SOFTWARE ENGINEERING LAB

Yesoda Bhargava

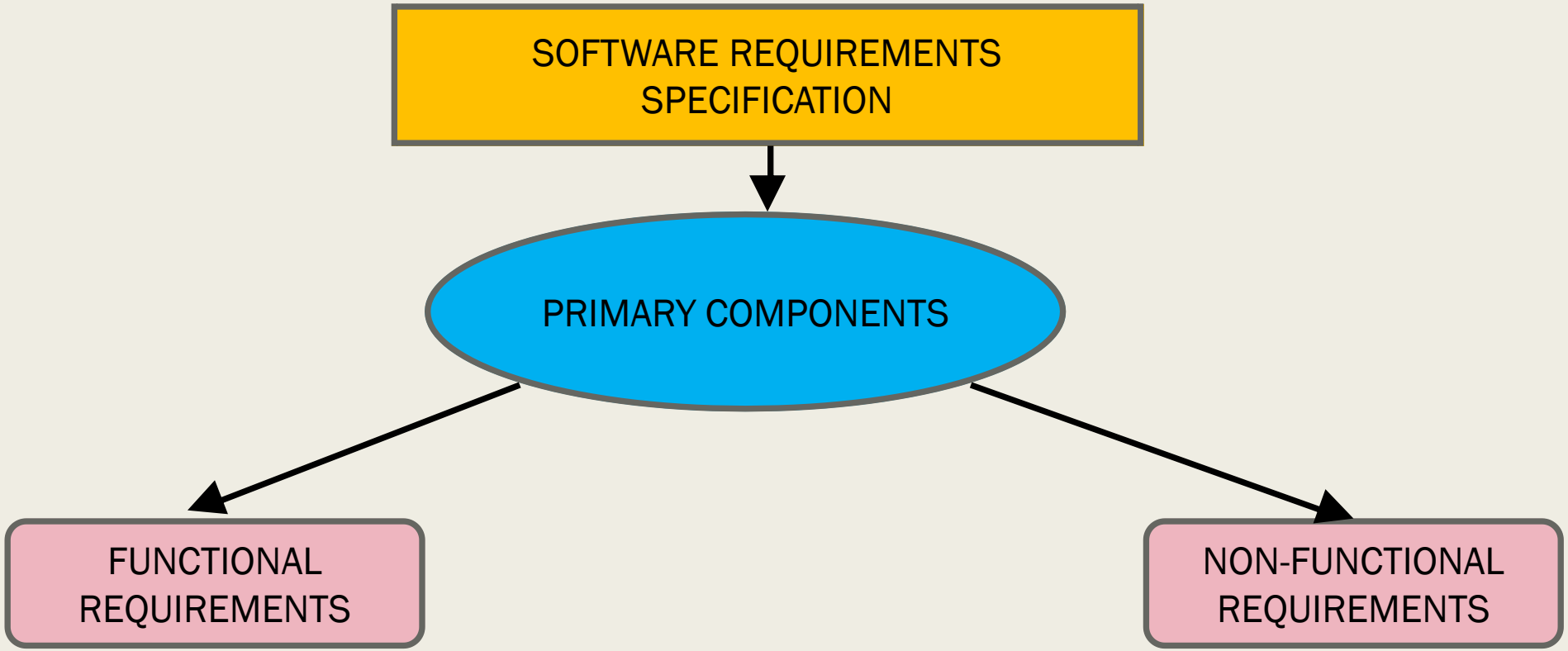
Goal

- Understand how to extract and identify requirements from a problem statement.
- Long term Goal : Understanding Software Requirement Specifications in general through your project work.

Why Bother?

- Requirement identification is the first step in software development project.
- Until the requirements of a client have been clearly,
  - *Identified,*
  - *Verified,*
- No other task can begin (design, coding, testing).

