SDLC

Software Development Life Cycle

Presented by – Manohar Prasad

Agenda:

- Introduction
- SDLC Overview
- SDLC Phases
- SDLC Flow
- SDLC Model
- Waterfall Model
- Agile Model
- Conclusion

Introduction:

- The SDLC is a framework that describes the activities performed at each stage of a software development project.

- SDLC process is used by the software industry to design, develop and test high quality software. It aims to produce the quality software that meets or exceeds customer expectations, reaches completion within time and budget.

- ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

- Software Engineering Process Technology Company, (SEPT) is a firm specializing in meeting the software process standards information needs of the professional community, particularly concerning ISO/IEC 12207.

  - International Electrotechnical Commission (**IEC**)
  - International Organization for Standardization (**ISO**)
  - For more info visit http://www.12207.com/

SDLC Phases:


Requirement Analysis
Design
Implementation
Testing
Evolution

SDLC Phases

1. Planning and Requirements Analysis
2. Defining Requirements
3. Designing the Software
4. Building or Developing the Software
5. Testing the Software
6. Deployment and Maintenance

1. Planning & Requirement Analysis

Requirement analysis is the most important and fundamental stage in SDLC.

It is performed by the senior members of the team with inputs from all the stakeholders and domain experts or SMEs in the industry.

Planning for the quality assurance requirements and identification of the risks associated with the project is also done at this stage.

Requirements Analysis

- Business Requirements
- Stakeholder Requirements
- Solution Requirements
  - Functional Requirements
  - Non-functional Requirements
- Transition Requirements

2. Defining Requirements

Once the requirement analysis is done the next step is to clearly define and document the software requirements and get them approved from the project stakeholders.

This is done through ‘SRS’ – Software Requirement Specification document which consists of all the product requirements to be designed and developed during the project life cycle.

Defining Requirements

- Enterprise Analysis
- Business Analysis Planning & Monitoring
- Elicitation
- Requirements Analysis
- Requirements Management & Communication
- Solution Assessment & Validation

3. Designing the Software

- Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification.
- This DDS is reviewed by all the stakeholders and based on various parameters as risk assessment, design modularity , budget and time constraints , the best design approach is selected for the software.

4. Developing the Software

- In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage.

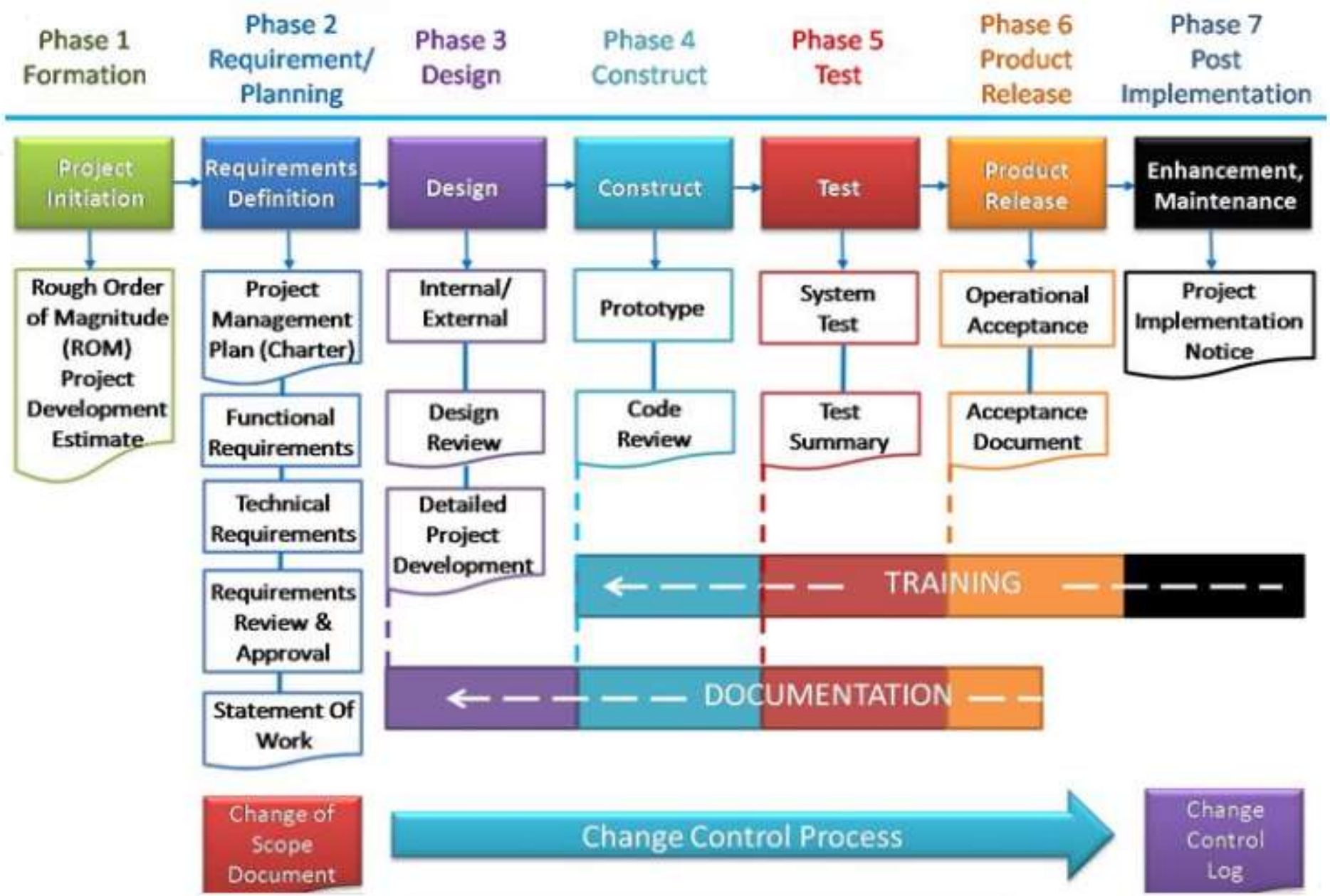
- Developers have to follow the coding guidelines defined by their organization and programming tools like compilers, interpreters, debuggers etc are used to generate and implement the code.

