



**Eötvös Loránd Tudományegyetem  
Informatikai Kar**

# Szoftverttechnológia

---

## 1. fejezet

# A szoftverfejlesztési folyamat

---

**Giachetta Roberto**

**A jegyzet az ELTE Informatikai Karának 2016. évi  
jegyzetpályázatának támogatásával készült**



(Marc Andreessen)

# A szoftverfejlesztési folyamat

## Szoftverfejlesztés

- A szoftverek nélkülözhetetlen alkotóelemei a modern világnak
  - számos célt szolgálhatnak
  - különböző felépítésűek, működési elvűek
  - megvalósításuk módja jelentősen eltérhet
- A szoftverekben sok hiba található, a szoftverfejlesztési munkák nagyrésze kudarcba fullad, ennek okai:
  - egyre nagyobb számban, egyre összetettebb szoftverekre van szükség
  - alacsonyak az elvárások a szoftverekkel szemben
- Nagy szükség van a *professzionális szoftverfejlesztésre*

# A szoftverfejlesztési folyamat

## Jelentős szoftverhibák

- *Intel*, Pentium processzor (1994): hibás lebegőpontos számítás
- *ESA*, Ariane-5 (1996): adat túlsordulás
- *NASA*, Mars Climate Orbiter (1998): mértékegység tévesztés
- *General Electric*, észak-amerikai áramkimaradás (2003): kiéheztetés
- *World of Warcraft*, Corrupted Blood Incident (2005)
- *OpenSSL*, Heartbleed (2012): adatszivárgás
- *Apple*, goto fail (2014): hibás elágazás kezelés
- *Toyota*, ETCS gyorsulás szabályozás (2014): verem túlsordulás
- *Steam*, felhasználói fiók törlése (2015): útvonal ellenőrzés hiánya
- *Boeing*, 787 Dreamliner (2015): adat túlsordulás

# A szoftverfejlesztési folyamat

## Minőségi mutatók

- A szoftvereknek megfelelő színvonalon kell biztosítani az elvárt funkciókat, amit a szoftver *minőségi mutatóival* (*quality characteristics*) írhatunk le
  - *karbantarthatóság* (*maintainability*): módosíthatóság, továbbfejleszthetőség lehetőségei
  - *megbízhatóság és biztonság* (*dependability and security*): meghibásodások valószínűsége, támadásokkal szembeni védelem, sebezhetőségi pontok
  - *hatékonyság* (*efficiency*): erőforrások használata, korlátai, válaszidő, skálázhatóság
  - *használhatóság* (*acceptability*): érthetőség, használat elsajátítása, ergonómia

# A szoftverfejlesztési folyamat

## A szoftvertechnológia

- Egy szoftvernek, mint terméknek gyártási technológiára van szüksége, amely garantálja a program funkcióit, minőségét, költségét és határidejét
- *A szoftvertechnológia feladata szoftverek rendszerezett, felügyelt, minősített fejlesztése, működtetése és karbantartása*
  - *a szoftver a program(ok), dokumentáció(k), konfiguráció(k), valamint adatok együttese*
- A szoftverek többsége nagy méretű, nagy bonyolultságú programrendszer, amely
  - rendszerint csapatmunkában készül
  - hosszú élettartamú, karbantartást és bővítést igényel

# A szoftverfejlesztési folyamat

## Szoftvertechnológiai projekt

- A szoftver fejlesztésének folyamatát *projektnek*, előállításának felügyeletét *projektmenedzselésnek* nevezzük
- A projektért felelős személy a *projektmenedzser* (*project manager*), aki
  - biztosítja, hogy a szoftver megfelel az előírt minőségnek, és elkészül a megadott határidőre a költségkereten belül
  - szervezi, irányítja, ütemezi a projektben részt vevő csapat munkáját, és biztosítja a szükséges hardver és szoftver erőforrásokat
  - garantálja a módszerek és szabványok alkalmazását
  - gondoskodik a projekt dokumentáltságáról

