Vizwik – visual data flow programming and its educational implications
Brief Paper for EdMedia 2015

Abstract
The new movement towards digital literacy in the 21st Century requires that the educational system shifts from developing consumers of technology to creators of technology. Given that software is pervasive in every aspect of our digital lives the ability to create software, or ‘coding’, has become the focus of this new digital literacy movement. Large challenges for schools, teachers, and students exist evidenced by only one in ten students in North America being exposed to coding during their education. In this brief paper we present a preliminary report on the use of a new visual data flow programming language platform, Vizwik, and its early adoption in the province of New Brunswick Canada. We outline our observations regarding efficacy and its rapid adoption by teachers followed by a proposal for a more comprehensive study of its educational impact upon students in a transdisciplinary setting.

Keywords: design-based research, communities of practice, professional development, authoring tools, coding, visual programming.

Introduction
Visual data flow programming languages (VPLs) have been the subject of academic research for the past 3 decades with the majority of the effort directed to the definition of languages for the purpose of creating their theoretical foundation based on various calculi (ACM 2004). However, there is a significant lack of evidence-based research on the efficacy of data flow VPLs as a means of developing computational thinking skills in middle-school (Grades 6-8) and high-school (Grades 9-12) students.

A resurgence of the importance of coding in education is occurring lead by the non-profit organization code.org (Partovi 2013) that has brought an hour of code to 112 million students (Partovi 2014). This work was motivated by the current educational crisis in America in which only 10% of students being exposed to coding in the middle-high school grades 6-12 (BLS 2014). Coding is being recognized worldwide as a critical component to education demonstrated by recent political decisions in countries like Estonia and England to make coding part of the core curriculum (Code Kids 2013). In a bold move this year the State of Arkansas have become the first state in the USA to make coding mandatory in its fall 2015 curriculum (Arkansas 2015). With this momentum the research and development of coding tools and services to help teachers bring coding to students is a key factor.

Early use of VPLs in academic setting dates back to the 1970’s with the introduction of LOGO (Papert et. al 1967) based on a dialect of Lisp, and introduced turtle graphics for elementary school children. This was followed by Etoys in the 1990’s by Alan Kay targeting primary school math and introduced Morphic Tile Scripting (Kay 1980). Scratch (Resnick 1991), developed by Mitchel Resnick at MIT Media Labs, also based on tile scripts has now become a standard elementary school coding approach and the basis of several commercial products (Tynker 2013) (App Inventor 2009). However, none of these languages have been able to penetrate the high school coding levels.

Another branch of VPL design is the class of data flow languages typified by commercial products Prograph (Cox 1984) and LabView (LabView 1986) of the 1980’s. Prograph was the first general-purpose data flow programming language targeted to professional developers creating Macintosh desktop applications. Prograph was also used in education teaching introductory programming at several universities in Atlantic Canada in the 1990’s but was discontinued when the company folded. LabView is domain-specific targeted at instrumentation and remains in use today. LabView is also the basis of the educational robotics platform LEGO Mindstorms (Ferrari 2001) targeted to elementary school.

Visual programming, both morphic and data flow models, have been used in educational settings with great success at the early and late stages of the learning process. However, as noted there is a significant lack of presence in the middle and high school levels. This phenomenon is currently having a dramatic negative impact on the North American educational system’s ability to produce enough qualified programmers for its economic requirements with an estimated 1 million jobs being unfilled by 2020 (Code.org 2015).
It should be noted that while our economy needs more human resources with computing skills the pedagogical motivation behind introducing coding into schools is the development of computational thinking skills within our students. These skills include the ability to analyze problems, use abstract reasoning, critical and logical reasoning, creative analysis, pattern identification, and solution searching. This cognitive change enhances the students’ ability to deal not only with code, but with every subject, resulting in improved chances for a productive and meaningful life.

The obstacles of integrating coding into the education can be summarized as: (1) teachers do not have time to learn coding, (2) coding not officially recognized to count towards graduation and there i not in the curriculum, (3) coding in textual form is perceived as hard to learn. These problems have been recognized and are being addressed by industry with the development of new software tools, such as Vizwik, which aims to transforming the education system globally while offering a new paradigm in design and implementation which is based on end users’ (students and teachers’) styles and needs.

