HADMÉRNÖK

XIII. Évfolyam 1. szám – 2018. március

HÍRADÓ-INFORMATIKAI FEJLESZTÉST TÁMOGATÓ AGILIS DOKUMENTÁCIÓS MÓDSZEREK - 2. RÉSZ

AGILE DOCUMENTATION METHODS FOR COMMUNICATION AND INFORMATION SYSTEM DEVELOPMENT - 2. PART

GEREVICH János; NÉGYESI Imre

(ORCID: 0000-0001-7236-4514); (ORCID: 0000-0003-1144-1912)

gerevich.janos@agilexpert.hu; negyesi.imre@uni-nke.hu

Absztrakt

A nagy szervezetek életében egy új informatikai rendszer kialakítása és bevezetése mindig erőt próbáló kihívás. Azonban itt nem ér véget a megpróbáltatások sora, az alkalmazásba vett informatikai szolgáltatások továbbfejlesztése és integrációja egyaránt újabb kihívások elé állítja az alkalmazókat és szoftverfejlesztőket is. A Magyar Honvédség útmutatása ebben a témakörben az MH Összhaderőnemi Híradó és Informatikai Doktrínája. Ebben a cikkben olyan dokumentációs technikákkal találkozhatunk, melyek segítségével magas szintű követelményelemzést lehet végezni a rendszeresített és alkalmazásba vett informatikai szolgáltatások továbbfejlesztése érdekében.

Kulcsszavak: *követelményrendszer, követelményelemzés, agilis szoftverfejlesztés, továbbfejlesztés, adatmigrálás, interfész, dokumentációs módszer, Military Scrum*

Abstract

In the life of the large organizations the design process and the introduction of a new IT system is always a big challenge. However, this is not the end of the trials, the development and integration of the IT services that are being applied will bring the users and software developers to new challenges. The according guidance of the Hungarian Defense Forces on this topic is the Joint Doctrine for Informatics. This article contains documentation techniques that can be used to perform a high-level requirement analysis to improve the regularly applied IT services.

Keywords: *requirements, requirement analysis, agile software development, improvement, data migration, interface, documentation method, Military Scrum*

A kézirat benyújtásának dátuma (Date of the submission): 2017.11.23.
A kézirat elfogadásának dátuma (Date of the acceptance): 2018.03.02.

BEVEZETÉS

A Magyar Honvédség középtávú informatikai törekvéseit a 2014-es Informatikai Stratégia [1] határozza meg, ezen túlmenően a híradó-informatikai szolgáltatásokra vonatkozó folyamatokat, eljárásokat a hatályos Informatikai Szabályzat [2] írja le. A szabályzat 3-5 fejezetei foglalkoznak a híradó-informatikai szolgáltatásokkal, azok tervezésével, fejlesztésével és a már rendszeresített, illetve használatba vett szolgáltatások üzemeltetésével. Az említett dokumentumok általános követelményeket határoznak meg a béke és tábori rendszerek vonatkozásában, azonban létezik egy gyakorlati követelményekre nagyobb hangsúlyt fektető hatályos dokumentum is, a Magyar Honvédség Összhaderőnemi Híradó és Informatikai Doktrínája [3], a továbbiakban MH HID.

Az MH HID előszavából megtudhatjuk, hogy a dokumentum célja az, hogy „összefoglalja *a katonai híradás és informatika alapelveit, meghatározza a honvédelmi célú híradó és informatikai rendszerek tervezésével, szervezésével és alkalmazásával kapcsolatos elveket és követelményeket*.” [3; 7. o.] A dokumentum az alapelveken túl bemutatja a jelenlegi NATO szabványokat és azok alkalmazásának rendjét is. Jelen tanulmányban a szerzők az MH HID-ben található általános informatikai követelményrendszert vetítik rá a szoftverfejlesztésre, mint speciális informatikai tevékenységre. Ezt követően olyan agilis dokumentációs módszerek kerülnek bemutatásra, amelyek segítségével az MH HID-ben szereplő elvárások kielégíthetőek egyedi szoftverek követelményrendszerének összeállítása során, ezzel megfelelő minőségben előkészítve a megvalósítást – a szoftver implementálásának fázisát.

AZ MH HID SZOFTVERTECHNOLÓGIAI ELEMZÉSE

Az MH HID 1. fejezetében a híradó és informatikai rendszerek alapjai kerülnek bemutatásra, ezen belül az informatikai rendszerekkel szemben támasztott 16 általános alapelvvel találkozhatunk. [3; 11-14 o.] Az alábbiakban a szoftverfejlesztéshez szervesen kapcsolható 10 elv kerül bemutatásra. Az eredeti dokumentumban az egyes alapelvekhez tartozó értelmezések általánosak, ez annak tudható be, hogy híradásra és informatikára együttesen vonatkozó iránymutatásról van szó. A felsorolt pontokban az MH HID-ben fellelhető alapelvek szoftvertechnológia elemzése következik.

1. *Megbízhatóság* – általános informatikai követelmény, szoftverek esetében jó minőségű, hibamentes, folyamatos működésként írható le az adott informatikai rendszer rendeltetésszerű használata mellett.
2. *Szabványosság* – általános műszaki követelmény, mely értelmezhető COTS [4] rendszerekre, egyedi alkalmazásokra, szoftver modulokra és szoftver komponensekre egyaránt.
    a. *Kompatibilitás* – azonos technológiák, ezen belül azonos fordítási folyamatok (compilation process), összeépítési módszerek (build methods), tesztelési eszközök, futtatókörnyezetek, azonos szoftververziók felhasználását jelenti.
    b. *Interoperabilitás* – más szoftverekkel történő kétirányú együttműködés kialakításának képessége. Létező modulok, illetve komponensek beágyazhatósága.
3. *Rugalmasság* vagy manőverező képesség – megfelelő jogosultságok mellett ugyanazon képességek, funkciók különböző felhasználókkal történő alkalmazása, illetve ugyanazon funkciók különböző felületekről történő igénybe vétele a rendszer konzisztenciájának megőrzése mellett.
4. *Hitelesség* – általános informatikai követelmény, mely a rendszerben tárolt adatok megbízhatóságára vonatkozik.

a. Alap szinten – az informatikai rendszerben vezetett eseménynapló, mely segítségével kinyerhető a rendszerben tárolt adatok változásának története.
b. Felső szinten – valamilyen hitelesítési eszköz alkalmazása, elektronikus aláírás, illetve elektronikus bélyegző integrált alkalmazása az adatok hitelességének garantálása érdekében.

