Honours Project Report

CodeSketch:
A Visual Programming Language for Arduino
HCI Approach

Bhavana Harrilal

Supervised by Gary Marsden and Marion Walton

Mark Breakdown

<table>
<thead>
<tr>
<th>Category</th>
<th>Min</th>
<th>Max</th>
<th>Chosen</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Requirement Analysis and Design</td>
<td>0</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>2 Theoretical Analysis</td>
<td>0</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>3 Experiment Design and Execution</td>
<td>0</td>
<td>20</td>
<td>17</td>
</tr>
<tr>
<td>4 System Development and Implementation</td>
<td>0</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>5 Results, Findings and Conclusion</td>
<td>10</td>
<td>20</td>
<td>11</td>
</tr>
<tr>
<td>6 Aim Formulation and Background Work</td>
<td>10</td>
<td>15</td>
<td>12</td>
</tr>
<tr>
<td>7 Quality of Report Writing and Presentation</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>8 Adherence to Project Proposal and Quality of Deliverables</td>
<td>10</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>9 Overall General Project Evaluation</td>
<td>0</td>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

Total Marks 80 80

Department of Computer Science

UNIVERSITY OF CAPE TOWN

2013
Abstract

Arduino's growing popularity and its capability of creating imaginative projects appeals to spectrum of various users, most of whom are designers and artists. The technology's shortcoming is the difficulty in learning how to program as a novice programmer. This paper looks at the possibility of formulating a visual programming language (VPL) for Arduino for novice programmers. The system developed in close connection with a non-profit organisation Ikamva Youth, who equip learners from disadvantaged communities with the knowledge, skills, networks and resources to access tertiary education. An iterative user centred design approach has been applied which involved participatory design sessions where learners from Ikamva Youth who have no programming experience were taught programming concepts using Arduino technology probes. The knowledge gained from lessons formed the basis of a visual programming language and the design of the system. Evaluation of the high fidelity prototype indicates that the VPL formulated is understood by its users and can be used to successfully complete tasks. Future work of the system can thus build on the current VPL as a programming educational tool.
Acknowledgments

Firstly, I would like to thank Professor Gary Marsden and Marion Walton for all their guidance, feedback and support throughout the course of this project. Thank you for always making the time to assist me when needed and maintaining my focus.

Next I would like to thank the students at Ikamva Youth for participating in the lessons and evaluations. I express a large amount of gratitude for choosing to spend your Friday afternoons and sharing my passion for learning with Arduino.

Lastly, I want to thank my family who have supported, inspired and motivated me during the project. Thank you for loving, understanding and believing in me.
Table of Contents

Abstract ..................................................................................................................... ii
Acknowledgments ..................................................................................................... iii
Table of Figures and Tables .................................................................................... vii

Figures .................................................................................................................... vii
Tables ...................................................................................................................... vii

1. Introduction ........................................................................................................ 1
   1.1. Problem outline ............................................................................................ 1
   1.2. Research Questions ..................................................................................... 2
   1.3. Proposed solution ......................................................................................... 3
   1.4. Report outline .............................................................................................. 3

2. Motivation and Background ............................................................................. 4
   2.1. Introduction ................................................................................................ 4
   2.2. Arduino ....................................................................................................... 4
       2.2.1. Using Arduino to teach programming .................................................. 4
       2.2.2. Tangible and graphical programming languages .............................. 4
       2.2.3. Existing Visual Programming Languages for Arduino ................. 5
   2.3. Ikamva Youth .............................................................................................. 6
       2.3.1. Human Access Point ......................................................................... 6
   2.4. Summary ..................................................................................................... 6

3. Design Chapter .................................................................................................. 7
   3.1. Introduction ................................................................................................ 7
   3.2. Understanding of users ............................................................................. 8
       3.2.1. Arduino lessons .................................................................................. 8
       3.2.2. MakeyMakey applications ................................................................ 8
   3.3. Technology probes ..................................................................................... 8
   3.4. Prototypes ................................................................................................... 8
   3.5. Evaluations ................................................................................................ 8
   3.6. Summary ..................................................................................................... 9

4. Technology probes and Lessons .................................................................... 10
   4.1. Introduction ................................................................................................ 10
   4.2. Learners ...................................................................................................... 10
       4.2.1. Permission and consent .................................................................... 11
   4.3. Lesson 1: Sequence .................................................................................. 11
       4.3.1. Probe ................................................................................................. 11
       4.3.2. Findings ............................................................................................ 11

...
4.4. Lesson 2: Selection........................................................................................................12
  4.4.1. Probe....................................................................................................................12
  4.4.2. Findings................................................................................................................12
4.5. Lesson 3: Iteration ......................................................................................................12
  4.5.1. Probe....................................................................................................................12
  4.5.2. Findings................................................................................................................13
4.6. Paper based feedback...............................................................................................13
  4.6.1. Requirements ......................................................................................................13
  4.6.2. Visual language vocabulary .................................................................................14
  4.6.3. Narrative representation vs. Conceptual representation....................................14
  4.6.4. Design based on lesson feedback .......................................................................14
4.7. Summary ..................................................................................................................16
5. High fidelity prototype and Final Iteration .................................................................17
  5.1. High fidelity prototype ..........................................................................................17
    5.1.1. Introduction ......................................................................................................17
    5.1.2. Design and implementation of Horizontal prototype .........................................17
      5.1.2.1. Comic book interface motivation ..............................................................17
      5.1.2.2. Development platform .............................................................................17
      5.1.2.3. Outcomes that were implemented ............................................................17
      5.1.2.4. Outcomes that were not implemented .......................................................18
    5.1.3. Evaluation and findings ....................................................................................18
      5.1.3.1. Test subjects ..............................................................................................18
      5.1.3.2. Permission and consent ..........................................................................18
      5.1.3.3. Methodology ............................................................................................19
      5.1.3.4. Structure of evaluation ............................................................................19
      5.1.3.5. Findings ....................................................................................................20
      5.1.3.6. Tasks Completion .....................................................................................21
      5.1.3.7. List of required changes ..........................................................................23
    5.2. Final Iteration .......................................................................................................23
      5.2.1. Introduction ....................................................................................................23
      5.2.2. Development changed made .........................................................................23
        5.2.2.1. Changes not implemented .....................................................................23
        5.2.2.2. Changes implemented ..........................................................................23
6. Reflection on ICT4D ....................................................................................................24
  6.1. Introduction ............................................................................................................24
6.2. UCD drawbacks.................................................................................................................. 24
6.3. Implications on ICT4D.......................................................................................................... 25
7. Conclusions............................................................................................................................. 26
  7.1. Research questions ............................................................................................................ 26
  7.2. Future work......................................................................................................................... 27
References ..................................................................................................................................... 28
Appendix A: Lesson Plan ............................................................................................................ 30
Appendix B: Sketches from Lessons ........................................................................................ 31
Appendix C: Table of Icons in Vocabulary list ......................................................................... 35
Appendix D: High fidelity prototype evaluation interview questions ...................................... 38
Appendix E: Description of High fidelity Tasks ........................................................................ 39
Table of Figures and Tables

Figures

Figure 1 User centred design .......................... 7
Figure 2 Sequence probe .................................. 11
Figure 3 Selection Probe .................................. 12
Figure 4 Iteration Probe .................................. 13
Figure 5 Student sketch illustrating sequence .................. 14
Figure 6 Student sketch of selection ........................ 14
Figure 7 Student sketch of iteration ........................ 15
Figure 8 Sketch of interface design ........................ 15
Figure 9 Example sketch of selection ......................... 17
Figure 10 Example sketch of iteration (TV example) ....... 17
Figure 11 Screen shot used in concept model extraction ........ 19
Figure 12 Screen shot of student creating a sketch of sequence .......... 22
Figure 13 Screenshots of student creating a selection concept sketch .......... 22

Tables

Table 1 Planning for each lesson at Ikamva Codes .................. 30
Table 2 A list of icons used in the system and their URL reference link ........ 35
1. Introduction

1.1. Problem outline

The Arduino prototyping platform is open to a variety of users, enabling them to create usable hardware and software. However, most users of Arduino are novice programmers and struggle with learning how to program for the Arduino platform. Arduino is an open source electronics prototyping platform that is used by expert to novice programmers [18]. These novice programmers can be anyone from a child to adult artists. The aim of Arduino is to provide flexible, easy-to-use hardware and software but the difficulty lies in the Arduino programming language which controls the hardware and is controlled by a set of C/C++ functions [18].

