Journal of Engineering and Applied Sciences 13 (Special Issue 2): 2951-2956, 2018
ISSN: 1816-949X

The Analysis on Research Trends for Software Education in Korea

[1]Eunkyoung Lee, [2]Seong-Won Kim and [2]Youngjun Lee
[1]Korea Institute for Curriculum and Evaluation, Seoul, Republic of Korea
[2]Department of Computer Education, Korea National University of Education,
Seoul, Republic of Korea

Abstract: The research trends of software education research in Korea are analyzed in this study. Research papers related to software education were collected by the research information sharing service using the words “software education” and “SW education”. The collected papers were selected for final analysis through expert review. The analytical criteria developed for the purposes of this study were based on previous studies. The results of the study were derived by analyzing the papers based on the developed analytical criteria. A considerable number of studies are being conducted to simplify teaching programming in software education. The research methods are primarily related to the development/application, analysis of theory/content and investigation of condition/awareness. Only a few studies have analyzed the effectiveness of software education. The study subjects of software education have been extremely biased. Elementary school students and the literature were the most frequent, followed by pre-service and in-service teachers. Relatively little research has been conducted on middle and high school students. In addition, to the research subjects being concentrated, they were concentrated in a specific area as a test tool used in the study. Moreover, many of the test tools that were used were developed on their own. Lastly, a block-based programming language was used as the programming language in the study and physical computing devices were rarely used. Software education research is extremely biased. Research using various research methods, subjects and languages is needed.

Key words: Software education, research trends, computational thinking, informatics education, methods, Korea

INTRODUCTION

The development of software is integrated with existing disciplines to promote the emergence of new disciplines and technology. As a result, new technologies are evolving rapidly, creating technologies that have not been experienced before. In addition, software facilitates the emergence of various industries and rapidly transforms the existing social and economic life. These changes are called the fourth industrial revolution; technologies such as artificial intelligence, robots and internet related software are leading this change (Schwab, 2017). Therefore, the importance of software is increasing by the day. However, since, human software resources are limited, software education is being actively introduced to cultivate the creative talents that will lead the fourth industrial revolution throughout the world (Choi *et al*., 2015).

In the United States, the Computer Science Teacher Association (CSTA) developed both a computer science curriculum model and a framework for computer science education through collaboration between CSTA and various educational organizations (Sung and Kim, 2015; Choe, 2016). These frameworks provide a level of practice and concept as well as a platform for introducing computer science education tailored to local and school environments. The United Kingdom reorganized its curriculum centered on computing thinking in the existing ICT-centered curriculum and taught it as essential in elementary, middle and high school (Kim and Lee, 2016a-d). In addition, organizations such as Computing At School (CAS) are creating an environment for revitalizing software education. To keep with this trend in Korea, the information curriculum which was an optional curriculum, was made compulsory for elementary and middle schools, in the 2015 revised curriculum (Park, 2015). The content of the information subject matter was also changed drastically (Lee, 2015, 2016; Kim and Lee, 2017). In addition, the government has been conducting various studies on such things as research-school, development of education programs, teacher training programs and teacher study group support (Kim, 2016). In partnership

Corresponding Author: Youngjun Lee, Department of Computer Education, Korea National University of Education, Seoul, Republic of Korea

with the industry, we are committed to creating a culture and environment for software education (Kim and Lee, 2016a-d; Yang, 2016).

Since, the activation of software education, numerous studies related to software education have been conducted (Jun, 2017). For software education to be successful in the schools, it is essential to analyze the existing research trends, identify the problems of the research and present the research directions for future research. However, since, the introduction of software education, research has been conducted to analyze news and text data (Jun, 2017; Park, 2017). Consequently, there is a lack of research analyzing the trends in software education research. With these considerations, this study aimed to collect research papers related to software education and to analyze the research trends of software education. The purpose of this study was to collect research papers and to analyze the research trends according to the analytical criteria. Based on the results of this analysis, we propose a plan for revitalizing software education in the future.

MATERIALS AND METHODS

Research procedure: In this study, to analyze the research trends of software education, software education related papers were collected from an academic database. Next, the title and abstract of the studies were read and the studies to be used in the research trend were selected. After reading these selected papers in their entirety, the researchers selected the papers to be analyzed in the final research trend. Finally, the analytical criteria were set and the trends in the research of software education were analyzed.

