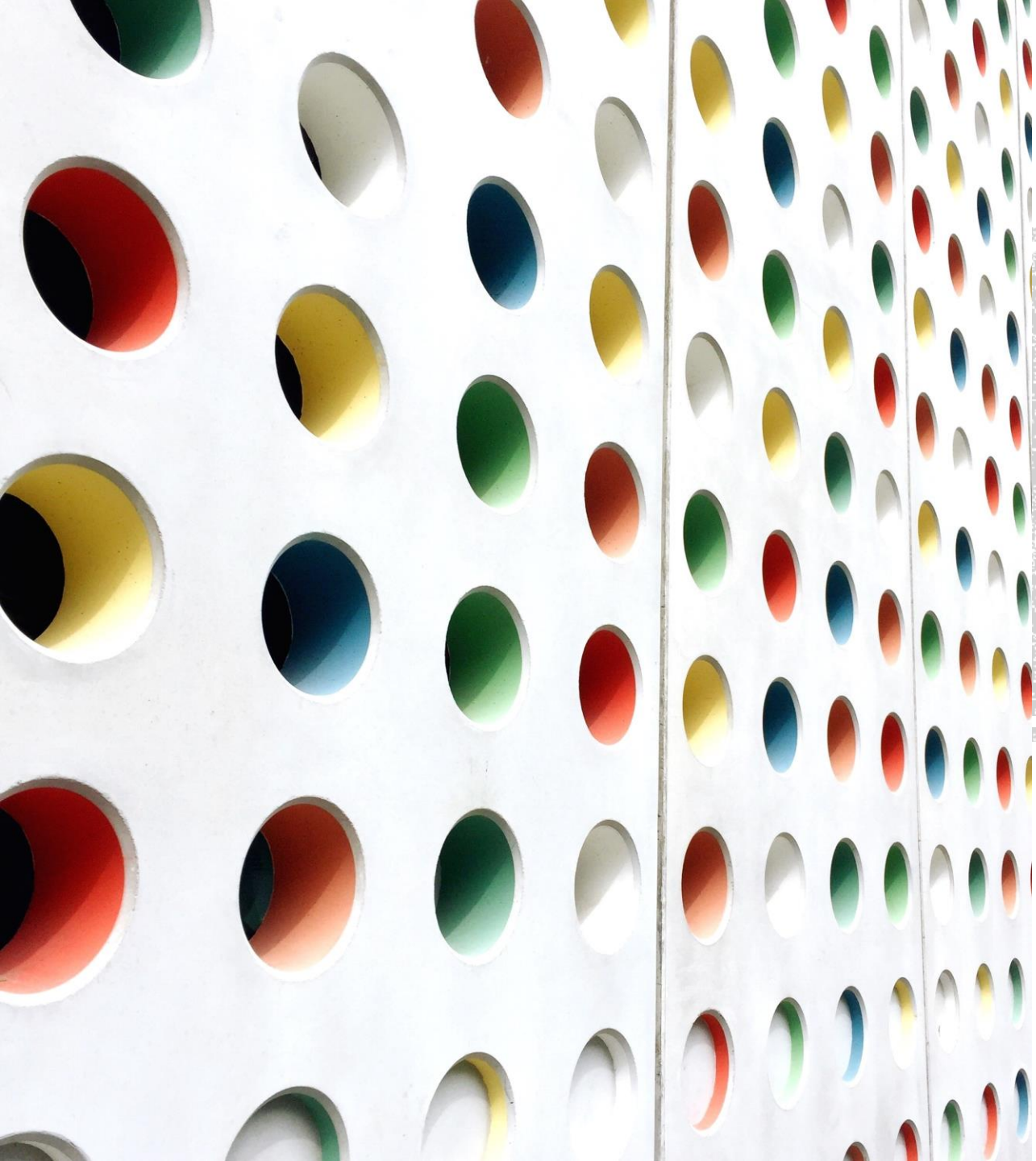


SOFTWARE REQUIREMENTS

Yesoda Bhargava

WHY BOTHER TO LOOK DEEPER INTO SOFTWARE REQUIREMENTS?

