Using Educational Programming Languages to enhance teaching in computer science

Dr. David Smith, The Austrian Federal Ministry of Education, Austria
Dr. Petros Lameras, Ellinogermaniki Agogi, Greece
Mr. Nektarios Moumoutzis, Technical University of Crete, Greece
Abstract

This paper examines whether embedding the use of an ‘Educational Programming Language’ (EPL) in a viable learning process can be effectively applied to the study of computer science in schools. Utilising technology to enhance learning and motivate children remains at the forefront of financial and strategic planning as the tiers of the education community research trial and process various methodologies to effectively enhance learning. Integrating technology in the school curriculum satisfies many demands from requiring schools to be more effective (Afshari et al., 2007) to investing in a country’s knowledge future (Venezky & Davis, 2002). Yet technology as an add-on has been haphazard in its effect depending more on individuals for successful adoption in schools and has led organisation to start using technology within sound pedagogical frameworks to realise effective practice. Technical adoption in a pedagogical context has a profound impact on learning (Straub, 2009) and reflects the purpose of this paper as it examines the impact of a European project on the study of computer science in schools.

The project (pSkills) explores the use of EPLs within a pedagogical framework that support the adoption of inquiry-based and problem-based learning for enhancing students’ programming skills. One of the main challenges facing computer science teachers in the current curriculum is the level of complexity due to the abstract nature of programming languages. The current computer science strategy has limited the development and hence availability of pedagogic approaches and assistance to computer science teachers. This scenario contributes to teachers’ limited knowledge about pedagogical types and their uncertainty about the most appropriate pedagogy to deploy for a curriculum theme. Arguably used in isolation the EPL, as with other technologies, has a limited appeal for students, short-lived due to its disconnectedness with a curriculum purpose. However, it has been found that teachers can encourage students to inquire and research by experimenting and utilising technology in a pedagogical context that incorporates models which foster autonomous, independent learning, where learners are able to work individually and in groups as well as applying their knowledge into real-world simulations. The development of a pedagogically rich approach to use EPLs for enhancing computer science teaching is at the forefront of the pSkills project. This paper examines the results of a quantitative survey completed by computer science teachers across four countries about using EPLs to enhance student learning within a pedagogical framework. Further to this, the paper outlines the criteria for the selection of EPLs and along with the survey results, details their combined impact on the development of learning scenarios and the design of a recommended computer science curriculum. A discussion of the training framework to incorporate the learning scenarios concludes this paper with some implications for the way forward.
1.0 Introduction
Using information communication technologies in education has generated much debate from the installation of hardware and software solutions to the utilisation of technology. There are a myriad of issues challenging governments, education institutions and communities such as filtering Internet browsing, using particular types of software or the adoption of one to one computing platform that emanate from an economic and learning viability. At the heart of these debates and challenges is the desire/demand to maximise the learning opportunity to prepare, perpetuate current and future generations for the knowledge society.

This paper examines whether changes in the learning process applied to the study of computer science in the school curriculum can effect change to student interest, understanding and motivation in the study of this subject. Utilising technology to enhance learning and motivate children remains at the forefront of financial and strategic planning processes for governments and in particular their education sectors, learning pedagogy remains at the fore of discussions surrounding effective practice. The pSkills project combines technology integration with learning pedagogy to create a learning scenario (platform) to provide not only a structure for the use of technology but also a means to effectively engage students.

This paper details some of the pSkills research that examines the viability of utilising learning theory such as Papert’s constructionism (Conole et al., 2004) in the delivery of computer science education. The paper commences with an explanation about the current state of computer science education, particularly in secondary and higher education. The snapshot of computer education practice shows that it is a contributing factor to the decline in the numbers of students studying this subject both in schools (Clark & Boyle, 2006) and in the higher education sector. Reversing the decline of student numbers through changes in education practice forms one of the main aims of the pSkills project and is described in the next section.

