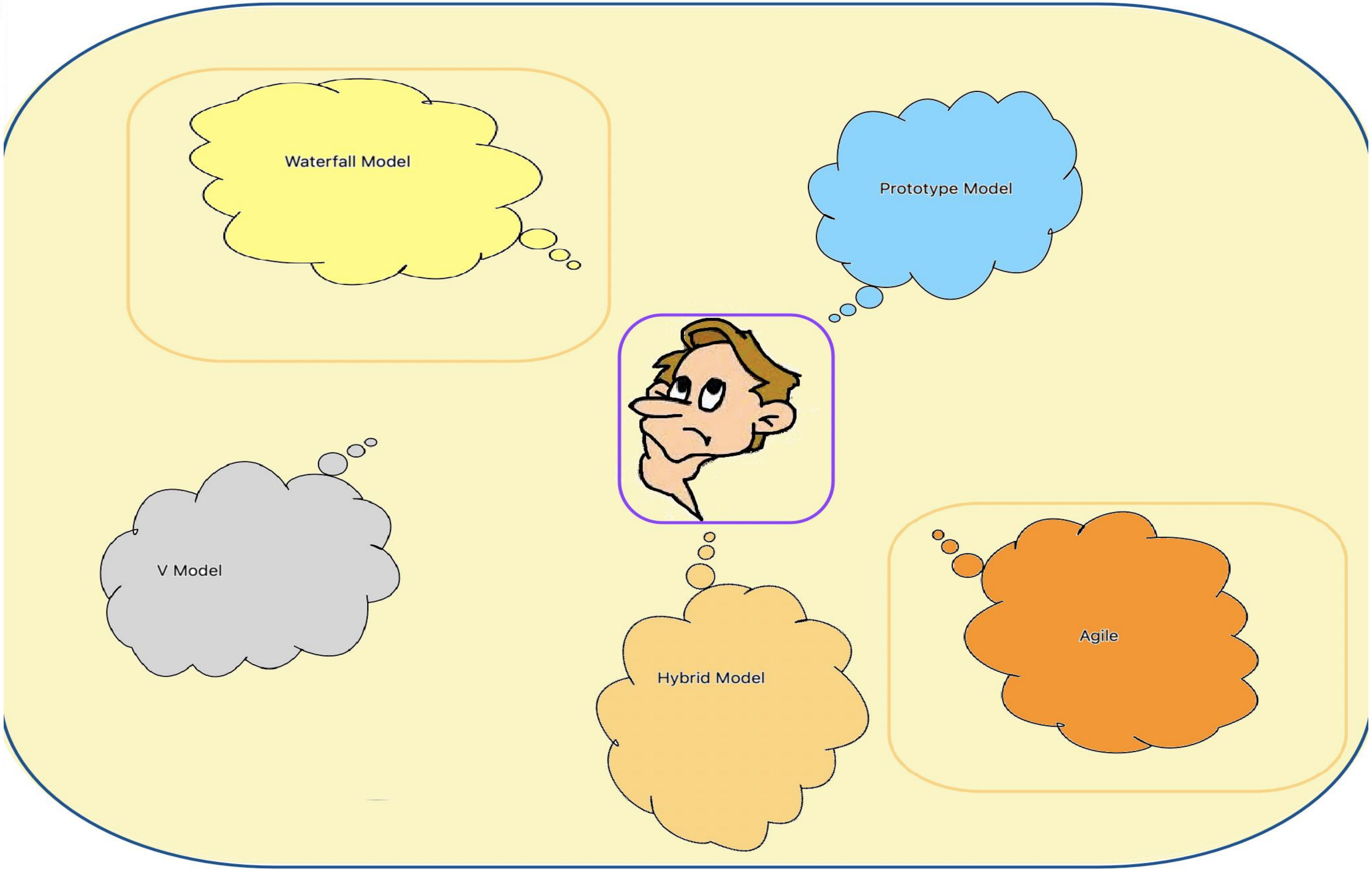
TESTERS ZONE
Learn before earn

MODELS IN SOFTWARE TESTING

By Mithilesh Singh


Waterfall Model
Prototype Model
V Model
Hybrid Model
Agile

TYPES OF MODEL

Waterfall model

Spiral model

Prototype model

V model

Hybrid model

Agile

WHY WE USE THESE MODELS?

- SDLC model gives the proper flow or better said roadmap for developing and testing the software. It's a kind of framework that says when to do what to do and how to do.This is designed based on requirement types, kinds of requirements,and challenges that we were facing to achieve a quality product. Client needs to come up with their requirements and budget and we can use specific model to fulfill that.

WATERFALL MODEL

Waterfall Model is the first model introduced in Software Development Life Cycle.

This model is similar to the water flows direction from top to bottom and can't go back in reverse direction to the previous phase.

Waterfall Model is very simple to understand, use and has distinct endpoints or goals at each phase.

In the waterfall model, the current phase should be completed to move to the next phase and current phase input is the output of the previous phase.

The waterfall model is a sequential design process through the phases of Requirements, Feasibility Analysis, Design, Coding, Testing, Deployment, and Maintenance.

WHY THIS MODEL?

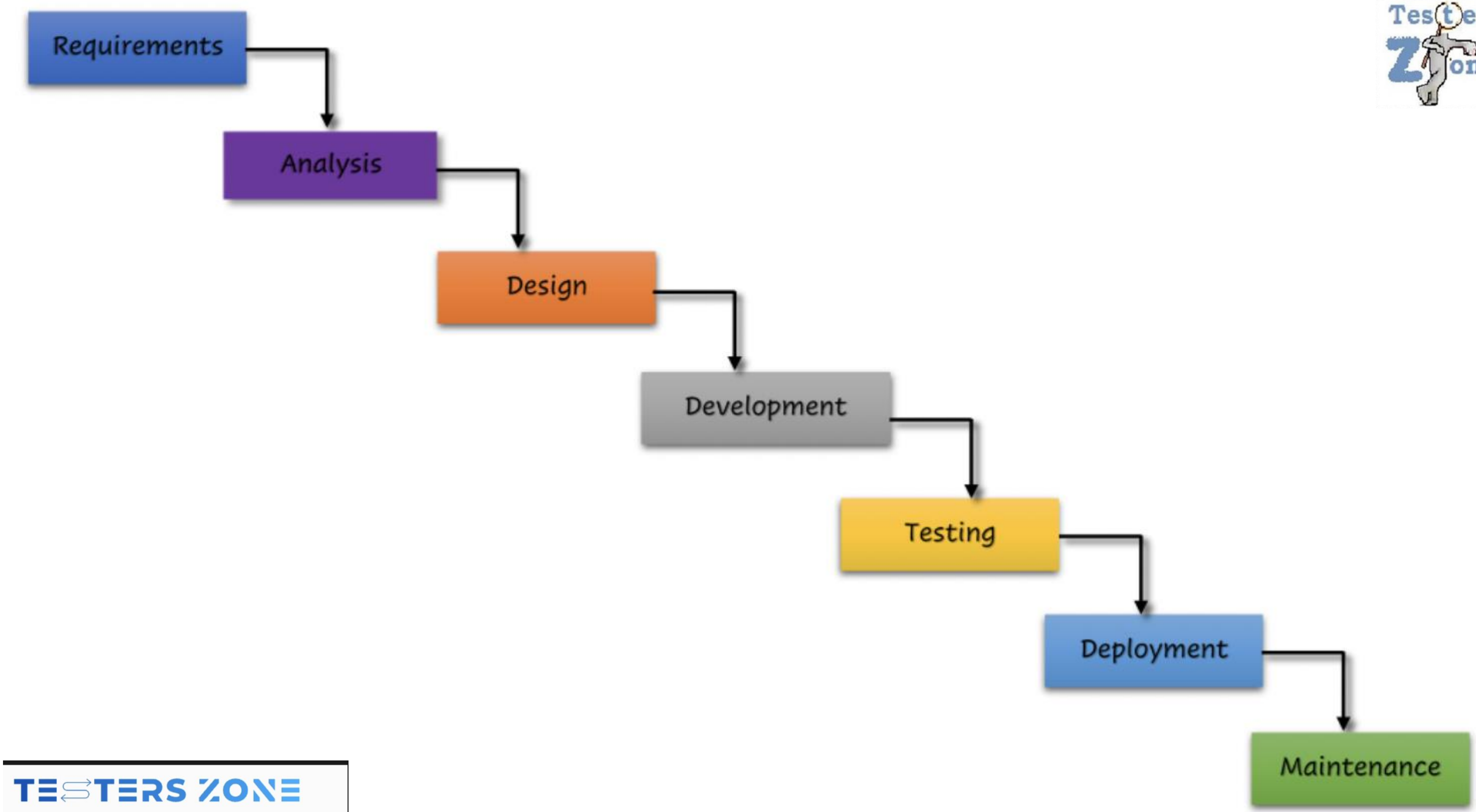
- This was very simple and first model introduced and there was no any other model at that time and also it fits to the any size(small, medium or large) project that why it got popularity in short span of time.