# A szoftverfejlesztési folyamat

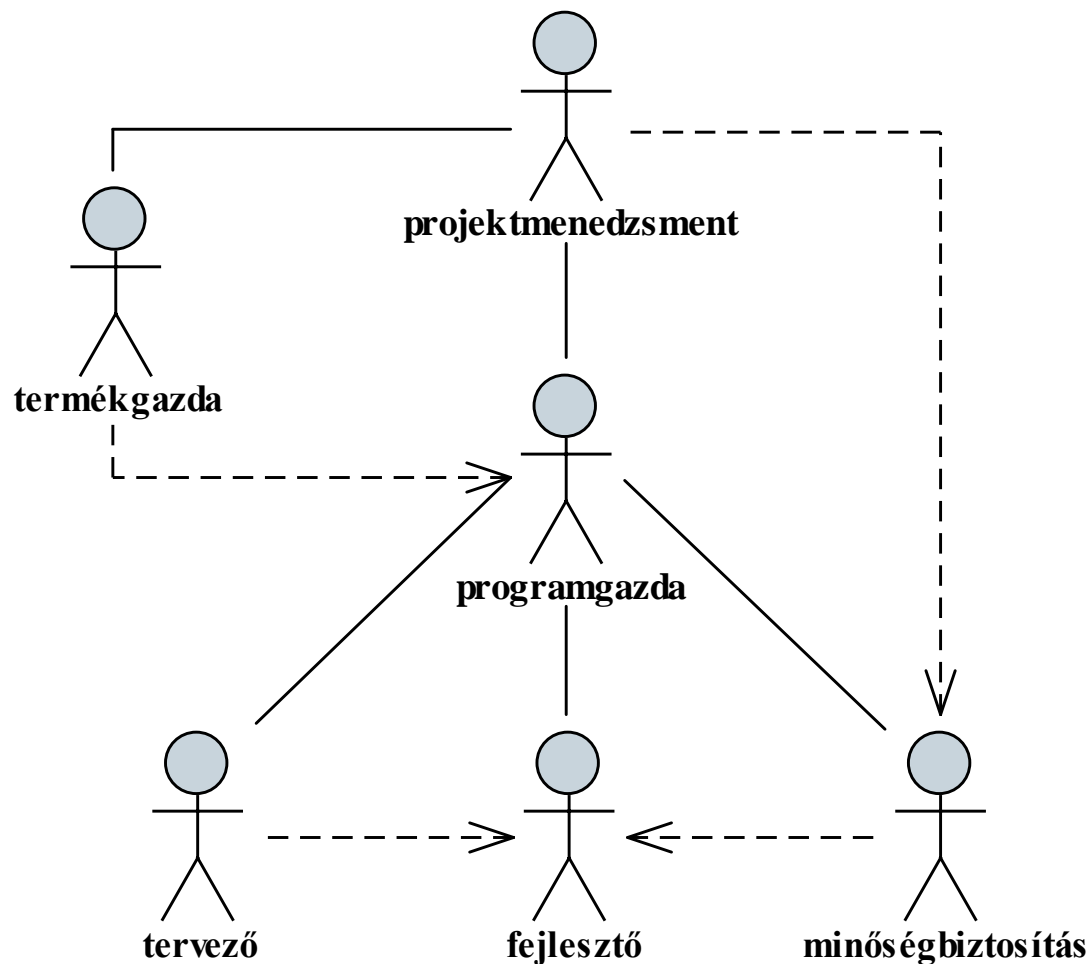
## Szoftvertechnológiai projekt

- A szoftverfejlesztési csapatnak számos további tagja lehet, akik különböző szerepeket töltenek be, pl.:
  - *termékgazda (product management)*: üzleti folyamatok, prioritások és elfogadási feltételek kezelése
  - *programgazda (program management)*: fejlesztés ütemezése, feladatok elosztása és követése
  - *tervező (architect)*: szoftver magas szintű tervének elkészítése, technikai döntések kezelése
  - *fejlesztő (developer)*: szoftver implementációja
  - *minőségbiztosítás (quality assurance)*: tesztelés tervezése, magvalósítása, minőségi kritériumok ellenőrzése