5. Testing the Software

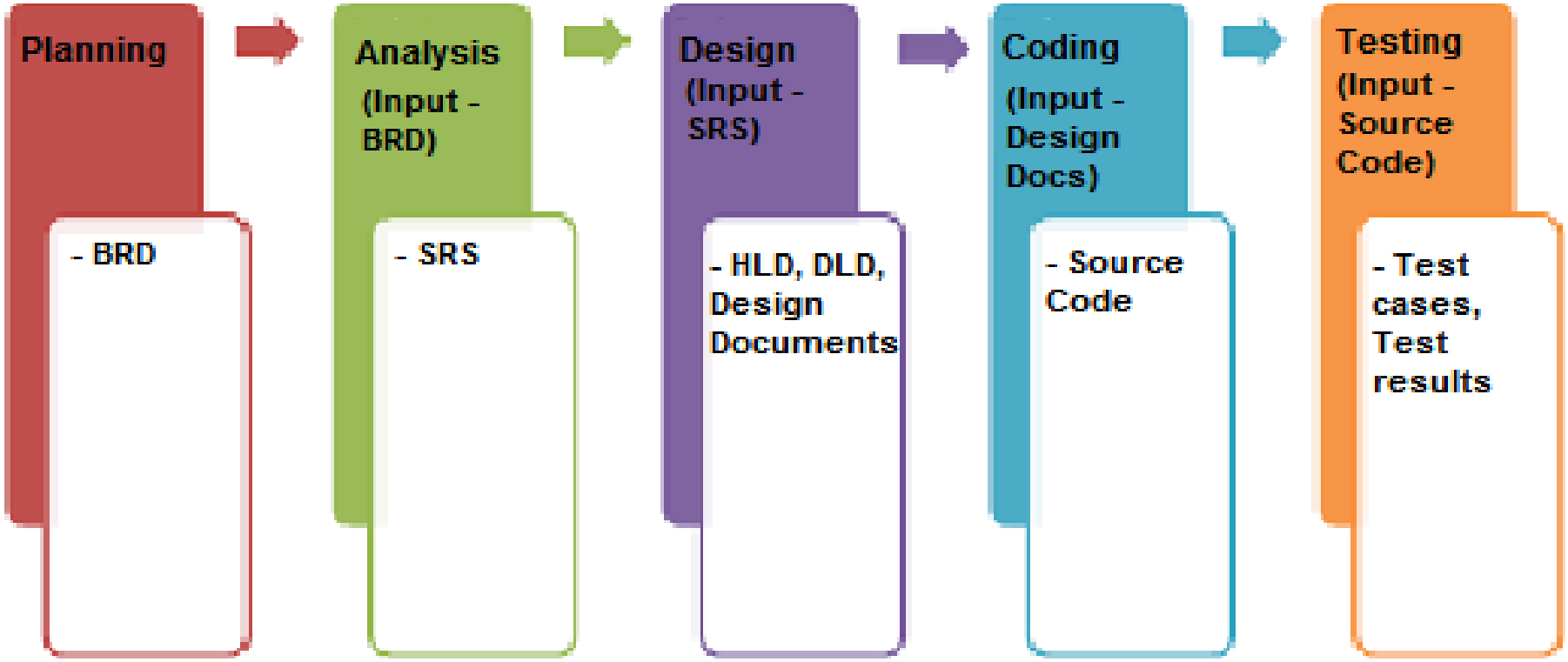
- This stage is usually a subset of all the stages as in the modern SDLC models, the testing activities are mostly involved in all the stages of SDLC.
- However this stage refers to the testing only that stage of the software where defects are reported, tracked, fixed and retested, until the software reaches the quality standards defined in the SRS.

