Szoftverrendszerek fejlesztésének oktatása projektfeladat keretén belül [1]

Horváth Zoltán, Kozsik Tamás, Lövei László

{hz, kto, lovei}@inf.elte.hu
ELTE IK

Absztrakt. Szoftverrendszerek fejlesztésének oktatásához egyetemi környezetben megvalósított, de ipari jellegű projektfeladatok megoldásába vonjuk be a hallgatókat. A projektekben alap-, mester- és doktori képzésben részt vevő hallgatók működnek együtt oktatókkal, kutatókkal éles K+F feladatok megoldásában. Az oktatásba bevont feladatokat úgy választjuk ki, hogy azok ipari partnerrel történő együttműködésben, nemzetközi kapcsolatrendszerbe ágyazva legyenek megoldhatóak. A projektek innovatív jellege lehetővé teszi, hogy a kutatási eredményeket konferencia- és folyóiratcikkekben mutassuk be, diákköri dolgozatok, diplomamunkák és doktori dolgozatok készüljenek. Ipari méretű projektfeladatok megoldása során a hallgatók olyan készségeket szereznek a szoftverfejlesztés területén, amelyek hagyományos oktatási keretek között nem alakíthatóak ki.

1. Bevezetés

A szoftverfejlesztés gyakorlatának oktatása során gyakran szoktak kisebb projektmunkát bízni a hallgatókra, illetve csapatmunka gyakorlásához hallgatók 3-4 fős csoportjaira. Az itt bemutatott módszer abban tér el ettől, hogy viszonylag nagy hallgatói csapatokkal, nagyobb kutatásfejlesztési projektekben végezzük a szoftvermérnökség tanítását. A projektekben alap-, mester- és doktori képzésben részt vevő hallgatók működnek együtt oktatókkal, kutatókkal éles K+F feladatok megoldásán. A feladat megoldásának körülményei olyanok, mint amilyenre a szoftveriparban számíthatnak a hallgatók: éles határidők, magas minőségi elvárások, futó projektbe való becsatlakozás szükségessége, nagy méretű kód készítése, karbantartása, dokumentálása vár a hallgatókra. Ezen körülmények között olyan tapasztalatokra tesznek szert a hallgatók, amelyeket hagyományos egyetemi képzés során nem szereznének. Az a tény, hogy a projekt eredményét az iparban is felhasználják, különösen motiválja őket.

Az első projekt három éve indult. Célja egy refaktoráló [2] eszköz fejlesztése az Erlang programozási nyelvhez. A feladatot az Ericsson Magyarország megbízásából végezzük. Az ERASMUS program mentén együttműködünk angol egyetemekkel és cégekkel is a kutatásfejlesztés során: a Kenti és a Sheffieldi Egyetemmel, valamint az Erlang Training and Consulting céggel [5].

A cikk szerkezete a következő. A második szakaszban áttekintést adunk az oktatásba bevont projektekről. Az ezután következő szakaszokban az egyik projektet, nevezetesen az Erlanghoz

1 Támogatta az Ericsson Magyarország és az ELTE IKKK.

készülő refaktoráló eszközt fejlesztését mutatjuk be részletesebben. A harmadik szakasz a tanterv vonatkozó részeit tárgyalja. Ismertetjük, hogy a projekteket befogadó *Szoftvertechnológia labor* tantárgy hogyan illeszkedik a tantervbe, és hogy a projektmunka eredményei hogyan hasznosíthatók az oktatásban. A negyedik szakaszban a projektkurzus során alkalmazott módszertant tárgyaljuk. Bemutatjuk, hogy hogyan alakítjuk ki a csapatot, és hogy milyen elvek alapján határozzuk meg a fejlesztett szoftver felépítését. Az ötödik szakaszban felsoroljuk azokat a készségeket, amelyeket fejleszteni kívánunk a hallgatókban. A hatodik szakaszban az elért eredményeket mutatjuk be. Végül a hetedik szakaszban összefoglaljuk a tanulságokat.

2. Projektkurzusok

Jelenleg négy projektkurzust biztosítunk hallgatóink számára A „Szoftvertechnológia labor” tantárgy keretében. Érdekes módon ezekből három a funkcionális programozás ipari felhasználását célozza. Mindegyik projektben 10-15 alap-, mester- és doktori képzésben résztvevő hallgató vesz részt 2-3 oktató irányítása mellett. A projektek témáját nagyon körültekintően kell kiválasztani. Olyan témákra van szükség, amelyek mind oktatási, mind kutatási szempontból érdekesek, mert így a projekt minden résztvevője számára hasznos a feladatok megoldása. Az általunk választott négy téma a következő.