The revolution that is Web 2.0 has changed more than just the availability of software, it questions the current thinking on computer literacy, computer competency and computer practice in today’s education climate. Learning practices in schools reflect the learning models from previous generations, the perpetuation of an obsolete model (Carroll et al., 2005) that ignores the real-time circumstance and needs of the students and teachers. This paper describes the survey and findings of computer science teachers about current school practices and the development of a pedagogical learning practice to address the need for a contemporary style of learning. The learning practice (scenarios) reflects the use of learning pedagogies such as constructivist theory that complement the teaching of computer science as well as the use of educational program languages (EPL) to be used in conjunction with the pedagogy. The process used to select the EPLs is detailed along with the training framework designed to support and mentor the teachers in the implementation of these new learning scenarios.
Finally this paper will conclude with a discussion of the anticipated outcomes of the implementation of the learning practice in the classroom. Some of the anticipated benefits of adopting a new learning approach are also discussed as well as disseminating the findings of the pSkills project in a broader educational context for further research.

2.0 The current status of computer science

Computer science draws its foundations from a wide variety of disciplines (Savage, 1998). Computer science is at the same time a mathematical, scientific and engineering discipline that has inherent social, cultural, and ethical issues (ACM, 1991; 1997). In that sense, computer science is an empirical discipline that seeks to discover new phenomena as well as to explain and analyse phenomena that already occur. Computer science is widely prevalent in Higher Education putting emphasis on technical as well as in theoretical aspects on how computers may be used by the society; and for scientific purposes aiming to enhance students’ professional career. However, in schools computer science education remains a fragmented, complex and misunderstood subject area, often being integrated as a technological tool for facilitating the teaching and learning of other non-computing subjects. In computing education the terms ‘information technology literacy’ and ‘information technology fluency’ often are used interchangeably to denote the ability to use technology in one’s own field as well as the capability to learn and use technology as it evolves (Tucker et al. 2006). While there is a logical transition from school to university level in other subjects such as physics, mathematics and language arts, children do not possess the necessary skills and knowledge for apprehending and understanding the philosophical and practical foundations of computer science needed for being able to build upon prior knowledge and understand the ubiquitous meanings that will be taught in university. Schools, universities and professional organisations seem to recognise this problem and have taken measures to re-structure computer science curriculum.

Most of the computer science topics are characterised by the increased level of abstraction which may cause certain difficulties in teaching and learning (Hagan and Sheard, 1998). For this reason, the computer itself may be used as a tool in the teaching and learning of computer science. For example, educational software in the form of simulations, microworlds and other multimedia are regarded as appropriate for teaching and learning in computer science. Through these teaching methods students have the opportunity to practice their skills in the laboratory and to gain practical experience in a variety of topics such as programming, hardware and the use of particular software (Hagan and Sheard, 1998). The Web also can be used as a tool for teaching and learning in computer science for accessing a vast amount of related subject information, and to use software tools beyond the classroom context such as Web2.0 tools. There have been recent studies that have made an attempt to adopt more student-led approaches to teaching in computer science subjects. For example, Cox et al. (2008) described and evaluated an information management course from an

Deek and Kimmel (1999) suggested some important progress indicators for providing professional recognition to the teachers and to the discipline. These are:

- Content standards for computer science education need to emulate relevant standards that already occur in science, mathematics and other disciplines.

- Educational departments must recognise the discipline of computer science for establishing teacher certification requirements in the discipline.

- Teacher training programs should address all teaching and learning aspects that are necessary for teachers to be able to teach complex and abstract computer-based subjects. Teachers already in the school systems must be trained appropriately to obtain relevant certifications.

The majority of computer science teachers may have studied computer science but some of them come from other ‘hard’ disciplines such as mathematics, chemistry, physics and engineering (ACM, 1991). While computer science teachers’ aims include helping students to know the field well and preparing them for their professional career, they adopt different teaching approaches for achieving these aims. Kordaki and Komis’s (2000) study of the epistemological views of teachers about Computer science showed that Greek teachers adopted different teaching approaches based on the nature of the taught course. For example, teachers that were teaching algorithms and operating systems courses emphasised a teacher-centred/content-oriented approach to teaching in terms of presenting information verbally related to theoretical rules and examples. Similarly, teachers who taught programming languages had a teacher-centred/content-oriented approach that included the presentation of syntactical rules of particular commands. Teachers who taught more open-ended/socially-focused subjects such as data bases and multimedia started to use more interactive approaches by initiating discussions and debates about particular applications. However, there were teachers who expressed their insecurity about group discussions and group projects. In particular, teachers expressed that they did not have the appropriate knowledge to manage projects that included research and data collection and they were also concerned about managing groups working together. Finally, teachers who taught general purpose software courses hold more student-centred approaches to teaching allowing students to collaborate in group-projects. Teachers also believed that discussions and collaborations are essential for conducting the course leading to a more student-centred environment.