This project builds on the premise that the steep learning curve faced by Arduino users could be made less arduous for them be managed if users were introduced to programming in ways which drew on the representational and cognitive resources which they already have.

Dr Marion Walton, acted as the human access point (HAP), to provide insights into the user group and has given good understanding of the users and what the users knowledge. Walton, is a senior lecturer in the Centre for Film and Media Studies at the University of Cape Town. She has been involved at Ikamva Youth through several teaching and community projects. Her current focus is the Ikamva Codes project. Walton works with developing new projects around visual approaches to programming, and introducing young people to procedural thinking through Arduino and Lego Mindstorm robotics. She has taught students from Ikamva Youth from 2012 and has a good understanding of the challenges involved in introducing students to coding.

Teaching coding to South African youth

The Ikamva Codes project aims to teach coding skills to learners from Ikamva Youth [3]. The Ikamva Codes learners are novice programmers with little prior exposure to computers and limited access to a computer lab.

Small groups of learners meet in the afternoons to engage with lessons designed to teach basic programming concepts and repertoires using the Processing language. Their assignments focus on creating visual designs, since, as the teacher, Dr Marion Walton explained the rationale for this decision:

Processing code is somewhat forbidding initially, and like all programming languages, all the magic goes on behind the scenes. I chose Processing because the language makes it really easy to create procedurally generated graphics and thus to visualise the hidden procedural operations of the code. (Personal communication, 22 October, 2013)

This project aimed to contribute to Ikamva Codes by investigating whether Arduino could be used to teach novice programmers some basic concepts needed for their subsequent lessons in Ikamva Codes. The learning theory of experiential education focuses on the
process of learning by constructing meanings from direct experiences [29]. The benefit of hand-on learning according to Dewey is that students are much more engaged and the benefits to the entire group of students promote the learning process [30]. Arduino was chosen because of the possibilities it suggested for embodied, tangible programming practice, and the opportunity to get away from the imperceptible procedures of many textual programming languages.

In order to do this, required getting to know a small group of young women from Ikamva Youth who were interested in programming but had joined the programme late, and thus had not attended any previous Ikamva Codes lessons. They had no prior experience of any computer languages, including Processing. These novice programmers were introduced to programming via Arduino, rather than Processing.

1.2. Research Questions

This report addresses the design question: Given South African students' background and prior knowledge, is it possible to create a visual programming language (VPL) for Arduino. This visual programming language would be a very helpful tool for Arduino users who are limited by their inability to program for Arduino.

*Can HCI participatory design methods help to formulate a visual programming language for Arduino?*

*What are the verbal and visual languages that can be extracted into a visual programming language?*

The Arduino programming language is difficult for novice programmers, who make up a large portion of Arduino users. Before developing an alternative language, it is important to get to know the young people who participated in Ikamva Codes in order to tease out what visual and verbal languages they already know, which could help suggest ideas for a new system. This is a complex process which requires the use of participatory design methods to understand the users and their context, and also to establish what kinds of programming languages users would find most accessible.

*Are technology probes and ethnographic study methods suitable to design a visual programming language?*

HCI and participatory design have been known to create great results when working with Arduino technology [8, 9]. Utilising technology probes along with ethnographic study to create a visual programming language. Technology probes are used to gauge the knowledge and understanding of the users. Probes are used to gain information about the learners at Ikamva, aiming to prompt responses in an engaging method [1]. Ethnographic methods focus on producing an account of what is going on in real situations by observing the moment-by-moment behaviour of people interacting with others and their environment over extended periods of time. Resulting in the collection of first-hand, eyewitness evidence of how the students learnt the materials taught in the lessons [1]. These design methods
hold the user as an important component to formulating the solution to the problem. The problem addresses the inability of Arduino users to program for Arduino; this is a core function of the Arduino technology. This project involved users in designing a visual language which they could easily comprehend and use.

1.3. Proposed solution

Unlike the internet and print media, radio and television reach almost all South Africans [4]. For this reason, the project made the assumption that most young South Africans would be familiar with a simple visual repertoire learned from television cartoons and the printed comics, which often appear in school textbooks or newspapers. This suggested that an application which provides a platform for users to code their programs visually might be preferable to the text-based code needed to control Processing and Arduino. The application would use the Ikamva Youth students’ process of learning and their visual representations of logical concepts during Arduino programming lessons as a major source of the visual language to be used in the interface design. The application would need to provide the necessary platform for students to express the concepts learnt from the lessons and a 3 week lesson plan focusing on developing a basic procedural repertoire (Appendix A).

The system should also represent the students’ understandings of concepts and the way it has been taught through the Arduino.

1.4. Report outline

The next chapter looks at the problems facing learning how to code with Arduino, explains the benefits of visual programming languages and looks at some existing visual programming languages for Arduino. Additional information is then given about Ikamva Youth and the representative is collaborating on this project.

Following from this, Chapter 3 discusses the overall design process used and explains the various iterative steps. Chapters 4 and 5, talk about the design, implementation and evaluation steps lessons, technology probes, high fidelity prototypes and the final iteration respectively. Chapter 6 reflects on the project and its implications on ICT4D.

Finally, Chapter 7 concludes this project report and gives direction for future work that can be conducted.
2. Motivation and Background

2.1. Introduction

This chapter explores the current visual programming systems developed Arduino and VPL that have been developed and used to teach programming.

2.2. Arduino

The higher level shortcomings of Arduino are the Arduino programming language itself. Programs are compiled into a C/C++ but Java, Python, Processing are used on a personal computer to run communications with the Arduino. The process of code to implementation can be a complex set of steps for a novice programmer as the code implementation is either C/C++, python or Java. This problem forms part of a larger set of problems related to programming in general and is not exclusive to Arduino. Dealing with the psychology of programming, the problem is described as a conflict between what users want and what users need [10].

2.2.1. Using Arduino to teach programming

Arduino has been used to teach child friendly programming languages [6]. There are conflicting ideas about the curriculum for introductory programming. Nonetheless central topics which appear in most curricula include recursion, procedure and variables. Common strategies involve the use of graphical programming or robots that are programmable [6]. It is shown that these topics are important to children learning of a programming language as it provides meaningful information to programming [6]. One of the implementations was a paper based Arduino prototype.

The AdMoVeo study that has been running since 2008 with 400 students was conducted for industrial design students with the main aim to increase the basic programming skills of students [5]. The study was conducting with the knowledge that the ability to link theory and practice is difficult. This ability to link the two frustrates students. The study created a robotic platform based on Arduino. Hu et al. designed their own software so there was no C++, only a Processing program environment [5]. The project was successful as it motivated and encouraged students to explore their creativity with their enthusiasm in graphical and behavioural design [5]. The project also invited the learners to apply and grow knowledge in their design projects [5].

