Telemedicina olcsó eszközökkel

Végh Ádám, Antal Gábor, Dr. Bilicki Vilmos
Szegedi Tudományegyetem, Szoftverfejlesztés Tanszék

Napjainkban olyan új trendeket, paradigmaváltásokat figyelhetünk meg, melyeknek jelentős hatása lehet a telemedicinára is. A mobil telefonok elterjedése, ezek használatának előtérbe kerülése, személyes mivolta egyedi lehetőségeket teremt úgy az életvitel, mint egyes vitális paraméterek monitorozására. A nagy adatmennyiség robosztus kezelése sem okoz problémát a felhő technológiák segítségével. Olyan új szenzorok jelennek meg, melyek egyes telemedicina területeken teljesen új távlatokat nyitnak meg. Ezen területekre fókuszál a TOMI projekt, melynek célja a megbízható telemedicina megvalósítása olcsó eszközökkel. A cikk e projekt eddigi tapasztalatairól számol be.

These days we encounter new trends, paradigms changes worldwide that may have an effect on telemedicine as well. The use of mobile phones is increasingly common; their personal nature is creating new opportunities for monitoring both lifestyle and certain vital parameters. With new cloud technologies even the robust management of great quantities of data is no problem any longer. New sensors are emerging that open completely new perspectives for certain fields in telemedicine. The TOMI project focuses on these fields and its objective is the implementation of reliable telemedicine with low-priced instruments. This article reports on the experiences gained thus far in the project.

BEVEZETŐ

A mobil előfizetések penetrációja a következő évben meg fogja haladni a Föld lakosságának létszámát [1], az előfizetők közül már most is 1,3 milliárd okostelefon felhasználó van. Ezen felhasználók 91%-a 7*24 órában karnyújtásnyira van a telefonjától. Az USA tévénézőinek 88%-a telefonját is nézi TV nézés közben. A telefonok tehát olyan egyedi eszközök, melyek amellett, hogy asztali számítógéppel összemérhető számítási, tároló és kommunikációs képességekkel bírnak, személyesek, és mindig a felhasználók közelében vannak. Ez egyedi lehetőségeket teremt az életvitel monitorozására. Egy másik paradigma az olcsó és egyedi képességű szenzorok [2] megjelenése. Ma már 100 USD alatt kapható olyan készülék mely a kéz 3D térképét mm pontosan leképezi valós időben. De ilyen eszköz az egycsatornás EEG is. A lépésszámláló, mozgásmennyiség-mérő és egyéb fitnesz kellékek szintén elérhető áron és jó minőségben kaphatóak. A tárolási és számítási kapacitás szintén könnyen és olcsón elérhető.

Ezen paradigmaváltások, trendek ellenére sem igazán látunk sikeres telemedicina szolgáltatást. A WHO 2011-es felmérése szerint a klasszikus képalkotással összefüggő telemedicina szolgáltatások tekinthetőek csak elterjedtnek (a vizsgált országok 1/3-ában van). Az olyan területek, mint az otthonápolás, neurológia, diabétesz kezelés [3] csak a vizsgált országok kicsiny (<5%) részében érték el a kereskedelmi szintet. Hasonló a helyzet az mHealth-hez tartozó szolgáltatásokkal is. A klasszikus szolgáltatások, mint a segélyhívó központ, információs vonalak a megkérdezett országok 40-50%-ában jelen vannak, azonban az olyan innovatívabb szolgáltatások, mint a döntéstámogatás [4] vagy a páciensmonitorozás kevesebb, mint az országok 10%-ában van jelen kereskedelmi szolgáltatásként. A mobil telefonok, a felhő és a speciális, olcsó szenzorok korához hogyan kell a szoftverfejlesztésnek alkalmazkodnia?

- Hogyan tudunk hatékonyan és gyorsan adott célra fókuszáló telemedicinális alkalmazásokat fejleszteni?
- Hogyan tudunk olyan szenzorokat illeszteni, amelyek felhő háttérrel is bírnak?
- Hogyan tudjuk alkalmazni a szabványokat?

Sok hasonló kérdést lehet feltenni, és e kérdések kerültek a TOMI telemedicinás projekt célkeresztjébe is. A hatékony szoftverfejlesztéshez megfelelő referencia architektúra kell. Ez esetünkben olyan egyedi termékvonal alapú fejlesztést jelent, ahol az alap elemkészletet a HL7 szabványcsalád adja, erről fog szólni a következő fejezet. Ezután részletesebben áttekintjük a kialakított fejlesztési metodológiánkat, illetve röviden bemutatjuk az eddig elért eredményeinket.

