**SOFTWARE DEVELOPMENT METHODOLOGİES**

**1. WATERFALL MODEL**

* Waterfall is the most traditional and ”old school” method.
* A new phase can’t begin until the previous phase has been completed.
* When a phase is finished and a new phase begins, the previous phase cannot be returned.
* Project owner can see the product after the project is completed.

**Phases of Waterfall Model**

1. **Requirements Analysis** **:** The aim of this phase is to understand the exact requirements of the customer and to document them properly. Software Requirement Specification (SRS) document is created in this phase.
2. **System Design:** aims to transform the requirements gathered in the SRS into a suitable form which permits further coding in a programming language. Software Design Document (SDD) is created in this phase.
3. **Implementation:** design is implemented. If the SDD is complete, the implementation or coding phase proceeds properly, because all the information needed by software developers is contained in the SDD.
4. **Testing:** The testing team tests the complete application and identifies any defects in the application.
5. **Deployement:** The product or application is deemed fully functional and is deployed to a live environment.
6. **Maintenance:** improve, update and enhance the final product if needed.

**Advantages of Waterfall Model**

* It is easy to use and manage.
* The requirements are well understood.
* It is easier to transfer project information.
* Better for small projects
* Roles remain as stable as possible
* Comprehensive documents are created.
* Testing phase is quick because of well defined requirement documents.
* Less amount of defect.

**Disadvantages of Waterfall Model**

* Change and innovation are difficult.
* Customer feedbacks and suggestions are ignored.
* There is no working product until the project is completed.
* Unable to handle unexpected risks.
* Documentation occupies a lot of time of developers and testers
* Small changes or errors that arise in the completed software may cause a lot of problems
* Preparing the comprehensive documents cost too much time.
* If the encountered defect is found to be caused by the analysis, the analysis, design, and test documents must be updated, and the code must be corrected, resulting in a significant waste of time and money.

**2. V MODEL**

V-Model also referred to as the Verification and Validation Model. In this, each phase of SDLC must complete before the next phase starts. It follows a sequential design process same as the waterfall model. Testing of the device is planned in parallel with a corresponding stage of development.

**Verification:** It involves a static analysis method (review) done without executing code. It is the process of evaluation of the product development process to find whether specified requirements meet.

**Validation:** It involves dynamic analysis method (functional, non-functional), testing is done by executing code. Validation is the process to classify the software after the completion of the development process to determine whether the software meets the customer expectations and requirements.

So V-Model contains Verification phases on one side of the Validation phases on the other side. Verification and Validation process is joined by coding phase in V-shape. Thus it is known as V-Model.

**The V model has the following advantages:**

* It is simple and easy to use.
* Test activities, such as planning and test design, carried out
* before coding activities . Thus, the model saves time.
* Because each stage is tested before moving on to the next stage, the V model has a better chance of success than the Waterfall model.
* Defects can be found at an early stage. Taking defects to the next stage is avoided.

**Disadvantages of the V model:**

* The method is quite strict, depends on strict rules.
* The software is developed in the development phase as in waterfall, so no early prototypes of the software are produced.
* If there is a change in the requirements at any stage, the test
* documents should be updated along with other documents.
* The V model can be used for projects where requirements are clearly defined.

**3. The Spiral Model**

Spiral Model is a risk-driven software development process model. This model is best used for large projects which involve continuous enhancements. There are specific activities that are done in one iteration (spiral) where the output is a small prototype of the large software. The same activities are then repeated for all the spirals until the entire software is built.

In this model, we create the application module by module and handed over to the customer so that they can start using the application at a very early stage. And we prepare this model only when the module is dependent on each other. In this model, we develop the application in the stages because sometimes the client gives the requirements in between the process.

**The different phases of the spiral model are as follows:**

**Determine objectives and find alternate solutions –** This phase includes requirement gathering and analysis. Based on the requirements, objectives are defined and different alternate solutions are proposed.

**Risk Analysis and resolving –** all the proposed solutions are analyzed and any potential risk is identified, analyzed, and resolved.

**Develop and test:** This phase includes the actual implementation of the different features. All the implemented features are then verified with thorough testing.

**Review and planning of the next phase –** In this phase, the software is evaluated by the customer. It also includes risk identification and monitoring like cost overrun or schedule slippage and after that planning of the next phase is started.

**Spiral Model Advantages**

* Perfect for projects that are large and complex.
* Because of its risk handling ability, the model is best suited for projects which are very critical.
* This model supports the client feedback and implementation of change requests (CRs)
* Since customer gets to see a prototype in each phase, so there are higher chances of customer satisfaction.

