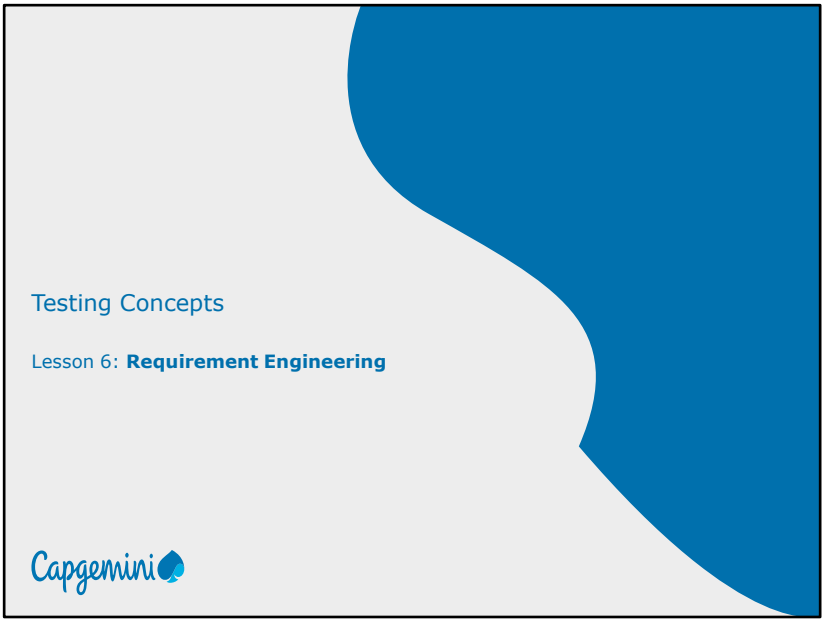

Testing Concepts
Lesson 6: Requirement Engineering
Capgemini

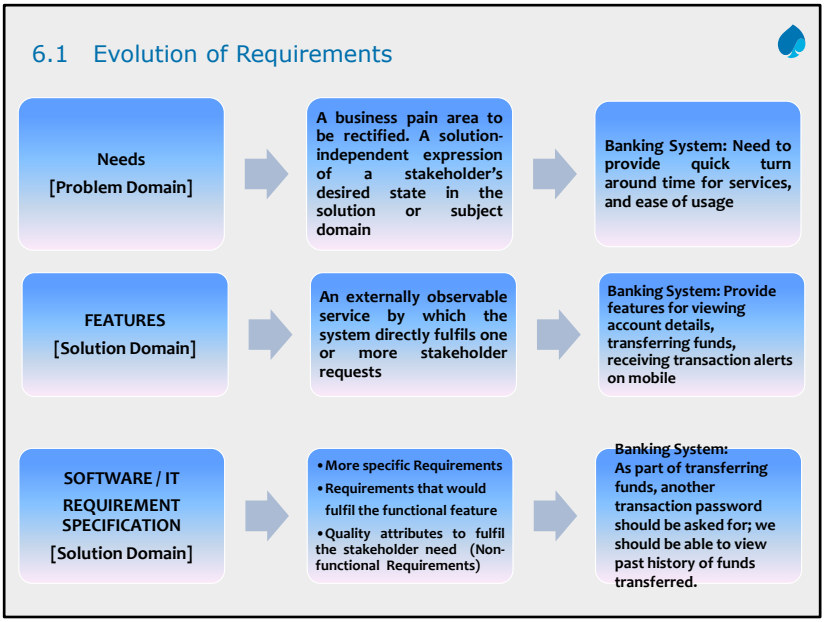
Lesson Objectives

To understand the following topics:

- Evolution of Requirements
- Who provides the Requirements?
- Challenges in Requirement Gathering
- Why do we need good requirements?
  - Characteristics & Impact of bad Requirements
- Requirement engineering
- Functional Vs Non-Functional Requirements
- Non Functional Requirements: FURPS +

