Software Development Life Cycle (SDLC)

Introduction

- Name
- College
- Skillset(Worked/Exposure)
- Hobby/Fun fact

AGENDA

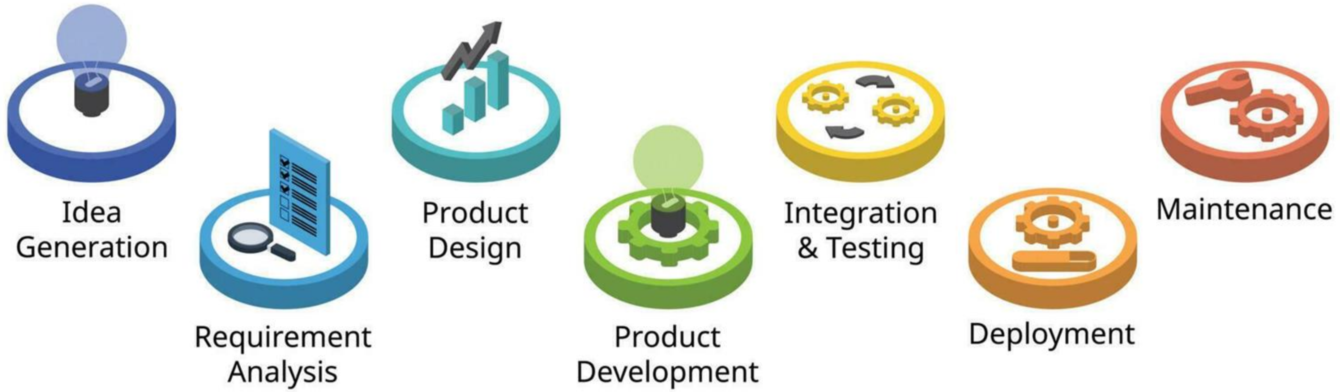
- Understanding the Software Development Lifecycle
- Working on an Agile Team
- Software Testing and Quality Assurance
- Software Testing Tools and Techniques
- Communication and Collaboration
- Common Tools (ServiceNow, JIRA, etc.)

What we are trying to Accomplish?

- Understand what is SDLC
- Purpose of SDLC
- Phases of the SDLC (requirements gathering, design, development, testing, deployment, and maintenance)
- Importance of following a structured SDLC process

Understanding the Software Development Lifecycle

- SDLC stands for ***Software Development Life Cycle***
- It is a structured process used to develop and maintain software systems
- SDLC consists of a set of phases or stages that guide the software development process from start to finish



CIRCLE of Life

1. Teams come, operate, evolve and disband
2. People come, grow, and eventually move on
3. Projects come, grow, evolve and enter stasis

Your project has to accommodate these facts of project lifecycle!!

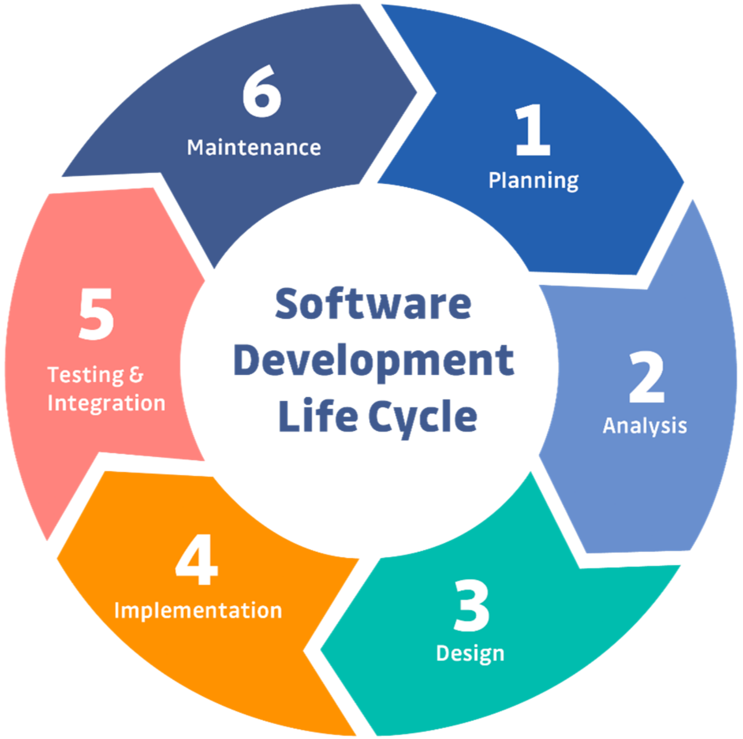
Software Development Life Cycle (SDLC): Why?

