SOFTWARE ENGINEERING MID NOTES

1.What is Software...?
Ans: Software is more than just a program code. A program is an executable code, which serves some computational purpose. Software is considered to be collection of executable programming code, associated libraries and documentations. Software, when made for a specific requirement is called software product.

2.What is Engineering ...?
Ans: Engineering on the other hand, is all about developing products, using well-defined, scientific principles and methods.

3.What is software engineering.?
Ans: Software engineering is an engineering branch associated with development of software product using well-defined scientific principles, methods and procedures. The outcome of software engineering is an efficient and reliable software product.

4.What are the essential attributes of good Software?
Ans: For a software to be classified as good, it must possess some key features.

1. Functionality: A good software must be able to do what it was designed to do. The software requirements must guide the design and implementation of the software.

2. Usability: The software must be usable; the users must not find it difficult to figure out how a good software works. A good software is user-centered and user-friendly.

3. Efficiency: Efficiency means that perform it's operations with minimal time and processing power. A good software uses the least amount of processing power and memory needed to achieve the desired result.

4. Maintainability: A good software must evolve with changing requirements.

5. Security: A good software must be secure. It should not cause physical or economic damage in the event of a system failure. Unauthorized users must not be allowed access to the system.

6. Reliability: A reliable system will rarely fail, and even when it does fail, there are recovery mechanisms in the software to recover from the failure with minimal losses.

5. What is a software process?

A software process is the set of activities and associated results that produce a software

product. There are four fundamental process activities (covered later in the

book) that are common to all software processes. These are:

1. Software specification where customers and engineers define the software to be produced and the constraints on its operation.

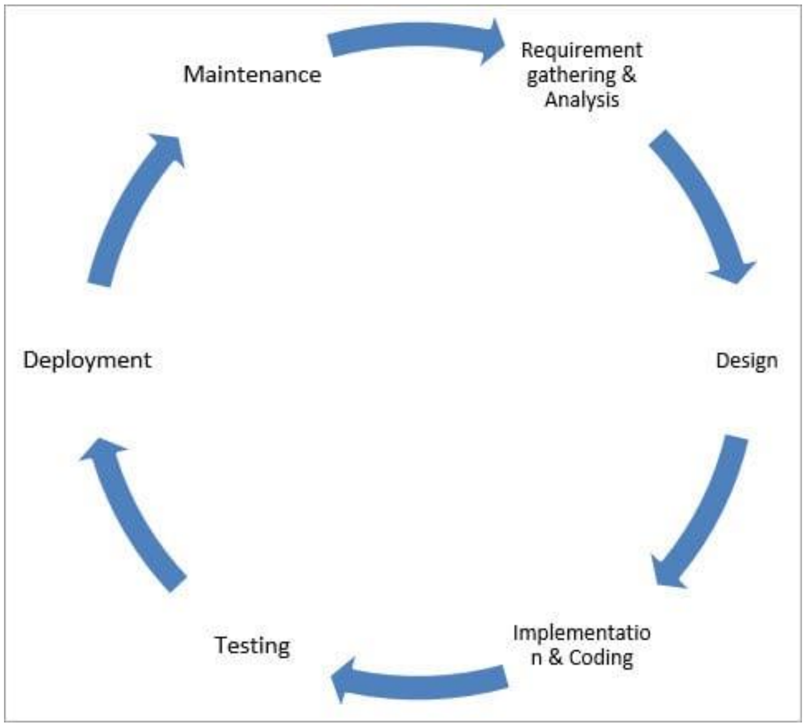
2. Software development where the software is designed and programmed.

3. Software validation where the software is checked to ensure that it is what the customer requires.

4. Software evolution where the software is modified to adapt it to changing customer and market requirements.

6.What is SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)?

Ans: SDLC is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software that meets customer expectations.



The system development should be complete in the pre-defined time frame and cost. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life Cycle has its own process and deliverables that feed into the next phase. SDLC stands for Software Development Life Cycle and is also referred to as the Application Development life-cycle. A structured set of activities required to develop a software system. The way we produce software, including:

- **Requirements Analysis:**
  The requirement is the first stage in the SDLC process. It is conducted by the senior team members with inputs from all the stakeholders and domain experts in the industry. Planning for the quality assurance requirements and recognition of the risks involved is also done at this stage. This stage gives a clearer picture of the scope of the entire project and the anticipated issues, opportunities, and directives which triggered the project. Requirements Gathering stage need teams to get detailed and precise requirements. This helps companies to finalize the necessary timeline to finish the work of that system. The requirements are collected using a number of practices as given:

  - studying the existing or obsolete system and software,
  - conducting interviews of users and developers,
  - referring to the database or
  - collecting answers from the questionnaires.

- **Feasibility study:**
  Once the requirement analysis phase is completed the next SDLC step is to define and document software needs. This process conducted with the help of 'Software Requirement Specification' document also known as 'SRS' document. It includes everything which should be designed and developed during the project life cycle. There are mainly five types of feasibilities checks:

  - **Economic**: Can we complete the project within the budget or not?
  - **Legal**: Can we handle this project as cyber law and other regulatory framework/compliances.
  - **Operation feasibility**: Can we create operations which is expected by the client?
  - **Technical**: Need to check whether the current computer system can support the software
  - **Schedule**: Decide that the project can be completed within the given schedule or not.

- **Designing/Modeling**
  In this third phase, the system and software design documents are prepared as per the requirement specification document. This helps define overall system architecture. This design phase serves as input for the next phase of the model.
- **Coding /Development**
  Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process.
  In this phase, Developer needs to follow certain predefined coding guidelines. They also need to use programming tools like compiler, interpreters, debugger to generate and implement the code.
- **Testing**
  Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality of the entire system. This is done to verify that the entire application works according to the customer requirement. During this phase, QA and testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug and send back to QA for a re-test. This process continues until the software is bug-free, stable, and working according to the business needs of that system. An estimate says that 50% of whole software development process should be tested. Errors may ruin the software from critical level to its own removal
- **Implementation / Integration phase**
  Once the software testing phase is over and no bugs or errors left in the system then the final deployment process starts. Based on the feedback given by the project manager, the final software is released and checked for deployment issues if any.

- **Operation/Maintenance**
  Once the system is deployed, and customers start using the developed system, following 3 activities occur
  - **Bug fixing -** bugs are reported because of some scenarios which are not tested at all.
  - **Upgrade -** Upgrading the application to the newer versions of the Software
  - **Enhancement -** Adding some new features into the existing software
- **Documentation**
  Different types of documents are created through the whole software development lifecycle (SDLC). Documentation exists to explain product functionality, unify project-related information, and allow for discussing all significant questions arising between stakeholders and developers.

The main focus of this SDLC phase is to ensure that needs continue to be met and that the system continues to perform as per the specification mentioned in the first phase.