Lesson Objectives

- Stable and Volatile Requirements
- Baselining Requirements
- Requirements Traceability
  - Requirement Traceability Matrix
  - Maintaining Requirement Traceability
  - Requirement Traceability Matrix – Example
- Requirements Change
  - Change Management Process
  - Requirement Creep


6.1 Evolution of Requirements
Needs
[Problem Domain]
A business pain area to be rectified. A solution-independent expression of a stakeholder's desired state in the solution or subject domain
Banking System: Need to provide quick turn around time for services, and ease of usage
FEATURES
[Solution Domain]
An externally observable service by which the system directly fulfils one or more stakeholder requests
Banking System: Provide features for viewing account details, transferring funds, receiving transaction alerts on mobile
SOFTWARE / IT REQUIREMENT SPECIFICATION
[Solution Domain]
• More specific Requirements
• Requirements that would fulfil the functional feature
• Quality attributes to fulfil the stakeholder need (Non-functional Requirements)
Banking System:
As part of transferring funds, another transaction password should be asked for; we should be able to view past history of funds transferred.

6.2 Who provides the Requirements? - Stakeholders

"Stakeholders" are the individuals who affect or are affected by the proposed software product.

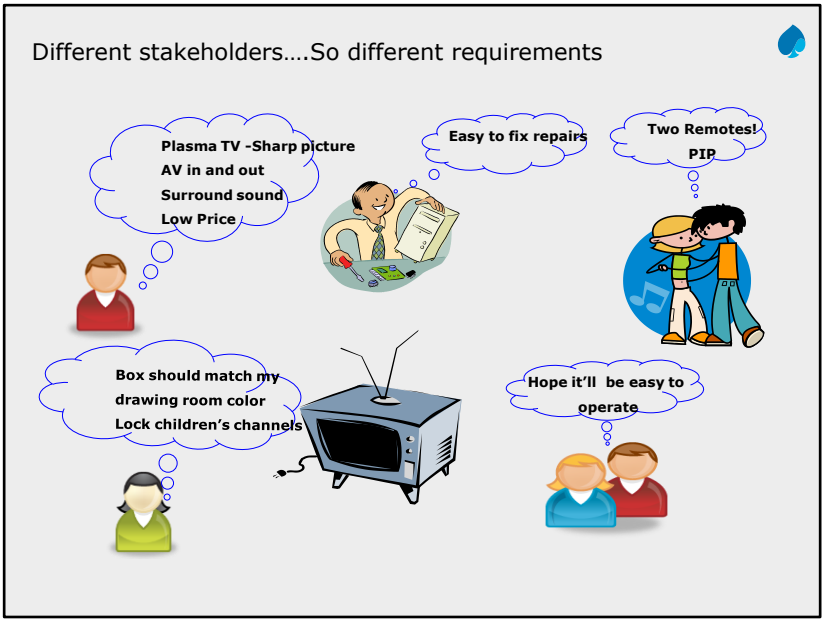
Following are the different stakeholders :

- Customers – These people purchase, and/or pay for the software product in order to meet their business objectives
- End Users – use the product directly or indirectly by receiving reports, outputs, or other information generated by the product
- Development Team – They include individuals/teams from the development organization :
  - Business Analyst - elicit the requirements from the customers, users, and other stakeholders; analyze requirements, write requirements specification, and communicate the requirements to development team and other stakeholders.
  - Designers –translate the requirements into the software's architectural and detailed designs specifying how the software will be implemented.
  - Developers – implement the designs by creating the software product
  - Testers – They use the requirements for designing the test cases that they use to execute and test the software under specific, known conditions to detect the defects and provide confidence that the software performs as specified.

Other Stakeholders: There may be other stakeholders interested in the requirements too.