Research data: The Research Information Sharing Service (RISS) from the Korea Education and Research Information Service (KERIS), a Korean academic database, was used to collect the papers to be used in the research. A total of 273 papers were collected by searching “SW education” and “Software Education” as keywords in the RISS (studies published until March 30, 2017 were included). Out of these papers, 104 were selected as the first study subjects, proceeding and duplicate papers were excluded. After reading the titles and abstracts of the selected studies, papers that were determined not to be related to software education were also excluded. After reading the entire contents of the remaining studies and discussing them with the researchers, 76 papers were selected as final research subjects.

Analytical criteria: The analytical criteria used in this study were developed based on previous studies (Kim and Lee, 2016a-d). The analytical criteria used are divided into years, research method, research design, research subject, programming language and physical computing device. The year is excluded in this study since software education has been active, since 2014. However, for the literature analysis of software education, we also added the variables observed in the study. These analytical criteria are shown in Table 1.

Table 1: Analytical criteria

Domain	Detailed criteria
Computational thinking	Yes, no
Research methods	Analysis of effect, development/application, investigation of condition/awareness, analysis of theory/content
Research designs	Quantitative research, qualitative research, mixed method research
Research subjects	Elementary school student, middle school student, high school student, pre-service teacher in-service teacher, literature, etc.
Test tools	Awareness, attention/attitude, computational thinking, educational needs, creativity, interest, satisfaction, educational effect, personality
Programming language	Scratch, App. inventor, etc.
Physical computing device	Arduino, LEGO wedo, etc.

RESULTS AND DISCUSSION

The analysis of computational thinking: The goal of software education is to cultivate competency with computing skills which is the ability to creatively solve real-world problems using computing devices (Kim, 2016). Students should be educated to improve their computational thinking. It is suggested that software education is not different from existing programming education and robot education. However, it is necessary to precisely define and study software education due to the difficulty of terminology such as coding education, informatics education and computer-science education.

In this study, we analyze how computing thinking is utilized in studies of software education. Of the 76 papers, only 24 (31.6%) studied software education to improve computing thinking which was not suitable for software education as shown in Table 2. In Korea, various efforts such as research in schools, revision of curriculums and the development of operational guides have been made to promote software education. However, in a situation where the ambiguity of the definition and the goal of software education are not studied properly, it is impossible to study to train talented people with computing thinking ability. This problem may cause difficulty in establishing software education in the schools. Therefore in software education, it is necessary to clarify the relationship between existing coding education, programming education, information education and computer science education.

Table 2: The result of analysis of computational thinking

Parameters	Yes	No	Total
Number of studies	24 (31.6%)	52 (68.4%)	76 (100%)

Table 3: The result of analysis of research designs

Parameters	Quantitative research	Qualitative research 1	Mixed method research	Total
Number of studies	26 (34.2%)	30 (39.5%)	20 (26.3%)	76 (100%)

Table 4: The result of analysis of research methods

Parameters	Analysis of effect	Development/ application	Investigation of condition/ awareness	Analysis of theory/ content	Total
No. of studies	4 (5.3%)	36 (47.4%)	16 (21.1%)	20 (26.3%)	76 (100%)

The analysis of research designs and methods: In software education, research design was the most studied, with qualitative research at 39.5%. This was followed by quantitative research at 34.2% and mixed method research at 26.3%. These results are related to the study method. In the analysis of research methods, research methods were the most common by development/application at 47.4%. The analysis of theory/content, at 26.3% was the next largest as shown in Table 3. Software training has been active, since 2014. In the 2015 revised curriculum, the informatics curriculum was organized as a mandatory curriculum and the teachers were busy introducing software education to the school.

An analysis is being conducted of the need for educational programs for software and the development of software education programs applying various teaching-learning theories. Therefore, it is considered that the studies have been done significantly by development/application and the analysis of theory/content. Since, these studies are typically made up of qualitative and mixed research methods, it is likely that the qualitative research will result in a high percentage of software education research. In the case of quantitative research, this research design is widely used in the study of the analysis of effect and the investigation of condition/awareness. As a result, the analysis of effect (5.3%) and the investigation of condition/awareness (21.1%) accounted for only a small percentage. Therefore, it is considered to be less than other research designs, as shown in Table 4.