2.1. Erlang programok refaktorálása

Ahogy korábban említettük, ebben a projektben egy refaktoráló eszközt fejlesztünk az Erlang programozási nyelvhez [23]. Az eszköz statikus programelemzésre és jelentésmegőrző programtranszformációk alkalmazására ad lehetőséget.

2.2. F# programok elemzése

Ebben a projektben is statikus programelemzéssel foglalkozunk. A cél olyan eszközök fejlesztése, amelyekkel F# programozók munkáját könnyíthetjük meg [21].

2.3. Domain-specifikus nyelv digitális jelfeldolgozáshoz

A digitális jelfeldolgozáshoz használt algoritmusok megvalósítását és a kód karbantartását nagy mértékben megkönnyítheti egy erre a célra specializált programozási nyelv kifejlesztése. Ebben a projektben egy ilyen nyelvhez készítünk fordítóprogram prototípust és egyéb kapcsolódó szoftvereszközt (például nyomkövetőt) [27].

2.4. HypereiDoc

Ezen projekt célja egy epigrafikus és papirologikus szövegek elosztott, kollaboratív, rétegelt feldolgozását, kritikai kiadásra való előkészítését támogató XML alapú rendszer kifejlesztése [9,24,25,28].

3. A projektkurzus helye a tantervben

Karunkon a programtervező matematikus és informatikus hallgatók számos programozási nyelvekkel és szoftvertechnológiával kapcsolatos tárgyat végeznek el, mielőtt a Szoftvertechnológia labor keretében egy nagyobb projektfeladat megoldására vállalkoznának. Ezen kurzusok közül néhány tudományos alapozást biztosít: formális módszerek szekvenciális, párhuzamos és elosztott programok specifikálására és levezetésére, fordítóprogramok, algoritmusok és adatszerkezetek. A hallgatók három félév alatt ismerkednek meg az imperatív (C++, Ada, Java) és funkcionális (Clean vagy Haskell) nyelvekkel; ezen tantárgyak során a programozási nyelvek alapvető fogalmai és eszközei között a párhuzamosságot (Ada taszkok) és üzenetküldést (PVM) is hangsúlyosan tárgyaljuk. Más tárgyak a széles körben használt tananyagok [3] alapján szoftvermérnöki ismereteket adnak át, illetve gyakoroltatnak kis létszámú (3-4 fős) csoportokban. Ezen utóbbi kurzusok projektjei nehézségüket tekintve megmaradnak az egyetemi gyakorlófeladatok szintjén, és meg sem közelítik egy tipikus ipari projekt bonyolultságát.

A következőkben az Erlang programok refaktorálására alkalmas eszköz (RefactorErl) elkészítését irányzó projektkurzus részletes bemutatása található. Az Erlang [1] egy mohó kiértékelésű, dinamikusan típusozott, nem tisztán funkcionális programozási nyelv, melyet az Ericsson fejlesztett ki. A nyelv különösen alkalmas konkurens és elosztott, hibatűrő, gyengén valósidejű rendszerek, például telekommunikációs alkalmazások készítésére. Az Erlangban a funkcionális nyelvi eszközök mellett folyamatok, valamint üzenetküldési és -fogadási szerkezetek vannak. Rendelkezik még a nyelv modulrendszerrel, kivételkezelést támogató nyelvi elemekkel, önelemzési lehetőségekkel, előfordítóval és jelentős méretű szabványos programkönyvtárral. A mesterképzés során az érdeklődő hallgatók választhatnak olyan kurzusokat is, amelyekben típusrendszerekről, funkcionális nyelvek fordításáról, lambda kalkulusról, elosztott és párhuzamos funkcionális programozásról, formális szemantikákról és helyességbizonyítási eszközökről van szó. Ezek a kurzusok mind kapcsolódnak a RefactorErl projektkurzus témájához.

3.1. Választható tantárgy az Erlang nyelvről

A RefactorErl projekthez kapcsolódó egyik fontos választható tantárgy az Erlang nyelvet mutatja be. A kurzust, mely egyre nagyobb érdeklődést vált ki a hallgatók körében, az Ericsson Magyarország javaslatára indítottuk. Korábban ez a cég – amely távközlési alkalmazásokat fejleszt Erlangban – sikeres előadássorozatot tartott az Erlang Training and Consulting cég közreműködésével a Central European Functional Programming School elnevezésű nyári iskolán [4], melyet tanszékünk szervezett.

