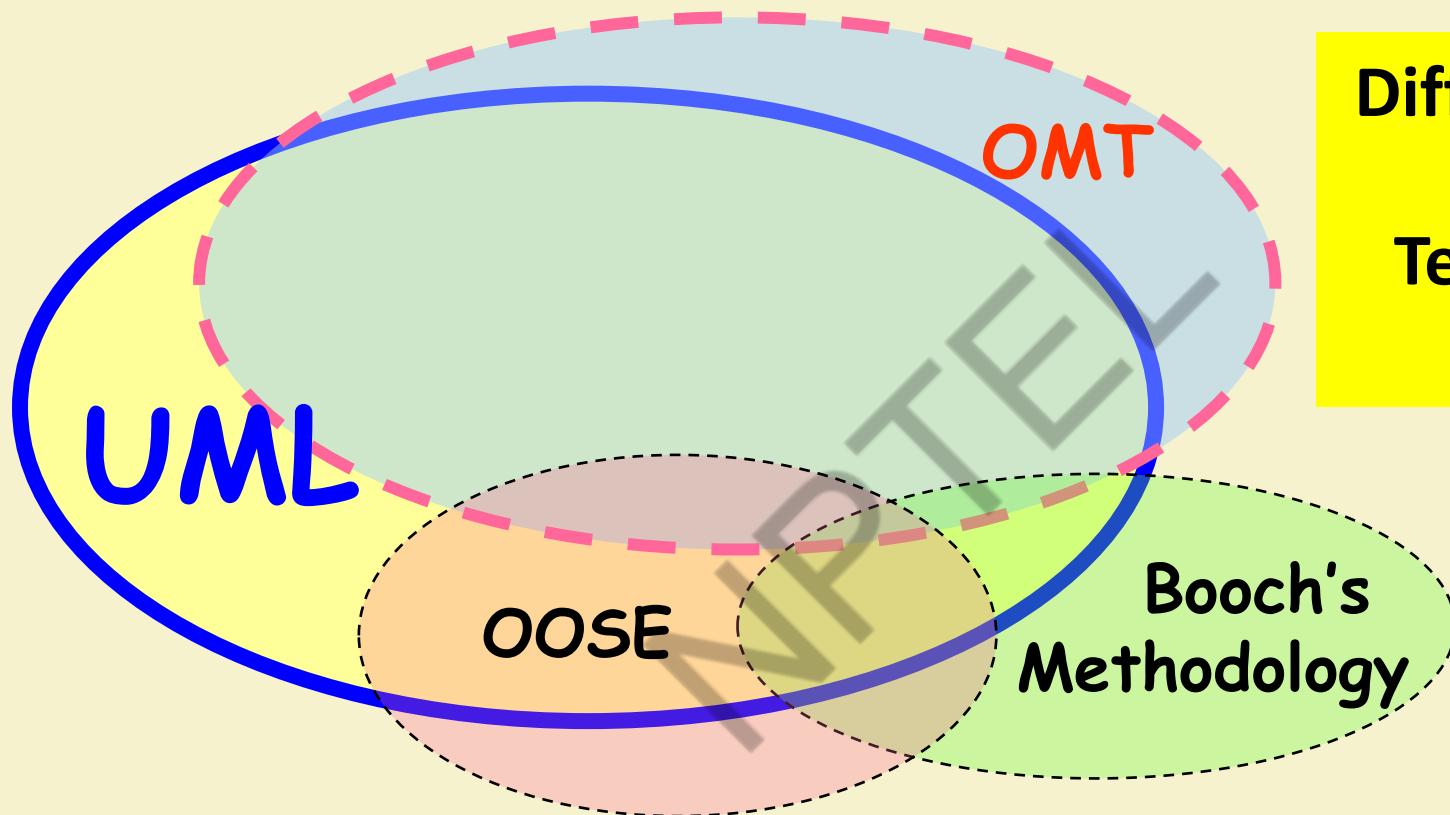


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Different Object Modeling Techniques in UML



IIT KHARAGPUR

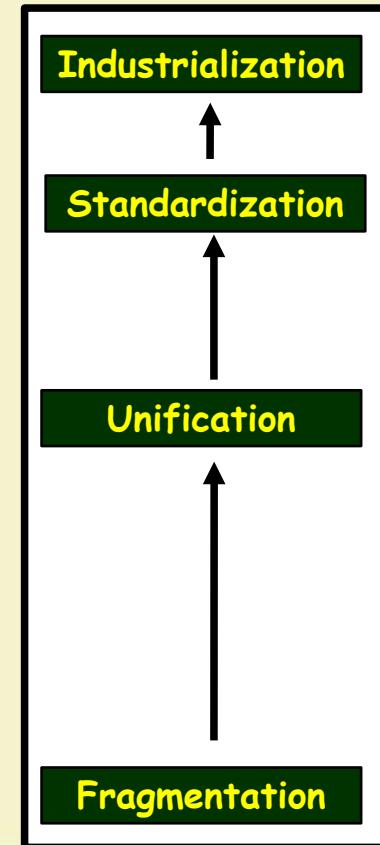
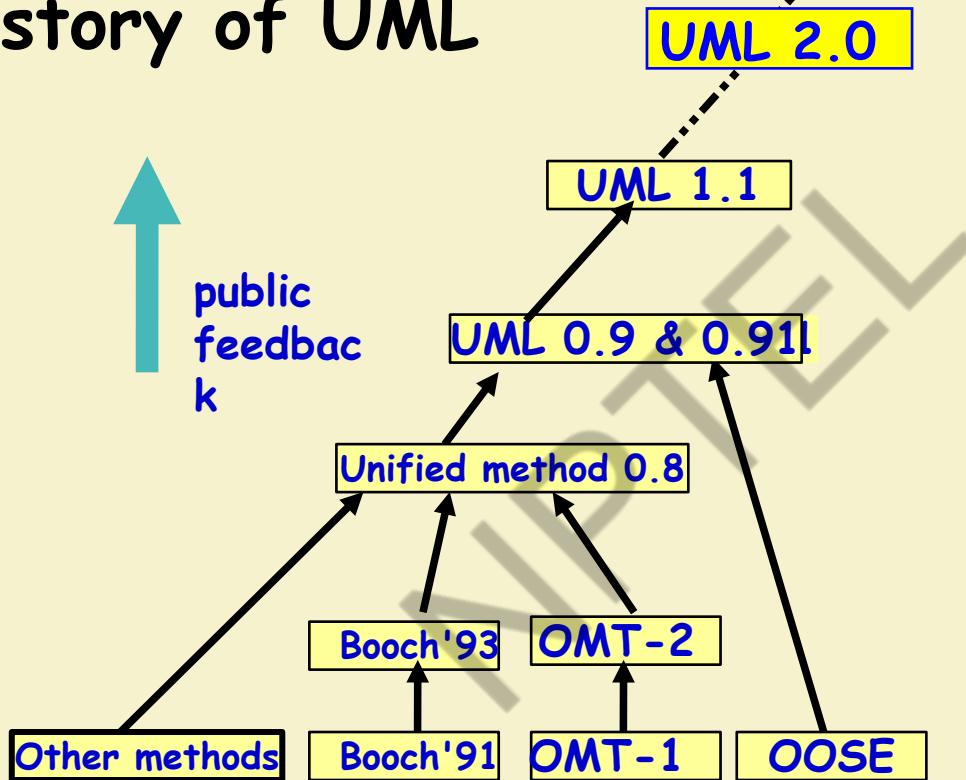


NPTEL  
ONLINE  
CERTIFICATION COURSES

# UML as A Standard

- Adopted by Object Management Group (OMG) in 1997.
- OMG is an association of industries
- Promotes consensus notations and techniques
- UML also being used outside software development area:
  - Example car manufacturing

# History of UML



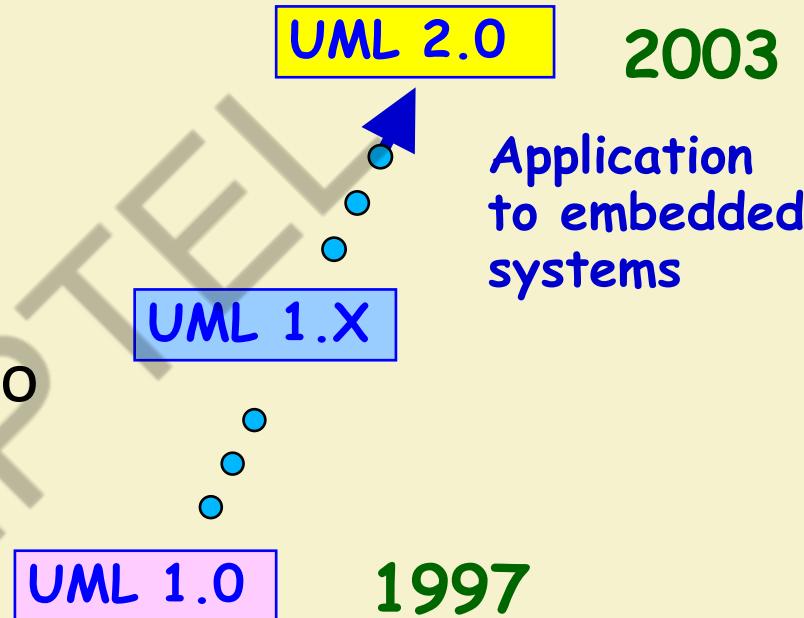
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Developments to UML

- UML continues to develop, due to:
  - Refinements
  - Making it applicable to new contexts



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Why are UML Models Required?

- Modelling is an abstraction mechanism:
  - Capture only important aspects and ignores the rest.
  - Different models obtained when different aspects are ignored.
  - An effective mechanism to handle complexity.
- UML is a graphical modelling technique
- Easy to understand and construct

# UML Diagrams

- Nine diagrams in UML1.x :
  - Used to capture 5 different views of a system.
- Views:
  - Provide different perspectives of a software system.
- Diagrams can be refined to get the actual implementation of a system.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# ● Views of a system:

- User's view
- Structural view
- Behavioral view
- Implementation view
- Environmental view

UML  
Model  
Views



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Diagrams and views in UML

## Structural View

- Class Diagram
- Object Diagram

## Behavioural View

- Sequence Diagram
- Collaboration Diagram
  - State-chart Diagram
  - Activity Diagram

## User's View

- Use Case Diagram

## Implementation View

- Component Diagram

## Environmental View

- Deployment Diagram



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Structural Diagrams

- **Class Diagram**
  - set of classes and their relationships.
- **Object Diagram**
  - set of objects (class instances) and their relationships
- **Component Diagram**
  - logical groupings of elements and their relationships
- **Deployment Diagram**
  - set of computational resources (nodes) that host each component.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Behavioral Diagrams

- **Use Case Diagram**
  - high-level behaviors of the system, user goals, external entities: actors
- **Sequence Diagram**
  - focus on time ordering of messages
- **Collaboration Diagram**
  - focus on structural organization of objects and messages
- **State Chart Diagram**
  - event driven state changes of system
- **Activity Diagram**
  - flow of control between activities



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Some Insights on Using UML

- “UML is a large and growing beast, but you don’t need all of it in every problem you solve...”  
– Martin Fowler
- “...when learning the UML, you need to be aware that certain constructs and notations are only helpful in detailed design while others are useful in requirements analysis ...” Brian Henderson-Sellers

# Are All Views Required for Developing A Typical System?

● NO

● For a simple system:

- Use case diagram, class diagram and one of the interaction diagrams only.

● State chart diagram:

- when class has significant states.
- When states are only one or two, state chart model becomes trivial

● Deployment diagram:

- In case several hardware components used to develop the system.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Use Case Modelling



IIT KHARAGPUR

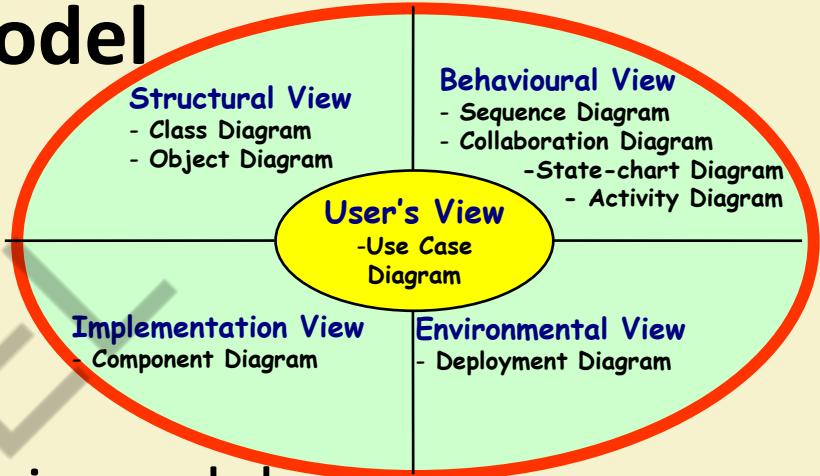
9/6/2018



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Use Case Model

- Consists of a set of “**use cases**”
- It is the central model:
  - Other models must conform to this model
  - Not really an object-oriented model, it is a functional model of a system



# A Use Case

- **A case of use:** A way in which a system can be used by the users to achieve specific goals
- Corresponds to a high-level requirement.
- Defines external behavior without revealing internal structure of system
- **Set of related scenarios tied together by a common goal.**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

–Use cases for a Library information system

- issue-book
- query-book
- return-book
- create-member
- add-book, etc.

Example  
Use Cases



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Are All Use Cases Independent?

- Use cases appear independent of each other
- However, Implicit dependencies may exist
- **Example:** In Library Automation System, renew-book and reserve-book are independent use cases.
  - But in actual implementation of renew-book--- **A check is made to see if any book has been reserved using reserve-book.**

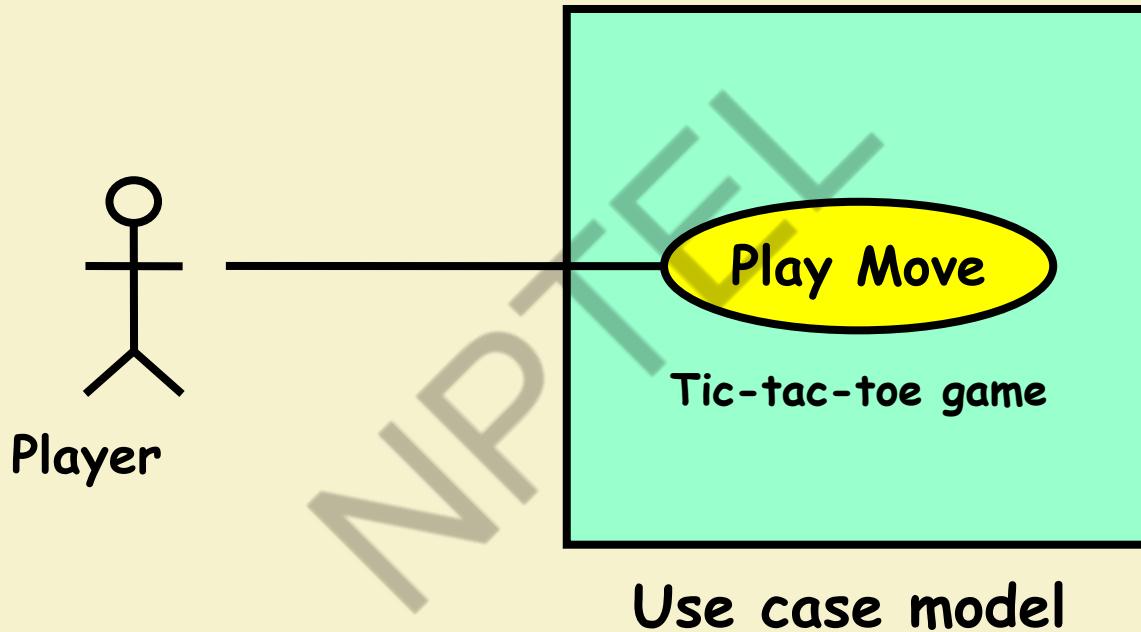


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# An Example Use Case Diagram



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Why Develop A Use Case Diagram?

- Serves as requirements specification
- How are actor identification useful in software development?
  - Identifies different categories of users:
    - Helps in implementing appropriate interfaces for each category of users.
    - Helps in preparing appropriate documents (e.g. users' manual).



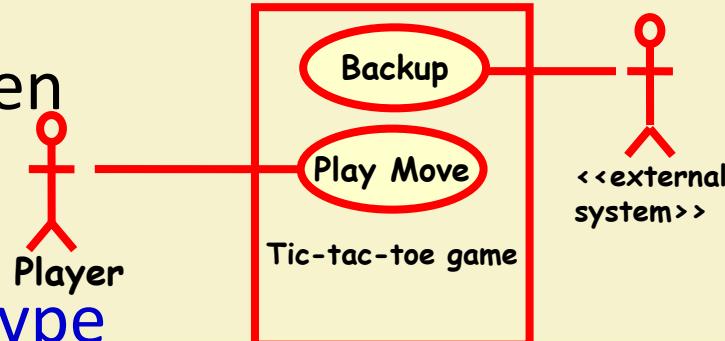
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

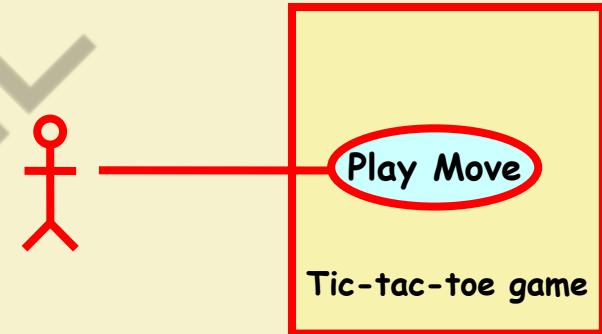
## Representation of Use Cases

- Represented in a use case diagram
- A use case is represented by an ellipse
- System boundary is represented by a rectangle
- Users are represented by stick person icons (actor)
- Communication relationship between actor and use case by a line
- External system by adding a stereotype



# What is a Connection?

- A connection is an association between an actor and a use case.
- Depicts a usage relationship
- Connection does not indicate data flow

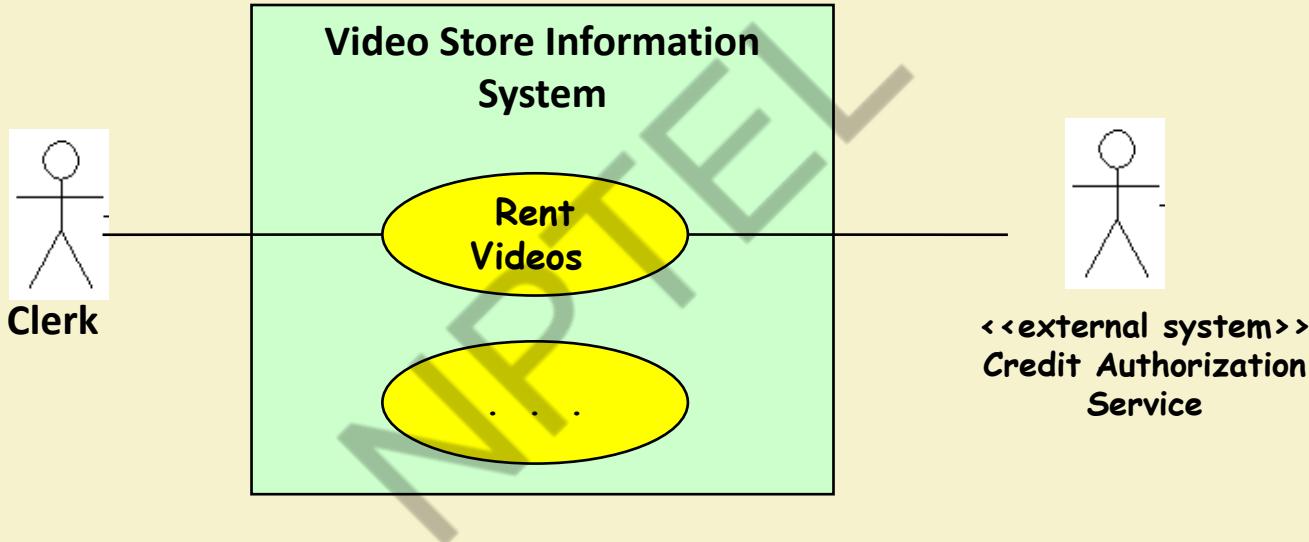


# Relationships between Use Cases and Actors

- Association relation indicates that the actor and the corresponding use case communicate with one another.



# Another Example Use Case Diagram

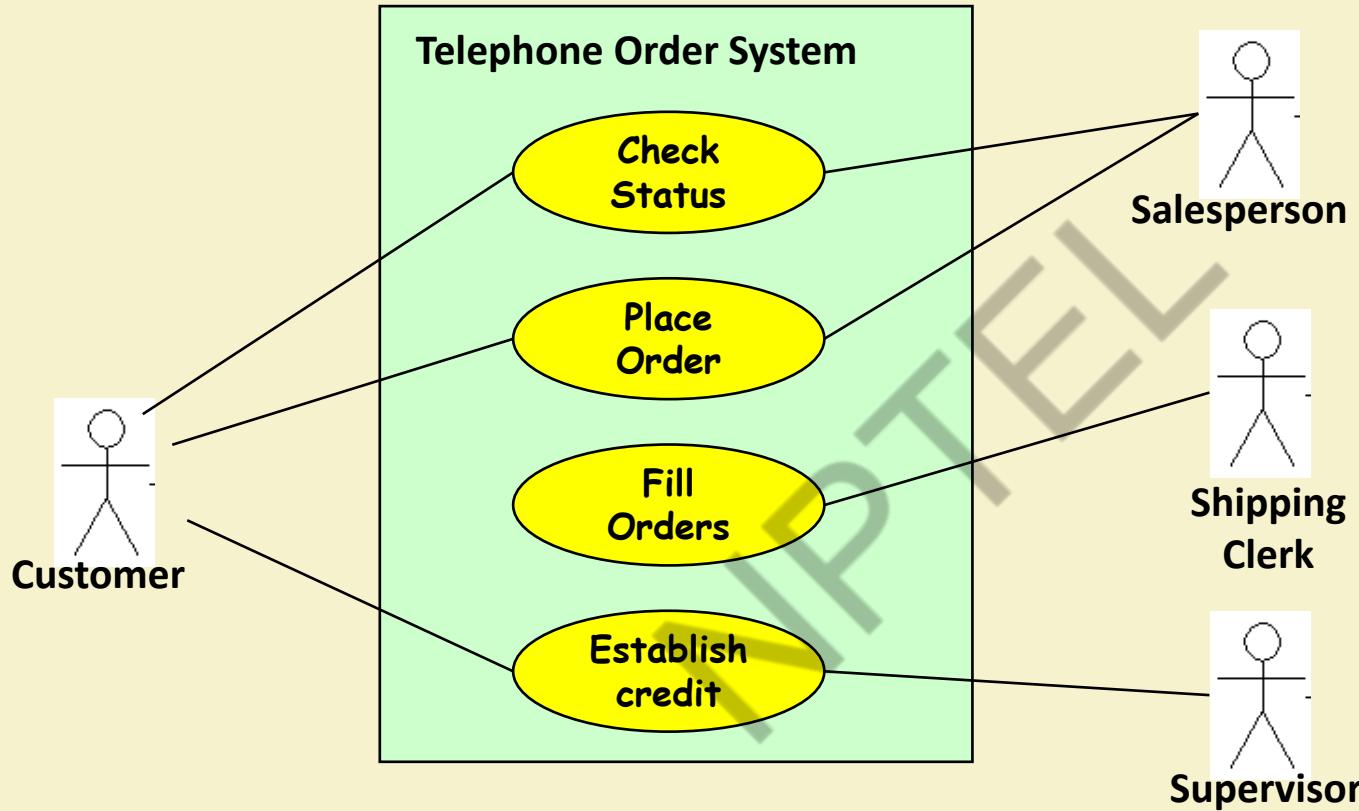


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Yet Another Use Case Example



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Factoring Use Cases

- Two main reasons for factoring:
  - Complex use cases need to be factored into simpler use cases
  - Helps represent common behavior across different use cases
- Three ways of factoring:
  - Generalization
  - Include
  - Extend



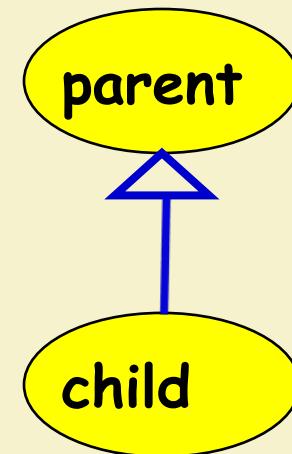
IIT KHARAGPUR



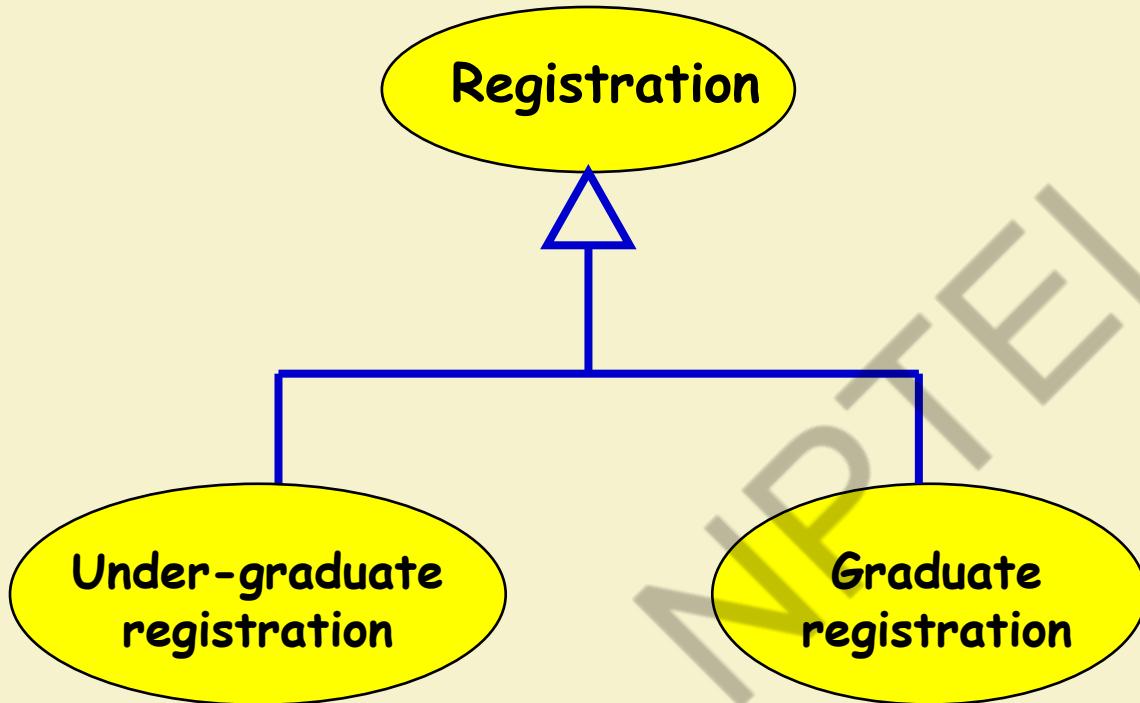
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Generalization

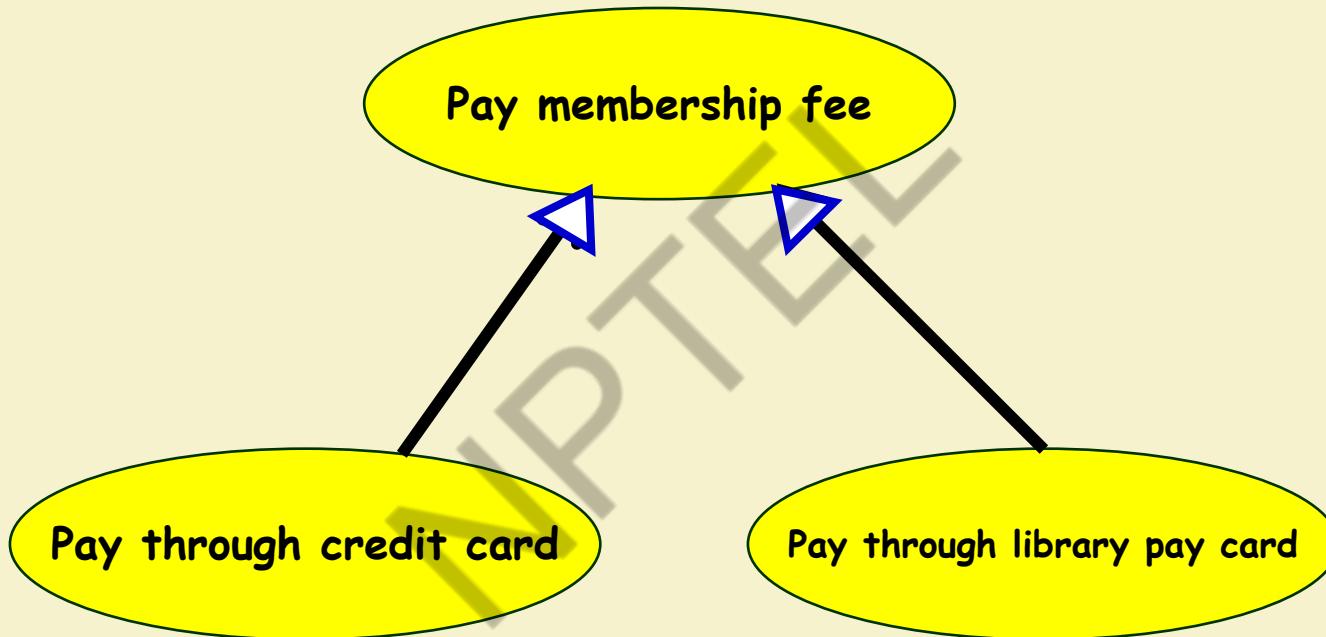
- The child use case inherits the behavior of the parent use case.
  - The child may add to or override some of the behavior of its parent.



## Generalization Example 1



# Factoring Use Cases Using Generalization

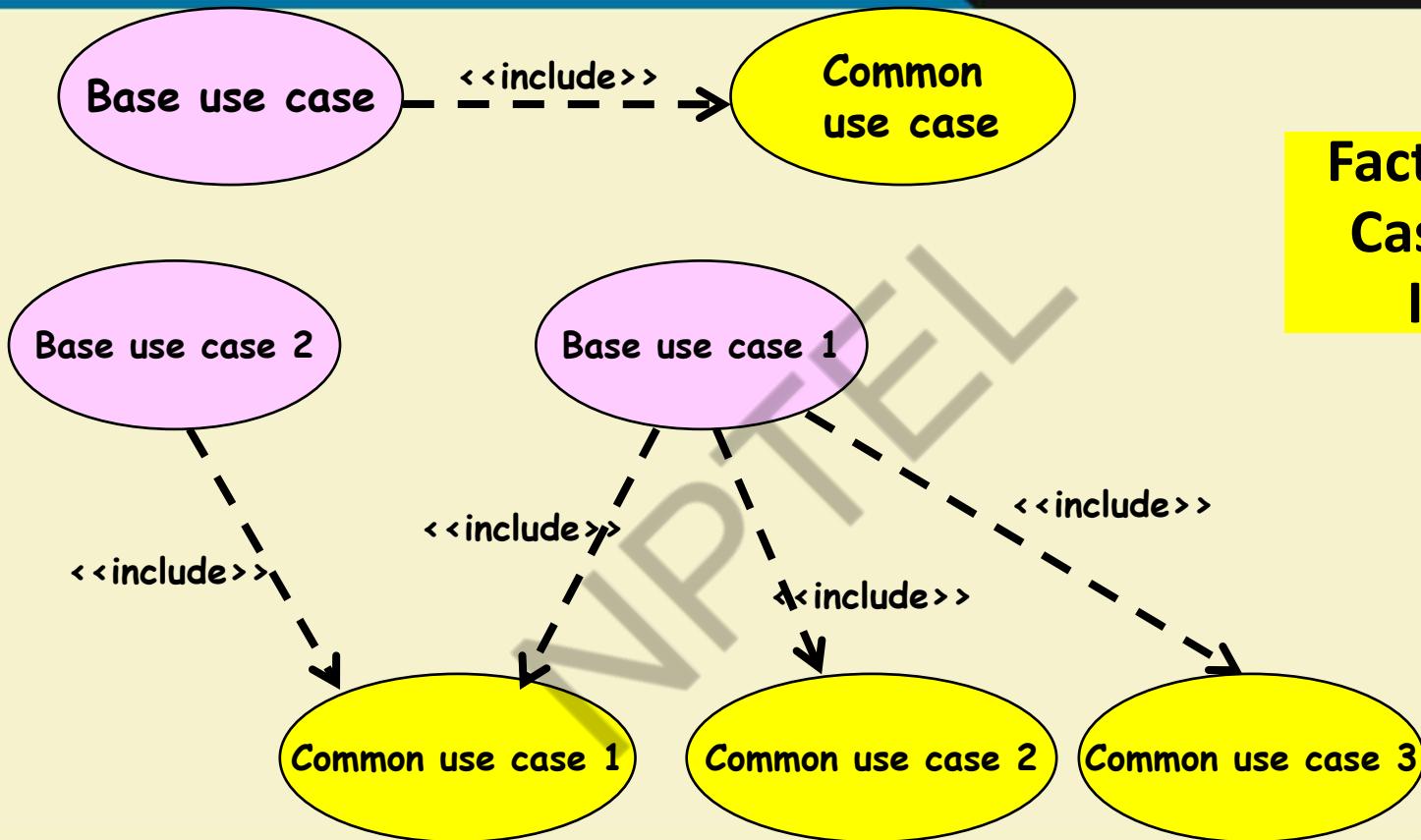


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

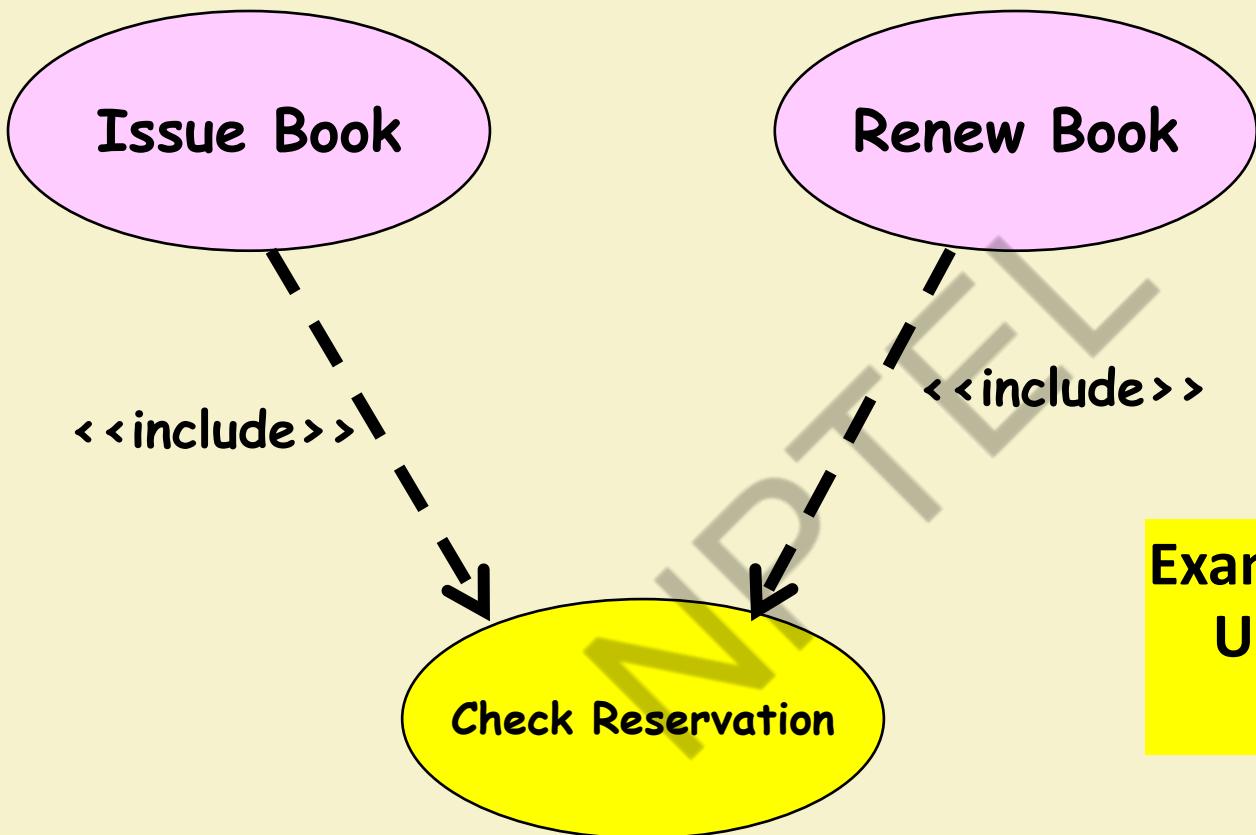
## Factoring Use Cases Using Include



IIT KHARAGPUR

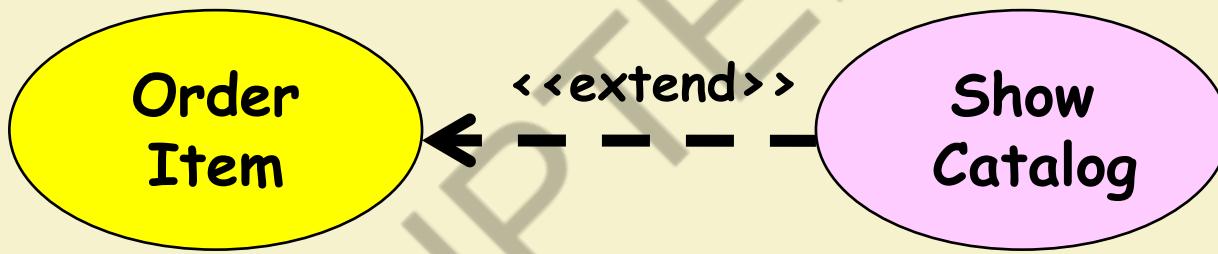


NPTEL  
ONLINE  
CERTIFICATION COURSES



**Example of Factoring  
Use Cases Using  
Include**

# Example Factoring A Use Case Using Extend



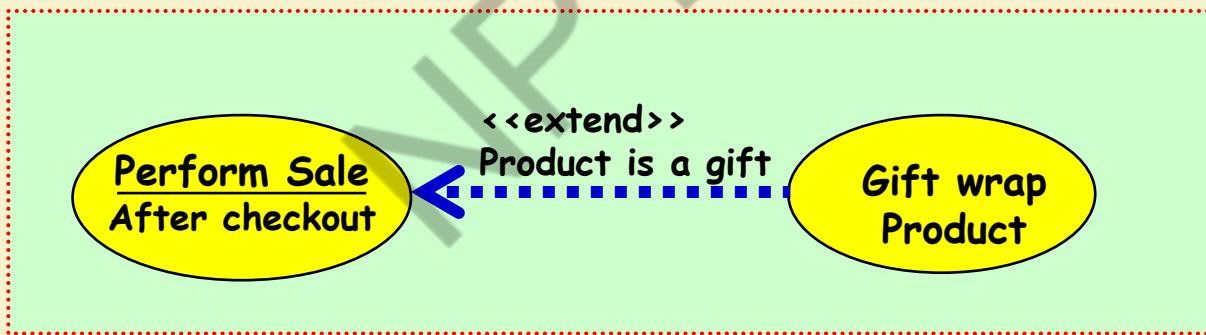
IIT KHARAGPUR

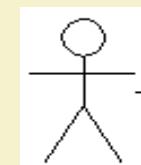


NPTEL  
ONLINE  
CERTIFICATION COURSES

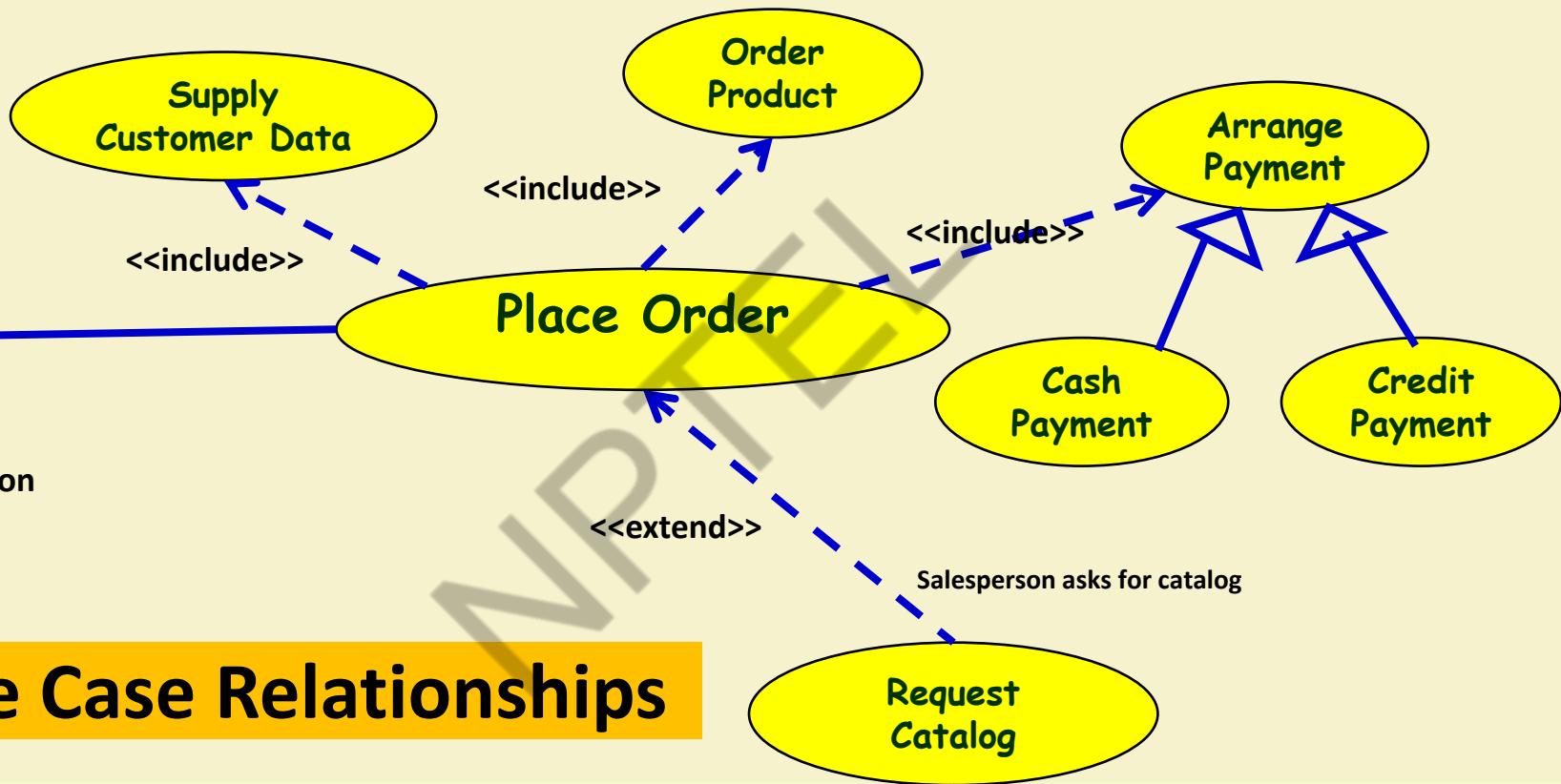
# Extension Point

- The base use case may include/extend other use cases:
  - At certain points, called extension points.
- Note the direction of the arrow





Sales Person



## Use Case Relationships

- Video Store Information System supports the following business functions:

### Example 1: Video Store Information System

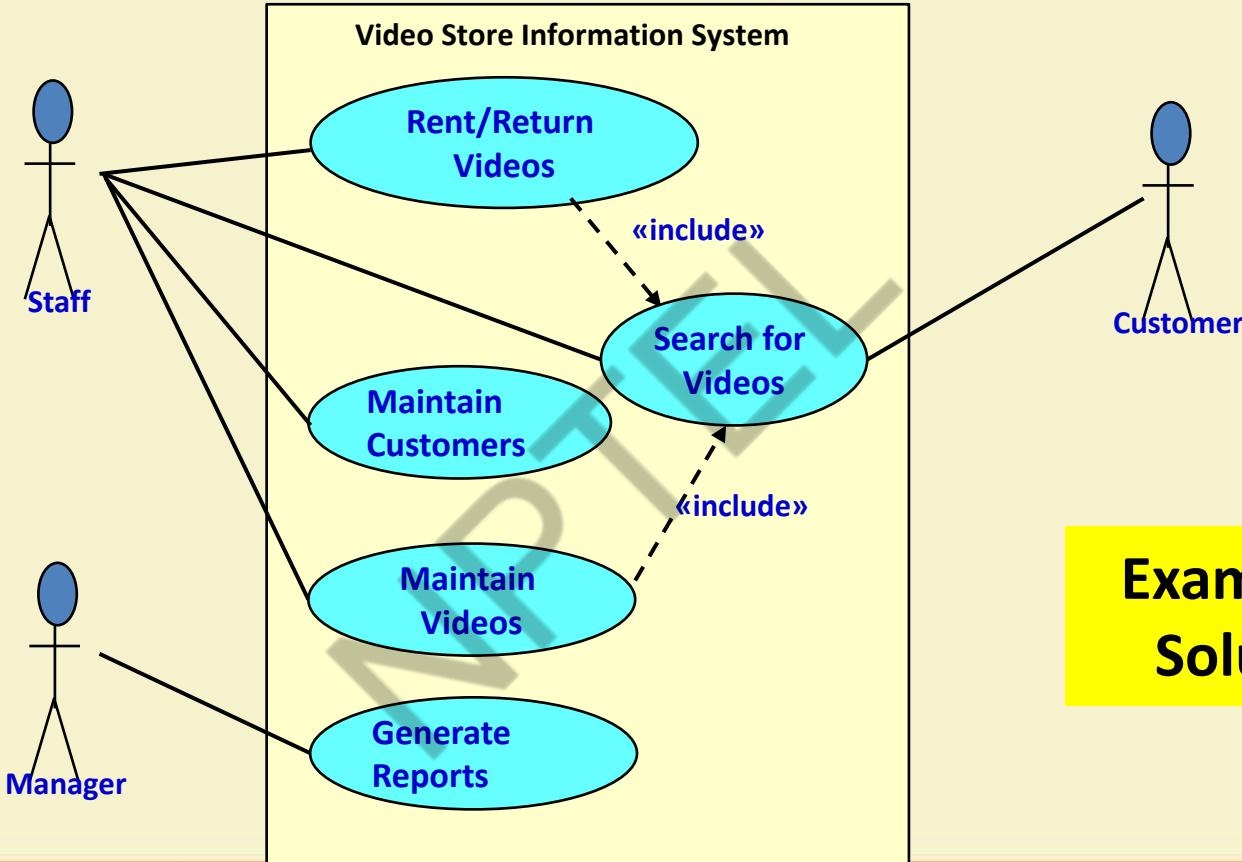
- Recording information about videos the store owns
  - This database is searchable by staff and all customers
- Information about a customer's borrowed videos
  - Access by staff and customer. It involves video database searching.
- Staff can record video rentals and returns by customers. It involves video database searching.
- Staff can maintain customer and video information.
- Managers of the store can generate various reports.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



## Example 1: Solution



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

Name

Actors

Trigger

Preconditions

Post conditions

Mainline Scenario

Alternatives flows

## Use Case Description

Alistair Cockburn  
“Writing Effective Use Cases”

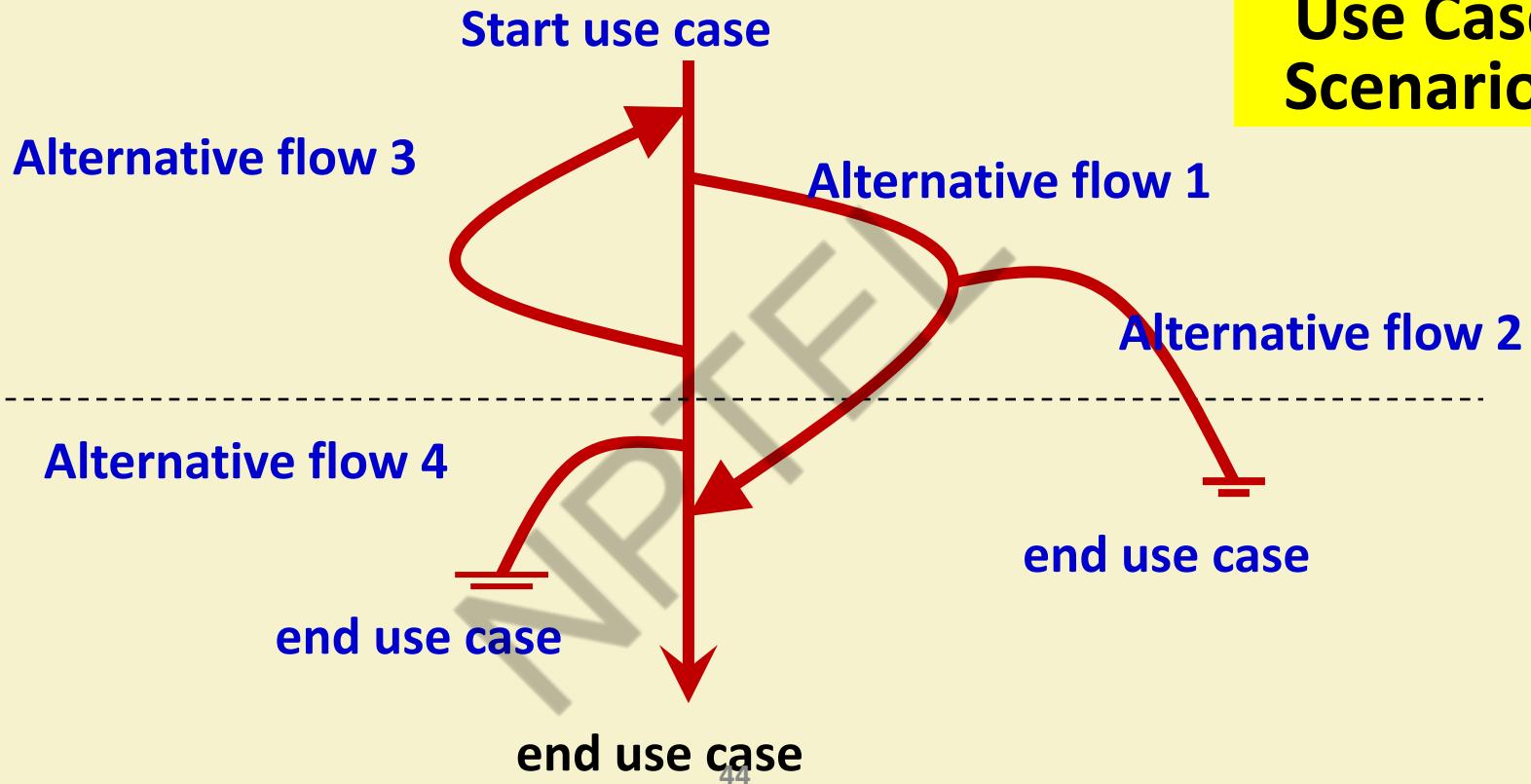


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Use Case Scenarios



44



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# ATM Money Withdraw Example

- **Actors:** Customer
- **Pre Condition:**
  - ATM must be in a state ready to accept transactions
  - ATM must have at least some cash it can dispense
  - ATM must have enough paper to print a receipt
- **Post Condition:**
  - The current amount of cash in the user account is the amount before withdraw minus withdraw amount
  - A receipt was printed on the withdraw amount



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## ATM Money Withdraw Mainline Scenario

Actor Actions	System Actions
1. Begins when a Customer arrives at ATM	
2. Customer inserts a Credit card into ATM	3. System verifies the customer ID and status
5. Customer chooses "Withdraw" operation	4. System asks for an operation type
7. Customer enters the cash amount	6. System asks for the withdraw amount
	8. System checks if withdraw amount is legal
	9. System dispenses the cash
	10. System deduces the withdraw amount from account
	11. System prints a receipt
13. Customer takes the cash and the receipt	12. System ejects the cash card

## ATM Money Withdraw (cont.)

- **Alternative flow of events:**
  - **Step 3:** Customer authorization failed. Display an error message, cancel the transaction and eject the card.
  - **Step 8:** Customer has insufficient funds in its account. Display an error message, and go to step 6.
  - **Step 8:** Customer exceeds its legal amount. Display an error message, and go to step 6.
- **Exceptional flow of events:**
  - Power failure in the process of the transaction before step 9, cancel the transaction and eject the card.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Use Case Description: Change Flight

- **Actors:** traveler

- **Preconditions:**

- - Traveler has logged on to the system and selected 'change flight itinerary' option

- **Basic course**

- 1. System retrieves traveler's account and flight itinerary from client account database
  2. System asks traveler to select itinerary segment she wants to change; traveler selects itinerary segment.
  3. System asks traveler for new departure and destination information; traveler provides information.
  4. If flights are available then
  5. ...
  6. System displays transaction summary.

- **Alternative courses**

- 4. If no flights are available then ...



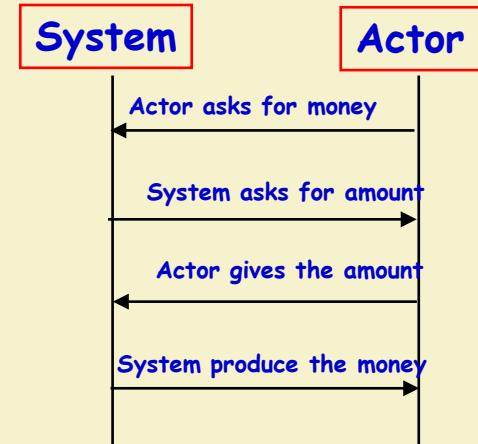
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Guidelines for Effective Use Case Writing

- Use simple sentence
- Do not have both system and actor doing something in a single step
  - Bad: “Get the amount from the user and give him the receipt.”
- Any step should lead to some tangible progress:
  - Bad: “User clicks a key”



# Identification of Use Cases

## 1. Actor-based:

- Identify the actors related to a system or organization.
- For each actor, identify the processes they initiate or participate in.

## 2. Event-based

- Identify the external events that the system must respond to.
- Relate the events to actors and use cases.



