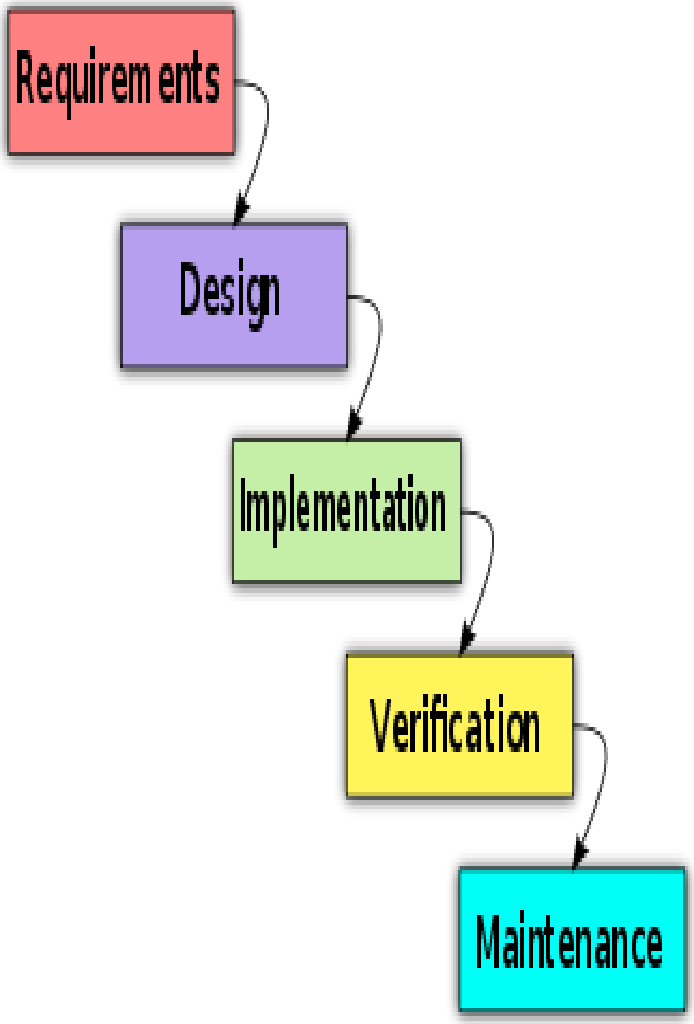
Software Life Cycle Model

An enlightenment of :
- Waterfall Model
- Agile Model

By Ananda Pramanik

Waterfall Model



The simplest software development life cycle model is the waterfall model, which states that the phases are organized in a linear order. A project begins with feasibility analysis. On the successful demonstration of the feasibility analysis, the requirements analysis and project planning begins. The design starts after the requirements analysis is done. And coding begins after the design is done. Once the programming is completed, the code is integrated and testing is done. On successful completion of testing, the system is installed. After this the regular operation and maintenance of the system takes place. The following figure demonstrates the steps involved in waterfall life cycle model.

Thus the waterfall model maintains that one should move to a phase only when its preceding phase is completed and perfected.

Waterfall Model - Pros

The waterfall model is the oldest and most widely used model in the field of software development. There are certain advantages of this model, which makes it, one of the most widely used models as yet. Some of them are:

Pros –

- Simple and easy to implement.
- Easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Documentation is produced at every stage of the software's development.
- Works well for smaller projects where requirements are very well understood.
- The amount of resources required to implement this model are minimal.
- After every major stage of software coding, testing is done to check the correct running of the code.