What are main types of main types of process models.?
Ans: Three main types of process models are:

- Linear process models – phases that happen sequentially, one after another
  - Waterfall Model
  - V-Model
- Iterative process models – phases that are repeated in cycles
  - Spiral model
- Parallel process models – activities that occur concurrently
  - Unified process model

What is Software Development Paradigm?
Ans: The software development paradigm helps developer to select a strategy to develop the software. A software development paradigm has its own set of tools, methods and procedures, which are expressed clearly and defines software development life cycle.

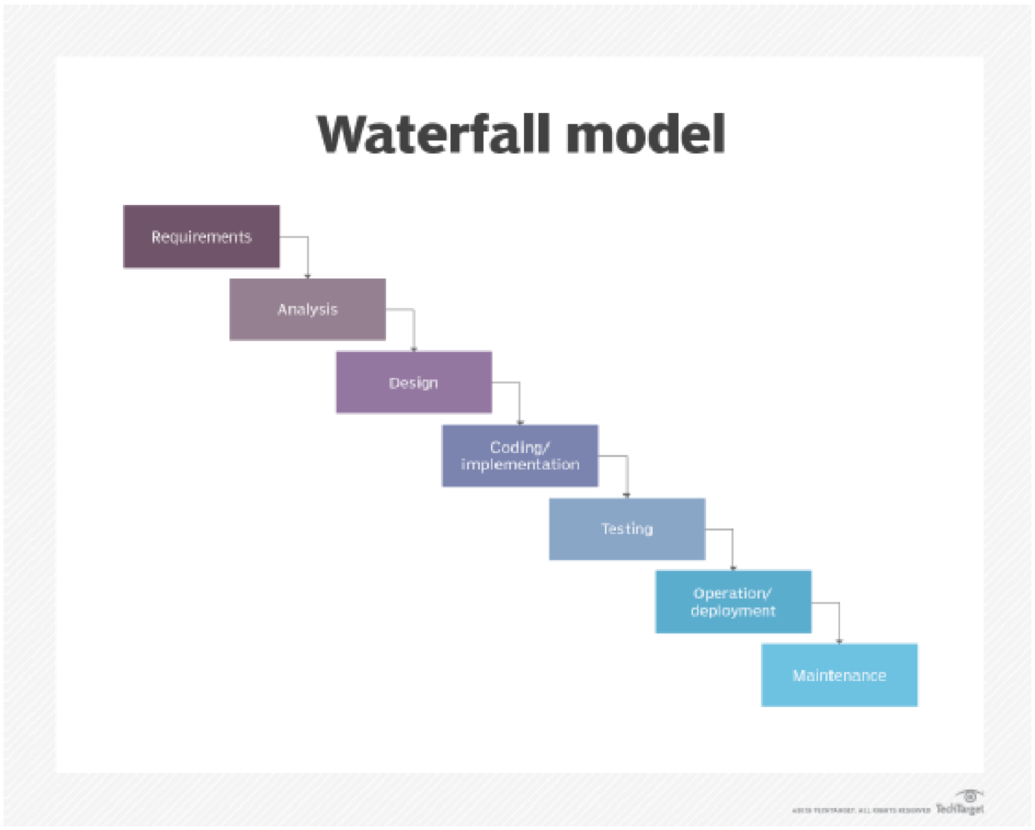
What is Software waterfall model?

Ans: The waterfall model is a classical model used in system development life cycle to create a system with a linear and sequential approach. It is termed as waterfall because the model develops systematically from one phase to another in a downward fashion. This model is named "Waterfall Model", because its diagrammatic representation resembles a cascade of waterfalls.

Here are the following steps-

1. **Requirement's analysis and specification phase**: The aim of this phase is to understand the exact requirements of the customer and to document them properly. Both the customer and the software developer work together so as to document all the functions, performance, and interfacing requirement of the software. It describes the "what" of the system to be produced and not "how."In this phase, a large document called Software Requirement Specification (SRS) document is created which contained a detailed description of what the system will do in the common language.

2. **Design Phase:** This phase aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language. It defines the overall software architecture together with high level and detailed design. All this work is documented as a Software Design Document (SDD).
3. **Implementation and unit testing:** During this phase, design is implemented. If the SDD is complete, the implementation or coding phase proceeds smoothly, because all the information needed by software developers is contained in the SDD. During testing, the code is thoroughly examined and modified. Small modules are tested in isolation initially. After that these modules are tested by writing some overhead code to check the interaction between these modules and the flow of intermediate output.
4. **Integration and System Testing:** This phase is highly crucial as the quality of the end product is determined by the effectiveness of the testing carried out. The better output will lead to satisfied customers, lower maintenance costs, and accurate results. Unit testing determines the efficiency of individual modules. However, in this phase, the modules are tested for their interactions with each other and with the system.
5. **Operation and maintenance phase:** Maintenance is the task performed by every user once the software has been delivered to the customer, installed, and operational.



When to use SDLC Waterfall Model?

Ans: Some Circumstances where the use of the Waterfall model is most suited are:

- When the requirements are constant and not changed regularly.
- A project is short
- The situation is calm

- Where the tools and technology used is consistent and is not changing
- When resources are well prepared and are available to use.

What are the Advantages and Disadvantages of Waterfall model?

Ans: Advantages of Waterfall model

- This model is simple to implement also the number of resources that are required for it is minimal.
- The requirements are simple and explicitly declared; they remain unchanged during the entire project development.
- The start and end points for each phase is fixed, which makes it easy to cover progress.
- The release date for the complete product, as well as its final cost, can be determined before development.
- It gives easy to control and clarity for the customer due to a strict reporting system.

Disadvantages of waterfall model:

- This model does not work well for large projects.
- We cannot go back at previous phase to change anything or any requirement.
- No changes can be made in the project.
- It can take a long time to complete the project.
- If customer does not satisfy with the project, then it is hard to change the requirements.
- High risk to make a project.

What is Iterative Waterfall Model?

Ans: Iterative Waterfall Model is the extension of the Waterfall model. This model is almost same as the waterfall model except some modifications are made to improve the performance of the software development. Iterative waterfall allows to go back on the previous phase and change the requirements and some modification can done if necessary. This model reduces the developer's effort and time required to detect and correct the errors.