# A szoftverfejlesztési folyamat

## Szoftvertechnológiai projekt



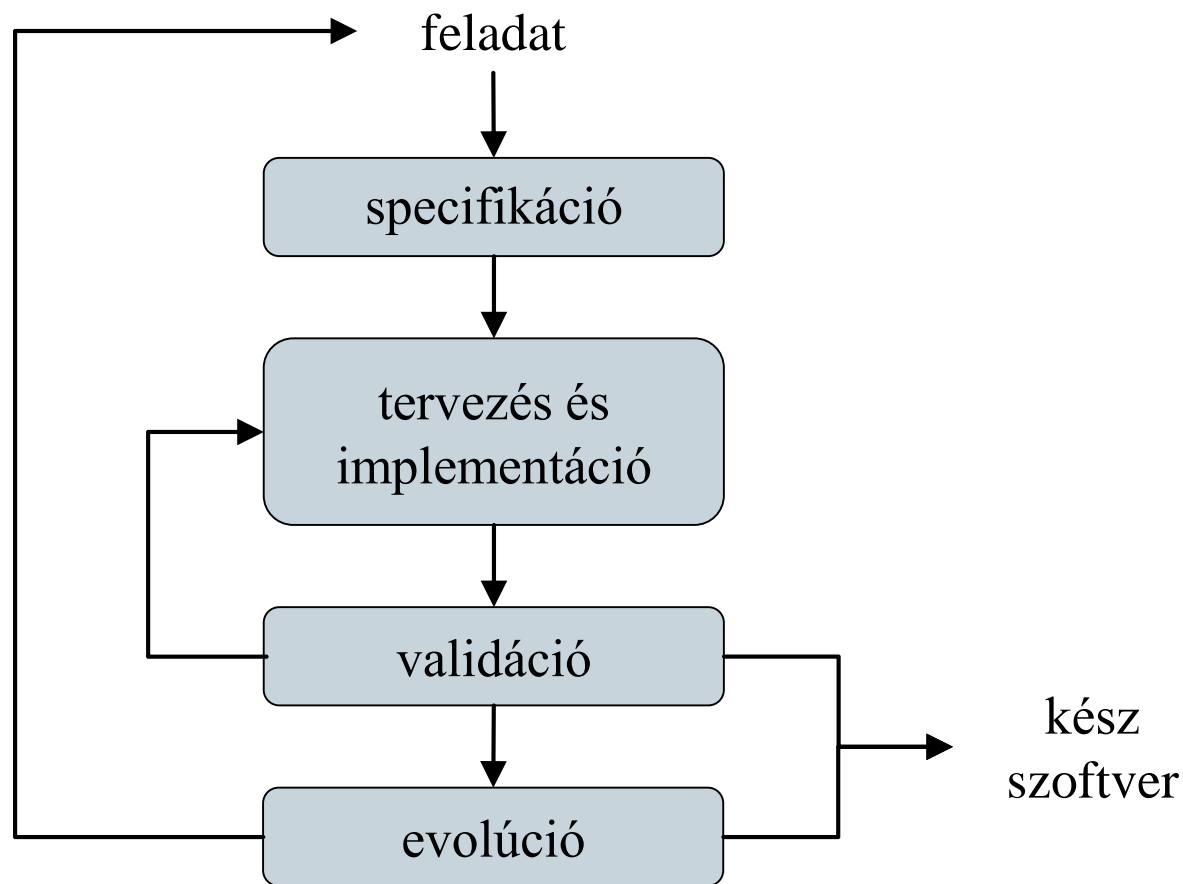
# A szoftverfejlesztési folyamat

## A szoftver életciklus

- Minden szoftver rendelkezik *életciklussal*, amely meghatározza létét a feladat kitűzésétől a program használatának befejeztéig
- Az életciklus általában négy fő fázisra bontható:
  1. *specifikáció*: a szoftver funkcionalitásának és megszorításainak megadása
  2. *tervezés és implementáció*: a specifikációnak megfelelő szoftver előállítása
  3. *verifikáció és validáció*: a szoftver ellenőrzése a specifikációnak történő megfelelésre
  4. *evolúció*: a szoftver továbbfejlesztése a változó elvárásoknak megfelelően

# A szoftverfejlesztési folyamat

## A szoftver életciklus



# A szoftverfejlesztési folyamat

## Specifikáció

- A *specifikáció* (*software specification*) célja a feladatot megoldó szoftver funkcióinak tisztázása, a rendszerre és a fejlesztésre vonatkozó elvárások megadása
  - feltérképezi a követelményeket felhasználói, valamint fejlesztői szemszögből, lépései:
    - megvalósíthatósági elemzés
    - követelmény feltárás és elemzés
    - követelmény specifikáció
    - követelmény validáció
  - eredménye a *szoftver követelmény-leírása* (*software requirements specification*)

# A szoftverfejlesztési folyamat

## Tervezés és implementáció

- A szoftver tervezése és implementációja (*software design and implementation*) feladata a specifikáció átalakítása egy végrehajtható rendszerre
  - meghatározza a rendszer szerkezetét (felépülés), felületét (be- és kimenet), működését (alkalmazott algoritmusok, kommunikációs folyamatok)
  - a folyamat során elkészül a *szoftver rendszerterve* (*software design description*), amely tartalmazza a program statikus és dinamikus szerkezetét, a kommunikációs csatornák feltérképezését, az implementációs és tesztelési tervet
  - elkészíthető a *szoftver prototípusa* (*prototype*), amely a program egyszerűsített megvalósítását tartalmazza

