**Lecture 2 and 3**

SW development life cycle(sdlc )models

\_\_\_

# Notes from slides

## Inherent problem with Software Development

1. **Requirements are complex** -(clients doesn't know abt the functional requirements in advance)
2. **Requirements may be changing**(technology enablers introducing new possibilities to deal with nonfunctional requirements.)
3. **Frequent changes are difficult to manage**(identifying milestones and cost estimation are difficult)
4. **There is more than one software system**(-new system must be backward compatible with the existing system.

-need to distinguished between the system under-development and already released systems)

* **Popular SDLC models**

1. **Waterfall model**
2. **V-shaped model**
3. **Iterative Incremental**
4. **Spiral model**
5. **Prototyping model**
6. **Agile development**
7. Waterfall model

-called as Linear sequential Model

-whole application is developed in a sequential approach

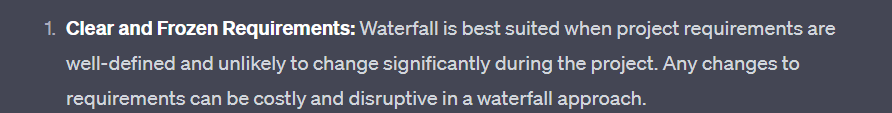
-each phase must be completed fully before the next phase begins

-does not define the process to go back to the previous phase to handle changes in the requirements

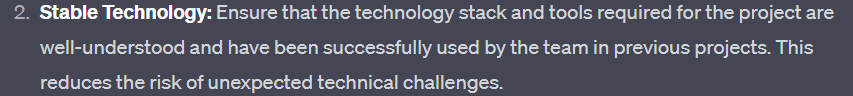
-provides structure to the inexperienced staff

* When to use this model??

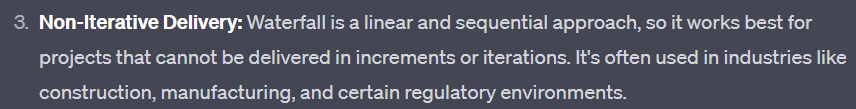
1. Requirements are clear and frozen



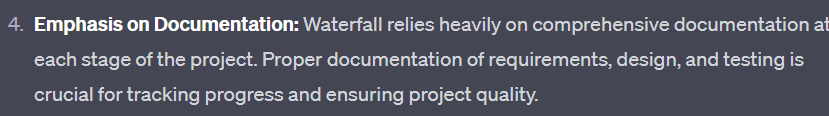
1. Technology is understood and used by the team in different projects



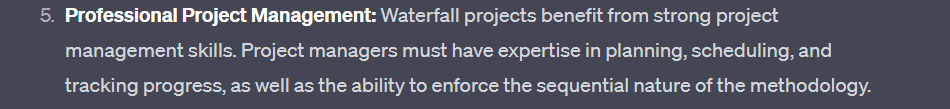
1. The projects cannot be delivered in iterative manner



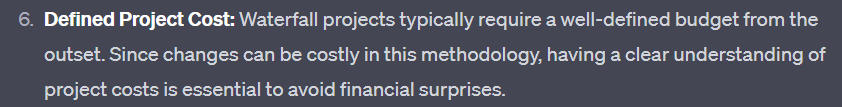
1. Documentation is essential



1. Professional project management skills



1. The project cost is defined



* Advantages

1. Very easy to explain the output of each phase and to the business user
2. Structured approach
3. Stages and activities are well defined
4. Easier for project managers to plan, schedule, utilize the resources and define the milestone easily
5. Validation, verification at each phase ensure early detection of error and misunderstandings at the same phase
6. Each phase has specific deliverables.

* DisAdvantages

1. Take full lifecycle to deliver workable solution
2. Very difficult to go back to any phase after it has finished
3. assume s the requirements are frozen without any changes
4. Little flexibility and adjusting scope is difficult and expensive
5. Requires more time for detailed plan upfront of the project, as requirements are clear,so its mandatory to deliver full detailed plan
6. Delays the testing phase which can discover issues in requirements,design and implementation .
7. V-shaped model

-extension of waterfall model, process are bent upwards

-major difference between waterfall and V-shaped is the early test planning in the V-shape model

* When to use this model??

-requirements are well defined,clearly documented and fixed

-product definition is stable

-tech is not dynamic and well understood by team

-no ambiguous or unclear requirements

-the project is short in nature

* Advantages

1. Simple and easy to use
2. Each phase has specific deliverables
3. Higher chance of success over waterfall due to the development of test plan early on during life cyle
4. Works well for where requirements are easily understood
5. Verification and validation of the product in the early stage of product development

* disAdvantages

1. Very inflexible ,like waterfall model
2. Adjusting scope is difficult and expensive
3. Software is developed in the implementation phase so no early prototype of the software are produced
4. Model cant provide clear path for the problems found in the testing phase
5. Costly and required more time, in addition to a detailed plan
6. Iterative Incremental model

-developed to overcome the weaknesses of waterfall model

-start with initial planning and end with deployment with cyclic interactions between them

-basic idea behind this to develop a system through repeated cycle(iterative) and smaller portion at a time(incremental)

-allowing software developer to take advantage of learning during the development of the earlier version /part of the system

-can be consisted of mini waterfall or mini V-shape model

* When to use this model??