Following is the list of other stakeholders:

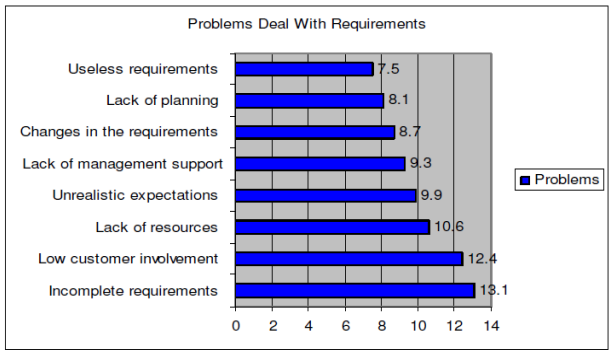
Legal or contract management
Manufacturing or product release management
Sales and marketing
Upper management
Government or regulatory agencies


Different stakeholders….So different requirements
Plasma TV -Sharp picture
AV in and out
Surround sound
Low Price
Easy to fix repairs
Two Remotes!
PIP
Box should match my drawing room color
Lock children's channels
Hope it'll be easy to operate

6.3 Challenges in Requirement Gathering

Requirement Problems are the single No.1 reason for projects to fail.

According to the Standish Group's 1995 CHAOS survey, the top two "project impaired" factors were **incomplete requirements** and **lack of user involvement.**

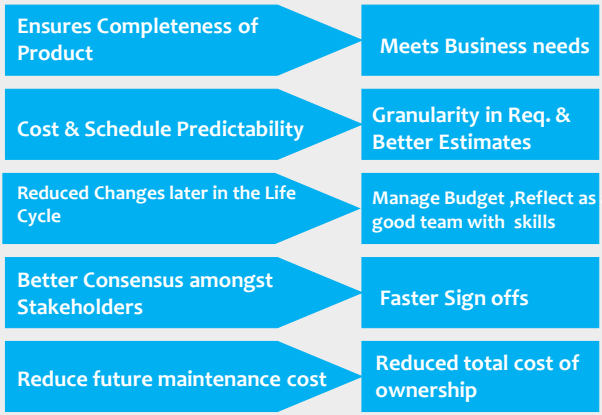


Challenges in Req. gathering :

1. Lack of User Involvement
   - "It is month end, quarter end ,year end. I have to do all my reports so can't spend time on requirements"
2. Unrealistic Customer Expectations :
   - Unreasonable, Infeasible, Conflicting in many occasions
3. Lack of Planning :
   - All requirements are critical, no priority is defined
4. Improper Change Management :
   - New requirements get added in the middle of the project
   - Users/customers are busy and not available to specify requirements
   - Signed-off requirements keep changing
5. Language barriers
   - Have interpreter(e.g. Japanese projects)
   - Use more visual communication tools

6.4 Why do we need good requirements?

Why do we need the requirement phase?
What is it supposed to achieve?



6.4.1 Characteristics & Impact of bad Req

Characteristics of defective Requirements :

- Lack of Cohesiveness
- Lack of Completeness
- Lack of Correctness
- Lack of Consistency
- Lack of Project Relevance
- Lack of Testability, Usability, Validatability
- Ambiguous

Impact of

- Function f
- Extensivel
- Extensivel
- Extensivel
- Poor quali
- Not consid delivered
- Sometime

The percentage of defects originating during requirement engineering are estimated as :

50% - Karl Wiegers 2001
42% - A. Winngrove
60%-64% - requirements and design EBG Consulting

6.5 Requirements Engineering

Requirements Engineering is a disciplined, process-
definition, documentation, and maintenance of soft
throughout the software development life cycle

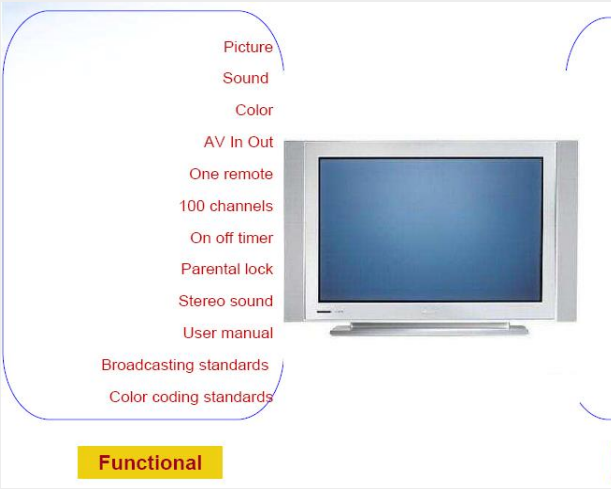
Software requirements engineering is made up of t
"**Requirements Development**" and "**Requireme**

