

Információs rendszerek tervezése – hallgatói prezentáció

Agilis szoftverfejlesztés és Scrum

Miskolc, 2008.10.15

Készítette:

Sereg Ákos
Varga Balázs

Tartalom

- Projektmenedzsment alapvető ismertetése
- Klasszikus modellek ismételése, hátrányai
- Agilis projektvezetés
- Scrum
- Esettanulmány

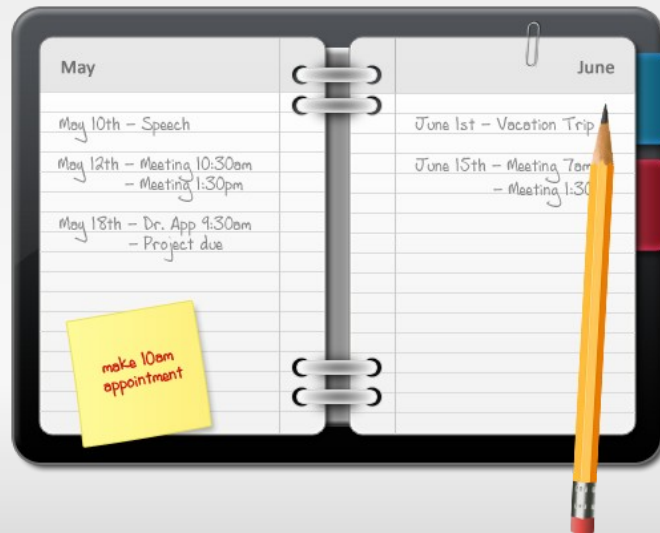


Projektmenedzsment

- Korlátok / tényezők felmérése
(projektenként változó kritériumok, különböző súlyokkal) - tradicionálisan:
 - Idő (time)
 - Pénz (cost)
 - Hatókör (scope)
 - Erőforrás (resource)
 - Teljesítmény
 - Minőség

Projektmenedzsment

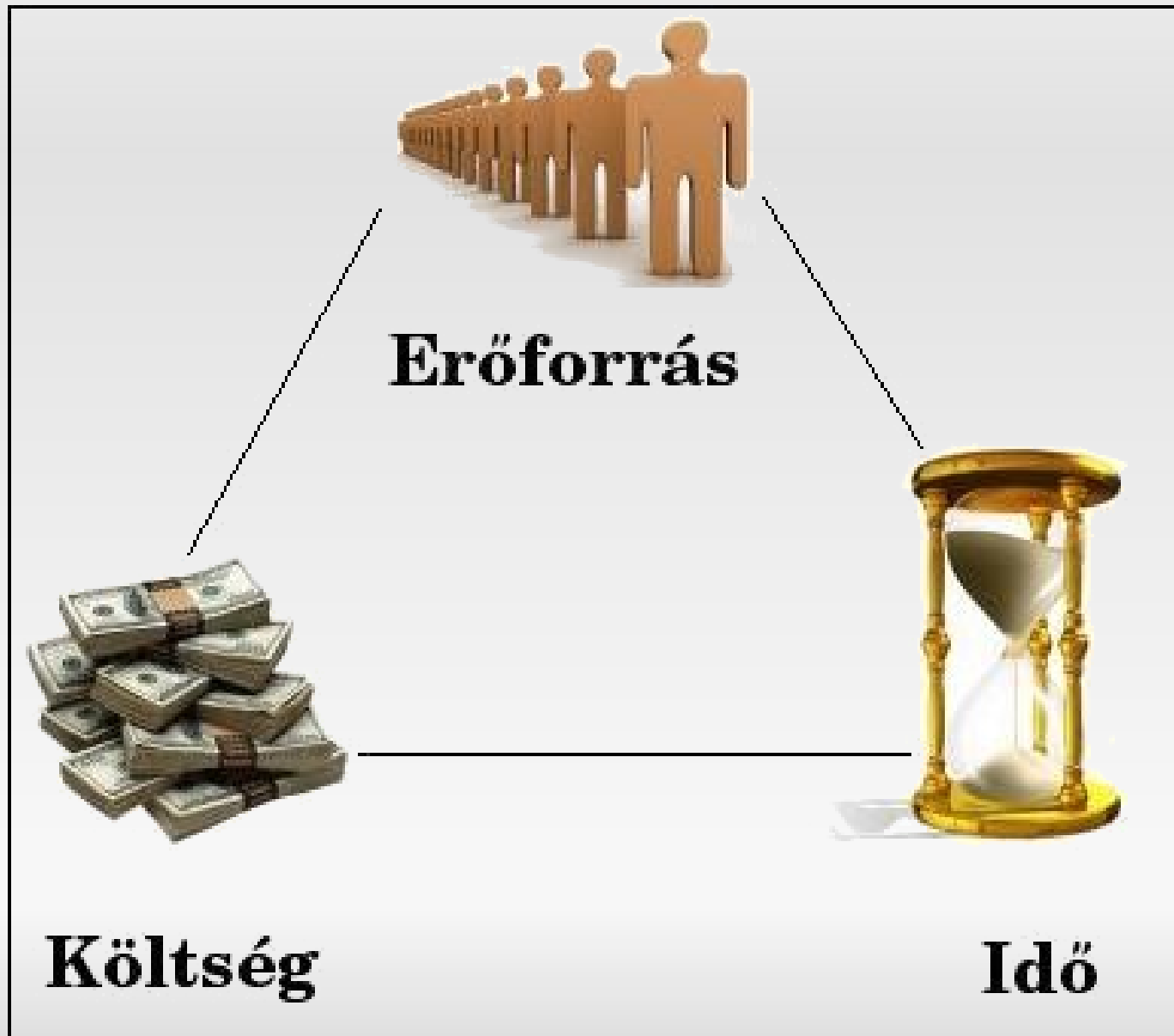
- Eredményességi mérce
(Költségek, bevételek, tartalmak, időbeli ütemezés betartása)
- Célja: Feladatok, erőforrások, határidők szervezése / összehangolása



