KLIMKÓ Gábor

AZ AGILIS SZEMLÉLET ELSŐ KÉT ÉVTIZEDE

Az agilitás az elmúlt húsz év egyik fontos fogalma, amelyet több területen (gyártás, szoftverfejlesztés és projektmenedzsment) is több-kevesebb eredménnyel alkalmaztak. A hazai akadémiai diskurzus eddig mostohagyermekként kezelte e témákat, holott az érintett szakmák gyakorlati művelői számos alkalommal foglalkoztak velük. A szerző rövid áttekintést ad e területek fontosabb gondolatairól, majd ezek közül néhányat a vezetés-szervezés általánosabb tárgyalásmódjával vizsgál. Az agilis szemlélet mögött rejlő alapgondolatok egy része nem nevezhető újnak. A cikk az agilitás témájának pozicionálásával zárul.[1]

Kulcsszavak: **agilitás, agilis gyártás, agilis szoftverfejlesztés, agilis projektmenedzsment**

Meglepő módon az agilitás, az agilis megközelítés vizsgálata nem része a magyar akadémiai diskurzusnak, holott a téma mind a szoftverfejlesztés, mind a projektmenedzsment szakmai berkeiben már évek óta terítéken van Magyarországon is (Hinsemkanpf, 2007; PMI, 2009; Prónay, 2011; PMSZ, 2011). Ez a hiátus azért is meglepő, mert kifejezetten az agilitás témájával foglalkozó magyar szakmai szervezetek is létrejöttek (Agilis Szoftverfejlesztők Egyesülete, PMSZ Agilis PM tagozat). Jelen tanulmányunk célja két évtized tapasztalatainak összegzéséből kiindulva, a projektmenedzsment területén használható agilis vezetés-szervezési technikák esszenciájának összegyűjtése.

Az agilitás kezdetét sokan (tévesen) a szoftverfejlesztés területén meghatározó módszertani szakértők által kibocsátott „agilis kiáltványhoz" kötik, holott az agilitás fogalma már jóval korábban megjelent a gyártás kontextusában. Manapság a vállalati szférában az agilis kifejezést már kiterjesztett értelemben, minőségjelzőként (is) használják, és számos fogalmat árnyalnak vele, így többek között beszélnek „agilis szervezetről" (Fister, 2012), „agilis projektmenedzsmentről" (Wysocky, 2002; Chin, 2004), „agilis termékfejlesztésről". Az agilis szemléletmód jelenségét már több évtizede tárgyalja a szakirodalom, az egyes fogalmakat azonban gyakran eltérő értelemben használják a szerzők. Az angol szakkifejezések eleve tág értelmezést tesznek lehetővé, lefordításuk magyarra az eredeti gondolatok esetleges torzulását eredményezhetné, ezért a tanulmányban több helyen az eredeti nyelven szerepel idézet.

Az agilis gyártás

A témakörben a legkorábbi említésre méltó forrás Nonakának és Takeuchi-nak az innovatív termékfejlesztés jellemzőiről írott cikke. A publikáció a változó körülményekhez való alkalmazkodás, a gyors, ugyanakkor flexibilis fejlesztés új megközelítését írja le, amelyet a szerzők a rögbiből kölcsönzött hasonlattal „scrum-nak" neveznek (Nonaka – Takeuchi, 1986). A rögbiben a scrum a játék megszakítás utáni újrakezdését jelenti, amikor alkalmazkodni kell az ellenfél játékához. Nonaka és Takeuchi nem használja az agilitás fogalmát, viszont az általuk vizsgált probléma éppen az, ami az agilis szemléletmód kialakulásához vezetett.

Az agilitás kifejezés az 1990-es évek elején került be a köztudatba, méghozzá az agilis gyártás *(agile manufacturing)* témakörében. 1991-ben több mint 150 szakértő együttes munkájának eredményeképpen született meg a *„21st Century Manufacturing Enterprise Strategy"* című jelentés (Nagel, 1992), innen származik az elnevezés.

Yusuf és szerzőtársainak cikke jó betekintést ad az agilis gyártás területére. A változó környezet általános fogalmát négy részterületre bontják, és véleményük szerint ezek az agilitás mozgatórúgói (drivers). Yusuf és társai az agilitás fogalmát így határozzák meg: *„A manufacturing system with extraordinary capabilities (internal capabilities: hard and soft technologies, human resources, educated management, information) to meet the rapidly changing needs of the marketplace*

(speed, flexibility, customers, competitors, suppliers, infrastructure, responsiveness)" (Yusuf et al., 1999). Gunasekaran irodalmi áttekintésében pedig így összegzi az agilis gyártás fogalom tartalmát: *„Agile manufacturing can be defined as the capability of surviving and prospering in a competitive environment of continuous and unpredictable change by reacting quickly and electively to changing markets, driven by customer-designed products and services."* (Gunasekaran, 1999)

Az agilis gyártás témájában számos publikáció jelent meg, ezeket Sanchez és Nagi 73 cikk feldolgozásával tekintette át. Kilenc fő kutatási területet azonosítottak, bár megjegyzik, hogy az általuk feldolgozott források jelentős része három szakfolyóiratból származik (azaz lehet itt némi partikularitás). A kilenc terület: gyártórendszerek tervezése; folyamattervezés; gyártástervezés, -ütemezés és -ellenőrzés; anyagtároló rendszerek; információs rendszerek; ellátásilánc-stratégiák; emberi tényezők és az üzleti gyakorlat és folyamatok leírása. Szám szerint a legtöbb publikáció (21 darab) az információs rendszerek kérdéseivel foglalkozott. Sanchez és Nagi területenként vetnek fel további kutatási kérdéseket a tárgykörben (Sanchez – Nagi, 2001).

Sanchez és Nagi cikkükben rámutatnak arra, hogy az *„agilis gyártás"* és a *„lean gyártás"* eltérő fogalmak. Ezt azért érdemes megjegyezni, mert egyes szerzők manapság is összekötik e két fogalmat, holott azok gyökeresen különböző jelenségeket írnak le. A lean szemlélet alapvetően a hatékonyságról szól, az értéket nem teremtő tevékenységek valamilyen értelemben vett minimalizálásáról (Womack – Jones, 2009). Az agilis megközelítés viszont egy adott feladat eredményes elvégzését teszi lehetővé turbulens környezetben. A hatékonyság nem azonos az eredményességgel.

Ezen a ponton lényeges az „agilis" (angolul „agile") szó jelentésének körbejárása. Jelen kontextusban a sportból vett analógiaként arra utal, hogy képesek vagyunk a gyorsan mozgó cél követésére. Nem tévesztendő össze tehát a fürgeséggel, bár kétségtelenül van némi áthallás, hiszen a követés képessége magával vonhatja az események felgyorsulását.

Az agilitás megjelenése természetesen maga után vonta az agilis gyártás megteremtését megalapozni szándékozó javaslatokat is, melyek meglehetősen általánosak (Gunasekaran, 1998; Shairif – Zhang, 1999; Zhang – Sharif, 2000). A szoftverfejlesztés területén ennél jóval konkrétabb és célirányosabb javaslatok születtek az agilis szemléletmód átvételére.

Az elmúlt évtizedben kevesebb publikáció jelent meg az agilis gyártás területén. Calvo és társai az agilitás fenntartását vizsgálják (Calvo et al., 2008), Inman és társai empirikusan támasztják alá azt az amúgy nyilvánvaló elvárást, hogy az agilitás hatékonyabbá teszi egy vállalat működését (Inman et al., 2011). Zhang 2011-ben az agilitást még mindig új, bár elfogadott koncepciónak nevezi (Zhang, 2011). Úgy tűnik, az agilis gyártás kérdése bizonyos értelemben kiforrottá vált, és így egyre inkább a kuhni értelemben vett normál tudomány vadászterületévé válik.

