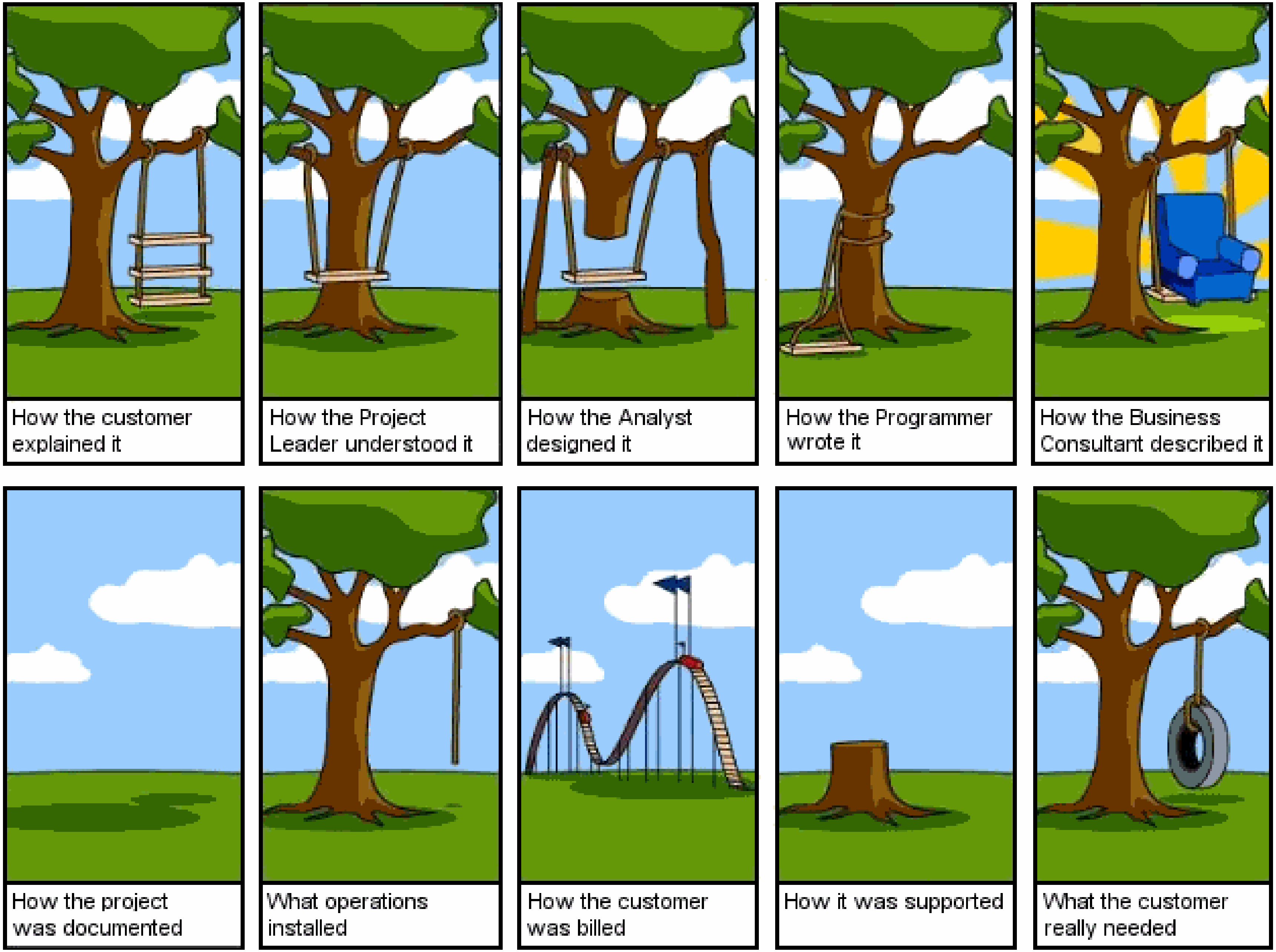
Software Development Lifecycle (SDLC)

Week 1 - DASS

Y. Raghu Reddy
Software Engineering Research Centre
IIIT Hyderabad, India


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Why Life cycle model?