5. *Modularitás* – általános szoftvertechnológiai követelmény, mely önálló továbbfejlesztési lehetőséget jelent az egyes rendszerelemek, modulok, komponensek esetében. Valamint az egyes modulok cseréjének, verzióváltoztatásának lehetőségét is magában foglalja.
6. *Skálázhatóság* – ugyanazon szoftver vagy szoftverkomponens különböző hardveres környezetben történő alkalmazásának lehetősége.
    a. Alap szinten – minimális hardverkövetelmények melletti működés, illetve korlátozott funkcionalitású működés képessége.
    b. Felső szinten – szoftver kapacitásának növelhetősége a hardveres erőforrások növelésével. Például: Egy időben kiszolgálható felhasználók száma.
7. *Biztonság és védettség* informatikai rendszerek esetében – az informatikai rendszeren belüli megfelelő azonosítási és jogosultsági rendszer kialakítása. Szoftverekre vonatkoztatva az egyes funkciók elérése csak valamilyen azonosítási eljárást követően megfelelő jogosultságokkal lehetséges.
8. *Felhasználhatóság* – szoftvertechnológiai követelményként értelmezve az adott rendszer grafikus felületeire, más rendszerek számára biztosított interfészeire, illetve a szoftveren belül megvalósított folyamatokra vonatkozó követelmény.
9. *Információ-megosztás* – az egyes funkciók, elérhető információk szerepkörökhöz, jogosultságokhoz történő kapcsolása. Speciális jogosultságként kezelhető egy adott felhasználó helye a katonai hierarchiában. Például: a magasabb szinten elhelyezkedő felhasználók több funkciót érhetnek el vagy a megismerhető információ mennyisége növekedhet a magasabb vezetési szinteken.
10. *Adat-konzisztencia* – szoftvertechnológiai szemszögből azt jelenti, hogy egy adatot lehetőség szerint egy informatikai rendszeren belül csak egyszer tároljunk. Eme követelményre lehet válasz a *Domain Driven Design* [5], amikor egy adott felhasználási terület fogalmi rendszerét képezzük le a szoftver üzleti logikájába és adatbázis modelljére. Ezzel a technikával megfeleltethetőek a valós életben létező fogalmak és fogalmi-kapcsolatok a szoftverben megjelenő fogalmakkal és a közöttük lévő logikai kapcsolatokkal.

A fenti értelmezés az MH HID egy lehetséges kiterjesztése a katonai célú szoftverfejlesztésre vonatkozóan. Az imént felsorolt pontok és a hozzájuk tartozó értelmezések egyaránt felhasználhatóak béke és tábori célú egyedi alkalmazások fejlesztésekor. Kiemelt alkalmazási terület lehet a híradó-informatikai rendszerekkel szemben támasztott követelményhalmazok összeállítása, valamint az elkészült szoftverek műszaki és minőségi felülvizsgálata. A Híradó és Informatikai Doktrínában fellelhető, hogy *„a vezetési fölény kivívásának szükséges, de nem elégséges feltétele az információs fölény kivívása”* [3; 16. old], ennek eléréséhez már az informatikai rendszerek követelményeinek meghatározásakor a legkifinomultabb technikákat kell alkalmazni. Az imént azonosított 10 alapelv tekinthető a Hadmérnök on-line folyóirat 2017-ben megjelent 1. és 3. számában már bemutatott *Military Scrum* [6][7] szoftverekkel szemben támasztott általános műszaki követelményrendszerének.

Híradó-Informatikai rendszerek tervezése és fejlesztése

Az alábbiakban az MH HID-ben megjelenő informatikai fejlesztésekre [3; 1.3.5.] és szolgáltatásokra [3; 1.3.6.] vonatkozó követelmények kerülnek bemutatásra. A felülről építkező informatikai fejlesztést a következő módon írja le a dokumentum: *„a klasszikus rendszerfejlesztési módszerek alkalmazásával logikai szinten kialakítják a szervezet információs rendszerét, kidolgozzák az információfeldolgozási rendszer, majd az adatfeldolgozási rendszer tervét, meghatározzák a megvalósításhoz szükséges erőforrásokat, majd vagy elkészítik a programtervet és végrehajtják a programozást, vagy a piacról beszerzik a szükséges alkalmazói szoftvert, illetve megvásárolják a szükséges informatikai szolgáltatást."* [3; 18. o.]

Napjaink szoftvereinek előállítása egyre jobban eltér a fentiektől, egyre nagyobb teret nyernek az iteratív, agilis módszerek, ezzel cseng egybe a *Scrum* agilis szoftverfejlesztési módszertan térnyerése, amiről a Forbes magazin hasábjain [8] is olvashatunk. A cikkből kiderül, hogy a megkérdezett szoftverfejlesztő cégek 50%-ánál alkalmazták legalább egy fejlesztési projekten a *Scrum*-ot 2015-ben. Mindez azt is jelenti, hogy a program- és rendszertervek helyét strukturált követelményrendszerek, fejlesztési sprintekre vonatkozó feladatlisták, tesztforgatókönyvek és automatizált tesztek veszik át. Természetesen egy modern fejlesztési módszer alkalmazása során is szükség van a hagyományos értelemben vett rendszerszervezésre, csak ez a tevékenység az egyes iterációk részeként valósul meg. Egy ilyen lépés a sprinttervezés, amikor a termékre vonatkozó feladatlista alapján alacsony szintű fejlesztési feladatok keletkeznek, ekkor a fejlesztőcsapat és terméktulajdonos együtt vesznek részt a rendszerszervezésben. A másik rendszerszervezési lépés, maga az új funkció implementálása, amit megelőz a tesztesetek azonosítása, ekkor az adott szoftverfejlesztő és a terméktulajdonos együtt végeznek rendszerszervezési feladatot.

Ahhoz, hogy egy új szoftver kialakítása során vagy egy létező szoftver továbbfejlesztésekor a fejlesztést végző csapat sikeres tudjon lenni, megfelelő minőségű követelményeket kell támasztani az adott szoftverrel, illetve szoftver komponensekkel szemben megrendelői oldalon. Hasonló megállapítással találkozhatunk az MH HID-ben is: *„a szolgáltatások fejlesztésével kapcsolatos követelmények a honvédségi szervezetekre általánosan jellemző információfeldolgozási folyamatok elemzéséből kerülnek meghatározásra az alkalmazó szervezetek és a híradó-informatikai szakmai szervezetek együttműködése eredményeként…"* [3; 19. o.]. Az informatikai szabályzat ezt még bővebben taglalja, külön kitér arra, hogy a híradó-informatikai szakmai szervezetek feladata az együttműködés koordinálása az alkalmazó szervezetekkel. [2; 2.3.4.2.]

Szoftvertechnológiai megközelítésből az a kérdés merül fel, hogy az említett "*általánosan jellemző információfeldolgozási folyamatok"* elegendőek-e egy szoftverrel szemben támasztott hagyományos követelményspecifikáció vagy egy agilis követelményrendszer összeállításához valamilyen deklarált és lefektetett módszertan nélkül? A válasz valószínűleg az, hogy nem vagy csak alacsony minőségben. Egy teljesen új szoftverrel szemben támasztott követelményrendszer meghatározására, strukturálására a korábban már bemutatott *Military Scrum* ad egy alkalmazható dokumentációs technikát.

Amiről még nem esett szó az a rendszeresített, illetve használatba vett szoftverek továbbfejlesztésének, kiváltásának előkészítése. Ilyen esetekben a forrás szoftver műszaki hátterének, funkcionalitásának feltárása, értelmezése komoly feladat elé állíthatja a követelményrendszer összeállítóját. A követelmények kinyerésének miként je a szolgáltatásokra vonatkozó irányelvek bemutatását követően kerül tárgyalásra.