Projektmenedzsment - korlátok

- Pénz, Idő: Egyértelmű
 - Hatókör:
 - Megvalósítandó funkcionalitás
 - Amennyiben bővül, változik a költség/határidő
 - Projekttől függően egyéb korlátok, amiket figyelembe kell venni
-
- Adott Idő- és Költségkereten belül sikeresen teljesüljenek a projekt céljai

Projektmenedzsment háromszög



Projektmenedzsment háromszög

- Irányítás: A P.M. Háromszög egyensúlyban tartása
- Ha bármelyik sarok irányába billen, a másik kettő hangsúlyozásával visszaállítható
- Példa
Határidő veszélybe kerül
 - Erőforrások átcsoportosításával /
 - Pótlólagos erőforrások bevonásával
- ... az egyensúly helyre billenthető!

Projektmenedzsment az informatikában

- Egyre összetettebb, jobb minőségű sw.-ek igénye a piacon
- Több időt igényel a fejlesztési tevékenység koordinálása
- Versenyhelyzet: állandó nyomás a fejlesztőcégeken
- → hatékony módszer szükséges a fejlesztés menedzsmentjéhez

Projektmenedzsment az informatikában

- Klasszikus sw fejlesztési modellek nem optimális hatékonyságúak (később részletezve)
- Versenyképesség megtartása + piaci igények kielégítése
- → Szükségessé vált újabb, hatékonyabb sw fejlesztési modellre

Projektmenedzsment az informatikában

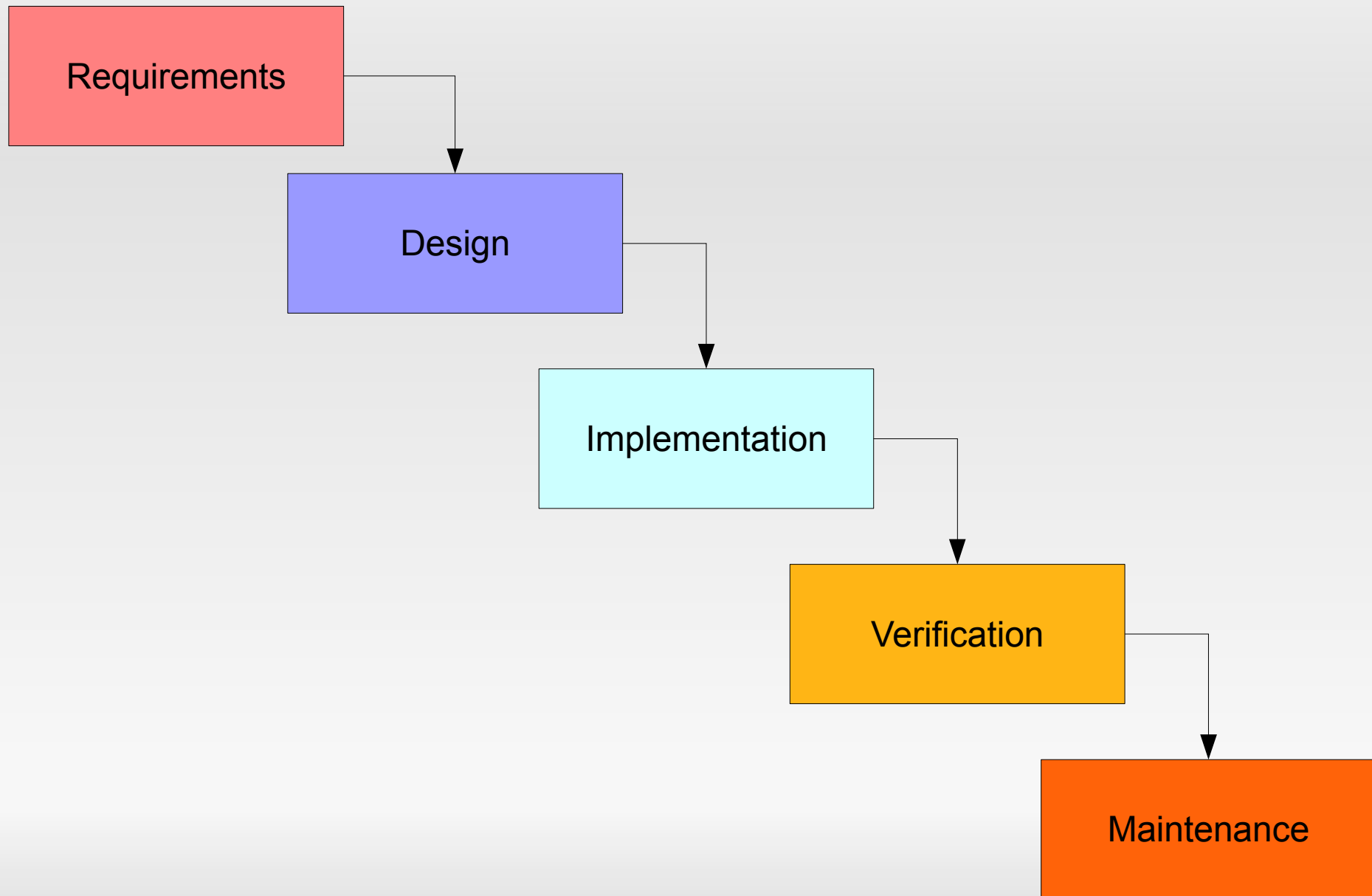
- Gyakori problémák
 - A megrendelő az esetek többségében nem tudja pontosan, mit akar
 - Igények megváltozása a fejlesztés során
 - Fejlesztési modellnek rugalmasnak kell lennie!