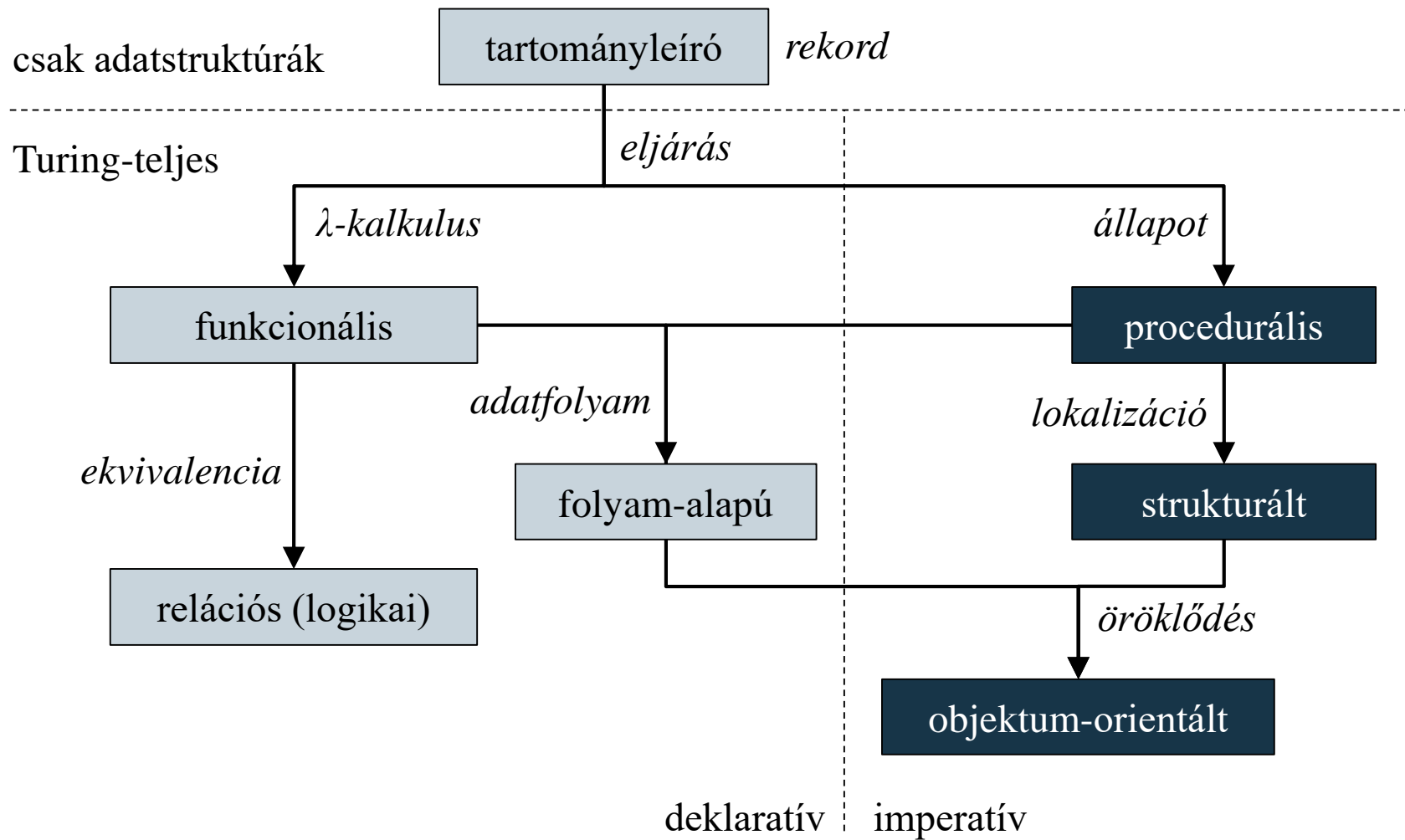
# A szoftverfejlesztési folyamat

## Tervezés és implementáció

- az implementációhoz megfelelő *szoftverfejlesztési környezetet* kell használnunk, a programkód változásait *verziókövetéssel* tartjuk nyilván
- az implementáció részeként az egyes programegységek tesztelése is megtörténhet
- a szoftverek tervezésének és programozásának módszerét nevezzük *programozási paradigmának*
  - meghatározza a programozási stílust, az absztrakciós szintet
  - meghatározza az alkalmazható programozási nyelvek körét is, és fordítva