Híradó-Informatikai szolgáltatásokra vonatkozó irányelvek

Az MH kötelékében rendszeresített informatikai szolgáltatásokkal kapcsolatban az alábbi lényeges követelmény jelenik meg: *„a híradó és informatikai szolgáltatás célja a szervezetek eltérő feladatainak végrehajtása érdekében alkalmazható olyan egységes információs képesség biztosítása, amelynek alkalmazásával a feladatokat rövidebb idő alatt, kevesebb erőforrás bevonásával és jobb minőségben hajthatók végre, mint a szolgáltatás igénybe vétele nélkül”*. [3; 19. o.] Itt egy a szolgáltatásokra vonatkozó irányelv jelenik meg, mely az informatikai rendszerek integrált szolgáltatásként történő kialakítását vetíti elő. Ehhez azonban szükséges az egyes felhasználási területek fogalomrendszerének tisztázása és olyan informatikai rendszerek tervezése, melyek letisztultan és konzisztensen képezik le azt a fogalmi rendszert, amelynek a problémáját meg kell oldaniuk. Minderre azért van szükség, mert az egységes szolgáltatásokat a fogalmak számának növekedésével egyre bonyolultabb egyetlen szoftverben, illetve szoftverfejlesztési projektben kezelni. A fejlesztés sebessége túlságosan lelassulhat, a bonyolultság kezelhetetlenné válhat. Ilyen esetben célszerű a modularizáció, az alrendszerekre bontás, mellyel a bonyolultság az alrendszereken belül csökkenthető, az alrendszerek funkciója, felelőssége tisztázható. Új modulok, alrendszerek bevezetésekor, illetve cseréjük során felül kell vizsgálni az egyes rendszerek határait és szükség esetén újra kell gondolni azokat.

A fejlesztések során kompatibilis technológiákat kell kiválasztani, melyekkel az elkészült alrendszerek könnyedén integrálhatók és együttesen megfelelő minőségű szolgáltatást tudnak képezni. Egybe cseng ezzel a következő MH híradó-informatikai szolgáltatásokra vonatkozó követelmény is: *„biztosítsák a honvédségi szervezetek vezetési-irányítási, végrehajtási és együttműködési információs folyamatainak hatékony támogatását, tegyék lehetővé a szervezetek és feladatok széles körében felhasználható híradó és informatikai szolgáltatások biztosítását, képezzék alapját a honvédségi szervezetek integrált, egységes elvek és követelmények alapján kialakított információs rendszerének.”* [3; 1.4.1] Ez egy teljes Magyar Honvédség viszonylatában értelmezhető irányelv, ennek megvalósításához magas szintű követelmény meghatározásra is szükség van. A *Military Scrum* az újonnan fejlesztett szoftverek követelményrendszerének meghatározására már alkalmas, de még nem tért ki a szoftverfejlesztési projektek jelentős részére, melyek más jellegű feladatokat takarnak. Ezekben az esetekben egyéb dokumentációs módszereket is be kell vetni ahhoz, hogy megfelelő minőségű követelményrendszer álljon rendelkezésre a megvalósítás megkezdéséhez. Az alábbiakban a már említett továbbfejlesztési és integrációs feladatok kerülnek további részletezésre.

1. új műszaki alapokra történő helyezés – az informatika adta lehetőségek bővülésével és az igények növekedésével megeshet, hogy egy sok éven át használt rendszert le kell cserélni, mert a jelenlegi formájában már nem fenntartható.
    a. új szoftver fejlesztése adatmigrálás[1] nélkül – elképzelhető, hogy a régi rendszernek van egy kifutási ideje, nem kell a rendszerben felhalmozott adatvagyont új rendszerbe költöztetni; ebben az esetben a feladat az új rendszer követelményeinek meghatározása és az új szoftver

[1] adatmigrálás - áttérés egy alacsonyabb verziójú adatstruktúráról egy magasabb verziójúra vagy egy hasonló feladatot ellátó rendszer más szerkezetű adatstruktúrájára

implementálása, nagyon gyakori igény a régi rendszer funkcionalitásának megőrzése, annak jobbá tétele;

b. új szoftver fejlesztése adatmigrálással – itt az előző ponton felül a forrásrendszer adatainak átmozgatása a célrendszerbe is feladat;

2. rendszerek közötti integráció – két rendszer közötti kommunikáció megteremtése, folyamatok kialakítása az integrált rendszeren belül.

LÉTEZŐ SZOFTVEREK TOVÁBBFEJLESZTÉSE

Agilis technikával fejlesztett szoftverek esetében feltételezhető, hogy létezik egy termékre vonatkozó feladatlista, melyre adaptálható valamely agilis szoftverfejlesztési módszertan. Abban az esetben, ha nincs ilyen termékre vonatkozó követelményrendszer, akkor alkalmazható a *Military Scrum* követelményelemző módszere [7; 215-220. o.], itt meg kell jegyezni, hogy ez a módszer új funkciókhoz tartozó követelmények feltárását mutatja be. Egy már létező szoftver működésének feltárásához más eszközökre van szükség, máshonnan indul a követelményelemzési folyamat. Speciális módszer kell ahhoz, hogy ki tudjuk nyerni az újrahasznosítható információt egy kiváltandó rendszerből olyan formában, hogy azt majd fel is tudjuk használni egy új rendszer tervezése során. Az alábbiakban erre a problémára keressük a választ.

Alkalmazott szoftver cseréje új szoftver fejlesztésével, adatmigrálás nélkül

Gyakran tapasztalható az informatikában, hogy már korábban alkalmazásba vett rendszereket az idő elteltével lecserélnek, helyüket új szoftverekkel váltják ki. Ennek oka szerteágazó lehet, egyik eshetőség, hogy műszakilag elévül az alkalmazott rendszer és új technológiai alapokra kell helyezni. Az is elképzelhető, hogy újabb igények jelennek meg a szoftverrel kapcsolatban, amelyeket már nem lehet az eredeti technológiai környezetben kezelni. Alkalmazói oldalon ilyenkor komoly problémát jelenthet az új szoftverrel szemben támasztott követelményrendszer összeállítása. Milyen eszközökkel, illetve módszerekkel lehet kinyerni egy már használatba vett rendszer funkcióinak felsorolását, azok működésének leírását? A kinyert információk alapján milyen eszközökkel lehet a régi és az új követelményeket egy egységes követelményrendszerré formálni?

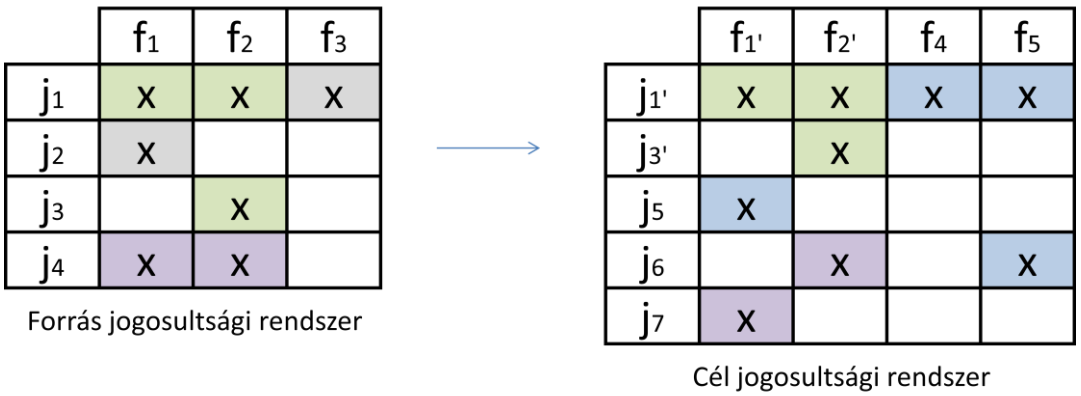
Vegyük kiindulási pontnak az agilis módszerek legalapvetőbb technikáját, hogy először a nagyobb követelmények kerülnek azonosításra, majd azokat bontják apróbb feladatokra és ezt követi a megvalósítás. Hasonló gondolatmenet alkalmazása elképzelhető egy már működő szoftver képességeinek feltérképezésére. A funkció-jogosultsági mátrix[2] már egy régóta ismert szoftvertechnológiai eszköz egy jogosultsági rendszerrel ellátott szoftver működésének leírására.