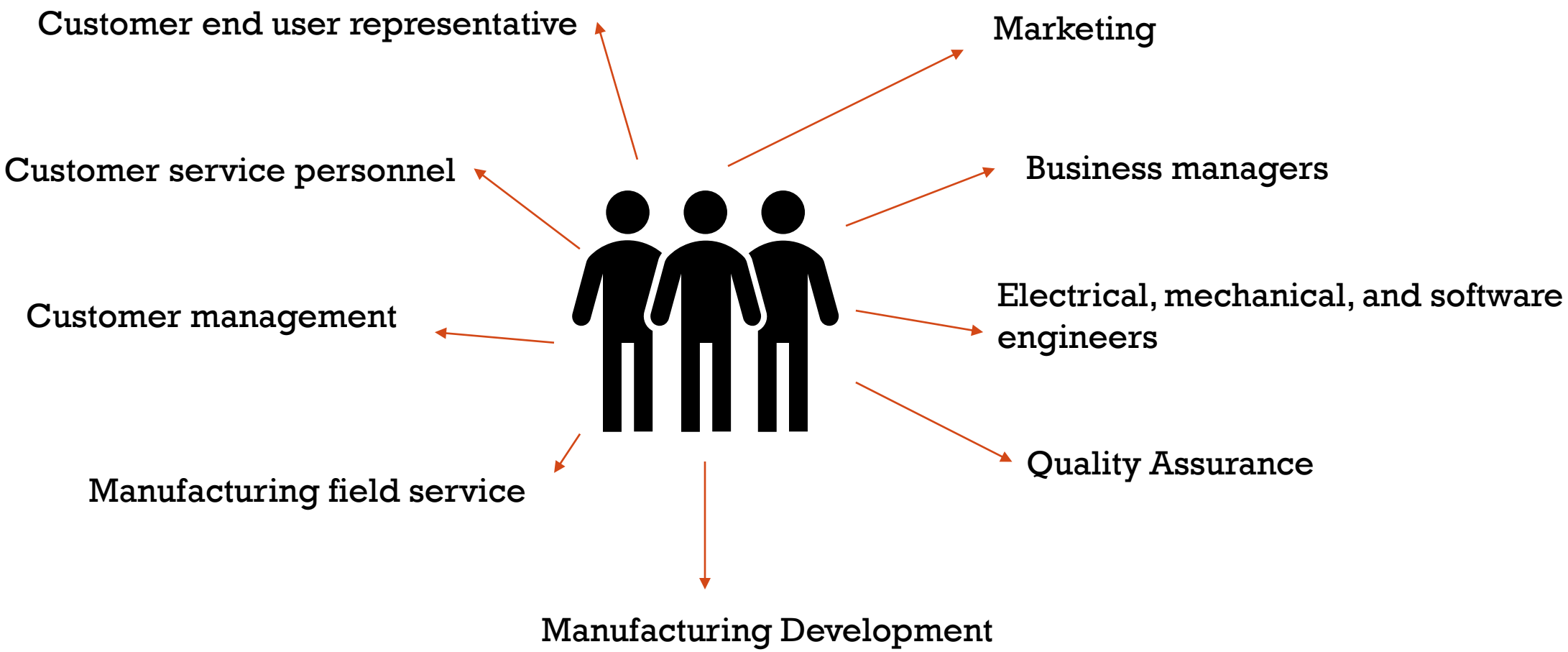
- Ever confused or not sure if you really want a certain feature in your product?
- Or if that feature will add value to your services?
- Without requirements testers do not know what to test.
- Developers wouldn't know what is considered complete.
- Customers do not know what to expect.
- bugs may be introduced due to unclear requirements or a misunderstanding of requirements.

WHAT IS REQUIREMENT GATHERING?

- Most important activity in the software development lifecycle.
- The phase consists of researching and discovering requirements for a certain product or application to be built or a business expectation to be fulfilled.
- Usually performed by a team of
  - Product Owners
  - Technical Leads
  - Designers
  - Developers
  - Involving clients, customers, users.

Requirements provide stakeholders with one uniform vision and set of goals. Each stakeholder will be able to understand the requirements and hold realistic expectations for the final product.

Who are stakeholders in requirements gathering?