Tradicionális modellek áttekintése

- Vízesés modell
- Inkrementális (iteratív) modell
- Spirál modell
- „Cleanroom” modell
- RAD modell
- RUP modell

Vízesés modell



Vízesés modell

- Jellemzők
 - Egymás után következő elhatárolt, de összefüggő fázisok (általában 5)
 - Követelményanalízis és definíció (requirements)
 - Rendszer- és szoftvertervezés (design)
 - Implementáció, részegységek tesztelése (impl.)
 - Részegységek tesztelése, rendszer tesztelés (verification)
 - Működtetés, karbantartás (maintenance)
- Hátrány
 - Már a korai szakaszokban komoly döntéseket kell hozni (rugalmatlan)

Inkrementális (iteratív) modell

- Jellemzők
 - Célszerűbb a nagy szoftverek fejlesztésénél
 - Rugalmasabb a vízesés modellnél
 - Első lépésben egy-egy kisebb probléma megoldása a cél
 - AI-változatok egymásutánija
 - Prototípus változat: UI (félreértések elkerülése)
 - Gyors visszacsatolás
- Hátrány
 - A folyamatos változások odavezetnek, hogy a rendszer rosszul strukturált lesz
 - Fejlődés mérése nehézkes

Spirál modell



Spirál modell

- Jellemzők
 - 4 lépésen keresztül, ciklikusan történik a fejlesztés
 - Újdonság: több alternatíva felajánlása (minimális kockázatú a megfelelő)
 - Minden ciklus egy célkitűzéssel kezdődik
- Hátrány
 - Alacsony kockázatú vagy kicsi projektnél költséges
 - Jelentős kockázatkezelési szakértelem szükséges

Cleanroom módszer

- Jellemzők
 - Statisztikai minőségbiztosítás (MTTF)
 - Létezik matematikai modell a megadására
 - MTTF -ben megadott megbízh. egy időben fejeszthető a termékkel (hiba korai kiszűrése)
 - Minden szakasz után bizonyítják, hogy hibátlan a szoftver
- Hátrány
 - Speciáli szakértők
 - Nem teszi hatékonyabbá a fejlesztést

