Software Engineering

Week 4

Muhammad Daniyal Liaquat

Agile Model

- The meaning of Agile is swift or versatile.“
- Agile process model" refers to a software development approach based on iterative development.
- Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning.
- The project scope and requirements are 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.

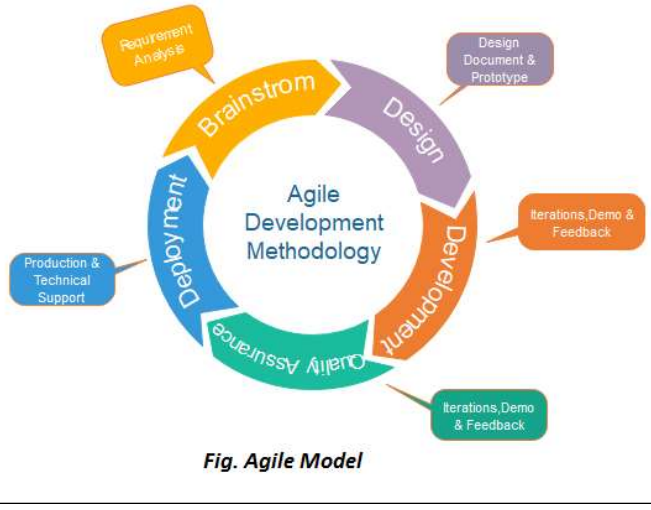


Fig. Agile Model

3

Phases of Agile Model:

- Following are the phases in the Agile model are as follows:
- Requirements gathering
- Design the requirements
- Construction/ iteration
- Testing/ Quality assurance
- Deployment
- Feedback

4

Agile Testing Methods:

- Scrum
- Crystal
- Dynamic Software Development Method(DSDM)
- Feature Driven Development(FDD)
- Lean Software Development
- eXtreme Programming(XP)

5

Scrum

- SCRUM is an agile development process focused primarily on ways to manage tasks in team-based development conditions.
- There are three roles in it, and their responsibilities are:
- **Scrum Master:** The scrum can set up the master team, arrange the meeting and remove obstacles for the process
- **Product owner:** The product owner makes the product backlog, prioritizes the delay and is responsible for the distribution of functionality on each repetition.
- **Scrum Team:** The team manages its work and organizes the work to complete the sprint or cycle.

6

eXtreme Programming(XP)

- This type of methodology is used when customers are constantly changing demands or requirements, or when they are not sure about the system's performance.
- Crystal:
- There are three concepts of this method-
- Chartering: Multi activities are involved in this phase such as making a development team, performing feasibility analysis, developing plans, etc.
- Cyclic delivery: under this, two more cycles consist, these are:
  - Team updates the release plan.
  - Integrated product delivers to the users.
- Wrap up: According to the user environment, this phase performs deployment, post-deployment.

7

Dynamic Software Development Method(DSDM):

- DSDM is a rapid application development strategy for software development and gives an agile project distribution structure. The essential features of DSDM are that users must be actively connected, and teams have been given the right to make decisions. The techniques used in DSDM are:

- Time Boxing
- MoSCoW Rules
- Prototyping

8

The DSDM project contains seven stages:

- Pre-project
- Feasibility Study
- Business Study
- Functional Model Iteration
- Design and build Iteration
- Implementation
- Post-project

9

Feature Driven Development(FDD):

- This method focuses on "Designing and Building" features.
- In contrast to other smart methods, FDD describes the small steps of the work that should be obtained separately per function.

10

Lean Software Development:

- Lean software development methodology follows the principle "just in time production." The lean method indicates the increasing speed of software development and reducing costs. Lean development can be summarized in seven phases.
- Eliminating Waste
- Amplifying learning
- Defer commitment (deciding as late as possible)
- Early delivery
- Empowering the team
- Building Integrity
- Optimize the whole

11

When to use the Agile Model?

- When frequent changes are required.
- When a highly qualified and experienced team is available.
- When a customer is ready to have a meeting with a software team all the time.
- When project size is small.