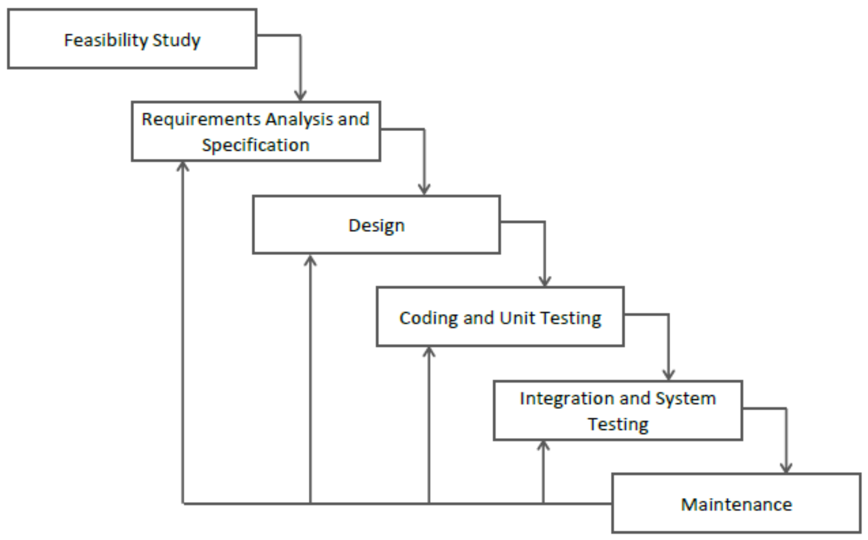
Phases of Iterative Waterfall Model:

- Requirement Analysis
- Feasibility Study
- Software Design
- Coding/Implementation
- Software Testing
- Software Deployment
- Software Maintenance

When to use Iterative Waterfall Model

- The requirement of the defined and clearly understood.

- New technology is being learned by the development team.
- There are some high-risk features and goals which might in the future.


Feasibility Study
Requirements Analysis and Specification
Design
Coding and Unit Testing
Integration and System Testing
Maintenance

Iterative Waterfall Model

Advantages of Iterative Waterfall Model: -

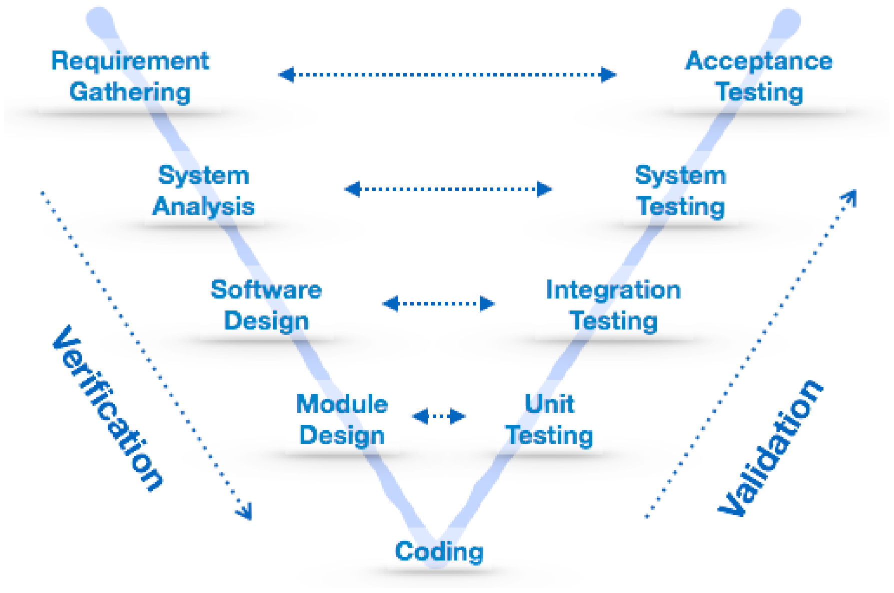
- Iterative waterfall model is very easy to understand and use.
- Every phase contains feedback path to its previous phase.
- This is an simple to make changes or any modifications at any phase.
- By using this model, developer can completer project earlier.
- Customer involvement is not required during the software development.
- This model is suitable for large and complex projects.

Disadvantages of Iterative Waterfall Model: -

- There is no feedback path for feasibility study phase.
- This model is not suitable if requirements are not clear.
- It can be more costly.
- There is no process for risk handling.
- Customer can view the final project. there is no prototype for taking customer reviews.
- This model does not work well for short projects.
- If modifications are required repeatedly then it can be more complex projects.

What is V model?

Ans: V- model is also called Verification and Validation model .This model is the extension of the Waterfall Model. In this model one phase for verification and other for validation and the coding phase joins the both phases verification and Validation .so that makes the V shape so this model is called V-model. This process starts from the top left i.e. verification phase towards the top right phases i.e. validation phases and follows a linear improvement like waterfall model.



Process of V Model

This model consists two main phases:

- **Verification phase:** Verification is the process to verify that the software product development phase to determine that specified requirements meet or not? In this phase, there is no need to execute the code for testing.
- **Validation phase:** Validation is the process to verify that the software product fulfills the customer requirements and expectations or not. In this phase, there is need of execution of the code.

Phases of Verification Stage:

- **Requirement Analysis:** In this phase, developers collect information from customers about the software.
- **System Design:** When the requirements are defined clearly then implement and design the complete hardware and communication setup for developing product and choose the programming language and databases.
- **System Architecture:** Architectural specifications are designed in this phase. The system design is splits further into modules taking up different working. This is also

called High Level Design (HLD). In this stage, communication and transformation of data between the internal modules and the outer world is clearly specified.

- **Module Design:** In this phase the system breaks down into small modules. The detailed design of modules is specified, it is also called the Low-Level Design (LLD).
- **Implementation/ Coding Phase:** This is the last phase of the V-Shape model. Module design is transformed into the code. The coding is done based on the coding principles and standards in a particular selected programming language.

Phases of Validation Stage:

- **Unit Testing:** Unit testing is a type of white box testing. These Unit Test Plans are executed to remove bugs at code level. Unit Test Plans are created during the module design phase.
- **Integration Phase:** In the integration testing, the integration test cases are executed which were developed in the High-level design phase. Integration testing is a testing process in which unit tested modules are integrated and evaluated. It verifies that the modules work together as expected or not.
- System Testing: System testing is done corresponds with the system design phase. It tests the functional and non-functional requirements and evaluate the whole system functionality and the communication of the system with external systems.
- **Acceptance Testing:** This testing is done to check that the delivered system meets user's requirement or not? Nonfunctional testing such as Load, Stress etc. are also done in this phase.

Advantages of V Model:

- It works very well for small project according to their requirement.
- This model is very simple, easy and useful.
- This is a high-quality model and all the phases are completed at once.
- This model is use to track the process of project management.
- This model saves a lot of time and efforts.
- Testing is starting at the initial phase so there is no issue of bugs.
- Client's requirements are not clearly specified.

