Educating future coders with a holistic ICT curriculum and new learning solutions

Pia NIEMELÄ
Computer Science, Tampere University of Technology
Tampere, Finland

Cristiano DI FLORA
Rovio
Helsinki, Finland

Martti HELEVIRTA
Tampere, Finland

and

Ville ISOMÖTTÖNEN
Mathematical Information Technology, University of Jyväskylä
Jyväskylä, Finland

ABSTRACT

Technology-orientation and coding are gaining momentum in Finnish curriculum planning for primary and secondary school. However, according to the existing plans, the scope of ICT teaching is limited to practical topics, e.g., how to drill basic control structures (if-then-else, for, while) without focusing on the high level epistemological view of ICT. This paper proposes some key extensions to such plans, targeted to highlight rather the epistemological factors of teaching than talk about concrete means of strengthening the craftsmanship of coding. The proposed approach stems from the qualitative data collected by interviewing ICT professionals (N=7, 4 males, 3 females), who have gained experience of the industry needs while working as ICT professionals (avg=11.3 y, s=3.9 y). This work illustrates a holistic model of ICT teaching as well as suggests a set of new methods and tools.

Keywords: ICT curriculum, teaching ICT in primary and secondary school, concept maps, modelling, holistic ICT model

1. INTRODUCTION

The new curriculum with information and communication technology (ICT) as its focus is currently being reviewed and prepared for publication. The need of more ICT experts in industry has been recognized in decision-making by governing bodies. Not only are various domestic directions promoting ICT education but also the EU and multinational corporations have been actively pursuing the instruction and assessment of e-skills. For example, the EU has outlined a strategy for improving e-skills for the 21st century to foster competitiveness, growth, and jobs. Moreover, distinguished pedagogues of especially the University of Helsinki [1, 2] are promoting more student-centered, informal learning: tablets for all students, using online material and social media to co-create, and better ICT and multi-literacy skills. Future needs have guided the planning of the 2016 curriculum. The way of working and living is rapidly changing, and the need for curriculum change is therefore acknowledged. Familiarizing students with technology and learning the basics of coding will be started already in primary school and the skills gained are further strengthened at the secondary level.

In the ICT curriculum, ICT literacy and skills are meant to be built gradually, starting from visual coding and tactile learning followed by a more formal approach at the secondary level, where ICT is integrated into math teaching. Hence, programmable calculators and other computational features are well represented in the plans. The introduction of new ICT concepts by experimenting relies on "Learning by doing" methodology. Graphical and other high-level languages with additional libraries meant for education are utilized. Learning goals are divided as learning packages that logically progress in according to the sequential grades of secondary school. For example, the 7th grade aims at acquainting pupils with computing basics, such as the statements, data types, the sequential execution of the program, 'if-then-else' flow control structures, and finding errors in syntax and correcting

them. Basics of logic are introduced, starting from a truth value of a sentence. In the 8th grade, variables and functions are introduced. State machines are used in playing with the construction kits (e.g. the switch states ON/OFF). Logic continues with deduction and reasoning. In the 9th grade, new variable classes such as collections, conditional iteration (while, do, for), and recursion are introduced. At a more general level, the learning goal is to model a problem and divide it into smaller executables.

2. RESEARCH QUESTIONS

The current proposal for ICT curriculum emphasizes gaining the craftsmanship of coding with small and valid incremental steps. The order of propagation is well-justified, but still arguable. Instead of addressing all the possible aspects of coding and computer science learning primitives-in-detail, the epistemology of the ICT teaching as a whole should be discussed to consider all essential needs. It is important to ensure that these findings are sanity checked and that the discussion is not lead by pedagogues only, but validated and augmented by ICT professionals, who know the real industry needs. The research questions to be replied are:

1. Which kind of model would give a more holistic view of ICT epistemology?
2. How to support the learners in their becoming not only good coders but also good software architects and designers?
3. Which kind of learning solutions would support the ICT teaching model proposed?