1. Most requirements are known up-front but are to expected evolve over time
2. Requirements are prioritized
3. Need to get the basic functionality delivered fast
4. Project has lengthy development schedule
5. Has new technology
6. Domain is new to team

* Advantages

1. Can develop prioritized requirements fast
2. Customer gets important functionality early
3. Initial product delivery is faster
4. Lowers initial delivery cost
5. Each release is product increment,so that customer will have working product all the time
6. Customer can provide feedback to each increment thus avoiding surprise in the end
7. Requirement changes can be easily accommodate

* DisAdvantages

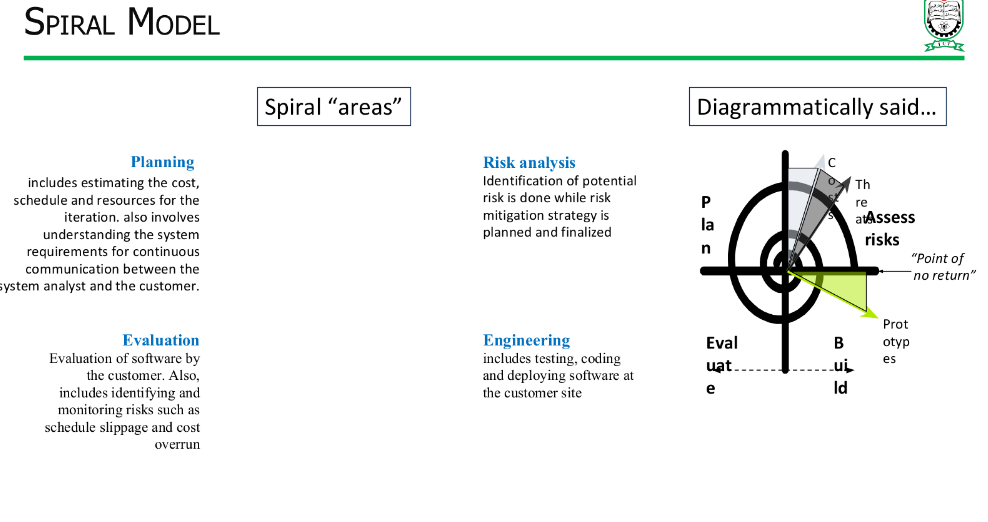
1. Requires effective planning of iterations
2. Requires effective design for inclusion of required functionality and provision for changes later
3. Requires early definition of complete and fully functional system to allow the definition of increments
4. Well defined modules interfaces are needed, as some are developed before the others are developed
5. Spiral model

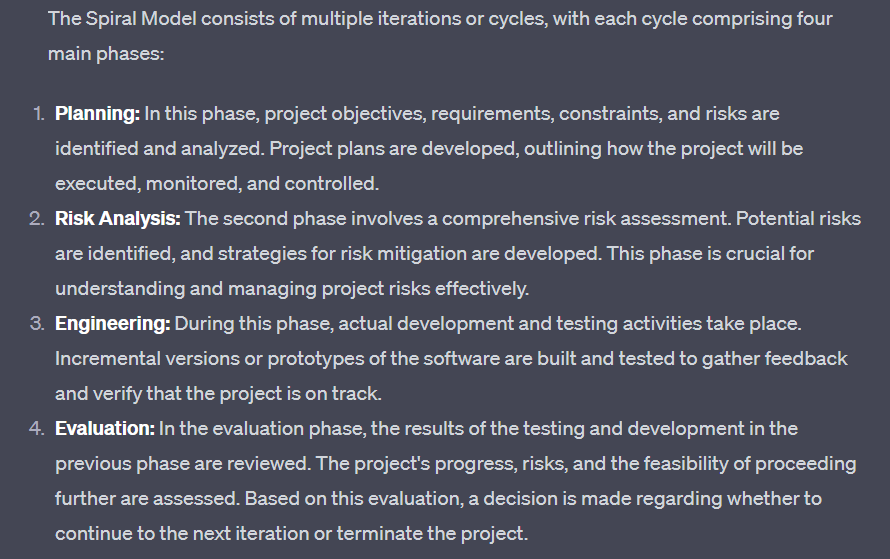
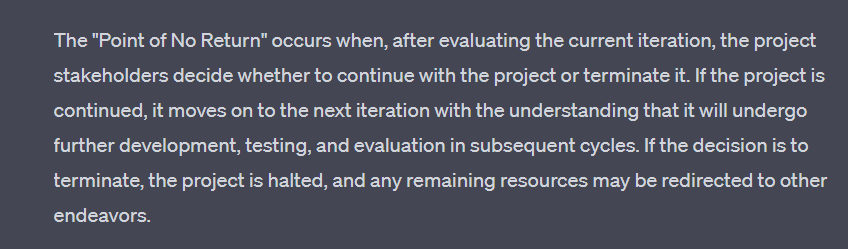
-combination of waterfall and iterative inc.