Disadvantages of V Model:

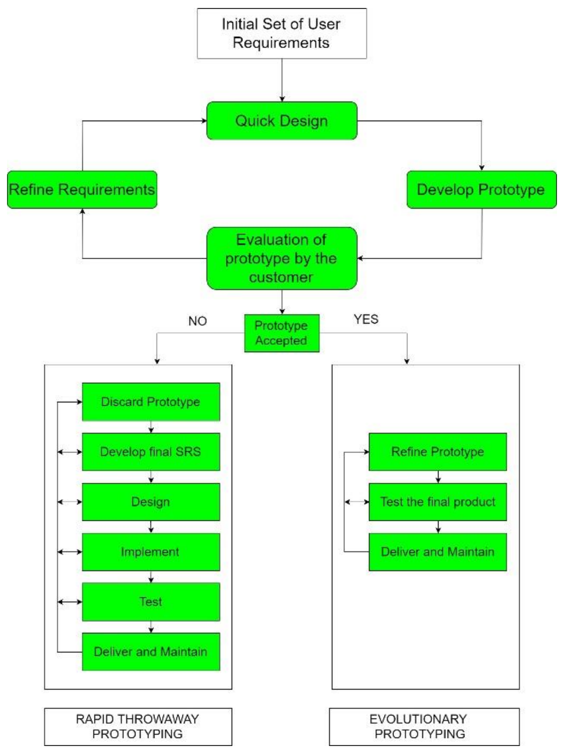
- This model cannot be use for large project.
- This model is not good if customer's requirements are not clear.
- There are lots of risk.
- This model is not easy for complex projects.
- Client have no prototype and involvement during the software development.
- This model contains less flexibility.
- It is hard to go back and alter the working of the system if new requirements are met.

What is PROTOTYPING Model?

Ans: Prototyping is defined as the process of developing a working replication of a product or system that has to be engineered. The Prototyping Model is one of the most popularly used Software Development Life Cycle Models (SDLC models). This model is used when the customers do not know the exact project requirements beforehand. In this model, a prototype of the end product is first developed, tested and refined as per customer feedback repeatedly till a final acceptable prototype is achieved which forms the basis for developing the final product.

There are Five types of models available:

- **Illustrative Prototype:** Mock-up, and Wireframe (Non-interactive) to illustrate the idea which is can be easily disposable. Often pen and paper used to sketch the idea.
- **Exploratory development:** Objective is to work with customers and to explore a final system from an initial outline specification. It allows the user to interact with system functionality (interactive mock-ups) and explore what is feasible.
- **Throw-away prototyping:** Objective is to understand the system requirements. Should start with poorly understood requirements. For example, first version of the product is just built to get a better understanding of building the second version.
- **Incremental prototyping:** When a product is built and released in increments, it is known an incremental prototype (working software). It use a triage system (the increments are build based on priorities, e.g. very important feature, important and optional features)
- **Evolutionary Prototype:** Develop early version of all the feature until they are fully mature in the final product. The feature evolves from initial functionalities to more mature versions.



Advantages –

- The customers get to see the partial product early in the life cycle. This ensures a greater level of customer satisfaction and comfort.
- New requirements can be easily accommodated as there is scope for refinement.
- Missing functionalities can be easily figured out.
- Errors can be detected much earlier thereby saving a lot of effort and cost, besides enhancing the quality of the software.
- The developed prototype can be reused by the developer for more complicated projects in the future.
- Flexibility in design.

Disadvantages –

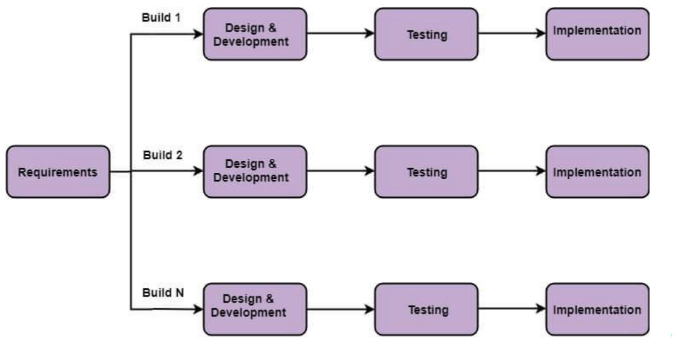
- Costly w.r.t time as well as money.
- There may be too much variation in requirements each time the prototype is evaluated by the customer.
- Poor Documentation due to continuously changing customer requirements.
- It is very difficult for developers to accommodate all the changes demanded by the customer.
- There is uncertainty in determining the number of iterations that would be required before the prototype is finally accepted by the customer.
- After seeing an early prototype, the customers sometimes demand the actual product to be delivered soon.
- Developers in a hurry to build prototypes may end up with sub-optimal solutions.
- The customer might lose interest in the product if he/she is not satisfied with the initial prototype.

Use –

The Prototyping Model should be used when the requirements of the product are not clearly understood or are unstable. It can also be used if requirements are changing quickly. This model can be successfully used for developing user interfaces, high technology software-intensive systems, and systems with complex algorithms and interfaces. It is also a very good choice to demonstrate the technical feasibility of the product.

What is INCREMENTAL Model?

Ans: Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle. In this model, each module goes through the requirements, design, implementation and testing phases. Every subsequent release of the module adds function to the previous release. The process continues until the complete system achieved. The evolutionary model is normally useful for very large products, where it is easier to find modules for incremental implementation.



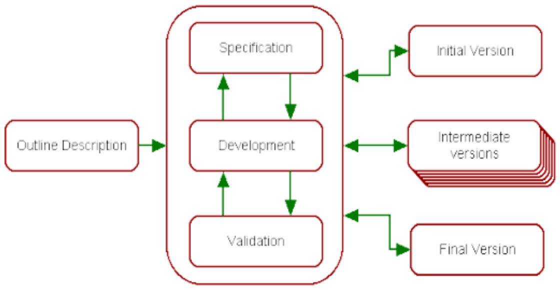


Fig: Incremental Model

When we use the Incremental Model?

- When the requirements are superior.
- A project has a lengthy development schedule.
- When Software team are not very well skilled or trained.
- When the customer demands a quick release of the product.
- You can develop prioritized requirements first.

Advantages of Incremental Model

- Large project: Evolutionary model is normally useful for very large products.
- Errors are easy to be recognized.
- Easier to test and debug
- More flexible.
- Simple to manage risk because it handled during its iteration.
- The Client gets important functionality early.