WHEN TO USE WATERFALL MODEL?

- Requirement is clear and receive in the starting of the project. Thers is no any in between requirement changes.
- Client is interested with end product and don't want to pitch into in between development.
- Having clear idea on what is the end product.

DIFFERENT PHASES:

- Requirement Collection
- Analysis
- Design
- Development
- Testing
- deployment



Requirements

Analysis

Design

Development

Testing

Deployment

Maintenance

REQUIREMENT COLLECTION

- **Requirement Gathering** is the most important and first phase of water fall model. In this phase, business analyst and project manager participate in the meetings with client to gather the requirements.
- Client: someone who will provide the requirement and tell what to build, who will use it or purpose of it to all the participants.
- Business Analyst: Person who gathers the requirement from the client.
- Project Manager : who handles the project
- **Who involves -** Client, Business Analyst, Project Manager and all Stake holders.

ANALYSIS PHASE

- Once the requirements gathering phase completed, Business Analyst (BA) converts the business requirements into technical requirements with the help senior team members like SMEs (Subject-matter experts) and team leads. The technical requirements documented in a document called SRS (Software Requirement Specification) document. SRS consists of all the technical requirements for all system involved (that includes cross-applications if exists)
- SRS document name varies from project to project.
  SRS (Software Requirement Specification) or Requirement Specification Document (RSD) or Project Specification Document (PSD).

DESIGN

- In this phase designer converts SRS into DDS(Design Document Specification). overall system architecture is designed by defining their functionality of each module that includes their interaction with cross-systems.

- The Design Document Specifications that are two types based on their level of design. Those are -
  High-level Design (HLD) - HLD contains the high-level architecture changes or design changes.
  Low-level Design (LLD) - LLD contains very detailed level architecture or design changes.

- Both HLD and LLD creates by **Senior developers or architects** creates HLD.
- **Note**: The architects usually provide more than one approach in a Design Document Specification. This DDS sent for review to all the stakeholders and the best approach would be selected based on the features such as robustness, budget and timing constraints.

CODING/DEVELOPMENT/IMPLEMENTATION

- Once Design is approved implementation gets started with code following organization coding standards, Once coding for small component is done developer use to compile and debug the developed code using different tools.
- Coding or Development or Implementation all are having same meaning and purpose. Implementation means developers are implementing the source code as per DDS, since the implementation helps to develop the piece of product/software this is also called as Development. Development needs some programming language understanding to develop the product, since development means nothing but piece of code using some programming language and that is coding in technical term so that why we call this phase as Coding also.

TESTING PHASE

- **Note**: This phase helps to understand whether implemented code is as per client expectation or not, Once the coding completed The developer performs the initial testing that are unit testing (UT) and/or Application Integration Testing (AIT) before handover the code to the testing team.
- Testers do lot of testing like Functional testing, system testing, acceptance testing etc and verify the BRS(Business requirement specification). If development is not as per BRS, testers raise or log the defect tagging to developer and dev needs to fix it and share the updated code with tester in form of build and then tester will do retesting. After retesting defect might reopen or close. This entire testing and bug logging activity follows certain process that is called as Software testing life cycle or Software testing process.

DEPLOYMENT PHASE

- Once the product is tested well and that is less buggy then client will review the test results and artifacts and then approve it for deployment
- Once the software got deployed to production, then the new functionality available to the end-users who are currently using the system.

MAINTENANCE PHASE

- Once the end-user starts using the newly deployed software, there might be a possibility that the real-time issues starts coming up. The team has to fix these issues to avoid the loss in business. if the issue is having less priority and less impact or high priority and has huge impact, client can take a decision to roll out or backout new changes and refine the functionalities as required. This process of taking care for the finished product is called as maintenance.