3. INDUSTRY NEEDS RULED IN

To get a better grasp of the current ICT landscape we interviewed seven ICT professionals by email, six of whom are software developers and one a program manager. The email questionnaire contained the following questions: What are the ICT skills needed today/in the future? Which are your best ICT courses/informal learning experiences? How should ICT be taught in the primary school? With seven replies, we are far from scientific significance and based on the data only rough recommendations can be given. However, the anticipated holistic model can be verified by referring to the answers. Based on the replies, we classified ICT related capabilities to four categories: the craftsmanship of coding, modeling, user-centered design, and project management. More generic skills such as critical thinking, future working qualifications, and global citizenship were also mentioned, however, not taken into account here as they were regarded more as all-encompassing, general capabilities to be taught in other subjects as well.

Since the majority of the interviewees represented the implementation side, the craftsmanship of coding perspective was well pronounced. In a list of needed skills, web computing was mentioned six times and mobile coding once, followed by data structure & algorithms (3), and testing (2). Among the most common computer languages Java (2), JavaScript (2) and C++ (1) were listed, but also such specialties as Rust (1), Clojure (1) and Go (1) received votes. High-availability engineering (1) and the ability to develop games (1) were seen as useful system level capabilities. Modeling was mentioned four times: design, UML, architecture, and being able to recognize meaningful entities were listed. User-centered design occurred once in the form of ‘seeing the needs of the customer’ emphasized by the only project manager involved, who considered also project management and selling as important skills.

Our interviewees regarded hands-on experience as the main building block in learning. They would include in the ICT lessons of primary school e.g. team-work and pair-programming exercises, increase motivation and inspiration by providing good examples, combine ICT with sports, and have students build their own e-portfolios. Working in teams or in pairs kids would learn informally scaffolding each other in the zone of proximal development [3]. Regarding good learning experiences the importance of teamwork was emphasized (3), the basic courses in the beginning have been the most meaningful (2), and those teaching techniques that remain the same regardless of the language such as data structures and algorithms (3) were valued high. Nevertheless, we also

received critical views regarding ICT teaching, for example: *At high school I never attended any good ICT courses. But all the math and physics at school helped me to learn problem solving and how to break down a problem in multiple pieces.* According to the interviews, the future is drifting in the direction of HTML5 (2), robots (1), internet of things (1), and visualizations (1). These findings were classified into four main categories.

The craftsmanship of coding

In the discipline of handicrafts and craftsmanship, learning happens through doing by hand, which is seen as a way of leveraging innovation and the creativity. Theories such as intelligent hands [4] and learning by doing, are the basis for the tactile learning language. In maths, the tactile exercises such as fraction pieces and decimal system learning tools are used while approaching the symbol language more in-depth. In ICT, bridging the connection between electronics and coding may be achieved with the help of different assembly kits (e.g. LEGO MindStorm and Robots, Arduino, Lilypad, littleBits). Electronic components, such as light emitting diodes, buzzers, and couplers can be controlled by coding and give a more concrete and clear response than visible feedback on a computer screen.

In addition to construction kits, visual programming languages may be used as primers. Scratch, for example, provides graphical support for a user preventing the faulty code or the connecting of incompatible code sequences. Control structures (such as if-then, for, while) are ready-built, a user only has to adjust parameters, such as iteration counters. Visual programming languages are limited in freedom degrees, which at the initial learning stage will be good to minimize the cognitive load: time is not wasted hunting syntactic errors. In the long run, the conciseness of such language starts to restrict freedom and creativity indicating the due date to expand to more expressive programming languages.