A MODELL/TERMÉKVONAL ALAPÚ FEJLESZTÉS A GYAKORLATBAN

Napjainkban a szoftverfejlesztési projektek körében egyre szélesebb teret nyernek a prototípus alapú fejlesztési modellek. Ennek elsődleges oka az, hogy a rövid fejlesztési idejű prototípusok segítségével gyorsabban és hatékonyabban feltérképezhetők a megrendelő követelményei a fejlesztendő alkalmazással kapcsolatban. Ezáltal minden iterációban pontosabb specifikáció és a megrendelői követelményeknek egyre inkább eleget tevő alkalmazás állítható elő.

Mivel a prototípus alapú fejlesztési modellekben az egyes iterációk rövid fejlesztési időket igényelnek, ezért olyan eszközökre van szükség, amelyek támogatják a prototípusok gyors előállítását, valamint a követelmények változásával a prototípus gyors módosítását. Ezen eszköztár egyik rendkívül fontos eleme a modell alapú fejlesztés (Model-Driven Development, MDD). Lényege, hogy a prototípusokról a fejlesztő modell-leíró nyelvek és eszközök segítségével olyan modelleket készít, amelyek magas absztrakciós szinten modellezik a megvalósítandó rendszer ele-

meit, az elemek közötti kapcsolatokat és interakciókat, valamint a kezelendő adatokat. Ezen modellekből az MDD eszközök képesek beépített automatikus transzformációk segítségével alacsonyabb szintű modelleket, a folyamat végén pedig konkrét implementációt, forráskódot generálni.

Ezt a metodológiát egészíti ki a termékvonal alapú fejlesztési modell (Product Line Engineering, PLE). Lényege, hogy első körben a fejlesztendő alkalmazások által lefedett területek, domain-ek összessége kerül magas szinten elemzésre, modellezésre és implementálásra, majd e modellekből kerülnek származtatásra a termék specifikus követelmények, modellek és implementációk. A modell kiemelt előnye, hogy a domain változása mellett egyszerűbben kezelhető a termékek változása is, illetve az azonos domain-re épülő termékek integrációja könnyebben megvalósítható.

A gyakorlati alkalmazások szempontjából tekintve a modell alapú és a termékvonal alapú fejlesztési modellek eszköztára is többnyire olyan eszközöket tartalmaz, amely a rendszerben definiált adatok vagy a rendszer komponenseinek magas szintű modelljeiből képes alkalmazás vázakat előállítani adott programozási nyelven. A legtöbb esetben a modellező nyelvet a UML (Unified Modeling Language) képezi (ezen belül is legtöbbször az osztálydiagram, a szekvencia diagram vagy az állapotátmenet diagram a kiindulási modell [5][6][7][8][9]), a végső implementációs nyelv pedig a legtöbb esetben Java, C# vagy C++. Ilyen UML alapú MDD eszközök például a Modelio [10], az IBM Rational termékcsalád Rose [11] és Rhapsody termékei [12], az AndroMDA [13], az ObjectIF [14] és a SOLoist Framework [15]. Illetve léteznek saját modellező nyelvet definiáló MDD eszközök is, például az Acceleo [16] (bővíthető saját metamodellekkel) és az ActifSource [17]. Ugyanakkor a modell fogalmát tágabb értelemben véve a meglévő forráskód is lehet modell, legtöbbször a rendszerben lévő entitások implementációi, amelyekből CRUD (Create-Read-Update-Delete) jellegű alkalmazás prototípus generálható. Ilyen entitás alapú generáló eszközök például a JBoss Forge [18], a seam-gen [19], a MyEclipse for Spring [20], és az OpenXava [21].

Az előbb említett kódgeneráló eszközök alapvető hiányossága, hogy viszonylag kis funkcionalitással rendelkező alkalmazás prototípust tudnak csak generálni, és azt is sok esetben pontatlanul. Illetve az UML alapú MDD eszközök többsége csak adott adattagokkal és üres metódusokkal feltöltött osztályvázakat képesek generálni. A prototípus alapú fejlesztések esetében viszont legtöbbször ennél sokkal hatékonyabb generatív eszközökre van szükség.