Az első lépés a használatban lévő rendszer funkcióinak összegyűjtése. Ez a mozzanat megtehető a forráskód elemzésével, valamint a felhasználói felület feltárásával. A legjobb eredmény elérése érdekében célszerű mindkét módszer együttes alkalmazása. Egyszerűbb rendszerek esetében egy lehetséges módszer lehet egy teljes jogosultsági körrel rendelkező tesztelő felhasználó alkalmazásával feltárni a kiváltandó rendszer funkcióit, egyfajta fekete doboz tesztelést végezni. Ha már rendelkezünk egy alapvető rendszerismerettel, akkor a jogosultsági rendszert az egyes jogosultságok elvételével indirekt módon lehet feltárni, jó

[2] Funkció-jogosultsági mátrix – egy olyan táblázat, melynek oszlopait egy adott informatikai rendszerben elérhető funkciók alkotják, sorai az adott rendszerben azonosított jogosultságokat reprezentálják. A táblázat egy cellája akkor van kitöltve, ha az adott oszlophoz tartozó funkciót el lehet érni az adott sorhoz tartozó jogosultsággal.

esetben egy addig elérhető funkció már nem lesz elérhető egy adott jogosultság elvételét követően. Ez a módszer erősen függ a szoftver jellegétől, a rendszerben megvalósított folyamatoktól, nem mindig alkalmazható.

Összetettebb rendszekerek esetén a rendelkezésre álló felhasználói és műszaki dokumentáció tanulmányozása is kiindulási alapja lehet a feltáró munkának. Amennyiben sokszereplős és összetett munkafolyamatok jelennek meg egy rendszeren belül, szükségszerű lehet tesztforgatókönyvek kidolgozása az egyes jogosultságok és funkciók kapcsolatának feltárásához. Segítség lehet még a forrásrendszer felhasználói és műszaki dokumentációjának tanulmányozása, valamint olyan tesztforgatókönyvek kialakítása, melyek végrehajtását követően a kiváltandó rendszer adatbázis modelljéből további információk nyerhetőek ki.



1. ábra Jogosultsági rendszerek közötti leképezés (saját szerkesztés)

A feltárt funkciók és a transzformált jogosultsági mátrix alapján előállítható az új szoftverrel szemben támasztott követelményrendszer, a felelősségi-körök meghatározhatóak, a *Military Scrum* előkészíthető. A függőségek feltárását is el kell végezni, ez lehet a fejlesztési sorrend kialakításának egyik alapja. Az új adminisztrációs felületek kialakítását célszerű már az elején elvégezni. Az 1. ábrán azt láthatjuk, hogy a forrásrendszer f_1, illetve f_2 funkciói leképezésre kerülnek a célrendszer $f_{1'}$ és $f_{2'}$ funkcióra, ugyanakkor az f_3 funkció megszűnik és új f_4, f_5 funkciók kerülnek kialakításra. A jogosultságok vonatkozásában azt lehet látni, hogy a j_1, j_3 jogosultságok leképezésre kerülnek a cél rendszerbe, míg a j_2 jogosultság megszűnik. Új jogosultság a j_5, míg a j_4 jogosultság két újonnan bevezetett jogosultsággá bomlik szét: j_6, j_7, melyekkel az $f_{1'}$ és $f_{2'}$ funkciók érhetőek el a célrendszerben. Egy valós projekt esetében is hasonló transzformációs szabályok figyelhetőek meg a forrás és célrendszerek között. Természetesen a különbségek mértéke határozza meg a szabályok mennyiségét és a feladat bonyolultságát.

Tegyük fel, hogy a cél rendszerrel szemben támasztott egyéb követelmények elemzése megtörtént. Az iménti példából és a feltételezett egyéb követelményekből kiindulva a célrendszerre vonatkozó feladatlista összeállítható. Az 1. táblázat néhány fiktív műszaki követelményt, egy kitalált jogosultsági rendszer kialakításának feladatait és az 1. ábrán bemutatott új funkcionális követelményeket tartalmazza.

A táblázatból a *Megjegyzés* és *Leírás* oszlopok helyén a *Jogosultságok* és a *Korábbi funkció* oszlopok kaptak helyet, eltérve ezzel a *Military Scrum* termékre vonatkozó feladatlistájától. A *Jogosultságok* oszlopban az új funkcióhoz kapcsolódó jogosultságok kerülnek felsorolásra. A *Korábbi funkció* oszlopban a forrásrendszerben azonosított korábbi funkciók és a hozzájuk tartozó jogosultságok szerepelnek. A forrásrendszer funkcióinak bevezetése a dokumentumban azért szükséges, hogy a fejlesztés során az újonnan fejlesztett funkció összevethető legyen az elődjével, a felmerülő kérdések egyszerűbben tisztázhatóak legyenek.

KhSsz.	Ssz.	Feladat	Prior.	Mod.	Állapot	St.pont	Jogosultságok	Korábbi funkció
MK-1	S-11	Vékony kliens architektúra kialakítása	100	-	Várakozik	40	-	-
MK-2	S-12	Relációs adatbázis kialakítása	90	-	Várakozik	40	-	-
FK-1	S-13	Szervezeti elemek adminisztrálása	80	-	Várakozik	20	-	-
FK-1	S-14	Felhasználók adminisztrálása	70	-	Várakozik	20	$j_{1'}$, $j_{3'}$, j_5, j_6, j_7	-
FK-2	S-15	$f_{1'}$ funkció kialakítása	60	-	Várakozik	13	$j_{1'}$, j_5, j_7	f_1 funkció j_1, j_2, j_4 jogosultságokkal
FK-3	S-16	$f_{2'}$ funkció kialakítása	60	-	Várakozik	8	$j_{1'}$, $j_{3'}$, j_6	f_2 funkció j_1, j_3, j_4 jogosultságokkal
FK-4	S-17	f_4 funkció kialakítása	50	-	Várakozik	8	$j_{1'}$	-
FK-4	S-18	f_5 funkció kialakítása	50	-	Várakozik	5	$j_{1'}$, j_6	-
.	.	.	.	.	.	.	.	.
FK-N	S-X	-	-	-	-	-	-	-

1. táblázat Termékre vonatkozó feladatlista létező rendszer feltárása alapján (saját szerkesztés)

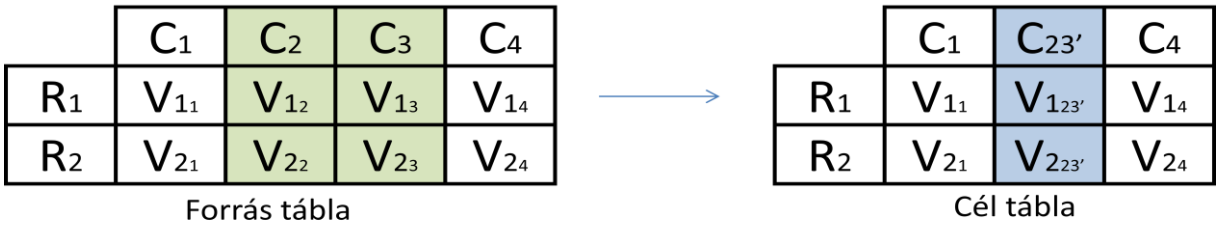
Az imént bemutatott módszerrel a szoftverek kiváltására vonatkozó követelményrendszer összeállítása magas színvonalon elvégezhető. Az új és az örökölt követelmények egy dokumentumban, strukturált formában kapnak helyet ezzel megalapozva az új szoftver elkészítésének megfelelő minőségű támogatását.