# A szoftverfejlesztési folyamat

## Programozási paradigmák



# A szoftverfejlesztési folyamat

## Validáció és evolúció

- A *verifikáció és validáció (software verification and validation)* célja megmutatni, hogy a rendszer megfelel a specifikációnak, valamint a felhasználói elvárásoknak
  - alapvetően tesztelés, amely több fázisban, több módszerrel történik (a felhasználói tesztek csak az utolsó lépésben történnek)
- Az *evolúció (software evolution)* során új követelményeknek megfelelően bővítjük a szoftvert, illetve korrigáljuk a felmerülő hibákat
  - átlagosan a szoftver élettartamának 80%-a, ezért eleve bővíthetőre, módosíthatóra kell kialakítani a szoftvert



# A szoftverfejlesztési folyamat

## A szoftver életciklus

- További lépések is kísérhetik a fejlesztési folyamatot, pl.
  - *kihelyezés (deployment)*: a program üzembe állítása, és elérhetővé tétele
  - *tréning és támogatás (training and support)*: a felhasználókkal való kapcsolattartás (annak biztosítása, hogy a szoftvert megfelelően tudják kezelni és használni)
- A szoftver dokumentációja két részből tevődik össze:
  - *felhasználói dokumentáció*, amely tartalmazza a szoftver üzembe helyezésének, funkcióinak bemutatását
  - *fejlesztői dokumentáció*, amely tartalmazza a szoftver megvalósítását folyamatát és részletes ismertetését

# A szoftverfejlesztési folyamat

## Ütemterv

- A szoftver életciklus fázisai (*feladatai*) további fázisokra (*részfeladatokra*) tagolhatóak, így egy hierarchikus feladatszerkezetet kapunk
  - az egyes feladatokra erőforrásokat és időkorlátot adhatunk
  - az egyes feladatok között *függőségeket* állapíthatunk meg (a feladat nem kezdhető el, amíg a függősége el nem készül)
  - ezek alapján elkészíthetjük a *projekt ütemtervét*
    - tartalmazza a feladatok időbeli beosztását, függőségeit, felelőseit, így áttekinthetővé teheti az erőforrás szükségleteket
    - általában a specifikáció során készül el, de később módosulhat