12

Rapid Application Development (RAD)

Rapid application development is another form of incremental model.

It is a high-speed adaptation of the linear sequential model in which fully functional system is developed in a very short time (2-3 months).

This model is only applicable in the projects where requirements are well understood and project scope is constrained.

Because of this reason, it is used primarily for information systems.

13

Synchronize and Stabilize Model

This is yet another form of incremental model adopted by Microsoft.

In this model, during the requirements analysis interviews of potential customers are conducted and requirements document is developed.

Once these requirements have been captured, specifications are drawn up.

The project is then divided into 3 or 4 builds. Each build is carried out by small teams working in parallel. At the end of each day the code is synchronized (test and debug) and at the end of the build it is stabilized by freezing the build and removing any remaining defects.

Because of the synchronizations, components always work together. The presence of an executable provides early insights into operation of product.

14

Spiral Model

This model was developed by **Barry Boehm**.

The main idea of this model is to avert risk, as there is always an element of risk in development of software.

For example, key personnel may resign at a critical juncture, the manufacturer of the software development may go bankrupt, etc.

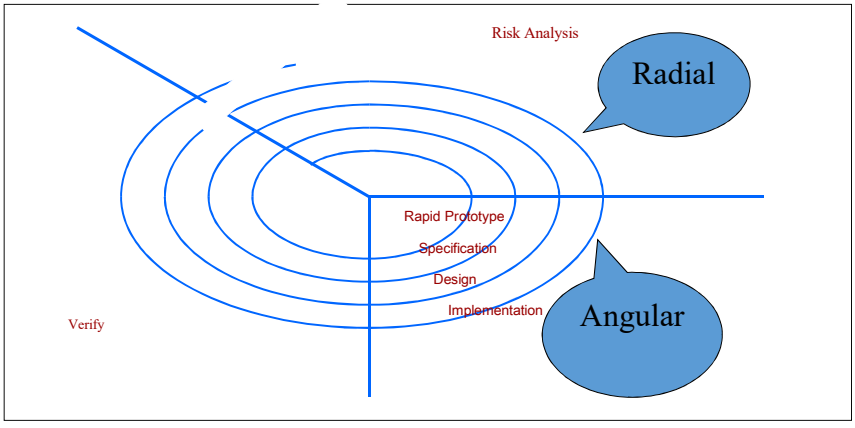
In its simplified form, the Spiral Model is Waterfall model plus risk analysis.

In this case each stage is preceded by **identification of alternatives** and **risk analysis** and is then followed by evaluation and planning for the next phase.

If risks cannot be resolved, project is immediately terminated. This is depicted in the following diagram