**Spiral Model Disadvantages**

* Because of the prototype development and risk analysis in each phase, it is very expensive and time taking.
* Risk analysis requires high expertise
* It is **not** suitable for a simpler and smaller project because of multiple phases.
* It requires more documentation as compared to other models.
* Project deadlines can be missed since the number of phases is unknown in the beginning.

**4. Prototype Model**

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.

**Phases of Prototype Model**

1. Requirement Identification
2. Design Stage
3. Build the Initial Prototype
4. Review of the Prototype
5. Enhancement of Prototype

**5. Agile Methodologies**

**The Agile methodology** is a method of project management that divides a project into phases. It involves ongoing collaboration with stakeholders as well as continuous improvement at each stage. When the work begins, teams go through a cycle of planning, execution, and evaluation. Collaboration is essential, both with team members and project stakeholders.

**Agile project management** is based on the idea that a project can be continuously improved upon throughout its life cycle, with changes made quickly and responsively.

Due to its flexibility, adaptability to change, and high level of customer input, agile is one of the most popular approaches to project management. Customer input refers to suggestions, enhancement requests, recommendations or other feedback provided by customer.

**Agile project management is not a singular framework** — rather, it can be used as an umbrella term to include many different frameworks. Agile project management can refer to terms including Scrum, Kanban, Extreme Programming (XP), and Adaptive Project Framework (APF).

1. **Scrum Methodology**

Scrum is an agile development methodology used in the development of Software based on an iterative and incremental processes. Scrum is adaptable, fast, flexible and effective agile framework that is designed to deliver value to the customer throughout the development of the project. The primary objective of Scrum is to satisfy the customer’s need through an environment of transparency in communication, collective responsibility and continuous progress. The development starts from a general idea of ​​what needs to be built, elaborating a list of characteristics ordered by priority (product backlog) that the owner of the product wants to obtain.

**What is Product Backlog?**

It is a list of requirements for the project. The product owner receives requirements from customers and organizes them in priority order. According to changing needs, the product owner can add or remove items from the product backlog. As a result, the change can be easily integrated into the project at any stage.

**Product Backlog Item (PBI):** The name given to each requirement in the product backlog.

**Backlog Grooming/Refinement(arıtmak, saflaştırmak) :** Product Backlog grooming (also known as backlog refinement) is a recurring event or meeting where product managers, product owners, and the rest of the team review and re-prioritize backlog items. The primary goal of product backlog grooming is to keep the backlog up to date and to ensure that backlog items are ready for future sprints. Regular product backlog grooming sessions ensure that the right stories are prioritized and that the product backlog does not turn into a black hole.

**What is Sprint Backlog?**

The product backlog is divided into sprint backlogs. The Sprint Backlog is a list of tasks the team plans to complete in a Sprint. During the Sprint Planning meeting, team members collaborate to create the list, which is obtained from the Product Backlog.

While determining the user stories to be included in the Sprint Backlog, the order of priority in the product backlog is considered. They do not put the low priority ones in the first sprints.

The amount of user stories to be included in the Sprint Backlog varies according to the points given by the team.

**What is Sprint?**

Scrum teams work in short timeframes known as sprints. Sprints can be as short as one week or as long as one month. Sprints are held one after the other with no breaks in between to maintain a consistent project timing.

Each sprint begins with a plan (sprint planning) and concludes with a review of the work completed (sprint review) and an additional look back at how the team worked together (sprint retrospective).

During each sprint, the entire scrum team collaborates to complete one or more increments of a larger overall product or project. Each completed increment must be potentially releasable, which means it could theoretically be delivered as-is. In other words, each increment must be thoroughly tested and approved.

**Scrum Ceremonies**

* **Sprint Planning:** The project manager gathers the team to determine how much work can be completed in one sprint. It is critical that there is enough work to fill the time frame, but not too much. Inadequate work planning can derail a project and result in budget and timeline overruns. Too much planning can result in team burnout and missed deadlines.
* Sprint planning is the first activity in each Sprint.
* The Product Owner and Development team will meet here to decide which Product Backlog Items (PBI) will be included in the Sprint.
* While the Product Owner has the authority to prioritize any PBI for inclusion in the Sprint, the Development team is encouraged to respond, raise issues, and provide feedback as needed.
* After learning about the speed, resources, and factors that can affect the time and resources available, the development team estimates how many PBIs they can add in the Sprint.
* The Sprint Planning Meeting results in a realistic and achievable Sprint Goal and Sprint Backlog that everyone agrees on.
* Sprint Planning answers the following questions;

1) What will we do?

