Perceived Acceptance and Use of Scratch Software for Teaching Programming: A Scale Development Study

Serife Nur Yildiz[1]
Alev Ates-Cobanoglu[2]
Tarik Kisla[3]

1 Izmir Institute of Technology
2 Ege University
3 Ege University

DOI: 10.21585/ijcses.v4i1.59

Abstract

This paper reports the development process of a scale for Information and Communication Technology (ICT) teachers' acceptance and use of Scratch for teaching programming. For early beginners of programming, Scratch is the most popular block-based software for facilitating programming teaching (Zhang and Nouri, 2019) worldwide. Also, in Turkey, Scratch is widely used and has been chosen as a block-based coding tool for developing problem solving ability, self-efficacy, motivation, interest (Kasalak and Altun, 2018). Despite that wide use, there is a lack of scale for understanding ICT teachers' acceptance and use of Scratch for teaching programming. Theoretically, this scale development study is based on Unified Theory of Acceptance and Use of Technology (UTAUT) which is also explained in the paper. The sample of the study includes 265 ICT teachers from Turkish Ministry of National Education (MoNE) secondary schools who used Scratch software in their courses. According to exploratory and confirmatory factor analyses results, the final version for the scale includes 28 items. The Cronbach Alpha coefficient is 0.97. "Perceived Acceptance and Use of Scratch Software for Teaching Programming Scale" (PAUSS-TPS) can help the practitioners who aim at understanding the ICT teachers' level of acceptance and use of Scratch; the researchers who wish to investigate perceived contribution according to various variables and the decision-makers for deciding to use Scratch or shift to another block-based tool for teaching programming at early ages.

Keywords: scratch; teaching programming; coding instruction; computer science instruction; block-based programming

1. Introduction: Defining Computational Thinking

In today's developing and changing world, individuals need to acquire 21st century skills such as critical thinking, creative thinking, problem solving, and cooperative working in line with the diversifying social requirements. To teach 21st century skills to learners and to become a technology-producing society, programming or coding education has been on the agenda of Turkey particularly as of the 2000s. As Gülbahar and Kalelioğlu (2018) reported that particularly since 2012, computer science topics such as problem solving, and programming have been the focus of the Turkish curriculum. The literature includes findings revealing that an effective teaching programming manages to teach these skills. For example, when learners are taught programming and designing tools, their product development, problem solving and analytical thinking skills

improve (Akpınar and Altun, 2014; Çakıroğlu, Sarı and Akkan, 2011). Besides using the products provided by technology, another quality expected from the learners of the 21st century is developing projects and products. It is recommended that programming should be taught at early ages to be able to raise individuals who can develop projects and products (Yükseltürk and Altınok, 2015). Great attention is attached to teaching programming in many countries like Korea, China, India, Spain, and Canada. In Korea, for instance, amendments were made in the curricula of the information and communication course at the secondary school level in 2010 and at the high school level in 2011. Programming is included in the computer sciences course curriculum at high school level in India and Israel (Gülmez, 2009); while in Canada it is offered within the scope of the courses delivered in connection with computer engineering and computer sciences at secondary education institutions (Stephenson, 2001).

On the other hand, teaching programming involves several challenges in terms of the programming language in use and the characteristics of the target population. One of these challenges is the structure of traditional programming languages especially for new beginners (Çatlak, Tekdal and Baz, 2015; Genç and Karakuş, 2011; Gomes and Mendes, 2007). Another difficulty is that some procedures and concepts concerning programming are abstract for learners (Ersoy, Madran and Gülbahar, 2011). In this regard, teaching programming at early ages, it becomes important to use software focusing on such aims as defining the problem, doing analysis, evaluation and creativity by taking the learner away from the challenges of code-writing (Kert and Uğraş, 2009). To this end, visual programming tools like Scratch, Code.org, Alice, Small basic and Toontalk were developed to provide new beginners with a more interest-provoking and enjoyable environment.

In Turkey, some additions were made into the ICT and software course as of 2012 regarding such topics as Microsoft Office applications, desktop settings, control panel, of which many learners today have some knowledge. In 2012, MoNE included the learning domains of problem solving, programming, and developing original products into the “ICT and software course” content, which was also mentioned by Gülbahar and Kalelioğlu (2018). Revised curriculum of the Information technology and Software Course also covered the “Problem solving and Programming” unit in 6th grade program (MoNE, 2018). Concerning these learning domains, specifically Scratch program was recommended as a block-based visual programming tool for introducing programming in the formal annual lesson plans of 6th graders of the mentioned course.

Weintrop (2019) claims that block-based programming tools like Scratch provide visual cues, drag-and-drop options, and syntax error prevention to the beginners of programming and computer science. Block-based programming has two main purposes: (i) Simplifying programming syntax, (ii) Getting attention of target learners for programming to increase the number of learners who are interested in programming (Durak, Karaoğlan Yılmaz, Yılmaz and Seferoğlu, 2017; Yükseltürk and Altıok, 2016). It is considered that these facilitating characteristics of block-based programming tools have the potential to encourage children to produce their own tiny programs easier than traditional text-based programming. In another study by Oluk and Oluk (2019), block-based programming tools such as Scratch, Mblock, Alisblock, code.org, thinkerCad circuits, blockly, alice, code game were compared with features such as "Turkish language support", "platform", "pricing", "age level", "robotic coding compatibility". It was stated that the selection between programming tools should be made according to variables such as the target audience, content, educational environment considering the specified criteria. In addition, it was reported that Scratch, code.org and Blockly are used more than others. Zhang and Nouri (2019) reported that Scratch is the most popular visual block programming language as of 2018 rankings. Scratch, which is an educational and social software tool, enables children particularly over eight to develop video games and interactive stories and to share these over the Internet easily. Revealing the worldwide popularity of Scratch, Scratch website for statistics (https://scratch.mit.edu/statistics/) reported that there are almost 50 million registered users, as of January 2, 2020. Also, in Turkey, ICT teachers in Turkish secondary schools are responsible for teaching programming at introductory level and they make use of block-based programming tools like Scratch, Mblock, Code.org etc. Among these tools for early stages of programming, Scratch is widely used in Turkey as a block-based coding tool for developing problem solving ability, self-efficacy, motivation, and interest (Kasalak and Altun, 2018). Therefore, among other tools, the authors chose Scratch software in the study. Besides the popularity of Scratch, it would also be impractical to measure acceptance and use of all block-based programming tools. Consequently, the scope of present study is narrowed down to acceptance and use of Scratch software in teaching programming.

