Unit 1
Requirements

Software Analysis and Design Project

Computer Science

Universidad Autónoma de Madrid

Index

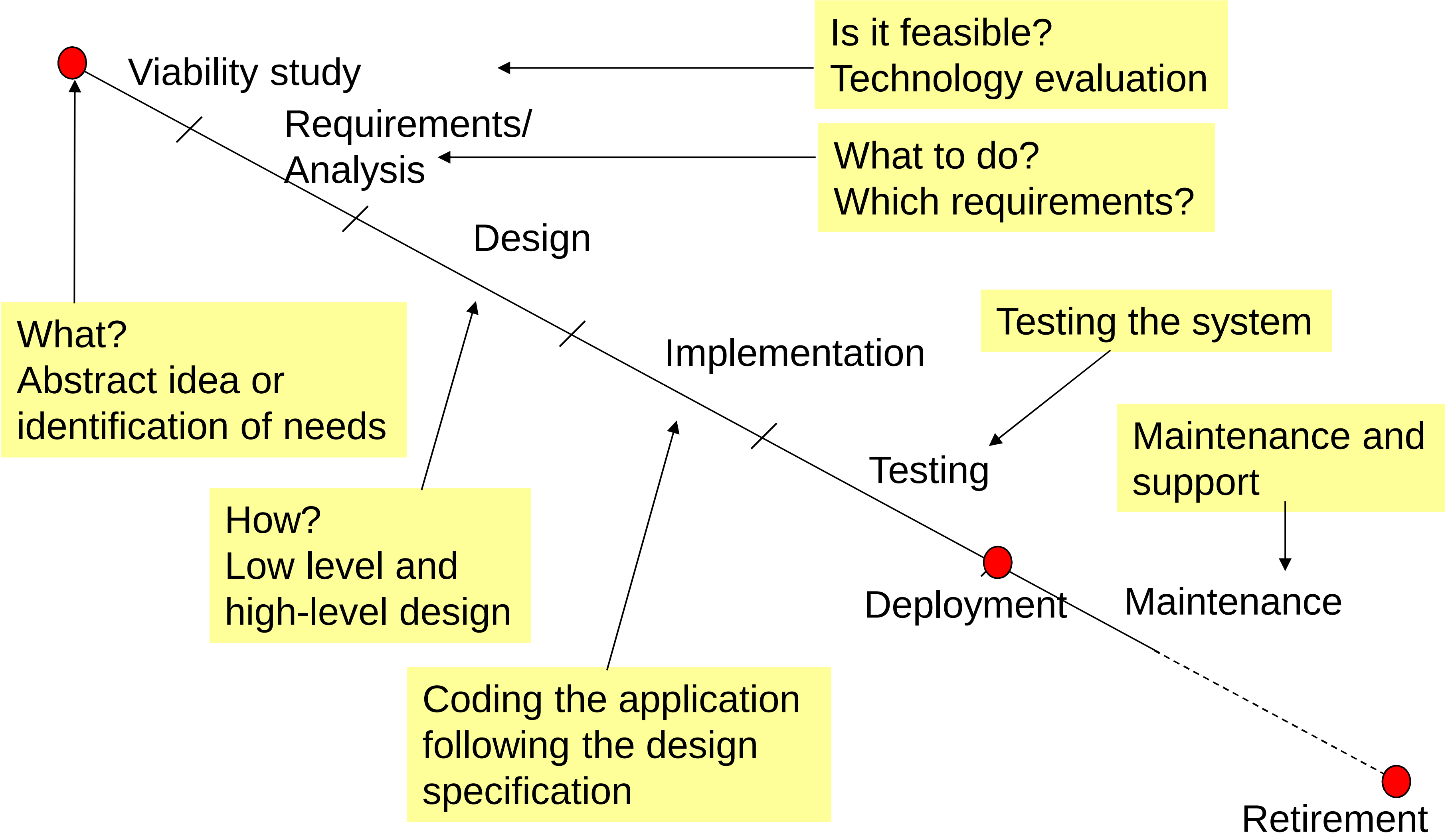
- **Introduction**
  - Phases and Software life-cycle model
- **Requirements**
  - Requirements elicitation techniques
  - Requirements representation techniques

Software development phases

Life-cycle

- Software development is not just programming
- Phases: Viability study, requirements, analysis, design, implementation, testing, maintenance.
- These phases are developed following a project plan.
- Similar to other fields:
    - Building a car is not only about soldering or screwing pieces together
    - Building a house is not just putting bricks
    - …