Conceptual modeling as a software architect

On the authority of our interviewees, the development of ICT talent requires strong modeling and conceptual skills on. Therefore, we propose conceptual modeling as one of the key expertise and concept mapping as a previous preparatory skill. In addition to visually luring and easy to use concept map applications, the tools of ICT professionals such as the Unified Modelling Language (UML) may be used to introduce students to software architecture concepts. Modeling skills are especially important for communication purposes, both between developers and other stakeholders. Nowadays, as the development is often geographically distributed, these skills are extremely necessary, yet undervalued. Even if one is not going to be a developer himself, a better understanding about the subject benefits him also as a user or buyer of ICT systems. For example, defining use cases, assessing interfaces, or reporting errors require the capability of speaking the same language with developers.

User-centered design for new innovations

Real innovation takes the user's needs into account; it focuses on the user and his context and incorporates his perspectives during the whole design process. New applications may be seen as innovations and future coders need tools for user-centered design. An essential prerequisite is observing the environment to identify its needs by perceiving its operation schemes and processes. For younger students, gaining the required level of situation awareness is a meta-cognitively demanding task and requires such future working skills as communication and collaboration. In software projects, when the snapshot of the situation has been obtained, the demands are dressed as formal use cases and requirements, which are a starting point to the following implementation.

Project management

While all the needed software expertise is gained, the management of the project is a lot easier. Agile methods such as Scrum or Canban help in divided the work into smaller tasks that are shared with the team [5]. Project management is about controlling the entirety from the customer needs to a graceful delivery of products. Agility methodologies target flexibility by taking into account the moving target. A customer may change his mind during the process that is called requirement volatility. By

iteratively ensuring that the direction is right the product will better respond to the need, hence the project management is linked with the user-centered design, too. Albeit of the biased sample (only one project manager interviewed) the interview data confirmed what was expected i.e. that the project management is more concerned about the user needs than developers.

4. THE HOLISTIC ICT TEACHING MODEL AND NEW LEARNING SOLUTIONS

The proposed teaching model is depicted in Figure 1. The most elementary building block, the craftsmanship of coding, merges both the visual coding and tactile learning to provide robust hands-on exercises to build the base. When systems gradually become more complicated, students will have to learn how to model the system; to this aim we propose the building block of conceptual modeling. The third thread illustrates the need for user-centered innovations. To be able to innovate, the student has to make observations in order to become acquainted with the system, after which it is possible to optimize and improve existing processes, and perhaps most importantly, come up with totally new approaches to the problems and challenges.

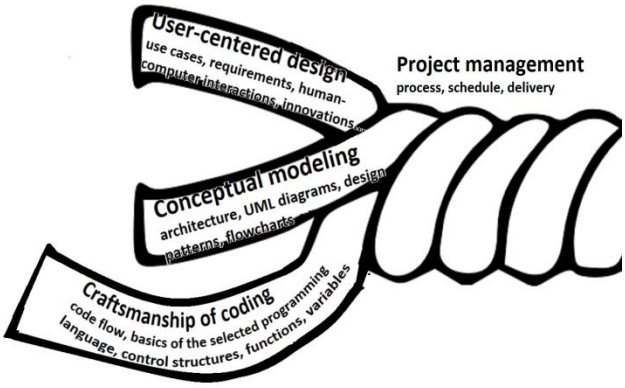

Figure 1: The proposed holistic model for ICT curriculum in primary and secondary school

Together threads form the cord of a more equipped ICT professional capable of managing ICT projects. The folding thread of project management includes mastering the process (e.g. agile), being able to divide the project in smaller tasks, schedule and stay within the deadlines.

As new learning solutions games have proven to be very powerful, since many students are passionate about playing. High motivation and engagement appear as a "flow" while playing. Fu et al. [6] examined engaging games and listed properties such as immersion, the clarity of the goal, autonomy, feedback, challenge, and social interaction as the ingredients of flow. Moreover, it has been reported that games can have a positive impact on pupils' perceptual templates, knowledge acquisition and affective outcomes [7]. By including the suitable features of games in learning environments, serious education may transform to edutainment. The level of shared fun increases interest in ICT in general and ICT learning objects may be gamified, too.