Az agilis szoftverfejlesztés

A szoftverfejlesztés területén az agilis szemlélet kialakulását az „agilis kiáltványhoz" szokás kötni, melyet meghatározó módszertani szakértők jegyeznek (Beck et al., 2001).

A szoftverfejlesztés sajátos kockázatokkal járó feladat. Az információtechnológiai rendszerek szociotechnikai volta miatt szinte szükségszerűek a tanulási ciklusok, melyek fontosságát rendre alá szokták becsülni. Ezért is olyan magas a sikertelen szoftverfejlesztési projektek aránya, mely állítás alátámasztására a The Standish Group 1994 óta évente publikált jelentéseit szokták felhozni.

A szoftverfejlesztés nehézségei már évtizedek óta ismertek. A múlt század hatvanas éveinek végén jelent meg a „szoftverkrízis" fogalma, amelyet egy Garmisch-Partenkirchenben tartott NATO-konferenciához szoktak kötni (Naur – Randell, 1969). A szoftverkrízis megoldását akkortájt a tervezés éthoszának felmagasztalásában vélték fellelni. A szoftverfejlesztés folyamatát lineárisnak vélték (ez az ún. vízesés-életciklusmodell), aminek az első szakaszaiban kell a megfelelő figyelmet és technikákat alkalmazni, és akkor bizonyos lesz a siker. Számos strukturált szoftverfejlesztési módszertan (SSADM, Merise, Vorgehensmodell, Method-1, SDM stb.) került kifejlesztésre, létrehozták ezek intézményes környezetét, azaz oktatási, vizsgáztatási és akkreditációs sémákat. A tervezés mindenhatósága felmagasztalásának eredményeképpen a tervezési dokumentumok száma és vastagsága lenyűgöző lett, összességében azonban a szoftverfejlesztési projektek sikerességi aránya nem javult lényegesen. A nagy szavak kedvelői hasonlíthatják e folyamatot a kuhni „paradigmatikus válság" jelenségéhez.

A múlt század utolsó évtizedében aztán számos újabb módszertani próbálkozás látott napvilágot. A következő felsorolás, a teljesség igénye nélkül, a vonatkozó referenciaműre való időrendi hivatkozás sorrendjében nevesít néhányat a szoftverfejlesztési folyamat eredményességének jobbítását célzó irányzatok közül:

- Dynamic Systems Development Method (Stapleton, 1997),

- Unified System Development Process (Jacobson – Booch – Rumbaugh, 1999),
- Extreme Programming (Beck, 1999),
- Adaptive Software Development (Highsmith, 2000).

A fenti irányzatok mindegyike a változó körülményekhez való gyorsabb alkalmazkodás képességét ígérték, meglehetősen hasonló utakat ajánlva. A Dynamic Systems Development Method és a Unified Process iteratív és inkrementális fejlesztési megközelítést ajánlottak, az Extreme Programming gyakori (szoftver) kibocsátási ciklusokat írt elő, melyek végén az új felhasználói követelményeket be lehet fogadni, az Adaptive Software Development pedig iteratív, feszes határidejű (time-boxed) fejlesztési megközelítést javasolt. Majd 2001-ben egy szinte már-már misztikus összejövetelen a szoftveripar nagyjai együttesen álltak elő egy „kiáltvánnyal az agilis szoftverfejlesztésért" (Beck et al., 2001). A kiáltványban a szerzők mellbevágó hangsúlyeltolódásokat fogalmaztak meg, úgymint:

- az egyén és a személyes kommunikáció fontosabb, mint a módszertanok és a fejlesztési eszközök,
- a működő szoftver fontosabb, mint a szoftver teljes dokumentációja,
- a megrendelővel való együttműködés fontosabb, mint a fejlesztési szerződés tárgyalása,
- a változásra való reagálás fontosabb, mint a tervek fegyelmezett követése.

A fenti állítások lényege az, hogy kiáltvány szerzői többre tartják az előresorolt dolgokat, bár az összehasonlításokban másodikként szereplő jellemzők is fontosak. A kiáltvány honlapján 12 „agilis alapelvet" is megfogalmaznak, sajnos egyedi azonosító nélkül, ami körülményessé teszi a hivatkozásukat, ezért itt szó szerint megismételjük őket, a honlap magyar nyelvű változata szerint.

„Mi a következő elveket követjük:

- *Legfontosabbnak azt tartjuk, hogy az ügyfél elégedettségét a működő szoftver mielőbbi és folyamatos szállításával vívjuk ki.*
- *Elfogadjuk, hogy a követelmények változhatnak akár a fejlesztés vége felé is. Az agilis eljárások a változásból versenyelőnyt kovácsolnak az ügyfél számára.*
- *Szállíts működő szoftvert gyakran, azaz néhány hetenként vagy havonként, lehetőség szerint a gyakoribb szállítást választva!*
- *Az üzleti szakértők és a szoftverfejlesztők dolgozzanak együtt mindennap, a projekt teljes időtartamában!*
- *Építsd a projektet sikerorientált egyénekre! Biztosítsd számukra a szükséges környezetet és támogatást, és bízz meg bennük, hogy elvégzik a munkát!*
- *A leghatásosabb és leghatékonyabb módszer az információ átadásának a fejlesztési csapaton belül, a személyes beszélgetés.*
- *A működő szoftver az elsődleges mércéje az előrehaladásnak.*
- *Az agilis eljárások a fenntartható fejlesztést pártolják. Fontos, hogy a szponzorok, a fejlesztők és a felhasználók folytonosan képesek legyenek tartani egy állandó ütemet.*
- *A műszaki kiválóság és a jó terv folyamatos szem előtt tartása fokozza az agilitást.*
- *Elengedhetetlen az egyszerűség, azaz az elvégezetlen munkamennyiség maximalizálásának művészete.*
- *A legjobb architektúrák, követelmények és rendszertervek az önszerveződő csapatoktól származnak.*
- *A csapat rendszeresen mérlegeli, hogy miképpen lehet emelni a hatékonyságot, és ehhez hangolja és igazítja az működését."* (Beck et al., 2001)

Látható, hogy az agilis elvek szervezési, és nem műszaki előírásokat adnak meg.

Az agilis szoftverfejlesztés néhány jellegzetessége

Az agilis kiáltványban megfogalmazott hangsúlyeltolódások következtében lényegesen megváltozott a szoftverfejlesztés módszere. A hagyományos felfogás szerint az elkészítendő cél alapján határozták meg az idő és költség/erőforrás igényeket, az agilis szemlélet szerint viszont pont fordítva kell eljárni: az idő és költség/erőforrás keretek alapján kell az elvégezhető feladatot meghatározni (Dalcher, 2009; hivatkozza Kosztyán – Kiss, 2011: p. 34.). Ennek megfelelően egy agilis fejlesztés iteratív és inkrementális lesz. Egy iterációban a megrendelő által kockázat és érték alapján legfontosabbnak ítélt funkciókat valósítják meg, és az iteráció végén pontosítani lehet a követelményeket. Ez a szemlélet gyökeres szakítást jelent a termékalapú tervezés mindenhatóságát hirdető hozzáállással, és valójában nem más, mint annak a nyílt beismerése, hogy ebben nem tudunk pontosan előre tervezni. Megjegyzendő, hogy a szoftverfejlesztés világában az iteratív szemlélet már Boehm 1986-ban publikált spirális fejlesztési modelljében megjelent (Boehm, 1986). A moduláris (avagy inkrementális) szoftverfejlesztés gondolata például Parnas nevéhez köthető (Parnas, 1971), de akár még korábbra is datálható lenne.

Az agilis kiáltvány után napvilágot látott számos újabb módszertani megközelítés az iteratív és inkrementális megközelítésre épített, úgymint:

- Feature Driven Development (Palmer – Felsing, 2002),
- Scrum (Schwaber - Beedle, 2002),
- Test Driven Development (Beck, 2002),
- Kanban (Anderson, 2003),
- Crystal Clear (Cockburn, 2004).