Scratch helps children to program a prototype of a video game or a story, to understand how its different parts run and to visualize an idea (Gonzalez, 2013). Developed by the Lifelong Kindergarten Group within the Massachusetts Institute of Technology (MIT), Scratch consists of three parts as Block Palette, Script Area and

Stage. Blocks include block palettes divided in groups within themselves; script area is where programming is implemented, and stage displays sprites (visual material). With block palettes, users bring codes together using the drag-and-drop method and they can test them with stage quickly. The projects designed can be shared online with open-resource codes. Scratch has three versions as 1.4, 2 and 3. Version 2 of Scratch can be used both online and offline.

Studies concerning the use of Scratch in teaching programming have varying results in the literature. Some positive results, for instance, reveal that Scratch helps delivering computational thinking skills according to a systematic review study (Zhang and Nouri, 2019); Scratch increases programming and computer achievement (Ferrer-Mico et al., 2012; Malan and Leitner, 2007; Meerbaum-Salant, Armoni and Ben-Ari, 2013); it affects thinking skills positively (Kert and Uğraş, 2009); Scratch learning is found easy (Genç and Karakuş, 2011; Tanrıkulu and Schaefer, 2011); students enjoy using Scratch (Genç and Karakuş, 2011); students come to class with greater motivation (Wilson and Moffat, 2010; Yükseltürk and Altıok, 2016), and it affects students' creativity positively (Kobsiripat, 2015). On the other hand, some studies report that Scratch has no significant effect on certain variables. For instance, it is stated that Scratch has no significant effect on programming knowledge by Wilson and Moffat (2010), on algorithm instruction by Tekerek, Altan and Akdağ (2012) and on problem solving skills by Kukul and Gökçearslan (2014). As a scale development study, Kasalak and Altun (2018) developed a perceived self-efficacy scale related to block-based programming, namely related to the Scratch tool. While there are a variety of study findings concerning different effects of Scratch on students' problem solving, programming skills and creative thinking and its effects on teaching and learning programming; it is noticeable that most of the reported results are positive and there is a gap for scale development studies in this era. In addition, the perceptions concerning its contribution to teaching programming are mostly determined through blog posts and interviews (Genç and Karakuş, 2011; Kalelioğlu and Gülbahar, 2014). Apparently, there is a lack of comprehensive studies related to Scratch and ICT teachers which points out the importance of current scale development study to measure the level of acceptance and use of Scratch software by the ICT teachers dealing with teaching introductory programming.

Ursavaş (2014) mentioned that teachers' technology acceptance has been one of the most critical factors in the integration of ICT in education. Šumak and Šorgo (2016) claim that the adoption of any technology or teaching practice of a teacher is affected by many factors if the adoption is voluntary. The case in Turkey is that including the use of block-based programming tools, more commonly Scratch, in the curriculum of "ICT and Software" course required teachers to use such tools in their lessons. Clearly, teachers have a key role as change agents in education (Van Der Heijden, Geldens, Beijaard and Popeijus, 2015), and they are a key factor in technology use as well (Mueller, Wood, Willoughby, Ross and Specht, 2008). Effective and efficient use of Scratch by the ICT teachers affects the quality of programming lessons. To do that, we need to know the level of ICT teachers' acceptance and use of Scratch. The results of this scale can lead researchers and decision makers to make judgements about different aspects and effectiveness of Scratch. It is remarkable that there are an inadequate number of studies dealing with the acceptance of Scratch within the scope of the TAM or UTAUT model. Therefore, teachers' acceptance and use of any new technology - in this case, the Scratch software- needs further attention for investigating the factors of the effectiveness of introducing programming via Scratch.

Davis (1989) suggests that user acceptance affects effective implementation of an information technology or information system. As a unifying theory of technology acceptance related models, UTAUT can explain the factors that have the highest effect on individuals' technology acceptance behaviours. UTAUT model is helpful for predicting system usage and making technology-adoption- and technology-usage-related decisions (Chao, 2019). It has four essential determining components as performance expectancy, effort expectancy, social influence, facilitating conditions, behavioural intention to use the system, and usage behaviour and four moderators as gender, age, experience, and willingness to use technology (Venkatesh, Morris, Davis, and Davis, 2003). And this study utilized the UTAUT model as a theoretical framework for measuring the acceptance of ICT teachers for the scale development process.

1.1 Research Questions

In this respect, the present study aims to develop an instrument that can measure acceptance and use of Scratch by ICT teachers' who practice Scratch in "ICT and Software" course. As a result, it is targeted to provide an instrument with which practitioners, researchers and decision makers working on ICT teaching curriculum development and beginner level coding instruction. In this scale development study for measuring ICT teachers' perceived acceptance and use of Scratch for teaching programming based on UTAUT, the research questions are as follows:

RQ1: Statistically, is the PAUSS-TPS a valid scale according to the results of exploratory factor analysis?