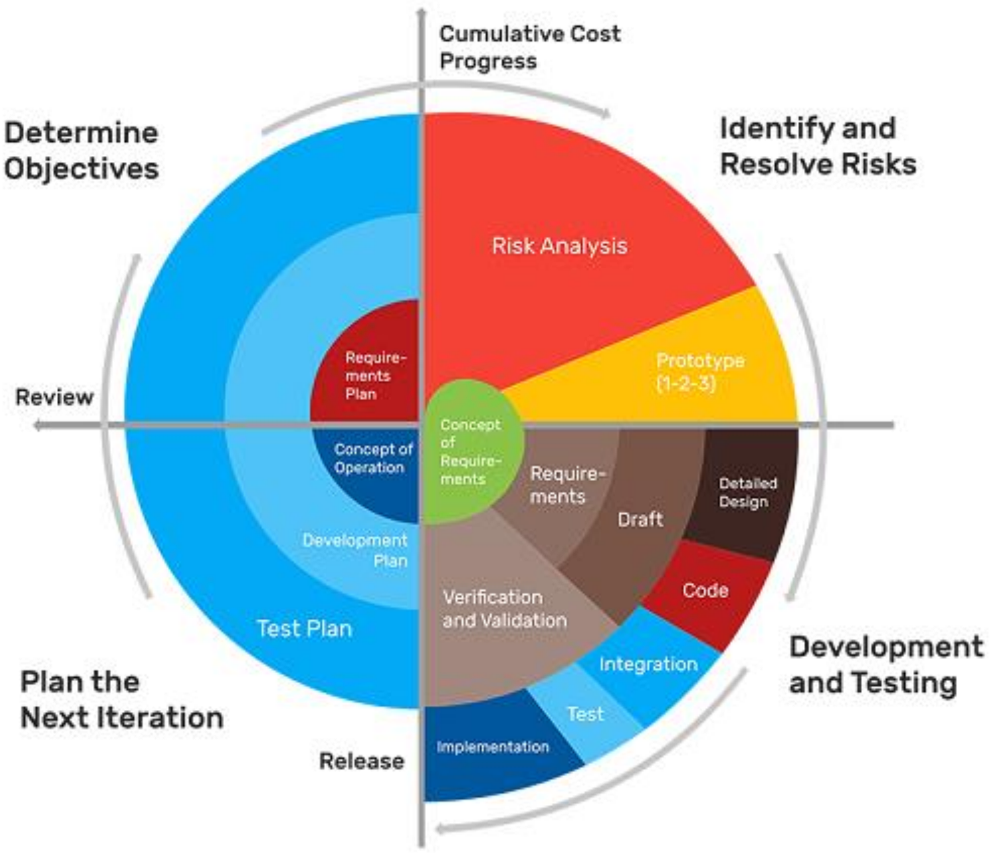
Disadvantages of Incremental Model

- Need for good planning
- Total Cost is high.
- Well defined module interfaces are needed.
- Can't see the progress / Early delivery

What is Spiral model?

Ans: Spiral model is a combination of both, iterative model and one of the SDLC model. It can be seen as if you choose one SDLC model and combine it with cyclic process (iterative model).

This model considers risk, which often goes un-noticed by most other models. The model starts with determining objectives and constraints of the software at the start of one iteration. Next phase is of prototyping the software. This includes risk analysis. Then one standard SDLC model is used to build the software. In the fourth phase of the plan of next iteration is prepared.



Spiral Model Methodology and its Phases

The whole development process is completed in the **4 phases**. The spiral model has four phases in a spiral:

- **Planning:** The requirements are collected from the clients. Feasibility study is done in this phase. It includes cost estimation, schedule, objectives are defined and other resources for the iteration to develop a software project.
- **Risk analysis:** In the second phase of development, risks are identified and find different solutions to remove risks. If any risk is found out during the phase, then alternate solutions are designed and implemented the best way among them.
- **Development and testing:** After risk analysis phase is completed, the software product is developed and tested at the end of each iteration so there is development and testing applied at same phase.
- **Evaluation:** In the last phase, the feedback is taken from the customer and evaluate the developed project after completion of each iteration. To carry on next iteration, evaluation phase is completed.

Advantages of Spiral Model:

- This model is good for large and complex projects.
- Updates are received to the customer at each iteration.
- Spiral model is suitable to change the requirements at any time, any phase.
- Risks are analyzed after each iteration.
- Documentation is clearly defined and understandable.

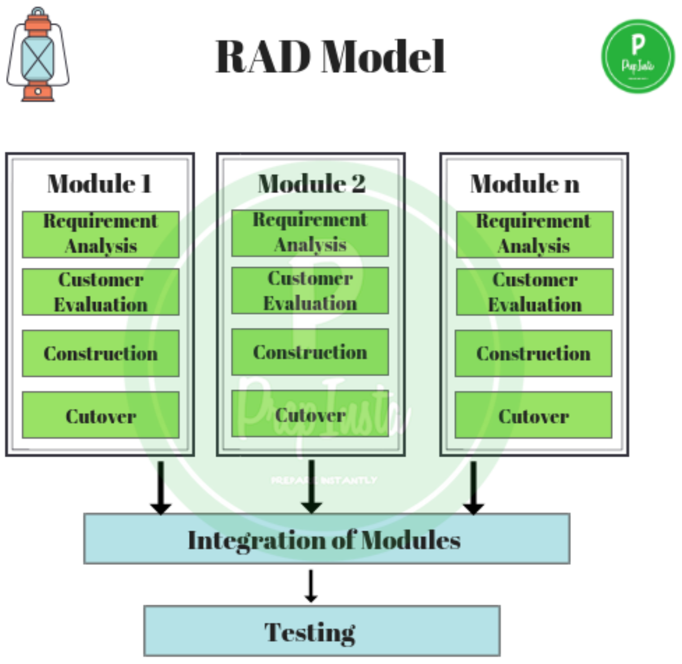
Disadvantages of Spiral Model:

- Spiral model is not good for small projects.
- This model is more complex and difficult to understand if a new employee is entered in the project development.
- It can be much expensive.
- Fast development and software is built at the SDLC.
- Not defined end points of the project, so it can take a long time to develop or iterations can be go infinitely

What is RAD(RAPID APPLICATION DEVELOPMENT)?

Ans: RAD Model is generally based on the prototype model and iterative approach.This model is used to completing the process of software product developing in a very short time.

The entire project is divided into various small modules and each module is allocated to different party to finish the working of the small modules. After that, all small modules are combined together to obtain the final project.



Process of RAD Model:

There are four phases in this model:

- **Requirement Analysis:** There are various approaches which is used in requirement planning like brainstorming, task analysis, form analysis, user scenario etc. This phase consists plan or designing of each module which contains data, methods and other resources.
- **Customer Evaluation:** In this phase, developer evaluates the customer satisfaction by delivering the prototype and taking the reviews from them. If the customer is satisfied then developer starts implementation.
- **Construction:** Prototype is refining and all the modification, correction and improvements is done in this phase. This phase helps us to convert the process and modules into the final working product.
- **Cut Over:** This is the last stage of the RAD model. In this phase, all the independent modules are evaluated separately. The tools and sub-parts of product makes the testing of the product very easy.