A fenti megközelítések közül az Egyesült Államokban évenként lebonyolított „agilis felmérés" (VersionOne, 2013) szerint a legnagyobb ismertsége a Scrum-nak van, ezért a továbbiakban elsősorban ennek a gondolatmenete szerint mutatjuk be az agilis szoftverfejlesztést.

A „scrum" a rögbijátékban használt alakzat, amely a labda védelmét szolgálja a támadókkal szemben, akik természetesen mozognak. Mint említettük, a „scrum" szót Nonaka és Takeuchi használta az innovatív termékfejlesztés új módjának leírására (Nonaka - Takeuchi, 1986). A zöldmezős szoftverfejlesztés tipikusan innovatív termékfejlesztés, így alkalmazható rá a szerzők módszere. Nonaka és Takeuchi az innovatív termékfejlesztés hat sajátosságát azonosította, melyek közül az első (az instabilitás) éppen arra hívja fel a figyelmet, hogy egy innovatív termék előállításának miértjét nem lehet precízen előre megtervezni.

A Scrum módszer köré komoly infrastruktúra épült ki az évek során. A módszer kézikönyve (*Scrum Guide,* lásd Sutherland – Schwaber, 2011) több nyelven elérhető, illetve a módszertan elsajátítása esetén többféle (egyéni) minősítés is szerezhető, lásd http://www.scrumalliance.org/certifications/. Az alábbiakban a *Scrum Guide* magyar fordításának terminológiáját használva összefoglaljuk a fejlesztés menetét, eljárásait és a betöltendő szerepeket.

A Scrum terminológiájában egy iteratív fejlesztési ciklust sprintnek neveznek, az adott sprint során a vizsgált időpontban még hátralevő, megvalósítandó szoftver funkcionalitást „sprint teendőlistának" (*sprint backlog*) hívják. A Scrum feltételezi, hogy nagyvonalakban ismert az összes, a sprint során elkészítendő termék terjedelme, ezért használja a „termék-teendőlista" (*product backlog*) fogalmát is.

A Scrum kisebb fejlesztőcsapatokat (3–9 fő) tételez fel. Két szerepet különböztet meg, a „terméktulajdonost" (*product owner*), egy olyan felhasználót értve ezalatt, aki lényegében meg tudja fogalmazni a fejlesztendő szoftvertől elvárt funkcionalitást, illetve „Scrum mestert" (*scrum master*), aki a Scrum szabályainak megértéséért és betartatásáért felelős. A fejlesztőcsapat további tagjait nem különbözteti meg névvel a módszer.

A Nonaka és Takeuchi által azonosított második sajátosság a fejlesztőcsapat önszerveződő volta (*self organizing project teams*). A Scrum-ban ennek megfelelően a csapat tagjai önként (önszerveződően) vállalják fel a sprint feladatait. Egy sprint időtartama legfeljebb egy hónap, egy feladat időigénye pedig a sprint első részeiben legfeljebb egy munkanap.

Nonaka és Takeuchi felhívta a figyelmet a fejlesztési folyamat kellő mélységű követésének (*subtle control*) szükségességére is. Ez a Scrum-ban *„napi scrum-nak"* nevezett munkaértekezleten történik, ami a következő 24 óra tervezését szolgálja. A *napi scrum* legfeljebb 15 percig tarthat, minden résztvevőnek három kérdésre kell válaszolnia:

- Mit sikerült elvégeznie az előző megbeszélés óta?
- Mit fog csinálni a következő megbeszélésig?
- Milyen akadályozó tényezőket lát?

A sprint végén kerül sor a „sprint áttekintőre" (*sprint review*), ahol áttekintik az elkészült és az el nem készült részeket, megvizsgálják ennek okait és lehetséges következményeit. Ezen a ponton lehet áttekinteni azt is, hogy miként áll a teljes termék megvalósítása. A „sprint visszatekintés" (*sprint retrospective*) keretében pedig felmérhetik a fejlesztőcsapat és módszer javítását célzó intézkedéseket. Ezek után kerülhet sor a következő sprint megtervezésére, ahol gyakran olyan funkcionalitást határoznak meg, amely a sprint eredményes befejezése esetén azonnal használható.

A Scrum útmutató nem írja ugyan elő, de számos esetben élnek azzal a lehetőséggel, hogy a fejlesztők egyetlen közös teremben dolgozzanak, amely csak a fejlesztési projekt munkatársainak áll a rendelkezésére. Az előrehaladás tervezésére és követésére lehetőség szerint vizuális eszközöket (ábrákat) kell használni, melyeket minden résztvevő jól tud követni.

Említést érdemel még ehelyütt a kanban megközelítés alkalmazása az agilis szoftverfejlesztés világában. A kanban (jelzőkártya) rendszer alapötlete szintén a gyártás világából származik, ahol egy kártya kíséri az éppen megmunkálás alatt álló munkadarabot (Womack – Jones, 2009). A szoftverfejlesztés világában a részfeladatokat egy áthelyezhető kártyára (pl. Post-it) jegyzik fel, és egy mindenki által jól látható táblán helyezik el. A tábla oszlopai az egyes fejlesztési feladatokon végezhető munkafázisokat jelentik. A kanban rendszerben azt írják elő, hogy minimalizálni kell az egy időben végrehajtott, azonos munkafázisban levő feladatok számát. A kanban másik fontos eleme az ún. „húzóelv" alkalmazása, ami jelen esetben azt jelenti, hogy a csapat választja meg, mikor és mennyi munkát vállal fel, azaz nem kívülről tolják rájuk (Kniberg – Skarin, 2010). A Scrum-ot és a kanban együttes alkalmazását időnként Scrumban névvel is szokás illetni. E megközelítésről Corona és Pani ad összefoglaló képet (Corona – Pani, 2013).

Az agilis fejlesztés jellemzőinek enciklopédikus, több mint 400 oldalas leírását adta közre Coplien és Harrison, ami tartalmában kilép a szoftverfejlesztés világából (Coplien – Harrison, 2006). A szerzők a *„pattern language”* fogalmát használják, melyet Christopher Alexander építész talált ki. A *„pattern language”* kifejezés nagyjából „összetartozó mintázatok nyelvének” fordítható. Alexander módszere strukturált technikát ad egy adott szakterület ismereteinek leírására, melynek segítségével az adott szakterület problémáit meg lehet oldani. A leírás része egy szótár, a mintázatok gyűjteménye. Minden egyes mintázat egy-egy probléma megoldását segíti, a mintázat leírásában szerepel az a kontextus, amiben a probléma felmerül, majd annak egy lehetséges megoldása. Az egyes mintázatok más mintázatokra is hivatkozhatnak.

Például Coplien és Harrison egyik mintázatának a neve „bizalmi légkör” (*Community of trust*) kialakítása, ami szükséges egy jól működő fejlesztő szervezetben. A kontextus az, hogy az együtt dolgozó emberek munkáját segítheti, vagy éppen hátráltathatja az, hogy milyenek az interperszonális kapcsolatok. Ha a csapat tagjai bíznak egymásban, akkor a munka is gördülékenyebb lesz (Coplien – Harrison, 2006: p. 57.)

Coplien és Harrison nem vizsgálja az általuk leírt mintázatok összességének teljességét, hanem azokat egyféle kiindulópontnak tartják, amit az alkalmazóknak testre kell szabniuk (Coplien – Harrison, 2006). A szerzők 93 ilyen mintát sorolnak fel, melyeket négy „nyelvbe”, azaz szakterületbe sorolnak. Ez a négy „nyelv” a projektmenedzsment (*Project Management Pattern Language*), a fokozatos (szervezeti) növekedés (*Piecemeal Growth Pattern Language*), a szervezeti stílus (*Organizational Style Pattern Language*) és maga a (program)kód (*Code Pattern Language*). E „nyelvekben” vegyesen találhatók olyan minták, amelyek a szoftverfejlesztésre vonatkoznak, illetve olyanok, melyek bizonyos értelemben túllépnek azon (Coplien – Harrison, 2006). A „projektmenedzsment” és „fokozatos (szervezeti) növekedés” nyelvére még visszatérünk.