Az Erlang kurzust az alapképzés utolsó évében és a mesterképzés során vehetik fel a programtervező informatikus hallgatók. Jelenleg a tárgy legtöbb hallgatója programtervező matematikus, azaz a kifutó 5 éves egyetemi képzésünkben vesz részt. Három évvel ezelőtt, a tárgy első meghirdetésekor három hallgató vette fel a kurzust. Azóta átlagosan évi 20 hallgató jelentkezik, melyeknek körülbelül a fele el is végzi a tárgyat.

3.2. A RefactorErl projektkurzus

A RefactorErl projektkurzust azoknak a hallgatóknak kínáljuk fel, akik a legjobban teljesítettek az Erlang kurzuson. A projekt keretében a hallgatók megtanulják, hogyan kell nagyméretű és -bonyolultságú szoftvereket fejleszteni. A kurzusért 16 ETCS kreditet kaphatnak, valamint folytathatják a kapott feladat megoldását szakdolgozatként [6,10,11,19] és diplomamunkaként [8] is.

A szoftver, amelyhez a hallgatók komponenseket fejlesztenek, körülbelül harmincezer sor méretű. Az a tapasztalatunk a RefactorErl és az F# projektek kapcsán, hogy a hallgatók könnyebben elboldogulnak ezzel a nagy funkcionális kódbázissal, mint egy ugyanekkora méretű Java kóddal: a funkcionális programozási paradigma, úgy tűnik, segít a szoftverek bonyolultságának kezelésében.

3.3. A RefactorErl hasznosítása az oktatásban

Érdekes módon a projektkurzus során elkészült refaktoráló eszköz nem csak a szoftveriparban hasznosul; visszaforgatható az oktatásba is. A refaktoráló eszköz jól használható az Erlang nyelv szintaxisának és szemantikájának magas szintű tanítása során, valamint a helyes programozási stílus kialakításához. Ezt mutatta a RefactorErl korábbi (prototípus) megvalósításának sikere a második Central European Functional Programming School nyári iskolán [13], ahol 35 hallgató vett részt egy Erlang/RefactorErl kurzuson [14]. Már kezdő szinten is jól használható egy refaktoráló eszköz a programozási nyelv oktatásában: segítségével a hallgatók elsajátíthatják a kódminőség lépésenkénti javításának technikáját.

4. A projektmunka során alkalmazott módszerek

A RefactorErl eszköz fejlesztése az Ericsson Magyarország által támogatott, ipari projektekre hasonlító kutatás-fejlesztési feladat. A munkatervről félévente állapodunk meg az Ericsson Magyarország munkatársaival, és heti-havi rendszerességgel egyeztetünk az eszközzel szembeni elvárásokról. A refaktoráló eszköz második verzióját az Ericsson Magyarország sikerrel alkalmazta egy nagyméretű és bonyolult távközlési szoftver modulszerkezetének átalakításához [16].

A projektben résztvevő csapat létszáma és szerkezete időben változó. Három oktató, 2-4 doktorandusz (akiknek a doktori kutatási témájuk is ehhez a projekthez kapcsolódik) és 7-12 hallgató működik együtt a feladat megoldásán. A csoport folyamatos változásának két oka is van. Egyrészt a résztvevő hallgatók cserélődnek, ahogy végeznek a tanulmányaikkal, illetve ahogy felveszik a projektkurzust. Másrészről több hallgató és doktorandusz is külföldön tölt egy-egy félévet a projektben szintén együttműködő angol egyetemeken és ipari partnernél. A csapat változása szükségessé teszi, hogy a készülő szoftver szerkezete ennek figyelembe vételével kerüljön megtervezésre, és hogy a szoftverkomponensek közötti interfészek pontosan definiáltak és dokumentáltak legyenek. A RefactorErl eszköz prototípusának elkészítése során nyilvánvalóvá vált, hogy az a programszerkezet, ami 2-3 hallgatóval történő fejlesztéséhez megfelelt, nem tette lehetővé több hallgató bevonását [12]. Ezért újra kellett tervezni és implementálni az eszközt. A jelenlegi szerkezetben már megfogalmazhatók 1 emberhónapnyi feladatok, de még így is szükség van folyamatos kommunikációra a hallgatók és a projektet vezető doktoranduszok, oktatók között: hetente kétszer két órában projektmegbeszélést tartunk.

