РОЛЯ И МЯСТО НА ПРОГРАМИРАНЕТО В ОБУЧЕНИЕТО ПО ИНФОРМАТИКА, КАТО ЧАСТ ОТ СЪВРЕМЕННИТЕ МЕТОДОЛОГИИ ЗА СЪЗДАВАНЕ НА СОФТУЕР

Христо Христов*, Антоанета Христова**
** ФМИ при ПУ „Паисий Хилендарски" гр. Пловдив, п.к. 4003, бул. България 236*
e-mail: hth@uni-plovdiv.bg; hristo.toshkov@gmail.com
*** ПМГ „Иван Вазов гр. Димитровград, п.к. 6400," ул. Климент Охридски 1 e-mail: pmg@escom.bg*

ROLE AND PLACE OF PROGRAMMING IN TEACHING INFORMATICS AS PART OF CONTEMPORARY METHODOLOGIES FOR CREATING SOFTWARE

Hristo Hristov*, Antoaneta Hristova**

ABSTRACT

This work explores the teaching of programming languages in informatics lessons. Their role and place in training have been analyzed as a part of the software process.

Keywords: *programming language, teaching methodology, software development process*

1. Въведение.

Характеризирането и оценяването на програмирането е въпрос от тясно научен характер. Съвременната преценка за неговата настояща и бъдеща роля и място в средното образование обаче, е свързана както със специализация на обучаващия се в областта на компютърните науки, така и с развитието на личностни качества в условията на формиращото се информационно общество. Взаимовръзката между придобиването на *технологични познания* и изграждането на *информационна култура* е в основата на пораждане на интелектуален капитал, който да създава нови информационни продукти, услуги и технологии, в полза на развитието на информационното общество.

Ако приемем условно, че информационните технологии (ИТ) движат развитието на информационното общество (ИО), то може да кажем, че разработването на софтуер е двигателят на това развитие. Затова подготовката и придобиването на базови познания в средното образование за съвременните методи и практики на разработване на софтуер, без непременно те да са тясно специализирани, е основополагащо в обучението на развиващия се ученик, предвид информационния век, в който живеем.

2. Програмирането в средното образование през годините.

Умението да програмираш не е единственото умение, което се изисква от компютърния специалист. Това не е достатъчно, за да се създаде една добре работеща програма. Независимо от това дейността програмиране е централна в компютърната наука [1]. Нейното практикуване, като обсег и концепция, зависи от характеристиките на езика за програмиране.

Програмните езици са се развили значително от края на 50-те години на миналия век, когато са били създадени първите езици на високо ниво - FORTRAN и COBOL[2]. Понастоящем има документирани повече от 2500 езика за програмиране[3]. Голямото множество езици за програмиране при изследване на програмирането поставя въпроса за тяхната категоризация и класификация. В научната литература съществуват различни критерии за групиране на езиците за програмиране - стил и парадигма на програмиране, независимост от архитектурата на компютъра, производителност, преносимост, многонишковост, сигурност, динамичност и др.

В [5] авторът, след като уточнява широкия смисъл на понятието парадигма, посочва, че съществуват четири вида парадигми на програмиране. Обектно ориентираното програмиране

е сравнително нова програмистка парадигма. Други такива са: императивно-програмната (реализирана в езиците C и Pascal), функционално-програмната (реализирана в езиците FP, Lisp, Haskell, ML), логическо-програмната (реализирана в езика Prolog)[4]. В училищното образование място са намерили езиците с общо предназначение, които спадат към първите две парадигми.