15

Spiral Model

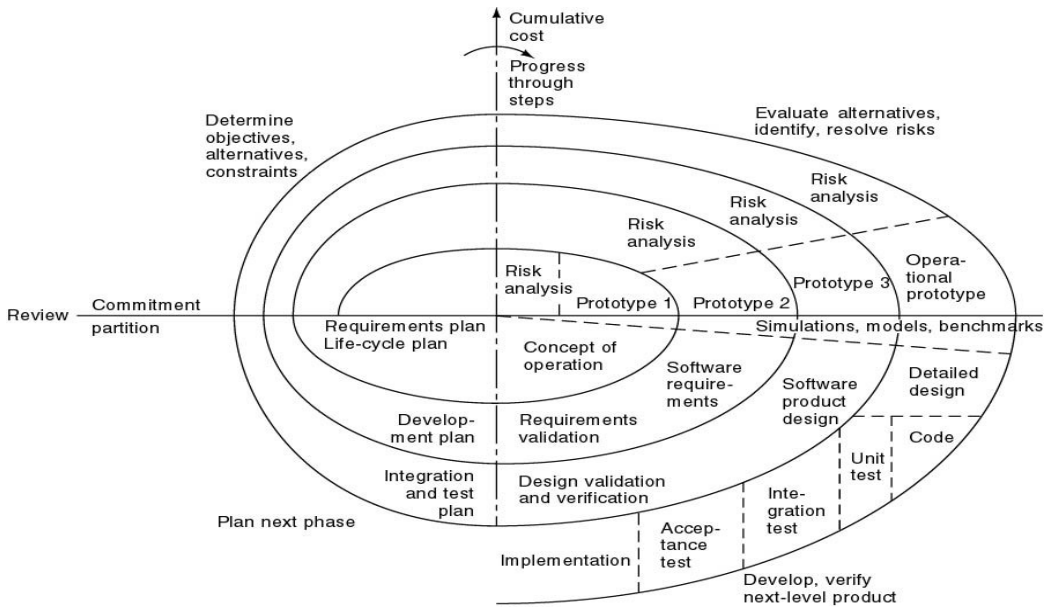


As can be seen, a Spiral Model has two dimensions. Radial dimension represents the cumulative cost to date and the angular dimension represents the progress through the spiral.

Each phase begins by determining objectives of that phase and at each phase; a new process model may be followed.

16

A Full version of Spiral Model



17

Spiral Model

The main strength of the Spiral Model comes from the fact that it is very sensitive to the risk.

Because of the spiral nature of development, it is easy to judge how much to test and there is no distinction between development and maintenance.

It however can only be used for large-scale software development and that too for internal (in-house) software only.

18

Object Oriented Models

Object-oriented lifecycle models appreciate the need for **iteration** within and between phases. There are a number of these models. All of these models incorporate some form of **iteration**, **parallelism**, and **incremental development**.

19

OOM (Extreme Programming)

It is a somewhat controversial new approach.

In this approach, user requirements are captured through stories which are the scenarios presenting the features needed by the client?

Estimate for duration and cost of each story is then carried out.

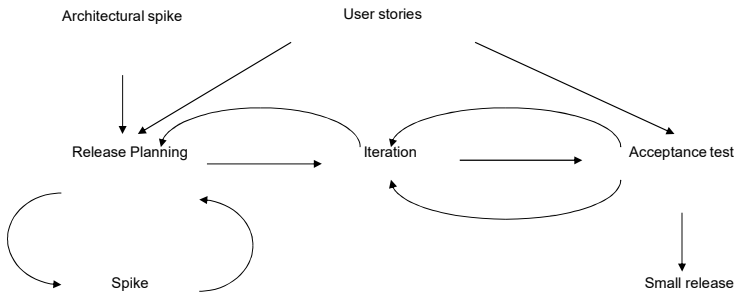
Stories for the next build are selected.

Then each build is divided into tasks.

Test cases for task are drawn up first before and development and continuous testing is performed throughout the development process.

20

Extreme Programming



One very important feature of eXtreme programming is the concept of pair programming.

In this, a team of two developers develops the software, working in team as a pair to the extent that they even share a single computer.

21

Extreme Programming

In eXtereme Programming model, computers are put in center of large room lined with cubicles and client representative is always present.

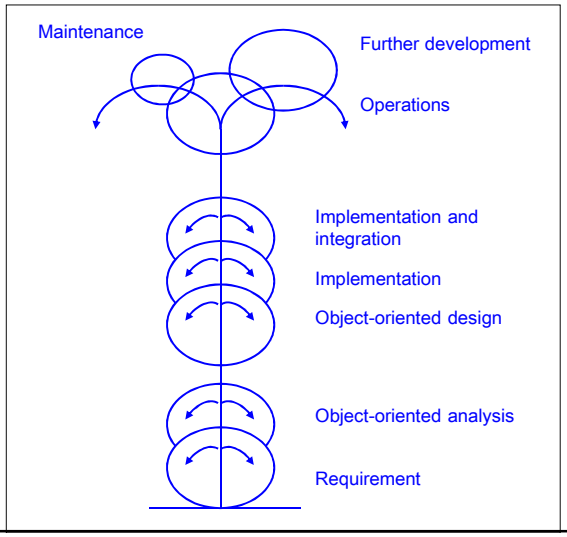
One very important restriction imposed in the model is that no team is allowed to work overtime for two successive weeks.

XP has had some successes. It is good when requirements are vague or changing and the overall scope of the project is limited. It is however too soon to evaluate XP.