Software development life-cycle



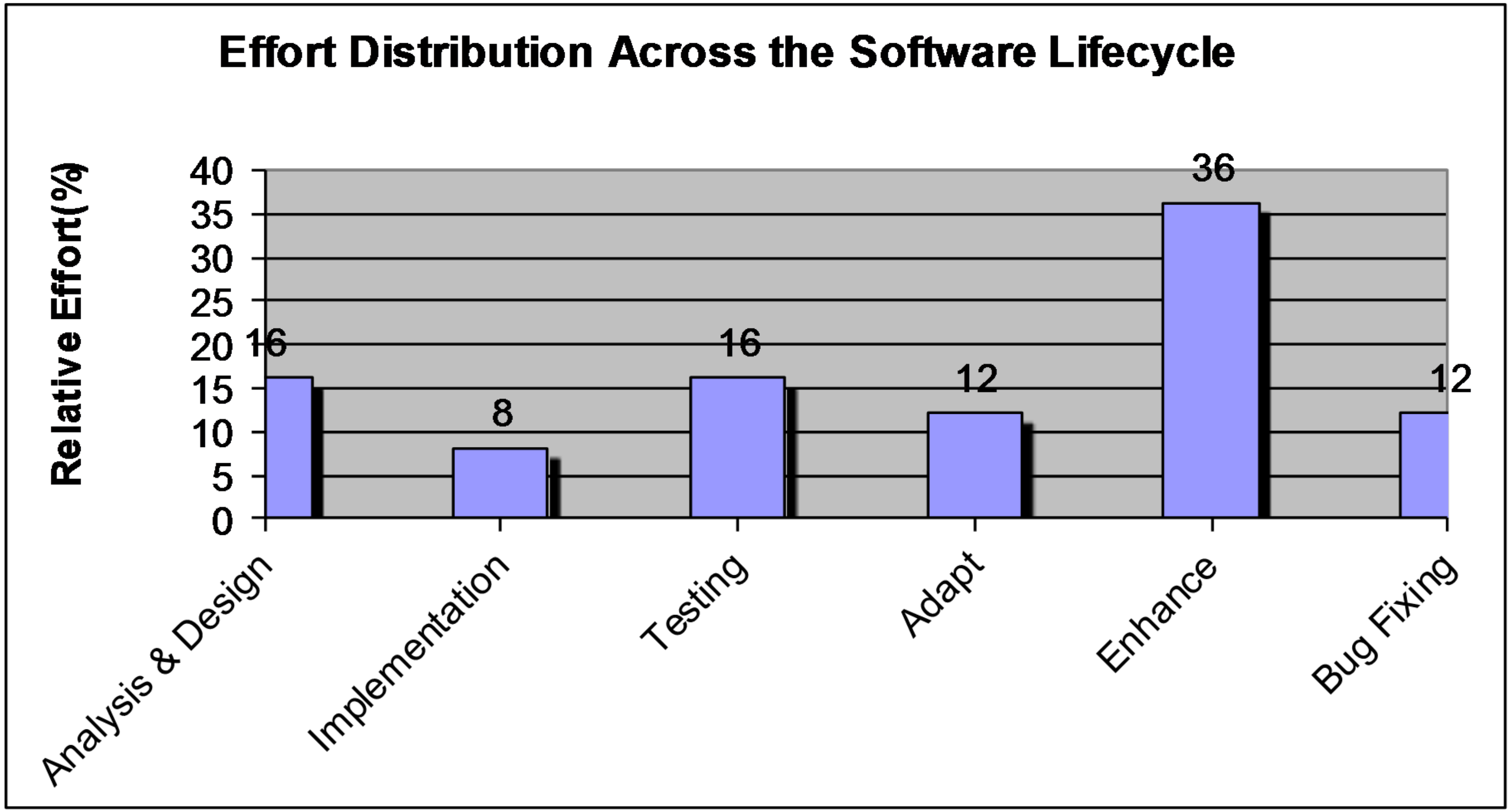
Life-cycle models

- There are different ways to arrange software development phases, in addition to a linear path

- Iterations:
  - □ Iterations between analysis and design phases

- Increments:
  - □ Firstly the application core is developed
  - □ Increments with secondary functionality