- Requirements development involves all of the activ
eliciting, analyzing, specifying, and validating the r
- Requirements management involves the activities
changes to the baselined requirements, performing
requested changes, approving or disapproving thos
implementing the approved changes

6.6 Functional & Non-functional Requirements

Functional Requirements	Non-functional Requirements
It specifies the input and output behaviour of a systems. It defines how software behaves to meet user needs. Example : Functional requirements of a health insurance company include : • Determining Claimant Eligibility • Paying Claims • Calculating Premium	It represents quality attributes of the system : Availability, Maintainability, Performance, Portability, Reliability, Robustness, Security, Scalability etc. Examples: • It should be easy to see the history of transactions • The system should be available 99.9% of the time • The system should use SSH public-key cryptography to authenticate the remote computer and allow the remote computer to authenticate the user, if necessary • The system should be able to serve at the most 100 concurrent users

Functional Vs Non-Functional Requirements



6.7 Non Functional Requirements: FURPS +

Functionality
Usability
Reliability
Performance
Supportability
+ other such quality attributes

Non Functional Requirements: FURPS +

1. **Functionality -** The degree to which the software satisfies stated needs as indicated by the following sub attributes: suitability, accuracy, interoperability, compliance, and security.
2. **Usability -** The degree to which the software is easy to use as indicated by the following aspects: understandability, learnability, operability.
3. **Reliability -** The amount of time that the software is available for use as indicated by the following aspects: maturity, fault tolerance, recoverability.
4. **Performance -** is concerned with characteristics such as throughput, response time, recovery time, start-up time, and shutdown time.
5. **Supportability -** is concerned with characteristics such as testability, adaptability, maintainability, compatibility, configurability, installability, scalability, and localizability.
    - Adaptability: The ability to change the system to deal with additional application domain concepts
    - Maintainability: The ability to change the system to deal with new technology or to fix defects
    - Internationalization: The ability to change the system to deal with additional international conventions such as languages, or number formats, styles
    - Portability: The ease with which a system or component can be transferred from one environment to another
    - Eg. Application should be available for German & Japanese users, Personalization to include user

details, configuring options, Application should work on Mac too

The "+" in FURPS+ helps us to remember concerns such as:

1. **Design requirements -** Design constraints for designing system, for example, if you specify that a relational database is required, that's a design constraint.
2. **Implementation requirements -** Specifies or constrains the coding or construction of a system. Examples might include required standards, implementation languages, and resource limits.
3. **Interface requirements -** Specifies an external item with which a system must interact, or constraints on formats or other factors used within such an interaction.
4. **Physical requirements -** Specifies a physical constraint imposed on the system , shape, size, or weight.

6.8 Stable and Volatile Requirements

Stable Requirements

- They are related to the core activities of the system and its domain
- For example, in an organization there will be requirements concerned with employees, departments, payroll etc.

Volatile Requirements

- These are requirements that are likely to change during the system development process or after the system has been become operational
- Examples of volatile requirements are requirements resulting from organization's leave policies or Income Tax policies enforced by the country's government bodies

6.9 Baselining Requirements

The requirements are baselined at the end of the Requirements Development phase & ideally signed-off by the customer

A requirements baseline is:

- A snapshot in time of a set of requirements
- Used as a mechanism to track changes as the project progresses
- Constitutes agreement on scope between customer and development team