Advantages of RAD Model:

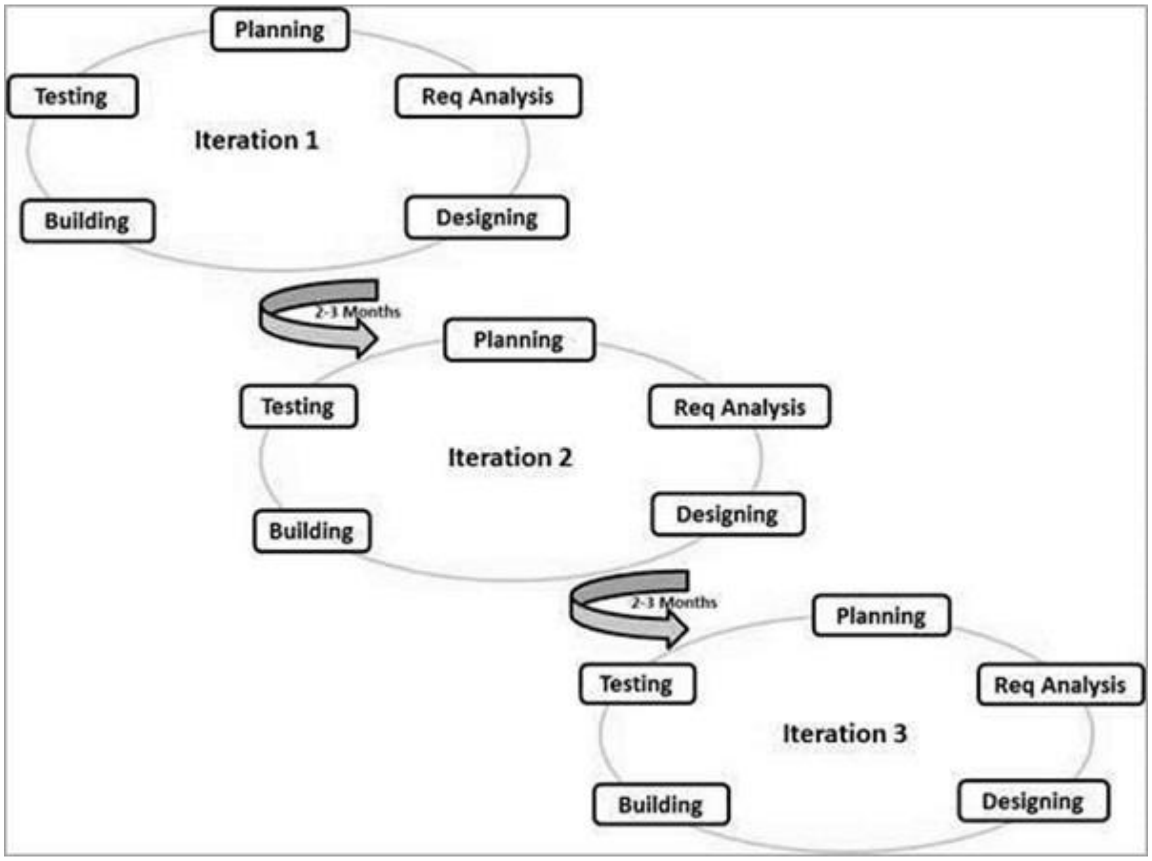
- RAD model completes the project in a short period of time.
- The progress and development of project can be check on various stages .
- This model uses the powerful techniques and tools.
- reduce cost because very less developers are needed.
- Prototype is delivered to the customer so the customer is satisfied.
- It has more flexibility and adaptability to acquire the new requirements.
- Reusability of the components is increased.
- fully functional system in 90 days, give or take 30 days

Disadvantages of RAD Model:

- Team leader must to do work with developers to complete the work on time.
- Customer involvement are needed.
- There is no reusable component are used to lead the failure of the project.
- This model works only when the requirements are clearly specified.
- This model can be more complex if prototype is refined again and again.
- RAD model is not suitable for the short projects.

What is Agile Model?

Ans: Agile model is the combination of iterative and incremental software development model. In the agile model, the requirements are break up into many parts, called iterations, and then developed incrementally. In this model, each iteration is planned, designed, implemented, tested and deployed to the customers to take the feedback.



AGILE METHODS

- Extreme Programming (XP)
- Scrum
- Dynamic Systems Development Method (DSDM)
- Feature-Driven Development (FDD)
- Crystal Methods
- Lean Development (LD)
- Adaptive Software Development (ASD)

Advantages Of Agile Model:

- Provides more flexibility.
- Agile model provides customer’s satisfaction at each iteration.
- No issue of bugs.
- Can change or modify the requirement very easily.
- It reduces development time of software product.
- This model saves time.

Disadvantages Of Agile Model:

- It takes a long time to complete a project.
- Agile model requires too much involvement of customers.
- This model does not define end point clearly.
- It can be more cost effective.
- Developers and testers have to work actively.
- This model is difficult to implement.
- More risk of maintainability.

What is EXTREME PROGRAMMING (XP)?

Ans: XP is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop a software.

extreme Programming (XP) was conceived and developed to address the specific needs of software development by small teams in the face of vague and changing requirements.

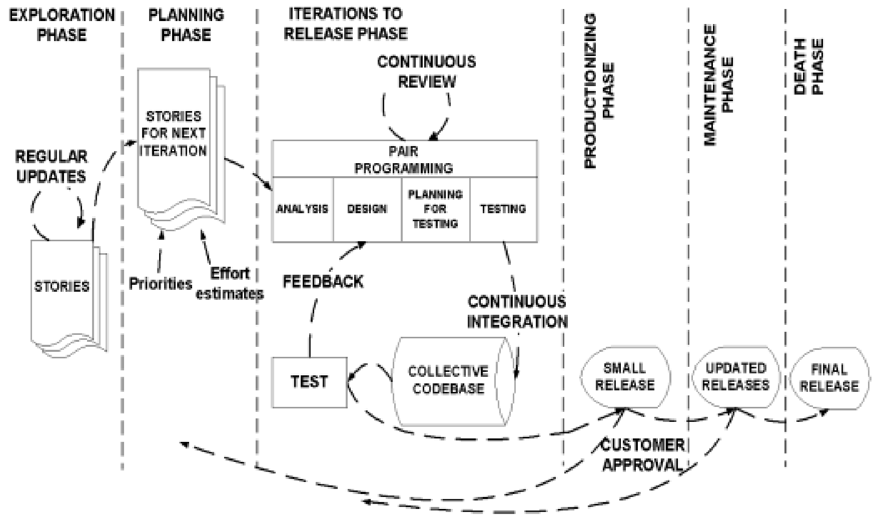
Extreme Programming is one of the Agile software development methodologies. It provides values and principles to guide the team behavior. The team is expected to self-organize. Extreme Programming provides specific core practices where −

- Each practice is simple and self-complete.
- Combination of practices produces more complex and emergent behavior.