IIT KHARAGPUR

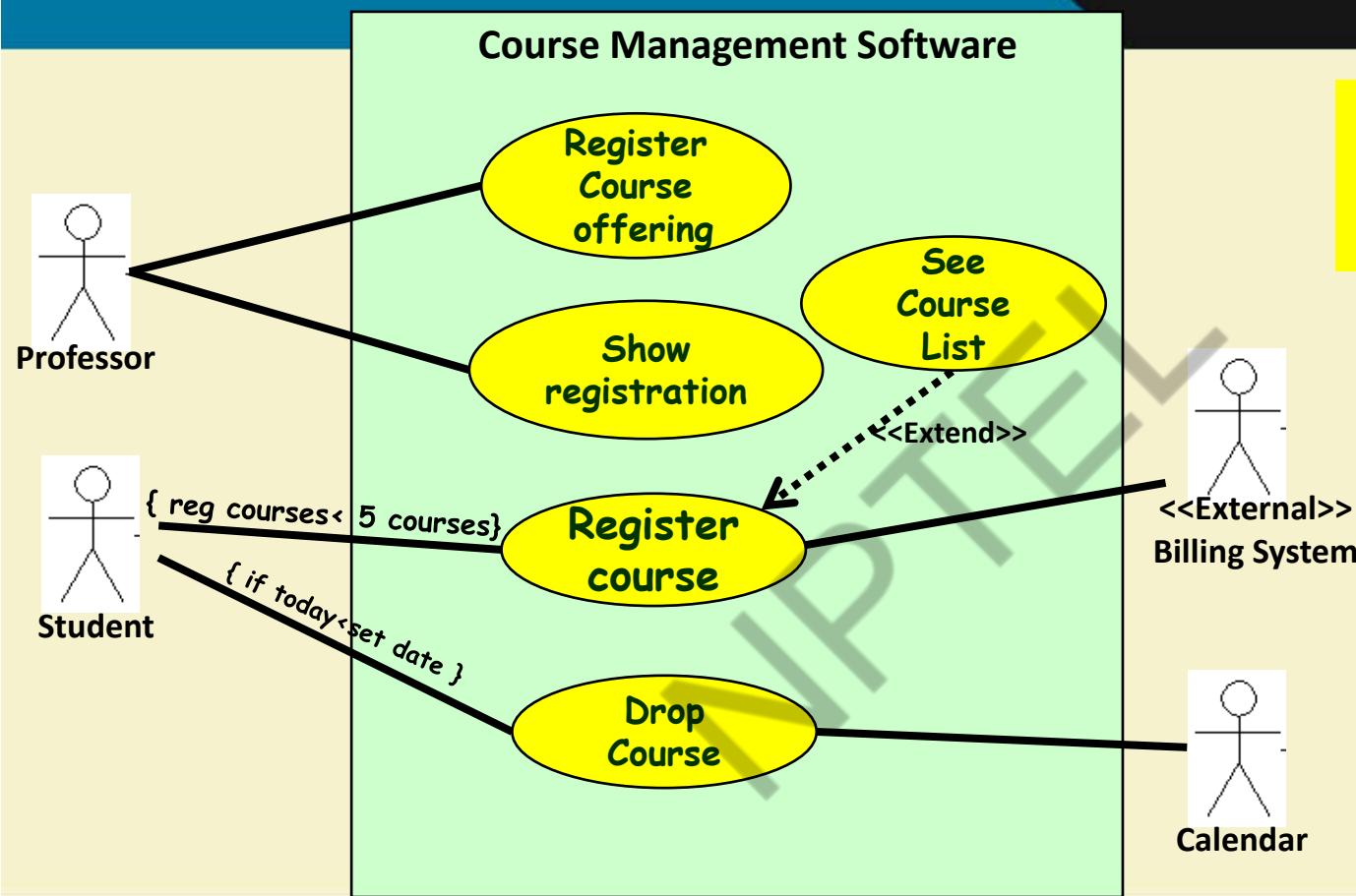


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example 2: Use Case Model for Course Management Software

- At the beginning of each semester,
  - Each professor shall register the courses that he is going to teach.
- A student can select up to four-course offerings.
  - **During registration a student can request a course catalogue showing course offerings for the semester.**
  - **Information about each course such as professor, department and prerequisites would be displayed.**
  - **The registration system sends information to the billing system, so that the students can be billed for the semester.**
- For each semester, there is a period of time during which dropping of courses is permitted.
- Professors must be able to access the system to see which students signed up for each of their course offerings.

## Example 2: Model Solution



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Use case name should begin with a verb.
- While use cases do not explicitly imply timing:
  - Order use cases from top to bottom to imply timing -- it improves readability.

- **The primary actors should appear in the left.**

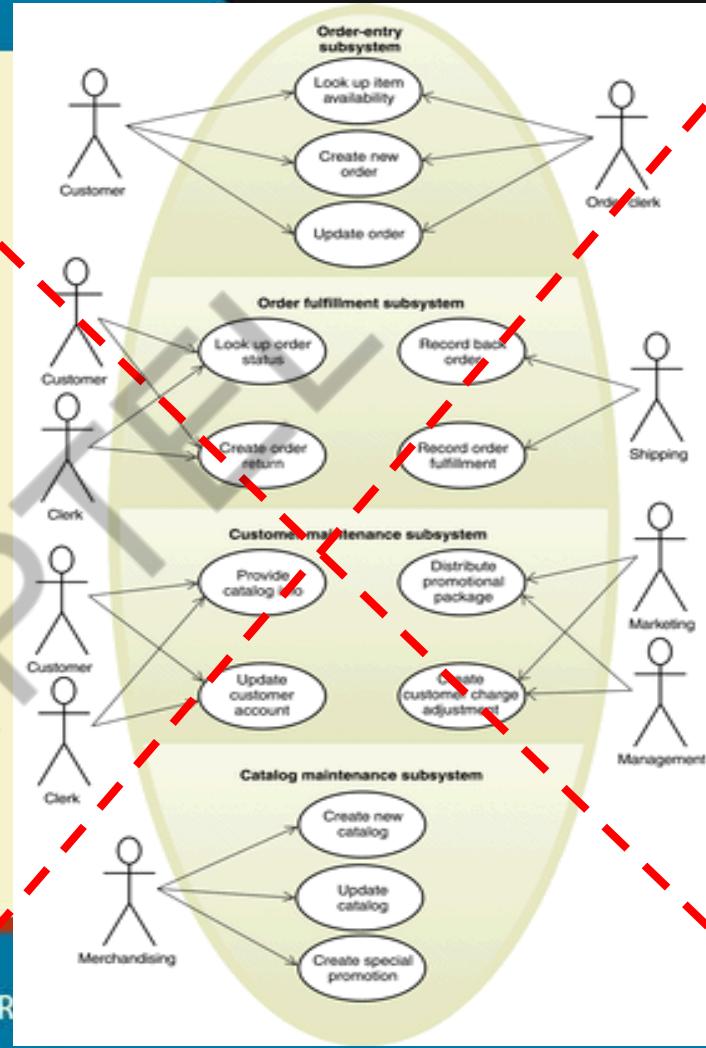
- Actors are associated with one or more use cases.
- Do not use arrows on the actor-use case relationship.
- **To initiate scheduled events include an actor called “time”, or “calendar”**
- **Do not show actors interacting with each other.**
- <<include>> should rarely nest more than 2 levels deep.

## Style Notes (Ambler, 2005)

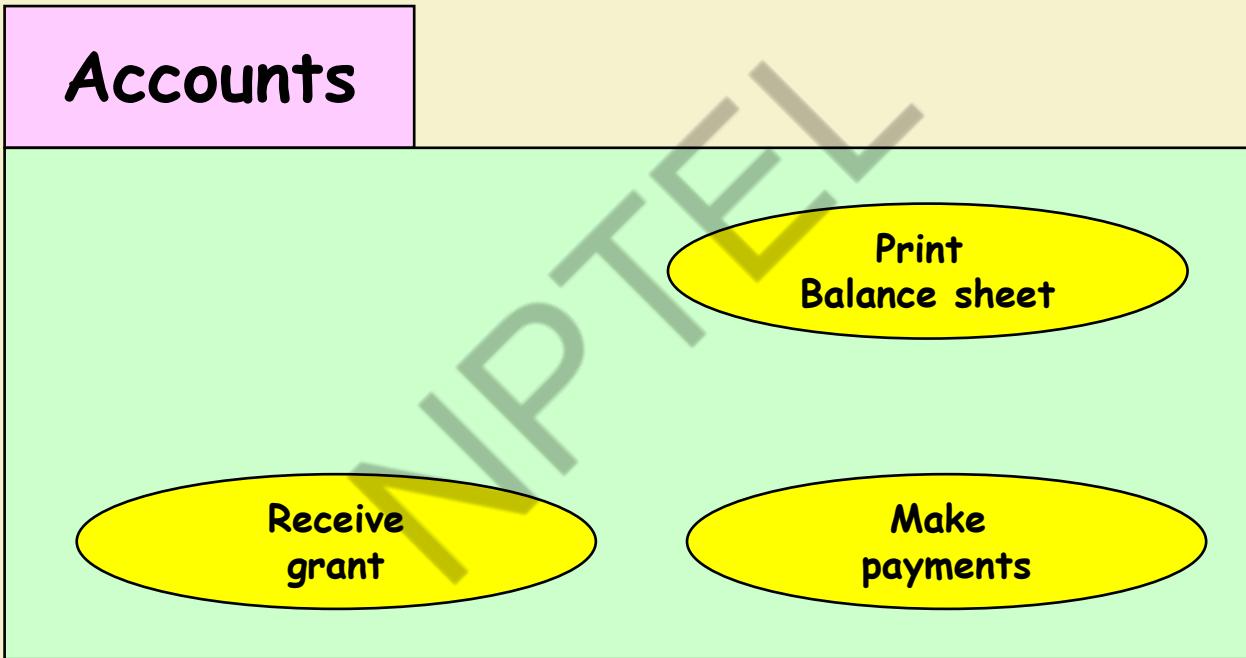
- Use cases should be named and organized from the perspective of the users.
- Use cases should start off simple and at as much high view as possible.
  - Can be refined and detailed further.
- Use case diagrams represent functionality:
  - **Should focus on the "what" and not the "how".**

**Effective Use Case  
Modelling**

Too many use cases  
at any level should  
be avoided!



# Use Case Packaging

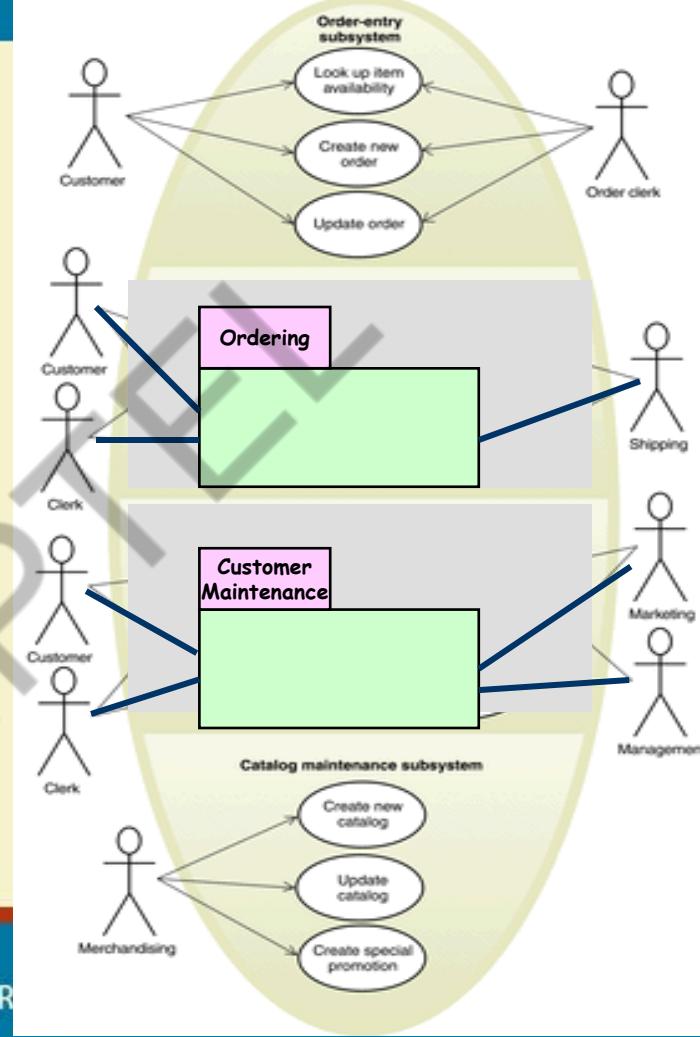


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

More acceptable!



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSE

# Class Diagram

- Classes:
  - Entities with common features, i.e. attributes and operations.
  - Represented as solid outline rectangle with compartments.
  - Compartments for **name, attributes, and operations.**
  - Attribute and operation compartments are optional depending on the purpose of a diagram.



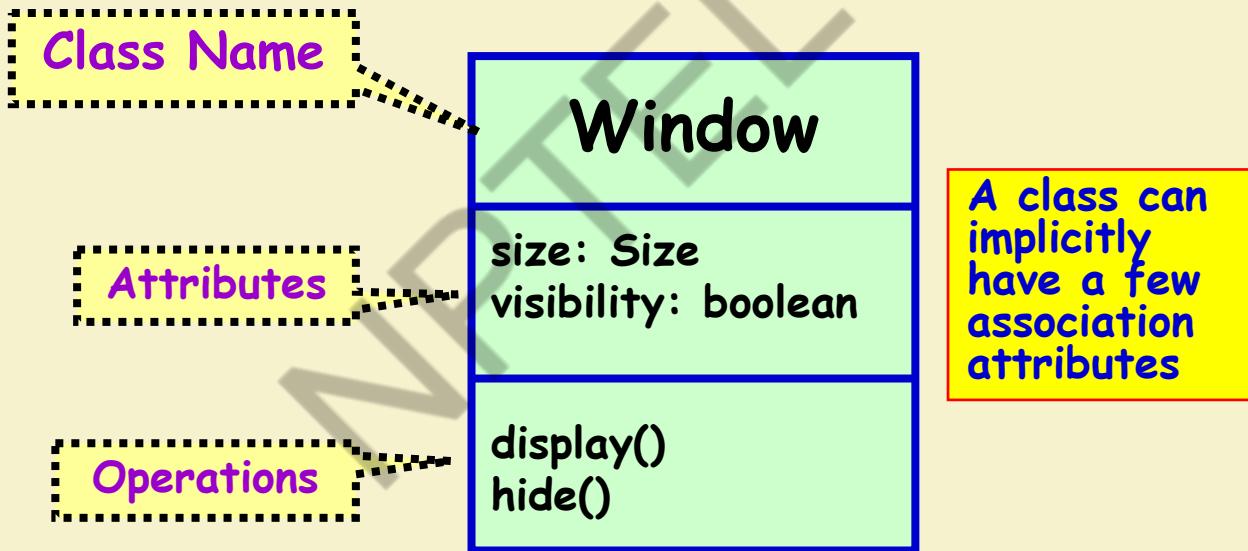
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# UML Class Representation

- A class represents a set of objects having similar attributes, operations, relationships and behavior.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Different representations of the LibraryMember class

LibraryMember

Member Name  
Membership Number  
Address  
Phone Number  
E-Mail Address  
Membership Admission Date  
Membership Expiry Date  
Books Issued

issueBook( );  
findPendingBooks( );  
findOverdueBooks( );  
returnBook( );  
findMembershipDetails( );

LibraryMember

issueBook( );  
findPendingBooks( );  
findOverdueBooks( );  
returnBook( );  
findMembershipDetails( );

LibraryMember

Example UML  
Classes



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What are the Different Types of Relationships Among Classes?

- Four types of relationships:
  - Inheritance
  - Association
  - Aggregation/Composition
  - Dependency



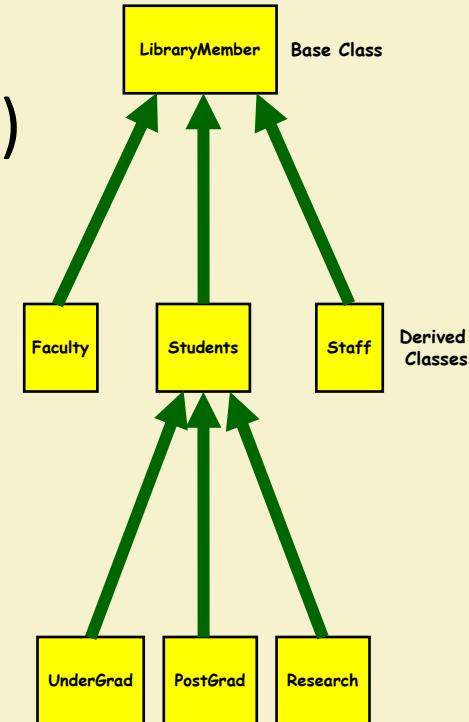
IIT KHARAGPUR



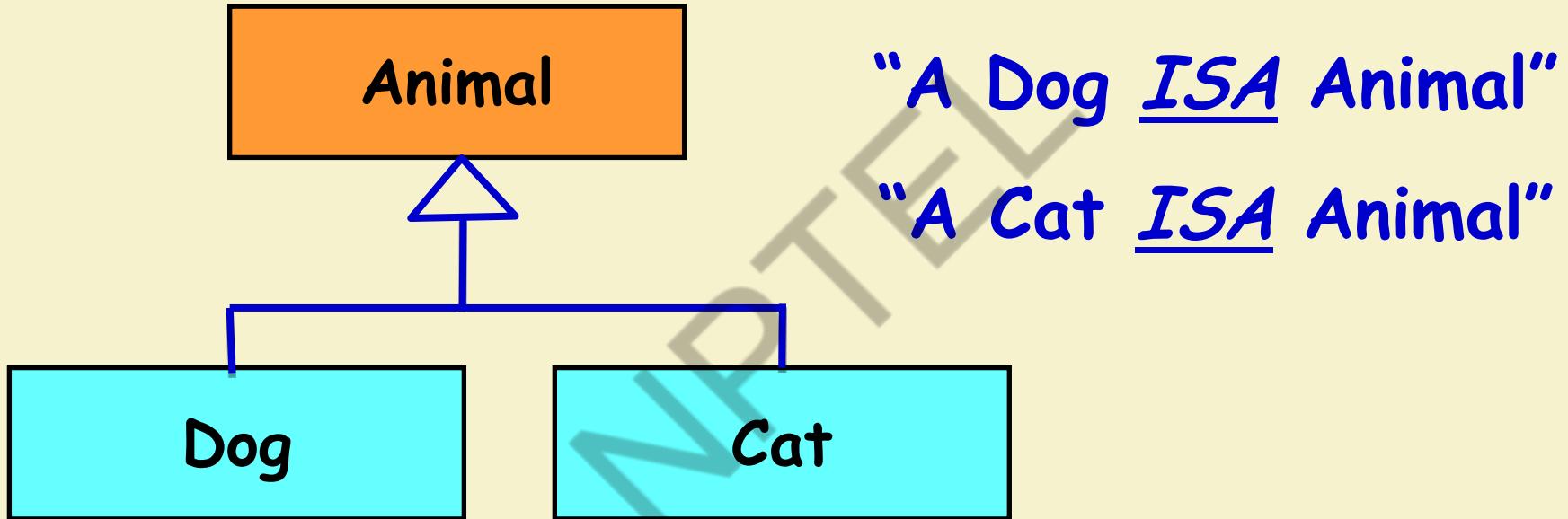
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Inheritance

- Allows to define a new class (derived class) by extending an existing class (base class).
  - Represents generalization-specialization
  - Allows redefinition of the existing methods (method overriding).



# Inheritance Example



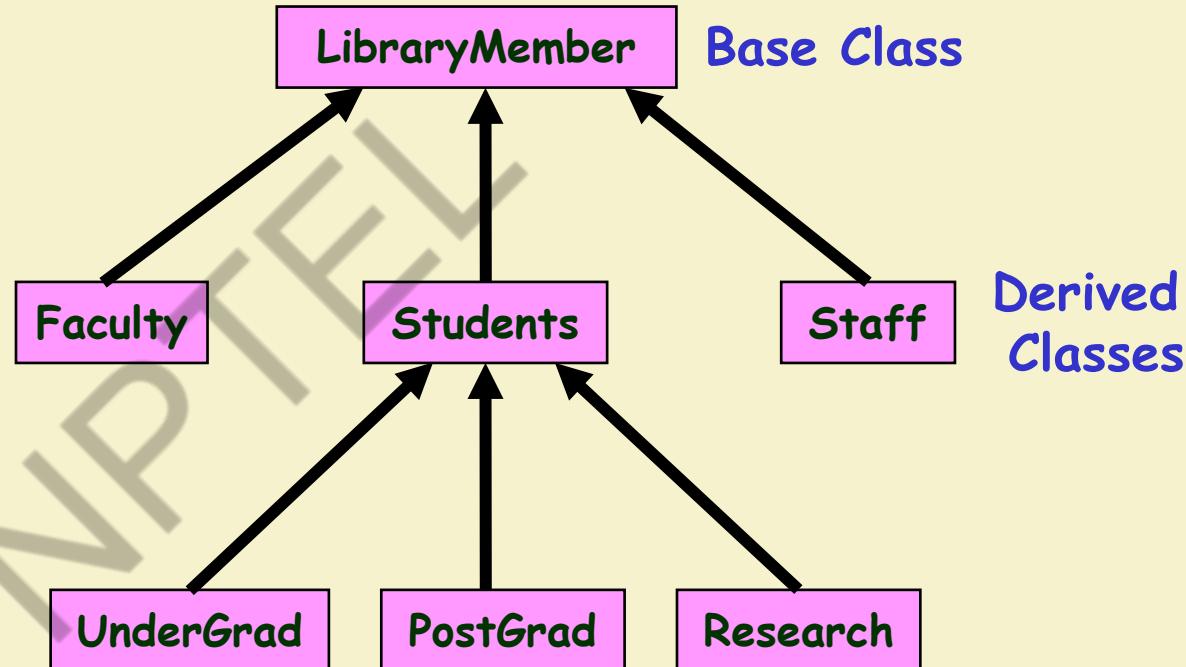
IIT KHARAGPUR



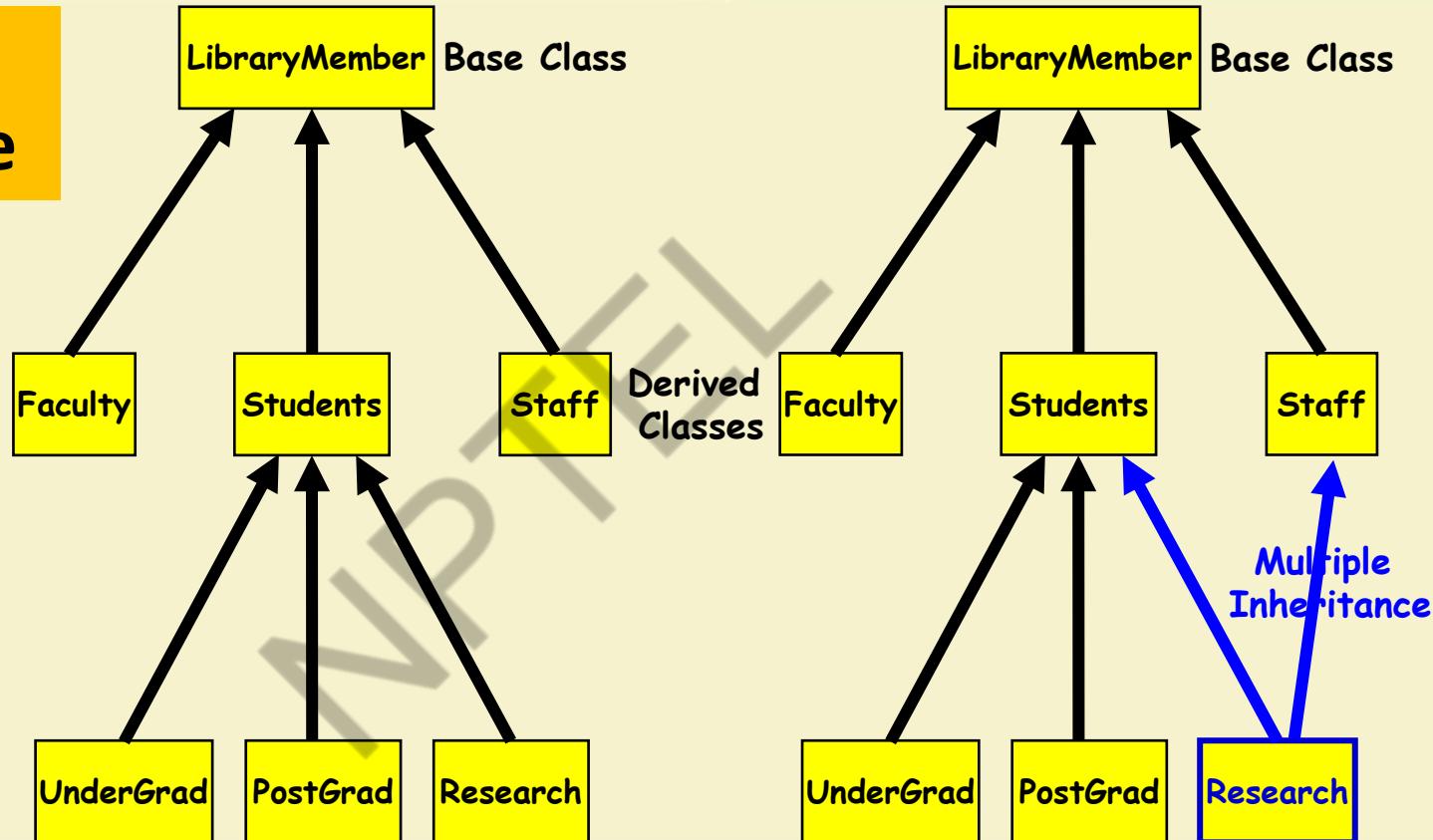
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Inheritance

- Lets a subclass inherit attributes and methods from a base class.



# Multiple Inheritance



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

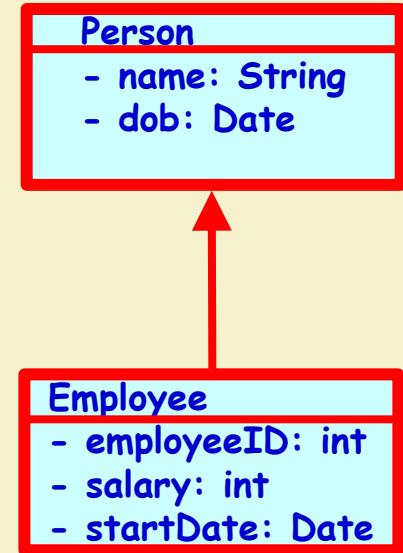
# Inheritance Implementation in Java

- Inheritance is declared using the "extends" keyword
  - Even when no inheritance defined, the class implicitly extends a class called Object.

```
class Person{  
    private String name;  
    private Date dob;  
    ...  
}
```

```
class Employee extends Person{  
    private int employeeID;  
    private int salary;  
    private Date startDate;  
    ...  
}
```

```
Employee anEmployee = new Employee();
```



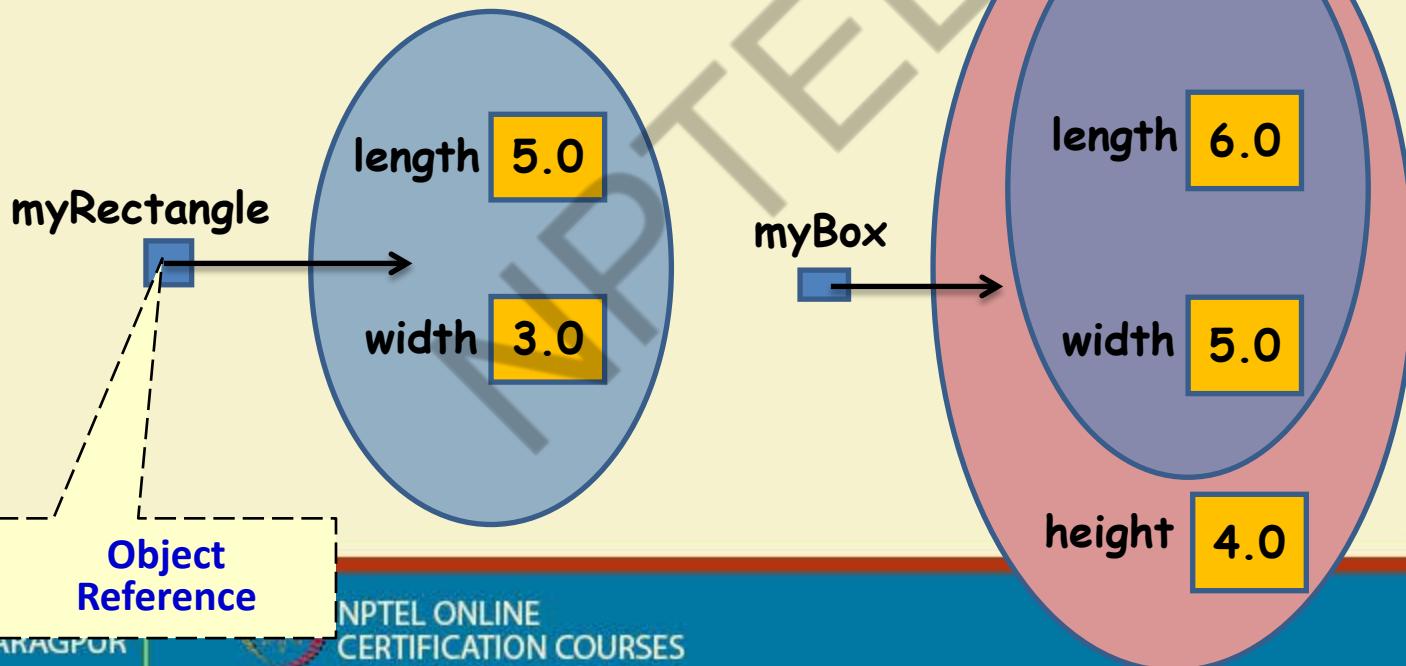
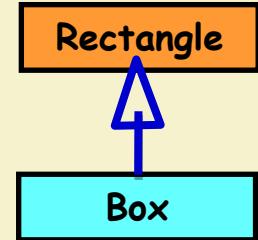
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Objects myRectangle and myBox

```
Rectangle myRectangle = new Rectangle(5, 3);  
Box myBox = new Box(6, 5, 4);
```

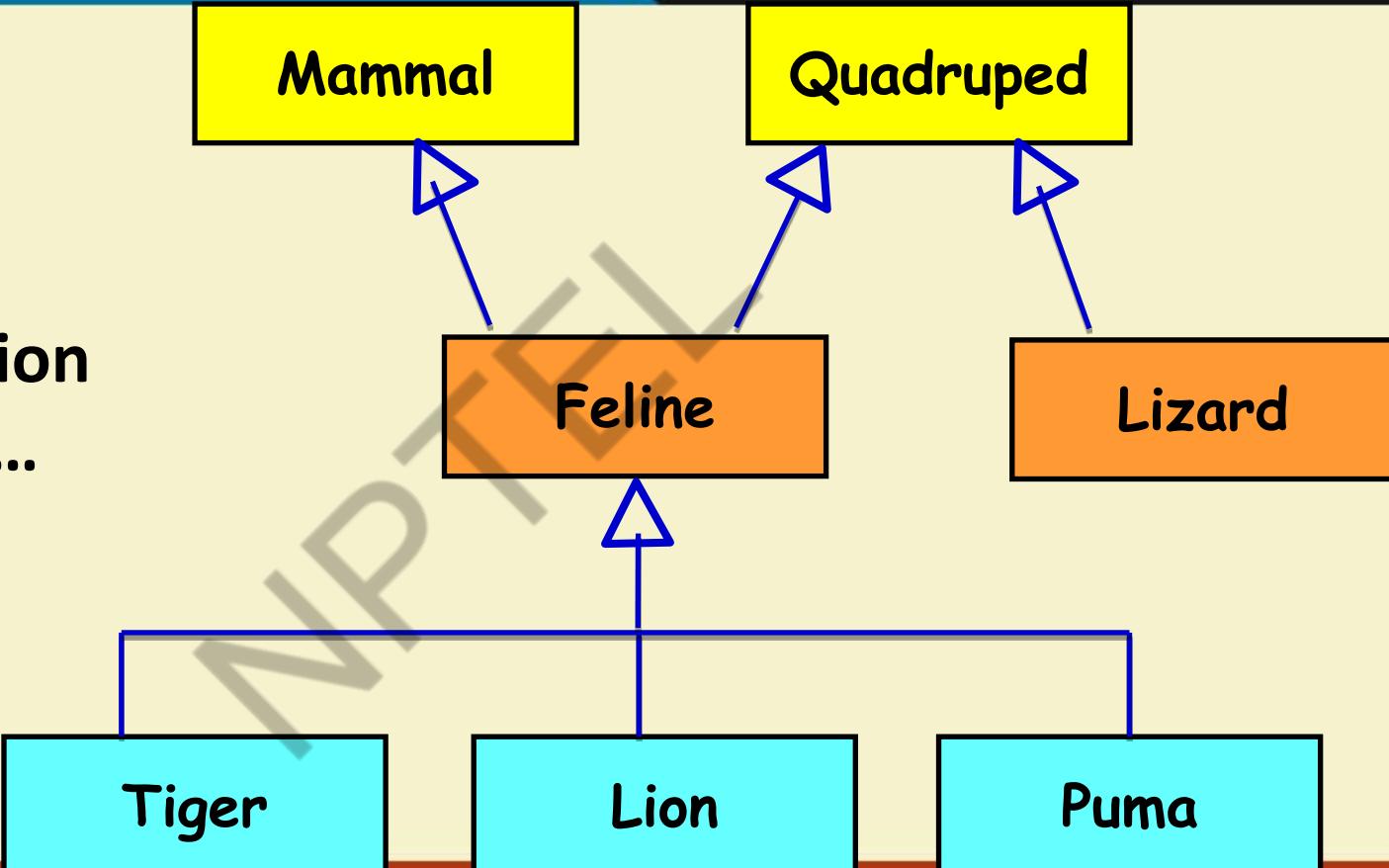


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## More Generalization Examples...



IIT KHARAGPUR



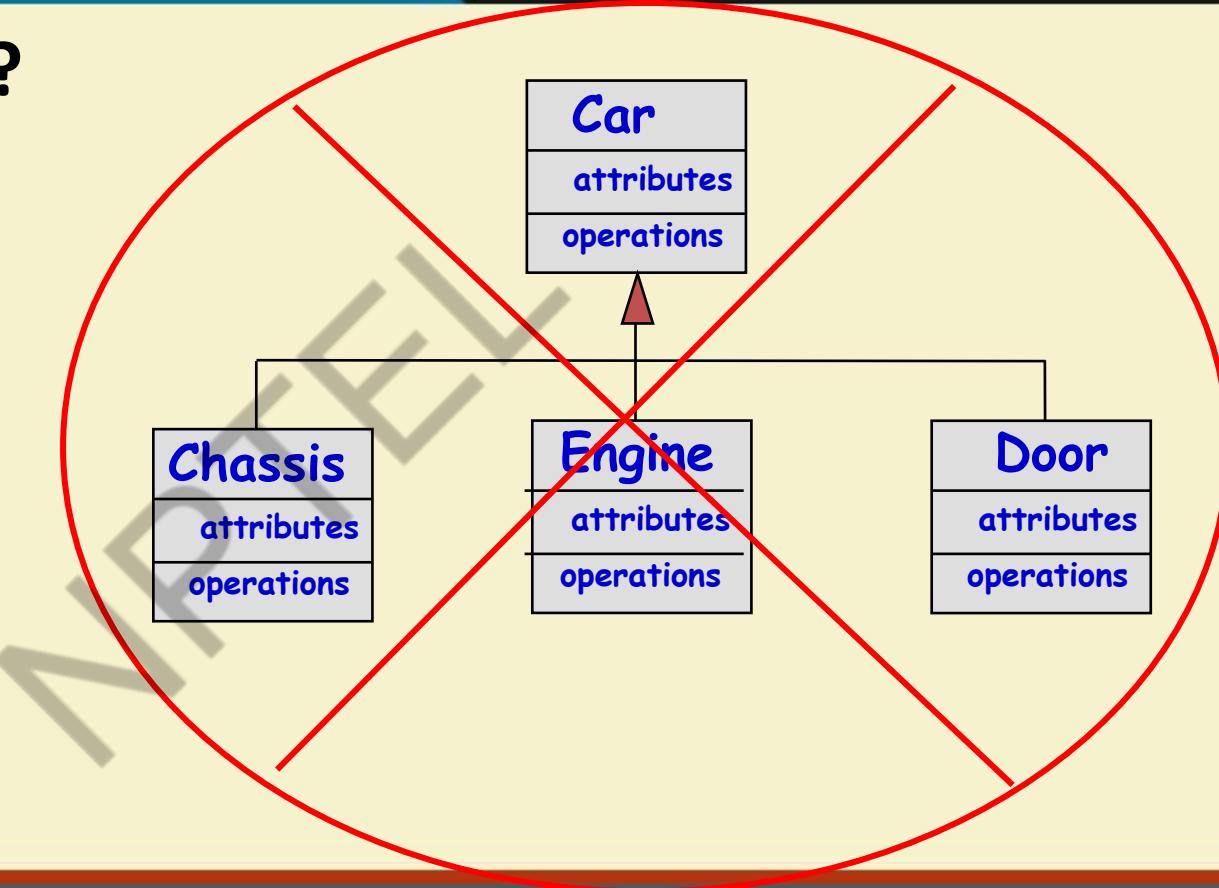
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Any problems?

Wrong  
Generalization

---

violates “is a”  
or “is a kind  
of” heuristic

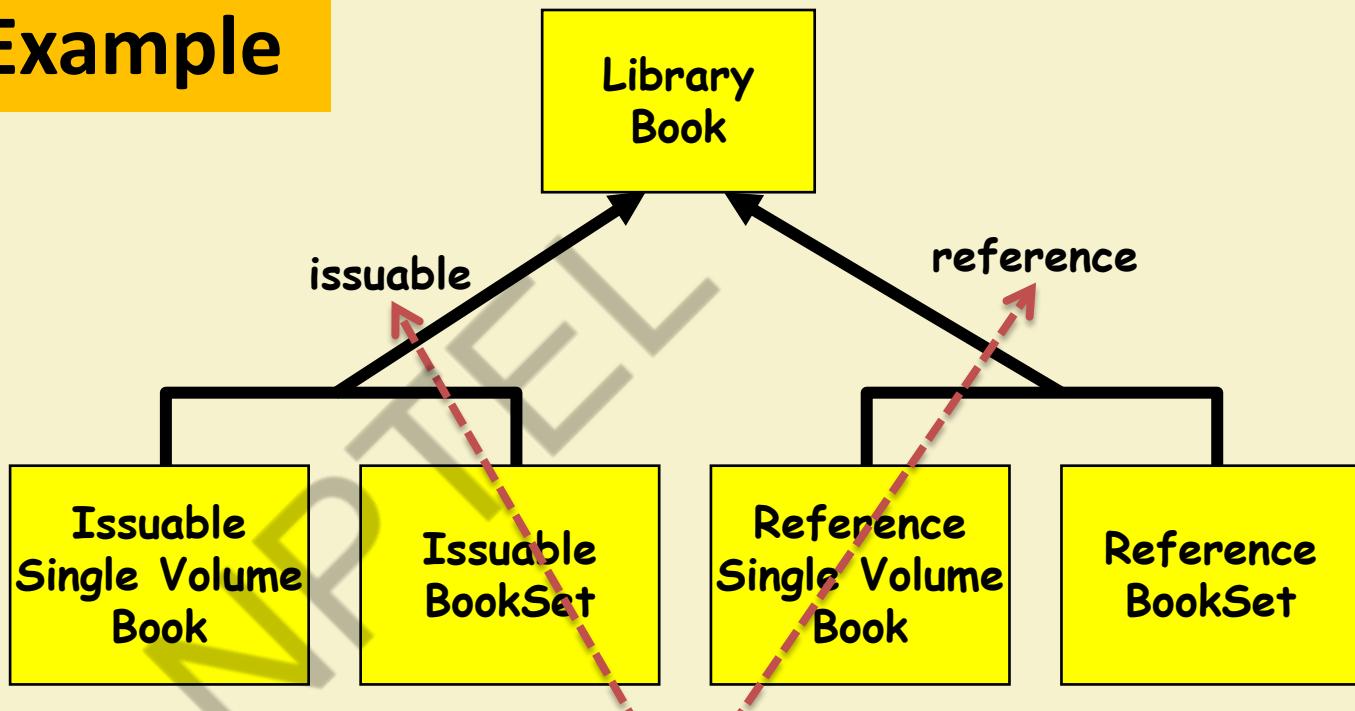


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Inheritance Example



**Discriminator:** allows one to group subclasses into clusters that correspond to a semantic category.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Inheritance Pitfalls

- Inheritance certainly promotes reuse.
- **Indiscriminate use can result in poor quality programs.**
- Base class attributes and methods visible in derived class...
  - Leads to tight coupling



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Association Relationship

- How implemented in program?
- Enables objects to communicate with each other:
  - One object must “know” the ID of the corresponding object in the association.
- Usually binary:
  - But in general can be n-ary.

# Association – example

- In a home theatre system,
  - A TV object has an association with a VCR object
    - It may receive a signal from the VCR
  - VCR may be associated with remote
    - It may receive a command to record

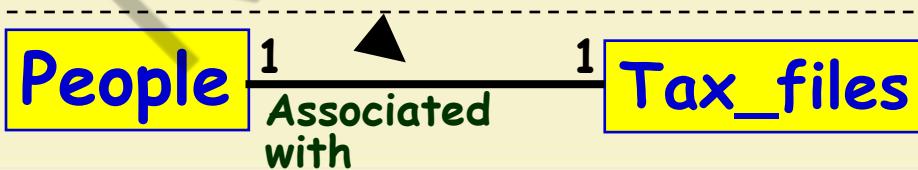
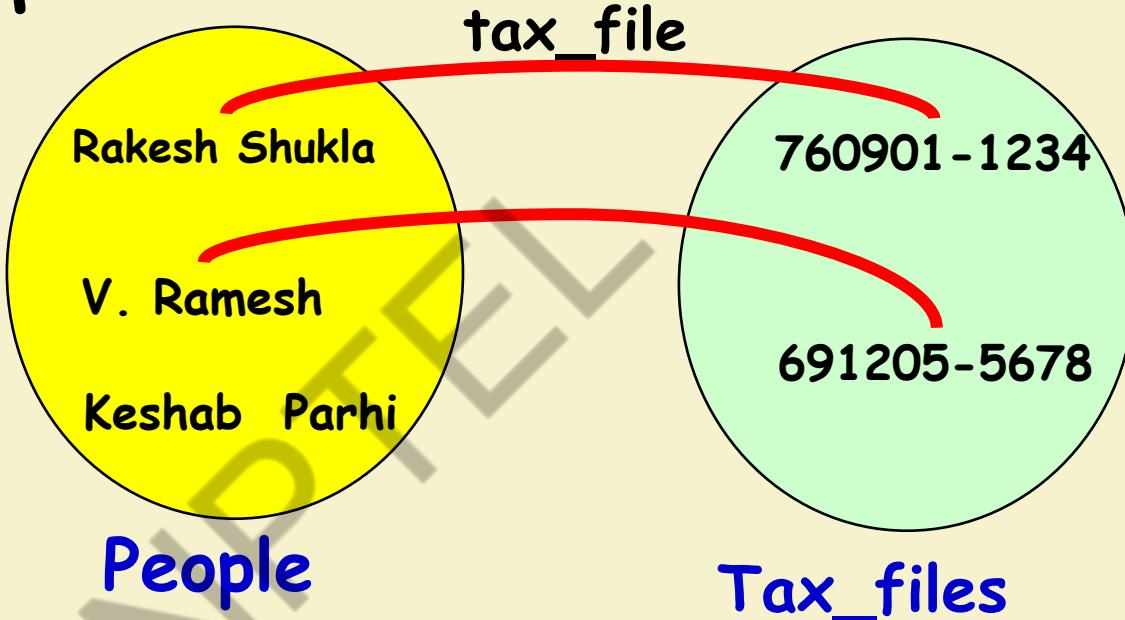


IIT KHARAGPUR

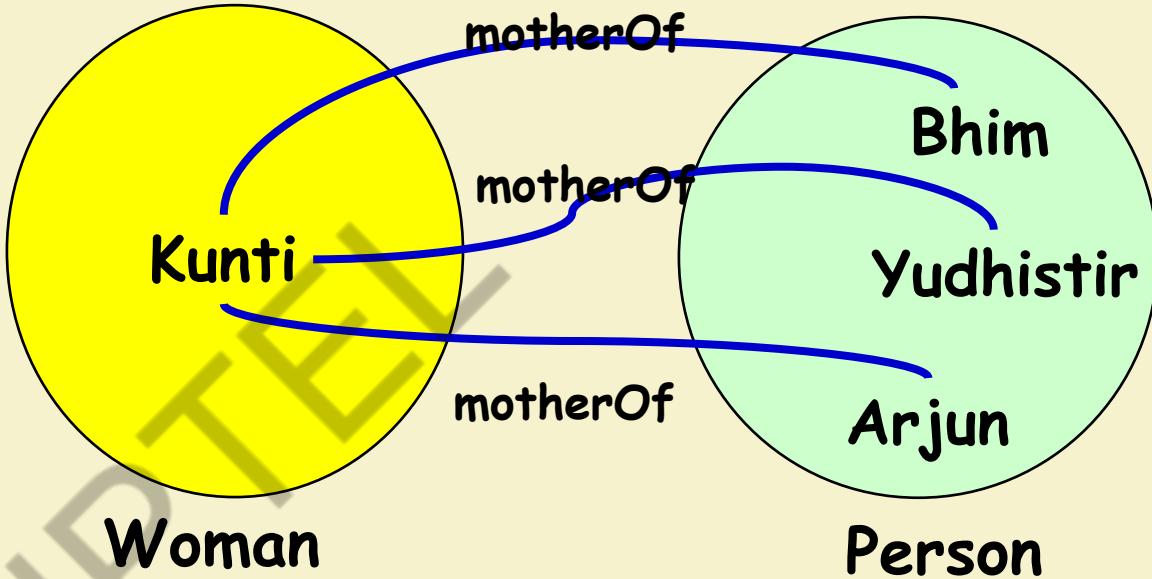


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# 1-1 Association - example



# Multiple Association - example

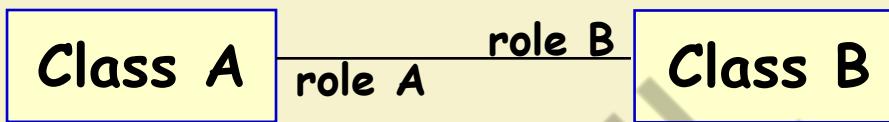


IIT KHARAGPUR

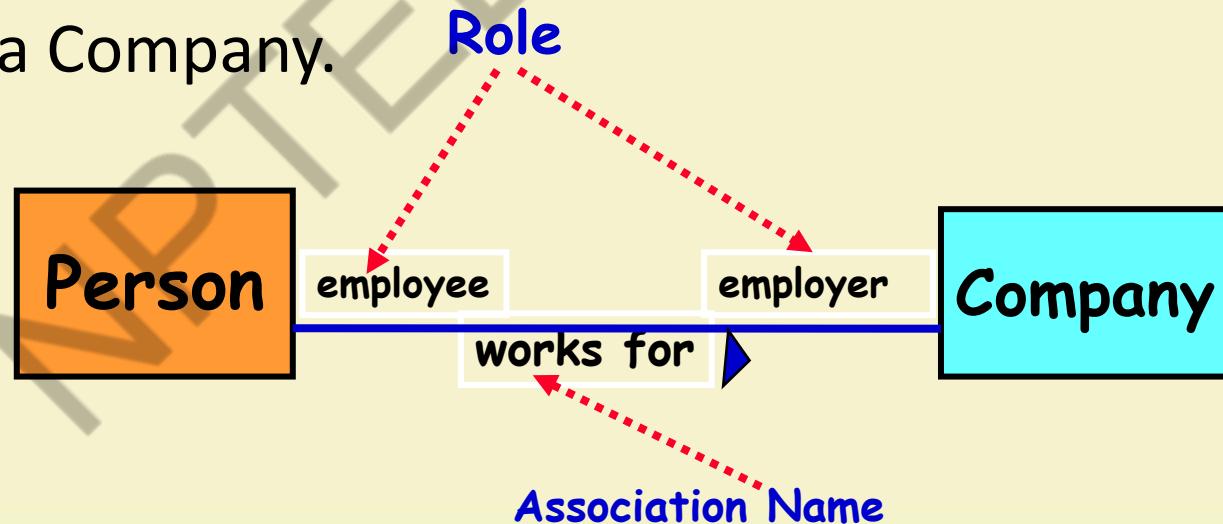


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

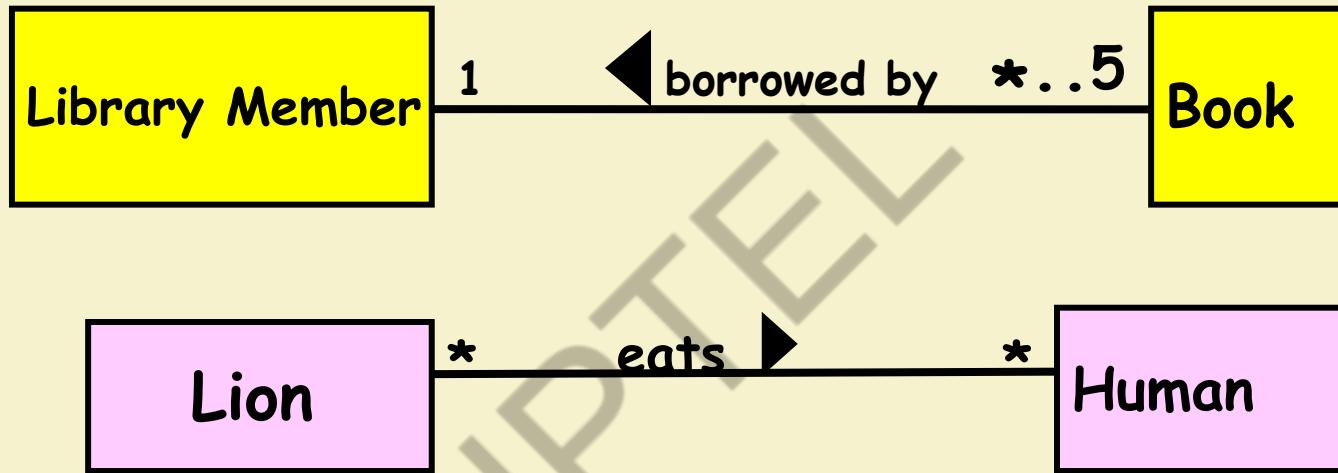
# Association UML Syntax



- A Person works for a Company.

