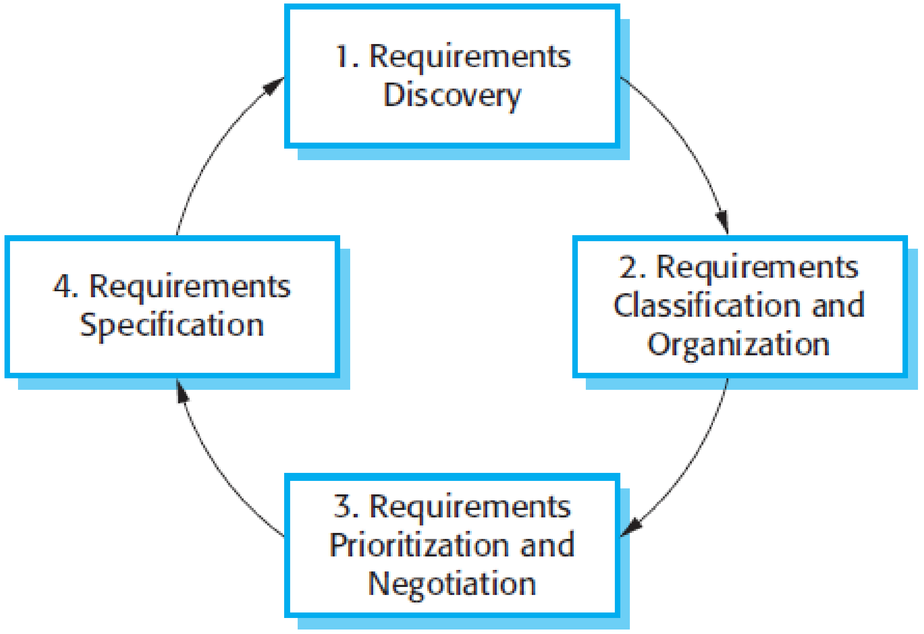
Requirement Elicitation

- In this activity, software engineers work with customers and system end-users to find out about the application domain, what services the system should provide, the required performance of the system, hardware constraints, and so on.


1. Requirements Discovery
2. Requirements Classification and Organization
3. Requirements Prioritization and Negotiation
4. Requirements Specification

Requirement Elicitation Cont.

- Requirements elicitation and analysis may involve a variety of system stakeholder who should have some direct or indirect influence on the system requirements.
- Stakeholders include end users who will interact with the system and anyone else in an organization who will be affected by it.
- Each organization will have its own version or instantiation of this general model depending on local factors such as the expertise of the staff, the type of system being developed, the standards used, etc.

Process of Req. Elicitation

- Requirement Discovery
  - This is the process of interacting with stakeholders of the system to discover their requirements. Some of the methods are questionnaires, JAD, Interview, Prototyping.
- Requirement classification and Organization
  - This activity takes the unstructured collection of requirements, groups related requirements, and organizes them into coherent clusters.
  - The most common way of grouping requirements is to use a model of the system architecture to identify sub-systems and to associate requirements with each sub-system.

Process of Req. Elicitation

- **Requirement Prioritization and negotiation**
  - Inevitably, when multiple stakeholders are involved, requirements will conflict. This activity is concerned with prioritizing requirements and finding and resolving requirements conflicts through negotiation.
- **Requirement Specification**
  - The requirements are documented and input into the next round of the spiral. Formal or informal requirements documents may be Produced.

Process of Req. Elicitation

- Stakeholders often don't know what they want from a computer system except in the most general terms.
- Stakeholders in a system naturally express requirements in their own terms and with implicit knowledge of their own work.
- Different stakeholders have different requirements and they may express these in different ways.
- Political factors may influence the requirements of a system.
- The economic and business environment in which the analysis takes place is dynamic.

Use cases

- A behaviorally related sequence of steps (a scenario), both automate and manual, for the purpose of completing single business task.
- It is a collection of success and failure scenarios that describe actor using a system to support a goal.
- It is made up of set of possible sequence of interaction between the system and actor in a particular environment.