More specifically, the focus of our research lies in asking if we can improve development of computational thinking skills more effectively with visual programming. Early experiences with Scratch and Logo seem to improve access of more students to programming by changing the representation of syntax to a graphical form that appears to be more attainable to the brain. Images and visual aids are interpreted by the brain 65,000 times faster than text (Brain 2008). Almost 50 percent of the brain is used in reading images versus a small percentage of the left side of the brain is used to work with text (Pink 2004).

Visual programming has the power of pedagogical transitivity with its ability to be used in a variety of curriculum. Rather than making coding a subject unto itself, as we do with computer science, programming can be considered as the fourth ‘R’ after reading, writing, and arithmetic which are recognized as basic literacies for all. Its ability to be used as a general purpose language of expression for computation elevates its functionality to a level of transitive empowerment for students to define their creations. Disregarding of particular disciplinary school domain, coding becomes a language of expression integrated in all subjects that becomes a mean to an end, allowing students to gain a new form of literacy.

**Context of the study**

The study is being conducted in the context of New Brunswick, Canada’s provincial school system by the research team being composed of an expert in computer science, experienced innovative teacher and ICT mentor, as well as specialist in techno-pedagogy from the university, a combination of different perspectives and expertise necessary to design and implement efficiently a new VPL platform that will help local students to develop computational thinking.

New Brunswick’s K-12 school system is composed of two sectors French and English which reflects its dual linguistic structure with French being a minority language. Our work focused on working mostly with middle-school and high-school populations of students. While having ICT-competence as transdisciplinary learning outcome explicitly stated in each subjects’ curriculum for all Grades, it remains unclear how it should be developed and taught and by whom. Especially it concerns computer science, absent from the general curriculum with few selective courses related to technology on the high-school level. Several ICT-related initiatives aimed to improve the situation with introducing individual laptops in Grades 7-8, robotics-based learning, use of interactive whiteboards what could be cited as examples which are limited, however, to few participating schools and teachers (XXXXXX et al., 2011, XXXXXX et al., 2010, XXXXXX, 2012). With few specialists trained in computer science and ICT, professional development is also limited to workshops offered to teachers by the XXXXXX (Name of association) association. Although being successful in promoting techno-pedagogy within francophone communities in the Atlantic Canada, XXXXXX (Name of association) does not provide specific training in programming and coding.

The question of ICT competences, of which computational thinking and higher-order IT skills are essential parts, and its development over the lifelong cycle is recently addressed by the research partnership network XXXXX (Name of the network) created to identify and to measure the acquisition and transfer of lifelong digital skills, has been created. It aims to establish collaboration between family and school, between elementary school and high school, between high school and postsecondary institutions and, finally, between educational institutions and the workplace, in order to (1) define the continuum of digital competences in varied contexts of life such as education,
work, community and family, (2) identify and describe best practices that develop these competences over the lifelong period, and finally (3) develop and implement new initiatives as result of increasing collaborative efforts (XXXXXX, 2015).

By working with partners from multiple sectors and disciplines, the network will generate evidence-based data about how digital competence, defined by Calvani, et al. (2009) as ability to explore and act with flexibility in situations involving new technologies to be able to analyze, select and evaluate critically the data and information, to exploit the potential of technology in order to represent and solve problems and also to be able to build and share new knowledge while having a sense of responsibility and respect of the reciprocity of rights and obligations is being acquired and transferred from one educational level to another, and from that to the workplace and everyday life.

Methodology

It is becoming a common trend to operationalize evidence-based approach and its underlying design-based research (DBR) methodology that blends empirical educational research with theory-driven design (Gunn and Peddy, 2008). Referring to the fundamental work of the Design-Based Research Collective (DBRC, 2003), the authors identify five main characteristics of the DBR that frame implementation and evaluation of innovative educational concepts based on the analysis of what occurs rather than working from hypotheses thus helping to deal with ‘consistently unreliable predictions about the impact of technology in education’ (Gunn and Peddy, 2008, p. 364).