RQ2: Statistically, is the PAUSS-TPS a reliable scale according to the results of internal reliability test?

2. The study

2.1. Method

This section explains the method, the sample, and the development process of the scale to measure the perceived acceptance and use of Scratch for teaching programming. In this scale development process, the authors adopted a mixed-method approach, specifically a sequential mixed model design. As Tashakkori and Teddlie (2003) suggests, the first strand of a sequential mixed model study is exploratory, and the second strand is confirmatory. The first phase of this study entails collecting qualitative data from literature records, and quantitative data afterwards. The steps of scale development are given in-detailed below section.

2.2. Participants

The participants of the study include the ICT teachers who work at secondary schools affiliated to MoNE in Turkey and use the Scratch program at 5th grade which is a part of the secondary school system in the current Turkish educational system. The participants were selected from the population using criterion sampling among purposeful sampling strategies. The participants are 256 ICT teachers who work at secondary schools affiliated to MoNE and use the Scratch program in their classes. It was assumed in the study that the participants were willing to take the survey and provide honest self-reflection. The demographics of the participants are shown in Table 1.

Table 1. Demographics of the participants

Variable	**Category**	**f**	**%**
Gender	Male	112	42.3
	Female	153	57.7
Professional Experience	1-5 years	134	50.6
	5-10 years	89	33.6
	10-15 years	42	15.8
School type	Public School	199	75.1
	Private School	66	24.9

2.3. Scale Development Steps

In scientific research, constructing a scale with specific measurement characteristics for the construct measured is the purpose of scaling and the Likert type, multiple choice, or forced-choice items are commonly used response formats (Kyriazos and Stalikas, 2018). In present scale development study, the construct to be measured is the acceptance and use of Scratch software in teaching programming. Mainly, the researchers followed the steps of scale development which Streiner, Norman and Cairney (2015) suggested. For making these steps clear, the authors drew a visual (Figure 1) and explained each step below Figure 1. Table 1 presents the questions that were developed by the researcher to give the participants the opportunity to demonstrate a range of computational thinking practices. These questions were similar in nature to questions commonly used by Computer Science teachers to stimulate students' thinking in their lessons. Many such examples are available to teachers making use of shared resources such as the Computing At School website (Computing At School, 2020). In addition, questions were developed in response to assessable themes identified in the literature around CT.

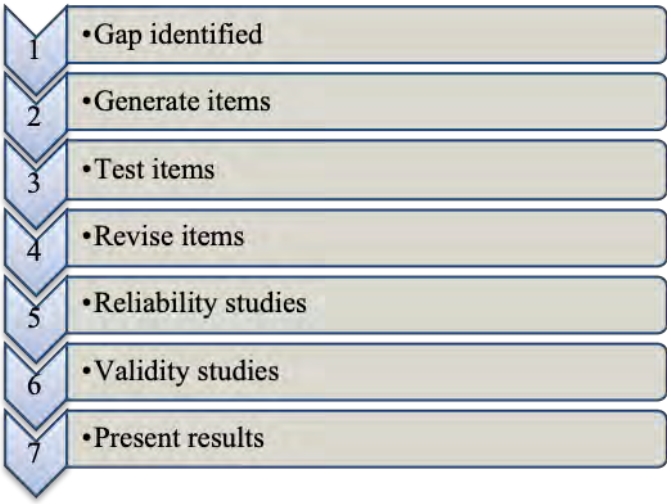

Figure 1. The steps of scale development process in the study (Streiner et al., 2015)

Step 1: Gap identified

For the first step of scale development, literature related to block-based teaching programming was reviewed, available scales were examined. About acceptance or effect of Scratch as a teaching tool, there are some studies with preservice ICT teachers (Choi, 2013; Fesakis and Serafeim, 2009; Saltan and Kara, 2016); and few scale development studies with students (Kasalak and Altun, 2018). Items of scale development studies based on technology acceptance in the literature were also examined (Çam, 2012; Dönmez and Akbulut, 2019; Esen and Büyük, 2014; Özer, Eriş and Özmen, 2012; Serçemeli and Kurnaz, 2016; Teo, 2015; Turan and Haşit, 2014; Ursavaş, 2014). For instance, Çam (2012) examined the factors which affect cloud-computing applications of IT experts and found out that behavioral intentions for applying this technology is affected by perceived usefulness at 99.8% level. In another study, Esen and Buyuk (2014) examined an electronic system, namely e-BYS, acceptance of the employee of the Higher Education Council of Turkey (YÖK). And they reported that self-efficacy, facilitating conditions and social norms positively affect perceived usefulness, perceived ease-of-use, and behavioral intention while anxiety negatively affects. Dönmez and Akbulut (2019) investigated teachers' acceptance of children's internet use. One of their findings is that perceived benefits, risks, and availability of online risks predicted acceptance. Regarding teachers' acceptance, Ursavaş (2014) examined teachers ICT acceptance based on TAM in his dissertation. He extended this framework with some extended variables such as anxiety, compatibility, self-efficacy, perceived enjoyment, facilitation conditions, technological complexity, and subjective norms. Neither of the reviewed studies of scale development were about teachers' acceptance of any of the block-based programming tool.

Step 2: Generate items

To generate the items of the pilot scale form, interviews were held with 15 ICT teachers working at secondary schools affiliated to the MoNE, located within Izmir province. The interviews were conducted using an 11-item semi-structured interview form developed by the first researcher taking the four subscales of the basic TAM into account. A 54-item pilot scale was developed considering the data obtained from the interviews, literature support and the subscales of the UTAUT. Regarding the scale items, opinions were collected from a panel of six experts in the field of Computer Education and Instructional Technology (CEIT). In line with the suggestions of these experts, a technology acceptance measure for teachers developed by Ursavaş (2014) was decided to be used. Of the items, those on which the experts agreed at a degree of 70- 80% were included in the scale (Büyüköztürk, 2017). As a result, a 39-item pilot scale was formed. Prior to the pre-implementation, 10 ICT teachers' feedback was collected about the clarity of the items and the whole scale, also it was observed that the scale was completed in 10 minutes on average.