- A software project will never succeed if activities are not coordinated:
  - 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

- Adherence can lead to accurate status reports

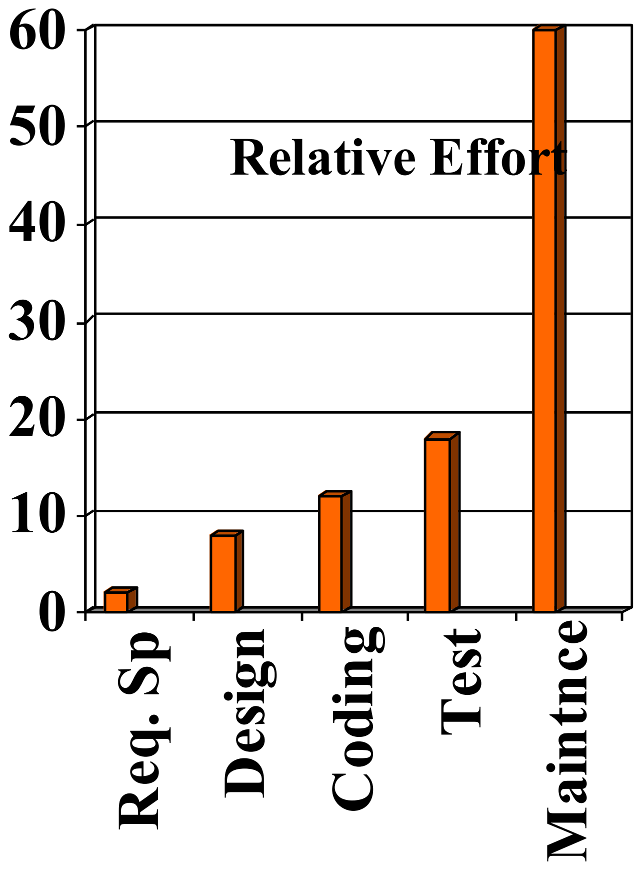
- Otherwise, it becomes very difficult to track the progress of the project
  - the project manager would have to depend on the guesses of the team members.

Software Development Life Cycle (SDLC)

- Typical software life cycle or software process consists of following phases:
  - Feasibility study (involves business case)
  - Requirements analysis and specification,
  - Design
  - Coding
  - Testing
  - Maintenance

Relative effort for Phases

- Phases between feasibility study and testing
  - known as development phases.
- Among all life cycle phases
  - maintenance phase consumes maximum effort.



Feasibility Study

- Main aim of feasibility study: determine whether developing the product
  - financially worthwhile
  - technically feasible.

- First roughly understand what the customer wants:
  - Inputs
  - Processing
  - Outputs
  - various constraints on the behaviour of the system

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
    - development time
- Perform a cost/benefit analysis:
    - you may determine that none of the solutions is feasible due to high cost, resource constraints, technical reasons.

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.

Goals of Requirements Analysis

- Collect all related data from the customer:
  - analyze the collected data to clearly understand what the customer wants,
  - ensure correctness, consistency and unambiguity.

Requirements Gathering

- Gathering relevant data:
  - usually collected from the end-users through interviews and discussions.
  - For example, for a business accounting software:
    - interview all the accountants of the organization to find out their requirements.

Requirements Analysis (CONT.)

- The data you initially collect from the users:
  - would usually contain several contradictions and ambiguities:
  - each user typically has only a partial and incomplete view of the system.
- Ambiguities and contradictions:
  - must be identified
  - resolved by discussions with the customers.
- Next, requirements are organized:
  - into a **Software Requirements Specification (SRS)** document

Design

- Design phase transforms requirements specification:
  - into a form suitable for implementation in some programming language.

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.

