UNIT 4
AGILE METHODOLOGY

What is Agility?

Agile Process

Agility Principles

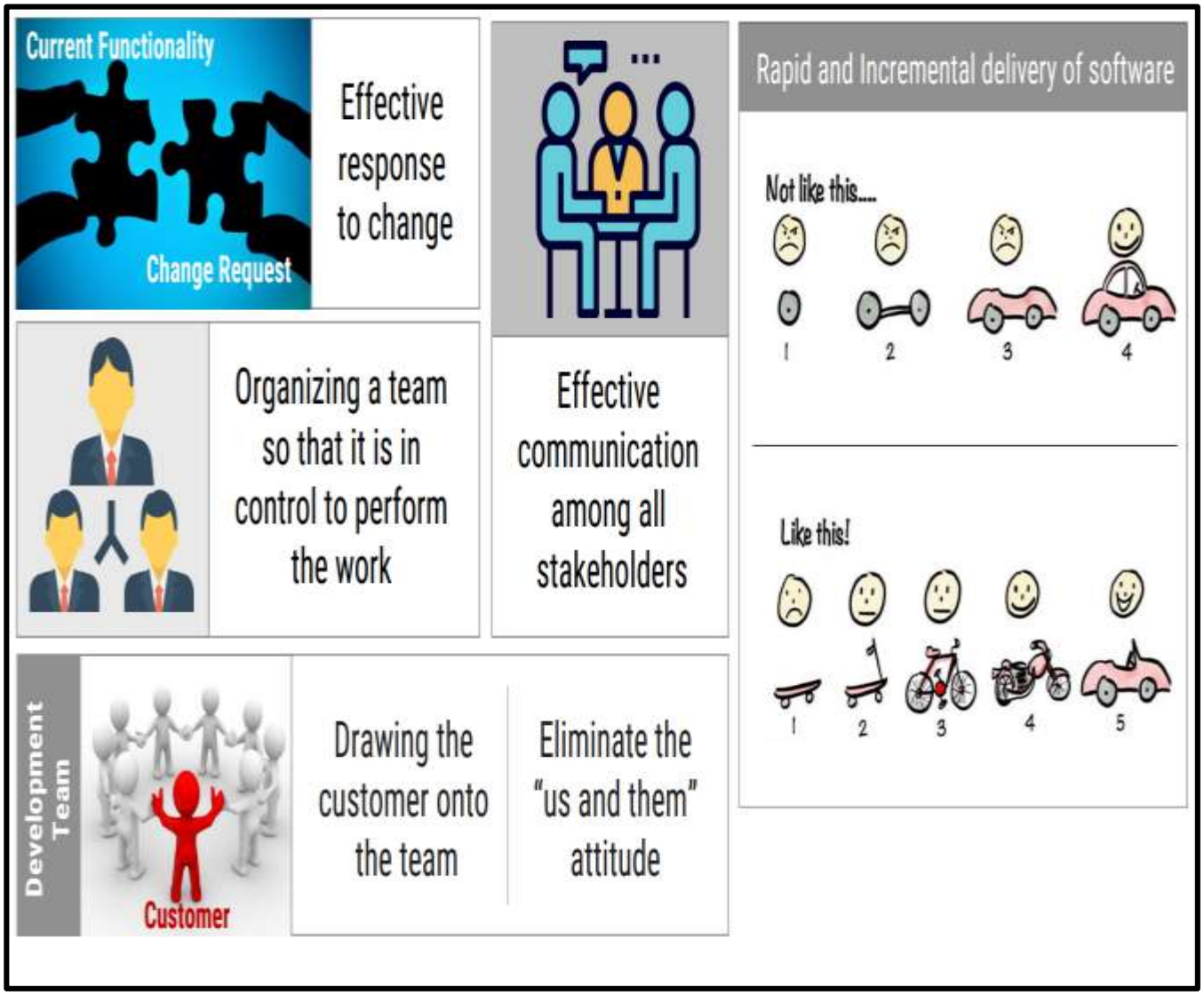
Where agile methodology not work?

Agile Process Models

- Extreme Programming (XP)
- Adaptive Software Development (ASD)
- Dynamic Systems Development Method (DSDM)
- Scrum
- Feature Driven Development (FDD)
- Crystal
- Agile Modelling (AM)

What is Agility?

