

Programiranje v parih v srednjih šolah

Gabrijela Krajnc¹, Viljan Mahnič²

¹T ŠC Kranj, Kidričeva 55, 4000 Kranj, Slovenija, gabrijela.krajnc@quest.arnes.si

²Univerza v Ljubljani, Fakulteta za računalništvo, Tržaška cesta 25, 1000 Ljubljana, viljan.mahnic@fri.uni-lj.si

Ekstremno programiranje je nov stil razvoja programske opreme, ki temelji na odličnih aplikacijah programskih tehnik, medsebojne komunikacije in teamskega dela. S pomočjo le tega dobimo rezultate, ki si jih nismo mogli niti zamisliti. Glavna praksa ekstremnega programiranja je programiranje v parih. Pri programiranju v paru dva programerja drug ob drugem, za istim računalnikom hkrati analizirata, načrtujeta in razvijata ter testirata programsko opremo. Zagovorniki te prakse trdijo, da so programi, ki so nastali pri delu v paru, boljši, z manj napakami in lepšim vmesnikom. Menimo, da je programiranje v parih zelo uspešna metoda tudi za učenje programiranja v srednji šoli. Začetni kvalitativni in kvantitativni rezultati prikazujejo, da je programiranje v parih pri učenju programiranja povečalo znanje in zadovoljstvo dijakov. Raziskali smo naravo programiranja v parih in preizkusili na kakšen način lahko s to prakso izboljšamo poučevanje, zvečamo motivacijo in povečamo znanje dijakov pri poučevanju programiranja v srednjih šolah.

Ključne besede: Agilne metodologije, ekstremno programiranje, programiranje v parih, kvalitativni in kvantitativni rezultati

1 Uvod

Temelji agilnih metodologij, med katerimi je tudi ekstremno programiranje, so bili postavljeni leta 2001. Na tem sestanku so obravnavali obstoječe metodologije in opisali nov trend pri razvoju metodologij, katerih temeljni lastnosti sta učinkovitost in prilagodljivost. Izpeljali so štiri načela:

- Posamezniki in njihova komunikacija so pomembnejši kot sam proces in orodja.
- Delujoča programska oprema je pomembnejša kot popolna dokumentacija.
- Vključevanje uporabnika je pomembnejše, kot pogajanje na osnovi pogodb.
- Upoštevanje sprememb je pomembnejše od sledenja planu. (The Agile Manifesto, 2001).

Ekstremno programiranje je metodologija razvoja programske opreme, ki temelji na enostavnosti, komunikaciji, povratni informaciji in ne nazadnje pogumu. Deluje z močno povezanostjo celotne razvojne skupine, katere pomemben del je tudi naročnik ali stranka. Le ta je del skupine in neprestano sodeluje pri razvoju projekta. Vsi razvijalci delujejo usklajeno. Razvilo se je kot rezultat kritičnega pogleda na celovite, težko obvladljive metodologije, ki imajo vsak korak predpisan do potankosti. Kljub kvaliteti produktov razvitih s pomočjo takšnih metodologij, je razvojni čas zelo dolg, testiranje se opravi šele na koncu, sodelovanje naročnika je omejeno na zgodnje faze razvoja. Novi trendi razvoja zahtevajo večjo prilagodljivost, krajše čase, nenehno testiranje in stalno sodelovanje naročnika, kar zagovarjata Kent (2005) in Jeffries (2001).

Programiranje v parih je glavna praksa ekstremnega programiranja. Osnovna definicija programiranja v parih je, da dva programerja, drug ob drugem, za enim računalnikom

analizirata, načrtujeta in razvijata ter testirata programsko opremo. (Beck, 2005; Jeffries, 2001; Williams, 1999).

Programiranje v parih, kot glavno prakso ekstremnega programiranja, smo poskusno uvedli v pouk programiranja v srednji šoli. Pri programiranju v parih smo preverili vpliv nove metode na znanje dijakov, samopodobo, strpnost, prilagajanje in zadovoljstvo. Zanimalo nas je, kako se ta praksa obnese pri učenju in utrjevanju programiranja ter razvijanju medsebojnih odnosov. Dokazali smo, da je izjemno pomembna priprava učitelja in priprava dijakov, ter obravnava psiholoških pogojev programiranja v parih. Le tako je metoda tudi pri pouku programiranja zelo uspešna. Dijaki so pridobili mnogo znanja ter razvili so strpnost in medsebojno pomoč.

2 Pouk programiranja v srednjih šolah

Pouk programiranja danes poteka v tehniških gimnazijah in srednjih strokovnih šolah na smeri elektrotehnik računalničar. Ne tako dolgo nazaj smo pri praktičnem pouku programiranja sedeli v parih za enim računalnikom. V to nas je prisililo pomanjkanje strojne opreme. Temu bi na nek način lahko rekli programiranje v parih. Delo takrat ni bilo primerno pripravljeno in vodeno in je pri paru po navadi eden delal, drugi pa je bil pasivni opazovalec. Ocenjevanje dela ni bilo objektivno, saj ni prikazovalo realnega znanja obeh udeležencev. Le redki so pri tem pridobili več znanja in spretnosti, kot bi ga pri samostojnem delu. Takoj, ko so šole dobile zadosti računalnikov, smo 'programiranje v parih' opustili in praktično delo danes poteka kot samostojno delo dijaka na svojem računalniku. Vsaka oblika ima svoje prednosti in slabosti. Gotovo smo glede na sedanje delo in izkušnje prepričani tako učitelji kot dijaki, da je takšna oblika najprimernejša za optimalno delo in objektivno

ocenjevanje. S pomočjo eksperimenta smo preverili ali je mogoče s pomočjo nove metode programiranja v parih izboljšati delo v razredu, potek pouka, in seveda znanje ter osebni razvoj dijakov.