Изучаването на програмни езици и програмиране в гимназиалния етап на българската образователна система се извършва в часовете по информатика. Според [5] проблемът за внедряването на предмета информатика в средното училище има няколко относително самостоятелни аспекта, сред които особено място заема езикът, на който се програмира. България е била една от първите държави, които въвеждат изучаването на информатика, в частност обучението по програмиране в средните училища. През 60-те години в много окръжни градове са се създали т. нар. "математически" паралелки. В тях учениците са изучавали предметите "Програмиране" и "Числени методи"[6]. В началото са се учили по два езика: Минск- 2, Алгол 60; Минск 22, Алгол 60; Туфак, Фортан IV[5]. През 70-те години учениците са обучавани на АЛГОЛ, ФОРТРАН, КОБОЛ, а по късно и на БЕЙСИК. В средата на 80-те с решение на просветното министерството информатиката става отделен общообразователен учебен предмет. Учебната програма е била съобразена със съществуващото положение в страната: широко разпространение на микрокомпютъра "Правец 82" и езика Бейсик[6]. След това, през 90-те е започнал да се изучава езикът Паскал, а няколко години по късно постепенно са започнали да се използват и езиците C и C++. През последните години в някои гимназии с профил „информатика" се провеждат експерименти в обучението по обектно-ориентирано програмиране, като предпочитани езици са Java и C#. Такъв експеримент, преподаване на обектно-ориентирания език за програмиране Java за профил „информатика", в ПМГ „Иван Вазов", гр. Димитровград, авторът е провел през учебните 2006/2007 и 2007/2008 г.

В Таблица №1 са показани по години повечето от изучаваните езици за програмиране в средното образование. Съпоставяйки езиците от таблицата на графиката[7], която показва историческото развитие на утвърдените и широко приети езици за програмиране, и извършвайки сравнителен анализ между двете схеми, правим извод, че: *в гимназиалния етап на българското общо образование преподаването на програмиране и изучаването на езици за програмиране е било на високо технологично ниво през годините и в синхрон с тенденциите, които са диктували развитието на компютърните технологии*.

Година	Език за програмиране
60-те	Минск, Алгол, Туфак, Фортан
70-те	Алгол, Фортран, Кобол, Бейсик
80-те	Бейсик, Лого, Паскал
90-те	Паскал, ИНФО, C, C++, Visual Basic
След 2000	Паскал, C, C++, Visual Basic, Java, C#, … ?

Таблица №1 Езици за програмиране изучавани в средното образование

3. Проблеми в обучението по програмиране и перспективи пред училищната информатика.

Независимо от технологично актуалното преподаване на програмиране и изучаването на съвременни езици за програмиране, проблемите в училищната информатика през последното десетилетие се задълбочават. Една част от тях са повлияни от социални явления - *слаба мотивация на ученика, недостиг на учители по информатика, ниско равнище на придобита квалификация по програмиране и практики за създаване на софтуер* и т.н.; друга част имат научен характер - *изучаването на стилове и езици за програмиране не се разглежда в техния контекст - създаването на софтуер посредством прилагане на процес; няма разработени методики за преподаване на методологии за създаване на софтуер* и др.;

най-проблемна обаче е *липсата на национална концепция и оценка за факторите, които играят стратегическа роля за развиването на софтуерните технологии.* ***Днешният ученик, след 5 или 10 години, с какви познания ще присъства в информационното общество - само потреблението на технологии и информация ли ще е характеристиката на неговото обучение, или той ще бъде част от създаването, развиването и внедряването на нови технологични решения?*** За разлика от много други научни области, компютърната информатика притежава потенциал и инструментариум за разрешаване на множество въпроси в сферата на администрацията, икономиката, правото, медицината, физиката, химията, дори и на математиката. За целта обаче е необходимо да се прозре, че не само *потреблението* на ИТ, но преди всичко *производството* на технологии и намирането на нови софтуерни решения допринася за развиването на споменатите области. Поглед от тази гледна точка показва колко е фундаментална за съвременното общество компютърната информатиката, в частност училищната информатика, и колко е необходимо да се оценят факторите, които развиват ИО. Един такъв фактор е *изграждането на интелектуален капитал (знания, умения и способности за вземане на решения, които да автоматизират различни по характер социални дейности и нужди чрез средствата на компютърната информатика) за създаване на нови технологии и решения.* Именно концепция, в която се залага на подготовка на ИТ специалисти, движещи развитието на ИО, чрез създаването на информационни технологии, е нужно да се заложи като фундамент в обучението по информатика.

4. Софтуерният процес в обучението по информатика.