-each phase begins with design goal and ends with client reviewing the progress

-dev team start with small sets of requirements and goes through each development phase for those set of requirements.

-teams add functionality for the additional requirement in every increasing spirals until the application is ready for production phase





Steps of spiral model-

1. Define requirements
2. Initial new system design
3. Construct and evaluate initial prototype
4. Construct further refined prototype
5. Overall (system wide) risk assessment
6. Prototype assessment(of step 4) and possible development of further prototype
7. Repeat steps 1-5 until refined prototype meets user expectations
8. Construct the system(based of final refined prototype)
9. Test and maintain the system

* When to use this model??
* When project is large
* When releases are required to be frequent
* When Creation of prototype is applicable
* For medium to high risk project
* When requirements are complex and unclear
* When changes may require at any time
* Advantages

1. Additional functionality/changes can be done at later stage
2. Cost estimation becomes easy as prototype building is done in small fragment
3. Continuous or repeated development helps in risk management
4. Development is fast and features are added in systematic way
5. There is always space for customer feedback

* disAdvantages

1. Risk of not meeting the budget or schedule
2. It works best for large projects only also it need risk assessment expertise
3. For smooth operation, model protocol need to follow strictly
4. Documentation is more as it has intermediate phase
5. Not advisable for smaller project , it might cost them a lot
6. Prototyping model

-is the process of quickly putting together a working model(a prototype) in order to test various aspects o a designs,illustrate ideas or features and gather early user feedback

Types of prototyping

* throwaway/rapid prototyping(used when final system architecture is not well understood)- using little efforts to build prototype with minimum requirement analysis.once the actual requirements are understood, this proto is discarded and actual system is developed with much clear understanding of user requirements.
* Incremental prototyping-refers to building multiple functional prototypes of diff subsystem and integrating all available prototypes to form a complete system
* Evolutionary Prototyping-

Focuses on building basic software that can evolve over time.begins with simple prototype that is continuously expanded and refined as more feedback is gathered and requirements become clearer.final product emerges through this iterations.

* Extreme Prototyping-

Is used in web development domain, consisting of 3 sequential phases.

* Basic prototype with all the existing pages is presented in the html format
* Thn data processing is simulated using a prototype service layer
* Finally services are implemented and integrated to the final prototype
* When to use this model??

1. The requirements are unclear
2. Requirements are changing quickly
3. Can be successfully used in developing user interfaces,high technology software-intensive systems and systems with complex algorithms and interfaces
4. A very good choice to demonstrate the technical feasibility of the product

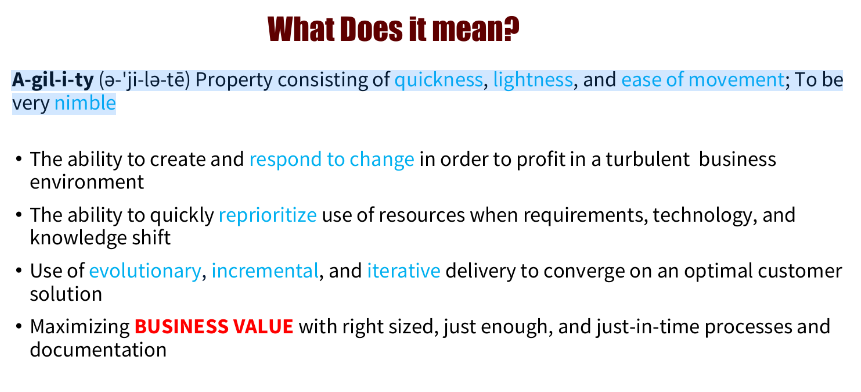
* Advantages

1. Errors can be detected in initial stage of software development process
2. Missing functionality can be identified which helps to reduce risk of failure
3. Customer satisfaction exists-> customer can feel the product at a early stage
4. As customer involved in early stage , hardly chance of rejection of software
5. Quicker user feedback helps to achieve better software development solutions
6. Its straightforward model easy understand
7. No need for specialized experts to build the model
8. Prototype serve as basis for deriving a system specification
9. Prototype helps to gain better understanding of customer needs
10. Proto may offer early training for future user of the software