ADVANTAGES OF WATERFALL

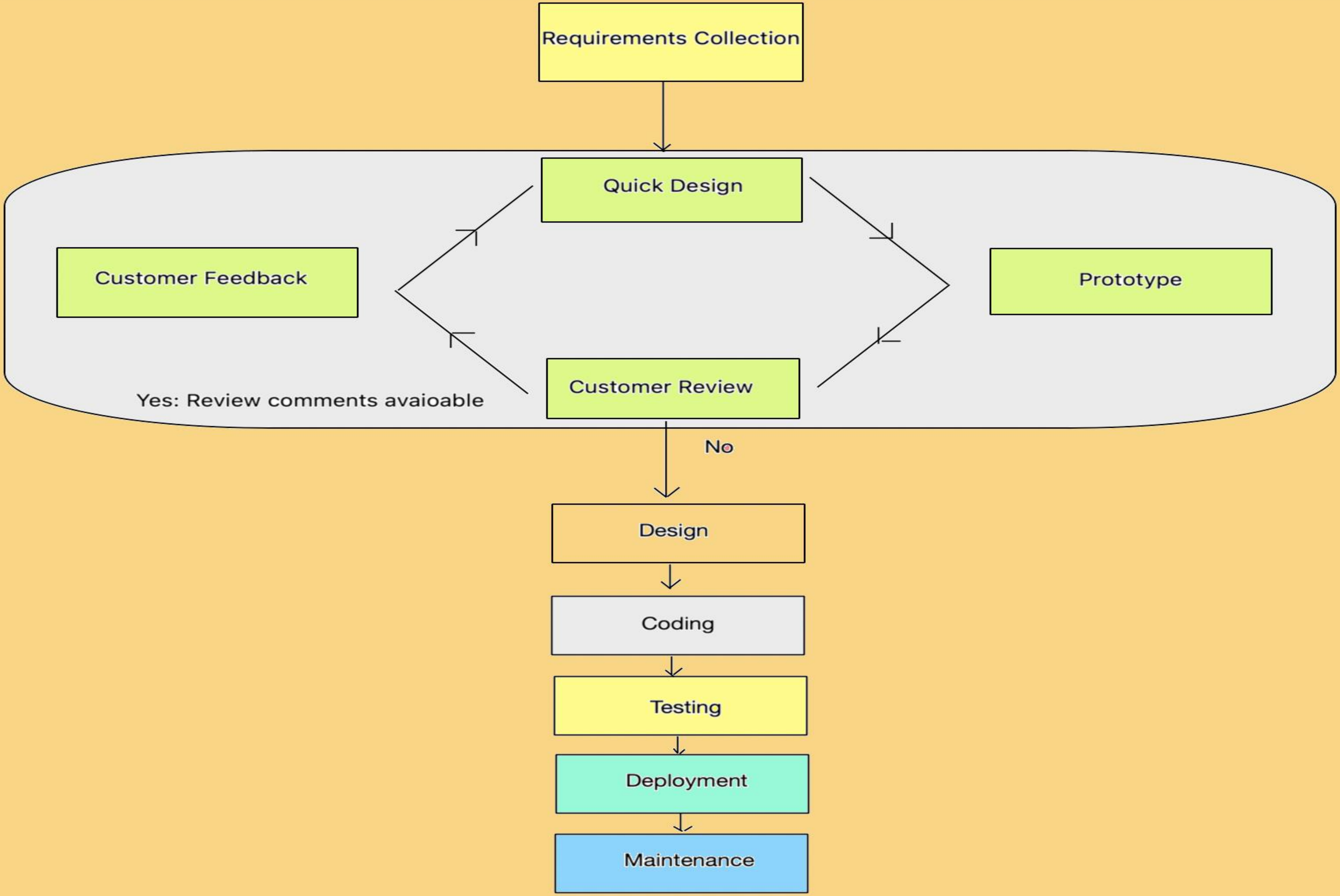
- Simple and easy to understand
- Requirements are clear at the beginning of the project.
- Easy to manage
- Each phase is done one at a time so No overlapping of the phases as it is a step by step process and it takes long delivery time
- Client intervention is hardly required.
- Best suitable for small projects as the requirements are clearly stated.

DISADVANTAGES

- Not good for complex and long on-going projects.
- Less client involvement hence No feedback path until the end of the project.
- Testing phase will come late during the development process.
- No intermediate deliveries until the deployment of full product. End-users has to wait until the deployment even it is urgent requirement.

PROTOTYPE MODEL

- A biggest drawback in the Water fall model was client interaction was very less and if there is any gap in product development and not as per client requirement then it increases development cost. To overcome this one new model came in SDLC and that is Prototype model.
- The basic idea of prototype model is that the prototype gets reviewed and approved by the client
- Using this prototype, client can get the "real behavior" of the developing product. Client provides his valuable feedback once he reviews the prototype. This feedback helps the developers and designers work on the prototype to incorporate feedback comments. Once the design modified, again sends for client review. This process will continue until no review comments from client or gets a client signoff to proceed with design and development.



Requirements Collection
Quick Design
Customer Feedback
Prototype
Customer Review
Yes: Review comments avaioable
No
Design
Coding
Testing
Deployment
Maintenance

DIFFERENT PHASES

Requirement Collection --> slide no 9

Quick Design

Prototype development

Customer Review

Customer Feedback

Analysis --> slide no 10

Design --> slide no 11

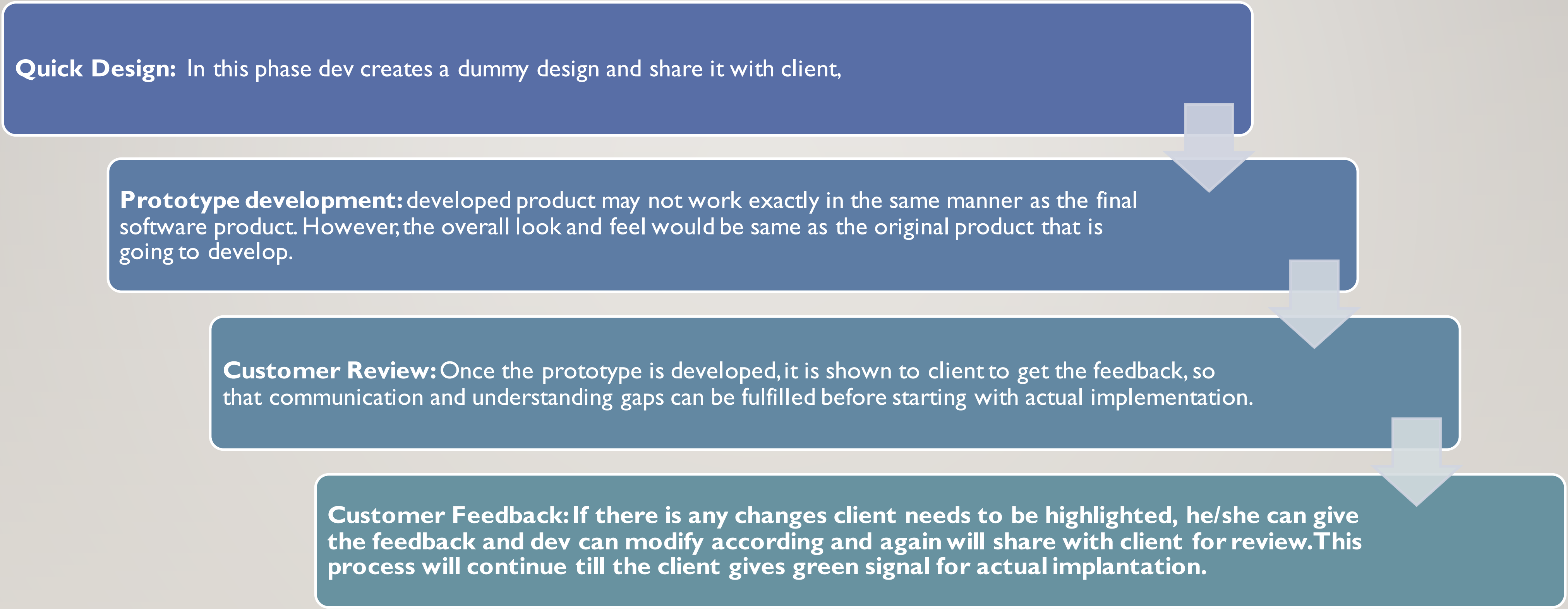
Development --> slide no 12

Testing --> slide no 13

Deployment --> slide no 14

Maintenance --> slide no 15

PHASES:



Quick Design: In this phase dev creates a dummy design and share it with client,

Prototype development: developed product may not work exactly in the same manner as the final software product. However, the overall look and feel would be same as the original product that is going to develop.

