2. Agilis modellek

# Agilis módszerek

# Extrém programozás

* XP, Extreme programming
* Inkrementális fejlesztés + OOP + Agilis elvek
* 4 aktivitásból áll
  + **Planning** (analízis)
  + **Design** (tervezés)
  + **Coding** (implementáció)
  + **Testing** (tesztelés)
* **Alapvető értékei**
  + **Kommunikáció:** Közös gondolkodás, aktív alkotói légkör
  + **Egyszerűség:** Core funkciók először, apróságok utoljára (core first, shine last)
  + **Visszacsatolás:** Tesztelők is, megrendelő is, fejlesztők is
  + **Bátorság:** Merjünk új technológiákhoz nyúlni
  + **Tisztelet:** Egymás munkája iránt maximálisan

## XP modellje

A képen diagram, tervezés látható

Automatikusan generált leírás

### Planning

* User story-k leírása
* Apró release-ek preferálása
* Minden iteráció elején analízis
* Minden nap elején stand-up meeting

### Design

* Egyszerűség, CRC kártyák használata, ne hozzunk előre semmit

### Coding

* Standard-ek betartása, pair-programming, nincs túlóra

### Testing

* Unit-test mindenre, ahol minden teszt legyen zöld a release előtt
* Elfogadási tesztek (acceptance tests) gyakori futtatása

## XP előnyök és hátrányok

* **Előnyök**
  + Jól kezeli a követelmények változását
  + Költséghatékony
  + Nagyobb megrendelői elégedettség
  + Alacsony kockázati tényező
* **Hátrányok**
  + Minőség erősen függ a megrendelő közreműködő szándékától
  + Nem terv centralizált, hanem kód-centralizált
    - Nehéz az újrafelhasználása később, nehéz később elővenni
  + Nem fektet elég hangsúlyt a dokumentáció készítésére

## 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.

# Hagyományos modellek és agilis modellek összehasonlítása

* Mindegyik jó a maga módján
* **Szekvenciális modellek**, akkor jók, ha jól át vannak gondolva a dolgok, tehát nem lesznek változtatások és biztosan jól értünk mindent.
  + **Terv alapú**
* **Agilis modellek,** a megrendelőknek nincsen terve, kb el tudja mondani, hogy mit szeretne, vagyis **Adaptív**, alkalmazkodik a különböző környezeti változásokhoz.
* **Iteratív modellek**, a kettő között helyezkednek el.

A képen szöveg, képernyőkép, sor, Téglalap látható

Automatikusan generált leírás

A képen kör, képernyőkép, szöveg, diagram látható

Automatikusan generált leírás

* **Hagyományos módszereknél**, a specifikáció fix, idő és költség változó, így a minőség rovására megy.
* **Agilis módszereknél,** a funkciók legyenek változóak, így az időt és költséget jobban lehet rögzíteni, ezzel biztosan kész lesz időre és akkor a minőséget magasabb szintre fogja emelni.

## Melyik a sikeresebb?

* Összességében az Agilis módszerek sikeresebben, mivel az elkészült projektek magasabb szinten, időre tudnak elkészülni.
* A hagyományos módszereknél így sokkal nagyobb az esélye, hogy a projekt „elbukik”.