Within the DBR paradigm, in our previous study of the XXXXXXX (Name of the community) virtual community, we identified three basic components of its dynamic developmental structure: needs analysis, pedagogical and technological background, and research feedback. The following schema illustrates the dynamic character of the model for each developmental cycle (XXXXXX, 2009, p. 24).

The authors of the DBR-model (DBRC, 2003) argue that this innovative research approach is suitable for studying complex problems in real, authentic contexts in collaboration with practitioners. Research and development happens through continuous cycles of design, enactment, analysis, and redesign which would lead to sharable theories that help communicate relevant implications to practitioners and to other educational designers. Integrating known and hypothetical design principles and conducting rigorous and reflective inquiry to test and refine innovative learning environments, the DBR-based research must account for how designs function in authentic settings using methods that can document and connect processes of enactment to outcomes of interest (DBRC, 2003).

This model is suitable to address challenges that the Vizwik platform faces while combining software design, testing and implementation in schools with students and teachers. As the first part of our paper provided with analysis of urgent needs to develop computational thinking in all students as a form of literacy which goes beyond the coding ability. In this paper we investigate the Vizwik developmental cycle by conducting (1) an analysis of background of the web-based platform and its connection to the end users’ needs, in our case, students and teachers, (2) an initial
testing with high school students and teachers (enactment) thus collecting first observations from the field that inform and determine next steps in the design. According to these objectives, we formulate research questions as following:

Research Questions:

1) How to connect Vizwik web-based platform with needs of end users, students and teachers?

2) What lessons do we learn from first field-observations of the Vizwik testing in the classroom?

In the next sections, we briefly report what was done at this stage while addressing each question.

Vizwik - A Visual Data Flow Programming Platform

Vizwik is the invention of XXX Inc. (name of the company) that consists of a web-based platform incorporating a visual data flow development system for mobile web apps and a social network of users for whom the mobile apps are shared and sold. The design of this platform emerged from research work at Dalhousie University to define a mathematical model of a class of languages defined as “controlled data flow” [XXX 2009]. The combination social development, deployment, and sale of software integrated into a unified platform was the subject of a patent (XXX 2010).

The visual data flow language within Vizwik is an extension of the work originally started in Prograph. The extensions consist of new control-flow annotations for conditionals and loops that map more directly to imperative languages as opposed to the unfamiliar Prolog-like control flow in Prograph [XXXXXX 2007]. Another modification was the removal of object-orientation in favour of a lambda-based model of first-class functions added to the data flow type-space.

The development of mobile software within Vizwik is based on the model-view-controller pattern thus reducing the complexity of mental modelling an App to three interconnected concepts. These are explicitly represented in the tool as three editor types for modifying the user interface (View), the program behaviour (Script) and storage of models (Data). This is further supported by use of drag and drop of items between each editors reinforcing the relations through immediate feedback and discoverable through the use of visual affordances. Providing immediate feedback is also supported through the use of a software simulator of the mobile phone and the automatic compilation of Scripts removing the code-compile-run cycle. These four components are combined into a “project” metaphor that is the basis of all construction within the platform. Projects are created from templates, and used to generate Apps.

Vizwik is a social network of users who connect through their shared use of an App which generates a social hypergraph of people and apps from which a recommendation engine generates suggestions of Apps to people based on current trends of users within the network following Apps.
Creation of a new programming platform that does not conform to any popular textual or visual programming languages means that any user introduced will be a new user and will require some form of training. As a result, success of Vizwik not only depends on its design, simplicity of learning, and easy of use, but on the supporting learning system that takes a Vizwik user from a novice to expert. The purpose of the tutorial system within Vizwik is to introduce the user, through a series of brief explanations and examples followed by task-based constructionist learning, the development of a full mental model of the system upon which computational thinking can mature and thrive. Built upon the existing project-based model of creation the tutorial system provides learners with a playground upon which to learn how to use Vizwik and apply this learning to creative projects.