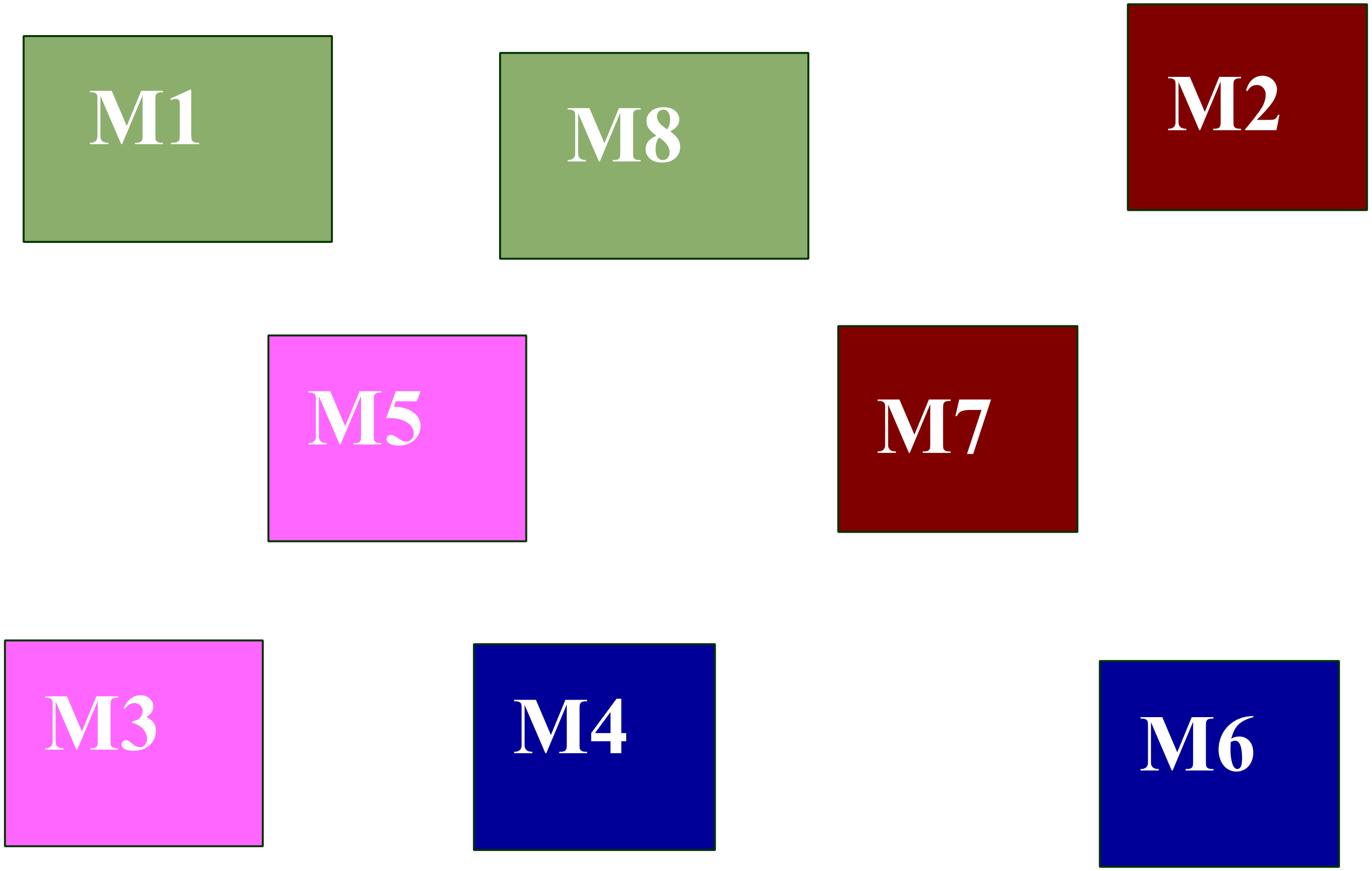
Implementation

- During the implementation phase:
  - each module of the design is coded,
  - each module is unit tested
    - tested independently as a stand alone unit, and debugged
- The purpose of unit testing:
  - test if individual modules work correctly.
- The end product of implementation phase:
  - a set of program modules that have been tested individually

Integration and System Testing

- Different modules are integrated in a planned manner:
  - modules are almost never integrated in one shot.
  - Normally integration is carried out through a number of steps.
- During each integration step,
  - the partially integrated system is tested.

Integration and System Testing



M1

M8

M2

M5

M7

M3

M4

M6

Maintenenace

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

Maintenance (CONT.)

- Preventive maintenance
  - Making appropriate changes to prevent the occurrence of errors
- 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

Summary

- A software life cycle model
  - is a descriptive and diagrammatic model of software life cycle
  - identifies all the activities required for product development,
  - establishes a precedence ordering among the different activities
  - divides life cycle into phases.
- A fundamental necessity while developing any large software product:
  - Adoption of a software development life cycle model (software process model).