- Agility is a property consisting of quickness, lightness and ease of movement.
- It is the ability to create and respond to change in order to profit in a unstable global business environment.
- It is the ability to quickly reprioritize use of resources when requirements, technology and knowledge shift.
- It is a very fast response to sudden market changes and emerging pressures by intensive customer interaction.
- It is the use of evolutionary, incremental and iterative delivery to converge on an optimal customer solution.
- It is maximizing "Business Value" with right sized, just-enough and just-in-time processes and documentation.



Agile Process

- Agile software process addresses few assumptions

    - Difficulty in predicting changes of requirements and customer priorities.
    - For many types of software; design and construction are interleaved (mixed).
    - Analysis, design, construction and testing are not as predictable as we might like.

- An agile process must be adaptable
- Requires customer feedback so that appropriate adaptations can be made.

Agility Principles

- ✓ Highest priority is to satisfy the customer through early & continuous delivery if software
- ✓ Welcome changing requirements
- ✓ Deliver working software frequently
- ✓ Business people and developers must work together
- ✓ Build projects around motivated individuals
- ✓ Emphasize face-to-face conversation
- ✓ Working software is the measure of progress
- ✓ Continuous attention to technical excellence and good design
- ✓ Simplicity – the art of maximizing the amount of work done
- ✓ The best designs emerge from self-organizing teams
- ✓ The team tunes and adjusts its behavior to become more effective

Where Agile Methodology not work ?

Agile Process Models

I. Extreme Programming (XP) :

- The most widely used approach to agile software development.
- A variant of XP called Industrial XP (IXP) has been proposed to target process for large organizations.
- It uses object oriented approach as its preferred development model.

XP Values:

Communication : To achieve effective communication, it emphasized close and informal (verbal) collaboration between customers and developers.

Simplicity : It restricts developers to design for immediate needs, do not make it complex by future needs.

Feedback : It is derived from three sources the implemented software, the customer and other software team members, it uses Unit testing as primary testing.

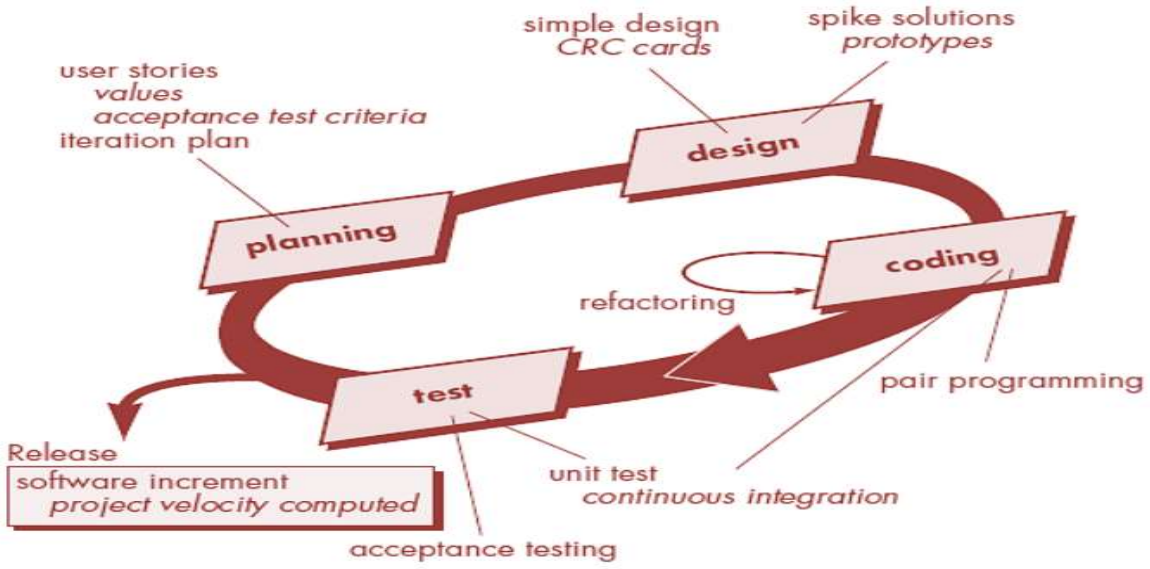
Courage : It demands courage (discipline), there is often significant pressure to design for future requirements, XP team must have discipline (courage) to design for today.

Respect : XP team respect among members.

The XP Process :

It considers four framework activities.