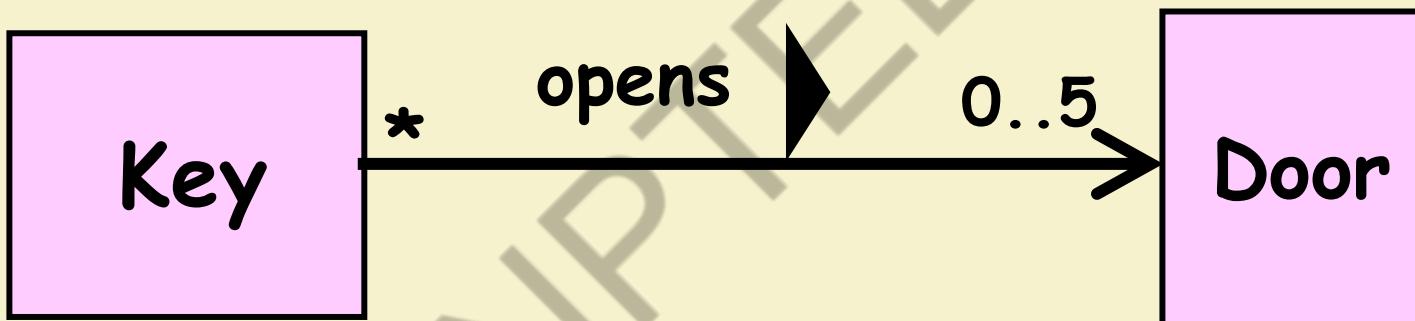


# Association - More Examples



**Multiplicity:** The number of objects from one class that relate with a single object in an associated class.

# Navigability



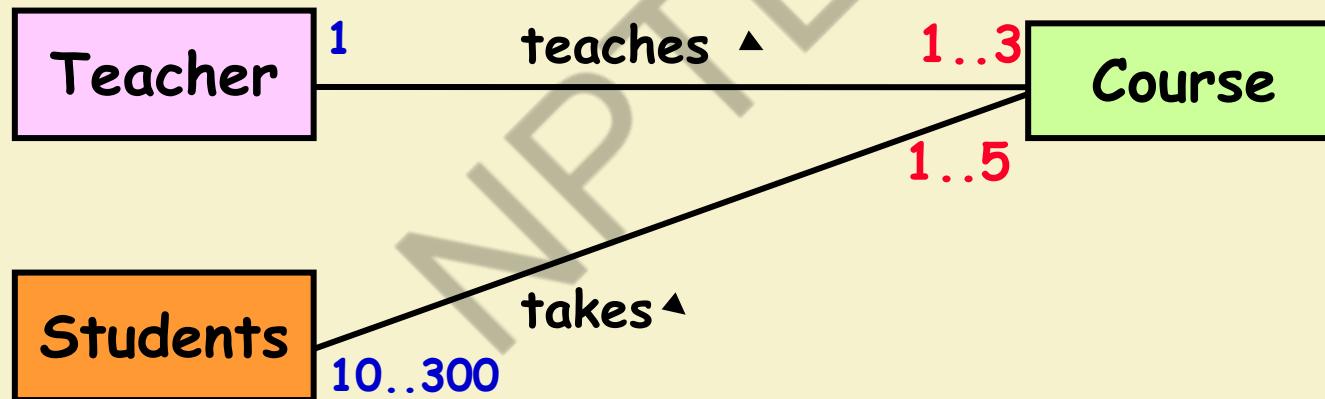
IIT KHARAGPUR



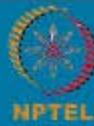
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Association - Multiplicity

- A teacher teaches 1 to 3 courses (subjects)
- Each course is taught by only one teacher.
- A student can take between 1 to 5 courses.
- A course can have 10 to 300 students.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Quiz: Draw Class Diagram

- A Student can take up to five Courses.
- A student needs to enroll in at least one course.
- Up to 300 students can enroll in a course.
- An offered subject in a semester should have at least 10 registered students.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

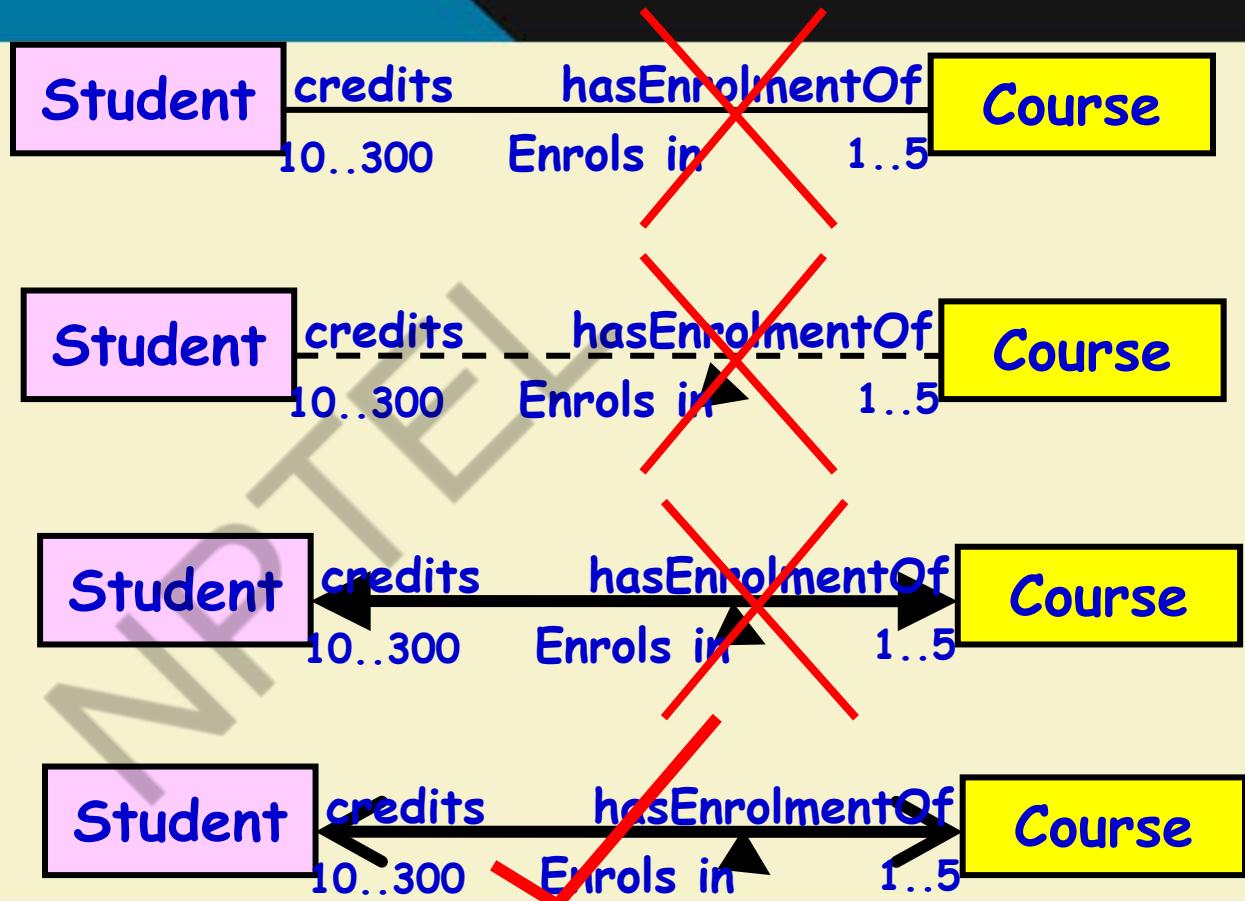


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Identify as Correct or Wrong

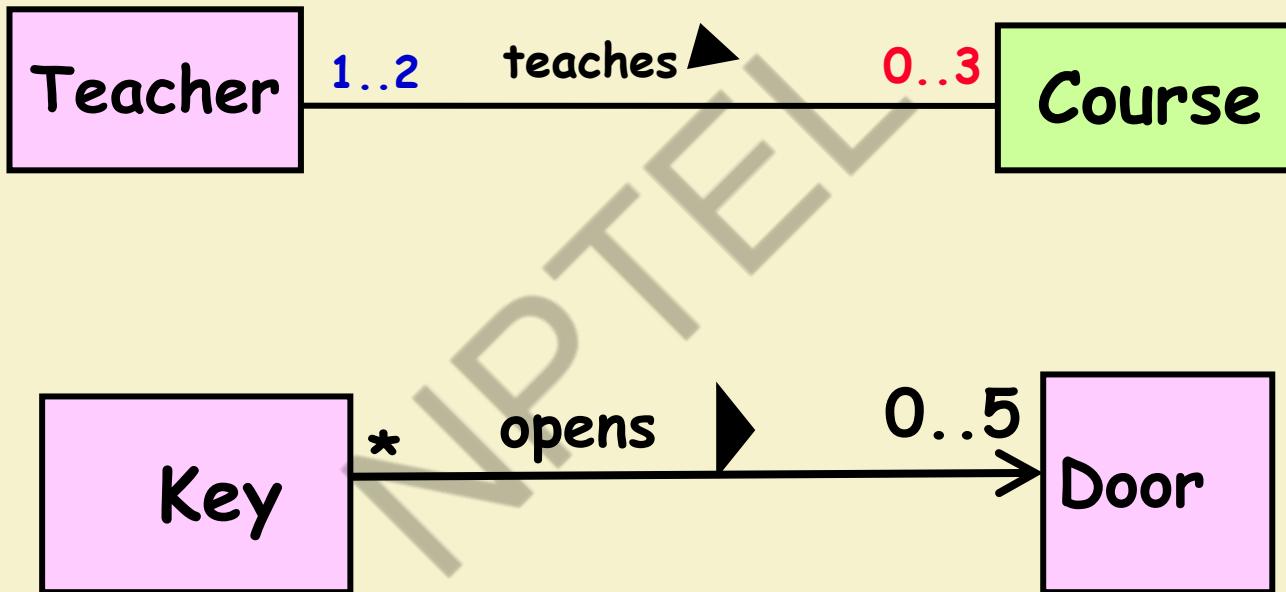


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Quiz: Read the Diagram



IIT KHARAGPUR



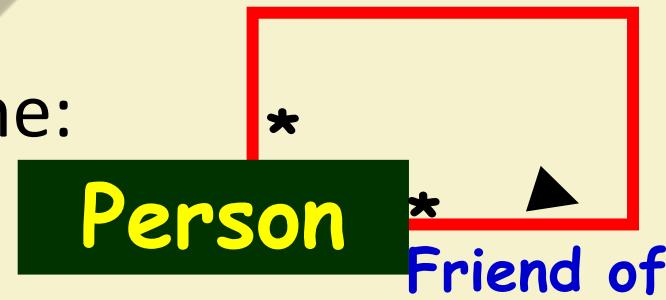
NPTEL  
ONLINE  
CERTIFICATION COURSES

# Association and Link

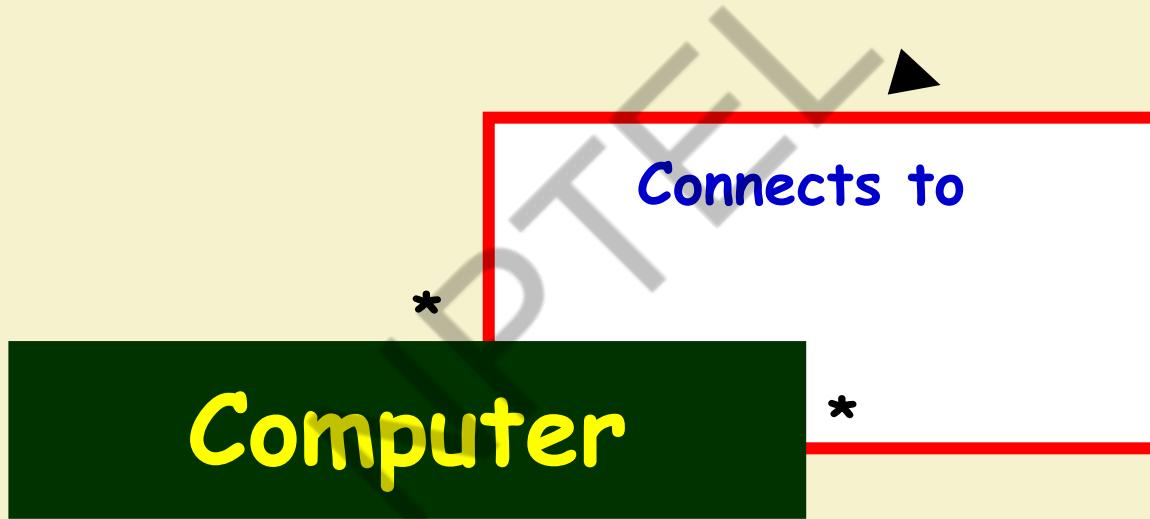
- A link:
  - An instance of an association
  - Exists between two or more objects
  - **Dynamically created and destroyed as the run of a system proceeds**
- For example:
  - An employee joins an organization.
  - Leaves that organization and joins a new organization.

# Association Relationship

- A class can be associated with itself (**unary association**).
  - Give an example?
- An arrowhead used along with name:
  - Indicates direction of association.
- Multiplicity indicates # of instances taking part in the association.



# Self Association: Example of Computer Network

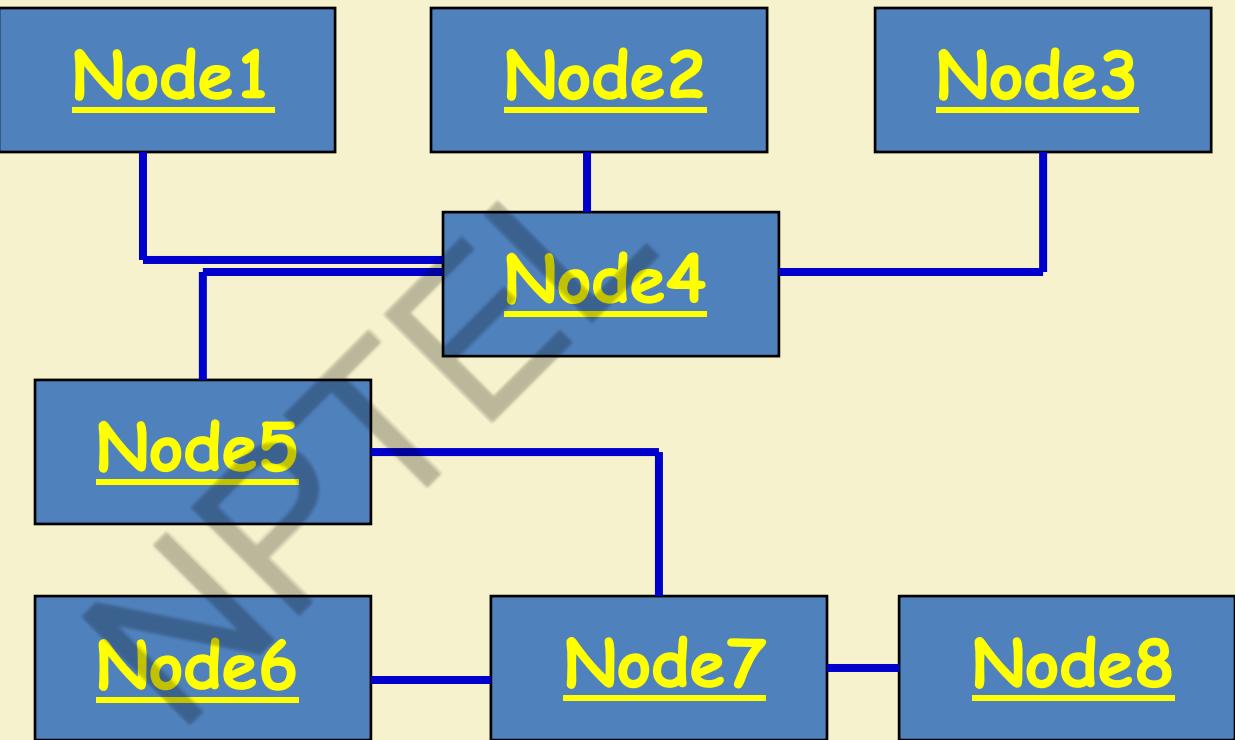


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Computer Network: Object Diagram



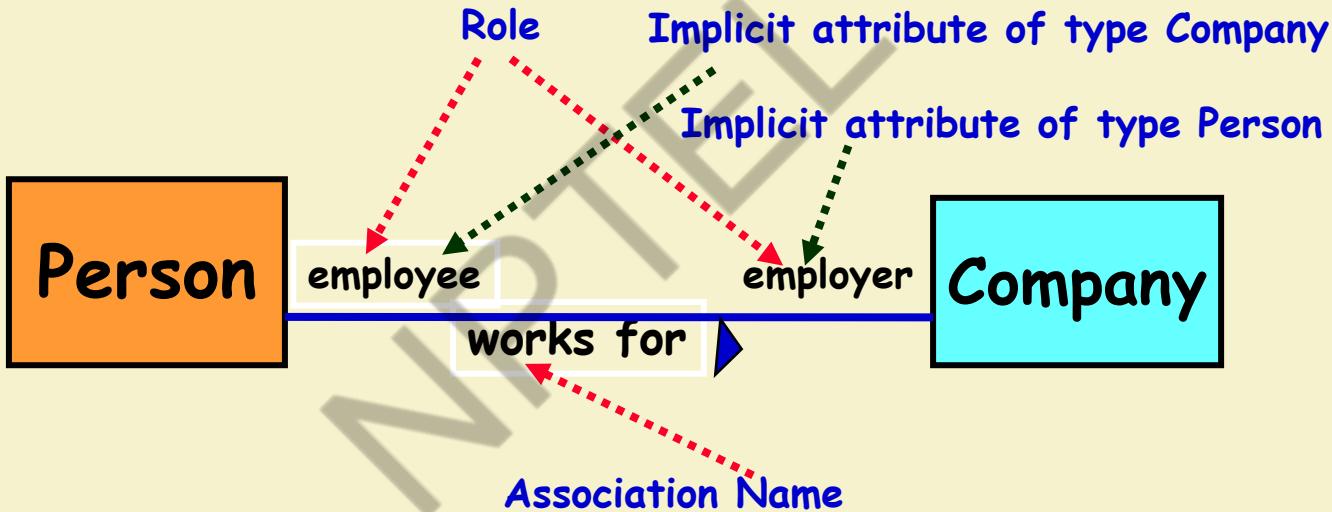
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

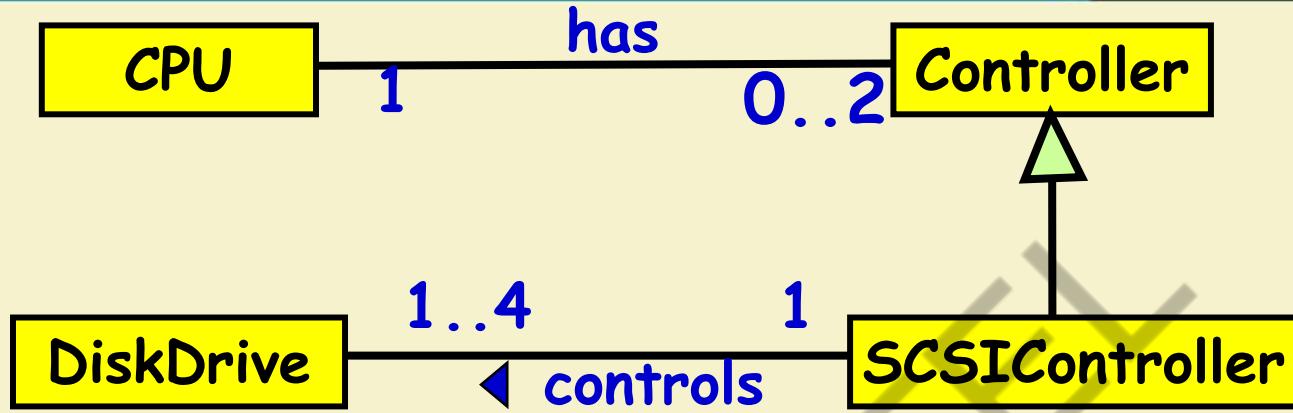
# Association Exercise 1

- A Person works for a Company.



Observe: Implicit bidirectional navigation

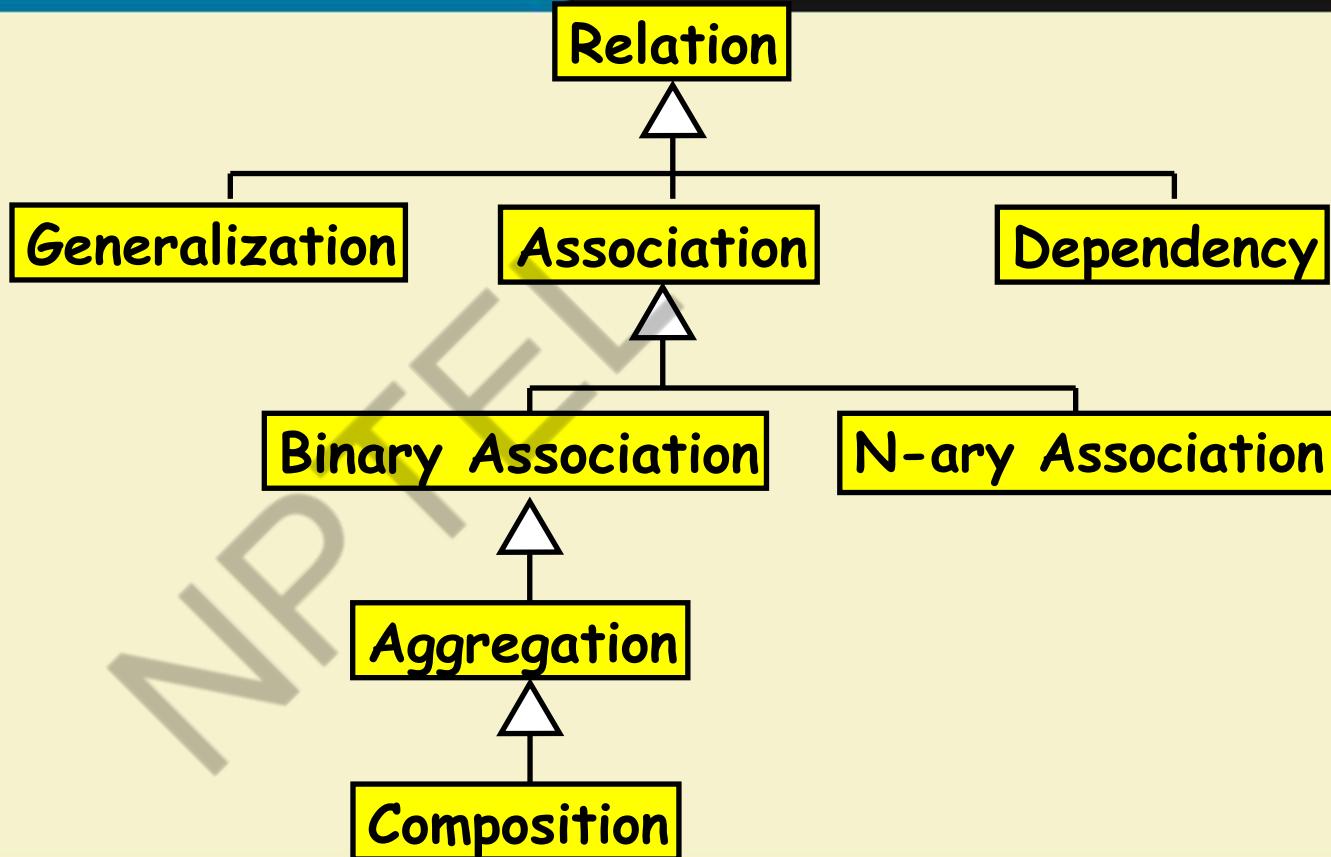
Implementation?



Quiz: Read and understand UML class diagram

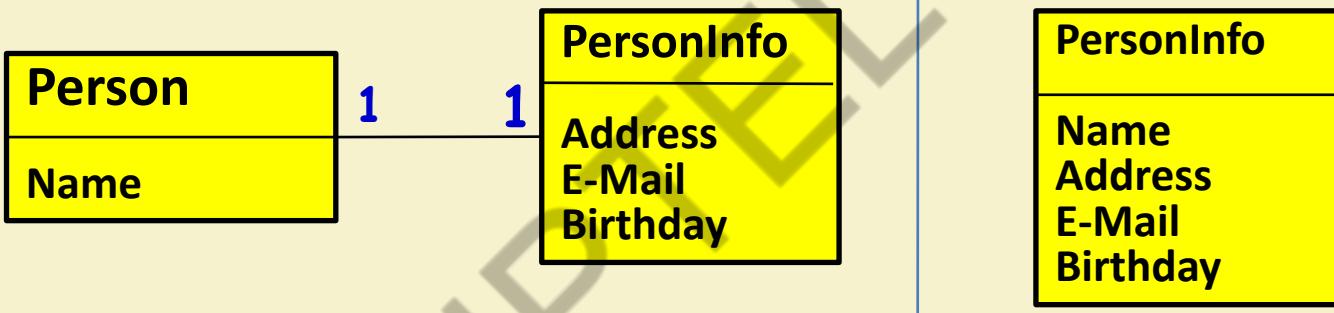
- 1 CPU has 0 to two Controllers
- 1-4 DiskDrives controlled by 1 SCSIController
- SCSIController is a (specialized) Controller

# Types of Class Relationships



# Overdoing Associations

- Avoid unnecessary Associations



Avoid This...

Do This



IIT KHARAGPUR

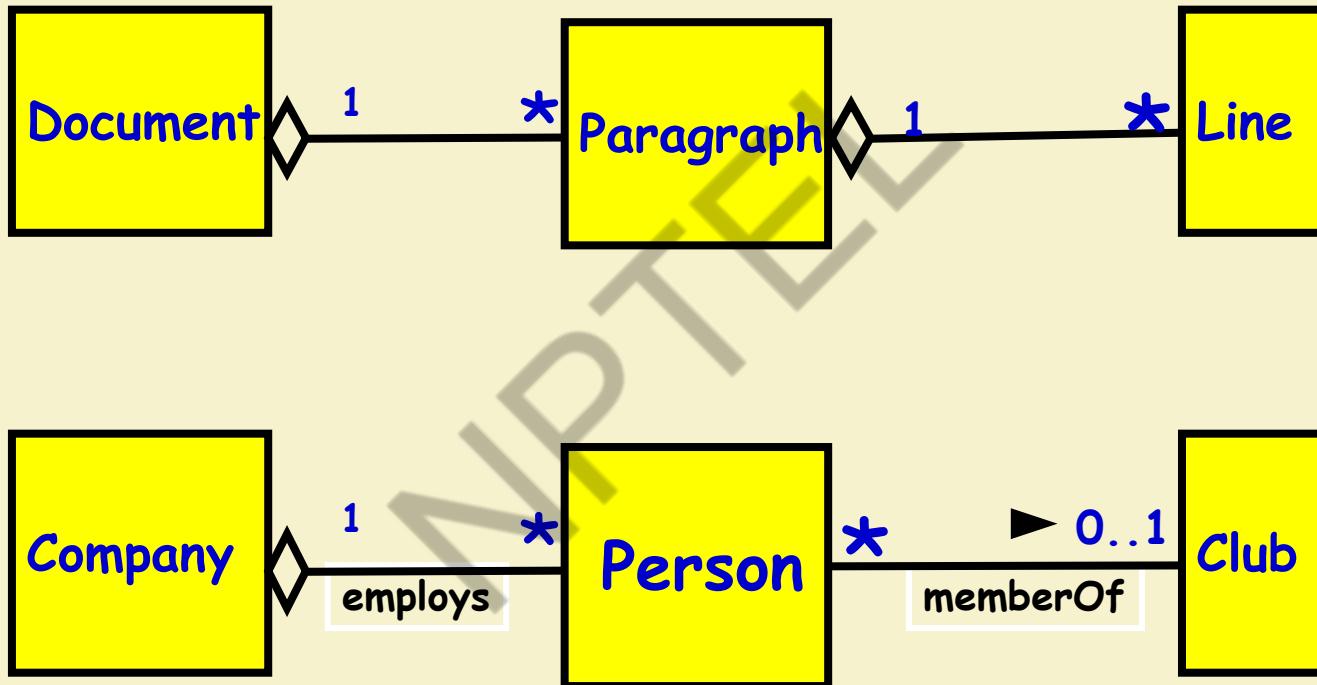


NPTEL ONLINE  
CERTIFICATION COURSES

# Aggregation Relationship

- Represents whole-part relationship
- Represented by a **diamond** symbol at the composite end.
- Usually creates the components:
  - Also, aggregate usually invokes the same operations of all its components.
  - This is in contrast to plain association
- Usually owner of the components:
  - But can share with other classes

# Aggregation Relationship



IIT KHARAGPUR

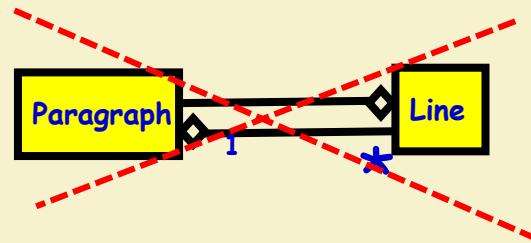


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Aggregation

cont...

- An aggregate object contains other objects.
- Aggregation limited to tree hierarchy:
  - No circular aggregate relation.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Aggregation vs. Inheritance

Cont...

## ● Inheritance:

- Different object types with similar features.
- Necessary semantics for similarity of behavior is in place.

## ● Aggregation:

- Containment allows construction of complex objects.



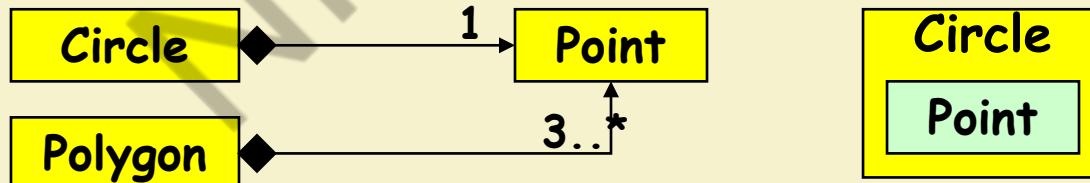
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Composition

- A stronger form of aggregation
  - The whole is the sole owner of its part.
    - A component can belong to only one whole
  - The life time of the part is dependent upon the whole.
    - **The composite must manage the creation and destruction of its parts.**



# Composition Relationship

- Life of item is same as that of order

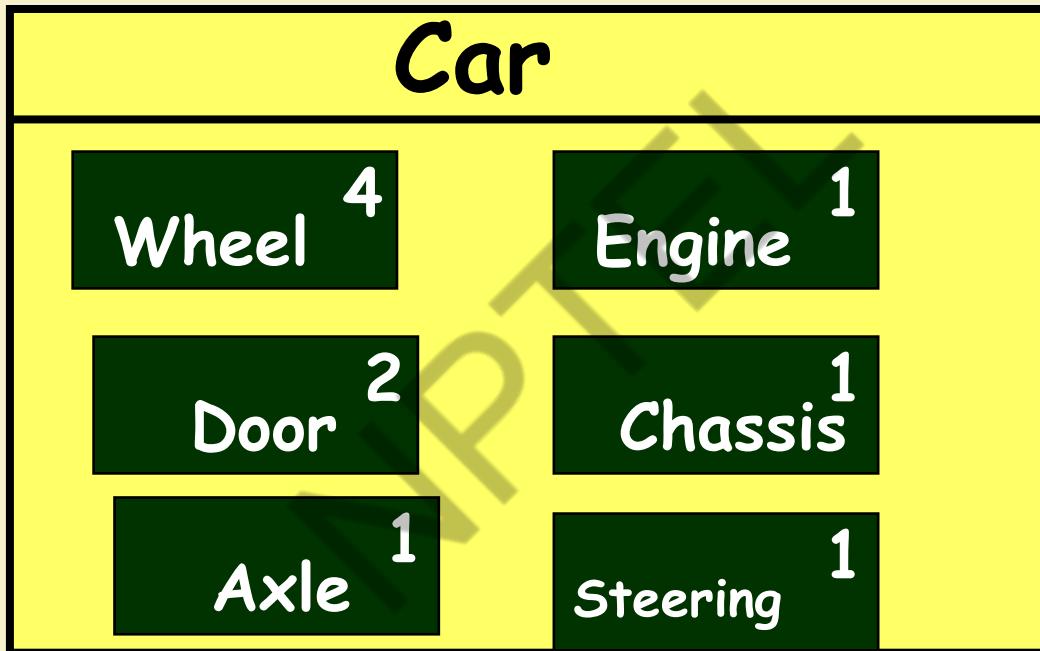


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Composition: Alternate Notation



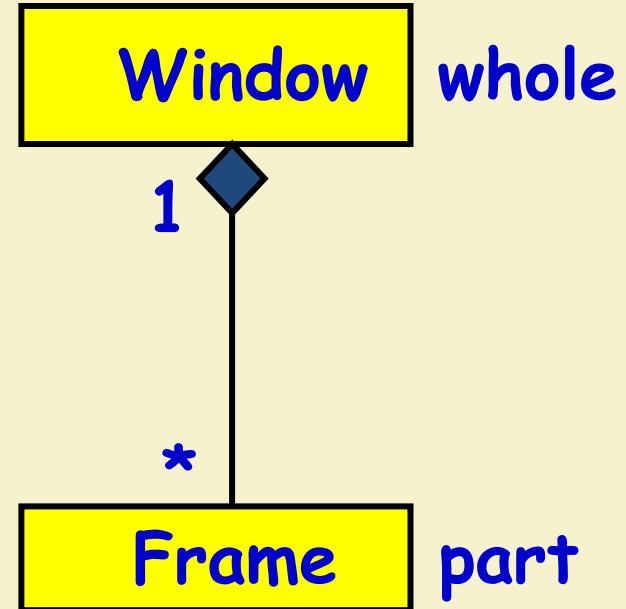
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Composition

- An object may be a part of ONLY one composite at a time.
  - Whole is responsible for the creation and disposition of its parts.

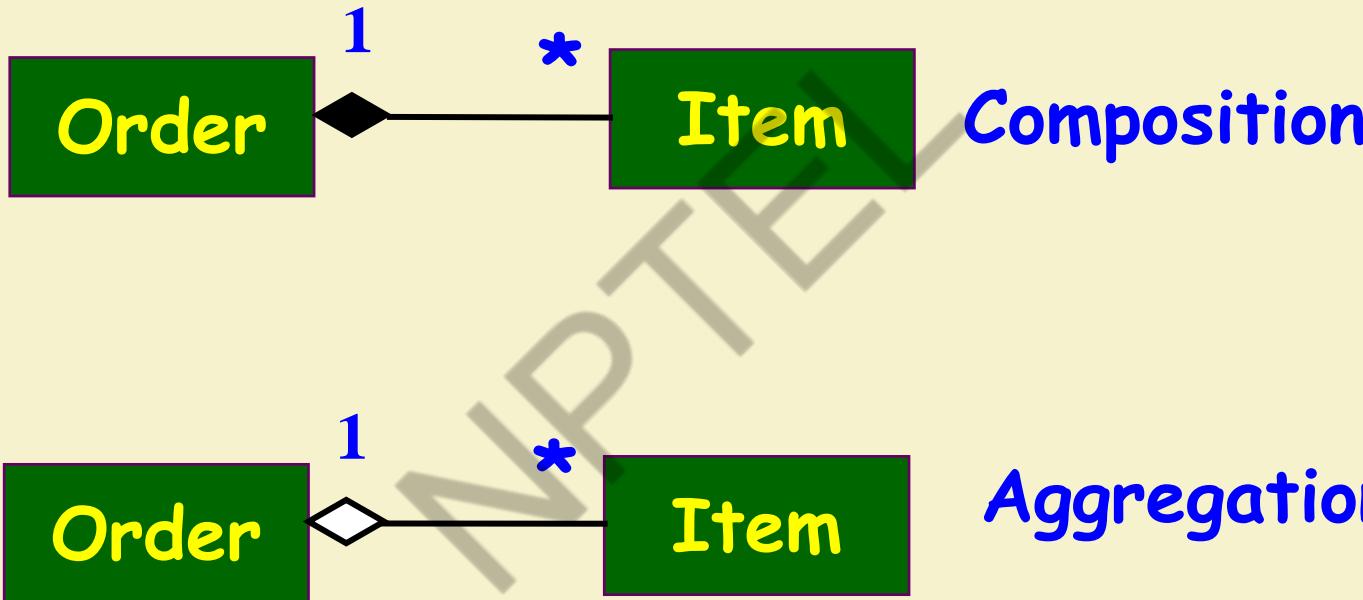


# Aggregation vs. Composition

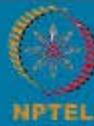
- Composition:
  - Composite and components have the same life line.
- Aggregation:
  - Lifelines are different.
- Consider an **order** object:
  - **Aggregation:** If order items can be changed or deleted after placing order.
  - **Composition:** Otherwise.



# Composition versus Aggregation



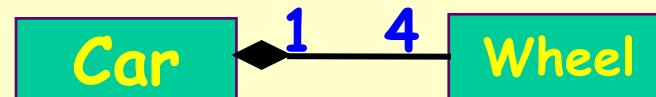
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Implementing Composition

```
public class Car{  
    private Wheel wheels[4];  
    public Car (){  
        wheels[0] = new Wheel();  
        wheels[1] = new Wheel();  
        wheels[2] = new Wheel();  
        wheels[3] = new Wheel();  
    }  
}
```



# How to identify aggregation/composition?

- Lifetime of part is bound within lifetime of composite
  - There is a create-delete dependency
- There is an obvious whole-part physical or logical assembly
- Some properties of composite propagate to parts (e.g., location)
- Operations applied to composite propagate to parts (e.g., destruction, movement, recording)

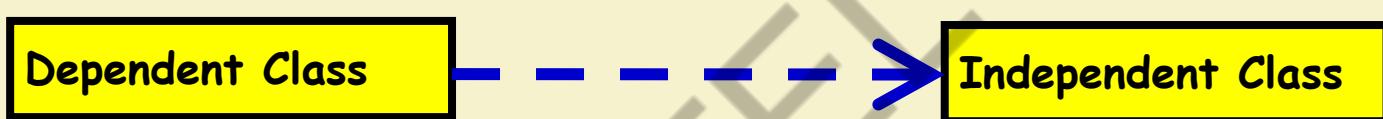


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Class Dependency



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Dependency

- Dependency relationship can arise due to a variety of reasons:
  - **Stereotypes are used to show the precise nature of the dependency.**

Type of dependency	Stereotype	Description
Abstraction	«abstraction»	Dependency of concrete class on its abstract class.
Binding	«bind»	Binds template arguments to create model elements from templates.
Realization	«realize»	Indicates that the client model element is an implementation of the supplier model element
Substitution	«substitute»	Indicates that the client model element takes place of the supplier.

# Association Vs. Aggregation

- Is aggregation an association?
- Is composition an aggregation?



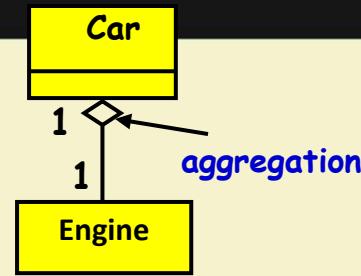
IIT KHARAGPUR



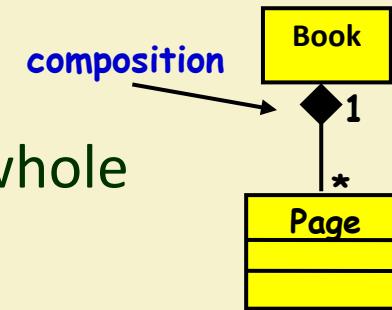
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Association Types

- **aggregation:** "is part of"
  - Symbolized by empty diamond

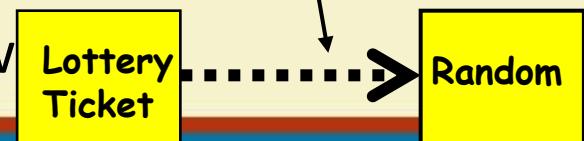


- **composition:** is made of
  - Stronger version of aggregation
  - The parts live and die with the whole
  - Symbolized by a filled diamond



- **dependency:** Depends on

- Represented by dotted arrow

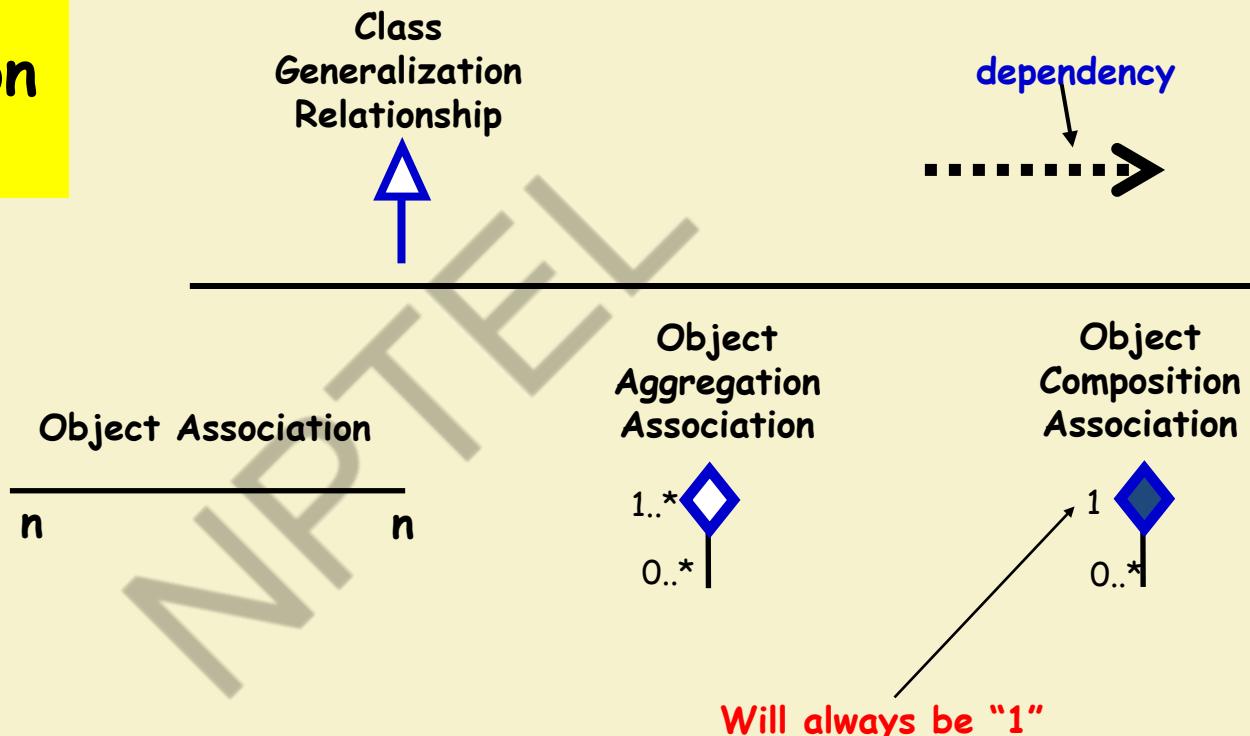


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# UML Class Relation Notation Summary

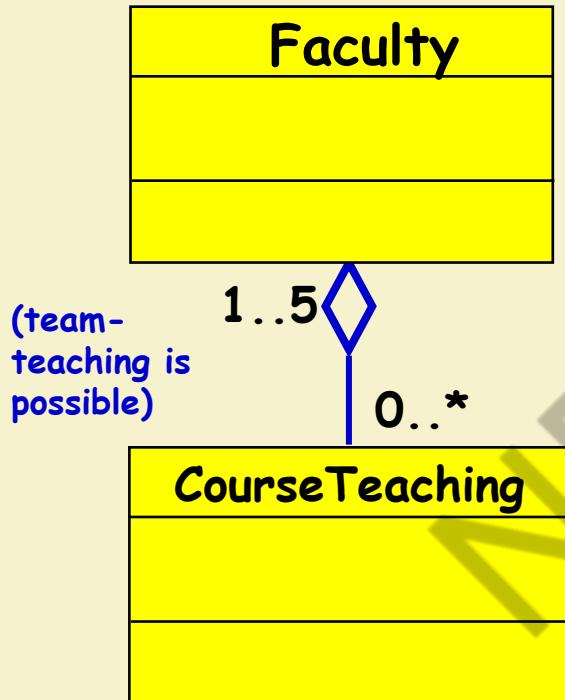


IIT KHARAGPUR

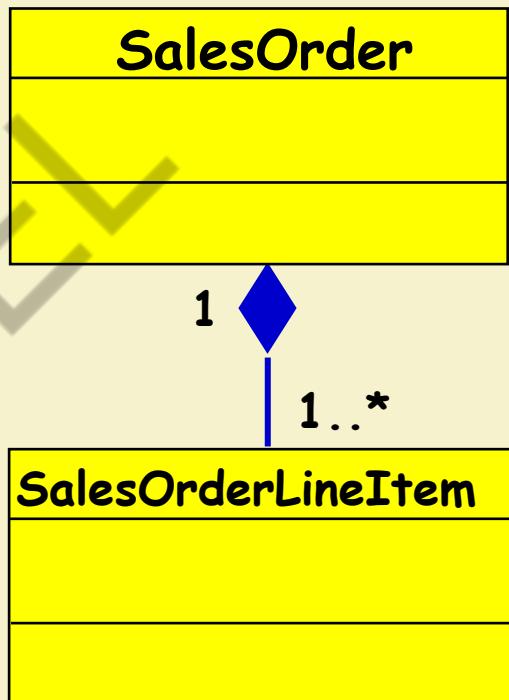


NPTEL ONLINE  
CERTIFICATION COURSES

# Aggregation



# Composition



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Class Relation Hints

- **Composition**
  - B is a permanent part of A
  - A contains B
  - A is a permanent collection of Bs
- **Subclass / Superclass**
  - A is a kind of B
  - A is a specialization of B
  - A behaves like B
- **Association (Collaboration)**
  - A delegates to B
  - A needs help from B
  - A and B are peers.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Class Diagram Inference Based on Text Analysis (based on Dennis, 2002)

- A common noun implies a class e.g. Book
- A proper noun implies an object (instance of a class): CSE Dept, OOSD, etc.
- An adjective implies an attribute e.g. price of book
- A “doing” verb implies an operation
  - Can also imply a relationship e.g. student issues Book
- A “having” verb implies an aggregation relationship
- A predicate or descriptive verb phrase elaborates an operation e.g. ISBN numbers are integers
- An adverb implies an attribute of an operation e.g. fast loading of image...

## Identify Class Relations

- Faculty & student
- Hospital & doctor
- Door & Car
- Member & Organization
- People & student
- Department & Faculty
- Employee & Faculty
- Computer Peripheral & Printer
- Account & Savings account



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- A square is a polygon
- Shyam is a student
- Every student has a name
- 100 paisa is one rupee
- Students live in hostels
- Every student is a member of the library
- A student can renew his borrowed books
- The Department has many students

**Identify Classes & Relations**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Identify Classes & Relations

- A country has a capital city
- A dining philosopher uses a fork
- A file is an ordinary file or a directory file
- Files contain records
- A class can have several attributes
- A relation can be association or generalization
- A polygon is composed of an ordered set of points
- A person uses a computer language on a project

- Describes static structure of a system
- Main constituents are classes and their relationships:
  - Generalization
  - Aggregation
  - Association
  - Various kinds of dependencies

## Class Diagram: Recap

# Summary of Relationships Between Classes

- **Association**
    - Permanent, structural, “has a”
    - Solid line (arrowhead optional)
  - **Aggregation**
    - Permanent, structural, a whole created from parts
    - Solid line with diamond from whole
  - **Dependency**
    - Temporary, “uses a”
    - Dotted line with arrowhead
  - **Generalization**
    - Inheritance, “is a”
    - Solid line with open (triangular) arrowhead
  - **Implementation**
    - Dotted line with open (triangular) arrowhead
- OR
- 
- The diagram illustrates four types of class relationships:
- Association:** Represented by a solid horizontal line.
  - Aggregation:** Represented by a solid horizontal line with a diamond symbol at the end closest to the source class.
  - Dependency:** Represented by a dotted horizontal line ending in a red arrowhead pointing to the target class.
  - Generalization:** Represented by a solid horizontal line ending in an open (triangular) arrowhead pointing to the target class.

# Object Diagram

LibraryMember

Mritunjay  
B10028  
C-108, Laksmikant Hall  
1119  
Mrituj@cse  
25-02-04  
25-03-06  
NIL

IssueBook( );  
findPendingBooks( );  
findOverdueBooks( );  
returnBook( );  
findMembershipDetails( );

LibraryMember

Mritunjay  
B10028  
C-108, Laksmikant Hall  
1119  
Mrituj@cse  
25-02-04  
25-03-06  
NIL

LibraryMember

Different representations of  
the LibraryMember object



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Interaction Diagram

- Can model the way a group of objects interact to realize some behaviour.
- How many interaction diagrams to draw?
  - Typically each interaction diagram realizes behaviour of a single use case
  - Draw one sequence diagram for each use case.

## A First Look at Sequence Diagrams

- **Captures how objects interact with each other:**
  - To realize some behavior (use case execution).
- Emphasizes time ordering of messages.
- Can model:
  - Simple sequential flow, branching, iteration, recursion, and concurrency.

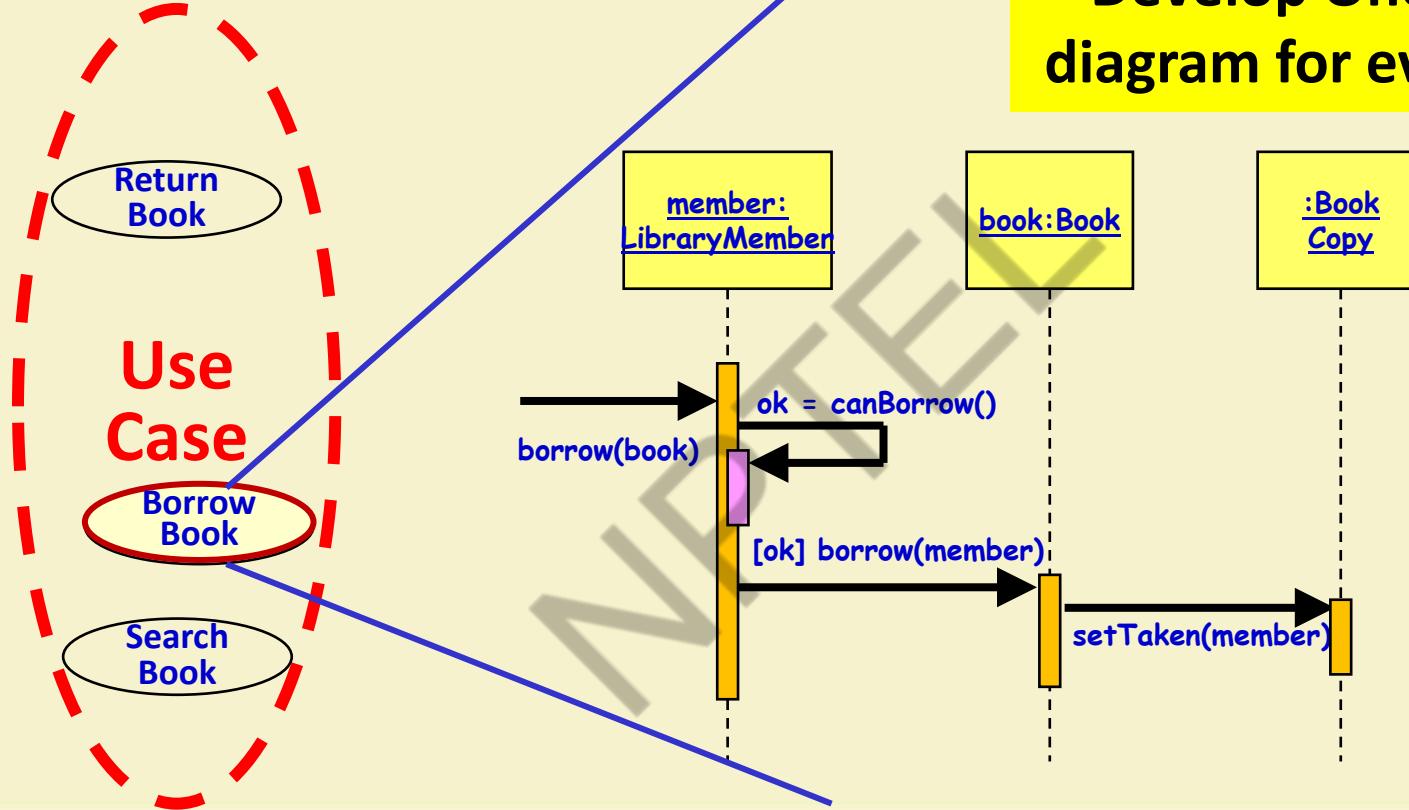


IIT KHARAGPUR



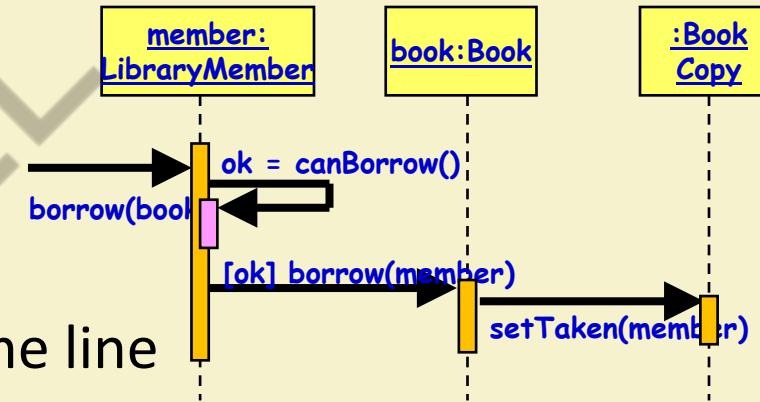
NPTEL  
ONLINE  
CERTIFICATION COURSES

Develop One Sequence  
diagram for every use case



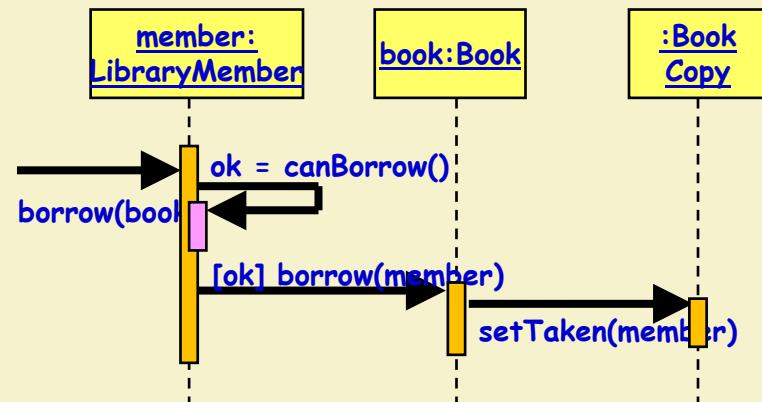
# Sequence Diagram

- Shows interaction among objects in a two-dimensional chart
- Objects are shown as boxes at top
- If object created during execution:
  - Then shown at appropriate place in time line
- Object existence is shown as dashed lines (lifeline)
- Object activeness, shown as a rectangle on lifeline



# Sequence Diagram cont...

- Messages are shown as arrows.
- Each message labelled with corresponding message name.
- Each message can be labelled with some control information.
- Two types of control information:
  - condition ([])
  - iteration (\*)



- **iteration marker** \*[for all objects]

- **[condition]**

- message is sent only if the condition is true

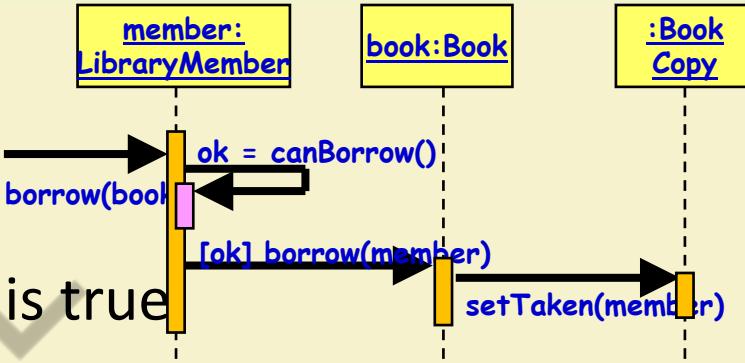
- **self-delegation**



- a message that an object sends to itself

- **Loops and conditionals:**

- UML2 uses a new notation called **interaction frames** to support these



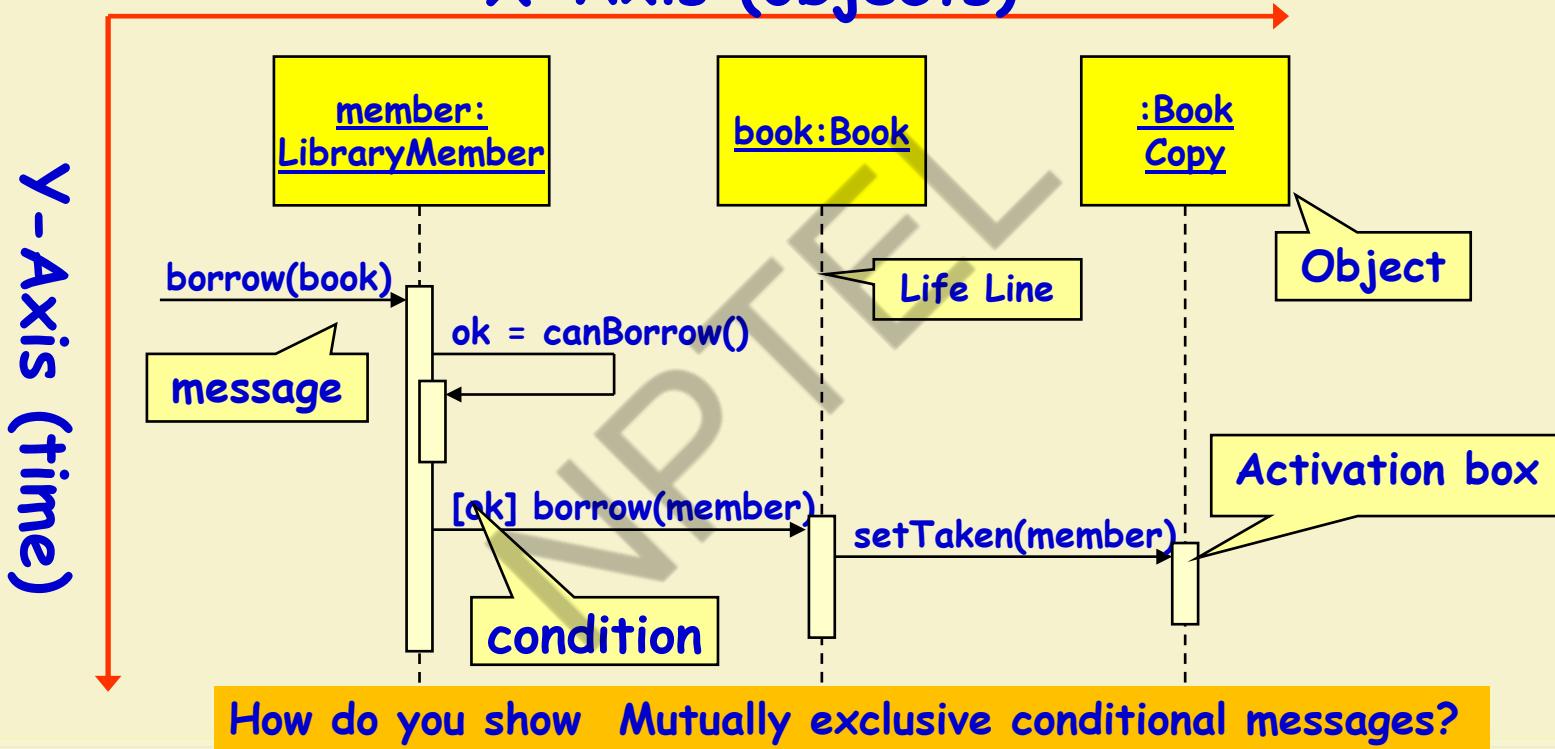
**Gist of Syntax**

# Control logic in Interaction Diagrams

- **Conditional Message**
  - [ variable = value ] message()
  - Message is sent only if clause evaluates to *true*
- **Iteration (Looping)**
  - \* [ i := 1..N ] message()
  - “\*” is required; [ ... ] clause is optional
  - The message is sent many times to possibly multiple receiver objects.