In recent years, concept maps have been recognized as an effective visualized learning tool that helps learners memorize and organize knowledge. Åhlberg [8] recommends maps in situations requiring data parsing and argues that they illustrate the conceptual and propositional structure of written text. He also considers conversions to both directions, from text to a map and vice versa, a good way to work and elaborate meanings. In computing good software architects are good modelers, too.

User-centered design implies the participatory methods of inquiry, modeling the situation, and means of improving. Nousiainen [9] emphasizes that the continuous involvement of users is a goal as such but also a way to empower users and promote workplace democracy and the means to practise the working life skills of participation, collaboration, and communication. In some schools students have already started to innovate e.g. new means to recycle [10].

Project management may be practised by open assignments i.e. with school projects that have to be scheduled and delivered in a timely fashion. In such projects, students should be responsible for the acquisition of information needed, for example, by interviews or searching data from the net. After the data is gathered its reliability has to be assessed and the data has to be parsed.

It would be good to attempt raise the abstraction level by modeling. The interviews suggest that the project management level is more situation aware and concerned about retrieving the big picture, whereas developers think more in terms of technical solutions.

5. CONCLUSIONS

Finland is planning to enrich its primary and secondary school curriculum with coding in order to prepare students for the future working life. Instead of settling on only the basics of coding, e.g. code flow and control structures, we claim that the ability to innovate and design software systems is in the very heart of software engineering. By adding the key areas of modeling and user-centered design to instruction schemes, we create a more holistic ICT curriculum. The conception and modeling ability is needed not only in ICT but in knowledge building and conceptual thinking in general. User-centered design improving the practices of one's own environment may also be seen as a tool of empowerment. An empowered member of the society, who is aware of user needs, will also grow innovative.

Hands-on experimentation was considered beneficial also among our interviewees: games, pair-programming, and learning from others informally were seen as ways to foster learning and engagement. Innovativeness and creativity are buzzwords used in curriculum planning, often in accordance with arts and crafts, which are assumed to enhance them. With new methods and learning solutions, creativity may be fostered with STEM subjects as well: building robots, making animations, and playing games (e.g. Angry Birds Space to assimilate gravity basics) are new, engaging and motivating ways of learning.

Coding starts in Finnish schools in the autumn of 2016. ICT classes become as a laboratory of new learning tools and methodology. By documenting experiments and applying continuous development cycles, we may iteratively improve the learning results. In addition to teaching to code, it is necessary to introduce new tools for modeling and user-centered design.

6. REFERENCES

[1] K. Hakkarainen, L. Hietajärvi, K. Alho, K. Lonka, K. Salmela-Aro, "A Paradigmatic Analysis of Contemporary Schools of IS Development", **International Ensyclopedia of the Social & Behavioral Sciences,** 2015 , pp. 918-923

[2] L. Krokfors, "Learning. Creatively. Together ", **Educational Change Report 2016.** 2015.

[3] L.Vygotsky, "Interaction between learning and development", **Readings on the development of children,** 23.3: pp. 34-41. 1978.

[4] L. Hyde, "Making it ", **New York Times,** 6, 2008

[5] H. Kniberg, and M. Skarin, "Kanban and Scrum - making the most of both". 2010.

[6] F. Fu, R. Su and S. Yu, "EGameFlow: A scale to measure learners' enjoyment of e-learning games", **Computers & Education**, vol. 52, pp. 101-112, 2009.

[7] T. Connolly, E. Boyle, E. MacArthur, T. Hainey, & J. Boyle, "A systematic literature review of empirical evidence on computer games and serious games", **Computers & Education**, 59(2), 661-686. 2012.

[8] M. Åhlberg, "Concept maps as a research method". 2002.

[9] T. Nousiainen, "Children's involvement in the design of game-based learning environments". 2009.

[10] M. Aineslahti, "A Journey in the Landscape of Sustainable School Development". 2009.