1. Planning
2. Design
3. Coding
4. Testing



1. **Planning :**

   User Stories :

   - Customer assigns value (Priority)
   - Developer assigns cost (number of development weeks)

   Project velocity

   - Computed at the end of first release
   - Number of stories implemented in first release
   - Estimates for future release
   - Guard against over-commitment

2. **Design :**

Keep it Simple (Design of extra functionality is rejected)

Preparation of CRC card is work project

- CRC cards identify and organize object oriented classes

Spike Solutions (in case of difficult design problem is encountered)

- Design Operational prototype intended to clear confusion

Refactoring

- Modify internals of code, No observable change

3. **Coding :**

- Develops a series of Unit test for stories included in current release.
- Complete code perform unit-test to get immediate feedback.
- XP recommend pair-programming. “Two heads are better than one”
- Integrate code with other team members, this “continuous integration” helps to avoid compatibility and interfacing problems, “smoke testing” environment to uncover errors early.

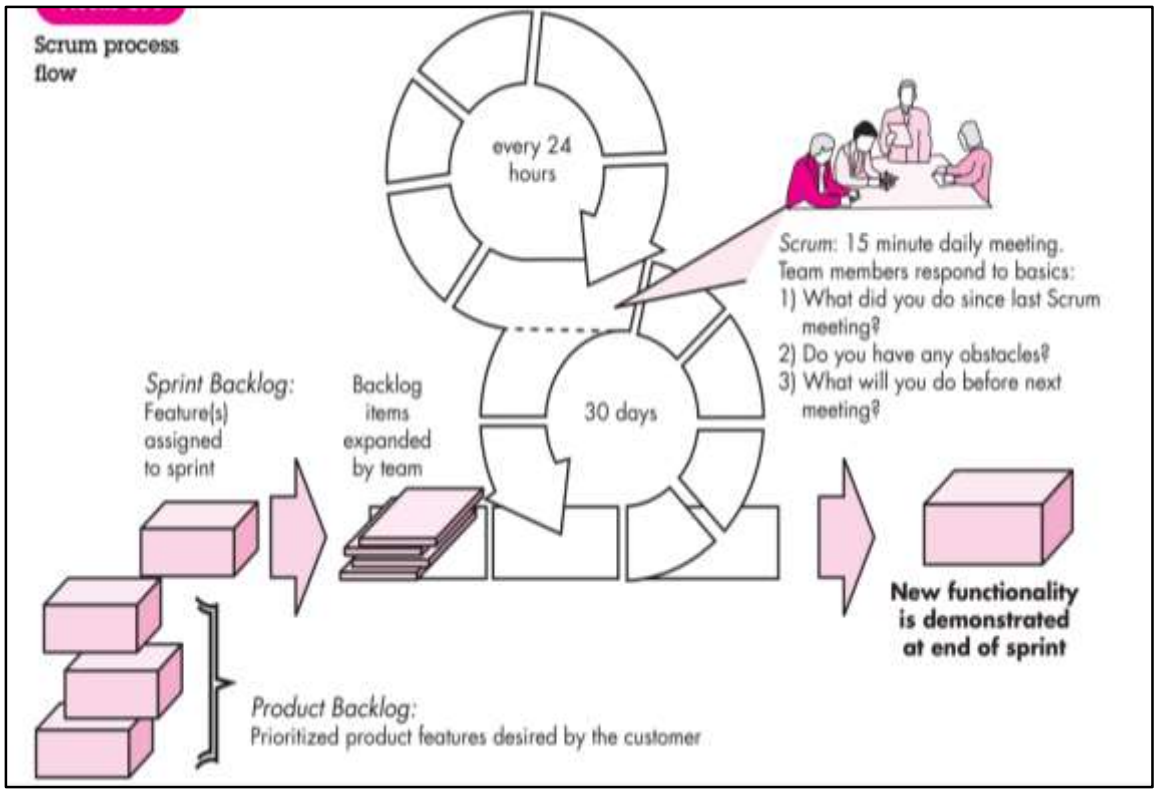
4. **Testing :**

- Unit test by developers and fix small problems
- Acceptance tests – Specified by customer
- This encourages regression testing strategy whenever code is modified

II. Scrum :

- Scrum is an agile process model which is used for developing the complex software systems.
- A Scrum is a method of restarting play in rugby that involves players packing closely together with their heads down and attempting to gain possession of the ball.
- Scrum is a lightweight process framework. Lightweight means the overhead of the process is kept as small as possible in order to maximize the productivity.
- Scrum incorporates a set of process patterns that emphasize project priorities, compartmentalized work units, communication, and frequent customer feedback.

