RUP, XP és SPEM

Budapest

2010. Április 21.

Ráth István

rath@mit.bme.hu

(Szőke Ákos fóliái alapján)

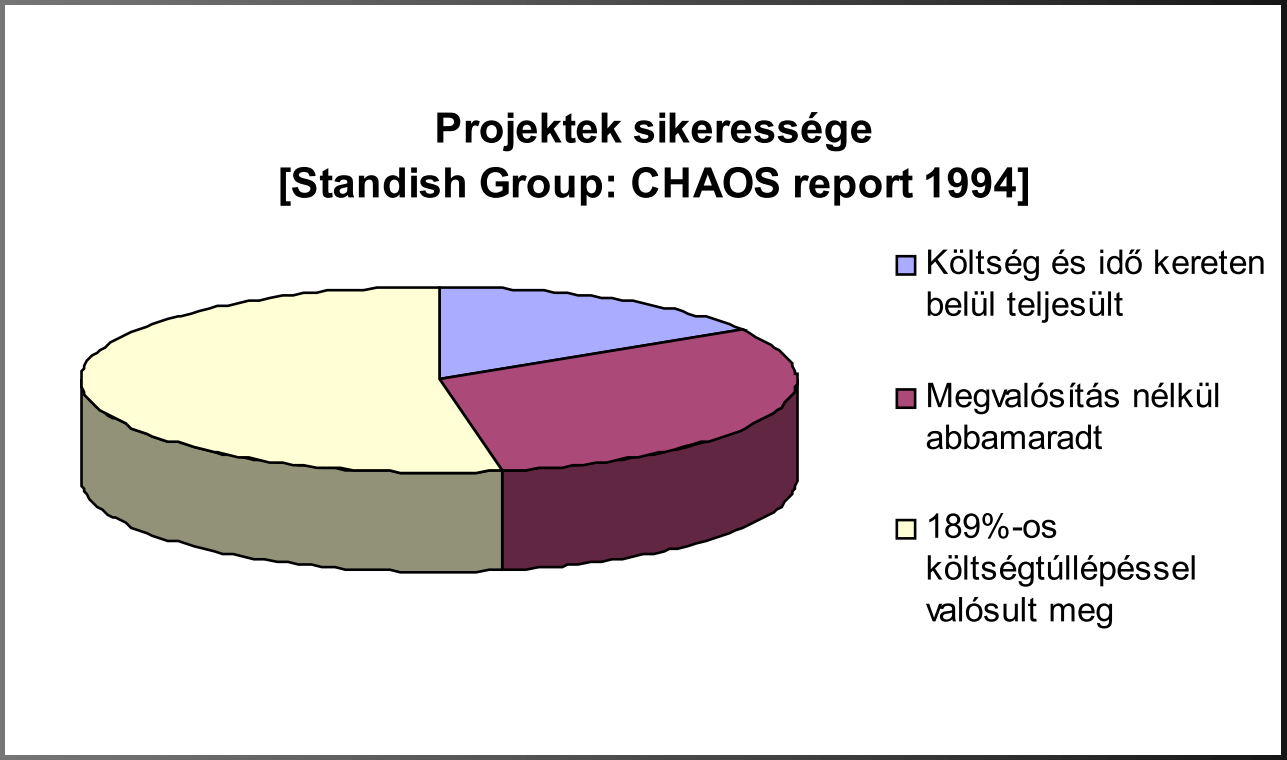
Tartalomjegyzék

- Best practices
- A Rational Unified Process (RUP)
  - RUP elveinek bemutatása
  - RUP elemeinek bemutatása
  - RUP fázisok bemutatása
  - RUP, mint termék bemutatása
- eXtreme programming
- Agilis metodikák összehasonlítása
- Folyamatmintáktól a SPEM-ig

Best Practices

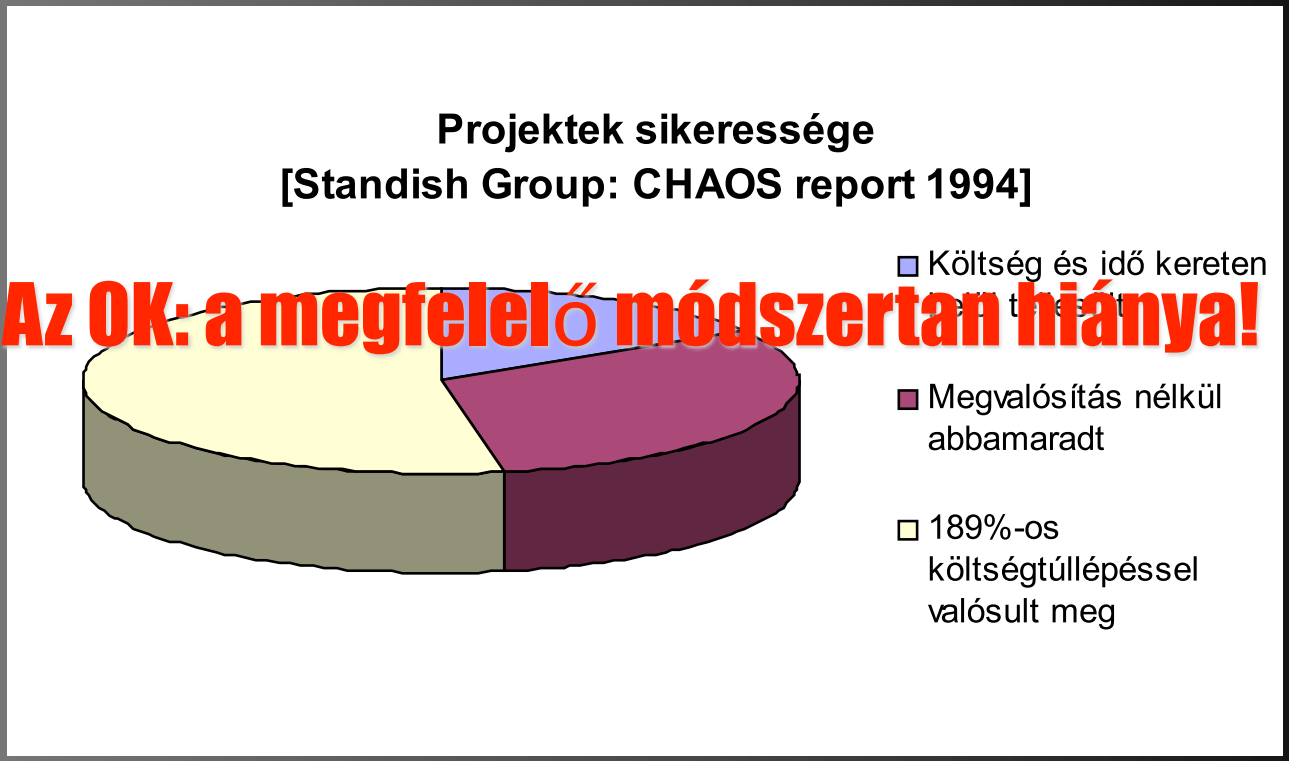
Trend a szoftvertechnológiában

- Egyre komplexebb rendszerek
- A meglévő fejlesztési folyamatok nem megfelelőek:



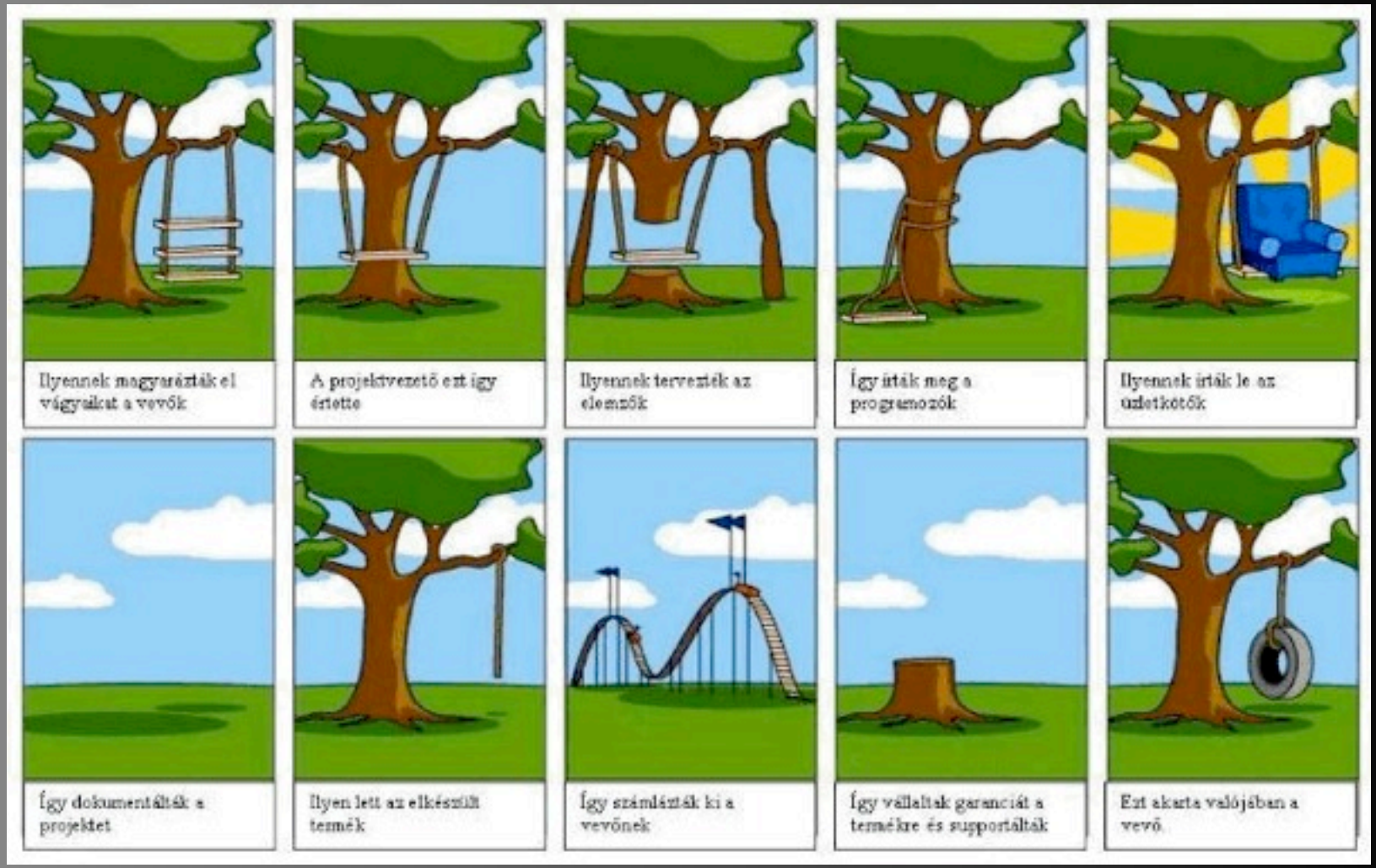
Trend a szoftvertechnológiában

- Egyre komplexebb rendszerek
- A meglévő fejlesztési folyamatok nem megfelelőek:



Az OK: a megfelelő módszertan hiánya!

A hinta


Ilyennek magyarázták el vágyaikat a vevők
A projektvezető ezt így értette
Ilyennek tervezték az elemzők
Így írták meg a programozók
Ilyennek írták le az üzletkötők
Így dokumentálták a projektet
Ilyen lett az elkészült termék
Így számlázták ki a vevőnek
Így vállalták garanciát a termékre és supportálták
Ezt akarta valójában a vevő

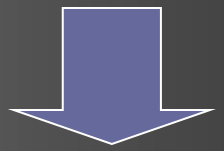
A probléma megoldás lépései:

A probléma megoldás lépései:

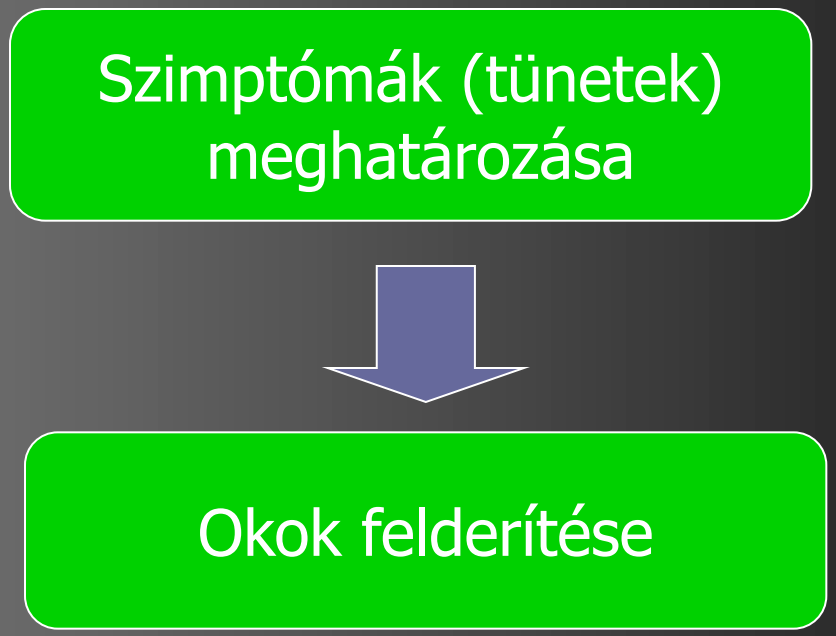
Szimptómák (tünetek) meghatározása

A probléma megoldás lépései:

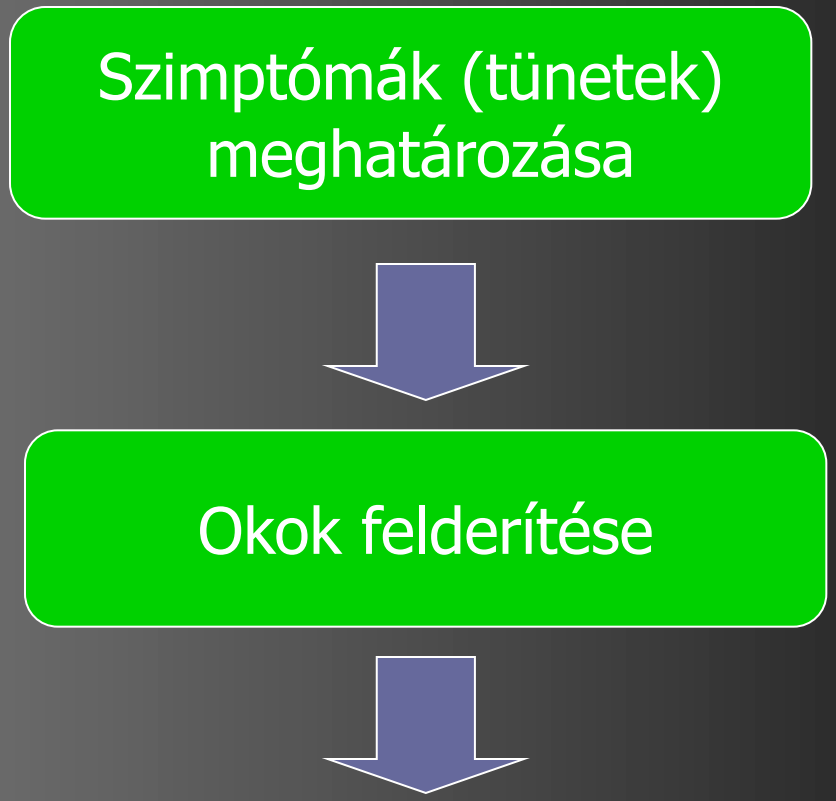
Szimptómák (tünetek) meghatározása



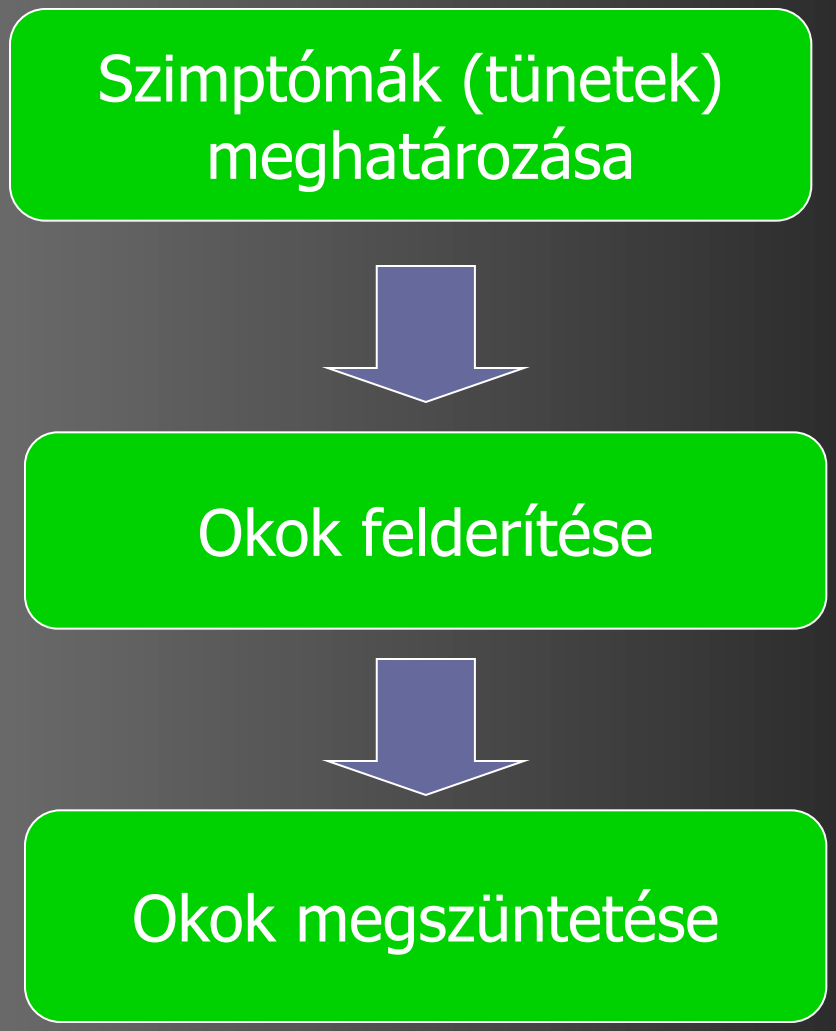
A probléma megoldás lépései:



A probléma megoldás lépései:



A probléma megoldás lépései:



Szimptómák meghatározása

- Elvárások nem teljesülnek
- Követelmények zavarosak
- Modulok nem illeszkednek egymáshoz
- Nehéz a karbantartás
- Hibák késő detektálása
- Gyenge minőség
- Gyenge performancia
- Fejlesztők közötti eltérések

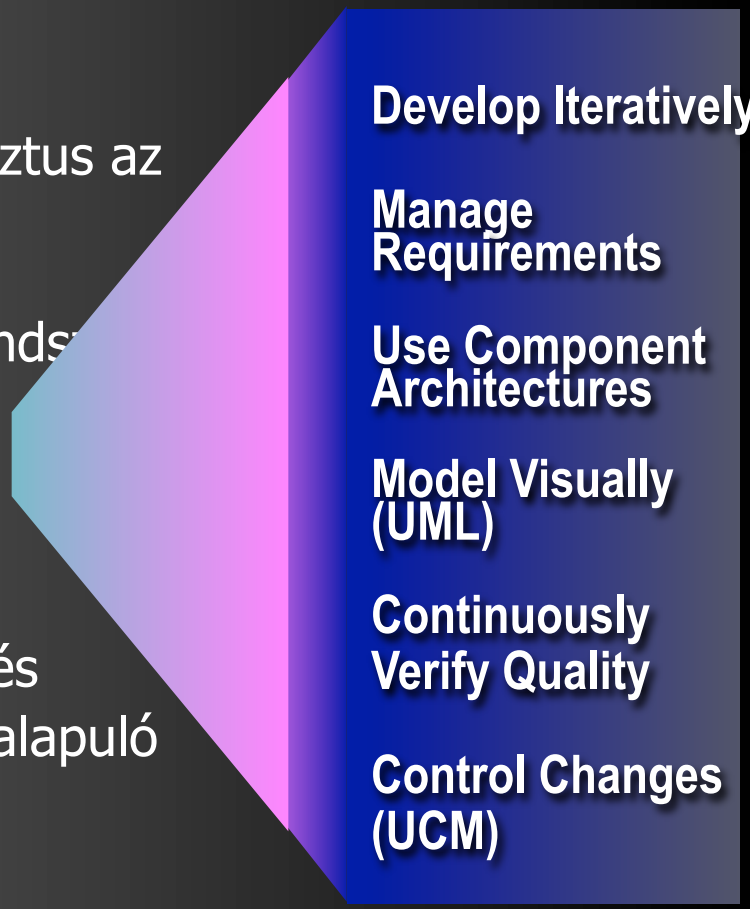
Okok felderítése

- Elégtelen követelmény-meghatározás
- Félreérthető kommunikáció
- Nem eléggé robosztus az architektúra
- Túlságosan nagy komplexitású a rendszer
- Nem detektált inkonzisztencia
- Alacsony fokú kiteszteltség
- Szubjektív értékelés
- Vízesés modellen alapuló fejlesztés
- Nem kontrollált változtatás
- Elégtelen automatizáció

Okok megszüntetése – Legjobb gyakorlatok

- Elvárások nem teljesülnek
- Követelmények zavarosak
- Modulok nem illeszkednek egymáshoz
- Nehéz a karbantartás
- Hibák késő detektálása
- Gyenge minőség
- Gyenge performancia
- Fejlesztők közötti eltérések

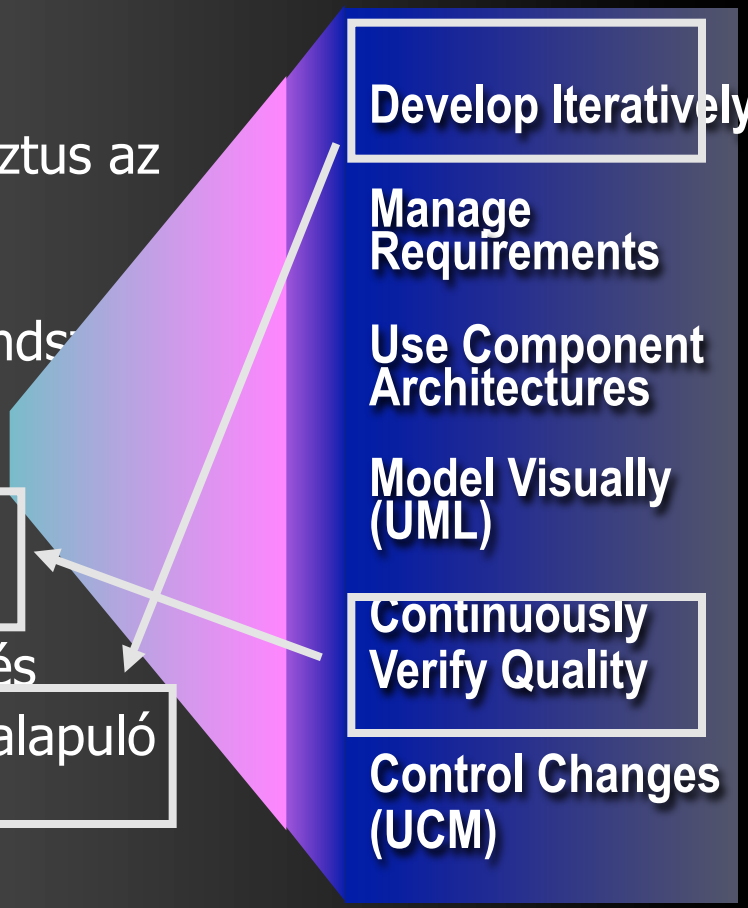
- Elégtelen követelmény-meghatározás
- Félreérthető kommunikáció
- Nem eléggé robosztus az architektúra
- Túlságosan nagy komplexitású a rends
- Nem detektált inkonzisztencia
- Alacsony fokú kiteszteltség
- Szubjektív értékelés
- Vízesés modellen alapuló fejlesztés
- Nem kontrollált változtatás
- Elégtelen automatizáció



Okok megszüntetése – Legjobb gyakorlatok

- Elvárások nem teljesülnek
- Követelmények zavarosak
- Modulok nem illeszkednek egymáshoz
- Nehéz a karbantartás
- Hibák késő detektálása
- Gyenge minőség
- Gyenge performancia
- Fejlesztők közötti eltérések

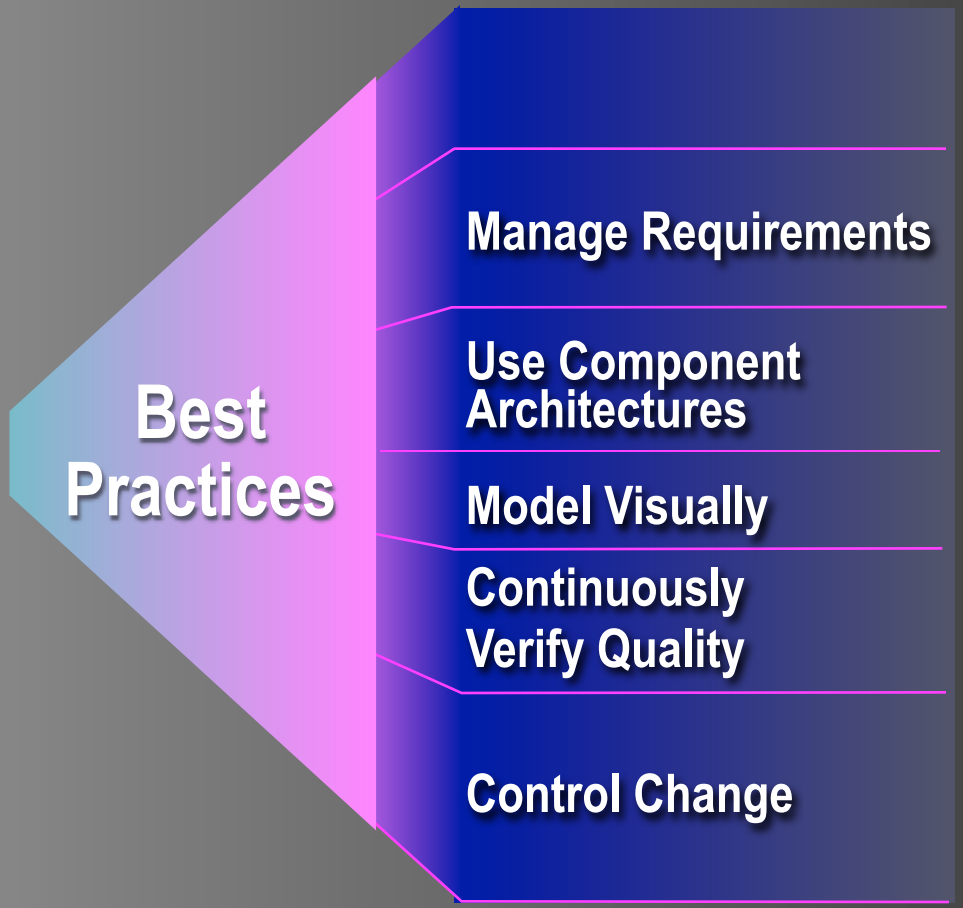
- Elégtelen követelmény-meghatározás
- Félreérthető kommunikáció
- Nem eléggé robosztus az architektúra
- Túlságosan nagy komplexitású a rends
- Nem detektált inkonzisztencia
- Alacsony fokú kiteszteltség
- Szubjektív értékelés
- Vízesés modellen alapuló fejlesztés
- Nem kontrollált változtatás
- Elégtelen automatizáció



Mit jelentenek a „legjobb gyakorlatok”?

