Szoftver management

# A fejlesztési folyamat részei

## Programkészítés lépései

1. **Specifikáció**
   1. A megrendelő kitalálja, hogy milyen programot szeretne.
   2. Megbeszéli a programozóval, hogy mit kell tudnia a programnak.
   3. **A tisztázott igények precíz leírását nevezzük specifikációnak.**
2. **Tervezés**
   1. A feladathoz megfelelő adatszerkezeteket és algoritmusokat kell találni vagy fejleszteni.
   2. Megtervezendő továbbá a program felhasználói felülete és gondolni kell a jövőbeli bővíthetőségre is.
3. **Kódolás**
   1. A kész terveket egy választott programozási nyelven kódoljuk.
   2. A forráskódból egy fordítóprogram hozza létre a futtatható programot.
4. **Tesztelés**
   1. Két szempont alapján kell tesztelni a programot:
      1. **Minden esetben működik-e és hatékony-e.**
5. **Hibajavítás**
   1. A felismert hibák kijavításához új specifikáció, új terv és újrakódolás lehet szükséges.
6. **Dokumentáció**
   1. Ha később is szeretnénk fejleszteni a programot és érteni akarjuk a működését, akkor egy dokumentációt kell készíteni hozzá.
   2. Dokumentációnak **két szintje** van:
      1. **Felhasználói** és **Fejlesztői**

## Szoftvertervezés egy lehetséges menete

1. **Deployment diagrammok**
   1. A rendszer és milyen körülmények között lesz használva.
2. **Behavioral diagrammok**
   1. Milyen funkciókat kell a rendszernek tudnia?
   2. Use-case + Activity + Wireframes diagrammok
   3. Megrendelővel közös tervezés
3. **Structural diagrammok**
   1. A működést milyen modulokkal, milyen felbontással lehet megoldani.
   2. Component + Sequence (Class diagrammok)
   3. Entity-relations diagram (Adatbázis tábla struktúrák)
4. **Időtervezés**
   1. Gantt diagram
5. **Implementálás**

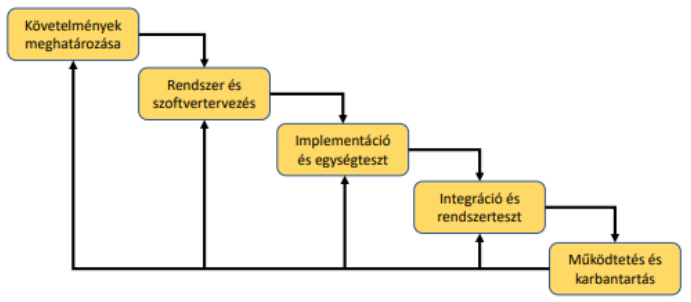
# Fejlesztési folyamat közbeni problémák

* **Követelmények változása:** Ügyfelek igényei fejlesztéskor folyamatosan változhatnak.
* **Hiányos tesztelés:** Hibás alkalmazásokat vagy váratlan problémákat okozhatnak.
* **Hibás kommunikáció**
  + Fejlesztők és az ügyfelek között, de lehet a fejlesztői csapaton belül is rossz kommunikáció.
* **Technikai korlátok**
  + Akadályozhatja a fejlesztés menetét, minőségét az, hogy nem állnak rendelkezésre a megfelelő eszközök, technológiák a fejlesztés során.

# Klasszikus modellek

## Vízesés modell

* Akkor hasznos, ha a követelmények jól ismertek és csak nagyon kis változások lehetségesek a fejlesztéskor.
  + Kevés üzleti rendszernek vannak stabil követelményei.
  + Főleg nagy rendszerek fejlesztésekor használják, ahol a fejlesztés több helyszínen történik.
* **Problémái**
  + Minden a specifikáció minőségétől függ.
  + Későn lát a megrendelő működő programot.
  + Kezdeti bizonytalanságot nehezen kezeli.
  + Tesztelés szerepe nem eléggé hangsúlyos.
* **Fázisai**



## V-modell

* Azért nevezik V-modellnek, mert két szára van: **Fejlesztési** és **tesztelési** szár.
* Vízesés modell kiegészítése teszteléssel.
  + Először végre kell hajtani a fejlesztés lépéseit, ezután jönnek a tesztelés lépései.
  + Ha valamelyik teszt hibát talál, akkor vissza kell menni a megfelelő fejlesztési lépésre.
* Szigorú dokumentálást követe és nem küszöböli ki a vízesés modell problémáit.

A képen szöveg látható

Automatikusan generált leírás

# Agilis módszerek

## Scrum

* **Vezetőség az erőforrásokat adja**: Emberi, anyagi és infrastrukturális
* Adott Scrum Team közvetlenül kommunikál a megrendelővel.
* Teljesen az ő felelősségük a projekt esetleges kudarca.

