Programozási technológia 2.

Szoftverfejlesztési folyamat

Dr. Szendrei Rudolf

ELTE Informatikai Kar

2018.

Információk

- **Képzés**
  - Programtervező Informatikus BSc, nappali tagozat, C szakirány
- **Tárgykód**:
  - IP-17cPROGT2EG
- **Előfeltétel** (erős):
  - Programozási technológia 1.
- **Kreditszám**: 5
  - 2 óra előadás
  - 2 óra gyakorlat
  - 1 óra konzultáció
- **Cél**:
  - Szoftverfejlesztés csapatmunkában, végigkísérve a szoftver teljes életciklusát

Információk

Számonkérés

- A gyakorlaton kapott feladatot csapatban kell megoldani, ahol a fejlesztett szoftver minden egyes mérföldkövénél a végzett munkáért 1-1 db jegy jár csapattagonként
  - Követelmény elemzés
  - Tervezés
  - Prototípus implementáció
  - Végleges implementáció, tesztelés, dokumentáció
- Minden csapattagnak részt kell vennie minden mérföldkő teljesítésében, és minden megszerzett jegynek legalább elégségesnek kell lennie
- Az összevont gyakorlati jegy (a jegyek átlagával egyezik meg) akkor szerezhető meg, ha a félév végi teszt zárthelyi sikeres

Információk

Elérhetőségek

- **Honlap**:
  - https://swap.web.elte.hu/
- **E-mail**:
  - swap@inf.elte.hu
- **Személyesen**:
  - Déli épület, 2.602

Szoftverfejlesztési folyamat

Szoftverfejlesztés

- A szoftverek nélkülözhetetlen alkotóelemei a modern világnak
  - számos célt szolgálhatnak
  - különböző felépítésűek, működési elvűek
  - emiatt megvalósításuk módja jelentősen eltérhet
- A szoftverekben jelentős része hibás, a szoftverfejlesztési munkák jelentős része kudarcba fullad, ennek okai:
  - egyre nagyobb számban, egyre összetettebb szoftverekre van szükség
  - alacsonyak az elvárások a szoftverekkel szemben
- Emiatt nagy szükség van a *professzionális szoftverfejlesztés*re

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

Szoftverfejlesztési folyamat

Minőségi mutatók

- A szoftvereknek megfelelő színvonalon kell biztosítania az elvárt funkciókat, amit a szoftver *minőségi mutatói*val (*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

Szoftverfejlesztési folyamat

Szoftvertechnológiai projekt

- A szoftver fejlesztésének folyamatát *projekt*nek, előállításának felügyeletét *projektmenedzselés*nek 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