„Elvek, módszerek és folyamatok egy szervezett és dokumentált halmaza, amely növeli a szoftverfejlesztési minőségét és produktivitását.”

Practice 1 – Fejlessz iteratívan!

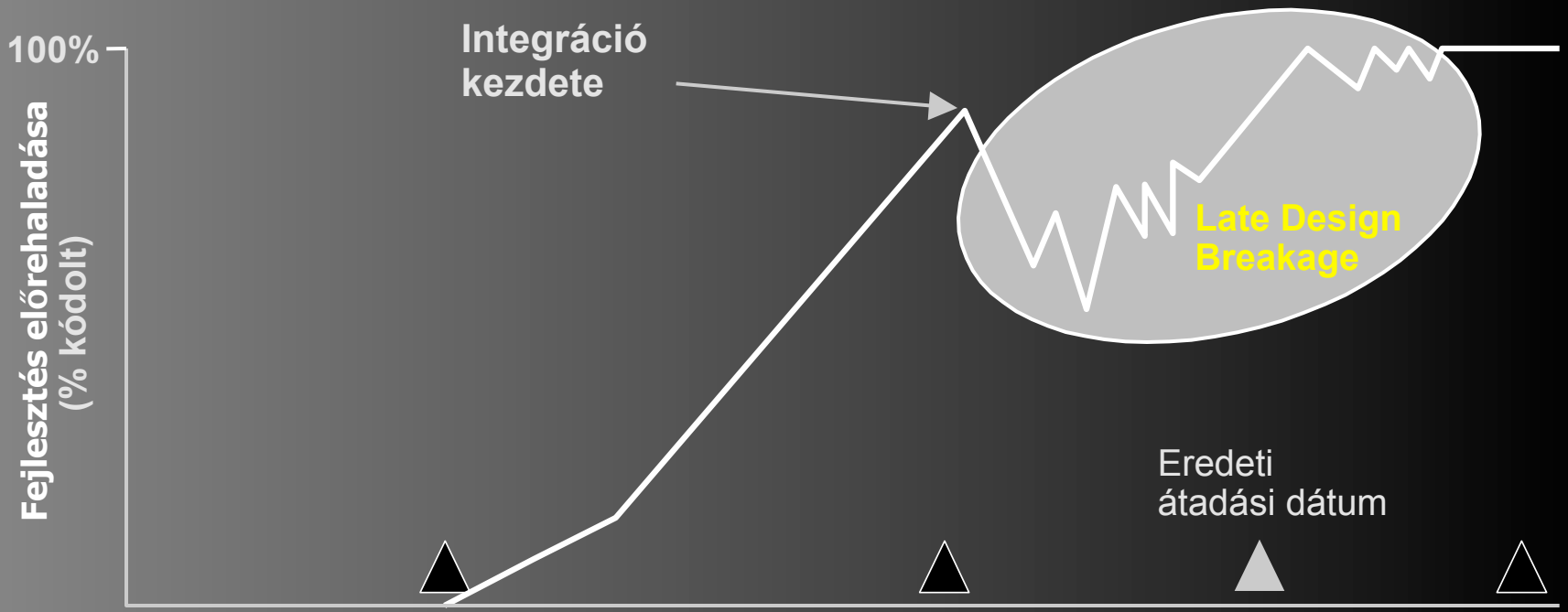


Develop Iteratively

Szekvenciális fejlesztés

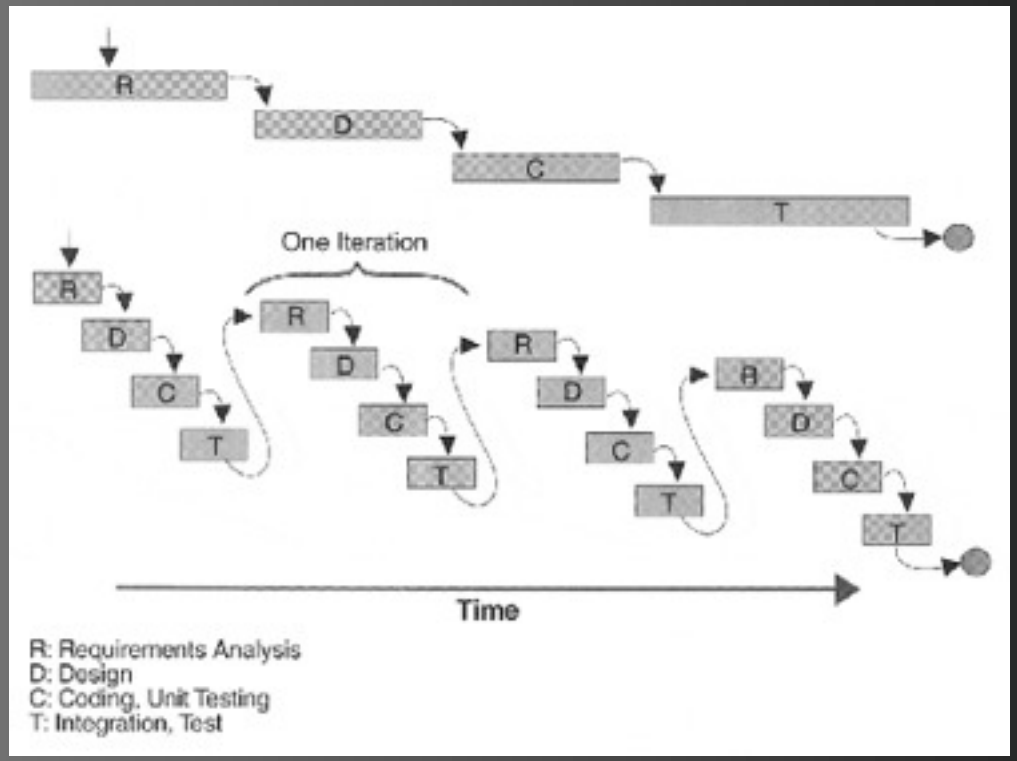
Szekvenciális tevékenységek:

Requirements → Design → Code → Integration → Test



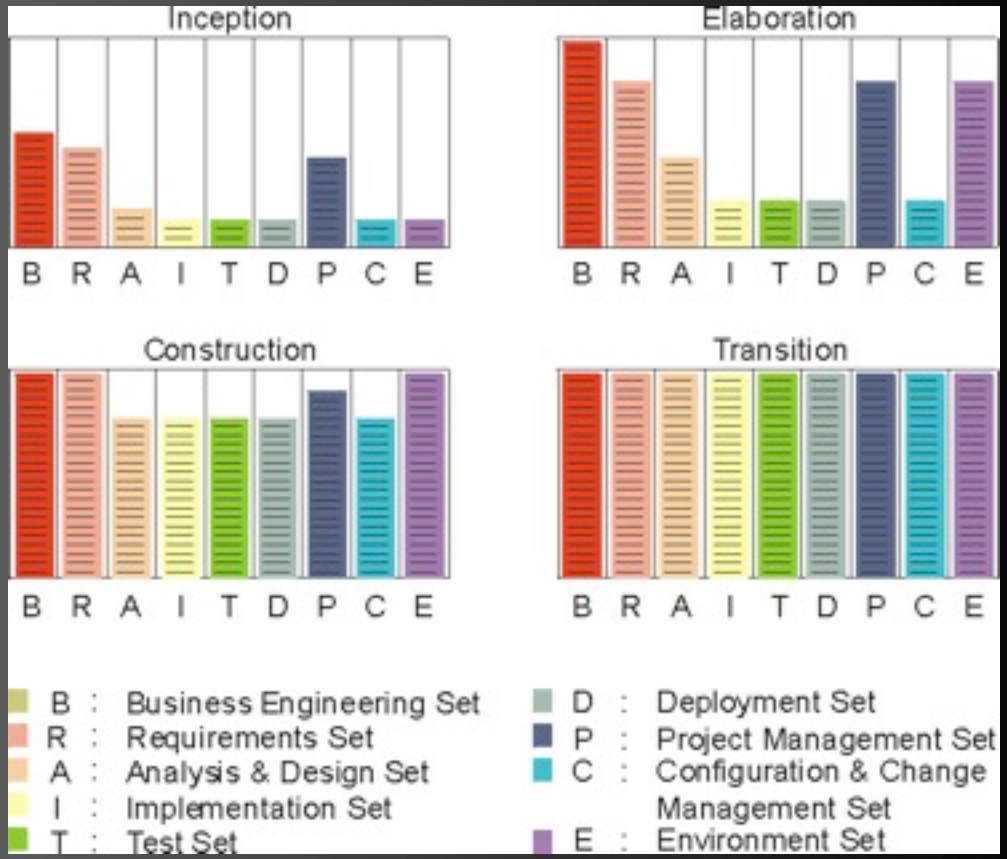
A szekvenciális és az iteratív fejlesztési folyamat összehasonlítása - 1

Folyamatok időbeli lefutása:

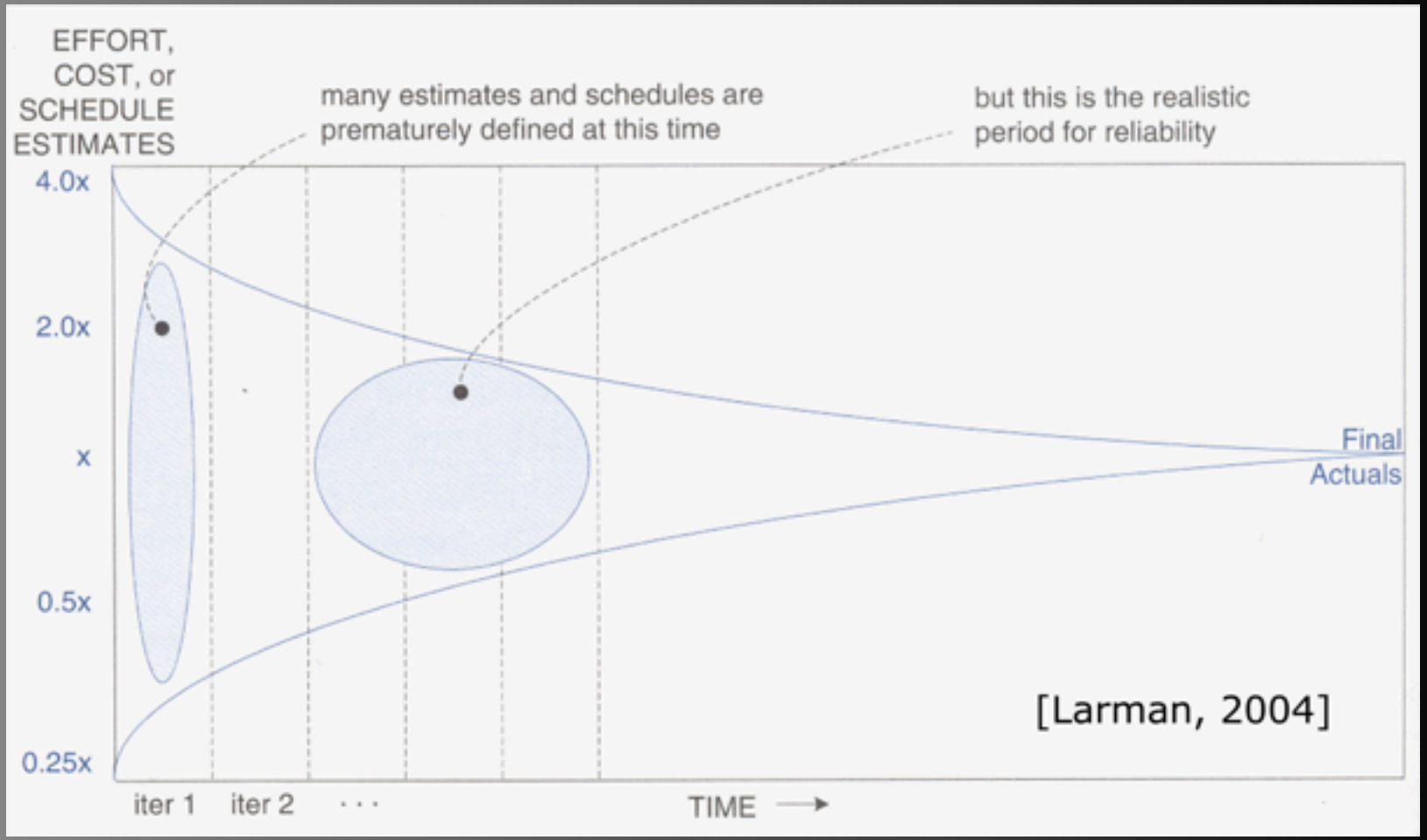


A termékek evolúciója a fejlesztési

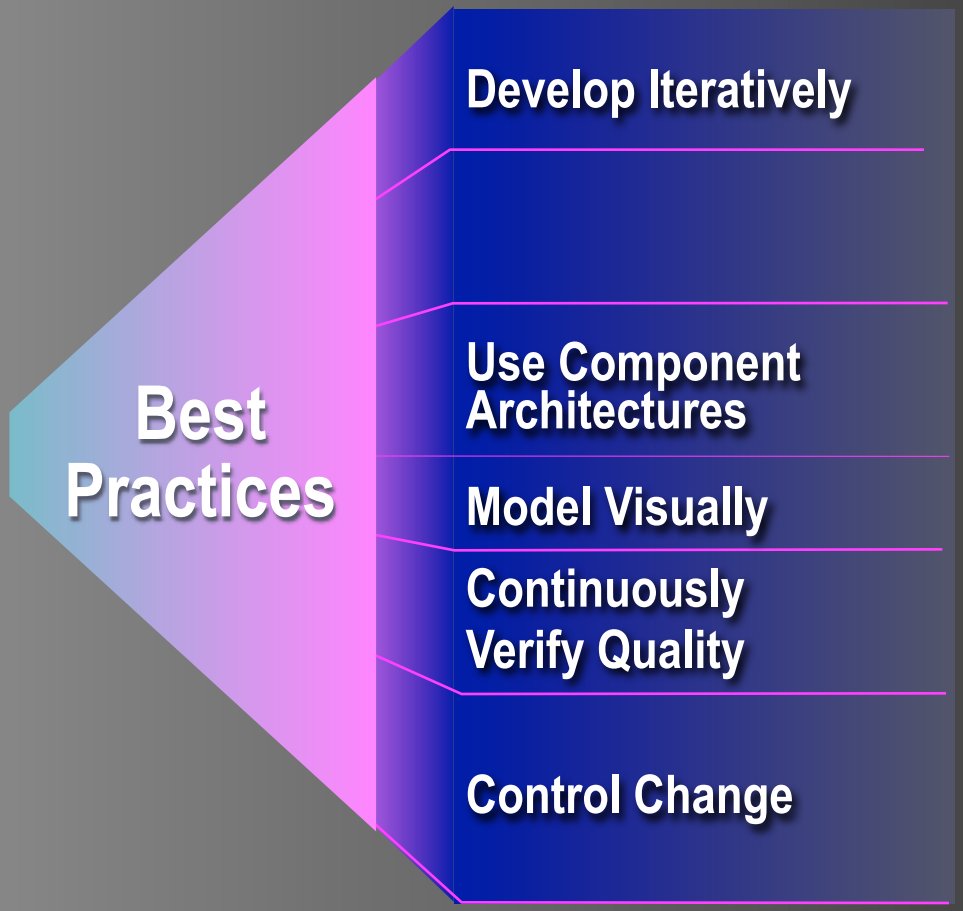
Az iteratív megközelítés miatt az idő előrehaladtával a termékek egyre „érettebbek” lesznek.



Bizonytalansági kúp

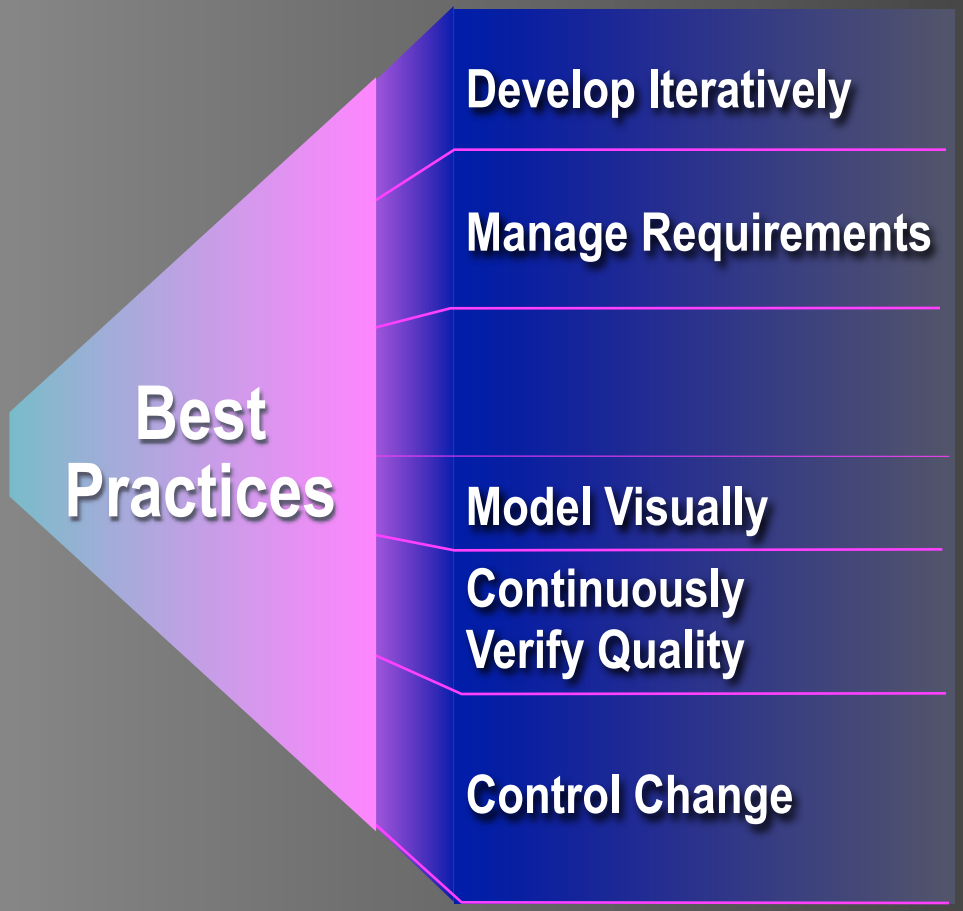


Practice 2 – Kezeld a követelményeket!