Nem célja jelen tanulmánynak a Scrum (és általában az agilis) szoftverfejlesztés alkalmazhatóságának és korlátainak vizsgálata, valamint kritikus értékelése, de megjegyzendő, hogy ez (sem) csodaszer. Léteznek olyan feladatok és körülmények, ahol az agilis szoftverfejlesztési megközelítés eleve nem lehet eredményes.

Az agilis szoftverfejlesztés sajátosságait, eredményeit és korlátait a VersionOne vállalat évente áttekinti az ún. agilis felmérésben (VersionOne, 2013). A 2012-es, hetedik agilis felmérés résztvevőinek 90%-a szerint az agilis módszerek használata javította a változó követelményekhez való alkalmazkodási képességüket, 85%-uk szerint javította a szoftverfejlesztés hatékonyságát, 84%-uk szerint mind a projekt átláthatóságát, mind a projektcsapat morálját (VersionOne, 2013). A felmérés reprezentativitását nehéz megítélni, de a számok kétségtelenül lenyűgözőek.

Az agilis projektmenedzsment

Az agilis szoftverfejlesztés módszerei nem műszaki jellegűek, hanem a fejlesztési folyamat kézbentartásának, menedzselésének miként jét változtatják meg. Nem meglepő tehát, hogy manapság a projektmenedzsment területén is előszeretettel használják az agilis szót minőségjelzőként. Az évek során több olyan mű is megjelent, amely kifejezetten a szoftverfejlesztés irányából közelíti meg az agilis projektmenedzsment témakörét (Highsmith, 2004; Beck, 2004; Highsmith, 2010; Goodpasture, 2010; Cobb, 2011). E művek közös jellemzője, hogy a szoftverfejlesztés specifikus életciklusmodelljéhez illesztik a sajátos javaslataikat. Az agilis jelző annyira sikeresnek bizonyult, hogy mára önálló, a piac által létrehozott egyéni projektmenedzseri minősítési rendszerek is léteznek (PMI Agile Certified Practitioner, illetve Agile Project Management, AgilePM®). Ezek a minősítések azonban tartalmilag még mindig a szoftverfejlesztés világához kötődnek.

Hass a szoftverfejlesztés területén az agilis projektmenedzsment kilenc összetevőjét (*management component*) írja le. Ezek az összetevők az alábbiak: vizuális (minden résztvevő által látható) előrehaladáskövetési technika, egy légtérben történő munkavégzés, a tesztek (azaz a visszacsatolás) által diktált fejlesztés, adaptív követés, együttműködésre alapozott munkavégzés, adott részletekre (*feature*) fókuszáló fejlesztés, az együttműködés hangsúlyozása az ellenőrzéshez képest, bevétel-központúság a költségfókusz helyett és folyamatos tanulási ciklusok alkalmazása (Hass, 2007). Hass felhívja a figyelmet arra is, hogy a fenti összetevők nem minden esetben alkalmazhatók.

A szoftverfejlesztés specifikumain túllépve, az elmúlt évtizedben olyan művek is megjelentek, melyek az agilis projektmenedzsment fogalmát általában tárgyalják. A továbbiakban ezek közül tallózunk.

Az agilis kiáltvány megjelenéséhez képest meglepően gyorsan, 2002-ben jelent meg Wysocky agilis projektmenedzsmentet keretbe foglaló műve, amely mára már öt kiadást ért meg (Wysocky, 2009). Wysocky általában tárgyalja projektmenedzsment-témakörét, amiben egy lehetséges megközelítés az agilis projektmenedzsment. Wysocky felismerte azt, hogy bár definíció szerint a projekt egyedi tevékenység, ugyanakkor

lényeges különbség van az egyes projekttípusok között azok egyedisége, vagy ha tetszik, kockázatossága között. Egy tipizált családi ház századik felépítése lényegesen eltér egy innovatív, termékfejlesztési projekttől. Wysocky az ilyen értelemben vett különbségek kezelését a „rugalmasság" és az „adaptív képesség" megfelelő adagolásában látja. Ebben a terminológiában akkor van szükség „agilis projektmenedzsmentre", amikor a cél világos, de annak elérésének módja nem az. Wysocky ilyen esetekre a projektéletciklus olyan szakaszolását javasolja, amely iteratív és inkrementális (Wysocky, 2009: p. 390.). Az agilis projektmenedzsment fogalmának bevezetése után öt szempontból vizsgálja annak a „tradicionális projektmenedzsmenttől" való eltérését, amire még visszatérünk.

Hasonló megközelítéssel tárgyalja a témát Chin, amikor (többek között) a projektek eltérő bizonytalanságával indokolja az „agilis projektmenedzsment" fogalmának bevezetését (Chin, 2004). Wysockyhoz hasonlóan Chin is vizsgálja az agilis megközelítés alkalmazhatóságának kérdését, azaz nála sem a mindent megoldó varázspálca szerepét kapja, hanem kontingenciaalapon közelíti meg a kérdést.

DeCarlo a bizonytalanság helyett a „nagy sebességű", gyorsan változó, magas komplexitású és kockázatú, stresszes projektekre javasolja a hagyományos projektmenedzsmenttől eltérő, „extrém projektmenedzsment" alkalmazását (DeCarlo, 2004). DeCarlo meglehetősen emelkedett szavakkal írja le az új megközelítés lényegét, amely holisztikus, az emberre fókuszál, a projektmenedzsmentjét az üzlet (az alaptevékenység) és a valóság vezérli. Az extrém projektmenedzsment öt kritikus sikertényezőjét azonosítja, melyek közül az egyik az „agilis szervezet", ahol a projekteket bonyolítják. DeCarlo világa érdekes, intuitív és elnagyolt.

Teljesen más irányból, a rendszerelmélet felől közelítette meg a témát Lyneis és Ford, akik a rendszerdinamika technikáinak alkalmazását vitték át a projektmenedzsment területére (Lyneis – Ford, 2007). A rendszerdinamika sajátos tárgyalásmódját követve, visszacsatolási rendszerek leírásával vizsgálták a projekt előrehaladásának és az átdolgozás szükségességének kapcsolatát (Ford már 1995-ös PhD-dolgozatában e témával foglalkozott). Tignor részlegesen áttekintette az agilis projektmenedzsment irodalmát, és összevetve azt Lyneis és Ford modelljével, arra a következtetésre jutott, hogy az *„kiállja az agilis projektmenedzsment tesztjét"*. Lyneis és Ford holisztikus modellje jól alkalmazható az agilis szemléletmóddal együtt (Tignor, 2009).

E rész befejezéseként visszatérünk Coplien és Harrison „nyelveire". A „projektmenedzsment" nyelvbe 26 minta tartozik, melyek túlnyomó része a nyelv nevének ellenére kifejezetten a szoftverfejlesztésre jellemzőek, pl. prototípus építése (*Build prototype*) vagy közvetett követelmények (*Implied requirements*). A további mintázatok viszont már a projektmenedzsment világába tartoznak, mint például az ütemezés ne legyen se túl laza, se túl feszes (*Size the schedule*), nem érdemes kicsit csúszni a határidőkkel (*Take no small slips*), a projekttagok maguk tervezzék meg rövid távú feladataikat (*Informal labour plan*). A „fokozatos (szervezeti) növekedés" nyelvben is számos olyan mintázatot találunk, melyek tartalmilag fellelhetők a projektmenedzsment-kézikönyvekben. Ilyen mintázat, hogy a projekt eredményeire váró felhasználót be kell vonni a munkálatokba (*Engage Customers*), a projektcsapat tagjai maguk választhassák meg, hogy kit fogadnak be a csapatba (*Self selecting teams*), mindenki legyen tisztában a projekt céljával (*Unity of purpose*), a csapatszellem és morál legyen erős (*Team pride*). Coplien és Harrison műve azonban nem csodafegyver, azt, hogy az adott mintázatot miképpen lehet kialakítani és megerősíteni, nem írják le. Az egyes mintázatok kiválasztását indokolják ugyan, de a mintázatok teljességét sem vizsgálják (Coplien – Harrison, 2006).