Primary Software Requirements Type

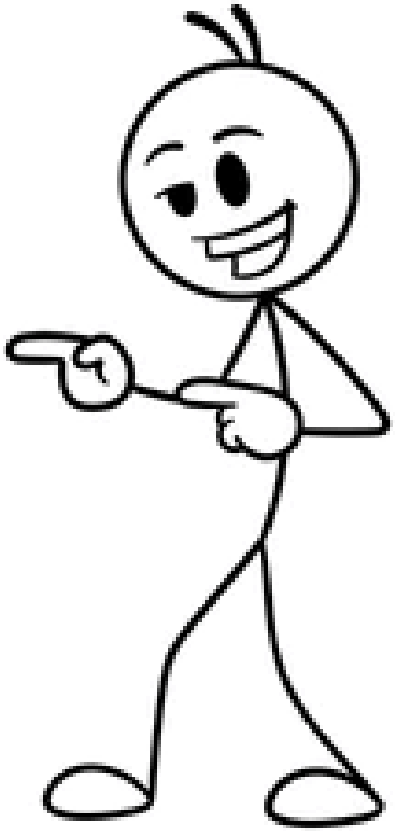


Lab Objective

- To identify functional and non-functional requirements from the problem statement.
- Post the lab session, you should be able to:
  - *Identify ambiguities, inconsistencies and incompleteness from a requirements specification*
  - *Identify and state functional requirements*
  - *Identify and state non-functional requirements*

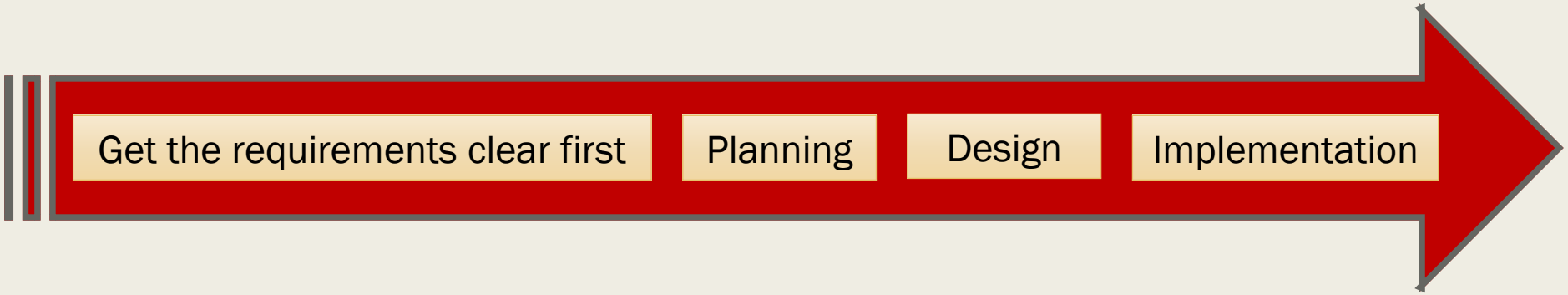
What is a requirement?

A specification of what should be implemented. How the target system should behave.



Requirements engineering

- The process of understanding what a customer expects from the system to be developed.
- To document these in a standard and easily readable and understandable format.



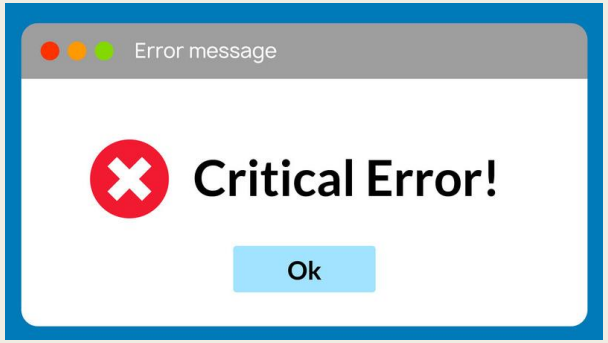
The cost of not understanding the requirements properly?