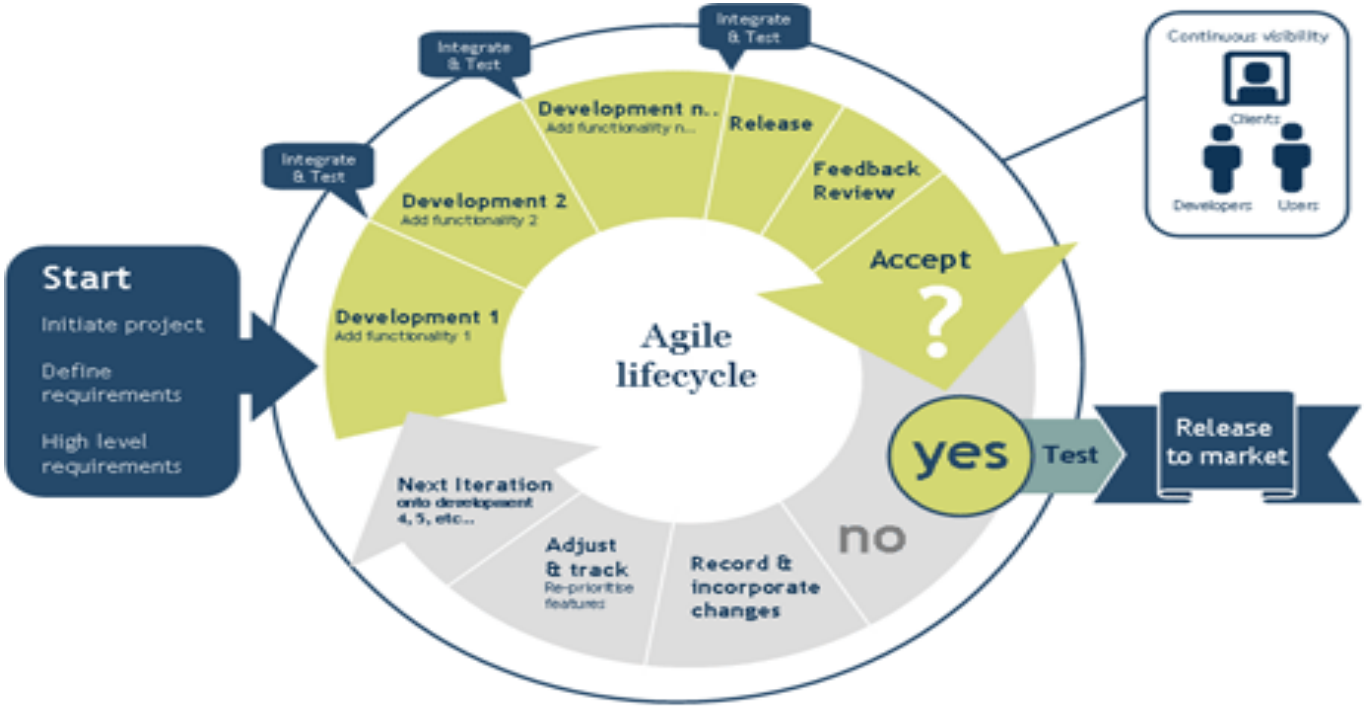
Waterfall Model - Cons

Even after knowing Waterfall model is most widely used model in the field of software development. Then what could be the possible disadvantages of the waterfall model? Here are a few :

Cons –

- We cannot go back a step; if the design phase has gone wrong, things can get very complicated in the implementation phase.
- The only way to amend something which has been already developed is to go back and start again.
- Adjusting scope during the life cycle can kill a project.
- Until the final stage of the development cycle is complete, a working model of the software does not lie in the hands of the client. Thus, he is hardly in a position to inform the developers, if what has been designed is exactly what he had asked for
- High amounts of risk and uncertainty.

Agile Model



Agile software development is a group of software development methodologies based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams.

Agile methods break tasks into smaller iterations or parts and do not directly involve long term planning. The project scope and requirements are clearly laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

Agile Model - Pros

Agile methodologies work on iterations i.e. successive approximation. After this the task is further broken into smaller tasks or iterations called sprints. Each sprint involves a small software development lifecycle. There are certain advantages of this model, Some of them are:

Pros –

- Agile methodology has an adaptive team which is able to respond to the changing requirements.
- The documentation is crisp and to the point to save time.
- Face to face communication and continuous inputs from customer representative leaves no space for guesswork.
- The end result is the high quality software in least possible time duration and satisfied customer.
- Less defects in the final product.
- Fewer “surprises” (scope changes).
- Significantly reducing the overall **risk** associated with software development

Agile Model - Cons

Agile methodology does not require long term planning. In fact, each iteration requires about a couple of weeks from planning to development. Later all these small tasks are integrated to form the required software. Then what could be the possible disadvantages of the Agile model? Here are a few :

Cons –

- Because agile methods are not process-oriented and require quick response to change, a lack of documentation is often a primary characteristic.
- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.
- Agile requirements are barely sufficient. Requirements are clarified just in time for development and can be documented in much less detail due to the timeliness of conversations.
- Frequent delivery of product and the need to sign off each feature as *done* before moving on to the next makes UAT (user acceptance testing) continuous and therefore potentially quite onerous.