6. Deployment and Maintenance

- Once the software is tested and no bugs or errors are reported then it is deployed.
- Then based on the feedback, the software may be released as it is or with suggested enhancements in the target segment.
- After the software is deployed then its maintenance starts.


Phase 1 Formation
Phase 2 Requirement/ Planning
Phase 3 Design
Phase 4 Construct
Phase 5 Test
Phase 6 Product Release
Phase 7 Post Implementation
Project Initiation
Requirements Definition
Design
Construct
Test
Product Release
Enhancement, Maintenance
Rough Order of Magnitude (ROM) Project Development Estimate
Project Management Plan (Charter)
Functional Requirements
Technical Requirements
Requirements Review & Approval
Statement Of Work
Internal/ External
Design Review
Detailed Project Development
Prototype
Code Review
System Test
Test Summary
Operational Acceptance
Acceptance Document
Project Implementation Notice
TRAINING
DOCUMENTATION
Change of Scope Document
Change Control Process
Change Control Log

SDLC Stages & Documents



SDLC Models

To help understand and implement the SDLC phases various SDLC models have been created by software development experts, universities, and standards organizations.

Reasons for Using SDLC Models

- Provides the base for project planning, estimating & scheduling.

- Provides framework for standard set of terminologies, activities & deliverables.

- Provides mechanism for project tracking & control.

- Increases visibility of project progress to all stakeholders.

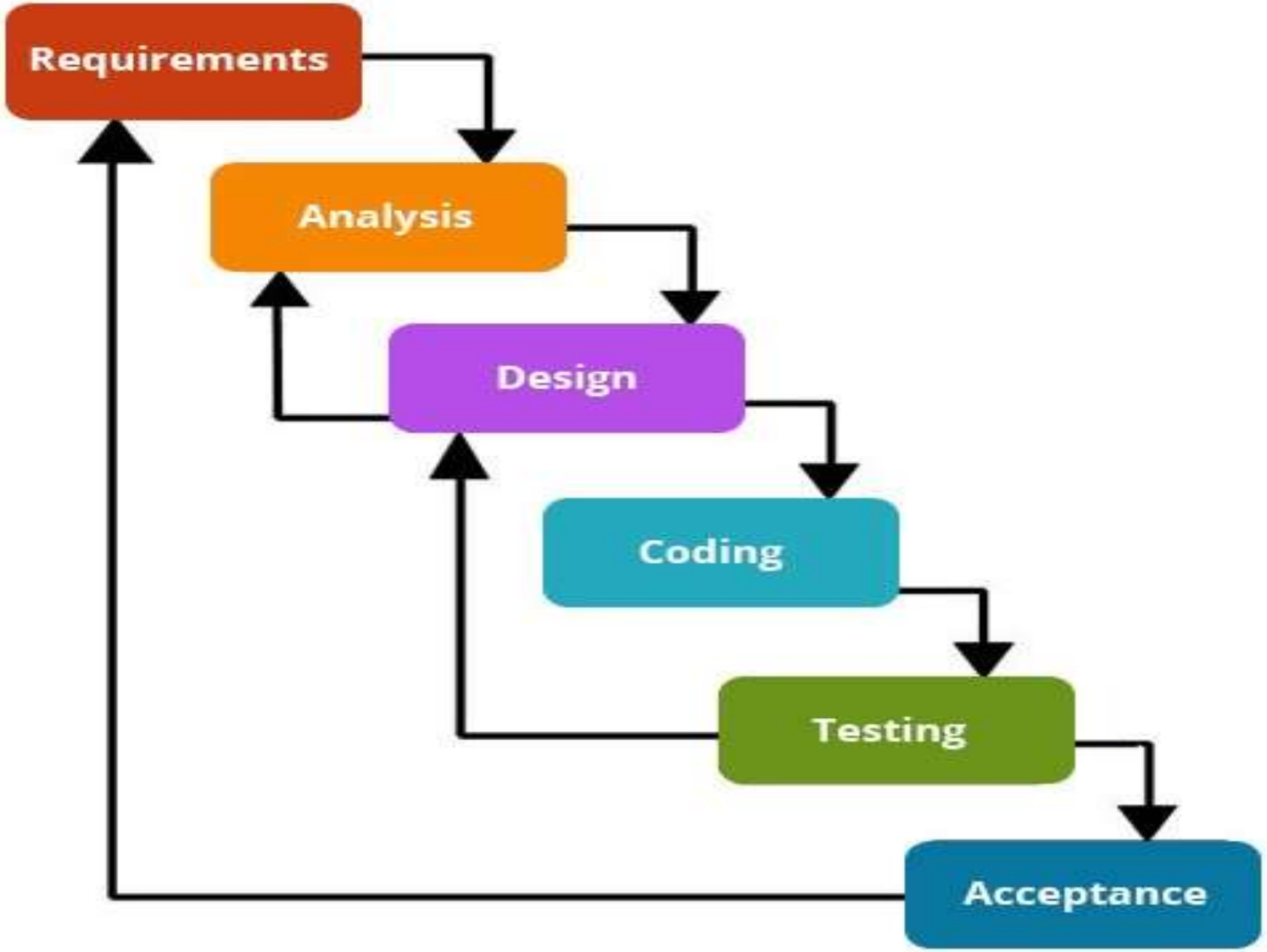
Advantages of Choosing an Appropriate SDLC