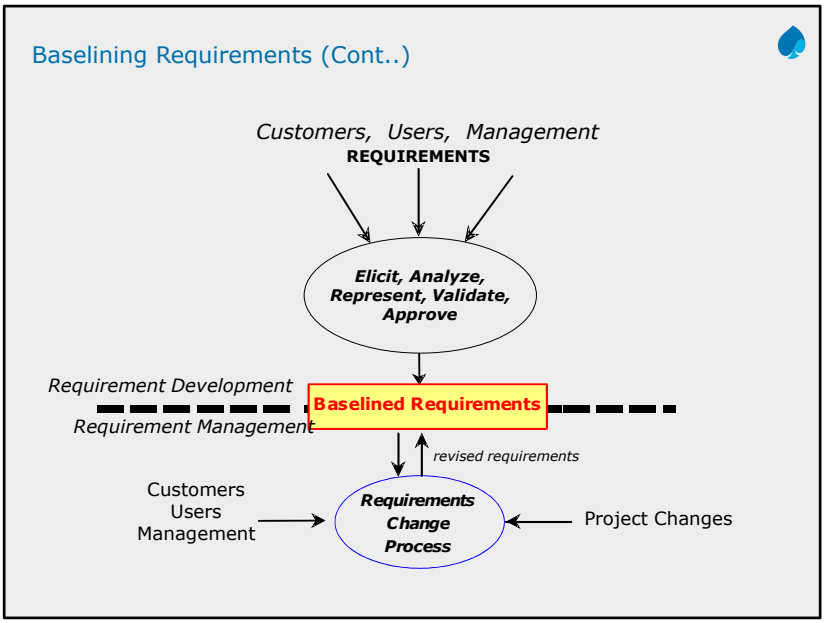
Scope drives estimate, schedule, staffing, deadlines

New baselines are typically created at major project milestones

Baselining Requirements

Usually projects start with unclear requirements and expectations. Lack of base lined requirements can result in chaos with lots of requirements changes resulting in requirements and scope creeps. Baselines can also help in acceptance testing and prototyping efforts. Baselines are especially valuable in fixed price contracts.

A baseline is all about getting to a common base agreement between stakeholders. It essentially involves setting the right expectations including responsibilities, risks, assumptions, deliverable and approaches. Once an agreement is reached; it could be put in source control to manage the base line going forward.



Baselining Requirements

Once sufficient requirements have been discovered and documented as an analyst you maybe required to help facilitate the planning of releases / versions.

Firstly you ensure that the requirements or features have been prioritized by the customer. This may require some diplomacy if you have a number of stakeholders with different priorities. This prioritization process is crucial and any conflicts must be resolved before starting development work where possible.

Secondly you will need to work with the development team to help them estimate the duration and risk of each requirement. Once completed you will be prepared to facilitate the discussion to start baseline planning.

The features that make up each version should be derived through a balance of priority value (determined by the project customer), effort required (determined by the development team) and perceived risk (determined by the development team).
Baselined requirements are the start point for Requirements Management.

6.10 Requirements Traceability

It is one of the essential activities of good requirements management

Requirement traceability helps in assessing the impact of requirements change

Traceability is used to track the relationship between each unique product-level requirement and its source

> ***"The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another."***
>
> ***The IEEE Definition***

Requirement Traceability

Requirements traceability is concerned with documenting the life of a requirement. It should be possible to trace back to the origin of each requirement and every change made to the requirement should therefore be documented in order to achieve traceability. Even the use of the requirement after the implemented features have been deployed and used should be traceable.

Requirements come from different sources, like the business person ordering the product, the marketing manager and the actual user. These people all have different requirements on the product. Using requirements traceability an implemented feature can be traced back to the person, or group, that wanted it during the requirements elicitation. This can be used during the development process to prioritize the requirement, determining how valuable the requirement is to a specific user. It can also be used after the deployment when user studies show that a feature is not used, to see why it was required in the first place.

6.11.1 Requirement Traceability Matrix

RTM is a table containing requirements of a project and their relation to the engineering work products

It ensures completeness in translating requirements to the delivered work products

Advantages

- Ensures completeness of testing against requirements
- Facilitates the impact analysis of the requirements change on all the related work products
- Enables scope analysis for regression testing
- Helps judging requirements stability from a customer
- Helps to analyze Requirements creep

Suggested Tools

- Rational Requisite Pro
- Excel Sheet (Traceability template)

The classic way to perform traceability is by constructing a traceability matrix

6.11.2 Requirement Traceability Matrix – Example

Example : XYZ Banking Application

This is a site that aims at computerizing the business process of XYZ bank. The following are the business targets that are to be achieved :