Step 3: Test items

The pilot scale was applied to the convenience sample of volunteer ICT teachers working at secondary schools

and using Scratch in their classes in Turkey. The data were collected via an electronic questionnaire for 2016-2018 years.

Step 4-5-6: Revise items; reliability and validity analysis

After revising items, validity and reliability studies were conducted. The results were reported for the pilot phase of the study. It was considered enough that the 39-item pilot form was filled out by 265 individuals for the analyses to be performed (Tabachnick and Fidell, 2001). The following analyses were carried out on the data collected; (i) exploratory factor analysis (EFA) on SPSS 23.0 using principal component analysis with the factor loadings accepted as minimum 0.30 through varimax rotation for the construct validity, and (ii) confirmatory factor analysis (CFA) on LISREL 8.72 package program to test construct validity. Additionally, Cronbach's Alpha internal consistency coefficient was calculated for the reliability of the scale.

Step 7: Present results

The results of analysis are given in the Results section below.

3. Results

3.1 Results of validity study

In the present scale development study, exploratory factor analysis was performed to reveal the construct validity and determine factor loadings. Before starting the analysis, Kaiser-Meyer-Olkin (KMO) coefficient was calculated to determine the suitability of data and Bartlett Sphericity test was performed. KMO was found as 0.97 and the result of the Bartlett Sphericity test (x^2 =14329.9, p=0.000) was significant according to George and Mallery (2011). As a result of the exploratory factor analysis, eigenvalue of the scale is seen to be gathered under 3 factors larger than 1. The variance explained by these factors is 75.47%. The items and their factor loadings are presented in Table 2.

Table 1. Item factor loadings obtained from the first factor analysis

Item	Factor 1	Factor 2	Factor 3	Item	Factor 1	Factor 2	Factor 3
m1	.902	-.040	.004	m21	.836	-.011	.061
m2	.920	-.094	.012	m22	.850	-.036	.064
m3	.900	-.142	.072	m23	.812	.014	-.071
m4	.914	-.125	.059	m24	.727	.719	.011
m5	.892	-.055	.065	m25	.804	.080	.055
m6	.901	-.109	.014	m26	.903	-.013	-.012
m7	.909	-.114	.003	m27	.636	-.616	-.014
m8	.853	-.076	.000	m28	-.020	.520	.454
m9	.589	.498	-.209	m29	.515	.456	-.222
m10	.889	-.053	.028	m30	.624	.586	.178
m11	.933	-.072	.022	m31	.475	.378	.308
m12	.927	-.083	-.019	m32	.551	.217	.456
m13	.928	-.108	-.016	m33	.356	.073	.368
m14	.934	-.079	-.037	m34	.883	-.116	.012
m15	.888	.032	-.097	m35	.932	-.066	.002
m16	.910	-.025	-.061	m36	.944	-.089	-.010
m17	.516	.542	-.507	m37	.919	-.062	-.010
m18	.574	.556	-.468	m38	.915	-.044	-.017
m19	.891	.015	-.048	m39	.878	-.076	-.049
m20	.835	-.027	-.094				

When Table 2 was examined, it was seen that items 9, 17, 18, 24, 27, 28 29, 30, 31, 32 and 33 are lower than 0.30 or overlapped items. After removing these items EFA was reapplied. The results of the analysis performed with 28 items show that the items gather under a single factor which explains 80.3% of the variance. Table 3 shows the item factor loadings. Table 3 shows the factor loadings obtained from EFA and CFA analyses and the t values estimated with CFA. Evaluating the t values in accordance with Table 2, the factor loadings are statistically significant. As a result of the EFA, the items included in the scale form consist of a total of 28 items gathering under a single factor with factor loadings ranging between 0.80 and 0.94. This factor explains 80.3% of the total variance. Eigenvalue distribution graph is shown in Figure 2.

Table 3. Item factor loadings obtained from the first factor analysis

	Factor 1			
Item	**EFA***	**CFA****	**t*****	**R_2**
m1	.90	.85	11.14	.72
m2	.93	.87	11.09	.75
m3	.91	.85	11.14	.73
m4	.92	.84	11.18	.71
m5	.89	.82	11.22	.68
m6	.91	.83	11.21	.68
m7	.92	.85	11.15	.72
m8	.88	.79	11.28	.62
m10	.89	.84	11.18	.70
m11	.94	.91	10.85	.83
m12	.93	.90	10.89	.82
m13	.94	.92	10.78	.84
m14	.94	.90	10.90	.81
m15	.89	.85	11.14	.73
m16	.91	.87	11.07	.76
m19	.89	.85	11.16	.72
m20	.84	.78	11.29	.61
m21	.83	.79	11.27	.63
m22	.85	.80	11.26	.64
m23	.81	.76	11.32	.58
m25	.80	.76	11.30	.60
m26	.90	.86	11.12	.74
m34	.89	.86	11.13	.73
m35	.93	.90	10.85	.92
m36	.95	.91	10.75	.85
m37	.92	.89	10.99	.79
m38	.92	.89	10.98	.79
m39	.87	.85	11.15	.72