2.2.2. Tangible and graphical programming languages

TurTan uses Logo which is a graphical programming language to encourage its users to be creative and explore the potentials of programming [11]. This example of a tangible programming language proved to be a positive with users, much of this positivity being credited to its tangibility attribute. Users found it to be enjoyable and easy to understand. Gallardo concluded that a tangible programming language yields good learning of basic programming language skills with its users. As stated by the HAP, teaching a programming language is forbidding, as with all programming languages like Processing the procedures are hidden in the code. TurTan much like Arduino can be used to create a tangible programming language learning experience.
Varesano’s master’s thesis explores the current situation in the Arduino programming language [13]. Currently, Arduino users program use the Arduino IDE which is the same as coding in C/C++. The current programming process is purely text based. Each program written is called a sketch. Before being deployed onto the Arduino board the sketch is compiled and converted into a c program. Throughout Varesano’s paper he implements a variety of technologies built on top of Arduino to achieve tangible human computer interaction technology. These examples are the Palla prototype, translated from Italian means ball, which was implemented as a spherical game controller [13]. The aim of Varesano’s future work is to incorporate user experience into the tangible component of prototypes he builds [13].

2.2.3. Existing Visual Programming Languages for Arduino

Max is a visual programming language for music and multimedia developed and maintained by San Francisco-based Software Company Cycling ’74 [26]. During its 20-year history, it has been widely used by composers, performers, software designers, researchers, and artists for creating innovative recordings, performances, and installations [26].

The NI LabVIEW Interface for Arduino Toolkit helps the user to interface with the Arduino microcontroller using LabVIEW [25]. Together the toolkit and LabVIEW can control or acquire data from the Arduino microcontroller. Once the information is in LabVIEW, it can be analyzed using the hundreds of built-in LabVIEW libraries, develop algorithms to control the Arduino hardware, and present findings on a polished user interface. A sketch for the Arduino microcontroller acts as an I/O engine that interfaces with LabVIEW VIs through a serial connection. This enables the user to quickly move information from Arduino pins to LabVIEW without adjusting the communication, synchronization, or even a single line of C code. Using the common Open, Read/Write, Close convention in LabVIEW, this provides access the digital, analog, pulse-width-modulated, I2C, and SPI signals of the Arduino microcontroller [25].

Simulink can be used to develop algorithms that run on Arduino, specifically Arduino Mega 2560 and Arduino Uno boards [27]. This support is useful for modelling and testing algorithms for control and robotics applications. Simulink built-in support for the Arduino platform includes automated installation and configuration, a library of Simulink blocks that connect to Arduino I/O, such as digital input and output, analog input and output, serial receive and transmit, and servo read and write, interactive parameter tuning and signal monitoring of applications running on the Arduino Mega and model deployment for stand-alone operation [27].

Modkit Micro is a graphical programming environment for many microcontroller platforms including Arduino [28]. Modkit Micro was created to bring the world of microcontroller programming to the masses [28]. The application helps users’ code complicated algorithms through a simple and powerful visual programming interface.
2.3. Ikamva Youth

Ikamva Youth’s full name is Ikamva lisezandleni zethu; it is isiXhosa for *The future is in our hands*. A key part of the organisation’s success is that the Ikamva students take responsibility for, and ownership of, their own education. [17] The organisation provides the support, learning resources and information that may be missing from the students’ external lives, on the principle that with this support, the learners themselves should generate their own culture of achievement and be able to close much of the gap between them and their more privileged counterparts [17].

Ikamva Youth equips learners from disadvantaged communities with the knowledge, skills, networks and resources to access tertiary education and/or employment opportunities once they matriculate [3]. Ikamva Youth aims to increase the collective skill level of the population, to grow the national knowledge base, and to replicate success in more communities. A non-profit organisation, established in 2003 and formally registered in 2004, with branches in three provinces in South Africa, Ikamva Youth currently operates from Khayelitsha, Nyanga and Masiphumelele in the Western Cape, Ivory Park in Gauteng, the greater Cato Manor area and Molweni in KwaZulu-Natal. While learners enroll at Ikamva Youth when they are in grades 9, 10 and 11, the programme’s success is ultimately determined by the number of grade 12 learners who access tertiary institutions and/or employment-based learning opportunities when they matriculate [3].

The Ikamva Youth model draws from a large and growing pool of volunteers made up of students (from nearby universities) and local professionals. Like Walton, I joined the organisation as a volunteer for six weeks, where I contributed to the Ikamva Codes project which aims to teach Ikamva Youth students coding skills.

2.3.1. Human Access Point

Walton, is a senior lecturer in the Centre for Film and Media Studies at the University of Cape Town. She has been involved at Ikamva Youth through several teaching and community projects. Her current focus is the Ikamva Codes project. Walton works with developing new projects around visual approaches to programming, and introducing young people to procedural thinking through Arduino and Lego Mindstorm robotics. She has taught students from Ikamva Youth from 2012 and has a good understanding of the challenges involved in introducing students to coding.

2.4. Summary

Using Arduino to teach introductory programming has been successful over many projects. The main problem of learning to code for Arduino for novice programmers in the context of Ikamva Youth is the text-based code and a visual alternative could be explored.
3. Design Chapter

3.1. Introduction
As motivated in the previous chapter, the use of Arduino with novice programmers yields effective learning of programming concepts through participatory design.

Ethnographic methods help provide insight to users. First-hand investigations help to understand the particular group of users. This is vital to designing a visual programming language for these users. A more intrusive technique such as direct questioning provides validation of findings; helps combine observations and gain additional information. Dr Marion Walton, acted as the human access point (HAP), to provide insights into the user group and has given good understanding of the users and what the users knowledge.

The time given with users was limited to a 6 week period of two hour lesson sessions every Friday afternoon. These lessons were formulated using Arduino to teach programming concepts. For this reason the participatory design technique of technology probes and user testing of prototypes were utilised. Since users were unfamiliar with the technology, contextual inquiries were made by interviewing users before the prototype was introduced. The initial technology probe will be prototyped by users, this probe allows for users to be part of a design process where they could communicate their preferences and discuss strengths and weaknesses in the design. Through users’ input, possible resources to be used in the visual programming language were identified.

These techniques helped me to develop an understanding of the young women’s mental models and how they conceptualised procedural control of the hardware. From the knowledge gained from the technology probes and through the design sessions done with the users in co-design; applying what is learnt from users to design the interface.

The design and development processes are described by the user centred design process below. This diagram shows the iterative process used for this project. Due to the time constraint of teaching the lessons and interviewing the users to formulate a design the user centred design process required an adaptation step.

The first step was to plan, determine all activities needed and the necessary resources. Followed by research, before designing, understand the users’ goals and tasks and the market needs. Then design, define the system from the user perspective. Adapt based on need for changes discovered during development. Lastly measure usability, which is comprised of effectiveness, efficiency, and satisfaction.

![Figure 1 User centred design](#)
3.2. Understanding of users

Understanding the user is undoubtedly fundamental to the process of user-centred design and as such, a significant amount of time and effort was put into understanding who this system was being designed for and how the young women’s responses to the lesson material can be extracted and augmented into the system.

3.2.1. Arduino lessons

The lessons are the only time spent with the users in gaining knowledge of their understanding of the Arduino and programming concepts being taught.

3.2.2. MakeyMakey applications

As an incentive of the lessons, time is left at the end of each lesson for the students to make use of the MakeyMakey. This involves playing interactive games as a class.

3.3. Technology probes

Technology probes used to gauge the knowledge and understanding of the users. Probes are used to gain information about the learners at Ikamva, aiming to prompt responses in an engaging method [1]. Probes encourage users to reinterpret them and use them in different ways. Another goal of technology probes is its ability to raise participants’ curiosity and attention. Using probes in this project is important in generating ideas from the users through interactive design techniques [1].