Using the abstract nature of teaching in computer science as a premise, there may be several opportunities and challenges to the extent to which options about
computer science curricula can be offered to students and still be applicable for
the teacher and the institution. When the student is given more opportunities and
choices especially in more social-based computer subjects, the teacher needs to
increasingly respond and individualise rather to just deliver and plan. Therefore,
by adopting an effective and flexible computer science curriculum the teacher
should: a) specify goals to pursue instead of content to learn, b) accept a
diversity of outcomes instead of demanding common results, c) request the
production rather than the communication of knowledge, d) evaluate at the task
rather than at the knowledge level, (e) build groups instead of assigning activities
that only have meaning to individuals and (f) promote communities of practice
instead of remaining localized and underutilized. The above ideas mean in
practice that roles, skills and tasks should be re-defined as teachers make a shift
from presenters to managers of activities. Feedback is given to students from
their contribution, which may include processes and strategies new to the
teachers that need to be mastered before they can respond.

Similarly, students need to become active and self-directed learners where they
are constantly seeking to enhance their learning experience through independent
and collaborative learning processes. Modular structures, credit accumulation
schemes, independent learning and collaborative knowledge building can create
a process where students can empower their skills and knowledge. However, not
all students wish to make their own choices or be responsible for the quality of
their choices. More process-oriented strategies bring more independence but
also the need for more self-direction and more self-motivation. These
approaches and traits are not familiar in many students. Giving students their
own choice of time, method, content and media, route and pace will mean less
chance of group interaction and discussion. Many students will need some kind
of support from a teacher who will make some learning choices for them. Thus, in
the Computer science curriculum it may be an option of selecting predetermined
choices, but this requires multiple versions of the same course or course
components.

The use of computers to facilitate the learning and teaching process for a
computing subject is a major task since information and digital tools are
increasingly prevalent in education and especially in current science and
mathematics education reform in Europe using inquiry-based approaches in
science classrooms (Rocard, 2007) particularly within the categories of: (a)
tutorials, (b) simulations, (c) applications and (d) communications. It is important
though to distinguish between the technology and the computer science
curriculum. Computer science must be perceived as a broader field that studies
the phenomena related to computer systems and technology should be
perceived as a tool that may facilitate the learning and teaching process. For
example, technology may be used in a software engineering course or in any
other discipline for helping students to exchange and share ideas through an
online discussion forum or to transfer information located in an online repository
tool. To situate this in a context, computer science is the subject matter and
technology is the tool that assists on communicating, delivering, planning and structuring it.

While efforts to promote computer science in schools have begun in most countries, some schools are trying to integrate computer science courses by designing and delivering curriculum in such a way that inspire students to choose computing subjects. For this reason, the Computer science Teachers Association (CSTA) was established in 2005 by the Association of Computing Machinery (ACM) for promoting computer science in high schools. Two surveys were conducted by CSTA, the first in 2005 and the second in 2007, about computer science educators in the United States. In 2008 a more detailed analysis was conducted based on the 2007 survey examining findings reported from individual states. The results from all surveys reported problems in relation to insufficient computer science teacher certification, ineffective professional development processes as well as under-representation of women and ethnic-minority students in U.S. high school computer science courses.

Educational policy makers, teachers and institutions need to consider several issues to successfully and meaningfully enhance computer science in schools. For example, there is a need to consider and implement appropriate computer science courses to develop an inspiring curriculum for computer science or consider the particular topics that these subjects will address. Furthermore, attention should be given to the relevancy of these courses and how they align with current trends in computer science. Moreover, there is a need to conduct empirical research for identifying computer science teachers’ conceptions of, and approaches to, teaching computer science (with or without the use of technological tools) courses (e.g. Lameras et al. 2008) to better understand the perceptions of computer science teachers and what they do in relation to contextual influences such as the educational environment, curriculum, level of students, approaches adopted etc.. To effectively address these concerns specific aims and objectives for the computer science curriculum should first be identified, and a well-defined subject area, including both hard (i.e. software engineering) and soft (i.e. information management) subjects for school level be highlighted and developed. Using the curriculum recommendations provided by curriculum committees such as ACM K-12 task force curriculum and UNESCO’s IFIP curriculum, pSkills is planning to propose some adaptations to the computer science curriculum at a European level. The project intends to utilise educational programming languages, include recommendations at a national level, while at the same time introduce an array of teaching approaches that teachers may adopt in teaching computer science and related subjects.

