User Centred Design

A Usability Report On Choregraphe an Education Software Programming Platform

Table of Contents

1. Introduction

As we rapidly progress further into the modern digital world it becomes much harder for industry's to stay stagnant and refuse change. One such industry that has recently had an overhaul to keep up with modern digital world is the primary and secondary school education system in Australia. As a result it has drastically changed not only how the students learn in and outside the classroom but also how the teacher's now most effectively teach the students.

Various usability study's have been conducted and evaluated around how both student and teacher best use technology within the educational sector and as a result the Technology Acceptance Model (TAM) has been created to best represent how users come to accept and use a given technology (Holden & Rada, 2011, p. 343-350)

The original TAM model is shown below in figure 1.

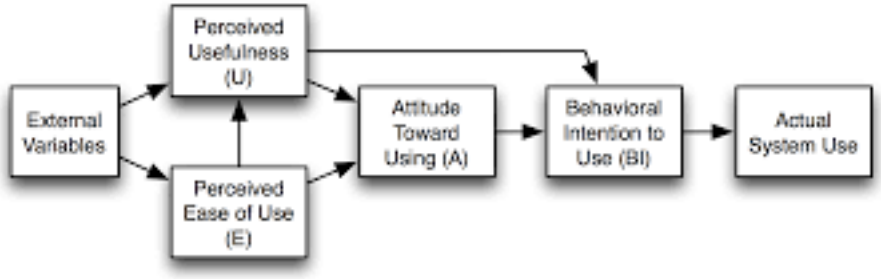

Figure 1: Original Technology Acceptance Model (TAM) (Davis et al. 1989)

Therefore through using the TAM model we can conclude that user acceptance, enjoyment and perceived usability of innovative technologies within education are crucial to the dispersal of those technologies within the class room in todays digital world to both the teacher and the student.

However while using the TAM model we need to also consider that while a specific type of technology might be considered highly usable and useful it will not always be accepted by the targeted users of the technology due to various reasons (Dillon, 2001, p. 12 - 14).

Due to the gaps in the TAM model for this study we will also be using the definition of usability provided by the Nielsen Norman Group (NNG) and The International Organization for Standardization 9241 (ISO) to ensure we encapsulate all areas of usability as well as reduce the risk of overlooking a significant area within the testing phase due to the flaws of the TAM model. The NNG (2012) define usability as an "attribute that assesses how easy user interfaces are to use", where as the ISO 9241 define it as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specific context of use.

Furthermore from the three definitions of usability we will be adapting them to conduct both the usability testing and usability evaluation to test the following areas of an application in accordance to both the User-Centred Design development methodology and the Software Development Life Cycle. As shown below in table 1:

SDLC/UCD stage	Evaluation Type	Evaluation Goal
Stage 1: Specify the needs of the testing and the user	Type 1: Day to day task Type 2: User-task on the system	In a controlled environment Describe/fine a functionality
Stage 2: System component development	Type 3: Basic System task Type 4: System User Task	In a controlled environment describe overall system performance Validate the accuracy and specifics of the system in use
Stage 3: Combine Components	Type 3: Basic System task Type 4: System User Task	In a controlled environment evaluate basic interaction performance Efficiency: How quick does the user learn the interface and functions Validate the accuracy and completeness of the system
Stage 4: Integrate system into setting	Type 3: Basic System Task Type 4: System User Task Type 5: System User Task Environment	Quality assessment in the working environment System effectiveness, accuracy, completeness, utilization, workflow evaluation Users efficiency in process speed and workflow Overall satisfaction from user perception
Stage 5: Routine use	Type 3: Basic System Task Type 4: System User Task Type 5: System User Task	Overall impact assessment of the application in a working environment

Table 1: Usability testing and usability evaluation criteria

2. Context of Use

For the purpose of this usability study the specific program used to evaluate has to be one currently in use by the Australian School system. One such program being used to teach students programming and robotics currently is Choregraphe (Daniela Conti etal. 2019, p. 2 - 34).

Choregraphe is a graphical user interface designed by Aldebaran robotics which enables it's users to create applications containing Dialogs, services and powerful behaviours such as interaction with people, dance and email sending on the Nao and Pepper robots, all without writing a single line of code.

The software platform uses a mixture of block programming paired with a sequential programming paradigm, which also allows its user's to not only block code but also code in various other high level languages such as:

- C++
- Python
- Java
- JavaScript

This type of programming progression allows its users to go from basic block programming to more advanced high-level text based languages such as Python. For this exact reason this is why schools are using this type of software to teach programming both inside and outside the classroom as the software is available on any desktop platform and accessible 24 hours a day.

2.1. Choregraphe's Interface and functionality – Initial Start up

In order to correctly evaluate Choregraphe's interface and functionality it will be evaluated against how well it performs from the initial start-up all the way to the end. It also needs to be noted that we conducted these tests on Choregraphe 2.1.4

When you first initially start up Choregraphe you get presented with the industry standard of being prompted would you like to open a new file or open a pre-existing project you have saved before. This can be seen below in Figure 2:

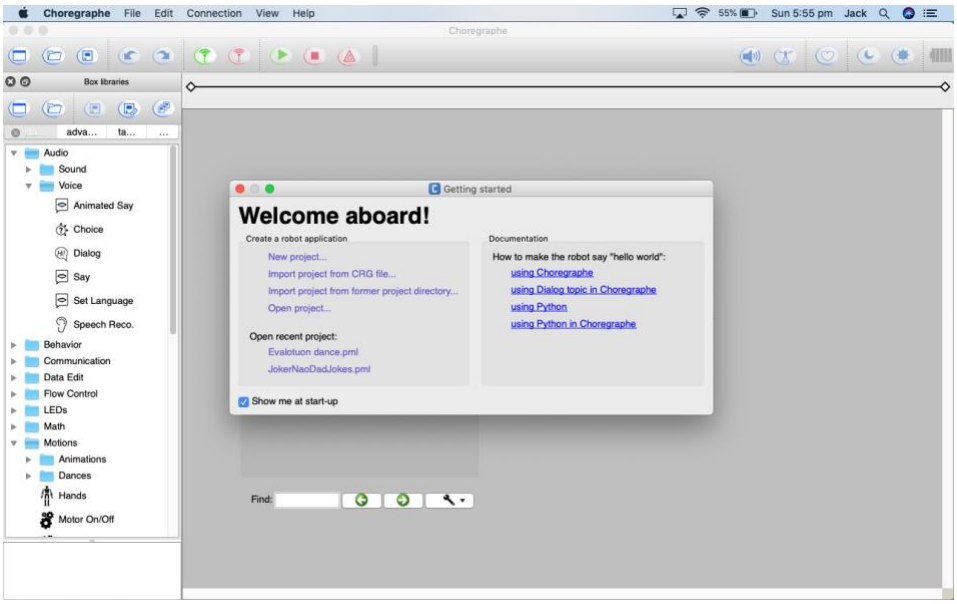

Figure 2: Choregraphe 2.1.4 initial start-up screen

As you can see from the figure above, Choregraphe's initial load screen follows the same rules as many other programs in which it allows its users to have a high level of familiarity for the start. However something unique about Choregraphe's start up screen is the option to skip the getting the start up screen all together.

This small little tick box in the bottom left of the getting started window significantly reduces not only the amount of time but also the amount of clicks required to get started to get through the initial start up screen for individuals who frequently use Choregraphe. Ultimately this small option increase's the functionality of the application for experienced users as well as increase affordance.

However the initial start-up of the program also contains specific functionality for user's who are new to using program as well. From the figure above you are able to see a specific dedicated section to key documentation to help user's learn how to use the program. Nevertheless they stand out even further by having these key documentation links embedded in the download file's, which ultimately allow the user to access these guide's without a internet connection which can be seen below in Figure 3 via the address shown in the search bar:

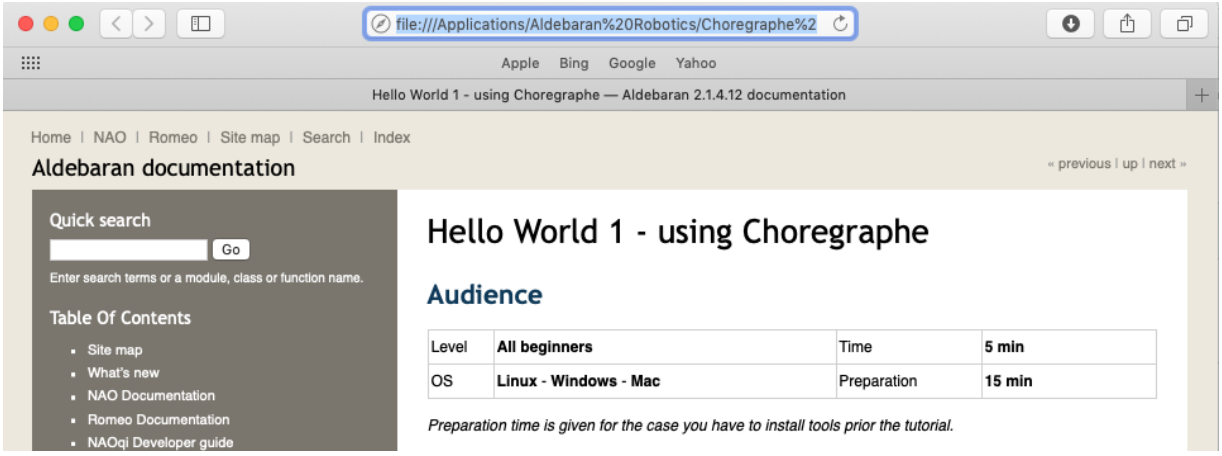

Figure 3: Choregraphe's unique embedded guides for individuals using Choregraphe 2.1.4 without an Internet connection

Ultimately this functionality allows new user's to learn how to use Choregraphe without the need of having an active Internet connection. Which ultimately inturn allows both the student and teacher the ability to teach and use the program regardless of the situation as long as they have the application installed properly.

Furthermore Choregraphe use's a very neutral colour palette paired with accent colours in their initial start-up screen, which highlights to the user the possible options that they can select as well they highlight the most likely options the user might choose by underlining the options. This can be seen especially in the start-up screen under the documentation section by how the links are specifically underlined to indicate that they are a hyperlink.

Continuing on, Choregraphe's start-up screen also allows both new and existing users the option to directly program and have the program file temporary stored in the ram

by clicking the close button. However this particle option is not highlighted anywhere or ever told to the user in any of the documentation or guides Choregraphe provides. So while this might provide the program with significantly more functionality it is never used because the user does not know it is there, therefore we can assume this does not provide any functionality towards the overall use of the software, as many individuals will never be aware of it.