Manage Requirements

Practice 3 – Használj komponens architektúrákat

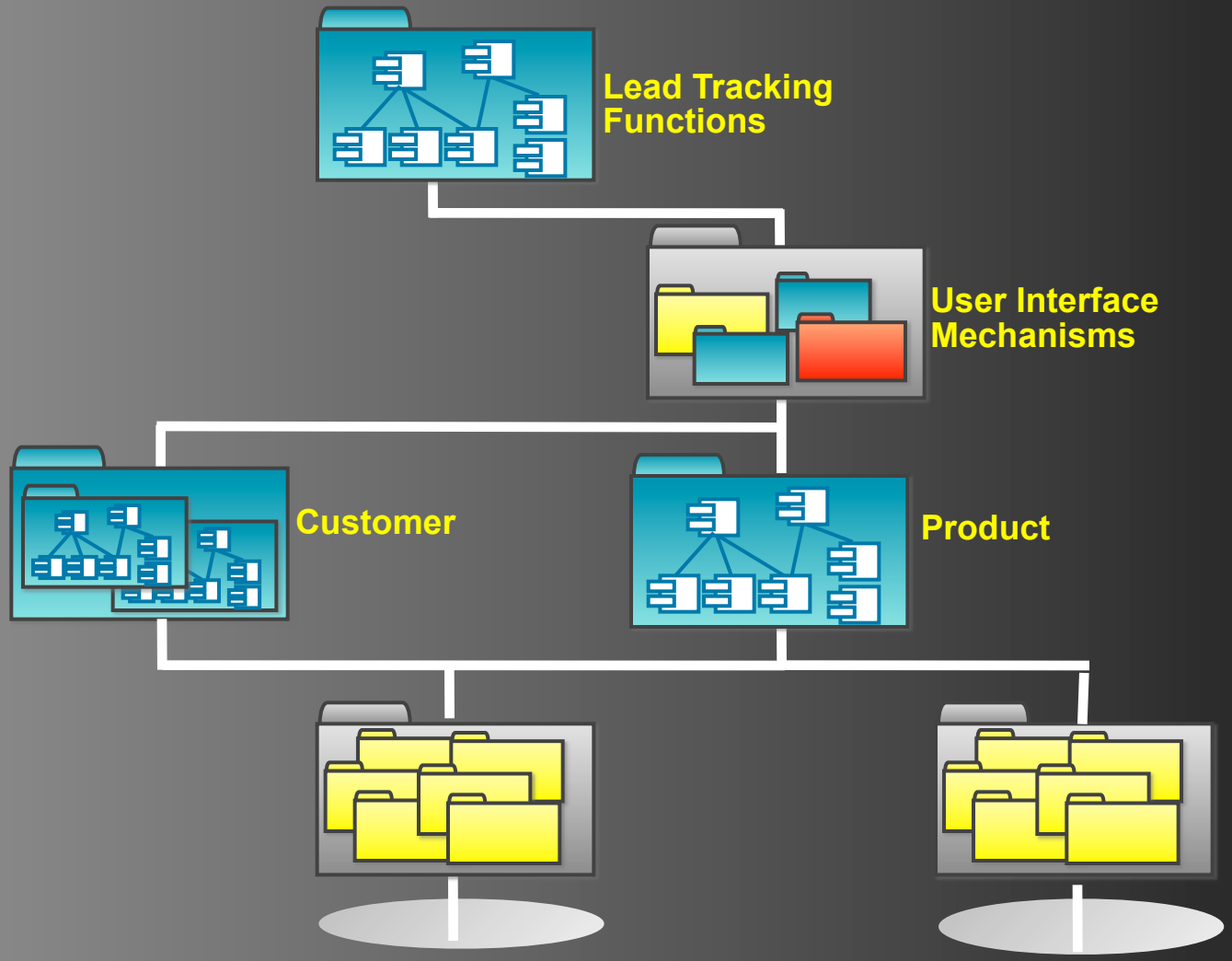


Use Component Architectures

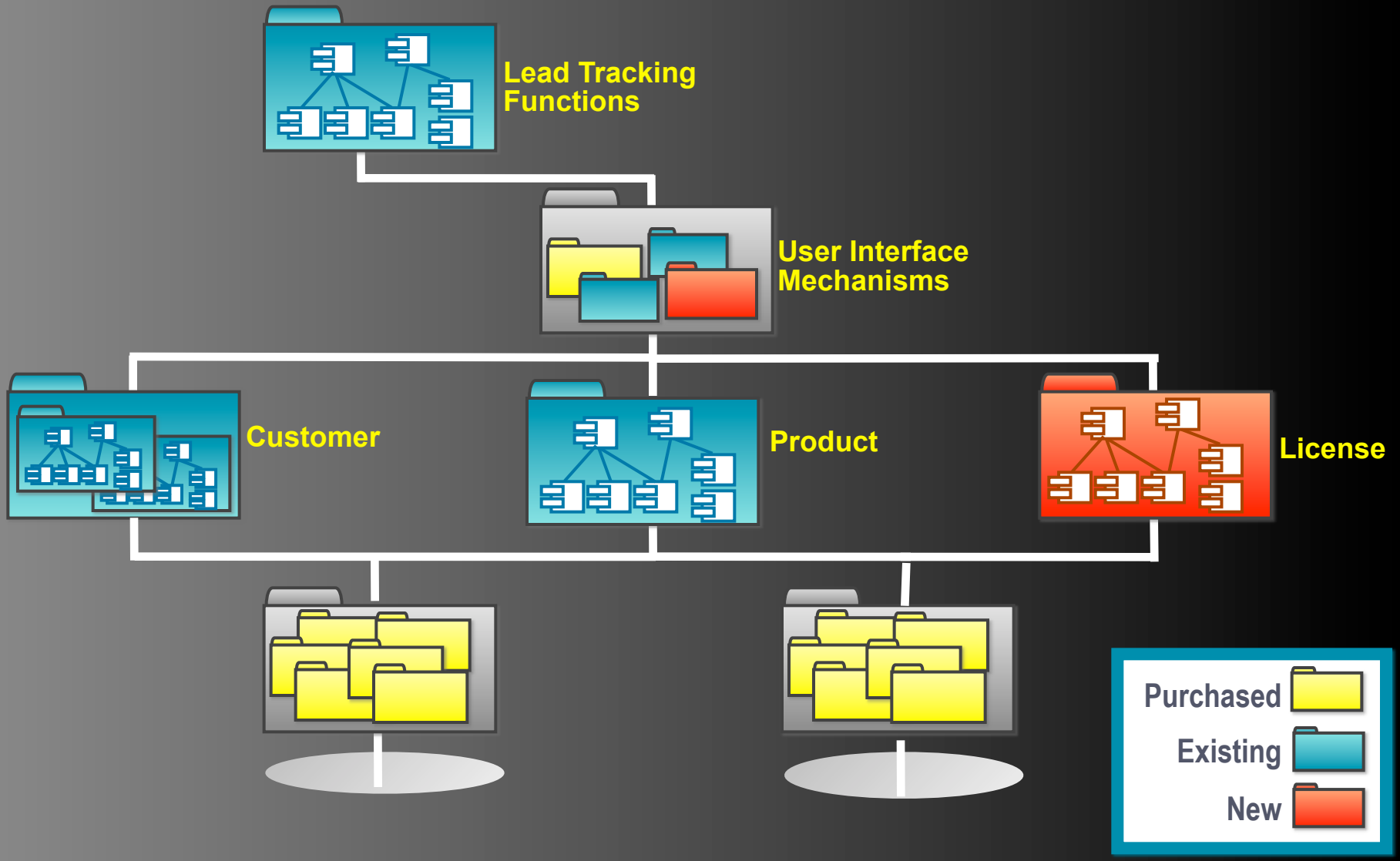
Komponens alapú architektúra előnye

- Komplexitás jobban kezelhető
- Integritás jobban biztosítható
- Újrafelhasználást támogatja:
  - Komponens újrafelhasználás
  - Architektúra újrafelhasználás
- Projekt menedzsmentet támogatja:
  - Szoftver tervezés
  - Emberi erőforrás tervezés

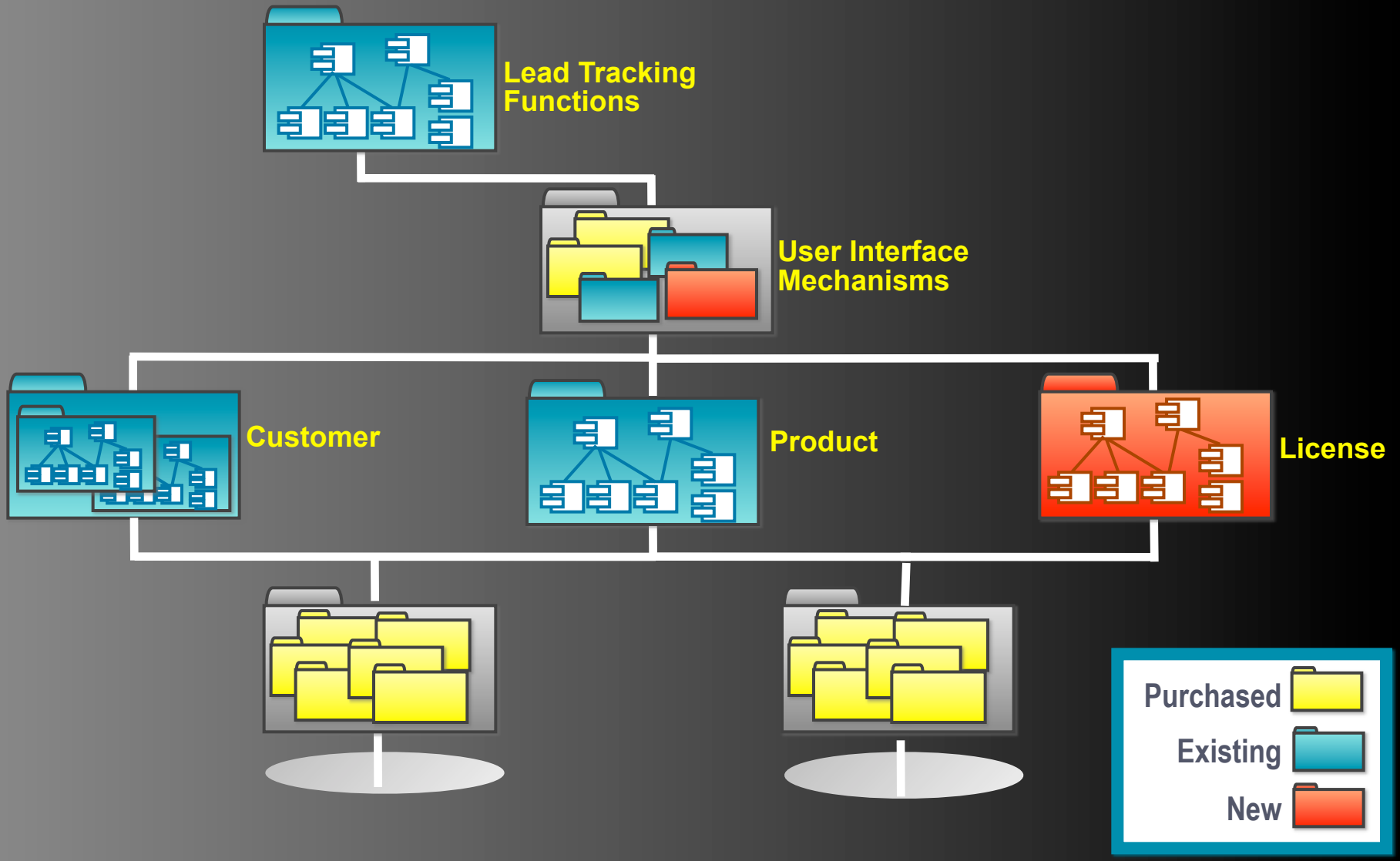
Példa: komponens alapú architektúra



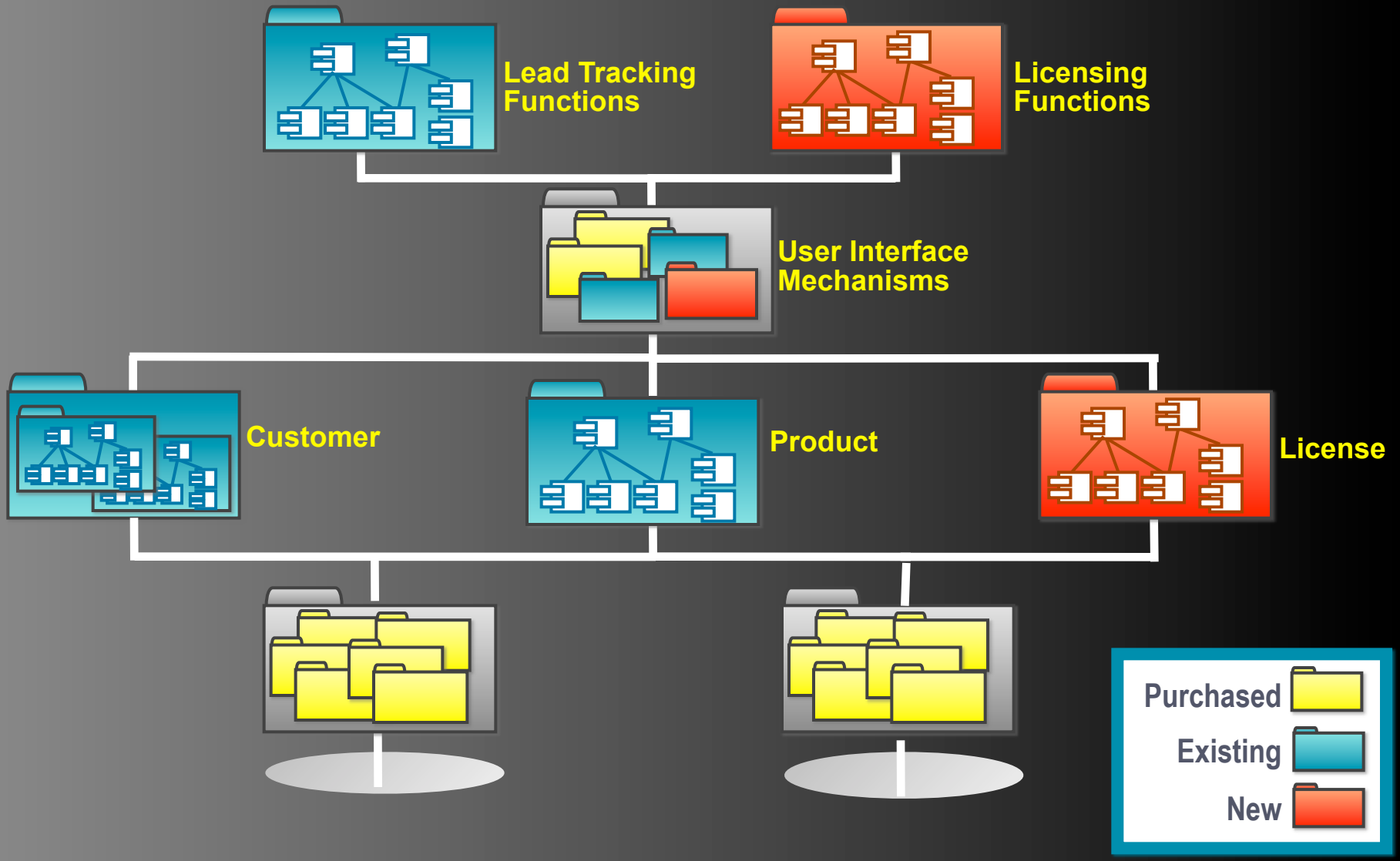
Példa: komponens alapú architektúra



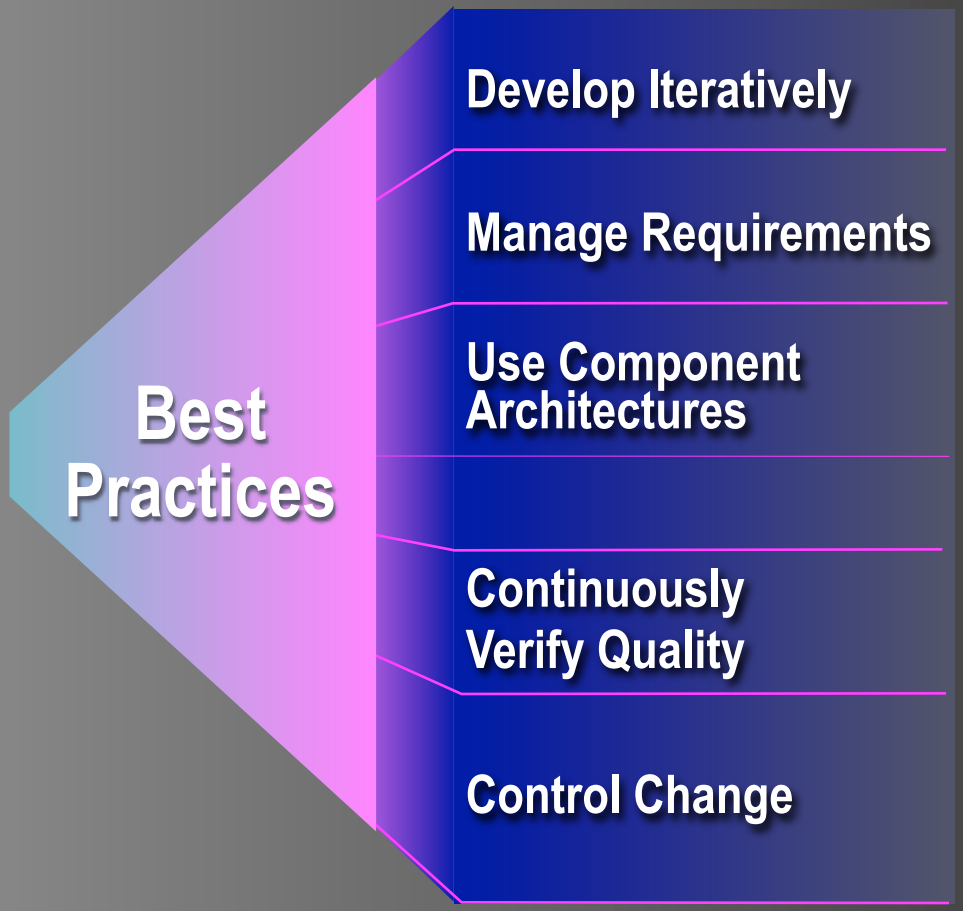
Példa: komponens alapú architektúra



Példa: komponens alapú architektúra



Practice 4 – Modellezz vizuálisan!



Model Visually (UML)

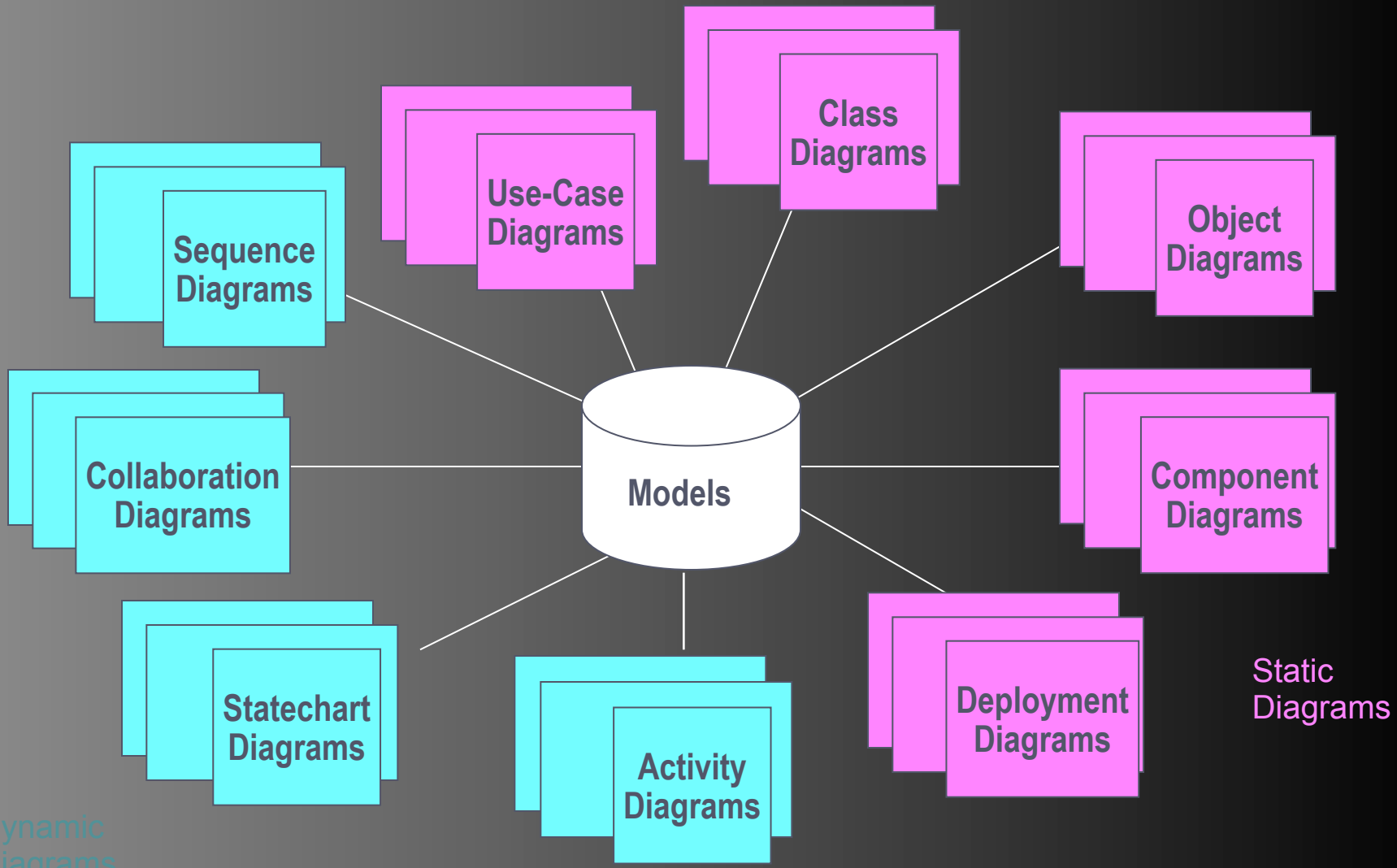
Reprezentáció: az UML

Az UML, a szoftver-intenzív rendszerek termékeinek

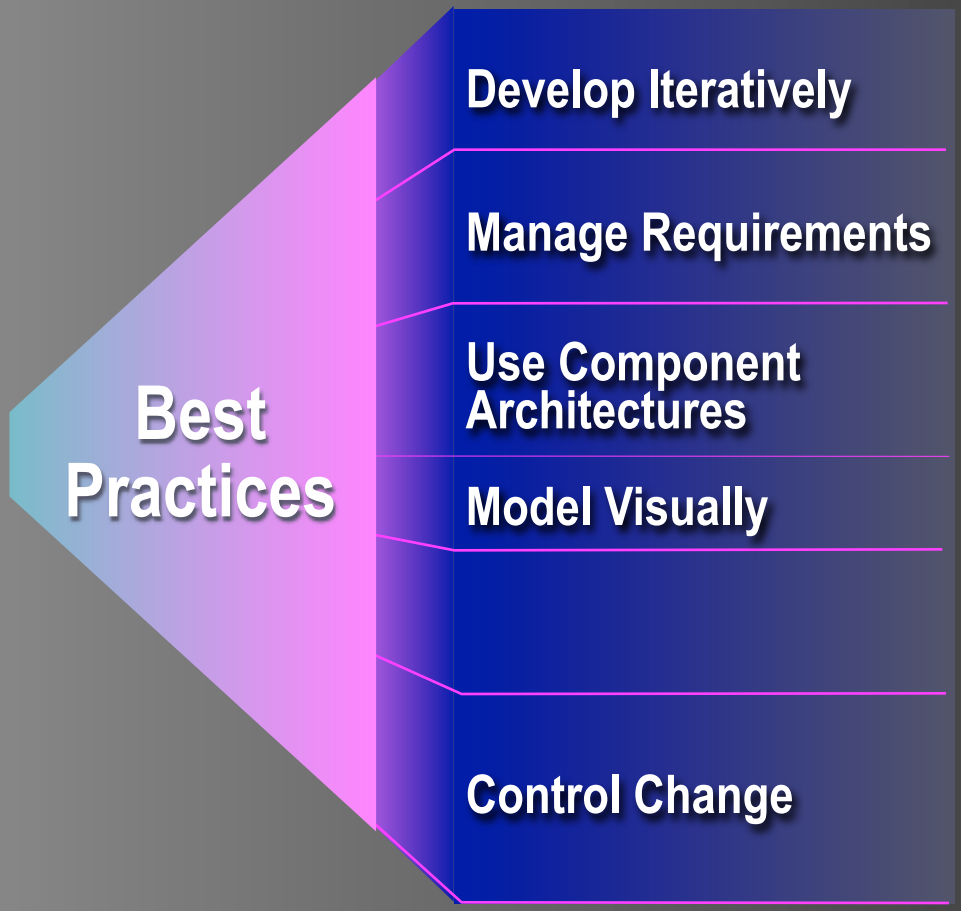
- vizuális megjelenítésre,
- specifikálásra,
- tervezésre, és
- dokumentálásra

kidolgozott jelölésrendszer.

Vizuális modellezés az UML-lel



Practice 5 – Folyamatosan ellenőrizd a szoftver minőségét!