3.4. Prototypes

This will be evaluated by the learners and feedback gathered will be used as key information for the next prototype i.e. high-fidelity prototype. The high-fidelity prototype will build from knowledge from previous designs done with the learners. This will be an interactive system that does not have functionality. This is then taken back to the learners to gather feedback again.

After the prototype phase, an implementation of the interface will be done. This will have functionality and follow from the design in the previous sessions and knowledge gathered from the learners. This implementation will address a subset of Arduino behaviours.

3.5. Evaluations

At the end of every iterative design cycle, an evaluation should be conducted on the given prototype. This allows the progress of the prototype in meeting the users’ needs to be assessed. There are many different evaluation methods that can be applied, as described in the book Mobile Interaction Design [1]. However, it is vital to choose the correct form of evaluation to gain the required insight and results appropriate to the prototype iteration level.

For the evaluation of a low fidelity paper prototype, conceptual model extraction techniques can be applied. This gives an indication of how the users interpret the interface and highlights things that they do not understand. The ‘quick and dirty’ approach can also be applied by meeting informally with representative users and asking them to give feedback on
the design. In this project, the HAP acted as a representative for the user in the technology probe iteration and a combination of ‘quick and dirty’ and conceptual model extraction techniques were applied. The findings of these evaluations are usually subjective and unstructured, with qualitative comments and suggestions.

Conceptual model extraction can also be applied in the high fidelity iteration on the horizontal prototype. This is particularly needed if users have not been included in the previous design iterations to ensure that they are able to understand the system. A formative evaluation can also be conducted with a number of users to highlight any errors that have been missed and discover navigational difficulties that arise. A formative evaluation is also used to validate that the system is on track to meet its goals and discover any improvements that can be made. As with all user evaluations, it is imperative to gain the relevant ethical clearance and ensure that the users fully understand that it is the system, and not them, that is being evaluated.

Interviews can be used as a means of allowing the user to reflect on their experience and describe the things they liked, their frustrations and the changes that they would recommend. However, it is important that an interview be conducted in conjunction with another form of evaluation to ensure that important information is not missed. Since user responses are often very subjective, and can be biased in an attempt to please the researcher, it is often not possible to draw significant conclusion from an interview alone. In this project, an interview was conducted at the end the high fidelity prototype iteration.

3.6. Summary

This chapter has introduced the methodology and tools used to develop the VPL application interface. The following three chapters will show how these have been used in the different iterations involved in designing and creating the final systems interface, functionality and evaluation.
4. Technology probes and Lessons

4.1. Introduction
The use of technology probes as an aid in teaching the programming lessons. The purpose of the Arduino programming lessons was to teach three important concepts and structures of structured programming. The lessons are different to the traditional teaching approaches which describe programming techniques by examples of code. By moving away from the traditional methods to demonstrating of programming the students are able to observe the composition of programming concepts which leads to students developing authentic problem solving skills in an introductory programming class [19].

The focus of the lessons is to teach students to effectively solve problems which would help them understand how the Arduino’s are programmed. As most programming languages are distinctive, there are concepts and structures found in common [20]. There are three main types of control structures:

- **Sequence** of instructions, in a sequence structure must all be carried out.
- **Selection** structure, certain instructions will only be carried out IF a certain condition is found to be true
- **Iteration** structure, certain instructions will be repeatedly carried out until a condition is no longer true (DO-WHILE) or alternatively until a condition is no longer false (DO-UNTIL). Such structures consist of a loop which takes the program back to the top of the instructions to be repeated

The lessons taught to the students at Ikamva were designed with the use of Arduino boards and MakeyMakey applications to further solidify concepts taught in the lessons.

After a concept was taught, students were asked questions in the classroom about how they would describe and code the demonstrated Arduino behaviour. This was done by direct questioning and by the students drawing out the understanding on cardboard.

The HAP, Dr Marion Walton, provided insights into the visual literacy significance of the students’ drawings and explanations.

4.2. Learners
Throughout the period of time whereby lessons were given there were a number of students who attended the lessons. Some of whom had programming experience already. It was important that these groups of students be taught separately from the group which this project focuses on. Namely, the four female Ikamva Youth students attended all of the lessons in the time period. These students represent the potential users of the system. These young women from Ikamva were interested in programming and had no prior experience of any computer languages. The four female students were either in grade 10 or 11. Their first language is Xhosa and they all took the same school subjects. The students have little prior exposure to computers and limited access to a computer lab.
The lessons took place at the Ikamva Youth, Makhaza in the computer lab that is part of the Nazeema Isaacs Library, Makhaza, Khayelitsha.

4.2.1. Permission and consent

All students each signed a release form with a parent or guardians at the beginning of the set of lessons that were taught at Ikamva Youth. The relevant ethical clearance was received from Ethics in Research Committee of the Faculty of Humanities from the University of Cape Town and they approved the attention to ethical principles of this study. This application was approved under Dr Walton’s research.

4.3. Lesson 1: Sequence

4.3.1. Probe
The probe used in this case was a simple blinking LED (light-emitting diodes) set up Arduino board, as seen in figure below. The circuit behaviour is such that the LED blinks on and off.

The reason for using this example is to illustrate the programming concept of sequence. To expand further, sequence involves completing a set of instructions in a particular order. The blinking LED probe demonstrates the concept of sequence.

The class example of this programming concept was how one would turn on the lights. The students were asked how they would describe the set of instructions to turn a room light on and off.

![Sequence probe](image)

4.3.2. Findings
The observations from the lessons lead the users to give a variety of example to explain the concept of sequence (Appendix B). One of the users expressed the drinking and refilling of a coffee cup to emulate the same behaviour of a flashing LED.

The students showed a firm grasp of the sequence concepts, this is supported by their evidence of their examples (Appendix B). Learners enjoyed the introductory lesson to Arduino and showed great enthusiasm toward the MakeyMakey. In this lesson the students used the MakeyMakey to play an online version of Pacman.
4.4. Lesson 2: Selection

4.4.1. Probe
The probe used to demonstrate selection was an Arduino set up with push buttons. The point was to illustrate the purpose each button has when pressed. The probe's behaviour was as follows, only if one of the buttons is pushed the LED will light up. If both buttons are pushed this triggers no behaviour.

This probe was designed to demonstrate there are certain conditions that need to be met in order for the Arduino to behave in a certain way. This illustrates the concept of selection where certain behaviours occur if a condition is met, in the case of this probe when a push button is pressed.

The class example of this programming concept was when it is raining you would use an umbrella, this was discussed on the white board.

4.4.2. Findings
The students interacted with this probe by a pair of students attempting to press the buttons simultaneously. They asked many questions on the hardware of the Arduino, as this probe had more components to explore than the previous lesson.

When students were asked to describe in their own words this example the feedback received varied from when it is raining a convertible car will have its roof up to the example of if a woman is pregnant she will give birth. The concept of having a condition and behaviour which followed was well learned by all students.

4.5. Lesson 3: Iteration

4.5.1. Probe
The probe used was an Arduino board with a potentiometer which changed its lighting pattern as the position of the potentiometer changed. Thus, illustrating that as a condition is met the Arduino behaviour alters. In this probe, while the potentiometer is in a range of values the LED is lights up at a certain pace depending on its value on the potentiometer.
The probe was designed to demonstrate the iteration concept that the Arduino will behave differently when a certain condition is met, in this case when the potentiometer is in a certain position the LED behaviour changes. This illustrated the DO-UNTIL iteration structure.

The class example discussed was that if a spaza shop owner asks you to code a program to write out the quantities and prices of a packet of chips per quantity. The students were then asked if they could do it in a way that uses what they learnt from the previous lessons.

