The Interdisciplinary Center, Herzliya

Efi Arazi School of Computer Science

M.Sc. program - Research Track

Music Technology Education and a Plugin-Based Platform as a Tool to Enhance Creativity, Multidisciplinarity, Creative Design, and Collaboration Skills.

by

Or Perets

M.Sc. dissertation, submitted in partial fulfillment of the requirements
for the M.Sc. degree, research track, School of Computer Science.
The Interdisciplinary Center, Herzliya.
December 2020

This work was carried out under the supervision of Dr. **Revital Hollander** from the Adelson School of Entrepreneurship, The Interdisciplinary Center, Herzliya, and **Prof. David Harel** from the department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot.

Acknowledgements

Throughout the writing of this master's work, I received a great deal of support and assistance. I would first like to thank my supervisors, Dr. Revital Hollander and Prof. David Harel. Dr. Hollander was invaluable in formulating the research questions and methodology, and providing insightful feedback, and prof. Harel, helped me to sharpen my thinking and bring my work to a higher level. You provided me with the tools that I needed to choose the right direction and successfully complete my work.

In addition, I would like to thank my parents, for their support in my career and the academic studies. You are always there for me. Also, to my best friend, Tal Guetta, who set me off on the road to this MSc a long time ago.

Finally, I could not have completed this work without the support of my partner in life, Omer Doron, who provided stimulating discussions as well as happy distractions to rest my mind outside of my research.

Abstract

"Creativity is essential to computer science students, and computer science makes it easy to be creative" (Romeike, 2007,87). Creativity, creative design capability, multidisciplinarity, collaborative ability, and artistry can improve computer scientists' and software engineers' abilities in problem-solving, innovation, software design, and development. It has also been recognized that music technology can be used effectively to enhance creative development. It is engaging, and "it can help develop creative thought in an academic environment and allow students to gain self-efficacy in their creative abilities" (Rosen, Schmidt & Kim, 2013, 344). When developing music technology projects, students can easily combine art, science, and technology. Whether music technology is used in theoretical research or for an applicative project, it naturally involves a merge between artistic and computational paradigms and a combination of several disciplines; music, art, sound, neuroscience, psychology, sports, education, gaming, and more. When students are creating and collaborating, music technology education helps them express their personalities, their passion for music, and other positive emotions (Brown & Theorell, 2006). The combination of academic studies, positive emotion, and enthusiasm is an integral part of optimal engagement, increasing creativity and innovation.

In this work, we have developed a creative education method based on music technology development that uses the Muzilator platform as a creative educational tool. Muzilator is a plugin-based web platform that enables developers to divide their project into a set of independent plugins that can be implemented, debugged, uploaded, and shared with the platform's community.

This study seeks to identify which project features and team combinations can optimize the students' learning outcomes and help students develop their creativity, innovation, artistry, design capabilities, and collaboration skills.

The research is based on 75 projects implemented by 183 computer science students who participated in the Computer Music course taught by Dr. Revital Hollander-Shabtai at the Interdisciplinary Center, Herzliya, between 2016 and 2020. The students developed ideas and prototypes (POCs) for innovative research or applicative music-tech projects. They worked in teams, using an Agile methodology, and developed the projects in three phases. For the purposes of the research, the projects were divided into five main categories and the projects' risk level, creativity, multidisciplinarity, interaction, artistry, and creative design were evaluated. The difference between theoretical research projects and applicative projects was examined and the students' self-evaluations, as well as a subjective report on the final project, were analyzed.

The analysis results show that high-risk projects were more creative and artistic than were low-risk projects. Students who considered themselves self-learners combined more disciplines in their projects than did others. Mixed-gender teams (men and women) developed the most artistic, and the most multidisciplinary projects, while other team combinations were less effective. Solo teams with only one member had the lowest rankings in all parameters and learning outcomes. Women tended to choose to develop interactive applications, while men tended to choose more theoretic (algorithmic), non-interactive research projects. Finally, teams that used the Muzilator platform developed projects that were more creative, multidisciplinary, and artistic, and which were ranked higher in creative design than were projects that were developed without use of the platform.

During the course of writing this thesis, some of its conclusions and work processes were presented at the following conferences:

1. The 8th Kinneret Conference for Software Engineering Education, February 2020, Israel.
2. The 4th MIC (Marconi Institute for Creativity) Conference – Nurturing Creative Potential (ISSCI), September 2020, Italy.

Table of Contents

List of Figures

List of Tables

1 Introduction

Traditional computer science education, both academic and non-academic, combines mathematical knowledge, theoretical computer science, computational thinking, computer programming, and software engineering. While all these skills are necessary for algorithm design and implementation, additional skills and techniques are essential for enabling computer scientists and software engineers to solve complex problems and to innovate:

1. *Creativity* involves the use of original ideas to create something new and effective (Runco & Jaeger, 2012). Creative thinking and creative design in software engineering are fundamental for improvising and devising solutions for controlling complex systems.
2. *Multidisciplinary* refers to the ability to draw from different disciplines for research and problem-solving.
3. *Collaborative ability* is an essential skill for computer scientists, without which there can be no communication or synchronization between individuals and teams. Collaborative ability contributes to code sharing, upgrades the quality of the products, accelerates coding and integration processes, and improves software design capability, testing, and quality assurance (QA).

The importance of a software product's design and development is "dependent on the team members' openness, analyzing a system design, and coding the various components" (Nelson, Brummel, Grove, Jorgenson, Sen & Gamble, 2010, 206).

4. *Software Design Capability* involves the use of software designs during development, which is essential for the future maintenance of the project or the product. A developer needs to have both a deep understanding of the global scope of any project and the ability to develop independent components which can still relate and interact with other elements of the system.

Music technology is a field that can offer an excellent tool for creative development (Rosen, Schmidt & Kim, 2013). A high level of engagement has been shown among students who studied and developed musical projects, and among students who were intellectually involved in the process of meaningful exploration (Newmann, Wehlage & Lanborn, 1992). When creating and collaborating, music technology becomes a tool for expressing positive emotions during the learning process. The combination of academic studies and positive emotion is an integral factor for optimal engagement (Khairuddin & Hashim, 2008). Music technology is a multidisciplinary domain that naturally merges the artistic and computational spheres.

When students develop a music technology project, they use their software design skills to build and combine different artistic or computational components, such as an interface to trace interactions, a synthesizer, an algorithm, and more.

This study investigated the characteristics of music technology projects and the key factors needed to improve computer science students' skills, such as creativity, artistry, multidisciplinarity, creative design capability, and some aspects of software design and collaboration skills.

The research is based on 75 projects developed by 183 computer science students (third-year undergraduates or masters students) who participated in the Computer Music course at the Interdisciplinary Center, Herzliya, between 2016 and 2020. In this course, students were provided with an introductory background on music technology and learned how to use music-tech tools to develop an innovative idea. Their projects could either involve theoretical research, such as an analysis or a generation algorithm, or be an applicative project, such as an intelligent interface or a proof of concept (POC) for a new application. The students worked in teams of one to four members, using an Agile methodology. The projects were divided into five main categories and were evaluated for several criteria: creativity, artistry, interactivity, multidisciplinarity, and risk.

The projects were evaulated in terms of team size, gender combinations (single or mixed gender groups), team members' skills and background in software development and music, and the students' self-evaluations assessing themselves as creative, multidisciplinary, and self-learners, or autodidacts.

In 2020, the last year of this research, we launched and tested Muzilator, a plugin-based web platform for sharing and collaboration. The platform is an innovative educational and collaboration tool and environment for all developers, projects, and teams. The efficacy of working with the platform, its abilities, and how it can enhance the students' creativity, multidisciplinarity, creative design, software design capability, and collaboration were examined.

This thesis is organized as follows. Section 2 contains background and related work. Section 3 provides a categorization of the Computer Music course projects based on the characteristics according to individuals, teams, and projects. In Section 4, the Muzilator experiment from 2020 is described. Section 5 presents several computational analysis methods. Finally, Section 6 contains the main conclusions and suggestions for future directions.

2 Background and Related Work

This section presents a review of related work divided into sections according to the subjects relevant to this study: creativity, creative education, project-based learning, and music technology education.

Creativity

There are two primary perspectives about creativity in computer science (CS), one is concerned with creativity and the person, and the second with creativity in the software development process (Romeike, 2007). When focusing specifically on motivation among students (Bergin & Reilly, 2005; Junius, 2015), those concerned with creating and the person in CS claim that highly motivated students exhibit greater creativity performance than others. Romeike describes multiple factors that can increase motivation, particularly the anticipation of being able to use the software in the future, and participation in an open-source community that can facilitate tracing, provide access to reports of other developers, integrate students into teams according to their goals, and enable students to expand and improve their programming skills by exposing them to different concepts. Those focusing on creativity in the software development process stress the importance of a multidisciplinary viewpoint and creative processes in software design.

When examining a multidisciplinary process over different domains (i.e., art, creativity, and engineering), these different disciplines may share common attributes, leading to similar processes (Charyton and Snelbecker, 2007). Charyton and Snelbecker conducted a study designed to understand the differences or similarities between these domains among music students and engineering students.

Several psychology methodologies have been used to estimate the differences between groups, such as the creative temperature scale (Gough, 1979), the cognitive risk tolerance survey (Snelbecker, McConologue & Feldman, 2001), the harmonic improvisation readiness record (Gordon, 2000; Kiehn, 2003) and the creativity test (Lawshe & Harris, 1960). The results from these analyses have indicated that engineers and musicians are approximately equal in terms of artistic creativity.

The enhancement of creative development among undergraduate computer science students can be described using conceptual frameworks (Ferreira, 2013). Ferreira presented a conceptual framework for students which focuses on programming, iteration and human-computer interaction (HCI). Ferreira's framework consisted of seven factors: immersion (solution adaptation to a relative problem); dependencies' recognition; exploration of complementary paths (elaboration and sharing); overcoming obstacles and limitations (generalization and high-level scenarios); expansion or combination of ideas;

discovery of unpredictable places (transforming ideas into novel solutions); and development. According to Ferreira's results, this framework enables the students to enhance their creative thinking, strategies, and programming skills.

In recent years, increased attention is given to estimating creativity using the domains of science and art. Agnoli, Corazza & Runco (2016) defined this challenge as multidimensional because it can be tested in several aspects: convergence, divergence, psychology, and more. They presented a test battery to assess creativity and to measure ideation and evaluation.

The test includes six steps: a Remote Associates Test (RAT) to determine associations between cue words; a title task involving suggesting alternative titles for classic books or movies; a figure task (Wallach & Kogan, 1965), in which participants were asked to provide three different explanations of three abstract drawings; an exploration of practical rather than abstract problems; a Creative Achievement Questionnaire (Carson, Peterson, & Higgins, 2005) to measure creative accomplishments in ten different domains; and a Creative Activity and Accomplishment Checklist (CAAC), where participants ranked creativity achievements in several domains. Other psychological tests were also incorporated, such as the Big Five Personality Test and Raven's Advanced Progressive Matrices (APM).

The participants (over 300 students from the University of Bologna) took several tests to strengthen the validity of the battery test. The researchers

found that diverse thinking abilities were positively associated with personality traits and with creative artistic achievement. They also noted that high levels of problem-solving abilities were essential indicators of creative achievement.

Nilsson (2011) suggested a methodology to measure innovation and creative design: the taxonomy of creative design (Figure 1). He presented five hierarchical levels of creative design: imitation, involving the question of, “is the creation the same as something that already exists;” (Nilsson, 58) variation, or whether it is “a slight change to an existing object;” (Nilsson, 59) combination, involving whether it is “a mixture of two or more things such that it can be said to be both;” (Ibid.) transformation, or whether it is “a re-creation of something in a new context;” (Ibid.) and original Creation, asking whether it appears “to have no discernible qualities of pre-existing objects.” (Nilsson, 60)

Novelty, according to Nilsson, is the taxonomy level of being novel, new, or unique, and scaled by the taxonomy. It can be measured according to the two-dimensional parameters of Novelty in Form and Novelty in Content. This taxonomy can be applied to creativity in non-related fields by scale adaptation: for example, by measuring creativity in education to determine novelty among students (Junius, 2015).

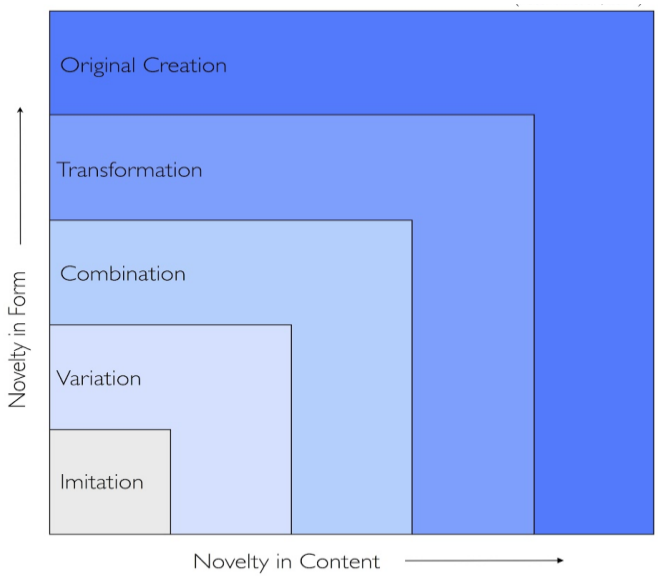


Figure 1: *The Taxonomy of Creative Design (Nilsson, 2011)*

Creative Education

Creative educational methods are relevant and necessary for encouraging creativity among students. After gathering observations from interviews conducted at institutions engaged in creative educational thinking, Rauth, Köppen, Jobst & Meinel (2010) presented an educational method to enhance the creative confidence level. Their interviews contained various questions regarding creative education design, based on creative education design (Lande & Leifer, 2010). At the beginning of the experiment, they informed the participants about the process and the creative assignments and challenges to ensure that the participants fully understood the questions.

According to their research findings, participants initiated creative challenges independently, without any relative background with respect to predefined creative challenges. The participants were aware of the uncertainty (risk) level of a given creative assignment.

Nelson, Brummel, Grove, Jorgenson, Sen & Gamble (2010) proposed the SEREBRO (Software Engineering REwards for BRainstorming Online) system for modeling creativity within a computer program. The system provided measurement opportunities to develop metrics around originality, elaboration, and overall creativity. Students worked in teams ("even when a single member is more creative or has an advanced skill set, the success of the project requires the contribution of all members, especially within a small team,", [.207]) and the projects were rated for fluency, flexibility, originality, elaboration, and overall creativity. The SEREBRO platform assigned reward points to each individual or team for their creative input. For example, the platform methodology rated usages with maximum K points (where K is a natural number). Each team's total score by usage, reuse, and sharing was measured by awarding the developer of a process K points for each usage of that process and awarding whomever was involved in the process 0.5*K points for each usage, thereby leading to precise results. Creativity ranged from 3.18 to 4.84, and in general, teams with higher quality ratings received high creativity ratings.