Continuously Verify Quality

Minőség méréséhez: szoftver metrikák alkalmazása

- Termék metrikák

pl. méret, funkcionalitás, komplexitás, struktúra

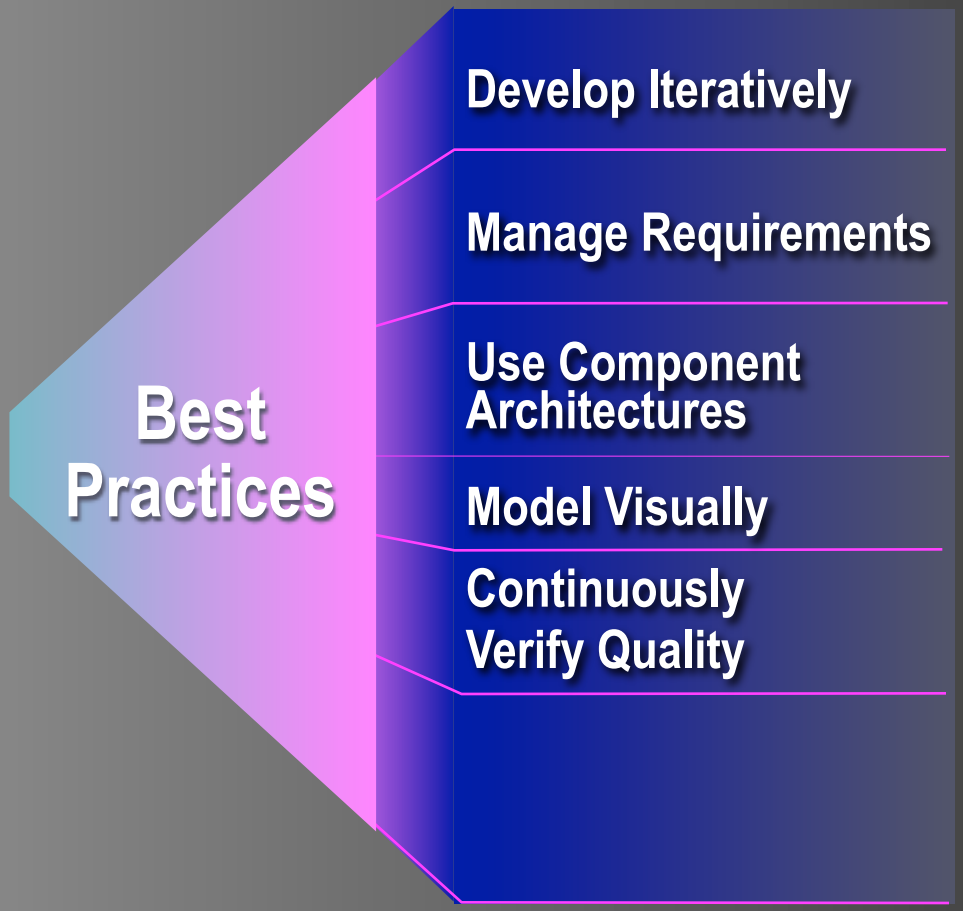
- Folyamat (fejlesztés és támogatás) metrikák

pl. hibaszám, követelményeknek való megfelelőség

- Projekt metrikák

pl. produktivitás, ütemezés, költség, munkaerő-minta

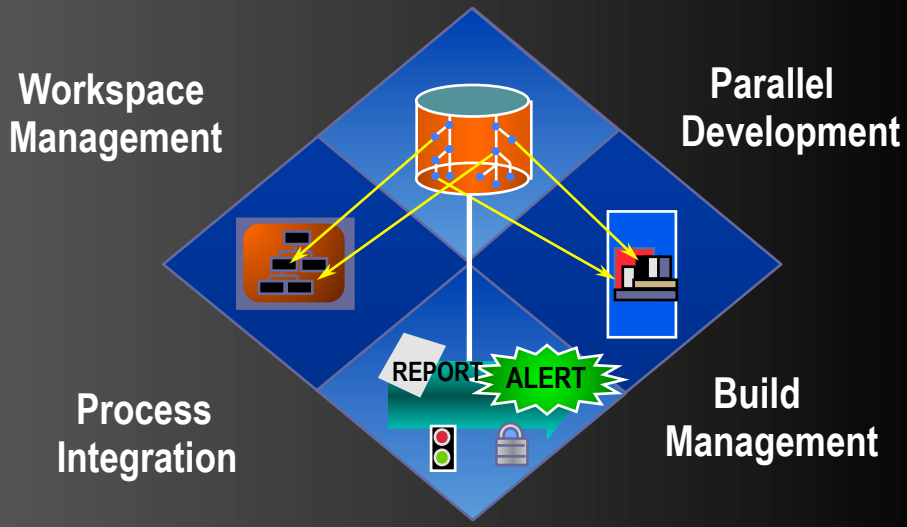
Practice 6 – Kezeld a szoftver változásait!



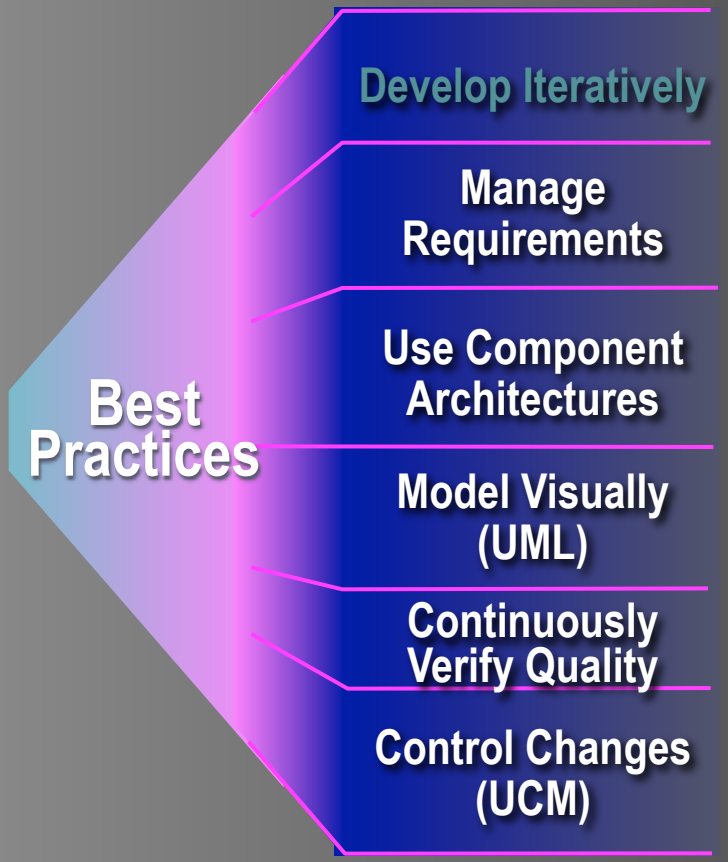
Control Change (UCM)

Mire való a változáskezelés?

- Iteratív fejlesztésből adódó változások vezérléséhez, nyomon követéséhez és monitorozásához.
- Minden fejlesztőnek biztonságos workspace-t biztosítson.
- Automatikus integrációhoz és build menedzsmenthez.



A Best Practice-ek egymást erősítik



A felhasználók bevonása a követelmények meghatározásába.

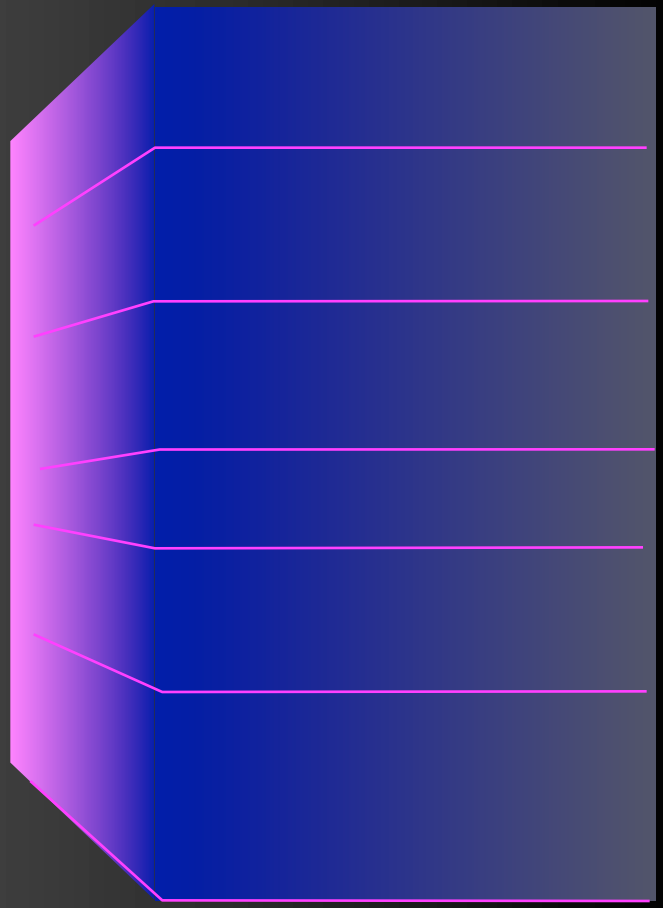
Architektúra korai validációja.

A tervezés és implementáció komplexitása fokozatosan nő.

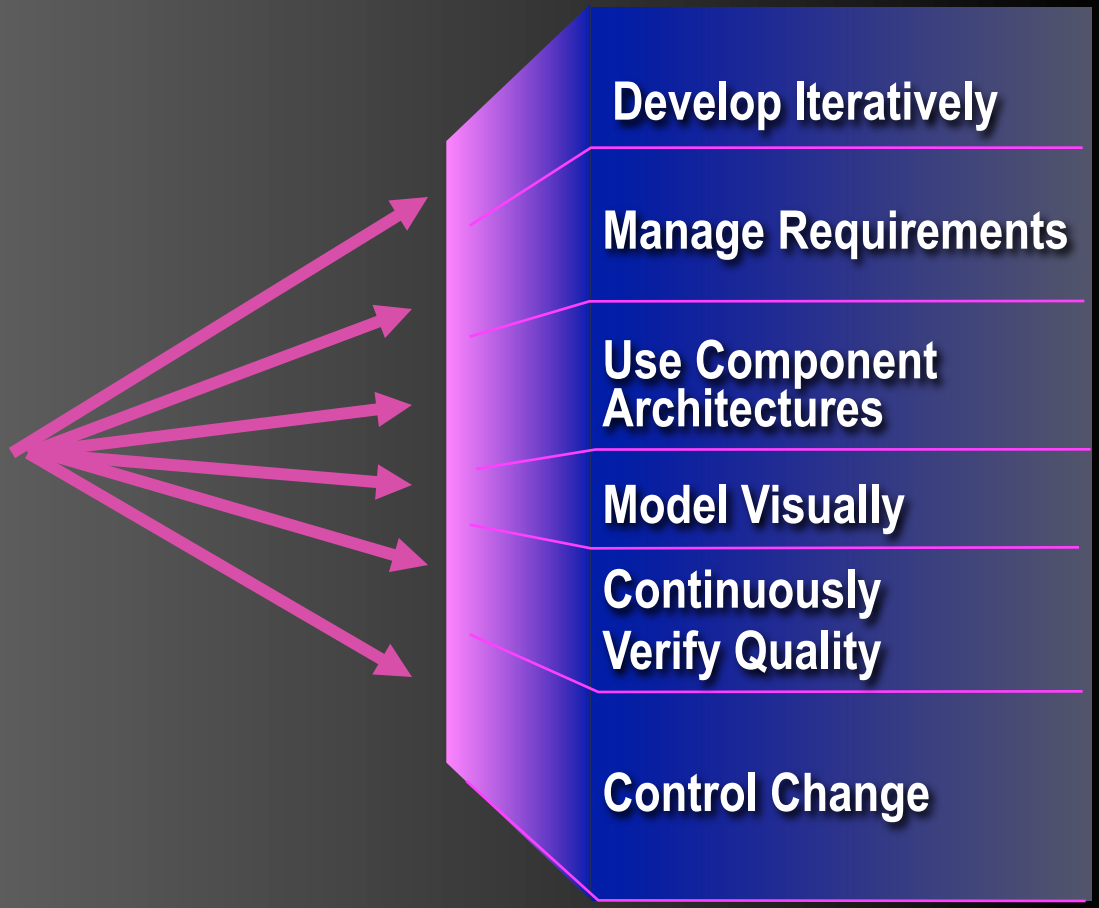
A minőség korai és gyakori mérése.

Változások folyamatos nyomonkövetése.

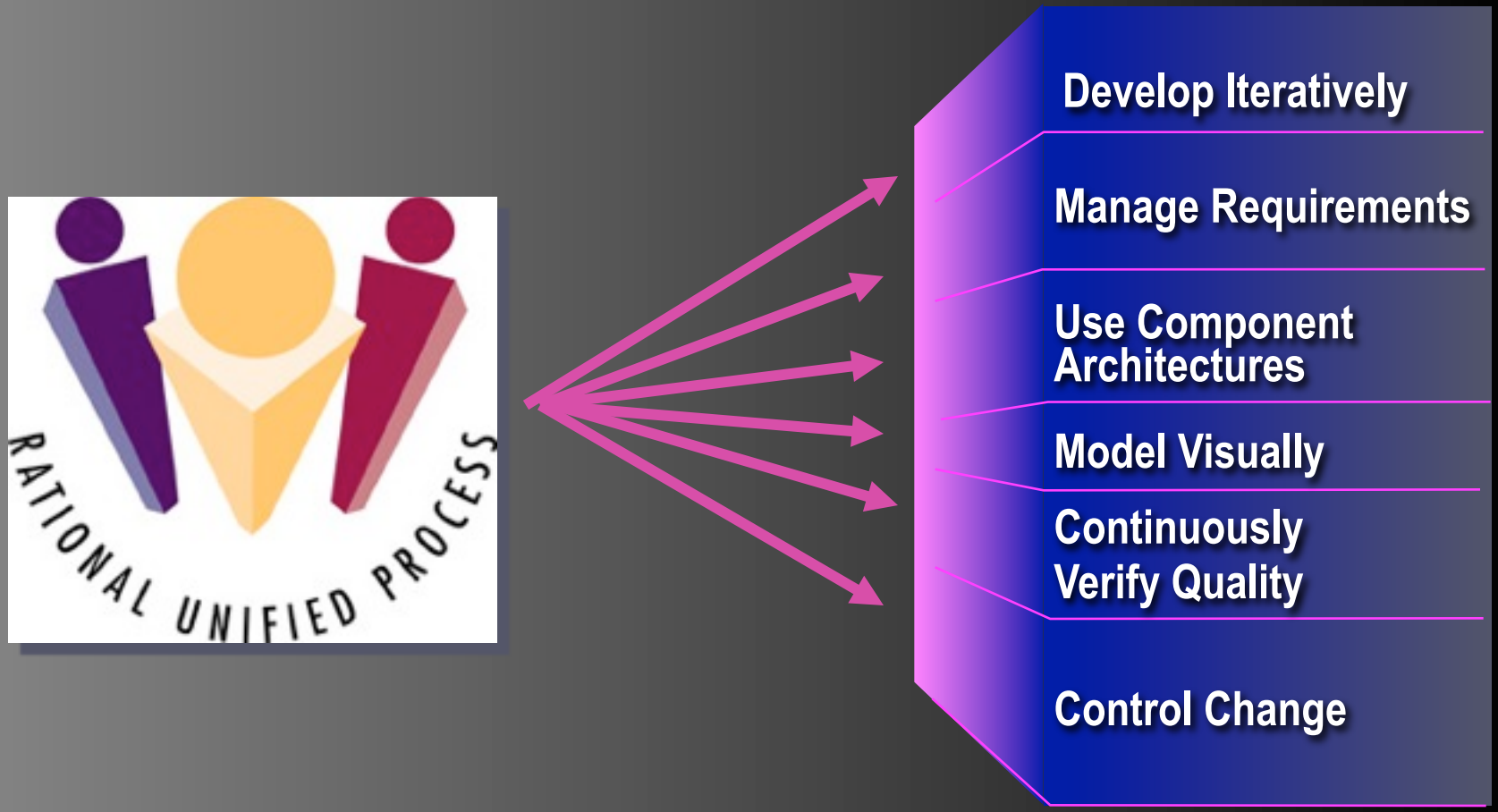
A RUP ezen Best Practice-okon alapul



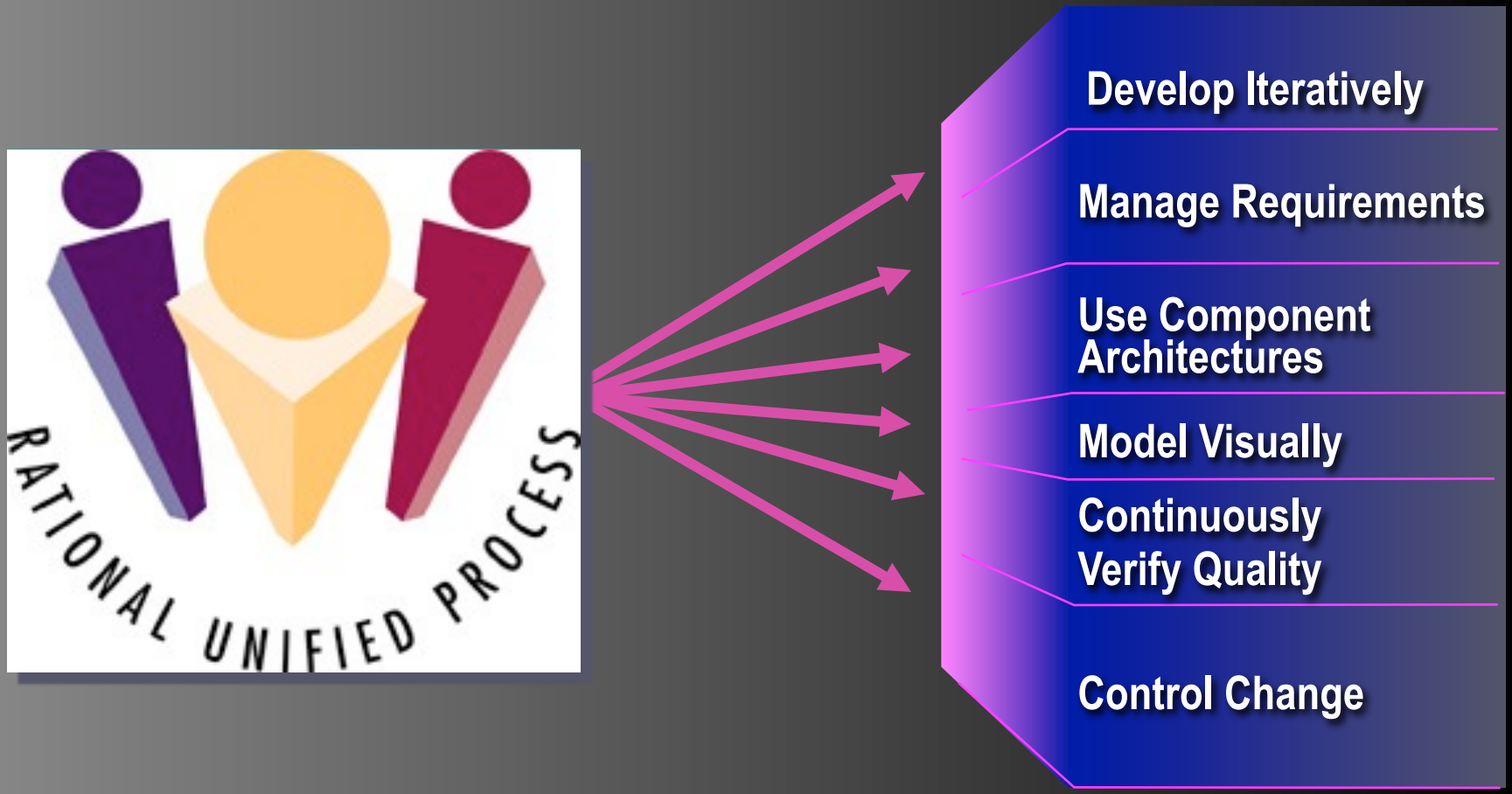
A RUP ezen Best Practice-okon alapul



A RUP ezen Best Practice-okon alapul



A RUP ezen Best Practice-okon alapul



A Rational Unified Process (RUP)

- RUP elveinek bemutatása
- RUP elemeinek bemutatása
- RUP fázisok bemutatása
- RUP, mint termék bemutatása

A Rational Unified Process (RUP)

- RUP elveinek bemutatása
- RUP elemeinek bemutatása
- RUP fázisok bemutatása
- RUP, mint termék bemutatása

Mi a RUP?

Mi a RUP?

Szoftverfejlesztési elv, mely:

- Use-case vezérelt
- Architektúra központú
- Iteratív

Mi a RUP?

Szoftverfejlesztési elv, mely:

- Use-case vezérelt
- Architektúra központú
- Iteratív

Szoftverfejlesztési folyamat, mely:

- Jól definiált (ki, mit, mikor, hogyan)
- Jól strukturált (életciklus, mérföldkövek, döntések)

Mi a RUP?

Szoftverfejlesztési elv, mely:

- Use-case vezérelt
- Architektúra központú
- Iteratív

Szoftverfejlesztési folyamat, mely:

- Jól definiált (ki, mit, mikor, hogyan)
- Jól strukturált (életciklus, mérföldkövek, döntések)

Termék a szoftverfejlesztéshez, mely

- Testre szabható (projekt méret, ceremónia)
- A projekt összes tagjának szolgáltatásokat nyújt

A RUP, mint szoftverfejlesztési elv motivációja

- A megrendelői igények kielégítése a cél, ezért ennek kell a központban lenni a folyamat során
- Minden jól működő rendszernek egy egyszerű, jól működő architektúrája van
- A szoftverfejlesztés bizonytalanságának a csökkentése a fokozatos előrehaladással

A RUP, mint szoftverfejlesztési elv motivációja

- A megrendelői igények kielégítése a cél, ezért ennek kell a központban lenni a folyamat során

Use case vezérelt

- Minden jól működő rendszernek egy egyszerű, jól működő architektúrája van

- A szoftverfejlesztés bizonytalanságának a csökkentése a fokozatos előrehaladással

A RUP, mint szoftverfejlesztési elv motivációja

- A megrendelői igények kielégítése a cél, ezért ennek kell a központban lenni a folyamat során

Use case vezérelt

- Minden jól működő rendszernek egy egyszerű, jól működő architektúrája van

Architektúra centrikus

- A szoftverfejlesztés bizonytalanságának a csökkentése a fokozatos előrehaladással

A RUP, mint szoftverfejlesztési elv motivációja

- A megrendelői igények kielégítése a cél, ezért ennek kell a központban lenni a folyamat során

Use case vezérelt

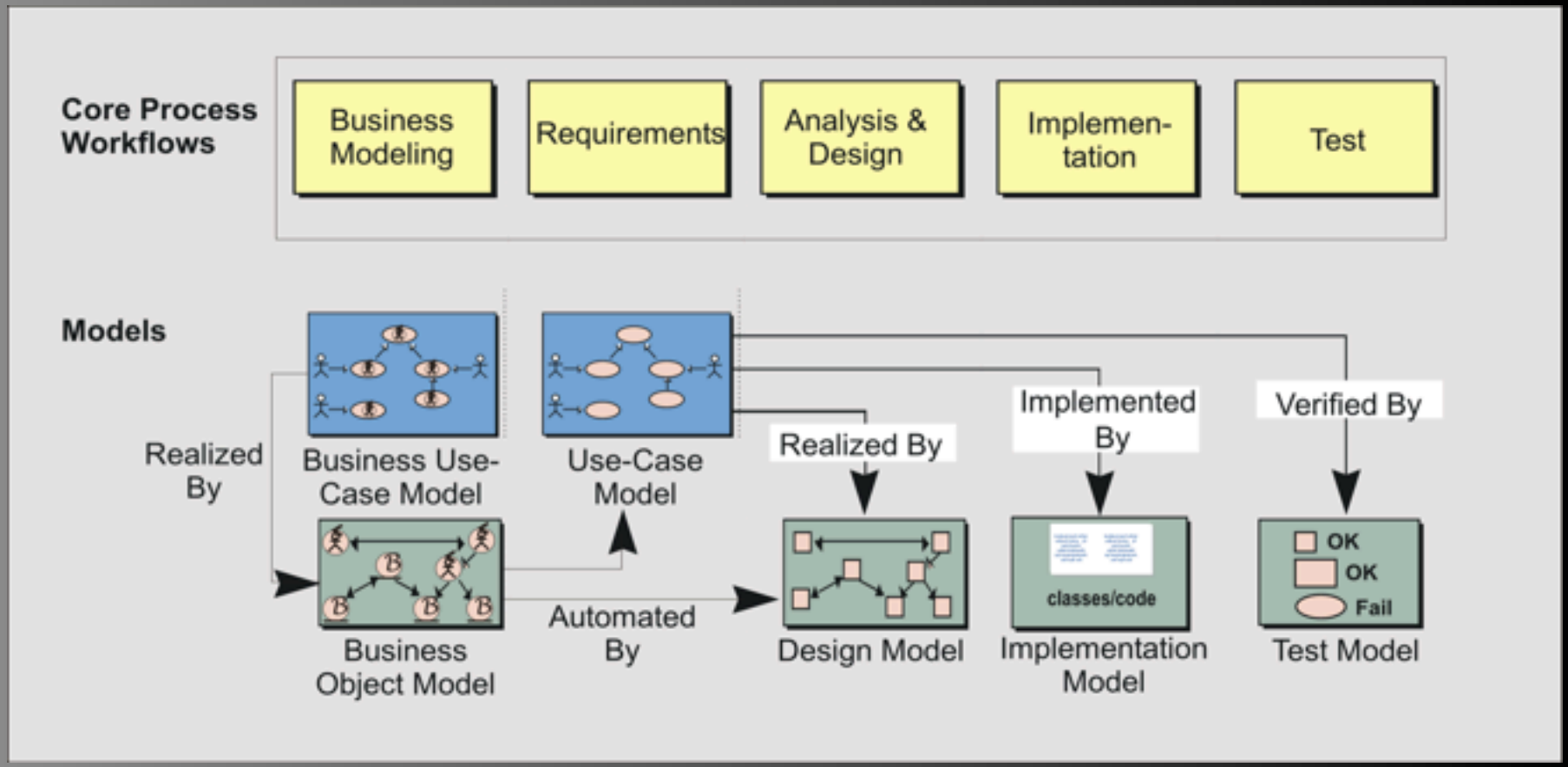
- Minden jól működő rendszernek egy egyszerű, jól működő architektúrája van