* EFA factor loadings

** DFA factor loadings

*** Value of significance (t) of the factor loadings estimated with CFA

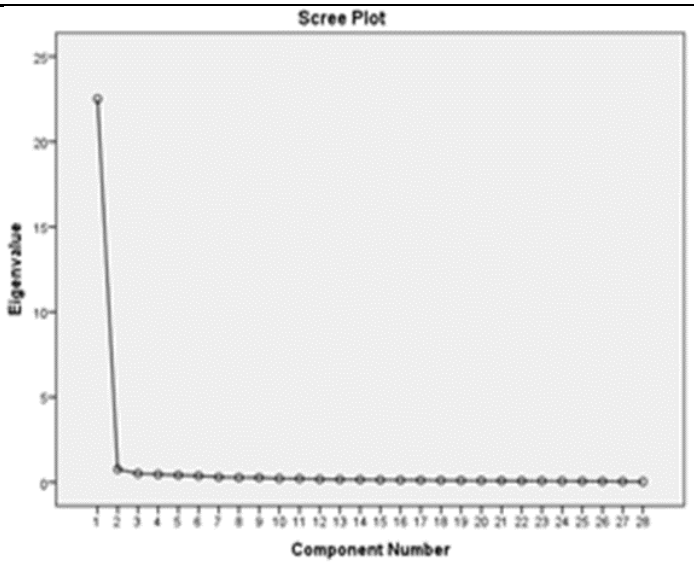

Figure 2. Eigenvalue distribution graph

As the next step of scale development, the items were renumbered, and CFA was performed to see whether the items predicted the factor or not. The model obtained from the analysis is presented in Figure 3. CFA reveals that the construct created is confirmed.

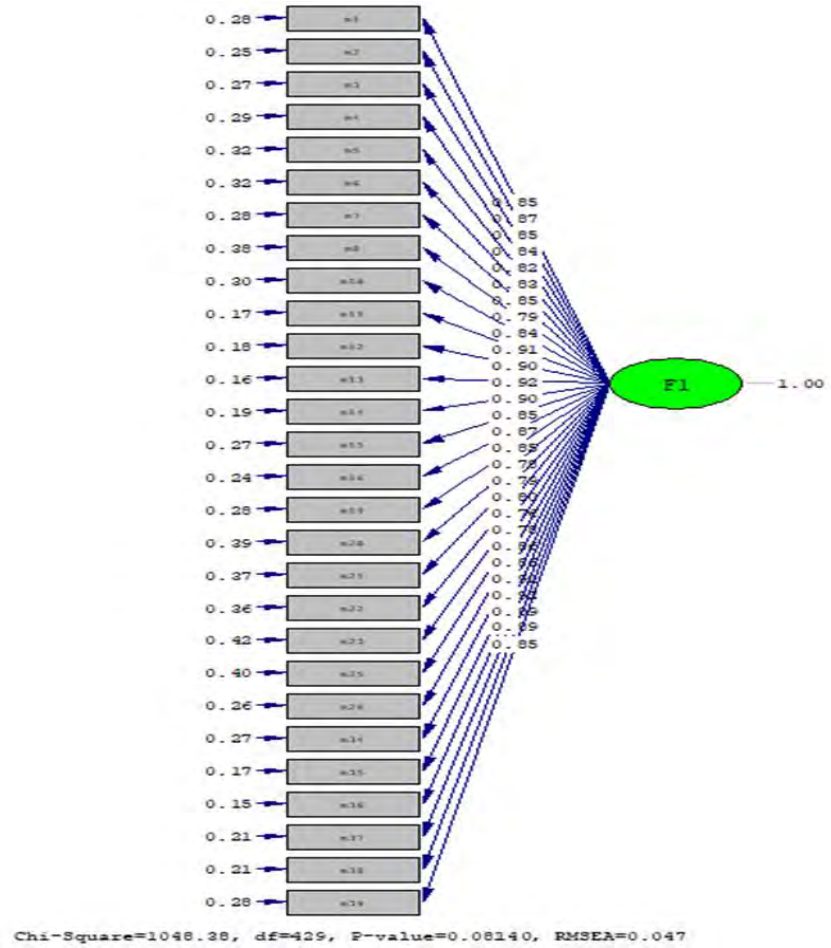

Figure 3. Path diagram pertaining to CFA

Statistical values obtained from CFA are shown in Table 4 together with their acceptable and goodness of fit values according to Schermelleh-Engel and Moosbrugger (2003).

Table 4. Statistical values pertaining to confirmatory factor analysis

	X_2	X_2/df	RMSEA	S-RMR	GFI	AGFI	CFI
Single factor construct	1048.38	2.44	0.047	0.039	0.93	0.87	0.97
Acceptable fit			.05<RMSA<.10	.05<SRMR<.1	.90<GFI<.95	.85<AGFI<.90	.90<CFI<.95
Goodness of fit value		<3	<.05	<.05	>.95	>.90	>0.95

RMSEA : Root Mean Square Error of Approximation

GFI : Goodness of Fit Index

AGFI : Adjusted Goodness of Fit Index

S- RMR : Standardized RMR

CFI : Comparative Fit Index

CFA shows that GFI and AGFI values are in the acceptable fit range, and RMSEA, S-RMR, CFI and X_2/df values are in the good fit range. These values indicate that the model generated is an acceptable one (Schermelleh-Engel et al., 2003; Kline, 2005; Hooper et al, 2008). In conclusion, the results of the confirmatory factor analysis support the construct validity of the scale.

3.2 Results of reliability study

Cronbach's Alpha internal consistency coefficient was calculated to determine the internal reliability of the PAUSS-TPS. Accordingly, the Cronbach's Alpha internal consistency coefficient was calculated as 0.99. A Cronbach's alpha value of .70 and over is enough for the reliability of scale scores (Büyüköztürk, 2017). This indicates that the internal reliability of the scale is quite good. The highest possible score to be obtained from the five-point type 28-item PAUSS-TPS scale is 140 while the lowest possible score is 28. The whole scale consists of positive items. The final form of the scale is presented in Appendix A.