Adatmigrálás

Napjainkban az adatok tárolására leggyakrabban a relációs adatbázis-kezelő rendszereket (RDBMS) [9] használják a szoftverfejlesztők. A technológia több mint 40 éve jelent meg és mára széleskörűen elterjedt, jelenleg is gyakori választás az új rendszerek kialakításakor. A továbbiakban a relációs adatbázisok közötti adatmigrálás agilis menete kerül bemutatásra. A fogalmak újragondolása is célszerű, mert a módszer nem csak egy kiváltandó rendszer és egy új rendszer közötti adatmigrálást mutatja be, természetesen itt is alkalmazható, hanem egy forrás- és célrendszer közötti adatmegfeleltetés szabályainak leírására szolgál.

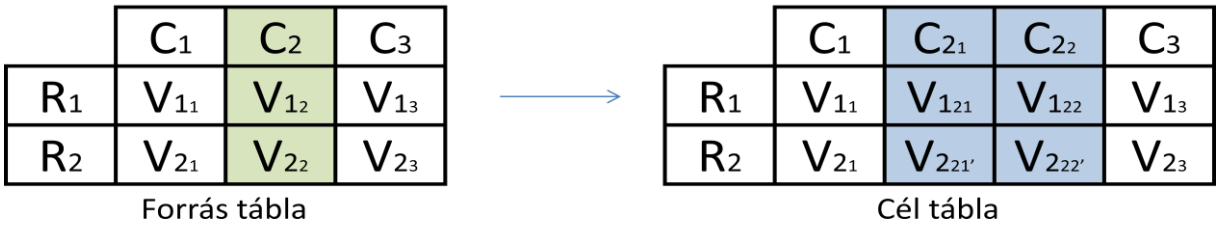
A relációs adatbázisok az adatok tárolását adattáblákkal, azokon belül különböző típusú oszlopokkal valósítják meg. Az egyes táblák közötti kapcsolatok leírására szolgálnak az idegen kulcsok. Kulcs lehet egyetlen oszlop vagy oszlopok halmaza, amelyekkel egyértelműen azonosítható egy sor egy adattáblában. Egy adatmigrálás során a problémát az okozza, hogy forrás és célrendszer adatstruktúrája, tárolt adatai merőben eltérhetnek egymástól.

Elképzelhető, hogy a forrás rendszer több oszlopot használ egy adott funkcióhoz tartozó adatok modellezésére. Tegyük fel, hogy adott egy *f* funkció és a működéséhez szükséges adatokat a forrás rendszer egy tetszőleges táblájában a C_2, C_3 oszlopok tartalmazzák. Mindemellett a cél rendszerben az *f* funkció *f'* leképezése a $C_{23'}$ oszlopot használja a működéséhez szükséges adatok tárolására.



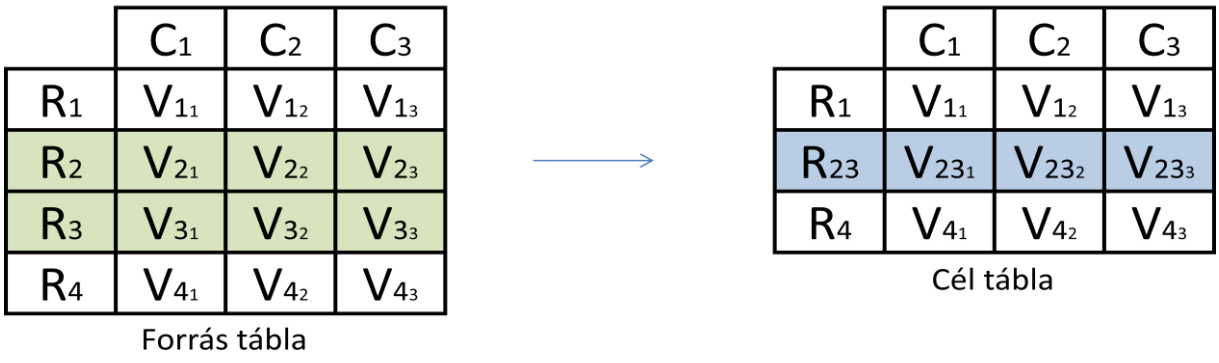
2. ábra Adatmigrálás során oszlopok összevonása (saját szerkesztés)

Az is lehetséges, hogy a forrásrendszer kevesebb oszlopot tartalmaz az adott funkció modellezéséhez. Tegyük fel, hogy az *f* funkció működéséhez szükséges adatokat a forrás rendszer egy tetszőleges táblájában a C_2 oszlop tartalmazza. Ehhez képest a cél rendszerben az *f* funkció *f'* leképezése a C_{21} és C_{22} oszlopokat használja a működése során.



3. ábra Adatmigrálás során oszlopok szétbontása (saját szerkesztés)

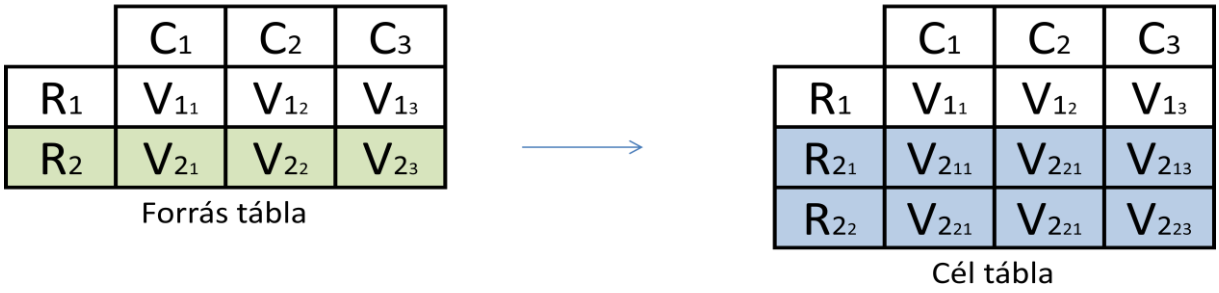
Egy adott funkció működése során nem csak a működéshez szükséges oszlopok száma lehet különböző, az is fennállhat, hogy a forrásrendszer több sorban tárolja le az adatokat, mint a célrendszer. Az is lehet, hogy a forrás adatbázisban a funkció működése több adatbázistáblát is érint. Az egyszerűség kedvéért most tételezzük fel, hogy ugyanazon tábla két sora egy funkció meghívásának végeredménye a forrásrendszerben, míg a célrendszerben az eredmény tárolására egyetlen sor szolgál. Tegyük fel, hogy az *f* funkció futásának végeredménye a forrásrendszerben az R_1, illetve R_2 sorok, míg az *f* funkció *f'* leképezésének meghívása a célrendszerben egy összevont R_{23} sorban tárolja ugyanazokat az adatokat más struktúrában más tartalommal.