![Iteration Probe](image)

**Figure 4 Iteration Probe**

4.5.2. Findings

This lesson brought in learning from the previous lessons. The students required some guidance but were able to conceptually think of how this probe behaviour would be coded.

The students were able to successfully describe what was happening with the Arduino when the interacted with it.

The examples given back from students included, a television will turn on when a specific TV is scheduled to appear on a channel. Another example was that a car will continue “drive” on condition it remains within a speed limit (Appendix B).

4.6. Paper based feedback

For each of the lessons there were 4 students who attended and participated. These are the students who provided the feedback to gain insight and motivate the design of the system.

4.6.1. Requirements

The requirements gathered from the lessons indicate that the students are comfortable with expressing their ideas visually through cartoon-like scenarios.
4.6.2. Visual language vocabulary

A visual language can be extracted from the lessons using the feedback from the students during lessons to formulate a visual language vocabulary of scenarios and examples which explain their understanding of concepts (Appendix A).

4.6.3. Narrative representation vs. Conceptual representation

Upon discussion with HAP, it can be seen that the diagrams produced by the students are using narrative processes to represent their understandings. This is seen as the visual structures of representation are presenting unfolding actions and events, processes of change or transitory spatial arrangements [2]. Conceptual representation represents participants in terms of a more generalized, stable and timeless essence, in terms of class, structure or meaning [2]. Analysing the diagrams from the lessons, a strong visual narrative process in visual communication is being portrayed.

Using what is learnt and observed in visual narrative processes the following can be said with regard to the narrative representations. Although very few vectors are used in the feedback from students, it is clear that there are non-transactional actions, non-transactional reactions and transactional reactions being communicated. These are evident in the students drawings depict elements that emanates from a participant but does not point to another participant.

4.6.4. Design based on lesson feedback

Using the students’ feedback from the lessons, the following design was created.

Figure 5 Student sketch illustrating sequence

Figure 6 Student sketch of selection
These sketches were used in establishing the following design:

Figure 8 is created by taking components of the sketches of the students and providing a canvas for students to create something very similar as seen in their paper based feedback. The Add Block button originates from students wishing to have either one block to draw their images on or multiple. This is seen from figures 5-7 where students have either used multiple smaller sketches to show their understandings of a particular concept.
The repeat button is associated when a user would like to create a condition which is the condition for repetition of a sequence of events as seen in figure 7. In figure 7 the sketch shows that until a speed limit condition is met the car will continue to drive. The repeat button is indicative of having the speed limit in a particular block to illustrate its meaning throughout an example of a sketch.

The order button is inspired by having a sequence of blocks as seen in figure 5. This allows users to order the blocks in particular sequence.

The buttons on the left are the objects that the user may utilize to draw their sketches. These originate from the visual language vocabulary defined by the students from the lessons and examples that were discussed.

4.7. Summary

This chapter dealt with the technology probes used in the lessons which were taught at Ikamva Youth. The lesson plans and structure of lessons using were discussed. Content which supported of lessons were discussed along with the technology probe which demonstrated the programming concept being taught in that lesson.

For each lesson, the probe used was described with its programmed behaviour. Afterward the lesson material and findings from each lesson were discussed.

Lastly the paper based feedback was analysed to extract the necessary visual representations to feed the design of the interface.
5. High fidelity prototype and Final Iteration

5.1. High fidelity prototype

5.1.1. Introduction
This chapter looks at the high fidelity prototype that was developed from the findings from the previous chapter. The horizontal prototype combines the designs found from the lessons into an interface. The interface suggested from the findings leads to a storyboard or comic book type application.

5.1.2. Design and implementation of Horizontal prototype

5.1.2.1. Comic book interface motivation
As seen from technology probes chapter, the students express their programming concept learning in the form of diagrammatic sketches of scenes which are familiar to them. The use of verbal language is minimal and many of the paper feedback points to cartoon like sketches.

![Figure 9 Example sketch of selection](image9.jpg)

![Figure 10 Example sketch of iteration (TV example)](image10.jpg)

5.1.2.2. Development platform
The development platform used to develop the prototype is a Java application. This platform was chosen to allow for a user interface to be developed that the users would be able to understand, given their limited experience with computer technology. The Java platform allows for relatively quick development of a system, given the time constraint of this project. A Java application is a good choice as it has the potential to be expanded into a fully functional system that can verify whether sketches are logically correct at a later stage.

5.1.2.3. Outcomes that were implemented
Each of the 3 types of programming concepts was accommodated in the design, namely sequence, selection and iteration.
This involved using the visual language vocabulary extracted from the lesson. This vocabulary consisted of the terminology and examples of programming concepts that the students drew and communicated in the lessons. This visual language vocabulary was implemented as drag and drop buttons that could be placed in the applications drawing canvas to be assigned to certain blocks in order to create the sketches.

A comic strip inspired interface was implemented that pieced together the visual language vocabulary defined from the students and an interface that would allow users to create their sketches. Current Arduino IDE’s focus on the iconic nature of the Arduino components rather than focus on closure while representations such as storyboards and comic strips give closure over iconography. Providing closure linked with what the students produced from the lessons and their understandings of the programming concepts from their paper sketches.

5.1.2.4. Outcomes that were not implemented
As this iteration required a high-fidelity, horizontal prototype, elements of the backend were not yet implemented. In this iteration the focus was on extracting the students understanding of concepts into a visual language that would make sense to them. The backend will be implemented once the necessary adjustments that this iteration’s evaluation stage requires have been addressed. The following is a list of the backend elements that have not yet been fully implemented:

- The functionality of verifying a sketch for each of the programming concepts is yet to be implemented. This gives users an indication as to whether their sketch is logically correct or not. The verification of a sketch requires the application to check the logic of a sketch and if it adheres to the programming concepts.

5.1.3. Evaluation and findings

5.1.3.1. Test subjects
Two Ikamva Youth students, both female, participated in this evaluation. These students represent the potential users of the system. The test subjects attended all of the lessons (chapter 4). This meant that the subjects had been taught the concepts and is familiar with the vocabulary used in the application as it was drawn from the lesson examples and discussions. The subjects were each given R50 airtime for their participation.

It is important to note that the evaluations took place at Ikamva Youth, Makhaza. The absence of the other two participants can be linked to either their commitment to other afternoon activities such as the tutoring program at Ikamva or personal. The absence of these users require the researcher to rethink the evaluations to focus more on the understanding of the prototype rather that the task completion of the evaluation.

5.1.3.2. Permission and consent
All students each sign a release form with a parent or guardians at the beginning of the set of lessons that were taught at Ikamva Youth. Verbal consent for participation in this evaluation was given by each student at the start of the evaluation. The relevant ethical clearance was also received from Ethics in Research Committee of the Faculty of
Humanities from the University of Cape Town and they approved the attention to ethical principles of this study. This application was approved under Dr Walton’s research.

Before the evaluation and interview the test subjects were explained what the evaluation would entail and asked questions on their understanding after the lessons and of their familiarity with visual literacy such as comic books. It was stressed that it was assessing the system and not themselves and that their answers, comments and suggestions would be anonymous and in no way linked to their names. They were told that they would be compensated with R50 airtime for their time. This was done in order to ensure that they were happy participating in the evaluation as this would lead them to have confidence and make them comfortable.

5.1.3.3. **Methodology**

5.1.3.3.1. **Concept model extraction**

Since this was the first time that the users had directly interacted with the system it was vital to test whether the design decisions made aligned with their understanding. Conceptual model extraction gives insight into the users’ first impressions of the system and highlights any misconceptions and confusion that they may have [1]. This evaluation by concept model extraction was conducted as follows; each student was walked through the system and explained what she expected to happen. If the results were not as she anticipated, the possibilities were discussed and recommended changes were made.