Customer dissatisfaction

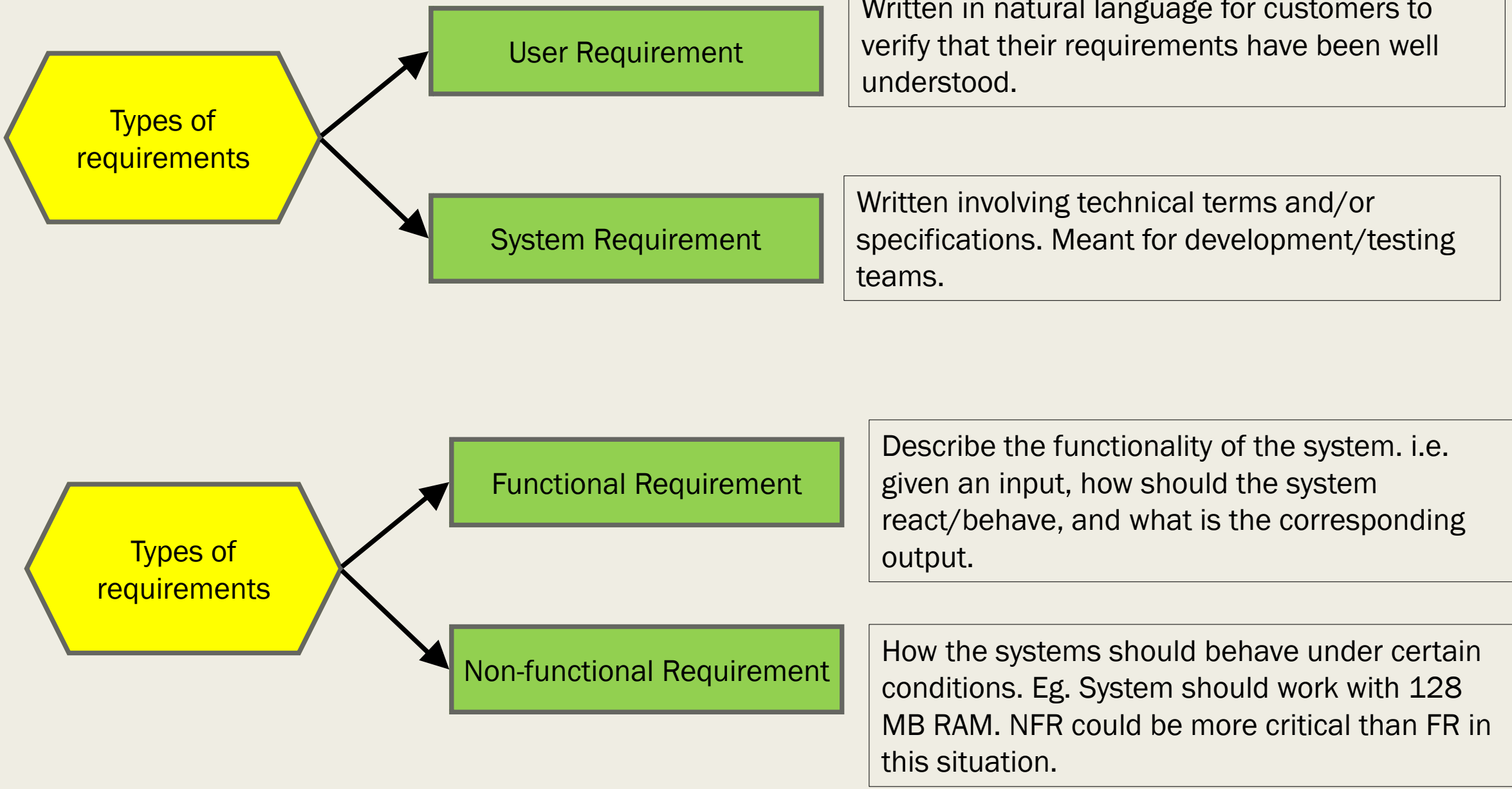
Loss of business

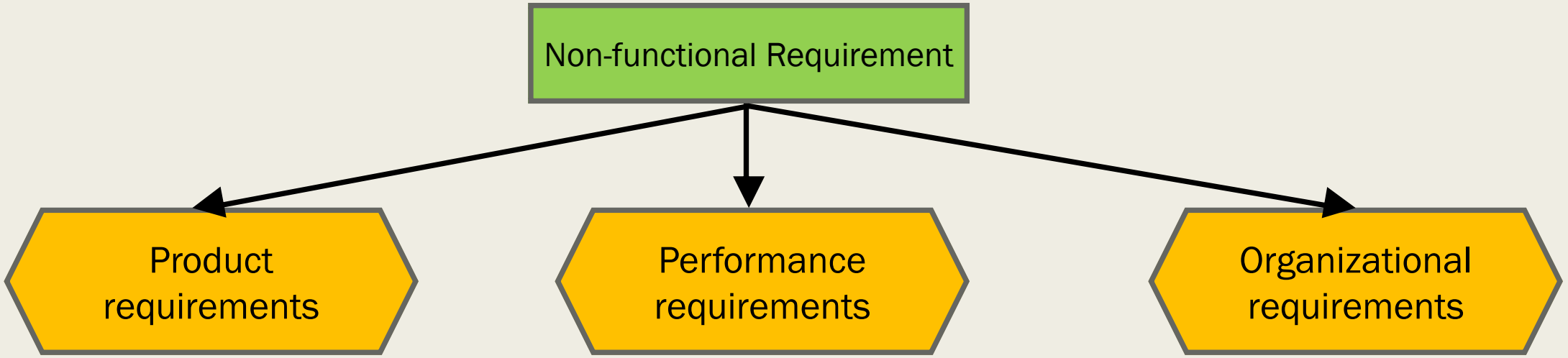
Characteristics of the requirements

- Requirements gathered for any new system should fulfill the following characteristics:
- **UNAMBIGUITY**: No ambiguity in what the system developed should do. Example: Website creation for a client. Client requires enough visitors should be able to access the website at the same time. What is “enough” to you? May be 10 to you, but 100 to the client. This is an ambiguity.
- **CONSISTENCY**: Consider automation of a nuclear plant. One developer says, that all reactors should shut down, when the radiation level exceed a threshold of R1, whereas, the other suggests the threshold to be kept at R2. There is an inconsistency among the developers for the level of threshold to be set in the nuclear plant to shut down the reactors.