Purpose

- Leads to good software
- Reduces risk
- Enables visibility and measurement
- Enables team's teaming

Key attributes

- Outcomes of processes or products are key deliverables
- Roles are clear
- Pre- and post-conditions are understood and held true



Phases of the SDLC

The SDLC (Software Development Life Cycle) comprises several essential phases:

- **Requirements Gathering:** Identifying and documenting project needs and objectives by consulting stakeholders.
- **System Analysis:** Evaluating existing systems and defining requirements for the new system.
- **System Design:** Planning the software architecture and user interface.
- **Coding and Implementation:** Writing and integrating the code to create the software.
- **Testing:** Executing tests to ensure the software's functionality and quality.
- **Deployment:** Launching the software for users to access and utilize.
- **Maintenance and Support:** Providing ongoing support, updates, and improvements to the software.

Benefits of the SDLC

Increased Quality

The SDLC helps ensure that software is developed to a high standard by following a structured process

Reduced Risk

The SDLC helps reduce the risk of project failure by identifying and mitigating risks early in the process

Improved Communication

The SDLC helps improve communication between stakeholders by defining clear roles and responsibilities

Increased Efficiency

The SDLC helps improve efficiency by providing a sustainable process that can be used to develop software more quickly and effectively

Objectives of SDLC

Ensure High-Quality Software

- SDLC provides a systematic approach to software development, ensuring the creation of high-quality software products.
- It emphasizes rigorous testing, quality assurance, and adherence to industry standards.

Manage Project Scope and Resources

- SDLC helps in defining the project scope, requirements, and goals, enabling effective resource allocation and management.
- It assists in estimating project timelines, costs, and risks, facilitating better project planning and control.

Enhance Communication and Collaboration

- SDLC promotes collaboration between development teams, stakeholders, and users.
- It establishes clear communication channels, facilitating effective interaction and understanding among all involved.

Ensure Customer Satisfaction

- SDLC focuses on understanding and meeting customer requirements, expectations, and needs.
- It helps deliver software solutions that align with user expectations, improving overall customer satisfaction.

Benefits of a Structured SDLC Process

Improved Project Management

Better planning and tracking of project progress, efficient utilization of resources, and timely delivery

Enhanced Quality Assurance

Early identification and resolution of defects, consistent and reliable software output

Effective Communication

Clear communication among team members and stakeholders, reduced misunderstandings and misinterpretations

Risk Mitigation

Identification and management of project risks, increased chances of project success

Cost and Time Efficiency

Accurate estimation and allocation of resources, minimized rework and cost overruns

