CHAPTER 2:

Concept And Requirement Management

2.1. OVERVIEW STATEMENT

Nam Xuan Phong Corporation has 400 employees, they specialized in Supply chain, Construction, Installation, Repair and Maintenance of industrial. Their head office locates at HCM city and have built many constructions. Each construction has their own warehouse, so does the head office. All the work was done by excel, via phone and email. That makes a big problem in management the supply chain and warehouse material, this current process is wasting time and cost. Nam Thinh Corporation has considered many software applications, but those have bad performance and hard to use. Moreover, they manage more than ten thousand materials, each material belongs to specific category and suppliers. They spend to manipulate materials to create receipts.

2.2. BUSINESS SYSTEM

- Supply chain operation

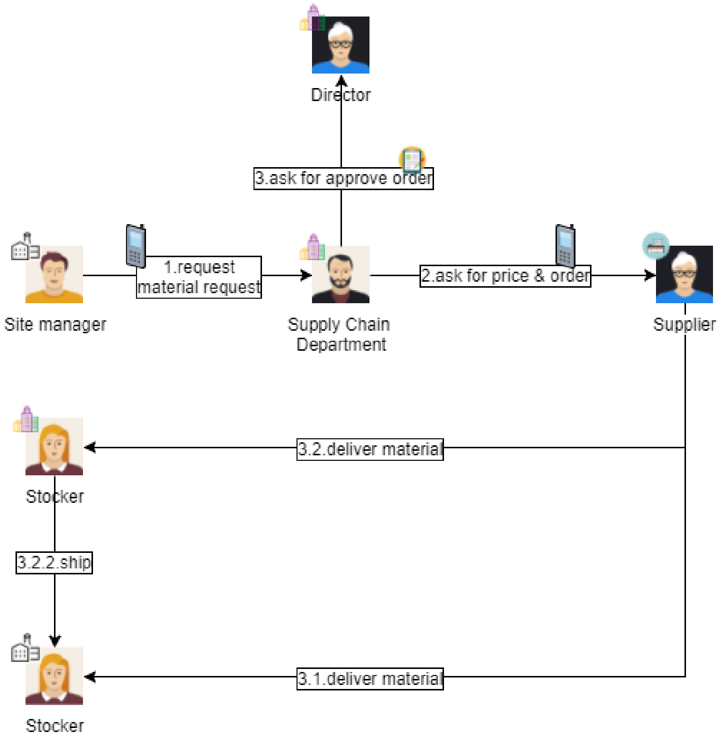


Figure 2.1 Supply Chain Operation

Table 2.1 Supply Chain Operation Description

#	Position	Activities
1	Site manager	SM calls SCD employee to tell what he wants for Material Request.
2	Supply chain department	SCD will ask Head stocker to check the quantity of material If available, they will prepare the material and ship to the construction. Else, SCD calls Suppliers to ask for ordering. Then the SCD create a Purchase Receipt by word to ask Director for approving.
3.1	Supplier	If the Suppliers are requested to ship the material to the construction, they will delivery material to construction stocker. Stocker will check and import it to construction store.
3.2	Supplier	If request is company warehouse, supplier will delivery to Head stock. Head stocker will check and import it to store.
3.2.2	Stocker	After receiving material, they will execute shipment later.

- Material Import Operation

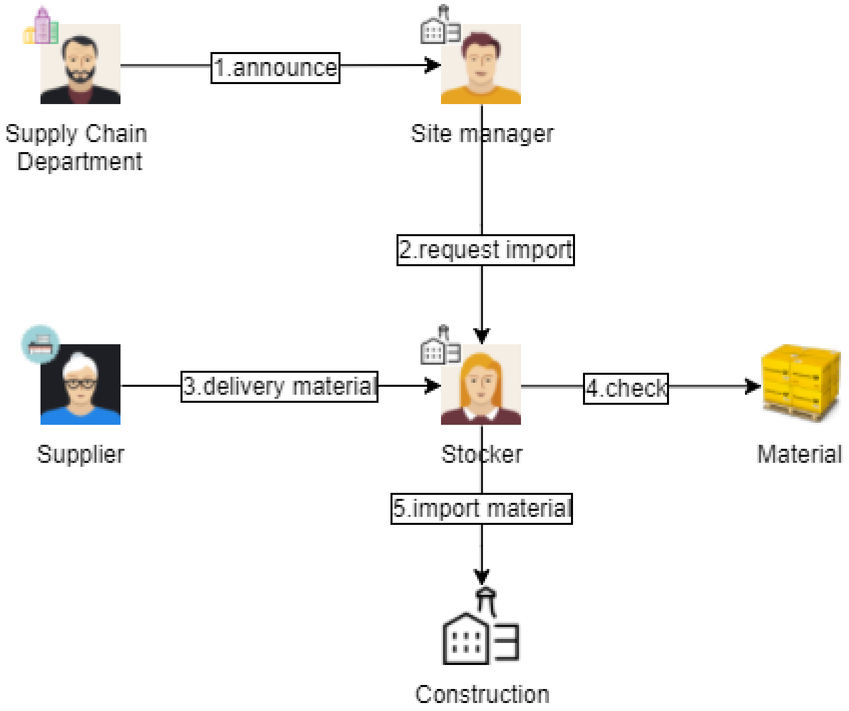


Figure 1.2 Material Import Operation

Table 1.2 Material Import Operation Description

#	Position	Activities
1	Supply chain department	SCD announce Site manager that Purchasing is successful and prepare to import material
2	Site manager	Site manager will announce Stocker to prepare for the importation.
3	Supplier	Suppliers delivery material to construction
4	Stocker	Stocker will check the material's quantity, quality. If the material does not meet the requirement, stocker do not accept the order. Else, stocker confirm the order is done.
5	Stocker	Stocker will import the material into construction physically

2.3. SYSTEM JUSTIFICATION

- Justification for Change

  This current process is wasting time and cost. It is also easy to cause deviations and loss of information. Site manager doesn't know where is his or her application and its status. Secretary hard to manage application. Director consuming cost for shipper. Market systems are also hard to use.

