Indian Institute of Information Technology, Vadodara

Meetup

SDLC MODEL

Author:
Pooja Gurjar
Radheyshayam
Shubham Singh

Date: 13-09-2018

Contents

1 Purpose

The purpose of this document is to find a suitable SDLC model for our project by weighing the pros and cons of each SDLC model.

2 Software Development Life Cycle Models

Software development is the continuous process of designing complete development of software. It involves various phases for getting the required output. Using SDLC Models in our project will ensure that the final product is completed in a systematic and cost-effective manner.
The various phases involved in the Software Development Life cycle are:

- Feasibility
- Requirements Specification
- Design
- Coding and Unit Testing
- Integration
- Testing
- Maintenance

Some of the SDLC Models we studied are:

- Classical Waterfall Model
- Iterative Waterfall Model
- Spiral Model
- Rapid Application Development Model

Let us look at each SDLC model in detail with advantages and disadvantages of each model.

2.1 Classical Waterfall Model

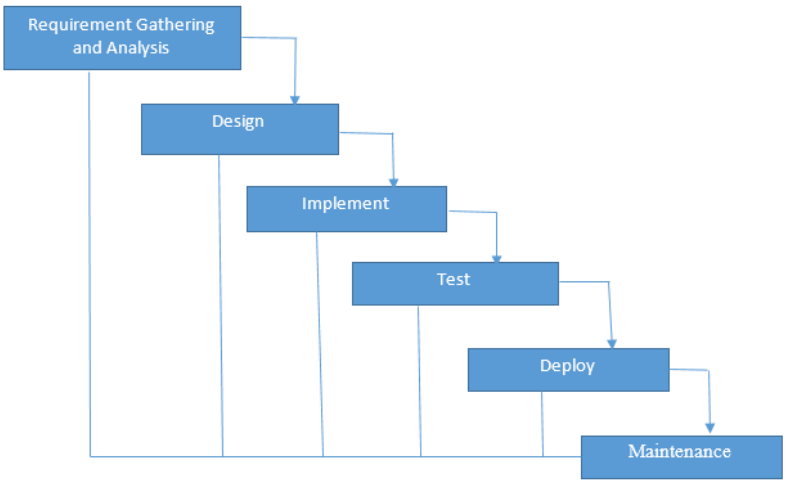
The waterfall model is a linear sequential flow in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of software implementation. In this model any phase in the development process begins only if the previous phase is complete.

Advantage:

1. This model is simple and easy to use.
2. This model is effective when the problem is well understood.
3. Rigidity of the model makes adhering to deadlines easier.
4. Model works better for smaller projects.

Disadvantages:

1. In this model reverting back to previous phases is costly and not feasible.
2. Huge amount of risk.
3. It requires all requirements to be stated explicitly.
4. Difficult to accommodate uncertainty.
5. A major mistake is as good as an irreversible mistake.



2.2 Iterative Waterfall Model

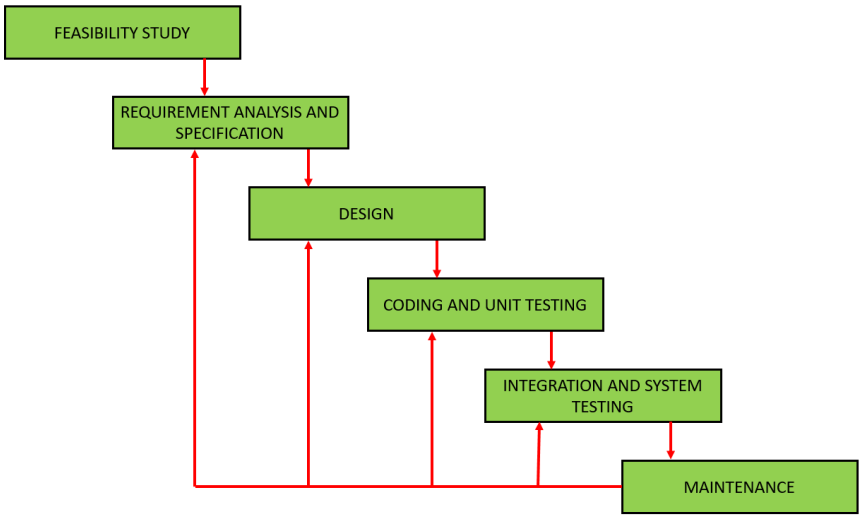
The iterative waterfall model works on the principle that the errors should be detected and rectified as soon as possible. In this model one can go the previous phase where the error can be detected to ensure that all the previous phases until now are error free.

