Computational Modeling and Simulation for Learning an Automation Concept in Programming Course

Yong Cheon Kim, Dai Young Kwon, and Won Gyu Lee

Abstract—**Computational thinking is a fundamental skill for living in the 21st century. The essence of computational thinking is abstraction and automation. Computer science and the concept of automation is affecting various parts our lives. The concept of automation can be improved through the programming activities to design an algorithm, and to implement using a programming language. In this paper, we propose modeling and simulation steps in a programming activity that can learn the concept of automation. And, we are trying to see how modeling and simulation steps can help in the process of modeling practical problems and how they can solve them.**

Index Terms—**Computational thinking, programming, automation, computational modeling, simulation.**

I. Introduction

Computational thinking is a fundamental skill for living in the 21st century [1]. Basic capabilities such as the ability to read, write, and do arithmetic problems require computational thinking, since it is an essential skill for people and not just for computer scientists [2]. Computational thinking affects many aspects of solving real-world problems by computer, designing systems as well as understanding people's behavior. And there is association with processing data in order to predict the weather, and processing large amounts of data automatically in the social network such as Facebook and Twitter.

The essence of computational thinking is abstraction and automation [3]. Abstraction is the process of generalization to express real-world problem by computer, and automation is executing repetitive tasks quickly to use the computer compared to the processing power of a human. In recent years, with the development of information and communication devices, the importance of the concept of automation is emphasized [4].

Concept of automation is essentially required to control the system successfully such as aircraft control systems, medical system, and life science & information systems. In particular, with the spread of smart devices, everyone takes advantage of computers in their daily life and is feasible at solving the problem, the importance of learning the concept of automation based on the principle of the computer is emphasized instead of utilizing the device simply.

In addition, computer science and the concept of automation is affecting a lot of our lives because, unlike the problems occurring in the industrial society, various social problems related to the information & communication society and information device occurs in the information and communication society. Thus, the presence or absence of the concept of automation to solve the problem by using the computer in their respective fields is very important.

The concept of automation can be improved by programming activities to design an algorithm, and to implement using a programming language [5], [6]. However, to create an algorithm is not easy to the novice programmer because it requires a technique of higher cognitive such as critical thinking, logical thinking, analytical skills and creative thinking. And programming languages were developed for a professional use and not to support learning. Complex syntax rules and seemingly arcane punctuation create an additional barrier for novice programmers [7].

It is a difficult process for both creating an algorithm and implementing using programming language. So, in general, programming activities requires a lot of time to learn the programming language instead of solving the problem. And this time consuming aspect can be minimized through automation [8]. It is important to know how to use the command and syntax of the programming language, but there are many difficulties to solve the problem using computer.

In order to learn the concept of automation through programming activities, it is necessary to provide help to allow the learners to the overall understand the real-world problem rather than emphasizing simple activities to create an algorithm, and to implement using programming language [9]. In other words, it must be provided all processes that searching data to solve the real-world problem and process to be implemented to the computer through the activities of abstraction. It is necessary to give scaffolding that can be learning of the concept of automation for novice programmer to help to enable procedural thinking and abstract thinking required to represent to the computer [10].

In this paper, we propose modeling and simulation steps in a programming activity that can learn the concept of automation. And, we are trying to see how modeling and simulation steps can help in the process of modeling the real-world problems and solving the problem.

II. Modeling and Simulation Steps

There are many processes such as data abstraction, algorithm design and implementation in order to express real-world problems to the computer. Therefore, various scaffolding that is appropriate to the level of the learners are necessary to help novice programmer abstract real-world problem and represent to the computer. In this paper, we

Manuscript received November 10, 2013; revised January 15, 2014.

The authors are with the Department of Computer Science Education, Korea University, Seoul 136-701, Korea (e-mail: {yongcheon.kim, daiyoung.kwon, lee}@inc.korea.ac.kr).