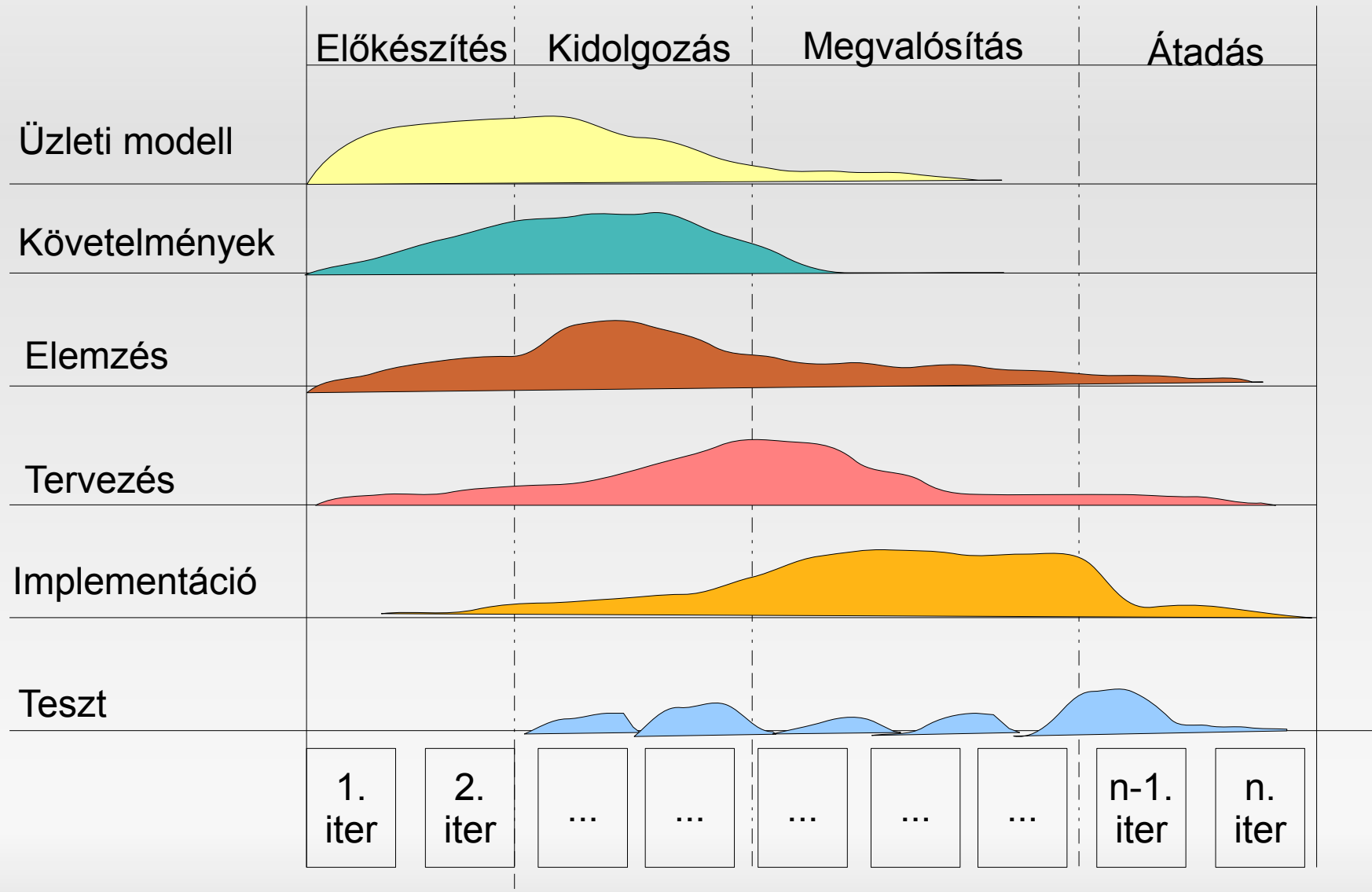
RAD – Gyors alkalmazásfejlesztés

- Jellemző
 - Rapid Application Development
 - Ciklikus fejlesztés
 - Működő prototípusok létrehozása
 - Integrált fejlesztői környezetek használata
 - SW komponensek újra felhasználása
 - Lecsökken a fejlesztési idő
- Hátrány
 - RAD módszertana gátat szabhat a használhatóság és futási sebesség területén

RUP modell

- Jellemző
 - Nem egy kész modell, inkább csak keret (ajánláscsomag)
 - Szervezetre / projektre igazítható
 - Nagy projektekre van kitalálva, ahol több csapat is dolgozik
 - Iteratív sw fejlesztési módszertan (teret ad a megbízói visszajelzésnek)
 - 4 fázis minden iterációban (vizsgálat, kidolgozás, létrehozás, átállás)
- Hátrány
 - ...

RUP modell



Igény a változásra..

- ...hiszen a szoftverfejlesztés NEM gyártás
- Változások gyors és rugalmas adaptálása
- Megszabadulni a klasszikus módszertanok hibáitól
- Cél
 - Minél gyorsabban
 - Minél költséghatékonyabban
 - Az elvárt igényt minél jobban kielégítő végeredmény

„A szoftverfejlesztés találékonyságra és kommunikációra épülő kooperatív játék.”

/Alistair Cockburn/



Megoldás alternatíva

- Agilis módszertanok
- Különböző területeken
 - Projektvezetés
 - Rendszertervezés
 - Szoftverfejlesztés



Agilis, agilitás

- Szó jelentése: fürgeség
- Kiegészítés
 - Az agilitás olyan adottság, amely egyaránt képes létrehozni és reagálni a változásokra és ezzel előnyt szerezni egy turbulens üzleti környezetben
 - Az agilis szervezetek befogadják, sőt generálják a változásokat, és ezzel versenyelőnyhöz jutnak
 - Az agilis szervezetek egyrészt fürgék és rugalmasak, másrészt képesek megtalálni a káosz és rend közötti egyensúlyt.

Agilis projektvezetés és szoftverfejlesztés

- Probléma felismerése: a szoftverfejlesztés nem gyártás
- Mindig új termék készül --> fontos a kommunikáció
- Nem lehet gyártási folyamatról beszélni
- A projektmenedzsment értékei másképp (idő,pénz,tartalom,minőség)

Kiáltvány az Agilis Szoftverfejlesztésért

Mi a jobb szoftverfejlesztés módjait fedezzük fel azáltal, hogy fejlesztünk és segítünk másokat fejleszteni.

Ezen munkában értékesebbnek tartjuk:

- **Az egyént és a személyes kommunikációt, a módszertanoknál és az eszközöknél.**
- **A működő szoftvert, az átfogó dokumentációnál.**

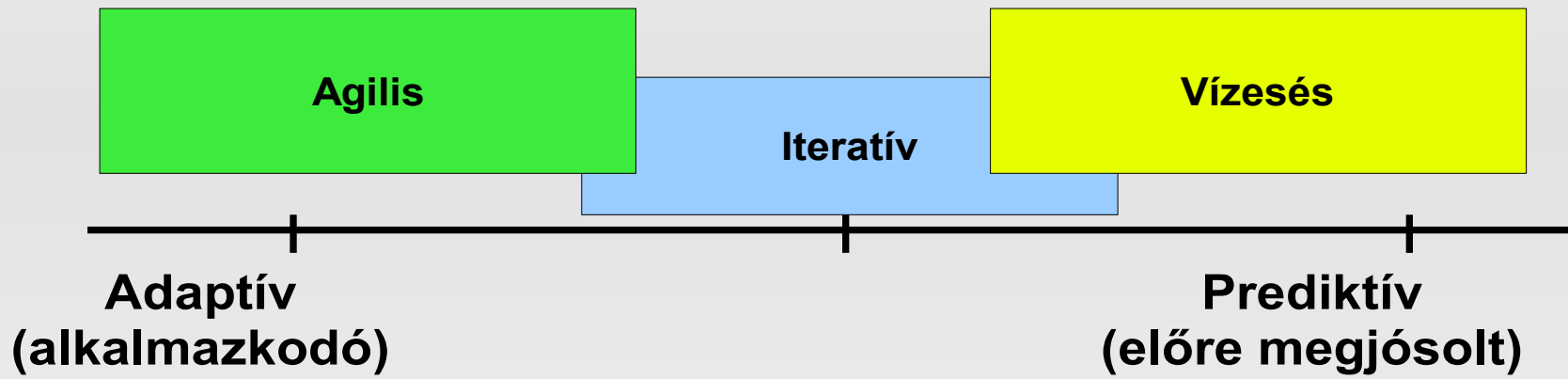
Kiáltvány az Agilis Szoftverfejlesztésért (folyt.)

