Journal of Education and Practice
ISSN 2222-1735 (Paper) ISSN 2222-288X (Online)
Vol.12, No.11, 2021

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

Or Peretz[1*] Revital Hollander[2] David Harel[3]
1.The Interdisciplinary Center, Herzliya, Israel
2.The Adelson School of Entrepreneurship, the Interdisciplinary Center, Herzliya, Israel
3.The Department of Computer Science and Applied Mathematics, the Weizmann Institute of Science, Rehovot, Israel
* E-mail of the corresponding author: or.perets@post.idc.ac.il

Abstract
Music technology is known to have the ability to enhance creativity and creative development among students. 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. When students develop a music technology project, they use their software design skills to build and combine different artistic and computational components. Here we present a creative education method for computer science and software engineering students, it uses Muzilator, a plugin-based web platform that enables developers to develop a project as a set of independent web applications (plugins). Students can share their plugins with others or use plugins developed by others. We examined 75 projects of teams of computer science students who participated in a Computer Music course. We studied the characteristics of these projects and Muzilator's effectiveness as a creative education and collaboration tool. Some of our results show that Muzilator-based projects received higher creativity and multidisciplinarity ratings than did other projects, and that high-risk projects were more creative and artistic than low-risk ones. We also found a gender-dependency: women tended more than men to develop interactive applications, while men tended to choose more theoretic (algorithmic), non-interactive projects.
Keywords: educational method, creativity, music education, software design, multidisciplinarity.
DOI: 10.7176/JEP/12-11-01
Publication date: April 30th 2021

1. Introduction

Traditional computer science education combines mathematical knowledge, theoretical computer science, computational thinking, computer programming, and software engineering (Hu, 2011; Pirker, Riffnaller-Schiefer & Gütl, 2014). While some of 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 a skill for communication or synchronization between individuals and teams.
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.

Music technology is a field that can offer an excellent tool for creative development (Rosen, Schmidt & Kim, 2013). Newmann, Wehlage & Lanborn (1992) described a high engagement level among students who studied and developed musical projects.

This study investigated the characteristics of music technology projects and the key factors needed to improve computer science students' skills. We based our study on 75 Computer Music projects that were developed in teams. The projects were divided into five categories and were evaluated for several criteria: creativity, artistry, interactivity, multidisciplinarity, and risk. We compared projects developed using the Muzilator platform with projects that did not, and evaluated its effectiveness as an educational tool.

The paper 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 platform experiment is described. Finally, Section 5 contains the main conclusions and suggestions for future directions.

2. Background and Related Work

There are at least two perspectives on creativity in computer science (CS), one focuses on students creativity, and the other on creativity on 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, such as participation in an open-source community that can facilitate tracing and enable students to expand and improve their programming skills by exposing them to different concepts.

When focusing on creativity, the importance of a multidisciplinary viewpoint is examined by Charyton and Snelbecker (2007). They conducted a study designed to understand the differences or similarities between these domains among music and engineering students. These analyses have indicated that engineers and musicians are approximately equal in terms of artistic creativity. Nilsson (2011) suggested a methodology to measure innovation and creative design of an idea: the taxonomy of creative design. He presented two scales, novelty in form and novelty in content, measured by five hierarchical levels of creative design: imitation, variation, combination transformation, and original creation. 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).

Creative educational methods are relevant and necessary for encouraging creativity among students. After gathering observations from interviews, Rauth, Köppen, Jobst & Meinel (2010) presented an educational method to enhance the creative confidence level. The interviews contained various questions regarding creative education design (Lande & Leifer, 2010). Their results show that participants initiated creative challenges independently, without any relative background. Nelson et al. (2010) proposed the SEREBRO (Software Engineering REwards for BRainstorming Online) system for modeling creativity within a computer program to explore creativity assessment. The SEREBRO platform assigned reward points to each individual or team for their creative input. Students worked in teams, and the projects were rated for originality, elaboration, and overall creativity. The results show that teams with higher quality ratings received high creativity ratings.