Actor

- Actor is something with behavior such as person computer system or organization that use the service of the system or provide the service to the system.
- It specifies a role played by a user or any other system that interact with the subject.
- Actors are play not only by people but also by organization, software and machine.
  - Types of Actor
    - Primary Actor
    - Supporting Actor
    - Offstage Actor

Types of Actors

- **Primary Actors** are the shareholder that primarily benefits from the execution of the use case by receiving something of measurable or observable value.
- **Supporting Actors** are those actors which provide assistance from other supporting system to fulfill primary actors goal. Example- ATM System in ATM Machine.
- **Offstage Actor** are persons who has an interest but not primary actor.

How to find Use Cases

- Choose the system Boundary.
- Identify the primary actor.
- Identify the goal for each primary actor.
- Define the use case that satisfy the user’s goal.

Use Case Diagram

- A diagram that depicts the interaction between the system, external system and users. In other words, It graphically describes who will use the system and in what ways the user expects to interact with the system.
- It show the relationship between the user and use case in which the user is involved.
- The use cases is represented by circle or eclipse in use case diagram.

Use Case Diagram

Relationship	Symbol	Meaning
Communicates	——	An actor is connected to a use case using a line with no arrowheads.
Includes	⇠	A use case contains a behavior that is common to more than one other use case. The arrow points to the common use case.
Extends	⇢	A different use case handles exceptions from the basic use case. The arrow points from the extended to the basic use case.
Generalizes	—▷	One UML "thing" is more general than another "thing." The arrow points to the general "thing."

Requirement Specification

- System Requirements specification is the process of writing down the user and system requirements in a requirements document.
- The user and system requirements should be clear, unambiguous, easy to understand, complete, and consistent.
- In practice, this is difficult to achieve as stakeholder interpret the requirements in different ways and there are often inherent conflicts and inconsistencies in the requirements.
- The requirements document should not include details of the system architecture or design.

The Structure of Requirement Document

- Preface
- Introduction
- Glossary
- User Requirement definition
- System Architecture
- System Requirement Specification
- System Models
- System Evolution
- Appendices
- Index

Steps to Write System Requirement Specification

- Define the purpose with an outline (or use SRS Template)
- Define your product's purpose
- Describe what you will build
- Detail your specific requirement
- Deliver for approval

Why it is practically impossible to execute all design information

- You may have to design an initial architecture of the system to help structure the requirements specification.
- In most cases, systems must interoperate with existing systems, which constrain the design and impose requirements on the new system.
- The use of a specific architecture to satisfy non-functional requirements may be necessary. An external regulator who needs to certify that the system is safe may specify that an already certified architectural design be used.

Ways of writing a simple requirement specification

- Natural Language Sentence
  - The requirements are written using numbered sentences in natural language. Each sentence should express one requirement.
- Structure Natural Language
  - The requirements are written in natural language on a standard form or template. Each field provides information about an aspect of the requirement.
- Graphical Notation
- Graphical models, supplemented by text annotations, are used to define the functional requirements for the system; UML use case and sequence diagrams are commonly used.

Ways of writing a simple requirement specification

- Design Description Language
  - This approach uses a language like a programming language, but with more abstract features to specify the requirements by defining an operational model of the system.
  - This approach is now rarely used although it can be useful for interface specification.
- Mathematical Specification
  - These notations are based on mathematical concepts such as finite-state machines or sets. Although these unambiguous specifications can reduce the ambiguity in a requirements document, most customers don’t understand a formal specification.

Change Control in Project Management

- Project management change control is the system a team uses to make major changes to a previously approved project.
- It can include budget additions and subtractions, deadlines or goalposts, and even new hiring as project needs evolve over time.
- Project change control ensures that all stakeholders have a say (or can at least agree on a process for someone else to carry out) for how foundational project elements can be revised when needed.
- It saves time, streamlines communication, and leads to a repeatable process for effective change.