The primary purpose of the Nelson et al. study was creativity assessment and enhancement of creativity while developing a system.

Project-Based Learning

Project-based learning (PBL) involves solving a given problem in educational activities, resulting in a complete product (Adderley, Ashwin, Bradbury, Freeman, Goodlad, Greene, & Uren, 1975). To understand the effect of PBL on students' creative thinking, Mihardi et al. (2013) used the Know, Want, Learn (KWL) worksheet, a framework that helps students organize knowledge before, during and after a lesson, thereby connecting their prior knowledge to active learning (Ogle, 1986). Mihardi et al. selected students through random sampling to participate in the experiment (Fraenkel, Wallen, & Hyun, 1993). The participants were asked to implement and solve a factory design problem and complete pre-project and post-project questionnaires. The results indicated that students' creative thinking using PBL was higher than when using other methods.

Furthermore, PBL enabled students to propose collaborative group ideas to achieve their final goal, an end-to-end project. PBL is considered a suitable method for preparing students with the skills needed for group creativity.

Zhou et al. (2009) tested group creativity in the development of PBL among engineering students. The participants were two groups of engineering students studying for master's degrees, with data collected through observations during the experiment.

The groups were asked to complete an assignment from the field of engineering and deliver a report. The research found that peer learning (learning from other group members) differed according to project type and field. They also found that PBL can build wide knowledge for students. These conclusions illustrate the influences of PBL on students' learning and learning by collaborative behavior.

Music Technology Education

As it has evolved, the New Interfaces for Musical Expression (NIME) field has increasingly focused on teaching the design and implementation of Digital Music Instruments (DMIs) and finding objective evaluation methods to assess the utility or success of these outcomes. Jorda & Mealla (2014) proposed a methodology for teaching NIME design with a set of tools meant to inform the design process. This approach has been applied in a master's degree course focused on exploring expressiveness and the role of mapping components in the NIME creation chain through a hands-on and self-reflective approach based on a restrictive framework consisting of smart-phones and the Pure Data (PD) programming language.

The outcomes gained by the students through this iterative methodology indicated that: all the students, some of whom had never performed music or programmed computers, were able to effectively engage in the NIME design processes, and to develop working NIME prototypes that fulfilled all the requirements; the assessment tools proved to be a consistent method for the evaluation of the NIME systems and performances; and analyzing the design processes leading to the evaluated outcome demonstrated traceable progress in the students' achievements. Although these findings were obtained within the specific context of a NIME course, the researchers believe that several of these solutions and methods could be extrapolated to more generic contexts, i.e., other NIME or even HCI courses, design methodologies, and evaluation methods in both fields, and could therefore assist teachers, designers and practitioners in general.

A less abstract example is the course Design and Development of Musical Controllers among Musicians and Novices, which was taught at New York University (NYU) by D'Arcangelo (2002). While no formal musical background was required for the course, musicality was considered as a driving force in the design process. The class was really an experiment with an educational approach that required each inventor to set his or her personal musical standards, even if minimal, as the basis for musical interface innovation. The design challenge was articulating expressive goals based on these musical standards, and then working with the tools and technologies required to achieve them.

Early discussions on the qualities of music and what constituted musical expression helped students to articulate the musical direction they wanted to pursue. Their approach to music was open and egalitarian. The course encouraged sensitivity to how musical styles vary across cultures and throughout history, ranging from the sacred to the profane, and popular to classical, and taking on novel forms with the advent of new technologies.

However flexible and open their definition of music was, each student needed to adopt some sense of musical style to root the invention of his or her new instrument. As a result, the projects were explicitly expected to break from the traditional paradigm of musical instruments and present new models of musical expression.

A framework enabling speedy design and prototyping of passive mobile device augmentations was introduced by Michon et al. (2017). This framework was suitable for developers with a background in music, sound design, and FAUST programming language for synthesis. The researchers organized a one-week workshop at Stanford University's Center for Computer Research in Music and Acoustics (CCRMA) and taught seven participants how to make basic musical smartphone apps using their Smart Keyboard App Generator. They also taught them how to use 3D printing for mobile device augmentations that enabled users to make sounds or even use the phone as an instrument.

The participants were free to invent, design, and make any musical instrument or sound toy for their final project. In one week, participants mastered all these techniques and designed and implemented highly original instrumental ideas.

3 Computer Music Education for Skills Development

This section describes the study's educational method, the categorization of music-tech projects and analysis results of project evaluation. Seventy-five music-tech projects developed by 183 students between 2016 and 2020 were examined.

The main questions we asked were:

RQ1: In what ways do students' and teams' characteristics align with the project's creativity, multidisciplinarity, artistry, and risk level?

RQ2: In what ways does the music-tech project type align with the project's quality and students' learning outcomes?

RQ3: How does a team's composition affect the project's quality?

RQ4: What music-tech characteristics are interdependent and affect the creativity level of the project?

Section 3.1 opens with a characterization of the music-tech projects. Section 3.2 includes definitions of students' skills and the projects' characteristics. In Section 3.3, the experimental method and students and projects grading methods are explained. The evaluation analysis results are presented in Section 3.4.

3.1 Computer Music Projects Categorization

First, computer music project categories were designated. The five categories were essential to artistic and computational models that can be used to develop a project in the category. All 75 projects were divided according to these categories, and the projects' characteristics in different categories were analyzed. Here are the categories:

1. **Music experience:** This refers to a project or an application with specific music functionality (i.e., playing or learning), which does not involve creation or generation. This category includes music games that combine musical elements, sounds or musical pieces, educational applications, players, streamers, recorders, editors, or digital controllers. Applications in this category are interactive, and the interaction is more functional than artistic, but the application does not make artistic decisions. Although some of the application is details can change during the development process, the developers can plan and design the project in detail before the development process. The quality of the product is not guaranteed with regard to the value of the user experience for the user. Nevertheless, the application can be defined and fully illustrated and planned, making the level of risk relatively low.

2. **Creative expression:** This refers to an interactive application that displays a musical interface to the user and can respond to the user's interaction. In this category, applications determine artistic considerations in the interaction with the user. An example is an application for music creation in which the user actually creates something musical.

3. **Analysis and Generation:** This refers to an algorithm that analyzes or generates music pieces, such as a Music Information Retrieval (MIR) algorithm for feature extraction of genre classification, a personalized playlist generator, or a generation algorithm (music, visualization, etc.). A generation project is not interactive.

4. **Smart Interaction:** This refers to an application that combines user interaction and creative expression with analysis or generation. For example, an application that analyzes users interaction or music improvised by the user, which generates a response that is played to him/her.

5. **Sonification:** This refers to a data-driven project that uses non-speech audio to convey information or to perceptualize data. A sonification algorithm builds an auditory representation for given data, such as sonification of stock prices, a text, or brain activity. Sonification can be used for scientific, experimental, or artistic purposes.

In a project involving sonification, the developer may have a general idea of how the data should be converted, but most details are determined during the development process when the data is processed and the developer is more familiar with it. This can be considered a generation project, but with an additional level of abstraction.

In sonification projects, data is first transformed from another domain prior to being generated, which differs from generating music using compositional rules or music pieces. Such projects were considered high-risk for the purposes of this study.

3.2 Characteristics of Students and Projects

This section reviews the characteristics by which we evaluated the students and projects involved in the study.

1. *Artistic ability* included skills and talent to create expressive works of art: painting, drawing, sculpting, musical composition, etc.

An *artistic application* is an application where the students used or combined musical elements or artistic aspects in their project. For example, an application that interacted with a human enabled the student to create a piece of art using an algorithm that analyzed or generated music.

2. An *interactive application* is an application that allows users to enter data or commands, such as a controller or instrument, a music player, a synthesizer, an educational application, a DAW (Digital Audio Workstation), an application for music creation, etc.

3. An *artistic-interactive* application is both an artistic and interactive application: for example, an application which enabled the user to improvise music through user interaction and with the application analyzing, generating, and playing a musical response. A music player is an interactive application, since the user interacts with the application by entering functional commands like "play," "stop," and "like," but it is not an artistic application, and therefore it could not be considered an artistic-interactive application. Another example is an application that analyzed musical pieces and generated a new musical piece based on another piece, which was an artistic application, but not an interactive one.

4. A *multidisciplinary skill* combine multiple disciplines to redefine problems outside of normal boundaries and reach solutions based on a new understanding of complex situations.

A *multidisciplinary project* is a project where a number of disciplines are incorporated into the project's development in order to arrive at a solution.

5. *Creativity* is the skill or talent that incorporates the imagination to create and solve a problem. A creative project is a project where a relatively high level of imagination and originality is used to solve the problem and to create an original, unique, and innovative project. A creative project is not necessarily an artistic project. For example, unlike any other game, a new game is a creative project, but a music player is not, since it imitates standard techniques and interfaces for music playing.

6. *Elaboration* is the ability to elaborate a part of the project (component), engage, describe the number of dependent components, and to isolate components.

7. The *novelty in form* and *novelty in content* is a two-dimensional creativity assessment model, from complete imitation to original creation (originality), according to Nillson's Taxonomy of Creative Design (2011). In our analysis, the dimensions were scaled from 1 to 5 (imitation, variation, combination, transformation, and original creation) and were interpreted according to:

- The *novelty in form*: This refers to the novelty in the project's source code: how many new components, how different the architecture (according to the initial project given in class, and the assignment upon which it is based), and more.

- The *novelty in content*: This refers to the novelty in the project content: algorithms, out-source libraries, complexity of run time, optimizations, etc.

8. A project with a *high level of risk* is a project where the main idea and the problem it seeks to solve are clearly defined before the development begins, but many designs and implementation details are unknown or unclear in advance. Some research and trial and error processes are needed to define these details and advance from one phase of the development process to another. Therefore, the project outcomes are uncertain, as is whether the students would succeed in achieving their goals and provide a solution for the problem they seek to solve.

- A *research project* refers to a scientific endeavor to answer a research question. Specifically, such projects take the form of case series, case-control studies, cohort studies, randomized, controlled trials, or secondary data analyses, such as decision analysis or meta-analysis. The students have some questions they want to answer, and they develop an algorithm or an application to accomplish this.

- *Applicative project,* in contrast to a research project, refers to a project where the students have an idea for a product that solves a problem. The project outcome is an application prototype, a POC that would be developed and tested on potential users.

3.3 Method

3.3.1 Educational Method

The course, “Computer Music” that taught by Dr. Revital Hollander-Shabtai at the Interdisciplinary Center, Herzliya, began by introducing music technology, followed by a discussion on current needs and future directions in this field. Over the following three weeks, the participants learned about musical elements in theory and practiced them using the SuperCollider language: notes, pitch, timbre, tempo, rhythm, melody, harmony, texture, structure, and the MIDI protocol.

After four weeks, the participants were divided into teams of one to four participants. They were asked to present and discuss three ideas for the final project in class and to choose one of the three. The next task was to build a presentation that described the project (Appendix 5). This presentation was updated after each phase and was used in the final presentation, on the course’s demonstration day.

During the development process, each team had two meetings with the course teachers. The first took place after the team had devised three ideas for their project, and the second occurred after the first development phase. During the first meeting with the course teachers, the team members presented themselves, their background, their interests, and the three ideas and possible solutions, in addition to the idea and development options.

Each idea's level of risk was discussed and how it matched the team members' interests and abilities.

After choosing one of the three ideas for the project, the students started planning an Agile development process and divided three phases. At the completion of each phase, a deliverable and working part of the project was delivered and tested with the potential users. In the case of a non-interactive project, the deliverable element was a preliminary output of the algorithm. Some audio output examples or videos that demonstrated the user using the prototype were submitted in both cases.

The teams learned the problem domain and solved the problem through a learning-by-doing or PBL process. Each team reviewed relevant papers and chose one paper most relevant to their project. They presented the paper in class, followed by the project presentation that described their project goals and its three phases. A class discussion took place and feedback was given in class. A demonstration day was held after the end of the semester, and the students presented their presentations and projects.

3.3.2 Project Evaluation Method

The course teachers and the students ranked the projects done in 2020 using the Muzilator platform. Data was collected from the students at the beginning and the end of the semester using pre-project and post-project questionnaires. At the beginning of the semester, the students were asked to rank their own creativity, multidisciplinarity, and self-learning (autodidactic) abilities and to provide information about their music and software programming backgrounds.

At the end of the semester, students were asked to rank their projects and their learning outcomes. The students were ranked according to the following criteria from the data collected in the pre-project questionnaire (Table 1).

Criteria	Question
Creativity	*"How do you define yourself Creative from 1 to 5?"*
Multidisciplinary	*"How do you define yourself Multidisciplinary from 1 to 5?"*
Autodidact	*"How do you define yourself Autodidact from 1 to 5?"*
Gender	*"Men/Women"*
Musical Background	*"Do you play an instrument? If yes, what instrument and how many years?"*
	"Do you play other instruments? If yes, how many?"
	"Do you read music notation?"
	"Do you have other music skills?"
Professional Background	*"Do you have experience in programming? How many years?"*
	"Did you work with the Agile methodology?"
	"What are your favorite methods? (server, algorithms, etc.)"
	"What programming languages are you familiar with?"
	"When did you start to learn programming?"
	"Do you prefer to work with a team or by yourself?"

Table 1: *The Participants' Pre-Project Questionnaire*

The grading method for musical background:

- **1** - No background.
- **2** – Beginner: Played an instrument for 1–2 years.
- **3** – Intermediate: Played an instrument for 3–5 years.
- **4** – Advanced: Played an instrument for at least 5 years, played additional instruments or sang, or majored in high school music or conservatory music.
- **5** – Expert: Academic background in music, or a professional musician.

The grading method for professional background:

- **1** - No experience.
- **2** – Trainee: 1–2 years of experience in a student position, or a technological position other than a developer in the Israel Defense Forces (IDF[1]).
- **3** – Junior: 1–2 years of experience as a senior developer in the industry (or the IDF).
- **4** – Senior: 3–5 years of experience as a programmer in the industry (or the IDF).
- **5** – Expert: More than 5 years of experience in the industry (or IDF) and additional experience as a team leader or specific expertise in machine learning, data science, backend, etc.