Az agilitás a vezetéstudomány szemszögéből

Madártávlatban áttekintve, az „agilis megközelítés" kifejezés égisze alatt végzett vezetési tevékenységeket, azonosítható néhány olyan meghatározó alapelv, melyek használata, mint látni fogjuk, egyáltalán nem új gondolat.

Egy terv legyen hihető (reális)

Az agilis tervezés sajátja az iteratív és inkrementális megközelítés. Ezt az elvet párosítva azzal a felismeréssel, hogy az innovatív (kockázatosabb, bizonytalanabb) feladatok (projektek) esetében elvileg nem lehet hosszabb távra tervezni, adódik a viszonylag rövid (hónapos) tervezési ciklus szükségessége.

Úgy is fogalmazhatunk, hogy csak olyan időtávra érdemes részletesen tervezni, melyen várhatóan a tervtől való eltérés mértéke nem lesz elfogadhatatlanul nagy. Ez a tervezési alapelv azonban a klasszikus projektmenedzsment eszköztárából származik! Néhány idézet ezen állítás alátámasztására:

- A PRINCE2 előírása szerint: *„Planning can only be done to a level of detail that is manageable and foreseeable."* (OGC, 2009: p. 13.)
- A PMBOK szerint: *„Predictive project life cycles… are the ones in which the project scope, the time and the cost required to deliver that scope, are determined as early in the project life cycle*

as practically possible." (PMBOK, 2013: p. 43.) *„Adaptive life cycles (known as change-driven or agile methods) are intended to respond high levels of change and ongoing stakeholder involvement… Adaptive methods are preferred when dealing with a rapidly changing environment, when requirements and scope are difficult to define in advance, and when it is possible to small incremental improvements that will deliver value to the stakeholders."* (PMBOK, 2013: p. 46.)

- *„Ha megkérdezzük az embereket, hogy mitől sikeres egy projekt, akkor legtöbben azt válaszolják, hogy a „reális tervezés miatt."* (Verzuh, 2006: p. 149.)

Ami újszerűnek tekinthető az agilis világban, az a tervezés mindenhatóságába vetett hit megkérdőjelezése. Gyakran egy terv puszta léte elhiteti a vezetéssel, hogy az abban foglaltak megvalósíthatók, ez most megkérdőjeleződik.

Van még egy nagy előnye az iteratív és inkrementális megközelítésnek, az, hogy megfelelő részcélok kiválasztása esetén gyorsan eredményeket lehet felmutatni (ez az ún. „quick win"). Ez szintén az agilis szoftverfejlesztés egyik jellegzetessége, ahol elvárják azt, hogy minél hamarabb „értéket" kapjon a felhasználó. A „quick win" alkalmazása azonkívül, hogy növeli a felhasználó bizalmát a fejlesztői csapatban, abban is segít, hogy a fejlesztői csoportból fejlesztői csapat legyen, ahol magas a csoportkohézió (Tudor – Trumble, 1996).

Természetesen számos olyan feladat van, ahol nincs sem szükség, sem lehetőség iteratív és inkrementális megvalósításra. Felettébb furcsa lenne, ha egy repülőgép gyártásakor egyes alkatrészeket újra és újra elkészítenének, vagy egy csatorna lefektetése során egyes vezetékszakaszokat átterveznének és újraásnának a munkálatok során. A fenti klasszikus megközelítés mögött rejlő kontingenciafelfogás szélesebb körű és rugalmasabb alkalmazhatóságot ad, mint az agilis megközelítés dogmatikus alkalmazása.

Az önszerveződés javíthatja a csoport teljesítményét

Az agilis munkavégzés további sajátossága, hogy a munkatársak maguk vállalkoznak egyes feladatok elvégzésére. Ez sem nevezhető új ötletnek, hiszen az önszerveződő csoportok előnyei már régóra ismeretesek (Cummings, 1978; Cohen, 1996). Cummings szerint az önszerveződő csoportok jellegzetessége az, hogy a csoport tagjai maguk döntik el, hogy mely munkafázissal foglalkoznak, az egyes tagok képesek többféle munka ellátására, és maguk határozzák meg a jutalmazás és munkakövetés módját (Cummings, 1978). Az önszerveződő csoportokban dolgozók csoportkohéziója erősebb, és így termelékenysége gyakran meghaladja a szokványos munkaszervezéssel vezetett csoportokét (Campbell – Martens, 2009). Nem meglepő tehát, hogy az agilis fejlesztési módszerek eredményesebbnek és hatékonyabbnak bizonyulnak, amennyiben az elvégzendő feladat természete lehetővé teszi annak önszerveződő csoport általi elvégzését.

Nyilvánvalóan a csoport tagjainak száma és személyisége is befolyásolja azt, hogy ez a fajta vezetési módszer alkalmazható-e. A páronkénti kommunikációs csatornák száma négyzetesen nő a csoport tagjainak számával, ezért 8-10 főnél nagyobb létszámú csoportoknál eleve gondokba ütközik a kommunikáció. A csoport tagjainak képesnek kell lenniük arra, hogy elvégezzék a vállalt feladatot, és elég ambiciózusnak ahhoz, hogy kellően kezdeményezőek legyenek. Ezek a tulajdonságok persze valamennyire idővel kialakíthatók, de egy ösztönös antipátia vagy rivalizálás az egész csoport teljesítményét sutba vághatja.

Az agilis munkaszervezés önállóságra és kezdeményezésre építő sajátossága tehát szintén már régóta ismert technika. A szoftverfejlesztés világának egyes részei, különösen a webes fejlesztés, kifejezetten alkalmas terep a technika használatára.

A feladatok elvégzésének követése legyen transzparens, és alapuljon tényeken

Mint korábban láttuk, a Scrum módszer előírja a *„daily scrum"* alkalmazását, ami tulajdonképpen az előrehaladás kontrollja. A csapat minden tagja láthatja, hogy a többiek mit végeztek egy nap alatt. Az átláthatóságot segíti az is, hogy a csapat tagjai egy helyiségben dolgoznak, az előrehaladást és a hátralevő feladatokat lehetőleg grafikusan, mindenki által jól látható helyen teszik közzé.

Ez a technika nem más, mint az ISO 9000 egyik alapelvének, a tényeken alapuló döntéshozatalnak a következetes alkalmazása (MSZ EN ISO 9000:2000). Nincs, és nem is lehet mellébeszélés, a tények magukért beszélnek. Nincs kizárólag a feladatok követésével foglalkozó munkatárs, mindenkinek egyszerre kell dolgoznia és a nyomon követéssel foglalkoznia. A napi visszacsatolás, illetve a *sprint review* nagyobb lélegzetű áttekintése elégséges. Nem véletlen, hogy VersionOne felmérése szerint a *daily scrum* a leginkább használt agilis technika (VersionOne, 2012).

Bármennyire tetszetős e visszacsatolási technika, alkalmazásának megvannak a maga korlátai. Ha a csapat sok tagból áll, ha a feladat dekomponálása egynapos részekre nem nyilvánvaló, vagy akár nem is lehetséges, akkor bizony más, ortodox kontrolltechnikák alkalmazására lesz szükség.