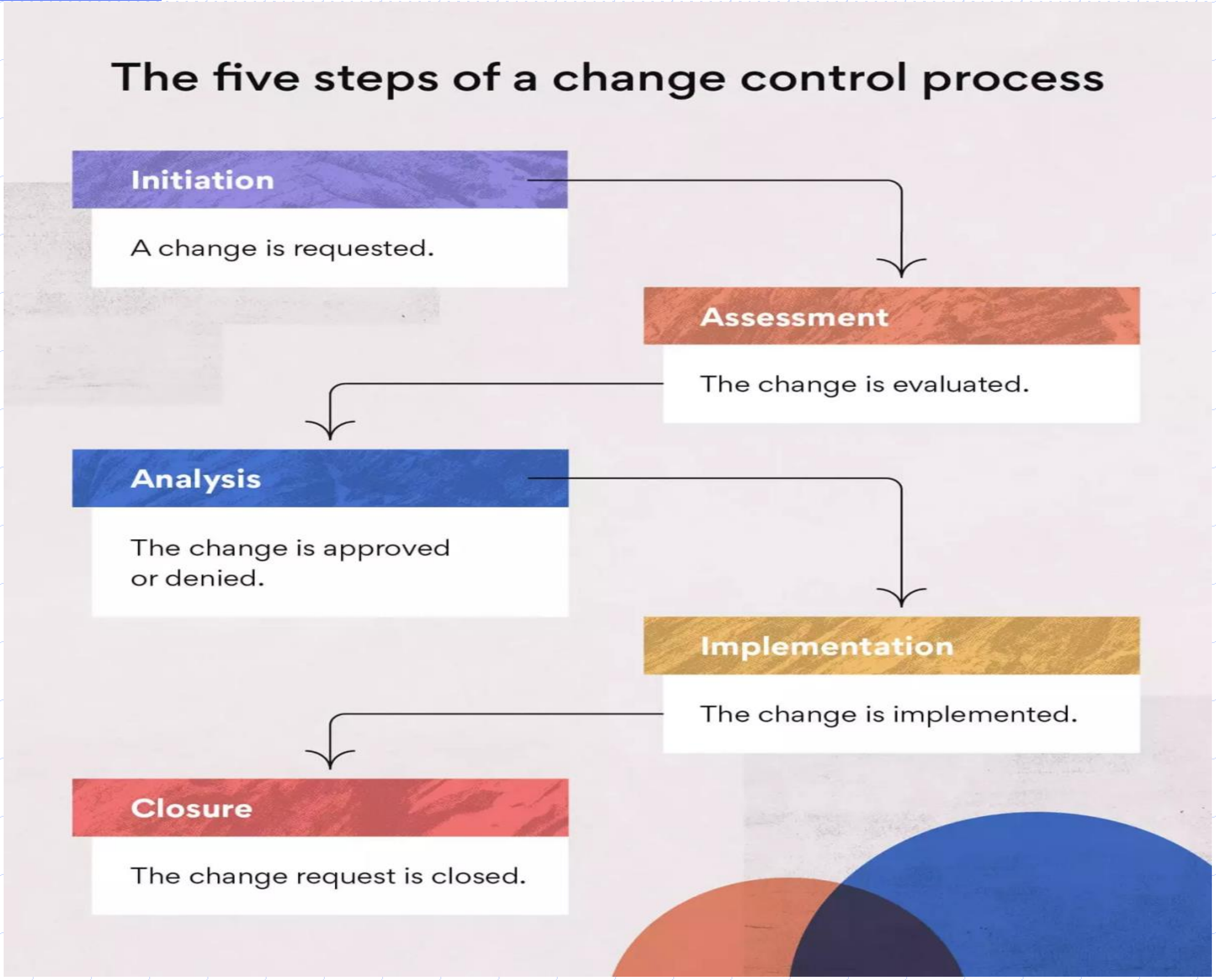
Change Control in Project Management

- It is a way to capture that change from the point where it's been identified through every step of the project cycle.
- It includes evaluating the request and then approving, rejecting or deferring it.
- The purpose of this process is to make sure that you're not changing things in the project that don't need to be changed.
- The change control process is part of the larger Change Control Management.