[1] The IDF has high-level elite computer units and as a result, some of the computer science students are graduates of the above units.

The projects were ranked according to the subjective criteria seen in Table 2, where 1 is the lowest level and 5 is the highest level:

Criteria	Scale
Creative	1 to 5
Artistic project	1 to 5
Artistic interaction	1 to 5
Interactive	1 to 5
Multidisciplinary	1 to 5
Risk level	1 to 3
Research	Boolean (0/1)
Entrepreneurial	Boolean (0/1)

Table 2: *The projects' evaluation criteria and scale*

3.4 Main Results

The students' project distribution was analyzed according to project categories (see Figure 2). Of all the projects, 67.6% were interactive projects from the music experience and creative expression categories, while 13.5% were analysis and generation projects, and 10.5% combined interaction and algorithms.

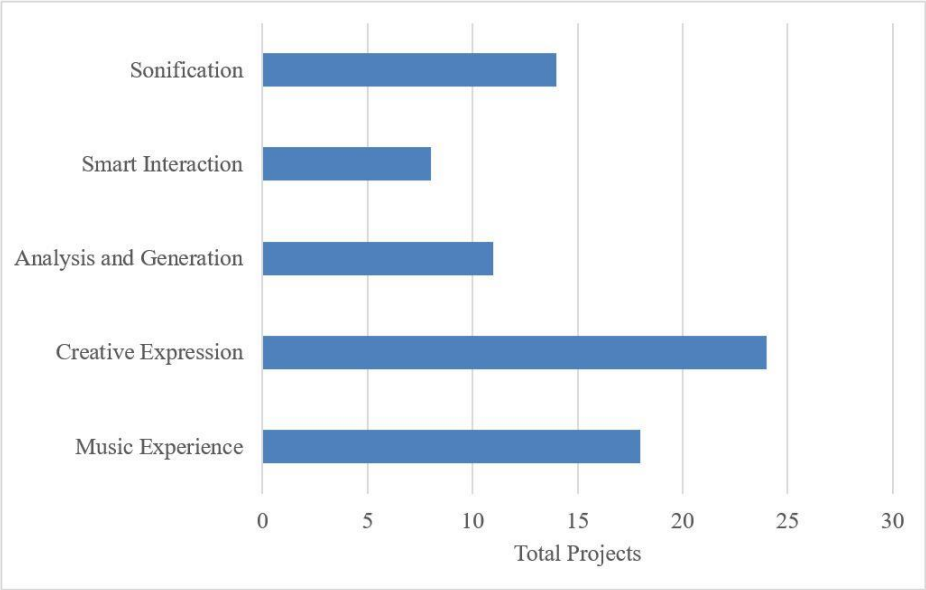


Figure 2: *The distribution of students' projects by category*

Here is a summary of the main results. The students' evaluations were first examined, followed by an analysis of the team and the project it developed.

3.4.1 Individual

Evaluating or measuring creativity was non-trivial. To achieve high marks for the projects' creativity ranking, projects were ranked in two ways. The first involved applying definition 5 for creativity presented in Section 3.2 ("Creativity is the skill or talent that incorporated the imagination to create and solve a problem"), to rank a project's creativity level by assessing the project concept and the overall imagination and originality demonstrated during implementation. The second involved applying Nilsson's taxonomy for creative design (Nilsson, 2011) to rank each project according to his model's two dimensions of novelty: in form and in content. Table 3 includes the projects' average grades in each category. The comparison (see Figure 3) shows a high correspondence between the two ranking methods.

	Creative Expression	Smart Interaction	Music Experience	Analysis and Generation	Sonification
Nillson's average rank	2.93 (.41)	2.95 (.47)	3.12 (.34)	3.3 (.31)	**4.1 (.21)**
Creativity average rank	2.82 (.30)	3.07 (.35)	3.37 (.28)	3.0 (.32)	**4.8 (.24)**

Note. The data is represented as M (SD).

Table 3: *Creativity average and standard deviation of projects by category*

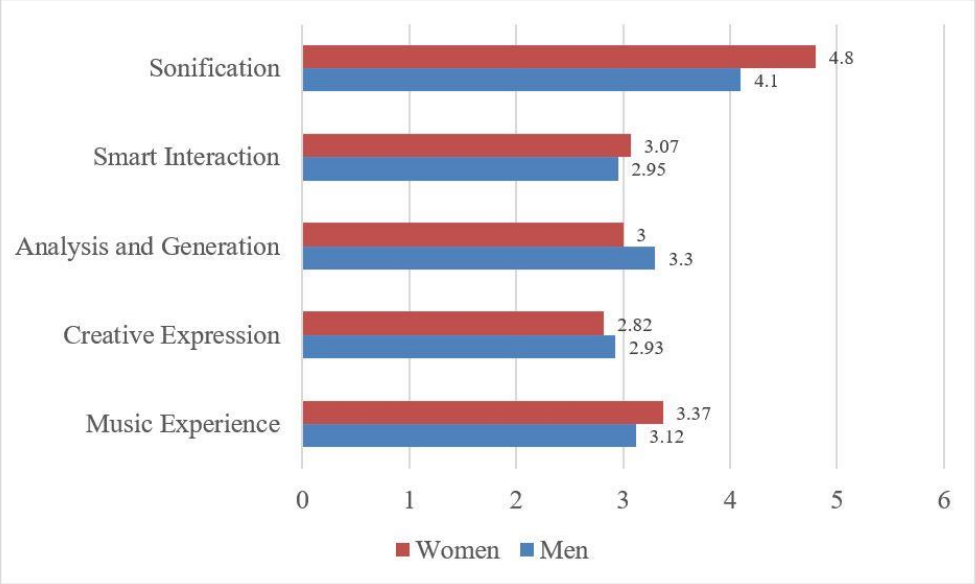


Figure 3: *A correspondence between Nilsson's creative design and the average creativity rank*

Here are the conclusions, on an individual level:

1. Participants who defined themselves as autodidacts were more willing to explore and combined more new disciplines in their projects. Their projects' multidisciplinarity and artistic ratings were relatively higher than those of other projects.

2. Participants who ranked themselves as highly creative developed a project with a higher creativity rating.

3. Participants who developed projects with the highest level of risk had high self-esteem in all the factors of autodidacticism, creativity, and multidisciplinarity.

	Autodidact	Creative	Multidisciplinary
Risk = 1	3.50	3.42	3.62
Risk = 2	3.72	3.63	3.9
Risk = 3	**4.02**	**4.09**	**3.97**

Table 4: *Participants' self-esteem characteristics' average compared according to the projects' risk level*

4. No significant difference was found between men's and women's self-esteem in terms of being autodidactic, creative, and multidisciplinary. Nevertheless, men's rates were slightly higher than women's in all categories.

	Autodidact	Creative	Multidisciplinary
Men	3.81	3.86	4.07
Women	3.73	3.66	3.98

Table 5: *Participants' self-esteem characteristics' average compared by gender*

3.4.2 Project

1. High-risk projects were more artistic and creative than other projects, and vice versa (RQ1).

2. Teams that developed a project with a low level of risk received lower creativity ratings (in both of the creativity ranking methods). In contrast, participants who developed a high level of risk projects received higher creativity ratings (RQ1).

3. A strong positive correlation (=0.876) was found between the projects' creativity and multidisciplinarity. Students who combined more disciplines in their projects tended to be rated as more creative (RQ1).

4. Students who developed projects with a high risk developed more creative projects and combined more disciplines in their projects.

Drawing on the previous findings that creativity and multidisciplinarity have a strong dependency that can affect how the project developed, those variables and risk levels were compared according to project type (Table 6). This analysis reinforced the conclusion that project type affected students' creativity. For example, students who developed sonification and generation projects received high multidisciplinarity and creativity ratings (RQ2).

	Music Experience	Creative Expression	Smart Interaction	Analysis and Generation	Sonification
Risk	1.3 (.63)	2.4 (.90)	2.6 (.80)	3.9 (1.14)	**4.78 (.64)**
Multidisciplinarity	3.4 (1.09)	3.1 (1.01)	3.8 (.70)	3.8 (.64)	**4.37 (.36)**
Creativity	2.58 (1.11)	2.9 (.94)	3.5 (1.07)	2.7 (1.12)	**4.42 (.57)**

Note. The data is represented as M (SD).

Table 6: *Creativity average and standard deviation of projects by the participants' characteristics*

4 A Collaborative Plugin-Based Platform as a Creative Education Tool

This chapter describes the Muzilator platform as a creative educational tool to enhance creativity, artistry, multidisciplinarity, and collaboration skills. This experiment was the first pilot done with the platform on a relatively large group of users: 47 Computer Music course students (32 men and 15 women) at the Interdisciplinary Center, Herzliya, who took the class (the same course that mentioned in the previous sections) in 2020. The goal was to learn about the platform's contribution to the students' and the teams' learning experience and outcomes, and the projects' creativity and quality.

4.1 About Muzilator

Muzilator is a plugin-based web platform for interactive applications, intended for musicians, novices, developers, and researchers (Hollander-Shabtai & Peretz, 2020). For app users, Muzilator improves creative musical expression, interaction, and creative skills by enabling users to interact with Muzilator's interactive musical interface and applications. For developers, Muzilator exposes APIs that can easily add their plugins to the platform. Muzilator records all interaction data and data transferred

between plugins, enabling researchers to build or use existing plugins or apps in their experiments and to analyze the recorded data.

The Muzilator platform was designed in a plugin manner (Figure 4), and a plugin may have any functionality.

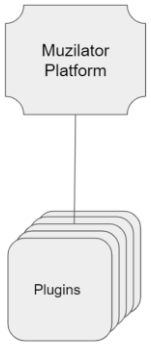


Figure 4: *Muzilator hosts web applications as Plugins*

There are two main types of plugins (Figure 5): applications (app) and libraries (libs). An app can be, for example, an interactive musical instrument, creation or educational app, or a game. Libs can be a controller, an external MIDI device, an analyzer (online/offline), a sound engine, a profiler, etc.

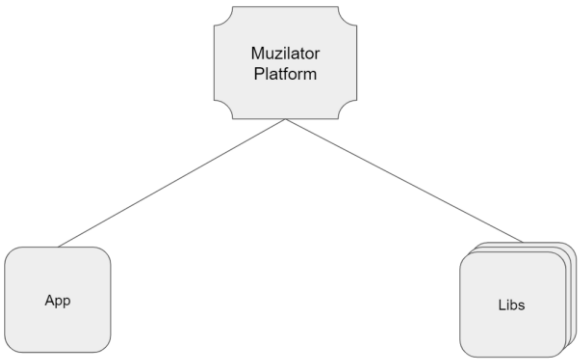


Figure 5: *Muzilator plugins types: Apps and Libs*

All plugins can communicate with each other through Muzilator channels (Figure 6). The channels transfer data from plugin A to plugin B if a channel exist between those plugins. Each Muzilator app can use any number of libs. The app is responsible for loading libs, connecting channels between plugins, and for the app logic that uses and synchronizes between the libs. The Muzilator architecture design allows any web application to be uploaded to the platform as an independent app or lib. Each plugin can be developed by a different developer and can be easily integrated with other plugins. Muzilator developers can benefit from being a part of a community of developers that create interactive musical applications and share any part of them with the community as plugins.

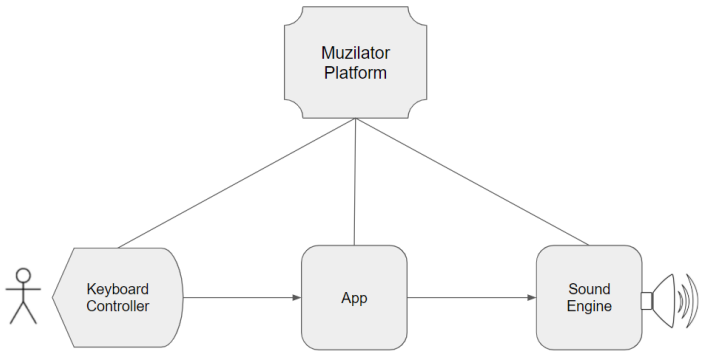


Figure 6: *An Instrument App Uses Two Lib Plugins: A Controller and A Sound Engine.*

As an initial set of plugins, a set of plugins and tools (integrated to Muzilator) was designed and developed for all students, including a dedicated debugger, which helped with communication between plugins, and as a tool for students with no musical background. Tutorials and guides were drawn up and handed out to the students with the basic set of plugins.

4.2 Creative Education and Collaboration Tool

The uniqueness of the educational method enabled by the Muzilator platform can be reflected in a number of areas:

1. *Development of independent shareable plugins:* The students developed their ability to focus on a specific entity as a plugin to write their plugin, or used independent entities that already existed in the platform as a vital software design capability skill. From our experience, without this mechanism, most students failed to separate the different components or layers of the projects, which resulted in poor coding and complex development or maintenance processes. Also, using the Muzilator, the students were able to focus on creative ideas regarding the responsibility of a specific plugin and optimize its functionality and uniqueness.

2. *A platform for everyone:* The platform architecture enables students to write plugins and easily add them to the platform, regardless of their level of programming skills or their level, if any, of musical or artistic background. The plugins are written in JavaScript, a widely common programming language for web development and the applications are browser-based applications that can use the Web-Audio API. It enables the students to focus on the innovative and musical aspects of their project.
3. *Use of existing plugins:* The students have a variety of artistic and computational projects. The participants' choices are based on their preferences: artistic HCIs, sound engines, players, recorders, profilers, applications, online/offline algorithms for analysis, music generation, prediction, profiling, and more.
4. *Software design and software engineering:* The platform includes a software development kit (SDK) to build and integrate plugins quickly and easily (Appendix 1). The SDK can be used in any JavaScript framework and installed via any web package installer. The platform also suggests a state machine interface that conveniently presents a state machine's concept and its use (Appendix 4). Developers and students can share their applications' Entity Relationship Diagram (ERD) with the community for future use.

5. *Work with a community:* Working as part of the platform community of at least 47 participants enabled the students to achieve a comprehensive perspective of the processes involved in platform design, components, and experience integration, to collaborate with individuals and other teams working on other projects, and to share their plugins.
6. *A unique Agile and artistic process:* The process required the students to develop a project in three phases, share the project deliverables at the end of each phase, use other projects, and give other students their feedback. This process was guided and monitored by the course team.
7. *Write and use APIs:* This platform enabled students to combine independent web applications through a unique API and to have them communicate with each other. The students learned how to bind an out-sourced platform, write their own plugins, how to expose their APIs to the community, and how to use an API of other shared plugins.
8. *Versatility:* Since a plugin can have any functionality in any domain, the students were able to easily combine art, technology, and science across different disciplines.
9. *Data recording and storage:* The platform has a built-in data storage mechanism in its recorder for facilitating information interaction between users, enabling them to perform analysis and achieve optimization for development processes.