Историята на програмирането е кратка, само няколко десетилетия, но развитието му е изминало дълъг път. Ако през първите няколко десетилетия плод на програмирането е била програмата, то през последните десет-петнадесет години *програмирането от писане на програми се трансформира в имплементация на дизайни* - продукт на което са независими един от друг класове, модули и компоненти с изходен код за многократна употреба. Основна роля за това е изиграла смяната на парадигмата на програмиране от структурен на обектно-ориентиран подход на програмиране. През 80-те години обектите са започнали да излизат от изследователските лаборатории и да правят своите първи стъпки в „реалния“ свят. По това време различни хора са започнали да мислят за обектно-ориентирани езици с графичен дизайн. Като резултат, от група методи за обектно-ориентиран анализ и дизайн е израснал унифицираният език за моделиране (Unified Modeling Language, UML). До известна степен всички методи са смесвали език за *графично моделиране* с *процес*, който описва как да се подходи към разработката на софтуер[8]. Обикновено има два проблема, които се отнасят за осъществяването на процесите. Първият, разработчиците трябва да подберат методи и практики за реализацията на софтуерната разработка. Вторият, че екипите трябва да определят как да приложат тези методи и практики по време на жизнения цикъл на проекта. Затова те имат нужда да приложат процес на разработка[9].

В наши дни разработчиците не знаят предварително аспектите на софтуерните проекти. Всяко изграждане на система има неопределени елементи по времето на прилагане на процеса. Клиентите нямат ясна представа за желаните от тях решения, разработчиците не са запознати с бизнес нуждите на клиентите, дори бизнес условията по време на разработката се променят. Софтуерният процес помага на разработчиците да се справят с най-честите причини за неуспех на софтуерните проекти като подобряват успеваемостта, качеството на решенията и влиянието върху бизнес намеренията, с цел намиране на решения, които да посрещат нуждите на клиентите. Процесите са се появили като средство за справяне с непрестанно нарастващата сложност на софтуерните разработки.

Има ли място софтуерният процес, неговите методи и практики, в обучението по информатика?

От една страна, съгласно списъка на професиите за професионално образование и обучение[10] в областта на информатиката, с направления *компютърни науки* (код 481) и *приложна информатика* (код 482) се подготвят професионалисти: *програмист*, *системен програмист* и *икономист-информатик*, съответно в специалности: *програмно осигуряване, системно програмиране* и *икономическа информатика*. Практикуването на всяка от тези специалности предполага задълбочени познания за методите и практиките на създаване на софтуерни системи. От друга страна, в съвременните софтуерни процеси, програмирането е само една от дейностите, която обединява различни софтуерни практики за изграждане на системи. В училищната практика обаче, всеки учител е запознат, че учениците и понастоящем се обучават да инструктират компютъра чрез език за програмиране, а не посредством програмирането да се имплементират абстракции на дизайни. Накратко, на обучението по програмиране в училище му липсва контекст, т.е. дейности като извличане и управляване на изисквания, извършване на анализ, избор на архитектура и изграждане на дизайн, оценяване на рисковете от неуспех, управление на обхвата на разработката, актуализиране на промените в изискванията към софтуера по време на работа и т.н. Това отделяне на обученото по програмиране от естеството на процеса на изграждане на софтуер лишава обучаващия се от придобиването на знания и умения, различни от дейността програмиране. По този начин за учениците реализацията на софтуерни проекти и изграждането на софтуерни системи остава енигма. От трета страна, в държавните образователни изисквания (ДОИ) за учебно съдържание на културно-образователна област, към която спада информатиката, въпросът за контекста на обучението на програмирането не е засегнат. В нормативните документи[11] изискванията към ученика за придобиване на общообразователен минимум от знания и умения, що се касае до създаването на софтуер, са пряко свързани с практики по програмиране. Видимо е, че обучението по програмиране не е достатъчно, за да се подготвят професионалисти в специалности с приложно и системно програмиране.

От направения анализ става ясно, че софтуерният процес, неговите методи и практики имат място в обучението по информатика. Считаме за удачно въпросът за мястото на софтуерния процес в обучението да се разглежда, като се разделя на: теоретична част с познавателен характер, която е по силите на всеки ученик; и практична част, в която да се изучава прилагането му. Разбира се, не бива да се забравя, че има разлика между професионални практики за създаване на софтуер и такива за обучение. Затова, понастоящем, изучаването на методите и практиките на софтуерните процеси са в сферата на педагогическите и методически експерименти.

5. Перспективи и решения.