Step 1: Business Requirement Document (BRD) :

BR ID	Module Name	Roles Involved	Description
BR_1	Login	Customer Manager	A customer can login using login module. A Manager can login using login module.
BR_2	Enquiry	Customer Manager	A customer can view balance of his accounts only. A manager can view balance of all customers who come under his supervision.
BR_3	Fund Transfer	Customer Manager	A customer can transfer funds from his 'own' account to any destination account. A manager can transfer funds from any account under his supervision.
BR_4	Loan Process	Customer Manager	A customer can access loan information and apply for loan. A manager can grant, reject, suggest changes to the loan request under his supervision.

Requirement Traceability Matrix – Example

Below is our Technical Requirement Document (TRD) based on the interpretation of the Business Requirements Document (BRD)

Step 2: Technical Requirement Document (TRD) for **BR_1 (Login)**

TR ID	Module Name	Description
TR_01	Login Module	User ID and Pwd must not be blank
TR_02	Login Module	If User ID and password are valid. Login.
TR_03	Login Module	The Pwd field should not allow copy paste from any source.

Requirement Traceability Matrix – Example

Step 3: On the basis of Business Requirement Document (BRD) and Technical Requirement Document (TRD), testers start writing test scenarios.

Test Scenarios :

Test Scenario ID	Test Scenarios
TS_1	Verify with valid User Credentials
TS_2	Verify with invalid User Credentials
TS_3	Verify the length provided for the user name field
TS_4	Verify the length provided for the Pwd field
TS_5	Verify copying the Pwd from external file and pasting it into Pwd field.

The BRD and TRD are not documented by QA teams. QA teams are mere the consumers of these documents along with the other project teams.
QA team then comes up with the list of high-level test scenarios to test.

Requirement Traceability Matrix – Example

Step 4: For each test scenario, write at least 1 or more test cases.

Test Cases :

Test Case ID	Test Condition	Test Steps	Test Data	Expected Result
TC_1	To validate that user is able to login successfully with valid User ID & valid Pwd.	1.Go to Login Page 2.Enter User ID 3.Enter Password 4. Click Login	user1 1234B	Login Successful
TC_2	To validate that user is unable to login with invalid User ID & valid Pwd.	1.Go to Login Page 2.Enter User ID 3.Enter Password 4. Click Login	sampleuser 1234B	Login Failure
TC_3	To validate that user is unable to login with valid User ID & invalid Pwd.	1.Go to Login Page 2.Enter User ID 3.Enter Password 4. Click Login	user1 samplePwd	Login Failure

Requirement Traceability Matrix – Example

Step 5: You can now start creating RTM.

Identify the Test scenarios, Technical Requirements and Business Requirements that the test cases are verifying.

BR ID	TR ID	TS ID	TC ID
BR_1	TR_01	TS_3	
		TS_4	
	TR_02	TS_1	TC_1
		TS_2	TC_2
		TS_2	TC_3
	TR_03	TS_5	
BR_2			
BR_3			
BR_4			

At this stage, the RTM can be used to find gaps. For example, in the above RTM, you see that there are no test cases written for TS_3, TS_4, TS_5.

it will indicate the places where the test team needs to work some more to ensure 100% coverage.

This is the preliminary version of your TM but generally, does not serve its purpose when you stop here.

Maximum benefits can be achieved from it when you extrapolate it all the way to defects.

Requirement Traceability Matrix – Example

Step 6: Expand the RTM to include test case execution status and defects.

BR ID	TR ID	TS ID	TC ID	Status	Defect ID
BR_1	TR_01	TS_3	TC_5	Pass	
		TS_3	TC_6	Fail	D_05
		TS_4	TC_7	Pass	
	TR_02	TS_1	TC_1	Pass	
		TS_2	TC_2	Fail	D_11
		TS_2	TC_3	Pass	
	TR_03	TS_5	TC_10	Fail	D_02

This RTM shows overall status of all requirements along with Test Cases.