Scrum Process flow :



Backlog :

- It is a prioritized list of project requirements or features that must be provided to the customer.
- The items can be included in the backlog at any time.
- The product manager analyses this list and updates the priorities as per the requirements.

Sprint :

- These are the work units that are needed to achieve the requirements mentioned in the backlogs.

- Typically the sprints have fixed duration or time box (of 2 to 4 weeks, 30 days).
- Change are not introduced during the sprint.
- Thus sprints allow the team members to work in stable and short-term environment.

Scrum Meetings :

- There are 15 minutes daily meetings to report the completed activities, obstacles and plan for next activities.
- Following are three questions that are mainly discussed during the meetings.
    a) What are the tasks done since last meeting ?
    b) What are the issues that team is facing ?
    c) What are the next activities that are planned ?
- The scrum master lead the meeting and analyses the response of each team member.
- Scrum meeting helps the team to uncover potential problems as early as possible.
- It leads to “knowledge socialization” and promotes “self-organizing team structure”

Demo :

- Deliver software increment to customer.
- Implement functionalities are demonstrated to customer.

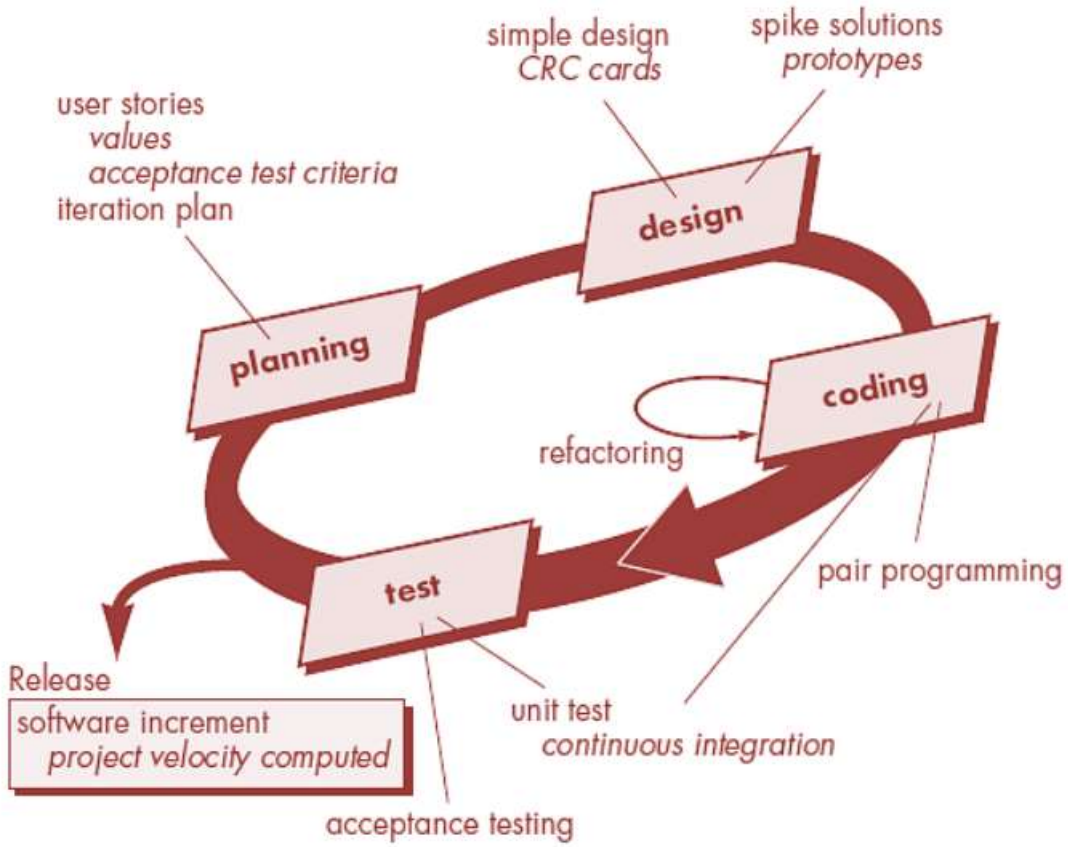
III. Adaptive Software Development (ASD) :

This is a technique for building complex software systems using iterative approach.

ASD focus on working in collaboration and team self-organization.

ASD incorporates three phases :

1. Speculation
2. Collaboration
3. Learning



1. **Speculation :**

The adaptive cycle planning is conducted.

In this cycle planning mainly three types of information us used

- Customer's mission statement
- Project constraints (Delivery date, budgets etc…)
- Basic requirements of the project

2. **Collaboration :** In this, collaboration among the members of development team is a key factor.