2.2. Choregraphe's Interface and functionality – Main Screen

At first glance at the workspace within Choregraphe we are able to clearly see that it also follows the same neutral colour palette paired with accent colours to highlight and show the user the functionality within the main screen. Furthermore we can see that the screen is broken up into 4 distinctive zones:

1: Box libraries on the left hand side of the screen

2: Quick action buttons on the top left hand corner of the screen

3: Quick action buttons on the top right hand corner of the screen

4: Drop down menu's located at the top left hand corner on of the screen

Within the main user interface the first library that always appears after the initial setup screen is the box library also on the left-hand side of the screen as shown below in figure 4:

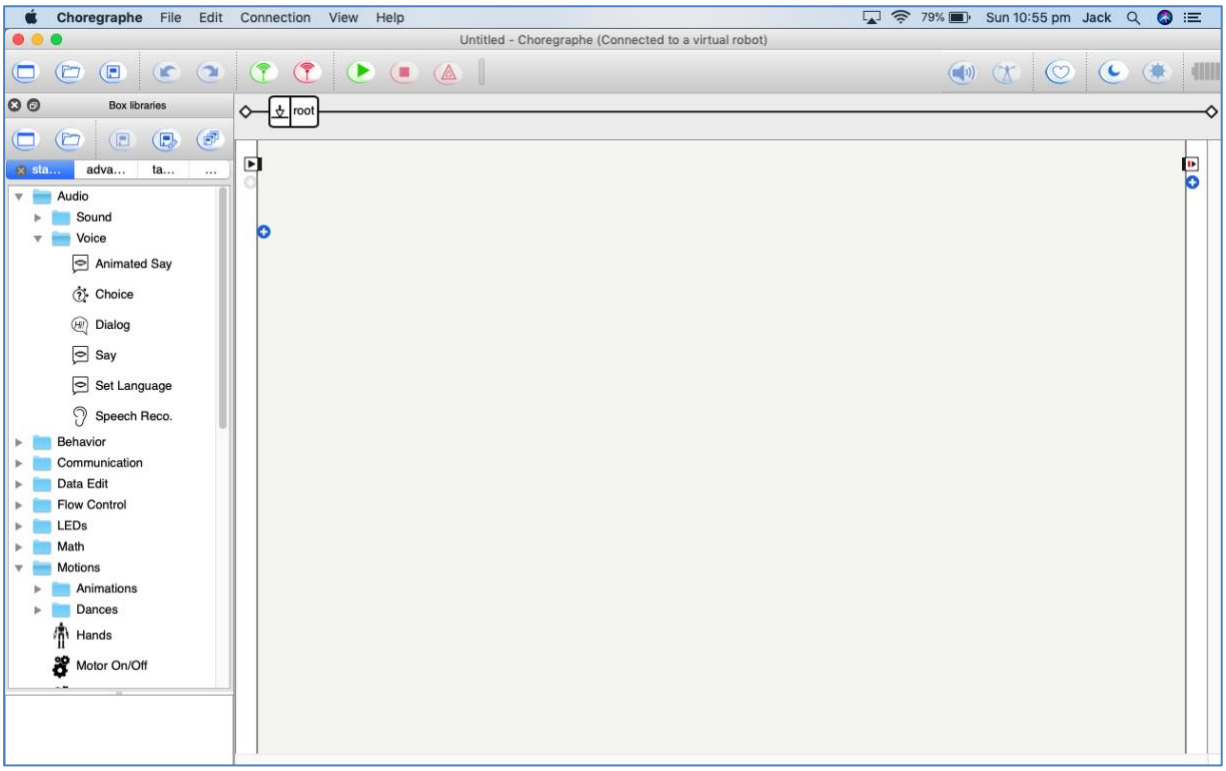

Figure 4: Choregraphe's Main Screen after Initial Start Up

This part of the interface is particularly important as the box libraries provide the user with the most amount of functionality within the program, as the user will use

the box libraries to create the program. Furthermore by only providing one box library upon initial start up it ensures that the interface maintains it simplistic and minimalist design that it is trying to go for.

Never the less Choregraphe also follows very common design trends present in many applications today. The most notable are the save, open, create new project, redo and undo buttons in the top right hand corner of the screen as seen below in Figure 5:

Figure 5: Choregraphe's Functional Quick Action Buttons

As you can see above in Figure 5 that Choregraphe follow's the exact same picture shapes as Microsoft Word. This ultimately allows the user to have a sense of familiarity while using the software, as they know what these buttons do.

Furthermore if they don't know what the buttons do they can alternatively however over them and a small text box will appear describing what that button specifically what that button does. This once again is very similar to how Microsoft Word functions as well. The comparison can be seen below in Figure 6:

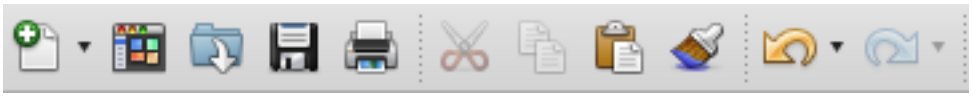

Figure 6: Microsoft Word's action bar

2.3. Choregraphe's Interface and functionality – Adaptable Interface and Functionality

The most notable and unique feature about Choregraphe's Interface and the functionality it holds is the ability to actively change and adapt how it is currently shown. They achieve this by using a scalable interface that the user is able to interact with any time they are actively using the Choregraphe interface. An example of how a user is able to change the interface is shown below in Figure 7:

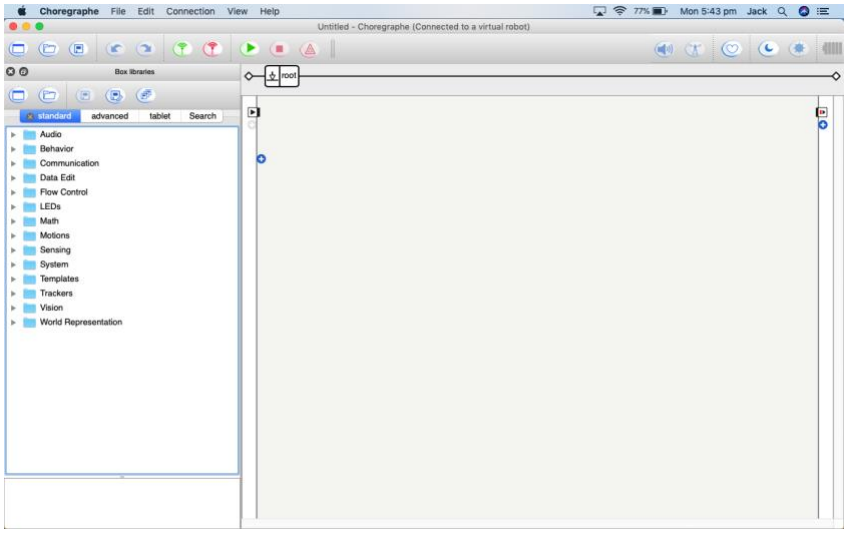

Figure 7: Choregraphe's Adaptable Interface – Changing Size

The comparison of the adaptable interface is explicitly evident when you compare Figure 7 and Figure 4 together. This is particularity shown when you view the amount of the workspace you have compared against the readability of the box library through adapting the interface.

The adaptable interface always the user to make compromises that they want while working within the application. As this is specifically important for all levels of user's as new users will most likely want a higher level of readability and are willing to sacrifice the amount of work space they have, whereas the more experienced user's will want to compromise readability to have more workspace to work in. This small tweak and functionality in the interface ultimately allows users to change the interface to what they currently need and want and essentially allows the user to create a personalised user interface very easily. Ultimately creating a higher level of user experience as the user is able to actively engage with the interface.

2.4. Choregraphe's Interface and functionality – Drop Down Menu's

A key feature within Choregraphe is the ability to add and remove various box libraries, as the user wants. They enable the user to do this by utilization drop down menu bars as by making use of the drop down menu bars they maintain a large enough workspace while still ensuring that the screen does not feel cluttered all together. An example of the drop down menu within Choregraphe can be found below in Figure 8:

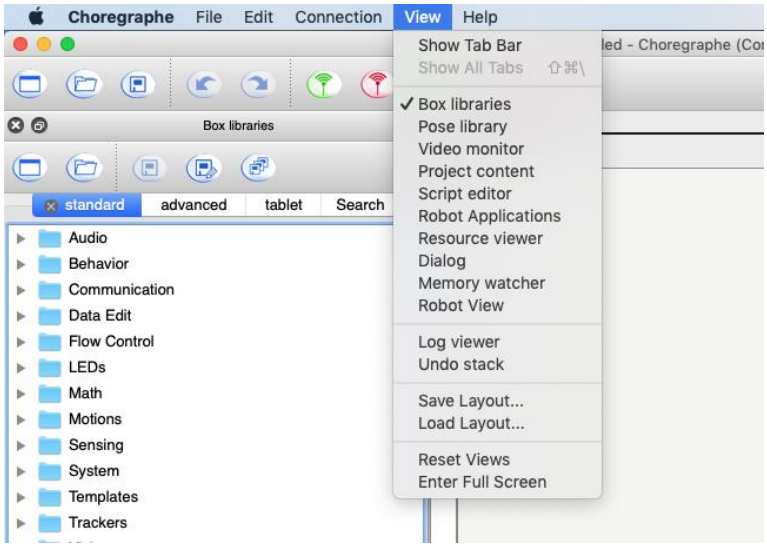

Figure 8: Choregraphe's Drop Down Menu Bars

As you can see Choregraphe expertly use's the drop down bars to not only increase the amount of functionality the program has but also to ensure that the overall user interface still maintains its appealing aesthetic appeal for all individuals regardless of any minor vision impairment such as red green colour blindness or vision loss.

3. Potential Users

Before analysing and creating potential user profiles we first need to know who will use Choregraphe for some type of educational purpose for either themselves or another individual. From various articles and published papers we can clearly see that the following are most likely to use Choregraphe:

- Primary School aged children, 5-12 years (Daniela Conti etal. 2019, p. 2 - 34)
- High School aged children, 13-18 years (Omar Mubin etal. 2013, p. 1 - 5)
- Special Needs children, 5-18 years (Shamsuddin etal. 2012, p. 1533 - 1538)
- Parents of children using the software, 18-80 years (Omar Mubin etal. 2013, p. 1-5)
- Teacher's at both a primary and high school level, 18-80 years (Omar Mubin etal. 2013, p. 1-5)
- University students, 17-80 years (Hadi Jahanshahi etal. 2019, p. 1 - 24)
- Researchers, 18-80 years (Hadi Jahanshahi etal. 2019, p. 1 - 24)

3.1. Participants

After reviewing 50 potential users a select group of 5 individuals were selected to conduct the usability study on as in accordance to the NNG groups research on user testing (Nielsen J, 2000). The selected users came from a diverse range of individuals based on there:

- Education Level
- Current Occupation
- Age
- Digital Literacy level
- English fluency based on international English language testing system
- Their accessibility level based on the Web Content Accessibility Guidelines

The participant data can be found below within table 2:

ID	Education Level	Occupation	Age	Digital Literacy	English Fluency	Accessibility Level
P1	Primary School	Student	13	Low	Medium	High
P2	High School	Student	17	Medium	High	High
P3	University	Student	21	High	Medium	High
P4	University	Teacher	26	High	High	High
P5	PhD	Researcher	43	Very High	High	High