The above version of the Traceability Matrix is generally prepared during or after Test execution.

During execution, it gives a consolidated snapshot of how work is progressing.

When 'Defect ID' column is used to establish the backward Traceability, RTM provides transparency back to the business requirement that has most defects thus showcasing the Quality in terms of what the client desires.

6.11 Requirements Change

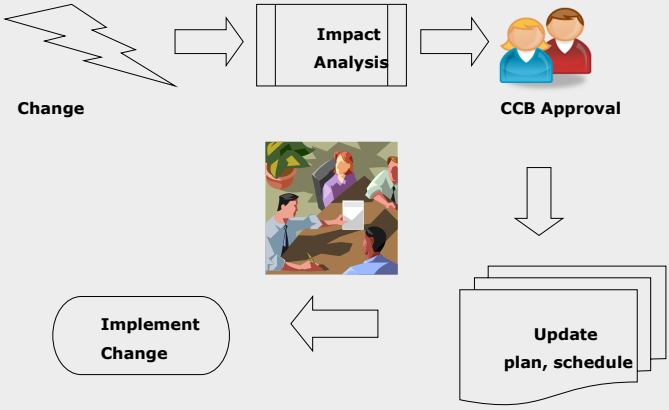
Users **can** propose **requirements change** at any stage of **SDLC.**

Requirements change because :

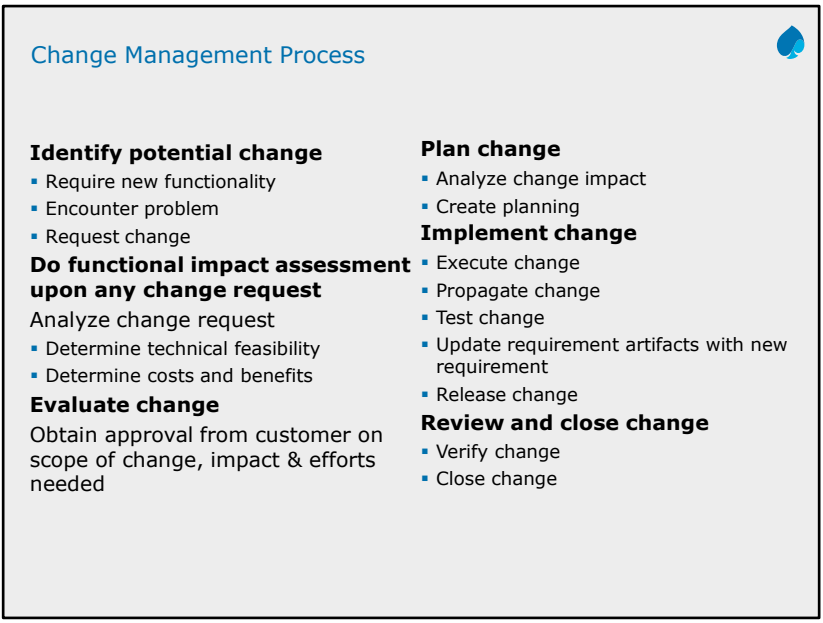
- Understanding of the problem improved : Customer's expectations change once they see the product taking shape
- Initial elicitation activities are imperfect : We failed to ask the right people the right questions at the right time.
- The priority of requirements from different viewpoints change during the development process
- Business needs evolve
- The users changed their perceptions : Customers may specify requirements from a business perspective that conflict with end-user requirements
- The external environment changed : The business and technical environment of the system changes during its development
- We either failed to create or follow a process to help manage change.

6.11.1 Change Management Process

Establish a formal process for managing changes to the system requirements



A change management process consists of number of predefined processes and standards to be followed to manage changes to the system requirements.
The Change Control Board (CCB) that includes the project stakeholders are involved in the execution of change management process.



There are six main activities, which jointly form the change management process. They are: Identify potential change, Analyze change request, Evaluate change, Plan change, Implement change and Review and close change.

1. **Identify potential change:** The change in requirements is identified. This could come from an analysis of the requirements, new customer needs, or operational problems with the system. The change may be initiated by a customer/user, business analyst or any other stakeholder involved in defining the requirement.