For successful collaboration and coordination it is necessary to have following qualities in every individual.

- Assist each other without resentment (Offense)
- Work hard.
- Posses the required skill set.
- Communicate problems and help each other.
- Criticize without any hate.

3. **Learning :** Emphasize is on learning new skills and techniques. There are three ways by which the team members learn.

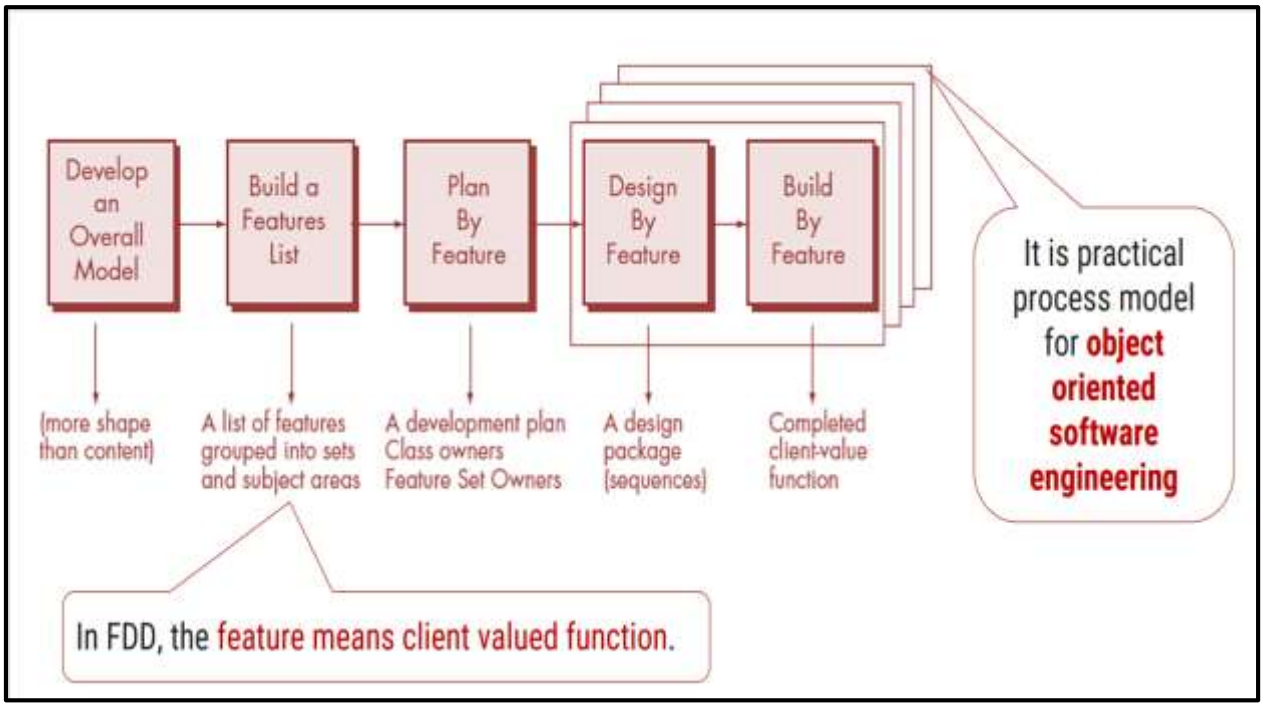
a) Focus groups : The feedback from the end-users is obtained.
b) Formal Technical review : This review is conducted for better quality.
c) Postmortems : Team analysis its own performance and makes appropriate improvements.

IV. Dynamic Systems Development Methods (DSDM) :

Various phases of this life cycle model :

- **Feasibility Study** : By analyzing the business requirements and constraints the viability of the application is determined.
- **Business Study** : The functional and informational requirements are identified and then the business value of the application is determined.
- **Functional model iteration** : The incremental approach is adopted for development.
- **Design and build iteration** : If possible design and build activities can be carried out in a parallel.
- **Implementation** : The software increment is placed in the working environment.

V. Feature Driven Development (FDD) :



- FDD, which stands for Feature-Driven Development, is a framework in the Agile methodology.
- FDD aims to ensure regular and on-time delivery to customers, in line with the values and principles of the Agile Manifesto.
- The iterative and incremental development process is central to FDD.
- FDD is suitable for large-scale, long-term projects, as it enables teams to manage changing requirements on an ongoing basis.