Project-based learning (PBL) involves solving a given problem in educational activities, resulting in a complete product (Adderley et al, 1975). Zhou et al. (2009) examined the influences of PBL on students' learning and learning by collaborative behavior. They tested engineering students studying for master's degrees at Aalborg University in Denmark. 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, and that PBL can build wide knowledge for students. Mihardi et al. (2013) used the Know, Want, Learn (KWL) worksheet, a framework that helps students organize knowledge in a lesson. The participants were asked to implement and solve a factory design problem and complete questionnaires. The results indicated that students' creative thinking using PBL was higher than when using other methods.

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). Jorda & Mealla (2014) proposed a methodology for teaching NIME design applied in a master's degree course at Pompeo Fabra University in Spain. The analysis of the design processes leading to the evaluated outcome demonstrated traceable progress in the students' achievements. 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). Musicality was considered as a driving force in the design process, and each student needed to adopt some sense of musical style. As a result, the projects were explicitly expected to break from the traditional paradigm of musical instruments and present new models of musical expression. Michon et al. (2017) presented a framework enabling speedy design and prototyping of passive mobile device augmentations. This framework was suitable for developers with a background in music and sound design. 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. 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 educational method, the categorization of music-tech projects, and analysis results of project evaluation of 75 music-tech projects developed by 183 students.

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 computer music project type align with the project's quality and students' learning outcomes?

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

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. Here are the categories:

1. *Music experience* - This refers to a project or an application with specific music functionality and logic (i.e., playing or learning), which does not involve creation or generation. This category includes music games, educational applications, players, and more.
2. *Creative expression* - This refers to an interactive application that displays a musical interface for music interaction and creation. A project in this category involves artistic considerations on the interaction with the user. This category includes NIMEs, controllers, synthesizers, and more.
3. *Analysis and Generation* - This refers to an algorithm that analyzes or generates music pieces. This category includes Music Information Retrieval (MIR) algorithms for feature extraction, music analysis or genre classification, recommendation systems such as playlist generators, or a visualization algorithm. A project in this category is not interactive.
4. *Smart Interaction* - This refers to an application that combines user interaction and creative expression with analysis or generation. This category includes applications that analyze users' interaction or music improvised by the user generates a response that is played to the user.
5. *Sonification* - This refers to a data-driven project that uses non-speech audio to convey information or to perceptualize data. The data is transformed from another domain prior to being generated, which differs from generating music using compositional rules or machine learning. Sonification can be used for scientific, experimental, or artistic purposes.

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, musical composition, improvisation, etc. An *artistic application* is an application where the students used or combined musical elements and artistic aspects in their project.
2. An *interactive application* is an application that allows users to enter data or commands.
3. An *artistic-interactive* application is both an artistic and interactive application.
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.
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 method (Nillson, 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* that refers to the novelty in the project's source code (how different the architecture, according to the initial project given in class), and the *novelty in content* that 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 are needed to advance from one phase of the development process to another. The project outcomes of such project are uncertain, as is whether the students would succeed in achieving their goals and provide a solution for the problem they seek to solve.

3.3 Method

3.3.1 Educational Method

The course, "Computer Music", 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. The students then learned about musical elements in theory and practiced them using the SuperCollider programming language. 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 choose one. The next task was to build a presentation that described the project.

The development process was divided into 3 sprint and developed in an Agile manner. After each sprint, a working part of the project was delivered and tested with the potential users. Audio output examples or videos that demonstrated the user using the prototype were submitted. The project presentation was updated after each sprint and was used in the final presentation.

During the development process, each team had two meetings with the course teachers. The first took place when the team had devised three ideas for their project. The team members presented themselves, their interests, and the three ideas and possible solutions. Each idea's risk level was discussed. The second meeting took place after the first sprint.

The students presented their implementation for the projects' main idea and planned the next sprint. In addition, the teams had to learn the problem domain in a learning-by-doing or PBL process. Each team reviewed relevant papers and presented on paper them in class. A demonstration day was held at the end of the semester, and the students presented their final presentations and projects.