3 Programiranje v parih pri pouku

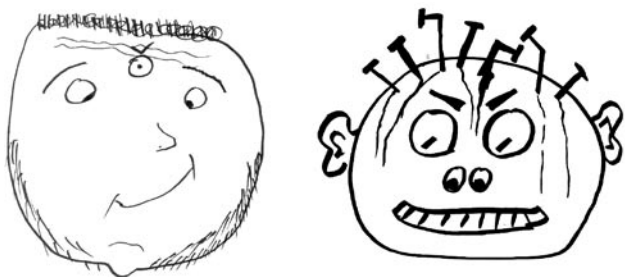
Pred uvedbo nove prakse v pouk, smo izvedli simulacijo risanja v parih. S pomočjo le te so dijaki dobili občutek, kako poteka programiranje v parih. Po simulaciji smo se pogovorili o psiholoških pogojih dela ter izbrali naloge za delo. Dijaki so se pri praktičnem pouku programiranja razdelili v dve skupini. Odločili smo se, da bomo z eno skupino delali po novi metodi, z drugo skupino pa bomo delali kot doslej. V skupini, kjer so dijaki delali v parih, so se pari na začetku formirali poljubno, nato pa so se po vsaki opravljeni nalogi zamenjali. Po mesecu dni smo primerjali rezultate obeh skupin.

3.1 Simulacija programiranja v parih

Risanje v parih je simulacija, ki je bila razvita kot pomoč pri predstavitvi metode programiranja v parih programerjem in menagerjem (Kerievsky, 2004). S pomočjo simulacije smo želeli dijakom prikazati potek programiranja v parih. Simulacija je potekala 30 minut. Tretina časa je bila namenjena simulaciji, dve tretini časa povzetku.

3.1.1 Opis simulacije

V skupini smo imeli parno število dijakov. Vsakemu dijaku smo dali dva bela lista in flumaster. Vsi so v dveh minutah na dva lista narisali dva obraza.



Slika 1: Primer samostojnega risanja obrazov.

Ko so dijaki končali (oziroma sta pretekli dve minuti), je avtorica Gabrijela Krajnc prosila, da so se razdelili v pare. Pare so formirali poljubno. Nato so si pari poiskali primeren prostor. Vsak par je ponovno dobil dva lista in dva flumastra različnih barv. Ponovno so morali risati obraze, tokrat v paru, vsakega na svoj list. Po končani simulaciji, smo pobrali izdelke. Vse izdelke smo izobesili tako, da so jih lahko vsi videli. Kot pravi oblikovalec simulacij Kerievsky (2004), so simulacije le izgovor za razpravo. Tudi mi smo se po koncu simulacije zbrali za razpravo.

3.1.2 Razprava

Ogledali smo si izdelke, ki so nastali pri samostojnem delu in pri delu v parih. Avtorica Gabrijela Krajnc je postavila naslednja vprašanja za analizo.

- Kako ste se lotili risanja v paru?
- Katere slike imajo po vašem mnenju večjo originalno vrednost?
- Kdaj ste se bolj koncentrirali in zakaj?
- Kdaj je bilo zabavneje?
- Kaj vam je, oziroma kaj vam ni bilo všeč pri risanju v parih?
- V čem menite se simulacija razlikuje od programiranja v parih?



Slika 2: Primer slik, ki sta nastali pri risanju v parih.

Risanja v paru so se dijaki lotili na različne načine. Nekateri so risali simetrično (napr. eden oko in drugi oko), drugi pa so se domenili in eno sliko risali simetrično, pri drugi pa je vsak narisal en del. Dijaki so povedali, da so se pri risanju v paru bolj zabavali, pri samostojnem risanju pa so lažje izrazili svojo kreativnost, saj so se pri risanju v parih morali ozirati na partnerja. Pri risanju v parih so se prilagajali partnerju. Zanimivejše so se jim seveda zdele slike, ki so nastale pri risanju v paru. Delo v paru je zahtevalo večjo koncentracijo. Pri združevanju v pare se morajo ljudje imeti pod nadzorom. Morajo biti sposobni se prilagoditi, uskladiti mnenje in zastopati svoje interese. Dijaki so menili, da je simulacija zanimivo predstavila programiranje v parih in so se le tega z veseljem lotili, saj jim je predstavljal izziv in nove možnosti za izboljšanje svojega dela, znanja in osebnosti.