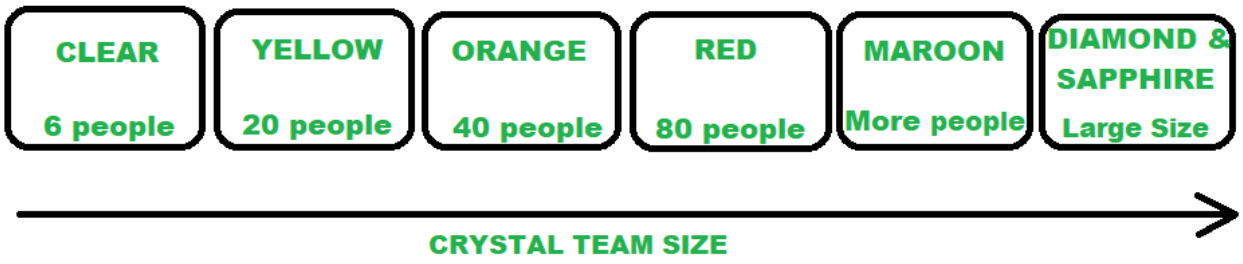
The five steps in FDD :

1. **Develop the overall model** : Here, an FDD team will determine the project scope. Multiple models will be proposed and merged to create one overall model.
2. **Build the features list** : Next, the team members will outline the customer-focused features to be developed. They will be small functions that can be completed in a short period of time. An example could be to create an automatic reminder for subscription renewal.
3. **Plan by feature** : The team will assess the individual features in the list and arrange them in the appropriate order. Then, the features will be assigned to team members.
4. **Design by feature** : At this stage, the team's chief programmer will choose which features to develop within a two-week period. A design package will be created for each feature, and team members will conduct a review before building commences.
5. **Build by feature** : Developers work to build the code for the aforementioned features. This code will be tested before the final version is created.

VI. Crystal :

- The crystal method is an agile framework that is considered a lightweight or agile methodology that focuses on individuals and their interactions.
- The methods are color-coded to significant risk to human life. It is mainly for short-term projects by a team of developers working out of a single workspace.
- Various methodologies in the Crystal family also known as weights of the Crystal approach are represented by different colors of the spectrum.
- Crystal family consists of many variants like Crystal Clear, Crystal Yellow, Crystal Red, Crystal Sapphire, Crystal Red, Crystal Orange Web, and Crystal Diamond.

1. **Crystal Clear** :- The team consists of only 1-6 members that is suitable for short-term projects where members work out in a single workspace.
2. **Crystal Yellow** :- It has a small team size of 7-20 members, where feedback is taken from Real Users. This variant involves automated testing which resolves bugs faster and reduces the use of too much documentation.
3. **Crystal Orange** :- It has a team size of 21-40 members, where the team is split according to their functional skills. Here the project generally lasts for 1-2 years and the release is required every 3 to 4 months.
4. **Crystal Orange Web** :- It has also a team size of 21-40 members were the projects that have a continually evolving code base that is being used by the public. It is also similar to Crystal Orange but here they do not deal with a single project but a series of initiatives that required programming.
5. **Crystal Red** :- The software development is led by 40-80 members where the teams can be formed and divided according to requirements.
6. **Crystal Maroon** :- It involves large-sized projects where the team size is 80-200 members and where methods are different and as per the requirement of the software.
7. **Crystal Diamond & Sapphire** :- This variant is used in large projects where there is a potential risk to human life.



VII. Agile Modelling (AM)

- Agile Modeling (AM) is a practice-based methodology for effective modeling and documentation of software-based systems.
- It is a collection of values, principles and practices for modeling software that can be applied on a software development project in an effective and light-weight manner.
- Although AM suggests a wide array of "core" and "supplementary" modeling principles, those that make AM unique are :

1. **Use multiple models** : There are many different models and notations that can be used to describe software AM suggests that to provide needed insight, each model should present a different aspect of the system and only those models that provide value to their intended audience should be used.
2. **Travel Light** : As software engineering work proceeds, keep only those models that will provide long-term value and jettison the rest.
3. **Content is more important than representation** : Modeling should impart information to its intended audience. A syntactically perfect model that imparts little useful content is not as valuable as a model with flawed notation that nevertheless provides valuable content for its audience.
4. **Know the models and the tools you use to create them** : Understand the strengths and weaknesses of each model and the tools that are used to create it.
5. **Adapt locally** : The modelling approach should be adapted to the needs of the agile team.