Agilis szoftverkészítés (Agile software development)

Szoftverkészítési módszertan/konceptuális keret

(Agilis: gyors, tettrekész, életrevaló)

Az Agilis módszerek alapelvei Az Agilis Manifesztum

1. Hasznos szoftvertermékek gyors, folyamatos szállításából fakadóan elégedett megrendelők.
2. Működő szoftver szállítása gyakran (inkább hetes, mint havi periódusban)
3. Az előrehaladás mércéje a működő szoftver
4. A követelményekben még a késői változásoknak is örülnek.
5. Szoros, napi kommunikáció a fejlesztők és a megrendelő között
6. Személyes kapcsolattartás
7. A projekteket motivált, megbízható munkatársak vezetik
8. Folyamatos figyelem kíséri a műszaki színvonalat és a tervet
9. Egyszerűség
10. Önszerveződő csapatmunka
11. A változó körülményekhez való gyors alkalmazkodás

Az agilis módszertan és más módszertanok

- Iteratív fejlesztési módszerek: hasonlatosak egymásra a ciklusos és inkrementális feljesztés kérdésében. DE: itt hetekben mérnek egy ciklust
- Vízesés modell: túlságosan tervkövető, fegyelmezett, meghatározott
- Partizán/(cowboy) kódolás: tervszerűtlen, nincsenek kiértékelési és minőségbiztosítási ciklusok
- Agilis módszertan: NEM tervezetlen vagy fegyelmezetlen, HANEM alkalmazkodókész (adaptív) vagy előretekintő(prediktív)

Az agilis módszerek használhatósága és határai

- Feltételei:
  1. A szervezet támogatja a párbeszédet
  2. Az emberekben megbíznak
  3. Kevesebb, de nagyobb szaktudású munkatárs
  4. A szervezet elfogadja a fejlesztők döntéseit
  5. A gyors kommunikációt lehetővé tevő környezet
1. Következmény: projektméret<20-40 fő (nehéz a kommunikáció)

- Agilis módszerek ellenjavallatai:
  - Nagy projektméret
  - Szétosztott, több helyszínű fejlesztés
  - Feladatkritikus vagy életfontosságú fejlesztések
  - Parancsosztó vállalatvezetési stílus
- Agilis javallatok:
  - Alacsony kritikussági fok
  - Tapasztalt fejlesztők
  - Változó követelmények
  - Kis projektméret
  - A káoszon virágzó fejlesztési kultúra
- Tervszerű javallatok:
  - Magas kritikussági fok
  - Pályakezdő fejlesztők
  - Rögzített követelmények
  - Nagy projektméret
  - Rendre alapuló fejlesztési kultúra

Agilis módszerek: XP Extrém programkészítés

- Kommunikáció: megrendelők, fejlesztők között
- Egyszerűség: a legegyszerűbb megoldást fejlesztjük ki legelőször
- Visszacsatolások:
  - A rendszertől: egységtesztekkel
  - A megrendelőtől: elfogadási teszteket a megrendelővel együtt írják
  - A fejlesztőktől: megrendelőkkel közösen tárgyalják az új fejlesztések igényeit
- Bátorság:
  - A mának kódolunk, nem a holnapnak.
  - Szükség esetén átírjuk a kódot (refactoring)
  - Szükségtelen kód törlése (bármennyi munka is van benne)
- Tisztelet: a munkatársaink iránt

XP gyakorlata

- Programozás kettesben
  - Megnövekedett fegyelemérzet
  - Jobb kódminőség
  - Rugalmas munkafolyamatok, megszakítástűrés
  - Több tervező/a párok cserélődhetnek
  - Élvezetesebb munka egyes programozók számára
  - Közös kód-tulajdonlás
  - Mentorszellem/tudás/tapasztalat terjesztés
  - A munkatársak gyorsabban megismerik egymást
  - Kevesebb megszakítás
  - Eggyel kevesebb munkaállomás

XP gyakorlata

- Tervezési játék: hetenkénti gyűlés során
- Tesztvezérelt fejlesztés (TDD):
  - előbb a teszteseteket írjuk meg, aztán a kódot
  - Automatizált egységtesztek – minden teszt rendszeres futtatása
  - Szükség szerint kódátírás (refactoring)
- Fejlesztési elvek:
  - „Keep It Simple Stupid” (KISS): ne bonyolítsuk feleslegesen
  - „You Ain’t Gonna Need It” (YAGNI)
- „Tesztvezérelt fejlesztési mantra”:
  - Tesztírás
  - Kódváz írás→a teszt nem sikerül, a hiba észlelése
  - Kódátírás→ a hiba kiküszöbölése, a teszt sikerül

Scrum (kavarodás, csetepaté, közelharc)

- Priorizált lista az elkészítendő fejlesztésekről
  - A lista egy részének elvégzése egy iteráció keretében
  - Scrum Tervező Gyűlés
  - Scrum napi gyűlés
  - Scrum kiértékelés
    - Scrum mester: a feladat végrehajtását akadályozó (elsősorban pszich- szoci-ológiai) tényezők kiküszöbölése
    - Disznó-csirke-Ham and Eggs-Committed/Involved
    - Max 5-9 ember

Feature Driven Development (FDD) (jegyvezérelt fejlesztés)

- Rövid iterációs(ciklus) idejű, modellvezérelt fejlesztés
- 5 alapvető tevékenység:
  - Átfogó modell megalkotása: a rendszer határainak meghatározása, minden modellterületre külön modell, kiscsoportos kidolgozás, ezek valamelyike, együtt vagy egyesítve adják az átfogó modellt
  - Jegyek listájának létrehozása: a teljes célterület lebontása tárgyakra-üzleti tevékenységekre, ezek lépései adják a jegyek listáját <tevékenység><eredmény><tárgy> formában.
  - Jegyenkénti projekttervezés: 1 jegy→1 osztály→1 felelős
  - Jegyenkénti szoftvertervezés: jegyekből fejlesztési csomagok… szekvenciadiagramok… osztálydiagramok… eljárásspecifikációlk
  - Jegyenkénti szoftverösszeállítás (build): kódolás, egységteszt, kódszemle→célszoftver összeállítása
- Mérföldkövek: átfogó modell, terv, tervszemle, kód, kódszemle, végső összeállítás

Feature Driven Development gyakorlati módszerei

- Domain Object Modelling: szakterületi modell készítése, amihez könnyen adhatunk jegyeket
- Jegyenkénti fejlesztés: Ha egy jegy 2 hétnél összetettebb, akkor szét kell bontani
- Egyéni kód (osztály) birtoklás
- Jegyfejlesztő csapatok: kicsi hatékony csapat
- Szemlézés: hibafelderítés, minőségbiztosítás céljából
- Konfigurációkezelés
- Rendszeres összeállítás (builds): mindig van egy naprakész futtatható rendszer
- A folyamat és az eredmények láthatósága

Feature Driven Development és a Metamodellek

- A feladat leírásához legmegfelelőbb nyelv megtervezése (később használata)
- Metaadat modellezés – a modellezési fogalmak rendszere
- Metafolyamat modellezés – a modellezési folyamatok rendszere
- Modellátalakító nyelv létrehozása
- UML Eszközök:
  - Adatmodell
  - Aktivitásdiagram
  - Folyamat-aktivitás diagram (tevékenységek és eredmények összerendelése)