# Elements of A Sequence Diagram

## X-Axis (objects)



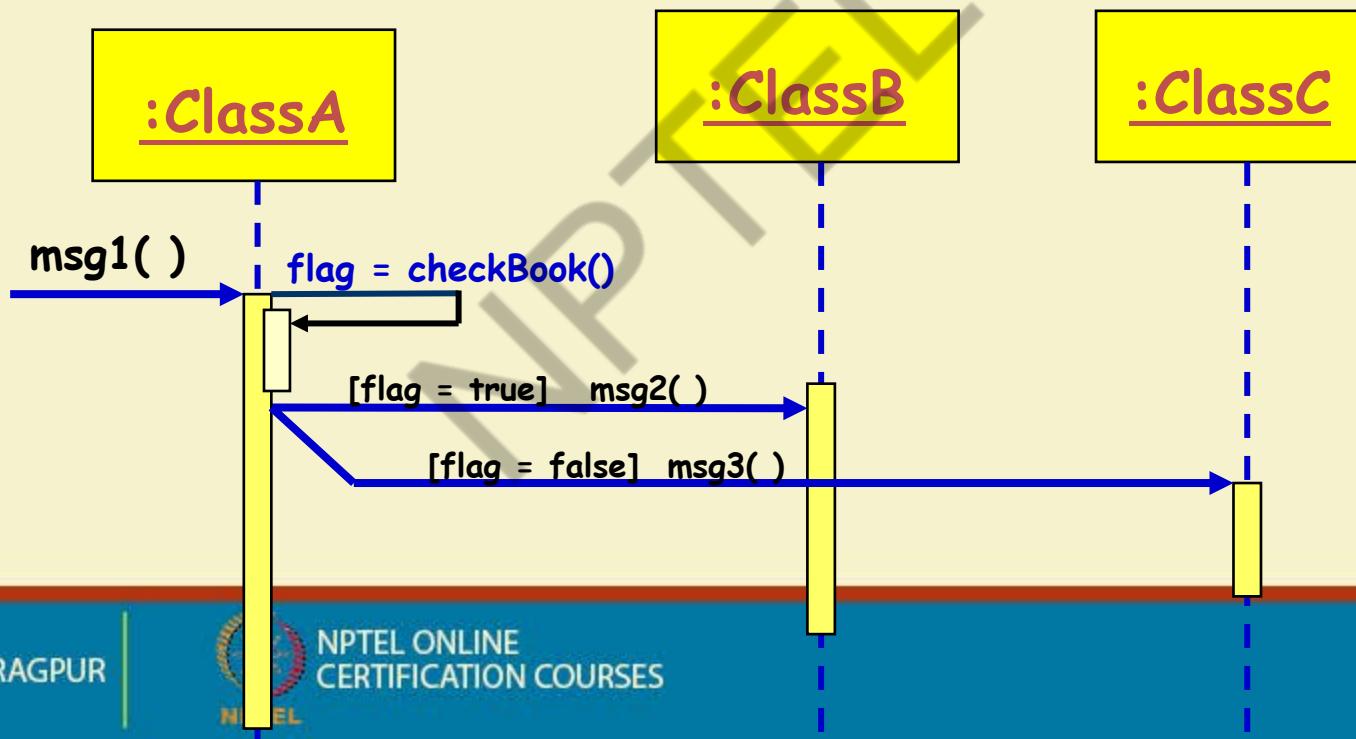
IIT KHARAGPUR

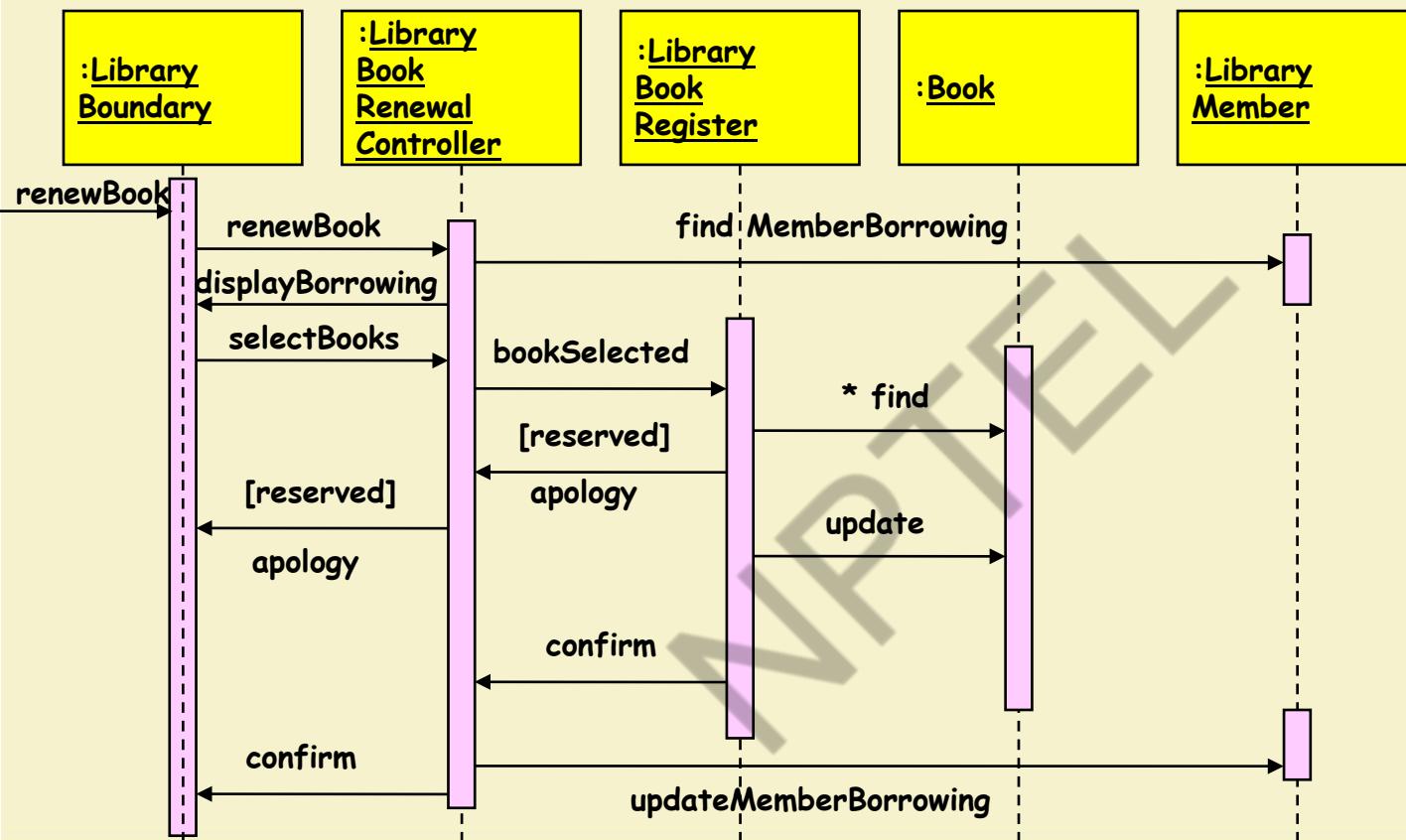


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Sequence Diagrams

How to represent Mutually exclusive conditional invocations? If book is available, invoke msg2 on ClassB else invoke msg3 on classC,





Sequence  
Diagram for  
the renew  
book use  
case



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Example: Develop Sequence Diagram

- A user can use a travel portal to plan a travel
- When the user presses the plan button, a travel agent applet appears in his window
- Once the user enters the source and destination,
  - The travel agent applet computes the route and displays the itinerary.
  - Travel agent widget disappears when user presses close button

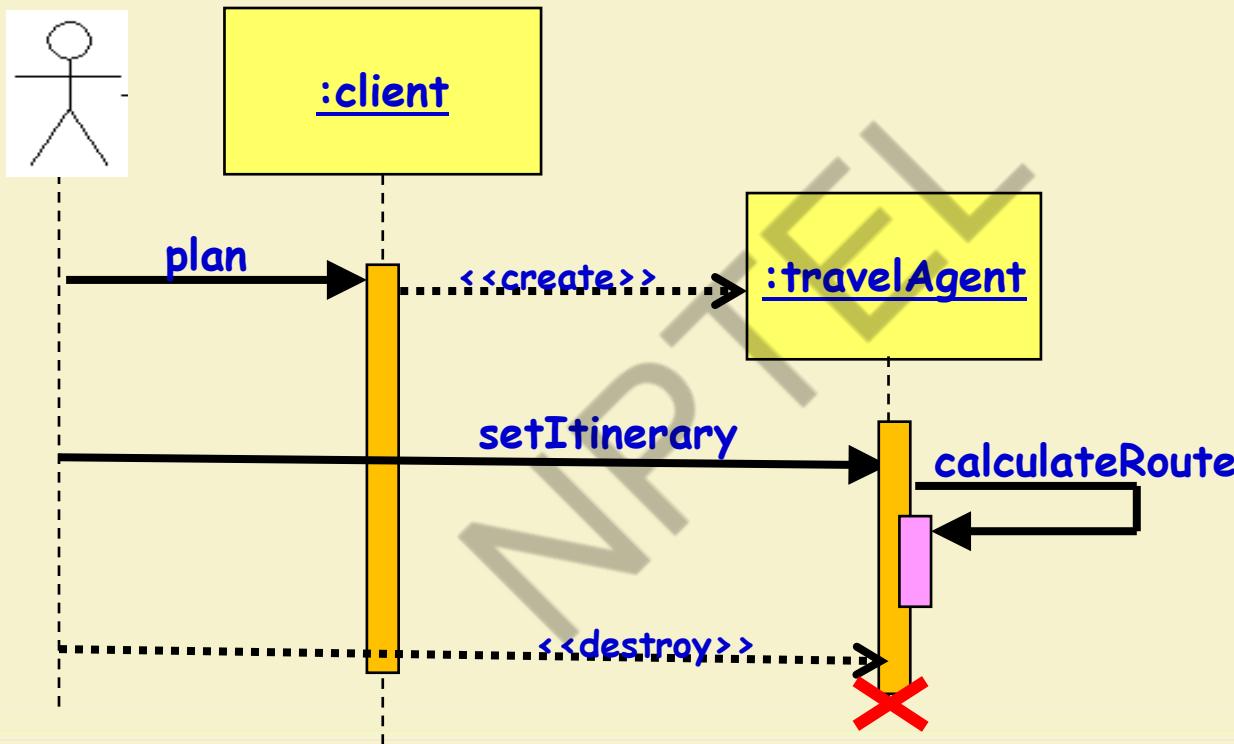


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example: Solution



IIT KHARAGPUR



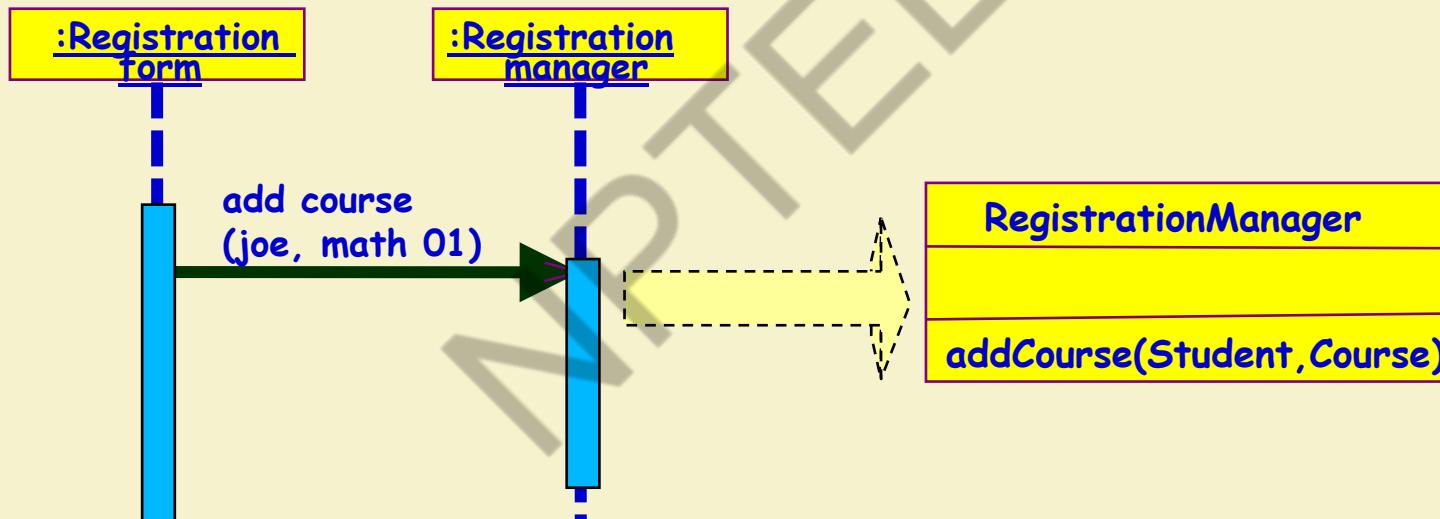
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Return Values

- Optionally indicated using a dashed arrow:
  - Label indicates the return value.
  - Don't need when it is obvious what is being returned, e.g.  
`getTotal()`
- **Model a return value only when you need to refer to it elsewhere:**
  - **Example:** A parameter passed to another message.

# Method Population in Classes

- Methods of a class are determined from the interaction diagrams...

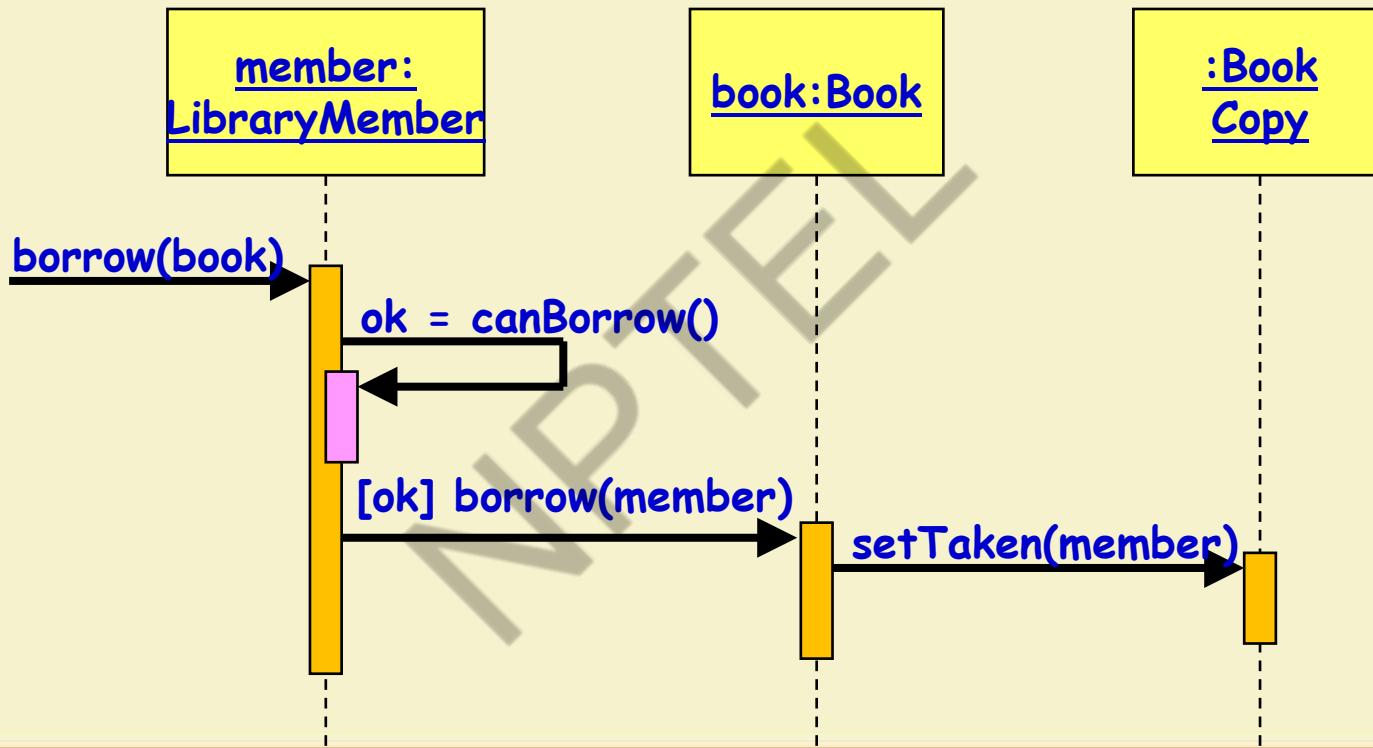


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Example Sequence Diagram: Borrow Book Use Case



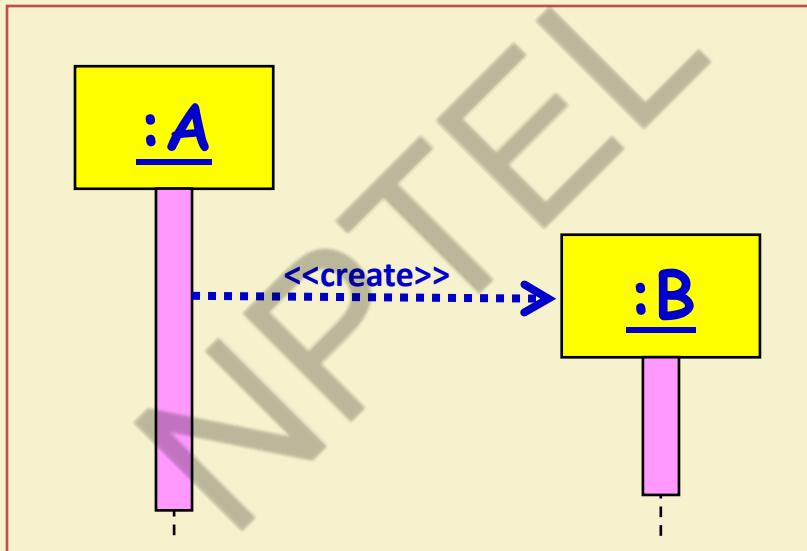
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

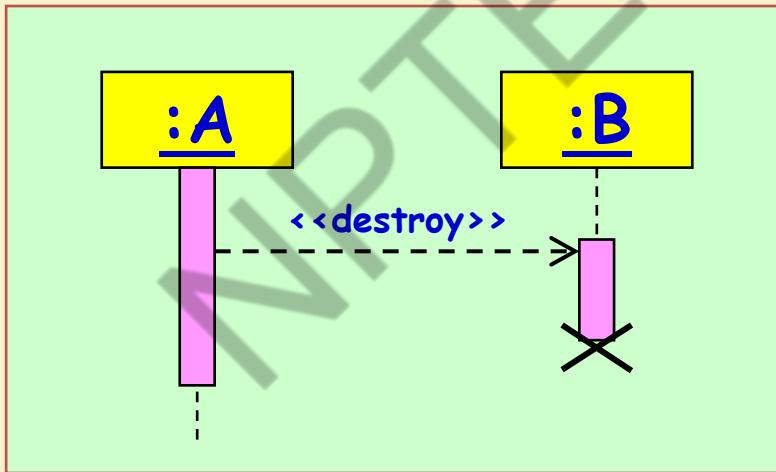
# Object Creation

- An object may create another object via a <<create>> message.



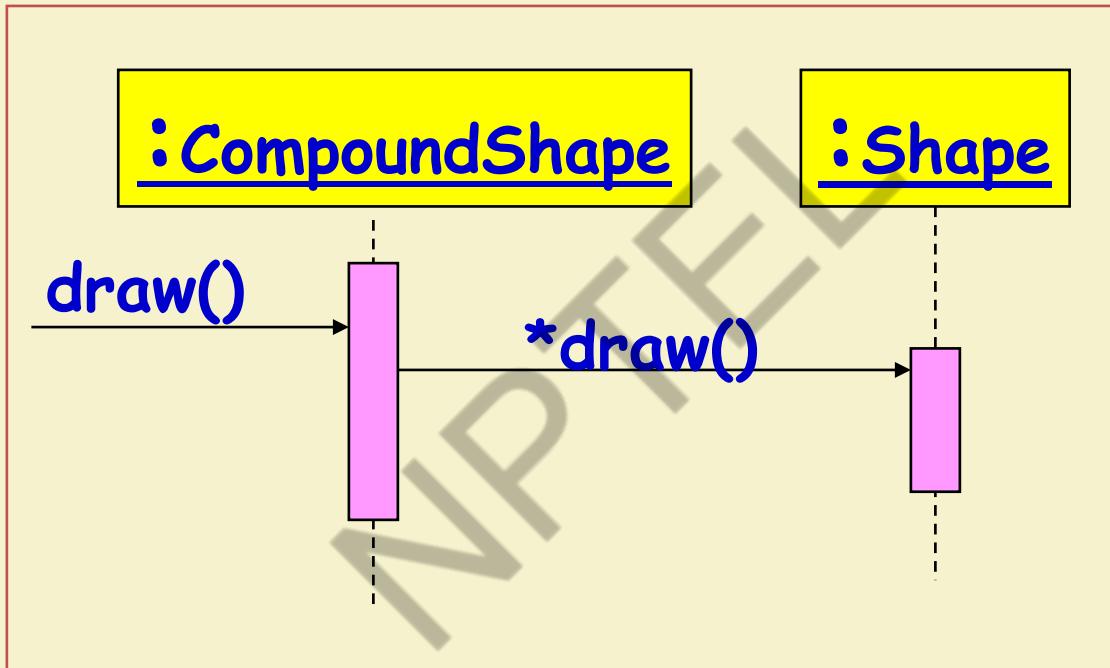
# Object Destruction

- An object may destroy another object via a <<destroy>> message.
  - An object may also destroy itself.
- **But, how do you destroy an object in Java?**
  - **Avoid modeling object destruction unless memory management is critical.**



# Control Information

Iteration  
example  
UML 1.x:

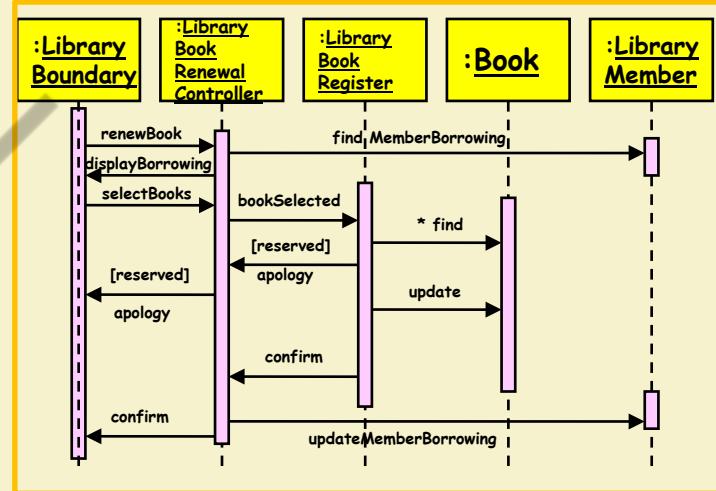
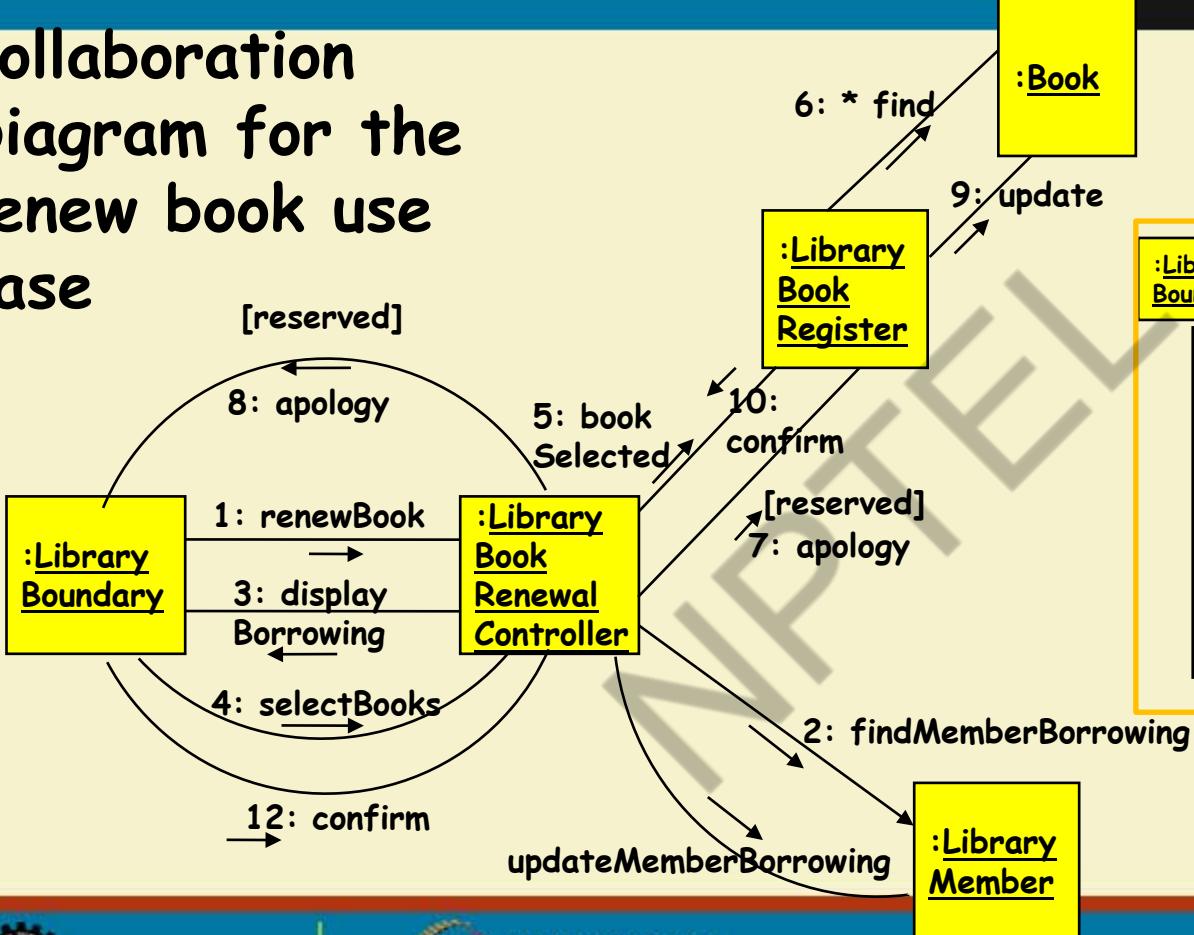


IIT KHARAGPUR

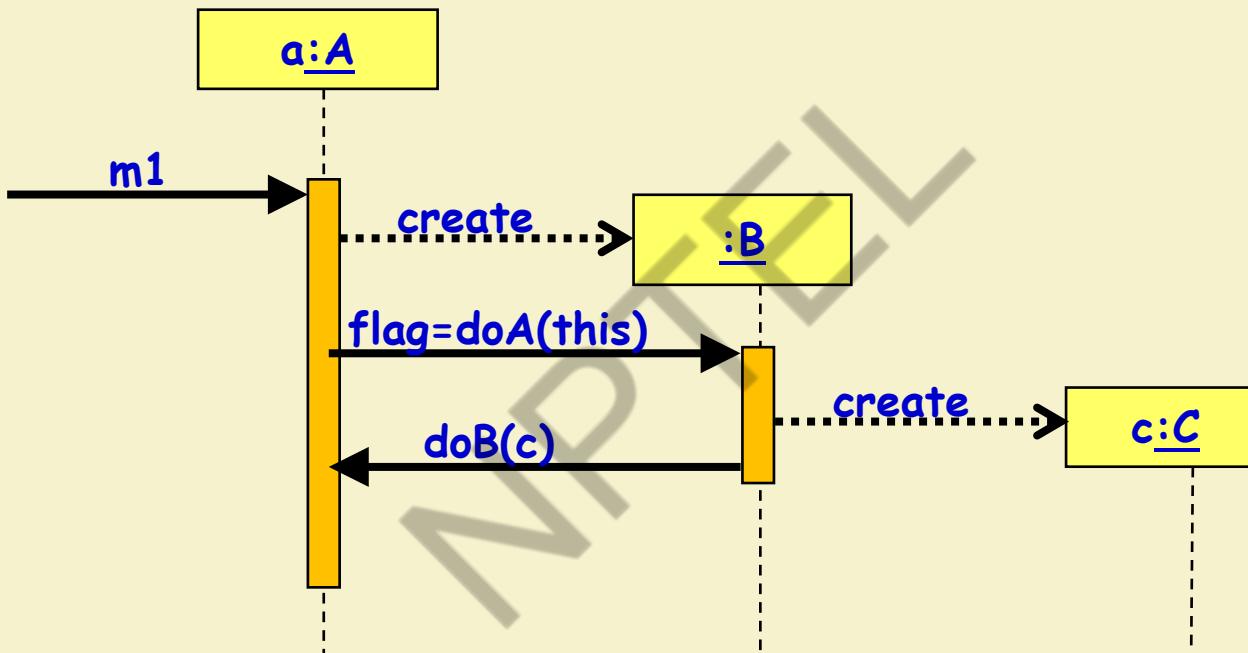


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Collaboration Diagram for the renew book use case



# Quiz: Write Code for class B



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Quiz: Ans

```
public class B {
```

```
...
```

```
int doA(A a) {  
    int flag;  
    C c = new C();  
    a.doB(c); ...  
    return flag; }  
}
```



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# State Machine Diagrams



IIT KHARAGPUR

9/13/2018



NPTEL ONLINE  
CERTIFICATION COURSES

# State Chart Diagram

Cont...

- State chart avoids two problems of FSM:
  - State explosion
  - Lack of support for representing concurrency
- A hierarchical state model:
  - Can have **composite states** --- OR and AND states.



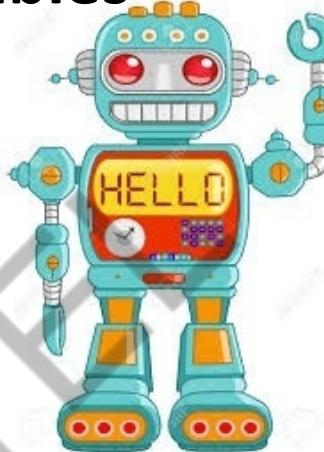
IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Robot: State Variables

- Movement: On, OFF
- Direction: Forward, Backward, left, Right
- Left hand: Raised, Down
- Right hand: Raised, down
- Head: Straight, turned left, turned right
- Headlight: On, Off
- Turn: Left, Right, Straight

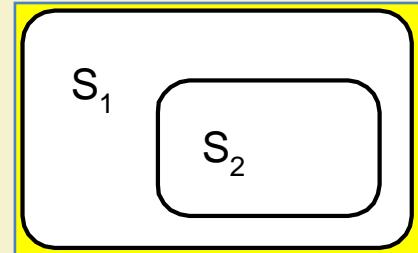


How many states in the state machine model?

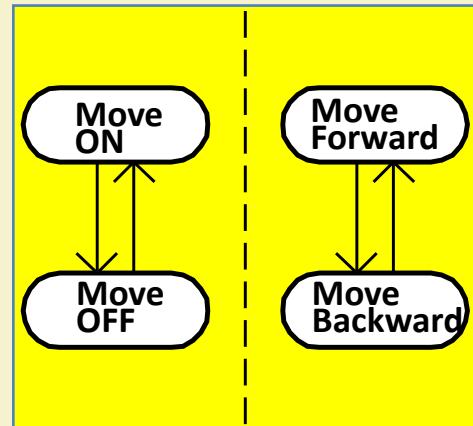
FSM: exponential rise in number of states with state variables

# Features of Statecharts

- Two major features are introduced :
  - **nested** states
  - **concurrent** states
- Many other features have also been added:
  - History state
  - broadcast messages
  - actions on state entry, exit
  - ...



Nested State Diagrams



Concurrent State Diagrams



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## State Chart Diagram

- Elements of state chart diagram
- **Initial State:** A filled circle
- **Final State:** A filled circle inside a larger circle
- **State:** Rectangle with rounded corners
- **Transitions:** Arrow between states, also boolean logic condition (**guard**)
- UML extended state chart is called state machine



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# How to Encode an FSM?

- Three main approaches:
  - **Doubly nested switch in loop**
  - **State table**
  - **State Design Pattern**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## 3 Principal Ways

- **Doubly nested switch in loop:**
  - Global variables store state --- Used as switch
  - Event type is discriminator in second level switch
  - Hard to handle concurrent states, composite state, history, etc.
- **State table:** Straightforward table lookup
- **Design Pattern:**
  - States are represented by classes
  - Transitions as methods in classes



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Doubly Nested Switch Approach

```
int state, event; /* state and event are variables */
```

```
while(TRUE){
```

```
    switch (state){
```

```
        Case state1: switch(event){
```

```
            case event1:  
                state=state2; etc...; break;
```

```
            case event2:....  
            default:
```

```
}
```

```
        Case state2: switch(event){...}
```

```
....
```

```
}
```



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# State Table Approach

- From the state machine, we can set up a **state transition table** with a column for the actions associated with each transition

Present state	Event	Next state	Actions
Light_off	e1	Light_off	none
	e2	Light_on	set red LED flashing
Light_on	e1	Light_on	none
	e2	Light_off	reset red LED flashing



IIT KHARAGPUR



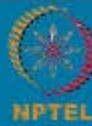
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# An Object-Oriented Design Process

- We discuss a design process based to a large extent on Larman's approach:
  - Also synthesizes features from various other methodologies.
- From requirements specification, an initial model is developed (OOA):
  - **Analysis model is iteratively refined into a design model**
- Design model is implemented using an OO language.



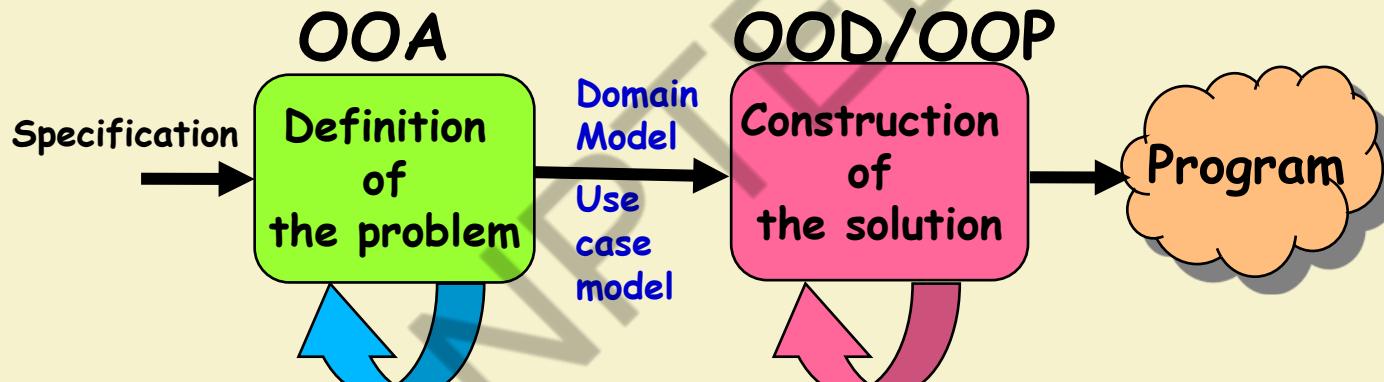
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# OOAD Process

Iterative and Incremental



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# OOA versus OOD?

- **Analysis:**
  - An elaboration of requirements.
  - Independent of any specific implementation
- **Design:**
  - A refinement of the analysis model.
  - Takes implementation constraints into account

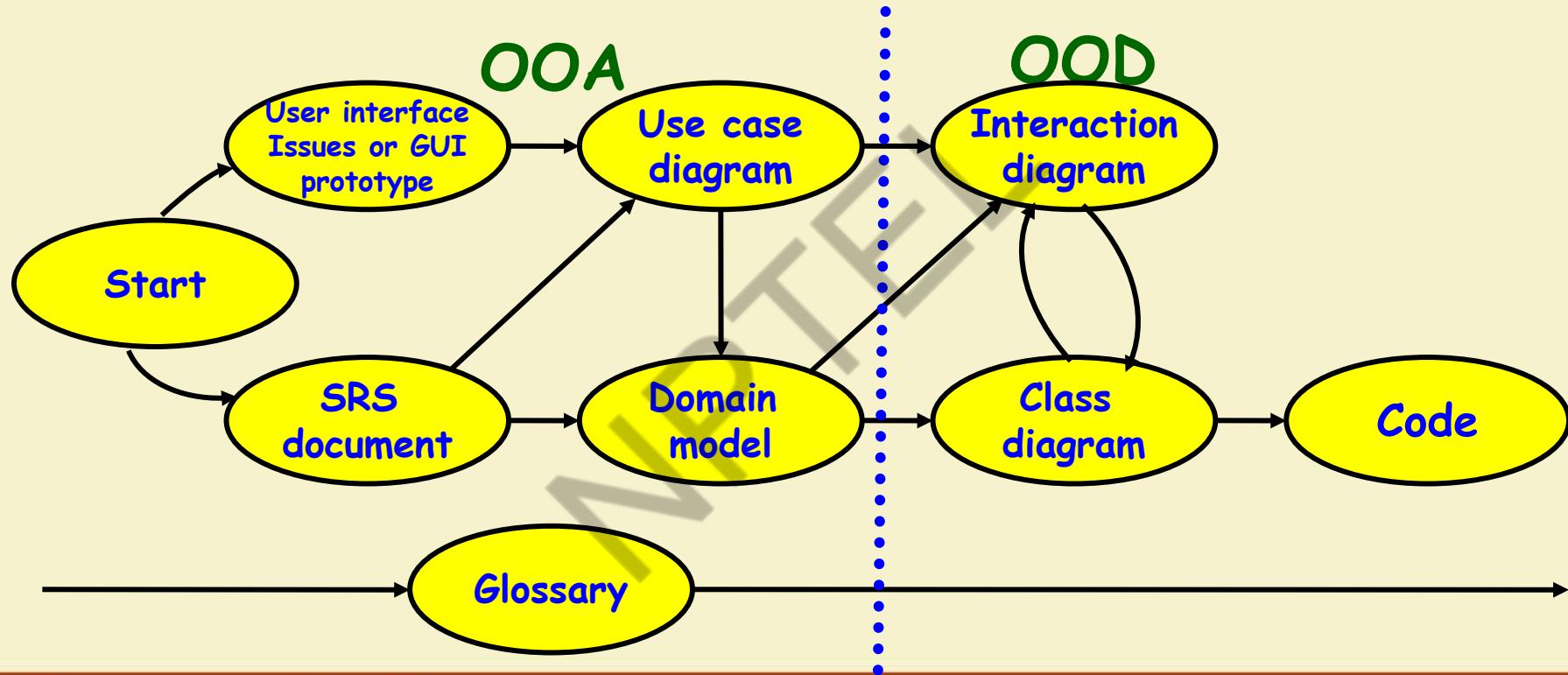


IIT KHARAGPUR

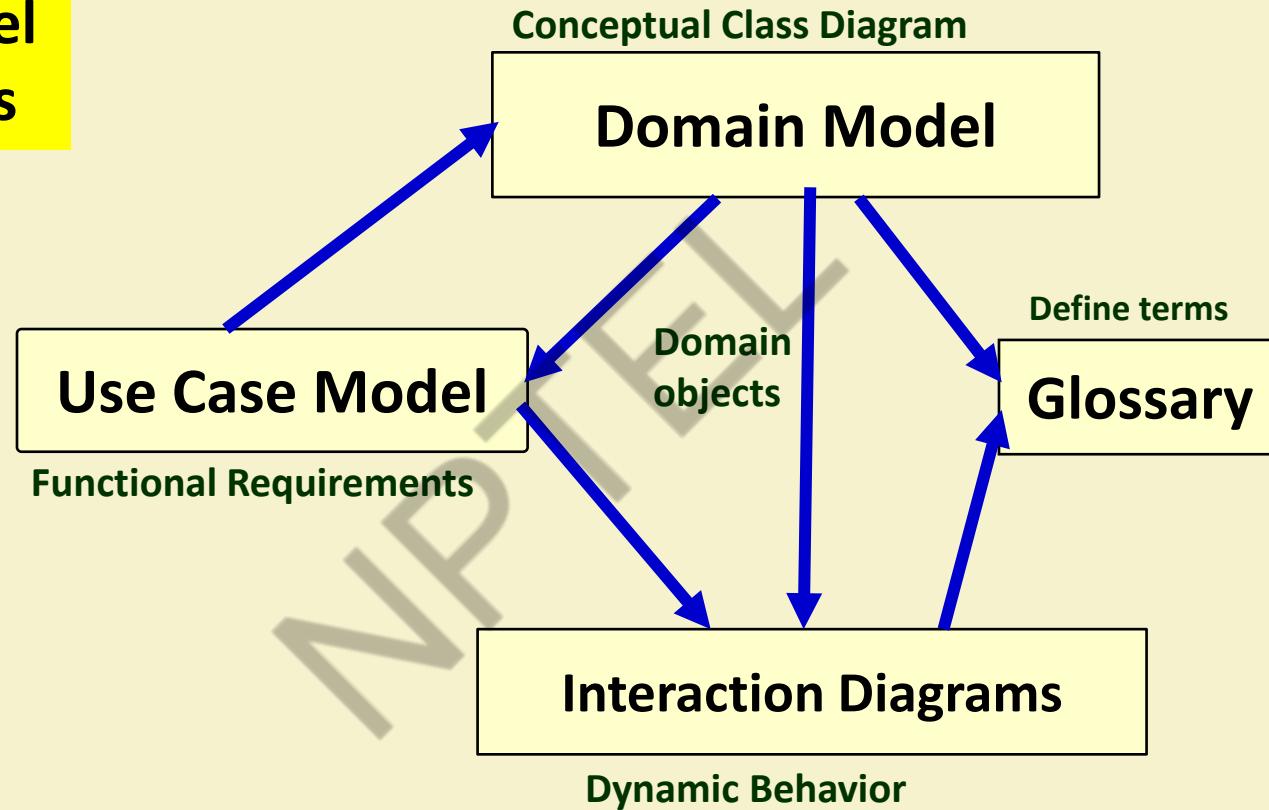


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Design Process



# Domain Model Relationships



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Domain Modelling

- Represent concepts or objects appearing in the problem domain.
  - Also capture object relationships.
- Three types of objects are identified:
  - **Boundary objects**
  - **Entity objects**
  - **Controller objects**



IIT KHARAGPUR

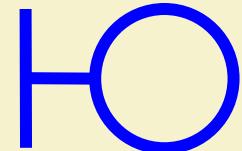


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

Three different stereotypes are used to represent classes :  
**<<boundary>>, <<control>>, <<entity>>.**

## Class Stereotypes

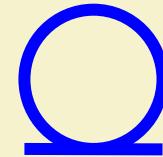
Boundary  
Cashier Interface



Control  
Withdrawal manager



Entity  
Account



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Boundary Objects

- Handle interaction with actors:
  - User interface objects**
- Often implemented as screens, menus, forms, dialogs etc.
- Do not perform processing:
  - But may validate input, format output, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Entity Objects

- Hold information over long term:
  - e.g. Book, BookRegister
- Normally are dumb servers:
  - Responsible for storing data, fetching data etc.
  - Elementary operations on data such as searching, sorting, etc.
- Often appear as nouns in the problem description...



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Controller Objects

- **Overall responsibility to realize use case behavior:**
  - Interface with the boundary objects
  - Coordinate the activities of a set of entity objects
- **Embody most of the business logic required for use case execution:**
- There can be more than one controller to realize a single use case



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Controller Classes

- Controller classes coordinate, sequence, transact, and otherwise control other objects...
- In Smalltalk MVC mechanism:
  - These are called controllers

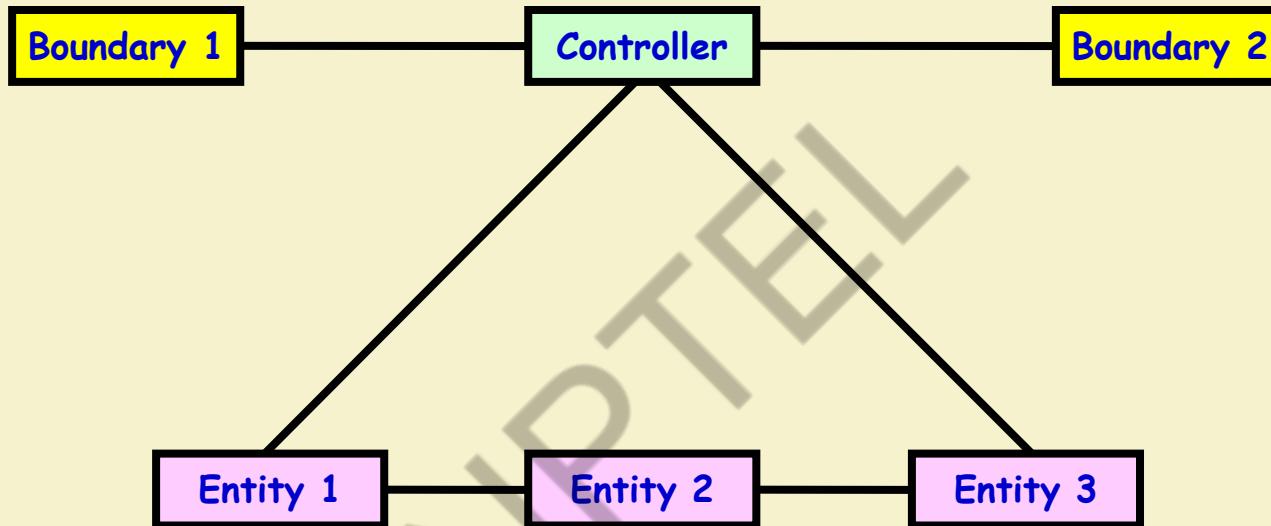


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Use Case Realization



Realization of use case through the collaboration of  
Boundary, controller and entity objects



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Domain Analysis

- Three types of classes are to be identified:
  - Boundary class (Actor-use case pair)
  - Controller class (One per use case)
  - Entity class (Noun analysis)



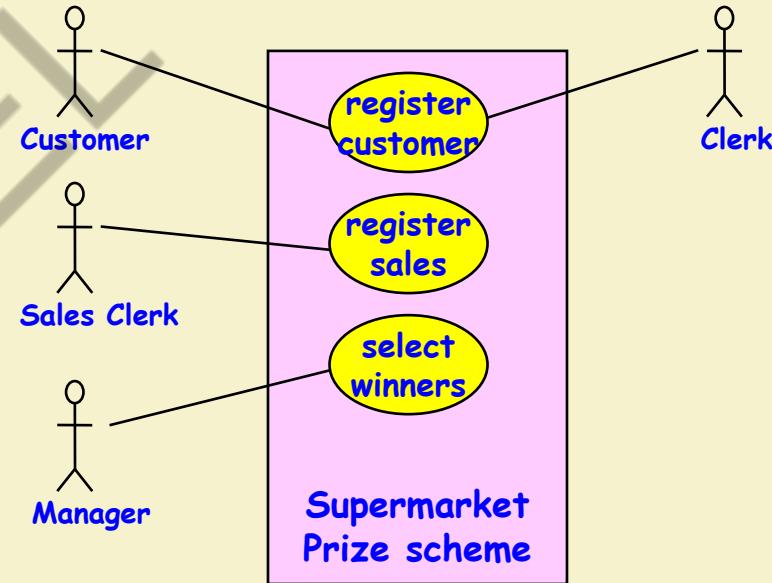
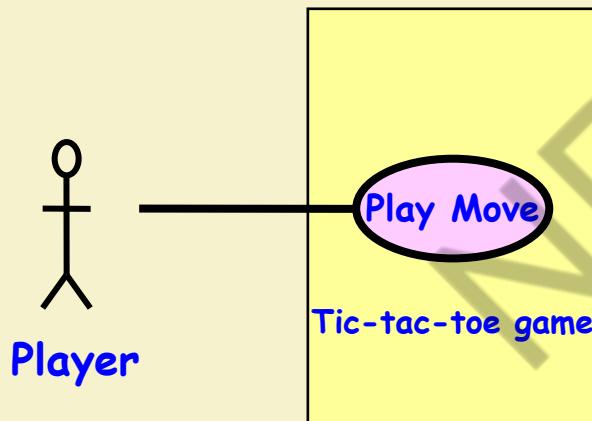
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Identification of Boundary Objects

- Need one boundary object :
  - For every actor-use case pair



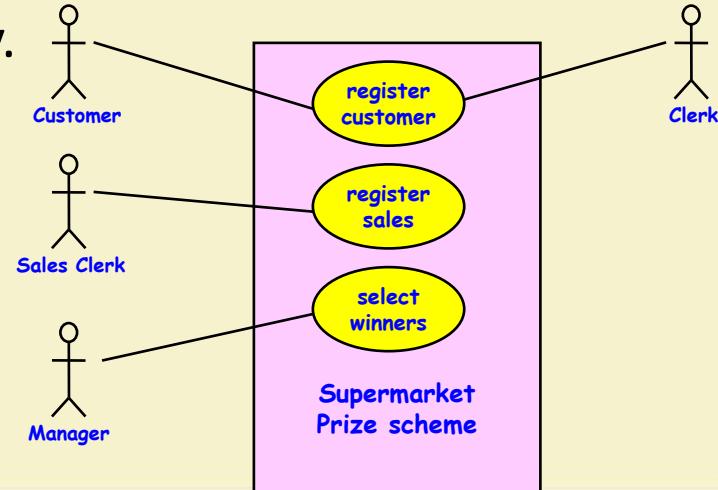
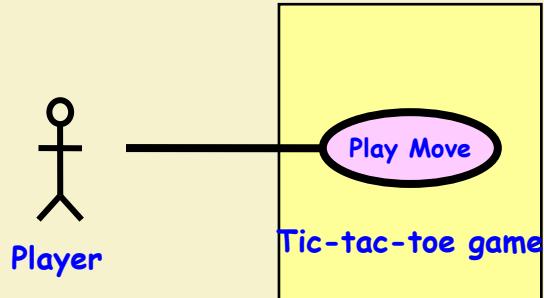
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Identification of Controller Objects

- Examine the use case diagram:
  - Add one controller class for each use case.
  - Some controllers may need to be split into two or more controller classes if they get assigned too much responsibility.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Identification of Entity Objects by Noun Analysis

- Entity objects usually appear as nouns in the problem description.
- From the list of nouns, need to exclude:
  - **Users (e.g. accountant, librarian, etc)**
  - **Passive verbs (e.g. Acknowledgment)**
  - **Those with which you can not associate any data to store**
  - **Those with which you can not associate any methods**
- Surrogate users may need to exist as classes:
  - **Library member**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Remember that a class represents a group (classification) of objects with the same behavior.
  - We should therefore look for existence of similar objects during noun analysis
- Even then, class names should be singular nouns:
  - Examples: **Book, Student, Member**

## Identifying Classes



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Noun Analysis: Example

A trading house maintains names and addresses of its regular customers. Each customer is assigned a unique customer identification number (CIN). As per current practice, when a customer places order, the accounts department first checks the credit-worthiness of the customer.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Identifying Classes by Noun Analysis

- A partial requirements document:

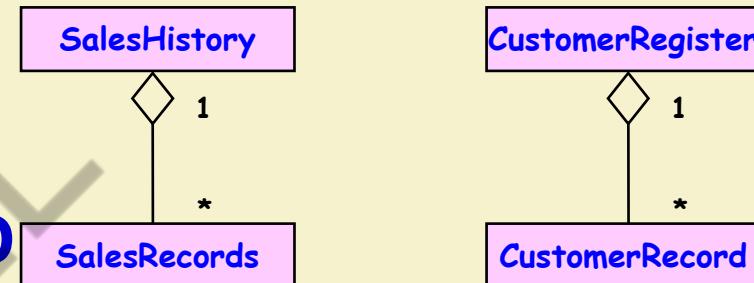
A ~~trading house~~ maintains ~~names~~ and ~~addresses~~ of its regular ~~customers~~. Each ~~customer~~ is assigned a unique customer identification number (CIN). As per current practice, when a customer places ~~order~~, The ~~accounts~~ department first checks the ~~credit-worthiness~~ of the ~~customer~~.

- Not all nouns correspond to a class in the domain model

# Identification of Entity Objects

- Usually:

- Appear as data stores in DFD
- Occur as group of objects that are aggregated
- The aggregator corresponds to registers in physical world

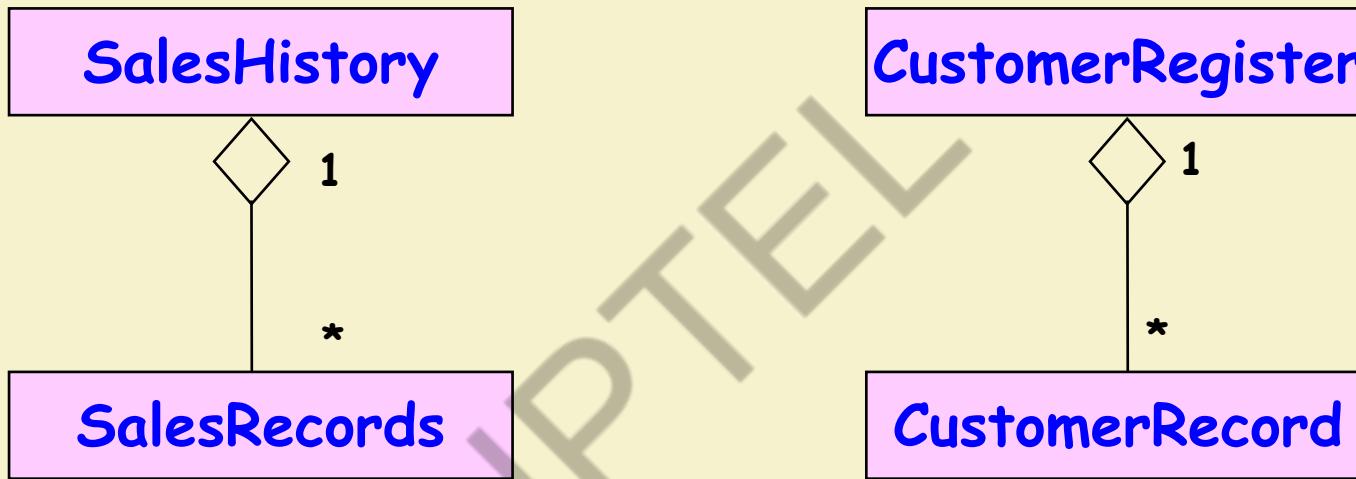


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 2: Initial Domain Model



Initial domain model



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 1: Tic-Tac-Toe Computer Game

- A human player and the computer make alternate moves on a 3X3 square.
- A move consists of marking a previously unmarked square.
- The user inputs a number between 1 and 9 to mark a square
- Whoever is first to place three consecutive marks along a straight line (i.e., along a row, column, or diagonal) on the square wins.

## Example 1: Tic-Tac-Toe Computer Game cont...

- As soon as either of the human player or the computer wins,
  - A message announcing the winner should be displayed.
- If neither player manages to get three consecutive marks along a straight line,
  - And all the squares on the board are filled up,
  - Then the game is drawn.
- The computer always tries to win a game.