2. **Analyze change request :** The proposed changes are analyzed. This includes how many other requirements and system components are affected by the change. The analysis are also carried out to understand approximate cost in terms of money, efforts & timer to successfully implement the change in the product and overall impact on the project schedule.

3. **Evaluate Change :** On the basis of change request analysis, the Change Control Board (CCB) decides whether to implement or not the change. This decision will be based on factors such as : the severity of the changes, cost versus benefit etc.

4. **Plan Change :** The first step in managing change is building awareness around the need for change and creating a desire among employees. If the CCB decides that the change is to be implemented, it is communicated to any stakeholders e.g. user/customer, project manager, project team etc. that are affected by that change. The change management team must develop a plan for successful implementation of the change. A key part of the change control process is to carry out an impact analysis of the proposed new or modified requirement. The impact analysis involves estimating the time, effort and cost of implementing the change and any other requirements that are affected by the change are also considered.a

6.11.2 Requirement Creep

"Scope creep (also called requirement creep and feature creep) in project management refers to uncontrolled changes or continuous growth in a project's scope. This can occur when the scope of a project is not properly defined, documented, or controlled."

Why does Requirement Creep occur ?

- Poor requirement analysis
- Not involving customers early though
- Insufficient detailing on the complexity of the project
- Lack of change control
- Gold Plating
- Unwillingness to say no to a client

Why does Requirement Creep occur ?

Poor requirement analysis

The customers are not sure enough about what they want from the system and they end up stating vague requirements

Not involving customers early though

This refers to having false confidence that you know exactly what the customers expect from the system

It is very important to involve customers in the requirement analysis as well as design phase

Insufficient detailing on the complexity of the project

Many projects runs into problems because they are executed for the first time and there is no enough detailed information available on what to expect from the project and how to implement the same in a standard manner

Lack of change control

You can expect requirement creep in most projects, therefore it is important to design a process to manage these changes

Gold Plating

This term is given to the practice of exceeding the scope of a project in the belief that a value is being added

As a team proceeds through the various phases of a project, frequently one or more of the team members will strive to improve or perfect the product

These changes inevitably consume time and budget and are not guaranteed to increase customer satisfaction

Unwillingness to say no to a client

The project team's or an individual's desire to please the customer and reluctance to say "no" can also lead to requirement creep

Requirement Creep (Cont..)

- Even when there is a clearly defined project scope, one must be aware that Requirement creep can still occur during project development.
- Requirement creep tends to additional requirements needed to achieve the new objectives. This can overwhelm the capacity of the resources allocated to the project resulting into project missing deadlines, budgets or complete failure.
- Therefore, preventing requirement creep and managing requirement creep is the key to successful project management.

Measures to control Requirement Creep

Following are some of the common measures those can be used to minimize requirement creep

- Utilize various techniques for more thoroughly defining user requirements up front
- Involve the customers in the earliest stages of the project possible
- Achievable goals should be set
- Prioritize requirements into must-haves versus nice-to-haves
- Project managers have to learn when to say no and when to say yes
- When the client wants to change or add a requirement, the change or addition should be analyzed for resource, cost, and schedule impacts
- Perform constant internal review to make sure the project is on track and within scope
- Set a timeline or due date for all tasks
- Have a tracking system for tasks, due dates, and action items

Answers:

Question1: True

Question2: False

Question3: Incremental approach

Question4: Inter-Personal Skills

Question5: Traditional approach

Review Question

Question 1: Which of the non functional requirements can ensure that the application should be available for German & Japanese users?

Question 2: In which of the requirement gathering patterns, requirements are specified in detail and passes thru multiple reviews and sign-offs?

Question 3: Volatile Requirements are likely to change during the system development process or after the system has been become operational. (T/F)

Question 4: The requirements are baselined at the beginning of the Requirements Development phase & ideally signed-off by the development team. (T/F)

Question 5: Which type of requirement traceability is used to validate whether the project is evolving in the desired direction and for the right product?