- **COMPLETENESS**: Not only specify what the system should do, but also what is SHOULD NOT do. For example: ATM machine. What happens if you do not specify the maximum permissible amount that can be withdrawn? How is it taken care of?



Once you have identified your requirements, go through this check of unambiguity, consistency and completeness.


Types of requirements
User Requirement
Written in natural language for customers to verify that their requirements have been well understood.
System Requirement
Written involving technical terms and/or specifications. Meant for development/testing teams.
Types of requirements
Functional Requirement
Describe the functionality of the system. i.e. given an input, how should the system react/behave, and what is the corresponding output.
Non-functional Requirement
How the systems should behave under certain conditions. Eg. System should work with 128 MB RAM. NFR could be more critical than FR in this situation.



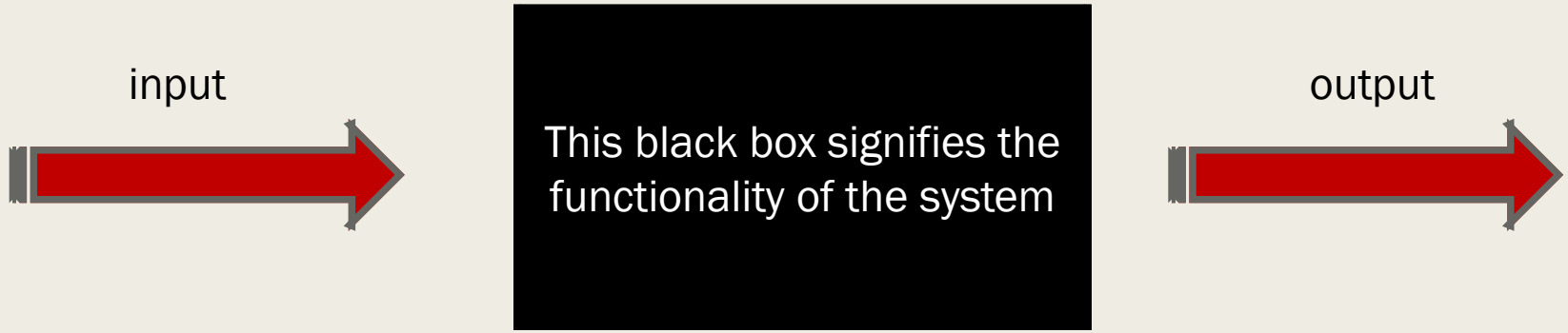
E.g. Website should only use plain HTML and no frames.