Table 2: Participant data from the usability study

Initially before any usability testing was conducted on the participants they were all asked what they look for when selecting or choosing educational programming learning software to learn from. This particular exercise took place in order to gain a better oversight of what each of the participants are looking for. The list below hereby contains the most frequent answers the participants responded:

- Flexible design in which I am able to manipulate to meet my learning needs and the ability to learn in multiple programming languages (P5)
- Training videos which specifically show how to best use the learning exercise being explained (P4)
- Help form where I am able to ask questions about specific exercises outside of the classroom (P3)
- Actively engaging platform which keeps me entertained (P2)
- Easy to read and use software which does not confuse me (P1)

Based on the various interviews, questionnaires, observations, specific needs of the participants and application specifications we are able to better gauge the overall prospective image of an every day user using the Choregraphe software. Therefore we are now able to create a user persona. We create a user persona to better determine how the target group of the application might respond to different situations that occur in the application. The representation of the user persona follows a qualitative and quantitative form of research (Johansson & Messeter, 2005) and can be found below in Table 3:

User Persona
Name: Rob  **Job Title:** Teacher **Age:** 25 **Geographical Location:** Melbourne, Berwick **Education:** University (Bachelors of Primary Education) **Family:** Wife and a son **First Language:** English **Digital Literacy Level:** High **English Fluency:** High **Hours spent a week studying:** 25 **Main environment while studying:** Home Office **Accessibility Level:** Very High **Nature of learning programming:** To teach it to his students who are currently learning it **Years of programming experience:** 0.5 **Programming Level:** Intermediate

Programming Languages Learnt: Python, C++

Confidence in block programming: Very High

Confidence in text based programming: Medium

Primary Motivation to learn programming: To be able to better teach his students programming

Primary Goal: To not only increase his programming skills but also his students

Secondary Goal: To ensure that class runs smoothly while running it

Primary challenge: Learning new concepts in a short period of time such as control flow statements

Secondary challenge: Actively teaching his students a new concept he just learnt

Overall IT skill level: Medium

Attitude towards educational programming software: Very positive as it makes his teaching more effective and efficient as a result of it

Fears: Not being able to learn a concept in time to be able to teach his students it

Hobbies: AFL, Woodworking, researching into behavioural adaptation

Most frequently used operating systems: Mac OS and Windows 10

Most frequently used devices – IPhone, MacBook pro 2012 and a old desktop running Windows 10

How will the person use the software:

- Learn from the software themselves the teaching concepts covered within the software
- Teach other individuals how to use the software and as well as the teaching concepts they had covered themselves
- Create activity plans around the software and timelines on how quickly the students should advance through the learning concepts related to the software

Key tasks to complete while using education programming software:

- Effectively teach themselves how to program
- Effectively teach others of all levels of skill levels how to program
- Create studying learning plans around the software for himself and the students to follow

Three most important components of an education programming software:

- User centred design which allows personalised learning geared towards the user
- Clear and legible interface that has a high level of readability for all levels of user regardless of their level of technical skills or digital literacy
- Customisable workspace which allows the user to create their own functions and higher level components which add extra applicable functionality's

Table 3: Sample User Persona For Educational Programming Software

3.2. Requirements

From the user persona and the five answers the users provided we are now able to create a bare minimum requirements list. This list is as follows:

- Aesthetically appealing user interaction
- Flexible and customisable user interface
- Easy to learn programming system usable by individuals of all ages and skill levels and adaptable towards their needs and wants
- A way of receiving help when stuck on a certain solution for example a discussion form
- Highly readable solution which allows individuals of limited language skills the ability to use the program
- Embedded training videos
- Responsive Interface which reacts to the users actions, for example if the user adds a new view, the screen changes based upon that view
- Way to track an individuals progress and analyse that data to provide constructive feedback on where

3.3. Evaluation Criteria Questions And Tasks

Therefore based upon the requirements, user persona, participant data and participant questionnaire answer results we are now able to create a set list of questions and tasks to most efficiently gauge the effectiveness of a software geared towards this specific genre of technology for the user.

The Tasks and questions will be related to and evaluated against the SDLC/UCD methodology listed earlier within this report to ensure we actively engage and focus on a user centric design model. The questionnaire can be found below within Table 4:

SDLC/UCD Stage	**Evaluation Type**	**Question/Task**
Stage 1: Specifying the needs of the user	**Type 1:** Day To Day Test	1: At first glance of Choregraphe would be happy looking at it for an extended period of time (). If not why?
	Type 2: User-Task On The System	2: Open up Choregraphe and create a new file and identify where the box library is, time period of 60 seconds
Stage 2: System Component Development	**Type 3:** Basic System task	3: Identify where the play button is within Choregraphe (60 seconds)
	Type 4: System User Task	4: Close the box libraries tab and then re-open it (60 seconds)

Stage 3: Combine Components	**Type 3:** Basic System Task	5: Save your program, close the program and reopen the program (60 seconds)
	Type 4: System User Task	6: Identify how you open a help guide within Choregraphe (60 seconds)
Stage 4: Integrate system into setting	**Type 3:** Basic System Task	7: Save the program as a CRG file (60 seconds)
	Type 4: System User Task	8: On a scale of 1-5 how easy is Choregraphe to use in reference of finding specific
	Type 5: System User Task Environment	9: Demonstrate to me and explain to me how you would re open a box library (graded on a 1-5 scale)
Stage 5: Routine Use	**Type 3:** Basic System Task	10: Would you change how the software looks, if yes explain why
	Type 4: System User Task	11: Name one thing about the Choregraphe software that you would change
	Type 5: System User Task	12: Overall how would you rate the ease of use of the software? What specifically would you change to increase the ease of use?