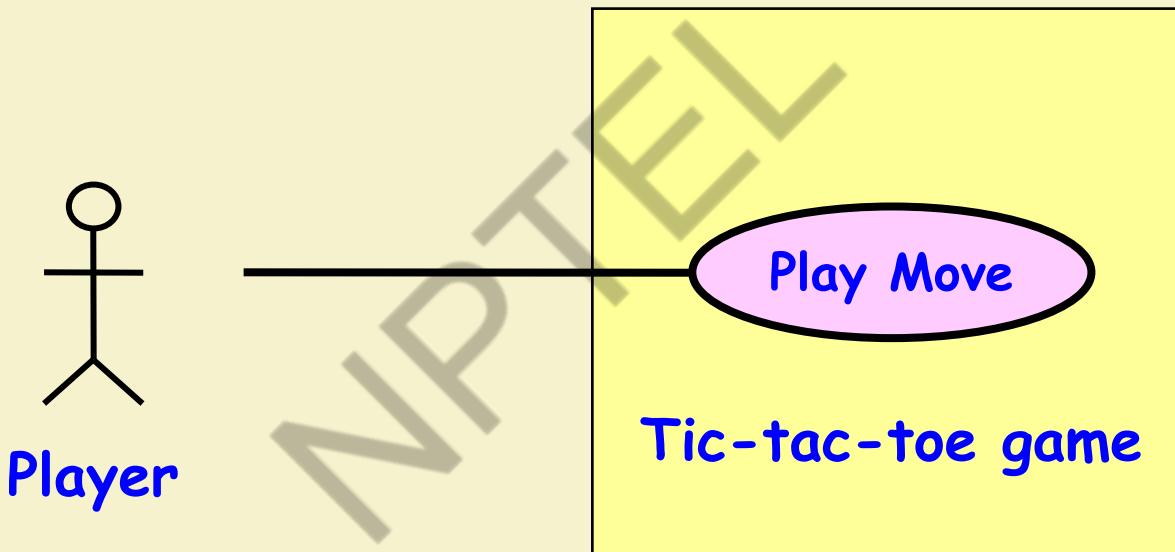


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example 1: Tic-Tac-Tie Use Case Model



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Example 1: Initial and Refined Domain Model

Board

Initial domain model

PlayMoveBoundary

PlayMoveController

Board

Refined domain model



IIT KHARAGPUR

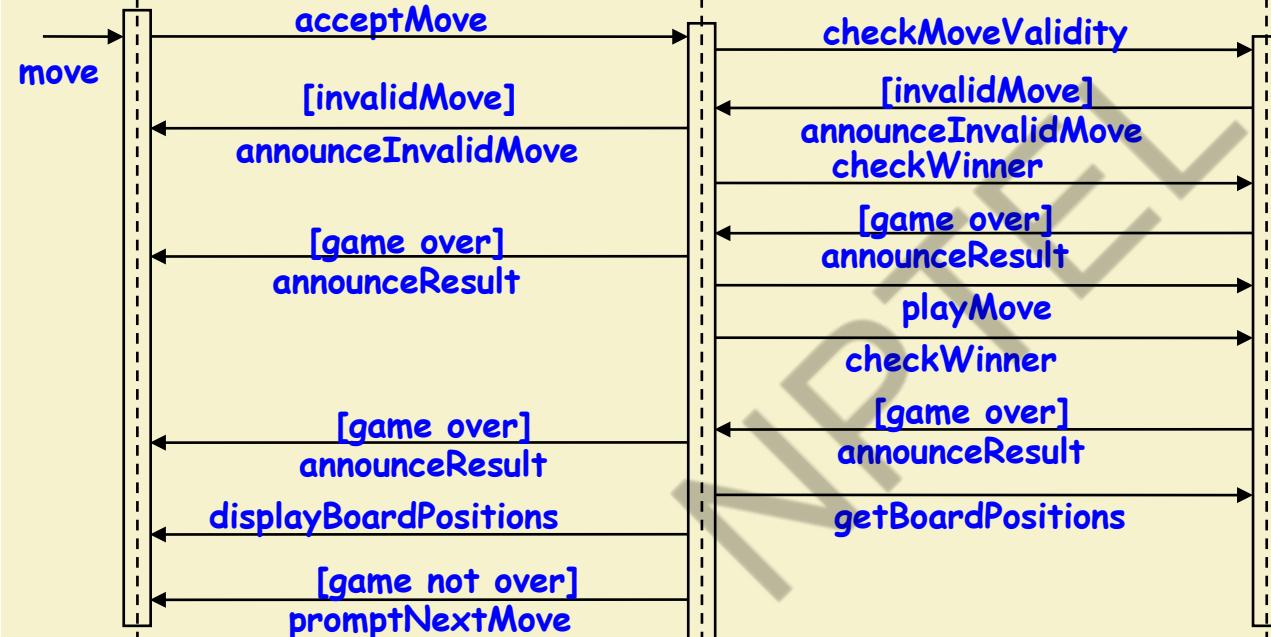


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

:playMove  
Boundary

:playMove  
Controller

:board



**Example 1:**  
**Sequence**  
**Diagram:**  
**play move**  
**use case**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# CRC Card

- Used to assign responsibilities (methods) to classes.
- Complex use cases:
  - Realized through collaborative actions of dozens of classes.
  - Without CRC cards, it becomes difficult to determine which class should have what responsibility.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Class-Responsibility-Collaborator(CRC) Cards

- Pioneered by Ward Cunningham and Kent Beck.
- Index cards prepared one each per class.
- Contains columns for:
  - Class responsibility
  - Collaborating objects

Class name	
Responsibility	Collaborator



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# CRC Cards

Cont...

- **Systematize development of interaction diagram for complex use cases.**
- Team members participate to determine:
  - The responsibility of classes involved during a use case execution

# CRC Cards cont...

- **Responsibility:**

- Method to be supported by the class.

- **Collaborator:**

- Class whose service (method) would have to be invoked

Class name	
Responsibility	Collaborator

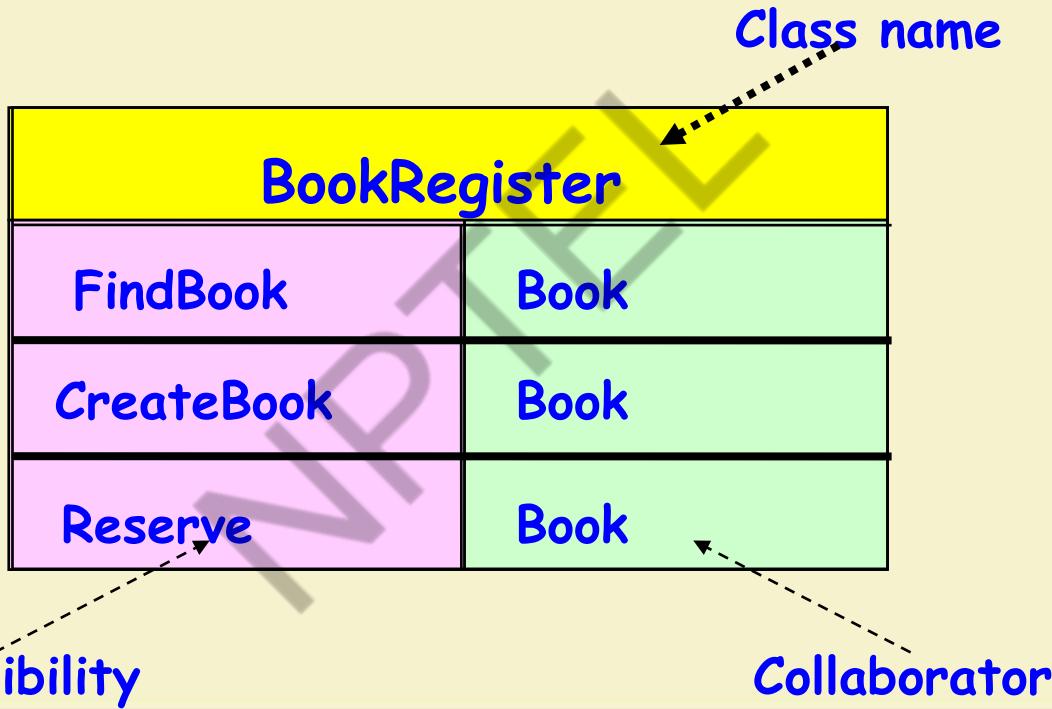


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# An Example: CRC Card for the BookRegister class



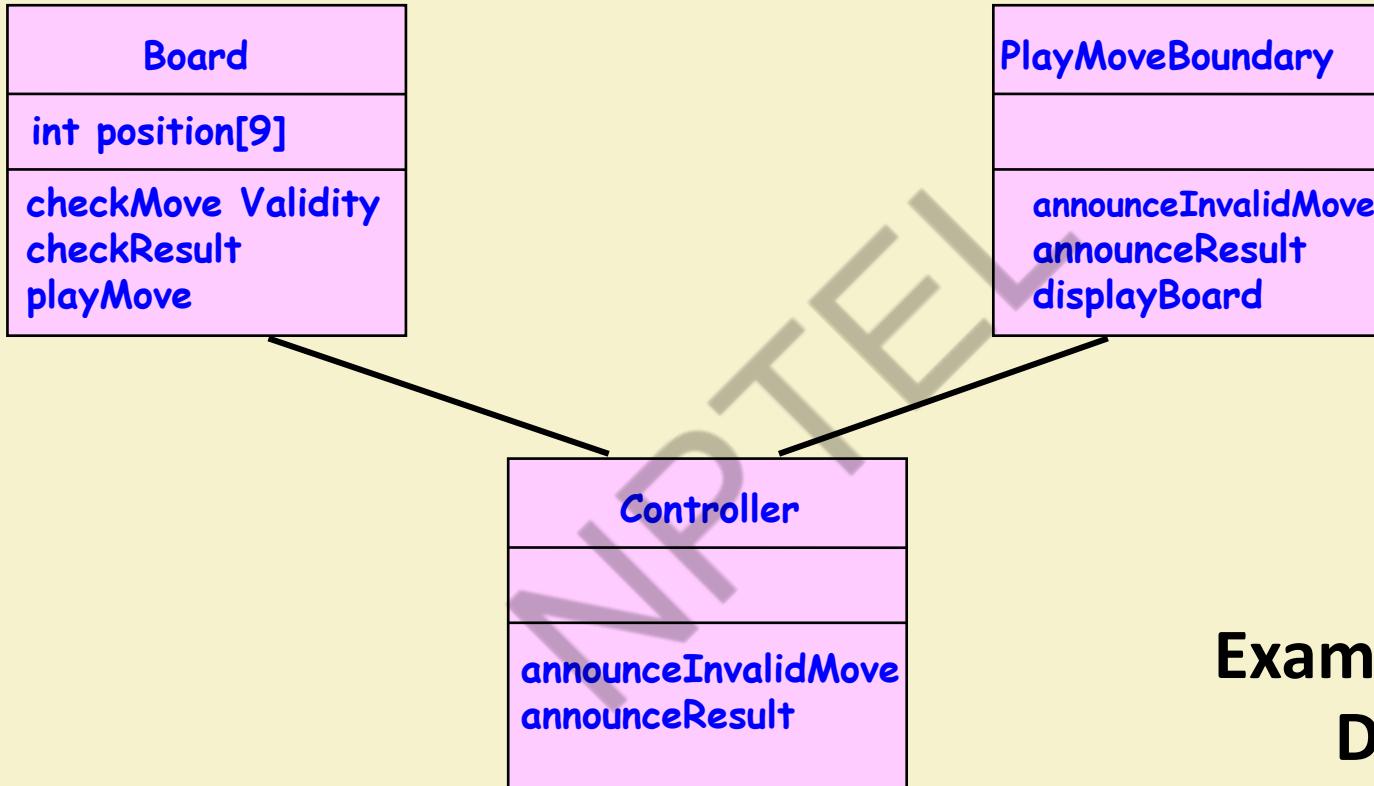
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Using CRC Cards

- After developing a set of CRC cards:
  - Run structured walkthrough scenarios
- Walkthrough of a scenario :
  - A class is responsible to perform some responsibilities
  - It may then pass control to a collaborator -- another class
  - You may discover missing responsibilities and classes



## Example 1: Class Diagram



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 2: Supermarket Prize Scheme

- Supermarket needs to develop software to encourage regular customers.
- Customer needs to supply his:
  - Residence address, telephone number, and the driving licence number.
- Each customer who registers is:
  - Assigned a unique customer number (CN) by the computer.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Example 2: Supermarket Prize Scheme

- A customer can present his CN to the staff when he makes any purchase.
- The value of his purchase is credited against his CN.
- At the end of each year:
  - The supermarket awards surprise gifts to ten customers who make highest purchase.

## Example 2: Supermarket Prize Scheme

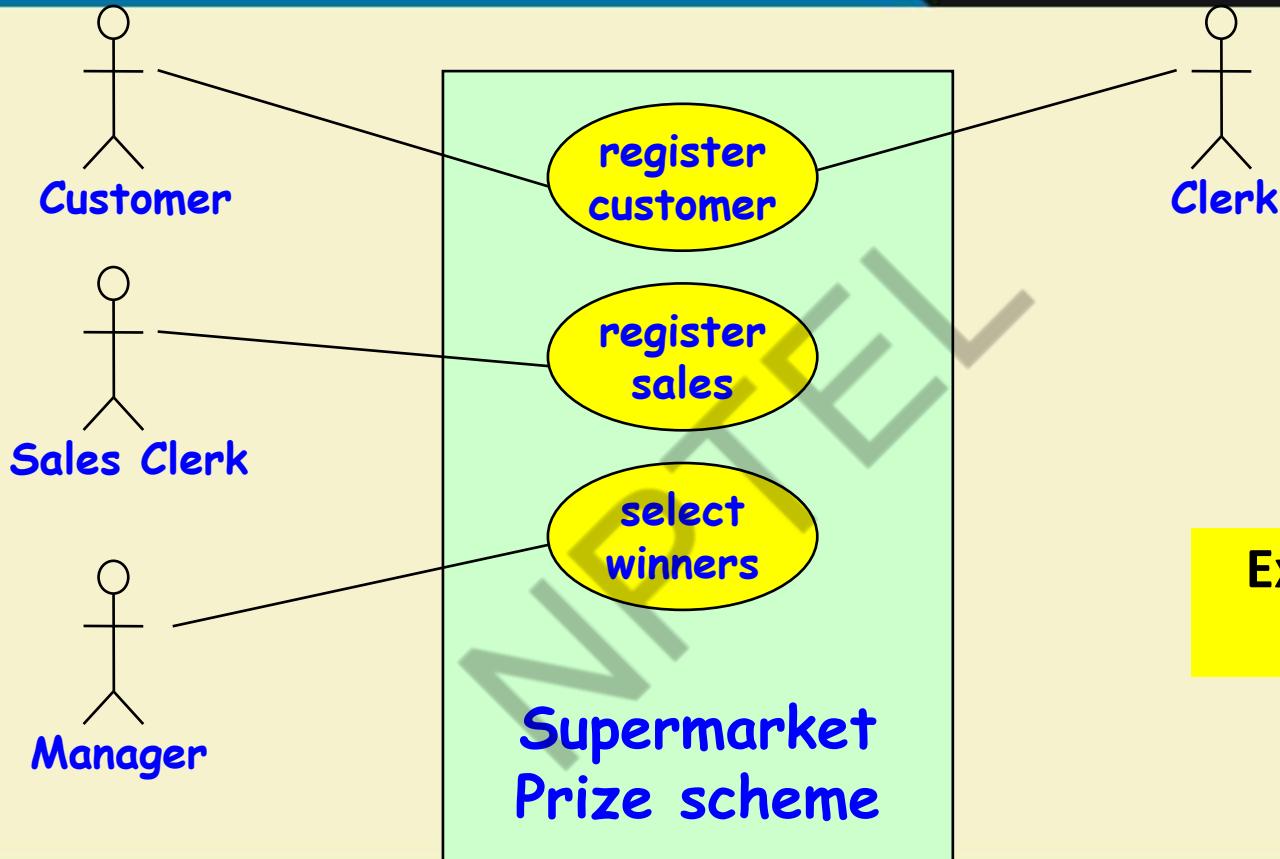
- It also, awards a 22 carat gold coin to every customer:
  - Whose purchases exceed Rs. 10,000.
- The entries against the CN are reset:
  - On the last day of every year after the prize winner's lists are generated.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



**Example 2: Use Case Model**

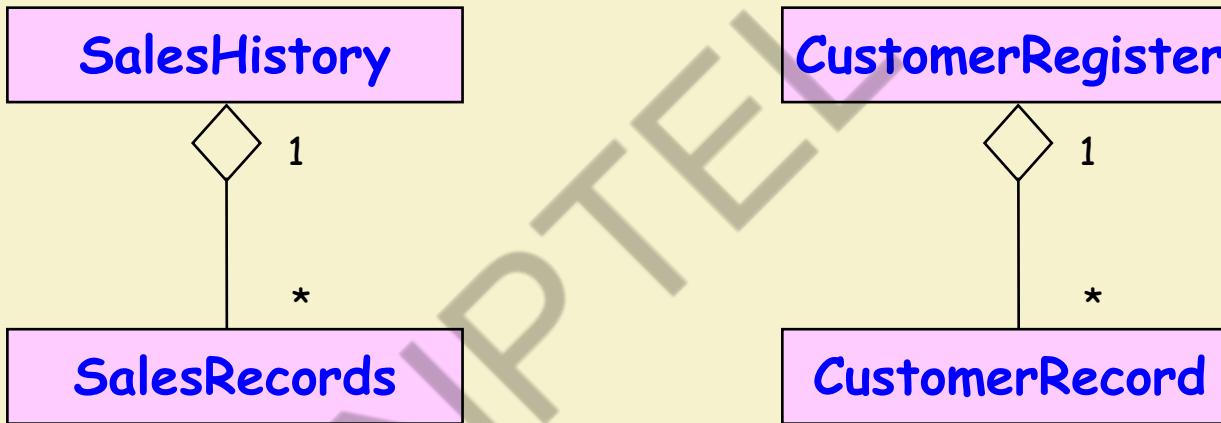


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Example 2: Initial Domain Model



Initial domain model

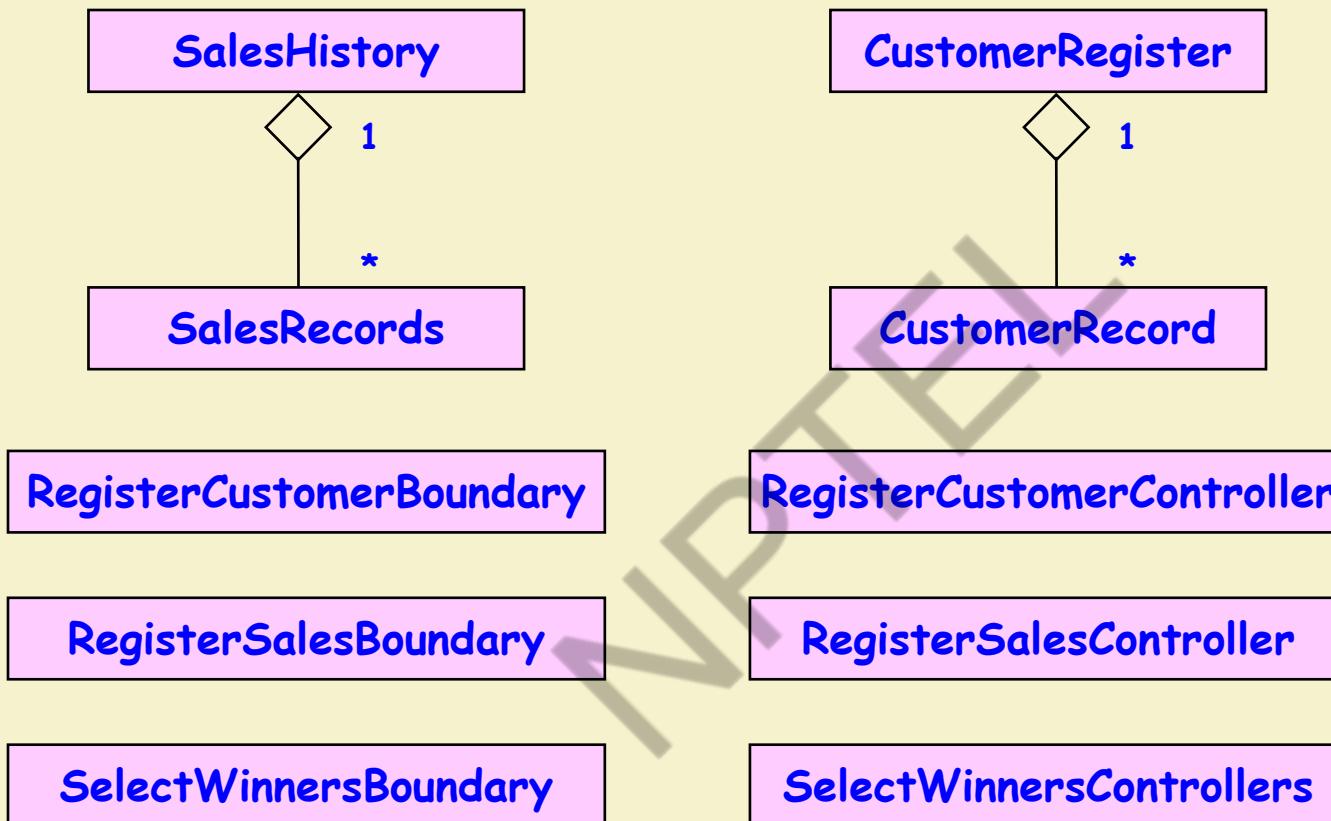


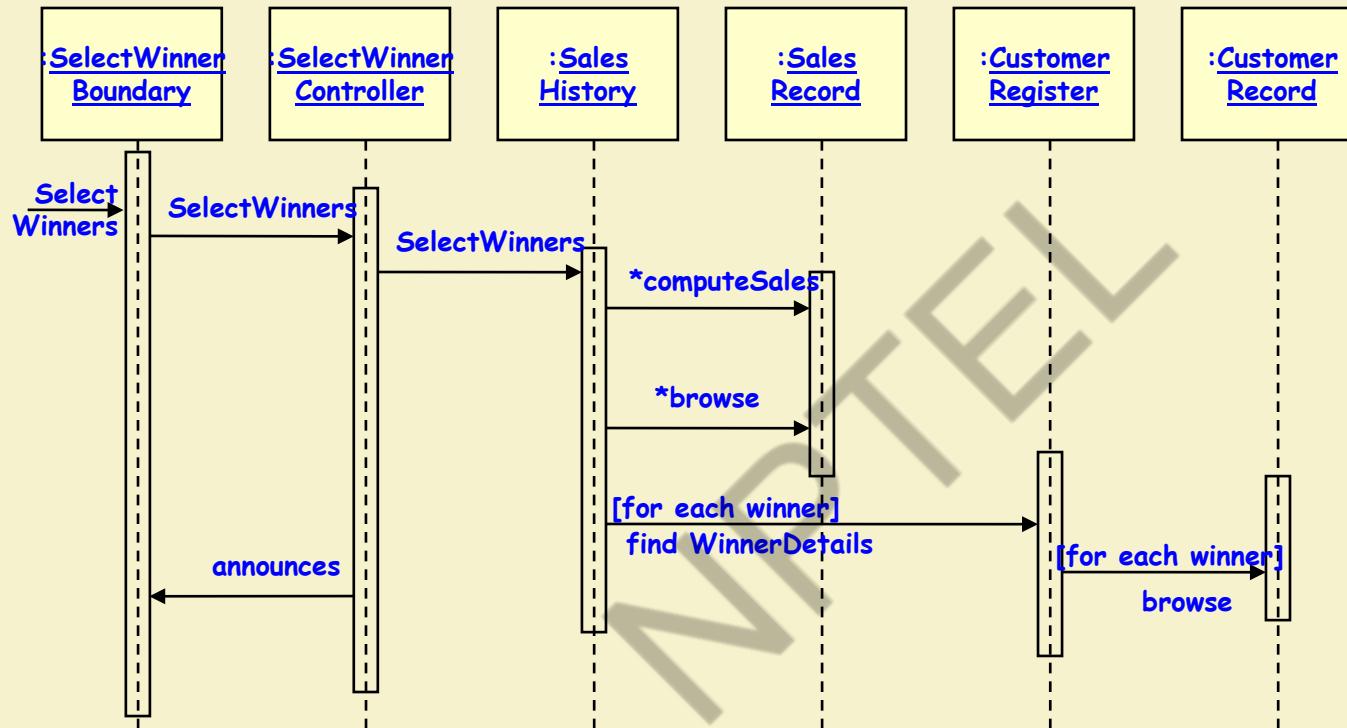
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

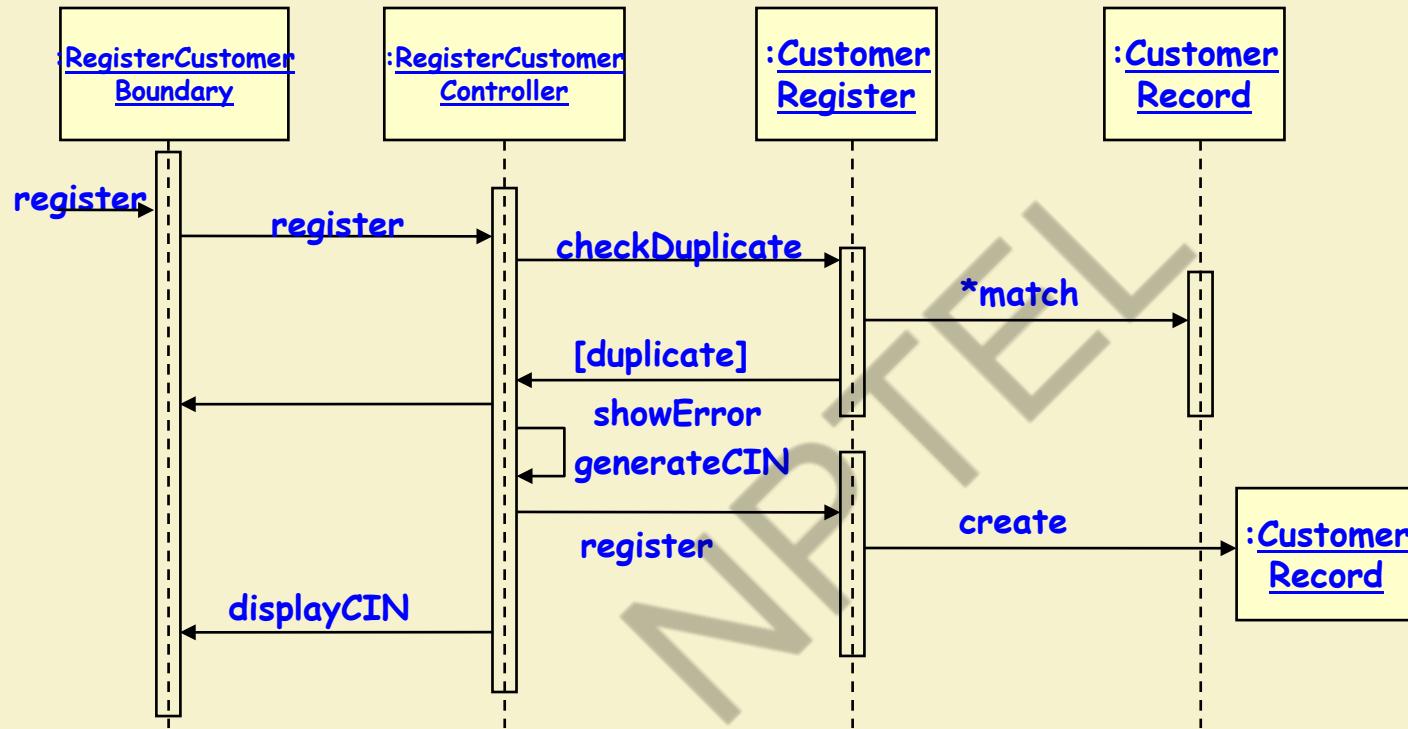
## Example 2: Refined Domain Model





Sequence Diagram for the select winners use case

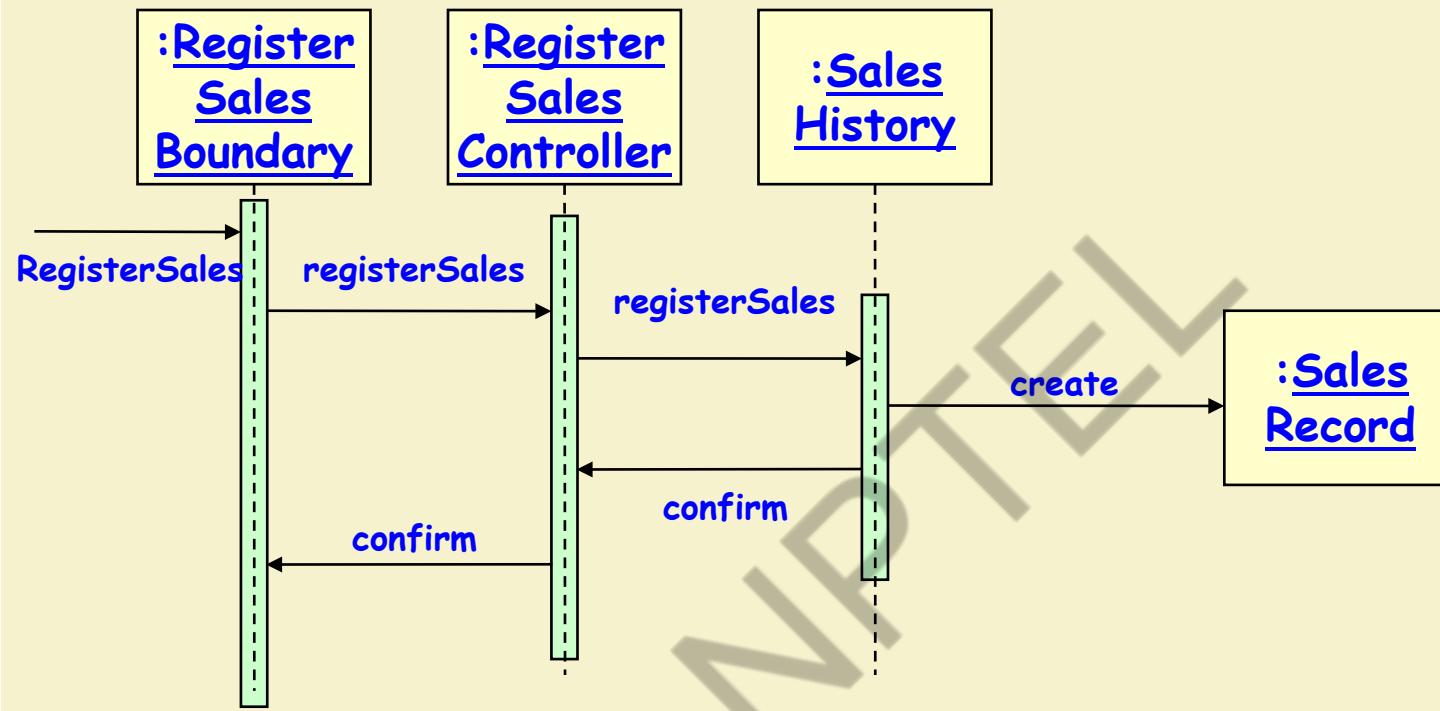
**Example 2:**  
**Sequence**  
**Diagram for the**  
**Select Winners**  
**Use Case**



Sequence Diagram for the register customer use case

**Example 2:**  
**Sequence**  
**Diagram for the**  
**Register**  
**Customer**  
**Case**

## Example 2: Sequence Diagram for the Register Sales Use Case



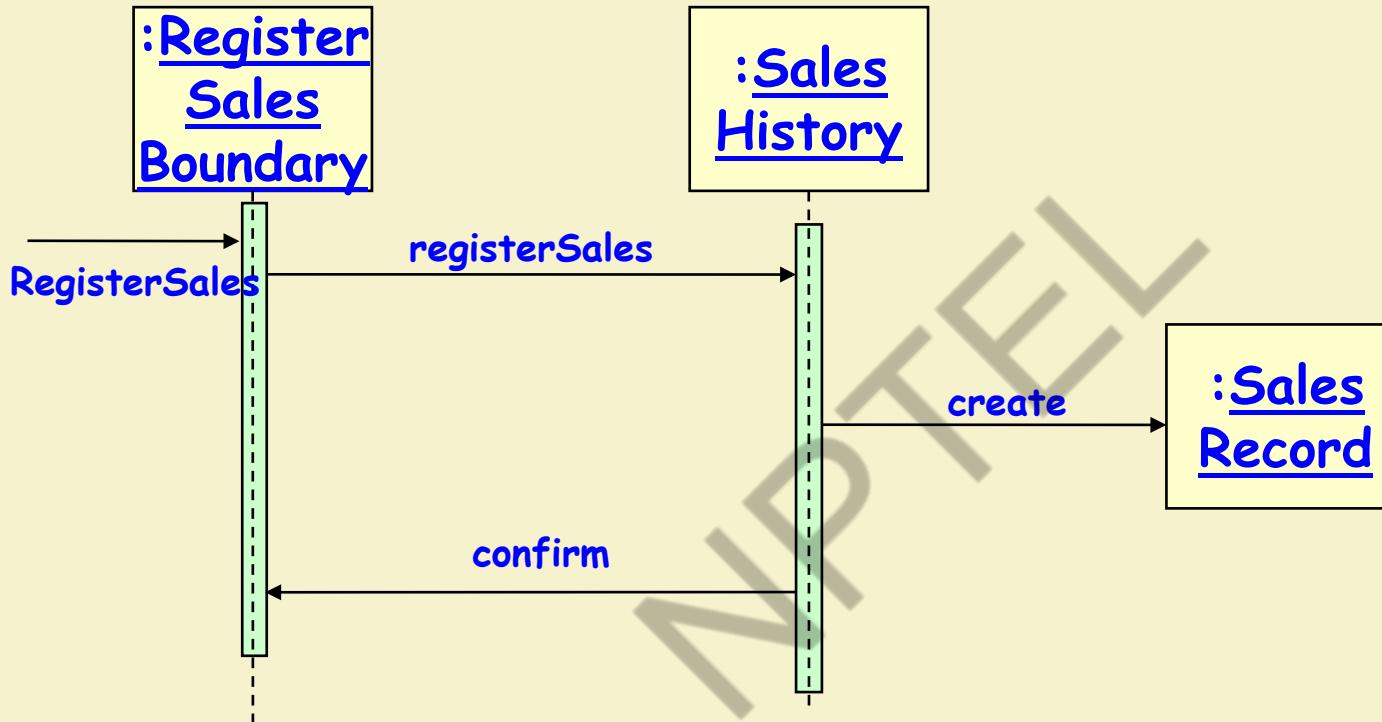
Sequence Diagram for the register sales use case



IIT KHARAGPUR



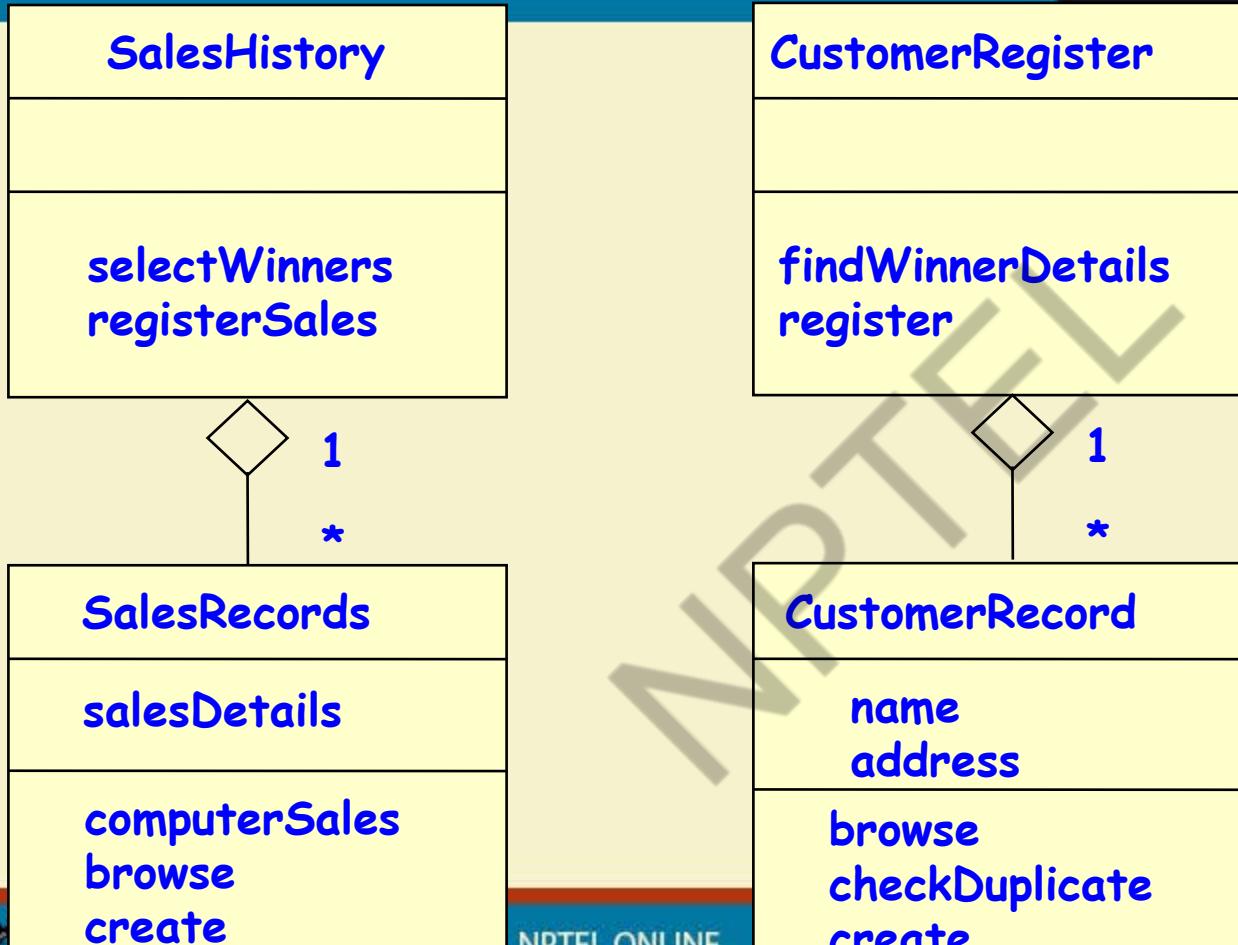
NPTEL  
ONLINE  
CERTIFICATION COURSES



Refined Sequence Diagram for the register sales use case

Example 2: Sequence Diagram for the Register Sales Use Case

## Example 2: Class Diagram



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Software Testing

Rajib Mall

CSE Department

IIT KHARAGPUR



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Faults and Failures

- A program may fail during testing:
  - A manifestation of a fault (also called defect or bug).
  - Mere presence of a fault may not lead to a failure.**



# Errors, Faults, Failures

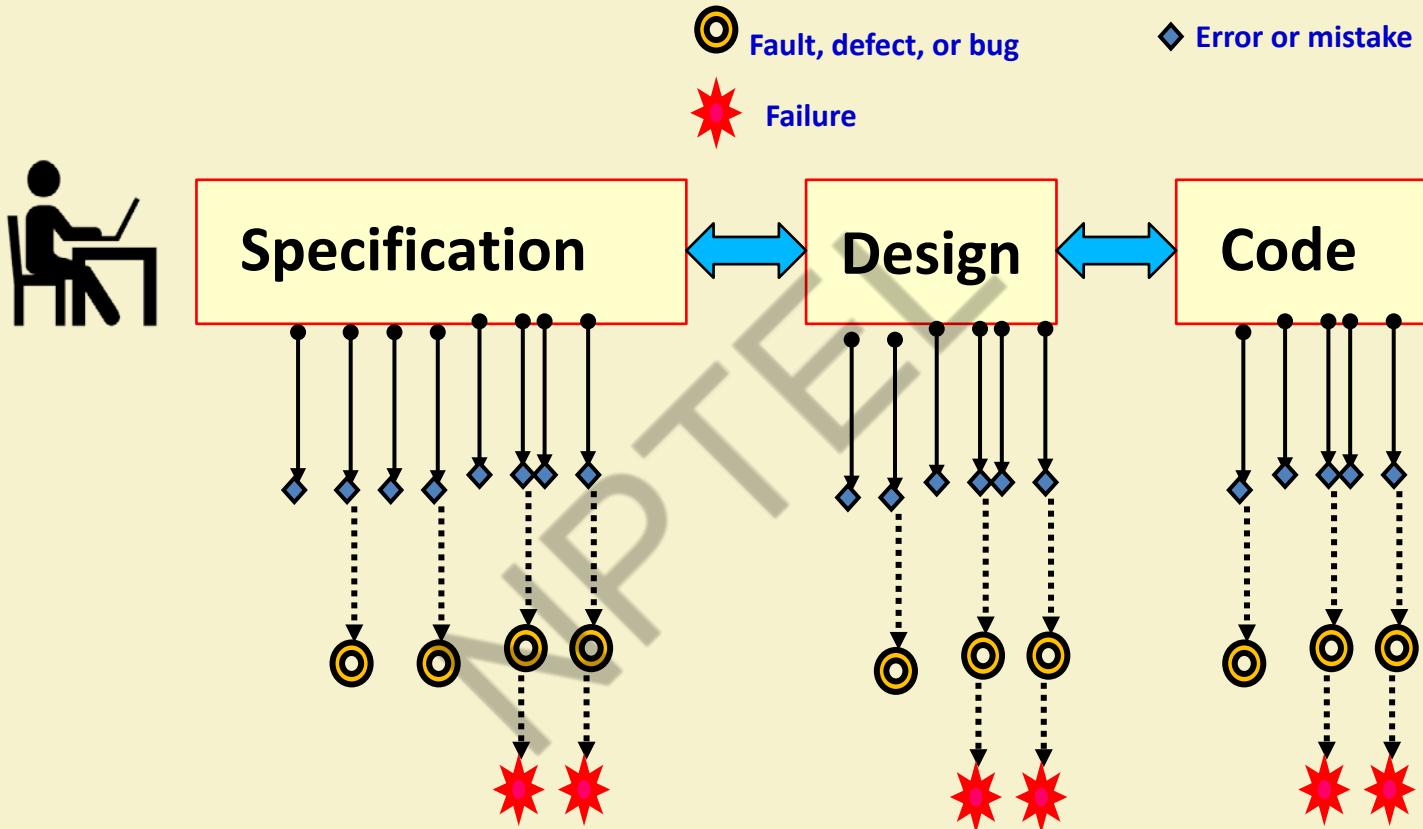
- Programming is human effort-intensive:
  - Therefore, inherently error prone.
- IEEE std 1044, 1993 defined errors and faults as synonyms :
- **IEEE Revision of std 1044 in 2010 introduced finer distinctions:**
  - To support more expressive communications, it distinguished between Errors and Faults



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES



# A Few Error Facts

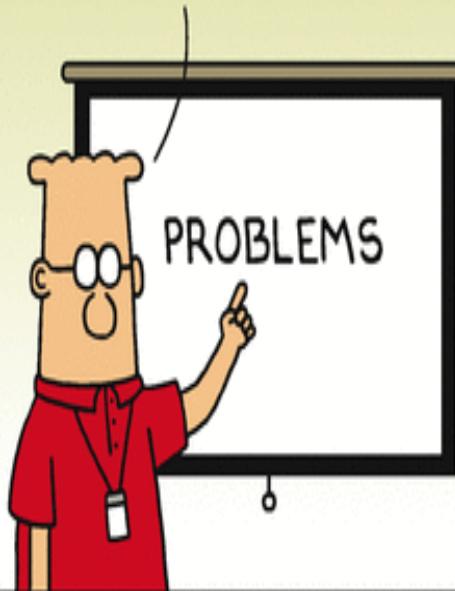
- Even experienced programmers make many errors:
  - Avg. 50 bugs per 1000 lines of source code
- Extensively tested software contains:
  - About 1 bug per 1000 lines of source code.
- Bug distribution:
  - 60% spec/design, 40% implementation.



Bug Source

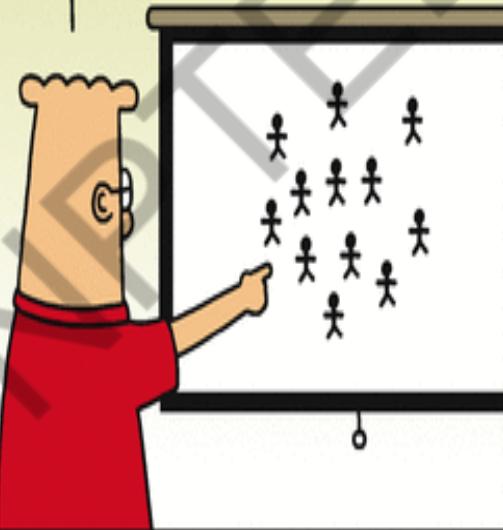


I FOUND THE  
ROOT CAUSE OF  
OUR PROBLEMS.



Dilbert.com DilbertCartoonist@gmail.com

IT'S  
PEOPLE.



4-24-15 © 2015 Scott Adams, Inc. /Dist. by Universal Uclick

THEY'RE  
BUGGY.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# How to Reduce Bugs?

- Review
- **Testing**
- Formal specification and verification
- Use of development process



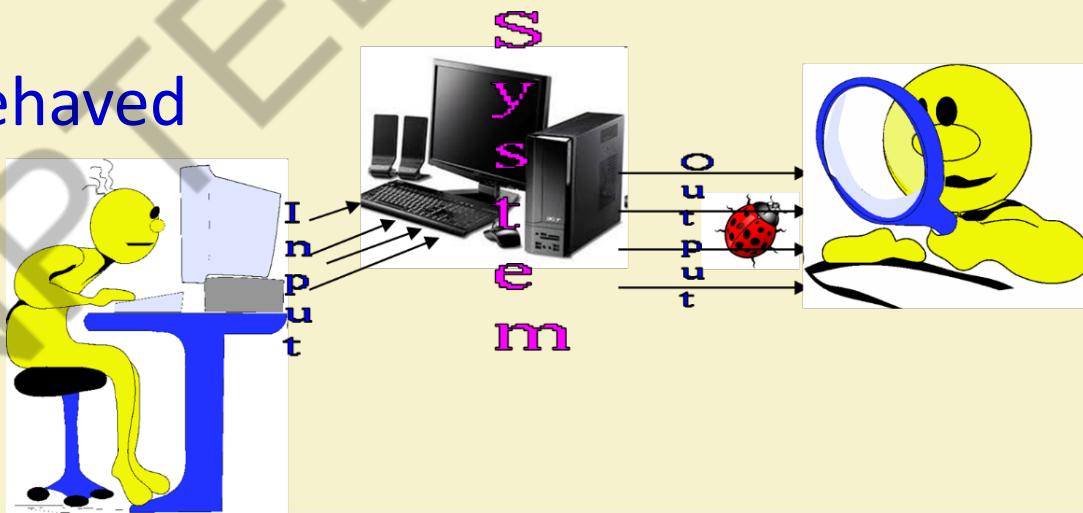
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# How to Test?

- Input test data to the program.
- Observe the output:
  - Check if the program behaved as expected.



# Examine Test Result...

- If the program does not behave as expected:
  - Note the conditions under which it failed (Test report).
  - Later debug and correct.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Testing Facts

- Consumes the largest effort among all development activities:
  - Largest manpower among all roles
  - Implies more job opportunities
- About 50% development effort
  - But 10% of development time?
  - How?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Testing Facts

- Testing is getting more complex and sophisticated every year.
  - Larger and more complex programs
  - Newer programming paradigms
  - Newer testing techniques
  - Test automation



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Testing Perception

- Testing is often viewed as not very challenging --- less preferred by novices, but:
  - Over the years testing has taken a center stage in all types of software development.
  - “**Monkey testing is passe**” --- Large number of innovations have taken place in testing area --- requiring tester to have good knowledge of test techniques.
  - Challenges of test automation



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Monkey Testing is Passe...



- Two types of monkeys:

- Dumb monkey



- Smart monkey



- Testing through random inputs.
- **Problems:**
  - Many program parts may not get tested.
  - Risky areas of a program may not get tested.
  - The tester may not be able to reproduce the failure.

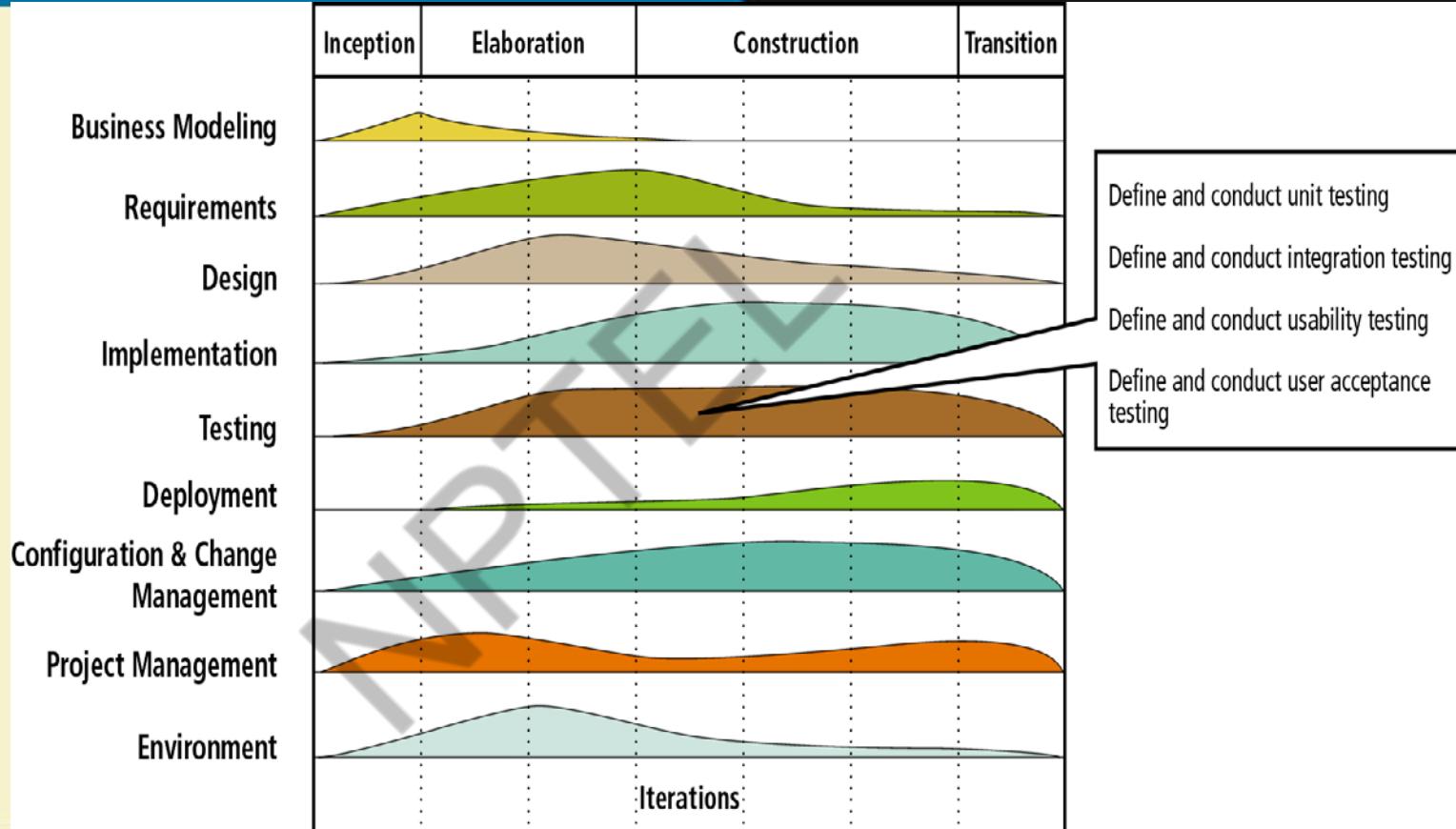


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

**Testing Activities Now Spread Over Entire Life Cycle**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Test How Long?

One way:



- Another way:
  - Seed bugs... run test cases
  - See if all (or most) are getting detected

# Verification versus Validation

- Verification is the process of determining:
  - Whether output of one phase of development conforms to its previous phase.
- Validation is the process of determining:
  - Whether a fully developed system conforms to its SRS document..



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Verification versus Validation

- Verification is concerned with phase containment of errors:
  - Whereas, the aim of validation is that the final product is error free.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Verification and Validation Techniques

- Review
- Simulation
- Unit testing
- Integration testing
- System testing



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Verification

Are you building it right?

Checks whether an artifact conforms to its previous artifact.

Done by developers.

Static and dynamic activities: reviews, unit testing.

## Validation

Have you built the right thing?

Checks the final product against the specification.

Done by Testers.

Dynamic activities: Execute software and check against requirements.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Testing Levels



IIT KHARAGPUR

9/18/2018



NPTEL  
ONLINE  
CERTIFICATION COURSES

52

# 4 Testing Levels

- Software tested at 4 levels:
  - Unit testing
  - Integration testing
  - System testing
  - Regression testing



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Test Levels

- **Unit testing**
  - Test each module (unit, or component) independently
  - **Mostly done by developers of the modules**
- **Integration and system testing**
  - Test the system as a whole
  - **Often done by separate testing or QA team**
- **Acceptance testing**
  - **Validation of system functions by the customer**

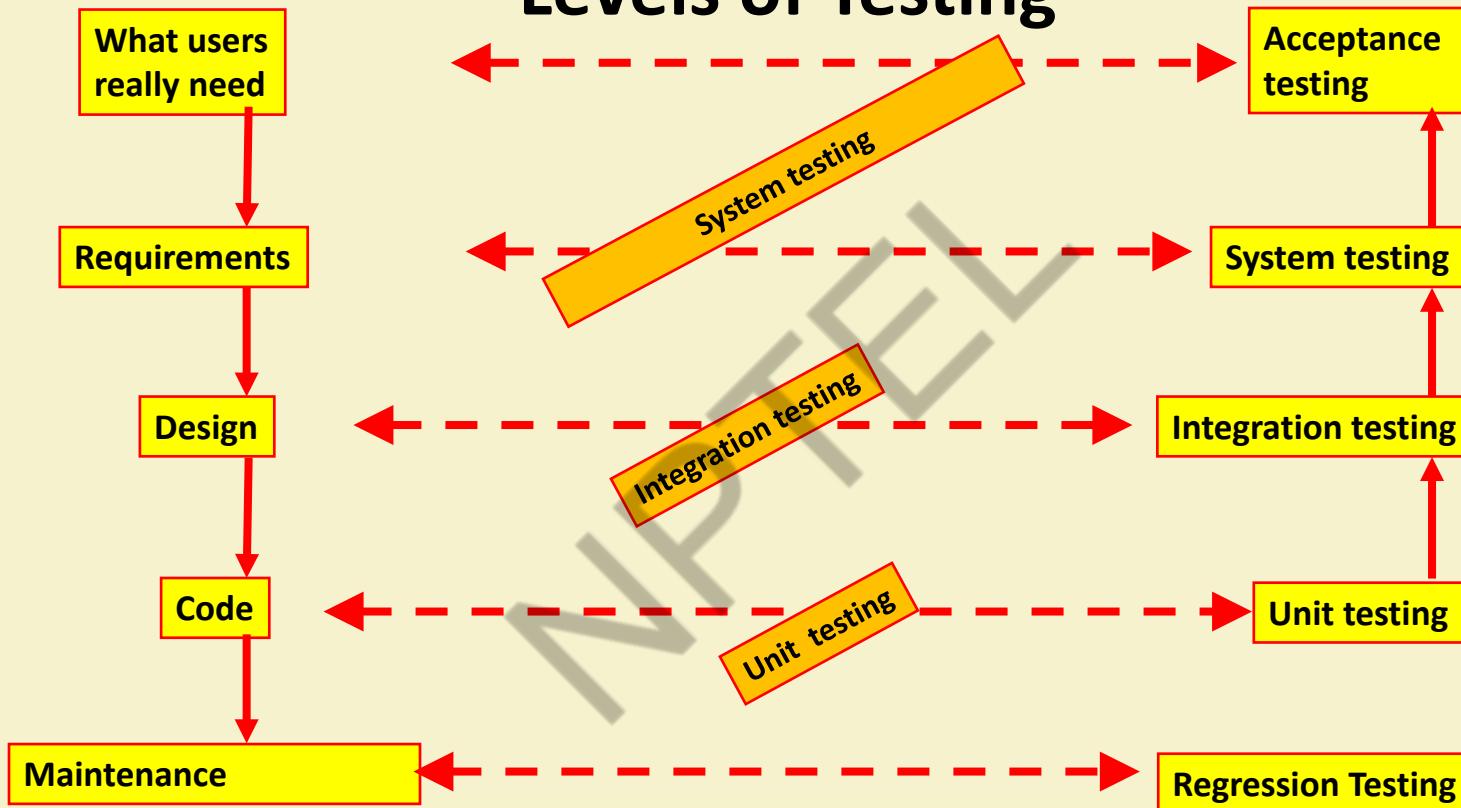


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Levels of Testing



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Overview of Activities During System and Integration Testing

- Test Suite Design
  - Run test cases
  - Check results to detect failures.
  - Prepare failure list
  - Debug to locate errors
  - Correct errors.
- 
- The diagram illustrates the activities of system and integration testing. A large bracket on the right side groups the activities into two categories: 'Tester' (top) and 'Developer' (bottom). The 'Tester' category includes 'Test Suite Design', 'Run test cases', 'Check results to detect failures.', 'Prepare failure list', and 'Correct errors.' The 'Developer' category includes 'Debug to locate errors.'