4. ábra Adatmigrálás során adatsorok összevonása (saját szerkesztés)

Végezetül az is elképzelhető, hogy a forrásrendszer egy adott funkciójának működése során kevesebb sor keletkezik az adott adatbázis táblában, mint a cél rendszerben. Hasonlóan az előző példához az is lehet, hogy a cél adatbázisban a funkció működése több adatbázistáblát is érint. Az egyszerűség kedvéért az alábbi példában is ugyanazon tábla

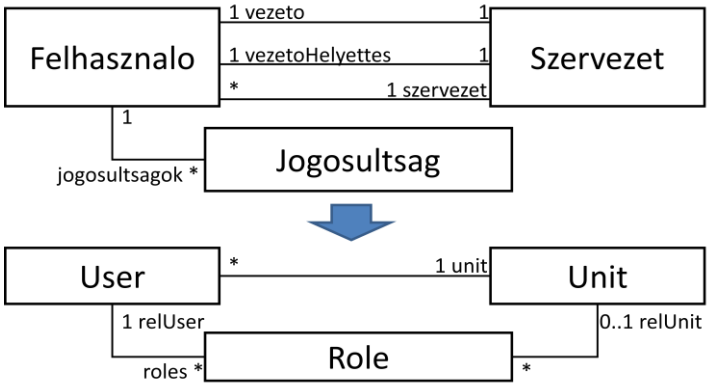
esetére térünk ki. Tegyük fel, hogy az *f* funkció futásának végeredménye a forrásrendszerben az R_2 sor, míg az *f* funkció *f'* leképezésének végrehajtása a célrendszerben R_{21} és R_{22} sorokban tárolja ugyanazokat az adatokat más struktúrában más tartalommal.



5. ábra Adatmigrálás során adatsor szétbontása (saját szerkesztés)

Az imént bemutatott esetek és ezek különböző kombinációi adják egy adatmigrálási projekt feladatrendszerét. Olyan rendszerek esetében, ahol funkciók százai érhetők el már érezhető a feladat bonyolultsága. A kérdés az, hogyan lehet strukturálni egy ilyen feladatrendszert, illetve milyen dokumentációs technika bevezetésével lehet egy adatmigrálási feladatot magas színvonalon támogatni.

A probléma szemléltetéséhez tegyük fel, hogy a forrás rendszerben 3 adatbázistábla modellezi a szervezeti elemeket, a bennük lévő felhasználókat, valamint a felhasználók jogosultságait. Minden felhasználónak pontosan egy szervezeti eleme van. Minden szervezeti elemnek pontosan egy vezetője és egy vezető helyettese van. Ezen felül a felhasználóknak lehetnek egyéb jogosultságai, például: adminisztrátor, ügyintéző. Tegyük fel, hogy a célrendszer máshogy modellezi a jogosultsági rendszerét. A szemléletesség kedvéért angol elnevezések kerültek bevezetésre a cél rendszerben. A *Felhasznalo* adattábla megfelelője a *User* tábla, a *Szervezet* tábla leképezése a *Unit* tábla. A célrendszerben *Role* tábla reprezentálja a jogosultságokat. Lényeges különbség a célrendszerben a forrásrendszerhez képest, hogy a vezető és a vezető helyettes kapcsolat egy-egy jogosultsággal van modellezve.



6. ábra Példa: forrás és cél jogosultsági rendszer bemutatása (saját szerkesztés)

Eme apró feladat megoldása során jól szemléltethető a *Military Scrum* erre a célra alkalmazható technikája. A módszer bemutatása előtt tegyük fel, hogy az adott forrásrendszer a jogosultsági rendszeren túl más fogalmakat is modellez, melyeket az egyszerűség kedvéért az *A, B, C...* táblákban tárolja. Az alábbi felsorolás a *Military Scrum* adatmigrálást támogató táblázatának oszlopait mutatja be, majd a 2. táblázat a fenti példa alapján keletkező feladatokat szemlélteti.

1. *Azonosító*: az adott táblát tartalmazó fogalomtér azonosítója
2. *Sorszám*: az adatbázistábla sorszáma
3. *Tábla*: az adatbázistábla neve
4. *Prioritás:* az adott feladat prioritása, egy pozitív egész szám, a termék-feladatlista rendezésének elvét ez az oszlop határozza meg [10; 103-104. o.]
5. *Állapot:* Az adott feladat megvalósításának állapota, mely lehet: Várakozik, Folyamatban, Kész, Törölt
6. *Storypont:* Az adott feladat megvalósítására becsült erőforrás fejlesztői napokban [10; 128-129. o.]
7. *Kapcsolatok feldolgozva?:* Igen/Nem – az adott tábla kapcsolataira vonatkozó transzformációs szabályok rendelkezésre állnak-e?
8. *Adatok feldolgozva?:* Igen/Nem – az adott tábla adataira vonatkozó transzformációs szabályok rendelkezésre állnak-e?
9. *Migrációs script kész?:* Igen/Nem – a tervek alapján elkészült-e a leképezéseket megvalósító migrációs script?

Azon.	**Ssz.**	**Tábla**	**Prior.**	**Állapot**	**St.pont**	**Kapcsolatok feldolgozva?**	**Adatok feldolgozva?**	**Migrációs script kész?**
Jog. R.	S-11	Felhasznalo	100	Kész	5	Igen	Igen	Igen
Jog. R.	S-12	Szervezet	90	Folyamatban	5	Igen	Igen	Nem
Jog. R.	S-13	Jogosultsag	80	Folyamatban	5	Igen	Nem	Nem
subd.-1	S-14	A tábla	70	Várakozik	3	Nem	Nem	Nem
subd.-1	S-15	B tábla	70	Várakozik	2	Nem	Nem	Nem
subd.-2	S-16	C tábla	60	Várakozik	3	Nem	Nem	Nem
subd.-2	S-17	D tábla	60	Várakozik	2	Nem	Nem	Nem
subd.-2	S-18	E tábla	60	Várakozik	5	Nem	Nem	Nem
.	.	.	.	.		.	.	

2. táblázat Adatmigrációs feladatlista adattáblák szintjén (saját szerkesztés)

A forrásrendszer fogalomterét célszerű további részekre bontani és így egyfajta altereket (angolul: *subdomain*) kialakítani. Minden altérhez lehetséges két újabb feladatlista kialakítása. Az első táblázat az altéren belüli és esetlegesen kifelé mutató kapcsolatokat gyűjti össze és rendszerezi az alábbiak szerint. Az egyes kapcsolatok feltárása során a megvalósítás sorrendje változhat. Ahhoz hogy egy kapcsolatot le tudjunk képezni szükséges mindkét tábla kialakítása és csak azt követően lehet beállítani az adott kapcsolatot egy idegen kulcs segítségével. Erre példa a 3. táblázatban az 1. lépéssel illusztrált *Felhasznalo* és *Szervezet* közötti kapcsolat leképezése a cél rendszerbe a *User, Unit* táblákra. A feladat elvégzéséhez azonnal két adatbázistáblával kell műveleteket végezni - az azonosítókat le kell képezni a cél rendszerbe, ezt követően lehet a kapcsolatokat átmozgatni.