Waterfall Vs Agile

Benefits over waterfall Model and Agile Model

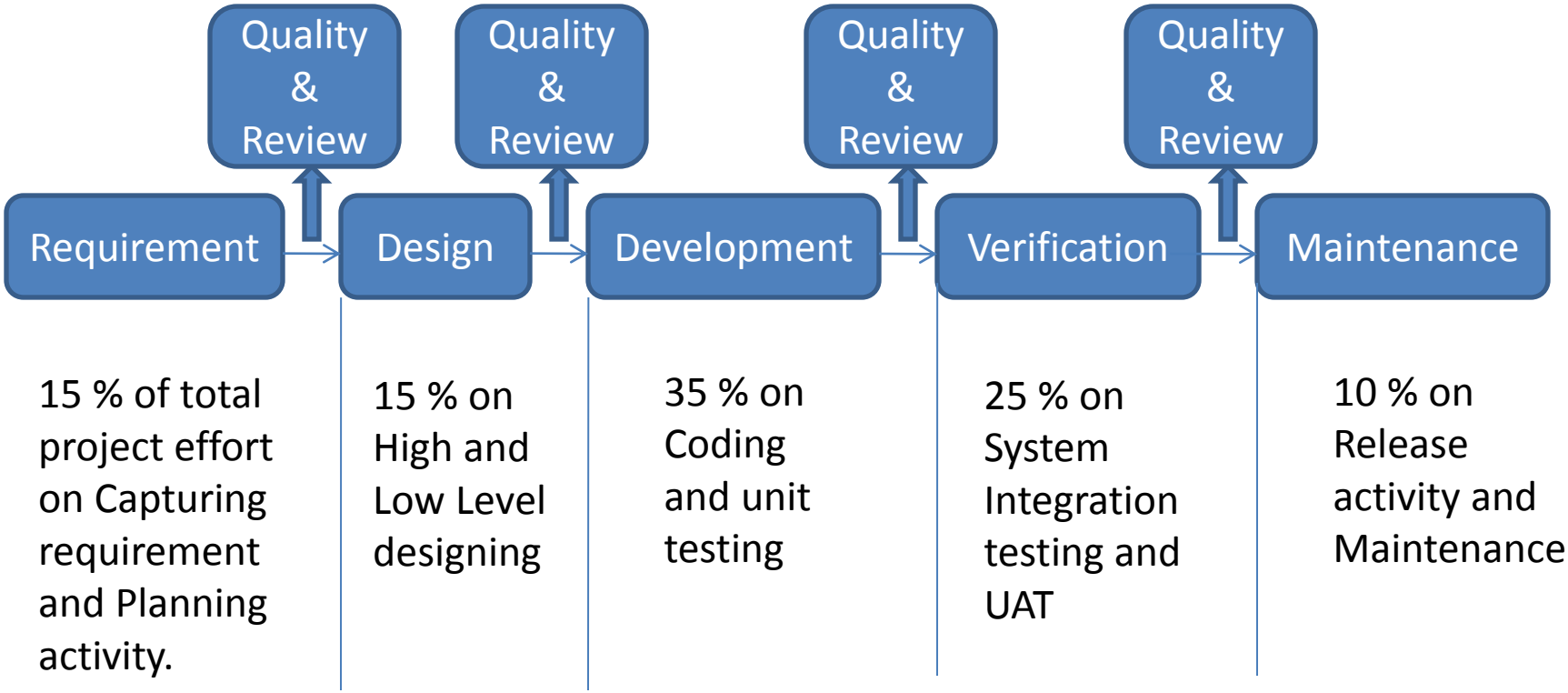
Waterfall	Agile
Proven Model to execute Projects	Suggested model for execution Products.
Sets expectations up front for cost, schedule	Continuous delivery and feedback cycles (iterative and incremental development)
Requirements must be validated and exit criteria must be met before proceeding to next phase	Changing requirements are welcome
Customer can focus on other things in the meantime	Early testing and continuous integration
“Measure twice, cut once” means less potential for rework	Customer collaboration and acceptance of each feature as it’s developed