4.3 Experiment and Educational Method

The educational method combined a learning process divided into three phases and used a plugin-based platform for musical applications. The three phases were: Assignment 1: Exploring an HCI: Designing an Interface Plugin; Assignment 2: A computational plugin; and Assignment 3: The Final Course Project. In the first two phases, the students focused on exploring a specific interactive musical application component. They received an initial plugin project and continued to develop it independently. The submitted project was then uploaded to the Muzilator platform. The following contains more details about the three phases:

- **Assignment 1:** Exploring an HCI: Designing an Interface Plugin. In this assignment, the students focused on user interaction and the user interface of a simple musical application (controller). Using creation methods from music and art, the students were provided with a theme or a trigger for a new idea. In this experiment, Bubbles, a simple and basic controller that displays random circles with random colors (Appendix 2) was used. Each circle is mapped to a random note (Figure 7a). The students were asked to develop a music controller or a simple musical application for some specific purpose: music interaction, a game, or a tool. They designed and implemented the controller's display while considering the target user’s interaction and experience.

The students had to combine programming and artistic abilities to design the HCI's features, including size, color, shape, configuration (spatial organization), graphics, animation, movement, gestures, mapping of graphical elements to musical elements, musical context, human factors, and the use of photos and videos. They also had to adjust some of the features to the potential user to optimize their interaction. In addition, the students were responsible for sending the user's interaction data from their plugins to other plugins for future use. Figures 7, 8 and 9 illustrate examples of the students' Assignment 1 with three levels of abstraction. Figure 7b-7f show five different uses with minor graphical changes (mostly in shape, configuration, colors, spatial organization, and pitch mapping to a circle). Figures 7b and 7c show a simple controller, where a significant focus was given to its design and the spatial organization that considers relations between notes and chords, Figures 7d-7f illustrate an eye tracker controller, where the user played a melody using his/her eye movements, and the primary focus was on human factors. Three different configurations were designed and used for three scenarios and musical contexts. Figure 8 shows the next level of abstraction with the use of Bubbles. In these projects, the students designed a tool or a game with a higher level of sophistication. Additional elements were combined in the interface and logic or animation was added to an interface for a tool or a game.

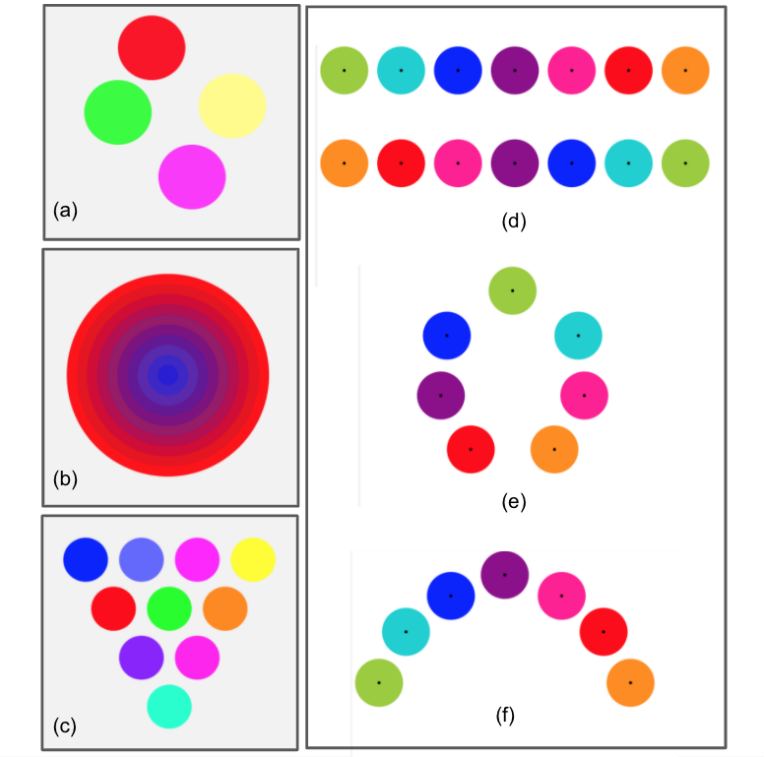


Figure 7: *Controllers developed by the participants in Assignment 1: Visual Transformation. Figure 7a is the given Bubble controller. Figures 7b and 7c demonstrate simple musical controllers with a specific design for specific musical elements, and Figures 7d, 7e, and 7f demonstrate three different configurations for an eye-tracking musical controller.*

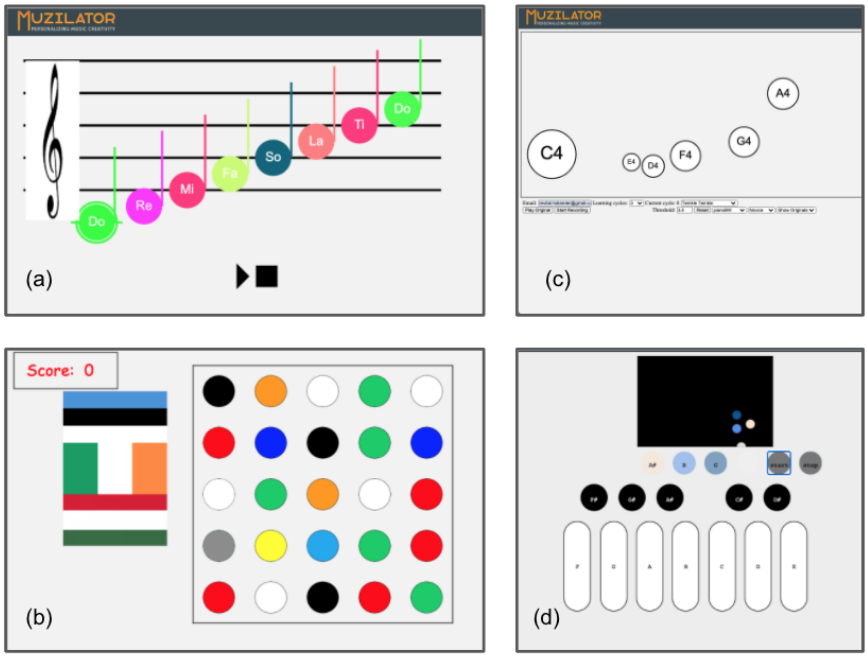


Figure 8: *Controllers developed by the participants in Assignment 1: Music Composition. Figure 8a shows a simple app for music composition, where the user composes a melody and the app continues the composition. Figure 8b shows a variation of a word-search game. A searched word is represented by triplets of colors of flags. When the user clicks on a circle, part of a national anthem is played. The user has to find a triplet where the same anthem is played and then choose the right flag. Figure 8c shows an application who learned how the user perceives a melody in a two-dimensional space. The user plays a melody on an empty canvas on the application several times, and the application generates a controller for the user. Figure 8d shows an animated chords game in which the user creates a chord by choosing three notes. The notes are mapped to the animated circle that moves in the black rectangle. Each time a circle hits one of the edges of the rectangle, a note is played.*

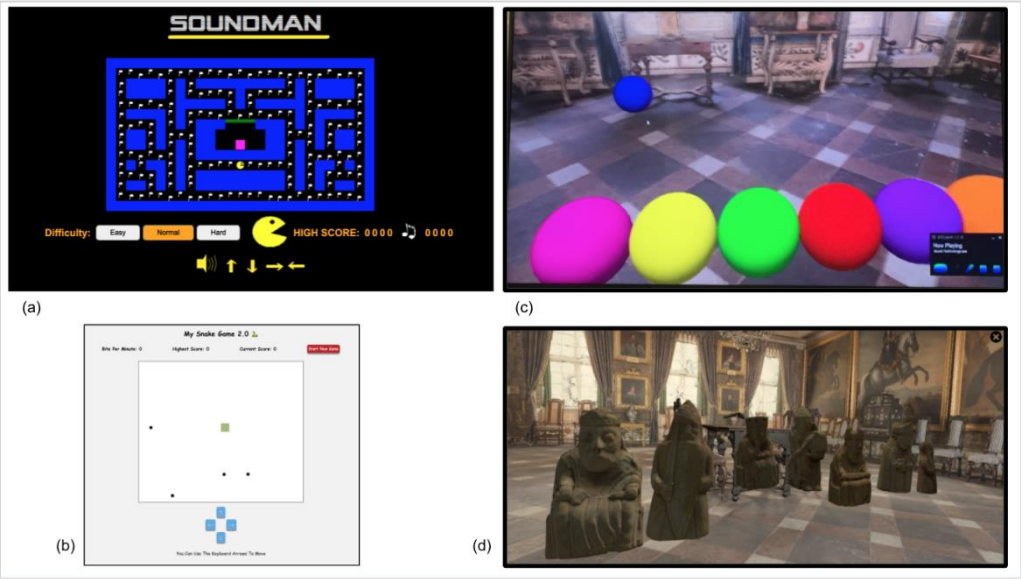


Figure 9: *Controllers developed by the participants in Assignment 1: Generalization. Figure 9a shows Soundman, a musical Pacman game where the user navigated with sound. Figure 9b shows a musical snake game, and Figure 9c shows the Bubbles controller converted to a 3D VR controller with additional abilities, such as drag and drop, that enable the user to organize the elements in a 3D space.*

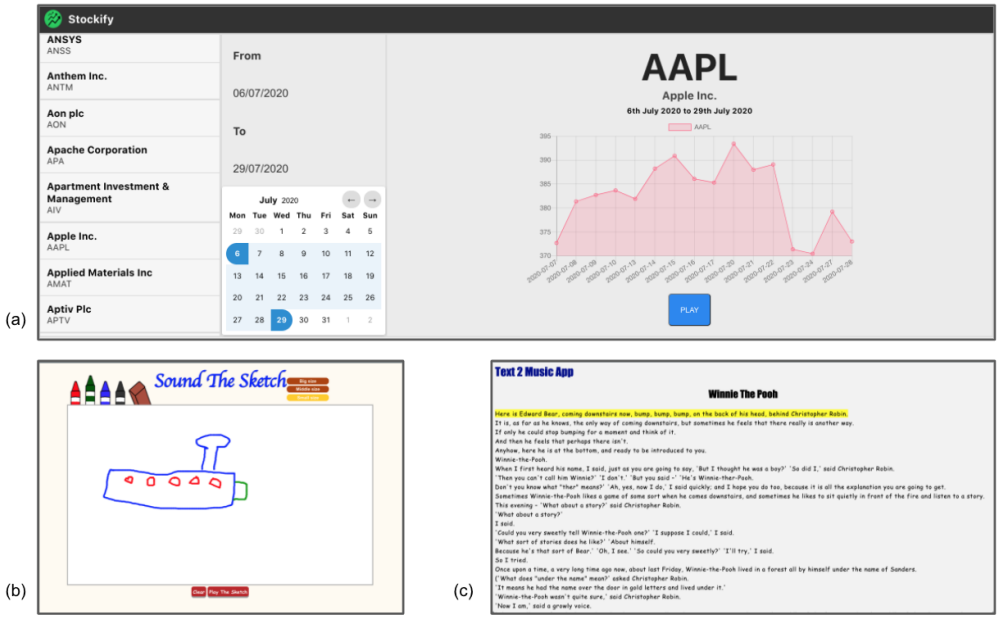


Figure 10: *Sonification projects developed by the participants. Figure 10a shows a stocks graph sonification. Figure 10b shows a musical painting app in which the user draws a painting, and the application plays the sonified painting. Figure 10c shows an application that takes short stories, and, using sentiment analysis and sonification, creates a playback for the reader while the user reads the story.*

- **Assignment 2** - A Computational Plugin. For this assignment, the students focused on a logical component of a musical application, such as an analyzer for analyzing user interaction and responding accordingly (Figure 11). The students were asked to build a computational plugin for another student's HCI. This assignment not only helped them learn the importance of collaboration, but also introduced them to the experience of being part of a developers' community. Computational plugins could employ a variety of approaches, such as a generative algorithm that generated music using computational models (Markov chains, genetic algorithms, Google Mangenta, etc.), an analysis of user input in a game and calculation of the score, or an analysis of music played by the user to help determine whether to switch the state in a state machine.

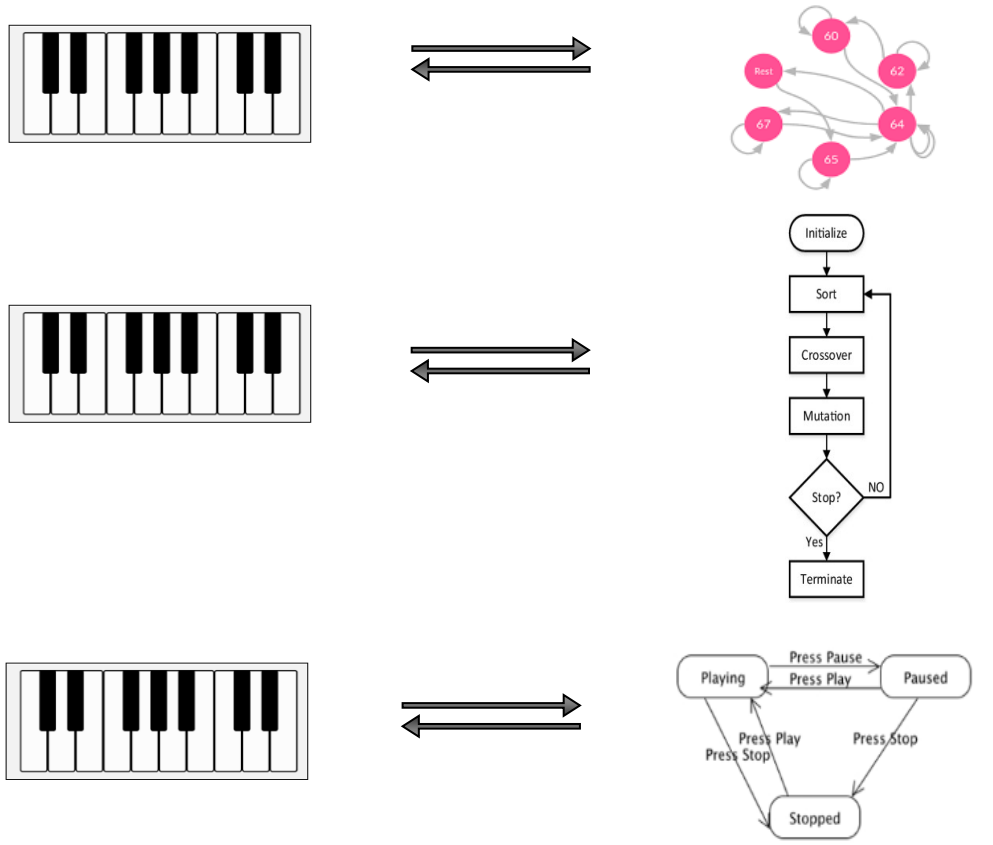


Figure 11: *An example of three applications that used the same controller with different analyzers. The first application used the Markov chain analyzer, the second used a genetic algorithm analyzer, and the third used the state machine analyzer.*

- **Assignment 3 – The Final Project:** For the final project, the students developed an idea for an original music plugin and implemented it:
    - The students were divided into 19 teams, of one to four participants each.
    - Each team designed and developed an original music plguin, such as an interactive app, instrument, generation algorithm, sound engine, sonification, or game.
    - The development process was divided into three phases (according to the Agile methodology), where at the end of each phase, the students submitted a deliverable project that could be used and tested by the plugin's potential user.