  With the website, Site Manager can easily determine application materials locate, status; Secretary can easily transfer reports and application materials and Director Manage can easily materials and approval of the application.

  Description of require changes

  - Ability: Data transparency and accuracy
  - Data processing: All applications will be real-time transferred. The change of materials will be updated promptly.
  - Display: Easy to use and manipulation not to complex
  - Environment: Support to access system online 24/7
  - Work: Staffs must do the right functions when they are authorized.
- Anticipated Users and Stakeholders

Table 2.2 Anticipated Users and Stakeholders

Stakeholder	Description
Director	Manage the company information and operation
Administration	Manage the system of company
Supply chain Department	Under inspection of Vice director Manage supply chain
Site Manager	Manage specific project, work in construction
Head Stocker	Manage the imported, exported operation of head warehouse
Stocker	Manage the imported, exported operation of construction warehouse
Supplier	Participate in supply chain operation

2.4. PROPOSED SYSTEM

2.4.1. Site Manager Request to supply material

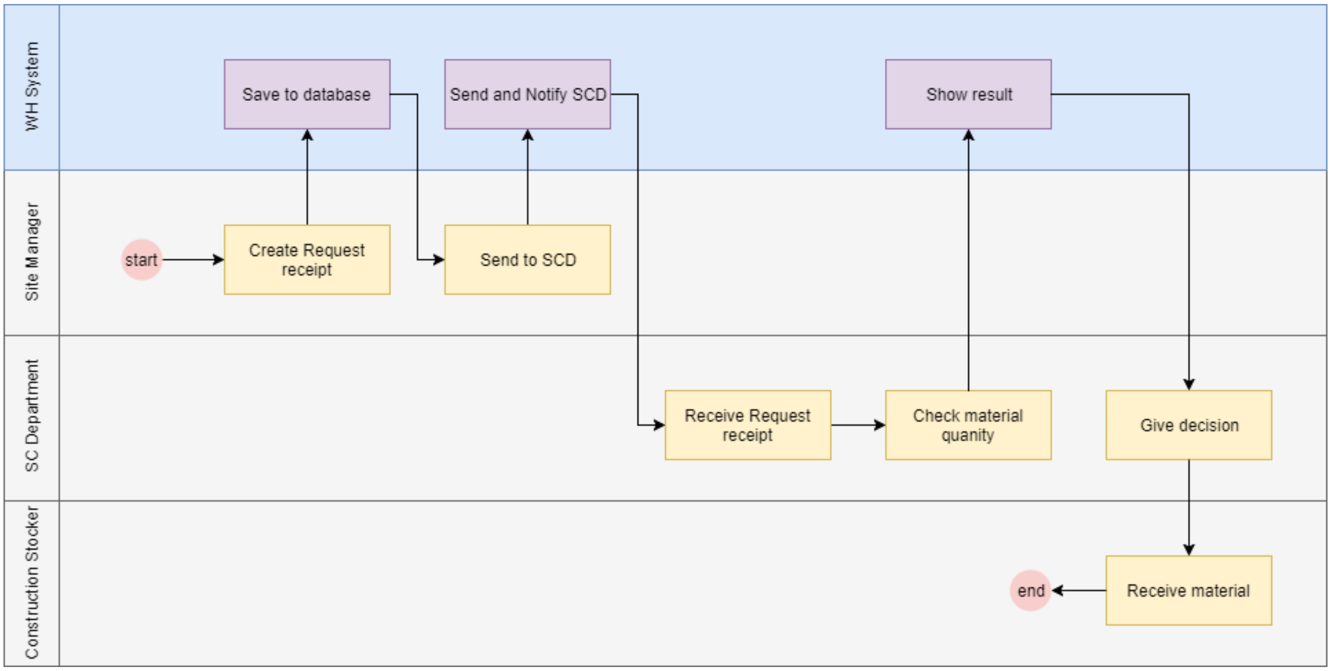


Figure 2.2 Supply Material Request

Table 2.3 Supply Material Request Description

#	Position	Activities
1	Site manager	SM access system to create Request receipt and send it to SCD. The system will save receipt and notify SCD to check the receipt from SM. The Request receipt is [Sent]
2	Supply chain department	SCD receive notification and receipt. SCD will check material quantity of Head store or other construction store. If the quantity is available, they will prepare the material and execute shipment. Else, SCD will execute purchasing The system will notify the SM when SCD make decision. [Wait for Shipment] or [Wait for Purchasing]
3	Stocker	End of this process, the Stocker will receive the needed material.