3.3.2 Project Evaluation Method

The course teachers and the students ranked the projects done between 2016 and 2020. In 2020, data was collected from the students at the beginning and the end of the semester. At the beginning of the semester, the students were asked to rank their own creativity, multidisciplinarity, and self-learning (autodidactic) abilities and to tell 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 criteria from the data collected in the pre-project questionnaire (Table 1). The projects were ranked according to the criteria in Section 3.2, where 1 is the lowest level and 5 is the highest level.

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

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?"

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: Have an 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).
- 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.

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.

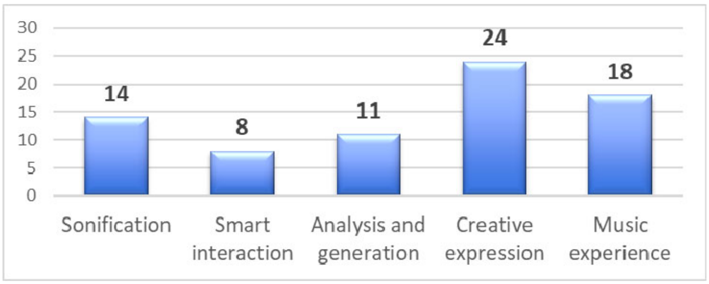


Figure 1: *The distribution of students' projects by category*

Here is a summary of the main results.

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, and the second involved applying Nilsson's taxonomy for creative design (Nilsson, 2011). The comparison (see Table 2) 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 2: *Creativity average and standard deviation of projects by category*

Here are the conclusions on the individual level:

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

3.4.2 Project

- High-risk projects were more artistic and creative than other projects, and vice versa (RQ1).
- 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). The same results achieved in multidisciplinarity rates.
- 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).

Drawing on the previous finding, creativity, multidisciplinarity and risk levels were compared according to project type (Table 4). 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 4: *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), who took the class 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.
The main question we asked was: does the Muzilator platform enhance multidisciplinarity, creativity, collaboration, and artistry levels?

4.1 About Muzilator

Muzilator is a plugin-based web platform for interactive applications, intended for musicians, novices, developers, and researchers (Hollander-Shabtai & Peretz, 2021). 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.

4.2 About Muzilator

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

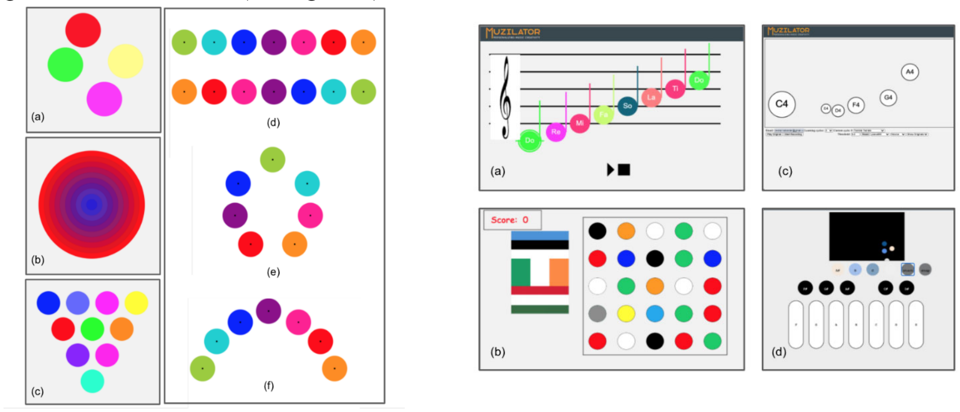
1. *A platform for everyone:* The platform architecture enables students to write plugins in Java Script, the most common programming language these days, and easily add them to the platform. This can be done by anyone, regardless of his/her level of programming skills or their level, if any, of musical or artistic background. It enables the students to focus on the innovative and musical aspects of their project.
2. *Development of independent shareable plugins:* The students develop their ability to focus on a creative ideas and implementation of a specific entity as a plugin, to write their plugin, they optimized a plugin functionality and uniqueness. Knowing that a plugin can be shared with the community requires the student to take it into consideration and thus improves its functionality, quality and interface.
3. *Use of existing plugins:* The students can use independent entities (plugins) that already exist in the platform. This can either help the student to focus on his/her plugins functionality and avoid re-inventing the wheel, to practice use and integration with another plugin and to collaborate with other developers.
4. *Versatility:* Since a plugin can have any functionality in any domain, the students are able to easily combine art, technology, and science across different disciplines. The students can choose from a variety of artistic and computational projects to develop: artistic HCIs, sound engines, players, recorders, profilers, applications, online/offline algorithms for analysis, music generation, prediction, profiling, and more. Choosing a topic which is meets his/her skills and interests and increases motivation and engagement.
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. *Software design and software engineering:* The platform includes a software development kit (SDK) to build and integrate plugins quickly and easily. In addition, the platform provides a state machine interface that conveniently presents a state machine's concept and its use for interactive applications.
8. *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.
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 easily perform analysis and achieve optimization for development processes and avoid parsing and data preparation for analysis.