![Figure 11 Screen shot used in concept model extraction](image)

5.1.3.4. **Structure of evaluation**

This formative evaluation contained unstructured questions that followed the flow of the system as well as a discussion around the test subject’s interpretation of the system. The questions that were asked during this evaluation attempted to gain an insight into what the test subjects understood of the application. In addition, the manner in which the students interacted with the system was observed and the times that they got confused or did not interact with the system’s interface as anticipated, was noted.
Test subjects were also asked background questions on technology in general and what technologies they use. Other questions were asked as to what experience the subjects have had with visual literature such as comics and graphic novels.

As the test subjects have not interacted with such an application, a basic introduction into how the application works was given. The application designed in such a way so as the vocabulary used in the drawing options were familiar to the examples taught and originated from the lessons. The students were asked to comment on their understanding of the application as well as if the drawing options made sense.

Each of the students was asked to redo a task from each of the lesson. These tasks being, for each lesson the students were asked to give an example of the programming concept taught in their own words or draw it out. The students would then walk through how they would create a sketch using the drawing options found in the applications interface to draw out their understanding.

These tasks were to use the application to create a drawing of each of the concepts learnt. These programming concepts being sequence, selection and iteration, as learnt and discussed in lessons.

The evaluation was concluded by asking the test subject if there was anything in the application that they found confusing, if they had any suggestions and if there was anything that they would like to be added. They were then asked if they had an option between using paper or the mobile phone application for capturing and reporting on cash-flow in their business, which they would prefer and why. They were then thanked for their valuable input and time spent on the evaluation and received their R50 airtime, after the evaluation had been concluded.

5.1.3.5. **Findings**
The observations gained during this evaluation led to the findings discussed in this section. In addition, the observations also led to a list of required changes that needed to be made. This can be found below. The main findings in this section have been broken down into four key areas: how easy the subjects found using the system to be, the navigational issues that arose, the subjects’ understanding of the system and the perceived value that the subjects believed the system would have for learning programming concepts.

**Ease of use**
The subjects have never use an application such as the system developed. When shown briefly how to use the application their uneasiness was diminished. As the subjects interacted and explored the application they developed much more confidence.

Although the subjects were often not able to infer what a certain button would do, they understood its purpose once they tried it out. Many of the subjects stated at the end that they did not find anything in the system confusing or counter intuitive. They expressed that “if I had more time to learn with it, I would be able to understand it better”.
Navigational Issues
The users did not find the drag and drop interface intuitively. One of the subjects expressed that it would be “easier if you could just press the picture” to get to the icon on the selected block.

Subjects were asked to create examples of programming concepts using the application; this was called creating a sketch. When creating a sketch, subjects needed to be reminded that they could add blocks to the screen.

Understanding the System
The subjects understood most of the icons which were part of the drawing vocabulary (Appendix C). This was due to each of the icons being extracted from examples which came across in lessons. When asked about the other icons these being adding blocks, deleting blocks or verifying the sketch, they were not familiar with having such functionality. Subjects expressed that they did not understand the purpose of the functions unless they were explained what it does.

The subjects did express if they had more time to understand the application, they would be able to use it effectively.

Perceived Value of System
When asked if the application was useful and an effective way of communicating a programming concept both subjects strongly agreed that it was useful and an easy way to understand the examples learnt in lessons. The subjects expressed that they enjoyed being able to make examples which they thought of in lessons. One of the subjects expressed that it was “clever” that her example could be made into the application.

5.1.3.6. Tasks Completion
As described in the evaluation, each of the subjects were asked to create a sketch illustrating each of the programming concepts learnt from the lessons.

Both were successful in creating an example of the first programming concept i.e. sequence. Neither required much assistance other that reminding them they could add blocks.

The task of creating a sketch which showed selection was the quickest of all three tasks performed by the subjects. Subjects were able to give multiple examples of selection. This was the most successful task of all three for both subjects.

The third concept, iteration, was successfully created by one of the subjects. The other subject required assistance. Although this subject was able to explain the programming concept effectively, the subject struggled to use the application to create a sketch without assistance.
Figure 12 Screen shot of student creating a sketch of sequence

Figure 13 Screenshots of student creating a selection concept sketch
5.1.3.7. List of required changes
This list is a summary of the changes that need to be made as a result of the conducted evaluation. Most of these changes were recommended by the test subjects themselves, while some of the less significant ones were discovered while observing the subjects’ use of the system.

- “Add block” needs to change to “Add new block”
- Adding text to some of the icons
- Increasing the size of icons
- Change the icons which are not easily understood to a clearer icon
- The groupings of the drawing options according to what category they fall under in the vocabulary extracted from the lessons should be grouped by a colour for each categorization.
- Remove the drag and drop interface to click-buttons.

5.2. Final Iteration

5.2.1. Introduction

The final iteration implemented changes from the high fidelity prototype evaluation. This iteration includes adding the functionality of a verification check to one of programming concepts using the vocabulary used for the application was implemented. This was done to check that the functionality of verifying code sketches or programs drawn in the application would be able to conduct a verification check for at least one of the programming concepts.

5.2.2. Development changed made

From the findings and results of the high fidelity prototype, all the recommended changes were implemented with one exception that is discussed below. In addition, the verification check for one of the programming concepts was implemented i.e. selection.

5.2.2.1. Changes not implemented
As mentioned, one of the changes that the users requested was not able to be implemented. It was recommended that the drag and drop interface be done with completely.

5.2.2.2. Changes implemented
The other changes suggested were implemented:

- “Add block” needs to change to “Add new block”
- Adding text to some of the icons
- Increasing the size of icons
- Change the icons which are not easily understood to a clearer icon
- The groupings of the drawing options according to what category they fall under in the vocabulary extracted from the lessons should be grouped by a colour for each categorization.
- Implementing the verification of one of the programming concepts, namely selection.
6. Reflection on ICT4D

6.1. Introduction

This chapter looks at the drawbacks of user centred design in developing projects and its implications to Information and communication technologies for development (ICT4D). It is a combination of a reflection on this project and the outcomes that can be extracted which may contribute to better HCI practices in developing projects.

6.2. UCD drawbacks

Successful user centred design is achieved by knowing the user (Section 3), while in development projects it is important to continually question and explain assumptions that may be implicit in design decisions. In order to manage with these unique challenges, we must re-examine many established areas in HCI including [21]:

- **Interaction Metaphors**: Exploring beyond the Western-centric Windows, Icons, Menus and Pointers (WIMP) metaphor to other interaction metaphors that are more culturally and socially applicable to the intended user groups [24]
- **User Analysis**: Developing methods to most effectively comprehend the users and their context, practices and wants by understanding their Socio-Cultural and Economic differences unique to them [22]
- **Interaction methods**: Localization and customization / alternatives to traditional input output methods [23]
- **Evaluation methods**: Thinking outside traditional methods by making evaluation more appropriate to the target user audience to elicit accurate and actionable feedback [24]

UCD and Participatory design are vital in assisting designers better understand their target users. Addressing UCD in development issues is one part of understanding users effectively, but it is a regular occurrence that researchers are also confronted with broader issues while work together on projects in the developing world. If an ICT project is to be successfully implemented, it is important to consider these factors as well. They include [21]:

- **Project aims**: In development projects one of the key goals is enhancing the long term capability of host communities after the project ends
- **Profoundly different contexts**: Developing world projects may widen the analysis and design space to take into account issues such as local economic conditions, historical context, political structures, resource availability, technological infrastructure, and long term financial sustainability of solutions.
- **Design constraints**: Developing-world projects are, by definition, conducted against a background of severely constrained resources.
- **Cultural disparities**: Developing-world projects may involve cultural and language barriers between technologists and hosts, as well as perceived differences in status. These factors can be more extreme than those encountered in projects in the developed world, leading to difficulty in establishing effective partnerships.
Sustainability: Ultimately, developing-world projects are only successful if they are sustainable by the community. Identifying the contextual, cultural, and technical factors are key to making projects sustainable

6.3. Implications on ICT4D

In this project the following issues found in UCD came up. The benefit of using UCD methods to interview with people work well in development in an office environment is good. In the context of this project it does not quite work out. In this project one of the key issues was getting regular attendance by students. Reasons for students not attending the lessons were either they had higher priority commitments such as working on school projects or due to poor weather.