Architektúra centrikus

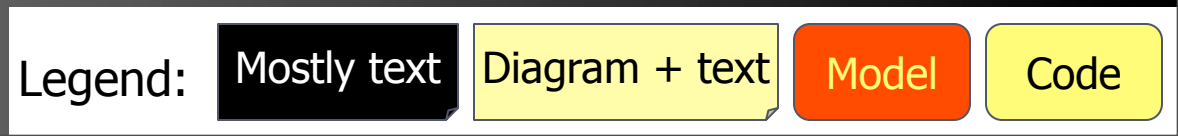
- A szoftverfejlesztés bizonytalanságának a csökkentése a fokozatos előrehaladással

Inkrementális és iteratív

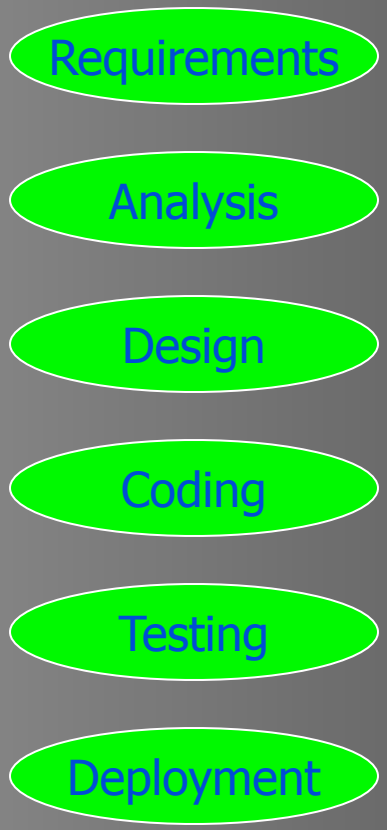
Use Case vezérelt



Architektúra centrikusság: MDA

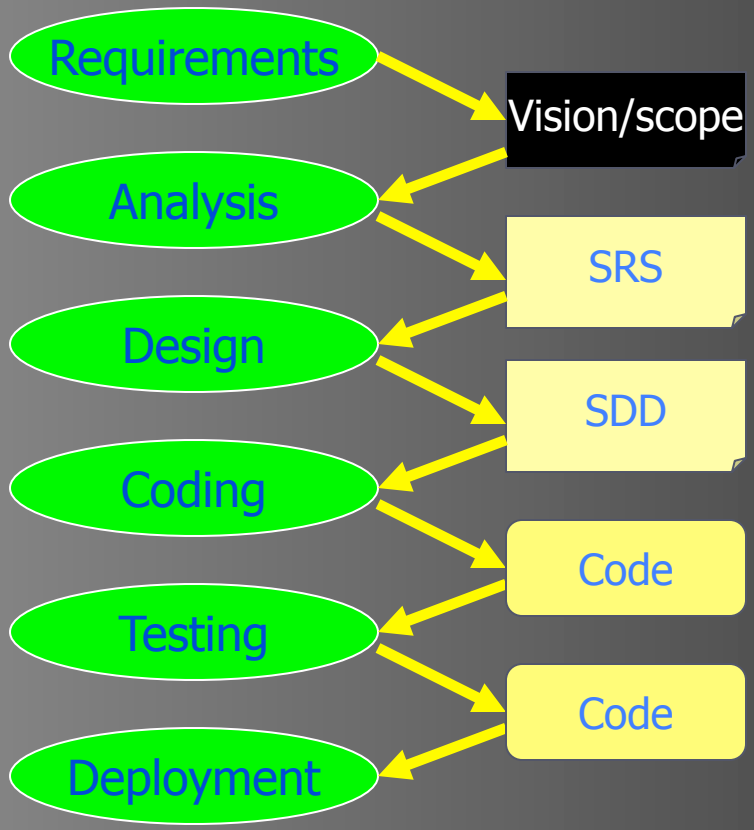


Architektúra centrikusság: MDA



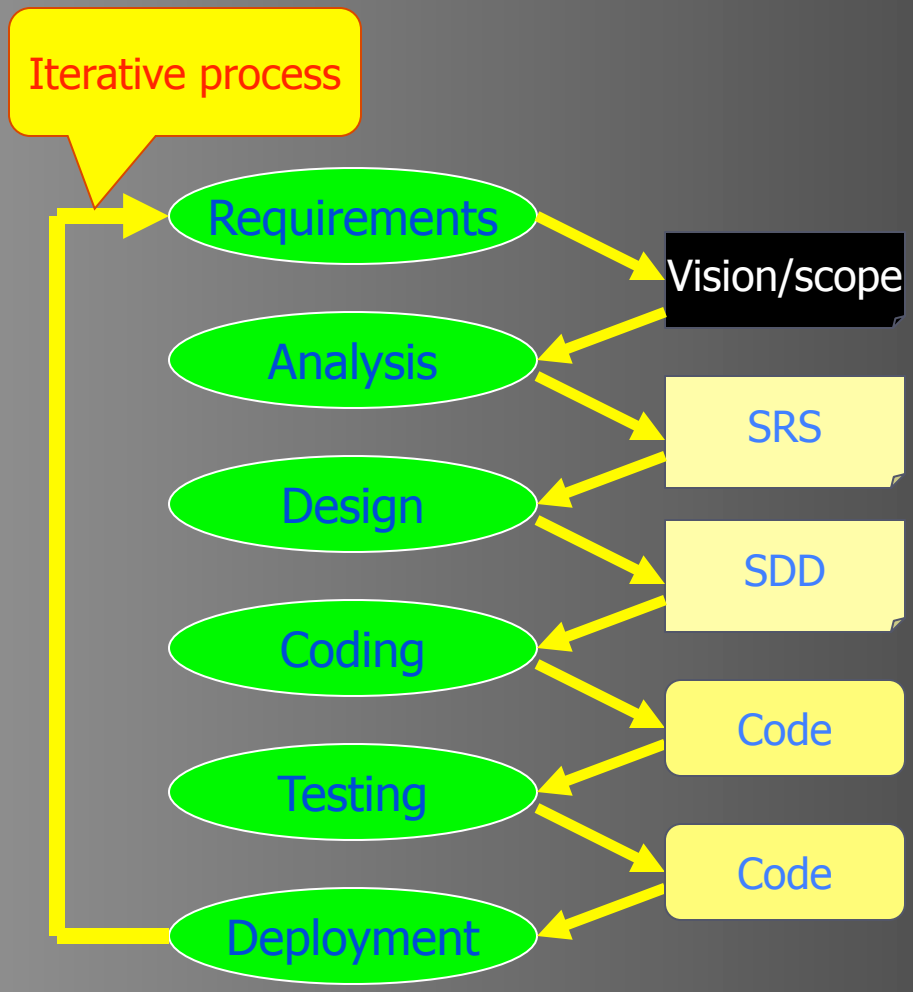
Legend: Mostly text | Diagram + text | Model | Code

Architektúra centrikusság: MDA



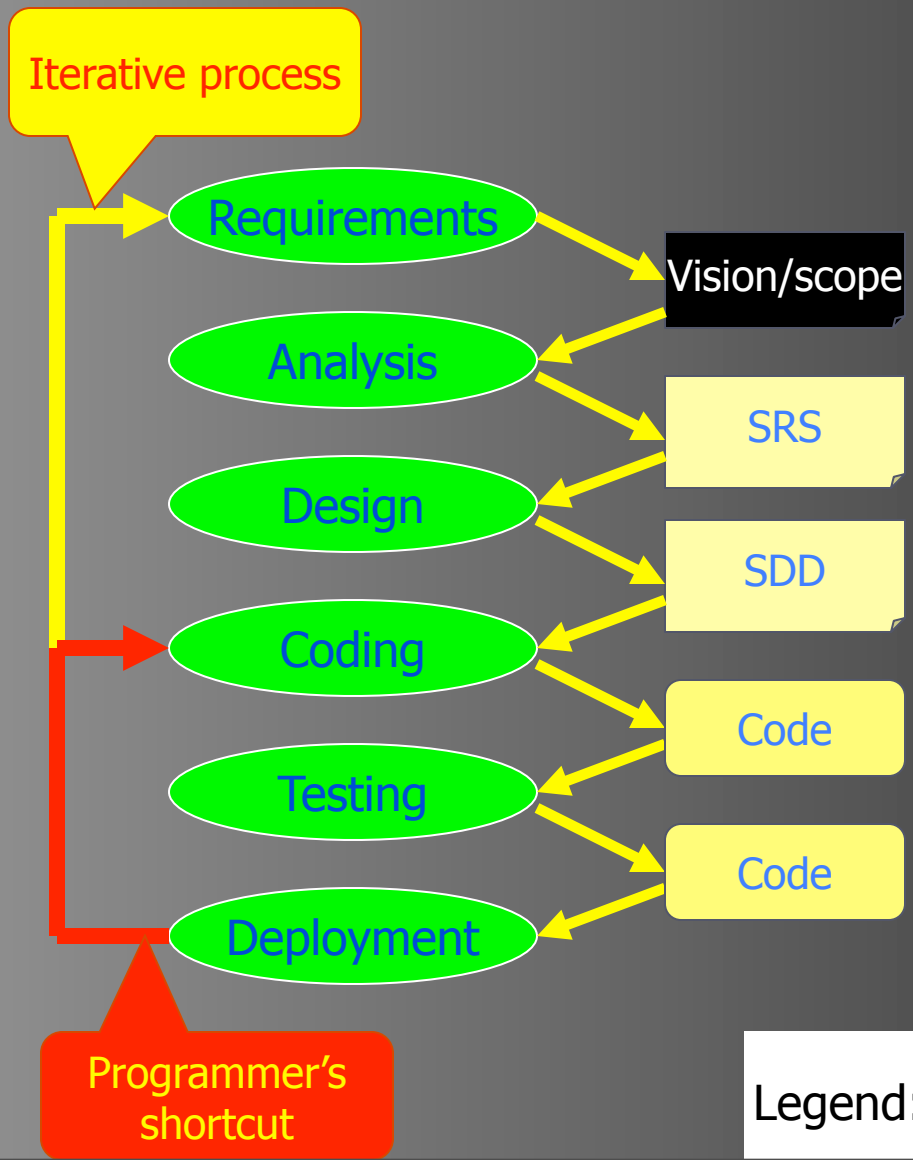
Legend: Mostly text | Diagram + text | Model | Code