Az agilis fejlesztések további jellegzetessége a vizualizáció, ami legtöbbször egy nyomon követést szolgáló tábla alkalmazását jelenti. E módszer – feltéve, hogy a táblán a valósággal egyező adatok szerepelnek, egyfajta demokratikus hozzáállást tükröz: mindenki számára egyértelmű, hogy ki, mit végzett el, és mi lesz a következő lépés. Tulajdonképpen ez a projekt követésének olyan eszköze, ami éppen a nyilvánossága miatt egyfajta kényszert is jelent a projekttagok számára, ugyanis nemcsak a vezető látja az elvégzett munkát, hanem a csoport összes tagja azonos képet kap.

Az agilis szemléletben a csapattagok fizikailag egymáshoz közel tartózkodnak, és rendszeresen kommunikálnak. Ez javítja a csoportszintű bizalmat, ami végső soron a transzparencia kialakulását segíti elő (Williams, 2005). A személyes, szemközti kommunikáció mindig alaposabb megismerést tesz lehetővé a kommunikáló felek számára, és elősegíti a bizalmi légkör kialakulását.

Az érintettekkel (a felhasználóval) legyen megfelelő a kapcsolattartás

Az agilis fejlesztés sajátossága, hogy szoros kapcsolatot tételez fel a szoftver felhasználója és fejlesztői között. Másképp fogalmazva, csak akkor lehet eredményes a fejlesztés, ha a kommunikáció az érintettek között megfelelő mélységű.

Az érintett fogalma, és a velük való kommunikáció kiemelt fontossága megjelenik többek között a vállalatgazdaságtanban (Chikán, 199: p. 23.), illetve a stratégiai menedzsment területén (Johnson – Scholes, 1993: p. 171–177.). Az érintettek megfelelő kezelése klasszikus terület a projektmenedzsment irodalmában is. Az érintett fogalma már a terület bibliájának számító, *„A Guide to the Project Management Book of Knowledge”* (PMBOK) 1996-os, első kiadásában is központi fogalomnak számított. A PMBOK 2012-es, ötödik kiadásában pedig önálló tudásterületként nevesítették az érintettek menedzsmentjét (PMI, 2012). A már említett PRINCE2 világában hasonlóképpen fontos szerepet kapnak az érintettek, akiket azonosítani kell, illetve biztosítani kell a szükség szerinti bevonásukat a projektbe (OGC, 2009: p. 41–42.).

A PMBOK megközelítése természetesen általánosabb, de bizonyos értelemben teljesebb, mint az agilis szoftverfejlesztésé: azonosítani kell a különböző igényű érintetteket, majd a sajátos igényeik szerint kell a velük történő kapcsolattartást megtervezni, majd kivitelezni. Az érintettekkel történő kapcsolattartás menedzselése és követése szintén önálló folyamat lett (PMI, 2012: p. 391–416.). Az agilis szoftverfejlesztés esetében egy mindig megjelenő, konkrét érintett a felhasználó, azaz úgy is tekinthetjük, hogy az agilis projektmenedzsment a PMBOK általános modelljét konkretizálja.

Összességében tehát az agilis szemlélet érintettekre fókuszáló sajátossága sem tekinthető új megközelítésnek, legfeljebb az adott környezetre történő alkalmazásnak.

Az önreflexió elősegíti a kéthurkos tanulást

Az agilis szemlélet jellegzetessége a gyakori visszacsatolás, azaz előírja az önreflexiót. A Scrum sprintjei végén előírt retrospektív áttekintés célja a munkavégzés módszerének javítása, azaz a kéthurkos tanulás (Argyris, 1977). Az önreflexió folyamatának tudatos és célirányos kézbentartása pedig gyorsabbá teheti a tanulási folyamatot (Daudelin, 1997). A tanulás végső soron javítja a csapat (szervezet) eredményességét és hatékonyságát.

A tapasztalatok kiértékelése szintén régóta része a projektmenedzsment technikai repertoárjának. A PMBOK szakaszzáráskor előírja a tanulságok (*lessons learned*) dokumentálását (PMI, 2012), a PRINCE2 hasonlóan jár el, csak itt más a megnevezés (*lessons log*, lásd OGC, 2009). Az agilis szemléletben ezek gyakorisága nagyobb a megszokottnál. Itt valójában megint annak felismerése köszön vissza, hogy bizonyos feladatok megoldása során nem tudunk elég pontosan tervezni, ezért szükséges a gyakori és hiteles visszacsatolás.

Az agilis szemlélet jövője

Az agilitás divatos lett az elmúlt húsz évben. Sokan használják e kifejezést, eltérő kontextusban és eltérő értelmezéssel. Láthattuk, hogy az agilitás számos gondolata a vezetés-szervezés körébe tartozik, és e kifejezés gyakran egyfajta csodavárással társul. A menedzsment irodalmában azonban sajnos már több paradigmaváltónak kikiáltott gondolat – például a *Business Process Reengineering,* a *learning organization* vagy a *knowledge management* – rövid tündöklés után egy szűkebb szakmai közösség diskurzusának tárgya maradt. Erre, a csodavárás után törvényszerűen bekövetkező csalódás jelenségére szokás a *„management fad”* kifejezést használni. E „csodaszerek”, mint előbb-utóbb kiderül, nem feltétlenül járnak egyértelműen kimutatható eredménnyel, de ezzel együtt is gyakran hasznosnak tartják alkalmazásukat (Gibson – Tesone, 2001, illetve Linden – Fenn, 2003). Természetesen ebből következik a kérdés, hogy az agilitás tündöklése után vajon a feledés homályába merül, vagy értékállónak bizonyul-e?

Az agilis gyártás témája esetében láttuk, hogy a kezdeti fellángolás után mára az agilitás fenntartásával, illetve e sokrétű fogalom letisztulása után az elvárható eredmények empirikus vizsgálatával foglalkoznak.

Az agilis szoftverfejlesztés esetében úgy tűnik, az agilis megközelítés az eredményessége miatt ugyancsak a kötelező szakmai repertoár részévé vált, és a hagyományos megközelítéssel szimbiózisban él majd tovább (Baskerville et al., 2011). Egy egyszerű teszt alátámasztja, hogy e terület mára az oktatás részévé vált: a Google kereső *„agile software development course"* kereső kifejezésre több mint 90 000 találatot ad (2013. szeptember 1-jén). Az alkalmas megközelítés kiválasztása a feladat természetétől a fejlesztést megrendelő szervezet kultúrájától, és természetesen a fejlesztést végző szervezet kultúrájától függ.

Az általában vett vezetés-szervezés területén már nem ennyire egyértelmű a helyzet. Az agilis projektmenedzsment kifejezést, annak ellenére, hogy a bemutatott irodalom a projektmenedzsment önálló területeként kezeli, egyelőre még elsődlegesen a szoftverfejlesztéshez kapcsolják (Fernandez – Fernandez, 2009). További kérdés, hogy más alkalmazási területeken, például az építőiparban, van-e létjogosultsága e fogalom használatának? Volt erre kísérlet (Owen – Koskela, 2006), de nem tűnik átütő erejűnek, ami azért nem meglepő, mert az építőipari projektek innovatív jellege alacsonyabb szintű, mint a zöldmezős szoftverfejlesztésé.

A gyártás területén és a szoftverfejlesztésben viszont az agilis szemlélet bizonyított. A cikk elején is hangsúlyozott „szemlélet" szó használata fontos ehelyütt; Cockburn szerint. *„I keep telling people that agile is mostly an attitude, not a methodology or fixed set of practices"* (idézi Fernandez – Fernandez, 2009: p. 16.). Az agilitás általános céljai: az eredményesség elérése, a termelékenység és termék minőségének javítása, az ügyfél elégedettségének növelése, a termelési költségek leszorítása mind-mind örökzöld témák (Fernandez – Fernandez, 2009). Az agilitás belátható ideig a gyakorlati és az akadémiai diskurzus része marad.