Customer Review: Once the prototype is developed, it is shown to client to get the feedback, so that communication and understanding gaps can be fulfilled before starting with actual implementation.

Customer Feedback: If there is any changes client needs to be highlighted, he/she can give the feedback and dev can modify according and again will share with client for review. This process will continue till the client gives green signal for actual implantation.

ADVANTAGES

- Easy to detect errors in the initial stage.
- Prototypes can be modified or removed.
- Lowers the risk.

DISADVANTAGES

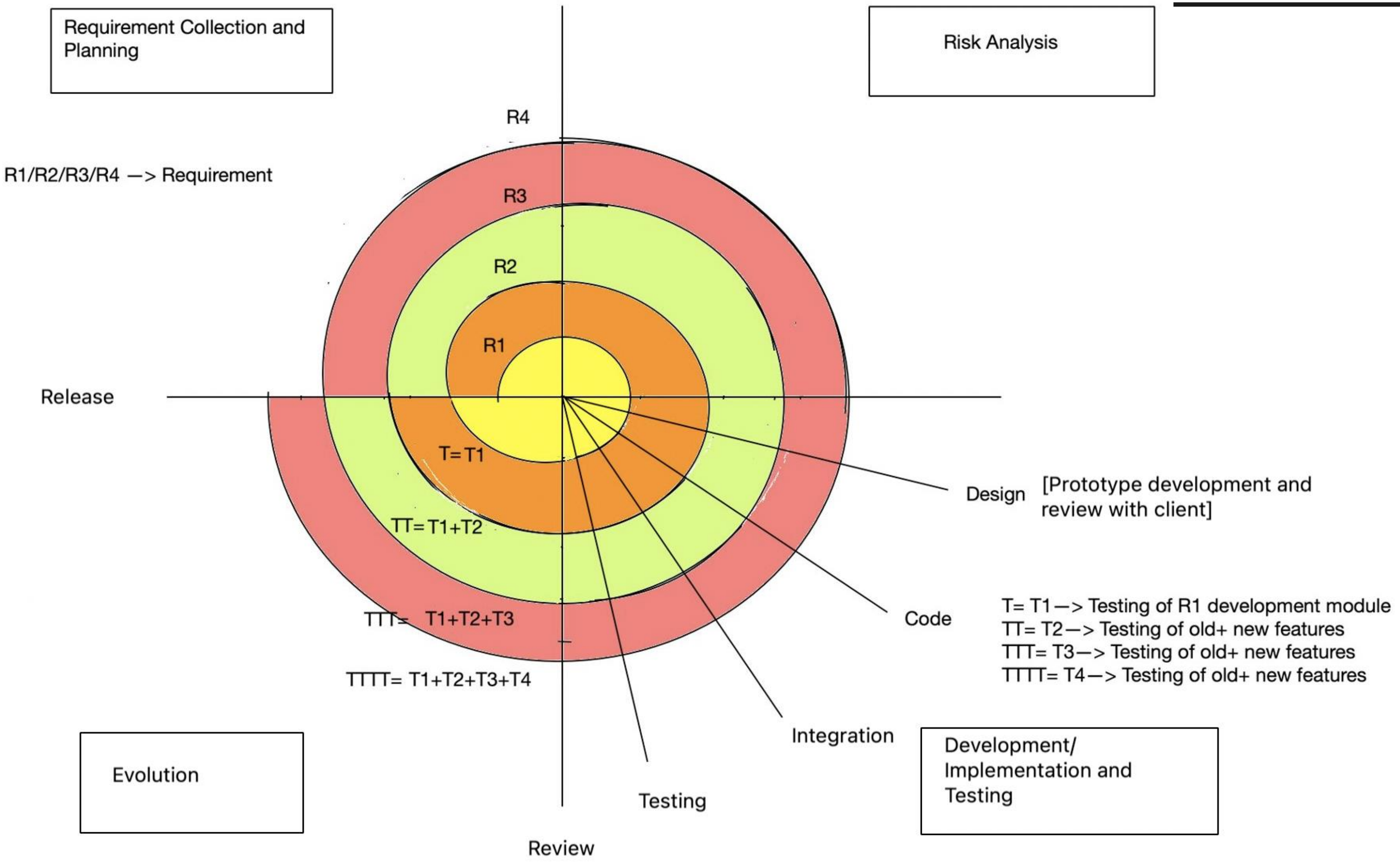
Slow and time-consuming process

Cost of design and developing prototype becoming waste if the prototype is rejected or doesn't met client expectations.

Large amount of change requests are accepted.

SPIRAL MODEL

- Spiral model is preferred for large and high-risk projects.
- Main agenda of this model is develop primary and important feature first since it is already defined and remaining features are added in spirals (iterations) of the project.
- The spiral model looks as a coil that has a starting point which call it as a "center" and the line that spirals shows the progress of the project development. The line that spirals outwards indicates the increase of time expenditure and effort by the developers and the project progress.
- Since it is a iterative model, in between changes are allowed
- It is divided into 4 phases Requirement collection, Risk Analysis, Development and Testing and Evolution.



Requirement Collection and Planning

Risk Analysis

R1/R2/R3/R4 —> Requirement

R4

R3

R2

R1

Release

T= T1

TT= T1+T2

TTT= T1+T2+T3

TTTT= T1+T2+T3+T4

Design [Prototype development and review with client]

Code

T= T1—> Testing of R1 development module
TT= T2—> Testing of old+ new features
TTT= T3—> Testing of old+ new features
TTTT= T4—> Testing of old+ new features

Integration

Testing

Review

Evolution

Development/ Implementation and Testing

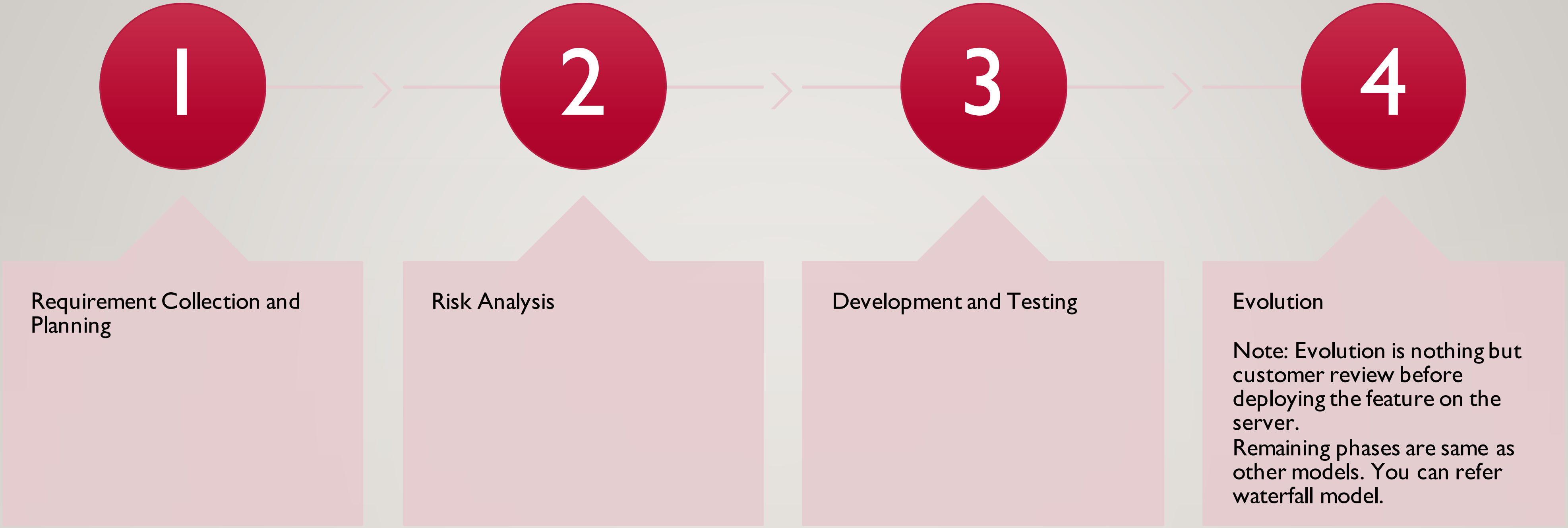
WHEN SHOULD WE GO WITH THIS MODEL?