2.4.2. Delivery Operation

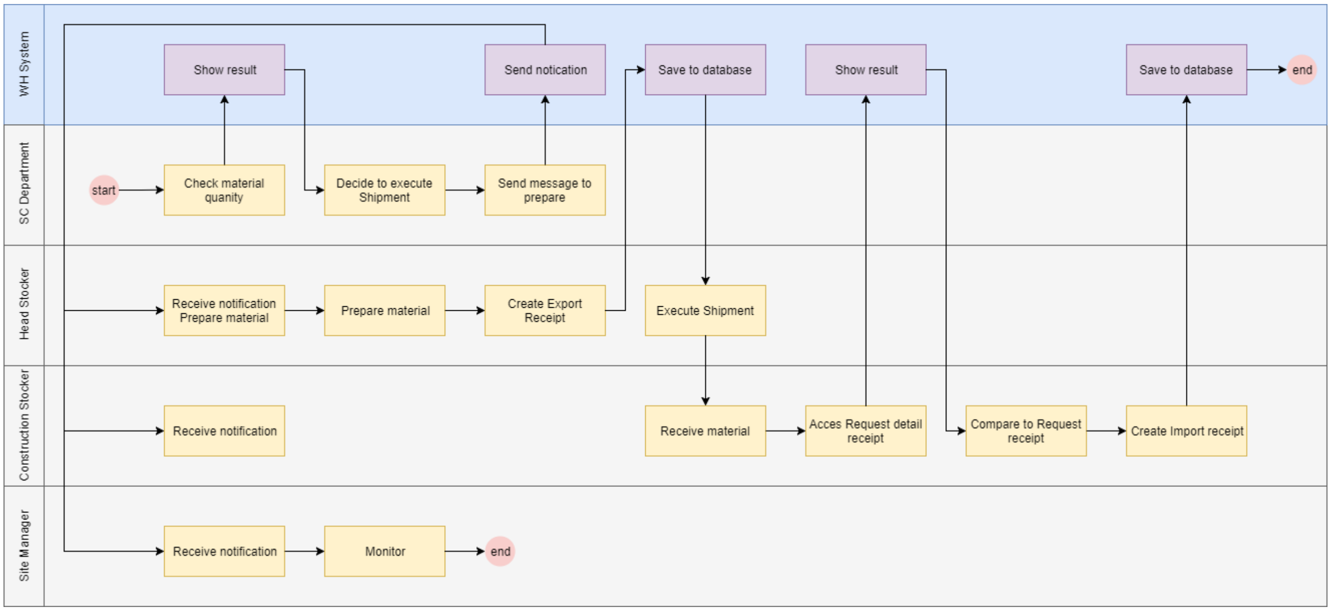


Figure 2.3 Delivery Operation

Table 2.4 Delivery Operation Description

#	Position	Activities
1	Supply chain department	SCD check material quantity of Head store or other construction store and the result is the quantity is available. SCD decide to execute shipment. The system will notify the SM and stocker the Request Receipt status is [Wait for Shipment] The Head Stocker will receive notification from system to prepare material for Request receipt
2	Head Stocker	After preparing material the HS will create Export receipt on the system to export material from Head Store.
3	Stocker	The stocker receive needed material and then access to the Request detail receipt to compare. If true, stocker will create Import receipt on the system. And announce SM the Request Operation is done. Else, stocker will announce SM to solve.
4	Site manager	After sending the Request receipt, the SM can check the status of it. If the Request Operation is done, SM will change the Request receipt status to [Done]

2.4.3. Purchase Operation

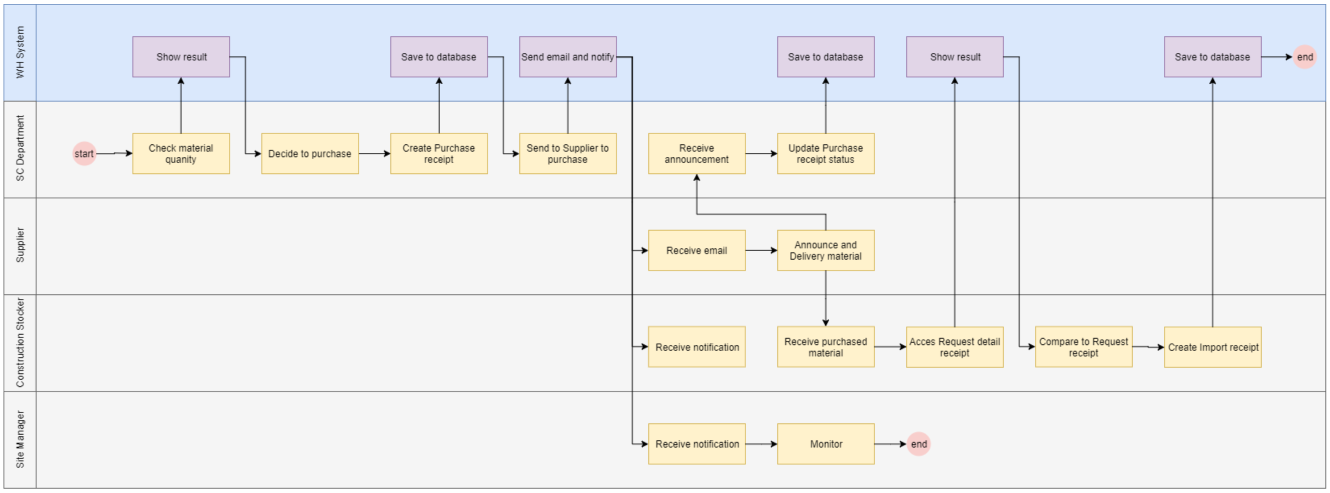


Figure 2.4 Purchase Operation

Table 2.5 Purchase Operation

#	Position	Activities
1	Supply chain department	SCD check material quantity of Head store or other construction store and the result is the quantity is not available. SCD decide to execute purchasing.
2	Supply chain department	SCD create Purchase receipt and send it to the Suppliers through the system. System will use email API to send it to each supplier. The System will notify the SM and stocker the Request Receipt status is [Wait for Purchasing]
3	Supplier	Supplier will announce SCD and delivery material
4	Stocker	The stocker will receive needed material and then access to the Request detail receipt to compare. If true, stocker will create Import receipt on the system. And announce SM the Request Operation is done. Else, stocker will announce SM to solve.
5	Site manager	After sending the Request receipt, the SM can check the status of it. If the Request Operation is done, SM will change the Request receipt status to [Done]

2.5. REQUIREMENTS ENGINEERING

Requirements engineering is a process of gathering and defining of what the services should be provided by the system.

It focuses on assessing if the system is useful to the business (feasibility study), discovering requirements (elicitation and analysis), converting these requirements into some standard format (specification), and checking that the requirements define the system that the customer wants (validation).