A projektmegbeszélések során a csapat áttekinti a projekt aktuális állapotát (elemzi az elért eredményeket, meghatározza a következő heti feladatokat), a hosszútávú feladatokról és megoldási lehetőségeikről vitázik (így az új csapattagok is kapnak egy általános képet a projektről, megismerik a szoftver szerkezetét, tervezésének szempontjait), algoritmusokat próbál megfogalmazni a bonyolultabb részfeladatok megoldására (ez gyakran a szakirodalom feldolgozásával is jár), és részleteiben átvizsgálja és véleményezi a hallgatók által írott kódokat.

A kódok átvizsgálása nagyon fontos módszernek bizonyult. A hallgatók, különösen a projektbe újonnan bevontak, gyakran hoznak hibás tervezési vagy implementációs döntéseket, sokszor alacsony minőségű, rossz stílusú, gyenge hatékonyságú kódot állítanak elő. Hajlamosak elfeledkezni a kódolási konvenciókról, után megválasztott azonosítókat használnak, gyatra megjegyzéseket írnak. A kódátvizsgálás során a csapat vezetőire nagy felelősség hárul: megfelelő pszichológiai érzékkel elkerülhetők a személyes konfliktusok, és a csapat egésze számára sok előnnyel jár a kódok elemzése.

- A csapat tagjai megtanulnak együttműködni egymással.
- Mások kódjának kritikai elemzése nem csak a kódminőségre, hanem a hallgatók tudására is kedvezően hat.
- A viták során letisztulnak az interfészek, a szoftver szerkezete egységesebbé válik.
- A hallgatók megismerik a szoftver egészét, így az általuk írt kód valóban jól integrálható lesz a többi komponenssel.
- A csapattagok el tudják kerülni programfunkciók többszöri leprogramozását.

Minden félévben készítünk egy jelentést. Ez a kb. 200-400 oldalas részletes dokumentum sokat segít abban, hogy az összegyűlt tudást átadhassuk az új csapattagoknak.

5. A projektmunka során kialakított készségek és ismeretek

A projektkurzus elvégzése több területen is fejleszti a hallgatók készségeit, ismereteit. Először is az, hogy a hallgatók pár ezer soros, összetett kódot terveznek és fejlesztenek Erlangban, *funkcionális programozási ismereteiket és az Erlang programozási nyelvről szerzett tudásukat* nagymértékben elmélyíti. Másodszor, a projekt során jelentésmegőrző programtranszformációk előfeltételeit kell elemezniük, és ennek hatására megismerkednek a *statikus programelemzés* technikáival, mely segít a programnyelvi konstrukciók *szemantikájának* alaposabb megértésében [17]. Harmadrészt, mivel a refaktoráló eszköz szintaktikus elemzője jelentősen különbözik a fordítóprogramokhoz készített szintaktikus elemzőktől, a hallgatók megtanulják a *fordítóprogramok* tárgyból szerzett ismereteiket alkotó módon alkalmazni. Végezetül a hallgatók tapasztalatokat gyűjtenek a *szoftverfejlesztés* elméleti és gyakorlati kérdéseivel kapcsolatban.

A hallgatók szoftverfejlesztésben legfontosabb készségeit alakítjuk ki, fejlesztjük a projektmunka során. Meg kell érteniük egy már létező, összetett szoftver működését és az általuk megoldandó feladat specifikációját, meglévő interfészeknek megfelelő programkomponenseket fejlesztenek, új interfészeket terveznek, kódot írnak, karbantartják, tesztelik és dokumentálják, és egymással szorosan együttműködve teszik mindezt.

A hallgatóknak jártasságot kell szerezni a csoportmunkát támogató szoftverek (verziókövető és projektvezetést támogató rendszerek, wiki, levelező listák), dokumentációs eszközök (edoc, LaTeX) és tesztelő eszközök (QuickCheck, CruiseControl) használatában, alkalmazásában. Előfordul az is, hogy saját, alkalmazásfüggő tesztelő eszközt és környezetet kell kialakítaniuk. Azzal is megismerkednek, hogy egy refaktoráló eszköz hogyan használható a szoftverfejlesztői munka során.

A projektben programok szemantikája és statikus elemzése témakörben kutatási feladatokat is meg kell oldani. A kutatás a szakirodalom feldolgozásával, algoritmusok tervezésével és elemzésével jár. A mesterképzésben részt vevő hallgatók a kutatási feladatokban részt vállalva felkészülhetnek a doktori képzésre, és diplomamunkájuk témájában folytathatják doktori tanulmányaikat.