4. Discussion and Conclusion

In the field of information technologies education, teaching programming / coding has gained importance both in Turkey and around the world particularly since the 2000s. Children's interaction with programming at early ages helps them to improve the 21st century skill of problem solving. In addition, product development, problem solving, and analytical thinking skills are also improved with the instruction of programming and designing tools (Akpınar and Altun, 2014; Çakıroğlu, Sarı and Akkan, 2011). In this respect, Scratch, which is a visual programming environment that facilitates learning programming with a visual interface that can attract children's attention, is used in ICT classes. The number of studies conducted on Scratch has increased since 2012. For instance, Çatlak, Tekdal and Baz (2015) reviewed studies carried out in connection with Scratch. Accordingly, Scratch is most commonly used at secondary school level; and the most frequently focused topics studied in relation with Scratch include its effect on algorithm and teaching programming, its effect on problem solving skills, student opinions, its effects on affective characteristics and its use in different courses.

Related literature includes studies examining the effects of Scratch on different variables (Kert and Uğraş, 2009; Kobsiripat, 2015; Kukul and Gökçearslan, 2014; Wilson and Moffat, 2010; Zhang and Nouri, 2019); and the perceptions about contribution of Scratch to teaching programming are mostly determined through blog posts and interviews (Genç and Karakuş, 2011; Kalelioğlu and Gülbahar, 2014). Moreover, some studies about using Scratch in teaching programming and perspectives of preservice teachers (Choi, 2013; Fesakis and Serafeim, 2009; Saltan and Kara, 2016) and students (Kasalak and Altun, 2018) exist. However, the acceptance and use of

Scratch by teachers play a key role for effective teaching programming. Introduction to programming via Scratch is undertaken by ICT teachers within the scope of "ICT and software" course at secondary school level in Turkey. Therefore, it is considered that the scale, which is developed in present study, namely "Perceived Acceptance and Use of Scratch Software for Teaching Programming Scale" (PAUSS-TPS), can provide insight about one of the most commonly used block-based programming tools (Zhang and Nouri, 2019) in teaching programming. Practically, the results of the scale can reveal the teachers' perspective for Scratch in teaching programming at early ages which is beneficial for understanding positive and negative aspects of the tool according to experiences of teachers. When implemented, the results of the scale may lead curriculum designers to suggest or avoid using Scratch which needs further investigations.

The scale consists of self-report questions which is a limitation of the study. Also because of practical reasons, the geographical location which the study took place is Turkey and that can be a delimitation. It is suggested researchers study at other geographical locations and compare acceptance and use of Scratch at other countries as well. It is recommended that researchers should use the PAUSS-TPS to examine ICT teachers' acceptance and use of Scratch for teaching programming in terms of several variables. These variables may include type of school, grade, ICT teacher's experience with Scratch, attitudes towards teaching programming. Additionally, it is also recommended that instruments for measuring perceived acceptance and use of other visual tools (Code.org, Alice etc.) that facilitate teaching programming should be developed, implemented and the results should be discussed.

As for decision-makers, it is suggested that the PAUSS-TPS should be applied to all ICT teachers teaching at secondary school level in Turkey and the contribution of Scratch software should be interpreted from the ICT teachers' perspective since they are given roles in formal curriculum of ICT and software course for using Scratch software and for introducing programming to the students. It is believed to help making decisions concerning adding or omitting activities about this software to/from the programming curriculum at secondary school level. It is suggested researchers study acceptance and use of Scratch and other block-based programming tools to describe the teachers' acceptance levels and make suggestions for better tools and increased effectiveness in teaching programming. The results of such studies help practitioners notice their colleagues' behaviors for acceptance and use of Scratch and can enable them to compare visual programming tools for teaching programming at early ages. The final form of the PAUSS-TPS consists of 28 items and a single factor, as seen in Appendix 1, is thought to be a useful measurement instrument for practitioners, researchers, and decision makers.

Acknowledgements

The present study was produced from the first author's master thesis and was supported by Ege University Scientific Research Projects Coordination Unit. Project Number: 15-EĞF-006.

References

Akpınar, Y. & Altun, Y. (2014). The need for programming education at schools of information society. *Elementary Education Online, 13*(1), dy:1-4.

Büyüköztürk, Ş. (2017). *Sosyal Bilimler için Veri Analizi El Kitabı: İstatistik, Araştırma Deseni, SPSS Uygulamaları ve Yorum [Handbook for Data Analysis in Social Sciences: Statistics, Research Design, SPSS Practices and Interpretation]*. Ankara: Pegem Academy.

Code.org (2018). About us. Retrieved 01.05.2019 from https://code.org/international/about

Choi, H. (2013). Pre-service Teachers' Conceptions and Reflections of Computer Programming using Scratch: Technological and Pedagogical Perspectives. *International Journal for Educational Media and Technology*, *7* (1), 15-25.

Chao, C. M. (2019). Factors Determining the Behavioral Intention to Use Mobile Learning: An Application and Extension of the UTAUT Model. *Frontiers in Psychology, 10.* http://dx.doi.org/10.3389/fpsyg.2019.01652

Çakıroğlu, Ü., Sarı, E. & Akkan, Y. (2011). The View of the Teachers About the Contribution of Teaching Programming To The Gifted Students In The Problem Solving, *5th International Computer & Instructional Technologies Symposium*, (September 22-24), Elazığ: Fırat University.

Çam, H. (2012). Determining the applicability of cloud computing technology in Turkish universities with the Technology Acceptance Model approach. (Unpublished doctoral thesis). Ataturk University, Erzurum.