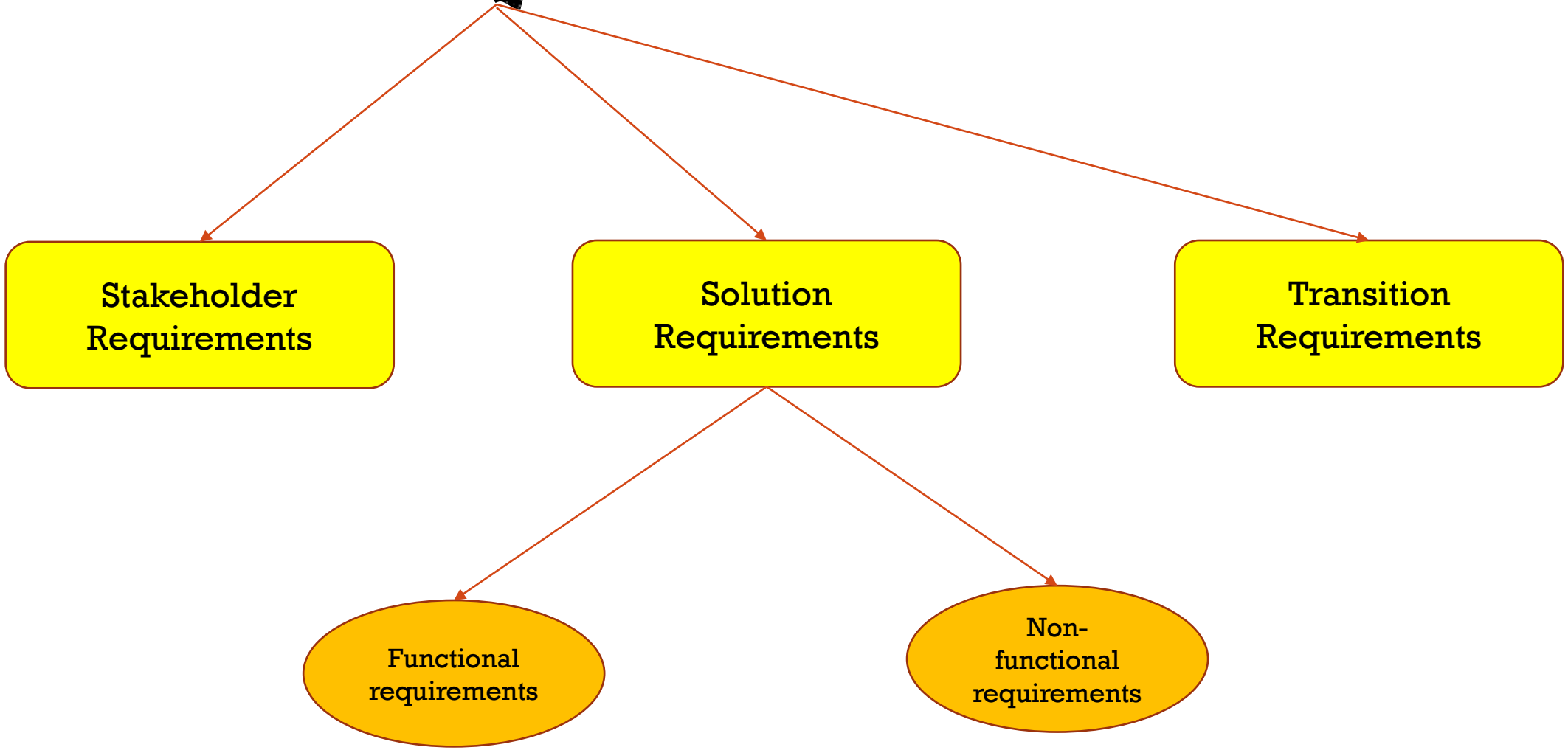


WHAT ARE SOFTWARE REQUIREMENTS?

- On the surface it sounds simple.
- The software must do X for Y so that Z.
- According to IEEE SWEBOK (Software Engineering Body of Knowledge), the area of Software Requirements is,
- **“concerned with the elicitation, analysis, specification, and validation of software requirements as well as the management of requirements during the whole life cycle of the software product”.**
- The piece of requirement should be ‘actionable’ in order to solve or trigger a certain scenario.
- ‘Actionable’ means something that can be implemented within a certain time frame.

There is an entire branch of engineering known as "Requirements Engineering" solely focused on requirements process.

TYPES OF REQUIREMENTS



STAKEHOLDER REQUIREMENT

- Problem statements from a user's, customer's or any other stakeholder's perspective.
- These requirements focus on 'what' of the problem, rather than the 'how' and 'why'.

SOLUTION REQUIREMENTS

- Solution Requirements describe how the execution of the project will be done in order to fulfil the conditions of the business stakeholders and users.
- **Functional requirements**
  - Detail how the product should respond and behave.
  - Could be the various features or functions the product must have which can be captured as a use case.
- **Non-functional requirements**
  - Describe the various characteristics of the solution such as quality, scalability, operations, performance etc, which in turn support the functional requirements.