When Requirement is complex and client is also not very much sure about the requirement.

Project duration is more.

In between changes are welcome.

SPIRAL MODEL PHASES:



1. Requirement Collection and Planning

2. Risk Analysis

3. Development and Testing

4. Evolution

Note: Evolution is nothing but customer review before deploying the feature on the server.
Remaining phases are same as other models. You can refer waterfall model.

ADVANTAGES

01

Customer Involvement is available hence customer would be more satisfied

02

Rist Handing Strategy we create in the beginning so product delivery would be fast

03

Fast development

04

It is a iterative and incremental Model so every release cycle will have integration feature testing with the older feature.

DISADVANTAGES

- High cost
- Chances to get clarity on product at the end will be maximum.
- It is suitable for long Project,
- Hard to manage
- Need more resources

V MODEL

- There was a biggest drawback in the waterfall model i.e. we can't identify bug in early stage of the cycle.And suppose we find major or critical defect in the closing phase then it's becoming challenging and costly to fix those defects. Because the project almost came to an end.To resolve this problem,there is a requirement to define new SDLC model.V-model defined as a solution to the above problem

ABOUT

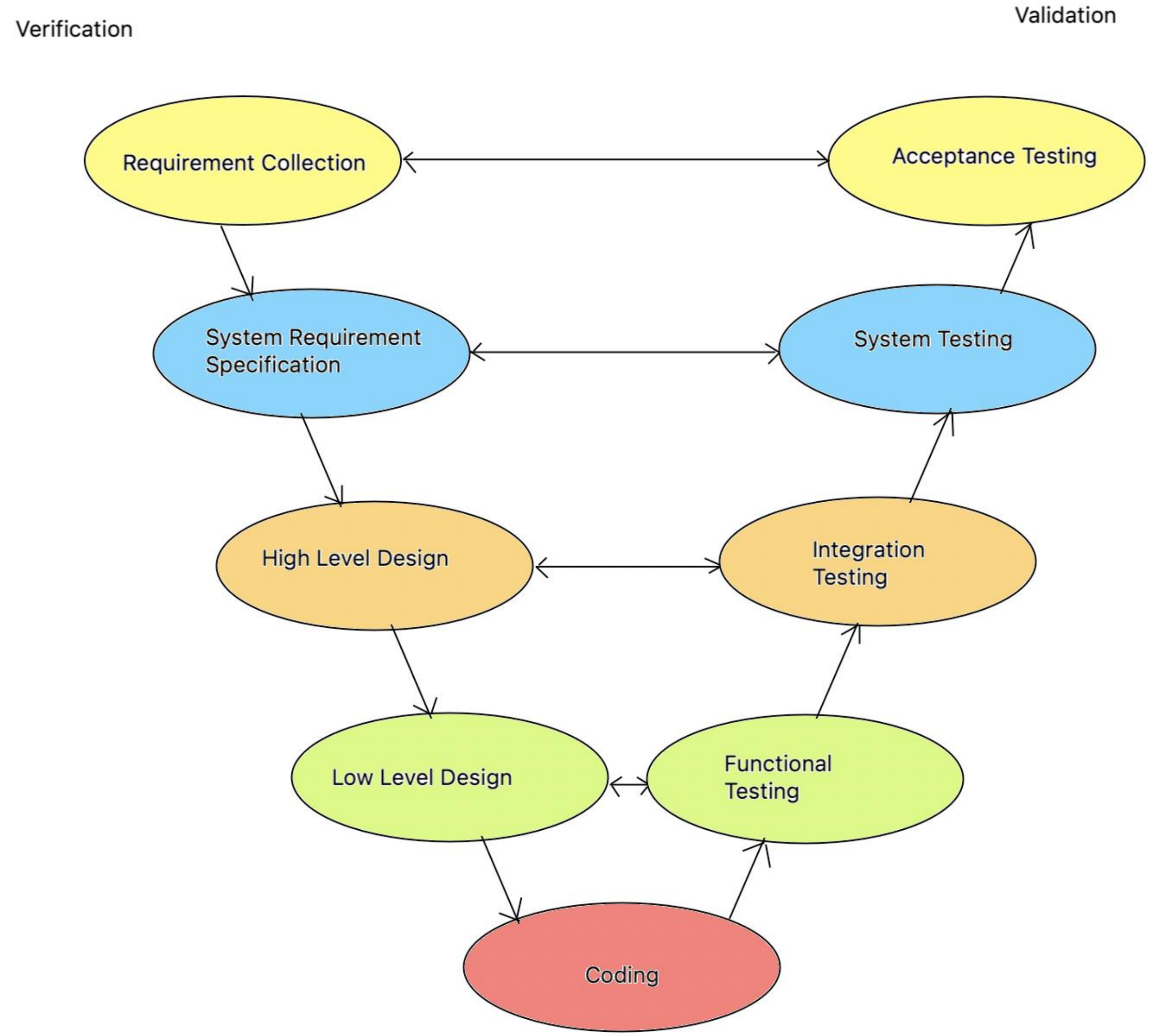
- Each phase of the development has its corresponding testing phase. Both development and testing activities performs together (parallelly).
- This model is also known as "Verification and Validation model".
- In this model testing use to perform at every stage but in between changes also not acceptable in this model too. Requirements are well defined, clear and documented in the Starting phase only.

PHASES

Verification - Involves in static analysis with code execution The evolution procedure carried out during the development to verify all the requirements covers in coding or not.

Validation - Involves dynamic analysis (both functional and non-functional) and testing is performed by code execution. The validation can be done once the development completed.