In addition to the project's code, the participants were asked to submit two additional files:

1. **API** (Table 7): An application program interface which included:

    i. **Plugin Description:** a description of the plugin functionality and how it worked.

    ii. **ID:** the unique plugin id as registered in the Muzilator platform for reuse and collaboration.

    iii. **Messages API:** the type of messages the controller used to handle their content, for both input and output messages.

2. **Channel-Diagram** (Figure 12): a diagram that include:
    i. Plugin scheme – A scheme that presents the communication channels between the plugins.
    ii. Active channels.

Description	This plugin recognizes chords in the user interaction data.
ID	chords-recognizer
Input Messages	**type**: set-pattern, **pattern**: array[0...127], **channel-name**: analyzer-channel **type**: note-on, **pitch**: number[0...127], **velocity**: number[0...127]
Output Messages	**type**: pattern-recognized, **channel-name**: analyzer-channel

Table 7: *An API example of a Muzilator plugin*

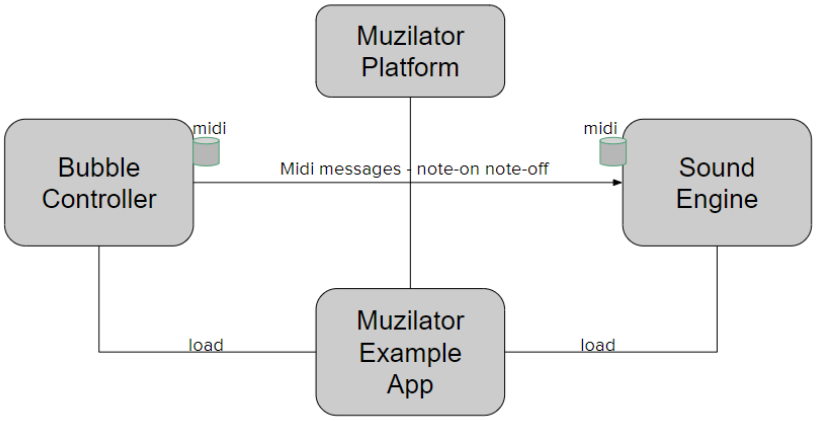


Figure 12: *Example of a channel-diagram. The controller and the sound engine communicate and send messages on the midi channel.*

The following is an examination of the differences in development between music technology projects in terms of music experience and sonification.

Example 1: The Cross Flags Game – A Music Experience Project

Cross Flags is a music experience game using a variation of a word search game, where the searched words are triplets of colors of flags. When the user clicks on a circle, a part of a specific country's national anthem is played. The user has to find a triplet where the same anthem is played for each circle, and then choose the right flag.

The development process was carried out in three phases:

- **Phase 1:** Creating a touch controller of a fixed size (four rows and four columns) and randomly spreading different colors with predefined constraints, such as the constraint that green-white-blue must appear at least once. At the end of this phase, the controller (HCI) handled user interaction (playing the sound according to the event), but there was no algorithmic thinking behind it.
- **Phase 2:** Designing and building a computational plugin that incorporates the user interaction data, analyzes the pattern, and searches for predefined sets of colors to create a known flag. At the

end of this phase, the application, consisting of controller and analyzer, identifies at least two different countries.

- **Phase 3:** Generalizing and completing the project. Using the prototype, defined in stages 1 and 2, the team was required to generalize the project and make it scalable; i.e., the size of the game board could be determined by the user, the collection of countries would be increased, the audio option was more in-depth than simple midi sounds, and more.

Since it was an already familiar game that was converted into a musical game with simple adaptations, once the game was planned and designed, the team started developing it and faced mostly technical concerns rather than user experience or other issues, which reduced the risk level and made the development process more manageable.

Example 2: "Stockify" – A Sonification Project

"Stockify" is a sonification project that transforms companies' stocks trading data into auditory data. The application displays a company list and a calendar to the user, the user selects a company and range of dates, and the application plays those stocks.

The development process was carried out in three phases:

- **Phase 1:** Creating a controller that displays the companies' list, the calendar, and the output chart. The chart is determined by the selected date range and displays the stock chart for that period.
- **Phase 2:** Designing and building a computational plugin that obtains a stock sequence as input and returns the MIDI notes mapping that described the stocks using an auditory medium.
- **Phase 3:** Generalizing and completing the end project. External APIs are added to extract information about the companies, stocks, and various dates to create a complete product.

Throughout the process, the team learned about the complexity of data transformation. Sonification projects, and data transformation into auditory data in general, were designed for several purposes:

1. **Scientific:** Transforming data into auditory data can be used for data exploration, finding patterns in data, and more.
2. **Experience:** Beyond the scientific goal of the project, the project's main aim was for participants to experience the data, hear it, and enjoy the musical experience generated from raw data.
3. **Musical:** Projects of this type dealt with the data's behavior and its translation into an audio representation to create a melody representing the data.

The challenge of data transformation is being able to map the data so that the output is melodic and has a musical sequence, enabling auditory conclusions to be drawn. Usually, this last challenge is the most difficult and requires analysis carried out throughout the process to find and define the most appropriate transformation to address the issue.

4.4 Experiment Results

The participants could choose whether or not to use the Muzilator platform in their final project development. The projects were divided into two groups: Muzilator projects and Independent projects. The following table demonstrates the participants' distribution between the groups:

	Muzilator Projects	Independent Projects
Number of participants	32	15
Number of projects	11	8

Table 8: *The distribution of the participants and the projects*

The participants from the 2020 course who chose to use the Muzilator were asked to complete a post-project questionnaire asking about teamwork, certainty level of their projects, creativity level, and the combination of art, science, and technology.

The participants' answers in their pre-project and post-project questionnaires were compared. A negative difference represented a student who defined him or herself at a high level for any given attribute (in comparison to the team or their post-questionnaire). A positive difference represented the fact that the student produced a product rated higher than his or her self-rating.

4.4.1 Individual

The participants' self-esteem ratios (Smith, Seger & Mackie, 2007) that were reported at the beginning of the semester were compared the results to the rating of the projects that they developed (Muzilator or independent project). The average ranking of all participants who developed Muzilator projects was consistently lower than that of participants who developed independent projects, as shown in the table below. There are two possible explanations for this. The first is that students who developed Muzilator projects were less confident or familiar with other environments, or wanted to learn more or use a more structured and dedicated tool. The second is that students who developed independent projects felt more confident developing in an environment that was more familiar to them, and/or did not want to spend more time learning a new environment.

By comparing the participants' self-esteem ratings to their projects' creativity, autodidacticism, and multidisciplinarity (Table 9), we found that participants who chose to develop their projects independently rated themselves higher than did others. However, participants who developed their project using the Muzilator platform rated themselves lower than did others.

The differences in the participants' levels of self-esteem can be attributed to the participants' professional knowledge, as such participants may have rated themselves as highly creative and autodidactic and may have developed their projects to reflect both their familiarity with their developmental environment and their abilities. Participants who developed their projects in Muzilator had a lower level of self-esteem. By developing independent plugins to Muzilator, the participants used other participants' plugins dedicated to a specific task or computation used by any platform user. In these cases, the participants may have felt comfortable using an existing platform with dedicated computational tasks, and did not develop their projects independently.

I consider myself as… *(scaled 1 to 5)*	Muzilator projects	Independent projects
Autodidact	3.26	4.37
Multidisciplinary	3.79	4.21
Creative	3.75	4.57
Musical Background	1.61	2.10
Professional Background	2.08	2.21

Table 9: *A comparison of participants' self-esteem by project category*.

Post-experiment analysis

1. Most of the participants who developed Muzilator projects and considered themselves highly creative developed more creative projects than did participants who developed an independent project.
2. Similarly, the participants' self-esteem with respect to their creativity and the projects' creativity ratings were compared. Of the participants who used the Muzilator platform, 70% received creativity ratings for their projects equal to or higher than their self-esteem ratings as creative. However, 73.3% of the participants who developed independent projects received creativity ratings for their project, that were lower than their self-esteem ratings of creativity.

Creativity	Muzilator projects	Independent projects
Equal or higher	70%	26.7%
Lower	30%	73.3%

Table 10: *The difference between pre-project and post-project questionnaires in creativity*

4.4.2 Team / Project

1. Muzliator projects received higher creativity and multidisciplinarity ratings than did independent projects (Table 11). The table below compares Muzilator projects and independent projects that were developed in 2020.
2. Elaboration and Nilsson's taxonomy rates were increased during the experiment milestones (Figure 13).

	Muzilator projects	Independent projects
Multidisciplinarity	3.97	2.87
Creativity	4.19	2.01
Risk	3.18	2.31
Artistic Project	4.01	2.91
Artistic Interaction	2.86	1.97
Interactive	3.87	3.51

Table 11: *Muzilator projects' ranking compared to that of independent projects*

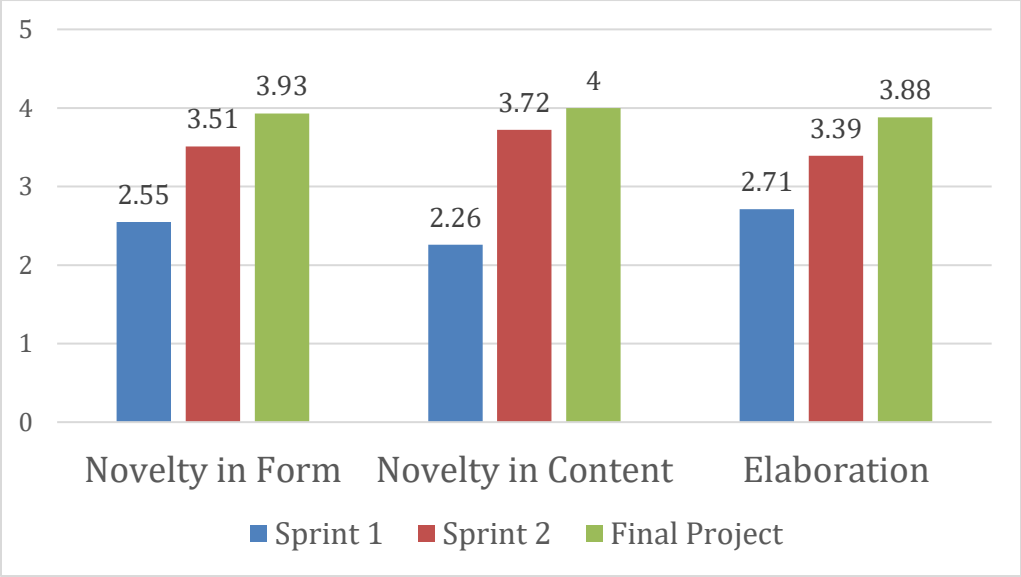


Figure 13: *Elaboration and Nilsson's taxonomy rates*

4.4.3 Gender Differences

For the purpose of comparing ratings based on gender, our findings were based on two analyses:

- **Pre- experiment analysis**

  A comparison was made between the answers of men and women regarding self-esteem, reported at the beginning of the semester. At that point, there was no significant difference in self-esteem between men and women with regard to autodidactism, creativity, and multidisciplinarism. A closer examination of men's and women's self-esteem reveals that women consistently rated their autodidactism, creativity, and multidisciplinarism lower than did men, a finding consistent with the observation that female programmers are less confident than male programmers (Kay & Shipman, 2014). As can be seen throughout this section, the result appeared to remain constant throughout various comparisons.

0	Autodidact	Creative	Multidisciplinary	Musical background	Professional background
Women	3.73	3.66	3.98	1.48	1.83
Men	3.81	3.86	4.07	2.06	2.74

Table 12: *The participants' self-esteem average according to gender*

- **Post- experiment analysis**

  To more fully explore the difference between women and men, various parameters were examined with respect to gender, such as project type (Figure 14), creativity, multidisciplinarism, risk, etc.

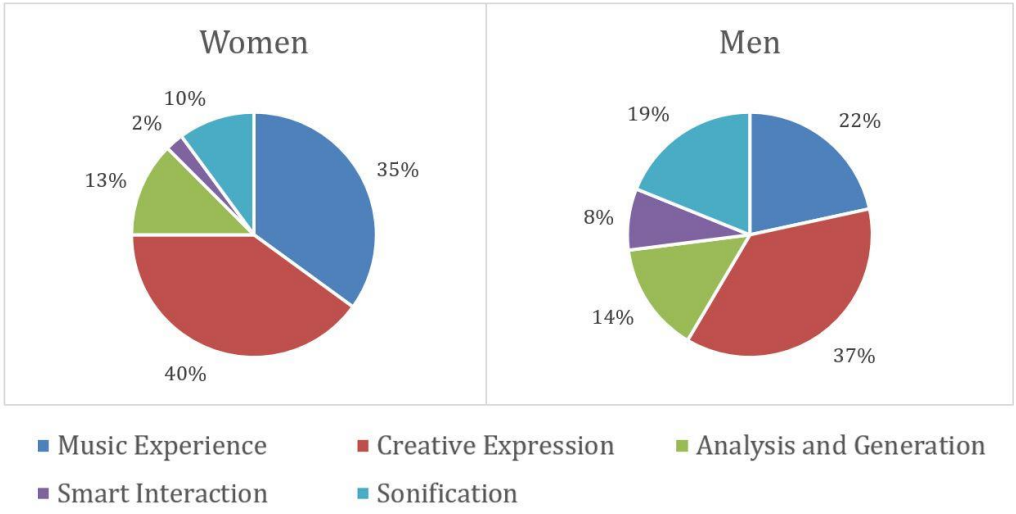


Figure 14: *Projects' category distribution by gender*

1. Teams that contained a certain percentage of women developed more artistic and interactive projects (Table 13). Teams with women only developed more artistic interaction and interactive projects than did other teams, and mixed-gender teams developed more artistic projects than did teams with men only (RQ3).