Lábjegyzet

[1] A tanulmány megírásához nélkülözhetetlen segítséget adtak a Kulcsár Bencével, Kupás Tiborral, Tornai Balázzsal, Prónay Gáborral és Véry Zoltánnal folytatott beszélgetések.

Felhasznált irodalom

Anderson, D.J. (2010): Kanban. Blue Hole Press

Argyris, C. (1977): Double-Loop Learning in Organizations. Harvard Business Review, Sep/Oct 77, Vol. 55, Issue 5: p. 115–125.

Baskerville et al. (2011): Post-agility: What follows a decade of agility? Information and Software Technology, Volume 53, Issue 5, May 2011: p. 543–555.

Beck, K. (1999): Extreme Programming Explained. Embrace Change. Kluwer Academic Publishers

Beck, K. et al. (2001): Kiáltvány az agilis szoftverfejlesztésért. Lásd http://agilemanifesto.org/iso/hu/ Letöltve: 2013. 08. 23-án

Beck, K. (2002): Test Driven Development. Upper Saddle River: Addison Wesley

Beck, K. (2004): Agile Project Management with Scrum. Microsoft Press

Beck, K. (2005): Extreme Programming Explained. Harlow: Pearson Education

Boehm, B.W. (1986): A spiral model of software development and enhancement. in: Jack C. Wileden - Mark Dowson (eds.): Proceedings of an International Workshop on the Software Process and Software Environments, March 1985, Coto de Caza, Trabuco Canyon, California, USA. ACM SIGSOFT Software Engineering Notes 11(4), August 1986

Calvo, R. et al. (2008): Systemic criterion of sustainability in agile manufacturing. International Journal of Production Research, Vol. 46, No. 12: p. 3345–3358.

Casey-Campbell, M. – Martens, M.L. (2009): Sticking it all together: A critical assessment of the group cohesion–performance literature. International Journal of Management Review, Vol. 11, Issue 2: p. 223–246.

Chikán A. (1995): Vállalatgazdaságtan. Budapest: Közgazdasági és Jogi Könyvkiadó – Aula

Chin, G. (2004): Agile Project Management: How to Succeed in the Face of Changing Project Requirements. AMACOM

Cobb, C.G. (2011): Making Sense of Agile Project Management: Balancing Control and Agility. Chichester: John Wiley & Sons

Cockburn, A. (2004): Crystal Clear: A Human-Powered Methodology for Small Teams. Harlow: Pearson Education

Coplien, J.O. – Harrison, N.B. (2006): Organizational Patterns of Agile Software Development. Upper Saddle River: Prentice Hall

Corona, E. – Pani, F.E. (2013): A Review of Lean-Kanban Approaches in the Software Development. WSEAS Transactions on Information Science and Applications, Volume 10, Issue 1: p. 1–12.

Cummings, T.G. (1978): Self-Regulating Work Groups: A Socio-Technical Synthesis. The Academy of Management Review, Vol. 3, No. 3: p. 625–634.

Cohen, E.G. et al. (1996): A Predictive Model of Self-Managing Work Team Effectiveness. Human Relations, Vol 49, No 5: p. 643–676.

Daudelin, M.W. (1997): Learning from experience through reflection. Organizational Dynamics, Volume 24, Issue 3, Winter 1996: p. 36–48.

DeCarlo, D. (2004): Extreme Project Management: Using Leadership, Principles, and Tools to Deliver Value in the Face of Volatility. San Fransisco: Jossey-Bass

Fernandez, D.J. – Fernandez, J.D. (2009): Project Management – Agilism versus Traditional Approaches. The Journal of Computer Information Systems, Winter 2008/2009: p. 10–17.

Fister, S.G. (2012): Organizational Agility. PM Network, October 2012, Vol. 26, No. 10: p. 54–60.

Gibson, J.W. – Tesone, D.V. (2001): Management fads: Emergence, evolution, and implications for managers. Academy of Managemeni Executive, Vol. 15, No. 4: p. 122–133.

Goodpasture, J.C. (2010): Project Management the Agile Way. J. Ross Publishing

Hass, K.B. (2007): The Blending of traditional and Agile project Management. PM World Today, May 2007. Vol. IX, Issue V.

Hinsemkampf, A. (2007): Agilis projektmenedzsment (előadás). PMI Budapest Chapter rendezvénye, 2007. 10. 17.

Highsmith, J.A. (1999): Adaptive Software Development: A Collaborative Approach to Managing Complex Systems. Dorset House Publishing

Highsmith, J.A. (2004): Agile project management. Upper Saddle River: Addison-Wesley

Highsmith, J.A. (2010): Agile Project Management: Creating Innovative Products (Agile Software Development). Upper Saddle River: Addison-Wesley

Inman, R.A. et al. (2011): Agile manufacturing: Relation to JIT, operational performance and firm performance. Journal of Operations Management, 29 (2011): p. 343–355.

Jackson, M.B. (2012): Agile: A Decade in; Jackson, PM Network, April 2012, Vol. 26, No. 4: p. 60–63.

Jacobson, I. – Booch, G. – Rumbaugh, J. (1999): Unified System Development Process. Harlow: Pearson Education

Johson, G. – Scholes, K. (1993): Exploring Corporate Strategy. Text and Cases, 3rd Edition. London: Prentice-Hall

Kniberg, H. – Skarin, M. (2010): Kanban és Scrum: mindkettőből a legjobbat. C4Media. Letölthető: http://www.adaptiveconsulting.hu/sites/default/files/KanbanEsScrum_MindkettobolALegjobbat_1.pdf (letöltve: 2013. 08. 23.)

Kosztyán, Zs.T.– Kiss, J. (2011): Matrixalapú projekttervezési módszerek. Vezetéstudomány, XLII. évf. 10. szám: p. 28–42.

Linden, A. – Fenn, J. (2003): Understanding Gartner's Hype Cycles. Gartner Research R-20-1971. Letölthető http://www.ask-force.org/web/Discourse/Linden-HypeCycle-2003.pdf (letöltve: 2013. 09. 04.)

Lyneis, J.M. – Ford, D.N. (2007): System dynamics applied to project management. System Dynamics Review, Volume 23, Number 2/3, Summer/Fall 2007: p. 157–189.

Mason-Jones, R. – Naylor, B. – Towill, D.R. (2000): Lean, agile or leagile? Matching your supply chain to the marketplace. International Journal of Production Research, 11/20/2000, Vol. 38, Issue 17: p. 4061–4070.

Nagel, R. (1992): 21st Century Manufacturing Enterprise Strategy. Iacocca Institute, Lehigh University. http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA257032 (letöltve: 2013. 08. 23.)

Naur, P. – Randell, B. (1969): Software Engineering: Report on a Conference sponsored by the NATO Science Committee. Garmisch, Germany, 7th to 11th October 1968, Brussels, Scientific Affairs Division, NATO, January 1969. Letölthető: http://homepages.cs.ncl.ac.uk/brian.randell/NATO/nato1968.PDF (letöltve: 2013. 08. 23.)

Naylor, J.B. – Naima, M.M. – Berry, D. (1999): Leagility: Integrating the lean and agile manufacturing paradigms in the total supply chain. International Journal of Production Economics, Volume 62, Issues 1–2: p. 107–118.

Nonaka, I. – Takeuchi, J. (1986): The new new product development game. Harvard Business Review, Vol. 64, Issue 1: p. 137–146.

OGC (2009): Managing Successful Projects with PRINCE2: 2009 Edition. The Stationery Office

Owen, R.L. – Koskela, L. (2006): An Agile Step Forward In Project Management. in: Proceedings of the 2nd Specialty Conference on Leadership and Management in Construction: p. 216–224.