Ssz.	**Lé-pés**	**Forrás tábla**	**Forrás oszlop**	**Kapcsolat**	**Cél tábla**	**Cél oszlop**	**Cél kapcs.**	**Eljárás szükséges?**	**Állapot**
S-11	1	Felhasznalo	Id	-	User	id	-	-	Kész
S-11	1	Felhasznalo	szervezet_id	Szervezet	User	unit_id	Unit	-	Kész
S-12	1	Szervezet	id	-	Unit	id	-	-	Kész
S-12	2-2	Szervezet	vezeto_id	Felhasznalo	Role	relUser_id	User	SP-1	Foly.-ban
S-12	2-2	Szervezet	vezeto_id	Felhasznalo	Role	relUnit_id	Unit	SP-1	Foly.-ban
S-12	2-3	Szervezet	helyettes_id	Felhasznalo	Role	relUser_id	User	SP-1	Foly.-ban
S-12	2-3	Szervezet	helyettes_id	Felhasznalo	Role	relUnit_id	Unit	SP-1	Foly.-ban
S-12	3	Szervezet	felettes_id	Szervezet	Unit	parent_id	Unit	-	Várakozik
S-13	2-1	Jogosultsag	id	-	Role	id	-	-	Kész
S-13	4	Jogosultsag	felhasznalo_id	Felhasznalo	Role	relUser_id	User	SP-2	Várakozik
.		.							

3. táblázat adattáblák közötti kapcsolatokra vonatkozó feladatlista (saját szerkesztés)

Az oszlopok tartalmának leképezése is különböző nehézségű feladat lehet. A legegyszerűbb feladat az oszlopnevek változása, ilyenkor nincs semmilyen transzformációs szabály. Az is előfordulhat, hogy valamilyen szabályrendszer alapján kell összevonni, illetve szétválasztani oszlopok tartalmát, ekkor olyan migrációs scriptekre van szükség, amelyek megvalósítják az adat-transzformációra vonatkozó szabályokat. Az is lehet, hogy valamilyen diszkrét értékkészlet van egy forrás oszlopban és a célrendszer máshogy modellezi ezeket az értékeket, ekkor hasonlóan valamilyen adatmigrációs szabályt kell implementálni. A következő táblázat ezt szemlélteti.

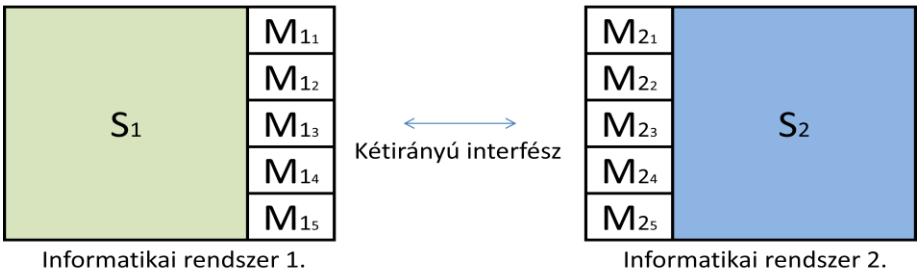
Ssz.	Lé-pés	Forrás tábla	Forrás oszlop	Értékkészlet	Cél tábla	Cél oszlop	Értékkészlet	Állapot
S-11	1	Felhasznalo	loginNev	-	User	login	-	Kész
S-11	1	Felhasznalo	nev	-	User	name	-	Kész
S-11	1	Felhasznalo	emailCim	-	User	email	-	Kész
S-12	2	Szervezet	nev	-	Unit	name	-	Foly.-ban
S-12	2	Szervezet	azonosito	-	Unit	shortName	-	Foly.-ban
S-12	2	Szervezet	tipus	-	Unit	type	-	Foly.-ban
S-12	2	Szervezet	emailCim	-	Unit	email	-	Foly.-ban
S-13	3	Jogosultsag	tipus	ADMIN, UGYINTEZO				Várakozik
.		.						
S-X		-						

4. táblázat adattáblák tartalmára (oszlopaira) vonatkozó feladatlista (saját szerkesztés)

A felvázolt módszer egy alapos eljárást ad a forrásrendszer feltárásához, valamint segítségével a fejlesztés menete is figyelemmel kísérhető, magas szinten támogatható. A bemutatott dokumentációs módszer megfelel a magas szintű elvárásoknak, így széleskörűen alkalmazható adatmigrálást igénylő feladatok során.

Rendszerek közötti integráció.

A létező rendszerek közötti magas szintű integráció kialakítása napjaink egyik leggyakoribb informatikai kihívása. Az alapszintű műszaki megoldást a problémára az interfész[3] tervezési minta alkalmazása jelenti. Az interfészek fejlesztése során ugyancsak alkalmazható a *Military Scrum*. A feladat mérete, az alkalmazott technológiák és a fejlesztést végző csapatok fizikai elhelyezkedése együttesen határozzák meg, hogy hány termékre – adott esetben interfészre – vonatkozó feladatlistát kell kialakítani. Az S_1 és S_2 informatikai rendszerek között kialakítandó $M_{11}...M_{15}$ és $M_{21}...M_{25}$ metódusokat tartalmazó kétirányú interfészt szemlélteti az alábbi ábra.



7. ábra Rendszerek közötti integráció (saját szerkesztés)

[3] Interfész - Az interfész szoftverek esetében két rendszer érintkezési felülete, ahol a kommunikáció menete mindkét rendszer számára ismert. Kétirányú interfészről beszélünk, ha mindkét fél megszólítható a másik által.

Amennyiben nem kivitelezhető az egy időben és kompatibilis eszközökkel történő fejlesztés, akkor külön-külön indított *Military Scrum*-okkal lefedhető az M_1 és M_2 interfészek fejlesztése. Ha azonban a megfelelő szoftverfejlesztési környezet adott, akkor az M_1 és M_2 interfészeket, valamint a velük megvalósított folyamatok implementálását egyazon agilis projektbe lehet szervezni. Az 5. táblázatban egy közös fejlesztési projektbe szervezett feladatlista néhány első sora jelenik meg.

KhSsz.	Ssz.	Feladat	Prior.	Mod.	Állapot	St.pont	Megjegyzés	Leírás	Képernyőterv
FK-1	S-11	M11 szolgáltatás kialakítása	100	S1	Várakozik	5		-	-
FK-1	S-12	M12 szolgáltatás kialakítása	90	S2	Várakozik	5		-	-
FK-2	S-13	M13 és M21 szolgáltatás kialakítása	80	S1, S2	Várakozik	8		-	-

5. táblázat kétirányú interfész kialakítására vonatkozó (saját szerkesztés)

A bemutatott rövid példa jól szemlélteti azt, hogy a különböző rendszerek közötti integrációs feladatok ily módon agilis technikával is támogathatók. A módszer azért különös fontosságú, mert segítségével különböző hálózati végpontokon lévő rendszerek integrációját is támogatni lehet. Az egyes rendszerek magas szintű illesztésével magas színvonalú hálózatalapú műveleti képesség [3; 17. old.] érhető el, amit az MH HID a sikeres információs műveletek zálogaként jelöl meg. A *Military Scrum* az informatikai rendszerek szervezésének egy modern megközelítését alkalmazza, mellyel külön-külön fejleszthető, folyamatosan karbantartható, frissíthető, egymással jól integrálható informatikai rendszerek tervezését és fejlesztését lehet támogatni, ezzel megteremtve a már említett hálózatalapú műveleti képesség szoftvertechnológiai előfeltételeit.