- Increased development speed
- Increased product quality
- Improved tracking & control
- Improved client relations
- Decreased project risk
- Decreased project management overhead

SDLC Models

- **Waterfall Model**
- Iterative Model
- Spiral Model
- **Agile Model**
- V – Model
- Big Bang Model

WATERFALL MODEL



Waterfall Model

- Oldest and most well-known SDLC model.
- Follows a sequential step-by-step process from requirements analysis to maintenance.
- Systems that have well-defined and understood requirements are a good fit for the Waterfall Model.

Waterfall Model: Strengths

- Easy to understand, easy to use
- Provides structure to inexperienced staff
- Milestones are well understood
- Sets requirements stability
- Good for management control (plan, staff, track)
- Works well when quality is more important than cost or schedule

Waterfall Model: Weaknesses

- All requirements must be fully specified upfront
- Deliverables created for each phase are considered frozen – inhibits flexibility
- Can give a false impression of progress
- Does not reflect problem-solving nature of software development – iterations of phases
- Integration is one big bang at the end
- Little opportunity for customer to preview the system (until it may be too late)

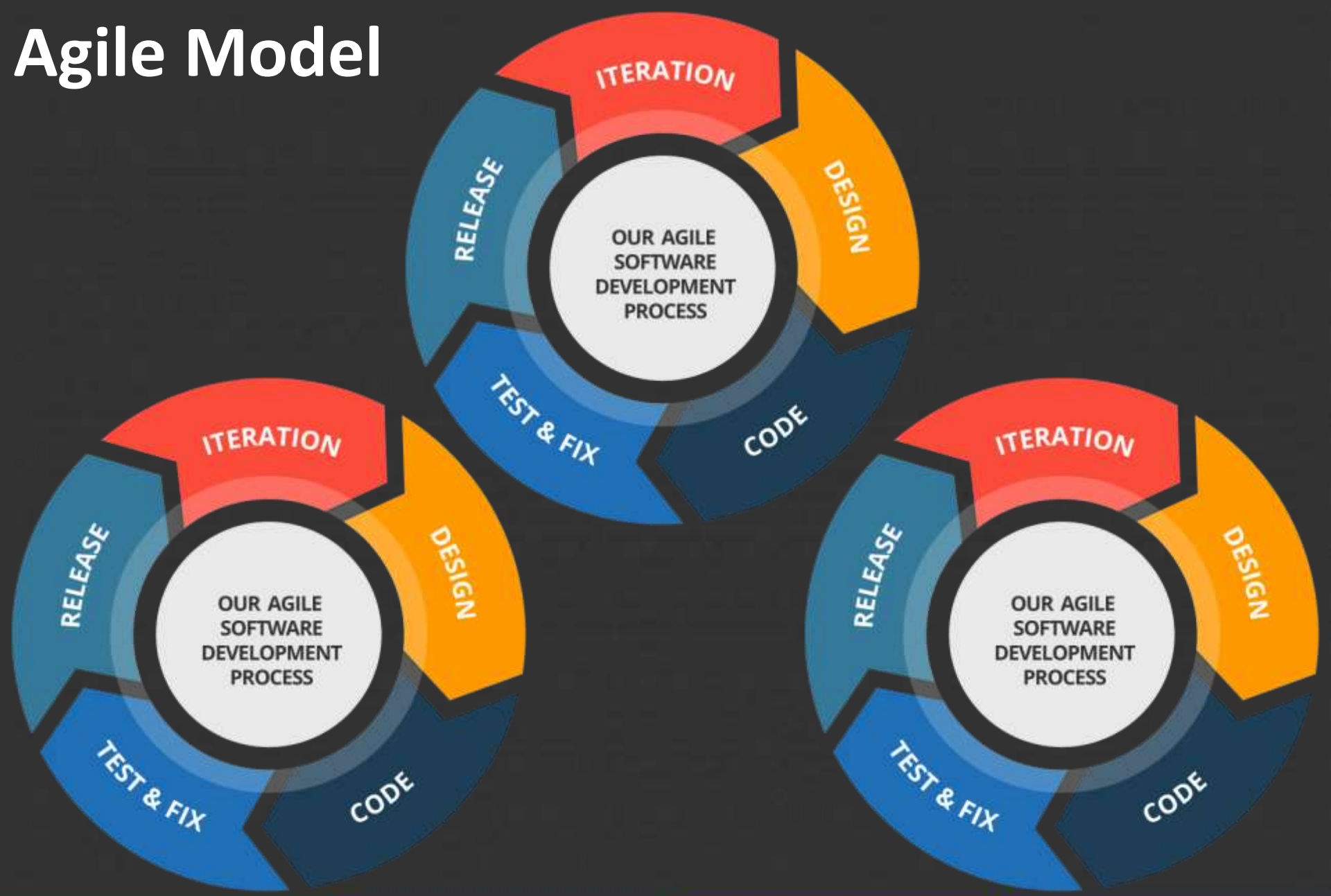
When to use the Waterfall Model

- Requirements are very well known
- Product definition is stable
- Technology is understood
- New version of an existing software/product
- Porting an existing product to a new platform

Agile Model

- Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements.
- In agile the tasks are divided to time boxes (small time frames) to deliver specific features for a release.
- Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer.

Agile Model


ITERATION
DESIGN
CODE
TEST & FIX
RELEASE
OUR AGILE SOFTWARE DEVELOPMENT PROCESS
ITERATION
DESIGN
CODE
TEST & FIX
RELEASE
OUR AGILE SOFTWARE DEVELOPMENT PROCESS
ITERATION
DESIGN
CODE
TEST & FIX
RELEASE
OUR AGILE SOFTWARE DEVELOPMENT PROCESS

Agile Model

- Speed up or bypass one or more life cycle phases
- Usually less formal and reduced scope