* disAdvantages

1. May occur too much variation every time customer evaluate prototype
2. Poor documentation due to continuously changing customer requirements
3. Difficult for the developer to accommodate all the changes demanded by the customer
4. Uncertainty in determining number of iterations would be required
5. After seeing early prototype, customer may demand fast delivery of the actual product
6. Developer in hurry to build proto may end up with suboptimal solution
7. Customer might lose interest in product is shes not interested with the initial prototype

Do we need prototyping??(check slides for 2 points of interest)

1. Agile Development

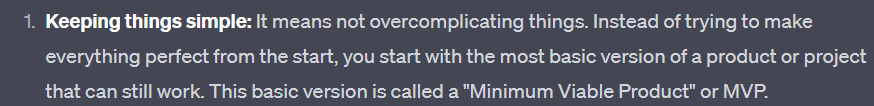
-is a set of practices, values, principles for software product development

– the core idea of agile development

* Adaptive– means teams and process should be flexible in the presence of “rapid fire change”
* iterative/incremental-produces working products in stages-a growing stage of completed and working software
* People oriented- team organization and processes will support good ppl(most imp ingredient to project success)

–agile fouses on lean and minimum viable products(MVPs)

That go through a number of iterations before anything is final



Manifesto for agile swd

1. **Individual and interactions** over processes and tools
2. **Working software** over comprehensive documentation
3. **Customer collaboration** over contract negotiation
4. **Responding to change** over following a plan

Agile manifesto principles;

1. Highest priority is to satisfy the customer through early and continuous delivery of software
2. Welcome changing requirements,even late in development
3. Deliver working software frequently(form a couple week to couple month)
4. Business ppl and developer must work together daily throughout the project
5. Build project around motivated individuals, providing support environment they need
6. Most effective method of conveying info to and within dev team is to face-to face conversation
7. Working software is the primary measure of progress
8. Agile process promote sustainable development
9. Continuous attention to technical excellence and good design enhances agility
10. Simplicity the art of maximizing the amount of work not done is essential
11. Best architecture, requirements and design emerge from self organizing teams
12. At regular intervals, the team reflects on how to become more effective, tunes and adjusts its behavior

* When to use this model??

-when new changes needed to be implemented and need to deliver the project in a short amount of time

-project involves iterative, cyclical processes in which incremental results will add value for project

-having budget flexibility

-trying to build something innovative that doesnt exist

-high product owner involvement

-organization doesnt have strict processes to follow, have the luxury of being able to working flexibly

* Advantages

1. Customer satisfaction by rapid continuous delivery of useful software
2. Ppl and interactions emphasized rather than process and tools
3. Customers, developers and testers interact continuously with eachother
4. Working software is delivered frequently
5. Face to face convo is best form of communication
6. Close daily co-operatipn between business ppl and developers
7. Continuous attention to technical excellence and good design
8. Regular adaptation to changing circumstances
9. Even late changes in requirements are welcomed

* disadvantages

1. In case of some large software deliverables , difficult to assess the effort required at the beginning of the sdlc
2. Lack of emphasis on necessary designing and documentation
3. Project can get take off from the track is the customer representative is not clear what outcome they want
4. Only senior programmer are capable of taking decisions required during development process.no place for newbie programmers unlcess combined with experienced resources

Why are SW industries adopting Agile development methods ????

For Success ….Organization success is about delivering value to its clients.

Go through

* Original xp methods based on 4 values - simplicity, communication, feedback, courage; has 12 supporting prwctices
* Agile development(lean) → helping businesses in all industries eleminate waste, improve processes and boost innovation.
* Software dev is line lean methodology
* Key concept of lean methodology→ continuous improvement, respect for ppl, lightweight leadership
* Kanban is highly visual workflow management that is popular among lean teams
* Kanban based on 3 basic principles-visualize what youll do today(workflow), limit the amount of work in progress(wip), enhance flow
* Agile (FDD) → iterative incre ental swd process and an agile method for developing software→ main purpose is to deliver tangible , working software repeatedly in timely manner
* Consist of 5 basic activities– development of an overall model, building of a feature list, planning by feature, designing by feature, building by feature
* Agile (DSDM) → dynamic system development method, is a framework made up of 8 principle, prioritizes schedule ad quality over functionality

-focus on the business need

-deliver on time

-collaborate

-never compromise quality

-build incrementally from firm foundation

-develop iteratively

-Communicate continuously and clearly

-demonstrate control

* breaks project in 4 different type of requirements-must should could wont have
* Agile(crystal)--> crystal methods are a family of methodologies → focused on-

1. Ppl
2. Interaction
3. Communication
4. Community
5. Talent
6. skill

* afile(scrum)Framework ppl cam address complex adaptive problems while productively and creatively delivering products of highest possible value
* Scrum is lightweight, simple to understand , difficult to master