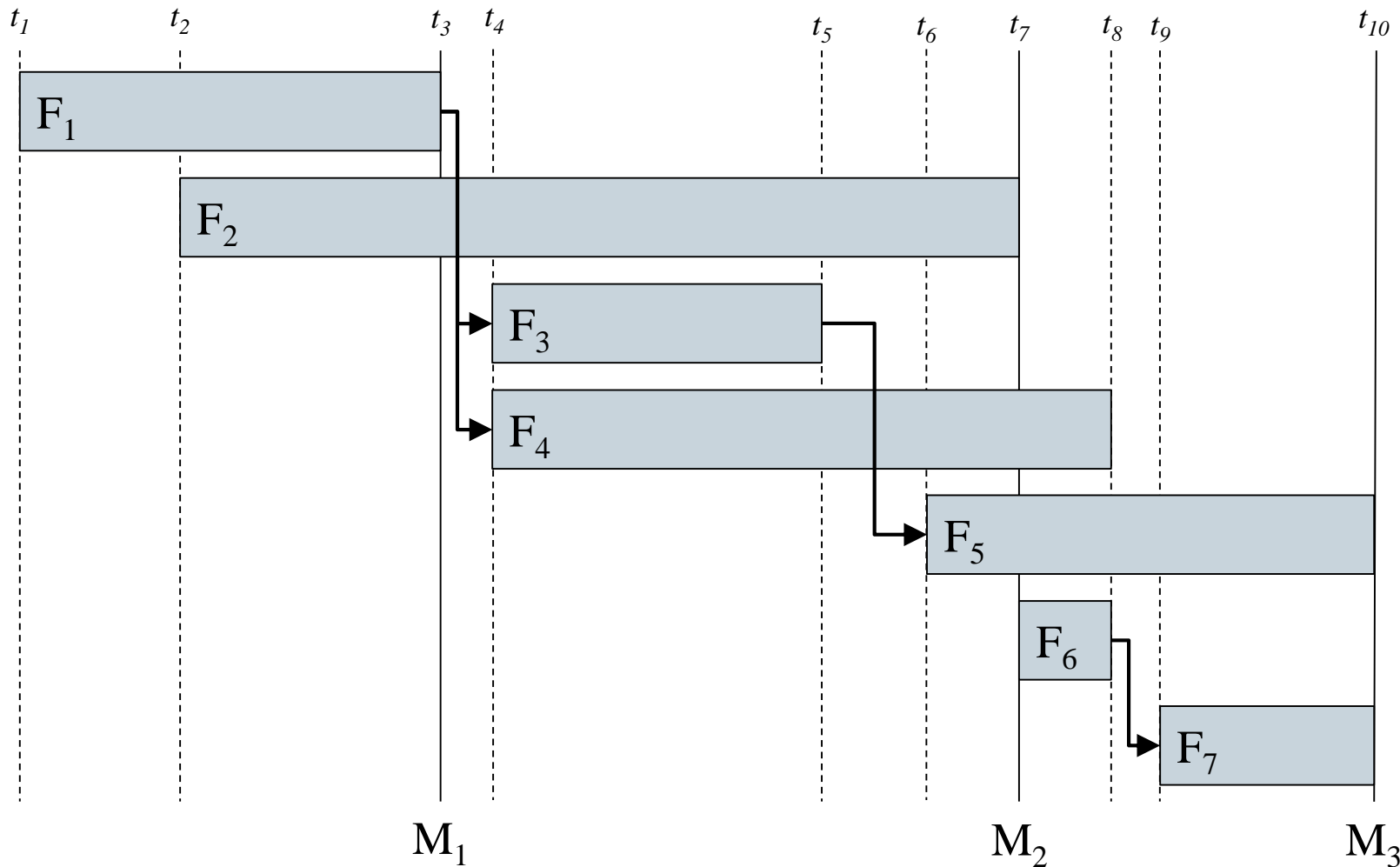
# A szoftverfejlesztési folyamat

## Mérföldkövek

- A feladatokhoz *mérföldköveket (milestone)* rendelhetünk, amelyek lehetőséget adnak a projekt haladásában történő betekintésre
  - a mérföldkő egy adott cél adott időpontra történő elérését jelenti, így *névvel, eseménnyel, céllal* rendelkezik
  - a mérföldkövek be nem tartása általában korrekciókat követel a projekt lefutásában
  - kellően konkrétnek, ellenőrizhetőnek, számon kérhetőnek kell lennie (akár a termékgazda számára is)
  - a fő mérföldkövek az egyes fázisok lezárását jelentik, ezen kívül számos további mérföldkő adható