Another issue was the cultural disparity in both cultural and language barriers between the researcher and the users. This problem relates to communicating ideas between both parties. It also ensures that participants feel at ease and the data extracted from the users is accurate. In this project it was clear that the female participants felt more comfortable without any of the male students in classes.

As part of learning from the project one has to rethink the way one conducts research in ICT4D. The constant rethinking and adaptation of plans due to the environmental changes in the project is a valuable lesson to be learnt.
7. Conclusions

The design question posed was to formulate a visual programming language for Arduino using HCI methods which resulted in a system that used the Ikamva Youth students’ process of learning and their visual representations of logical concepts during Arduino programming lessons as the source to the visual language implemented in the interface design. The system was required to be appropriate for the Ikamva Youth context and applicable to the Ikamva Codes lessons that teach learners programming skills.

In order to develop the proposed system, a user-centred approach was taken, applying participatory design techniques. The user-centred design approach recommends including the users in the design process as early as possible, however, when introducing a new technology and concepts with are unfamiliar, it is very difficult for users to give useful input when they cannot visualise the system. However, using the HAP, someone who understands both teaching the learners and the users’ needs, as a resource of designing lesson structure and analysing the feedback received from each of the lesson it was possible to extract data for the designs of the interface.

By adopting this approach, the feedback received from the students learning and their visual representations of logical concepts during Arduino programming lessons was successfully translated into the high fidelity horizontal prototype. This can be seen by the findings of the conceptual model extraction formative evaluation not raising any significant interface changes that needed to be made. Without the teaching the lessons and working through the students understandings of Arduino and the programming concepts, there would likely have been more terminology needing to be corrected and icons that the users did not understand, amongst other things. Instead, the changes that resulted were mainly from usability issues connected with the drag and drop interface, which could only be evaluated at the high fidelity level.

During the high fidelity prototype phase of the project, user input was vital for evaluating the interface. From the evaluations of the high fidelity prototype, there were a number of changes that needed to be implemented to the system. In the final iteration these minor changes were made to application to enhance their user friendliness and make them easier to understand. The code verification of the programming concepts was implemented for a subset of the vocabulary developed for the application.

7.1. Research questions

The three research questions that this research project aimed to answer were:

1. What are the verbal and visual languages that can be extracted into a visual programming language?
2. Are technology probes and ethnographic study methods suitable to design a visual programming language?
3. Can HCI participatory design methods help to formulate a visual programming language for Arduino?
The system developed for the Ikamva aimed to extract visual and verbal language of students into a visual programming language for the students. What was extracted was a language that accommodates for simple code structures that solidify the programming concepts learnt in the students' lessons. The language formulated was close to simple comic book like sketches and simple in terms of logical complexities.

It is felt that the suitability of technology probes and ethnographical study methods in designing a visual programming language in this project were correct in formulating a visual programming language given the students background and prior knowledge. The evaluation of the designs, however were not deal. This was due to user group’s absenteeism during the evaluation stages and the project running out of time to provide evaluations of the final iteration.

The research in this project confirms that participatory design methods can be used to extract a visual programming language. Technology probes and ethnographic study are suitable when working closely with a user group to understand their visual and verbal languages. The research conducted supports the use of these HCI participatory design methods as the data and feedback received from the user group yielded valuable insights and formed a foundational understanding of programming concepts.

7.2. Future work

In this research project, there are improvements that could be made to enhance the design. The biggest constraint in the project was spending time with the users, getting users to attend lessons and participate in evaluations of the high fidelity prototypes. This opens an opportunity into formulating a viable model of developing with such a user group so that the process it not as dependent on the researcher and user meeting in that environment. This would be a design process that is independent of the researcher going on site to gain evaluation.

Possible areas to consider would be to teach and gain insights from students using a web application. This would allow students to use the application at their pace and suitability of time and convenience. Other technologies which are more familiar to the user group such as mobile technology could be an opportunity for further development. The possibility of creating a mobile application for smartphones or a system that works on the MXIT platform. These mobile solutions fall under technology which is more familiar to the user group’s technological experience and could further reinforce the learning from lessons to possibly find use as a programming educational tool.
References


In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 975-984). ACM.


### Appendix A: Lesson Plan

#### Table 1 Planning for each lesson at Ikamva Codes

<table>
<thead>
<tr>
<th>Lesson</th>
<th>Introduction</th>
<th>Demo</th>
<th>Background Logic</th>
<th>Activity</th>
<th>Outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>interview students about their preconceptions about programming and computers</td>
<td>Introductory Arduino behaviours. Flashing LED-illustrates sequence</td>
<td>Discuss having a variable</td>
<td>In groups, students discuss how that would be coded</td>
<td>Introduce students to the potential rewards programming can give and the concept of sequence</td>
</tr>
<tr>
<td>2</td>
<td>questions from previous lesson</td>
<td>Arduino behaviour that uses if else statements (LED that changes colours)</td>
<td>Discuss real life examples which we use if else statements for. e.g. buying sweets, opening a closed door</td>
<td>In groups, students discuss how that would be coded. Using cardboard sketches to give a representation as to how that would look if coded up</td>
<td>Students use cardboard sketches to give a &quot;program&quot; which models some behaviour. Able to gain mental model to if else behaviour</td>
</tr>
<tr>
<td>3</td>
<td>Show students what their cardboard &quot;programs&quot; coded on the Arduino. Questions from previous lesson (recap)</td>
<td>Arduino behaviour that uses for loops (LED that changes ad position of potentiometer reaches certain positions)</td>
<td>Discuss real life examples which we use for loop statements for. e.g. given a list of names, count the number of names</td>
<td>In groups, students discuss how that would be coded. Using cardboard sketches to give a representation as to how that would look if coded up</td>
<td>Students use cardboard sketches to give a &quot;program&quot; which models some behaviour. Able to gain mental model to for loops behaviour</td>
</tr>
</tbody>
</table>
Appendix B: Sketches from Lessons

Sequence Lessons
My understanding of coding

1.
	→

2.
	→

Mendie

road
Selection Lessons
Creating a Code

**Example A**

- When a car is empty, it's starting, moving slowly, and slowly then it stops.
- When the car goes to the garage, it's starting to be normal.
- When the driver is driving again, the car will move normally until the driver stops it or reduce the speed of the car.

**Today's Lessons**

**My own example**

**Example A**

**My Example**

- Generations with: due to children's age?
- Reduce children's age.
- 

---

34
Appendix C: Table of Icons in Vocabulary list

The icons used in the final system are described in Table C-1 below. They come from three different icon sets, as indicated by their URL reference link or have been created for the application.

Clker.com is owned by Rolera LLC, an Illinois Limited Liability Corporation and is under the latest version of the creative commons CC0 public domain dedication (www.creativecommons.org/publicdomain/zero/1.0/). Clipart-Finder.com is a library of free vector clip art. The icons from Wikimedia Commons were created by Everaldo Coelho and YellowIcon [LGPL (www.gnu.org/licenses/lgpl.html)].