Architektúra centrikusság: MDA

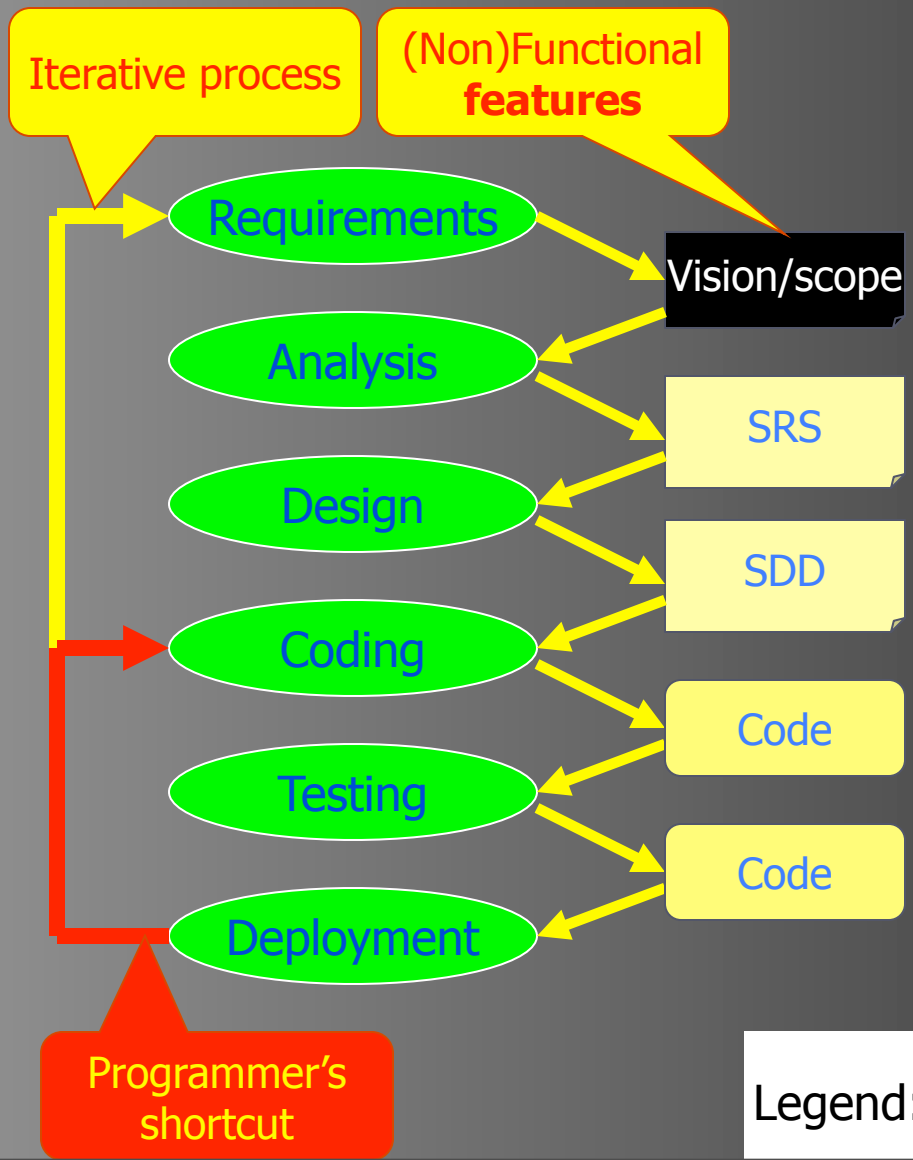


Legend: Mostly text | Diagram + text | Model | Code

Architektúra centrikusság: MDA

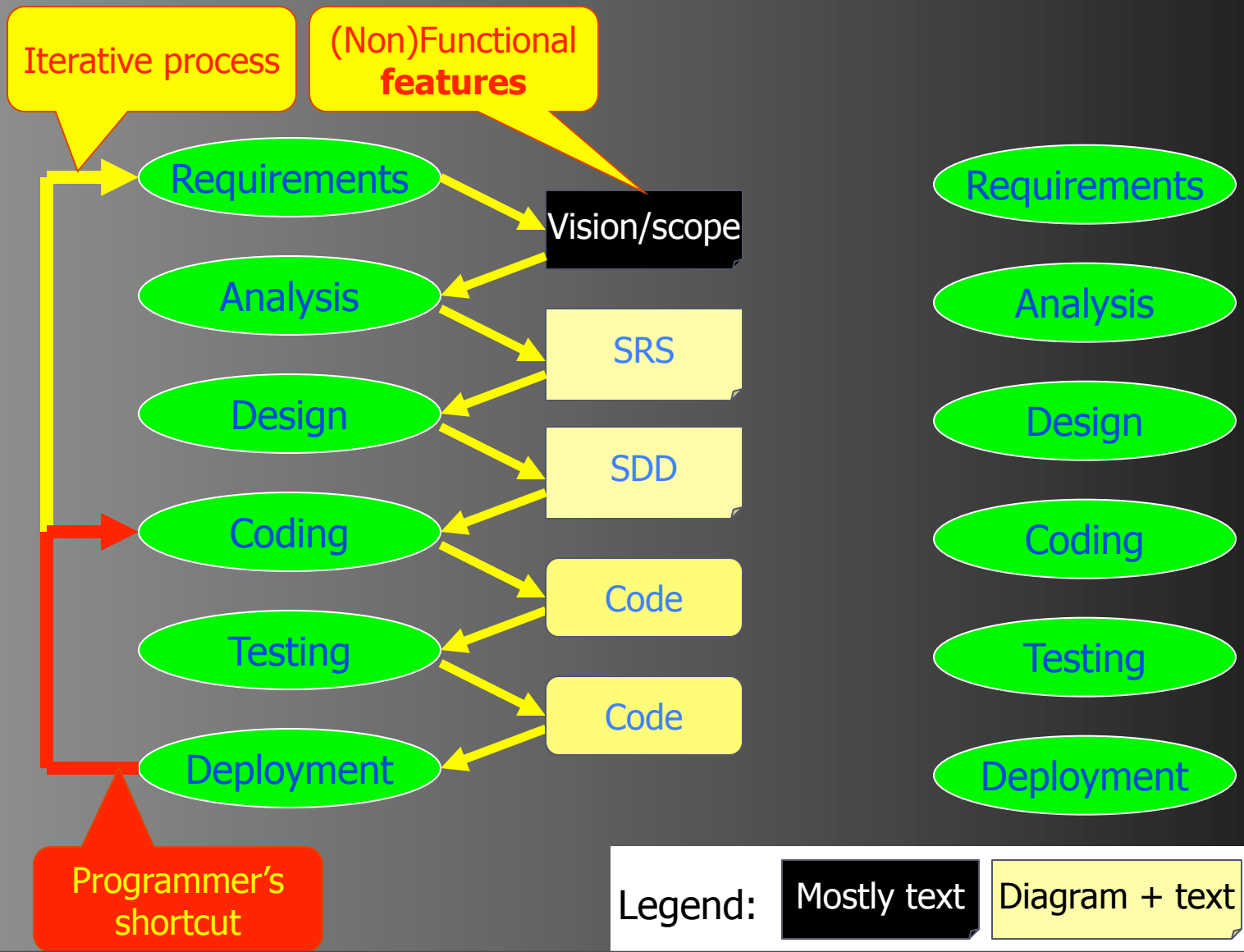


Architektúra centrikusság: MDA

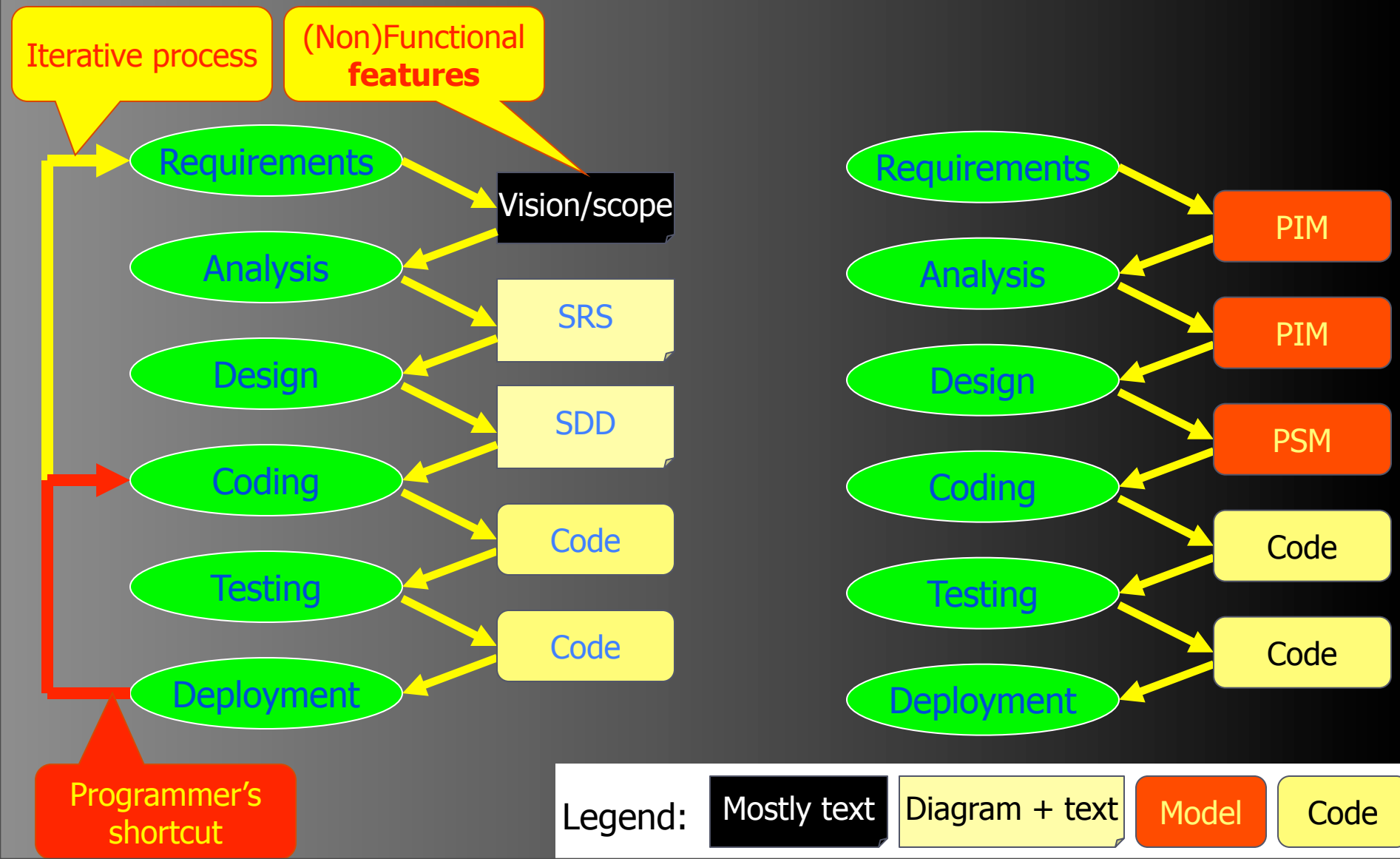


Legend: Mostly text | Diagram + text | Model | Code

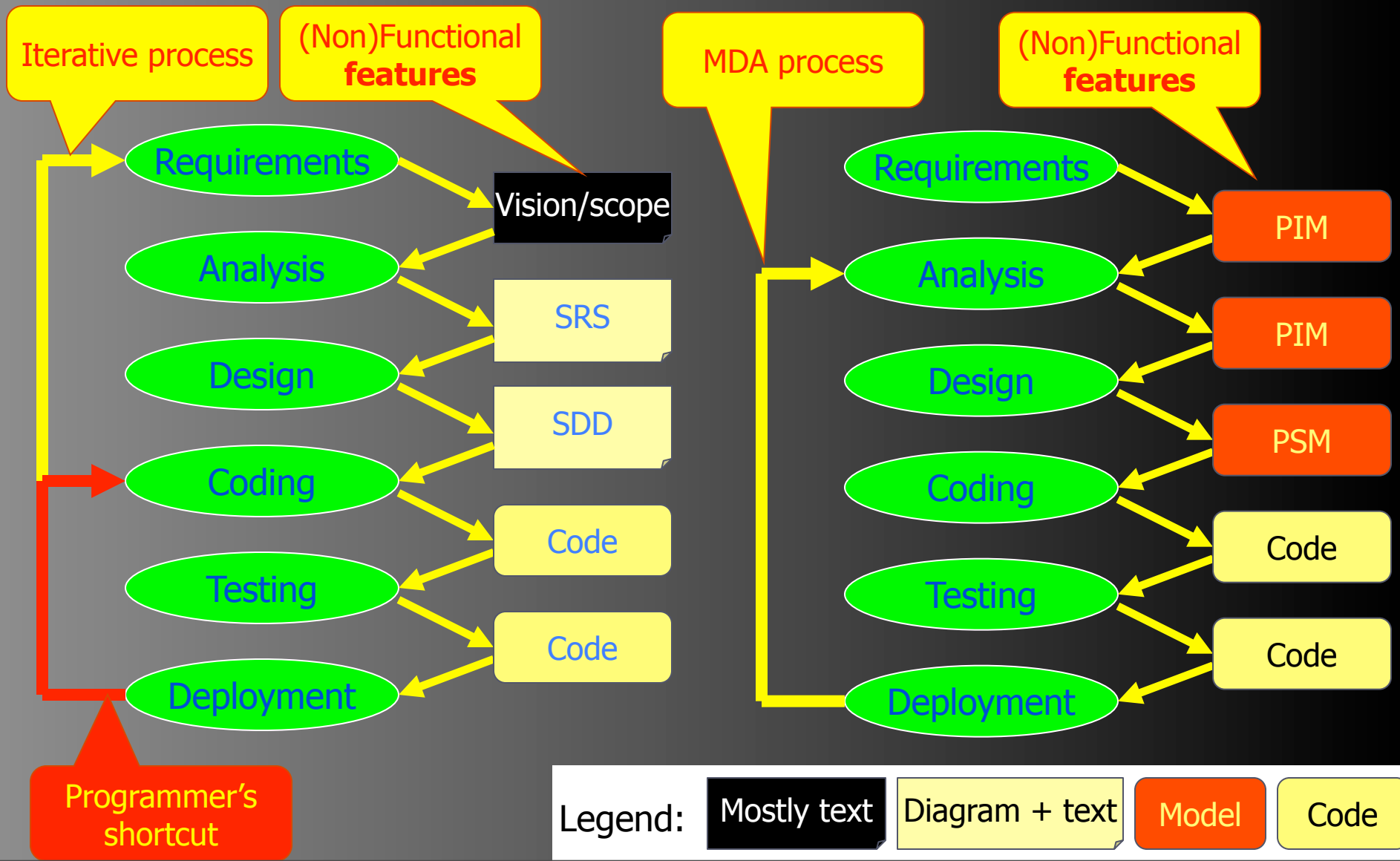
Architektúra centrikusság: MDA



Architektúra centrikusság: MDA



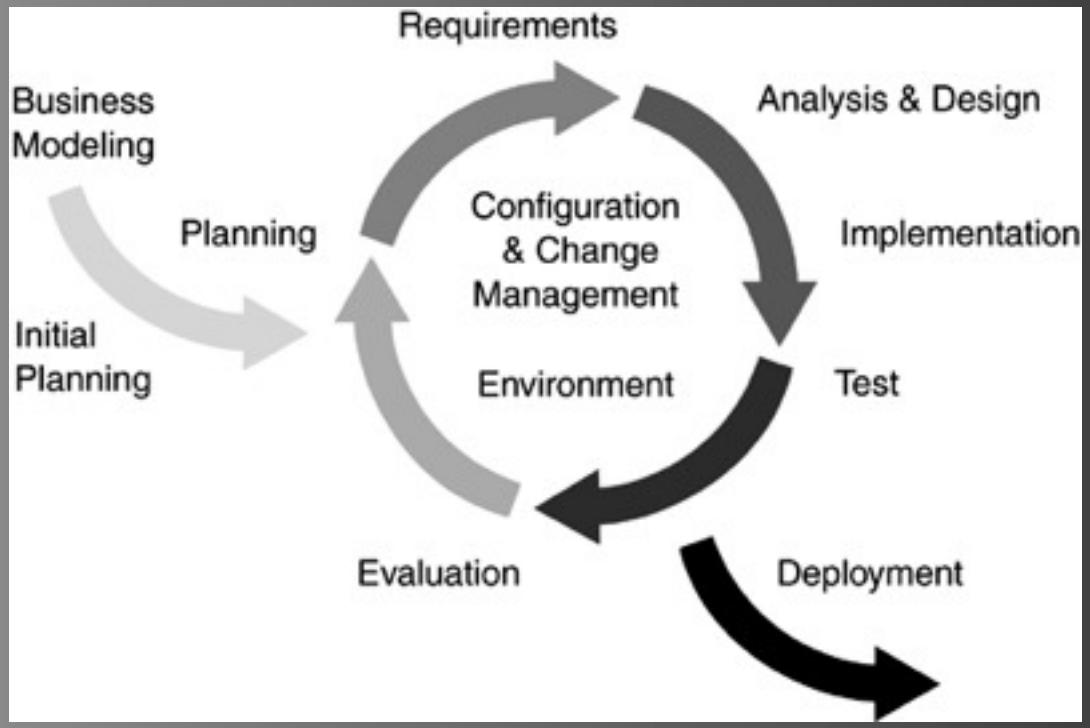
Architektúra centrikusság: MDA



Az MDA előnyei

- A befektetett tudás megőrzése
  - Különböző implementációs platform
  - Tudás explicitté tétele
- Fejlesztés felgyorsítása
  - Implementáció jelentős része generált
- Implementáció minősége
  - Szakértők transzformációs template-eket biztosítanak
- Karbantartás és dokumentáció
  - Analízis és design modell továbbírása történik meg
  - 100% nyomon követés specifikációtól a tesztig

Iteratív és inkrementális folyamat

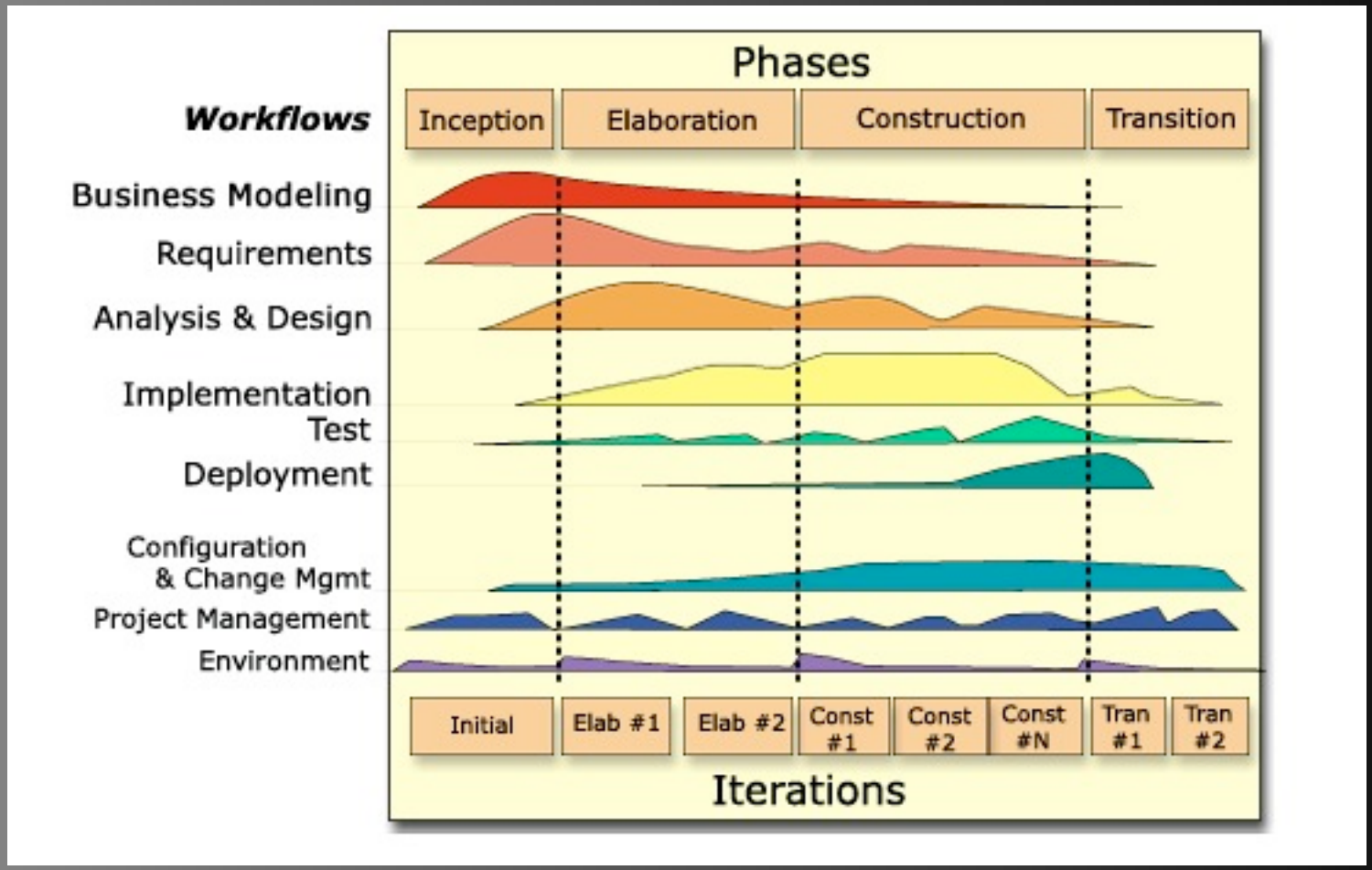


- Minden iterációban: RADIT lépések
- Minden iteráció az előzőre épül
- Konvergál a végleges termékhez

A Rational Unified Process (RUP)

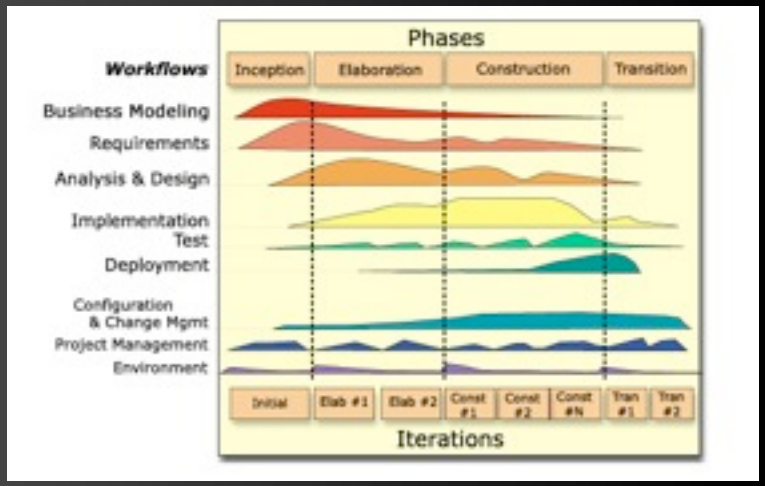
- RUP elveinek bemutatása
- RUP elemeinek bemutatása
- RUP fázisok bemutatása
- RUP, mint termék bemutatása

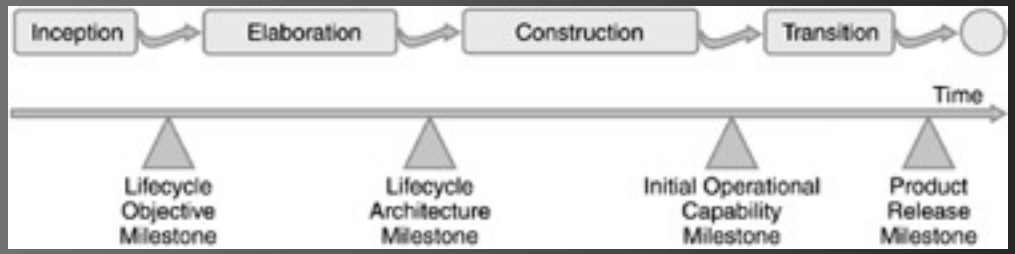
A fejlesztési folyamat két dimenziója



A folyamat dinamikus struktúrája

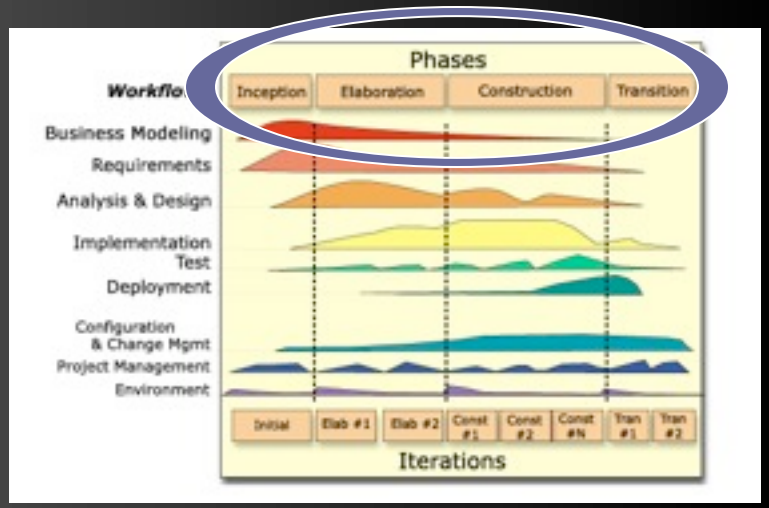
- Mind a 4 fázis végén mérföldkő és jól definiált termékek:

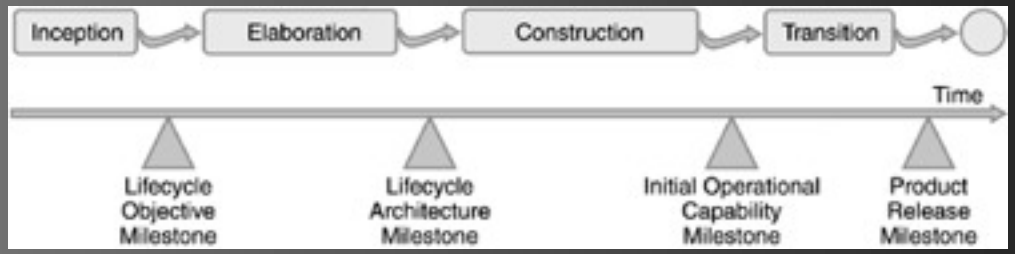




A folyamat dinamikus struktúrája

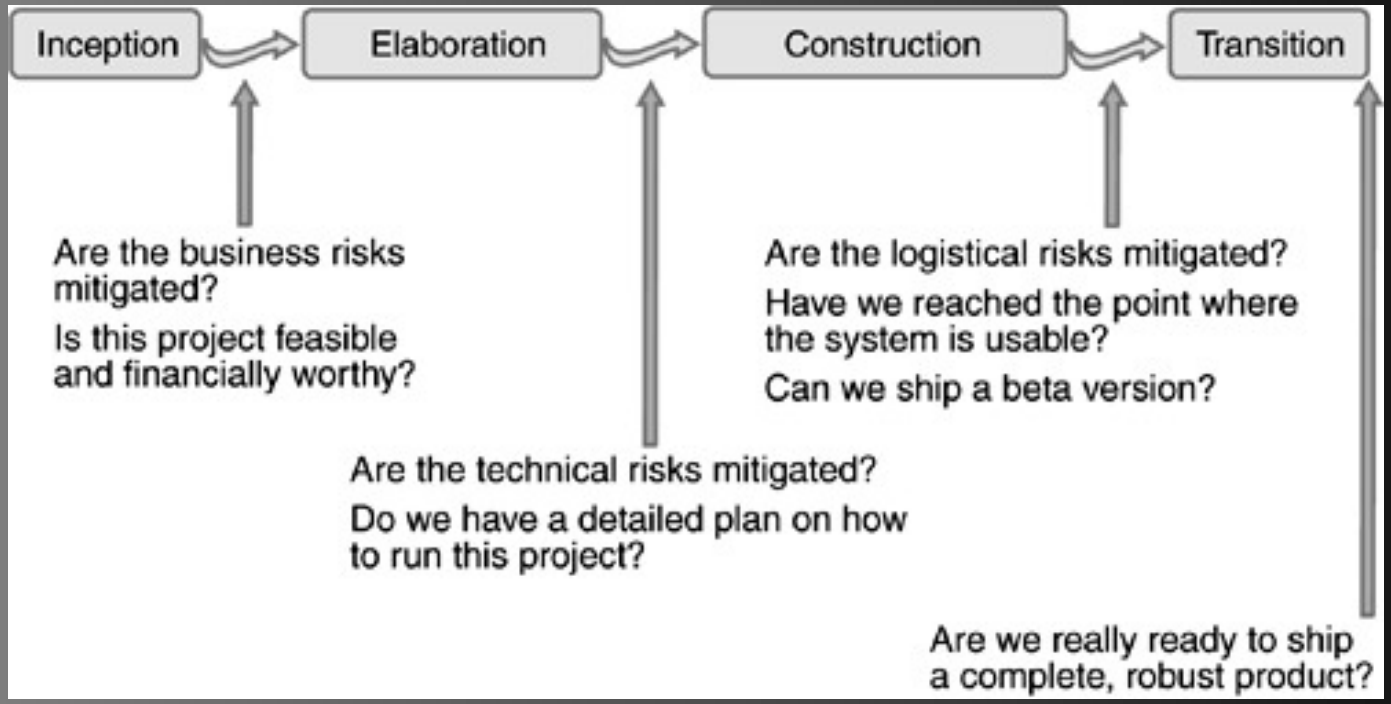
- Mind a 4 fázis végén mérföldkő és jól definiált termékek:



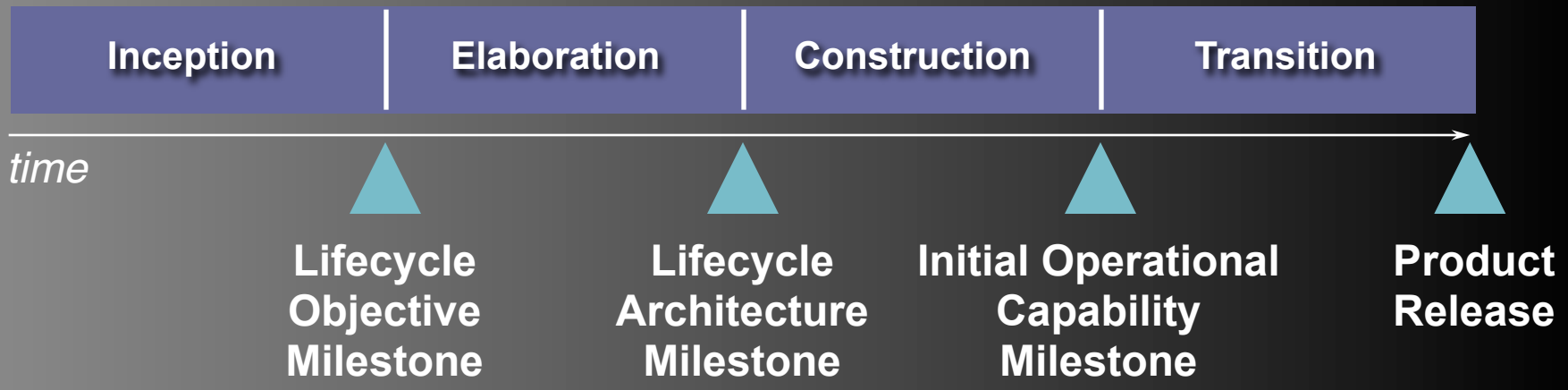


A folyamat fázisai

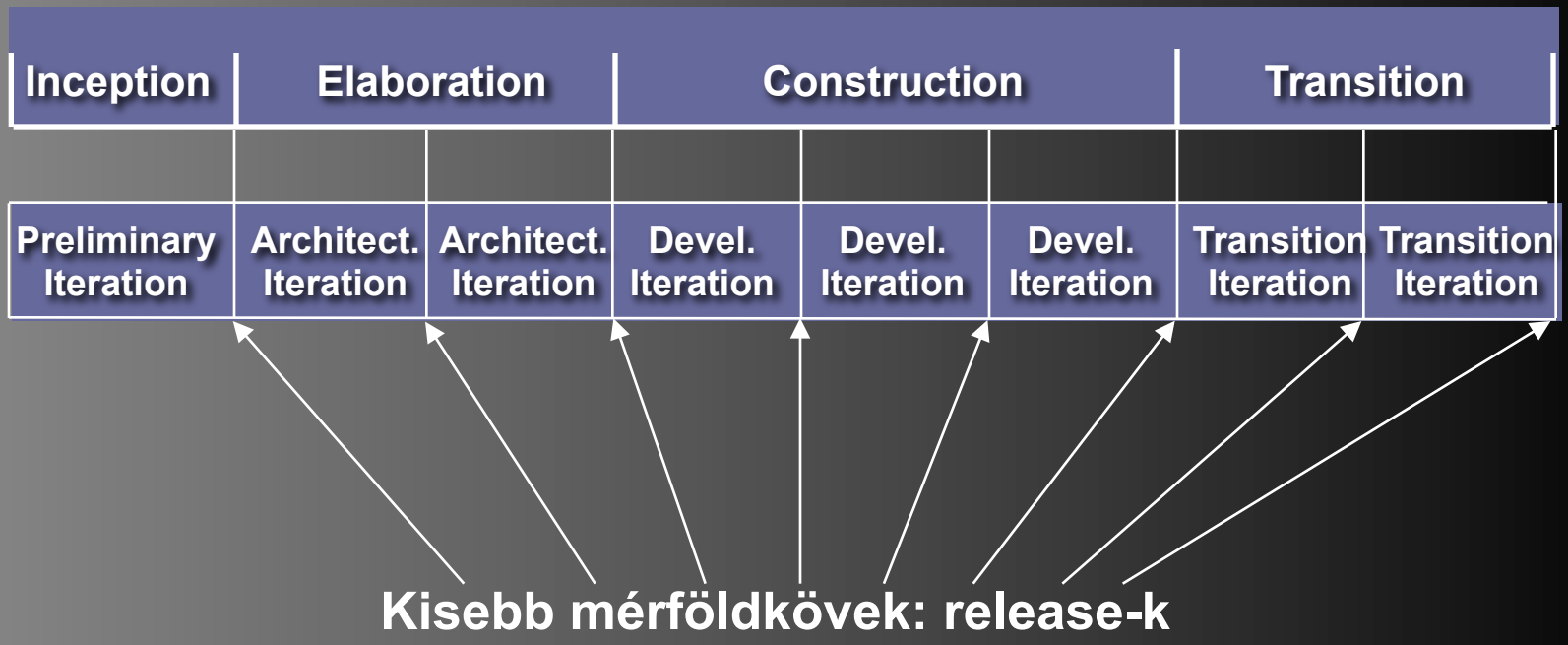
- Kezdeti fázis (Inception Phase)
- Kidolgozási fázis (Elaboration Phase)
- Megvalósítási fázis (Construction Phase)
- Átadási fázis (Transition Phase)



A folyamat fontosabb mérföldkövei



Iterációk és fázisok kapcsolata



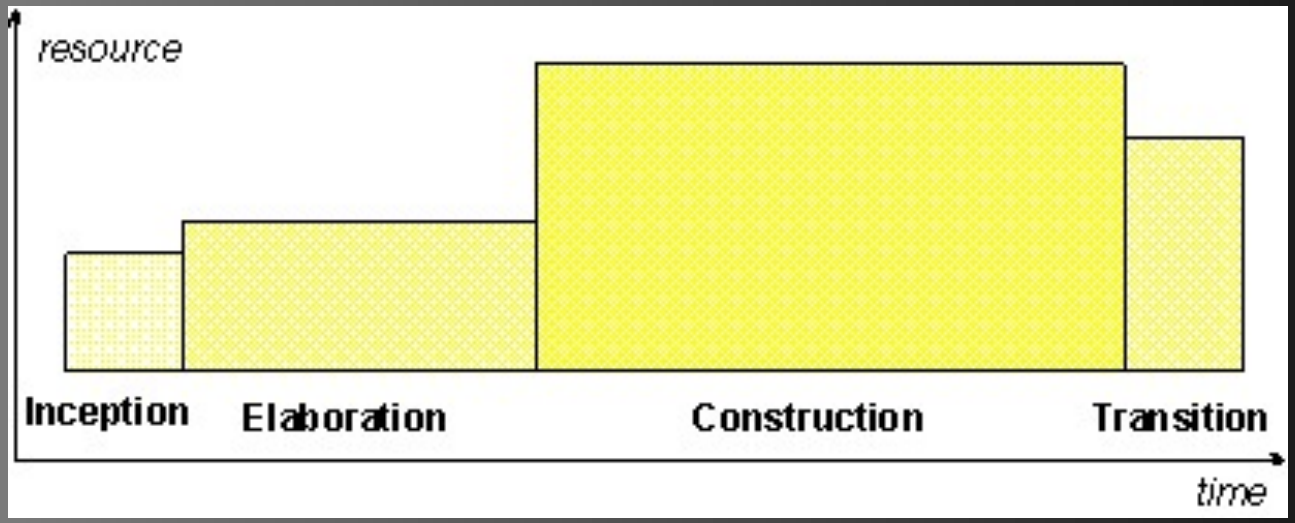
Az iteráció egy jól körülhatárolt tevékenység sor, mely egy megalapozott terven és kiértékelhető kritériumon alapul, mely egy futtatható release-t (belső/külső) hoz létre.