# A szoftverfejlesztési folyamat

## Ütemterv



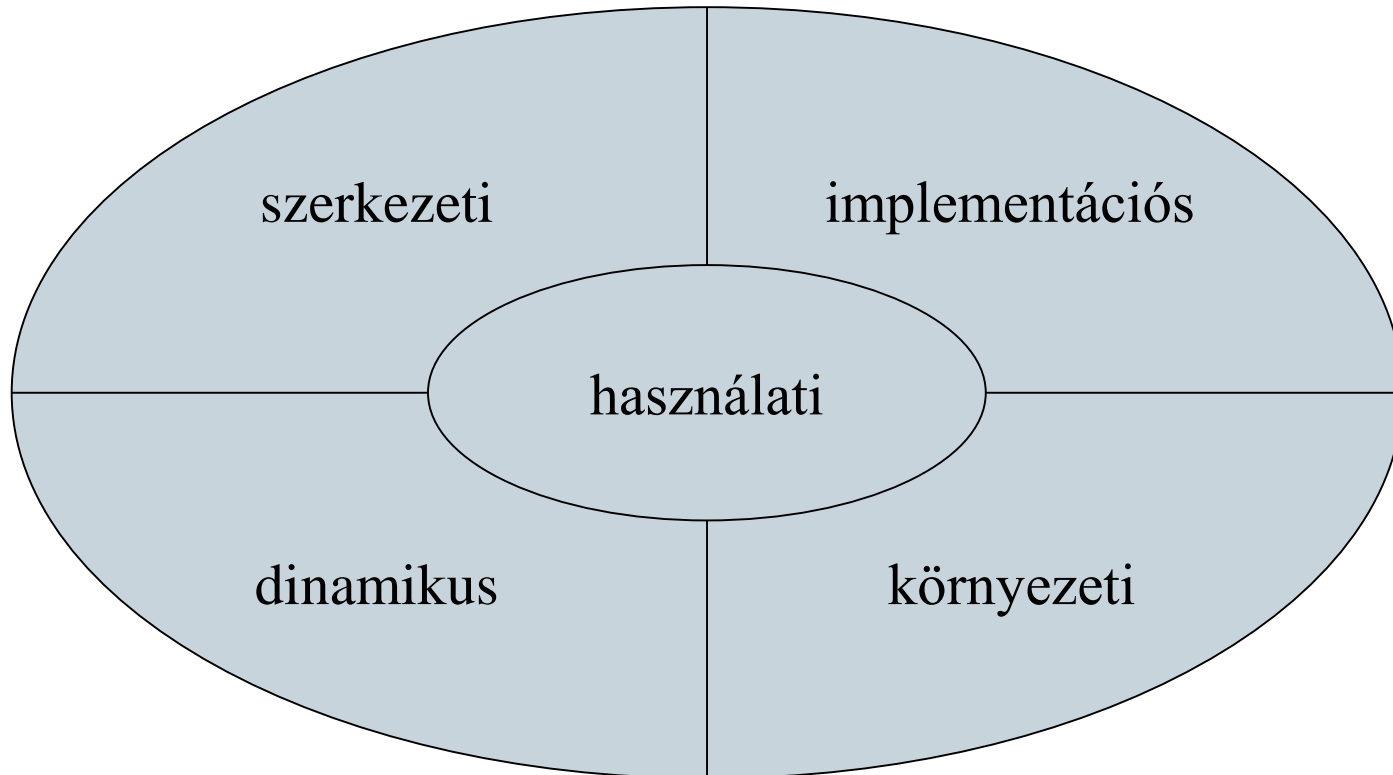
# A szoftverfejlesztési folyamat

## Az UML

- A szoftverfejlesztési életciklust folyamatosan követi a modellezés, ennek eszköze az *egységes modellezési nyelv* (*Unified Modeling Language, UML*), amely egy öt pillérű szemléletrendszerrel rendelkezik:
  - *használati*: a szoftver szolgáltatásai és azok kapcsolata a felhasználókkal
  - *szerkezeti* (statikus): a rendszer és a programegységek felépítése, kapcsolatai
  - *dinamikus*: a programegységek viselkedése
  - *implementációs*: a megvalósítás szempontjai, komponensei
  - *környezeti*: hardver és szoftver erőforrások

# A szoftverfejlesztési folyamat

## Az UML



# A szoftverfejlesztési folyamat

## Szoftverfejlesztési modellek

- Amellett, hogy a szoftverfejlesztés betartja az életciklus fázisait, a folyamat lefolyása különféle módokon történhet, amiket *szoftverfejlesztési módszereknek* nevezünk
  - klasszikus módszerek: *vizesés, prototipizálás, inkrementális, iteratív, spirális, V-Model*
  - agilis módszerek: *Scrum, Lean, Kanban, XP* (extreme programming), *RAD* (rapid application development)
  - speciális célú módszerek: *BDD* (behavior-driven development), *TDD* (test-driven development), *FDD* (feature-driven development)
  - formális módszerek: *B-módszer*

# A szoftverfejlesztési folyamat

## Szoftvereszközök

- A fejlesztőcsapat munkáját megfelelő szoftvereszközökkel kell alátámasztani
  - *projektirányítási eszközzel (project tracking system)*, amely támogatja a dokumentálást és a feladatok követését
  - fejlett *tervezőeszközzel (case tool)*, ahol a fejlesztés folyamata és a felelősség is nyomon követhető
  - *integrált fejlesztőkörnyezettel (IDE)*, amely elősegíti a csapatmunkát
  - *verziókövető rendszerrel (revision control system)*, amely lehetővé teszi a programkód változásainak követését
  - *folyamatos integrációs eszközzel (continuous integration)*, amely elősegíti a tevékenységek automatizálását



# A szoftverfejlesztési folyamat

## Projektirányítási eszközök lehetőségei

- A *projektirányítási eszköz* lehetőséget ad a projekt menedzselésének ellátására
  - általában webes felületű eszköz, amely bárhonnan elérhető és használható
  - főbb funkciói:
    - fejlesztés ütemtervének, kockázatainak rögzítése
    - egyszerű és folyamatos dokumentálás lehetősége
    - feladatok, tevékenységek, hibák rögzítése, és a kapcsolatos tevékenységek követése
    - integrált forráskód böngészés, és forrástörténet áttekintés (verziókezelés)
  - pl.: *Trac, Redmine, Team Foundation Server (TFS)*