TELEMEDICINÁBAN HASZNÁLHATÓ SZABVÁNYOK

Az integrált orvosi informatikai rendszerek szükségessége motiválta az orvosi informatikai szabványok kidolgozását és elterjedését. Az orvosoknak kommunikálniuk kell egymással, a gyógyszerészekkel, a betegekkel, illetve a kórházaknak is kapcsolatban kell állniuk egymással. A lokális rendszerek közötti kapcsolat és kommunikáció lényegesen hatékonyabb lehet, ha valamilyen egységes szabvány képezi a kommunikáció alapját. Továbbá új egészségügyi infrastruktúra kialakításakor, amennyiben az új rendszer valamely egészségügyi informatikai szabványt követi, akkor jelentősen könnyebben integrálható a meglévő egészségügyi rendszerekkel. Az egészségügyi informatikai szabványok fő céljai közé tartozik a klinikai ellátások gyorsítása és minőségének javítása, a költség- és kockázat csökkentés, hatékonyság növelés, az egészségügyi munkafolyamatok optimalizálása, és a tudástranszfer fejlesztés.

A telemedicina egyik fontos szabványa a HL7 (Health Level 7) [22], amely az egészségügyi információk cseréjére, megosztására, visszakeresésére és integrációjára biztosít szabványos megoldást. A HL7 tulajdonképpen egy szabványcsalád, amely többek között tartalmazza a klinikai üzenetek (HL7 Messaging) és dokumentumok (Clinical Document Architecture, CDA [23]) tartalmának formátumára vonatkozó ajánlást, az EHR (Electronic Health Record) bejegyzések tárolására vonatkozó szabványt és a klinikai döntéstámogató rendszerek (Clinical Decision Support System, CDSS) szabványát. A szabványok egy széles területet lefedő, hierarchikusan felépített domain modell struktúrán alapulnak, amelynek központi modelljét a HL7 RIM (Reference Information Model) képezi. Ezzel tulajdonképpen a HL7 fő célja a telemedicina strukturális jellegű szabványosítása.

Az egészségügyi informatika másik fontos szabványa a SNOMED CT (Systematized NOmenclature of MEDicine Clinical Terms) [24, 25], amelynek célja az egészségügyi terminológia elemeinek összegyűjtése és egységesítése, áthidalva a nyelvi és dialektusbeli különbségekből adódó problémákat. A szabvány tulajdonképpen egy ontológia-szerű felépítéssel rendelkező többnyelvű szinonimaszótárnak tekinthető. Olyan fogalmak összessége, amelyek között definiált az öröklődési reláció és az ok-okozati összefüggés relációja is. Ezáltal a SNOMED CT célja a lexikális jellegű szabványosítás.

A MI MEGKÖZELÍTÉSMÓDUNK

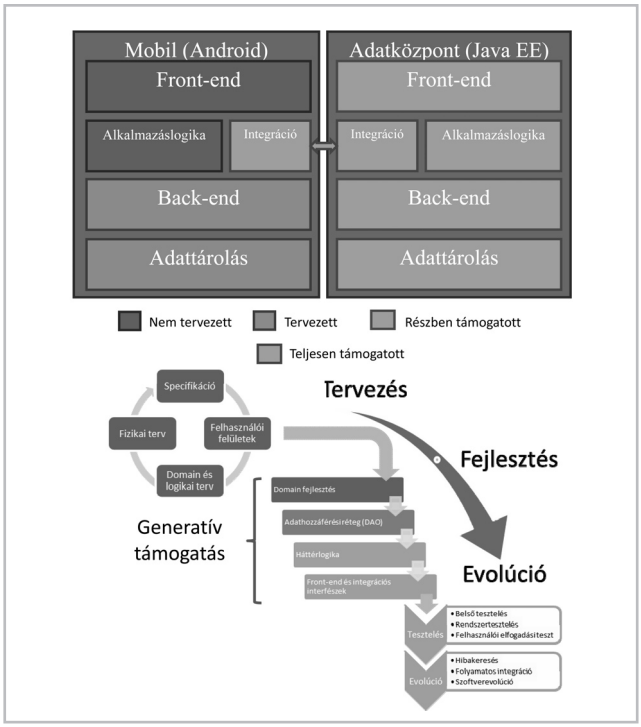
Egy szoftverfejlesztési folyamat három részből áll: tervezés, fejlesztés és tesztelés. Ezek lineáris vagy éppen iteratív megközelítése adja a fejlesztési folyamat dinamizmusát. Lineáris esetben nagyon lassú a fejlesztési folyamat, és nagyon kicsi a fejlesztett rendszerbe történő beavatkozási lehetőség. Az iteratív fejlesztési modellek viszont éppen az agilis mivoltukkal nem minden esetben tudnak olyan minőséget garantálni, melyek megfelelnek az orvosi adatok kezelésére vonatkozó elvárásoknak.