- **A megrendelővel való együttműködést, a szerződéshez való merev ragaszkodással szemben.**
- **A változásra való reagálást, a tervek rigorózus követésével szemben.**

Noha, fontosak az utóbbiak is, mi fontosabbnak tartjuk az előzőeket.

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

Összehasonlítás a klasszikus módszertanokkal



- Átfedés a módszerek között
- Iteratív használata
- Agilis ↔ „Vízesés”-szerű

Főbb jellemzők

- Adaptív (alkalmazkodó)
 - Nincsen előre jóslás, hosszú távú tervezés
 - Csak a közvetlen problémára koncentrálnak
 - DE arra hajszál pontosan
 - Csak azt tudják, mit fognak „a héten” csinálni
- Prediktív (előre megjósolt)
 - Előre megtervezett lépések
 - Minden lépés az EGÉSZRE optimalizálva --> nehézkes változás követés
 - Néha külön változáskezelő bizottság

Iteratív, mint alap

- Nincs éles elhatárolódás
- Iteratív megjelenésének oka: Vízesés modell rugalmatlanság javítása
- Módszer kulcsa: a szoftvert rövidebb időközönként kiadni
- Ezen jellemző átemelése, csak kicsit másképp...

Az idő szerepe az agilis fejlesztésben

- Hónapok helyett hetek
- A határidők nem flexibilisek
- „Kőbe vannak vésve”
- Az idő az alappillér a feladat helyett!!!!
- NEM LEHET KICSÚSZNI A HATÁRIDŐBŐL
- Ha mégis, akkor
 - A feladat „megoldhatatlan”
 - Visszamondás
 - Részfeladatokra bontás



Agilis fejlesztés alapelvei I.

- Összesen 12
- Legfontosabbak
 - legnagyobb prioritása a megrendelő igényeinek megfelelő, értékes szoftver korai, és folyamatos kiadása
 - A követelmények változása elfogadott, még a fejlesztés késői szakaszában is.
 - A rövidebb periódust kell előnyben részesíteni.
 - A megrendelőknak, üzleti szakembereknek és a szoftverfejlesztőknek naponta együtt kell dolgozniuk a teljes projekt során.

Agilis fejlesztés alapelvei II.

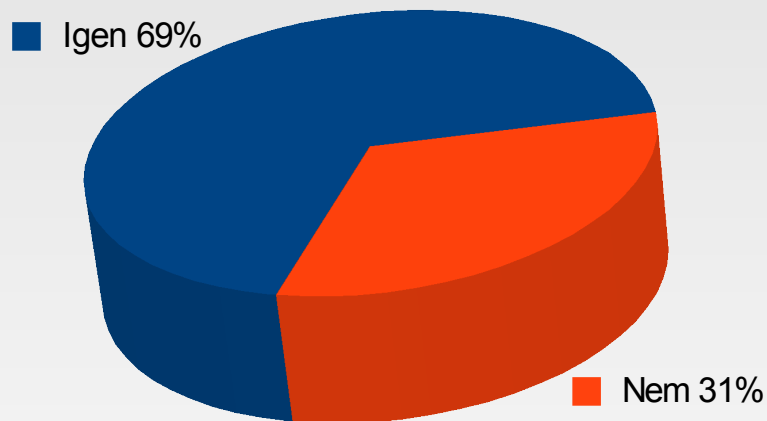
- A projekteket motivált emberekre kell építeni, kiknek megteremteni a megfelelő környezetet
- Hatékonyabb módszer az információ átadásának a fejlesztési csapaton belül, a személyes beszélgetés.
- A legjobb architektúrák, követelmények és rendszertervek az önszerveződő csapatmunkából alakul ki.
- A fejlesztői csapat, rendszeresen időközönként, megfontolja, hogy hogyan válhatnak hatékonyabbá és ennek megfelelően finomítják viselkedésüket.

Hátrányok

- Kiforratlanság (2001-ben „született”)
- Csak gyakorlott fejlesztőknél használható
- Nincs eléggé megtervezve a módszer
- Túlságosan meg kell változtatni a fejlesztési kultúrát, hogy jól működjön
- Sokan, TÉVESEN, azt állítják, hogy „cowboy” módszer