TRANSITION REQUIREMENTS

- Conditions that need to be fulfilled when a certain product or a solution is getting shifted to a different or a new phase in order to support smooth migration.

FUNCTIONAL REQUIREMENTS

- Describe the functions that the software is to execute.
- E.g. Providing a communication channel for a user or transferring data from one format to another.
- Also known as product's features or capabilities.
- Basic facilities that the system should offer.
- Represented or stated in the form of input to be given to the system, the operation performed and the output expected.
- Basically requirements one can see directly in the final product, unlike the non-functional requirements.
- Functional requirements define what a software system must do or not do. Typically expressed as responses to inputs or conditions.
- E.g. In a hospital management system, a doctor should be able to retrieve the information of his patients.

EXAMPLES OF FUNCTIONAL REQUIREMENTS

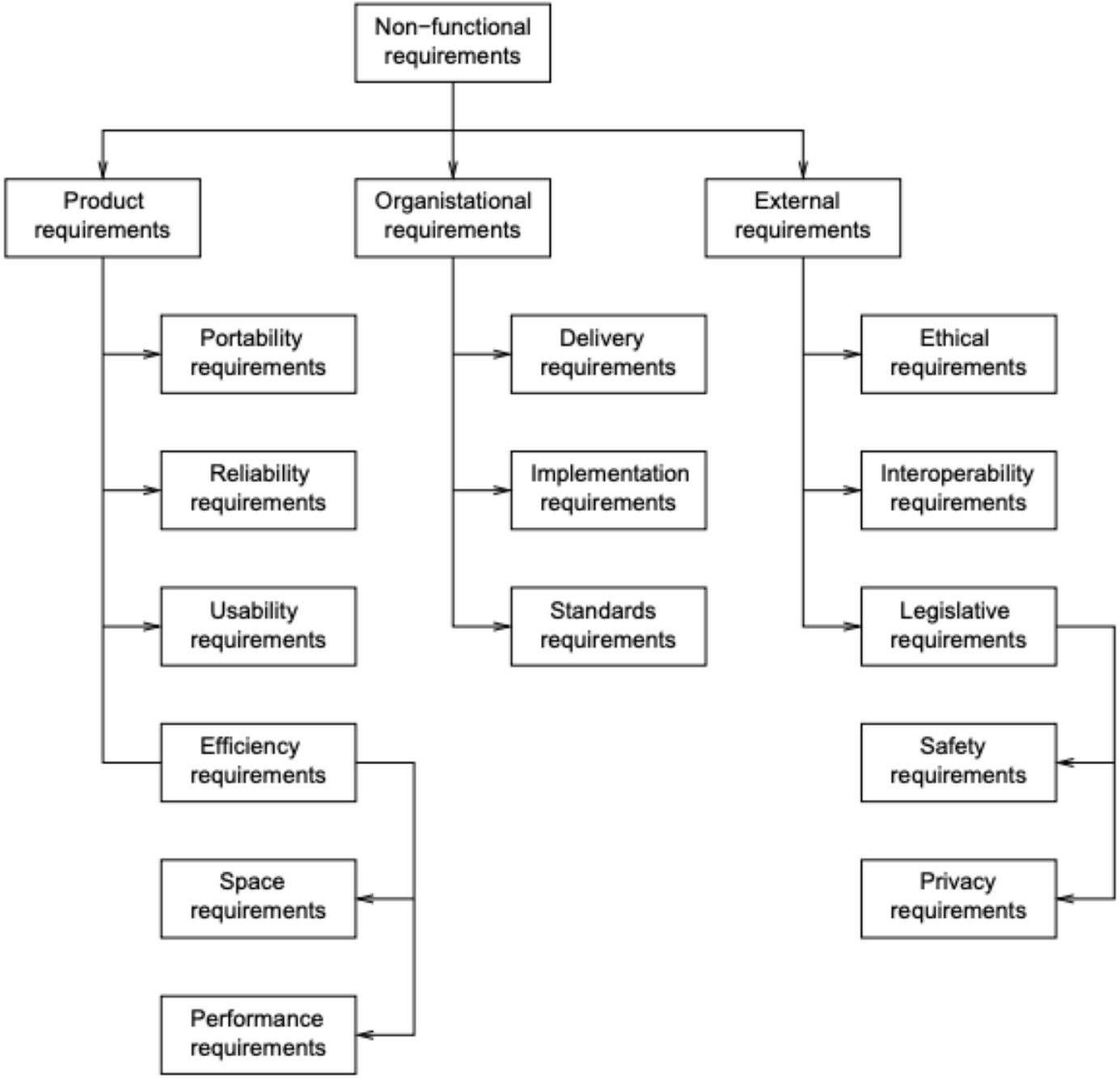
- Search option given to users to search from various invoices.
- User can email any report to management from a portal.