3.0 Learning and Pedagogy
Developing practices conducive for learning where students are motivated and inspired to achieve has long captured the attention of educationalists and the inception of technology has added a dynamic parameter. The utilisation of technology in education has the potential to inspire learning amongst students in the broad educational spectrum and there are a wide range of hardware devices
and software applications to be harnessed. Technology use has been chequered with some educators and administrations being distracted by its popularity, uncertain how to use IT or thinking that technology was the end solution. However, there has been an increasing focus on the integration of technology within a pedagogical framework to stimulate learning according to the situational circumstance. This type of approach is gaining popularity amongst educators to develop an understanding of the learning potential of the pedagogy (Stefani, Mason & Pegler, 2007) and use technology to inspire and motivate their students.

Using pedagogy as a foundation for learning, allows the educator to personalise the learning and precipitate the content, interaction and assessment designed to mine the learning potential of the student. Different pedagogies are suited to various curriculum subjects and the style of the teacher, however, whatever the pedagogy it is designed to stimulate thinking and allow students to analyse and process information better than with passive content. Coupling pedagogy with technology gives teachers the opportunity to utilise tools that students have some familiarity and interest with thereby raising the potential motivation that students can have with learning (Keengwe, 2007). Such a coupling also creates the potential to access higher order knowledge skills such as collaboration, initiative and creativity (Bereiter and Scardamalia, 2005) further challenging students and prompting paths to a deeper understanding of the subject. There are many examples ranging from mobile applications to second life where learning scapes are created to engage the learner. The value of such scapes depends upon the context in which they are used, what value and purpose is given by the teacher for the technology itself may have entertainment value but the learning involved is linked to its authentic purpose (Smith, 2010). Using technology within a pedagogical construct is crucial as it gives the technology an authentic purpose for all those involved in its use, particularly teachers as they empathise with the educational impact of said technology.

The use of pedagogy is designed to stimulate interaction so that when technology use is integrated it should disrupt passive learning exchange and stimulate active thinking and synthesis. Using technological aids within a computing context may seem a misnomer but the theoretical components can be complex so that applications designed to promote collaboration and problem solving are beneficial to increasing understanding and the incentive to pursue study in this field. The development of Educational Programming Languages (EPL) over recent years has given educators an opportunity to create collaborative and problem solving tasks using software that has a game based interface. In the computing sciences it is possible to connect the theoretical components of the school curriculum enabling teachers to combine pedagogy and technology and develop scenarios that are dynamic, interactive and linked. These type of scenarios form the basis of the learning scenario development for the pSkills project where two EPLs were selected to develop learning content for the computer science school curriculum in four European countries.
4.0 Using EPLs in the classroom

The current level of computer science education in schools focuses on the manipulation of programs considered as predetermined. To address the decline in numbers choosing computer science, alluded to in section 2, requires the level of school education change by focusing on applications as open systems that can be adapted to users' needs through programming. The application tools used should focus on programming languages that hide the syntactic complexity of general purpose programming that prevent students from comprehending the general concepts that are language-independent and instead give students the ability to utilise the program output for interpretation and manipulation. EPLs are designed to serve this purpose and provide an ideal tool to teach the core concepts that are necessary for students to use the computer in a cooperative and creative way.

It is apparent that each EPL is not something that can be used in every educational situation and to select the most appropriate EPL computer science teachers and their students in secondary education, a set of criteria has been specified to address their needs.