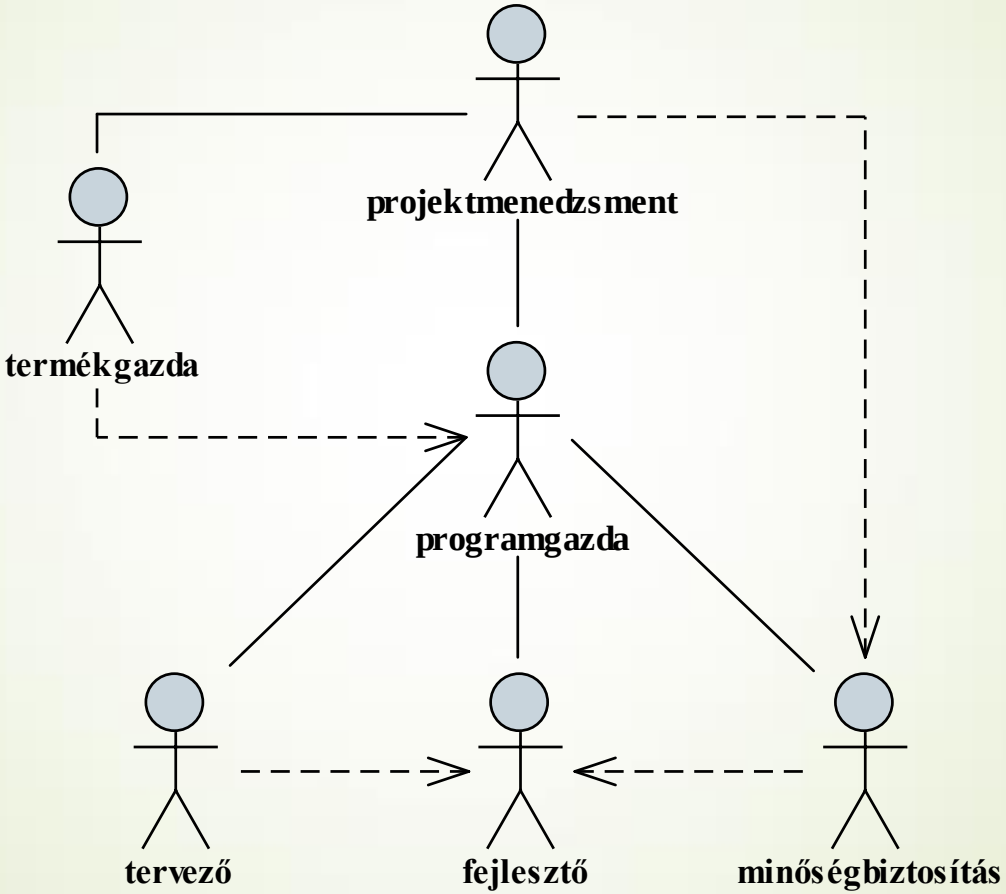
Szoftverfejlesztési folyamat

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

Szoftverfejlesztési folyamat

Szoftvertechnológiai projekt



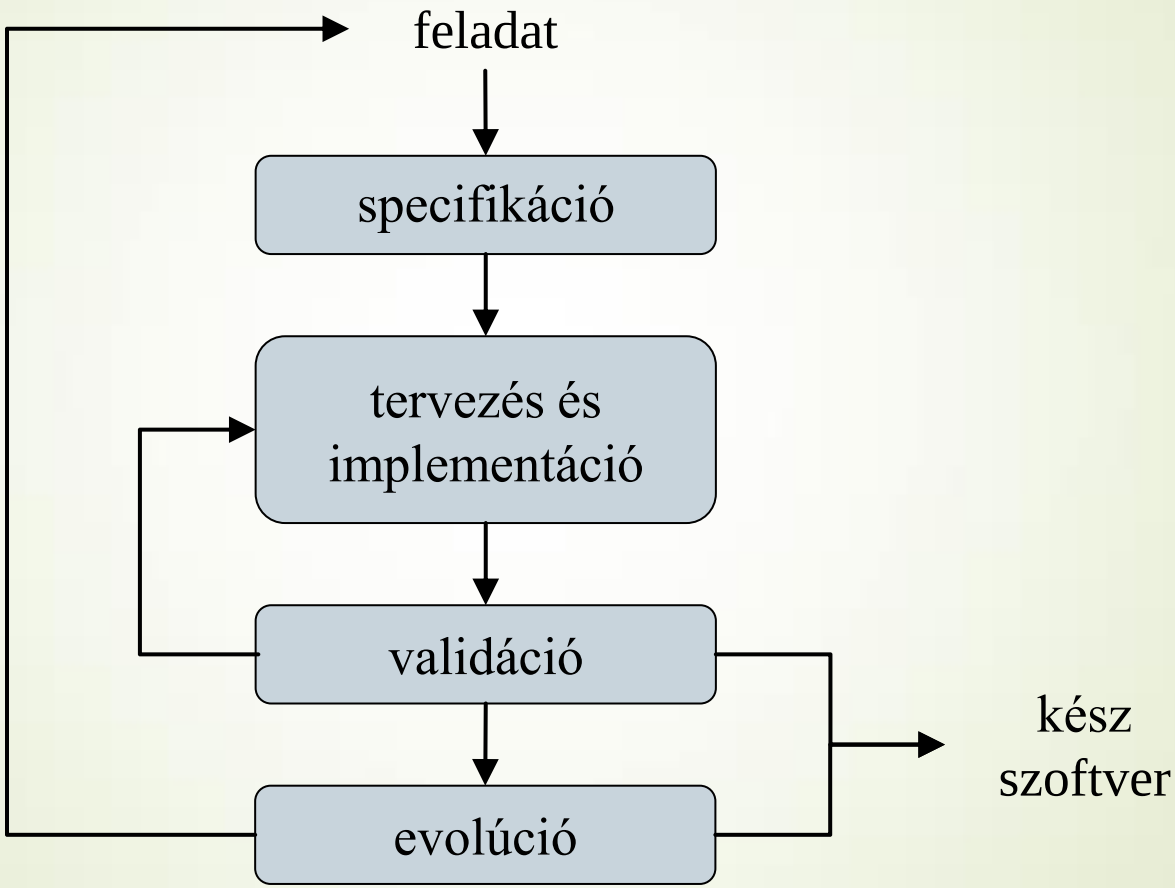
Szoftverfejlesztési folyamat

A szoftver életciklus

- Minden szoftver rendelkezik *életciklus*sal, 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

Szoftverfejlesztési folyamat

A szoftver életciklus



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*)

Szoftverfejlesztési folyamat

Tervezés és implementáció

- A *szoftver tervezése és implementációja* (*software design and implementation*) feladata a specifikáció átalakítása egy végrehajtható rendszerré
  - 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