2) How will we do?

* If there are any actions that affect the Sprint Planning Meeting among the decisions made at the previous Sprint Retrospective Meeting, they must be carried out.
* A good plan is the foundation of everything. If you start well, you have a good chance of finishing it successfully. A good Scrum Master ensures that the Scrum Team plans well.
* **Daily Scrum:** Daily Scrum meetings, also known as stand-up meetings, ensure that sprints run on time and that all team members are kept informed when problems arise. Sprint stand-ups are typically 15 minutes long and require each team member to discuss what they've accomplished since the last meeting, what they'll work on before the next meeting, and if any obstacles are in the way.
* Daily stand-up meetings should be brief. More in-depth meetings should be scheduled separately from stand-ups.
* What did we do yesterday? What will we work on today? Have we encountered any difficulties or obstacles? are discussed.
* Answers to questions should be brief and clear.
* If we consider a Development Team of seven people, each member will have 128 seconds to speak. Each question requires 43 seconds to answer.
* **Sprint Review/Demo:** After completing a sprint, the project manager holds a sprint review meeting with all team members and stakeholders to demonstrate sprint outputs, determine what was accomplished and what wasn't, and review project forecasts. Untested or incomplete work is not displayed, but is instead saved for the planning round of the next sprint.
* It is usually held on the last day of the Sprint and gives Stakeholders (customers, management and everyone involved) an opportunity to demonstrate the work done.
* In addition to showing the features of work produced during the sprint, you also get helpful feedback where you can add the Product Backlog that can help guide work for future sprints.
* The owner of the SprintReview Meeting is the Product Owner.
* It is necessary to emphasize to the stakeholders that "the product is their product." It is to inform them that it will be determined in this meeting whether what you will tell them and what you intend to show corresponds to (örtüşmek) what they expect from you.
* Stakeholders can see the working product and tell what new features they want to see added to it, as well as what quality, visuality, and competence the features should have.
* **Sprint Retrospective/Retro:** The final step in the sprint management process is the sprint retrospective. This takes place after the sprint review and before the next sprint planning session. This collaborative session allows team members to discuss accomplishments and challenges during the previous sprint so that processes can be altered, if needed. The goal is to fix one thing at a time and make small, incremental changes from sprint to sprint.
* Retrospective meetings are held to review the previous Sprint in order to improve business processes in the next Sprint and to determine "how can we perform better?" The development team, scrum master, and product owner all attend these meetings.
* It takes place right after the "sprint review" meeting and is the sprint's final meeting.
* The Scrum team discusses what can be improved for future Sprints and how they should be performed.
* They evaluate the difficulties they encountered, as well as the ideas and updates that contributed to their improvement.

**Scrum Team**

**Chicken Roles:** They are people who are not actively involved in the operation of Scrum. Like customers, vendors.

**Pig Roles:** Those who are involved in the Scrum process, that is, the people who do the main work in the project. These;

1) Product Owner (PO)

2) Scrum Master

3) Development Team

**Scrum Team Roles**

Scrum has three roles: product owner, scrum master and the development team members.

1. **Product Owner:**

* provides communication between the development team and the customer.
* represents the stakeholders in the team.
* defines the features of the project.
* creates a product backlog according to the priorities of the project.
* The Product Backlog should be accessible and understandable by everyone.
* In each Sprint, PO decides which user stories will be included in the sprint.
* organizes sprint review meetings. He is the owner of sprint review meetings.

**PO is not;**

* Not a manager of the Development Team and Scrum
* In technical matters, the Development Team is self-managed.
* Although the PO may have a technical background, this does not give the PO the authority to decide who does what job.
* PO cannot assign jobs or tasks!
* PO cannot interfere(müdahale etmek) with how the job is performed!

**2. Scrum Master**

* The Scrum master knows the rules, theories and practices of Scrum well and is the person responsible for the team's implementation of these rules. He is not the manager of the team. The Scrum master removes the situations that disturb the Team and prevent them from working efficiently.
* Scrum master is a member of the scrum team, is collaborative, protective, helpful, problem solver, determined, available and knowledgeable.
* Scrum master guides and coaches the team, helping them overcome the obstacles encountered.
* Scrum master strives to increase the harmony and communication between the team members.
* Scrum master facilitates scrum rituals and meetings such as sprint planning, sprint retrospective, daily scrum, and sprint review.
* Scrum master should ensure that the team can work in a safe and trouble-free environment, and should provide many services, including individual coaching, to team members when necessary.
* The Scrum master manages the relationship with the product owner. The Scrum master makes sure that the work set by the product owner is understood by everyone on the team, assists the product owner by finding techniques to organize the backlog effectively.