Az általánosan vett tervezés alapvetően kétféle megközelítést ismer: az alulról felfelé (bottom-up) és a felülről lefelé (top-down) történő tervezési, így fejlesztési megoldásokat. Míg az előbbi előnye, hogy az eredmény amint elkészül, kiértékelhető és használható, viszont hátránya, hogy lassabban készül el. A top-down fejlesztés előnye, hogy rögtön „kattintható” felhasználói felületeket, szoftver integrációs interfészeket készítjük el legelőször, nagy hátránya viszont, hogy nem használható, mely összezavarhatja a leendő fel-

használót. Számos esetben a felhasználó azt hiheti, hogy az adott szoftver termék készen van, pedig ez nem igaz, mivel a funkcionális működés hiányzik.

Azt tapasztaltuk, hogy egyik fenti módszer sem támogatja igazán hatékonyan a prototípus alapú fejlesztést az orvosi kutatás-fejlesztési területen. Olyan megoldásra van szükségünk, amely biztosítja az extrém gyors prototípusok fejlesztését, és azok kiértékelését, mindemellett biztosítja az orvosi rendszerekkel kapcsolatos biztonsági és megbízhatósági igényeket.

Az orvosi alkalmazásfejlesztés jellegzetessége, hogy az orvosok a napi munkájuk mellett rendkívül kis időt tudnak szánni arra, hogy a folyamataikat optimalizáló, vagy éppen kutatásukat támogató alkalmazásokat specifikálják. Azt tapasztaltuk, hogy egy meglévő prototípus használata során a kiértékelés lényegesen hatékonyabb, mint tiszta lappal a semmiből kiindulva definiáltatni adott alkalmazásokat vagy felületeket. Továbbá azt tapasztaltuk, hogy az úgynevezett funkcionális mock-up-ok, (esetleg kattintható rajzolt felületek) nem alkalmasak orvosi szempontból a kiértékelésre, a nem informatikus vénájú emberek nem minden esetben tudják elképzelni azt, hogy az adott felület hogyan funkcionál majd végleges alkalmazásként. Éppen ezért létrehoztunk egy olyan modell alapú alkalmazásgeneráló rendszert, mellyel képesek vagyunk funkcionális prototípusokat fejleszteni. A rendszer felépítését és működését az 1. ábra szemlélteti.



1. ábra
Modell alapú alkalmazásgeneráló rendszer

A szoftverfejlesztési folyamatot úgy gyorsítottuk, hogy tanulmányoztuk eddigi fejlesztéseinket, és azokat a repetitív tevékenységeket, melyek számos esetben hasonlóak, megpróbáltuk egy modell alapú kódgeneráló rendszer, a JBoss [26] ökoszisztémában működő Forge [18] keretrendszer segítségével generálni.

A szoftver fejlesztését tekintve alapvetően egyelőre a szerver oldali Java nyelvű, és az ehhez kapcsolódó JSF technológia által igényelt leírónyelvek (JSF, CSS) forráskódjai generálódnak. Interfészgenerálást tekintve a szerver oldalon REST szolgáltatások kerülnek generálásra, az Android kliensben pedig a szolgáltatásokat hívó REST kliens generálódik. A szerver és a kliens közötti kommunikáció JSON adatformátumban történik.

Azt tapasztaltuk, hogy egy szoftver fejlesztése során olyan – minden esetben hasonló – felületek jönnek létre, mint a felhasználó, szerepkörkezelés, regisztrációs folyamat, elfelejtett jelszó. Ezek kezelése egyszerű, könnyen modularizálható. Azonban jellemzően olyan felületekre is szükség van, mint adott entitások szerkesztése, kapcsolódó entitások megjelenítése, szűrő-kereső táblák, valamint integrációs adatkapcsolati interfészek. Ezek felépítését megvizsgálva arra a következtetésre jutottunk, hogy nagyon sok a repetitív hasonlóság, melyre forráskód generáló megoldásokat fejlesztettük.