	Artistic project	Artistic interaction	Interactive
Men	3.21	1.94	3.04
Women	3.52	**2.65**	**3.65**
Mixed	**3.70**	2.37	3.07

Table 13: *Artistic project, artistic interaction, and interactive levels according to gender*

2. Teams with women only developed more artistic projects than did other teams, and mixed-gender teams developed more interactive projects than did men-only teams (Figure 15).

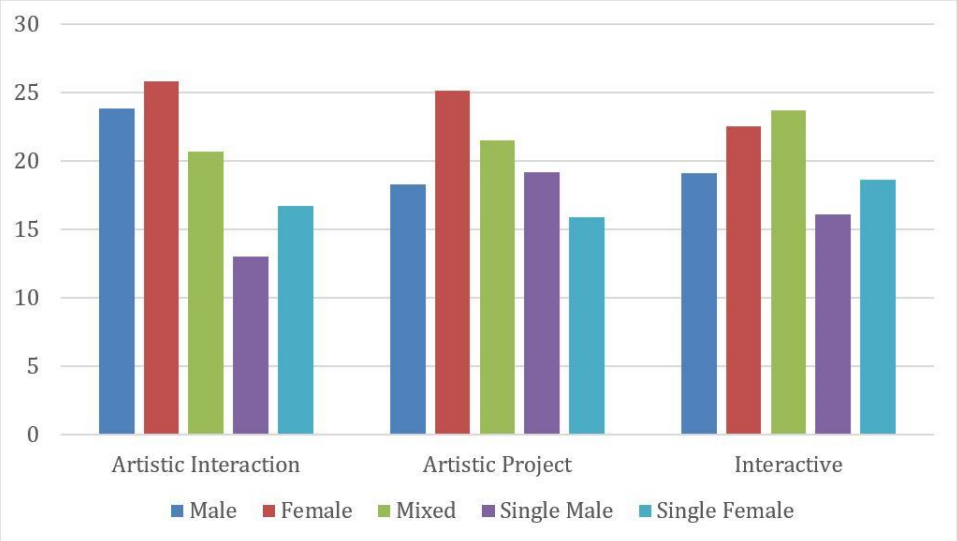


Figure 15: *A comparison of artistic project, artistic interaction, and interactive levels by team composition*

3. The above factors were also examined for the 2020 experiment, using Muzilator. Teams with women only developed more artistic projects and interactions than did other teams. The interactive level was almost equal between men and women, with teams with men tending to have a slightly higher interactive level.

	Artistic project	Artistic interaction	Interactive
Men	4.01	2.60	3.56
Women	**4.24**	**3.14**	3.01
Mixed	4.11	2.46	**3.59**
Single Man	3.20	1.08	2.97
Single Woman	3.22	1.42	2.95

Table 14: *Artistic project, artistic interaction, and interactive levels according to gender composition in the Muzilator experiment*

4. Both genders developed more applicative projects than research projects.

	Research projects	Entrepreneurial projects
Men	45%	75%
Women	31%	92%

Table 15: *The distribution of research and applicative projects*

5. Musical background (MB) affected women's creativity more than that of men. A strong negative correlation was found between creativity and musical background among women (p=-0.64), while the same comparison among men revealed a weak positive correlation (p=0.24). As the musical background was generally lower among women, they nonetheless developed more creative projects than did men with a low musical background.
6. Professional background (PB) affected women's artistry level (p=0.56). Women with professional backgrounds developed more artistic projects than did other women. There was no correlation between men's professional background and the artistic level of their projects. (p=0.06).
7. Men developed projects with a higher level of risk project than did women (Figure 16).

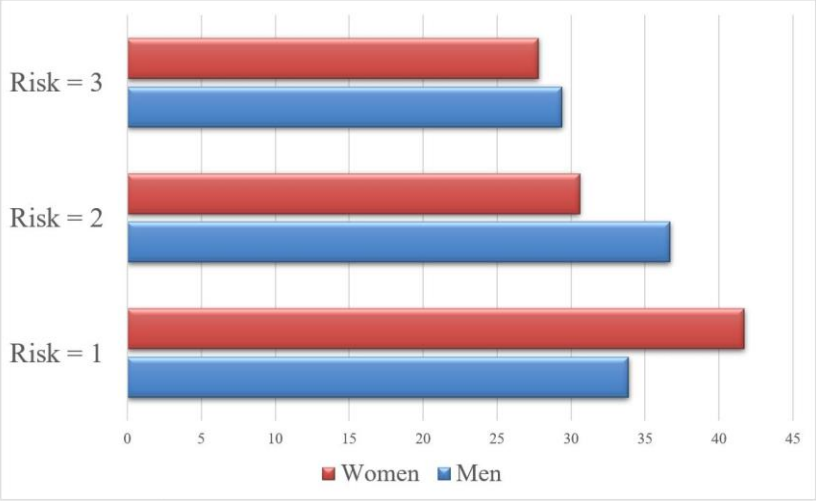


Figure 16: *A comparison of projects' risk level according to gender*

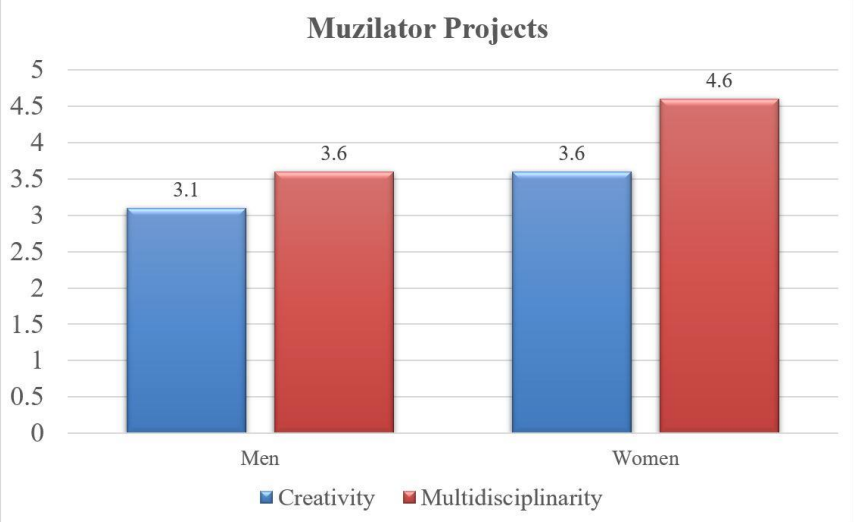
4.4.4 Muzilator Experiment Results

The participants chose whether they wanted to use the Muzilator platform in their final project development or not. The projects were divided into two groups: Muzilator projects and independent projects. The following table shows the gender distribution between the groups:

	Muzilator Projects	Independent Projects	Total
Men	21	11	32
Women	9	6	15

Table 16: *The distribution of the participants' gender and projects*

The results indicate that women developed creative and multidisciplinary projects more than men did, in both types of projects (Figure 17).



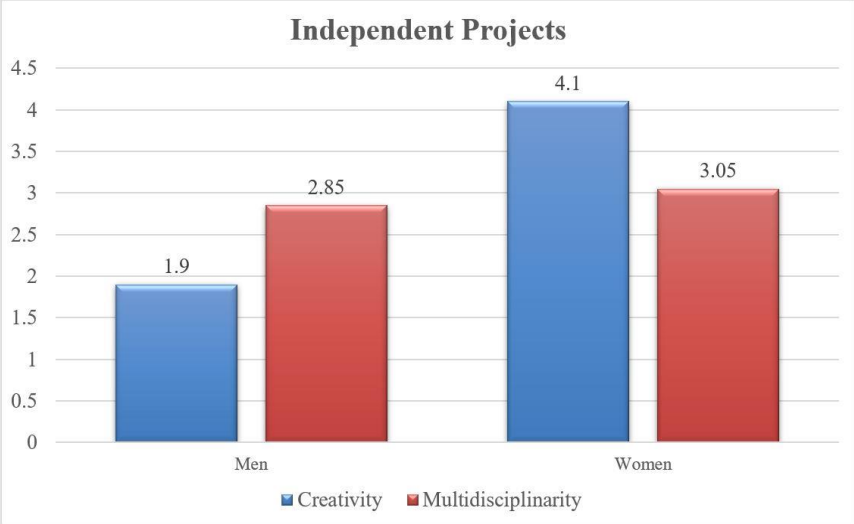


Figure 17: *A comparison of creativity and multidisciplinary levels according to gender*

5 Analysis

This section presents an analysis, using several statistical methods, of the influential characteristics of creativity in the projects. The goal was to find an estimator for creativity, given project characteristics, in order to be able to suggest to students a specific project type that would encourage their creativity level. Seventy-five projects developed by students between 2016 and 2020 were analyzed. Here are the variables we used:

1. ***V*** — a features vector of a given project:

   $$\mathbf{V} = (M, CMPT, R, A, AIN, RE, ENT, MG, FG),$$

   where:

   M = multidisciplinarity level;

   CMPT = computer music project type;

   R = risk level;

   A = artistic project level;

   AIN = artistic interaction level;

   RE = research project indicator;

   ENT = applicative project indicator;

   ME = total number of members;

   MJI = gender majority indicator (1 - men, 2 - equal, 3 - women).

2. ***C(V_0)*** — the creativity level of V_0, where $C(V_0) \in \{1, 2, 3, 4, 5\}$.

3. **PrM** — a 75x9 matrix, where the i^{th} row represents the projects' vector V_i:

$$PrM = \begin{pmatrix} M(v_1) & \cdots & MJI(v_1) \\ \vdots & \ddots & \vdots \\ M(v_{75}) & \cdots & MJI(v_{75}) \end{pmatrix}$$

4. **CrV** — a 75x1 matrix, where the i^{th} row represents the projects' creativity level, $\boldsymbol{C(V_i)}$.

5.1 Statistical Tests

The ranked projects were treated as classified data and as evaluated tests in order to understand the relationship between the ranked projects and creativity. We used the Kendall Tau-b test (Kendall, 1938) and the Somers' Delta test (Somers, 1962). As in these tests, the PrM matrix contains categorical data as well as the CrV matrix.

We defined the following hypotheses:

Let F_i be the i^{th} column in *PrM*, where $i \in \{1, 2, \dots, 9\}$.

H₀: F_i and *CrV* are independent (not associated) variables.

H₁: F_i and *CrV* are dependent (associated) variables.

The following subsections discuss the tests results.

5.1.1 Kendall's Tau-b test

A Kendall's Tau test can be used for hypothesis testing on a small sample size. In this case, the sample size was 75. This is a non-directional test (i.e., for two ordinal variables A and B, where the results are the same for A-B and B-A), and it was used to generalize associations between creativity and all other characteristics. Kendall's correlation coefficient (T_b) scaled from -1 to 1, where:

1. $T_b = -1$ indicates a perfect negative monotonous relation.
2. $T_b = 0$ indicates no monotonous relation at all.
3. $T_b = 1$ indicates a perfect positive monotonous relation.

After Kendall's test was performed, the results were converted into a spearman correlation (Walker, 2003).

The results of the test revealed that that multidisciplinarity, project type, risk level, artistry, and research level scored the highest T_b correlation coefficient value with creativity, at .693, .246, .284, .610, and .314, respectively (Table 17). Multidisciplinarity, project type, risk level, artistry, and research level also resulted in a p-value significantly lower than 5%. Consequently, these results refute H_0 and confirme H_1 with a 95% confidence level. Kendall's Tau-b coefficients were converted to spearman coefficients to strengthen this result, which justified the strong dependency with creativity.

Our main conclusion is that these characteristics affected the creativity level. To confirm and compare this conclusion with others, a Somers' Delta test was applied to gain an additional perspective.

	T_b	ρ	p	Accepted Hypothesis
M	.693	.876	.05*	H_1
CMPT	.246	.362	.05*	H_1
R	.284	.415	.05*	H_1
A	.610	.804	.05*	H_1
AIN	.128	.191	.193	H_0
RE	.314	.456	.05*	H_1
ENT	-.149	-.221	.149	H_0
ME	-.051	-.076	.584	H_0
MJI	-.085	-.127	.404	H_0

Note. $^*p < .05$

Table 17: *T_b correlation coefficient values and significance levels between creativity and the project's features*

5.1.2 Somers' Delta Test

Somers' Delta test is a directional test (i.e., for two ordinal variables A and B, the A-B result is not the same as B-A) of association between two variables. Somers' D results (S_d) take values between -1, when all variables values disagree, and 1 when they all agree.

Creativity was defined as the dependent variable and was tested with every column in PrM. As a result, multidisciplinarity, project type, risk level, artistry, and research level scored the highest S_d value, with creativity as a dependent variable, at .717, .248, .310, .631, and .403, respectively (Table 18).

	S_d	p	Accepted Hypothesis
M	.717	.05*	H_1
CMPT	.248	.05*	H_1
R	.310	.05*	H_1
A	.631	.05*	H_1
AIN	.139	.210	H_0
RE	.403	.05*	H_1
ENT	-.229	.156	H_0
ME	-.051	.596	H_0
MJI	.108	.369	H_0

Note. $^*p < .05$

Table 18: *S_d Somers' delta values and significance levels between creativity and the project's features*

5.2 Model Evaluation

According to the statistical tests, the significant features (multidisciplinarity, project type, artistry, risk level, and research indicator) were selected and evaluated according to an ordinal classification model (Frank & Hall, 2001). This version of classification was used because creativity, the dependent outcome, was an ordinal variable. The output was a probability vector, where the i^{th} element was the probability that the input belonged to class i. With these results, a project's chances of being creative could be estimated based on its characteristics.

The following discusses the classification process and its results.

5.2.1 Ordinal Classification

First, due to the relatively low sample size (75 samples), the model is an initiative proposal for estimating a project's level of creativity, and further research on an extensive data set is needed.

According to the suggested ordinal classification described above, four binary classifiers were evaluated. Of the 75 projects, 65 were randomly selected and their vectors used as input to each classifier. The models transformed the problem from a five-class ordinal problem to four binary-class problems.

This model was then tested with the remaining ten projects. The classifier estimated a relative creativity level in most cases. There were instances in which the classifier showed no distinct choice between two levels of creativity, such as projects with a creativity level of three. In some other instances, the classifier did not decide between the creativity levels but returned a probability priority to the relevant creativity it tried to classify. There were instances where the classifier failed to estimate vectors, and as can be seen, these failures occurred when the creativity level was one (Table 19).