A Rational Unified Process (RUP)

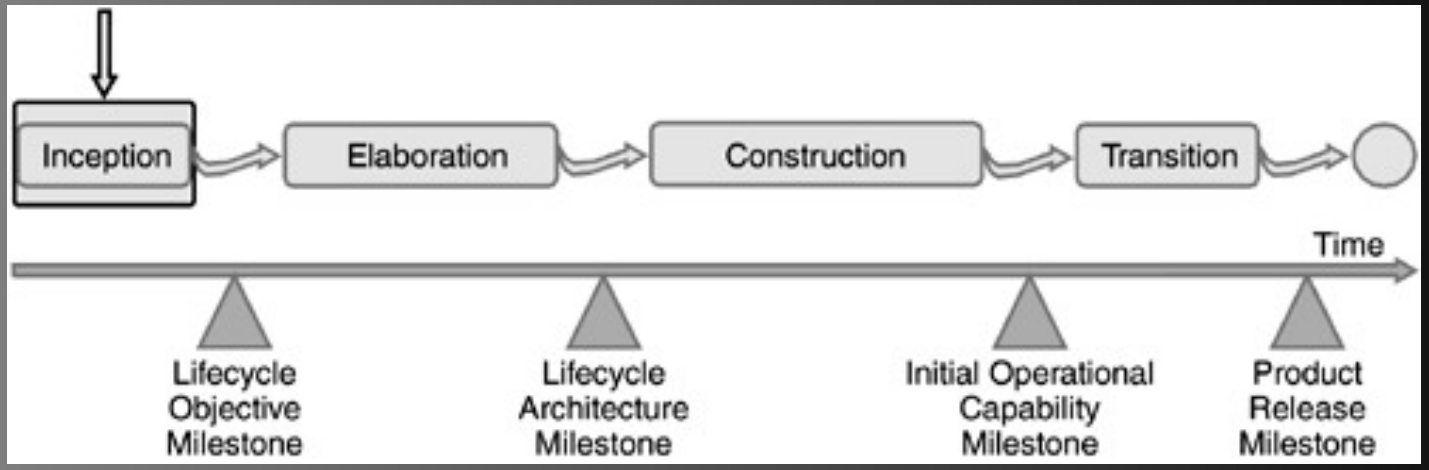
- RUP elveinek bemutatása
- RUP elemeinek bemutatása
- RUP fázisok bemutatása
- RUP, mint termék bemutatása

Az egyes fázisok ráfordítási szükséglete

	Inception	Elaboration	Construction	Transition
Munka	~5%	20 %	65 %	10%
Idő	10 %	30 %	50 %	10%



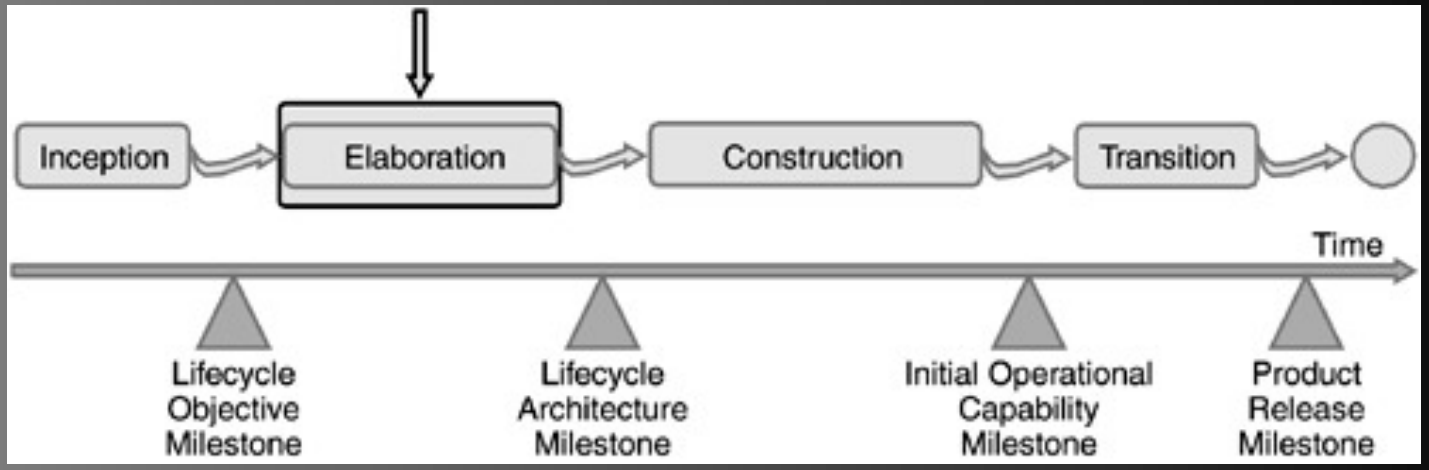
Kezdeti fázis helye a



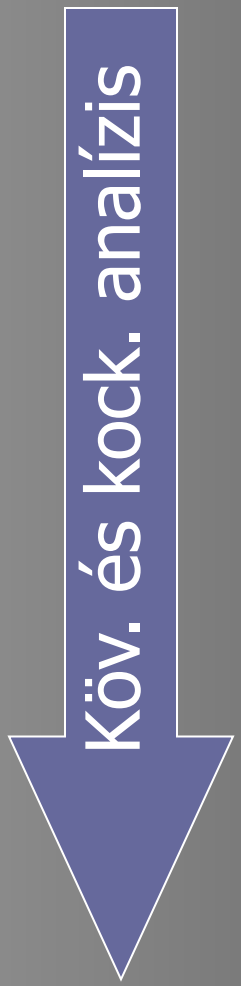
Kezdeti fázis főbb tevékenységei

- Projekt vízió elkészítése:
  - Probléma megfogalmazása
  - Kulcs felhasználók, stakeholder-ek analízise
  - Magasszintű funkcionális, nem funkcionális követelmények megfogalmazása
- „mile-wide, inch-deep” felmérése:
  - Aktorok azonosítása
  - Magasszintű use-case-ek elkészítése
- Fejlesztési folyamat és eszközök meghatározása
- Üzleti eset (business case) elkészítése: költség, ütemezés és kockázat becslés

Kidolgozási fázis helye a

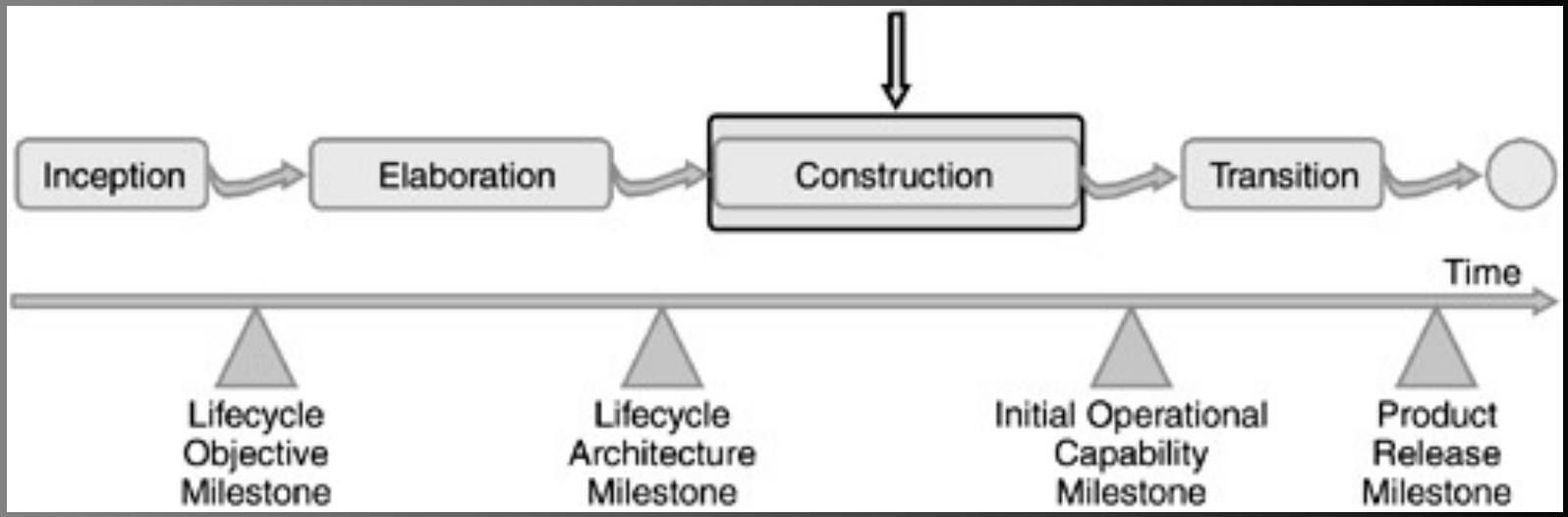


Kidolgozási fázis főbb tevékenységei



- Követelmények részletes felmérése, kidolgozása (lényegiek)
- Architektúra megtervezése, implementálása és validációja:
  - Hw és Sw architektúra
  - Építőkockák (building blocks) és IF-ek
  - Technológiai döntések
- Kockázatok csökkentése: ütemezés, költség és terjedelem finomítása
- Fejlesztési folyamat finomítása

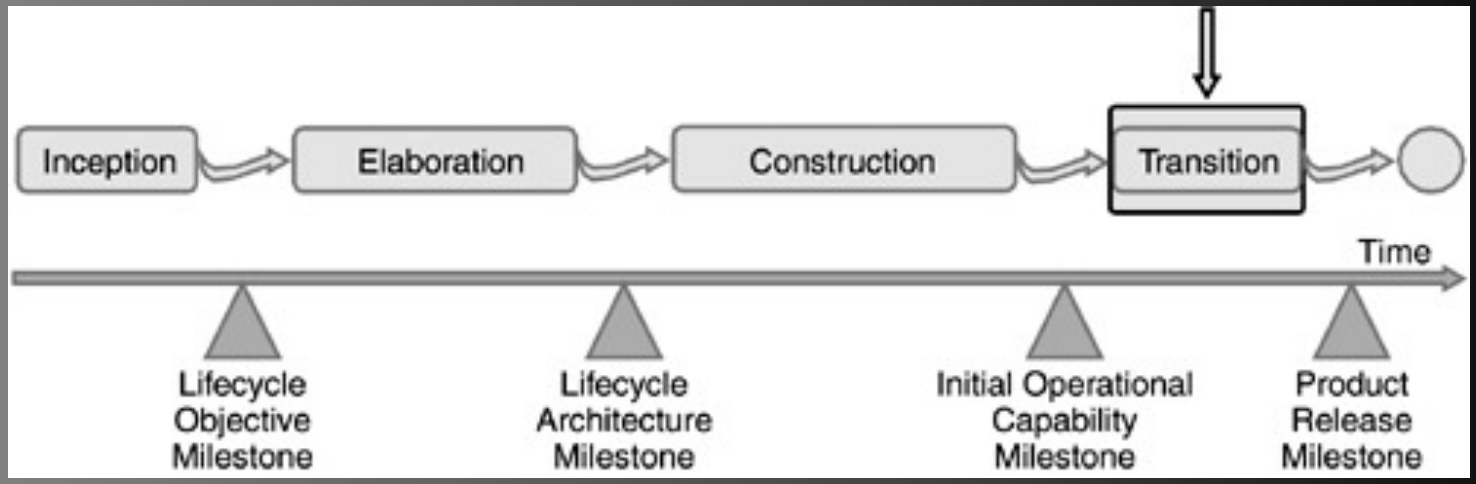
Megvalósítási fázis helye a



Megvalósítási fázis főbb tevékenységei

- Jól megalapozott architektúra ->fejlesztési tevékenység párhuzamosítása
- Kimaradt követelmények részletes felmérése, kidolgozása
- Implementáció és unit-teszt
- Folyamatos integráció és regressziós tesztelés
- Megrendelő intenzív bevonása: alfa és béta tesztelés

Átadási fázis helye a

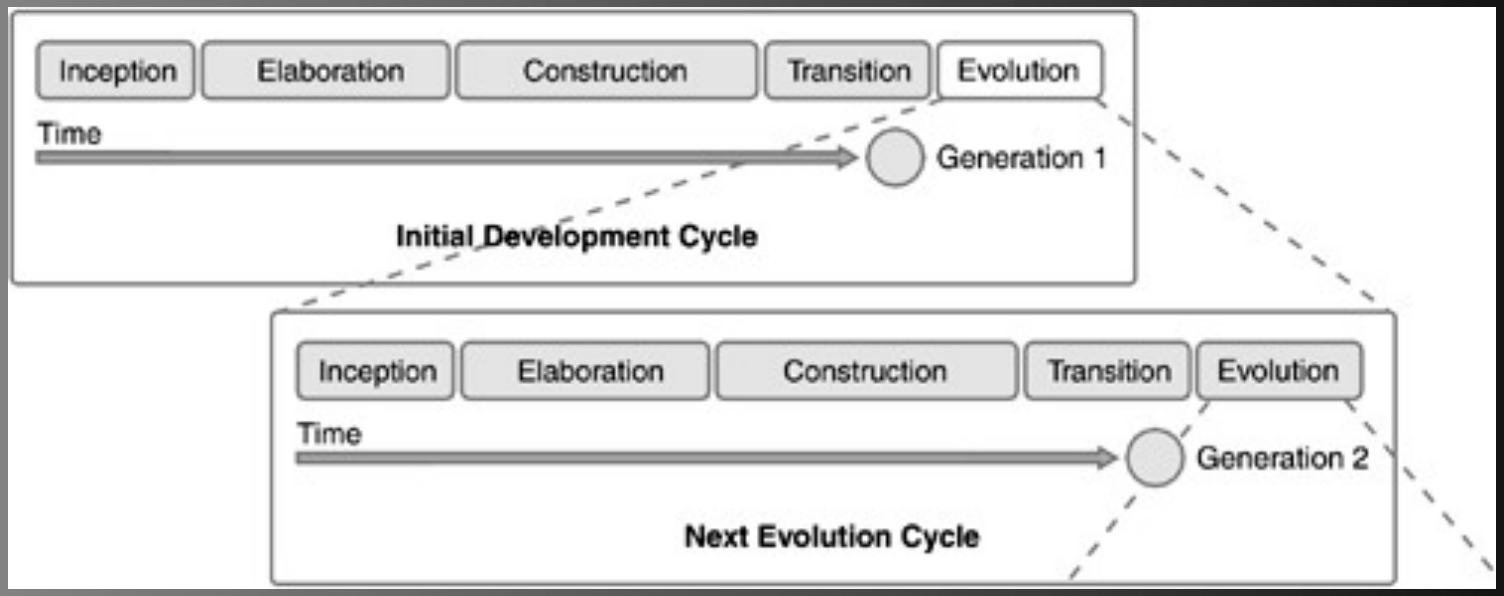


Átadási fázis főbb tevékenységei

- Rendszer finomhangolása
- Felhasználók oktatása, oktatási anyagok
- Telepítés a megrendelő telephelyére, konfigurálás
- Termék elfogadási teszt (kritériumok definiálása a teszt előtt!)
- Tanulságok levonása

Mi történik átadás után?

- Fejlesztési ciklus – Evolúciós ciklusok emberi tevékenység - termék (gyakori az átlapolódás)
- Patch-ek készítése



A fázisok gyakori

- Nem tévesztendő össze a vízesésmodell hagyományos fázisaival!
- Figyelembe kell venni mindkét dimenziót („tevékenység púpok”)!
- Nincs rögzített munkafolyamat!
- Nincsenek rögzített termékek!

A Rational Unified Process (RUP)

- RUP elveinek bemutatása
- RUP elemeinek bemutatása
- RUP fázisok bemutatása
- RUP, mint termék bemutatása

A Rational Unified Process (RUP)

- RUP elveinek bemutatása
- RUP elemeinek bemutatása
- RUP fázisok bemutatása
- RUP, mint termék bemutatása

Live show!

eXtreme programming (XP)

Közvélemény kutatás

Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?

Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?

Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?

Ki szeret az első követelmény meghallását követően rögtön kódolni?

Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)

Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás

Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?

Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?

Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?

Ki szeret az első követelmény meghallását követően rögtön kódolni?

Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)

Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás

Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?

Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?

Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?

Ki szeret az első követelmény meghallását követően rögtön kódolni?

Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)

Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás

- ☹ Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?
- ☹ Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?
- ☹ Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?
- Ki szeret az első követelmény meghallását követően rögtön kódolni?
- Ki szereti a Java-t, vagy a C++-t?
  (megengedő vagy! ☺)
- Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás

Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?

Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?

Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?

Ki szeret az első követelmény meghallását követően rögtön kódolni?

Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)

Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás

Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?

Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?

Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?

Ki szeret az első követelmény meghallását követően rögtön kódolni?

Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)

Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás


Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?


Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?


Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?


Ki szeret az első követelmény meghallását követően rögtön kódolni?


Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)


Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás

- ☹ Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?
- ☹ Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?
- ☹ Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?
- ☺ Ki szeret az első követelmény meghallását követően rögtön kódolni?
- ☺ Ki szereti a Java-t, vagy a C++-t? (megengedő vagy! ☺)
- ☺ Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Szép lenne az élet mi?

Közvélemény kutatás


Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?


Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?


Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?


Ki szeret az első követelmény meghallását követően rögtön kódolni?


Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)


Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

Közvélemény kutatás


Ki szeret napokat eltölteni azzal, hogy követelmény specifikáció és rendszer terv megbeszéléseken részt venni?


Ki szeret hosszú és részletes specifikációkat és terv dokumentumokat írni?


Ki szeret hosszú és részletesen megtervezett projekt státusz megbeszéléseken ülni?


Ki szeret az első követelmény meghallását követően rögtön kódolni?


Ki szereti a Java-t, vagy a C++-t?
(megengedő vagy! ☺)

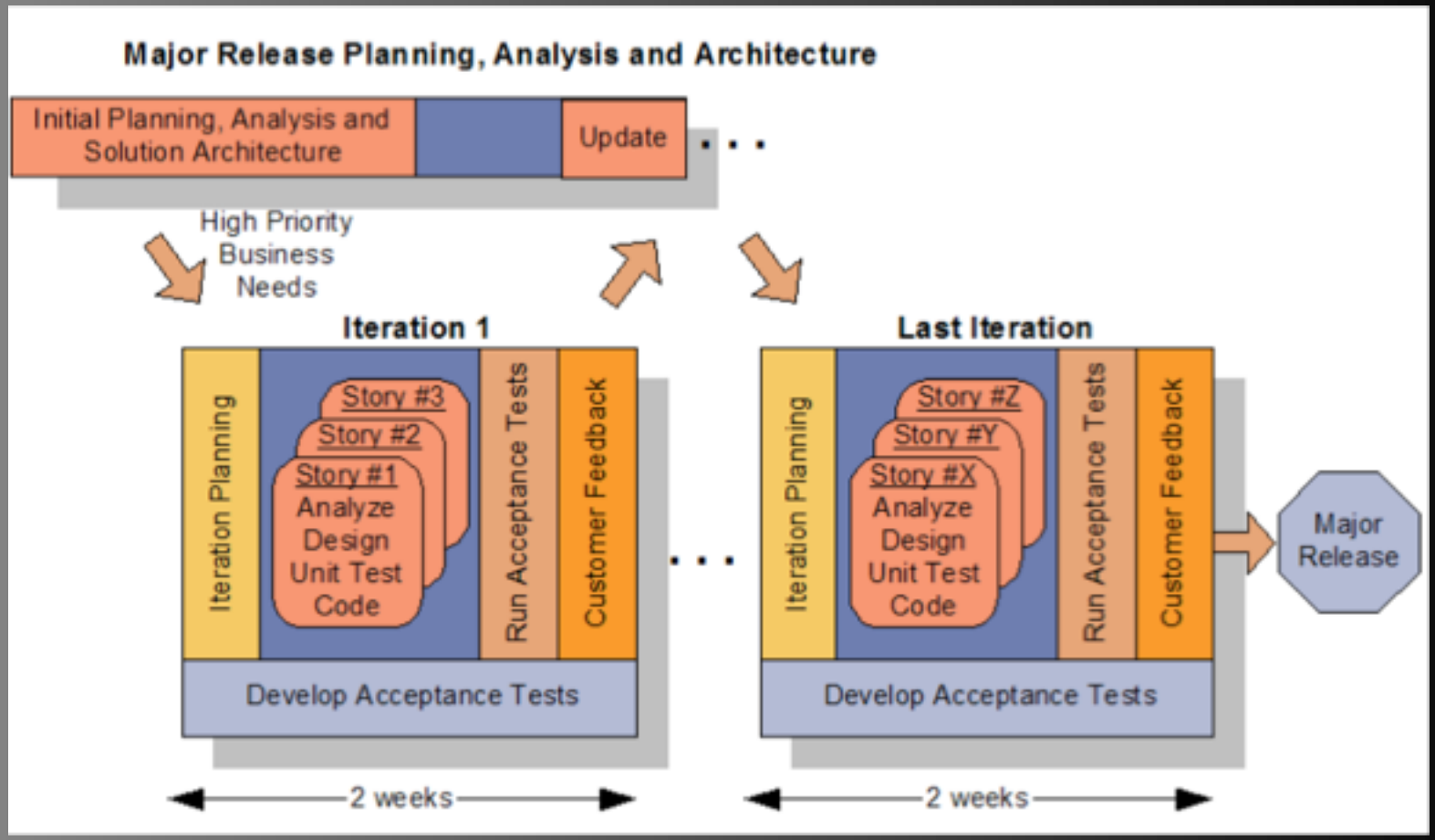

Ki szeretne heti 40 órát programozni a munkahelyén, a többi 128-at pedig hobby programozással tölteni? ☺

eXtreme programming!