Table 4: User Interaction Questions based off of the SDLC/UCD Methodology

Furthermore in conjunction with evaluation criteria we will also be evaluating the software application Choregraphe 2.14.3 against the most prevalent current design trends and guidelines found within a usability report by Jack Murley on Education programming software aimed at school aged individuals (Murley 2018, p. 5 - 11).

The most prominent design guidelines that will be used to assist us evaluate the program are:

- User Interaction
- Reactive Screen Size
- Minimalist Design
- Error Handling

4. Analysis of Choregraphe's Interface

After all of the ten participants had completed all twelve questions their answers were recorded into a spreadsheet. Once the spreadsheet held all 120 answers both manual and machine learning data analysis was conducted on all 120 answers.

Furthermore both quantitative and qualitative analysis was conducted on the:

- Survey results
- User Interaction Questions
- Participant data
- Any other questions asked

This was to ensure that all data collected for data analysis and review was of a high enough quality and held a specific level of validation to maintain a high level of data integrity.

4.1. Aesthetics

Out of the 5 participants most of them reported a relatively high level of aesthetical appeal (4.5/5) towards the overall interface at first glance with only the box library on the screen. However as more and more views and libraries were being added the participant's level of aesthetical appeal. This decrease in the participant's aesthetical appeal can be calculated by the following formula where A is the aesthetical appeal level and V and L are the views and libraries:

$A = 6 - L - V$

The main point the majority of the participants (3) had about the design interface was that after 7 views were added the screen became to cluttered and the workspace became to small and as a result ultimately being unusable.

However two of the participants who had university degrees preferred to have this, as they rather liked how they were able to make this compromise, as many other programs don't allow this. Ultimately this resulted in a mixed structure score, which can be evaluated by an individual's level of education, which can be seen in the graphs below:

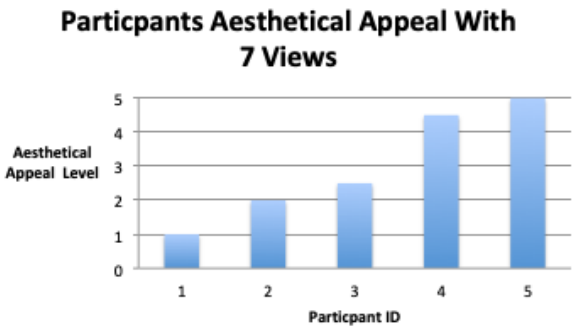

Figure 9: Participants Aesthetical Appeal With 7 Views

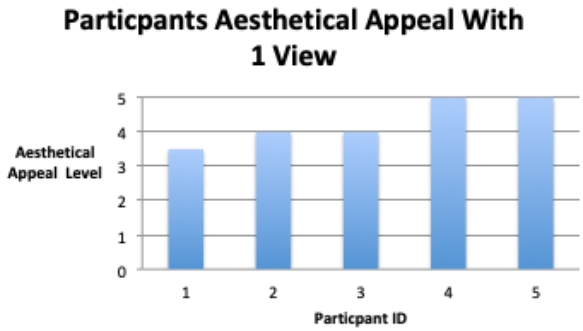

Figure 10: Participants Aesthetical Appeal With A Single View

4.2. Responsive Screen Design

Overall the majority of the participants (4) enjoyed and strongly commented on how they enjoyed using the interface and how fluent the transition was between different views, which resulted in of an average score of 4.8 out of 5 when you exclude the outlier. Furthermore they went on to comment and liken how it followed current design trends and knew when to downsize or upsize a specific view. As well as they commented on how it maintained a type of simplistic view while still being very complex at the same time.

However one individual, participant 1 gave the responsive screen design a score of 0.5 that ultimately dragged down the overall average score, which resulted in a 78.4% result of enjoyment of the responsive screen design feature overall for the group.

Participant 1's reasoning behind it was that he could not see the point behind have the screen change so often as it kept confusing him where certain view's were being held, this was seen in person during testing as the participant became agitated and confused when they were unable to keep track of certain views. The screen view can be seen below within Figure 11:

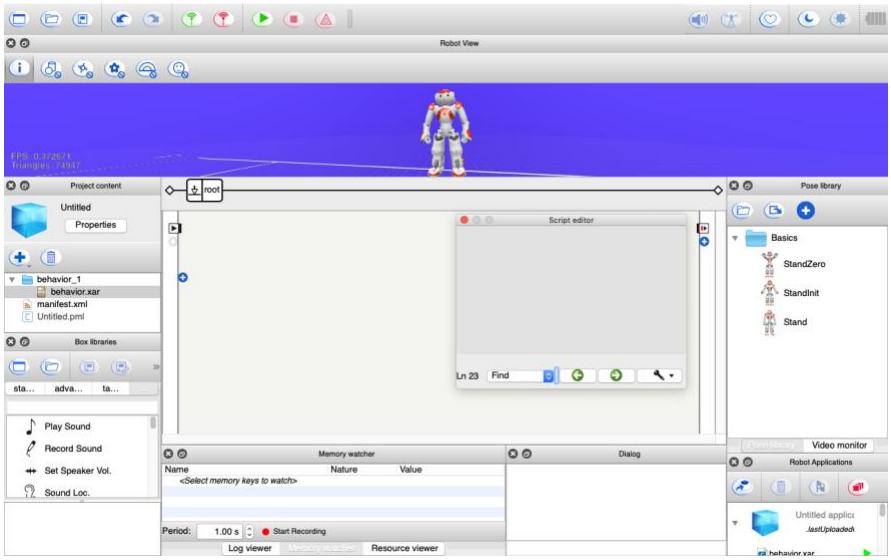

Figure 11: Screen View Of Participant 1's Screen Which Caused Frustration And Confusion

Therefore further tests were conducted and through the quantitative data analysis through a density based machine-learning algorithm to detect outliers it was found that this individual had a high likelihood of being an outlier. For that exact reason we will exclude this individual when calculating the overall User Experience however not when evaluating the functionality for all users.