22

Fountain Model

Fountain model is another object-oriented lifecycle model. This is depicted in the following diagram.



23

Fountain Model

In this model the circles representing the various phases overlap, explicitly representing an overlap between activities.

The arrows within a phase represent iteration within the phase.

The maintenance cycle is smaller, to symbolize reduced maintenance effort when the object oriented paradigm is used.

24

Rational Unified Process

Rational Unified Process is very closely associated with UML and Krutchen's architectural model.

In this model, a software product is designed and built in a succession of incremental iterations.

It incorporates early testing and validation of design ideas and early risk mitigation.

The horizontal dimension represents the *dynamic aspect* of the process.

This includes cycles, phases, iterations, and milestones.

25

Rational Unified Process

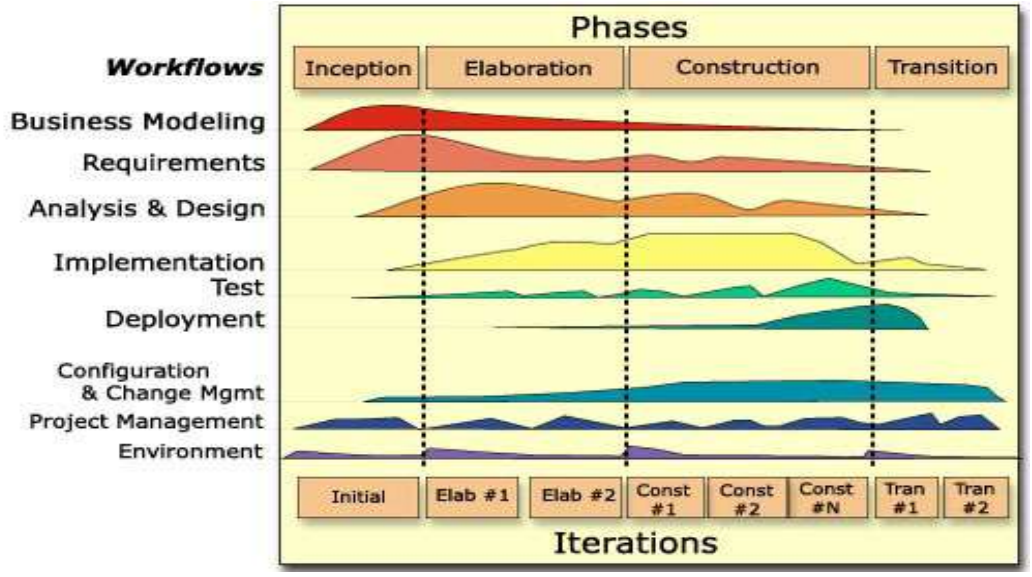
The vertical dimension represents the *static aspect* of the process described in terms of process components which include activities, disciplines, artifacts, and roles.

The process emphasizes that during development, all activities are performed in parallel, however, and at a given time one activity may have more emphasis than the other.

The following figure depicting RUP is taken from Krutchen's paper.

26

Rational Unified Process



27

Comparison of Process Models

Process Model	Strengths	Weaknesses
Build-and-fix model	• Fine for short programs that not require maintenance	• Totally unsatisfactory for nontrivial programs
Waterfall model	• Disciplined approach • Document driven	• Delivered product may not meet client's needs
Rapid prototyping model	• Ensures that delivered product meets client's needs	• Not yet proven
Incremental model	• Maximizes early return on investment • Promotes maintainability	• Requires open architecture • May degenerate into build-and-fix
Extreme programming	• Maximizes early return on investment • Fine when requirements are vague	• Small projects, small teams • Lack of design documentation • Has not yet been widely used
Synchronize-and-stabilize model	• Components always work together • Early insights into operation of product	• Has not been widely used other than at Microsoft
Spiral model	• "How much to test ?" in terms of risk • Maintenance is another cycle	• Only for large-scale, in-house products
Object-oriented models	• Iteration within phases • Parallelism between phases	• May degenerate into CABTAB

28