## Quiz 1

- As testing proceeds more and more bugs are discovered.
  - How to know when to stop testing?
- Give examples of the types of bugs detected during:
  - Unit testing?
  - Integration testing?
  - System testing?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Unit testing

- During unit testing, functions (or modules) are tested in isolation:
  - What if all modules were to be tested together (i.e. system testing)?
  - It would become difficult to determine which module has the error.



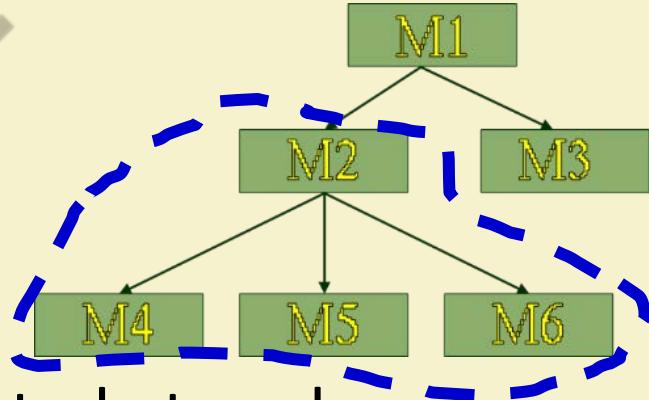
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Integration Testing

- After modules of a system have been coded and unit tested:
  - Modules are integrated in steps according to an integration plan
  - The partially integrated system is tested at each integration step.



# Integration and System Testing

- **Integration test evaluates a group of functions or classes:**
  - Identifies interface compatibility, unexpected parameter values or state interactions, and run-time exceptions
  - **System test tests working of the entire system**
- **Smoke test:**
  - System test performed daily or several times a week after every build.

# Types of System Testing

- Based on types test:
  - **Functionality test**
  - **Performance test**
- Based on who performs testing:
  - **Alpha**
  - **Beta**
  - **Acceptance test**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Performance test

- Determines whether a system or subsystem meets its non-functional requirements:
  - Response times
  - Throughput
  - Usability
  - Stress
  - Recovery
  - Configuration, etc.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# User Acceptance Testing

- User determines whether the system fulfills his requirements
  - **Accepts or rejects delivered system based on the test results.**



IIT KHARAGPUR

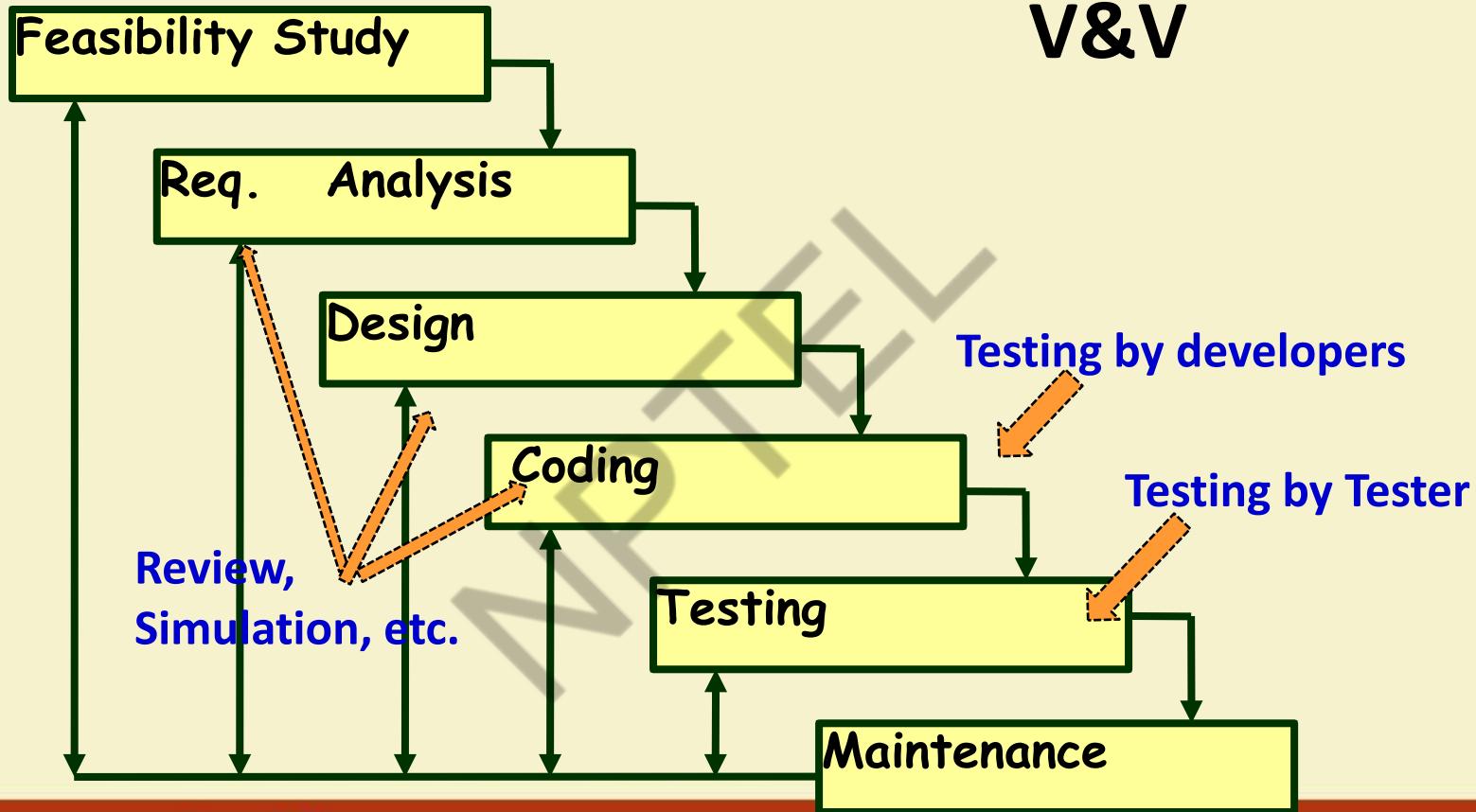


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Who Tests Software?

- **Programmers:**
  - Unit testing
  - Test their own or other's programmer's code
- **Users:**
  - Usability and acceptance testing
  - Volunteers are frequently used to test beta versions
- **Test team:**
  - All types of testing except unit and acceptance
  - Develop test plans and strategy

# V&V



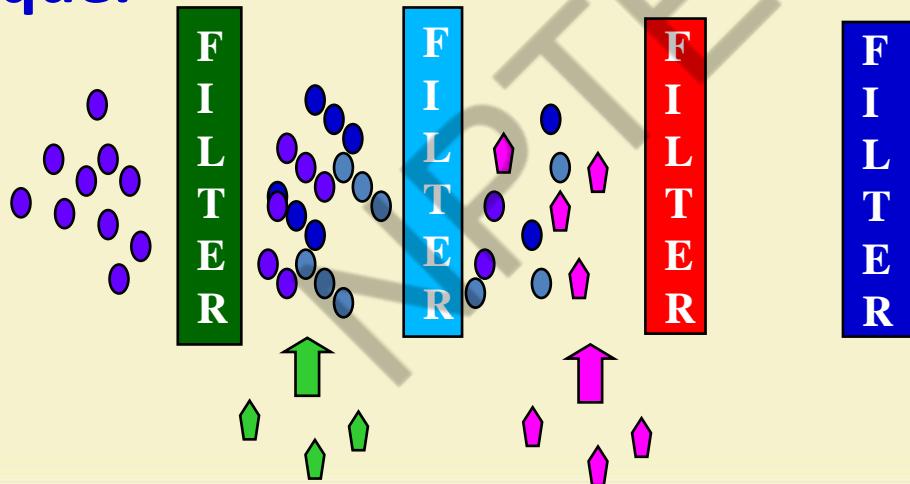
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Pesticide Effect

- Errors that escape a fault detection technique:
  - Can not be detected by further applications of that technique.



# Capers Jones Rule of Thumb

- Each of software review, inspection, and test step will find 30% of the bugs present.

In IEEE Computer, 1996



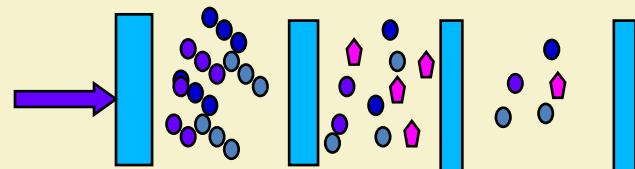
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Pesticide Effect

- Assume to start with 1000 bugs
- We use 4 fault detection techniques :
  - Each detects only 70% bugs existing at that time
  - How many bugs would remain at end?
  - $1000 * (0.3)^4 = 81$  bugs

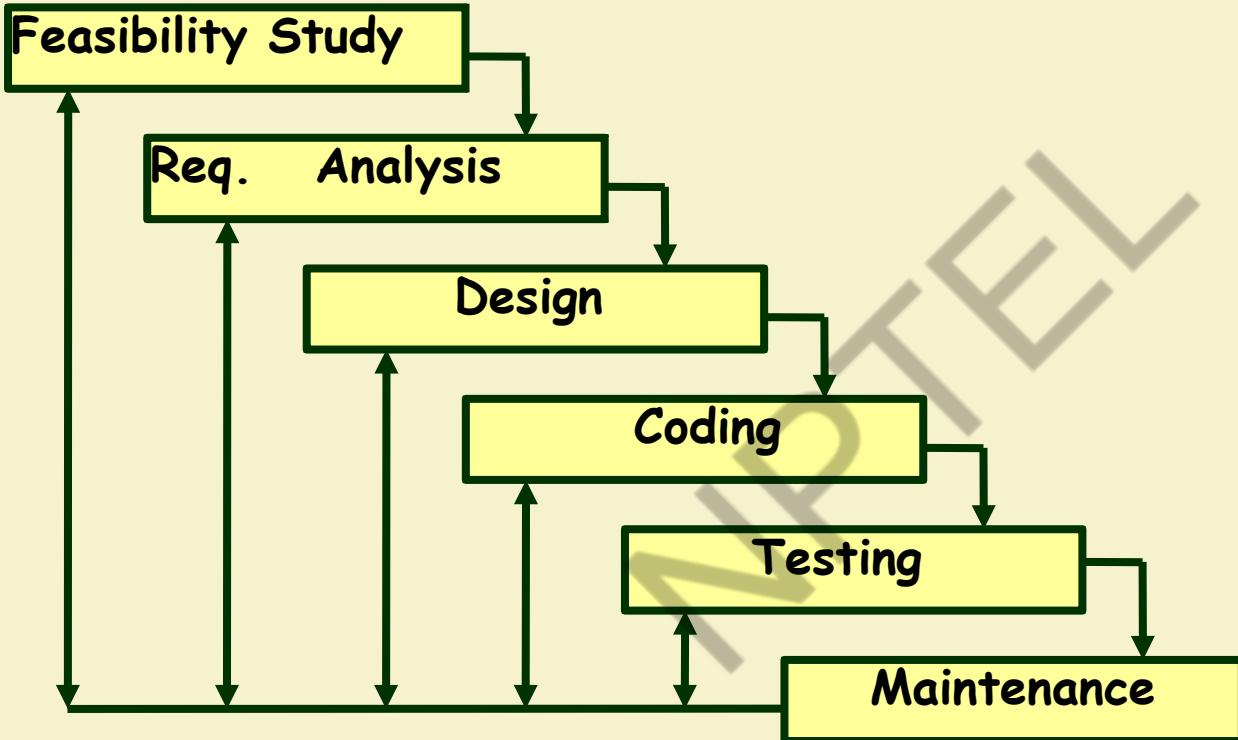


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Quiz



1. When are verification undertaken in waterfall model?
2. When is testing undertaken in waterfall model?
3. When is validation undertaken in waterfall model?

# Basic Concepts in Testing



IIT KHARAGPUR

9/18/2018



NPTEL  
ONLINE  
CERTIFICATION COURSES

70

- Several independent studies [Jones], [schroeder], etc. conclude:
  - 85% errors get removed at the end of a typical testing process.
  - Why not more?
  - All practical test techniques are basically heuristics... they help to reduce bugs... but do not guarantee complete bug removal...

## How Many Latent Errors?

# Test Cases

- Each test case typically tries to establish correct working of some functionality:
  - Executes (covers) some program elements.
  - For certain restricted types of faults, fault-based testing can be used.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Test data versus test cases

- **Test data:**

- Inputs used to test the system

- **Test cases:**

- Inputs to test the system,
  - State of the software, and
  - The predicted outputs from the inputs

# Test Cases and Test Suites

- A **test case** is a triplet [I,S,O]
  - I is the data to be input to the system,
  - S is the state of the system at which the data will be input,
  - O is the expected output of the system.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Test Cases and Test Suites

- Test a software using a set of carefully designed test cases:
  - The set of all test cases is called the **test suite**.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# What are Negative Test Cases?

- **Purpose:**
  - Helps to ensure that the application gracefully handles invalid and unexpected user inputs and the application does not crash.
- **Example:**
  - If user types letter in a numeric field, it should not crash but politely display the message: **“incorrect data type, please enter a number...”**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Test Execution Example: Return Book

- **Test case [I,S,O]**

- 1. Set the program in the required state:** Book record created, member record created, Book issued
- 2. Give the defined input:** Select renew book option and request renew for a further 2 week period.
- 3. Observe the output:**
  - Compare it to the expected output.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

**Test Case number**

## Sample: Recording of Test Case & Results

**Test Case author**

**Test purpose**

**Pre-condition:**

**Test inputs:**

**Expected outputs (if any):**

**Post-condition:**

**Test Execution history:**

**Test execution date**

**Person executing Test**

**Test execution result (s) : Pass/Fail**

**If failed : Failure information and fix status**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Test Team- Human Resources

- **Test Planning:** Experienced people
- **Test scenario and test case design:** Experienced and test qualified people
- **Test execution:** semi-experienced to inexperienced
- **Test result analysis:** experienced people
- **Test tool support:** experienced people
- May include external people:
  - **Users**
  - **Industry experts**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Why Design of Test Cases?

- Exhaustive testing of any non-trivial system is impractical:
  - Input data domain is extremely large.
- Design an **optimal test suite**, meaning:
  - Of reasonable size, and
  - Uncovers as many errors as possible.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Design of Test Cases

- If test cases are selected randomly:
  - Many test cases would not contribute to the significance of the test suite,
  - Would only detect errors that are already detected by other test cases in the suite.
- Therefore, the number of test cases in a randomly selected test suite:
  - Does not indicate the effectiveness of testing.

# Design of Test Cases

- Testing a system using a large number of randomly selected test cases:
  - Does not mean that most errors in the system will be uncovered.
- Consider following example:
  - Find the maximum of two integers x and y.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Design of Test Cases

- The code has a simple programming error:
- **If (x>y) max = x;**  
**else max = x; // should be max=y;**
- Test suite **{(x=3,y=2);(x=2,y=3)}** can detect the bug,
- A larger test suite **{(x=3,y=2);(x=4,y=3); (x=5,y=1)}** does not detect the bug.

- Before testing activities start, a test plan is developed.
- The test plan documents the following:
  - Features to be tested
  - Features not to be tested
  - Test strategy
  - Test suspension criteria
  - stopping criteria
  - Test effort
  - Test schedule

**Test Plan**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Software Testing

Rajib Mall

CSE Department

IIT KHARAGPUR



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Design of Test Cases

- Systematic approaches are required to design an effective test suite:
  - Each test case in the suite should target different faults.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Testing Strategy

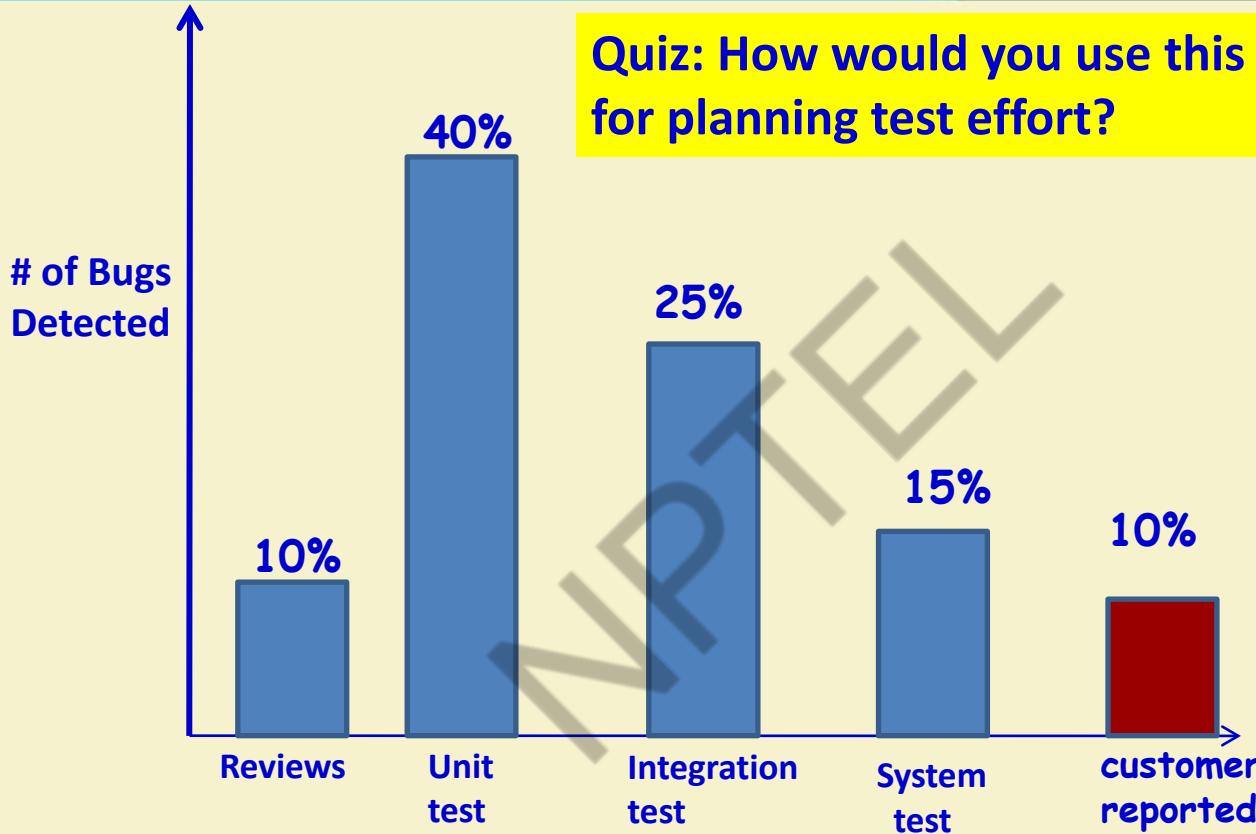
- Test Strategy primarily addresses:
  - **Which types of tests to deploy?**
  - **How much effort to devote to which type of testing?**
    - **Black-box: Usage-based testing** (based on customers' actual usage pattern)
    - **White-box testing** can be guided by black box testing results



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



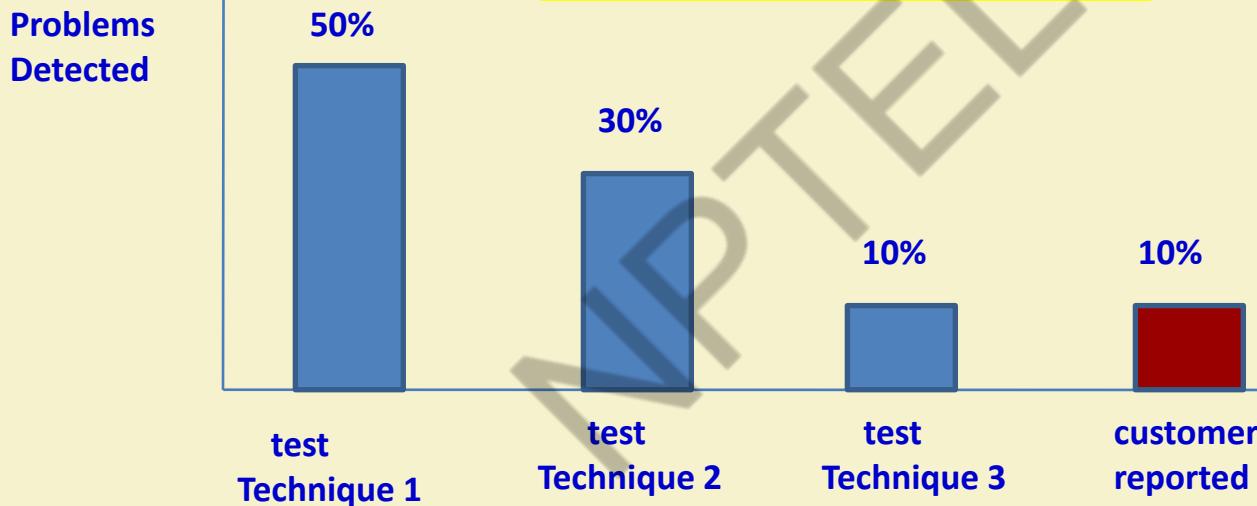
Considering  
Past Bug  
Detection  
Data...



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES



Quiz: How would you use this for planning unit test effort?

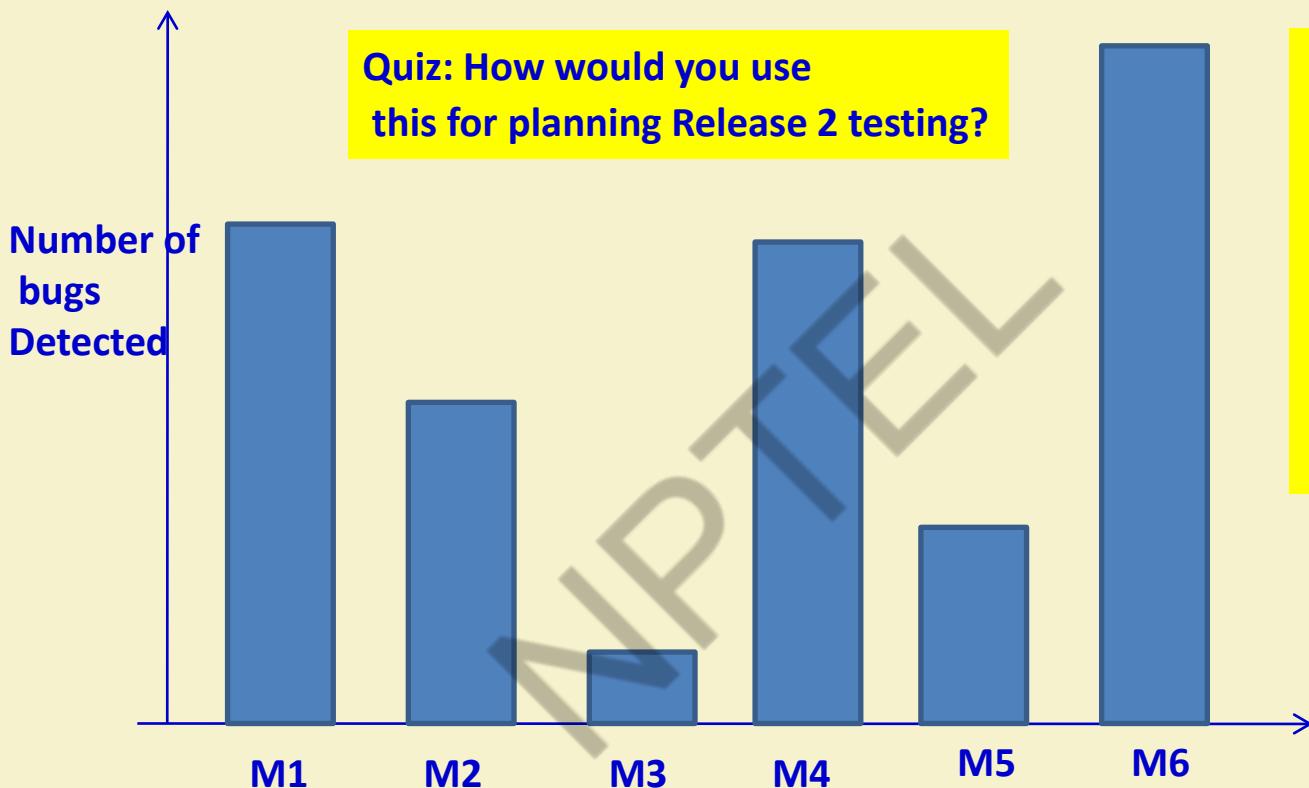
Consider Past Bug Detection Data...



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



Distribution of Error Prone Modules customer reported bugs for Release 1



IIT KHARAGPUR

Defect clustering: A few modules usually contain most defects...



NPTEL ONLINE  
CERTIFICATION COURSES

# Unit Testing



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# When and Why of Unit Testing?

- Unit testing carried out:
  - After coding of a unit is complete and it compiles successfully.
- Unit testing reduces debugging effort substantially.



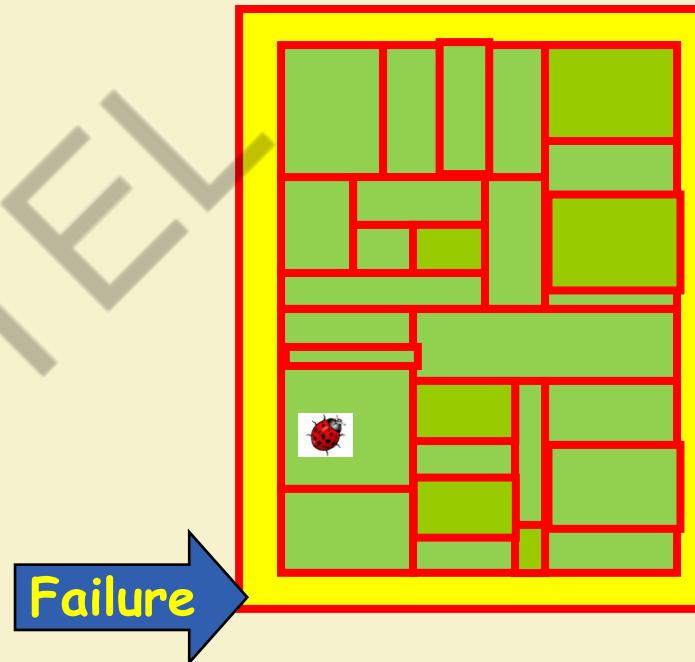
IIT KHARAGPUR



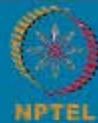
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Why unit test?

- Without unit test:
  - Errors become difficult to track down.
  - Debugging cost increases substantially...



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## • Testing of individual methods, modules, classes, or components in isolation:

### Unit Testing

- Carried out before integrating with other parts of the software being developed.

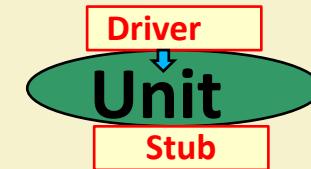
## • Following support required for Unit testing:

### – Driver

- Simulates the behavior of a function that calls and supplies necessary data to the function being tested.

### – Stub

- Simulates the behavior of a function that has not yet been written.

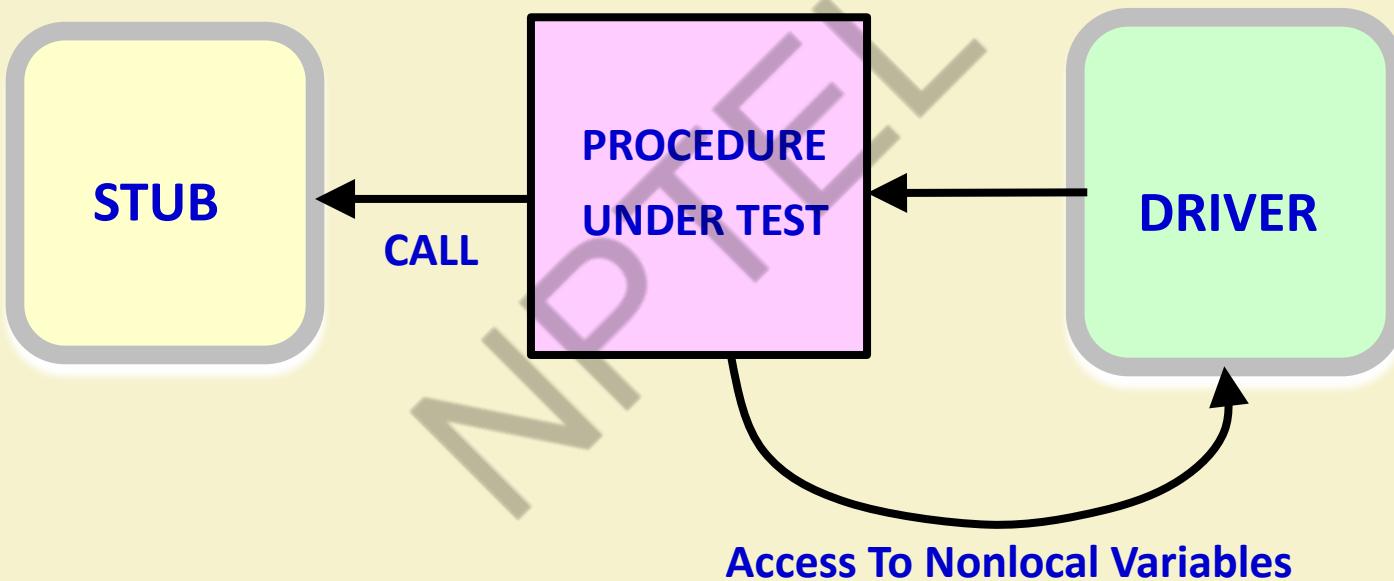


IIT KHARAGPUR

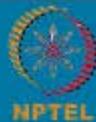


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Unit Testing



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Quiz

- Unit testing can be considered as which one of the following types of activities?
  - **Verification**
  - **Validation**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Design of Unit Test Cases

- There are essentially three main approaches to design test cases:
  - Black-box approach
  - White-box (or glass-box) approach
  - Grey-box approach

# Black-Box Testing

- Test cases are designed using only **functional specification** of the software:

—Without any knowledge of the internal structure of the software.



- Black-box testing is also known as **functional testing**.

# What is Hard about BB Testing

- Data domain is large
- A function may take multiple parameters:
  - We need to consider the combinations of the values of the different parameters.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# What's So Hard About Testing?

- Consider **int check-equal(int x, int y)**
- Assuming a 64 bit computer
  - Input space =  $2^{128}$
- Assuming it takes 10secs to key-in an integer pair:
  - It would take about a billion years to enter all possible values!
  - Automatic testing has its own problems!



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Solution

- Construct model of the data domain:
  - Called Domain based testing
  - Select data based on the domain model



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# White-box Testing

- To design test cases:
  - Knowledge of internal structure of software necessary.
  - White-box testing is also called structural testing.



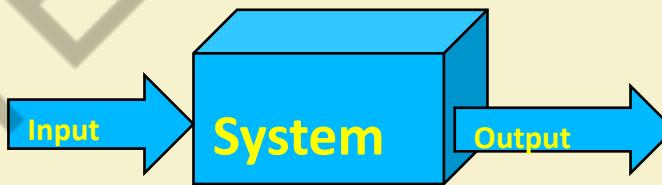
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Black Box Testing

- Software considered as a black box:
  - Test data derived from the specification
    - No knowledge of code necessary
- Also known as:
  - Data-driven or
  - Input/output driven testing
- The goal is to achieve the thoroughness of exhaustive input testing:
  - With much less effort!!!!



## Black-Box Testing

- Scenario coverage
- Equivalence class partitioning
- Boundary value testing
- Cause-effect (Decision Table) testing
- Combinatorial testing
- Orthogonal array testing

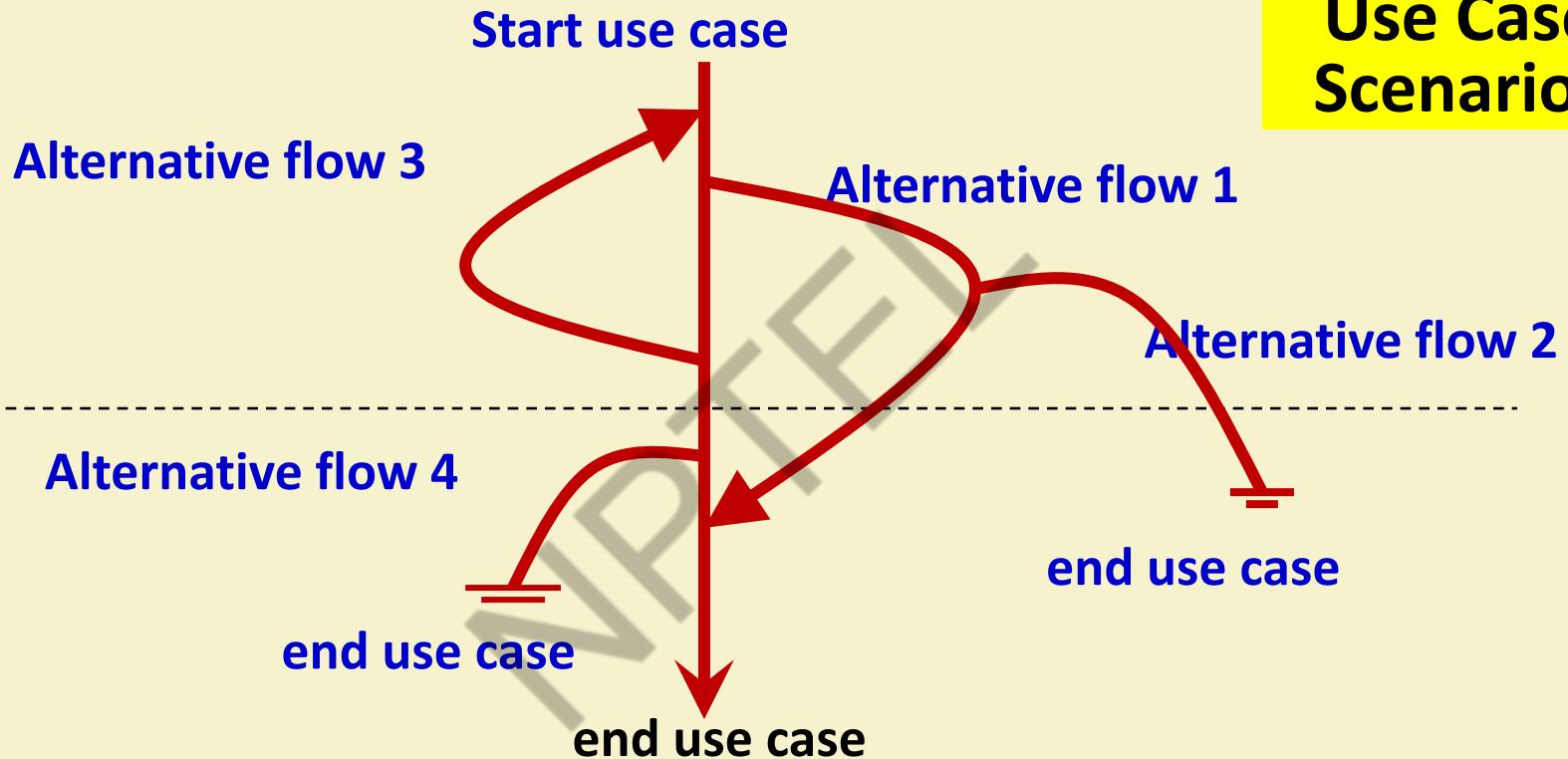


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Use Case Scenarios



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Deriving test cases from use cases

1. Identify the use case scenarios
2. For each scenario, identify one or more test cases
3. For each test case, identify the conditions that will cause it to execute.
4. Complete the test case by adding data values



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

Scenario number	Originating flow	Alternative flow	Next alternative	Next alternative
1	<b>Basic flow</b>			
2	<b>Basic flow</b>	<b>Alt. flow 1</b>		
3	<b>Basic flow</b>	<b>Alt. flow 1</b>	<b>Alt. flow 2</b>	
4	<b>Basic flow</b>	<b>Alt. flow 3</b>		
5	<b>Basic flow</b>	<b>Alt. flow 3</b>	<b>Alt. flow 1</b>	
6	<b>Basic flow</b>	<b>Alt. flow 3</b>	<b>Alt. flow 1</b>	<b>Alt. flow 2</b>
7	<b>Basic flow</b>	<b>Alt. flow 4</b>		
8	<b>Basic flow</b>	<b>Alt. flow 3</b>	<b>Alt. flow 4</b>	

**Identify  
use case  
scenarios:  
Example**

## Identify the test cases

- Parameters of any test case:
  - Conditions
  - Input (data values)
  - Expected result
  - Actual result

Test case ID	Scenario/ condition	Data value 1	Data value 2	Data value N	Exp. results	Actual results
1	Scenario 1					
2	Scenario 2					
3	Scenario 3					



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Equivalence Class Testing



IIT KHARAGPUR

03/08/10

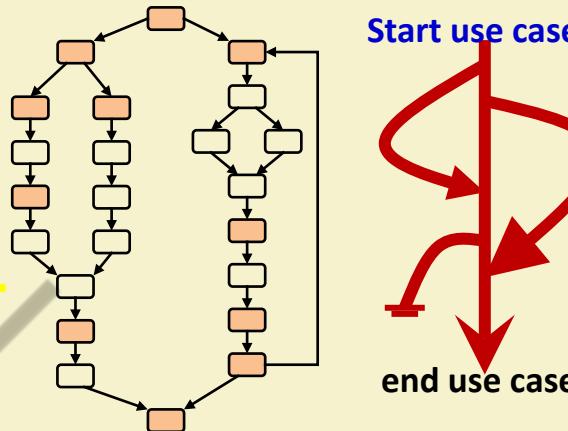


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

25

# Equivalence Class Partitioning

- The input values to a program:
  - Partitioned into equivalence classes.
- Partitioning is done such that:
  - Program behaves in similar ways to every input value belonging to an equivalence class.
  - At the least, there should be as many equivalence classes as scenarios.

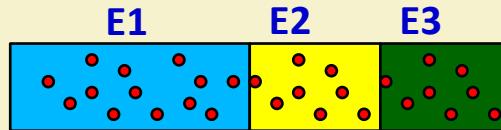


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Why Define Equivalence Classes?



- Premise:
  - Testing code with any one representative value from a equivalence class:
  - As good as testing using any other values from the equivalence class.

# Equivalence Class Partitioning

- How do you identify equivalence classes?
  - Identify scenarios
  - Examine the input data.
  - Examine output
- Few guidelines for determining the equivalence classes can be given...



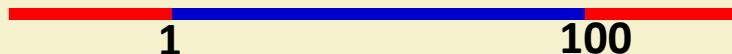
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- If an input is a range, one valid and two invalid equivalence classes are defined.

Example: 1 to 100



- If an input is a set, one valid and one invalid equivalence classes are defined. Example: {a,b,c}
- If an input is a Boolean value, one valid and one invalid class are defined.

### Example:

- **Area code:** input value defined between 10000 and 90000--- **range**
- **Password:** string of six characters --- **set**

**Guidelines  
to Identify  
Equivalence  
Classes**

## Equivalent class partition: Example

- Given three sides, determine the type of the triangle:
  - Isosceles
  - Scalene
  - Equilateral, etc.
- Hint: scenarios correspond to output in this case.



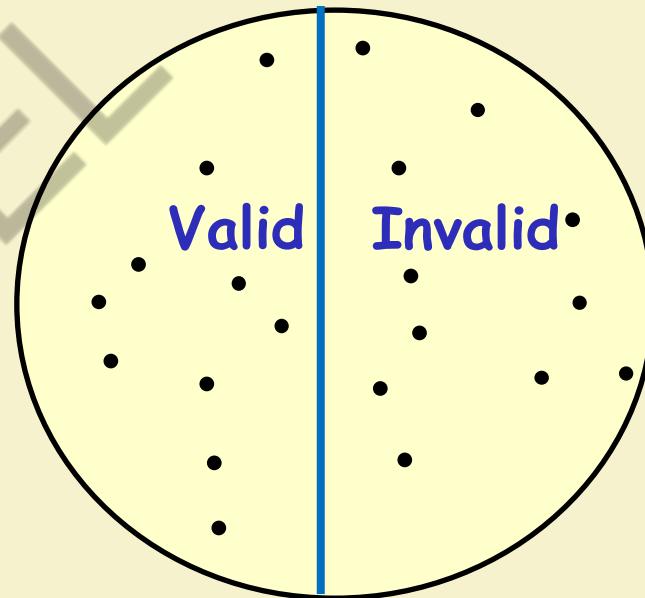
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

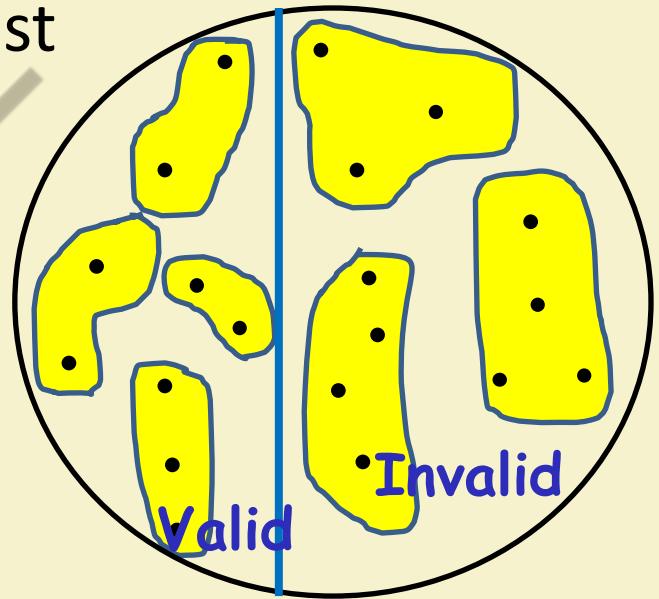
# Equivalence Partitioning

- First-level partitioning:
  - Valid vs. Invalid test cases



# Equivalence Partitioning

- Further partition valid and invalid test cases into equivalence classes



IIT KHARAGPUR

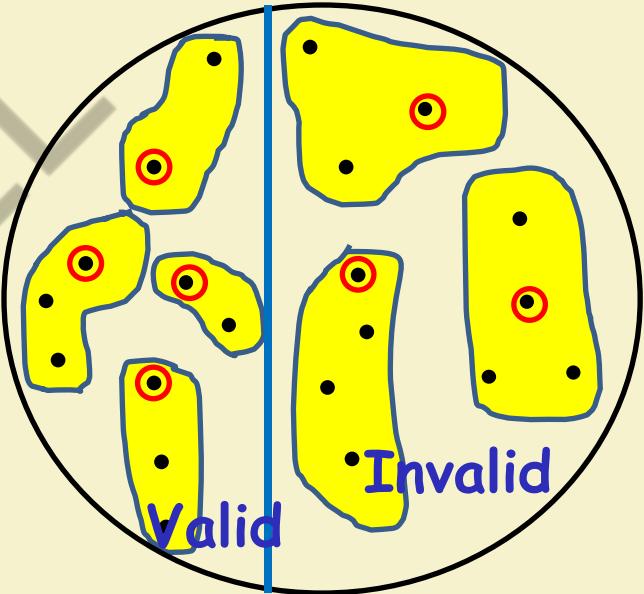


NPTEL  
ONLINE  
CERTIFICATION COURSES

# Equivalence Partitioning

- Create a test case using at least one value from each equivalence class

NPTEL



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Equivalence Class Partitioning

- If the input data to the program is specified by a range of values:
  - e.g. numbers between 1 to 5000.
  - One valid and two invalid equivalence classes are defined.



# Equivalence Class Partitioning

- If input is an enumerated set of values, e.g. :
  - {a,b,c}
- Define:
  - One equivalence class for valid input values.
  - Another equivalence class for invalid input values..



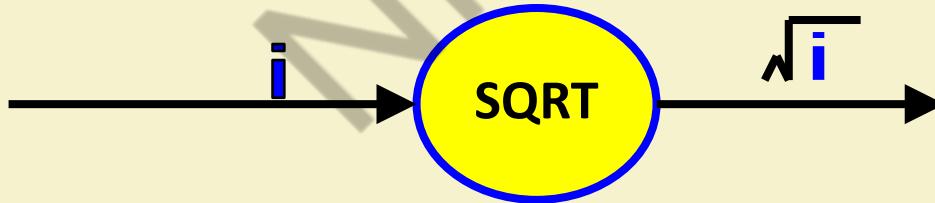
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Example

- A program reads an input value in the range of 1 and 5000:
  - Computes the square root of the input number



# Example (cont.)

- Three equivalence classes:
  - The set of negative integers,
  - Set of integers in the range of 1 and 5000,
  - Integers larger than 5000.



## Example (cont.)

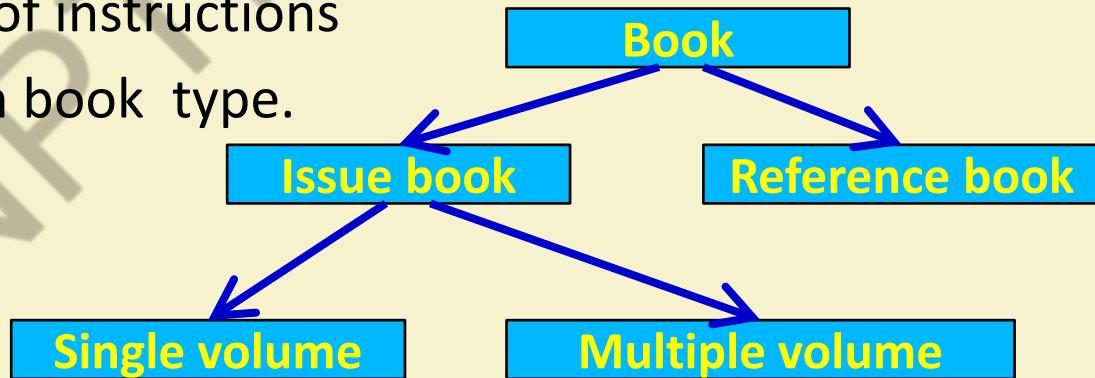
- The test suite must include:
  - Representatives from each of the three equivalence classes:
  - A possible test suite can be:  
 $\{-5, 500, 6000\}$ .



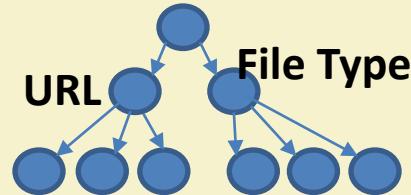
# Equivalence Partitioning

- A set of input values constitute an equivalence class if the tester believes that these are processed identically:

- Example : `issue book(book id);`
- Different set or sequence of instructions may be executed based on book type.



# Equivalence Partitioning: Example 1

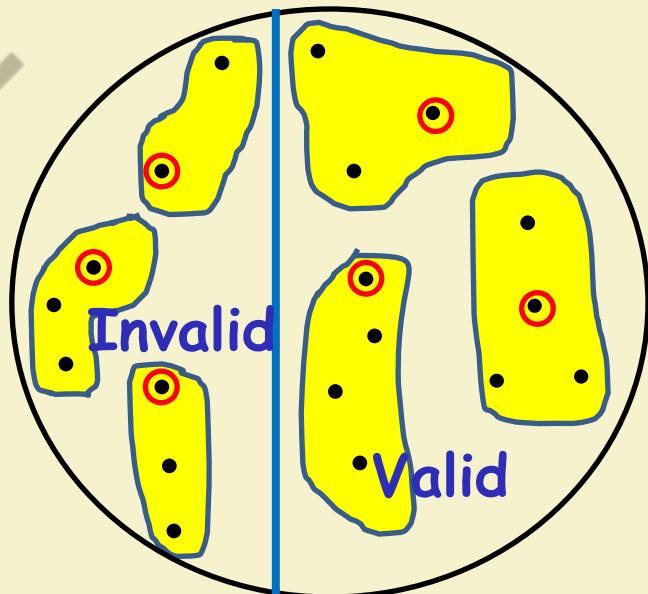
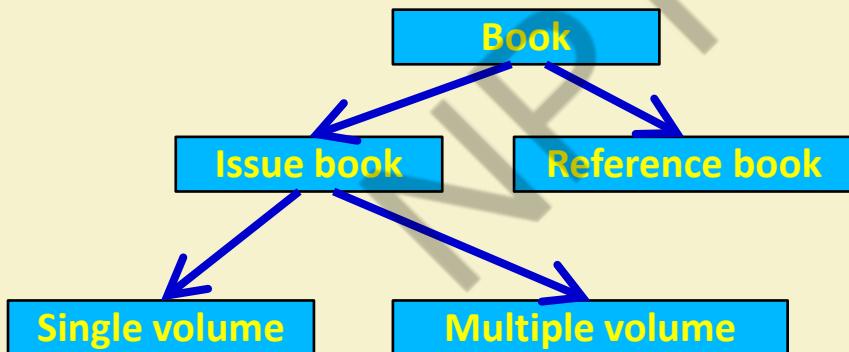


- Example: **Image Fetch-image(URL)**
  - **Equivalence Definition 1:** Partition based on URL protocol (“http”, “https”, “ftp”, “file”, etc.)
  - **Equivalence Definition 2:** Partition based on type of file being retrieved (HTML, GIF, JPEG, Plain Text, etc.)



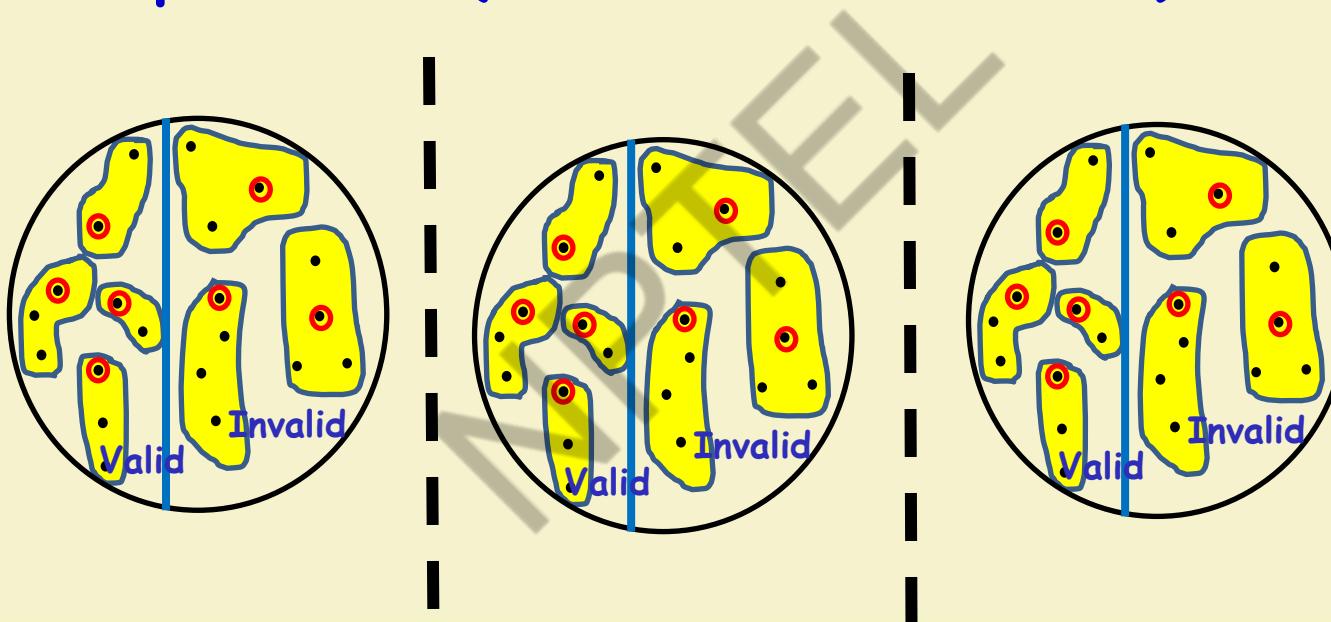
# Equivalence Partitioning: Single Parameter Function

- **issue-book(int book-id)**
- Create a test case for at least one value from each equivalence class



# Multiparameter Functions

- `postGrade(Roll, CourseNo, Grade)`



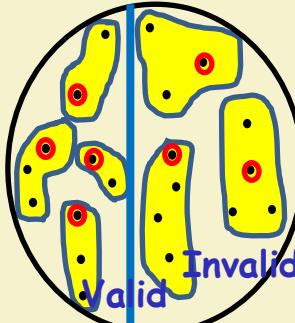
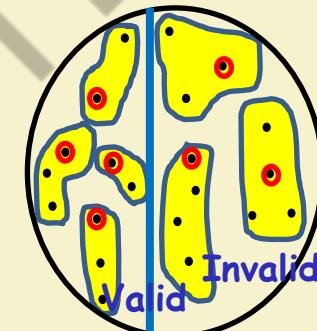
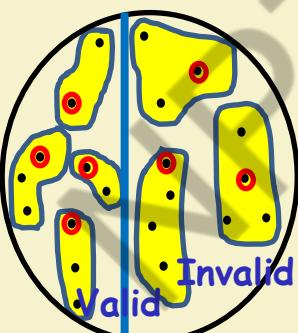
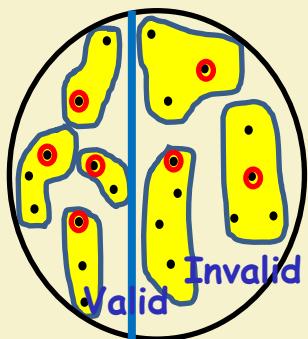
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

Multiparameter  
Function Accessing  
Global Variables

```
int Normalization Factor;  
postGrade(Roll, CourseNo, Grade)  
{ Grade=Grade*NormalizationFactor  
----- }
```



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Multi Parameter Equivalence Partitioning: Example

Input Parameter	Valid Equivalence Classes	Invalid Equivalence Classes
<p>An integer N such that: <math>-99 \leq N \leq 99</math></p>	?	?
<p>Phone Number Area code: [11,..., 999] Suffix: Any 6 digits</p>	?	?