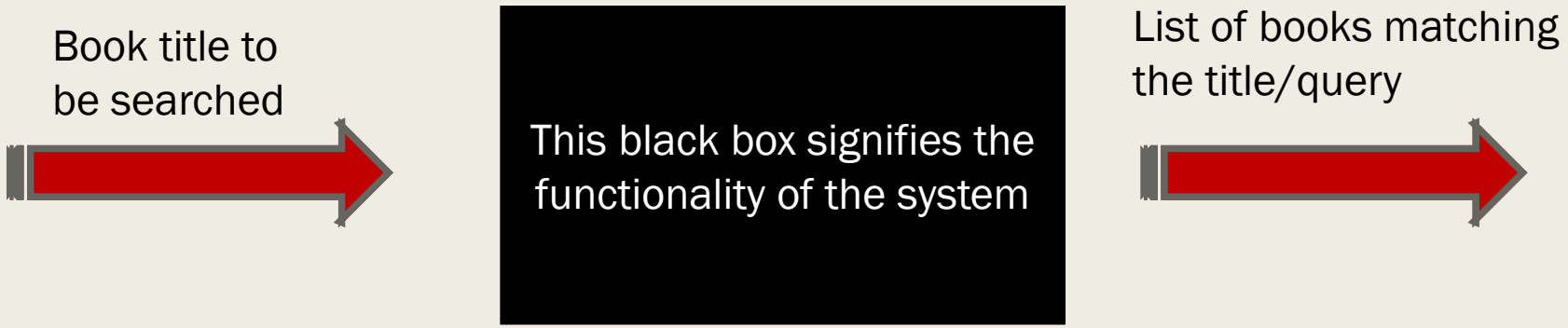
E.g. System should remain available 24 X 7

E.g. Development process should comply to SEI CMM Level 4.

SEI stands for 'Software Engineering Institute' at Carnegie-Mellon University, initiated by the U.S. Defense Department to help improve software development processes.
CMM stands for 'Capability Maturity Model', developed by the SEI. It's a model of 5 levels of organizational 'Maturity' that determine effectiveness in delivering quality software.

What are functional requirements?