Továbbá a szoftvergenerálást úgy egészítettük ki, hogy mind adattárolás során, mind adatkommunikációban legyen lehetőség szabvány alapú (pl.: HL7) kompatibilis megoldások generálására. Erre a generált rendszer moduláris kivitelezése ad lehetőséget.

ELSŐ EREDMÉNYEK

Több prototípusnál elkezdtük alkalmazni a kódgenerálás megközelítését. Ahhoz, hogy pontos összehasonlításokat tudjunk végezni, a korábbiakban az Eclipse [27] fejlesztőkörnyezetben kiegészített produktivitás mérő beépülőt (plugint) [28, 29] alkalmaztuk. A plugin képes mérni az aktívan töltött fejlesztési időt, a fejlesztési eseményeket, megnyitott fájlokat, és a fájlokon történt fejlesztői interakciót. A korábbi, részben egészségügyi-informatikai projektek során egy olyan adathalmaz és tapasztalat keletkezett, mely segítségével pontosan becsülni tudtuk az adott típusú felületek és a mögötte lévő inf-

Feladat	Átlagos fejlesztési idő embernapban	Átlagos fejlesztési idő kódgenerálással támogatott fejlesztés esetén	%	A teljes ráfordítási idő %-ban
Adatformátum leíró réteg (domain modell) fejlesztés	0.1	0.02	20%	5%
Adatkapcsolati réteg (DAO)	3	1	33%	10%
Megjelenítő-listázó-szerkesztő-törlő felületek fejlesztése (CRUD)	10	0.2	2%	50%
Authentikációs és authorizációs biztonsági réteg infrastruktúrájának interfészei (beléptetés, szerepkör kezelés)	20	0.2	1%	15%
Adatvizualizációs felületek	3	1	33%	10%
Rendszerintegrációs interfészek	3	1	33%	10%

1. táblázat
Átlagos fejlesztési idő felgyorsulása

rastruktúra megvalósítási költségeit. Ezzel az adathalmazzal hasonlítottuk össze a kódgenerálással keletkező mini alkalmazások fejlesztési költségeit. Az eredmények a generálható részeknél jól mérhető fejlesztési sebességnövekedést produkáltak, melyeket az 1. táblázat szemléltet.

A táblázatban szereplő számok az eddigi fejlesztések átlagai. Továbbá figyelembe kell venni, hogy a táblázatban szereplő számok embernapban értendők (egy ember egy munkanapnyi munkája), mely a generált kód előállítását és a finomhangolást, utómunkát is tartalmazza.

ÖSSZEFOGLALÓ

Cikkünkben áttekintettük a telemedicina hátterét formáló technológiák aktuális trendjeit, majd egy áttekintést adtunk a termékvonal alapú fejlesztés jelenleg elérhető megoldásairól. A mi megoldásunk abban különbözik ezen megközelítésektől, hogy megpróbálja kihasználni a szabványokban lévő mély tudást, azok megfelelő beillesztésével a termékvonal alapú referencia architektúránkba. Ezen metodológia és eszközkészlet segítségével valós telemedicina alkalmazásokat fejlesztünk/fejlesztettünk, a keretrendszerünk által biztosított hatékonyság növekedés átlagosan kb. 90%-kal csökkentette a ráfordításainkat. Azaz ennyivel gyorsabban és kevesebb munkával lehet telemedicina fókuszú alkalmazásokat fejleszteni.

Jelen kutatási eredmények megjelenését „Telemedicína fókuszú kutatások Orvosi, Matematikai és Informatikai tudományterületeken” című, TÁMOP-4.2.2.A-11/1/KONV-2012-0073 számú projekt támogatja.

A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg [30].

IRODALOMJEGYZÉK

[1] Tomi Ahonen: Latest Mobile Numbers for End of Year 2012

[2] M. Kasza, V. Szűcs, és Á. Végh: „Passive vs. Active Measurement: the Role of Smart Sensors”, He Third Int. Work. Pervasive Comput. Embed. Syst., vol PECES 2011, 2011.

[3] Global Observatory for eHealth series – Volume 2, Telemedicine – Opportunities and developments in Member States, ISBN 978 92 4 156414 4 ,13 January 2011

[4] Global Observatory for eHealth series – Volume 3, mHealth: New horizons for health through mobile technologies,ISBN 978 92 4 156425 0, 7 June 2011

[5] Abilio G. Parada, Eliane Siegert, Lisane B. de Brisolara: „Generating Java code from UML Class and Sequence Diagrams”. Computing System Engineering (SBESC), 2011 Brazilian Symposium, 07-nov-2011.