Çatlak, Ş., Tekdal, M.& Baz, F. Ç. (2015). Scratch Yazılımı ile Programlama Öğretiminin Durumu: Bir Doküman İnceleme Çalışması (The Status of Teaching Programming with Scratch: A Document Review Work). *Journal of Instructional Technologies & Teacher Education, 4*(3), 13-25.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.* 13:319. http://dx.doi.org/ 10.2307/249008

Dönmez, O. & Akbulut, Y. (2019). Modelling teachers' acceptance of children's internet use: A risk-focused inquiry. *The Social Science Journal, 56*(4), 518-529, http://dx.doi.org/ 10.1016/j.soscij.2018.08.014

Durak, H., Karaoğlan Yılmaz, F. G., Yılmaz, R. & Seferoğlu, S. S. (2017). Erken Yaşta Programlama Eğitimi: Araştırmalardaki Güncel Eğilimlerle İlgili Bir İnceleme [Early Programming Education: An Investigation About Recent Trends in Studies]. B. Akkoyunlu, A. İşman and H. F. Odabaşı (Eds.) In *Eğitim Teknolojileri Okumaları 2017 [Educational Technology Readings 2017],* 205-236.

Ersoy, H., Madran, R. O., & Gülbahar, Y. (2011). A Model Proposed for Teaching Programming Languages: Robotic Programming. XIII. *Academic Computing Conference*, 731–736, Malatya.

Esen, M. & Büyük, K. (2014). An investigation of electronic document management system in the context of technology acceptance model: Case of the Council of Higher Education. *Dumlupınar University Journal of Social Sciences, 42,* 313-325. Retrieved from https://dergipark.org.tr/tr/pub/dpusbe/issue/4784/66004

Ferrer-Mico, T. and Prats Fernandez, M. A. & Redo-Sanchez, A. (2012). Impact of Scratch Programming on Students' Understanding of Their Own Learning Process. *Procedia - Social and Behavioral Sciences,* 46, 1219-1223.

Fesakis, G. & Serafeim, K. (2009). Influence of the Familiarization with "Scratch" on Future Teachers' Opinions and Attitudes about Programming and ICT in Education. *ITiCSE'09*, July 6–9, 2009, Paris, France.

Genç, Z. & Karakuş, S. (2011). Learning by design: Using Scratch for developing educational computer games. *5th International Computer & Instructional Technologies Symposium*, Elazığ, Turkey.

Gomes, A. & Mendes, A. (2007). Learning to program - difficulties and solutions, *International Conference on Engineering Education – ICEE 2007*, Coimbra, Portugal.

Gonzalez, C. (2013). *Student Usability in Educational Software and Games: Improving Experiences*. USA: IGI Global.

Gülmez, I. (2009). The effect of using visual tools on learner achievement and motivation in programming instruction. (Unpublished master thesis). Marmara University, İstanbul.

Kalelioğlu, F. & Gülbahar, Y. (2014). The effects of teaching programming via Scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education, 13*(1), 33–50.

Kasalak, İ. & Altun, A. (2018). Blok Temelli Programlamaya İlişkin Öz Yeterlik Algısı Ölçeği Geliştirme Çalışması: Scratch Örneği. *Eğitim Teknolojisi Kuram Ve Uygulama, 8*(1), 209-225.

Kert, S.B. & Uğraş, T. (2009). Simplicity and fun in programming education. *1st International Educational Studies Congress,* Çanakkale, Turkey.

Kobsiripat, W. (2015). Effects of the Media to Promote the Scratch Programming Capabilities Creativity of Elementary School Students. *Procedia-Social and Behavioral Sciences,* 174, 227-232.

Kyriazos, T. A., & Stalikas, A. (2018). Applied Psychometrics: The Steps of Scale Development and

Standardization Process. *Psychology, 9*, 2531-2560. https://doi.org/10.4236/psych.2018.911145

Kukul, V. & Gökçearslan, Ş. (2014). Investigating The Problem Solving Skills Of Students Attended Scratch Programming Course. *8th International Computer & Instructional Technologies Symposium*, Trakya.

Malan, D. J. & Leitner, H. H. (2007), Scratch for Budding Computer Scientists, *SIGSCE'07*, Covington, KY, 223-227.

Meerbaum-Salant, S., Armoni, M. & Benari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education*, *23*(3), 239-264. https://doi.org/10.1080/08993408.2013.832022

Ministry of National Education (MoNE). (2018). Bilişim Teknolojileri ve Yazılım (Information Technology and Software). Retrieved 01.02.2020 from http://mufredat.meb.gov.tr/ProgramDetay.aspx?PID=374

Mueller, J., Wood, E., Willoughby, T., Ross, C., & Specht, J. (2008). Identifying discriminating variables between teachers who fully integrate computers and teachers with limited integration. *Computers & Education, 51*(4), 1523-1537. http://dx.doi.org/ 10.1016/j.compedu.2008.02.003

Oluk A., Oluk, A.H. (2019). Blok Tabanlı Programlama, Korkmaz Ö. (Eds) In *Programlama Öğretimi Yaklaşımları*, 978-605-7846-90-7. Nobel, pp.69- 89.

Ortiz-Colon, A., & Maroto Romo, J. (2016). Teaching with Scratch in Compulsory Secondary Education. *International Journal Of Emerging Technologies In Learning (IJET), 11*(02), 67-70. doi:http://dx.doi.org/10.3991/ijet.v11i02.5094

Özer, P. S., Eriş, E. D. & Timurcanday Özmen, Ö. N. (2012). An Integrative Model Proposition On Behavioral Factors Affecting Intention of Use In Information Technologies Implications. *Dokuz Eylul University Faculty of Economics and Administrative Sciences Journal*, 27, 94-114.