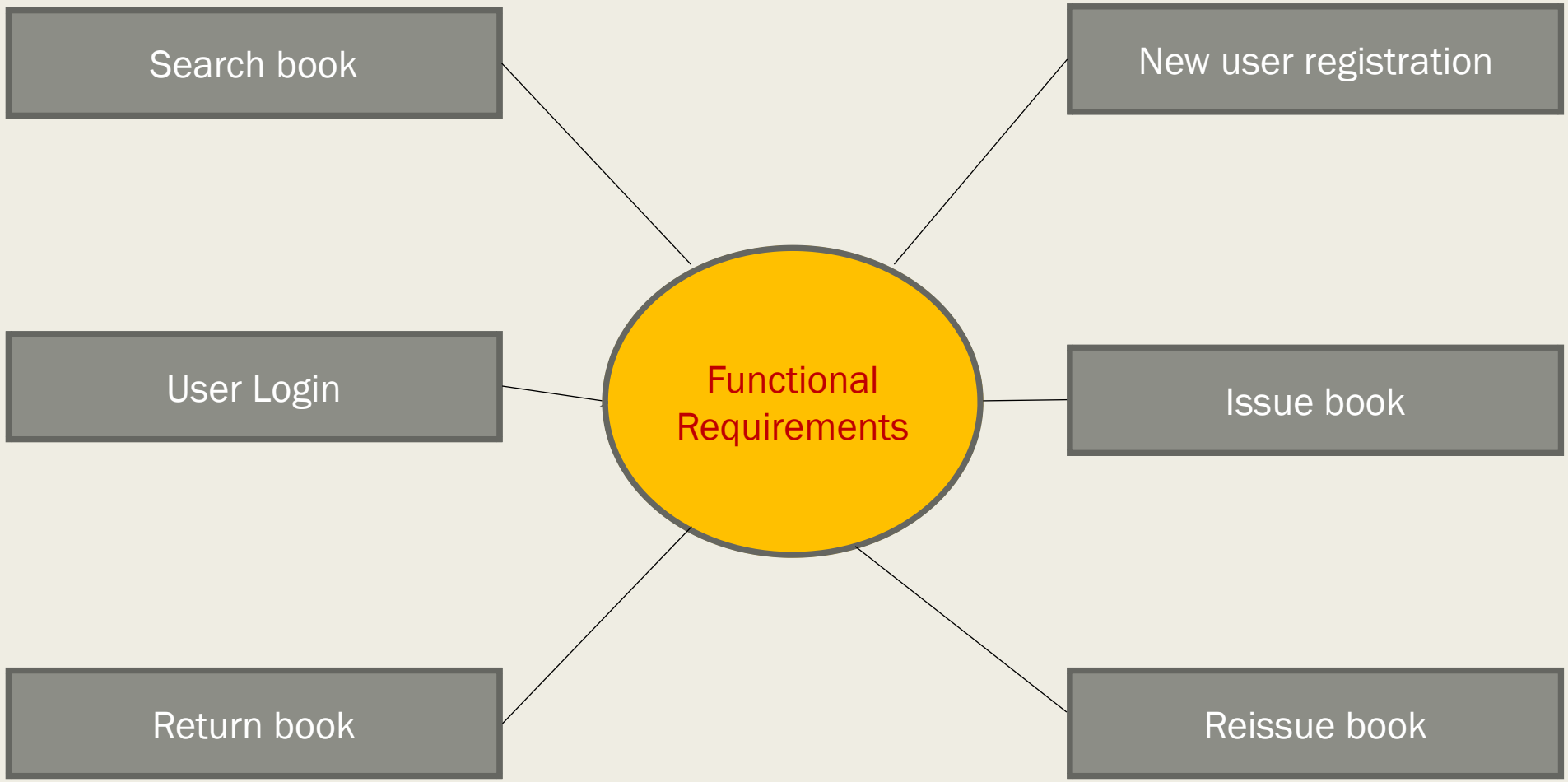




Identifying functional requirements

- Given a problem statement, the functional requirements could be identified by using following approach:
- Identify the high level functional requirement, from the conceptual understanding of the problem. Eg. A library management system (LMS) should be able to issue and return the books.
- Identify the use-case scenarios. Eg. The user is able to search the books available to obtain information about the desired book.
- High level requirement could have sub-requirements. Eg. “Issue book module” could behave differently for different classes of users. Those who have got the book issued thrice in a sequence. Or, Amazon customers who have been continuously refusing the orders at the door step could be issued a warning.

Example : Library Information System



All these are evidence problem functional requirements.

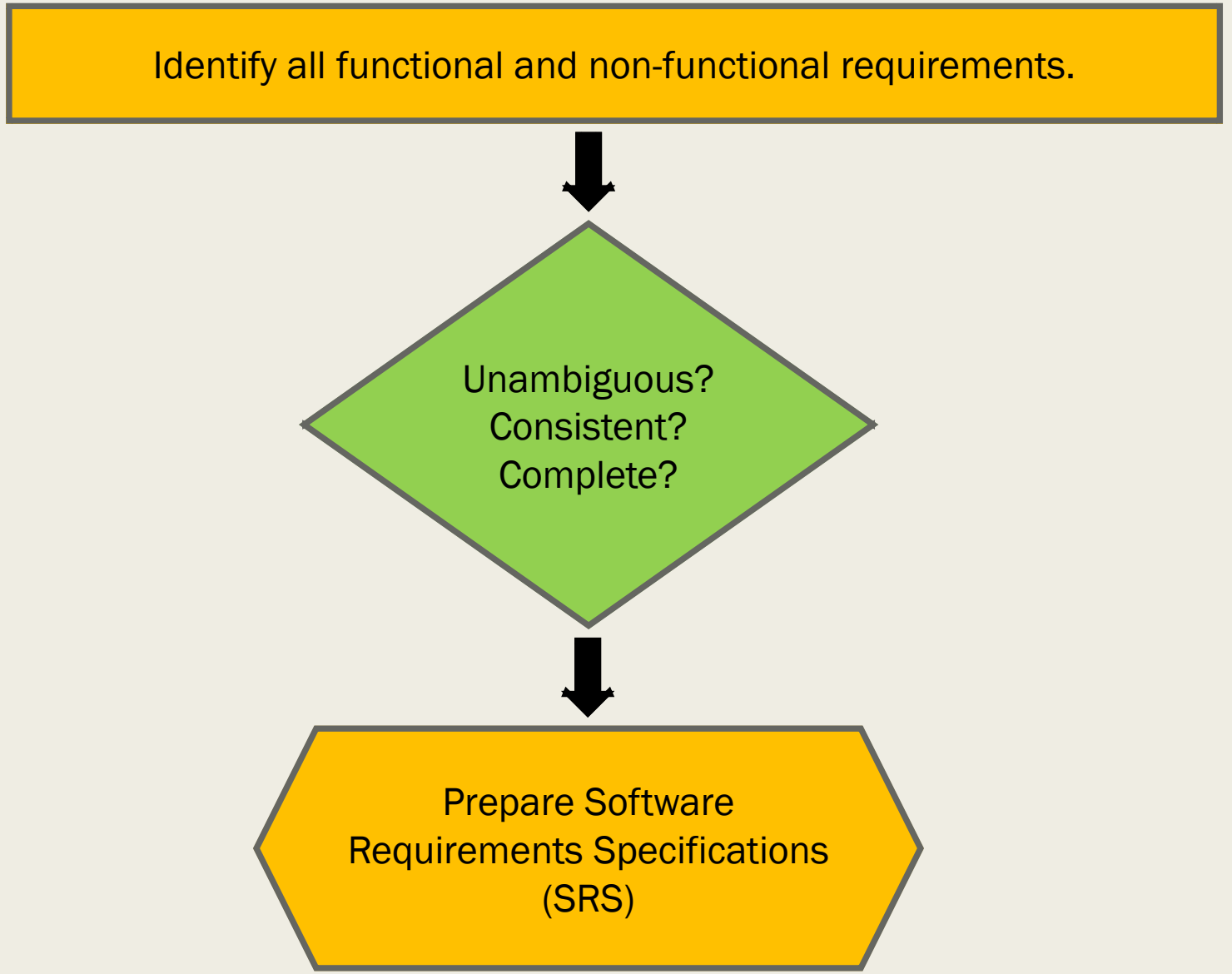
- Other not-so-visible requirements could be:
  - *User verification*
- How to figure out these?
  - *Think deeply about what you are creating, from client point of view.*
  - *Discuss with client actively to ensure you have understood the requirements.*
- Prioritize your software requirements.