3.2 Psihološki pogoji dela

Po uspešno opravljeni simulaciji smo se temeljito pogovorili o psiholoških pogojih dela. Te je izjemno zanimivo predstavil Fulghum (1988): »All I really Need to Know about Pair Programming I Learned in Kindergarten. Share everything, play fair, don't hit people, put things back where you found them, clean up your own mess, don't take things that aren't yours, say sorry when you hurt somebody, warm cookies and cold milk are good for you ...«. Poglejmo si osnova pravila programiranja v parih po Fulghumu (1988), ki smo jih pregledali tudi z dijaki:

- **Vse moraš deliti**

Pri programiranju v parih, morata oba programerja

skrbno pripraviti en izdelek, načrt, algoritem, kodo, test. Oba programerja sta kot povezan, inteligenten organizem, ki deluje kot eni možgani pri vseh vidikih izdelka. Ena oseba tipka oziroma piše, medtem, ko druga neprestano nadzira delo. Oba sta enakovredna udeleženca v procesu. Nihče ne sme niti pomisliti, da bi rekel: «Naredil si napako v načrtovanju», ali pa »Ta napaka je tvoja«. Vedno in povsod mora biti prisotna misel: »Tukaj sva naredila narobe« ali pa »Pravkar se nama je uspelo prebiti skozi test brez napak.«

■ *Bodi pošten*

Pri programiranju v paru, ena oseba »vozi« (ima nadzor nad tipkovnico ali pa načrtuje izgled), medtem ko druga neprestano nadzira delo. Tudi v primeru, ko je eden spretnější in boljši, je pomembno, da obrne »vožnjo«, da se drugi ne počuti izrinjen oziroma nepomemben. Oseba, ki nadzira delo, ni pasivni opazovalec. Nasprotno. Vedno je zaposlen in aktiven.

»Samo gledati nekoga, ki piše program je približno tako zanimivo, kot gledati, kako v puščavi umira trava«, je dejal Beck (2005). Pri programiranju v parih mora oseba, ki ne tipka, neprestano opravljati analizo, načrtovati in pregledovati kodo. Medtem, ko ena oseba tipka, je druga zaposlena na strateškem nivoju – kam vodi ta linija razvoja, ali bo na koncu program delal pravilno? Ali obstaja boljši način? Kako bi to lepše naredila?

■ *Ne škodi svojemu sosedu*

Vedno se je potrebno prepričati, da je partner zbran in z mislimi pri nalogi. Prednost dela v paru je seveda tudi to, da nihče ne zapravlja časa z branjem elektronske pošte, surfanjem po internetu, gledanjem skozi okno, sanjarjenjem. Partner venomer čaka na nadaljevanje razvoja in vsak tudi pričakuje, da bo partner sledil načrtovanemu razvoju. S partnerjem gledata, razmišljata. Ko ima eden občutek, da bo šlo vse k vragu, ima mogoče drugi 'svoj dan'. Tako z optimizmom povleče drugega za sabo.

■ *Pospravi stvari na svoje mesto*

Kako delujejo naši možgani? Izjemno hitro jih v nekaj prepričamo. Če mislimo, da je nekaj dobro, bodo možgani to sprejeli za resnico. Če si govorimo nekaj negativnega, kot je naprimer »Sem slab programer«, nam možgani zelo hitro verjamejo. Vsak, ki želi nadzorovati takšne negativne misli, jih mora postaviti na stranski tir vedno, ko želijo preiti v možgane. Programerji v parih so povedali, da je zelo težko delati z nekom, ki ima veliko nezaupanje v svoje sposobnosti. Programerji z nezaupanjem, naj pogledajo na programiranje v parih kot možnost za izboljšanje svoje spretnosti z neprestanim nadzorom in povratno informacijo od partnerja.

Tudi misli kot so: »Sem zelo dober, zakaj moram delati s to zgubo« ne sodijo semkaj. Nihče od nas, ne glede na to, koliko izkušenj ima, ni nezmotljiv in vsemogočen, zato je vedno dobrodošlo posredovanje nekoga drugega.

Programerji v parih so dejali, da je osupljivo, koliko manj očitnih napak naredi človek, kadar mu nekdo gleda čez ramo.

■ *Ne jemlji stvari preresno*

EGO programerji. The Psychology of Computer

Programming (Weinberg, 1998), je pravi scenarij za programerje, ki ponovno pregledujejo svojo kodo. Ob posebno »slabih dneh za programiranje« je zelo koristno, da se znamo nasmejati svojim napakam, ki jih je lahko zelo veliko. Spet je to, da nekdo ob takih dnevih pregleduje tvoje delo in ti svetuje, zelo koristno. Človeško oko ima zavirljive sposobnosti, da ne vidi stvari, ki jih ne želi videti. Tako svojih napak po navadi ne vidimo.

Partner, ki pa se vedno strinja s partnerjevo odločitvijo, ravno tako ni ravno v veliko pomoč. Za učinkovito izmenjavo mnenj, potrebujemo zdrav razgovor z argumenti.

Potrebno je ravnatežje med prikazovanjem preveč ali premalo ega. Učinkoviti programerji v parih, le to pridobijo skozi prakso in šele po nekem določenem času. Ta čas lahko traja ure, dneve, tedne, odvisno od posameznikov, narave dela in preteklih izkušenj dela v parih.

■ *Če nekoga prizadeneš, se mu opraviči*

Večin programerjev v parih se strinja, da je primeren delovni prostor kritični dejavnik pri uspehu. Programerji morajo biti sposobni sedeti drug ob drugem in programirati, simultano gledati ekran in si deliti tipkovnico in miško.

■ *Otresi se dvomov preden začneš*

Mnogo programerjev se polašča dvom, preden prvič pričnejo z delom v paru. Ne pričakujejo koristi, prednosti ali pa zabave. Združiti dva skeptična programerja zagotovo ne prinese uspeha.

■ *Malica je zdrava in koristna*

Ker programiranje v parih zahteva stalno koncentracijo obeh programerjev, ki se nenehno osredotočata na reševanje problema, je odmor še toliko bolj pomemben. Odmor naj se ponovi periodično pred ponovnim ciklom programiranja v parih. V odmoru se morata oba »odklopiti« od problema in ob ponovnem zagonu bosta lažje in bolj sveže začela. Priporočene aktivnosti med odmorom: Pregled pošte, telefoniranje, brskanje po internetu, malica.