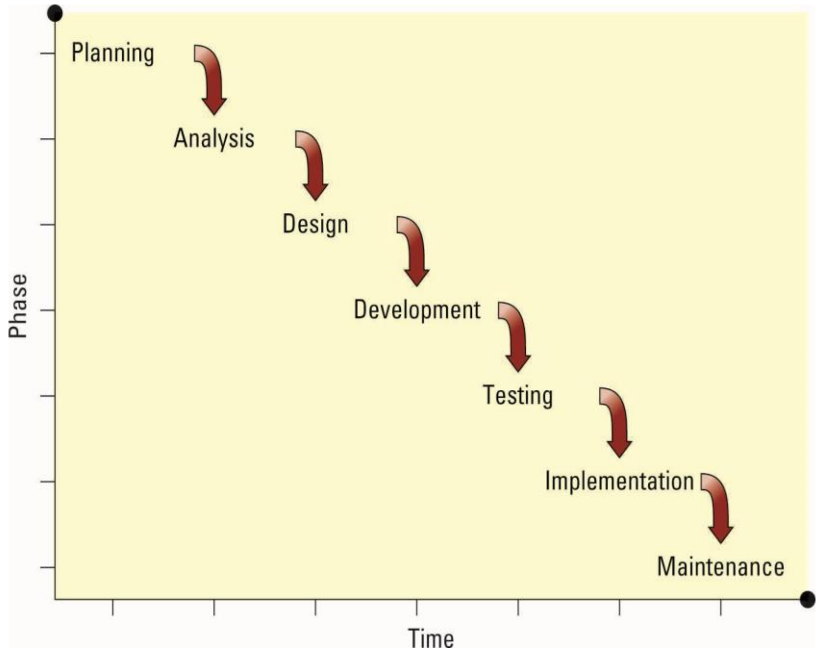
Software Development Methodologies

There are a number of different software development methodologies including

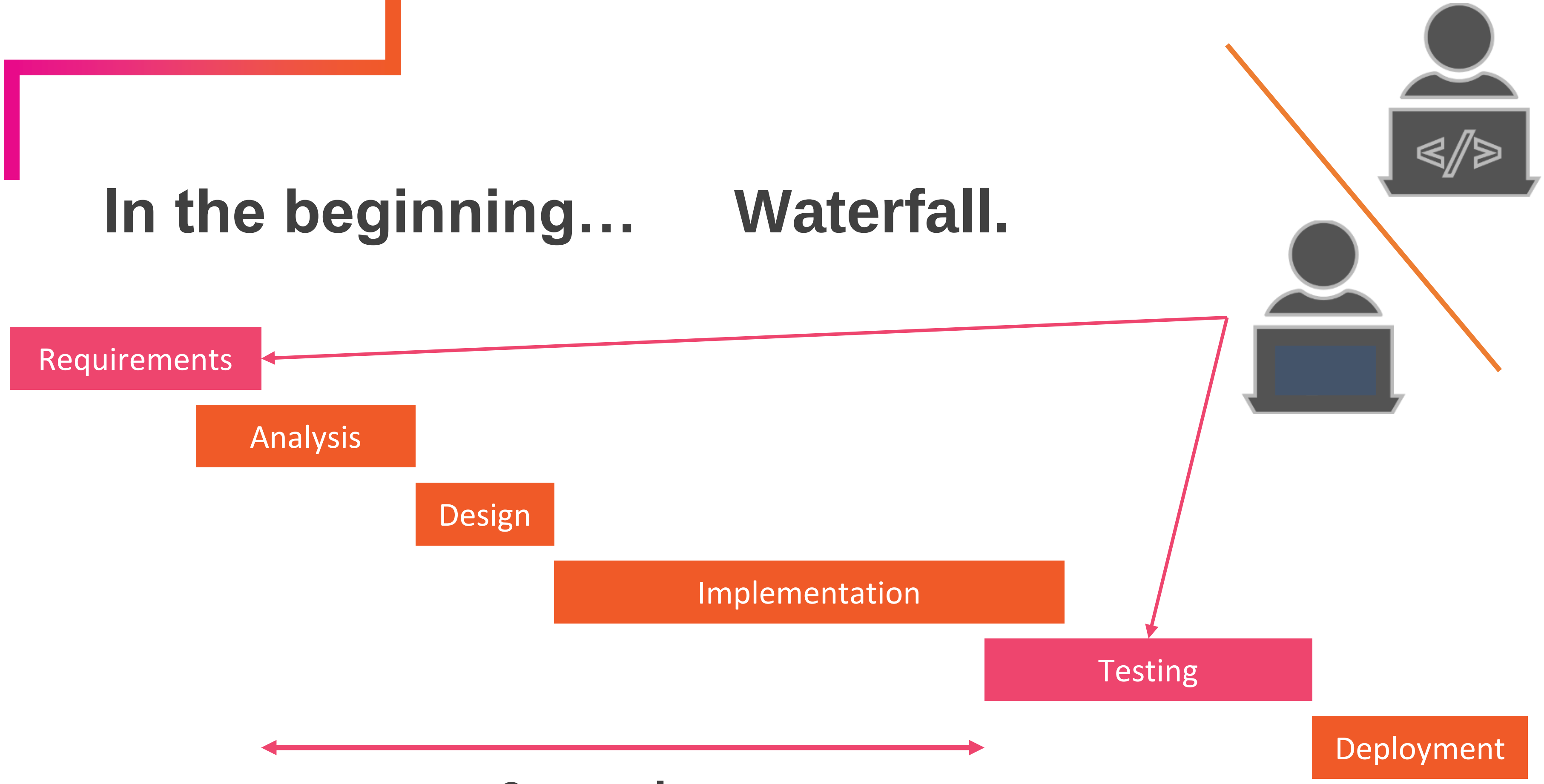
- **Waterfall**
- **Agile**
- **Rapid application development (RAD)**
- **Extreme programming (XP)**
- **Rational unified process (RUP)**
- **Scrum**