**3. Development Team**

It is a self-organizing, cross-functional team of people who are at the core of the Scrum development team structure. It is the team that is responsible for building the actual product increment and meeting the sprint goal.

Self-organizing: *No one (not even the Scrum Master) tells the Development Team how to turn Product Backlog into Increments of potentially releasable functionality.*

Cross-functional: *they should possess all the skills necessary to complete the sprint goal. They shouldn't need help from people outside the Scrum team to finish their sprint work.*

* Responsibility for technical issues belongs to the development team.
* The product owner and scrum master can determine the priority of the work to be done by the development team, but they do not interfere with what they will do.
* Scrum recognizes no titles for Development Team members, regardless of the work being performed by the person.
* Scrum recognizes no titles for Development Team members, regardless of the work being performed by the person.
* Regardless of the roles and abilities of the people within the team, the entire team is responsible for the completion of a job to the outside.
* The development team is obliged to put a product backlog item that it pulled from the product backlog as a working code snippet in front of the product owner.
* The development team should be “Self Organized”. Doesn't need a job description or a job tracker to get the job done.

1. **Kanban Methodology**

Kanban methodology is an agile method that aims at continuous improvement, flexibility in task management, and enhanced workflow. With this illustrative approach, the progress of the whole project can be easily understood in a glance.

Kanban method revolves around the kanban board. It is a tool that visualizes the entire project to track the flow of their project. Through this graphical approach of Kanban boards, a new member or an external entity can understand what’s happening right now, tasks completed and future tasks.

**Principles of Kanban**

1. **Start With What You Are Doing Now**

Kanban must be applied directly to current workflow. Any changes needed can occur gradually over a period of time at a pace the team is comfortable with.

1. **Agree To Pursue Incremental, Evolutionary Change**

Kanban encourages you to make small incremental changes rather than making radical changes that might lead to resistance within the team and organization.

1. **Respect Current Roles, Responsibilities, And Job-Titles**

Kanban does not impose any organizational changes by itself. So, it is not necessary to make changes to your existing roles and functions which may be performing well.

1. **Encourage Acts Of Leadership At All Levels**

Kanban encourages continuous improvement at all the levels of the organization and it says that leadership acts don't have to originate from senior managers only.

**Kanban Practices**

1. **Visualize the workflow**: In software development, work is quite abstract, for it does not have a physical form or it can be touched. A Kanban board is used to visualize the work using a physical board and post-it notes. It is an intuitive way to see how works flow from one stage to another.
2. **Limit Work-in-Progress (WIP):** WiP stands for Work in Progress. It is a key term in the Kanban method. Usually, people would think the number of WiP is the bigger, the better. However, any work in progress means that it is not completed. Therefore, it is a waste of time and effort. The final output is much valued, so to limit the work in progress is to make sure the work actually flows from "upstream" to "downstream" in a team and no blockage to affect the team productivity. Balance, a value of Kanban, is closely related to this practice.
3. **Manage flow :**Lead-time is everything for managing the flow of work. It is the time from when the work is started till the end of the work done. Flow is the measurement of work is done in a steady and predictable mode. The staff and work are allocated reasonably.
4. **Make policies explicit:** Transparency is valued in the Kanban method, so making policies simple and clear is a good way to improve team collaboration. As Kanban is a pull system, the team has to set the pull-criteria. Only when the work downstream is done, the upstream work can be started. It is like a pull system, rather than a push system. Then, fix the WiP limit which is immutable afterward. Other process rules can be set accordingly.
5. **Improve feedback loops:** On a Kanban board, feedback is not displayed. Feedback is done via meetings which are cadences of Kanban. A brief introduction of Kanban cadences can be found HERE.
6. **Improve collaboratively, evolve experimentally:** The first part of this practice, improve collaboratively, is how to drive the change in your team, while the second part, evolve experimentally, is how you implement the change. PDCA is often used by the Kanban community. It is a four-step iterative way to improve continuously. PDCA stands for plan-do-check-act. The team needs to do implement change together and cooperate.

**Scrum vs Kanban**