■ *Živimo prijetno življenje*

Komunikacija s soljudmi, je že v osnovi ključ, ki vodi do prijetnega življenja. Večina programerjev bi verjetno rekla, da raje dela samostojno, na mestu, kjer jih nihče ne moti, kot trdi Weinberg (1998), toda po pogovoru s programerji, ki so delali v paru, so le ti bolj dovzetni za menjavo idej in učinkovit prenos informacij. Primer: The Psychology of Computer Programming (Weinberg, 1998) razpravlja o velikem univerzitetnem računalniškem centru. Skupni prostor z veliko zbirko prodajnih avtomatov je bil v zadnjem delu sobe. Nekateri resno-misleči študenti so se pritoževali, da jih moti hrup v skupnem prostoru in odstranili so prodajne avtomate. Po odstranitvi prodajnih avtomatov, so prišle na dan mnoge pomanjkljivosti. Ni bilo več računalniških posvetovanj. Posledica sprememb: neformalni klepet okoli prodajnih avtomatov je pripomogel k velikemu pretoku informacij in prenosu idej med številnimi programerji.

■ *Popoldan si privoščimo odmor*

Gotovo ni potrebno delati samostojno vsak dan. Vendar je okoli 50% izprašanih programerjev dejalo, da je

dobrodošlo 10 do 50% časa, ko delajo samostojno. Mnogi so pri globoki koncentraciji, razmišljanju, logičnemu razmišljanju, raje sami. Večina pa se strinja, da so dobro definirani problemi in mehanično kodiranje učinkovitejši, kadar si lahko programerji izmenjujejo mnenja.

■ **Ko gremo skupaj ven, pazimo na promet in se držimo skupaj**

S programiranjem v parih, postaneta dva programerja eno. Med njima ne sme biti tekmovanja. Obtoževanje za napake programa ne sme nihče nikoli pripisati drugemu. Par si mora zaupati, vsak mora zaupati presoji drugega.

■ **Pazimo se moči dvojnih možganov**

Človek si lahko zapomni in se nauči le omejene količine. Torej, mora se posvetovati, da lahko poveča te omejene količine. Vsak prispeva svoj nabor znanja in izkušenj. Velik nabor znanja in izkušenj, ter spretnosti postane med dvema skupen, kar jima dovojuje učinkovitejšo medsebojno delovanje. Kakorkoli, edinstven nabor izkušenj vsakega posameznika, jima dovoljuje zaposliti se v interakciji, ki spodkoplje njune vire za rešitev naloge.

3.3 Priprava delovnega okolja

Dijaki so se pri vajah razdelili v dve skupini. V eni skupini smo preizkusili programiranje v parih, v drugi skupini pa so dijaki delali samostojno, vsak na svojem računalniku. Pripravili smo učilnico tako, da so dijaki sedeli drug ob drugem. Pri tem smo upoštevali, da vsak potrebuje okoli sebe določen prostor. Pari so se zamenjali po vsaki opravljeni nalogi. Ob končani nalogi smo se pogovorili, odkrivali pomanjkljivosti in prednosti. Rešitve smo primerjali z rešitvami skupine, v kateri so dijaki delali samostojno. Naloge so se razlikovale po obsežnosti glede na učni načrt in predhodno znanje ter izkušnje. V obeh skupinah smo imeli slabše in boljše dijake. Uspešnost metode programiranja v parih smo ocenjevali po naslednjih kriterijih:

- Pomoč učitelja
- Kvaliteta rešitve
- Dokončana rešitev
- Napake
- Testiranje
- Sodelovanje
- Samozaupanje
- Pridobivanje znanja in spretnosti
- Pridobivanje znanja in spretnosti.

3.3.1 Delo v 3. letnikih

V tretjih letnikih se dijaki učijo osnovnih algoritmov in principov programiranja v programskem jeziku Pascal. Vaje potekajo enkrat tedensko v dveh skupinah eno šolsko uro. Vaje so pripravljene v obliki navodil za izvedbo enostavnejšega programa, v katerem uporabijo znane enostavne in sestavljene podatkovne tipe.

Primer naloge:

Za regijsko srečanje mladih raziskovalcev se prijavljajo dijaki z naslednjimi podatki: imenom in priimkom (niz 30 znakov), imenom šole (niz 20 znakov), imenom in



Slika 3. Delovno okolje in pričetek dela.

priimkom mentorja (niz 30 znakov), naslovom naloge (niz 20 znakov) in področjem v katero spada naloga (niz 15 znakov). Napišite program, ki bo omogočal vpis podatkov na datoteko.

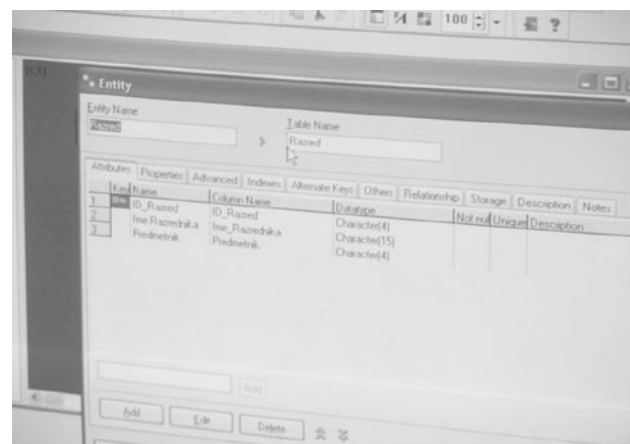
Po opravljeni nalogi smo pregledali rešitve, pripravljene teste, napake, ocenili pomoč učitelja in ocenili sodelovanje udeležencev v parih.