Advantage:

1. If the errors are detected at an early stage, this model becomes extremely systematic and easy to implement.
2. Results are obtained periodically.
3. Risks are identified and resolved during iteration.
4. Easy to measure progress.
5. Supports changing requirements.

Disadvantages:

1. If errors are not detected early, the whole development process fails.
2. More management is needed.



2.3 Spiral Model

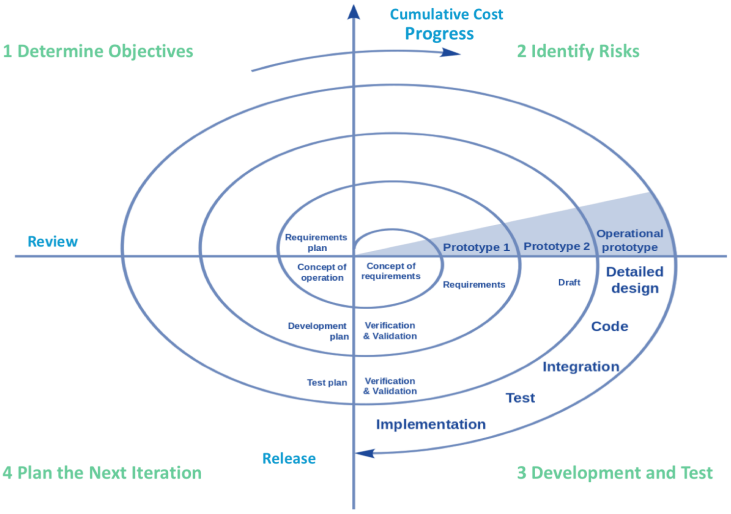
Spiral Model The spiral model combines the idea of iterative development with the systematic, controlled aspects of the waterfall model. This Spiral model is a combination of iterative development process model and sequential linear development model i.e. the waterfall model with a very high emphasis on risk analysis. It allows incremental releases of the product or incremental refinement through each iteration around the spiral.

Advantages:

1. High amount of risk analysis hence, avoidance of Risk is enhanced.
2. Good for large and mission-critical projects.
3. Strong approval and documentation control.
4. Additional Functionality can be added at a later date.
5. Software is produced early in the software life cycle.

Disadvantages:

1. Can be a costly model to use.
2. Risk analysis requires highly specific expertise.
3. Project's success is highly dependent on the risk analysis phase.
4. Does not work well for smaller projects.



2.4 Rapid Application Development Model

Rapid application development RAD is a software development methodology that uses minimal planning in favor of rapid prototyping. A prototype is a working model that is functionally equivalent to a component of the product. In RAD model the functional modules are developed in parallel as prototypes and are integrated to make the complete product for faster product delivery. Since there is no detailed pre-planning, it makes it easier to incorporate the changes within the development process. RAD projects follow iterative and incremental model and has small teams comprising of developers, domain experts, customer representatives and other IT resources working progressively on their component or prototype.

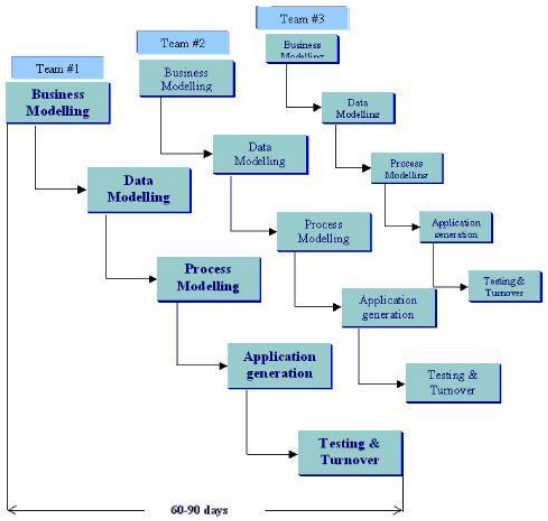
Advantages:

1. Changing requirements can be accommodated.
2. Progress can be measured.
3. Iteration time can be short with use of powerful RAD tools.
4. Productivity with fewer people in short time.
5. Reduced development time.
6. Increases re-usability of components.
7. Quick initial reviews occur.
8. Encourages customer feedback.
9. Integration from very beginning solves a lot of integration issues.

Disadvantages:

1. Dependency on technically strong team members for identifying project requirements.
2. Only system that can be modularized can be built using RAD.
3. Requires highly skilled developers/designers.
4. High dependency on modeling skills.

5. Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.
6. Management complexity is more.
7. Suitable for systems that are component based and scalable.
8. Requires user involvement throughout the life cycle.
9. Suitable for project requiring shorter development times.