Actual creativity level	Estimated creativity probability vector
1	(.64, .36, .0, .0, .0)
5	(.0, .0, .0, .0, 1.)
1	(.39, .61, .0, .0, .0)
4	(.0, .0, .39, .61, .0)
2	(.0, .68, .32, .0, .0)
3	(.0, .5, .5, .0, .0)
5	(.0, .0, .0, .0, 1.)
4	(.0, .0, .5, .5, .0)
1	(.23, .71, .06, .0, .0)
2	(.21, .61, .18, .0, .0)

Table 19: *Estimated creativity probability vector compared to the Actual creativity level*

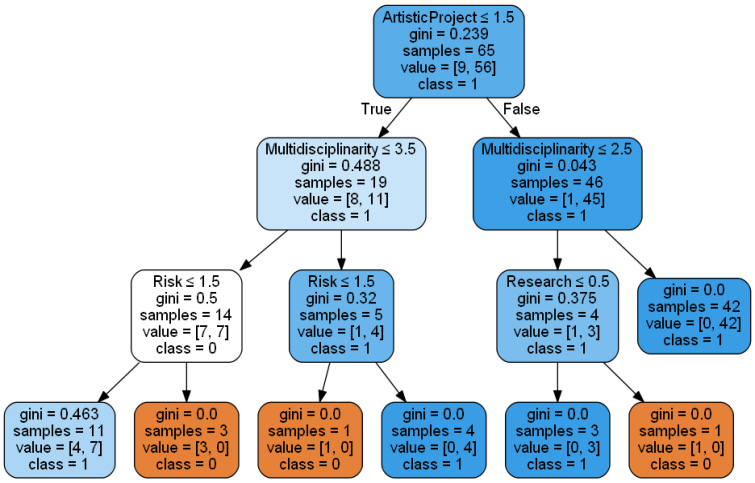
The four binary classifiers are decision trees of depth three, and each classifier contributes its decision to the probability output vector (Figure 18). Each classifier's influential characteristics were analyzed on a scale of 1 to 5, where 1 was the most noteworthy feature. It can be assumed that multidisciplinarity and artistry levels were the most homogeneous features in estimating creativity (Table 18). Generally, it can be concluded that multidisciplinarity and artistry had a substantial effect on the creativity level (RQ4).

Classifier	M	R	A	RE	CMPT
Creativity = 1	3	4	**1**	2	5
Creativity = 2	**1**	5	2	4	3
Creativity = 3	**1**	4	2	5	3
Creativity = 4	**1**	5	2	3	4

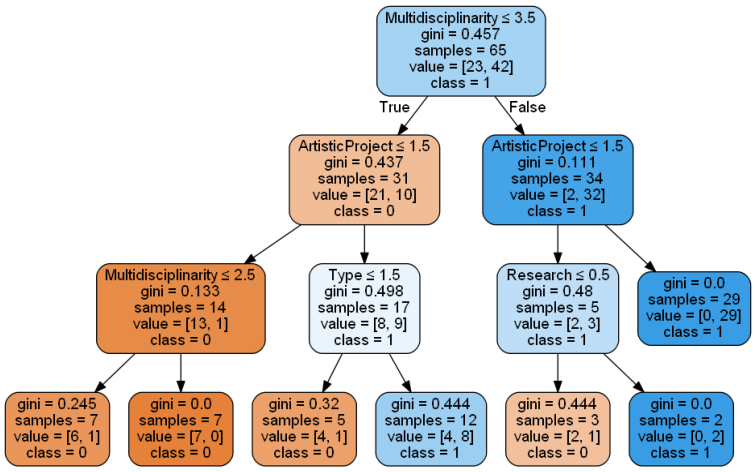
Table 20: *Features importance of the four binary classifiers.*

The first *c*lassifier (creativity level is 1) rated artistry as the most noteworthy feature, while the other classifiers rated multidisciplinarity as the most noteworthy feature.

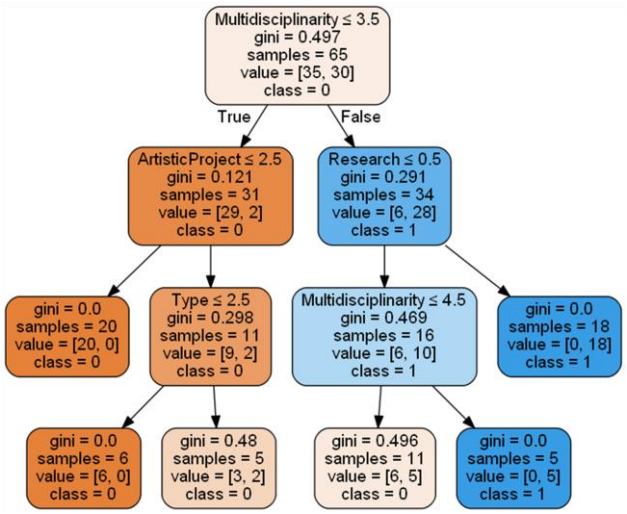
Creativity = 1



Creativity = 2



Creativity = 3



Creativity = 4

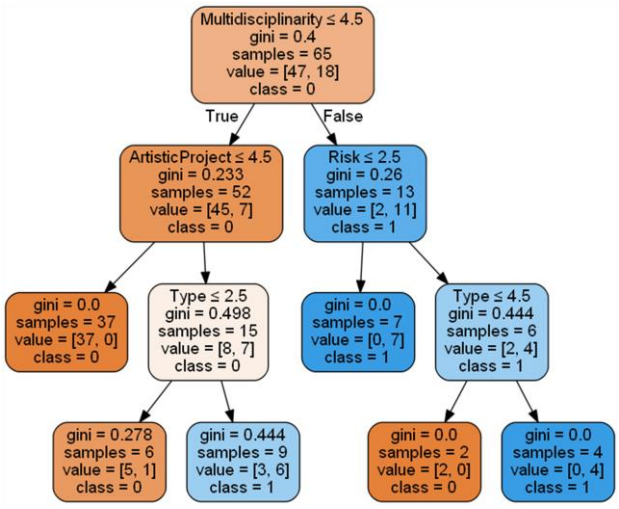


Figure 18: *Visualization of the four binary decision tree classifiers*

6 Conclusions and Future Work

This work presented an educational method to enhance creativity, multidisciplinarity, artistry, and collaboration among computer science students. The method was divided into three phases and used Muzilator, a novel plugin-based platform, as a creative educational and collaboration tool. Throughout the development process, the students were introduced to Muzilator abilities and concepts, such as separating projects into independent plugins, handling and transferring data between plugins, collaborating, reusing other developers' components, and more. Data from the projects themselves, and from the evaluated supplied by the students, we collected and analyzed, from the perspectives of both the team and the individual students.

The results indicate that multidisciplinarity, artistry, risk level, and project type were the projects' meaningful characteristics. Of the five categories of projects undertaken in the Computer Music course, sonification was the riskiest type of project that combined multiple disciplines. Such risky projects were rated as the most creative. Students who defined themselves as self-learners combined more disciplines in their projects than did others. The plugins that were developed during the study were built using components with dedicated roles, and thus gave students a deeper understanding of software architecture.

The computational analysis reinforced the hypothesis that multidisciplinarity, artistry, risk level, and project type influenced creativity regardless of whether the project was a research project or an applicative project.

Future investigations are recommended to validate the nature of the conclusions that can be drawn from this study. They could investigate creativity, multidisciplinarity, and artistry among students using the Muzilator platform and the method described here, based on three phases: HCI, computational plugin and the final project. The experiment proposed in this work can be repeated with many participants during an academic course or a hackathon to improve the ordinal classification model and the estimated results. Also, in future studies, the collaboration process should be tested.

Ethics

The studies involving human participants were reviewed and approved by the Herzylia Interdisciplinary Center's Adelson School of Entrepreneurship Ethics Committee. The participants provided their written, informed consent to participate in this study.

References

1. Adderley, K., Ashwin, C., Bradbury, P., Freeman, J., Goodlad, S., Greene, J., ... & Uren, O. (1975). Project methods in higher education (London, Society for Research into Higher Education).

2. Agnoli, S., Corazza, G. E., & Runco, M. A. (2016). Estimating creativity with a multiple-measurement approach within scientific and artistic domains. Creativity Research Journal, 28(2), 171-176.

3. Bergin, S., & Reilly, R. (2005). The influence of motivation and comfort-level on learning to program.

4. Brown, S., & Theorell, T. (2006). The social uses of background music for personal enhancement. *Music and manipulation: On the social uses and social control of music*, 126-161.

5. Campe, S., Denner, J., Green, E., & Torres, D. (2019). Pair programming in middle school: variations in interactions and behaviors. *Computer Science Education*, 1-25.

6. Charyton, C., & Snelbecker, G. E. (2007). General, artistic and scientific creativity attributes of engineering and music students. *Creativity Research Journal*, *19*(2-3), 213-225.

7. Cover, T. M., & Thomas, J. A. (2006). Elements of information theory 2nd edition (wiley series in telecommunications and signal processing).

8. D'Arcangelo, G. (2002). Creating a context for musical innovation: a NIME curriculum. In *Proceedings of the 2002 conference on New interfaces for musical expression* (pp. 1-4). National University of Singapore.

9. Ferreira, D. J. (2013). Fostering the creative development of computer science students in programming and interaction design. *Procedia Computer Science*, *18*, 1446-1455.

10. Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (1993). *How to design and evaluate research in education* (Vol. 7). New York: McGraw-Hill.

11. Frank, E., & Hall, M. (2001, September). A simple approach to ordinal classification. In *European Conference on Machine Learning* (pp. 145-156). Springer, Berlin, Heidelberg.

12. Fraser, D. A. S., Wong, A., & Wu, J. (1999). Regression analysis, nonlinear or nonnormal: Simple and accurate p values from likelihood analysis. *Journal of the American Statistical Association*, *94*(448), 1286-1294.

13. Gordon, E. E. (2000). *Studies in harmonic and rhythm improvisation readiness*. Chicago: GIA Publications.

14. Gough, H. G. (1979). A creative personality scale for the adjective check list. *Journal of Personality and Social Psychology*, 37, 1398–1405.

15.Hanif, S., Wijaya, A. F. C., & Winarno, N. (2019). Enhancing Students' Creativity through STEM Project-Based Learning. *Journal of Science Learning*, *2*(2), 50-57.

16.Helle, L., Tynjälä, P., & Olkinuora, E. (2006). Project-based learning in post-secondary education–theory, practice and rubber sling shots. *Higher education*, *51*(2), 287-314.

17.Hollander-Shabtai R., Peretz. O., A Plugin-Based Web Platform as a Collaborative Virtual Lab for Personal Creative Expression in Music, 2020.

18.Jacobson, S. K., Seavey, J. R., & Mueller, R. C. (2016). Integrated science and art education for creative climate change communication. *Ecology and Society*, *21*(3).

19.Jordà, S., & Mealla, S. (2014). A methodological framework for teaching, evaluating and informing NIME design with a focus on expressiveness and mapping. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (Vol. 30, pp. 233-238).

20.Junius, W. Y. (2015). The three factors of creativity management: Visual, number, and word creativity. *DLSU Business & Economics Review*, *25*(1), 1-1.

21.Kay, K., & Shipman, C. (2014). The confidence code. *The science and art of self*.

22.Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, *30*(1/2), 81-93.

23.Khairuddin, N. N., & Hashim, K. (2008, November). Application of Bloom's taxonomy in software engineering assessments. In *Proceedings of the 8th WSEAS International Conference on Applied Computer Science* (pp. 66-69).

24.Kiehn, M. (2003). Development of music creativity among elementary school students. *Journal of Research in Music Education*, 51, 278–288.

25.Kraskov, A., Stögbauer, H., & Grassberger, P. (2004). Estimating mutual information. *Physical review E*, *69*(6), 066138.

26.Lande, M., & Leifer, L. (2010). Difficulties student engineers face designing the future. *International Journal of Engineering Education*, *26*(2), 271.

27.Lawshe, C. H., & Harris, D. H. (1960). *Manual of instructions to accompany Purdue Creativity Test forms G and H*. Princeton, NJ: Educational Testing Services.

28.Lou, S. J., Chou, Y. C., Shih, R. C., & Chung, C. C. (2017). A study of creativity in CaC2 steamship-derived STEM project-based learning. *Eurasia Journal of Mathematics, Science and Technology Education*, *13*(6), 2387-2404.

29.Michon, R., Smith, J., Wright, M., Chafe, C., Granzow, J., & Wang, G. (2017). Mobile music, sensors, physical modeling, and digital fabrication: Articulating the augmented mobile instrument. *Applied Sciences*, *7*(12), 1-31.

30.Mihardi, S., Harahap, M. B., & Sani, R. A. (2013). The effect of project based learning model with kwl worksheet on student creative thinking process in physics problems. *Journal of Education and Practice*, *4*(25), 188-200.

31.Mokaram, A. A. K., Al-Shabatat, A. M., Fong, F. S., & Abdallah, A. A. (2011). Enhancing Creative Thinking through Designing Electronic Slides. *International Education Studies*, *4*(1), 39-43.

32.Nelson, C., Brummel, B. J., Grove, F., Jorgenson, N., Sen, S., & Gamble, R. F. (2010). Measuring Creativity in Software Development. In *ICCC* (pp. 205-214).

33.Newmann, F. M., Wehlage, G. G., and Lanborn, S. D. *Student engagement and achievement in American secondary schools*. Teachers College Press, 1992, ch. The significance and sources of student engagement, 11–39.

34.Nilsson, P. (2011). The Challenge of Innovation. In Critical Thinking and Creativity: Learning Outside the Box. In *Proceedings of the 9th International Conference of the Bilkent University Graduate School of Education (Turkey), Ankara* (pp. 54-62).

35.Ogle, D. M. (1986). KWL: A teaching model that develops active reading of expository text. *The reading teacher*, *39*(6), 564-570.

36.Rauth, I., Köppen, E., Jobst, B., & Meinel, C. (2010). Design thinking: An educational model towards creative confidence. In *DS 66-2: Proceedings of the 1st international conference on design creativity (ICDC 2010)*.

37.Rich, L., Perry, H., & Guzdial, M. (2004). A CS1 course designed to address interests of women. *ACM SIGCSE Bulletin*, *36*(1), 190-194.