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: Developing an HCI: Designing an Interactive 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:

1. **Assignment 1 -** Developing 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). In this experiment, Bubbles, a simple and basic controller that displays random circles with random colors and notes was used. 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 had to combine programming and artistic abilities to design the HCI's features (see figure 2).



2.1 - Visual transformation *2.2 – Music composition*

Figure 2: *Controllers developed by the participants in Assignment 1: Visual Transformation and Music Composition. Figure 2.1a is the given Bubble controller. Figures 2.1b and 2.1c demonstrate simple musical controllers with a specific design for specific musical elements, and Figures 2.1d, 2.1e, and 2.1f demonstrate three different configurations for an eye-tracking musical controller. Figure 2.2a shows a simple app for music composition, where the user composes a melody and the app continues the composition. Figure 2.2b 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 2.2c shows an application who learned how the user perceives a melody in a two-dimensional space. Figure 2.2d 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.*

2. **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. 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 as presents in figure 3.

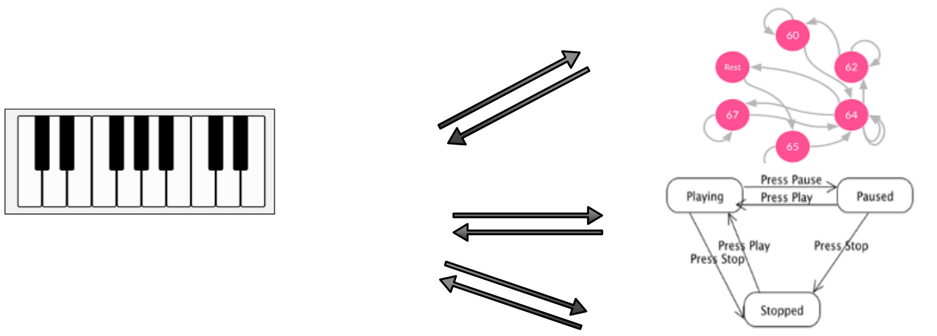


Figure 3: *An example of three applications that used the same controller with different analyzers. The first application used the Markov chain analyzer, the second used a genetic algorithm analyzer, and the third used the state machine analyzer.*

3. **Assignment 3 – The Final Project:** The development process of the final project 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.

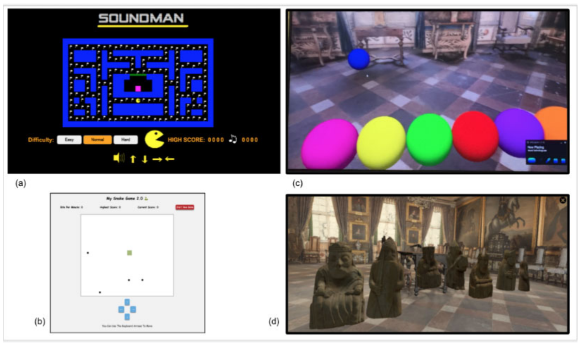


Figure 4: *Example Music experience projects developed by the participants. Figure 4a shows Soundman, a musical Pacman game where the user navigated with sound. Figure 4b shows a musical snake game, and Figure 4c shows the Bubbles controller converted to a 3D VR controller with additional abilities, such as drag and drop, that enable the user to organize the elements in a 3D space.*