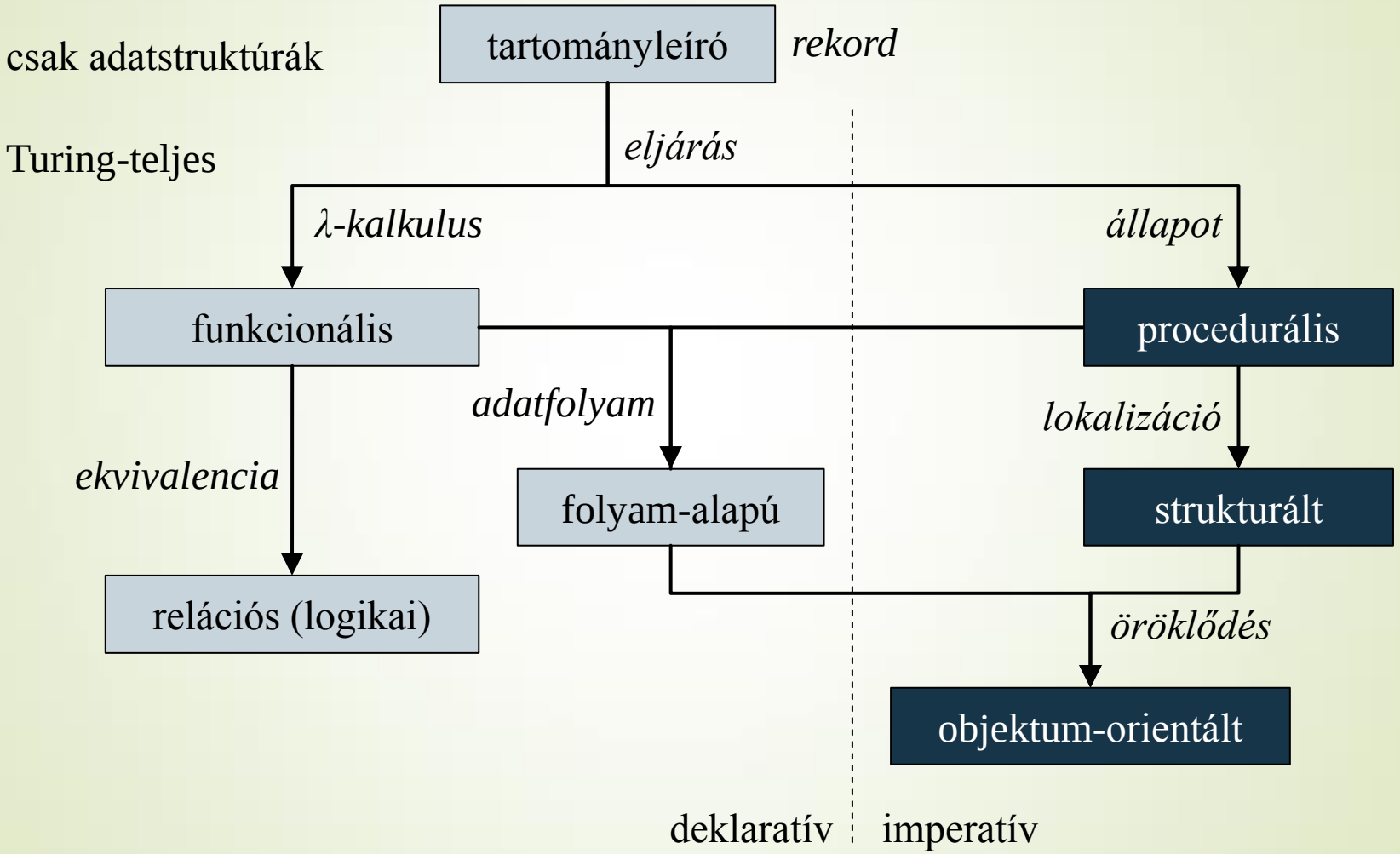
Szoftverfejlesztési folyamat

Tervezés és implementáció

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

Szoftverfejlesztési folyamat

Programozási paradigmák



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

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

Szoftverfejlesztési folyamat

Ütemterv

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

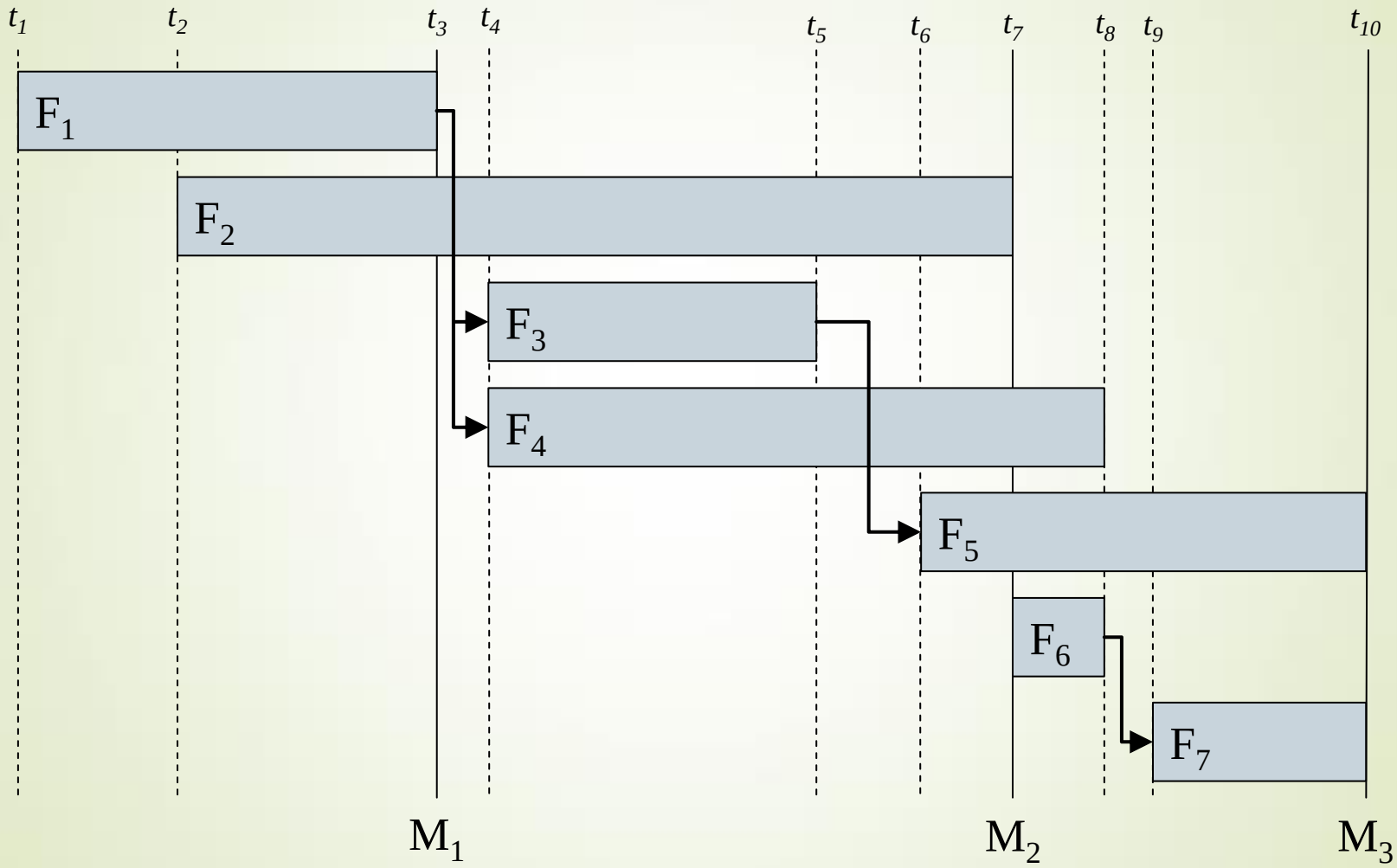
Szoftverfejlesztési folyamat

Mérföldkövek

- A feladatokhoz *mérföldkövek*et (*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 konkrétnak, ellenőrizhetőnek, számon kérhetőnek kell lennie (akár a termékgazda számára is)
  - a fő mérföldkövek az egyes fázisok lezárását jelentik, ezen kívül számos további mérföldkő adható