Software life-cycle



Index

- **Introduction**
  - Phases and Software life-cycle model
- **Requirements**
  - Requirement elicitation techniques
  - Requirement representation techniques

Requirements

- Needs that are to be solved by the software in order to be accepted by the client

- **Functional**:
  - Specific behaviours or functions. What a system is supposed to do

- **Non Functional**:
  - Operational (e.g. backup, recovery)
  - Security (e.g. access levels, protection)
  - Maintainability & portability (e.g. compatible with Windows & Linux)
  - Resources (e.g. memory, storage)
  - Performance (e.g. response time, scalability)
  - User interface & usability (e.g. size of visual elements)
  - Reliability (abnormal situations, or failures)
  - …

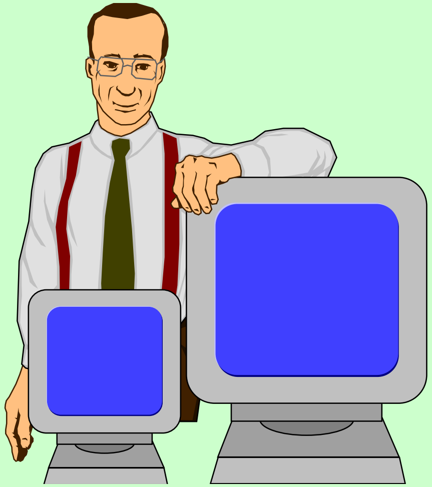
Requirements analysis

- Find answers for the following questions:
  - □ What needs to be done?
  - □ Which functionality do we need to implement?
  - □ Which are the non functional requirements? (e.g. performance, reliability)

"Problem analysis and complete specification of the external behaviour of the software system to be built, and the flow of information and control."

Requirements analysis

Customers and users describe their current problem, and the results and conditions they expect.



The software engineer makes questions, analyses, assimilates and presents the right solution.

Output: Requirements Analysis Document (RAD)

Requirements analysis

Important facts

- A program that solves the wrong problem is not useful to the client
- Understand what the customer wants before starting building the software
- If any requirement is missing:
  - The final application will be of lower quality
  - The customer is unhappy with the application
  - Has to be added in the maintenance phase: exponential cost
- If a requirement is incorrect:
  - Detect it as soon as possible (exponential cost as the project progresses)
- If the application does not meet the requirements:
  - The customer will not accept it
  - Modifications in the testing phase are very costly

Requirements analysis

Tasks

- Requirements **Elicitation** **(gathering)**
  - □ Identify requirements obtained from users and customers.

- **Analysis** of the problem and the requirements
  - □ Reasoning about requirements, combining related requirements, prioritize them, determine their feasibility, etc..