3.3.2 Delo v 4. letnikih

Dijaki so že poznali programska jezika Pascal in Delphi. Spoznali so kaskadni pristop pri razvijanju programske opreme. Vsi so naredili manjšo aplikacijo za delo z relacijskimi podatkovnimi bazami. Pri vajah, ki so dve šolski uri na teden, so razdeljeni v dve skupini. V eni skupini smo preizkusili programiranje v parih, pri eni skupini, pa so dijaki razvijali samostojno ob pomoči učitelja. Učitelj predstavlja naročnika, ki poda zahteve in želje.

Primer elektronske redovalnice:

Vsak učitelj lahko s pomočjo gesla dostopa do elektronske redovalnice tistih dijakov, ki jih uči. Dijaki so razdeljeni po oddelkih. Učitelj ima pri predmetih in



Slika 4: Primer zasnove relacijske podatkovne baze



Slika 5: Delo v paru

dijakih, ki jih uči, poln dostop, kar pomeni, da lahko podatke dodaja, spreminja, briše. V redovalnici so ocene razdeljene po konferenčnih obdobjih, na pisne, usne ocene, ter ocene vaj, projektne dela in drugo. Poleg ocen je mogoče vpisovanje tudi oznake, kot je na primer nms. Poleg učiteljev, ki učijo v posameznem oddelku, ima dostop do vseh dijakov posameznega oddelka in njihovih ocen, tudi razrednik, ki ima za dostop posebno šifro. Razrednik lahko ocene le preglejuje in pripravlja razna poročila. Dostop za pregled svojih ocen pa imajo tudi dijaki s svojo šifro. Pripraviti je potrebno podatkovne zbirke in aplikacijo, ki vsebuje prijavo, pregled, poročila, povpraševanje, vnos...

Vsi dijaki so reševali isto nalogo. Pari so se zamenjali po opravljeni zaključeni celoti.

3.4 Potek dela

Dijaki so se z vso nemo lotili dela. Združevanje v pare je bilo na začetku poljubno. Zame je bil sam pristop združevanja dijakov v pare izjemno poučen. Slabši dijaki so se zbal, da ne bodo rešili problema, vendar smo se pogovorili in delo je steklo. Pri poljubnem združevanju v pare nismo naleteli na probleme, kot sta na primer: »Zakaj moram delati s to zgubo« ali pa »on je veliko boljši, kako mu bom sledil?«. Smo pa opazili prve prednosti. Predvsem so to boljša zbranost in koncentracija, vzpodbujanje drug drugega in prenos znanja. Slabši dijaki so pri samostojnem delu potrebovali veliko moje pomoči, sedaj pa je bilo te pomoči opazno manj. Ob začetku vsakega novega cikla smo se pogovorili in pregledali opravljeno delo. Večina je imela po opravljenem delu zelo dober občutek. S programi so bili zadovoljni in pri delu so se zabavali.

4 Analiza dela

Programiranja v paru smo se lotili v zadnjih dveh mesecih pouka. Po uvodni predstavitvi dela, simulaciji risanja v paru, razgovoru in pripravi delovnega okolja smo se lotili dela. V četrth letnikih je programiranje v parih dalo izjemne rezultate. Pristop dijakov je bil že na začetku izjemno pozitiven. Z velikim interesom so sprejeli izziv. Najbolje

pokaže vzdušje v razredu izjava dijaka: »Projekt sva končala prej, kvaliteta je zelo dobra. Bil sem boljši v programiranju, kot moj sošolec, vendar pa je on predlagal boljšo rešitev. Znal je tudi dobro predstaviti program, ki sva ga naredila. Naučil sem se postavljati vprašanja, če česa nisem razumel in poslušati mnenje drugega. Na začetku sem bil jaz za računalnikom, sošolec pa je nadzoroval moje delo. Bil sem zelo presenečen in užaljen kadar je karkoli rekel, saj sem vedel, da sem boljši v programiranju. Nato pa sem spoznal, da zelo dobro opazuje in ima mnogokrat zelo dobre ideje. Sprejel sem njegove pripombe in potem nama je šlo dobro od rok.«

Pri delu v parih so bili dijaki v večini zelo uspešni in zadovoljni. Učitelj, ki je predstavljal naročnika, je neprestano nadzoroval delo in postavljajl pogoje. Učitelj kot pomočnik in svetovalec je bil potreben le na začetku, ko so se dijaki šele učili in privajali na nov način dela. Pri programih, ki so nastajali, smo neprestano preverjali kvaliteto, testirali in upoštevali dodatne zahteve. Izmed petih parov so štirje oddali dokončane rešitve, ki so v veliki meri upoštevale vse zahteve. Rešitve so imele lepe in prijazne vmesnike. Napak je bilo malo, oziroma smo jih odpravili že v fazi razvoja. Sodelovanje med dijaki je bilo odlično, problem se je pojavil le, kadar je nekdo izostal od pouka. Odločili smo se, da v tem primeru dijak, ki je brez para, razvija samostojno naprej, vendar o vsem svojem delu obvesti svojega partnerja in se naknadno z njim posvetuje. Par, ki ni dokončal svoje naloge v celoti pa je kljub temu vsaj delno opravil nalogo. Para, ki ni opravil naloge, ni bilo. V drugi skupini so dijaki delali samostojno. Le eden izmed dijakov je oddal program, ki je v celoti upošteval naše zahteve. Šest dijakov je oddalo delno delujoče programe s številnimi napakami in nezanimivimi vmesniki. Štirje dijaki niso naredili delujočega programa. Ob koncu smo se pogovarjali o metodi in občutkih. Dijaki, ki so delali v parih, so bili izjemno zadovoljni, imeli so dober občutek, dokončali so nalogo, veliko so se naučili in se pri delu še zabavali.