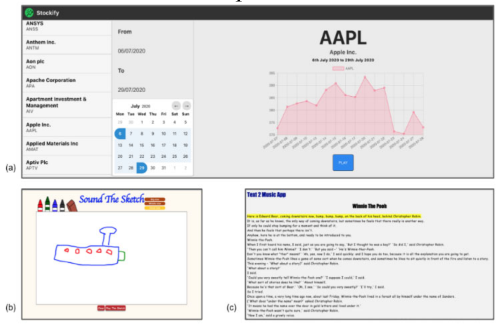


Figure 5: *Example of Sonification projects developed by the participants. Figure 5a shows a stocks graph sonification. Figure 5b shows a musical painting app in which the user draws a painting, and the application plays the sonified painting. Figure 5c shows an application that takes short stories, and, using sentiment analysis and sonification, creates a playback for the reader while the user reads the story.*

In addition to the project's code, the participants were asked to submit an application program interface (Appendix 1), and a channel-diagram (Figure 6), that presents the communication channels between the plugins.

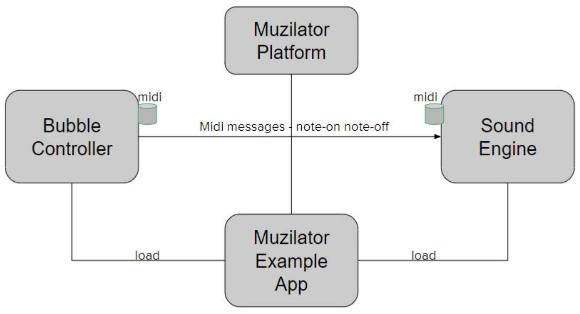


Figure 6: *Example of a channel-diagram. The controller and the sound engine communicate and send messages on the midi channel.*

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. Of all the 47 participants, 32 developed totals of 11 projects using the Muzilator, and 15 developed totals of 8 independent projects. 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

4.4.1.1 Pre-Experiment Analysis

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. 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.

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

There are two possible explanations for the difference. 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.

4.4.1.2 Post-Experiment Analysis

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. Similarly, the participants' self-esteem with respect to their creativity and the projects' creativity ratings were compared (see table 6).

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

Table 6: *The difference between pre-project and post-project questionnaires in creativity*

4.4.2 Team / Project

Muzilator projects received higher creativity and multidisciplinarity ratings than did independent projects (RQ).

	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 7: *Muzilator projects' ranking compared to that of independent projects*

4.4.3 Gender Differences

4.4.3.1 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 themselves 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.

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

4.4.3.2 Post-Experiment Analysis

To more fully explore the difference between women and men, various parameters were examined with respect to gender. Figure 7 present the examination of project type.

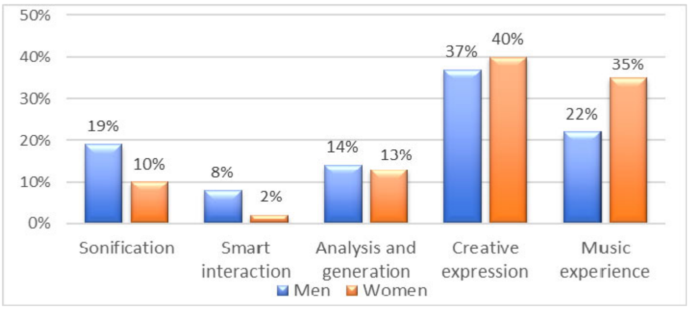


Figure 7: *Projects' category distribution by gender*

The results show that teams with women only developed more artistic projects and interactive projects than did other teams (Table 9).

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

Also, women with professional backgrounds developed more artistic projects than did other women (ρ=0.56). However, there was no correlation between men's professional background and their projects' artistic level (ρ=0.06). The risk level was examined, and the results show that men developed projects with a higher level of risk than did women.

5. Conclusions and Future Work

This work presented an educational method to enhance creativity, multidisciplinarity, artistry, and collaboration among computer science students. The process was divided into three phases: HCI, computational plugin, and the final project. In 2020 we added the Muzilator platform, as a creative educational and collaboration tool, to our method.