The EPLs surveyed were selected on the basis of fulfilling all the following criteria that ensure alignment with the core aims of the pSkills project:

- support object oriented programming which is the most widely used programming paradigm for application development
- offering an Integrated Development Environment with an intuitive Graphical User Interface
- support for visual programming to enhance user friendliness and minimize syntactic error during code development
- free and open source provision of the software to facilitate the use of project results in the widest possible range of schools
- existence of an active development community to ensure long-term support and evolution
- existence of stable versions to facilitate usage in actual educational settings
- age range that covers at least the secondary education students
- maximum possible number of supported languages including, if possible, all the languages needed in the context of the project (English, German, Italian, Estonian, Greek)
- localisation support to ensure transferability of project results to other languages not currently supported by the EPL
- running on most popular operating systems (MS Windows, MAC OS, Linux) to ensure wide usage of project results

Two EPLs, SCRATCH and Squeak etoys, were selected as fulfilling the above criteria.
The pSkills consortium then developed a common European computer science curriculum that is closely linked to the use of the selected EPLs and will develop and implement learning scenarios in schools to evaluate the effectiveness of the suggested curriculum but more importantly the use of EPLs to teach the abstract concepts of computer science.

Integrating the use of an EPL in a learning scenario underpinned by a pedagogical framework has crystallised as the focus of the pSkills project after survey and analysis of the state of computer science education in both school and higher education instrumentalities for the four European countries involved in the project. The provision of a pedagogical scaffold enriches the education value of the learning experience (de Freitas, 2008) and assists the learners in becoming more adept in acquiring knowledge (Kiili, 2005). Paralleling the development of learning scenarios for students is the development of a training framework for teachers, designed to mentor them on the use of an EPL within the pedagogical scaffold as well as empowering them to develop further scenarios using EPLs for other aspects of computer programming.

The implementation phase of the training framework and learning scenarios will take place towards the end of 2010 and in early 2011. Whilst some developed material has been tested on a small pilot scale in two countries, the larger trial periods allows for increased feedback from students and teachers as to the effectiveness of the developed material for learning computer science. It is necessary for teachers and the project personnel to be able to assess the perceived competence in knowledge ability (Bereiter, 2002) to make recommendations about the developed material. Feedback from the all four countries will be consolidated and used to modify training material and learning scenarios before publishing complete findings and materials.

5.0 Conclusion
The current status of computer science education in schools and the decreasing numbers of students studying computer science in higher education has prompted an examination worldwide into the current practice of computer science education. The pSkills project forms part of that global inquiry and has researched the current education practices in computer science in four European countries looking at both the curriculum and the classroom practice.

There is a deliberate focus on the part of the project consortium to not only use game-based technology in the form of EPLs but to structure the use of such technology within a pedagogical structure. The explicit application of learning theory within the project targets not only the educational content and resources but the dynamics of classroom and school educational practices. The project aims to address both content and practice with the development of learning scenarios for students and training frameworks for teachers with online support materials and network. The project further aims to trial both scenarios and training framework in four countries to test the effectiveness of their developed
resources. It is anticipated that the final outcome will reflect resources that integrate technology and pedagogy as well as practices that invigorated learning in computer science.

A methodology for the implementation of the training scenarios has already been established emphasising processes and strategies surrounding the concept of learning design in terms of planning, structuring, sequencing and representing lesson plans. The project aims to explore current trends in design for learning research for delivering pre-structured training scenarios through which computer science teachers will be able to create appropriate lesson plans for teaching EPLs (e.g. SCRATCH) while at the same time making explicit some of the pedagogical considerations they use to create teaching and learning activities. In this line, training scenarios may be perceived not only as a tool for helping teachers organise a particular session or an entire course, but also to consider some issues related to pedagogy, design and practice of different learning patterns that would likely be adopted in different learning contexts. Training scenarios will emphasise different themes of the learning design process such as the educational environments and intended learning outcomes, learning activities, tools and resources etc. All these different aspects may contribute to providing structure and guidance in designing learning activities for enhancing learning and teaching programming courses at a school level.
6.0 References


Keengwe, J. (2007). Faculty integration of technology into instruction and students perception of computer technology to improve student learning. Journal of Information Technology in Education v. 6, pp. 169-180


K-12 task force computer science curriculum committee, 2nd ed. New York: Association for Computing Machinery.


Acknowledgement
The research work reported in this paper was done within LLP Comenius Multilateral Project 502843-2009-LLP-GR-COMENIUS-CMP: “pSkills - Programming Skills Development in Secondary Education by means of Modern Educational Programming Languages”. This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.