However it needs to be noted that Participant 1's age was the youngest individual in this usability as well as had the lowest level of digital literacy. Therefore a possible usability issue could commonly occur for individuals who have a low level of digital, which could lead to the assumption that the lower a individuals digital literacy level the higher the likelihood of multiple stimulus's agitating or confusing that individual.

Nevertheless a larger group of individuals with a low level of digital literacy would be needed to test this assumption and verify it however in the context of this study it is reasonable assumption.

Ultimately from this we can conclude that further testing was required specifically within the Specifying the needs of the user stage of the SDLC/UCD methodology used to create this application.

4.3. Customisable Screen Design

On average the participants rated the usefulness of Choregraphe's customisable screen design based on the user's need to be very useful. This can be seen how on average the 5 participants had an average score of 4.3/5. The only negative overall comment that was made by the participants was that at times it was hard to resize the screen due to other views, which managed to collide with the scaling property. This can be seen down below in Figure 12:

Figure 12: Collision Of The Scaling Screens

Furthermore the participants also picked up on how some text when you scaled it to a specific size became clustered and very hard to read. Ultimately resulting in a lower readability when this happen. This can clearly be seen within Figure 12 in the bottom left hand corner of the screen.

4.4. User Experience

Overall the User Experience of the interface as a whole was considered a success. This is because the final overall mark for user Experience was 4.2/5. This was calculated based upon:

- Aesthetics overall score
- Responsive Screen Design overall score
- Customisable Screen Design overall score
- A separate overall interface design score

Furthermore it can be concluded that while the overall user experience for the usability of the interface might be high it does not necessary mean that it is free of usability issue's. In this study this was found out while testing the Responsive screen design, which identified that Choregraphe as a program lacked specific testing on a younger age group during the first stage of the UCD/SDLC methodology when they had to specify the needs of the user.

4.5. Overall Usability of Choregraphe's Interface

Ultimately through the testing conducted the overall usability was at a relatively high standard as the participants had the following comments to say about it after this section of testing:

Participant 1 – I would be happy to learn from this interface as long as it allows me to fully change and manipulate the views and doesn't restrict me on what size I can make them

Participant 2 – I would gladly learn from this interface as long as more extensive guides on how to use the different views were provided

Participant 3 – If the interface allowed me to fully edit and change certain aspects such as colour I would want to use the interface for all my studying needs

Participant 4 – I wish all over interface's interacted and transitioned as smoothly as this one did

Participant 5 – The interface did everything I needed and wanted and even more. The advanced customisation within the interface ensures that the interface can be tailored to almost anyone's needs which is rare to see in a programming interface

After extensively reviewing the participants comments and results from the subsequent usability testing on the interface it can be concluded that Choregraphe has a reasonable interface with a high level of usability for individuals of all levels of digital literacy and education level

However it needs to be noted that the lower an individual's level of digital literacy the less likely the high level of usability will be able to be used and also it increases the likelihood that the high level of usability might frustrate or confuse the individual.

Therefore we can conclude that while Choregraphe might have a higher level of usability for individuals who have a higher level of digital literacy, Choregraphe's high level of literacy can have the reverse effect on individuals of a lower digital literacy. Thus this can ultimately be decided and found out through the SDLC/UCD methodology used within this usability report that further testing during the first, second and last stage as these are crucial towards finding out information about all types of user's regardless of any factors that might impact their learning.

5. Functionality of Choregraphe's interface

In order to effectively evaluate the functionality of Choregraphe's interface we first need to create a usability metric to evaluate the functionality. The metric that will be used to evaluate the functionality of Choregraphe's interface is as follows:

Metric	**Sub-Characteristic's of that metric**
Learnability	Ease of use, Effectiveness of guides, Readability
Operability	Self Explanatory Error Messages, Error Correction, Input undo-ability, Error undo-ability, Customizability of functionality
Attractiveness	Interface Appearance, Interface Customizability, Customizability of Colour (Colour Blindness mode)

Table 5: Usability Metric to measure the functionality of Choregraphe's Interface

5.1. Overall Usability

On average the majority of the participants reported that while the guides provided detailed information on how to get started with Choregraphe they did not provide them with enough to use other box libraries. As a result, on average the user's reported a 3.5/5 result for the effectiveness of guides. Furthermore when compared against the results for the Attractiveness as discussed above we can clearly see that their was a .7 change in effectiveness.

However nevertheless the participants reported a relatively high result for operability when compared to the learnability. This is because on average the participants reported a 4.5/5. However I believe that the participants mistook operability to be the same as attractiveness. As this is especially seen within the comments found below in participants comments.