Key elements of Change Control

- Change Control Board
- Change Request
- Change Order
- Change Log

Key elements of Change Control



The five steps of a change control process

Initiation
A change is requested.

Assessment
The change is evaluated.

Analysis
The change is approved or denied.

Implementation
The change is implemented.

Closure
The change request is closed.

How to implement Change Control

- Define the scope Change
- Evaluate the impact of potential Change
- Submit a format change request to higher level management.
- Adjust Project Plans
- Communicate and implement the approved change request.

Tips for Change Control

- Use Project Management Tools
- Understand how to identify risk
- Develop a change management plan
- Use a change log and create a change request form

How to achieve Integrate Change Control

- Create a change Control System
- Develop Change Control Procedure
- Write down the impact of possible Changes
- List out all the task associated with the project.
- Draft templates

Process of Software Inspection

- Inspection Meeting
  - Reviewer goes over the product line by line. At any line, all issues are raised.
  - Discussion follows to identify if a defect.
  - Decisions are recorded at the end of the inspection meeting.
  - Scribe present the list of defects. If few defects, the work product is accepted; else, it might be asked for another review.
  - Group does not propose solutions through some suggestions that may be recorded.
  - A summary of the inspection is prepared, which is useful for evaluating effectiveness.

Process of Software Inspection

- Rework and Follow-up
  - Defects bin the defect list are fixed later by the author. These modifications are made to repair the discovered errors. Once fixed, the author gets it OKed by the moderator or goes for another review.
  - A reinspection may or may not be required.
  - Once all defects are satisfactorily addressed, the review is completed, and collected data is submitted.

Process of Software Inspection

- Inspection Roles
  - Author or owner
  - Inspector
  - Moderator or Chairman
  - Scribe
  - Reader
  - Chief Moderator

Advantages of Inspection

- The goal of this method is to detect all faults, violation and other side effects.
- Authors and other reviewers do complete preparation before conducting an inspection.
- A group of people are involved in the inspection procedure; multiple diverse views are enlisted.
- Every person in the inspection team is assigned a specific role.
- The reader in the inspection reads out the document sequentially in a structured manner so that all the points and all the code is inspected thoroughly.

Disadvantages of Inspection

- scheduling can become an issue since multiple people are involved.
- Time-consuming as it needs preparation as well as formal meetings.
- It is not always possible to go through every line of code with several parameters and their combination to ensure the correctness of the logic, side effects and appropriate error handling.

Deskchecks

- The term "desk checking" refers to the manual approach of reviewing source code (sitting a desk), rather than running it through a debugger or another automated process.
- In some cases, a programmer may even use a pencil and paper to record the process and output of functions within a program.
- Developers may desk check their code before releasing a software program to make sure the algorithms are functioning efficiently and correctly.
- It is useful for uncovering logic errors and other issues within a program's source code, it is time-consuming and subject to human error.

A desk check is normally done as a table with columns for:

- Pseudo code **line number** column (Pseudo code doesn't normally have lines numbers, but these are necessary in a desk check to specify the line(s) being executed)
- **One column per variable** used. The columns should be in alphabetical order on variable name with the variable name at the top of the column.
  - As the algorithm is executed, the new values of the variables are put in the appropriate column. Show working for calculations. e.g. the variable column heading *discount Price* could be used rather than the actual variable name *discountPrice*.

A desk check is normally done as a table with columns for:

- A **condition** column. The result of the condition will be true (T) or false (F). As the algorithm is executed, conditions are evaluated and the details are recorded in the column. Show working when evaluating the conditions. This is used whenever a condition is evaluated.
- An **Input/Output** column is used to show what is input by the user and displayed by the program. Show inputs with: the variable name, a "?" and the value input e.g. price ? 200.