Waterfall Vs Agile

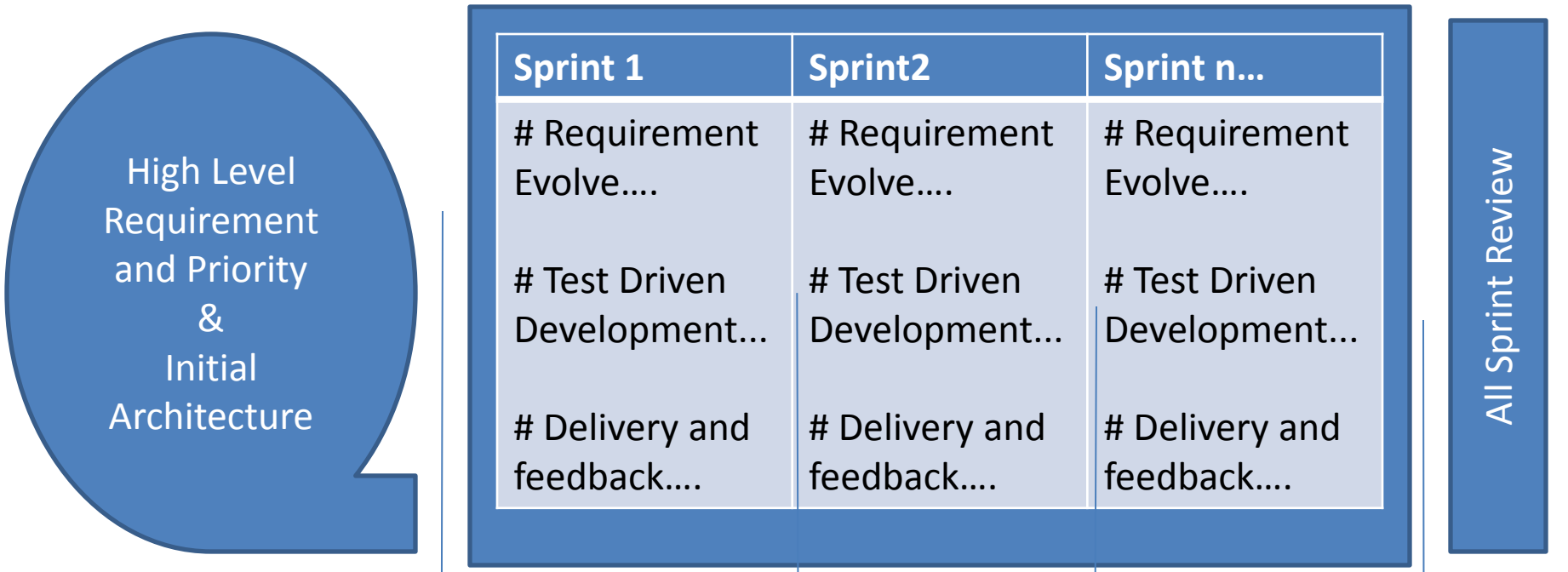
Conceptual differences over waterfall Model and Agile Model

Waterfall	Agile
Sequentially structured approach, the development team goes ahead to the next stage of development, only after the previous stage is fully accomplished. Considerable amount of time in each stage of development, till all doubts are cleared and all requirements are met.	Agile models involve multiple iterative development schedules that seek to improve the output with every iteration. Each iteration goes through all the steps of design, coding and testing. The iterative cycle continues till the customer is delivered with a product which exactly meets his expectations.
The belief that drives this kind of software development model is that considerable time spent in initial design effort corrects bugs in advance.	The design idea is never totally frozen or set in stone, but it's allowed to evolve as new ideas come in with each release. Seek to improve the output with every iteration.

Effort Distribution on Waterfall



Effort Distribution on Agile



10 % of total project effort spend on High Level Requirement and Planning

20 % on Sprint1

15 % on Sprint2

50 % on Sprint n …

5 % on Review

After high level Requirement and Planning, generally 85 % of effort distribute among different Sprints. Each Sprint contains separate effort, depending on its size.

Thank you