Szoftverfejlesztési folyamat

Ütemterv



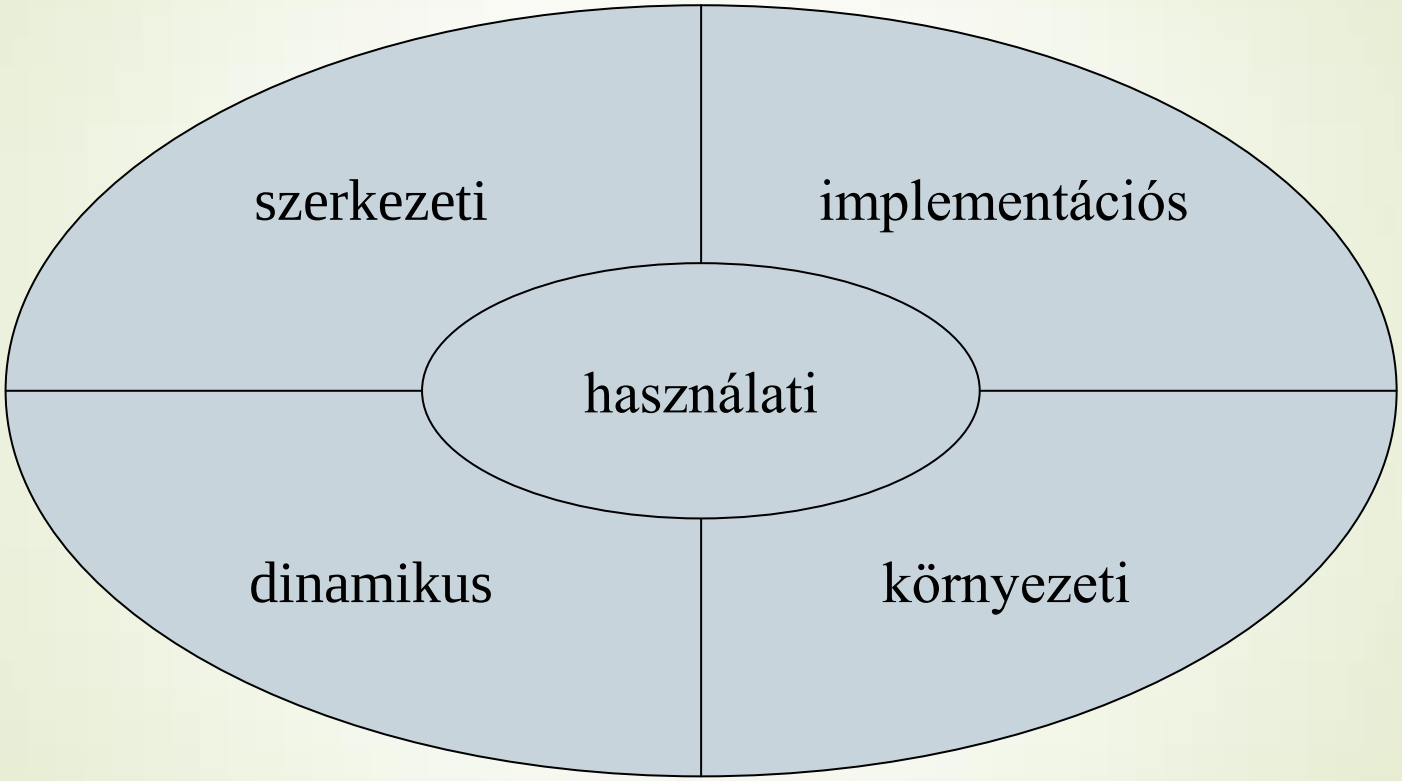
Szoftverfejlesztési folyamat

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

Szoftverfejlesztési folyamat

Az UML



Szoftverfejlesztési folyamat

Szoftvereszközök

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

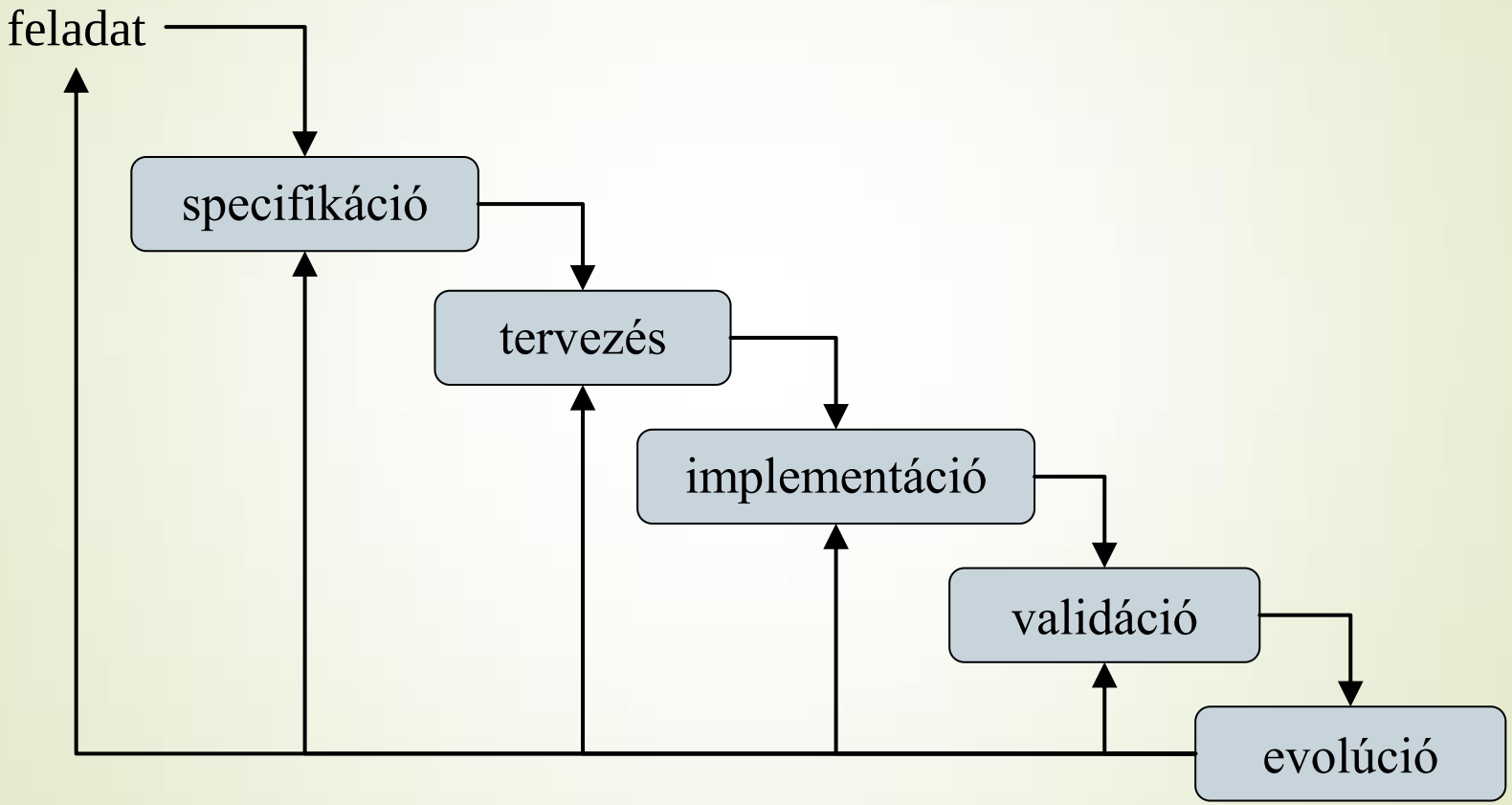
Szoftverfejlesztési folyamat

A vízesés modell

- A szoftverfejlesztési modell határozza meg az életciklus egyes fázisai közötti kapcsolatot, időbeliséget
- A legegyszerűbb fejlesztési modell a *vízesés (waterfall) modell*, amelyben az egyes fázisok lineárisan követik egymást
  - előre megtervezi a projekt időtartamát, ráfordításait
  - elvárja minden fázis megfelelő dokumentálását, amely tartalmazza annak eredményeit
  - *előnyei*: jól strukturált, dokumentált folyamatot biztosít
  - *hátrányai*: nem teszi lehetővé a követelmények megváltoztatását, nem készül fel az esetleges nehézségekre (nincs kockázatkezelés)

Szoftverfejlesztési folyamat

A vízesés modell



Szoftverfejlesztési folyamat

Prototípusok

- A szoftverfejlesztés során felmerülő nehézségek könnyebben előreláthatóak, ha a szoftvernek elkészítjük a *prototípusa*it (*prototyping*), amely lehet:
  - *horizontális prototípus*: interakciós szempontból mutatja be szoftvert (pl. felhasználói felület)
  - *vertikális prototípus*: egy adott funkció(csoport) részletes megvalósítása (pl. adatkezelés)
- A folyamat során megvalósított prototípusok a szoftver részévé válhatnak (*evolutionary prototyping*), vagy szolgálhatják csak a bemutatást/ellenőrzést, és ténylegesen nem kerülnek felhasználásra (*throwaway prototyping*)