38.Romeike, R. (2007). Three drivers for creativity in computer science education. *Proc of Informatics, Mathematics and ICT: a'golden triangle'. Boston, USA.*

39.Rosen, D., Schmidt, E. M., & Kim, Y. E. (2013, June). Utilizing music technology as a model for creativity development in K-12 education. In *Proceedings of the 9th ACM Conference on Creativity & Cognition* (pp. 341-344).

40.Runco, M. A., & Jaeger, G. J. (2012). The standard definition of creativity. *Creativity research journal*, *24*(1), 92-96.

41.Smith, E. R., Seger, C. R., & Mackie, D. M. (2007). Can emotions be truly group level? Evidence regarding four conceptual criteria. *Journal of personality and social psychology*, *93*(3), 431.

42. Snelbecker, G. E., McConologue, T., & Feldman, J. M. (2001). *Cognitive risk tolerance survey*. Unpublished manuscript.

43. Somers, R. H. (1962). A new asymmetric measure of association for ordinal variables. *American sociological review*, 799-811.

44. Walker, D. A. (2003). JMASM9: converting Kendall's tau for correlational or meta-analytic analyses. *Journal of Modern Applied Statistical Methods*, *2*(2), 26.

45. Wallach, M. A., & Kogan, N. (1965). A new look at the creativity-intelligence distinction 1. *Journal of personality*, *33*(3), 348-369.

46. Zhou, C., Dalsgaard, N. J., Kolmos, A., & Xiangyun, D. (2009, December). Group creativity development in engineering students in a problem and project based learning environment. In *Proceedings of the 2nd International Research Symposium on PBL* (Vol. 34, p. 18).

Appendix

1 Muzilator platform API

Types

View

export type **View** = ‘primary’ | ‘secondary’

Endpoint

export interface **Endpoint** {

libraryName: string

channelName: string

}

Library Platform

export interface **LibraryPlatform** {

createChannel(channelName**:** string)**:** Promise<MessagePort>

setSessionTerminationListener(onSessionTermination**:** () =>

romise<void>)**:** Promise<void>

}

Application Platform

export interface **AppPlatform** extends **LibraryPlatform** {

loadLibrary(pluginId**:** string, libraryName**:** string, view**?:** View)**:** Promise<void>

connectChannels(source**:** Endpoint, target**:** Endpoint)**:** Promise<void>

disconnectChannels(source**:** Endpoint, target**:** Endpoint)**:** Promise<void>

}

Functions

The platform SDK exposes two functions: **initializeApplication** and **initializePlugin**. Use **initializeApplication** when you are developing an application and **initializeLibrary** when you are developing a library.

Platform functions

---

export const **initializeApplication =** () => Promise<AppPlatform>
Call this function in your application as early as possible.

Initialize Library

export const **initializeLibrary =** () => Promise< LibraryPlatform >

Call this function in your application as early as possible.

Library and Application functions

Create Channel

createChannel(**channelName**: string): Promise**<**MessagePort**>**
Create a channel with a given name and return a Port through which you can send and receive messages.

Set Session Termination Listener

setSessionTerminationListener(onSessionTermination: () => Promise**<void>**)
: Promise**<void>**
Registers a callback function which will be called before the session is terminated. The platform will terminate the session only after the callback function returns.

Application functions

loadLibrary(pluginId: string, libraryName: string, view**?:** View):Promise<void>

- The pluginId is the ID of the plugin in the platform.
- The libraryName is used to refer to this library instance when connecting channels.
- The optimal view is used to open the library UI in either the primary or the secondary view.

Connect Channel

connectChannels(source: Endpoint, target: Endpoint): Promise**<void>**

Connects a source channel to a target channel. When calling the **connectChannel** function from an application, you can use the helper Self function to denote a channel of the application itself

Disconnect Channel

disconnectChannels(source: Endpoint, target: Endpoint): Promise**<void>**

2 Controller API Example

Description: Touch screen controller.

id: bubbles-vanilla-example

API:

- **Output**

  note-on: sent when a note is played

  channel-name: midi

  message-params: type: ‘note-on’, pitch: int[0, …, 127],

  velocity: int[0, …, 127]

- **Output**

  note-off: sent when a note is released

  channel-name: midi

  message-params: type: ‘note-off’, pitch: int[0, …, 127],

  velocity: int[0, …, 127]

3 Audio Player API Example

Description: Gets a **public** URL of an audio file and plays it. Audio Player first loads the file, stores it locally, and plays the file. A loading message can be sent only after all other plugins are loaded. **It is recommended** to load when the user performs the first event (click) of the session.

id: audio-player

API:

- **Input**

  load: receives ID and public URL of an audio file and loads the file with a given ID.

  channel-name: audio

  message-params: command: ‘load’, clipId: string, fileUrl: string

- **Input**

  play: receives ID and plays the file.

  channel-name: audio

  message-params: command: ‘play’, clipId: string

- **Input**

  stop: receives ID and stops playing the file.

  channel-name: audio

  message-params: command: ‘stop’, clipId: string

- **Output**

  loaded: send ‘loaded’ message when the file is successfully loaded.

  channel-name: audio

  message-params: command: ‘loaded’, clipId: string

- **Output**

  load-failed: send ‘load-failed’ message when an error occurs upon file load.

  channel-name: audio

  message-params: command: ‘load-failed’, clipId: string

4 StateMachine API

```
export interface Action {
   type: string;
}
export declare type Matcher<S extends string, D, A extends Action> = (state: S,
data: D, action: A) => boolean;
export declare type DataHandler<S extends string, D, A extends Action> =
(data: D, state: S, action: A) => D;
export declare type StateHandler<S extends string, D, A extends Action> =
(state: S, data: D, action: A) => Transition<S>;
export interface ConditionalDataHandler<S extends string, D, A extends
Action> {
   matcher: Matcher<S, D, A>;
   handler: DataHandler<S, D, A>;
}
export interface ConditionalStateHandler<S extends string, D, A extends
Action> {
   matcher: Matcher<S, D, A>;
   handler: StateHandler<S, D, A>;
}
export interface Stay {
   type: 'stay';
}
export interface Move<S extends string> {
   type: 'move';
   state: S;
}


export declare type Transition<S extends string> = Stay | Move<S>;
export declare const Move: <S extends string>(state: S) => Transition<S>;
export declare const Stay: Stay;
```

```
export declare type StateMatcher<S extends string> = (states: S) => boolean;
export declare type ActionMatcher<A extends Action> = (action: A) => boolean;
export declare type DataMatcher<D> = (data: D) => boolean;
export declare type Listener<S extends string, D> = (state: S, data: D) => void;
export declare type Updater<D> = (data: D) => D;
export interface StateMachine<S extends string, D, A extends Action> {
    currentState(): S;
    currentData(): D;
    process(action: A): void;
    process(actionType: A['type']): void;
    addEntryListener(state: S, listener: Listener<S, D>): void;
    addTransitionListener(from: S, to: S, listener: Listener<S, D>): void;
    addLeaveListener(state: S, listener: Listener<S, D>): void;
    addEntryUpdater(state: S, updater: Updater<D>): void;
    addTransitionUpdater(from: S, to: S, updater: Updater<D>): void;
    addLeaveUpdater(state: S, updater: Updater<D>): void;
}
export declare const createStateMachine: <S extends string, D, A extends Action>(initialState: S, initialData: D, dataHandler: ConditionalDataHandler<S, D, A>, stateHandler: ConditionalStateHandler<S, D, A>) => StateMachine<S, D, A>;
```

5 Computer Music Final Project: Presentation Requirements

1. Project name, team members' names, and one-liner.

2. Problem definition. The project's value is supposed to bring (for applicative projects) or an academic justification (for research projects). At the end of this phase, the participants should be able to describe what is the added value of their product.

3. The solution/algorithm developed in the project. For example, software, mobile, web, controller, interactive music instrument, app, music piece, sonification, etc.

4. A full description of the solution.

5. Advantages and innovation of the proposed solution/algorithm.

6. Musical and technical methods/paradigms that are implemented in their product.

7. User Interface schemes (if needed).

8. Development process plan in Agile methodology.

9. Bibliography.

במסגרת המחקר הייתי שותף בצוות הפיתוח הפלטפורמה. תכננתי ופיתחתי סט תוספים כללים, וספרייה עבור מכונת המצבים לאפליקציות אינטראקטיביות, התומכים בשיטה הלימודית לשימוש הסטודנטים, ולביצוע הניסוי וכן שיטות לתכנון הפרוייקטים, לשיתוף ולהצגתם, אשר מומשו על ידי הסטודנטים. הם פיתחו רעיונות ואבות טיפוס (POC) לפרויקטים מחקריים או אפליקטיביים חדשניים בתחום הטכנולוגיות למוסיקה. הם עבדו בצוותים, במתודולוגיה Agile לאורך שלושה ספרינטים.

חילקנו את הפרויקטים לחמש קטגוריות עיקריות והערכנו את הפרוייקטים מבחינת רמת סיכון, יצירתיות, רב תחומיות, אינטראקציה, אומנות ועיצוב יצירתי. בחנו את התוצרים מ-3 אספקטים: התלמיד כאינדיבידואל, הצוות והפרויקט. ניתחנו את הערכת הפרויקטים, תוך השוואה מול הערכה עצמית של הסטודנטים בנושא יצירתיות, רב תחומיות, למידה עצמית, רקע מוזיקלי ורקע מקצועי וכן מול דיווח סובייקטיבי של הסטודנטים על פרויקט הגמר, הצוות ותוצרי הלמידה. הבחנו בין פרויקטים מחקריים תיאורטיים לבין פרויקטים יישומיים או יזמיים. ביצענו השוואה בין פרויקטים שהשתמשו בפלטפורמת לבין פרויקטים עצמאיים שלא השתמשו בפלטפורמה. לבסוף, בחנו את גודל הצוות ואת תמהיל הגברים והנשים בפרויקט.

תוצאות הניתוח מראות כי פרויקטים בסיכון גבוה היו יצירתיים ואמנותיים יותר מאשר פרויקטים בסיכון נמוך. סטודנטים שהעריכו את עצמם אוטודידקטים שילבו יותר דיסציפלינות לעומת סטודנטים אחרים. נשים בחרו לפתח יישומים אינטראקטיביים, ואילו גברים נטו לבחור בפרויקטים מחקרים תיאורטיים יותר (לא אינטראקטיביים). צוותים מעורבים (עם גברים ונשים כאחד) פיתחו את הפרויקטים היצירתיים ביותר, האמנותיים והרב תחומיים, ואילו סולנים (צוותים עם חבר אחד בלבד) הציגו את הדירוג הנמוך ביותר בכל הפרמטרים. פרויקטים שפותחו עם Muzilator היו פרוייקטים יותר יצירתיים, רב-תחומיים, אמנותיים ובעלי רמת עיצוב יצירתי גבוהה יותר.

תקציר

"יצירתיות חיונית לסטודנטים למדעי המחשב, ומדעי המחשב מעודד יצירתיות" (Romeike, 2007). יכולת עיצוב יצירתי, רב תחומי, יכולת שיתוף ואומנות יכולים לשפר את יכולות מדעני המחשב ומהנדסי תוכנה בפתרון בעיות, חדשנות, עיצוב תוכנה ופיתוח. טכנולוגיה למוסיקה היא תחום שיכול לשמש ככלי מצוין לפיתוח יצירתי. זה מרתק ו"זה יכול לעזור בפיתוח חשיבה יצירתית בסביבה אקדמית, ולאפשר לסטודנטים להשיג יכולת עצמית ביכולות היצירתיות שלהם" (Rosen, Schmidt & Kim, 2013). בעת פיתוח פרויקטים של טכנולוגיות מוסיקה, התלמידים יכולים לשלב בקלות אמנות, מדע וטכנולוגיה. בין אם מדובר במחקר תיאורטי ובין אם מדובר בפרויקט יישומי, באופן טבעי נדרש מיזוג בין פרדיגמות אמנותיות וחישוביות ושילוב של כמה תחומים כמו מוסיקה, אמנות, סאונד, חינוך, משחק, ספורט, מדעי המוח ופסיכולוגיה. תוך כדי יצירה ושיתוף פעולה, חינוך טכנולוגי למוסיקה מסייע לתלמידים להביע את אישיותם, תשוקתם למוזיקה ורגשות חיוביים אחרים (Brown & Theorell, 2006). השילוב בין לימודים אקדמיים, רגש חיובי והתלהבות הוא חלק בלתי נפרד מהגברת היצירתיות והחדשנות.

בעבודה זו פיתחנו ובחנו את יעילותה של מתודה לחינוך יצירתי המשתמשת בפלטפורמה Muzilator ככלי לחינוך יצירתי, כאשר הסטודנטים פיתחו פרויקטים טכנולוגיים-מוזיקליים. פלטפורמת Muzilator הינה פלטפורמת אינטרנט מבוססת תוספים המאפשרת למפתחים לפצל את הפרויקט שלהם לאוסף תוספים עצמאיים המתקשרים ביניהם על ידי שליחת הודעות בערוצי תקשורת המוצעים דרך הפלטפורמה. Muzilator מאפשרת העלאה ושיתוף תוספים עם קהילת המפתחים המשתמשים בפלטפורמה. המתודולוגיה נועדה לפתח כישורים יצירתיים, להקנות יכולות לשילוב שיטות חישוביות ממדעי המחשב עם פרדיגמות מעולם המוזיקה והאמנות ודיסציפלינות נוספות, לשפר כישורי שיתוף פעולה, עבודת צוות ויכולות תיכון תוכנה. המחקר שלנו מבוסס על 75 פרויקטים שיושמו על ידי 183 סטודנטים למדעי המחשב שהשתתפו בקורס "מוסיקה ממוחשבת" בשנים 2016-2020.

עבודה זו בוצעה בהדרכתם של דר׳ רויטל הולנדר מביי״ס אדלסון ליזמות, המרכז הבינתחומי, הרצליה ופרופ׳ דוד הראל מהמחלקה למדעי המחשב ומתמטיקה שימושית, מכון ויצמן, רחובות.

המרכז הבינתחומי בהרצליה

בית-ספר אפי ארזי למדעי המחשב

התכנית לתואר שני (M.Sc.) - מסלול מחקרי

טכנולוגיה למוסיקה ופלטפורמה מבוססת תוספים ככלי לחינוך יצירתי והעצמת רב תחומיות, אומנות ושיתופיות.

מאת

אור פרץ

עבודת תזה המוגשת כחלק מהדרישות לשם קבלת תואר מוסמך M.Sc.

במסלול המחקרי בבית ספר אפי ארזי למדעי המחשב, המרכז הבינתחומי הרצליה

דצמבר 2020