In practice, requirements engineering isn’t sequential process, it’s an interative process in which activities are interleaved.

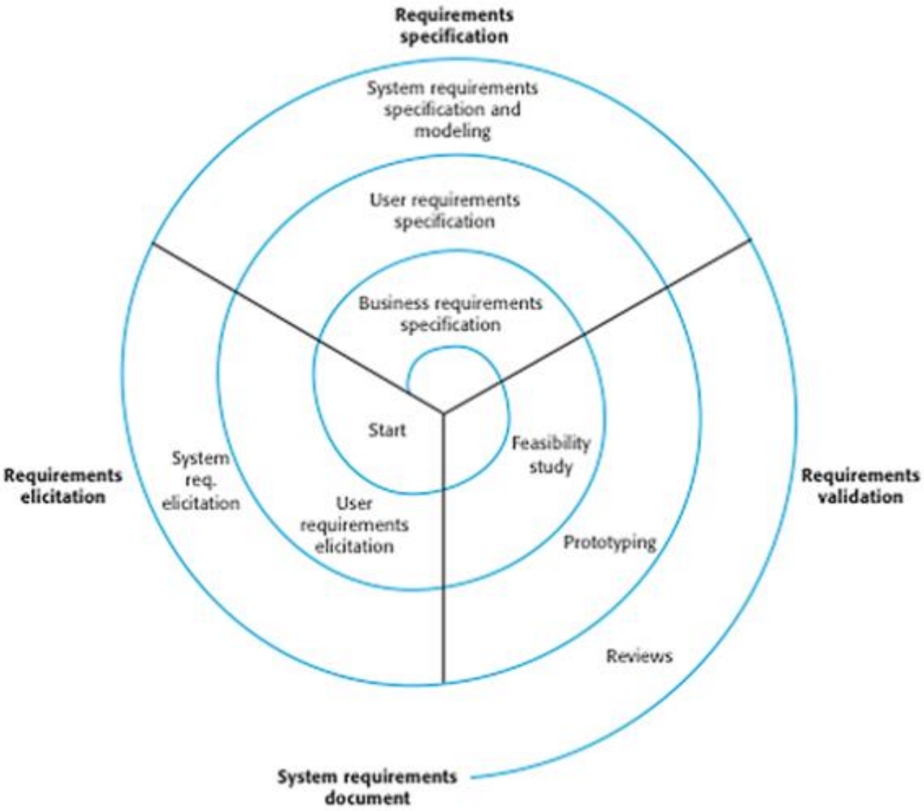

Requirements specification
System requirements specification and modeling
User requirements specification
Business requirements specification
Start
Feasibility study
Requirements elicitation
System req. elicitation
User requirements elicitation
Prototyping
Requirements validation
Reviews
System requirements document

Figure 2.6 The process of requirements engineering

Early in the process, most effort will be spent on understanding high-level business and user requirements. Later in the process, more efforts will be spent on elicitation and understanding detailed system reqiurements.

2.5.1. User and System Requirements

Typically, requirements are presented into two level of detail; user and system requirements, where user need a high-level statements of the requirements, while system developers need a more detailed system specification. So, user and system requirements are just refer to different level of detail.

Having different level of details is useful because it communicates information about the system being developed for different types of readers.

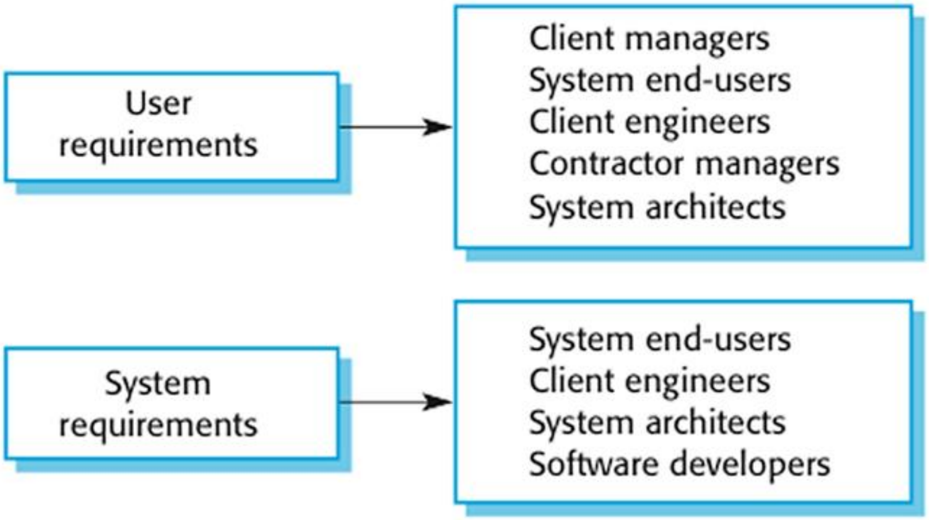


Figure 2.7 Readers of different types of requirements specification

So, end-users will not be concerned with the detail, they need a generic, abstracted written requirement.

While the people who are involved in the development, they need what exactly the system should do.

2.5.2. Functional & Non-Functional Requirements

The software requirements are classified into functional requirements and non-functional requirements.