TESTERS ZONE
Learn before earn



V MODEL FLOW:

- One big disadvantages in the other models were, stage wise validation was missing and that is the biggest advantage of V model.
- Both verification and validation happens parallelly for each phase so chances of downstream of bug is very less.
- As soon as requirement collection will be completed, team will work on feasibility analysis. Feasibility analysis is kind of requirement acceptance testing from team to check all the bottle necks at the beginning stage.
- Once BA converts CRS to SRS(Software requirement Specification) end to end feature explanation and flow will be explained in the SRS so Testers can go through and develop the System testing Cases based on that.
- Once SRS is completed, Designer will take and create Hight level Design, design will also say module wise integration and data flow that will help Testers to develop integration testing cases.
- Later point designer will create Low level design which will be component level explanation using virtual design. LLD will help testers to develop component /functional testing cases.
- Once LLD is completed dev will pick design to develop real module using any programming language. That falls into coding stage.
- Once coding is done, testing cycle will be getting started. Testers will follow Functional, Integration and System test cases for different types of testing and finally user acceptance testing will be performed from client end.

ADVANTAGES:

- Easy and Simple to understand.
- Avoid more defects
- Downward flow of defects can be reduced
- Development and progress is very organized and systematic.

DISADVANTAGES

In between changes are not allowed

No need to produce prototype.

No software is produced at the intermediate phases and only produces at the end.

AGILE METHODOLOGY

Agile model/Methodology helps to handle in between requirement changes of client and deliver product to client on time. There are many advantages of this model that will be explaining in further slides.

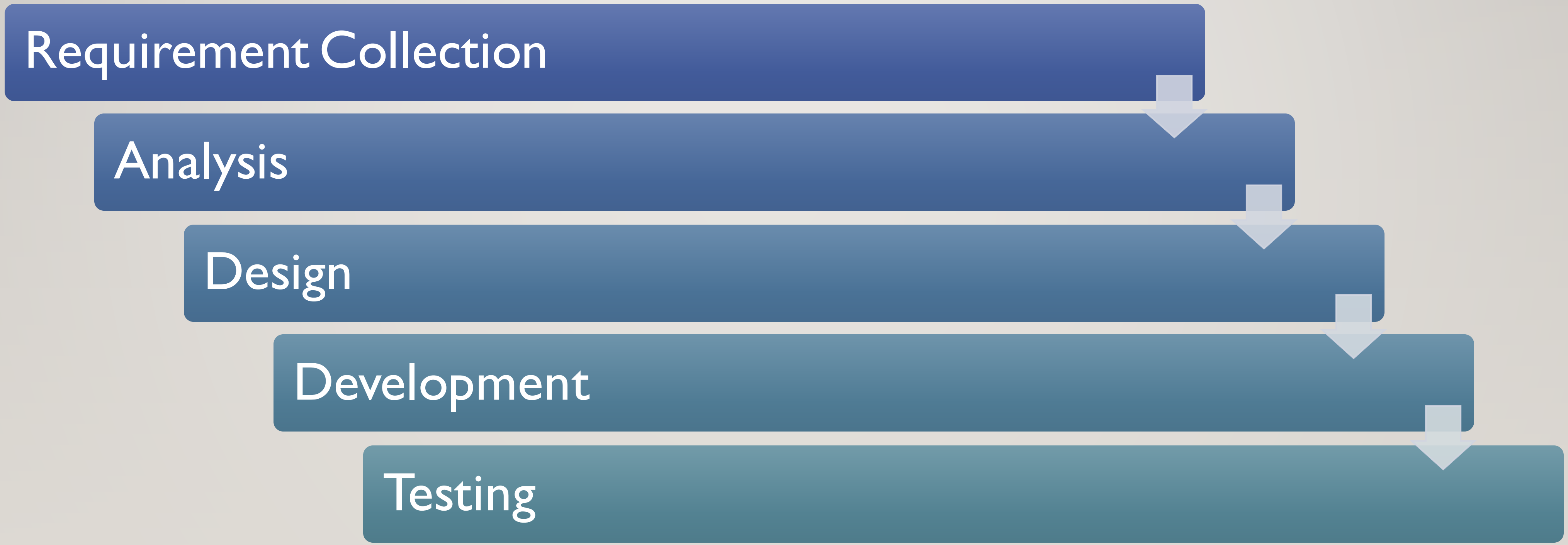
Agile Model is a people-focused, results-focused continuous iterative process. It involves the development and testing activities together throughout the Software Development Life Cycle (SDLC). It is a combination of both Iterative and spiral model.

In Agile, every project is divided into releases and each release is further divided into small time frames called iterations/sprints. Each iteration lasts for 2-4 weeks.

WHY AGILE ?

- Gathering requirement at once and analyzing takes more time, instead in agile we break the requirements in the chunks and that can be shared with BA in some time interval. This helps team to Analyse the part of requirements at a time and start will other process quickly.
- Customer interaction increases in this SDLC model hence we can deliver more quality product to client at the end.
- The traditional process is too slow. There is no parallel work progress in the traditional approach. Due to this, client has to wait more time to complete all phases to deploy in production. This might costs a profit sometimes.

PHASES:



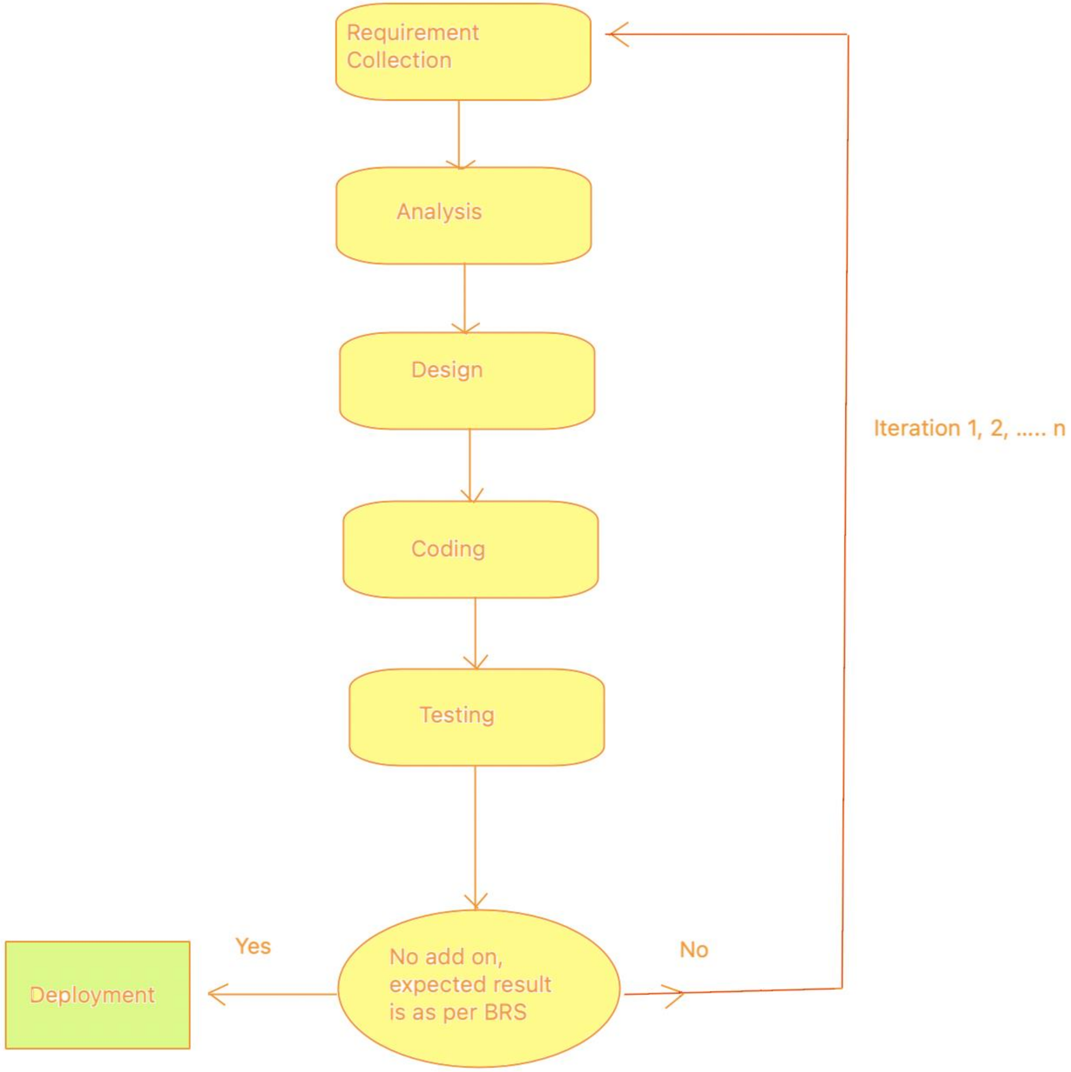
Requirement Collection

Analysis

Design

Development

Testing



TESTERS ZONE

Learn before earn

REQUIREMENT COLLECTION

Requirement Gathering is the most important and first phase of this model. In this phase, business analyst and project manager participate in the meetings with client to gather the requirements.

Client: someone who will provide the requirement and tell what to build, who will use it or purpose of it to all the participants.

Business Analyst: Person who gathers the requirement from the client.

Project Manager : who handles the project

Who involves - Client, Business Analyst, Project Manager and all Stake holders.

ANALYSIS

- Once the requirements gathering phase completed, Business Analyst (BA) converts the business requirements into technical requirements with the help senior team members like SMEs (Subject-matter experts) and team leads. The technical requirements documented in a document called SRS (Software Requirement Specification) document. SRS consists of all the technical requirements for all system involved (that includes cross-applications if exists)
- SRS document name varies from project to project.
  SRS (Software Requirement Specification) or Requirement Specification Document (RSD) or Project Specification Document (PSD).

DESIGN

- In this phase designer converts SRS into DDS(Design Document Specification). overall system architecture is designed by defining their functionality of each module that includes their interaction with cross-systems.

- The Design Document Specifications that are two types based on their level of design. Those are -
  High-level Design (HLD) - HLD contains the high-level architecture changes or design changes.
  Low-level Design (LLD) - LLD contains very detailed level architecture or design changes.

- Both HLD and LLD creates by **Senior developers or architects** creates HLD.

- **Note**: The architects usually provide more than one approach in a Design Document Specification. This DDS sent for review to all the stakeholders and the best approach would be selected based on the features such as robustness, budget and timing constraints.

CODING

- Once Design is approved implementation gets started with code, followed by organization coding standards, Once coding for small component is done developer use to compile and debug the developed code using different tools.
- Coding or Development or Implementation all are having same meaning and purpose.

  Implementation means developers are implementing the source code as per DDS, since the implementation helps to develop the piece of product/software this is also called as Development. Development needs some programming language understanding to develop the product, since development means nothing but piece of code using some programming language and that is coding in technical term so that why we call this phase as Coding also.

TESTING

Note:This phase helps to understand whether implemented code is as per client expectation or not, Once the coding completed The developer performs the initial testing that are unit testing (UT) and/or Application Integration Testing (AIT) before handover the code to the testing team.

Testers do lot of testing like Functional testing, system testing, acceptance testing etc and verify the BRS(Business requirement specification). If development is not as per BRS, testers raise or log the defect tagging to developer and dev needs to fix it and share the updated code with tester in form of build and then tester will do retesting. After retesting defect might reopen or close. This entire testing and bug logging activity follows certain process that is called as Software testing life cycle or Software testing process.

DEPLOYMENT

Once the product is tested well and that is less buggy then client will review the test results and artifacts and then approve it for deployment

Once the software got deployed to production, then the new functionality availabls to the end-users.

ADVANTAGES

- More customer satisfaction
- Welcome in between changes
- Continuous delivery of working product at the end of every iteration.
- Easily manageable.
- Less chances to downflow of bug hence Faster detection of issues and defects.
- Not required to get ready the whole requirements before starting project.

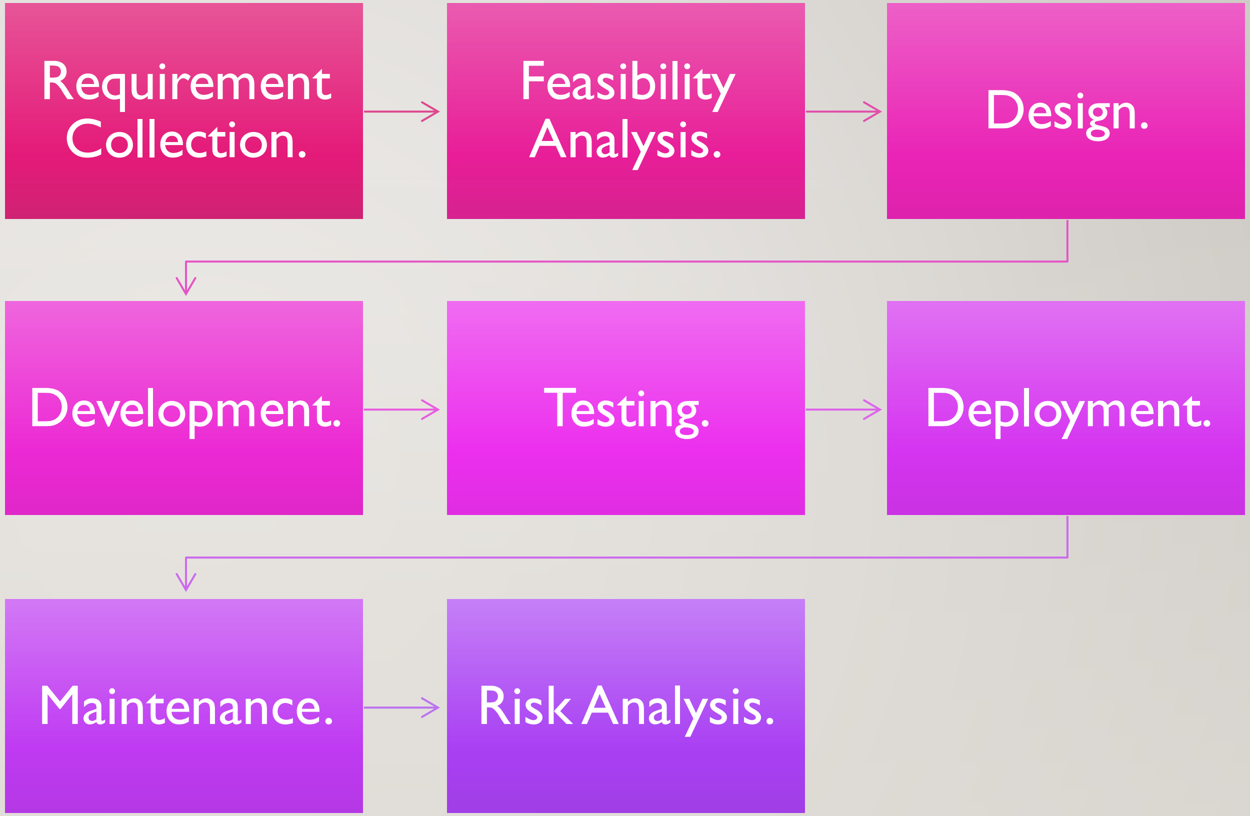
DISADVANTAGES

- Very less documentation
- Not suitable for projects having complex dependencies.
- Cost and duration is more as compare to other models.
- Huge dependency on the customer interaction, so if customer is not clear, team would be driven in wrong direction.