The analysis of research subjects (some studies have been conducted on several research subjects and the sum of the research subjects is larger than the total number of studies): The research subjects were extremely concentrated in software education. While there were very few middle (1.3%) and high school students (5.1%), elementary school students were the most frequently used research subjects (28.8%). Software education needs to be linked to middle, high school and university as well as elementary school. However, it was confirmed that the software education research was concentrated on elementary school students. In addition, numerous studies have been conducted on both pre-service (10.1%) and in-service teachers (11.4%). Most of these studies were conducted to investigate the perceptions and needs of software education in the school. Moreover, a large number of studies were conducted on the literature (21.5%). Since, the analysis of theory/content in the research method has been done a lot, there are thought to be many studies using the literature as the research subject. Studies have also been conducted on various research subjects. To revitalize SW education, the Ministry of Science, ICT and Future Planning (MSIP) operates a SW centered university to train professional SW workers. Therefore, numerous studies have been conducted on non-major college students (7.6%). There have also been some studies (6.3%) in which students have made code through programming. The examination of code was based on studies analyzing the difficulties of programming in automation and the system for evaluating the results of the programming by the students. Research has also been conducted on the analysis of SW trends by analyzing news data (2.5%), parents (1.3%), students of private education institutions (1.3%) and student circles (1.3%) as shown in Table 5.

The results of this study suggest that there are numerous studies for analyzing the awareness and educational need of SW education in the school field and for the development of SW curriculum as well as teaching-learning programs. In addition, there were also studies that applied to actual students and studies focused on elementary school students were in process.

The analysis of test tools: In software education, 34 papers, 44.7% of the total number of studies, examined changes in the study subjects. The most common measure was academic achievement (21.4%). Since, the development and application of educational programs are the most common in SW education, most of the studies investigated the educational effects of the developed educational programs. There have also been a number of studies on the satisfaction of developed educational programs (11.9%). Numerous studies have also measured awareness and interest/attitude/condition (14.3%). These studies investigated the awareness, interest and attitude of SW education for both pre-service and in-service teachers and suggested the implications for the study of SW education. Research related to measuring the educational needs, expected effects and education

Table 5: The result of analysis of research subjects

Parameters	Elementary school student	Middle school student	High school student	Pre-service teacher	In-service teacher	Literature	etc.	Total
Number of studies	23 (29.1%)	1 (1.3%)	4 (5.1%)	8 (10.1%)	9 (11.4%)	17 (21.5%)	17 (21.5%)	79 (100%)

Table 6: The result of analysis of test tools

Parameters	Awareness	Attention/attitude	Computational thinking	Educational needs	Creativity	Interest	Satisfaction	Educational effect	Personality	Total
No. of studies	6 (14.3%)	6 (14.3%)	4 (9.5%)	4 (9.5%)	3 (7.1%)	3 (7.1%)	5 (11.9%)	9 (21.4%)	2 (48.%)	42 (100%)

priorities was also conducted (9.5%). Because software education emphasizes the importance of computing thinking, research has been conducted on computing thinking power as well. However, because there is a lack of testing tools to measure computing thinking, only a handful of studies have actually measured computational thinking (9.5%). Instead of computational thinking, research was conducted to measure academic achievement and creativity. In addition, research was also conducted to measure the interest (7.1%) in the subject, the programming, the physical computing devices and changes in personality (4.8%) as shown in Table 6.

Regarding the test tool results, there were few test tools for measuring computing thinking ability. Therefore, only a few studies measured computing thinking when measuring the effect of the software education. All middle school students should learn the informatics curriculum based on computing thinking from 2018 in 2015 revised curriculum. Nonetheless, the absence of a testing tool to measure computing thinking is a fatal problem in information curriculum. The development of a computational thinking test tool is critical to solving this problem. Moreover, 24 studies (70.6%) did not utilize validated or reliable test tools but used self-developed test tools within the study instead. To obtain the validity and reliability of the study, it is imperative to solve the problem regarding the test tool.