Extreme Programming Values

Extreme Programming (XP) is based on the five values −

- **Communication:** XP has a culture of oral communication and its practices are designed to encourage interaction.
- **Simplicity:** Design the simplest product that meets the customer’s needs. An important aspect of the value is to only design and code what is in the current requirements rather than to anticipate and plan for unstated requirements.
- **Feedback:** The development team obtains feedback from the customers at the end of each iteration and external release. This feedback drives the next iteration.
- **Courage:** Allow the team to have courage in its actions and decision making. For example, the development team might have the courage to resist pressure to make unrealistic commitments.
- **Respect:** Team members need to care about each other and about the project.



The life cycle of XP consists of five phases:

- **Exploration:**
  - The customers write out the story cards that they wish to be included in the first release
  - At the same time the project team familiarize themselves with the tools, technology and practices they will be using in the project
  - He exploration phase takes between a few weeks to a few months, depending largely on how familiar the technology is to the programmers

- **Planning:**
  - User's stories are written
  - Estimate the effort of working with the user stories
  - Priorities are given to the user stories to be implemented
  - Release planning creates the release schedule
- **Iterations to Release:**
  - Includes several iterations of the systems before the first release
  - Each take one to four weeks to implement
  - The first iteration creates a system with the architecture of the whole system.
  - This is achieved by selecting the stories that will enforce building the structure for the whole system
  - The customer decides the stories to be selected for each iteration
  - At the end of the last iteration the system is ready for production
- **Productionizing:**
  - Requires extra testing and checking of the performance of the system before the system
  - can be released to the customer
  - New changes may still be found and the decision has to be made if they are included in the
  - current release
  - The iterations may need to be quickened from three weeks to one week
  - The postponed ideas and suggestions are documented for later implementation

- **Maintenance and Death:**
  - After the first release is productionized for customer use, the XP project must both keep
  - the system in the production running while also producing new iterations
  - Requires an effort also for customer support tasks
  - Development velocity may decelerate after the system is in production
  - May require incorporating new people into the team and changing the team structure

  - When the customer does no longer have any stories to be implemented
  - System satisfies customer needs also in other respects (e.g., concerning performance and reliability)
  - Necessary documentation of the system is finally written as no more changes to the architecture, design or code are made
  - Death may also occur if the system is not delivering the desired outcomes, or if it becomes
  - too expensive for further development

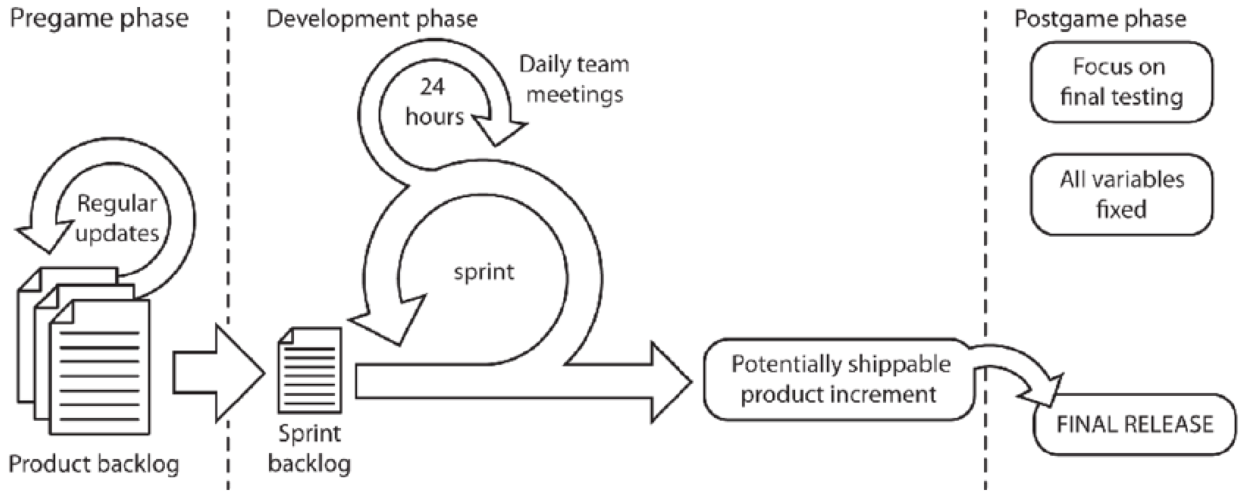
XP - PRACTICES:

- **Simple design:** The emphasis is on designing the simplest possible solution that is implementable at the moment
- **Testing:** Software development is test driven. Unit tests are implemented continuously
- **Refactoring:** Restructuring the system by removing duplication, improving communication, simplifying and adding flexibility
- **Collective ownership:** Anyone can change any part of the code at any time
- **Pair programming:**
  - Two people write the code at one computer.
  - One programmer, the driver, has control of the keyboard/mouse and actively implements the program. The other programmer, the observer, continuously observes the work of the driver to identify tactical defects (syntactic, spelling, etc.) and also thinks strategically about the direction of the work.
  - Two programmers can brainstorm any challenging problem. Because they periodically switch roles.
- **Continuous integration:** A new piece of code is integrated into the code-base as soon as it is ready.
- **40-hour week:** A maximum of 40-hour working week

What is Scrum?

Ans: Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value.

Scrum is not a process or a technique for building products; rather, it is a framework within which you can employ various processes and techniques. Scrum makes clear the relative efficacy of your product management and development practices so that you can improve.



SCRUM process includes three phases

- **Pre-game:**
    - **Planning**: A Product Backlog list is created, Definition of the system being developed, project team, tools, controlling issues, training needs and verification management approval.
    - **Architecture**: A design review meeting is held, Backlog items are identified, The high level design of the system including the architecture is planned
- **Development (game phase):** This phase is treated as a "black box" where the unpredictable is expected
- **Post-game:** This phase is entered when an agreement has been made such as the requirements are completed. In this case, no more items and issues can be found nor can any new ones be invented.

SCRUM PRACTICES

- **Product Backlog:** Defines the work to be done in the project
- **Sprint:** Sprint is the procedure of adapting to the changing environmental variables.
- **Sprint Backlog:** Sprint Backlog is the starting point for each Sprint. It is a list of Product Backlog items selected to be implemented in the next Sprint.
- **Sprint Planning meeting**: A Sprint Planning Meeting is a two-phase meeting organized by the Scrum Master.
- **Daily Scrum meeting:** Daily Scrum meetings are organized to keep track of the progress of the Scrum Team continuously and they also serve as planning meetings

- **Sprint Review meeting:** On the last day of the Sprint, the Scrum Team and the Scrum Master present the results (i.e. working product increment) of the Sprint to the management, customers, users, and the Product Owner in an informal meeting.