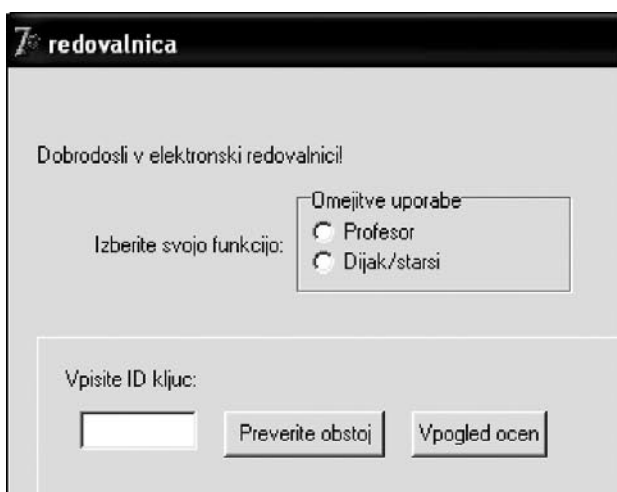
V tretjem letniku programiranje v parih ni tako dobro uspelo. Dijaki so se tudi tukaj razdelili v dve skupini. V eni so se formirali v pare, v drugi so delali samostojno. Naloge so bile, glede na njihovo znanje, enostavnejše. Razvojni cikli so bili zaradi enostavnejših nalog mnogo krajši. Po vsaki nalogi smo zamenjali skupini, v kateri so bili pari. Tudi pari so bili za vsako nalogo drugače zasnovani. Prvo nalogo so dijaki opravili sledeče:

	Pari (6 parov)	Samostojno (14 dijakov)
Kompleten program	2 ali 33.3%	1 ali 7%
Delno delujoč program	3 ali 50%	5 ali 36%
Ne delujoč program	1 ali 16.7%	8 ali 57%

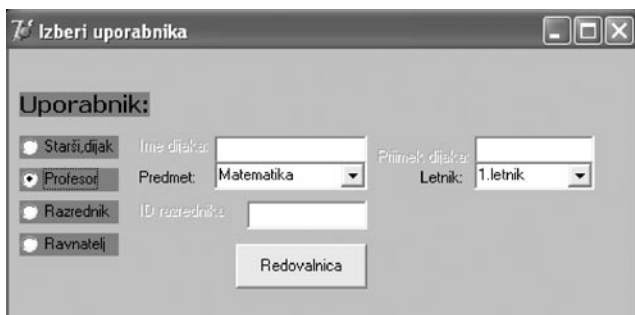
Po drugi nalogi, ko sta se skupini zamenjali, smo dobili naslednje rezultate:

	Pari (7parov)	Ne pari (12 dijakov)
Kompleten program	2 ali 28.5%	2 ali 17%
Delno delujoč program	3 ali 43%	6 ali 50%
Ne delujoč program	2 ali 28.5%	4 ali 33%

Pri dijakih v tretjem letniku je bilo čutiti do dela v parih več odpora. Ker so se pari menjali, naloge pa so bile krajše, so bili potrebni neprestana koncentracija, prilagajane in samokontrola. Kljub odporu dijakov in pogostejšega



Slika 6. Zaslonski sliki delujočega programa.



Slika 7: Primer pogovornega okna delujočega programa

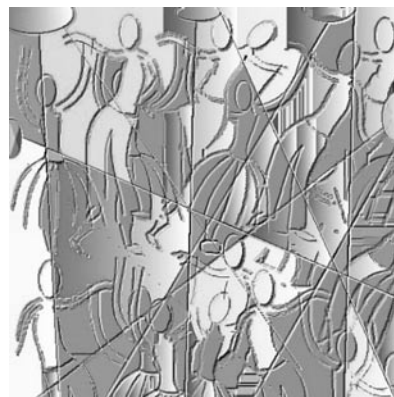
posredovanja učitelja so vidni boljši rezultati dela v parih. Verjetno pa so odporu dijakov botrovali krajši cikli in neprestano menjavanje metode dela.

5 Zaključek

Metodo programiranja v parih smo preizkusili v zadnjih mesecih pouka v tretjih in četrth letnikih tehniške gimnazije pri pouku računalništva. Metoda je zahtevala veliko priprav učitelja, skrbno načrtovano delo in dobro izbrane naloge. Rezultati, ki jih je prinesla, so zelo dobri. V pouk je prinesla svežino in dinamiko. Pri dijakih, ki so bili negativno usmerjeni, je bilo potrebno več dela učitelja, vendar je metoda kljub temu pokazala prednosti pred samostojnim delom. Menimo, da so k odporu dijakov pripomogli krajši cikli in pogosto menjavanje parov. Metoda je izjemna in uporabna za praktično delo pri vseh predmetih. Seveda se moramo metode najprej naučiti učitelji. V času, ko sta pri pouku zelo pomembna medpredmetno sodelovanje in integriran pouk, pa je vpeljava te metode izjemo koristna. Pri pouku računalništva na tehniški gimnaziji bomo s pomočjo programiranja v parih utrjevali znanje, pridobivali nove spretnosti in razvijali teamsko delo.