6. A projekt eredményei

A RefactorErl projekt kutatás-fejlesztési jellege miatt két szempontból is érdekes a hallgatók számára. Először is, az elkészített szoftver kurrens szoftvertechnológiai ismereteket, módszereket mutat be a hallgatóknak: modellalapú architektúra, adatbázisok, felhasználói felületek, elosztott rendszerek köztesrétege, XML stb. Másrészt a kutatási feladatok kapcsán a hallgatók izgalmas és bonyolult feladatok megoldását, adatok hatékony ábrázolását, algoritmusok tervezését gyakorolhatják. Még a doktori hallgatók számára is tartogat a téma kihívásokat, hiszen a dinamikusan típusozott, elosztott és konkurens programozást támogató nyelvi elemeket is tartalmazó Erlang nyelv statikus elemzése komoly kutatási feladat. A projekt témájának megválasztása tehát biztosíthatja, hogy a projekt résztvevői: hallgatók, doktori hallgatók, és az oktatók egyaránt megtalálják a számukra megfelelő szintű részfeladatokat. A projektek oktatási- kutatási szempontból mérhető eredményességéhez nagyban hozzájárul az is, hogy az ipari partner, az Ericsson Magyarország engedélyezte, hogy a kifejlesztett szoftver nyílt forráskódú legyen, és hogy a kutatási eredmények is szabadon publikálhatók legyenek.

Az erős kutatási tevékenység eredményei konferencia- és folyóiratcikkek, doktori disszertációk és diplomamunkák formájában is megjelennek, illetve a tehetséggondozás, a Tudományos Diákkör számára is színvonalas témakínálatot biztosít a projekt [7,22,26]. ERASMUS csereprogramban résztvevő hallgatóink sikeresen kapcsolódtak be a társegyetemeken futó K+F projektekbe is [15,18,22].

Végezetül, projektünk gyakorlati sikerességét az is bizonyítja, hogy az Ericsson Magyarország fel tudta használni eszközünket egy nagyméretű távközlési szoftver átstrukturálásához.

7. Összefoglalás

Innovatív kutatás-fejlesztési projektek bevonása az oktatásba számos előnnyel jár. Egy éles ipari feladat megoldása a szoftvertechnológiai projektkurzuson jobban motiválja a hallgatókat, mint egy egyetemi gyakorlófeladat, sokkal inkább fejleszti készségeiket és ismereteiket, és segíti kollektív és egyéni felelősségérzetük kialakítását a magas minőségi követelmények és határidők betartására vonatkozóan.

A projektmunka témájának helyes kiválasztása szükséges ahhoz, hogy alap-, mester- és doktori képzésben résztvevő hallgatók sikeresen együtt tudjanak működni a projektben, és hogy minden résztvevő megfelelő szintű feladatot kapjon. A hallgatóknak kiadott részfeladatokat úgy kell meghatározni, hogy a projektmunka hosszabb távon is eredményes lehessen, és alkalmazkodni tudjunk a csoport méretének és szerkezetének időbeli változásaihoz. Ez a megközelítés biztosítja az ismeretek folyamatos átadását is a régi és az új hallgatók között.