- Functional Requirements
  - It covers the main functions that should be provided by the system. When expressed as user requirement, they are usually described in an abstract way.
  - However, more specific functional system requirement describe the system functions, it’s inputs, processing; how it’s going to react to a particular input, and what’s the expected output.
- Non-Functional Requirements
  - These are the constrains on the functions provided by the system.
  - The constrains, like how many process the system can handle (performance), what are the (security) issues the system needs to take care of such as SQL injections, the rate of failure (reliability), what are the languages and tools will be used (development)…

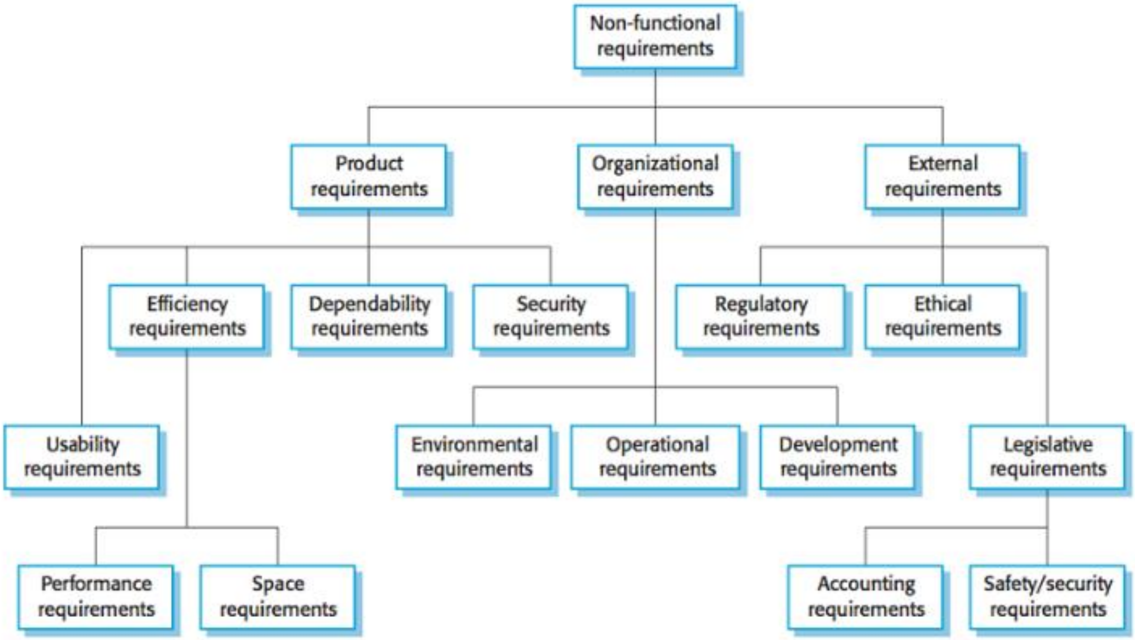
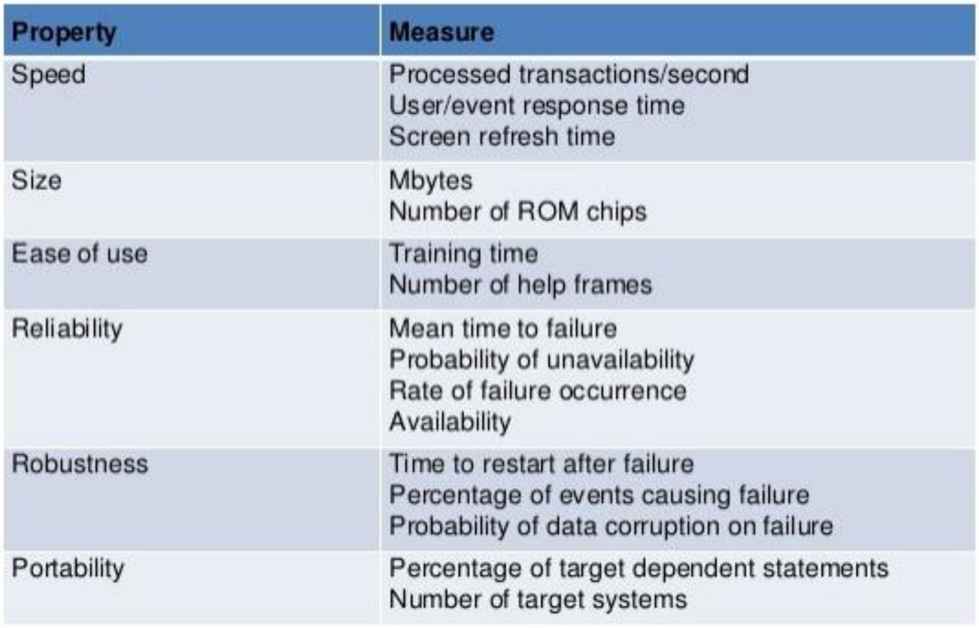


Figure 2.8 Non-Functional Requirements

- Non-functional requirements are often critical than individual functional requirements. Users can usually find ways to work around a system function that doesn't really meet their needs. However, failing to meet a non-functional requirement can mean that the whole system is unusable.

- Non-functional requirements should be measurable
  - Whenever possible, we should write non-functional requirements quantitatively, so that they can be tested. You can measure them when the system being tested to check whether the system meet it's non-functional requirements.



Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
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

Figure 2.9 Metrics for non-functional requirements

- In practice, customers for a system often find it difficult to translate their goals into measurable requirements. They don’t understand what some number defining the required speed or reliability. For some goals, such as maintainability, there’re no metrics that can be used.
- The cost of verifying measurable non-functional requirements can be very high and the customers may not think that these costs are justified.

2.6. FEASIBILITY REPORT

The input to the feasibility study is a set of preliminary business requirements, an outline description of the system and how the system is intended to support business processes.

The source for information is the managers of departments where the system will be used, software engineers who are familiar with the type of proposed system, end-users of the system, etc. We will complete a feasibility study in three weeks.

The results of the feasibility study should be a report that recommends whether to go forward to the next process or you won’t be able to implement the software at all.

2.7. REQUIREMENTS ELICITATION & ANALYSIS

It’s a process of interacting with customers and end-users to find out about the domain requirements, what services the system should provide, and the other constrains.

It may also involve a different kinds of stockholders; end-users, managers, system engineers, test engineers, developer, etc.