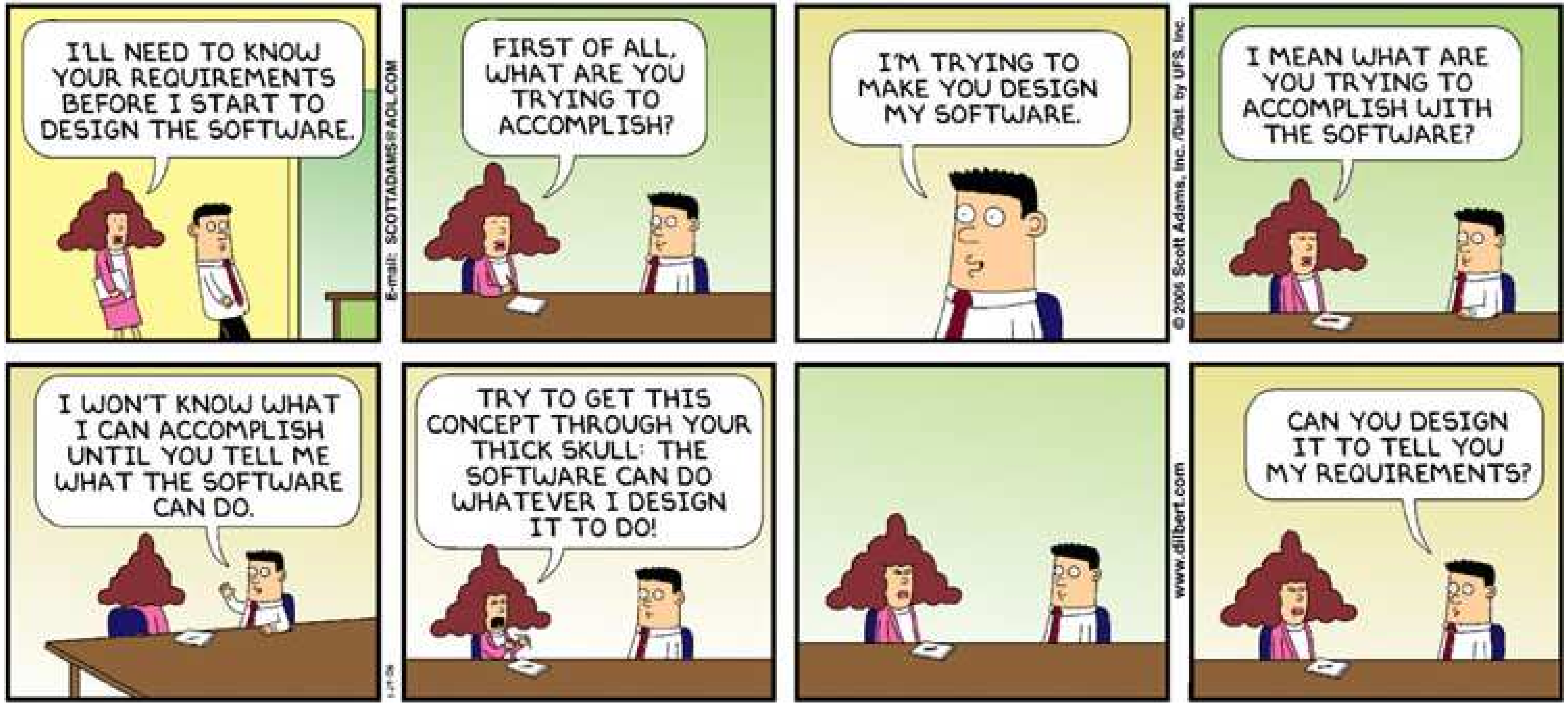
- **Representation** (modelling).
  - □ Register requirements using some method, including natural language, formal languages, models, mockups, etc..

- **Validation**.
  - □ Browse inconsistencies between requirements, determine their correctness, ambiguity, etc.. Establish criteria to ensure that the software meets the requirements as soon as possible (with which the client, user and developer should agree).

Requirements elicitation

- Also called “*requirements gathering*”
- It is not trivial, can not just be collected
- Techniques:
  - Interviews
  - Brainstorming
  - JAD (Joint Application Design)

Requirements elicitation


I'LL NEED TO KNOW YOUR REQUIREMENTS BEFORE I START TO DESIGN THE SOFTWARE.
E-mail: SCOTTADAMS@AOL.COM
FIRST OF ALL, WHAT ARE YOU TRYING TO ACCOMPLISH?
I'M TRYING TO MAKE YOU DESIGN MY SOFTWARE.
© 2006 Scott Adams, Inc. /Dist. by UFS, Inc.
I MEAN WHAT ARE YOU TRYING TO ACCOMPLISH WITH THE SOFTWARE?
I WON'T KNOW WHAT I CAN ACCOMPLISH UNTIL YOU TELL ME WHAT THE SOFTWARE CAN DO.
TRY TO GET THIS CONCEPT THROUGH YOUR THICK SKULL: THE SOFTWARE CAN DO WHATEVER I DESIGN IT TO DO!
www.dilbert.com
CAN YOU DESIGN IT TO TELL YOU MY REQUIREMENTS?

Requirements elicitation

1. Analysis of the problem.
2. Identify potential users.
3. Identify relevant sources of knowledge / information.
4. Gather information and facts.
5. Prepare and ask concise and direct questions.
6. Analyse the information obtained.
7. Evaluate findings (with respect to cost, techniques,…).
8. Prioritize.
9. Check the result with users.
10. Synthesize information in the form of specifications.
11. Determine issues not analysed yet.

Project

- Capture project requirements to be developed throughout the course.
- Write the Requirements Analysis Document, using the provided template that you can find in Moodle.

Bibliography

- Basic bibliography:
  - Software engineering a practitioner's approach, 7ªed. Roger Pressman. McGraw Hill Higher Education, 2010. INF/681.3.06/PRE.
  - Software engineering, 9ª ed. Addison Wesley. Ian Sommerville. INF/681.3.06/SOM.
- Recommended bibliography:
  - 830-1998 IEEE Recommended Practice for Software Requirements Specifications (Available using IEEE Xplore).