Analysis of programming languages and physical computing devices: Among the previous studies related to software education, few have analyzed the effect of educational programs on elementary, middle and high school students. Therefore, many studies do not utilize programming languages or physical computing devices. Only 23 out of the total number of studies used programming languages (30.3%). Moreover, the programming languages used in this manner were also biased. Scratch was used in more than half of the studies (65.3%) and the rest used App. inventor (13.0%). Other programming languages were Kodu (4.3%), m-Bizmaker (4.3%), Madewithcode (4.3%), Code.org (4.3%) and Unwritten (4.3%) as shown in Table 7. These results indicate that the programming language used in most of the studies was a block-based programming language.

Table 7: The result of analysis of programming language

Parameters	Scratch	App. inventor	etc.	Total
No. of studies	15 (65.3%)	3 (13.0%)	5 (21.7%)	23 (100%)

Table 8: The result of analysis of physical computing devices

Parameters	Arduino	LEGO wedo	etc.	Total
No. of studies	3 (42.9%)	1 (14.2%)	3 (42.9%)	7 (100%)

This result is related to the research and numerous other studies conducted for elementary and middle school students as compared to high school students. A block-based programming language is recommended in the elementary and middle school curriculum and almost all of the studies have been done using a block-based programming language (Park, 2015; Lee, 2015; Kim, 2016). Software education aims not only to end in elementary and junior high school but also to revitalize high school and university and to cultivate the human resources needed in the fourth industrial revolution. To achieve this goal, it is imperative to not only study block programming languages but various text-based programming languages as well.

Physical computing devices help achieve more effective academic achievement in programming education (Benitti, 2012; Lee and Lee, 2008, 2007; Jun *et al*., 2009; Kim and Kim, 2016). Moreover, in the 2015 revised education curriculum, a physical computing area was newly established to teach the content of controlling a program that controls the input/output by building a computing system (Eom *et al*., 2016). Therefore, the importance of physical computing has increased over the existing curriculum. However, in the case of software education research, physical computing devices have been completely ignored. Of all the studies, <10% conducted research using physical computing (9.2%). Arduino was used in nearly half of the studies (42.9%); LEGO Wedo was used in 14.2% and other devices (42.9%) were also utilized. The other devices included creative robots (14.2%), Picoboard (14.2%) and Chocopi board (14.2%) as shown in Table 8. To develop the ability to creatively solve real life problems, it is necessary to develop the competency to build both a physical computing system and a programming language. Therefore, research using physical computing tools needs to be conducted more actively.

CONCLUSION

This study examined the research trends in Korea by analyzing software education study. The conclusions are as follows. First, a definition of software education is needed. In examining the software education research, most of the studies were no different from the existing programming, computer science and information education. To activate the software education at the school site it is imperative to examine the relationship with the existing education and establish the research direction accordingly.

Second, the research methods were extremely biased. To reflect the demands of the schools regarding software education, many studies analyzed the recognition, education needs and interest. Moreover, the majority of the research on the development of educational programs required teachers to teach software education in class. Nonetheless, only a small number of studies have analyzed the effects of software education.

Third, there are numerous studies on elementary school students, teachers and literature as subjects of software education. There have been numerous studies on both pre-service and in-service teachers to reflect the needs of school sites. In addition, many studies analyze the literature for the development of software education related materials. Unlike middle and high school students, elementary school students accounted for the highest percentage of research subjects. To activate software education, more research should be conducted to study middle and high school students.

Fourth, the software education research includes only a few studies that measure computing thinking. Unlike the importance of computing thinking in software education, most of these studies investigated academic achievement, perception, attitude and satisfaction. These results were due to the lack of testing tools to measure computing thinking.

Finally, the programming language used in software education research was concentrated. Most programming languages were block-based programming languages such as scratch and App. inventor. This phenomenon is due to the fact that the subject of educational program development using programming language is concentrated on elementary and middle schools. Moreover, physical computing tools have not been used in most studies.

Four years have passed, since, the introduction of software education, numerous software education policies introduced in the schools have been centered on the government. However, regarding the trends of software education research, it is confirmed that the identity of the software education is being studied in an ambiguous state and the research subjects and tools are extremely biased. Therefore, conducting additional research on various subjects is imperative and basic research related to the definition of software education is urgent. In addition, software education that leads to elementary, middle, high school and college needs to be conducted on the related software education. In the previous research, research subjects were also concentrated and software education was studied separately in elementary, middle, high school and college. Therefore, it is imperative to develop an education model to teach the goal of software education according to the school level.