ROLES AND RESPONSIBILITIES

- **Scrum Master:** Scrum Master is responsible for ensuring that the project
- **Product Owner:** Product Owner is officially responsible for the project
- **Scrum Team**: Scrum Team is the project team that has the authority to decide on the necessary actions and to organize itself in order to achieve the goals of each Sprint.
- **Customer**: Customer participates in the tasks related to product Backlog items for the system being developed or enhanced.
- **Management:** Management is in charge of final decision making

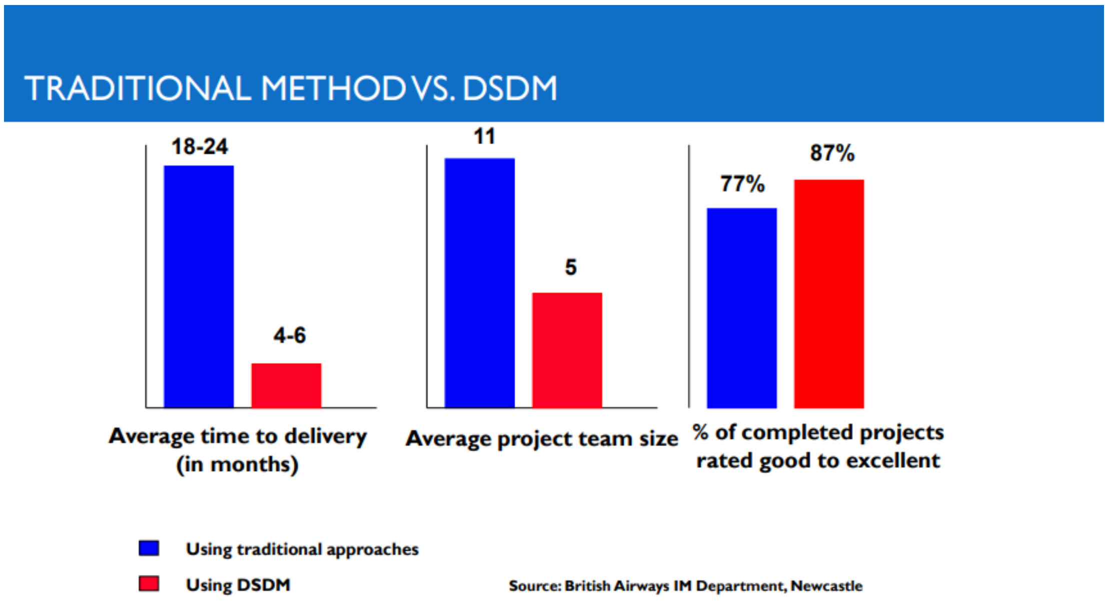
What is Dynamic Systems Development Method (DSDM)?

Ans: The Dynamic Systems Development Method (DSDM) is a public domain Rapid Application

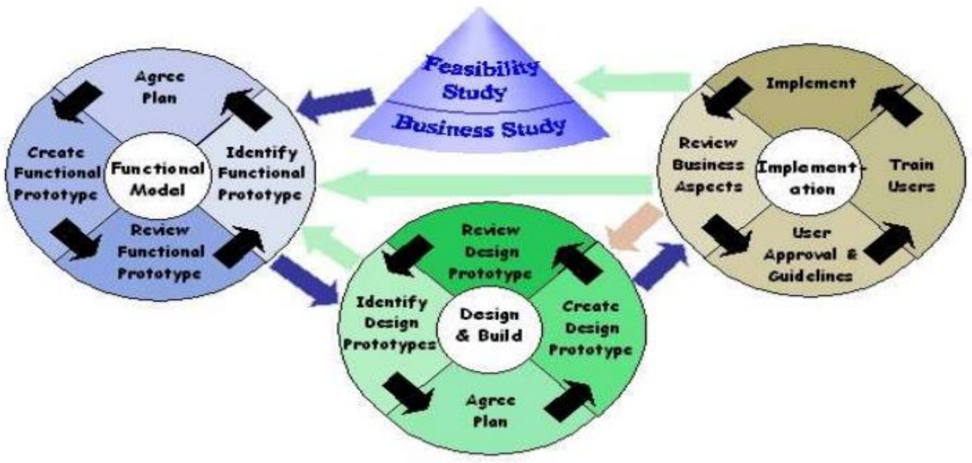
Development method which has been developed through capturing the experience of a large group of vendor and user organizations. It is now considered to be the UK's de-facto standard for RAD.

A fundamental assumption of DSDM is that nothing is built perfectly first time

80:20 Rule: assumes that a usable and useful 80% of the proposed system can be produced in 20% of the time it would take to produce the total system.



DSDM PROCESS VIEW



What is FEATURE DRIVEN DEVELOPMENT (FDD)?

Ans:

- FDD is an agile software development process
- FDD uses a short-iteration model
- FDD combines key advantages of other popular agile approaches along with other industry-recognized best practices
- FDD was created to easily scale to much larger projects and teams.
- FDD delivers the system feature by feature.

FDD Primary Roles

- Project Manager
- Chief Architect
- Class Owners
- Domain Experts
- Chief Programmers

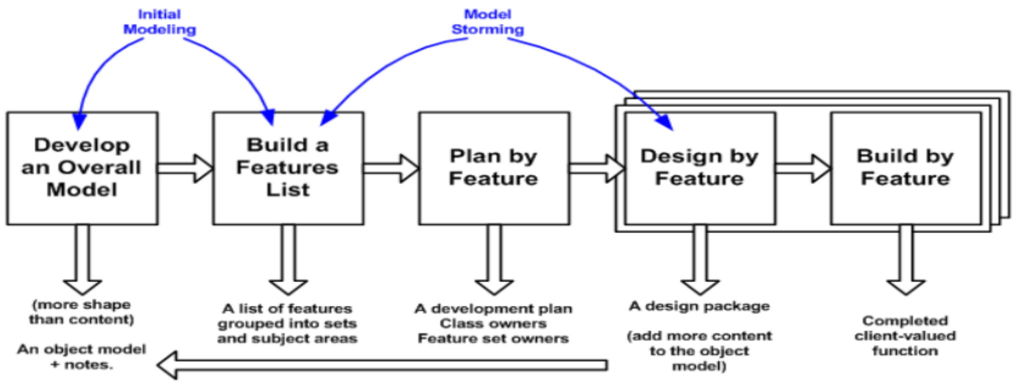
FDD Supporting Roles

- Language Guru (shared vocabulary)
- Tool smith (making tools for application)
- Tester
- Technical Writer (documentation)

FDD PROCESS

- Process #1: Develop an Overall Model
- Process #2: Build a Features List
- Process #3: Plan By Feature

- Process #4: Design By Feature
- Process #5: Build By Feature


Initial Modeling
Model Storming
Develop an Overall Model
Build a Features List
Plan by Feature
Design by Feature
Build by Feature
(more shape than content)
An object model + notes.
A list of features grouped into sets and subject areas
A development plan Class owners Feature set owners
A design package
(add more content to the object model)
Completed client-valued function