Walkthrough

- A *walkthrough* is an informal way of presenting a technical document in a meeting.
- Unlike other kinds of reviews, the author runs the walkthrough: calling the meeting, inviting the reviewers, soliciting comments, and ensuring that everyone present understands the work product.
- It typically does not follow a rigid procedure; rather, the author presents the work product to the audience in a manner that makes sense.

Walkthrough

- Walkthroughs are used when the author of a work product needs to take into account the perspective of someone who does not have the technical expertise to review the document.
- For example, a requirements analyst must make sure that the use cases she builds will provide the functionality that the users need, but the user representatives may not have seen use cases before and would be overwhelmed by them.
- The user sit in the meeting silently

Pair Programming

- Innovative practice, Introduced in XP is that programmers work in pairs to develop the software.
- A pair of programmers are sit together at the same workstation to develop the software.
- However, the same pairs do not always program together.
- Pairs are created dynamically so that all team members work with each other during the development process.

Advantages of Pair Programming

- It supports the idea of collective ownership and responsibility for the system. The team has common responsibility for resolving problems.
- Pair programming is a less formal process that probably doesn't find as many errors as code inspections, it is a much cheaper inspection process than formal program inspections.
- It helps support refactoring, which is a process of software improvement. Where pair programming and collective ownership are used.

Summarize Advantages of Pair Programming

- Fewer Coding Mistakes
- Knowledge is spread among the pairs.
- Reduced effort to Co-ordinate.
- Increase Resiliency

Challenges of Pair Programming

- Efficiency
- Equally Engaging pairs
- Social and Interactive Process
- Sustainability

Code Review

- Code Review, is the act of consciously and systematically convening with one’s fellow programmers to check each other’s code for mistakes and has been repeatedly shown to accelerate and streamline the process of software development like few other practices can.
- A perfect code should be easy to understand, flexible to modify, and readable. But since the work develops quickly, it may overlook these factors. That's why code reviews procedures are necessary to improve its quality.
- A code review accelerates and streamlines the software development process.

Benefits of Code Review

- Ensures consistency in design and implementation.
- Optimizing code for better performance.
- Collaborating and sharing new techniques
- Tracking project requirement and quality

Process of Code Review

- Know what you're looking for in a code review.
- Understand the different ways to conduct a code review.
- Hold regular group meetings where participants can receive feedback on their particular areas while also receiving notes about any issues.
- Make comments clear and specific.
- Be open to improvement.
- Be available for discussions.
- Start with small changes, then review more complex ones.
- Keep the status of a commit updated.

Where should code review be done

- Over the shoulder.
  - It happen at a developer's desk, where an experienced team member walks through the new code and makes suggestions.
- Email Passaround
  - The code is sent by e-mail once it is ready. Although e-mails offer a more passive approach to code review, content can become nested in multiple replies and difficult to manage and search.
- Pair Programming
- Tool assisted
  - It could be open source or paid, like GitHub, BitBucket, etc. Today, most people prefer it.

Use inspection to manage commitments

- A successful project needs more than just a blanket agreement between team members.
- It's very easy for someone to "agree" to a document, only to turn around later and decide that he didn't fully understand what he was agreeing to.
- The goal of an inspection is to build consensus on the document by gaining a real commitment from everyone who has read it.
- When a reviewer approves a document, he takes responsibility for its contents, and if the document has defects, he shares some of the blame for missing the mistake.

Use inspection to manage commitments

- The best way to reach consensus among the inspection team is for each person to feel like he or she made a real contribution to the document.
- The inspection meeting accomplishes that by allowing each person to find problems in the document and help the rest of the team find a solution to each problem.
- This is why it's important for the team to go beyond just pointing out the defects in the document and actually come up with replacement wordings that fix the defects.