Requirement	Identification	Priority
R1	New user registration	High
R2	User Login	High
R3	Search book	High
R4	Issue book	High
R5	Return book	High
R6	Reissue book	Low

Identifying non-functional requirements

- **Performance requirements**
  - *Accessibility of the system 24 X 7*
  - *At a given point of time, at least 60 users should be able to access the system.*
- **Security requirements**
  - *System available only within institute LAN.*
  - *Passwords should be saved via hash value.*
- **Software Quality Attributes**
  - *Reliability, Compatibility, Maintenance etc.*
- **Data base requirements**
  - *Kind of data to be stored.*
  - *Dynamic or static data?*
  - *Data organization, access, protection?*
- **Design constraints**
  - *Should be developed as a web application, work with all major browsers (Chrome, Firefox, Safari, Opera etc.)*
  - *System developed by HTML5.*

Preparing software requirements specification



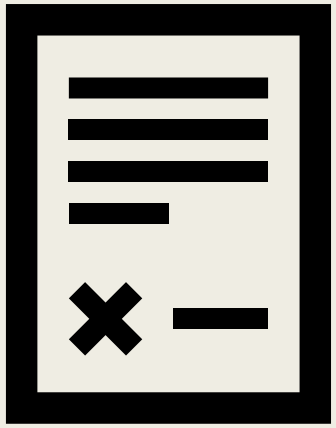
Software Requirements Specification

- Prepared by the service provider.
- Verified by the client.
- Serves as a legal agreement between the two parties.

Final software

Client check

Proposed feature not found.

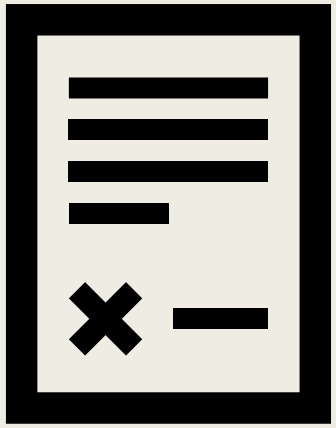


Client checks the SRS.

Final software

Client check

New feature required.



Service providers point out to SRS.

To do

- See this SRS IEEE template.
- Create an SRS document for your project.
- This is the submission required for your Formative Assessment.
- Deadline: 31st August 2021.