### Scrum szerepkörök

* **Product Owner**
  + Csapat tagja, megrendelő érdekeit képviseli.
  + Megszervezi a rendszeres demókat.
  + Nem fejlesztő, de érti a gazdasági folyamatokat.
  + Product Backlog kezelője.
* **Scrum Master**
  + Csapat tagja, egy hagyományos projektmenedzser szereppel egyezik a feladata.
  + Felügyeli a folyamatokat.
  + Konfliktusokat kezel, akadályok elhárítását irányítja.
  + Meetingeket ő szervezi, ő vezeti.
  + Scrum csapat feje.
* **Scrum Team**
  + 5-9 fő alkotja, **szükséges:** Elemző, fejlesztő, tesztelő
  + Ők végzik a tényleges fejlesztést.
  + Felelősségük, hogy egy sprintre bevállalt feladatokat elvégezzék.
  + **Fejlesztői fokozatok**: Junior > Medior > Senior
  + **Tesztelő**
    - Mindenki a saját kódját teszteli.
    - Unit tesztek után peer review más fejlesztővel.
    - Van manuális teszt, amit nem a fejlesztők végeznek.

### Scrum entitások

* **User Story, Task**
  + Specifikációból eredő feladat.
  + Taskokra bomlik szét, ezeket veszik magukra a fejlesztők.
* **Sprint**
  + 1-4 hét hosszú fejlesztési szakasz.
  + Addig jönnek újabb sprintek, amíg a Product Backlogból el nem tűnnek a User Storyk.
  + A sprint vége egy kész szoftver.
* **Product Backlog**
  + Elkészítésre váró Story-k gyűjtőhelye.
  + Product Owner tartja karban, tehát prioritásokat rendel a story-khoz.
  + **ROI:** Return of Investment = üzleti érték / ráfordítás.
* **Sprint Backlog**
  + A Product Backlog, de csak az adott sprintre bevállalt storykra szűrve.
* **Burn down/up chart**
  + Napi eredmény diagram és megmutatja, hogy a csapat mennyire tartja az eredeti ütemtervet.
* **Impediment**
  + Akadály, ami a munkát hátráltatja, valamilyen munkahelyi probléma.
  + Scrum Master feladata eljárítani.

### Indító meetingek

* **Sprint (pre)grooming**
  + Architektek + Scrum Master
  + Storyk ellenőrzése, pontozása (nehézség/idő > Fibonacci számokkal)
* **Sprint planning**
  + Team bevállal storykat.
  + Figyelembe veszik a pontozást.

### Folyamatos meetingek

* **Daily stand-up**
  + Minden nap ugyanakkor és az egész team részt vesz benne.
  + Szóban, kb 15 perc és egyéb eszközök nélkül.
  + Megbeszélik, hogy ki mit csinált vagy ma mit fog csinálni, és kinek mi a problémája.
* **Sprint refinement**
  + Hetente 1 maximum (3 hetes sprintben maximum 2 alkalom)
  + Storyk/Taskok áttekintése
  + Gyors hibaelhárítások
  + Nem része a Scrumnak, de alkalmazzák.
* **Egyéb meeting**
  + Csapat részhalmaza egyeztet.

### Lezáró meetingek

* **Sprint review**
  + Team + Scrum Master + Product Owner
  + Eredmények bemutatása, Product Owner dönti el, hogy sikeres-e a sprint.
  + Kimaradt Storyk/Taskok a következő sprintre mennek.
* **Sprint retrospect**
  + Review után tartják, Team + Scrum Master
  + Személyes tapasztalatok/javaslatok megvitatása.
  + Céges szintű problémákat a Scrum Master továbbítja felfelé.
  + Személyes konfliktusok megbeszélése.

## Kanban (jelzőtábla vagy hirdetőtábla.

* Lean módszer emberek csoportos munkájának menedzselésére és fejlesztésére.
  + **Lean**: Vállalatirányítási módszer, aminek célja, hogy a vállalat minél gazdaságosabban állítsa elő a termékeit, szolgáltatásait.
* Kiegyensúlyozza az igényeket és elérhető munkakapacitást.
* Munkafolyamat megjelenítésére használják a Kanban táblát.
* A munkafolyamatok elvégzését úgy ütemezik, ahogy a dolgozók kapacitása megengedi és nem engedik, hogy a munkafolyamatok sürgőssége határozza meg a munka időbeosztását.

### Kanban célja

* Célja, hogy létrejöjjön egy olyan vizuális folyamatmenedzsment-rendszer, ami segít meghozni azokat a döntéseket, hogy mit, mikor és hogyan gyártsanak.

### Hol használják?

* Maintenance/support jellegű feladatoknál
* HR-folyamatokra
* Értékesítésre