Literatura in viri

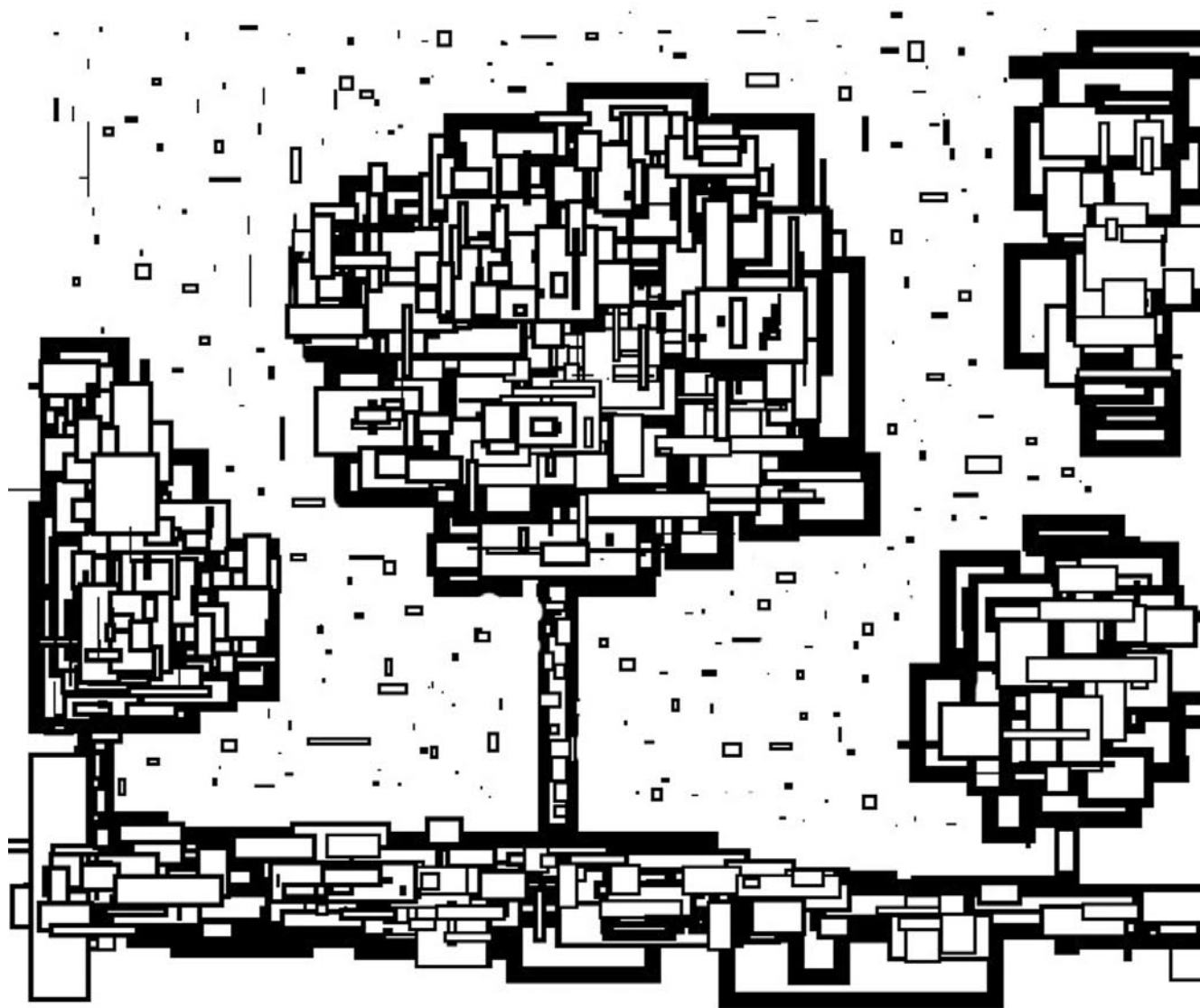
- Manifesto for Agile Software Development (2001), dostopno na <http://agilemanifesto.org/> (29.9.2006)
- Beck K. (2005). *Extreme Programming Explained*, Addison –Wesley
- Cunningham & Cunningham, Inc. (2006). Programming In Pairs Testimonials, dostopno na <http://www.c2.com/cgi/wiki?ProgrammingInPairsTestimonials> (28. 9. 2006).
- Fulghum R. (1988). *All I Really Need to Know I Learned in Kindergarten*, Villard Books, New York
- Jeffries R., (2001). What is Extreme programming, dostopno na <http://www.xprogramming.com/xpmag/whatisxp.htm> (28. 3. 2006)
- Kerievsky J. (2004). PairDraw, dostopno na <http://industriallogic.com/games/pairdraw.html> (28. 9. 2006)
- Randall, W. J. (2003). A Pair Programming Experience, dostopno na <http://www.stsc.hill.af.mil/crosstalk/2003/03/jensen.html> (28. 9. 2006)
- Weinberg, G.M. (1972). *The Psychology of Computer Programming*, Silver Anniversary Edition, New York
- Williams, L. (1999). Pair Programming questionnaire, dostopno na <http://limes.cs.utah.edu/questionnaire/questionnaire.htm> (28. 9. 2006)



Avtor: Sandra Tolić, 13 let
mentorica likovne vzgoje: Natalija Orlič
Mentor računalništva: Sonja Malnarič
OŠ Mirana Jarca, Črnomelj

Gabrijela Krajnc je diplomirala na Fakulteti za elektrotehniko in računalništvo Univerze v Ljubljani, trenutno je študentka podiplomskega študija Informacijski sistemi in odločanje na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Poučuje računalniške predmete na Tehniškem šolskem centru v Kranju. Sodelovala je pri prenovi programov tehniške gimnazije in srednjega tehniškega izobraževanja na področju računalništva. Je članica kurikularne komisije za strokovne predmete na tehniški gimnaziji na zavodu za šolstvo. Je članica republiške predmetne komisije za računalništvo za splošno maturo, zunanja ocenjevalka in izvedenka za pritožbe na maturi.