The participants comments related to the metric can be found below within table 6:

Metric	**Participants Comments**
Learnability	"The guides provided gave little in-depth information" P1, P2, P4 "The guides could have been improved through implementing video guides instead" P3 "The interface is relatively easy to use however the guides provided little to no information" P5
Operability	"I really enjoyed and liked how you could customise the interface" P1, P3

	"I enjoyed how the different views interacted with each other and how fluent they transformed" P2, P4 "The transfusion between the interfaces and the customizability within them ultimately created the ideal working environment especially when you compare it to other programs" P5
Attractiveness	"Overall the interface as a whole was just ideal and enjoying to look at" P1, P2, P4 "The interface was what you would expect in a programming piece of software aimed at all ages" P3 "I wish all graphical user interfaces for programs we as aesthetically pleasing and useful as this one" P5

Table 6: Participants Comments Based Upon The Functionality Metric

5.2. Affordance

Affordance within this usability was estimated based upon the amount of clicks and the total time it took the users to complete a specific task. This was found to be on average 9 clicks within a 45 second time period.

This is particularly important when we compare it to a developer doing the same task when on average it took them 3 clicks and 10 seconds to complete a task. Therefore we can assume that the more experience an individual has within the program the higher the likelihood that their level of ease of use within the program will also increase.

The distribution of time taken and number of clicks per task are seen below within Figure 13 and Figure 14:

Figure 13: Time Taken for both participant and developer to finish tasks

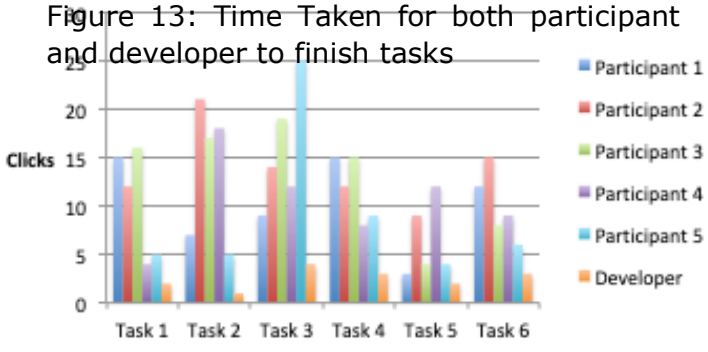

Figure 14: Amount of clicks needed for both participant and developer to finish tasks

It needs to be noted that the participants were stoped if they reached the 60-second time limit. Nevertheless we are now able to estimate the affordance based upon the amount of clicks and time taken from the users to complete their tasks.

Overall we can now assume from the results from the tests that the affordance for the application Choregraphe is relatively poor when compared to the developer results. Nevertheless while we assume that the developer will have better results than the participants the gap should not of been as large as it is currently.

6. Conclusion

Ultimately from the tests conducted within this usability report we can assume that an ideal user interface for an educational based programming software should consist of:

- Easy to use interface which contains minimal jargon
- Ability to program in multiple languages
- Ability to adapt and remove functionality's If the user wants to
- Advanced and well detailed documentation which explains not how to use the program but also includes instructional videos
- Minimal/greatly reduced navigation and click's based on specific tasks
- Logically structured and aesthetically pleasing layout
- The ability to change the interface/adapt the interface based on the users needs and wants

The need for specific and well-refined educational programming software is evident in not only the affordance but also the functionality testing on the interface shown within this usability report. This especially important In todays day in age as many educational programming software now heavily focus's around the user and what they need to achieve their learning goals.

Nevertheless the need for a refined system can be evident in the results by participant one who had the lowest level of digital literacy and as a result ultimately had the hardest time completing the tasks within the time limit.

Therefore the findings of this study can be solved and evaluated against the SDLC/UCD methodology as listed earlier to find where the specific gaps in the lifecycle lead to participant 1's confusion within Choregraphe and where the

developers of Choregraphe miss calculated the ability of individuals with a lower level of digital literacy.

Furthermore this study could also serve as a framework for future developers who are wishing to develop a educational based programming software.

7. Reference's

Davis, F., Bagozzi, R. and Warshaw, P. (1989), User Acceptance of Computer Technology: A Comparison of Two Theoretical Models. *Management Science*, [online] 35(8), pp.982-1003. retrieved 8 Apr. 2019.

Diloon, A. (2001). *User Acceptance of information technology*. 1st ed. London: London: Taylor and Francis, pp.12 – 14, retrieved 9 Apr. 2019.

Holden, H. and Rada, R. (2011). Understanding the influence of perceived usability and technology self-efficacy on Teachers' Technology Acceptance. *Journel of Research on Technology in Education*, 43(4), pp.343 – 350, retrieved 10 Apr. 2019.

Jahanshahi, H., Jafarzadeh, M., Sari, N., Pham, V., Huynh, V. and Nguyen, X. (2019). Robot Motion Pallning In An Unknown Environment With Danger Space. *MDPI*, pp.1 – 24, retrieved 7 Apr. 2019.

Mubin, O., Stevens, C., Shahid, S., Mahmud, A. and Dong, J. (2013). A Review Of The Applicability Of Robots In Education. *Technology For Education And Learning*, pp.1-5, retrieved 2 Apr. 2019.

Nielson, J. (2010). *Why You Only Need to Test with 5 Users*. [online] Nielsen Norman Group. Available at: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users, retrieved 3 Apr. 2019.

Nielson, J. (2012). *Usability 101: Introduction to Usability*. [online] Nielsen Norman Group. Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability, retrieved 4 Apr. 2019.

"Robot, tell me a tale!": A Social Robot as tool for Teachers in Kindergarten. (2019). *InteractionStudies*, 5(28), pp.2 – 34, retrieved 5 Apr. 2019.

Shamsuddin, S., Yussof, H., Ismail, L., Mohamed, S., Hanapiah, F. and Zahari, N. (2012). Humanoid Robot NAO Interacting with Autistic Children of Moderately Impaired Intelligence to Augment Communication Skills. *Engineering Procedia*, 41(41), pp.1533 – 1538, retrieved 1 Apr. 2019.