[6] Abilio G. Parada, Eliane Siegert, Lisane B. de Brisolara: „GenCode: A tool for generation of Java code from UML class models”. SIM 2011 – 26th South Symposium on Microelectronics, 2011.

[7] Muhammad Usman, Aamer Nadeem: „Automatic Generation of Java Code from UML Diagrams using UJECTOR”. International Journal of Software Engineering and Its Applications Vol.3, No.2, ápr-2009.

[8] Iftikhar Azim Niaz: „Automatic Code Generation From UML Class and Statechart Diagrams”. Computer Science Doctoral Program in Engineering University of Tsukuba, Japan, nov-2005.

[9] El Beggar Oma, Bousetta Brahim, Gadi Taoufiq: „Automatic code generation by model transformation from sequence diagram of system’s internal behavior”. International Journal of Computer and Information Technology (ISSN: 2279 – 0764), nov-2012.

[10] „Modelio, UML modeling tool – MDA, business process and software architecture design”. [Online]. Available at: http://www.modeliosoft.com/. [Elérés: 16-szept-2013].

[11] „IBM – Software – Rational Rose family – United States”. [Online]. Available at: http://www-03.ibm.com/software/products/us/en/ratirosefami. [Elérés: 16-szept-2013].

[12] „IBM – Software – Rational Rhapsody family – United States”. [Online]. Available at: http://www-03.ibm.com/software/products/us/en/ratirhapfami/. [Elérés: 16-szept-2013].

[13] „AndroMDA Model Driven Architecture Framework – AndroMDA – Homepage”. [Online]. Available at: http://www.andromda.org/index.html. [Elérés: 16-szept-2013].

[14] „objectiF – the Tool for Model-Driven Software Development with UML and BPMN in Java, C#, C++, BPEL, XSD and WSDL”. [Online]. Available at: http://www.microtool.de/objectif/en/. [Elérés: 16-szept-2013].

[15] „SOLoist Framework – Java Web Framework for Model-driven Development”. [Online]. Available at: http://www.soloist4uml.com/. [Elérés: 16-szept-2013].

[16] „Acceleo”. [Online]. Available at: http://www.eclipse.org/acceleo/. [Elérés: 16-szept-2013].

[17] „actifsource – Domain Model Driven Code Generator for any Language (C#, Cobol, C++, Groovy, html, Java, Ruby, xml, ...) – Eclipse Plugin”. [Online]. Available at: http://www.actifsource.com/index.html. [Elérés: 16-szept-2013].

[18] Forge I: [Online]. Available at: http://forge.jboss.org/. [Elérés: 16-szept-2013].

[19] „Chapter 2. Getting started with Seam, using seam-gen”. [Online]. Available at: http://docs.jboss.org/seam/2.3.1.Final/reference/html/gettingstarted.html. [Elérés: 16-szept-2013].

[20] „MyEclipse for Spring I Spring Development with MyEclipse I Spring Scaffolding I Spring Tutorial IIII”. [Online]. Available at: http://www.myeclipseide.com/me4s/. [Elérés: 16-szept-2013].