Waterfall Methodology

Waterfall methodology – A sequence of phases in which the output of each phase becomes the input for the next



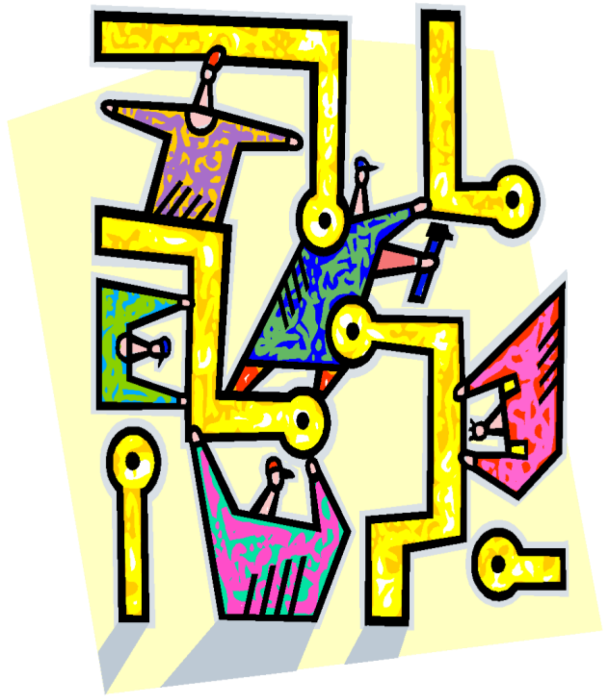
In the beginning… Waterfall.



6 months

Agile Methodology

- **Iterative development –** Consists of a series of tiny projects
- **Agile methodology –** Aims for customer satisfaction through early and continuous delivery of useful software components developed by an iterative process using the bare minimum requirements

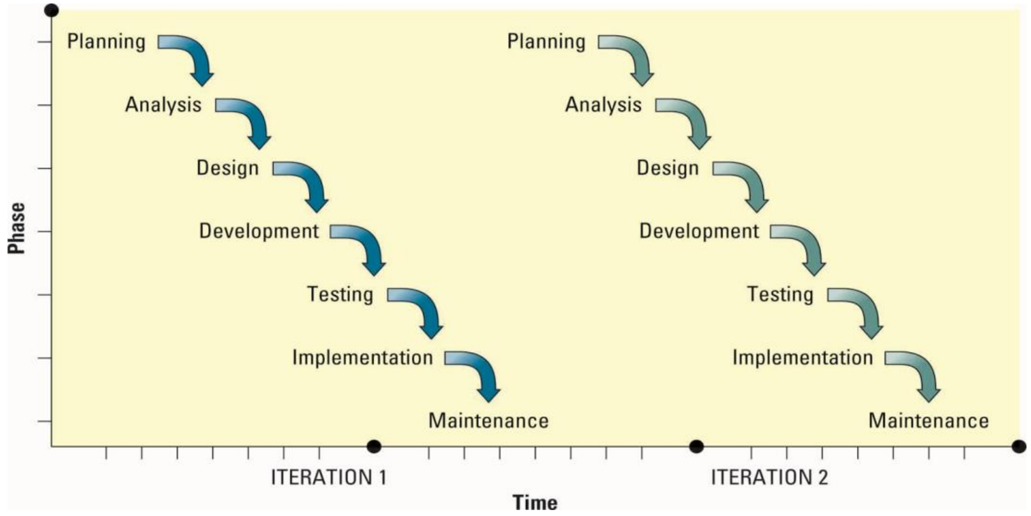


Rapid Application Development Methodology (RAD)