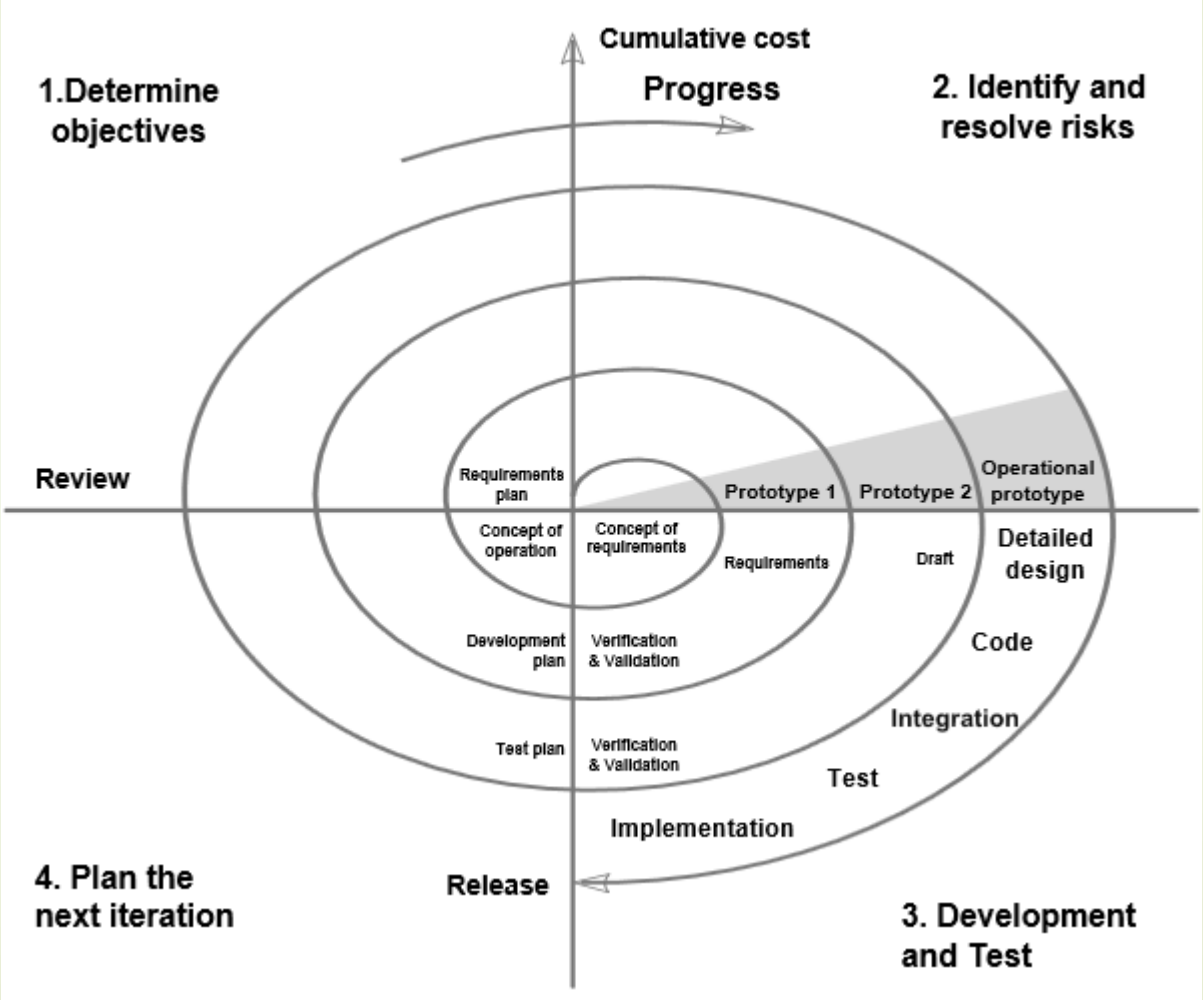
Szoftverfejlesztési folyamat

A spirális modell

- A (*Boehm-féle*) *spirális* (*spiral*) *modell* egy kockázatvezérelt fejlesztési modell, amelyben a folyamat során elsőként prototípusok kerülnek megvalósításra, amelyek kiértékelése után kerül megvalósításra a tényleges szoftver
  - a fejlesztés ciklusokban történik, amelyben az elkészített prototípusok, valamint a továbbfejlesztésével kapcsolatos kockázatok kiértékelésre kerülnek
  - *előnyei*: jobban alkalmazkodik a változó követelményekhez, a prototípusok lehetővé teszik a nehézségek előrelátását
  - *hátrányai*: költségesebb a prototípus elkészítése és a kockázatkiértékelés végett, továbbá a prototípusok megzavarhatják a felhasználót