Ha a jövőben valamilyen célterületen a Magyar Honvédség a jelenleginél magasabb szintű informatikai integrációra törekszik a szövetségeseivel, akkor a tervezés és a megvalósítás során alkalmazhat agilis technikákat is. A rendelkezésre álló NATO szabványok alapjai lehetnek úgy a műszaki, mint funkcionális követelményhalmazok meghatározásának, amelyek alapján már összeáll egy agilis szoftverfejlesztési projekt kiinduló követelményrendszere.

KÖVETKEZTETÉSEK

A tanulmány első szakaszában feltárt és bemutatott általános informatikai iránymutatás szoftvertechnológiai értelmezése konkrét követelményrendszerként használható fel különböző szoftverfejlesztési tevékenységek előkészítési fázisában. A kialakított értelmezés minőségi elvként a megvalósítás- és a bevezetés fázisaiban is tovább hasznosítható. A cikkben bemutatott továbbfejlesztést, kiváltást, adatmigrálást és interfészekkel történő összekapcsolást támogató dokumentációs módszerek – megfelelő eszközökkel és szoftvertechnológiai támogatással – megfelelnek a Magyar Honvédség magas szintű informatikai követelményeinek. A módszer során keletkező dokumentumok magas szinten támogatják a különböző tervezési, fejlesztési lépéseket is. A korábban és a jelen értekezésben bemutatott *Military Scrum* szoftverfejlesztési módszertan lehetőséget biztosít a követelmények stratégiai, hadműveleti és harcászati szinten történő rendszerezéséhez. A módszer külön-külön dokumentációs technikát ad a különböző jellegű szoftverfejlesztési feladatokhoz és minden esetben lehetőséget biztosít az egyes katonai szakterületek azonosításához és követelményekhez rendeléséhez.

A szerzők fontosnak tartják megjegyezni, hogy napjainkban az agilis szoftverfejlesztés egyelőre nem egy elterjedt eljárás a katonai alkalmazás területén, ugyanakkor úgy gondolják, hogy a közeljövőben ez a technika egyre nagyobb teret fog nyerni, ezért célszerű lenne a jelenlegi irányelveket és szabályokat tovább gondolni és a kapcsolódó eljárásokat időben kialakítani, felkészülendő a jövő kihívásaira. Emellett szól az is, hogy korunk szoftvereinek bonyolultsága már egyre nehezebben teszi lehetővé azt, hogy hagyományos statikus eszközökkel határozzák meg a rendszerekkel szemben támasztott követelményeket. Ha az informatikai rendszerek elmúlt évtizedekben végbemenő hatalmas térnyerését tekintjük, akkor látható, hogy a megrendelőknek is meg kell változtatniuk a követelménytámasztás során alkalmazott módszereiket. Folyamatos megrendelői részvételre van szükség egy szoftver követelményeinek meghatározása, fejlesztése és fenntartása során. Napjaink agilis módszerei alapvetően azt várják el a megrendelőktől, hogy tudatosan felkészüljenek a folyamatos jelenlétre, a fejlesztők üzleti tudással történő támogatására. Az együttműködés során a megrendelői oldalon olyan képességek alakulhatnak ki, melyek segítségével önállóan, illetve külső támogatással lehetségessé válik az agilis tervezés. A megrendelőknek ez azért kifizetődő, mert részesévé válhatnak a szoftverek előállításának és valóban saját rendszereket tudhatnak magukénak az átadást követően a szoftver élettartama során.

Végezetül visszakanyarodva a jelenbe – elmondható, hogy a Magyar Honvédség Informatikai Stratégiájában, Szabályzatában és az Összhaderőnemi Híradó és Informatikai Doktrínájában jelenleg fellelhető szoftvertechnológiai igények kielégítéséhez megfelelő választás lehet a *Military Scrum* szoftverfejlesztési módszertan.

FELHASZNÁLT IRODALOM

[1] *A honvédelmi miniszter 58/2014. (IX. 10.) HM utasítása a Magyar Honvédség Informatikai Stratégiájának kiadásáról*. – Hivatalos Értesítő, 2014. évi 46. szám 5997-6006 o.

[2] *A honvédelmi miniszter 39/2014. (V. 30.) HM utasítása a Magyar Honvédség Informatikai Szabályzatának kiadásáról*. – Honvédelmi Közlöny, 2014. évi 7. szám 3614-3660 o.

[3] *Magyar Honvédség Összhaderőnemi Híradó és Informatikai Doktrína. 1. kiadás*. – Magyar Honvédség, 2013

[4] NÉGYESI I.: *COTS rendszerek alkalmazási lehetőségeinek vizsgálata,* HADTUDOMÁNYI SZEMLE IV. 4. 111-116. o. http://uni-nke.hu/downloads/kutatas/folyoiratok/hadtudomanyi_szemle/szamok/2011/_4/2011_4_tt_negyesi_imre_111_116.pdf (letöltve: 2017.04.26.)

[5] EVANS E.: *Domain-Driven Design: Tackling Complexity inthe Heart of Software. Courier in Westford,* Massachusetts, Pearson Education, Inc., 2003.

[6] GEREVICH J.: *Az agilis szoftverfejlesztés alkalmazásának lehetőségei a Magyar Honvédség számára*. In: Hadmérnök XII. 1. (2017) 170-181. o. http://hadmernok.hu/171_14_gerevich.pdf (letöltve: 2017. 04. 26.)

[7] GEREVICH J.: *Híradó-informatikai fejlesztést támogató agilis dokumentációs módszerek.* In: Hadmérnök XII. 3. (2017) 210-222. o. http://hadmernok.hu/173_19_gerevich.pdf (letöltve: 2017. 10. 15.)

[8] DENNING S.: *Agile: The World's Most Popular Innovation Engine*. In: Forbes.com https://www.forbes.com/sites/stevedenning/2015/07/23/the-worlds-most-popular-innovation-engine/#5aad99957c76 (letöltve: 2017. 10. 10)

[9] *DB-Engines Ranking* https://db-engines.com/en/ranking (letöltés: 2017. 10. 20)

[10] RUBIN K. S.: *Essential Scrum. Ann Arbor*, Michigan, USA, Pearson Education, Inc., 2013.

[11] MUNK S.: *Híradó-informatikai szolgáltatások alapjai II. Híradó-informatikai szolgáltatások fogalma, értelmezése.* In: Hadmérnök X. 4. (2015) 149-165. o. http://hadmernok.hu/154_14_munks.pdf (letöltve: 2017. 04. 15.)

[12] NÉGYESI I.: *Die überprüfung der voraussetzungen von COTS systemen*, Hadmérnök VII. 2. (2012) 371-376. o. ISSN: 1788-1919 http://www.hadmernok.hu/2012_2_negyesi.pdf (letöltve: 2017.04.26.)