**Preliminary observations from testing**

Preliminary testing was performed in NB schools over a 4-months period. Launched in October 2014, Vizwik is already adopted by 500 high school teachers from a total number of 4238 working in the provincial high school system. A small group (45) of teachers were provides an hour of basic instruction on Vizwik while the the others learned it through access to the tutorial system which helps them to develop projects with their students.

Following is a summary of first observations, recorded as field-notes, of high school students across curriculum and main language of instruction (French and English):

1. Early on it was observed that the structure of the data flow language within Vizwik was similar to the diagrams used to teach mathematical expressions and equations, thus making Vizwik coding immediately familiar to students who had passed through this stage of mathematical curriculum.
2. Students identified with Attention Deficit Syndrome who were previously unable to concentrate on academic studies for more than 30 minutes were able to focus using Vizwik for periods longer than 60 minutes.
3. Students demonstrated positive emotional responses to the task by completing their Apps and sharing their sense of accomplishment with other students.
4. Students were able to work effectively in small groups of three or four around a single desktop computer, each of them being involved with equal levels of motivation and engagement to learn, and often participating in collaborative discussions while designing their applications.
5. Teachers were surprised at the level of engagement students demonstrated during sessions, some lasting three hours, which involved simultaneously about 50 students from middle school levels ranging from Grades 6 to Grade 8.
6. Recent testing of the tutorial system to 100+ local high school students showed that nearly 95 percent were able to complete the development of a complex game (consisting of animation, timers, conditionals, and data storage) over a 5-day self-directed session with little supervision.

7. Teachers with basic scientific backgrounds, or some simple coding experience, were able to learn Vizwik and follow up their students’ work by creating their own activities.

Moreover, several advanced teachers who learned Vizwik, began creating their own project-based curriculum which was then shared between teachers through social media and word-of-mouth communication thus contributing to the emergence of learning community. Also, a group of local non-profit organizations adopted Vizwik as a platform for teaching coding. With a minimal amount of training they were able to deliver training to students in from more than 20 schools.

Discussion and conclusion

In a context of one of Canadian provincial school system which does not have computer programming explicitly integrated in school curricula, a need for the development of computational thinking as new form of literacy for all becomes an important educational issue.

The paper aims to present first results of an on-going study of design and implementation of a new web-based platform incorporating a visual data flow development system for mobile web apps, which is being conducted by a multidisciplinary research team within a DBR-paradigm. After the first year of analysis-testing (enacting)-design cycle, first field-observations were conducted in several middle and high-school classrooms.

From our observations on students and teachers, we learn that the process of coding within graphical interface becomes more familiar as allows analogy with school mathematics, connection, already well documented in the literature from mathematics education with technology (see, for example, XXXXXXX (Eds., 2013)). Positive impact on students’ engagement, motivation, collaboration, and effective and productive work while interacting with technology-rich environments, and this being observed in students of various abilities also confirms similar findings from previous studies. Also, having some teachers being active users and creators of didactical knowledge, and sharing them with others using social media (similar to what happened, for example with GeoGebra community) is also a positive indicator of an overall positive first welcome of the new web-based platform from end-users, students and teachers. However, more rigorous studies are needed at the next cycles of the project, including case studies measuring impact on students’ learning, both disciplinary (in terms of computer science) and trans disciplinary (development of computational thinking). Fine tuning of tutorial system and other didactical tools is also needed.

References


H. Partovi (2013) [www.code.org/about](http://www.code.org/about)

H. Partovi (2014) [www.code.org/about/2014](http://www.code.org/about/2014)


Daniel G. Bobrow, Wally Feurzeig, Seymour Papert and Cynthia Solomon, The LOGO programming language.

BLS 2014
Code Kids 2013
Arkansas 2015
Kay 1980
Resnick 1991
App Inventor 2009
Tynker 2013
LabView 1986
Code.org 2015
Brain 2008
Pink 2004
Gauvin, Cox 2009
Gauvin, Banyasad 2010
Gauvin, Banyasad 2007