Table 2 A list of icons used in the system and their URL reference link

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Icon</th>
<th>URL reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Save</td>
<td>Disk</td>
<td><img src="http://commons.wikimedia.org/wiki/File:Crystal_Clear_device_floppy_unmount.png" alt="Disk Icon" /></td>
<td><a href="http://commons.wikimedia.org/wiki/File:Crystal_ClearDevice_floppy_unmount.png">http://commons.wikimedia.org/wiki/File:Crystal_ClearDevice_floppy_unmount.png</a></td>
</tr>
<tr>
<td>Verify</td>
<td>Tick</td>
<td><img src="http://commons.wikimedia.org/wiki/File:Crystal_Clear_app_clean.png" alt="Tick Icon" /></td>
<td><a href="http://commons.wikimedia.org/wiki/File:Crystal_Clear_app_clean.png">http://commons.wikimedia.org/wiki/File:Crystal_Clear_app_clean.png</a></td>
</tr>
<tr>
<td>Delete</td>
<td>Red arrow out</td>
<td><img src="http://commons.wikimedia.org/wiki/File:Crystal_Clear_action_button_cancel.png" alt="Delete Icon" /></td>
<td><a href="http://commons.wikimedia.org/wiki/File:Crystal_Clear_action_button_cancel.png">http://commons.wikimedia.org/wiki/File:Crystal_Clear_action_button_cancel.png</a></td>
</tr>
<tr>
<td>Ordering</td>
<td>Three numbered white rectangles</td>
<td><img src="http://commons.wikimedia.org/wiki/File:Crystal_Clear.png" alt="Ordering Icon" /></td>
<td>Created for application</td>
</tr>
<tr>
<td>Repeat</td>
<td>White rectangle with blue arrow on each side</td>
<td><img src="http://commons.wikimedia.org/wiki/File:Crystal_Clear.png" alt="Repeat Icon" /></td>
<td>Created for application</td>
</tr>
<tr>
<td>Add Block</td>
<td>White rectangle with addition symbol</td>
<td><img src="http://commons.wikimedia.org/wiki/File:Crystal_Clear.png" alt="Add Block Icon" /></td>
<td>Created for application</td>
</tr>
<tr>
<td>Stickman</td>
<td>Stick figured man</td>
<td><img src="http://www.clker.com/cliparts/e/5/d/9/1194984539216837021man02.svg.med.png" alt="Stickman Icon" /></td>
<td><a href="http://www.clker.com/cliparts/e/5/d/9/1194984539216837021man02.svg.med.png">http://www.clker.com/cliparts/e/5/d/9/1194984539216837021man02.svg.med.png</a></td>
</tr>
<tr>
<td>Icon</td>
<td>Description</td>
<td>URL</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td><img src="http://www.clker.com/cliparts/1/7/e/1/11949845541183631071womann02.svg.med.png" alt="Stick woman" /></td>
<td>Stick woman</td>
<td><a href="http://www.clker.com/cliparts/1/7/e/1/11949845541183631071womann02.svg.med.png">http://www.clker.com/cliparts/1/7/e/1/11949845541183631071womann02.svg.med.png</a></td>
<td></td>
</tr>
<tr>
<td><img src="http://clipart-finder.com/data/png/1294538687.png" alt="Coffee cup" /></td>
<td>Coffee cup</td>
<td><a href="http://clipart-finder.com/data/png/1294538687.png">http://clipart-finder.com/data/png/1294538687.png</a></td>
<td></td>
</tr>
<tr>
<td><img src="http://www.clker.com/cliparts/7/2/9/7/1194986337299109881cone2_bw_ganson.svg.med.png" alt="Ice cream" /></td>
<td>Ice cream</td>
<td><a href="http://www.clker.com/cliparts/7/2/9/7/1194986337299109881cone2_bw_ganson.svg.med.png">http://www.clker.com/cliparts/7/2/9/7/1194986337299109881cone2_bw_ganson.svg.med.png</a></td>
<td></td>
</tr>
<tr>
<td><img src="http://www.clker.com/cliparts/n/v/t/V/0/e/8-o-clock-md.png" alt="Clock" /></td>
<td>Clock</td>
<td><a href="http://www.clker.com/cliparts/n/v/t/V/0/e/8-o-clock-md.png">http://www.clker.com/cliparts/n/v/t/V/0/e/8-o-clock-md.png</a></td>
<td></td>
</tr>
<tr>
<td>Weather Condition</td>
<td>Description</td>
<td>Image Link</td>
<td></td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Partly cloudy</td>
<td>Sun and clouds</td>
<td><img src="http://www.clker.com/cliparts/a/f/c/12284285012032586819sivvus_weather_symbols_2.svg.med.png" alt="Image" /></td>
<td></td>
</tr>
<tr>
<td>Lightning</td>
<td>Cloud with yellow thunder bolts</td>
<td><img src="http://www.clker.com/cliparts/c/9/e/1195427644181646594thunder_raoul_rene_melce_01.svg.med.png" alt="Image" /></td>
<td></td>
</tr>
<tr>
<td>Television</td>
<td>Television</td>
<td><img src="http://www.clker.com/cliparts/d/e/b/0/1224784690715841627schoolfreeware_Television.svg.med.png" alt="Image" /></td>
<td></td>
</tr>
</tbody>
</table>
Appendix D: High fidelity prototype evaluation interview questions

Background questions

1. What forms of technology are you familiar with (mobile, desktop computers etc)?
2. Which technologies do you prefer?
3. Which technologies are you interested in?
4. What do you know of comic books/cartoons (exposure in newspapers, school text books, graphic novels, television)?

Recap of Lessons

Show a picture of each of the technology probes used:

5. What does the probe do (explain the Arduino behaviour)?
6. Can you explain the concept learnt from the probe?
Appendix E: Description of High fidelity Tasks

Task 1: List of instructions

“Using the application, create a drawing of any of one the examples we learnt of the first lesson. Remember the circuit that had a blinking light.”

This task tested if the user could replicate an example that they came up with or learnt from the sequence lessons.

The task required the user to use any of the items found in the visual language vocabulary to draw out the sequence programming concept. This could have been achieved by the user creating a sketch of a cup of coffee being made with a flask. Then being drunk by a person. The task was open to any scenario that could sketch out a sequence of events. The subject needed to have used the add block buttons and order button.

Task 2: Condition and action

“Using the application, create a drawing of any of one the examples we learnt of the second lesson. Remember the circuit that had the buttons that you could press.”

This task tested if the user could replicate an example that they came up with or learnt from the selection lessons.

The task required the user to use any of the items found in the visual language vocabulary to draw out the selection programming concept. This could have been achieved by the user creating a sketch of a condition under which an action is made. One of the many examples could have been a sketch of the sun and a person using an umbrella or eating an ice cream. Another example could be the clock had at 8 o'clock and the next block a bed. The task was open to any scenario that could sketch out a condition that imposed an action. The subject needed to have used the add block buttons and order button.

Task 3: Do this until a condition

“Using the application, create a drawing of any of one the examples we learnt of the third lesson. Remember the circuit that had the dial that you moved and the light did something different after a certain point”

This task tested if the user could replicate an example that they came up with or learnt from the iteration lessons.

The task required the user to use any of the items found in the visual language vocabulary to draw out the iteration programming concept. This could have been achieved by the user creating a sketch of an action happening then stopping after a condition is made. One of the examples could have been a sketch of the television only turning on when a certain show was scheduled to start. The task was open to any scenario that could sketch out a condition that imposed an action until a condition was met. The subject needed to have used the add block buttons, order button and repeat condition button.