Viljan Mahnič je izredni profesor in prodekan za pedagoško delo na Fakulteti za računalništvo in informatiko Univerze v Ljubljani. Od leta 2000 je predstojnik Laboratorija za tehnologijo programske opreme, kjer se ukvarja se z s posebnim poudarkom na informacijskih sistemih. Od leta 1996 je predstavnik Slovenije v EUNIS (European University Information Systems Association), od leta 2002 pa tudi član sveta direktorjev omenjene organizacije. Poleg tega aktivno deluje na področju računalniškega izobraževanja, je član Republiške predmetne komisije za predmet Računalništvo in član Programskega sveta za informatizacijo šolstva.



Avtor: 6. razred, 12 let
Mentorica likovne vzgoje: Natalija Orlič
Mentor računalništva: Sonja Malnarič
OŠ Mirana Jarca, Črnomelj

the classroom but it should also support the understanding of the chemical processes. The goal of the research is to check capability of acknowledgement and understanding of changes, which were observed by the pupils on the three different multimedia footage of the chemical experiments; main goal is to determine the added value of added visual elements (names and formulas of reagents, equation of the chemical reaction) in multimedia footage of the chemical experiments and school success of the pupil on the capability of perception and the proper understanding of it.

Key words: experiment, visual literacy, perception, explanation, school success

**Eva Jereb,
Igor Bernik**

Students' Opinion about Electronic Examinations before and after E-testing

This paper is about one of the essential matters in electronic learning: taking electronic exams. It presents students' opinion about electronic examinations before and after electronic testing. The studies in the years 2004 and 2005 confirmed that the majority of participants were prepared to take electronic exams. They were enthusiastic about the immediate feedback and time and place flexibility. However they had some reservations about the technological issues. Motivated by the positive students' response we performed a pilot e-testing. After the testing we checked the students' opinion again. The majority was enthusiastic and even more certain in introducing e-exams. Some of them think that this kind of taking exams is possible but they still do not see any advantages in it.

Keywords: e-learning, e-examination, students' opinion, e-testing tool Perception

**Bogdana Borota,
Andrej Brodnik**

Learning Music with ICT Technology

Currently ICT offers teacher many opportunities to improve processes of learning and teaching (of music). The results of research indicate that the successful inte-

gration of technology depends on teacher's competencies, on educational standards of music and technology, and on designed strategies of modeling, implementing and following. We designed a flexible software application for music teaching that permits use of some means of contemporary learning, such as: (1) differencing particularly based on individualization; (2) design of self learning strategy; (3) problem based and constructivist learning; (4) possibility of achieving of higher cognitive and connotational goals; (5) possibility of learners to participate in virtual community. The architecture we used was a standard server based architecture, where the server has a triple role: (1) provides the necessary software; (2) storing settings and learners' portfolio; and (3) provides a medium for the exchange of messages between the learners forming a virtual society.

Key words: ICT, music, primary school, strategies

**Gabrijela Kranjc,
Viljan Mahnič**

Programming in Pairs in High Schools

Extreme programming (XP) is a new style of software development focusing on excellent applications of programming techniques, clear communication, and team work, which gives unimaginable results. A major practice of Extreme programming is Pair Programming. There are two programmers working side by side at the same computer, collaborating on the same analysis, the same design, implementation and test. Proponents of pair programming argue that programs produced by pairs are of higher quality, with less errors, better design than those produced by one programmer. And they are made in the shortest time possible as well. We think that pair programming model has also been found to be beneficial for students. Initial quantitative and qualitative results demonstrate that the use of pair programming in the computer science classroom enhances student learning and satisfaction. We explore the nature of pair programming, then examine the ways such a practise may enhance teaching and learning in computer science education.

Keywords: agile methods, extreme programming, pair programming, quantitative and qualitative results.

Zvone Balantič

Software Spiral Development as the Continuing e-Health Process

The creative process can help to create software spiral development, where beside the settled development of an idea we have a large concentration of flashes of wit and intuition. E-teaching models that take into consideration given facts of present time are building in a high level of information technology (IT). It can be said that in this area as well the life expectancy of an e-product is getting shorter and needs constant improvements. The e-health process has to be constant, adjusted, growing and set in the newest theoretical happenings and practical realizations. Software spiral development is very dynamical regarding the education in medicine. With the spiral model our work can be structured very clearly through next steps: analysis, specification, design, implementation, testing, integration and maintenance. The software spiral development process was presented on the example of e-materials for the respiratory physiology. The construction of the spiral model makes IT in medicine clearer and more effective. In the final phase the IT grows into the Health Life-Style Portal for general public and into Professional Health Portal for professional and expert public.

Key words: spiral development, software, multimedia, lung function, education, e-health