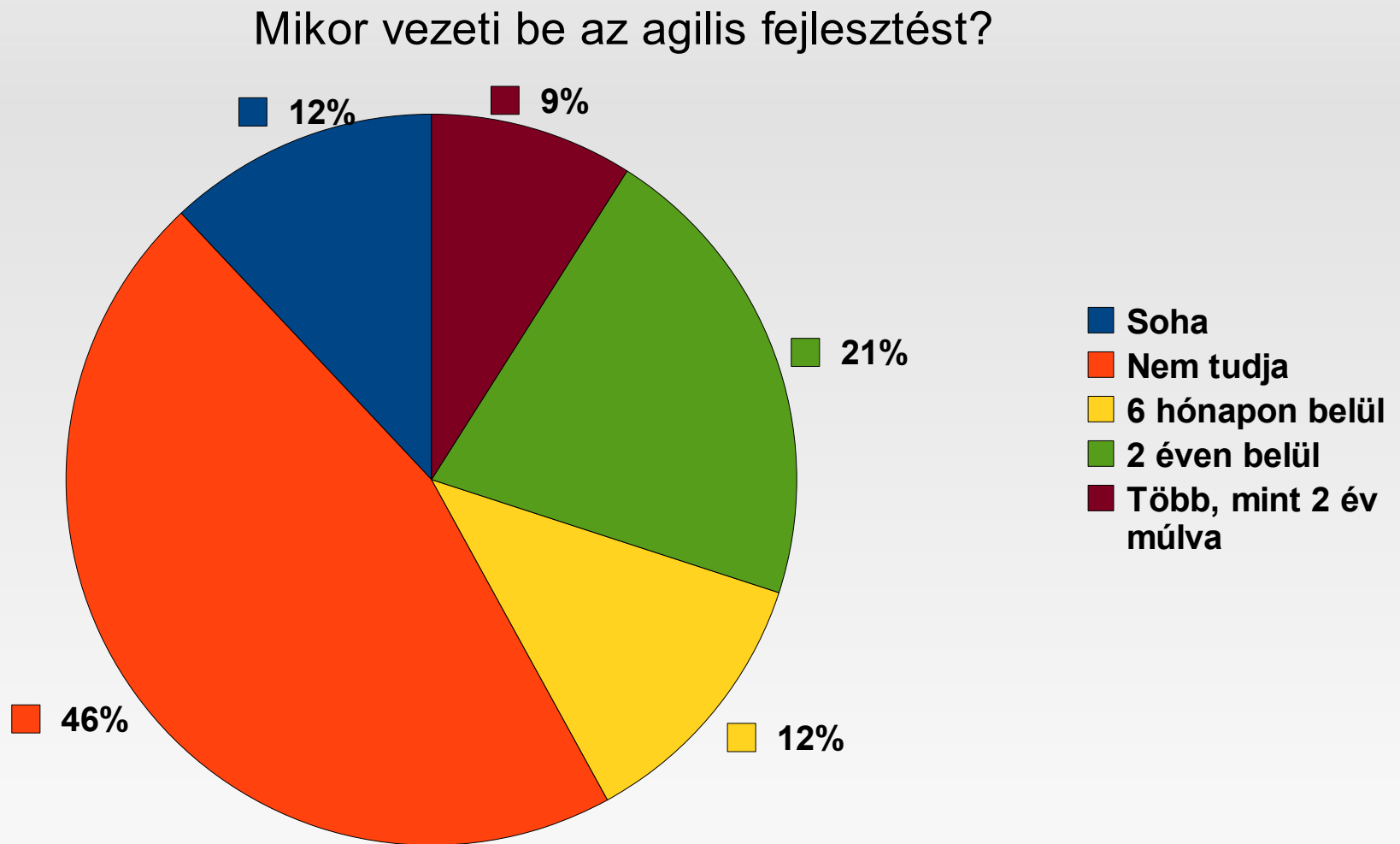
Statisztika, felmérés (2007.03)

Bevezetettek-e már agilis projektet?

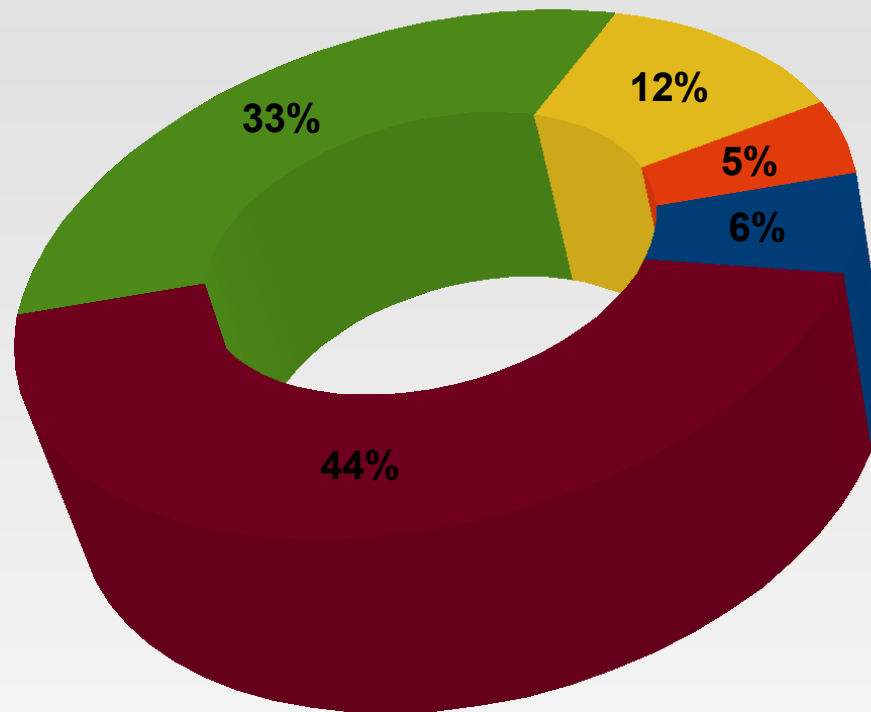


- Forrás: Agile Adoption Rate Survey
- A válaszadók 69%-a jelezte, hogy szervezetükben egy vagy több agilis projektet hajtanak végre.