Here are the 4 main process of requirements elicitation & analysis:

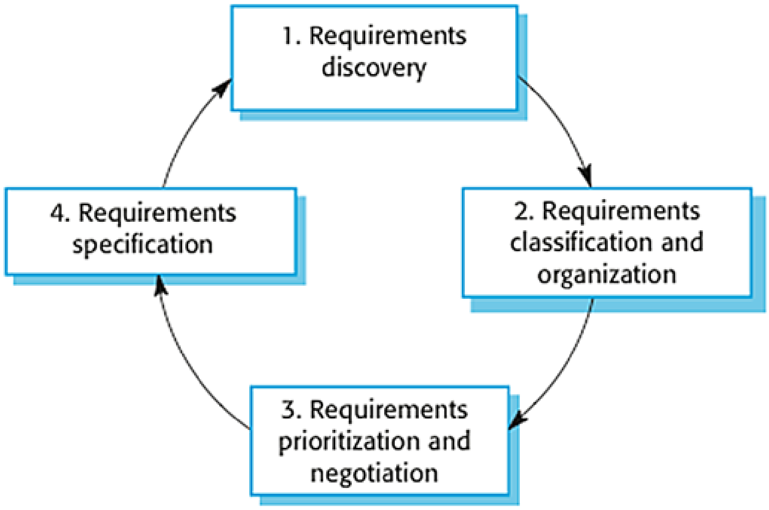

1. Requirements discovery
2. Requirements classification and organization
3. Requirements prioritization and negotiation
4. Requirements specification

Figure 2.10 The process of requirements elicitation & analysis

It shows that it’s an iterative process with a feedback from each activity to another. The process cycle starts with requirements discovery and ends with the requirements document. The cycle ends when the requirements document is complete.

- Requirements Discovery

  It’s the process of interacting with, and gathering the requirements from, the stakeholders about the required system and the existing system (if exist).

  Gathering and understanding the requirements is a difficult process

  That’s because stakeholders may not know what exactly they want the software to do, or they may give unrealistic requirements. They may give different requirements, which will result in conflict between the requirements, so we have to discover and resolve these conflicts.

  It can be done using some techniques, like interviews, scenarios, prototypes, etc, which help the stakeholders to understand what the system will be like.

  - Interviews

    In Interviews, our requirements engineering team puts the questions to the stakeholder about the system that’s currently used, and the system to be developed, and hence we can gather the requirements from the answers.

  The questions fall under two categories:

  - Closed-Ended questions: A pre-defined set of question.
  - Open-Ended questions: There is no a pre-defined expected answer, they are more of generic questions. It’s used to explore issues that’s not clear in a less structured way.

  Interviews are good to get an overall understanding of what stakeholders need, how they might interact with the new system, and the difficulties they face with the current system.

However, interviews aren't so helpful in understanding the domain requirements. This is for two reasons:

- Domain requirements may be expressed using special domain terminologies, and software engineers often find it difficult to understand and it's easy for them to misunderstand.
- Sometimes stakeholders won't tell you some requirements because they assume it's so fundamental and it doesn't worth mentioning, or they find it difficult to explain, which won't be taken into consideration in the requirements.

- Task Analysis

Employees working in some domain perform a number of tasks, such as handling material requests, cataloging new materials, ordering materials, etc.

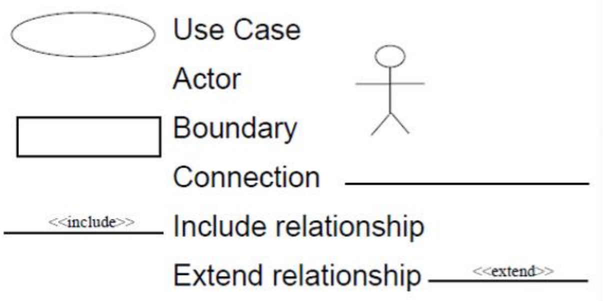
Higher-level tasks may be decomposed into subtasks. For example, the task 'handle a material request' may lead to the following subtasks:

- Check if that material is still in stock
- Check the quantity of that material
- The request form approved by whom?

- Use Cases & Scenarios

Use cases identify interactions between the system and it's users or even other external systems (using graphical notations), while a scenario is a textual description of one or more of these interactions.

Use case involves some symbols to describe the system:



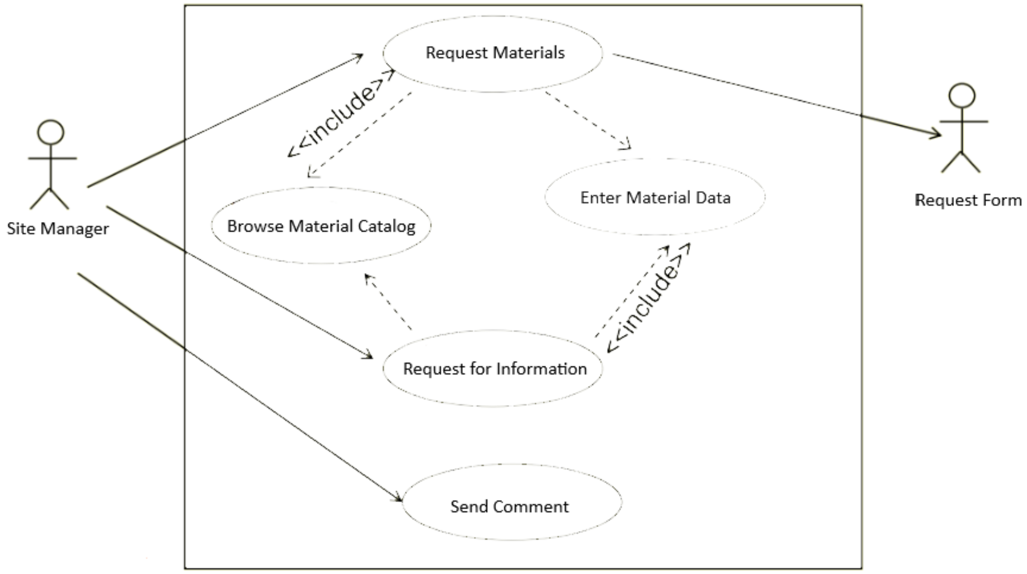


Figure 2.11 Request Material Use Cases

- Actors: Are those who interact with the system; human or other systems
- Interaction (Use Case): It denotes the name of the interaction (verb). It’s represented as a named ellipse.
- Connection: Lines that links between the actors and the interactions.
- Include Relationship: It denotes a connection between two interactions when an interaction is invoked by another. As an example, splitting a large interaction into several interactions.
- Exclude Relationship: It denotes a connection between two interactions when you want to extend an interaction by adding an optional behavior, but you can use the main interaction on it’s own without the extending interaction.

Now, we are going to use scenarios to describe the interactions in each use case textually. They should have a format and include the following:

- A description of the initial situation.
- A description of the flow of the events or interactions with the system
- A description of the exceptions, or in other words, what can go wrong, and how it can be handled.
- Any concurrent activities should be mentioned
- A description of the final state.

Table 2.7 Request materials Scenario

Name	Request Materials
Actors	Site Manager and Warehouse Management System
Description	It shows how Site Manager can request materials and view request form information
Pre-condition	The Site Manager is logged in
Post-condition	The Site Manager received materials from the Stocker
Actions	1. Site Manager will press on “Request” from Home Page 2. Enter the request form’s info 3. Enter the materials he needs and that material quantity 4. Press on “Save” 5. Confirmation message upon success 6. Site Manager can view his request form
Exceptions	User entered invalid input, thus, an error message will be displayed

- Use case and scenarios are effective techniques for eliciting the requirements. But, because they focus on the interactions with the system, they aren’t effective for eliciting high-level business, non-functional, or domain requirements.
- Requirements Classification & Organization

  It’s very important to organize the overall structure of the system.

  Putting related requirements together, and decomposing the system into sub components of related requirements. Then, we define the relationship between these components.

  What we do here will help us in the decision of identifying the most suitable architectural design patterns.

- Requirements Prioritization

  In most cases, not all requirements can be realized, so we have to make a selection. In the Kano model, user preferences are classified into five categories:

  - Attractive: The customer will be more satisfied if these requirements are met, but not less satisfied if they are not. For example, the Site Manager will get an automatic notification to his smartphone if the Stocker confirms his Request form.

  - Must-be: The customer will be dissatisfied if these requirements are not met, but his satisfaction will never raise above neutral. An example is the ability to search the material catalog.
  - One-dimensional: For requirements of this category, satisfaction is proportional to how many of these requirements are being met. Alternative ways to search the materials catalog could fall into this category.
  - Indifferent: The customer does not really care about these requirements. For example, the customer might not care whether different categories of materials are displayed in a different color on the screen.
  - Reverse: The customer's judgement of the requirements is the opposite of what the analysy thought a priori.
  - Questionable: The customer's preferences are not clear. He both seems to like and dislike a certain feature.
- Requirements Specification

  The requirements are then documented.

2.8. REQUIREMENTS SPECIFICATION

Besides readability and understandability, the requirements specification should be correct, unambiguous, complete, consistent and verifiable, modifiable and traceable.

We choose the natural language specification. It's a way of writing the requirements in normal plain text, there is no defined format by default.

"A/The (Actor) shall (do something), By (how; explain how the user can trigger this feature), In order to/so that (why; explain the benefits or the objects of this requirement).

Example: "A system shall allow the users to register by entering their username and password, In order to get an access to the system".

2.8.1. Software Requirements Document

The software requirements document (also called software requirements specification or SRS) is an official document of what should be implemented. It's also used as a contract between the system buyer and the software developers.

The requirements document has to be a compromise between communicating the requirements to customers, defining the requirements in detail for developers and testers, and information about predicted changes can help system designers to avoid restrictive design decisions, and help the system maintenance engineers to adapt the system to new requirements.

In agile methods, since the requirements change so rapidly, it's a waste of time to deliver a full document at once, instead, collects the requirements incrementally, and write them on a cards as user stories.

Each user story has estimated time of completion, and priority. Related user stories are grouped together.

2.8.2. Requirements Management

- Requirements identification

  Each requirement has to have a unique identification, the simplest form is to just number them. Since requirements are often changed and updated, it is expedient to include versioning information as well. Finally we will add some attributes to each requirement such as its status, priority, main stakeholder involved.

- Requirements change management
  - Should apply to all proposed changes to the requirements.
  - Principal stages:
    - Problem analysis. Discuss requirements problem and propose solution.
    - Change analysis and costing. Assess effects of change on other requirements.
    - Change implementation. Modify requirements document and other documents to reflect change.

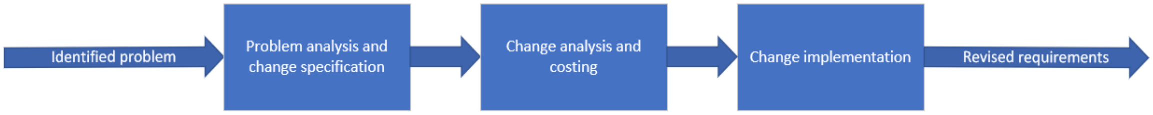

Identified problem
Problem analysis and change specification
Change analysis and costing
Change implementation
Revised requirements

Figure 2.12 Requirements Change Process

- Requirements traceability

  We may connect requirements to solution elements such as design elements or

  even software components that realize those requirements. In this way, we establish traceability from requirements to code and vice versa. This allows us to trace where requirements are realized (forward traceability), and why certain solutions are chosen (backward traceability). Traceability information is important in all development phases. It can be used to answer a variety of questions, such as:

  - Where is this requirement implemented?