Szoftverfejlesztési folyamat

A spirális modell



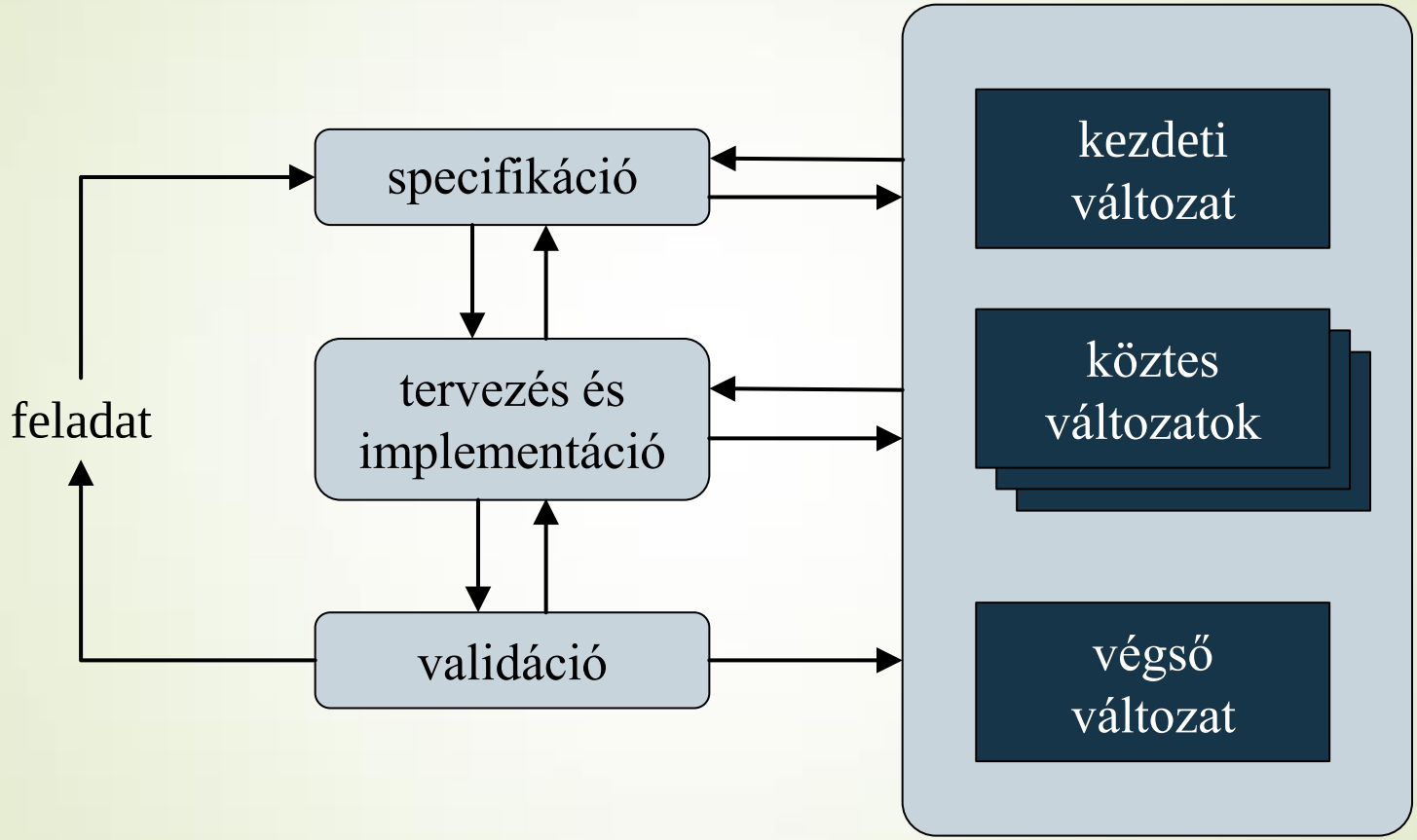
Szoftverfejlesztési folyamat

Az inkrementális modell

- Az *inkrementális* (*incremental*) *modell* több lépésből építi fel a folyamatot, és több változatban állítja elő a szoftvert
  - minden változat egy újabb funkcionalitással bővíti a szoftvert, a fázisok rövidek, gyors visszajelzésekkel (a felhasználói oldalról)
  - az egyes fázisok átfedésben vannak, és kihatnak egymásra
  - *előnyei*: gyorsan alkalmazkodik a változó követelményekhez, a felhasználó jobban követheti a fejlesztési folyamatot
  - *hátrányai*: kevésbé menedzselhető, tervezhető, áttekinthető, nehezebben validálható

Szoftverfejlesztési folyamat

Az inkrementális modell



Szoftverfejlesztési folyamat

Agilis szoftverfejlesztés

- Az *agilis szoftverfejlesztés* (*agile software development*) célja a gyors alkalmazásfejlesztés megvalósítása, inkrementális alapon
  - a szoftver folyamatos fejlesztés és kiadás alatt áll (*continuous delivery*), a sebesség állandó, a változtatások minden lépésben beépíthetőek (*welcome changes*)
  - a működő szoftver az előrehaladás mérőeszköze, előtérben az egyszerűség, ugyanakkor folyamatos odafigyelés a megfelelő tervezésre, optimalizációra
  - a fejlesztést általában önszervező, kis csapatok végzik, megosztott felelősséggel, folytonos interakcióval, gyors visszajelzésekkel

Szoftverfejlesztési folyamat

Az Agilis Kiáltvány

- Azzal leplezzük le a szoftverfejlesztés jobb módjait, hogy csináljuk és segítünk másoknak is csinálni. Ezen a munkán keresztül következő értékekhez jutottunk el:
  - *Egyének és kölcsönhatások* előnyben részesítése a folyamatok- és eszközökkel szemben
  - *Működő szoftver* előnyben részesítése az átfogó dokumentációval szemben
  - *Ügyféllel való együttműködés* előnyben részesítése a szerződéses megállapodással szemben
  - *Változásokra adandó válasz* előnyben részesítése egy terv követésével szemben

  Habár a jobb oldali elemekben is van érték, mi sokkal értékesebbnek tartjuk a baloldali elemeket.
  (© 2001, Beck, K., et. al.)

Szoftverfejlesztési folyamat

Agilis szoftverfejlesztés

- Az agilis fejlesztés
  - *előnyei*: jól alkalmazkodik a változtatásokhoz, hatékonyabbá teszi a fejlesztési folyamatot
  - *hátrányai*: egyes tényezői nehezen megvalósíthatóak, különösen nagyobb szoftverek esetén a megvalósításhoz képzett fejlesztők kellenek, a dokumentáció hiánya megnehezíti a későbbi evolúciót
- Speciális agilis fejlesztési módszer az *Extreme programming* (*XP*), amely elvárja a követelmények viselkedés alapú felbontásával (BDD), a tesztek előre történő megadását (TDD), a folyamatos integrációt és refaktorálást, valamint támogatja a párban történő programozást

Szoftverfejlesztési folyamat

Scrum módszer

- Az agilis fejlesztés menedzselését az egyes változatok előállítása szempontjából közelítik meg, amelyhez a *Scrum* módszer ad egy általános modellt
  - fő lépései:
    1. *architekturális tervezés*, amely megadja a szoftver magas szintű vázát
    2. *futamok* (*sprint*), amelyek az egyes változatokat állítják elő, és rögzített hosszúságúak (2-4 hét)
    3. *projektzárás*, a szükséges dokumentáció előállítása
  - nincs projektmenedzser, de minden futamnak van felelőse (*scrum master*), akinek a személye futamonként változik