2.5 Agile Model

In the agile methodology after every development iteration, the customer is able to see the result and understand if he is satisfied with it or he is not. This is one of the advantages of the agile software development life cycle model. One of its disadvantages is that with the absence of defined requirements it is difficult to estimate the resources and development cost.

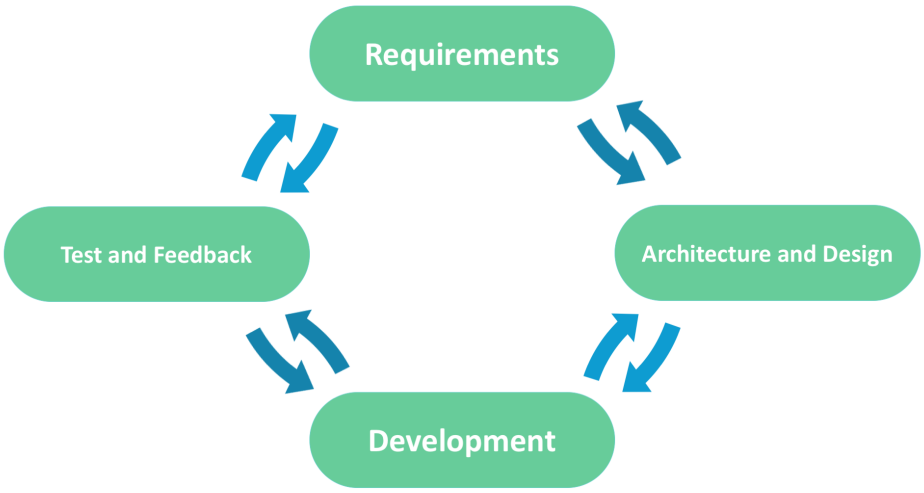
Extreme programming is one of the practical use of the agile model. The basis of such model consists of short weekly meetings – Sprints which are the part of the Scrum approach.

Advantages:

1. Corrections of functional requirements are implemented into the development process to provide the competitiveness
2. Project is divided by short and transparent iterations
3. Risks are minimized thanks to the flexible change process
4. Fast release of the first product version

Disadvantages:

1. Difficulties with measuring the final cost because of permanent changes
2. The team should be highly professional and client-oriented
3. New requirements may conflict with the existing architecture
4. With all the corrections and changes there is possibility that the project will exceed expected time



2.6 Prototyping Model

During Prototyping model, the software development team, clarify requirements and/or design elements, that generate mockups and prototypes of screens, reports, and processes. Although some of the prototypes may appear to be very substantial, they're generally similar to a movie set: everything looks good from the front but there's nothing in the back.

Advantages of Prototyping:
The software designer and implementer can obtain feedback from the users early in the project. The client and the contractor can compare if the software made matches the software specification, according to which the software program is built. It also allows the software engineer some insight into the accuracy of initial project estimates and whether the deadlines and milestones proposed can be successfully met.

Disadvantages of Prototyping:
Often clients expect that a few minor changes to the prototype will more than suffice their needs. They fail to realise that no consideration was given to the overall quality of the software in the rush to develop the prototype. The developers may lose focus on the real purpose of the prototype and compromise the quality of the product. For example, they may employ some of the inefficient algorithms or inappropriate programming languages used in developing the prototype. This mainly due to laziness and an over reliance on familiarity with seemingly easier methods.

A prototype will hardly be acceptable in court in the event that the client does not agree that the developer has discharged his/her obligations. For this reason using the prototype as the software specification is normally reserved for software development within an organisation.

3 Decided Model- Iterative Waterfall Model

The main reason for selecting this model for our project is because of this model allows us to develop a system through repeated cycles (iterative) and in smaller portions at a time, allowing software developers to take the advantage of what was learned during the development of earlier parts or versions of the system. Learning comes from both the development and use of the system, where possible key steps in the process start with a simple implementation of a subset of the software requirements and iteratively enhance the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added.
The procedure itself consists of the initialization step, the iteration step, and the Project Control List. The initialization step creates a base version of the system. The goal for this initial implementation is to create a product to which the user can react.To guide the iteration process, a project control list

is created that contains a record of all tasks that need to be performed.
The iteration involves the redesign and implementation of iteration is to be simple, straightforward, and modular, supporting redesign at that stage or as a task added to the project control list.
The analysis of an iteration is based on user feedback, and the program analysis facilities available. It involves the analysis of the structure, modularity, usability, reliability, efficiency, achievement of goals. The project control list is modified in light of the analysis results.