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Equivalence Partitioning: Example

Input	Valid Equivalence Classes	Invalid Equivalence Classes
<b>A integer N such that: <math>-99 \leq N \leq 99</math></b>	<b>[-99, 99]</b>	< -99 > 99 Malformed numbers {12-, 1-2-3, ...} Non-numeric strings {junk, 1E2, \$13} Empty value
<b>Phone Number Prefix: [11, 999] Suffix: Any 6 digits</b>	<b>[11,999][000000, 999999]</b>	Invalid format 5555555, Area code < 11 or > 999 Area code with non-numeric characters

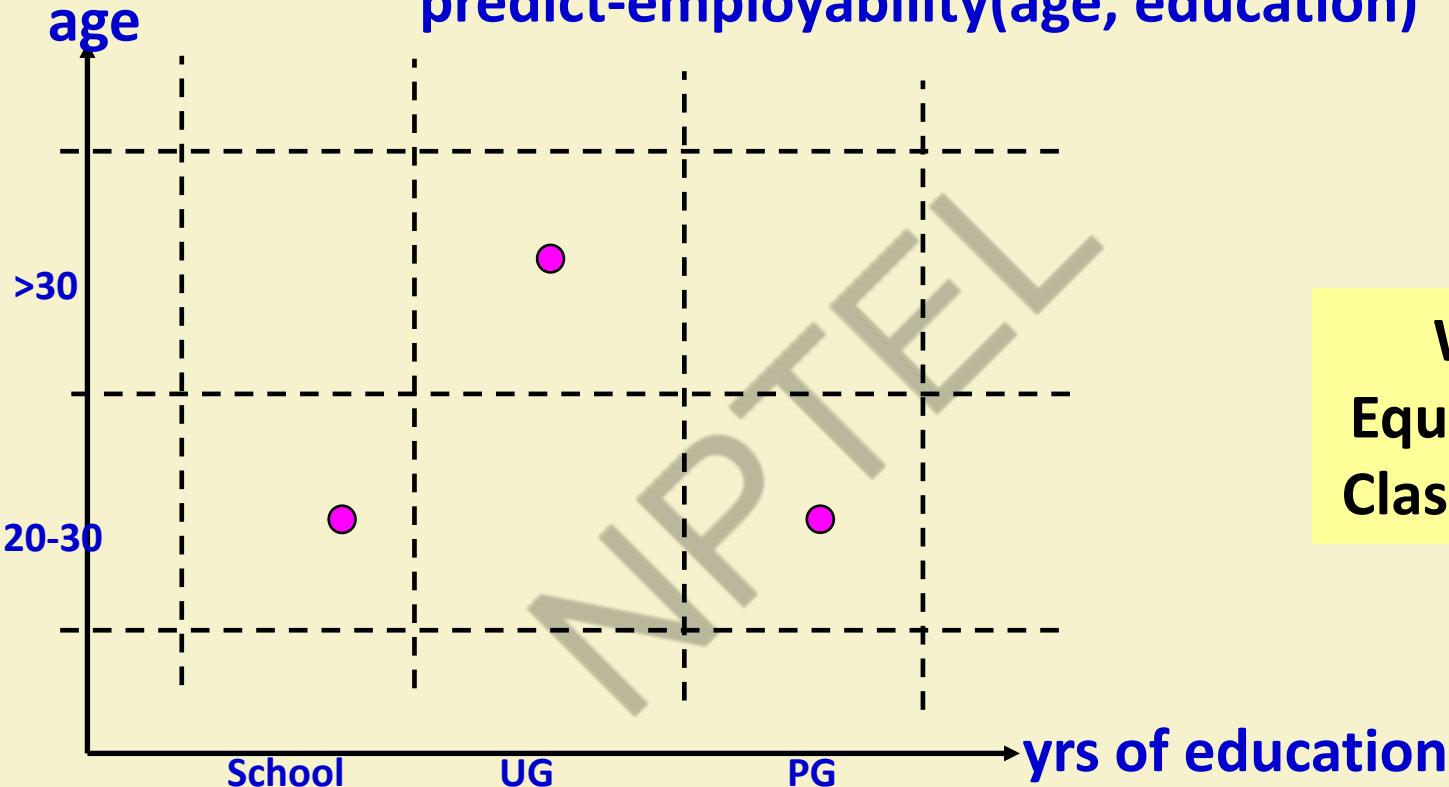


IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

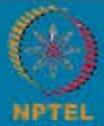
## **predict-employability(age, education)**



**Weak  
Equivalence  
Class Testing**

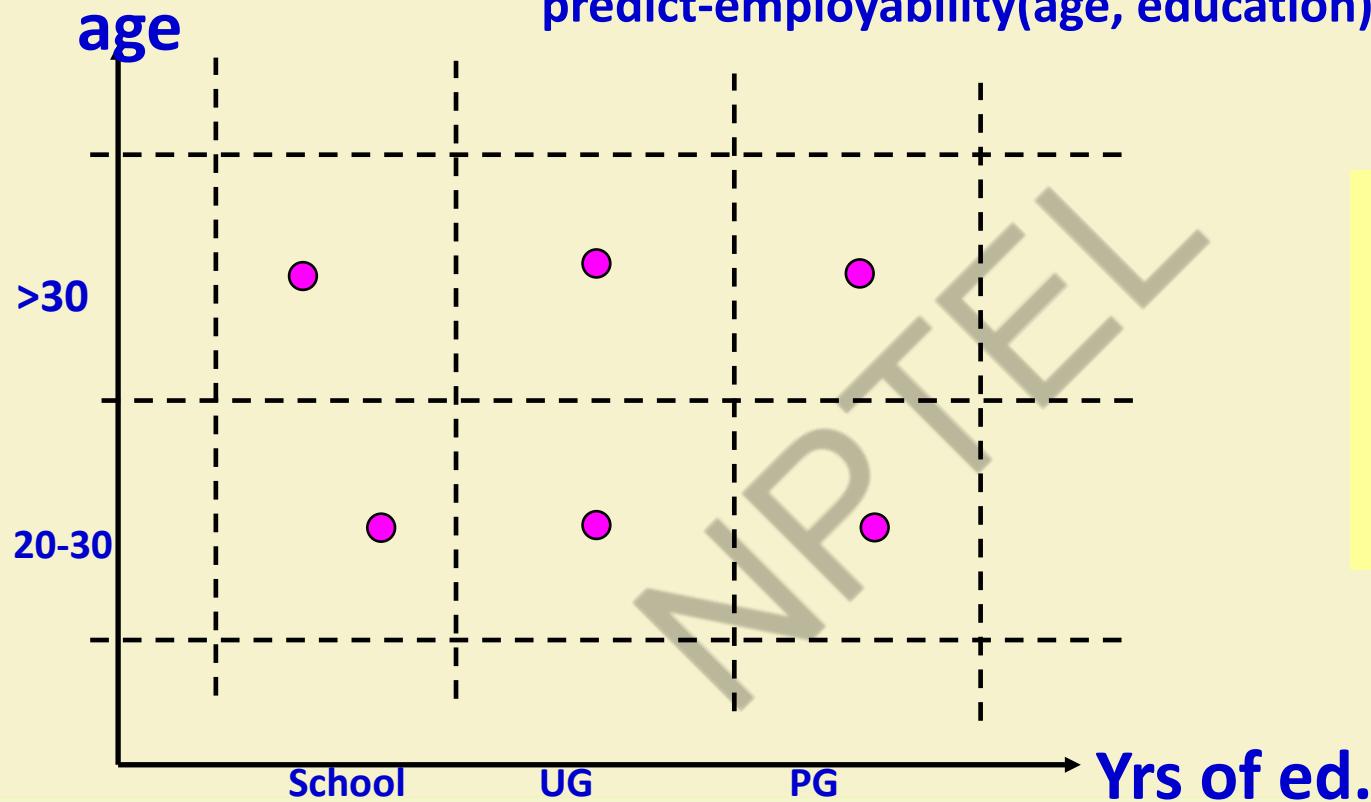


IIT KHARAGPUR



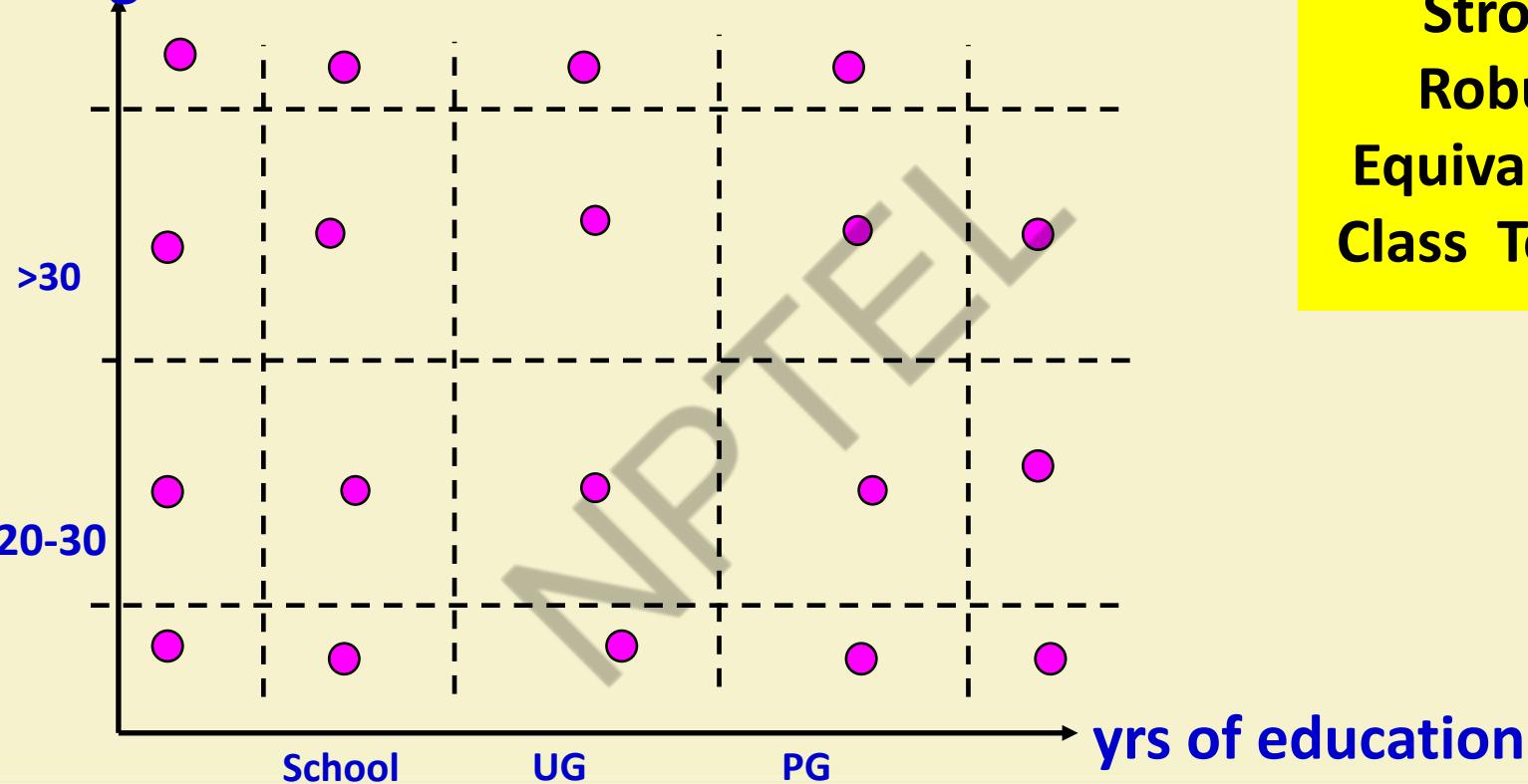
NPTEL  
ONLINE  
CERTIFICATION COURSES

**predict-employability(age, education)**



**Strong  
Equivalence  
Class Testing**

## age predict-employability(age, education)



Strong  
Robust  
Equivalence  
Class Testing

- Design Equivalence class test cases: **compute-interest(days)**
- A bank pays different rates of interest on a deposit depending on the deposit period.
  - 3% for deposit up to 15 days
  - 4% for deposit over 15 days and up to 180 days
  - 6% for deposit over 180 days upto 1 year
  - 7% for deposit over 1 year but less than 3 years
  - 8% for deposit 3 years and above

## Quiz 1



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Design Equivalence class test cases: **compute-interest(principal, days)**
- For deposits of less than or equal to Rs. 1 Lakh, rate of interest:
  - 6% for deposit upto 1 year
  - 7% for deposit over 1 year but less than 3 years
  - 8% for deposit 3 years and above
- For deposits of more than Rs. 1 Lakh, rate of interest:
  - 7% for deposit upto 1 year
  - 8% for deposit over 1 year but less than 3 years
  - 9% for deposit 3 years and above

## Quiz 2

## Quiz 3

- Design equivalence class test cases.
  - Consider a program that takes 2 strings of maximum length 20 and 5 characters respectively
  - Checks if the second is a substring of the first
  - **substr(s1,s2);**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Special Value Testing



IIT KHARAGPUR

03/08/10



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

52

52

# Special Value Testing

- What are special values?
  - The tester has reasons to believe that execution with certain values may expose bugs:
  - **General risk:** Example-- Boundary value testing
  - **Special risk:** Example-- Leap year not considered



IIT KHARAGPUR



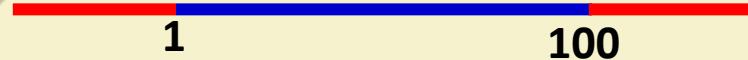
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Some typical programming errors occur:

- At boundaries of equivalence classes

Boundary  
Value Analysis

- Might be purely due to psychological factors.



- Programmers often commit mistakes in the:

- Special processing at the boundaries of equivalence classes.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Boundary Value Analysis

- Programmers may improperly use < instead of  $\leq$
- Boundary value analysis:  
—Select test cases at the boundaries of different equivalence classes.



## Boundary Value Analysis: Guidelines

- If an input is a range, bounded by values a and b:
  - Test cases should be designed with value a and b, just above and below a and b.
- **Example 1:** Integer D with input range [-3, 10],
  - test values: -3, 10, 11, -2, 0
- **Example 2:** Input in the range: [3,102]
  - test values: 3, 102, -1, 200, 5

# Boundary Value Testing Example

- Process employment applications based on a person's age.

0-16	Do not hire
16-18	May hire on part time basis
18-55	May hire full time
55-99	Do not hire

- Notice the problem at the boundaries.
  - Age "16" is included in two different equivalence classes (as are 18 and 55).

# Boundary Value Testing: Code Example

- If (applicantAge >= 0 && applicantAge <=16) hireStatus="NO";
- If (applicantAge >= 16 && applicantAge <=18) hireStatus="PART";
- If (applicantAge >= 18 && applicantAge <=55) hireStatus="FULL";
- If (applicantAge >= 55 && applicantAge <=99) hireStatus="NO";



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Corrected boundaries:

## Boundary Value Testing Example (cont)

0–15      Don't hire

16–17      Can hire on a part-time basis only

18–54      Can hire as full-time employees

55–99      Don't hire

- What about ages -3 and 101?
- The requirements do not specify how these values should be treated.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Boundary Value Testing Example (cont)

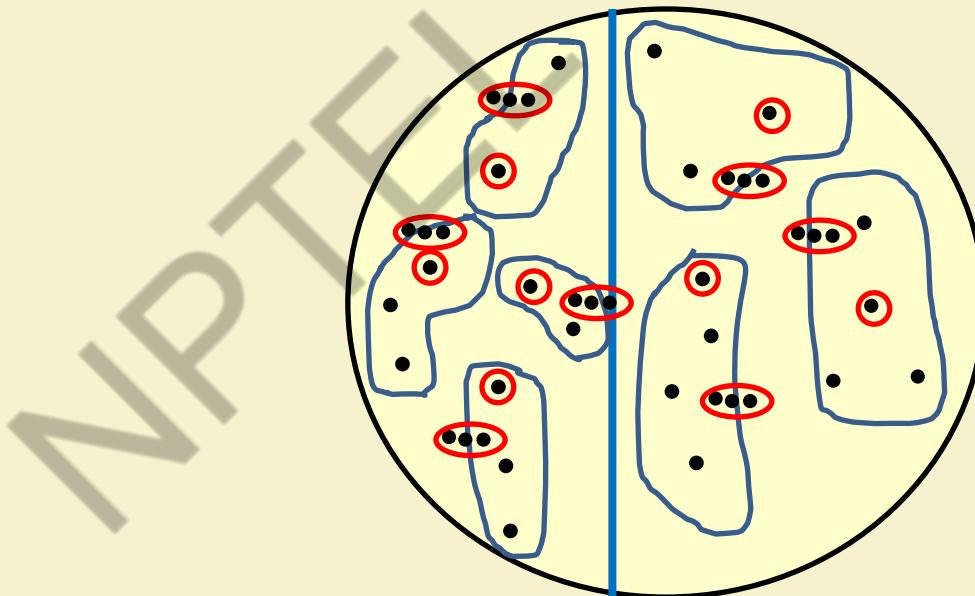
- The code to implement the corrected rules is:

If (applicantAge >= 0 && applicantAge <=15)	hireStatus="NO";
If (applicantAge >= 16 && applicantAge <=17)	hireStatus="PART";
If (applicantAge >= 18 && applicantAge <=54)	hireStatus="FULL";
If (applicantAge >= 55 && applicantAge <=99)	hireStatus="NO";

- Special values on or near the boundaries in this example are {-1, 0, 1}, {14, 15, 16}, {17, 18, 19}, {54, 55, 56}, and {98, 99, 100}.

# Boundary Value Analysis

- Create test cases to test boundaries of equivalence classes



# Example 1

- For a function that computes the square root of an integer in the range of 1 and 5000:
  - Test cases must include the values: $\{0, 1, 2, 4999, 5000, 5001\}$ .



- Consider a program that reads the “age” of employees and computes the average age.

### Example 2

input (ages) → Program → output: average age

Assume valid age is 1 to 150



- How would you test this?
  - How many test cases would you generate?
  - What type of test data would you input to test this program?



IIT KHARAGPUR



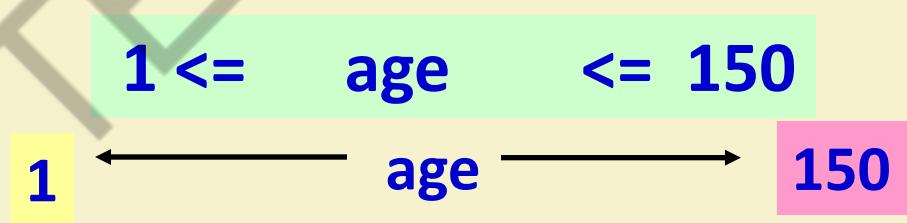
NPTEL  
ONLINE  
CERTIFICATION COURSES

# Boundaries of the inputs

The “basic” boundary value testing would include 5 test cases:

1. - at minimum boundary
2. - immediately above minimum
3. - between minimum and maximum (nominal)
4. - immediately below maximum
5. - at maximum boundary

**predict-longevity(age)**



- How many test cases for the example ?

- answer : **5**

- Test input values? :

- 1 at the minimum

- 2 at one above minimum

- 45 at middle

- 149 at one below maximum

- 150 at maximum

## Test Cases for the Example

**predict-longevity(age)**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

## Multiple Parameters: Independent distinct Data

- Suppose there are 2 “distinct” inputs that are assumed to be independent of each other.
  - Input field 1: years of education ( say 1 to 23 )
  - Input field 2: age (1 to 150)
- If they are independent of each other, then we can start with  $5 + 5 = 10$  sets.

coverage of input data: yrs of ed

1. n=1 ;      age = whatever(37)
2. n=2;      age = whatever
3. n = 12;      age = whatever
4. n = 22;      age = whatever
5. n = 23;      age = whatever

coverage of input data: age

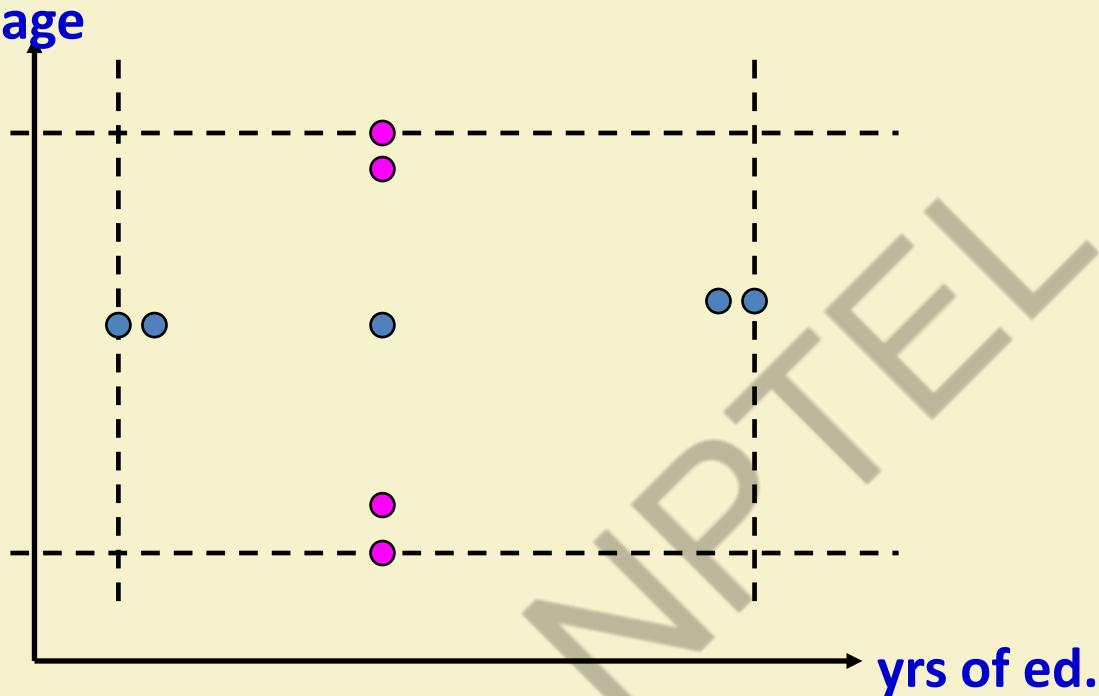
1. n= 12;      age = 1
2. n =12;      age = 2
3. n = 12;      age = 37
4. n = 12;      age = 149
5. n = 12;      age = 150



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



2 –  
Independent  
inputs

- Note that there needs to be only 9 test cases for 2 independent inputs.
- In general, need  $(4z + 1)$  test cases for  $z$  independent inputs.

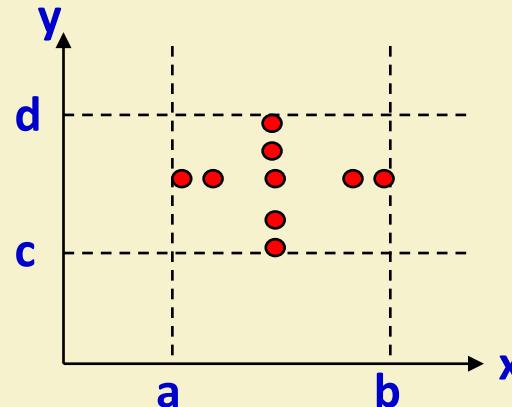
# Boundary Value Test

Given  $f(x,y)$  with constraints  $\rightarrow$

$$\begin{aligned} a \leq x \leq b \\ c \leq y \leq d \end{aligned}$$

Boundary Value analysis focuses on the boundary of the input space to identify test cases.

Defined as input variable value at min, just above min, a nominal value, just above max, and at max.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Weak Testing: Single Fault Assumption

- **Premise:** “Failures rarely occur as the result of the simultaneous occurrence of two (or more) faults”
- Under this:
  - Hold the values of all but one variable at their nominal values, and let that one variable assume its extreme values.



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

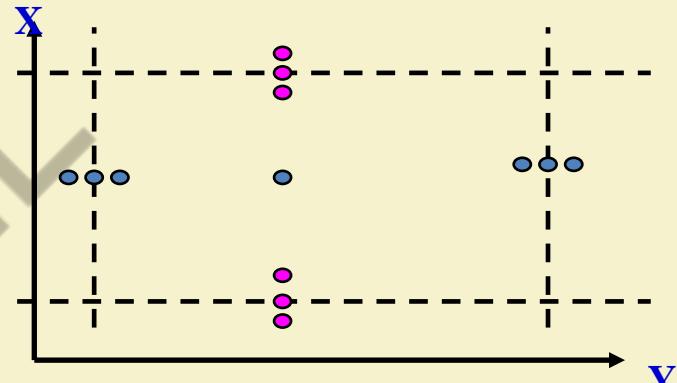
# Boundary Value Analysis: Robustness

- Numeric values are often entered as strings :
  - Internally converted to numbers [int x = atoi(str); val=x-48;]
- This conversion requires the program to distinguish between digits and non-digits
- A boundary case to consider: Will the program accept / and : as digits?

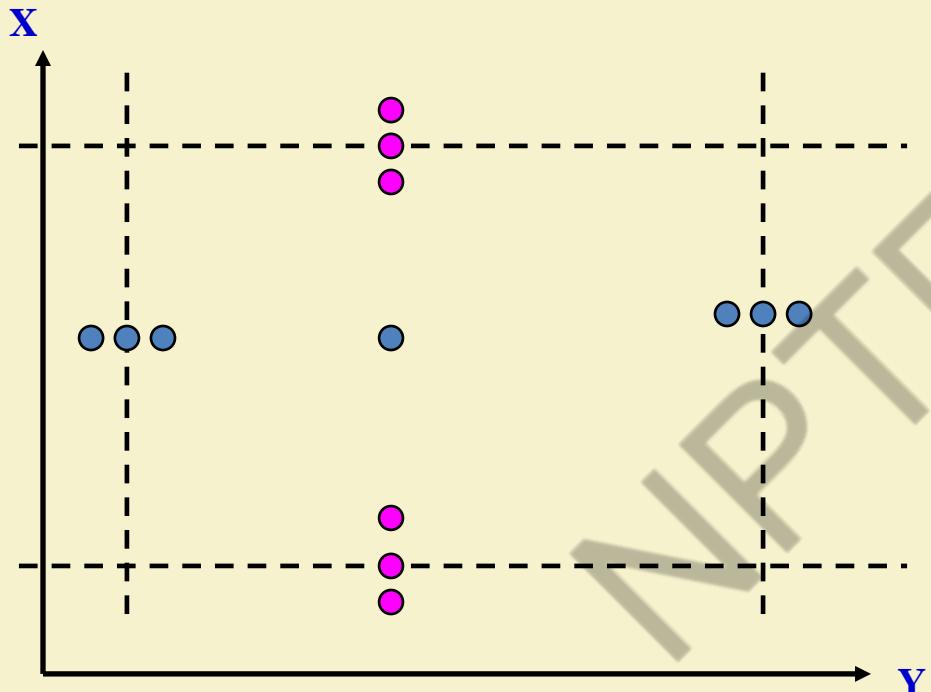
Char	/	0	1	2	3	4	5	6	7	8	9	:
ASCII	47	48	49	50	51	52	53	54	55	56	57	58

# Robustness testing

- This is just an extension of the Boundary Values to include invalid values:
  - Less than minimum
  - Greater than maximum
- There are **7 test cases** for each input
- The testing of robustness is really a test of “error” handling.
  1. *Did we anticipate the error situations?*
  2. *Do we issue informative error messages?*
  3. *Do we support some kind of recovery from the error?*



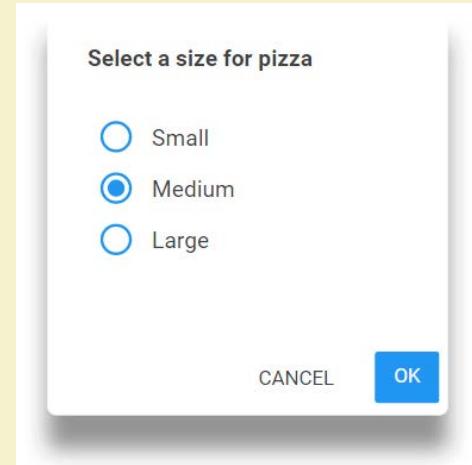
## 2 – independent inputs for robustness test



- Note that there needs to be only 13 test cases for 2 independent variables or inputs.
- In general, there will be  $(6n+1)$  test cases for  $n$  independent inputs.

## Some Limitations of Boundary Value Testing

- How to handle a set of values?
- How about set of Boolean variables?
  - True
  - False
- May be radio buttons
- What about a non-numerical variable whose values are text?



# Quiz: BB Test Design

- Design black box test suite for a function that solves a quadratic equation of the form  $ax^2+bx+c=0$ .
- Equivalence classes
  - Invalid Equation
  - Valid Equation: Roots?
    - Complex
    - Real
      - Coincident
      - Unique

# Combinatorial Testing



IIT KHARAGPUR

03/08/10



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

1

# Combinatorial Testing: Motivation

- The behavior of a program may be affected by many factors:
  - **Input parameters,**
  - **Environment configurations (global variables),**
  - **State variables. ...**
- Equivalence partitioning of an input variable:
  - Identify the possible types of input values requiring different processing.
- If the factors are many:
  - It is impractical to test all possible combinations of values of all factors.

**Combinatorial:** Relating to, or involving combinations



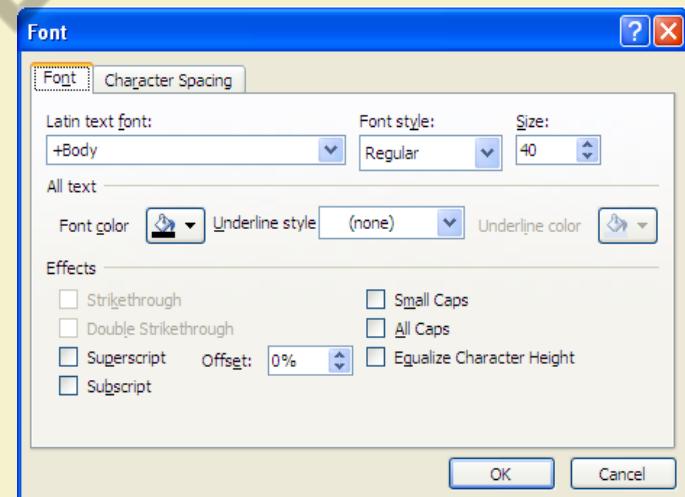
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Combinatorial Testing: Motivation

- Many times, the specific action to be performed depends on the value of a set of Boolean variable:
  - Controller applications
  - User interfaces



# Combinatorial Testing

- Several combinatorial testing strategies exist:
  - Decision table-based testing
  - Cause-effect graphing
  - Pair-wise testing (reduced number of test cases)



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- Applicable to requirements involving conditional actions.
- This is represented as a decision table:
  - Conditions = inputs
  - Actions = outputs
  - Rules = test cases
- Assume independence of inputs
- Example
  - If c1 AND c2 OR c3 then A1

## Decision table-based Testing (DTT)

	Rule1	Rule2	Rule3	Rule4
Condition1	Yes	Yes	No	No
Condition2	Yes	X	No	X
Condition3	No	Yes	No	X
Condition4	No	Yes	No	Yes
Action1	Yes	Yes	No	No
Action2	No	No	Yes	No
Action3	No	No	No	Yes

## Combinations

	Rule1	Rule2	Rule3	Rule4
Condition1	Yes	Yes	No	No
Condition2	Yes	X	No	X
Condition3	No	Yes	No	X
Condition4	No	Yes	No	Yes
Action1	Yes	Yes	No	No
Action2	No	No	Yes	No
Action3	No	No	No	Yes



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- A decision table consists of a number of columns (rules) that comprise all test situations
- Example: the triangle problem
  - C1: a, b,c form a triangle
  - C2: a=b
  - C3: a= c
  - C4: b= c
  - A1: Not a triangle
  - A2:scalene
  - A3: Isosceles
  - A4:equilateral
  - A5: Right angled

## Sample Decision table

	r1	r2	...				rn
C1	0	1					0
c2	-	1					0
C3	-	1					1
C4	-	1					0
a1	1	0					0
a2	0	0					1
a3	0	0					0
a4	0	1					0
a5	0	0					

# Test cases from Decision Tables

Test Case ID	a	b	c	Expected output
TC1	4	1	2	Not a Triangle
TC2	2888	2888	2888	Equilateral
TC3	?		)	Impossible
TC4				
...				
TC11				

- C1: a, b,c form a triangle
- C2: a=b
- C3: a= c
- C4: b= c

# More Complete Decision Table for the Triangle Problem

Conditions												
C1: $a < b+c?$	F	T	T	T	T	T	T	T	T	T	T	T
C2: $b < a+c?$	-	F	T	T	T	T	T	T	T	T	T	T
C3: $c < a+b?$	-	-	F	T	T	T	T	T	T	T	T	T
C4: $a=b?$	-	-	-	T	T	T	T	F	F	F	F	F
C5: $a=c?$	-	-	-	T	T	F	F	T	T	F	F	F
C6: $b=c?$	-	-	-	T	F	T	F	T	F	T	F	F
Actions												
A1: Not a Triangle	X	X	X									
A2: Scalene												X
A3: Isosceles									X	X	X	
A4: Equilateral					X							
A5: Impossible						X	X		X			



## Test Cases for the Triangle Problem

Case ID	a	b	c	Expected Output
DT1	4	1	2	Not a Triangle
DT2	1	4	2	Not a Triangle
DT3	1	2	4	Not a Triangle
DT4	5	5	5	Equilateral
DT5	?	?	?	Impossible
DT6	?	?	?	Impossible
DT7	2	2	3	Isosceles
DT8	?	?	?	Impossible
DT9	2	3	2	Isosceles
DT10	3	2	2	Isosceles
DT11	3	4	5	Scalene



## Decision Table – Example 2

### Printer Troubleshooting

Conditions	Printer does not print	Y	Y	Y	Y	N	N	N	N
	A red light is flashing	Y	Y	N	N	Y	Y	N	N
	Printer is unrecognized	Y	N	Y	N	Y	N	Y	N
Actions	Check the power cable				X				
	Check the printer-computer cable	X		X					
	Ensure printer software is installed	X		X	X	X		X	
	Check/replace ink	X	X			X	X		
	Check for paper jam		X		X				

# Quiz: Develop BB Test Cases

- Policy for charging customers for certain in-flight services:

If the flight is more than half-full and ticket cost is more than Rs. 3000, free meals are served unless it is a domestic flight. Otherwise, no meals are served. Meals are charged on all domestic flights.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

Fill all  
combinations  
in the table.

		POSSIBLE COMBINATIONS							
CONDITIONS	<i>more than half-full</i>	N	N	N	N	Y	Y	Y	Y
	<i>more than Rs.3000 per seat</i>	N	N	Y	Y	N	N	Y	Y
	<i>domestic flight</i>	N	Y	N	Y	N	Y	N	Y
ACTIONS									

Analyze  
column by  
column to  
determine  
which actions  
are appropriate  
for each  
combination

### POSSIBLE COMBINATIONS

CONDITIONS	<i>more than half-full</i>	N	N	N	N	Y	Y	Y	Y
	<i>more than Rs. 3000 per seat</i>	N	N	Y	Y	N	N	Y	Y
	<i>domestic flight</i>	N	Y	N	Y	N	Y	N	Y
ACTIONS	<i>serve meals</i>						Y	Y	Y
	<i>free</i>								Y

Reduce the table by eliminating redundant columns.

		POSSIBLE COMBINATIONS							
CONDITIONS	<i>more than half-full</i>	N	N	N	N	Y	Y	Y	Y
	<i>more than Rs. 3000 per seat</i>	N	N	Y	Y	N	N	Y	Y
	<i>domestic flight</i>	N	Y	N	Y	N	Y	N	Y
ACTIONS	<i>serve meals</i>					X	X	X	X
	<i>free</i>							X	

# Final solution

		Combinations			
CONDITIONS	<i>more than half-full</i>	N	Y	Y	Y
	<i>more than 3000 per seat</i>	-	N	Y	Y
	<i>domestic flight</i>	-	-	N	Y
	<i>serve meals</i>		X	X	X
	<i>free</i>			X	
ACTIONS					

## Assumptions regarding rules

–Rules need to be complete:

- That is, every combination of decision table values including default combinations are present.

–Rules need to be consistent:

- That is, there is no two different actions for the same combinations of conditions



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Guidelines and Observations

- Decision table testing is appropriate for programs:
  - There is a lot of decision making
  - Output is a logical relationship among input variables
  - Results depend on calculations involving subsets of inputs
  - There are cause and effect relationships between input and output
- **Decision tables do not scale up very well**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Quiz: Design test Cases

- Customers on a e-commerce site get following discount:
  - A member gets 10% discount for purchases lower than Rs. 2000, else 15% discount
  - Purchase using SBI card fetches 5% discount
  - If the purchase amount after all discounts exceeds Rs. 2000/- then shipping is free.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Cause-effect Graphs

- Overview:
  - Explores combinations of possible inputs
  - Specific combination of inputs (causes) results in outputs (effects)
  - Represented as nodes of a cause effect graph
  - The graph also includes constraints and a number of intermediate nodes linking causes and effects



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Cause-Effect Graph Example

- If depositing less than Rs. 1 Lakh, rate of interest:
  - 6% for deposit upto 1 year
  - 7% for deposit over 1 year but less than 3 yrs
  - 8% for deposit 3 years and above
- If depositing more than Rs. 1 Lakh, rate of interest:
  - 7% for deposit upto 1 year
  - 8% for deposit over 1 year but less than 3 yrs
  - 9% for deposit 3 years and above



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Cause-Effect Graph Example

## Causes

C1: Deposit<1yr

C2: 1yr<Deposit<3yrs

C3: Deposit>3yrs

C4: Deposit <1 Lakh

C5: Deposit >=1Lakh

## Effects

e1: Rate 6%

e2: Rate 7%

e3: Rate 8%

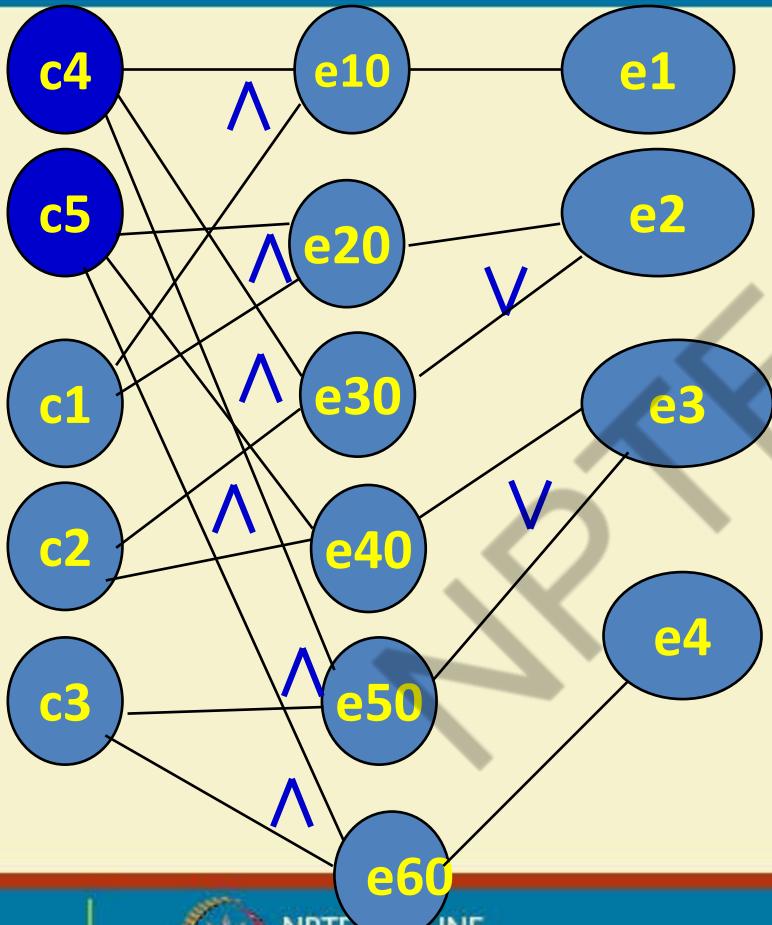
e4: Rate 9%



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES



## Cause-Effect Graphing



## Develop a Decision Table

C1	C2	C3	C4	C5	e1	e2	e3	e4
1	0	0	1	0	1	0	0	0
1	0	0	0	1	0	1	0	0
0	1	0	1	0	0	1	0	0
0	1	0	0	1	1	0	1	0

- Convert each row to a test case



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Pair-wise Testing



IIT KHARAGPUR  
03/08/10



NPTEL ONLINE  
CERTIFICATION COURSES

25

25

# Combinatorial Testing of User Interface

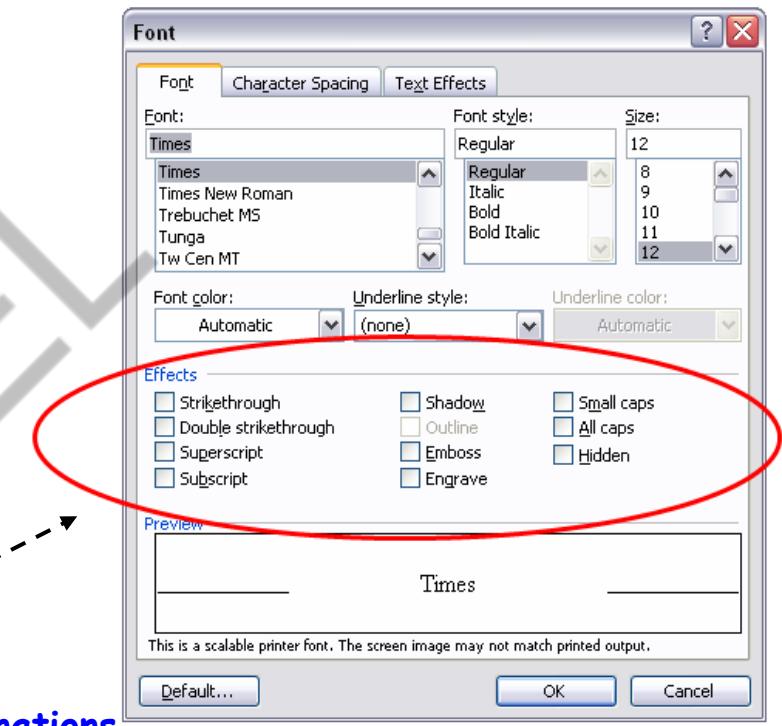
0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	0	0	0	0	1	
1	0	1	1	0	1	0	1	0	0	0
1	0	0	0	1	1	1	0	0	0	0
0	1	1	0	0	1	0	0	1	0	
0	0	1	0	1	0	1	1	1	0	
1	1	0	1	0	0	1	0	1	0	
0	0	0	1	1	1	0	0	1	1	
0	0	1	1	0	0	1	0	0	1	
0	1	0	1	1	0	0	1	0	0	
1	0	0	0	0	0	1	1	1	1	
0	1	0	0	1	1	1	0	1	0	

0 = effect off

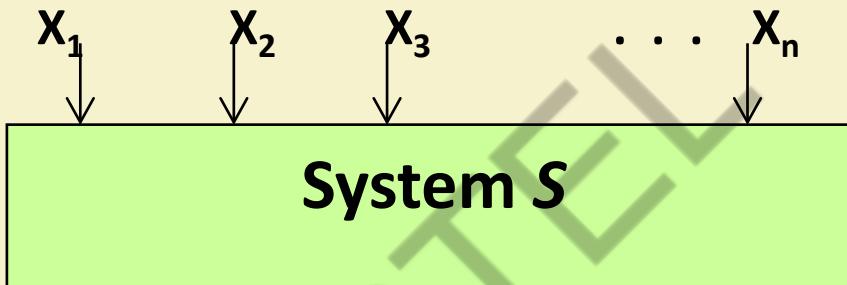
1 = effect on

$2^{10} = 1,024$  tests for all combinations

\*  $10^3 = 1024 * 1000$  .... Just too many to tests



# Combinatorial Testing Problem



- Combinatorial testing problems generally follow a simple input-process-output model;
- The “state” of the system is not the focus of combinatorial testing.



- Instead of testing all possible combinations:

- A subset of combinations is generated.

## t-way Testing

- Key observation:

- It is often the case that a fault is caused by interactions among a few factors.**

- t-way testing can dramatically reduce the number of test cases:
  - but remains effective in terms of fault detection.



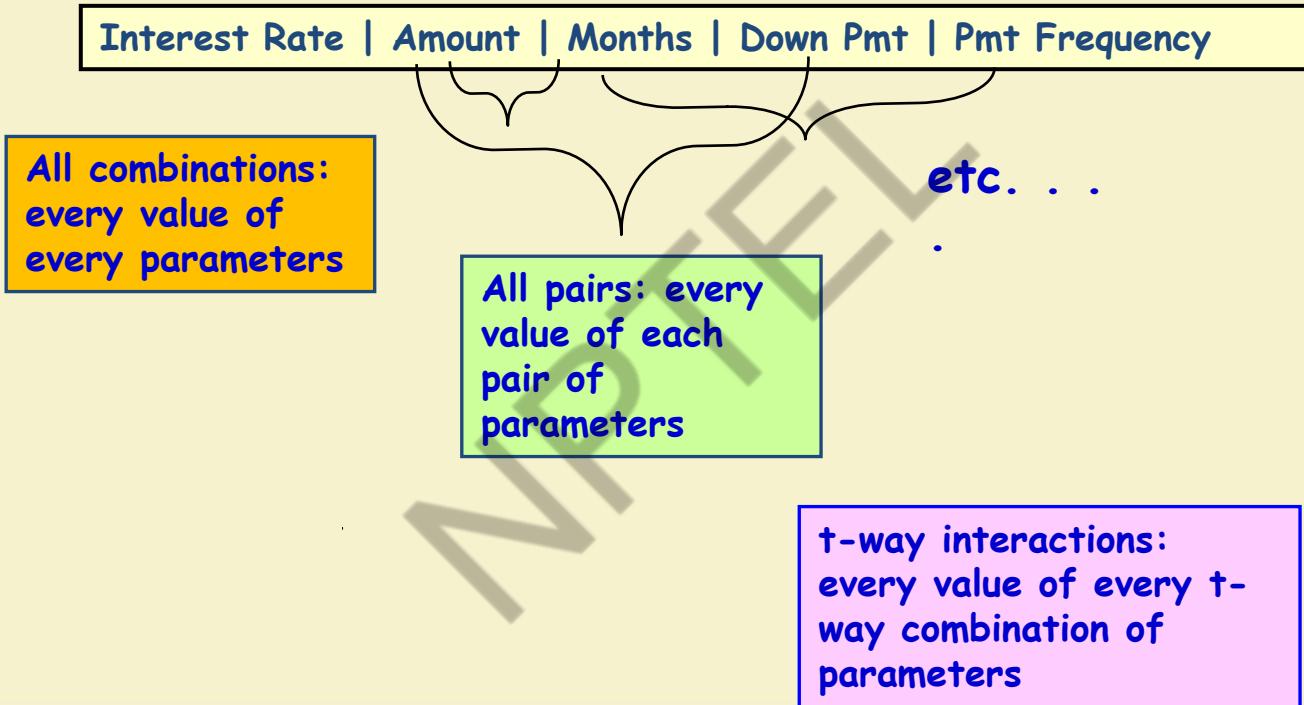
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

28

# t-way Interaction Testing



# Pairwise Testing

Pressure | Temperature | Velocity | Acceleration | Air Density

Pressure	Temperature
A	T1
A	T2
A	T3
B	T1
B	T2
B	T3

A	T1	1	10	1.1
A	T2	2	0	2.1
A	T3	3	20	3.1
		4	0	
		5		
		6		



# Pairwise Reductions

Number of inputs	Number of selected test data values	Number of combinations	Size of pairwise test set
7	2	128	8
13	3	$1.6 \times 10^6$	15
40	3	$1.2 \times 10^{19}$	21



## Fault-Model

- **A t-way interaction fault:**

- Triggered by a certain combination of  $t$  input values.
- A simple fault is a 1-way fault
- Pairwise fault is a  $t$ -way fault where  $t = 2$ .

- **In practice, a majority of software faults consist of simple and pairwise faults.**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Single-mode Bugs

- The simplest bugs are single-mode faults:
  - Occur when one option causes a problem regardless of the other settings
  - Example: A printout is always gets smeared when you choose the duplex option in the print dialog box
    - Regardless of the printer or the other selected options



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Double-mode Faults

- **Double-mode faults**
  - Occurs when two options are combined
  - Example:** The printout is smeared only when duplex is selected and the printer selected is model 394



# Multi-mode Faults

- **Multi-mode faults**
  - Occur when three or more settings produce the bug
  - This is the type of problems that make complete coverage necessary



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- begin
  - int x, y, z;
  - input (x, y, z);
  - if (x == x1 and y == y2)
    - output (f(x, y, z));
  - else if (x == x2 and y == y1)
    - output (g(x, y));
  - Else // Missing (x == x2 and y == y1)  $f(x, y, z) - g(x, y)$ ;
    - output ( $f(x, y, z) + g(x, y)$ )
- end
- Expected:  $x = x1 \text{ and } y = y1 \Rightarrow f(x, y, z) - g(x, y);$   
 $x = x2, y = y2 \Rightarrow f(x, y, z) + g(x, y)$



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

36

# Example: Android smart phone testing

- Apps should work on all combinations of platform options, but there are  $3 \times 3 \times 4 \times 3 \times 5 \times 4 \times 4 \times 5 \times 4 = 172,800$  configurations

HARDKEYBOARDHIDDEN_NO	ORIENTATION_LANDSCAPE
HARDKEYBOARDHIDDEN_UNDEFINED	ORIENTATION_PORTRAIT
HARDKEYBOARDHIDDEN_YES	ORIENTATION_SQUARE
KEYBOARDHIDDEN_NO	ORIENTATION_UNDEFINED
KEYBOARDHIDDEN_UNDEFINED	SCREENLAYOUT_LONG_MASK
KEYBOARDHIDDEN_YES	SCREENLAYOUT_LONG_NO
KEYBOARD_12KEY	SCREENLAYOUT_LONG_UNDEFINED
KEYBOARD_NOKEYS	SCREENLAYOUT_LONG_YES
KEYBOARD_QWERTY	SCREENLAYOUT_SIZE_LARGE
KEYBOARD_UNDEFINED	SCREENLAYOUT_SIZE_MASK
NAVIGATIONHIDDEN_NO	SCREENLAYOUT_SIZE_NORMAL
NAVIGATIONHIDDEN_UNDEFINED	SCREENLAYOUT_SIZE_SMALL
NAVIGATIONHIDDEN_YES	SCREENLAYOUT_SIZE_UNDEFINED
NAVIGATION_DPAD	TOUCHSCREEN_FINGER
NAVIGATION_NONAV	TOUCHSCREEN_NOTOUCH
NAVIGATION_TRACKBALL	TOUCHSCREEN_STYLUS
NAVIGATION_UNDEFINED	TOUCHSCREEN_UNDEFINED
NAVIGATION_WHEEL	

# White-Box Testing



IIT KHARAGPUR

03/08/10



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

38

# What is White-box Testing?

- White-box test cases designed based on:
  - Code structure of program.
  - White-box testing is also called structural testing.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# White-Box Testing Strategies

- **Coverage-based:**
  - Design test cases to cover certain program elements.
- **Fault-based:**
  - Design test cases to expose some category of faults



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

- Several white-box testing strategies have become very popular :
  - Statement coverage
  - Branch coverage
  - Path coverage
  - Condition coverage
  - MC/DC coverage
  - Mutation testing
  - Data flow-based testing

## White-Box Testing



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Why Both BB and WB Testing?

## Black-box

- Impossible to write a test case for every possible set of inputs and outputs
- Some code parts may not be reachable
- **Does not tell if extra functionality has been implemented.**

## White-box

- Does not address the question of whether a program matches the specification
- Does not tell if all functionalities have been implemented
- **Does not uncover any missing program logic**

# Coverage-Based Testing Versus Fault-Based Testing

- Idea behind coverage-based testing:
  - Design test cases so that certain program elements are executed (or covered).
  - Example: statement coverage, path coverage, etc.
- Idea behind fault-based testing:
  - Design test cases that focus on discovering certain types of faults.
  - Example: Mutation testing.

- **Statement:** each statement executed at least once
- **Branch:** each branch traversed (and every entry point taken) at least once
- **Condition:** each condition True at least once and False at least once
- **Multiple Condition:** All combination of Condition covered
- **Path:**
- **Dependency:**

Types of program element Coverage



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES



**Stronger and  
Weaker Testing**

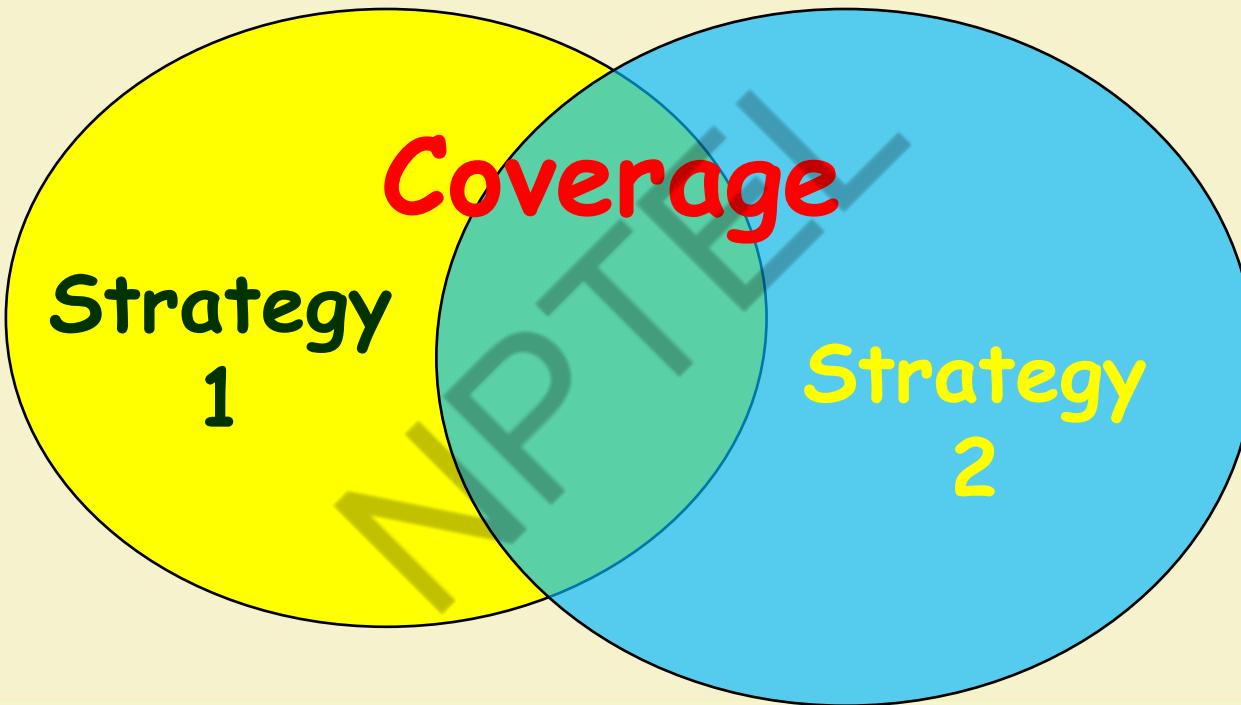


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Complementary Testing

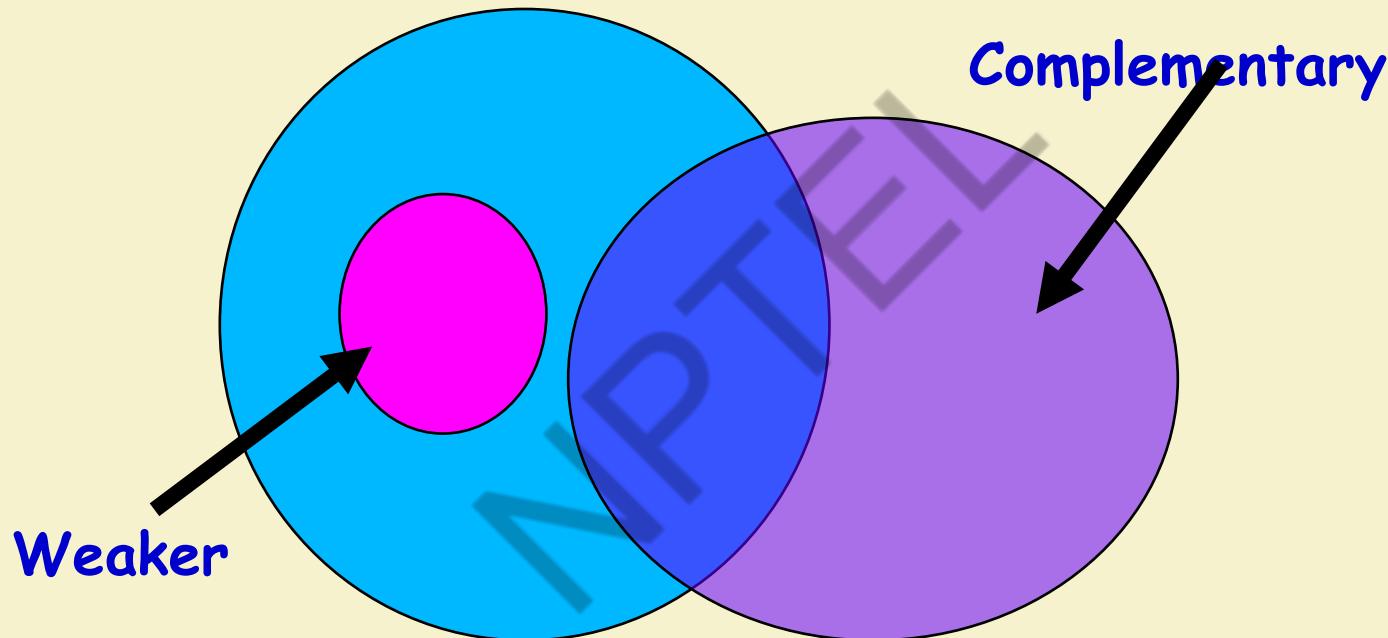


IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Stronger, Weaker, and Complementary Testing



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Statement Coverage

- Statement coverage strategy:
  - Design test cases so that every statement in the program is executed at least once.