Statisztika, felmérés (2007.03)



Statisztika, felmérés (2007.03)



Agilis fejlesztések sikeressége

■ Kevesebb, mint 25% ■ 25-49% ■ 50-74% ■ 75-90% ■ Több, mint 90%

Agilis módszertanok

- eXtreme Programming (XP)
- Test Driven Development (TDD)
- Feature Driven Development
- Scrum
- ...



Scrum

- Rögbiből átvett kifejezés
- Jelentése: Viaskodik, összecsap, dulakodik
- Egyéb jelentése: kavarodás
- „scrummy” = pompás, remek



Kialakulás

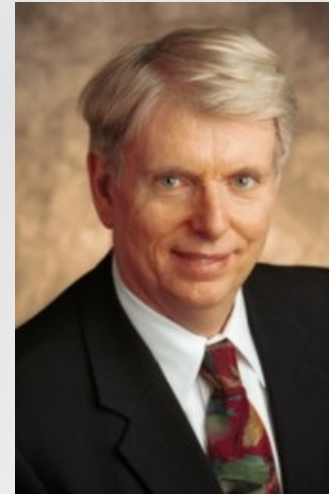
- Hirotaka Takeuchi és Ikujiro Nonaka
- Vízesés modell, mint váltófutás
 - A stafétát a program
 - A fázisok a futók
 - Ha egy futó „rossz”, a csapat veszít
- Megmaradtak a sport szemléletnél (innen a rögbi kifejezés)

Alapgondolat

- Módszer, ahol a fázisok erősen átlapolódnak
- Több terület emberei kisebb csoportokban
- És az összes fázisban együtt dolgoznak
- Lásd rögbi – együtt futnak, és közben passzolgatnak



Publikálás (1990)



- Ken Schwaber és Jeff Sutherland
- Megfigyelések, tanulmányozások --> SCRUM kialakulása

Scrum, mint agilis módszer

- Magában hordozza az adaptív jellemvonásokat
- Nincs „forgatókönyve”
- Sokan emiatt csak hozzáállásnak tartják, módszertan helyett



Scrum jellemzői

- Fejlesztő csapat – „EGYSZERRE” kezd dolgozni
 - Üzleti elemző
 - Tervező
 - Fejlesztő
- Teljes mértékben együtt felelősek a végeredményért
- Scrum Team

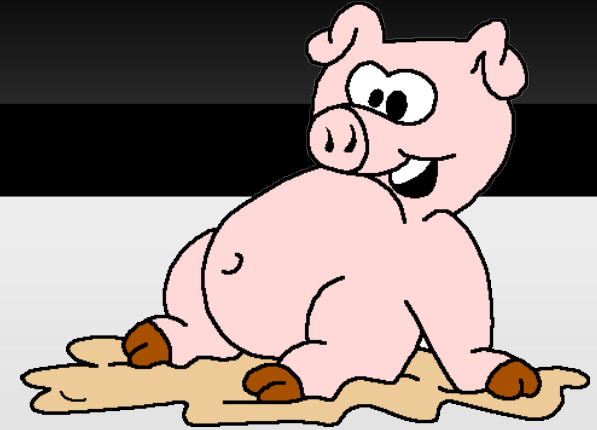
Scrum jellemzői (folyt.)

- Adaptív menedzsment biztosítása
- Megfelelő kommunikáció
 - Különböző szakterületek
- Inkrementális fejlesztés
 - Köztes termék
 - Mielőbbi hiba felfedezés
- Átlátható, világos, moduláris tervezetek
 - Ki, miért, milyen határidővel felelős
- Hatékony munkaóra kihasználás
 - Túlóra nem feltétlen pozitív!

Szerepkörök

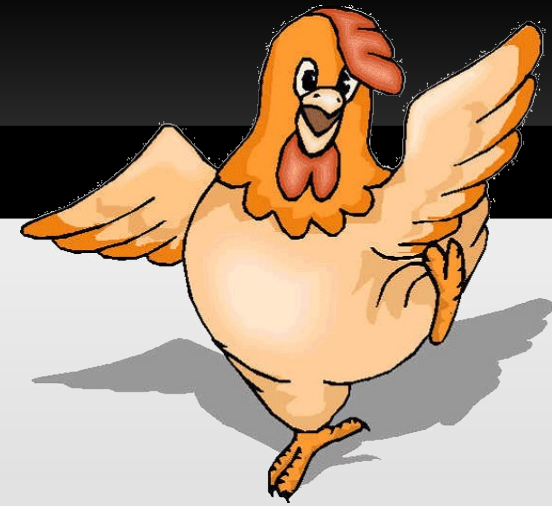
- A disznó és a csirke mennek az utcán. Egyszer csak a csirke megszólal:
- „Te, nyissunk egy éttermet!”
 - Mire a disznó:
- „Jó ötlet, mi legyen a neve?”
 - A csirke erre gondolkozik, majd azt feleli:
- „Nevezzük Sonkás-tojásnak!” (Ham and eggs)
 - A disznó erre:
- „Nem tetszik valahogy, mert én biztosan mindent beleadnék, te meg éppen csak hogy részt vennél benne.”