# A szoftverfejlesztési folyamat

## Projektirányítási eszközök lehetőségei

- A rendszerek lehetőséget adnak a fejlesztők számára feladatok kitűzésére, valamint a tesztelők számára a programban fellelhető hibák jelzésére
  - a feladatokat úgynevezett *cédulák* (*ticket*, *issue*) segítségével írhatóak ki
    - jelölhetnek új funkcionalitást (*feature*), hibát (*bug*), egyéb fejlesztési feladatot (*task*), vagy dokumentációs feladatot (*documentation*)
    - megadható a leírása, felelőse, határideje
    - kommentálhatóak, lezárhatóak, újra kinyithatóak
- a cédulák biztosítják a fejlesztési és tesztelési folyamat naplózását

# A szoftverfejlesztési folyamat

## Projektvezető szolgáltatások

- A *projektvezető szolgáltatások* (*project hosting services*) olyan online szolgáltatások, amelyek a projekttel kapcsolatos legtöbb funkcionalitást integrálják
  - projektmenedzsment, kód tárolás, verziókövetés, dokumentáció, folyamatos integráció, kihelyezés
  - integrálhatóak projektirányítási eszközökkel, fejlesztőeszközzel
  - garantálják a kód épségét, a folyamatos rendelkezésre állást
  - általában nyílt forráskódú szoftverek esetén ingyenes a használatuk
  - pl.: *SourceForge*, *CodePlex*, *GitHub*, *GitLab*

# Esettanulmányok

## Tic-Tac-Toe játék

*Feladat:* Készítsünk egy Tic-Tac-Toe programot, amelyben két játékos küzdhet egymás ellen.

- a programban jelenjen meg egy játéktábla, amelyen végig követjük a játék állását (a két játékost az ,X' és ,0' jelekkel ábrázoljuk)
- legyen lehetőség a játékosok neveinek megadására, új játék indítására, valamint játékban történő lépésre (felváltva)
- a program kövesse végig, melyik játékos hány kört nyert
- program automatikusan jelezzen, ha vége egy játéknak, és jelenítse meg a játékosok pontszámait

# Esettanulmányok

## Marika néni kávézója

*Feladat:* Készítsük el Marika néni kávézójának eladási nyilvántartását végigkövető programot.

- a kávézóban 3 féle étel (hamburger, ufó, palacsinta), illetve 3 féle ital (tea, narancslé, kóla) közül lehet választani
- az ételek ezen belül különfélék lehetnek, amelyre egyenként lehet árat szabni, és elnevezni, az italok nevei és árai rögzítettek
- a program kezelje a rendeléseket, amelyekben tetszőleges tételek szerepelhetnek, illetve a rendelés kapcsolódhat egy törzsvásárlóhoz
- biztosítsunk lehetőséget a függőben lévő rendeléseket lekérdezésére, valamint napi, havi és törzsvásárlói számra összesített nettó/bruttó fogyasztási statisztikák követésére

# Esettanulmányok

## Memory játék

*Feladat:* Készítsünk egy *Memory* kártyajátékot, amelyben két játékos küzd egymás ellen, és a cél kártyapárok megtalálása a játéktáblán.

- a játékosok felváltva lépnek, minden lépésben felfordíthatnak két kártyát
- amennyiben a kártyák egyeznek, úgy felfordítva maradnak és a játékos ismét léphet, különben visszafordulnak, és a másik játékos következik
- a játékot az nyeri, aki több kártyapárt talált meg
- lehessen a játékosok neveit megadni, kártyacsomagot választani, valamint a kártyák számát (a játéktábla méretét) szabályozni

# Esettanulmányok

## Utazási ügynökség

*Feladat:* Készítsük el egy utazási ügynökség apartmanokkal foglalkozó rendszerét.

- az apartmanok épületekben találhatóak, amelyek városokban helyezkednek el
- az épületek különböző adatokkal (leírás, szolgáltatások, pontos hely, tengerpart távolság, ...), valamint képekkel rendelkeznek
- a vendégek számára biztosítsunk egy webes felületet, amelyen keresztül apartmanokat kereshetnek, foglalhatnak
- a munkatársak számára biztosítsunk egy alkalmazást, amelyben szerkeszthetik az apartmanok adatait, képeit, valamint kezelhetik a foglalásokat