Tipikus XP projekt

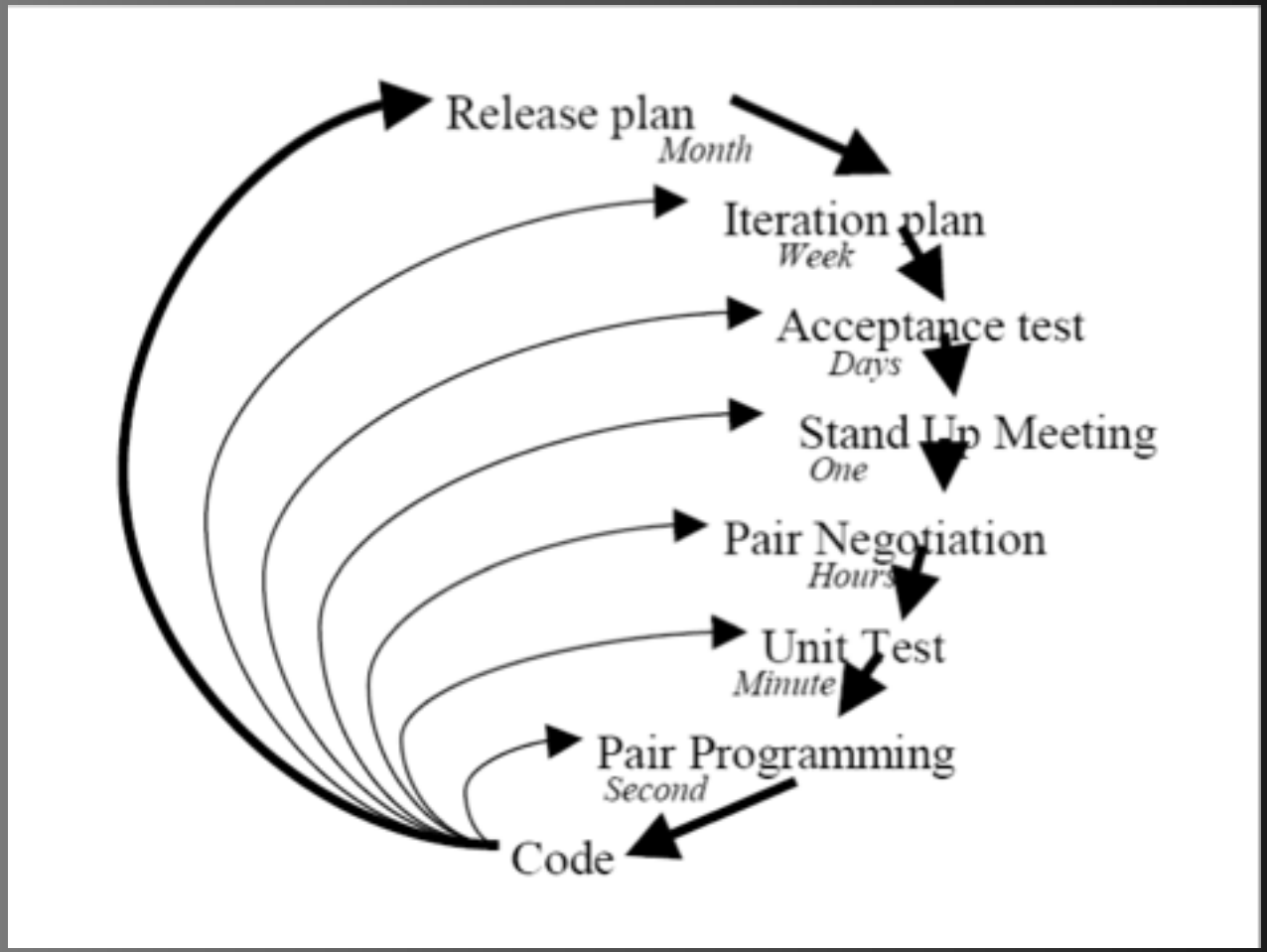
- Rövid iterációk (2-3 hét) sorozata
- Minden iterációban üzleti értékek kerülnek átadásra
- Működő kód centrikus (állandó tesztelés)
- Szoros megrendelői együttműködés („közös” fejlesztés a megrendelővel)
- Naponta „stand up” megbeszélések (státusz nyomon követés)
- Kis létszámú programozókból álló „tiger team”
- Kis méretű szoftver

Az XP projekt folyamata



Forrás: Linda Farrenkopf - Software Methodologies and Impacts to Quality, 2004

Tervezési és visszajelzési hurkok az XP-ben



Forrás: Per Runeson and Peter Greberg - Extreme Programming and Rational Unified Process, 2004

XP 12 szabálya

- Tervezési folyamat (planning game):
  - Követelményeket storycard-okra írd! (cards on a wall)
  - Teszt eseteket már a követelményeknél definiáld!
  - Felhasználó válasszon az általad írandó funkciók közül a költség becslés alapján!
- Kis „release”-ek, rövid ciklusok
  - Mielőbbi futtasd a programot!
  - Mielőbbi szerezz visszajelzést a felhasználótól!
- Metafora használata
  - Közösen használt nevek és leírásokat használj!
- Egyszerű tervezés
  - UML-t használhatod, de egyáltalán nem szükséges! (elegendő a code model)

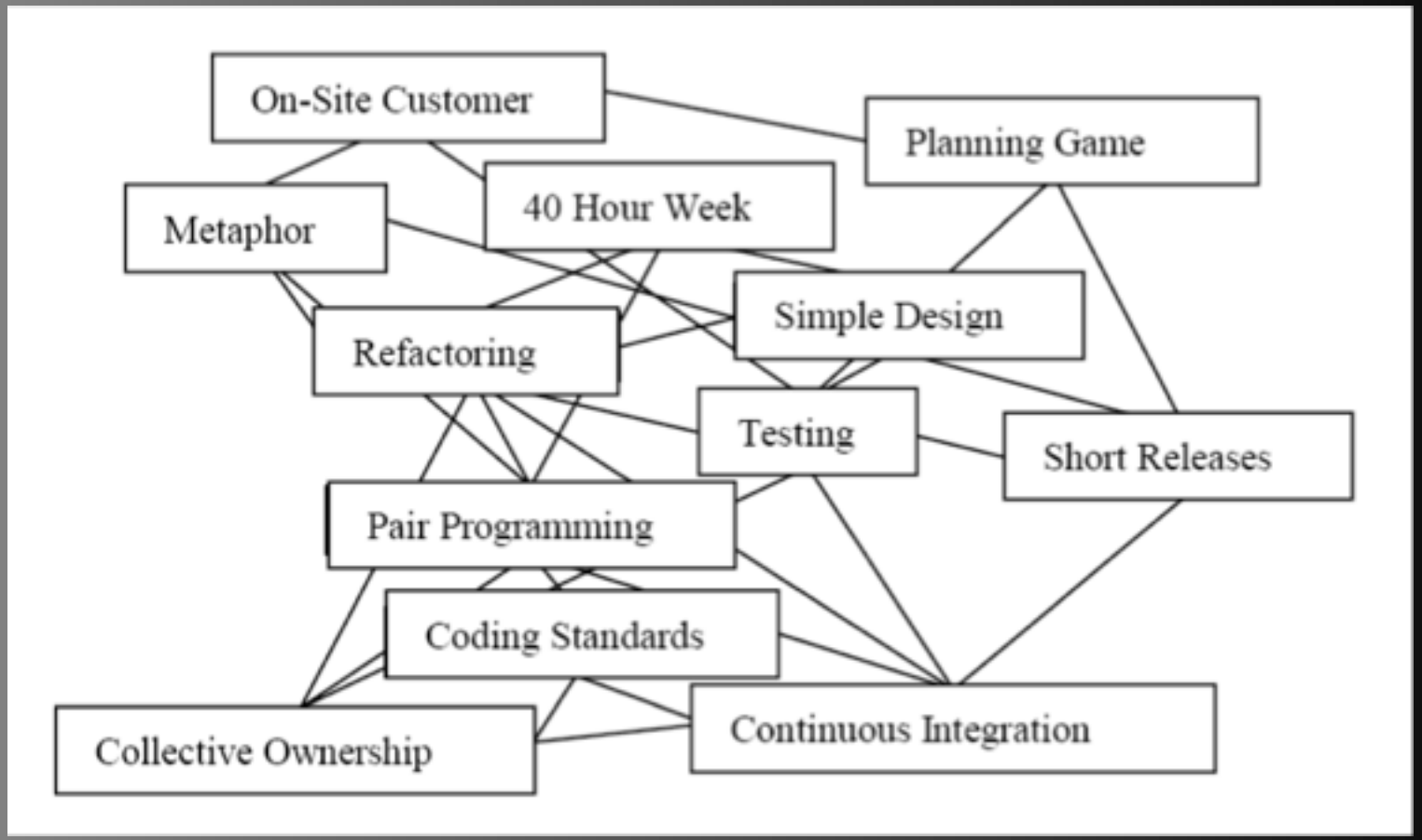
XP 12 szabálya (folyt.)

- Tesztelés
  - Tesztelj állandóan! (verifikáció)
  - Ellenőrizesd a felhasználóval! (User acceptance test)
- Refactoring
  - Írd újra egyszerűbben!
- Pair programming
  - Minden program funkciót ketten írjatok egy számítógép előtt ülve! (állandó code review)
- Kollektív tulajdonú kód
  - Minden kódot közös felelősséggel kezeld és vele közös tulajdonként bánj!

XP 12 szabálya (folyt.)

- Állandó integráció
  - Minden nap legalább egy kód integrációt hajtsatok végre, amely működik is ☺!
- 40 óra egy hét
  - Ne dolgozz többet egy héten 40 óránál, különben a minőség rovására megy!
- On-site customer
  - A felhasználóval szorosan dolgozz együtt!
- Kódolási standardok
  - Tiszta, érthető, közös standardot használjatok a kód megírása során!

XP szabályok összefüggései



Forrás: Per Runeson and Peter Greberg - Extreme Programming and Rational Unified Process, 2004

XP összegzése

- Nagyon produktív
- Projekt kockázatokat hatékonyan kezeli
- Helyesen futó kódra helyezi a hangsúlyt
- Vevői megelégedést kiemelten kezeli

- 12 fő alatt alkalmas
- Maximum fél éves kifutású projektek
- Környezeti megkötés (közös légtér)
- Kultúrális megkötés (kooperáló közösség)
- Nem specifikál közbenső termékeket

Agilis metodikák összehasonlítása

A RUP, mint agilis szoftverfejlesztési elv

Az agilitás...

- annak a képessége, hogy gyorsabban cselekszünk, mint azok a dolgok, melyek negatívan befolyásolhatják a projektet.

- annak a képessége, hogy a releváns változásokkal:
  - megrendelői igényekkel
  - üzleti célokkal
  - technológiai szükségeletekkel

  képes lépést tartani.

Agilis metodikák

- (Rational) Unified Process – „a 3 amigo”: Bootch, Rumbaugh, Jacobson
- eXtreme programming (XP) – Kent Beck
- SCRUM - Ken Schwaber, Mike Beedle
- Dynamic Systems Development Method (DSDM)
- Feature Driven Development (FDD) - Peter Coad
- Lean Software Development
- Adaptive Software Development - Jim Highsmith
- Crystal Methodologies - Alistair Cockburn

A vízeséses, a RUP és az XP metodika összehasonlítása

	Vízeséses	RUP	XP
Prj mgt	Szekvenciális	Iteratív	Iteratív
Team formáció	Nagy, erősen struktúrált	Közepes méretű, tech. specialisták	Programozók párban és prjmgr
Követelmény mgt	Szigorúan definiált és kezelt	Szigorúan definiált és kezelt	Programozás közben (stry crd)
Elemzés és tervezés	Széleskörűen és szigorú exit crit.	Adott fázis részletes OO terve	Követelmény elemzéssel együtt
Tesztelés	Modul tesztre koncentrál	Tesztelési tervek szerint (U,I,UAT)	Automatizált unit tesztre épít és napi

Software Process Engineering Metamodel

- Folyamat mintáktól a SPEM-ig
- RUP és XP modellek

Software Process Engineering Metamodel

- Folyamat mintáktól a SPEM-ig
- RUP és XP modellek

Út a folyamat mintákig

- Számos különböző szoftverfejlesztési folyamat jött létre. Mindegyik „best practice”-eket foglal magába néhány új ötlettel kiegészítve.
- A Design pattern-ök bebizonyították az előnyüket a szoftver tervezése során, ezért ez az ötlet lett alkalmazva a szoftverfejlesztési folyamatok szintjén is.

Út a folyamat mintákig

- Számos különböző szoftverfejlesztési folyamat jött létre. Mindegyik „best practice”-eket foglal magába néhány új ötlettel kiegészítve.
- A Design pattern-ök bebizonyították az előnyüket a szoftver tervezése során, ezért ez az ötlet lett alkalmazva a szoftverfejlesztési folyamatok szintjén is.

Út a folyamat mintákig

- Számos különböző szoftverfejlesztési folyamat jött létre. Mindegyik „best practice”-eket foglal magába néhány új ötlettel kiegészítve.
- A Design pattern-ök bebizonyították az előnyüket a szoftver tervezése során, ezért ez az ötlet lett alkalmazva a szoftverfejlesztési folyamatok szintjén is.

Folyamat minták:
A folyamat minták kipróbált és sikeresnek talált folyamatlépések sorozatát foglalja magába.

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

SPEM – Software Process Engineering Meta-model

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

SPEM lehetővé teszi a formális definíciót!

SPEM – Software Process Engineering Meta-model

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

Folyamat minták Gamma-szerű leírása lehetségesé válik!

SPEM lehetővé teszi a formális definíciót!

SPEM – Software Process Engineering Meta-model

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

Folyamat minták Gamma-szerű leírása lehetségesé válik!

SPEM lehetővé teszi a formális definíciót!

SPEM – Software Process Engineering Meta-model

Észrevételek a folyamat mintákra

- A szöveges folyamat minta leírások alkalmazása nehézkes
- A minták testreszabása nagy munka
- Nincs formális leírásuk

Folyamat minták Gamma-szerű leírása lehetségesé válik!

SPEM lehetővé teszi a formális definíciót!

SPEM – Software Process Engineering Meta-model

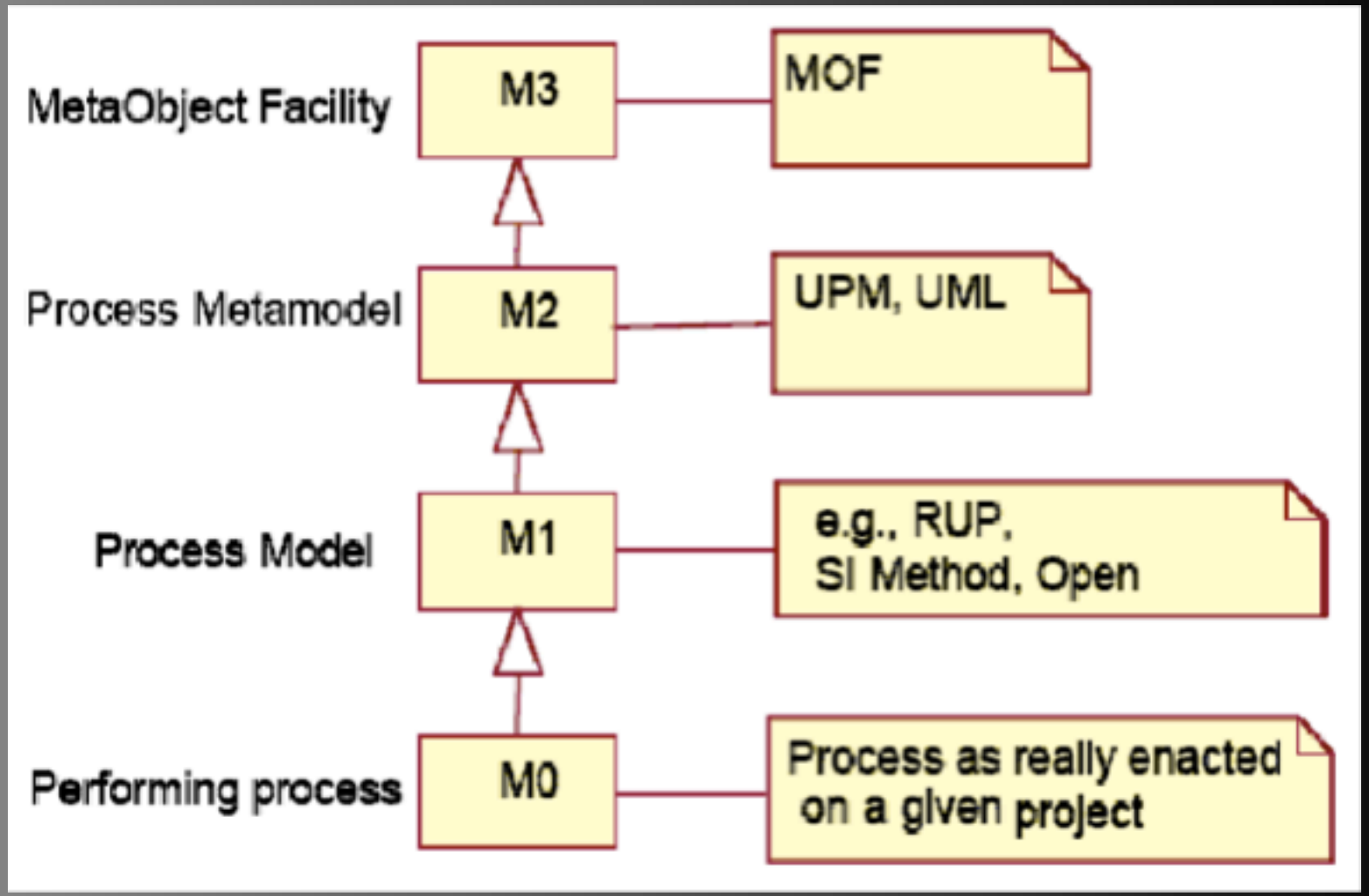
Eszköztámogatás is megoldható!

Software Process Engineering Metamodel

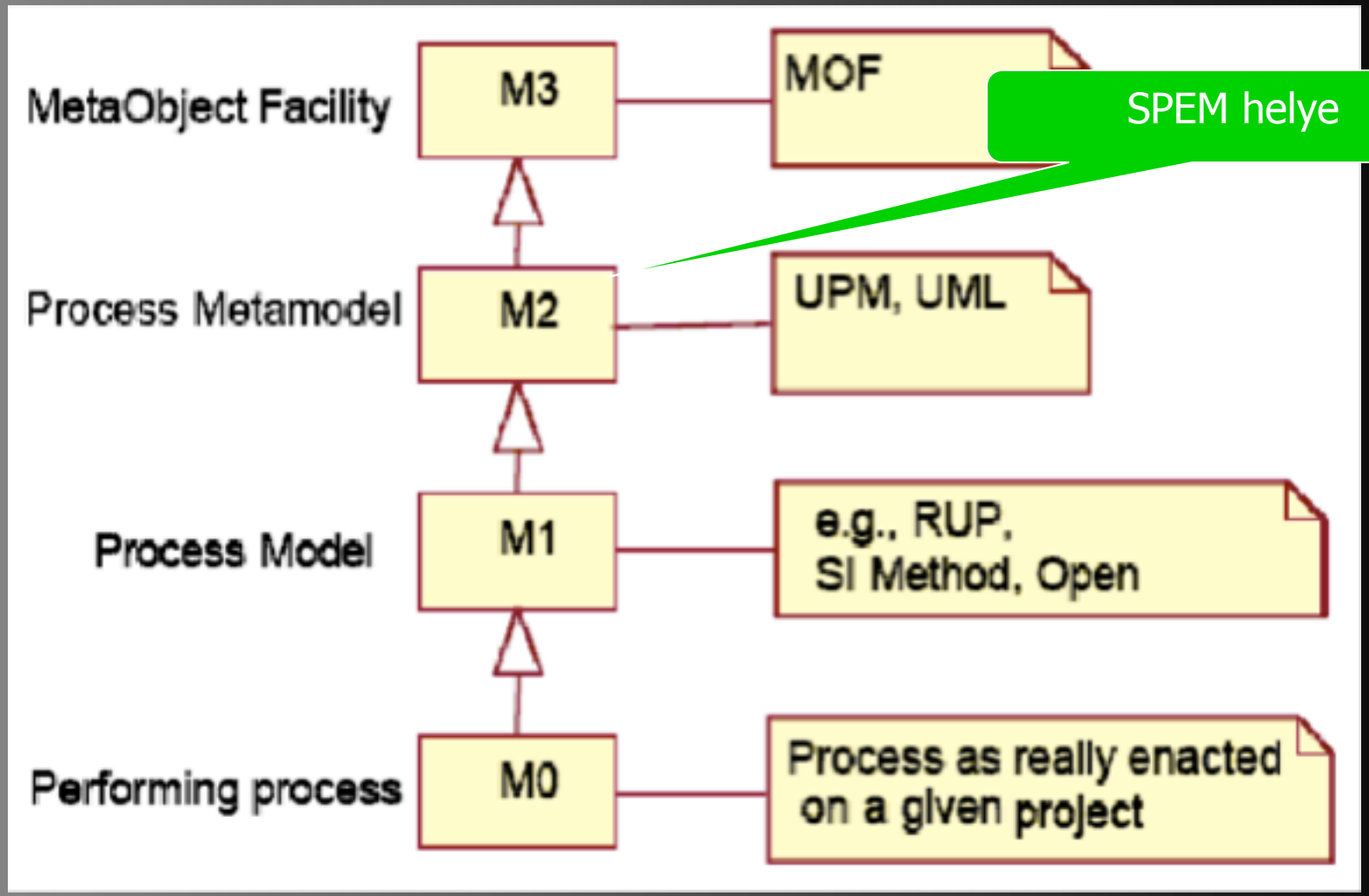
- SPEM:
  - egy UML profile, amely szoftver process modellezésre lett kifejlesztve
  - egy OMG specifikáció

- **Folyamat struktúra**:
  - WorkProduct, WorkDefinition, Activity, Step
- **Folyamat komponens**:
  - Process, Discipline
- **Folyamat életciklus**:
  - Phase, Iteration

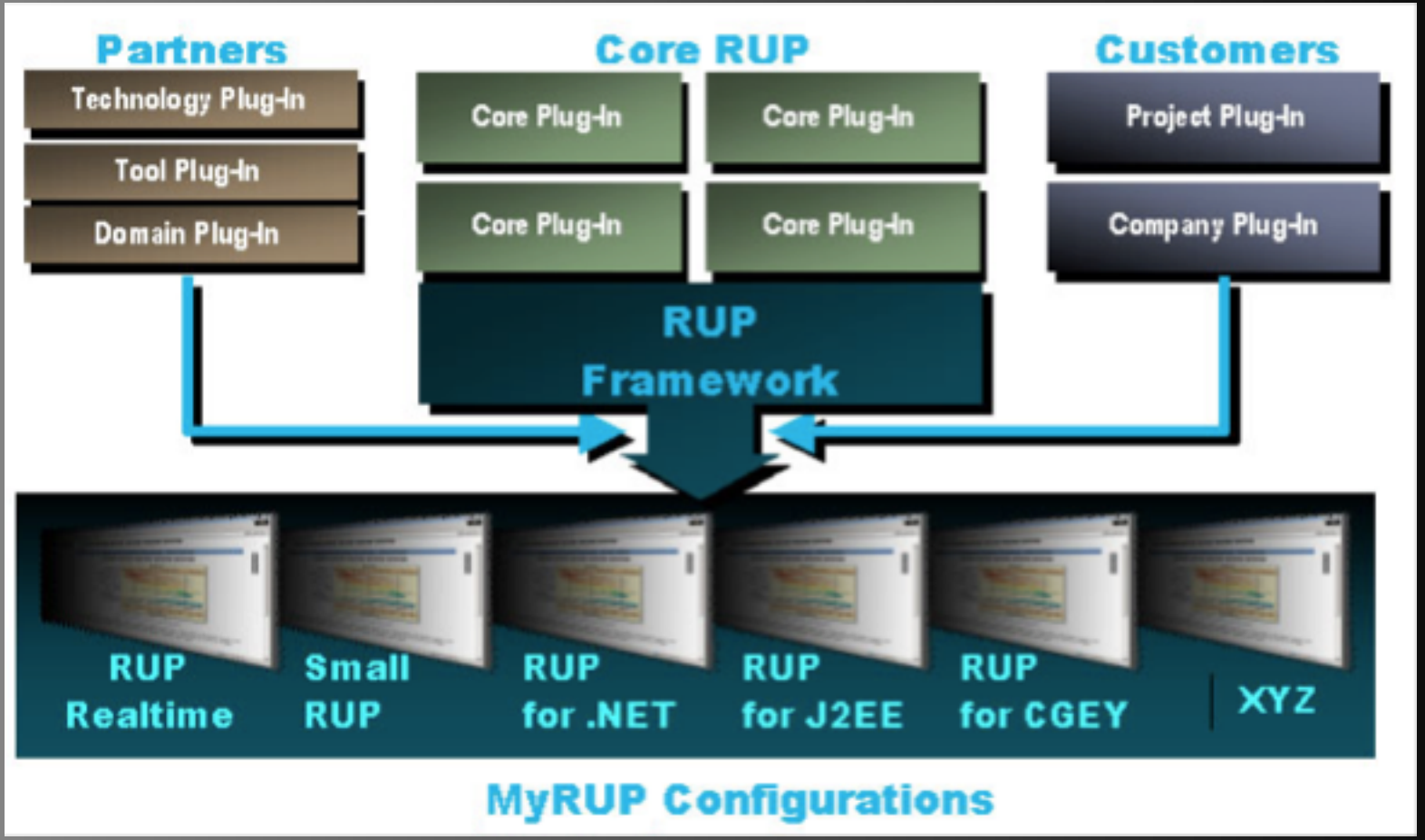
SPEM helye a modellek között



SPEM helye a modellek között



RUP konfigurációk



Software Process Engineering Metamodel

- Folyamat mintáktól a SPEM-ig
- RUP és XP modellek

Software Process Engineering Metamodel

- Folyamat mintáktól a SPEM-ig
- RUP és XP modellek

Live show!

RUP referenciák

- I. Jacobson, G. Booch, J. Rumbaugh,
  The Unified Software Development Process, Addison Wesley 1999
- P. Kroll, P. Kruchten,
  Rational Unified Process Made Easy, Addison Wesley 2003
- P. Robillard, P. Kruchten,
  Software Engineering Process with the UPEDU, Addison Wesley 2003
- M. Paulk, B. Curtis, M. Chrissis, C. Weber,
  Capability Maturity Model for Software, Tech. Report 1993 CMU SEI
- Website:
  www.ibm.com

XP referenciák

- Kent Beck. *Extreme Programming Explained: Embrace Change.*
- Jeffries, Ron, Ann Anderson, and Chet Hendrickson. *Extreme Programming Installed.*
- Beck, Kent and Martin Fowler. *Planning Extreme Programming.*
- Craig Larman: *Agile and Iterative Development: Manager's Guide*
- Websites:
  www.xprogramming.com
  www.extremeprogramming.org
  www.agilealliance.com