HYBRID MODEL

- Every SDLC model has their own advantages so we can't replace any model advantages with others so in order to merge the different model advantages in single model and use that in project we introduced one new model that is known as Hybrid Model.
- Hybrid model is the combination of more than one model together.
- We mainly consider Waterfall and Agile combination as Hybrid model. It is also known as Agile-Waterfall hybrid model
- Agile-Waterfall hybrid model allows teams developing software using Agile methodology where as hardware development teams and product managers using the Waterfall model.
- The main objective of waterfall-agile hybrid model is to be able use up to requirements phase (planning, requirement gathering etc.,.) using waterfall model and then move to agile model for the remaining phases of SDLC (design, development, testing, deployment and maintenance).

PHASES:



Requirement Collection.

Feasibility Analysis.

Design.

Development.

Testing.

Deployment.

Maintenance.

Risk Analysis.

REQUIREMENT COLLECTION

This is the very first stage of the SDLC Process. In this phase, business analyst and project manager participate in the meetings with client to gather the requirements.

Client: someone who will provide the requirement and tell what to build, who will use it or purpose of it to all the participants.

Business Analyst: Person who gathers the requirement from the client.

Project Manager : who handles the project

Who involves - Client, Business Analyst, Project Manager and all Stake holders.

ANALYSIS

Once the requirements gathering phase completed, Business Analyst (BA) converts the business requirements into technical requirements with the help senior team members like SMEs (Subject-matter experts) and team leads. The technical requirements documented in a document called SRS (Software Requirement Specification) document. SRS consists of all the technical requirements for all system involved (that includes cross-applications if exists)

SRS document name varies from project to project.
SRS (Software Requirement Specification) or Requirement Specification Document (RSD) or Project Specification Document (PSD).

DESIGN

- In this phase designer converts SRS into DDS(Design Document Specification). overall system architecture is designed by defining their functionality of each module that includes their interaction with cross-systems.

- The Design Document Specifications that are two types based on their level of design. Those are -
  High-level Design (HLD) - HLD contains the high-level architecture changes or design changes.
  Low-level Design (LLD) - LLD contains very detailed level architecture or design changes.

- Both HLD and LLD creates by **Senior developers or architects** creates HLD.
- **Note**: The architects usually provide more than one approach in a Design Document Specification. This DDS sent for review to all the stakeholders and the best approach would be selected based on the features such as robustness, budget and timing constraints.

CODING

- Once Design is approved implementation gets started with code followed by organization coding standards,Once coding for small component is done developer use to compile and debug the developed code using different tools.
- Coding or Development or Implementation all are having same meaning and purpose.
  Implementation means developers are implementing the source code as per DDS, since the implementation helps to develop the piece of product/software this is also called as Development. Development needs some programming language understanding to develop the product,since development means nothing but piece of code using some programming language and that is coding in technical term so that why we call this phase as Coding also.

TESTING

- **Note**: This phase helps to understand whether implemented code is as per client expectation or not, Once the coding completed The developer performs the initial testing that are unit testing (UT) and/or Application Integration Testing (AIT) before handover the code to the testing team.
- Testers do lot of testing like Functional testing, system testing, acceptance testing etc. and verify the BRS(Business requirement specification). If development is not as per BRS, testers raise or log the defect tagging to developer and dev needs to fix it and share the updated code with tester in form of build and then tester will do retesting. After retesting defect might reopen or close. This entire testing and bug logging activity follows certain process that is called as Software testing life cycle or Software testing process.

DEPLOYMENT

Once the product is tested well and that is less buggy then client will review the test results and artifacts and then approve it for deployment

Once the software got deployed to production, then the new functionality available to the end-users.

MAINTENANCE PHASE:

- Once the end-user starts using the newly deployed software, there might be a possibility that the real-time issues starts coming up. The team has to fix these issues to avoid the loss in business if the issue has less priority or less impact. If the issue has high priority and has huge impact, client can take a decision to roll out or backout new changes and refine the functionalities as required. This process of taking care for the finished product is called as maintenance.

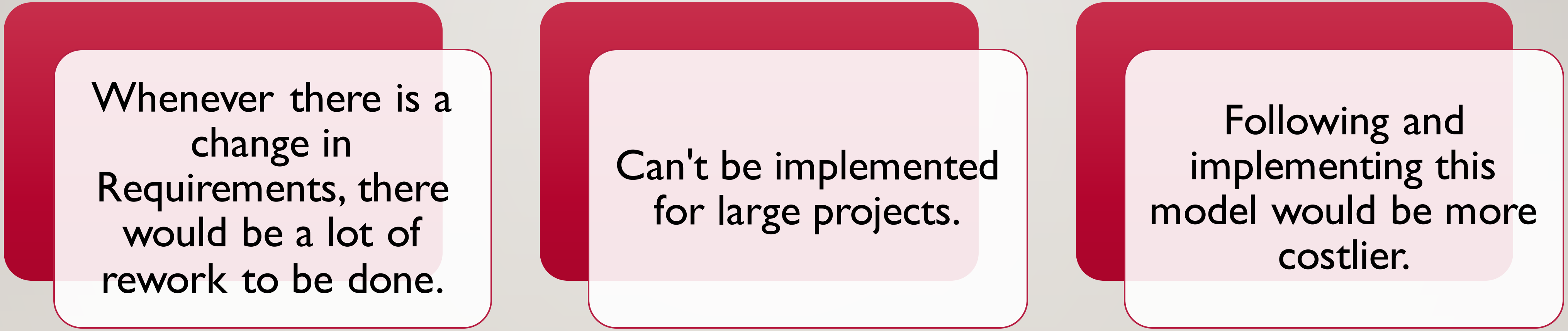
RISK ANALYSIS

- This phase parallelly run with all development phases starting with planning and finishes with maintenance.
- This phase is used to lists all expected risks and provides all the necessary activities to avoid such risks.

ADVANTAGES:

- Simple and easy to use.
- Easy to implement in small and medium projects.
- Chances of rejection would be very less.
- Easy to understand and implement.

DISADVANTAGES:



Whenever there is a change in Requirements, there would be a lot of rework to be done.

Can't be implemented for large projects.

Following and implementing this model would be more costlier.

THANK YOU!