- Used for time-critical applications
- Used in organizations that employ disciplined methods

Some Agile Methods

- Rapid Application Development (RAD)
- **Scrum**
- Extreme Programming (XP)
- Adaptive Software Development (ASD)
- Feature Driven Development (FDD)
- Crystal Clear
- Dynamic Software Development Method (DSDM)
- Rational Unify Process (RUP)

Agile Model: Strengths

- Is a very realistic approach to software development.
- Promotes teamwork and cross training.
- Functionality can be developed rapidly and demonstrated.
- Resource requirements are minimum.
- Suitable for fixed or changing requirements.
- Delivers early partial working solutions.
- Little or no planning required.
- Easy to manage.

Agile Model: Weaknesses

- For larger projects, it is difficult to judge the efforts and the time required for the project in the SDLC.
- Since the requirements are ever changing, there is hardly any emphasis, which is laid on designing and documentation. Therefore, chances of the project going off the track easily are much more.
- Depends heavily on customer interaction, so if customer is not clear, team can be driven in the wrong direction.

Conclusion

- Project Management – Lead to an endeavor.
- Planning is a map, a guide, especially for a team. (Relatively simple and helpful techniques)
- SDLC is mostly about people, process, time management & communication.
- Risks are inevitable, planning helps to avoid stupid ones (Experience counts).
- Assessing the scope of work, timing, risks and resources will lead to the project success.

Quote of the Day……

"Risks are Essential in Achievements, Luck is an Element of Success"

Ask your Queries….

IIBA International Institute of Business Analysis Member™

International Institute of Business Analysis

Recognizes

MANOHAR PRASAD

As a member in good standing with IIBA®

From: 13 Apr 2016 to 30 Apr 2017

Kelly L. Frankhouser

Membership Program Manager
International Institute of Business Analysis

Member ID: 142938

IIBA.org

Thank you!

Presented by: Manohar Prasad

Sr. Business Analyst

Ask you queries!

15700+ Followers

1700+ Followers

14600+ Followers

1000+ Followers