Palmer, S.R. – Felsing, J.M. (2002): A Practical Guide to Feature-Driven Development. Upper Saddle River: Prentice Hall

Parnas, D.L. (1971): On the criteria to be used in decomposing systems. Carnegie Mellon University, Computer Science Department. Paper 1980

PMI (2009): Agilis vagy Klasszikus projektmenedzsment? Körkapcsolás 12., 2009. 11. 05, lásd http://www.pmi.hu/pmi/rendezvenyek/korkapcsolas-12 (letöltve: 2013. 08. 23.)

PMI (2012): A Guide to the Project Management Book of Knowledge, 5th ed. Pennsylvania: PMI Inc.

PMSZ (2011): Agilis módszertanok az IT-n kívül. http://www.pmi.hu/pmi/rendezvenyek/projektmenedzsment-szakmai-teadelutan-10 (letöltve: 2013. 08. 23.)

Prónay, G. (2011): Extrém/agilis projektmenedzsment. lásd http://blog.mfor.hu/projekt/6243.html (letöltve: 2013. 08. 23.)

Sanchez, L.M. – Nagi, R. (2001): A review of agile manufacturing systems. International Journal of Production Research, Volume 39, Issue 16: p. 3561–3600.

Schwaber, K. – Beedle, M. (2002): Agile Software Development with Scrum. Harlow: Pearson Education International

Sharifi, H. – Zhang, Z. (1999): A methodology for achieving agility in manufacturing organisations: an introduction. International Journal Production Economics, 62 (1999): p. 7–22.

Stapleton, J. (1997): Dynamic Systems Development Method. The Method in Practice. Upper Saddle River: Addison-Wesley Longman

Sutherland, J. – Schwaber, K. (2011): A Scrum Útmutató. Lásd: http://www.scrum.org/Portals/0/ScrumGuide – HU.pdf (letöltve: 2013. 08. 23.)

The Standish Group (2010): CHAOS Summary for 2010

Tignor, W.W. (2009): Agile Project Management. International Conference of the System Dynamics Society. Albuquerque, NM 2009, July 26 – July 30, 2009

Tudor, T.R. – Trumble, R.R. (1996): Work-teams: Why do they often fail? S.A.M. Advanced Management Journal, Vol. 61, Issue 4: p. 31–38.

VersionOne (2013): 7th Annual State Of Agile Development Survey. http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf (letöltve: 2013. 08. 23.)

Verzuh, E. (2006): Projektmenedzsment. Budapest: HVG Kiadó

Williams, C.C. (2005): Trust Diffusion: The Effect of Interpersonal Trust on Structure, Function, and Organizational Transparency. Business & Society, Vol. 44, No. 3: p. 357–368.

Womack, J.P. – Jones, D.T. (2009): Lean szemlélet. Budapest: HVG

Wysocki, R.K. (2009): Effective Project Management: Traditional, Agile, Extreme. 5th ed., London: Wiley

Zhang, Z. – Sharifi, H. (2000): A methodology for achieving agility in manufacturing organisations. International Journal of Operations & Production Management, Vol 20, Issue 4: p. 496–512.

Zhang, Z. (2011): Towards theory building in agile manufacturing strategies – Case studies of an agility taxonomy. International Journal Production Economics, 131 (211): p. 303–312.

Szerzőinknek

A Vezetéstudomány a Budapesti Corvinus Egyetem Gazdálkodástudományi Karának havi, referált folyóirata. A lapban a vezetési és gazdálkodási tudományterületekhez kapcsolódó témakörök elméleti és gyakorlati kérdéseit elemző és vizsgáló írások jelennek meg. A szerkesztőség *(sandor.kerekes@uni-corvinus.hu)* elektronikus formában kéri az írásokat.

A cikkeket elektronikus levélben *(MS Word fájl formátumban)* lehet a szerkesztőséghez eljuttatni. A Vezetéstudományban megjelent cikkek magyar és angol nyelvű összefoglalói elérhetőek a http://www.vezetestudomany.hu és a http://vezetestudomany.hu címeken.

A lap tudományos folyóirat, ezért szövegközi forráshivatkozások és ezek jegyzéke nélküli írásokat nem jelentet meg. A Vezetéstudományban megjelentetni szándékozott kéziratok szerzőitől az alábbi követelmények figyelembevételét kérjük:

- A cikkek szokásos terjedelme a hivatkozásokkal, ábrákkal és táblázatokkal együtt 20–24 oldal, 1,5-es sortávolsággal *(12-es betűméret, Times New Roman betűtípus)*.
- A cikkek első oldalának alján tüntessék fel a szerző foglalkozását, munkahelyét és beosztását, elektronikus levelezési címét, a tanulmány elkészítésével kapcsolatos információkat és az esetleges köszönetnyilvánításokat.
- A kézirathoz csatolandó egy magyar nyelvű és lehetőség szerint egy angol nyelvű rövid összefoglaló *(200 szót nem meghaladó terjedelemben)*, valamint a cikk fő témaköreit megnevező kulcsszavak jegyzéke.
- Kiemeléshez **félkövér** és *dőlt betű* használható, aláhúzás nem. Jegyzeteket lehetőleg ne használjanak, amennyiben azok feltétlenül szükségesek, szövegvégi jegyzetként adják meg.
- A táblázatoknak és ábráknak legyen sorszáma és címe, valamint – átvett forrás esetén – pontos hivatkozása.
- Az ábrákat és a táblázatokat a kézirat végén, külön oldalakon, sorszámmal és címmel ellátva kérjük csatolni, helyüket a szövegben egyértelműen jelölve (pl. „Kérem az 1. táblázatot kb. itt elhelyezni!").
- A szövegközi bibliográfiai hivatkozásokat zárójelben, a vezetéknév és az évszám feltüntetésével kérjük jelölni: pl. *(Veress, 1999)*; szó szerinti, idézőjeles hivatkozás esetén kiegészítve az oldal(ak) számával *(pl. Prahalad – Hamel, 1990: 85.)*.
- Amennyiben egy hivatkozott szerzőnek több bibliográfiai tétele van ugyanazon évben, ezeket 1999a, 1999b stb. módon kell megkülönböztetni.
- A felhasznált források cikk végén elhelyezett jegyzékét ábécérendben kérjük, a következő formában:
  1. *példa* (könyv): Porter, M.E. (1980): Competitive Strategy; New York: The Free Press
  2. *példa* (folyóiratcikk): Prahalad, C.K. – Hamel, G. (1990): The Core Competence of the Corporation; Harvard Business Review, május–június, 79–91. o.

A formai követelmények fentiekben érvényesített, ún. „Harvard" rendszeréről *(más néven „szerző/év" vagy „név/dátum" hivatkozási módszerről)* részletes tájékoztatást nyújtanak az alábbi WEB-címeken elérhető források:

http://education.exeter.ac.uk/dll/studyskills/harvard_referencing.htm

http://sydney.edu.au/library/subjects/downloads/citation/Harvard_Complete.pdf

Havi folyóirat lévén és a megjelenés átfutási idejének csökkentése érdekében a Vezetéstudomány kefelevonatot nem küld, elfogadás előtt azonban a szerzőknek egyeztetés céljából elküldi a cikk szerkesztett változatát.

2009. januártól a Vezetéstudományban publikált cikkek elérhetőek az ISI ISI Emerging Markets „www.securities.com" internetcímen található strukturált on-line információs adatbázisban. 2009 júniusától a Vezetéstudományban közölt írások elérhetőek az EBSCO Academic Search Complete adatbázisában a *http://web.ebscohost.com/ehost/search?vid=20&hid=102&sid=747a764f-362f-4683-9255-4e54f5ba0df7%40sessionmgr112* oldalon is.

2012. március 1-jétől a Vezetéstudomány egyes cikkei elérhetőek a http://unipub.lib.uni-corvinus.hu/500/ oldalon is.

Külön kívánságra 2004-ig visszamenőleg az összes korábbi kiadás publikációit elektronikus változatban is elküldjük. Ha a szerző nem járul hozzá cikkének eseti kérésre, elektronikus úton való továbbadásához, kérjük, előre közölje ezt.