- Do we need this requirement?
- Are all requirements linked to solution elements?
- Which requirement does this test case cover?
- What is the impact of this requirement?

2.9. REQUIREMENTS VALIDATION

It's a process of ensuring the specified requirements meet the customer needs. It's concerned with finding problems with the requirements.

These problems can lead to extensive rework costs when these they are discovered in the later stages, or after the system is in service.

The cost of fixing a requirements problem by making a system change is usually much greater than repairing design or code errors. Because a change to the requirements usually means the design and implementation must also be changed, and re-tested.

During the requirements validation process, different types of checks should be carried out on the requirements. These checks include:

- **Validity** *checks*: The functions proposed by stakeholders should be aligned with what the system needs to perform. You may find later that there are additional or different functions are required instead.
- **Consistency** *checks*: Requirements in the document shouldn't conflict or different description of the same function
- **Completeness** *checks*: The document should include all the requirements and constrains.
- **Realism** *checks*: Ensure the requirements can actually be implemented using the knowledge of existing technology, the budget, schedule, etc.
- **Verifiability**: Requirements should be written so that they can be tested. This means you should be able to write a set of tests that demonstrate that the system meets the specified requirements.
-

Requirements Validation Techniques

- Requirements Reviews

Systems developers of my team will read the requirements in the document, and investigate in a great detail to check for errors, inconsistency, conflicts, and any ambiguity. Then they may negotiate with the customer on how to solve the problems and errors found.

- Prototyping

Prototyping is usually used when the requirements aren’t clear. So, we make a quick design of the system to validate the requirements. If it fails, we then refine it, and check again, until it meets the customer needs.

2.10. REQUIREMENT CHECKLIST

2.10.1. General Requirement Document

- Is a functional overview of the system provided?
- Have the software and hardware environments been specified?
- Are implementation assumptions stated?
- Have the functionality of hardware or software interacting with the system been properly specified?
- Has every acronym been defined?
- Is each requirement traceable to its source?
- Employ a numbering scheme to facilitate traceability and control.
- Are relationships among requirements clearly defined and organized to show how they may relate to form subsystems and a complete system?
- Are boundaries, scope, and context of the requirements identified?
- Is the document easily searchable for modification, and addition of requirements?
- Do the requirements avoid specifying design?
- Are the requirements at a fairly consistent level of detail?
- Are the requirements clear enough to be turned over to an independent group for implementation and still be understood?
- Does the set of requirements adequately address all appropriate exception conditions?
- Can the requirements be implemented within known constraints?
- Are all cross-references to other requirements correct?
- Are all missing items or unresolved issues identified with a TBD, an owner, and a time-line for closing it?

2.10.2. Business and Functional Requirements

- Are the high-level business objectives described?
- Are the requirements understandable by all stakeholders?
- Is the value to the business identified? (Cost savings, reduced inventory, etc.)
- Is the value to the customer identified? (New features, improved usability, etc.)
- Does this requirement answer the question ‘Why is this needed’?
- Does the set of functional requirements meet the needs outlined by business requirements? (e.g. complete, sufficient, etc.)

- Is the relation between functional and the non-functional requirements clear?

2.10.3.Interface Requirements

- Are all inputs to the system specified, including their source, accuracy, range of values, parameters and frequency?
- Are all outputs from the system specified, including their destination, accuracy, range of values, parameters and format?
- Are all screen formats specified?
- Are all report formats specified?
- Are all interface requirements between hardware, software, personnel, and procedures included?
- Are all communication interfaces specified, including handshaking, error-checking, and communication protocols?

2.10.4.Technical Requirements

- Are all inputs to a function sufficient to perform the required function?
- Are undesired events/inputs considered and their required responses specified?
- Have the types, initial values, units been defined for every object attribute?
- Have the parameter and return types of all object operations been defined?
- Has the accuracy, precision, range, type, rate, units, frequency of inputs and outputs been specified for each function?
- Is the expected response time, from the user's point of view, specified for all operations?
- Is the level of security specified?
- Is the reliability specified, including the consequences of software failure, the vital information that needs to be protected from failure, and the strategy for error detection and recovery?
- Is the maximum memory specified?
- Is the maximum storage specified?

2.10.5.Individual Requirements

- Is the requirement clear and concise?
- Is the requirement stated in as simple a form as possible?
- Is the requirement testable/verifiable?

- Is the requirement correct?
- Is the requirement in scope? (i.e., the system will be considered incomplete if even one requirement is left out)
- Is the requirement as modifiable as possible?
- Is the requirement written in the customer’s language, using the customer’s terminology?
- Is the requirement necessary?
- Does this requirement answer the question ‘How Well’?
- Is each requirement implementation independent?

2.10.6.System Requirements

- Reliability
    - Are the reliability (MTBF) requirements specified?
    - Is the availability (up time) requirements specified?
    - Are the serviceability (MTTR) requirements specified?
    - Are the robustness requirements specified?
- Performance
    - Are the response time or latency requirements specified?
    - Are the throughput requirements specified?
    - Are the data volume requirements specified? (input, stored, output)
    - Are the peak or short-term load requirements specified?
- Safety/Security
    - ▪ Are the security requirements specified?
    - ▪ Are the safety requirements specified?
- Configuration
    - Are the supported configurations specified?
    - Are the compatibility requirements specified? (backwards, other applications, etc.)
- Usability
    - Are the usability requirements specified?
    - Are the internationalization/localization requirements specified?
    - Are the look and feel requirements specified? (e.g. color schemes, standards, etc.)
- Operational
    - Are all operational constraints or requirements specified? (e.g. network limitations, memory limitations, processor speed.)