„Disznók”



- Akik „mindenüket” beleadják
- Terméktulajdonos
 - A vásárlót reprezentálja
- Scrum Master
 - Maga a scrum működtetés, akadálymentesítés
- Scrum Team
 - 5-9fős csapat, különböző területekről

„Csirkék”



- Közvetetten részei a folyamatnak
- Felhasználók
 - Vélemény, részeredmény (visszacsatolás)
- Stakeholder-ek
 - Pl.: rendszergazda, igazgató
- Tanácsadó szakértők
 - Akik nem szükségesek folyamatosan, csak 1-1 szakaszban válnak „disznóvá”

Dokumentumok

- Story
 - a megrendelőtől érkező lényegi leírás
- Product backlog
 - a story feldolgozása és priorizálása
- Sprint backlog
 - a konkrét feladatok feltüntetése az adott sprintre (ki, mit, milyen határidővel vállalt be)



Egyéb definíciók

- Backlog item
 - Egy teendő (funkció) a backlog dokumentumból
- Sprint
 - Rövid időszak
 - Ez alatt kell megvalósítani az adott backlog item-eket
- Burn down chart
 - egyfajta kimutatás; a csapat teljesítőképeség

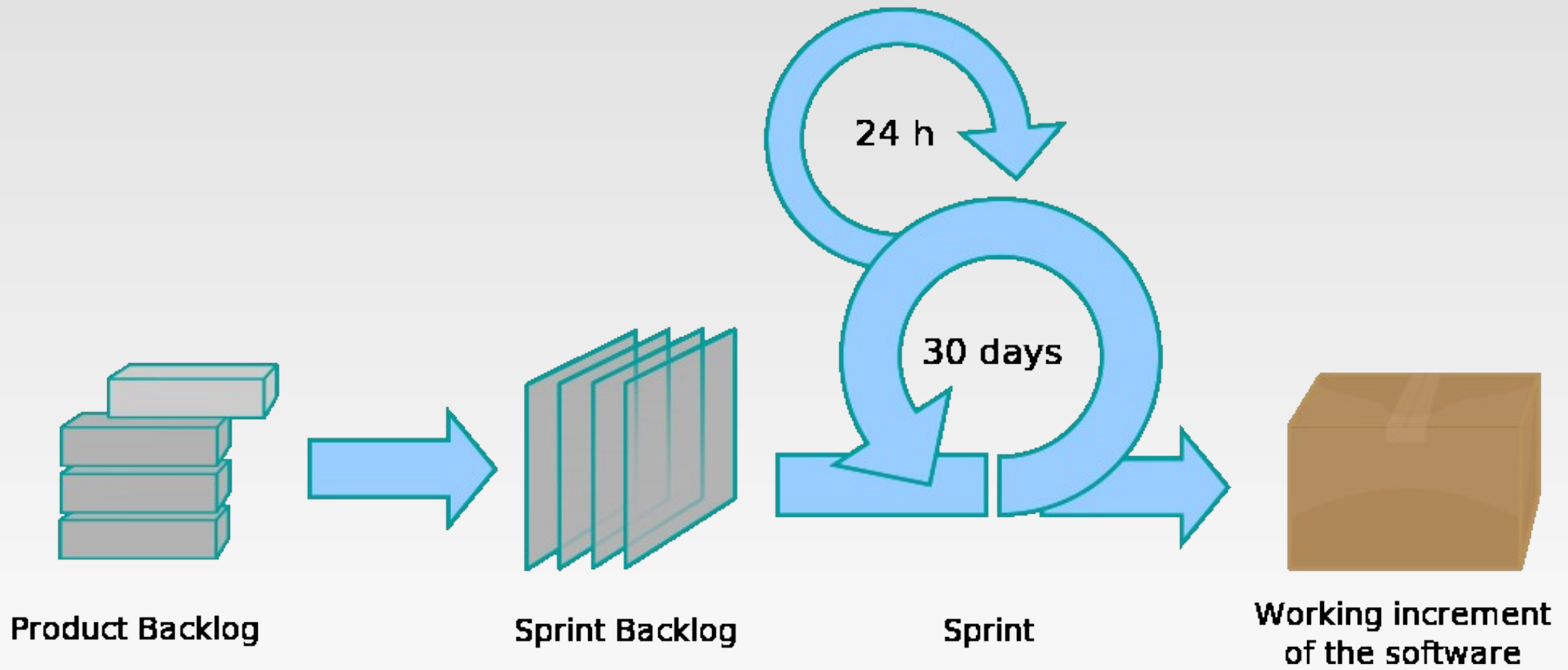
Scrum meeting

- A biztos kommunikáció
- Mindennap
 - rövid megbeszélés (~15-30perc)
- 3 alapvető kérdés mindenkihez
 - Mit csináltál a tegnapi scrum óta?
 - Mit fogsz csinálni a következő scrum -ig?
 - Van-e valami, ami akadályoz az előrehaladásban?
- Scrum master vezeti le
- Fő a NYÍLT beszélgetés



Fejlesztés menete

- ...bár nincsen igazi forgatókönyve...



Határidő és demo

- Kulcsfontosságú a határidő
- Csúsztatni nem lehet, csak
 - Visszamondani
 - Korrigálni az adott backlog item-t
- Minden sprint végén: DEMO
 - A megrendelő ellenőrzi az aktuális állapotot
 - Értékelés függvényében iterálódik tovább (visszacsatolás)

Esettanulmány, tapasztalat

- DocuGuard2 (evosoft Hungary Kft.)
- Story (Requirement Specification)
- Komplexitás becslés
- Product Backlog
- Sprint backlog
- Burn down chart



Összefoglalás

- Egyre nagyobb igények
- Egyre kisebb erőforrás használás mellett
- Változások gyors követése
- Nincs objektív megoldás
- „Van akinek tetszik, van akinek nem!”





Köszönjük a figyelmet! :)

Sereg Ákos <sigterm@c3.hu>
Varga Balázs <varga27@gmail.com>