NON-FUNCTIONAL REQUIREMENTS

- Basically the quality constraints that the system must satisfy according to the project contract.
- Non-functional requirements deal with issues like:
  - Portability
  - Security
  - Maintainability
  - Reliability
  - Scalability
  - Performance
  - Reusability
  - Flexibility
- Identifying the non-functional requirements requires the knowledge of the functionality of the system, and the knowledge of the context within which the system will operate.
- Not related to the functionality of the product.
- Many projects make the mistake of not specifying these explicitly.
- These are also known as **quality requirements.**

CONTD....

- Non-functional requirements arise due to user requirements, budget constraints, organizational policies, and so on.
- Should be accomplished in software to make it perform efficiently.
- Example: if an aeroplane is unable to fulfil reliability requirements, it is not approved for safe operation.
- If a real time control system is ineffective in accomplishing non-functional requirements, the control functions cannot operate correctly.
- Example: the website page should load in 3 seconds with the total number of simultaneous users <5k.
- The system should be able to handle 20 million users without performance deterioration.


Non-functional requirements
Product requirements
Organistational requirements
External requirements
Portability requirements
Reliability requirements
Usability requirements
Efficiency requirements
Space requirements
Performance requirements
Delivery requirements
Implementation requirements
Standards requirements
Ethical requirements
Interoperability requirements
Legislative requirements
Safety requirements
Privacy requirements

PRODUCT REQUIREMENTS

- Specify how software product performs. Product requirements comprise the following:
- **Efficiency requirements**: describe the extend to which the software makes optimal use of resources, the speed with which the system executes, and the memory it consumes for its operation.
- **Reliability requirements**: describes the acceptable failure rate of the software. E.g. software should be able to operate even if a hazard occurs.
- **Portability requirements**: ability to port the software to a different OS or platform without the need to redesign the software.
- **Usability requirements**: relates to the ease with which the users are able to operate the software. Think about fewer key strokes and mouse clicks.

ORGANIZATIONAL REQUIREMENTS

- Derived from the policies and procedures of an organization. These comprise the following:
- **Delivery requirements**: specify when the software and its documentation are to be delivered to the user.
- **Implementation requirements:** describes the programming language and the design method to be used in implementation.
- **Standards requirements**: specification of the process standard used during software development. E.g. ISO and IEEE standards.

EXTERNAL REQUIREMENTS

- Requirements which affect the software or its development process externally. These comprise of:
- **Interoperability requirements:** how different computer based systems interact with each other. E.g. How data entered in mobile clinical app may be transferred to the hospital management system.
- **Ethical requirements**: specification of rules and regulations of the software so that they are acceptable to the users.
- **Legislative requirements**: to ensure that the software operates within the legal jurisdiction. E.g. Pirated software should not be sold.