Saltan, F. & Kara, M. (2016). ICT Teachers' Acceptance of "Scratch" as Algorithm Visualization Software. Higher Education Studies, 6 (4), 146-155. doi:10.5539/hes.v6n4p146

Serçemeli, M. & Kurnaz, E. (2016). Investigation of Using Information Technology Products with Technology Acceptance Model (TAM) in Auditing. *Istanbul University Journal of the School of Business, 45*(1), 43-52. Retrieved from https://dergipark.org.tr/tr/pub/iuisletme/issue/30530/330269

Streiner, D. L., Norman, G. R., & Cairney, J. (2015). *Health Measurement Scales: A Practical Guide to Their Development and Use* (5th ed.). Oxford, UK: Oxford University Press. https://doi.org/10.1093/med/9780199685219.001.0001

Šumak, B., & Šorgo, S. (2016). The Acceptance and Use of Interactive Whiteboards Among Teachers: Differences in UTAUT Determinants Between pre-and Post-Adopters. *Computers in Human Behavior, 64,* 602–620. http://dx.doi.org/ 10.1016/j.chb.2016.07.037

Tanrıkulu, E., & Schaefer, B. C. (2011). The users who touched the ceiling of scratch. *Procedia – Social and Behavioral Sciences, 28,* 764 – 769.

Tashakkori, A. & Teddlie, C. (2003). *Handbook of Mixed Methods in Social & Behavioral Research*. Thousand Oaks: Sage.

Teo, T. (2015). Comparing pre-service and in-service teachers' acceptance of technology: Assessment of measurement invariance and latent mean differences. *Computers & Education*, *83* (2015), 22-31. http://dx.doi.org/10.1016/j.compedu.2014.11.015

Turan, B. & Haşit, G. (2014). Technology Acceptance Model and An Implementation on Elementary School Teachers. *International Journal of Alanya Administration Faculty*, *6*, 109-119.

Ursavaş, Ö. F. (2014). Modeling and examining teachers' ICT acceptance (Unpublished doctoral thesis). Gazi University, Ankara.

Van Der Heijden, H.R.M.A., Geldens, J.J.M., Beijaard, D. & Popeijus, H.L. (2015). Characteristics of teachers

as change agents, *Teachers and Teaching, 21*(6), 681-699. http://dx.doi.org/10.1080/13540602.2015.1044328

Venkatesh, V., Morris, M. G., Davis, G. B., and Davis, F. D. (2003). User acceptance of information technology: toward a unified view. MIS Quarterly, *27*(3), 425–478.

Yükseltürk, E. & Altıok, S. (2015). Bilişim Teknolojileri Öğretmen Adaylarının Bilgisayar Programlama Öğretimine Yönelik Görüşleri (Pre-Service Information Technologies Teachers' Views on Computer Programming Teaching). *Amasya Education Journal, 4*(1), 50-65.

Yükseltürk, E. & Altıok, S. (2016). Pre-Service Information Technology Teachers` Perceptions about Using Scratch Tool in Teaching Programming. *Mersin University Journal of the Faculty of Education*, *12*(1), 39-52. http://dx.doi.org/10.17860/efd.94270

Wilson, A. & Moffat, D. C. (2010). *Evaluating Scratch to introduce younger schoolchildren to programming*, 1–12. Retreived 10.02.2018 from http://scratched.media.mit.edu/sites/default/files/wilson-moffat-ppig2010-final.pdf

Zhang, L. & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education, 141* (2019), 1-25. https://doi.org/10.1016/j.compedu.2019.103607

Appendix A. Perceived Acceptance and Use of Scratch Software for Teaching Programming Scale

Dear teacher, you are kindly invited to the study titled "Perceived Acceptance and Use of Scratch Software for Teaching Programming". Before making your decision whether to participate in this study or not, you need to know why and how the study will be carried out. In this respect, it is highly important that this form is well-read and understood. Please feel free to ask us if you happen to fail to understand anything and if there are any unclear points, or you wish to get further information. Participation in this study is completely **voluntary**. You hold the right to **not participate** in the study or to **leave** the study at any point after participation. **Your response to the study** will be evaluated as **your consent for participation in the study**. Please do not get under anybody's pressure or suggestion while responding to the questions on the **forms** given to you. The personal information to be obtained from these forms will be kept completely confidential and will only be used for study purposes. Please read the statements given below. Mark your degree of agreement to the statements with an X. Thank you.

	Totally Agree	Agree	Undecided	Disagree	Totally Disagree
1. Drag-and-drop method facilitates the use of Scratch.					
2. Turkish language support facilitates the use of Scratch.					
3. Grouping of commands in menus facilitates the use of Scratch.					
4. Testing code blocks quickly facilitates the use of Scratch					
5. Scratch increases course efficiency.					
6. Scratch facilitates coding instruction.					
7. Scratch enriches the course.					
8. Scratch improves students' problem-solving skills.					
9. Scratch improves students' algorithm skills.					
10. I enjoy teaching Scratch.					
11. Using Scratch is interesting.					
12. I enjoy taking interest in Scratch.					
13. I will use Scratch in my future courses too.					
14. I will suggest Scratch to others.					
15. I think I will use Scratch frequently.					
16. Students support my use of Scratch.					
17. I can develop different applications using Scratch.					
18. I am self-confident in using Scratch.					
19. If I have a problem using Scratch, I can solve it myself.					

20. I use Scratch because it is a free software tool.					
21. It is easy for students to understand Scratch.					
22. Students are willing to learn Scratch.					
23. It is fun to use Scratch.					
24. I enjoy having classes Scratch.					
25. I like using Scratch.					
26. Scratch is an appropriate program for course objectives.					
27. Scratch is a suitable program for students' level.					
28. Scratch is important to my profession.					