* Requirement specification 🡪 defining what the system should do
* Design 🡪 Design the organization
* Implementation
* Testing
* Maintenance
* Evolution 🡪 Changing the system in response to changing customer needs
* Validation 🡪 Check it does what the customer wants
* Project management

PLAN-DRIVEN: All the process activities are planned in advance

AGILE: Planning is incremental, do plan step by step, no big plan. Easier to change the process to reflect changing customer requirements.

**SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)**

Waterfall

* Ciddi güvenli sistem (muharebe uçağı)
* Plan-driven
* Separate and distinct phases of specification and development

Diagram

Description automatically generated

We cannot go back, that’s why we say waterfall. You have to go to top.

Hata varsa da en sona kadar gitmelisin ve başa dönüp tekrar etmelisin.

It’s easy to manage and track processes 🡪 good side

It’s better to use waterfall for large projects.

Incremental Development

* Çok etkileşimli
* Specification, development and validation are interleaved
* Plan-driven or AGILE

Diagram

Description automatically generated

Process is not visible 🡪 bad side

When new increments are added system structure tends to degrade (Refactoring (içeriğinde herhangi bir değişiklik yapmadan bir programın iç yapısını yeniden düzenleyerek iyileştirme) is solution)

Integration and Configuration

* The system is assembled from existing configurable components
* Plan-driven or AGILE
* Software reuse

Configure application system

application system available



Software discovery

Adapt components

Requirements refinement

Requirement specification

Integrate system

components available

Software evaluation

Develop new components

**PROCESS ACTIVITIES**

Requirements engineering process:

Diagram

Description automatically generated

Diagram

Description automatically generated

**Agile methods**: reduce overheads

Principles

* Customer involvement
* Incremental delivery
* People not process (Team skills recognized)
* Embrace change
* Maintain simplicity

**Extreme Programming (XP)**

Agile development started with extreme programming

Increments delivered in each 2 weeks

Test the every build

Diagram

Description automatically generated

XP Practices

* Incremental planning
* Small releases
* Simple design
* Test-first development (Test after every change)
* Refactoring (Code improvement)
* Pair programming (Developers work in pair)
* Collective ownership (Pairs work on all areas)
* Continuous integration
* On-site customer

XP is not easy to integrate

User stories 🡪 we get all requirements by stories

Test driven 🡪 Write tests before code

Story point 🡪 Fazlaysa olayın yapılma süresi fazla

Diagram

Description automatically generated

Product backlog: Yapılacaklar listesi

User requirements 🡪 for users  
System requirements 🡪 for developers, more detailed

Stakeholders 🡪 Any person or organization affected by the system in soe way

* patients
* doctors
* nurses
* …

Agile 🡪 We don’t usually get system requirements document

Functional requirements 🡪 How system behaves for particular inputs. User should be able to search the appointmens lists for all clinics.

Non-Functional requirements:

* Security, …
* Constraints on the services, development process, standards, etc.
* Doğru zamanda yanıt veriyor mu?
* Veriler güvenli mi?
* Sistem çoktüğünde 3 dk içinde geri çalışabiliyor mu?
* Yedekleme sağlanıyor mu?
* Ölçülebilir olmalı ------> Easy to use can not be a requirement

Requirements should be consistent and complete

Metrics:

* Screen refresh time
* Size
* Training time
* Availability time
* Time to restart after failure
* Number of target system

Requirement engineering processes:

* Requirements elicitation
* Requirements analysis/specification
* Requirements validation
* Requirements management

User scenarios should include:

* Description of starting situation
* Description of normal flow of events
* Description of what can go wront
* Information about other concurrent activities
* Description of the state when the scenario finishes

Use case diyagramda stakeholderlar vardır, yaptıkları şeyler 🡪 use cases