The results show that of the five categories of projects presented here and 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. Teams with women only developed more artistic projects than did other teams. The plugins that were developed during the study were built using components with dedicated roles or shared with other, which gave students a deeper understanding of software architecture.

Future investigations could investigate creativity, multidisciplinarity, collaboration, and artistry among students using the Muzilator platform and the method described in section 4. In phase two, the development of a computational plugin, we examined the effectiveness of using HCI of another team, and with the help of the educational method, the collaborative process was accelerated. The experiment proposed in this work can be repeated with many participants during an academic course or a hackathon. Also, this process can be designed as a full-day workshop, where the participants aware in advance of the platform architecture and abilities. The workshop would split into three phases as the educational method and would examine collaboration and overall creativity among the participants.

The studies involving human participants were reviewed and approved by the Herzliya 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. Bergin, S., & Reilly, R. (2005). The influence of motivation and comfort-level on learning to program.
3. 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.
4. 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.
5. Ferreira, D. J. (2013). Fostering the creative development of computer science students in programming and interaction design. *Procedia Computer Science*, *18*, 1446-1455.

6. Fraenkel, J. R., Wallen, N. E., & Hyun, H. H. (1993). *How to design and evaluate research in education* (Vol. 7). New York: McGraw-Hill.
7. Gordon, E. E. (2000). *Studies in harmonic and rhythm improvisation readiness*. Chicago: GIA Publications.
8. Gough, H. G. (1979). A creative personality scale for the adjective check list. *Journal of Personality and Social Psychology*, 37, 1398–1405.
9. 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.
10. Hollander-Shabtai R., Peretz. O., A Plugin-Based Web Platform as a Collaborative Virtual Lab for Personal Creative Expression in Music, 2020.
11. Hu, C. (2011, June). Computational thinking: what it might mean and what we might do about it. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education* (pp. 223-227).
12. 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).
13. Junius, W. Y. (2015). The three factors of creativity management: Visual, number, and word creativity. *DLSU Business & Economics Review*, *25*(1), 1-1.
14. Kay, K., & Shipman, C. (2014). The confidence code. *The science and art of self.*
15. Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, *30*(1/2), 81-93.
16. Kiehn, M. (2003). Development of music creativity among elementary school students. *Journal of Research in Music Education*, 51, 278–288.
17. Lande, M., & Leifer, L. (2010). Difficulties student engineers face designing the future. *International Journal of Engineering Education*, *26*(2), 271.
18. Lawshe, C. H., & Harris, D. H. (1960). *Manual of instructions to accompany Purdue Creativity Test forms G and H*. Princeton, NJ: Educational Testing Services.
19. 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.
20. 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.
21. 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.
22. 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).
23. 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.
24. 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).
25. Ogle, D. M. (1986). KWL: A teaching model that develops active reading of expository text. *The reading teacher*, *39*(6), 564-570.
26. Pirker, J., Riffnaller-Schiefer, M., & Gütl, C. (2014, June). Motivational active learning: engaging university students in computer science education. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 297-302).
27. 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)*.
28. Romeike, R. (2007). Three drivers for creativity in computer science education. *Proc of Informatics, Mathematics and ICT: a'golden triangle'. Boston, USA.*
29. 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).
30. Runco, M. A., & Jaeger, G. J. (2012). The standard definition of creativity. *Creativity research journal*, *24*(1), 92-96.
31. 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.

32. Snelbecker, G. E., McConologue, T., & Feldman, J. M. (2001). *Cognitive risk tolerance survey*. Unpublished manuscript.
33. Somers, R. H. (1962). A new asymmetric measure of association for ordinal variables. *American sociological review*, 799-811.
34. Walker, D. A. (2003). JMASM9: converting Kendall's tau for correlational or meta-analytic analyses. *Journal of Modern Applied Statistical Methods*, *2*(2), 26.
35. Wallach, M. A., & Kogan, N. (1965). A new look at the creativity-intelligence distinction 1. *Journal of personality*, *33*(3), 348-369.
36. 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

The following is an API example of a Muzilator plugin. The API included plugin description, ID (as registered in the Muzilator) and messages API, both input and output:

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