Irodalom

1. J. Armstrong, R. Virding, M. Williams, C. Wikstrom: *Concurrent Programming in Erlang*. Prentice Hall (1996)
2. M. Fowler, K. Beck, J. Brant, W. Opdyke, D. Roberts: *Refactoring: Improving the Design of Existing Code*. Addison-Wesley (1999)
3. I. Sommerville: *Software Engineering (7th Edition)*. International Computer Science Series, Addison Wesley (2004)
4. Z. Horváth (szerk.) *Central European Functional Programming School. First Summer School, CEFP 2005, Budapest, Hungary, July 2005. Revised Selected Lectures*. LNCS 4164. Springer (2006)
5. H. Li, S. Thompson, L. Lövei, Z. Horváth, T. Kozsik, A. Víg, T. Nagy: *Refactoring Erlang programs*. The Proceedings of 12th International Erlang/OTP User Conference, Stockholm, Sweden (2006), 10 oldal, http://www.erlang.se/euc/06/
6. T. Nagy, A. Víg: *Storing Erlang source code in database*. Szakdolgozat, Eötvös Loránd Tudományegyetem, Informatikai Kar, Budapest (2006)
7. T. Nagy, A. Víg: *An Erlang refactor step: Tuple function arguments*. XXVIII. Országos Tudományos Diákköri Konferencia, Miskolc, 2007. (második helyezés)
8. T. Nagy, A. Víg: *Erlang refactor tool*. Diplomamunka, Eötvös Loránd Tudományegyetem, Informatikai Kar, Budapest (2007)
9. P. Bauer, Zs. Hernáth, Z. Horváth, Gy. Mayer, Zs. Parragi, Z. Porkoláb, Zs. Sz. Sztupák: *HypereiDoc - an XML based framework for supporting cooperative text editions*. Advances in Databases and Information Systems (ADBIS 2008), LNCS 5207, 14–29, Springer (2008)
10. I. Bozó: *Erlang Refactoring: Inline Function*. Szakdolgozat, Eötvös Loránd Tudományegyetem, Informatikai Kar, Budapest (2008)
11. D. Horpácsi: *Erlang modul refaktorálás - függvények áthelyezése modulok között*. Szakdolgozat, Eötvös Loránd Tudományegyetem, Informatikai Kar, Budapest (2008)
12. Z. Horváth, L. Lövei, T. Kozsik, R. Kitlei, T. Nagy, M. Tóth, A. Víg, R. Király: *Building a Refactoring Tool for Erlang*. Int'l Workshop on Advanced Software Development Tools and Techniques, ECOOP 2008, Paphos, Cyprus, July 8, 2008. http://smallwiki.unibe.ch/wasdett2008/
13. Z. Horváth, R. Plasmeijer, A. Soós, V. Zsók (szerk.): *Central European Functional Programming School. Second Summer School, CEFP 2007. Revised Selected Lectures*. LNCS 5161, Springer (2008)
14. T. Kozsik, Z. Csörnyei, Z. Horváth, R. Király, R. Kitlei, L. Lövei, T. Nagy, M. Tóth, A. Víg: *Use Cases for Refactoring in Erlang*. Central European Functional Programming School, Revised Selected Lectures, LNCS 5161, 250–285, Springer (2008)

15. H. Li, S. Thompson, Gy. Orosz, M. Tóth: *Refactoring with Wrangler, updated*. Proceedings of the 2008 SIGPLAN Erlang Workshop, 61–72, ACM (2008)

16. L. Lövei, Cs. Hoch, D. Horpácsi, H. Köllő, T. Nagy, A Víg: *Refactoring Module Structure*. Proceedings of the 2008 SIGPLAN Erlang Workshop, 83–89, ACM (2008)

17. L. Lövei, Z. Horváth, T. Kozsik, R. Király, R. Kitlei: *Static rules for variable scoping in Erlang*. Proceedings of the 7th International Conference on Applied Informatics, vol. 2, 137–145 (2008)

18. T. Nagy, A. Víg: *Erlang Testing and Tools Survey*. Proceedings of the 2008 SIGPLAN Erlang Workshop, 21–28, ACM (2008)

19. M. Tóth: *Erlang Refactoring: Extract Functions*. Szakdolgozat, Eötvös Loránd Tudományegyetem, Informatikai Kar, Budapest (2008)

20. P. Bauer: *XML modell tervezése a HypereiDoc keretrendszerhez*. XXIX. Országos Tudományos Diákköri Konferencia, Debrecen, 2009.

21. P. Diviánszky és mások: *Infrastructure for Analysis of F# Programs*. Poszter. ELTE Innovációs Nap, 2009.

22. Cs. Hoch: *Testing Erlang with QuickCheck*. XXIX. Országos Tudományos Diákköri Konferencia, Debrecen, 2009.

23. Z. Horváth és mások: *Refactoring Erlang Programs*. (2009) http://plc.inf.elte.hu/erlang/

24. M. Kovács, Zs. Sz. Sztupák: *Rétegelt adatállományok kezelése XML segítségével*. XXIX. Országos Tudományos Diákköri Konferencia, Debrecen, 2009. (második helyezés)

25. Zs. Parragi, Zs. Sz. Sztupák: *Ókori szövegek digitális tárolása, megjelenítése és kollaboratív szerkesztése*. XXIX. Országos Tudományos Diáköri Konferencia, Debrecen, 2009. (harmadik helyezés)

26. M. Tóth, I. Bozó: *Függvényközpontú refaktoring funkcionális nyelvekben: függvény általánosítás, függvény kiemelés, függvény behelyettesítés*. XXIX. Országos Tudományos Diáköri Konferencia, Debrecen, 2009. (1. helyezés)

27. *Domain specific language for digital signal processing*. (2009) http://dsl4dsp.inf.elte.hu/

28. *HypereiDoc*. (2009) http://hypereidoc.elte.hu/