METRICS FOR NON-FUNCTIONAL REQUIREMENT

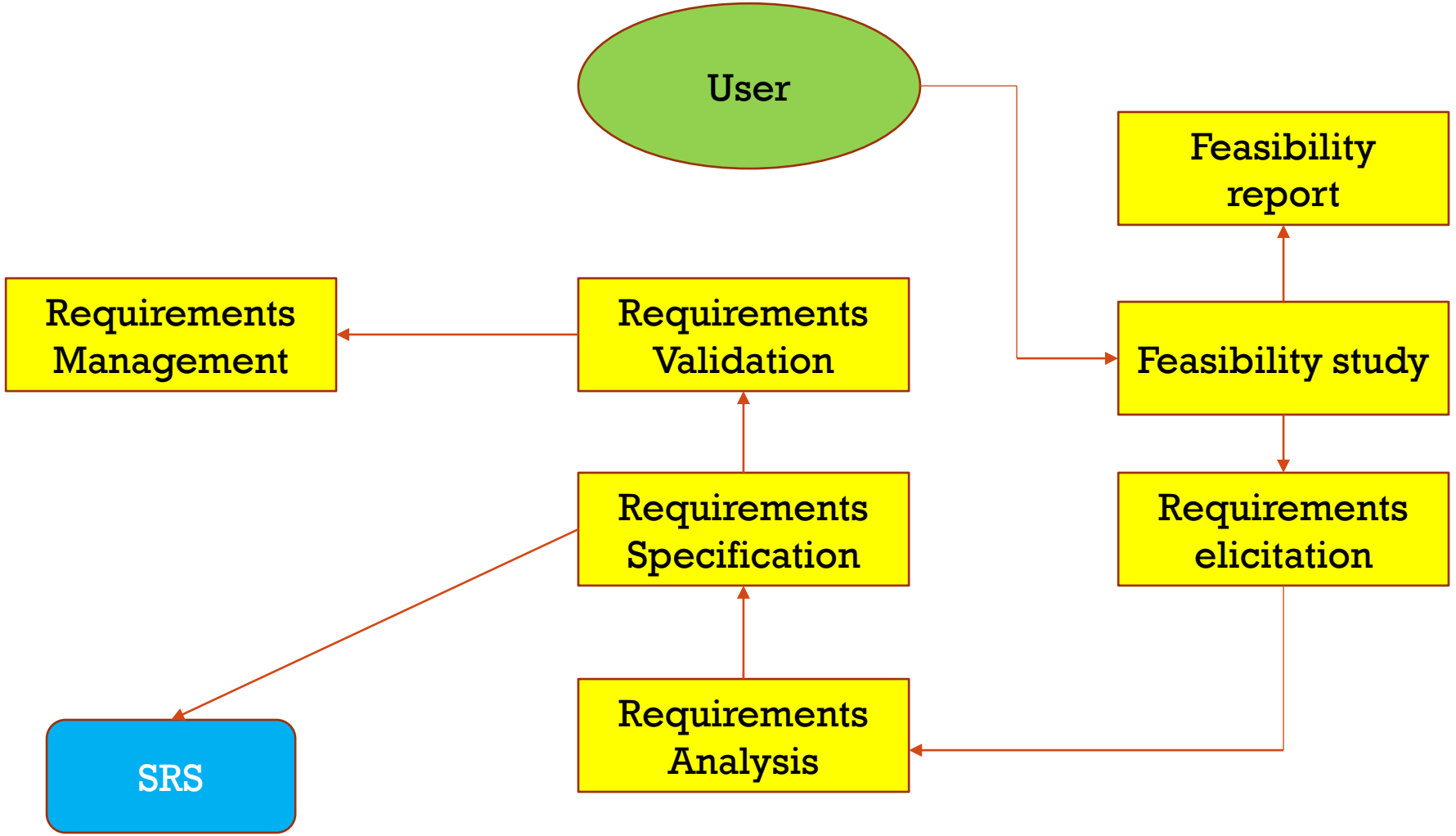
- Non-functional requirements are difficult to verify.
- Hence, it is essential to write them quantitatively, so that they may be tested.

Features	Measures
Speed	Processed transactions/second, Event response time, Screen refresh rate
Size	Amount of memory (KB), Number of RAM chips
Ease of use	Training time, Number of help windows
Reliability	Mean time to failure (MTTF), Rate of failure occurrence
Robustness	Time to restart after failure, Percentage of events causing failure, probability of data corruption on failure.
Portability	Percentage of target-dependent statements, number of target systems.

FUNCTIONAL VS. NON-FUNCTIONAL REQUIREMENTS

	Functional requirements	Non-functional requirements
Objective	Describe what the product does.	Describe how the product works.
End result	Define product features.	Define product properties.
Focus	Focus on user requirements.	Focus on user expectations.
Documentation	Captured in use cases.	Captured as quality attribute.
Essentiality	They are mandatory.	Not mandatory, but desirable.
Origin type	Usually defined by the user.	Usually defined by developers or other tech experts.
Testing	Tested before non-functional testing.	Tested after functional testing.
Types	External interface, authentication, authorization levels, business rules.	Usability, reliability, scalability, performance etc.

REQUIREMENTS ENGINEERING PROCESS



VALIDATION OF SOFTWARE REQUIREMENTS

- Software requirements need to be validated with the stakeholders for accuracy and whether or not they actually fulfil the needs.
- The validation of the requirements may be performed regularly if you are following an iterative development method.

END