- **Rapid application development methodology–** Emphasizes extensive user involvement in the rapid and evolutionary construction of working prototypes of a system to accelerate the systems development process
- **Prototype –** A smaller-scale representation or working model of the users’ requirements or a proposed design for an information system
- The prototype is an essential part of the analysis phase when using a RAD methodology

Extreme Programming Methodology

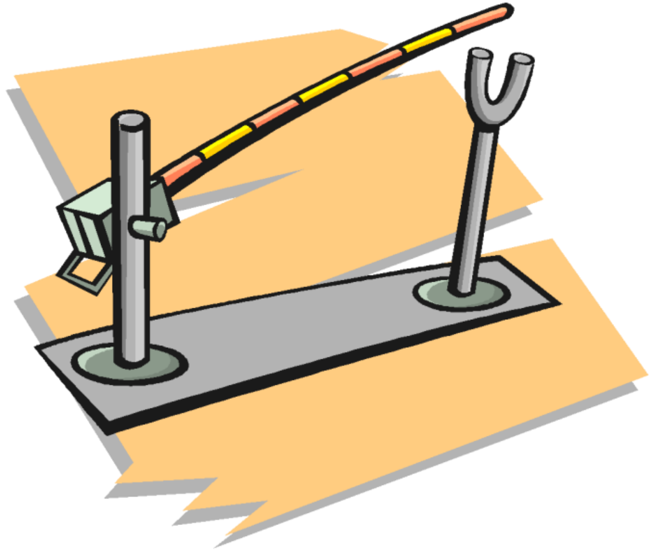
Extreme programming (XP) methodology – Breaks a project into tiny phases, and developers cannot continue on to the next phase until the first phase is complete



Rational Unified Process (RUP) Methodology

Rational unified process (RUP) – Provides a framework for breaking down the development of software into four gates

- Gate one: inception
- Gate two: elaboration
- Gate three: construction
- Gate four: transition



Agile Scrum

Emphasis on small, semi-independent teams ideally delivering discrete pieces of a system. Team ideally has total responsibility for the components it produces.

- **Team**
  - Small, cross-functional, self-organizing units
- **Scope**
  - Small deliverable scope delivered in consensus priority order
  - Priorities can be adjusted (typically at sprint start)
- **Timeline**
  - Small iterations (2-3 weeks is typical) emphasizing delivery at the end

SCRUM Methodology

- Scrum – Uses small teams to produce small pieces of deliverable software using sprints, or 30-day intervals, to achieve an appointed goal
- Under this methodology, each day ends or begins with a stand-up meeting to monitor and control the development effort

Minimizing waste and maximize feedback

Waste

❝ *The difference between what I think is valuable and what is really valuable creates waste*

Paul the master gardener, has a intuitive sense of what needs to be done next. He knows in his bones what matters and what doesn’t.

I might think perfectly straight rows are really important. I put a lot of effort into making my rows straight.

Along comes Paul and says "*Why are you working so hard at making the rows straight? What you need is more compost.*"

Software development is full of waste of overproduction

- Fat requirement documents that rapidly grow obsolete
- Elaborate architectures that are never used
- Code that goes months without being integrated, tested and executed in a production environment
- Documentation no one reads until it is irrelevant or misleading

While all these activities are important to software development, we need to use their output immediately in order to get the feedback we need to eliminate waste

The Toyota Story

Controlling waste

Taiichi Ohno

- Industrial engineer with Toyota
- Post WWII reconstruction
- Goal of competing with US auto industry
  - Limited resources
  - Focus on controlling what could be controlled
  - Believed in the potential of the team

"Progress cannot be generated when we are satisfied with existing solutions."
- Taiichi Ohno

`/reads/toyota`

Activity: Toyota story

1. How is the waste controlled in Toyota?
2. List some of the advantages to the organization by controlling waste.
3. Suggest some of ways how waste can be controlled in software development

Thank You