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Statement Coverage

- The principal idea:
  - Unless a statement is executed,
  - We have no way of knowing if an error exists in that statement.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Statement Coverage Criterion

- However, observing that a statement behaves properly for one input value:
  - No guarantee that it will behave correctly for all input values!**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Statement Coverage

- Coverage measurement:

# executed statements

# statements

- **Rationale:** a fault in a statement can only be revealed by executing the faulty statement



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Example

- int f1(int x, int y){
- 1 while (x != y){
- 2   if (x>y) then
- 3       x=x-y;
- 4   else y=y-x;
- 5 }
- 6 return x;      }

Euclid's GCD Algorithm



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

```
int f1(int x,int y){  
1 while (x != y){  
2   if (x>y) then  
3     x=x-y;  
4   else y=y-x;  
5 }  
6 return x;      }
```

## Example

Euclid's GCD Algorithm



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Euclid's GCD Algorithm

- By choosing the test set  $\{(x=3, y=3), (x=4, y=3), (x=3, y=4)\}$ 
  - All statements are executed at least once.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Branch Coverage

- Also called decision coverage.
- Test cases are designed such that:
  - Each branch condition
    - Assumes true as well as false value.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example

```
int f1(int x,int y){  
1 while (x != y){  
2   if (x>y) then  
3     x=x-y;  
4   else y=y-x;  
5 }  
6 return x;      }
```



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Example

- Test cases for branch coverage can be:
- $\{(x=3,y=3), (x=3,y=2), (x=4,y=3), (x=3,y=4)\}$



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Branch Testing

- **Adequacy criterion:** Each branch (edge in the CFG) must be executed at least once
- Coverage:

# executed branches

# branches



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Quiz 1: Branch and Statement Coverage: Which is Stronger?

- Branch testing guarantees statement coverage:
  - A stronger testing compared to the statement coverage-based testing.

# Stronger Testing

- Stronger testing:
  - Superset of weaker testing
  - A stronger testing covers all the elements covered by a weaker testing.
  - Covers some additional elements not covered by weaker testing



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Sample Coverage Report

Coverage Report

http://www.cafeconleche.org/coverage/coverage.html

Packages

- All
- [org.jaxen](#)
- [org.jaxen.dom](#)
- [org.jaxen.dom.html](#)

All Packages

Classes

- [BaseXPath \(77%\)](#)
- [Context \(93%\)](#)
- [ContextSupport \(91%\)](#)
- [DefaultNavigator \(38%\)](#)
- [DocumentNavigator](#)
- [DOMXPath \(100%\)](#)
- [DocumentNavigator](#)
- [HTMLXPath \(0%\)](#)
- [NamespaceNode \(0%\)](#)
- [DocumentNavigator](#)
- [Dom4jXPath \(100%\)](#)

Reports generated by [Cobertura](#).

Package	# Classes	Line Coverage	Branch Coverage	Complexity
All Packages	205	69%	80%	2.811
<a href="#">org.jaxen</a>	24	77%	73%	1.38
<a href="#">org.jaxen.dom</a>	3	55%	60%	1.907
<a href="#">org.jaxen.dom.html</a>	2	0%	0%	1.364
<a href="#">org.jaxen.dom4j</a>	2	78%	85%	2.395
<a href="#">org.jaxen.expr</a>	73	73%	84%	1.566
<a href="#">org.jaxen.expr.iter</a>	14	98%	100%	1.029
<a href="#">org.jaxen.function</a>	27	64%	76%	5.373
<a href="#">org.jaxen.function.ext</a>	6	63%	72%	4.235
<a href="#">org.jaxen.function.xslt</a>	1	86%	100%	2.5
<a href="#">org.jaxen.javabean</a>	4	44%	72%	1.87
<a href="#">org.jaxen.jdom</a>	3	62%	63%	2.897
<a href="#">org.jaxen.pattern</a>	13	49%	52%	2.135
<a href="#">org.jaxen.xpath</a>	8	51%	81%	1.887
<a href="#">org.jaxen.xpath.base</a>	6	95%	100%	10.723
<a href="#">org.jaxen.xpath.helpers</a>	2	28%	83%	1.34
<a href="#">org.jaxen.util</a>	15	41%	50%	2.432
<a href="#">org.jaxen.xom</a>	2	71%	66%	1.783

Disabled

## Coverage Report

```
110 128        else if ( nav.isElement( first ) )
111
112 100        {
113            return nav.getElementQName( first );
114 28        }
115        else if ( nav.isAttribute( first ) )
116 0            return nav.getAttributeQName( first );
117        }
118 28        else if ( nav.isProcessingInstruction( first ) )
119        {
120 0            return nav.getProcessingInstructionTarget( first );
121        }
122 28        else if ( nav.isNamespace( first ) )
123        {
124 0            return nav.getNamespacePrefix( first );
125        }
126 28        else if ( nav.isDocument( first ) )
127        {
128 28            return "";
129        }
130 0        else if ( nav.isComment( first ) )
131        {
132 0            return "";
133        }
134 0        else if ( nav.isText( first ) )
135        {
136 0            return "";
137        }
138        else {
139 0            throw new FunctionCallException("The argument to the name
140        )
141    }
142
143 8        return "";
144
```

Done

Disabled



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Statements vs Branch Testing

- Traversing all edges of a graph causes all nodes to be visited
  - So a test suite that satisfies branch adequacy criterion also satisfies statement adequacy criterion for the same program.
- The converse is not true:
  - A statement-adequate (or node-adequate) test suite may not be branch-adequate (edge-adequate).



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

- Statement coverage
- Branch coverage (aka decision coverage)
- Basic condition coverage
- Condition/Decision coverage
- Multiple condition coverage
- MC/DC coverage
- Path coverage
- Data flow-based testing
- Mutation testing

## White-box Testing

# All Branches can still miss testing specific conditions

- Assume failure occurs when  $c==\text{DIGIT}$   
 $\text{if}((c == \text{ALPHABET}) \mid\mid (c == \text{DIGIT}))$
- Branch adequacy criterion can be satisfied by  $c==\text{alphabet}$  and  $c==\text{splchar}$ 
  - **The faulty sub-expression might not be tested!**
  - Even though we test both outcomes of the branch

# Basic Condition Coverage

- Also called condition coverage or simple condition coverage .
- Test case design:  $((c == \text{ALPHABET}) || (c == \text{DIGIT}))$ 
  - Each component of a composite conditional expression
    - Made to assume both true and false values.

# Basic Condition Testing

- **Simple or (basic) Condition Testing:**
  - Test cases make each atomic condition assume T and F values
  - Example: **if (a>10 && b<50)**
- **Following test inputs would achieve basic condition coverage**
  - **a=15, b=30**
  - **a=5, b=60**
- Does basic condition coverage subsume decision coverage?

# Example: BCC

- Consider the conditional expression  
 $\neg((c1.\text{and}.c2).\text{or}.c3)$ :
- Each of  $c1$ ,  $c2$ , and  $c3$  is exercised with all possible values,
  - That is, given true and false values.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Basic condition testing

- Adequacy criterion: each basic condition must be executed at least once
- Coverage:

**# truth values taken by all basic conditions**

---

**2 \* # basic conditions**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Is BCC Stronger than Decision Coverage?

- Consider the conditional statement:  
**–If(((a>5).and.(b<3)).or.(c==0)) a=10;**
- Two test cases can achieve basic condition coverage: (a=10, b=2, c=2) and (a=1, b=10, c=0)
- **BCC does not imply Decision coverage and vice versa**

# Condition/Decision Coverage Testing

- **Condition/decision coverage:**

- Each atomic condition made to assume both T and F values
- Decisions are also made to get T and F values

- **Multiple condition coverage (MCC):**

- Atomic conditions made to assume all possible combinations of truth values



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# MCC

- Test cases make Conditions to assume all possible combinations of truth values.
- Consider: **if (a || b && c) then ...**

Test	a	b	c
(1)	T	T	T
(2)	T	T	F
(3)	T	F	T
(4)	T	F	F
(5)	F	T	T
(6)	T	T	F
(7)	F	F	T
(8)	F	F	F

Exponential in the  
number of basic  
conditions



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Multiple Condition Coverage (MCC)

- Consider a Boolean expression having n components:
  - For condition coverage we require  $2^n$  test cases.
- MCC testing technique:
  - Practical only if n (the number of component conditions) is small.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# MCC for Compound conditions: Exponential complexity

$((a \mid\mid b) \&\& c) \mid\mid d) \&\& e$

$$2^5=32$$

Test Case

(1)  
(2)  
(3)  
(4)  
(5)  
(6)  
(7)  
(8)  
(9)  
(10)  
(11)  
(12)  
(13)

Test Case	a	b	c	d	e
(1)	T	-	T	-	T
(2)	F	T	T	-	T
(3)	T	-	F	T	T
(4)	F	T	F	T	T
(5)	F	F	-	T	T
(6)	T	-	T	-	F
(7)	F	T	T	-	F
(8)	T	-	F	T	F
(9)	F	T	F	T	F
(10)	F	F	-	T	F
(11)	T	-	F	F	-
(12)	F	T	F	F	-
(13)	F	F	-	F	-

- Short-circuit evaluation often reduces number of test cases to a more manageable number, but not always...



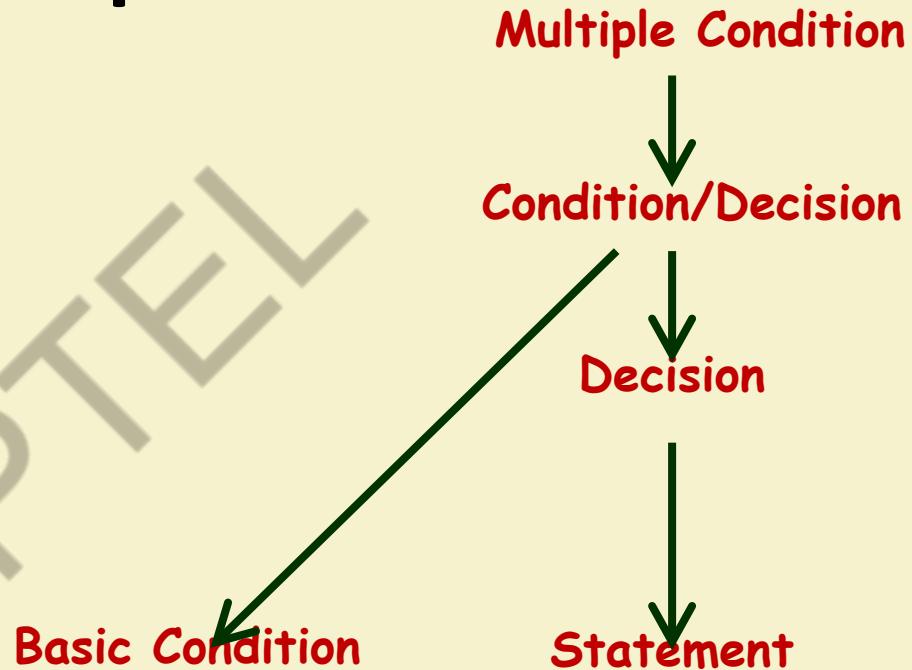
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Subsumption

- Condition testing:
  - Stronger testing than branch testing.
- Branch testing:
  - Stronger than statement coverage testing.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Shortcomings of Condition Testing

- **Redundancy of test cases:** Condition evaluation could be compiler-dependent:
  - Reason: Short circuit evaluation of conditions
- **Coverage may be Unachievable:** Possible dependencies among variables:
  - Example: `((chr=='A') || (chr=='E'))` can not both be true at the same time

# Short-circuit Evaluation

- **if(a>30 && b<50)...**
  - If a>30 is FALSE compiler need not evaluate (b<50)
- Similarly, **if(a>30 || b<50)...**
  - If a>30 is TRUE compiler need not evaluate (b<50)

# Software Testing

Rajib Mall

CSE Department  
IIT KHARAGPUR



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# MC/DC Testing



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Modified Condition/Decision Coverage (MC/DC)

- **Motivation:** Effectively test important combinations of conditions, without exponential blowup to test suite size:
  - “**Important**” combinations means: Each basic condition should independently affect the outcome of each decision
- **Requires:**  $\text{If}(\text{ (A==0)} \vee (\text{B}>5) \wedge (\text{C}<100)) \dots$ 
  - For each **basic condition c**, **Compound condition as a whole evaluates to true or false as ac becomes T or F**

## Condition/Decision Coverage

- Condition: true, false.
- Decision: true, false.

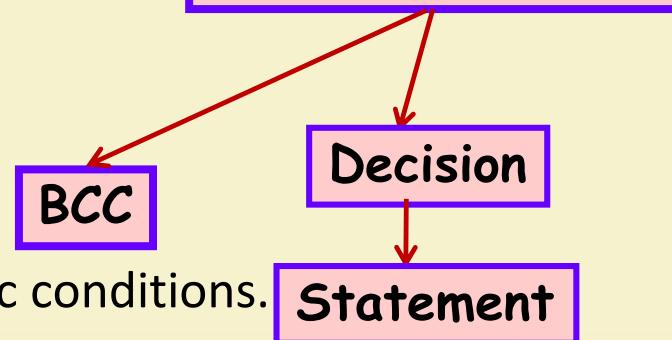
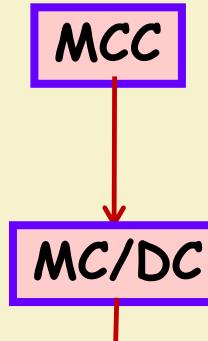
## Multiple Condition coverage (MCC)

- all possible combinations of condition outcomes in a decision
  - for a decision with  $n$  conditions
- $2^n$  test cases are required

## Modified Condition/Decision coverage (MC/DC)

- Bug-detection effectiveness almost similar to MCC
- Number of test cases linear in the number of basic conditions.

## Subsumption hierarchy



## What is MC/DC?

- MC/DC stands for **Modified Condition / Decision Coverage**
- It is a condition coverage technique
  - **Condition:** Atomic conditions in expression.
  - **Decision:** Controls the program flow.
- **Main idea:** Each condition must be shown to independently affect the outcome of a decision.
  - The outcome of a decision changes as a result of changing a single condition.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Three Requirements for MC/DC

## Requirement 1:

- Every decision in a program must take T/F values.

## Requirement 2:

- Every condition in each decision must take T/F values.

## Requirement 3:

- Each condition in a decision should independently affect the decision's outcome.



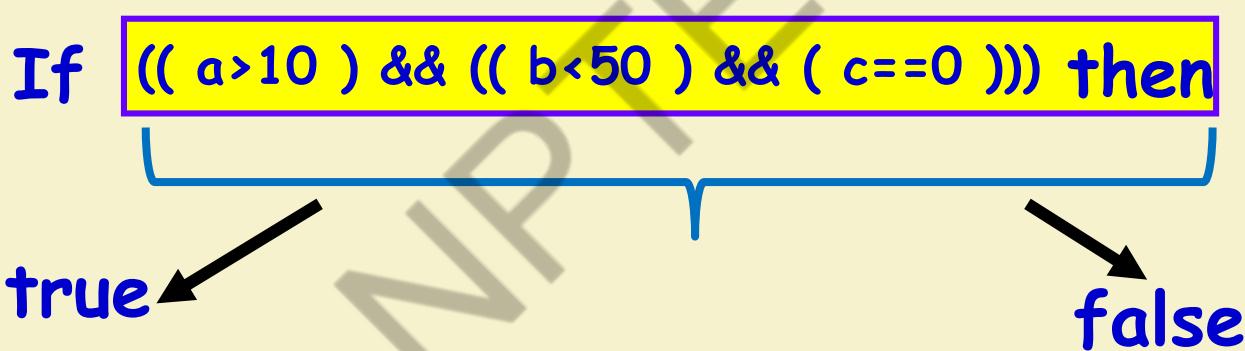
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# MC/DC Requirement 1

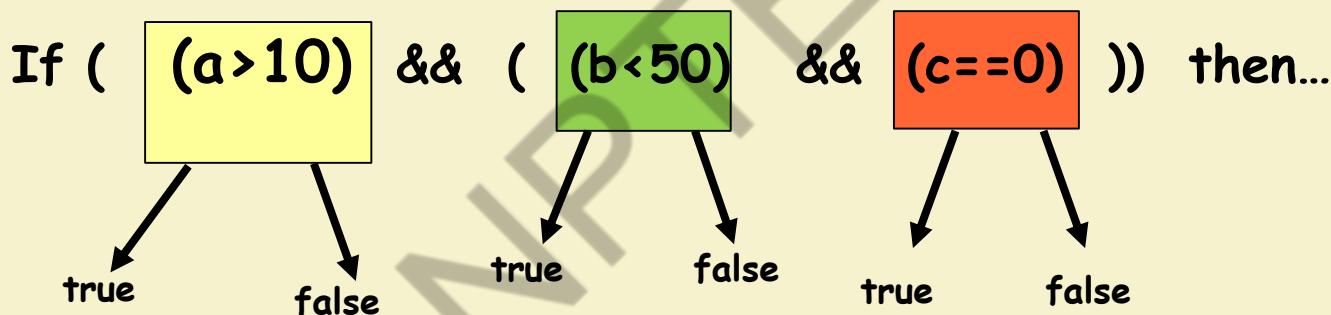
- The decision is made to take both T/F values.



- This is as in Branch coverage.

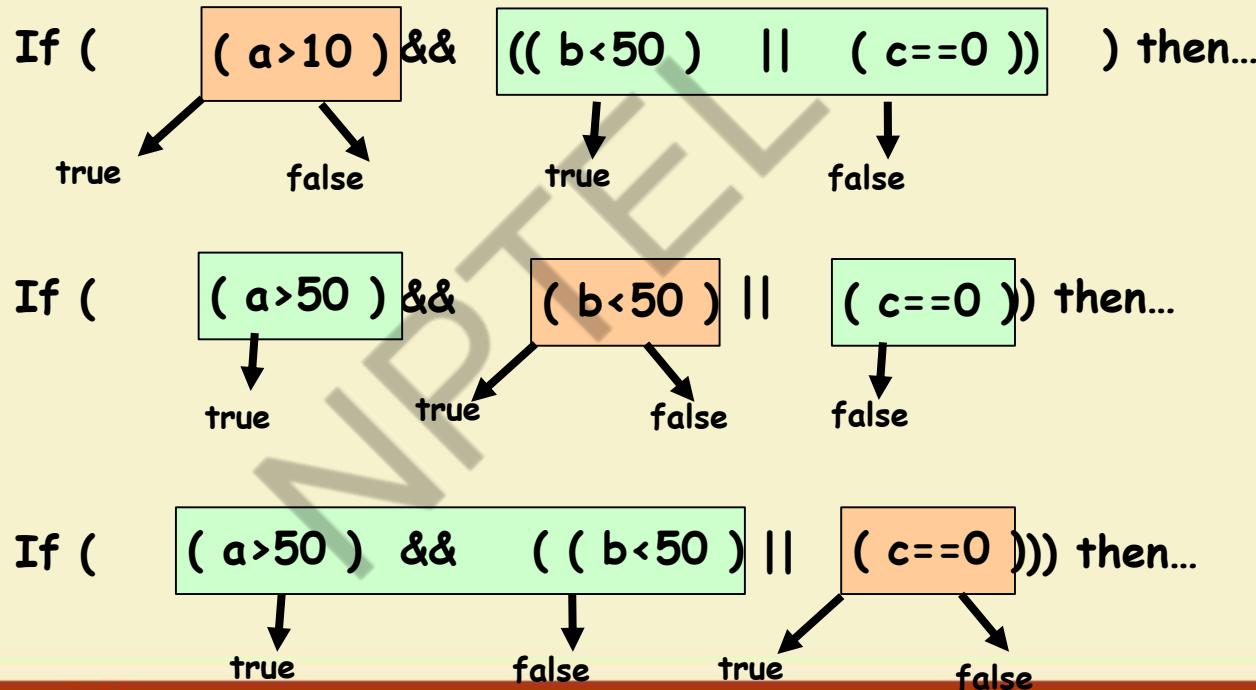
## MC/DC Requirement 2

- Test cases make every condition in the decision to evaluate to both T and F at least once.



# MC/DC Requirement 3

- Every condition in the decision independently affects the decision's outcome.



# MC/DC: An Example

- N+1 test cases required for N basic conditions
- Example:

$((((a>10 \mid\mid b<50) \&\& c==0) \mid\mid d<5) \&\& e==10)$

Test Case	a>10	b<50	c==0	d<5	e==10	outcome
(1)	<u>true</u>	false	<u>true</u>	false	<u>true</u>	true
(2)	false	<u>true</u>	true	false	true	true
(3)	true	false	false	<u>true</u>	true	true
(6)	true	false	true	false	<u>false</u>	false
(11)	true	false	<u>false</u>	<u>false</u>	true	false
(13)	<u>false</u>	<u>false</u>	true	false	true	false

- Underlined values independently affect the output of the decision

## Creating MC/DC test cases

- Create truth table for conditions.
- Extend the truth table to represent test case pair that lead to show the independence influence of each condition.

Example : If ( A and B ) then ...

Test Case Number	A	B	Decision	Test case pair for A	Test case pair for B
1	T	T	T	3	2
2	T	F	F		1
3	F	T	F	1	
4	F	F	F		

- Show independence of A :
  - Take 1 + 3
- Show independence of B :
  - Take 1 + 2
- Resulting test cases are
  - 1 + 2 + 3



If( (A  $\vee$  B)  $\wedge$  C) ....

	A	B	C	Result	A	B	C	MC/DC
1	1	1	1	1			*	*
2	1	1	0	0			*	*
3	1	0	1	1	*			*
4	0	1	1	1		*		*
5	1	0	0	0				
6	0	1	0	0				
7	0	0	1	0	*	*		*
8	0	0	0	0				

Another Example

# Minimal Set Example

If (A and (B or C)) then...

TC#	ABC	Result	A	B	C
1	TTT	T	5		
2	TTF	T	6	4	
3	TFT	T	7		4
4	TFF	F		2	3
5	FTT	F	1		
6	FTF	F	2		
7	FFT	F	3		
8	FFF	F			

We want to determine the MINIMAL set of test cases

Here:

- {2,3,4,6}
- {2,3,4,7}

Non-minimal set is:

- {1,2,3,4,5}



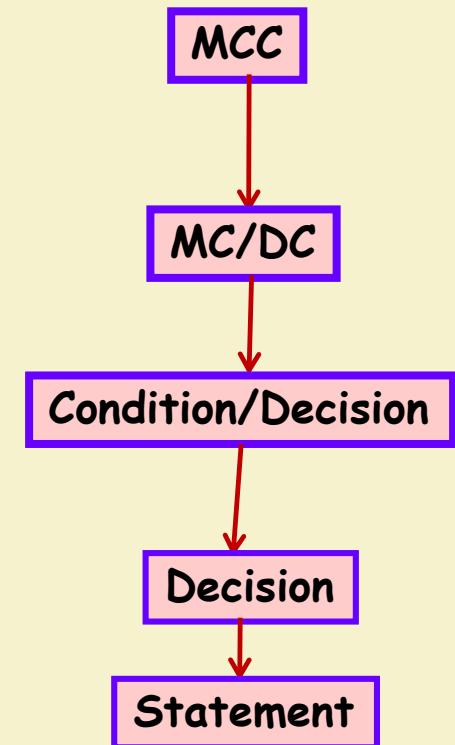
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Observation

- MC/DC criterion is stronger than condition/decision coverage criterion,
  - but the number of test cases to achieve the MC/DC still linear in the number of conditions  $n$  in the decisions.



# MC/DC: Summary

- MC/DC essentially is :
  - basic condition coverage (C)
  - branch coverage (DC)
  - plus one additional condition (M):  
every condition must *independently affect* the decision's output
- It is subsumed by MCC and subsumes all other criteria discussed so far
  - stronger than statement and branch coverage
- **A good balance of thoroughness and test size and therefore widely used...**

# Path Testing



IIT KHARAGPUR  
03/08/10



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Path Coverage

- Design test cases such that:
  - All linearly independent paths in the program are executed at least once.
- Defined in terms of
  - Control flow graph (CFG) of a program.

# Path Coverage-Based Testing

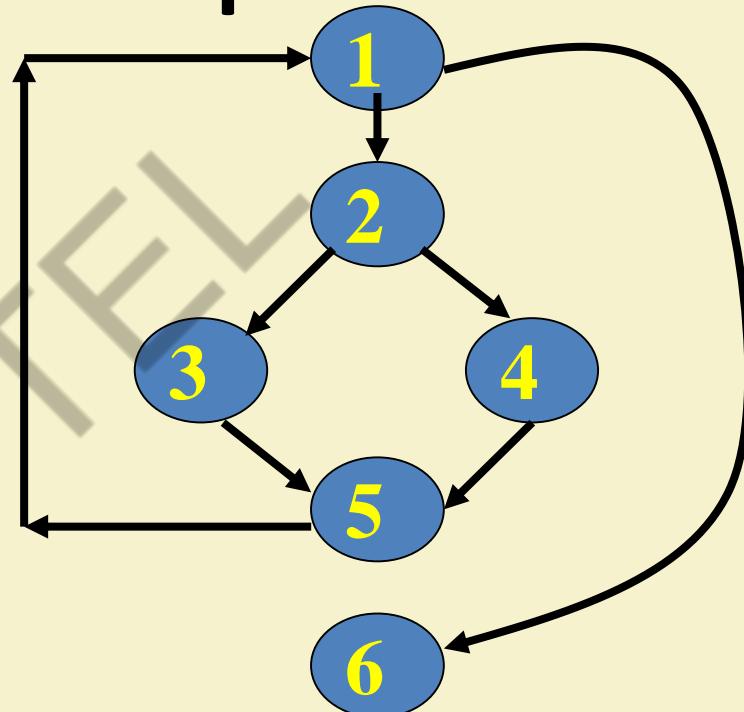
- To understand the path coverage-based testing:
  - We need to learn how to draw control flow graph of a program.
- A control flow graph (CFG) describes:
  - The sequence in which different instructions of a program get executed.
  - The way control flows through the program.

# How to Draw Control Flow Graph?

- **Number all statements of a program.**
- Numbered statements:
  - Represent nodes of control flow graph.
- Draw an edge from one node to another node:
  - **If execution of the statement representing the first node can result in transfer of control to the other node.**

```
int f1(int x,int y){  
1 while (x != y){  
2   if (x>y) then  
3     x=x-y;  
4   else y=y-x;  
5 }  
6 return x;      }
```

## Example



IIT KHARAGPUR



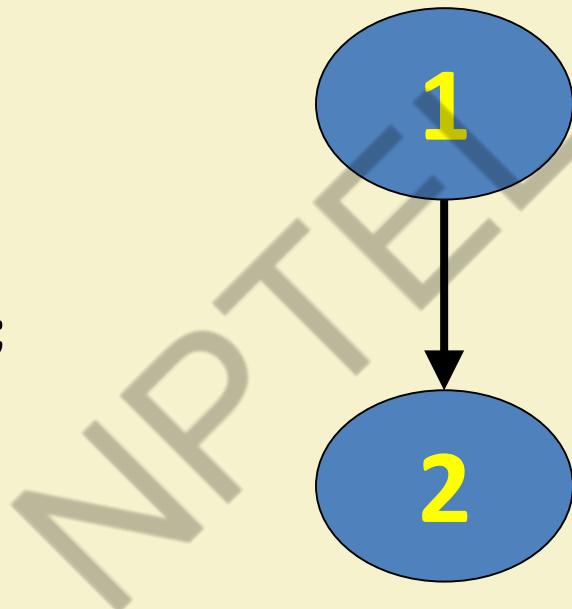
NPTEL ONLINE  
CERTIFICATION COURSES

- Every program is composed of:
  - Sequence
  - Selection
  - Iteration
- If we know how to draw CFG corresponding these basic statements:
  - We can draw CFG for any program.

How to Draw Control flow Graph?

# How to Draw Control flow Graph?

- Sequence:
  - 1 a=5;
  - 2 b=a\*b-1;



IIT KHARAGPUR

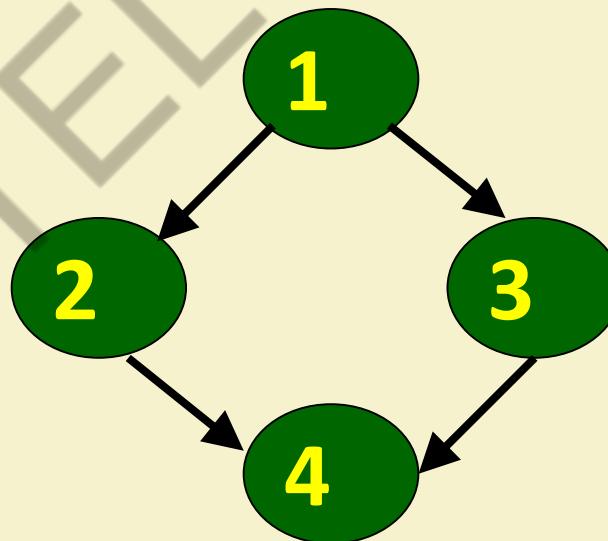


NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# How to Draw Control Flow Graph?

- Selection:

- 1 if( $a>b$ ) then
  - 2         $c=3;$
  - 3 else     $c=5;$
  - 4  $c=c*c;$



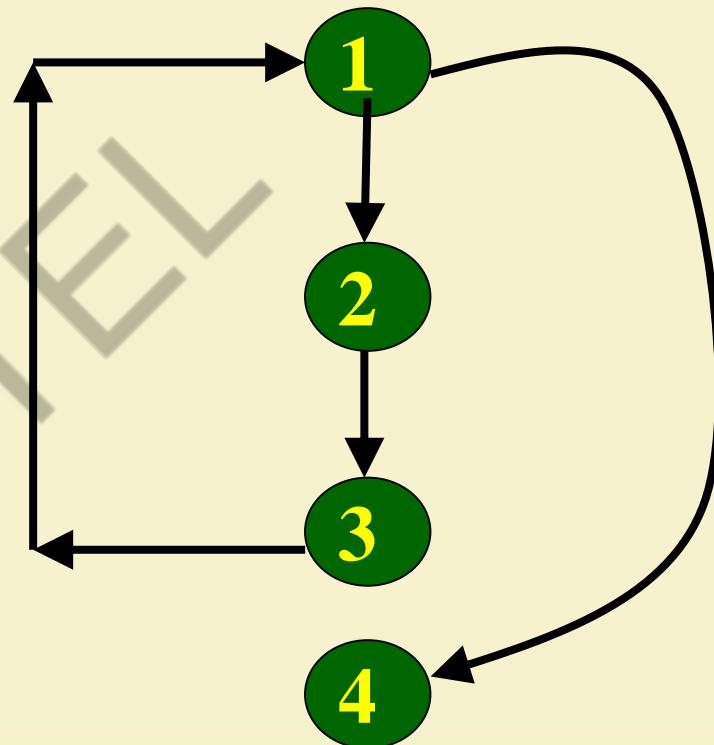
IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# How to Draw Control Flow Graph?

- Iteration:
  - 1 while( $a > b$ ){
  - 2      $b = b * a;$
  - 3      $b = b - 1;$ }
  - 4      $c = b + d;$



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Path

- A path through a program:
  - **A node and edge sequence from the starting node to a terminal node of the control flow graph.**
  - There may be several terminal nodes for program.



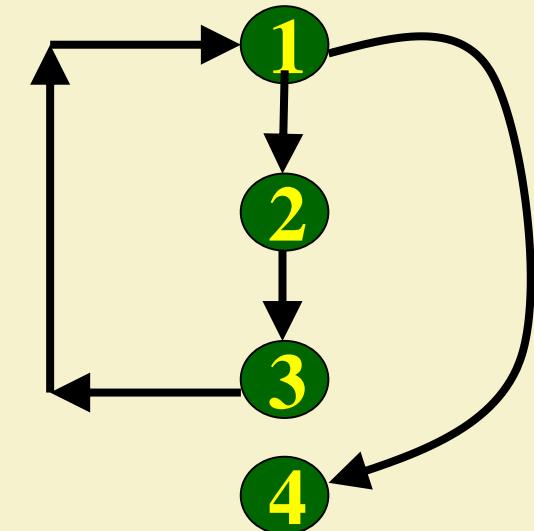
IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

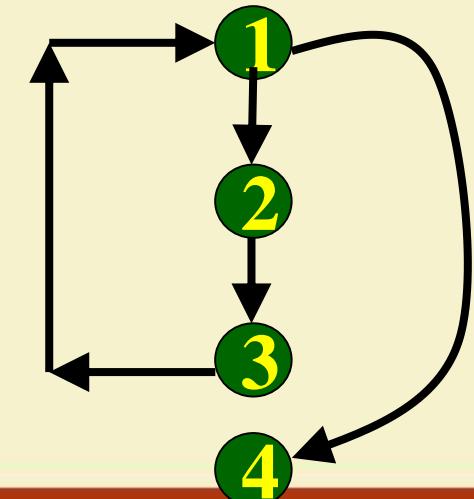
# All Path Criterion

- In the presence of loops, the number paths can become extremely large:
  - This makes all path testing impractical



# Linearly Independent Path

- Any path through the program that:
  - Introduces at least one new edge:
    - Not included in any other independent paths.



## Independent path

- It is straight forward:
  - To identify linearly independent paths of simple programs.
- For complicated programs:
  - It is not easy to determine the number of independent paths.

# McCabe's Cyclomatic Metric

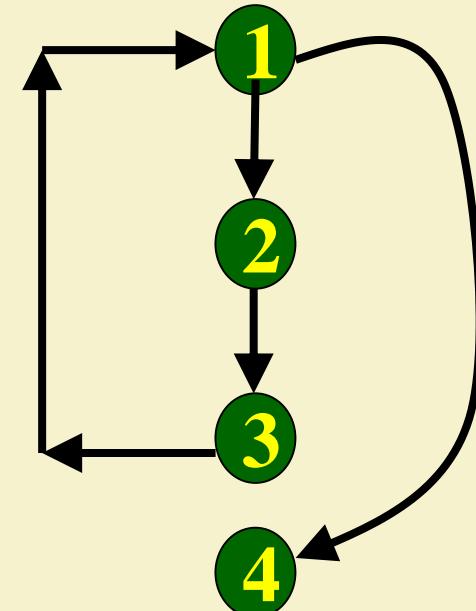
- An upper bound:
  - For the number of linearly independent paths of a program
- Provides a practical way of determining:
  - The maximum number of test cases required for basis path testing.

# McCabe's Cyclomatic Metric

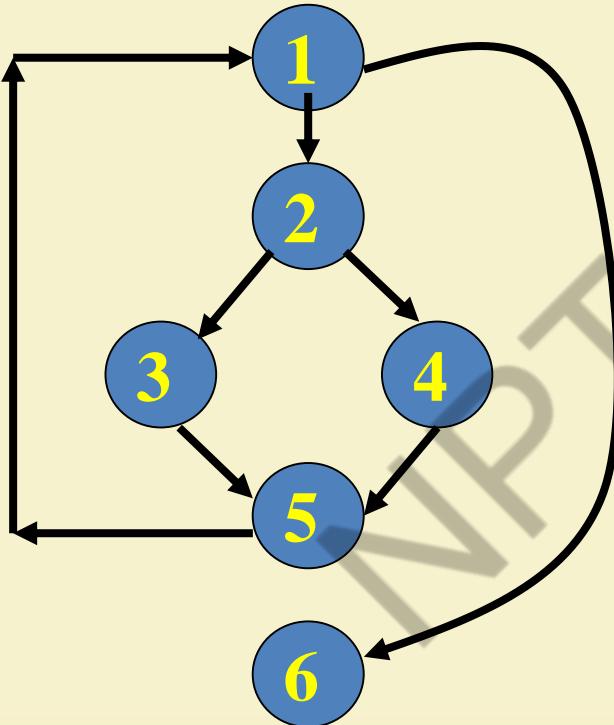
- Given a control flow graph  $G$ , cyclomatic complexity  $V(G)$ :

- $- V(G) = E - N + 2$

- $N$  is the number of nodes in  $G$
- $E$  is the number of edges in  $G$



# Example Control Flow Graph



Cyclomatic complexity =  
 $7 - 6 + 2 = 3.$



IIT KHARAGPUR

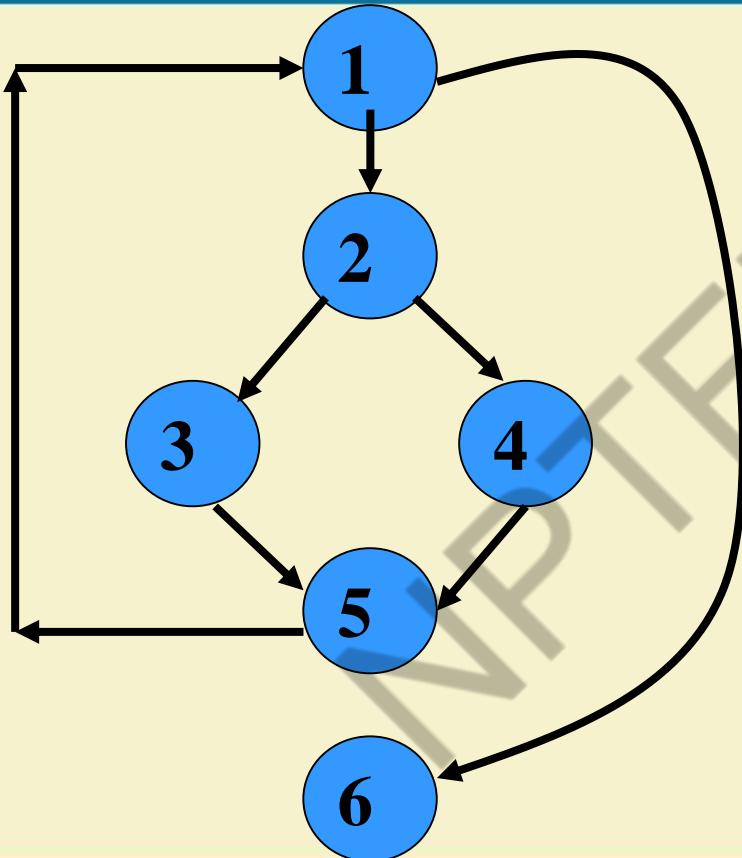


NPTEL ONLINE  
CERTIFICATION COURSES

# Cyclomatic Complexity

- Another way of computing cyclomatic complexity:
  - inspect control flow graph
  - determine number of bounded areas in the graph
- $V(G) = \text{Total number of bounded areas} + 1$ 
  - Any region enclosed by a nodes and edge sequence.

# Example Control Flow Graph



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Example

- From a visual examination of the CFG:
  - Number of bounded areas is 2.
  - Cyclomatic complexity =  $2+1=3$ .

# Cyclomatic Complexity

- McCabe's metric provides:
  - **A quantitative measure of testing difficulty and the reliability**
- Intuitively,
  - Number of bounded areas increases with the number of decision nodes and loops.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Cyclomatic Complexity

- The first method of computing  $V(G)$  is amenable to automation:
  - You can write a program which determines the number of nodes and edges of a graph
  - Applies the formula to find  $V(G)$ .



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Cyclomatic Complexity

- The cyclomatic complexity of a program provides:
  - A lower bound on the number of test cases to be designed
  - To guarantee coverage of all linearly independent paths.

# Cyclomatic Complexity

- Knowing the number of test cases required:
  - Does not make it any easier to derive the test cases,
  - Only gives an indication of the minimum number of test cases required.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Practical Path Testing

- The tester proposes initial set of test data :
  - Using his experience and judgment.
- A dynamic program analyzer used:
  - Measures which parts of the program have been tested
  - Result used to determine when to stop testing.



IIT KHARAGPUR



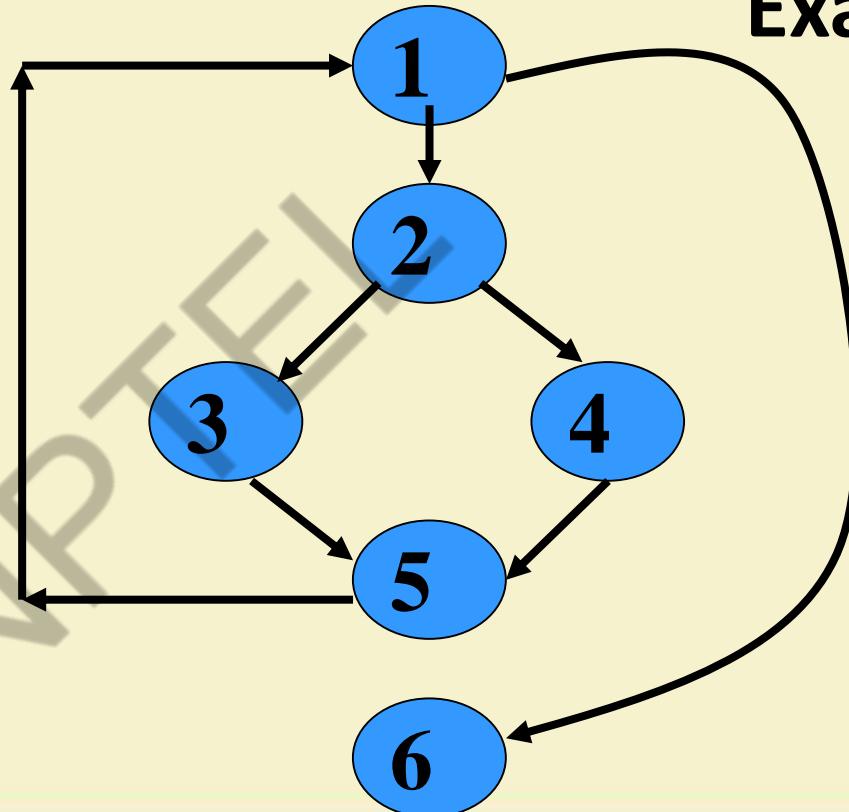
NPTEL ONLINE  
CERTIFICATION COURSES

# Derivation of Test Cases

- Draw control flow graph.
- Determine  $V(G)$ .
- Determine the set of linearly independent paths.
- Prepare test cases:
  - Force execution along each path.
  - Not practical for larger programs.

```
int f1(int x,int y){  
1 while (x != y){  
2   if (x>y) then  
3     x=x-y;  
4   else y=y-x;  
5 }  
6 return x;      }
```

## Example



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Derivation of Test Cases

- Number of independent paths: 3

–1,6    **test case (x=1, y=1)**

–1,2,3,5,1,6 **test case(x=1, y=2)**

–1,2,4,5,1,6 **test case(x=2, y=1)**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# An Interesting Application of Cyclomatic Complexity

- Relationship exists between:
  - McCabe's metric
  - The number of errors existing in the code,
  - Time required to correct the errors.
  - Time required to understand the program

# Cyclomatic Complexity

- Cyclomatic complexity of a program:
  - Indicates the psychological complexity of a program.
  - Difficulty level of understanding the program.

# Cyclomatic Complexity

- From maintenance perspective,
  - Limit cyclomatic complexity of modules
    - To some reasonable value.
  - Good software development organizations:
    - Restrict cyclomatic complexity of functions to a maximum of ten or so.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Dataflow and Mutation Testing



IIT KHARAGPUR

03/08/10



NPTEL

NPTEL ONLINE  
CERTIFICATION COURSES

46

# White Box Testing: Quiz

1. What do you mean by coverage-based testing?
2. What are the different types of coverage based testing?
3. How is a specific coverage-based testing carried out?
4. What do you understand by fault-based testing?
5. Give an example of fault-based testing?



IIT KHARAGPUR



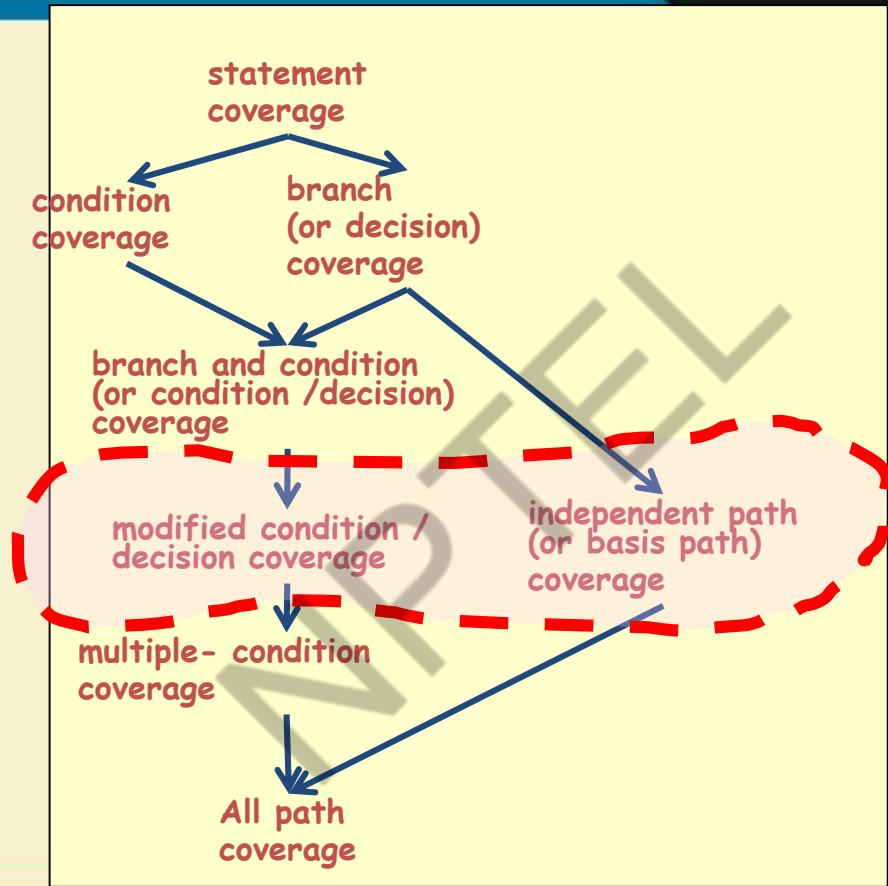
NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# White-Box Testing

Practically important coverage techniques

strongest

weakest



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data flow Testing



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Data Flow-Based Testing

- Selects test paths of a program:
  - According to the locations of
    - Definitions and uses of different variables in a program.

```
1 X(){}
2 int a=5; /* Defines variable a */
.....
3 While(c>5) {
4   if (d<50)
5     b=a*a; /*Uses variable a */
6     a=a-1; /* Defines as well uses variable a */
...
7 }
8 print(a); } /*Uses variable a */
```



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Data Flow-Based Testing

- For a statement numbered S,
  - $\text{DEF}(S) = \{X/\text{statement } S \text{ contains a definition of } X\}$
  - $\text{USES}(S) = \{X/\text{statement } S \text{ contains a use of } X\}$
  - Example: **1:  $a=b$** ;  $\text{DEF}(1)=\{a\}$ ,  $\text{USES}(1)=\{b\}$ .
  - Example: **2:  $a=a+b$** ;  $\text{DEF}(1)=\{a\}$ ,  $\text{USES}(1)=\{a,b\}$ .

# Data Flow-Based Testing

- A variable X is said to be **live** at statement S1, if
  - X is defined at a statement S:
  - There exists a path from S to S1 not containing any definition of X.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# DU Chain Example

```
1 X(){  
2 int a=5; /* Defines variable a */  
3 While(c>5) {  
4 if (d<50)  
5 b=a*a; /*Uses variable a */  
6 a=a-1; /* Defines variable a */  
7 }  
8 print(a); } /*Uses variable a */
```



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Definition-use chain (DU chain)

- $[X, S, S_1]$ ,
  - S and  $S_1$  are statement numbers,
  - $X$  in  $\text{DEF}(S)$
  - $X$  in  $\text{USES}(S_1)$ , and
  - the definition of  $X$  in the statement  $S$  is live at statement  $S_1$ .

# Data Flow-Based Testing

- One simple data flow testing strategy:
  - Every DU chain in a program be covered at least once.
- Data flow testing strategies:
  - Useful for selecting test paths of a program containing nested if and loop statements.

- 1 X(){
- 2 B1; /\* Defines variable a \*/
- 3 While(C1) {
- 4 if (C2)
- 5 if(C4) B4; /\*Uses variable a \*/
- 6 else B5;
- 7 else if (C3) B2;
- 8 else B3; }
- 9 B6 }

## Data Flow- Based Testing



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

## Data Flow-Based Testing

- [a,1,5]: a DU chain.
- Assume:
  - $\text{DEF}(X) = \{\text{B1, B2, B3, B4, B5}\}$
  - $\text{USES}(X) = \{\text{B2, B3, B4, B5, B6}\}$
  - There are 25 DU chains.
- However only 5 paths are needed to cover these chains.

# Mutation Testing



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Mutation Testing

- In this, software is first tested:
  - Using an initial test suite designed using white-box strategies we already discussed.
- After the initial testing is complete,
  - Mutation testing is taken up.
- The idea behind mutation testing:
  - Make a few arbitrary small changes to a program at a time.**



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES

# Main Idea

- Insert faults into a program:
  - Check whether the test suite is able to detect these.
  - This either validates or invalidates the test suite.



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Mutation Testing Terminology

- Each time the program is changed:
  - It is called a **mutated program**
  - The change is called a **mutant.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

# Mutation Testing

- A mutated program:
  - Tested against the full test suite of the program.
- If there exists at least one test case in the test suite for which:
  - A mutant gives an incorrect result,
  - Then the mutant is said to be dead.**

# Mutation Testing

- If a mutant remains alive ---even after all test cases have been exhausted,
  - The test suite is enhanced to kill the mutant.**
- The process of generation and killing of mutants:
  - Can be automated by predefining a set of primitive changes that can be applied to the program.**



IIT KHARAGPUR



NPTEL ONLINE  
CERTIFICATION COURSES

# Mutation Testing

- Example primitive changes to a program:

- Deleting a statement**
- Altering an arithmetic operator,**
- Changing the value of a constant,**
- Changing a data type, etc.**



IIT KHARAGPUR



NPTEL  
NPTEL ONLINE  
CERTIFICATION COURSES

## Traditional Mutation Operators

- **Deletion of a statement**
- Boolean:
  - **Replacement of a statement with another**  
eg. == and >=, < and <=
  - Replacement of **boolean expressions** with *true* or *false* eg. **a || b** with **true**
- **Replacement of arithmetic operator**  
eg. \* and +, / and -
- **Replacement of a variable** (ensuring same scope/type)

# Underlying Hypotheses

- Mutation testing is based on the following two hypotheses:
  - **The Competent Programmer Hypothesis**
  - **The Coupling Effect**

**Both of these were proposed by DeMillo *et al.*, 1978**

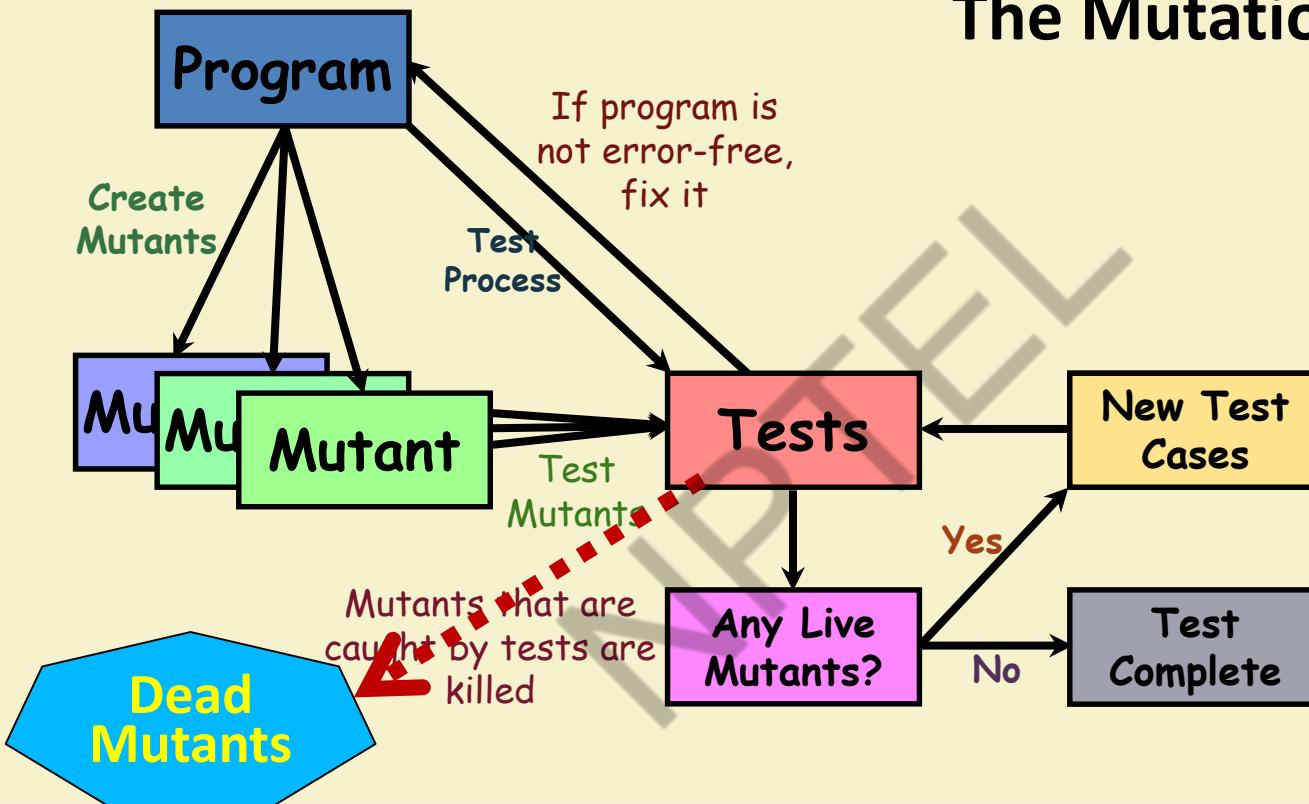
# The Competent Programmer Hypothesis

- Programmers create programs that are close to being correct:
  - Differ from the correct program by some simple errors.

# The Coupling Effect

- **Complex errors are caused due to several simple errors.**
- It therefore suffices to check for the presence of the simple errors

# The Mutation Process



IIT KHARAGPUR



NPTEL  
ONLINE  
CERTIFICATION COURSES