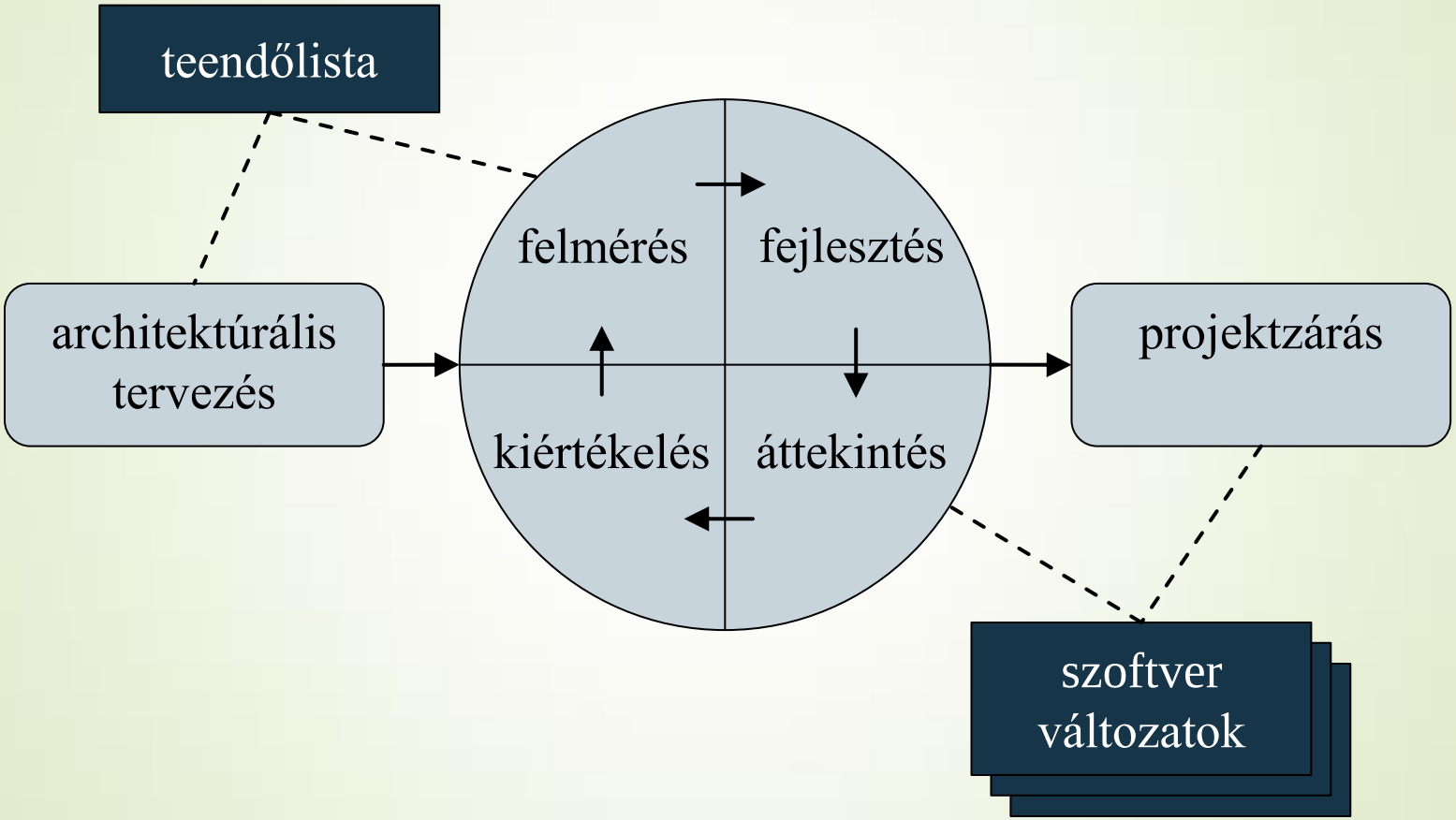
Szoftverfejlesztési folyamat

Scrum módszer

- Minden futam egy összetett folyamat megtervezett lépésekkel
  - feladatok felmérése (*select*), lefejlesztése (*develop*), áttekintése (*review*), kiértékelése (*assess*)
  - a megvalósítandó funkciók a termékgazdával egyetértésben kerülnek kiválasztásra a teendők listájából (*product backlog*)
  - naponta rövidebb megbeszélések (*stand-up meeting*) a teljes csapat számára
  - ciklus elején/végén hosszabb megbeszélések (*sprint planning*, *sprint review*), valamint visszatekintés (*retrospective*) a termékgazdával

Szoftverfejlesztési folyamat

Scrum módszer



Szoftverfejlesztési folyamat

Csoportosítás

- A szoftverfejlesztési modelleket 3 csoportba soroljuk
  - *terv-vezérelt* (*plan-driven*): célja a rend fenntartása, a szoftver fejlesztése előzetes specifikáció és tervezés alapján történik, igyekszik garantálni a minőséget
  - *agilis*: célja a változáshoz történő alkalmazkodás, az egyszerűség, így kevésbé garantálja a minőséget
  - *formális*: garantálja a minőséget, az implementáció bizonyíthatóan helyes megoldását adja a specifikációnak
- A gyakorlatban a fejlesztőcsapat és a feladat befolyásolja leginkább a választott módszert
  - sokszor a különböző módszerek vegyítve jelennek meg