Отворени за дискусия остават въпроси свързани, с избора и адаптацията на методология за създаване на софтуер, подходяща за обучение в училище, както и създаване на методика за нейното преподаване. Едно такова решение, подходящо за провеждане на експерименти за целите на средното образование и удачно за преподаване на съвременни софтуерни практики, е проектът Eclipse Process Framework[12], разработван от общността Eclipse и поддържан от IBM. Според авторите проектът е подходящ за целите на висшето образование, първо, за да се преподават най-добрите софтуерни практики, съгласно учебните планове на университетите, и второ, да се усъвършенства развитието на тези практики[13]. Въпреки това, той лесно може да се адаптира и за целите на средното образование. Интегрирана среда за моделиране с графична и текстова нотация - Eclipse Process Framework Composer[14] е лека за работа и по силите на средношколците. Екосистемата на проекта предлага богат набор от софтуерни практики с отворен код, достъпни от интернет страницата и интегрираната среда за разработка на проекта. Основното предимство на тази платформа е

подборът на голям набор от методи, практики, техники, съветници, шаблони, пътни карти и други средства за създаване на софтуер.

6. Заключение.

Днес програмирането е неизменима част от процесите за изграждане на софтуерни системи, а употребата и практикуването му обединява множество софтуерни практики.

В съвремието ни информатиката, като наука за автоматична обработка на информацията, е изправена пред предизвикателството - не само да обучава учениците в писане на програми, но и да им покаже и научи как се реализират софтуерни проекти. Тяхното прилагане включва обединяването и автоматизирането на множество бизнес решения и социални дейности. Интегрираното изучаване на тези дейности като част от софтуерния процес има място в обучението по информатика.

Относно въпросите кои и какви методи и практики на софтуерния процес е необходимо да се изучават в часовете по информатика? – еднозначен отговор може би не би могъл да се намери, и все пак, наложително е образованието да се придържа към новостите, тенденциите и нуждите на пазара. Това ни кара да считаме, че изучаването на софтуерни процеси има място в обучението по информатика.

Цитирана литература:

1. Хорстман, Г., 2000. Принципи на програмирането със C++. II., ИК СОФТЕХ, София.
2. Fischer, A., F. Grodzinsky, 1992. Anatomy of Programming Languages. Prentice Hall, USA.
3. O'REILLY-Media, 2005. The History of Programming Languages, http://oreilly.com/pub/a/oreilly/news/languageposter_0504.html, последно посещение на 14.03.2012.
4. Тодорова, М., 2011. Обектно-ориентирано програмиране на базата на езика C++, Сиела, София.
5. Атанасов, К., 2010, Състояние и проблеми на изучаването на предмета информатика в средните училища, Сборник доклади на Национална конференция: „Образованието в информационното общество", Пловдив.
6. Гъров, К., 2000, Обучението по информатика и информационни технологии в средното училище - състояние и перспективи, Доклади на Юбилейна научна сесия - 30 години ФМИ на ПУ "Паисий Хилендарски", Пловдив, стр. 28-37.
7. Diagram of programming languages history, http://rigaux.org/language-study/diagram-light.png, последно посещение на 10.03.2012.
8. Фаулър, М., 2004, UML основи, СофтПрес ООД, София.
9. Haumer, P., 2007, Eclipse Process Framework Composer - Part 1: Key Concepts, Eclipse Process Framework Project, http://www.eclipse.org/epf/general/EPFComposerOverviewPart1.pdf , последно посещение на 03.04.2012.
10. Официален сайт на МОН, Професионално образование, Списък на професиите за професионално образование и обучение, http://mon.bg/top_menu/vocational, последно посещение на 23.03.2012.
11. Официален сайт на МОН, ДОИ, Приложение№3, http://mon.bg/top_menu/general/doi, последно посещение на 23.03.2012.
12. Официален сайт на фондация Eclipse, проект Eclipse Process Framework, http://www.eclipse.org/projects/project.php?id=technology.epf, последно посещение 11.04.2012.
13. Официален сайт на фондация Eclipse, Who will benefit from EPF, http://www.eclipse.org/epf/general/getting_started.php, последно посещение на 11.04.2012.
14. Официален сайт на фондация Eclipse, EPF Composer,
15. http://www.eclipse.org/epf/downloads/downloads.php, последно посещение на 11.04.2012.