In software education, improving the student's computational thinking is extremely important. However, in the absence of a tool to measure computational thinking, most of the research has utilized its own tools. Because of the reliability problem, it is difficult to generalize the results obtained through the research. Therefore, it is necessary to actively develop an inspection tool that can measure the goals of software education.

ACKNOWLEDGEMENT

This research was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP) (No. 2016R1A2B4010522).

REFERENCES

Benitti, F. B.V., 2012. Exploring the educational potential of robotics in schools: A systematic review. Comput. Educ., 58: 978-988.

Choe, H.J., 2016. Comparison between informatics curriculum in Korea and computer science framework of CSTA in US. Educ. Res. Inst., 18: 111-129.

Choi, J., S. An and Y. Lee, 2015. Computing education in Korea-current issues and endeavors. ACM. Trans. Comput. Educ., 15: 1-8.

Eom, K., Y. Jang, J. Kim and W. Lee, 2016. Development of a board for physical computing education in secondary schools informatics education. J. Korean Assoc. Comput. Educ., 19: 41-50.

Jun, S., 2017. Analysis of research trends and learners preference for subject area of SW education content. J. Korean Assoc. Comput. Educ., 20: 39-47.

Jun, U., E. Lee and Y. Lee, 2009. A robot programming teaching and learning model to stimulate and maintain professional high school students learning motivation. J. Korean Assoc. Comput. Educ., 12: 13-21.

Kim, H., 2016. A study of the direction for developing software education operating guide. J. Learn. Centered Curriculum Instruction, 16: 529-548.

Kim, J. and T. Kim, 2016. The effect of physical computing education to improve the convergence capability of secondary mathematics-science gifted students. J. Korean Assoc. Comput. Educ., 19: 87-98.

Kim, S.W. and Y. Lee, 2016d. Development of a software education curriculum for secondary schools. J. Korea Soc. Comput. Inf., 21: 127-141.

Kim, S.W. and Y. Lee, 2016b. The analysis on research trends in programming based STEAM education in Korea. Indian J. Sci. Technol., 9: 1-10.

Kim, S.W. and Y. Lee, 2016a. The effect of robot programming education on attitudes towards robots. Indian J. Sci. Technol., 9: 1-10.

Kim, S.W. and Y. Lee, 2016c. The effects of robot programming on the attitudes toward robot of pre-service teachers. J. Korean Assoc. Comput. Educ., 19: 91-103.

Kim, S.W. and Y. Lee, 2017. The effects of programming education using app inventor on problem-solving ability and self-efficacy, perception. J. Korea Soc. Comput. Inf., 22: 123-134.

Lee, C.H., 2015. Direction and model of software education in elementary education. J. Korean Pract. Arts Educ., 28: 207-222.

Lee, C.H., 2016. Development of computational thinking based problem solving model (CT-PS Model) for software education. J. Korean Pract. Arts Educ., 22: 97-117.

Lee, E. and Y. Lee, 2007. The effect of a robot programming learning on problem solving ability. J. Korean Assoc. Comput. Educ., 10: 1-9.

Lee, E. and Y. Lee, 2008. The effects of a robot based programming learning on learners creative problem solving potential. J. Korean Inst. Ind. Educ., 33: 120-136.

Park, C.H., 2015. Relevancy to draft policy of 2015 revised national curriculum for elementary school. J. Learn. Centered Curriculum Instruction, 15: 335-354.

Park, S., 2017. Sentimental analysis of SW education news data. J. Korean Assoc. Inf. Educ., 21: 89-96.

Schwab, K., 2017. The Fourth Industrial Revolution. Crown Business, New York, USA., ISBN:9781524758875, Pages: 192.

Sung, J. and H. Kim, 2015. Analysis on the international comparison of computer education in schools. J. Korean Assoc. Comput. Educ., 18: 45-54.

Yang, D., 2016. Computational thinking vs coding. Rev. Korean Soc. Internet Inf., 17: 55-60.