[21] „AJAX Java Web Framework for Rapid Application Development of Enterprise Applications – OpenXava”. [Online]. Available at: http://openxava.org/home. [Elérés: 16-szept-2013].
[22] „Health Level Seven International – Homepage”. [Online]. Available at: http://www.hl7.org/. [Elérés: 16-szept-2013].
[23] R. H. Dolin, L. Alschuler, C. Beebe, P. V. Biron, S. L. Boyer, D. Essin, E. Kimber, T. Lincoln, és J. E. Mattison: „The HL7 clinical document architecture”, J. Am. Med. Inform. Assoc., vol 8, sz 6, o 552, 2001.
[24] „SNOMED CT”. [Online]. Available at: http://www.ihtsdo.org/snomed-ct/. [Elérés: 16-szept-2013].
[25] L. Bos at al: „SNOMED-CT: The advanced terminology and coding system for eHealth”, Med. Care N.a 3, vol 121, o 279, 2006.
[26] „Community driven open source middleware”. [Online]. Available at: http://www.jboss.org/overview/. [Elérés: 17-szept-2013].
[27] „Eclipse – The Eclipse Foundation open source community website.” [Online]. Available at: http://eclipse.org/. [Elérés: 17-szept-2013].
[28] Kakuja-Tóth Gabriella, Végh Ádám Zoltán, Beszédes Árpád, Schrettner Lajos, Gergely Tamás, Gyimóthy Tibor: „Adjusting effort estimation using micro-productivity profiles”. 12th Symposium on Programming Languages and Software Tools. SPLST’11. Tallinn, Estonia, 05-okt-2011.
[29] Kakuja-Tóth Gabriella, Végh Ádám Zoltán, Beszédes Árpád, Gyimóthy Tibor: „Adding process metrics to enhance modification complexity prediction”. Proceedings of the 2011 IEEE 19th International Conference on Program Comprehension (ICPC 2011). Kingston (ON), 22-jún-2011.
[30] „TÁMOP-4.2.2.A-11/1/KONV-2012-0073 | Publikációk és előadások”. [Online]. Available at: http://www.u-szeged.hu/tamop422a0073/letoltheto-dokumentumok/publikaciok-eloadasok. [Elérés: 17-szept-2013].

A SZERZŐK BEMUTATÁSA

Végh Ádám Zoltán 2003-ban kezdte tanulmányait a Szegedi Tudományegyetem Természettudományi és Informatikai Karának Programtervező Matematikus szakán. 2007-ben nyert felvételt a Szoftverfejlesztés Tanszék gyakornoki pozíciójára. Ekkor kezdett el foglalkozni kezdetben hálózat-monitorozással majd szoftver rendszerek tervezési mintáival, fejlesztési feladataival. 2008-ban szerzett Programtervező Matematikus egyetemi diplomát, ebben az évben nyert felvételt a Szegedi Tudományegyetem Informatikai Doktori iskolájába. Tudományos tevékenysége elsősorban a Java EE alapú szoftver referencia architektúra fejlesztéséhez, a fejlesztési és karbantartási teljesítménymméréshez kapcsolódik. 2008 óta foglalkozik egészségügyi informatikai projektek fejlesztésével és szakmai vezetésével, 2010 óta fejlesztői produktivitás méréssel. 2012-től kezdett foglalkozni kódgenerálással egybekötött modell alapú fejlesztéssel, hatékonyságával. Kutatási területe a szoftver architektúra tervezési minták és a szoftverfejlesztői hatékonyság vizsgálata.

Antal Gábor 2011-ben szerzett programtervező informatikus BSc diplomát a Szegedi Tudományegyetem Természettudományi és Informatikai Karán, ahol 2010-től dolgozik a Szoftverfejlesztés Tanszéken. Szemantikus web keretrendszerekkel, ontológia megfeleltetésekkel kapcsolatban végzett kutatómunkát, jelenleg pedig tudományos segédmunkatársként a JPA entitás alapú Java EE alkalmazás-komponensek automatikus generálásával foglalkozik. 2013-ban szerzett programtervező informatikus MSc diplomát a SZTE-TTIK szakán. Diplomamunkáját „Felhasználói felület és logika generálás JEE 6 szabvány szerint JBoss környezetben” témában készítette. Jelenleg az SZTE Informatika Doktori Iskola PhD hallgatója.

Dr. Bilicki Vilmos a Budapesti Műszaki Egyetem villamosmérnök szakán végzett 1999-ben. 2001-től tudományos segédmunkatársként dolgozott a SZTE Informatikai Tanszékcsoportjánál, ahol elosztott tárolással és SIP kód tömörítéssel kezdett el foglalkozni. Kutatásokat végzett az elosztott kivonat tábla alapú BotNet-ek felderíthetőségével és a kis fokszámú elosztott kivonat tábla alapú P2P megoldások skálázhatóságával. 2002-től 2010-ig a Szegedi Tudományegyetem tanársegédje, majd 2011-től adjunktusa, ahol a Programrendszerek Fejlesztése, az IP alapú Hálózatok Tervezése és üzemeltetése I. és II., valamint a Hálózati Operációs Rendszerek szakirányos tantárgyak előadója. Számos hazai és külföldi ipari és kutatás fejlesztési projekt szakmai koordinátora. Doktori értekezését az Infrastruktúrához Alkalmazkodó alkalmazások témakörében írta meg, és 2011-ben nyerte el az informatikatudomány doktora (PhD) fokozatot summa cum laude minősítéssel.