DOI: 10.7763/IJCTE.2014.V6.886

proposed a class of simulation based on scaffolding for learning the concept of automation. Class is composed of steps as in Fig. 1. The first step is "Collection". In this step, the learners searches and collects the data to solve the problem. The second step is "Abstraction". The learners extract the important information associated with problems in the collected data. The third step is "Representation". This is when the learners are able to understand the relationship between data and consider how to represent to the computer. The forth step is "Formula". In this step, the learners express the relationship between data by mathematical formula. The fifth step is "Instruction". The learners are possible to comprehend how to use the tools to implement the real-world problems to the computer. The sixth step is "Algorithm". In this step, the learners create a problem solving process for implementing on a computer. The seventh step is "Implementation". In this step, the learners implement an algorithm using programming language.

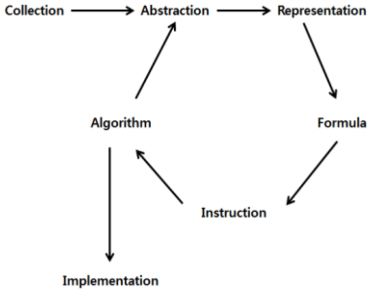


Fig. 1. Modeling and simulation steps.

III. Method

A. Participants and Procedure

In this paper, we are trying to see how modeling and simulation steps can help in the process of modeling the real-world problems and solving the problem. 30 middle school students (27 males, 3 females) with a mean age of 13.3 participated in the experiment. They have no programming experience.

The experiment was conducted 3 hours a day for every 3 weeks. The content of the class is the rotation and revolution of the Earth. Implementation of an algorithms for 3D object animation and analysis used C++ and OpenGL. OpenGL(Open Graphics Library) is a cross-language, multi-platform application programming interface (API) for rendering 2D and 3D computer graphics. The API is typically used to interact with a Graphics processing unit (GPU), to achieve hardware-accelerated rendering [11].

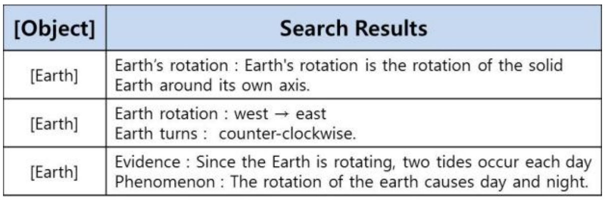
Class was conducted by the following Table I.

TABLE I: Instructional Design

Week	Class	Contents	Teaching Method
1	Modeling	Data Collection Data Abstraction Data Representation Mathematical Formula	Worksheet
2	Implementation	Instruction Algorithm Implementation	Programming
3	Application	Application Discussion	Presentation

B. Materials

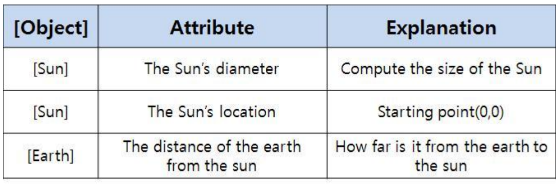
In "Collection" step, the learners search and collect the data to solve the problem. This step consisted of two parts, "Object Name" and "Search Results". Because the object name is used in the next step, we instructed the learners to display using the "[ ]". The following Fig. 2 is an example of the worksheet to be provided to the learners.



[Object]	Search Results
[Earth]	Earth's rotation : Earth's rotation is the rotation of the solid Earth around its own axis.
[Earth]	Earth rotation : west → east Earth turns : counter-clockwise.
[Earth]	Evidence : Since the Earth is rotating, two tides occur each day Phenomenon : The rotation of the earth causes day and night.

Fig. 2. Collection worksheet.

In "Abstraction" step, the learners extract the important information associated with problems among the collected data. Learner can be defined properties of the object that they create in step1. The following Fig. 3 is an example of the data abstraction activity.



[Object]	Attribute	Explanation
[Sun]	The Sun's diameter	Compute the size of the Sun
[Sun]	The Sun's location	Starting point(0,0)
[Earth]	The distance of the earth from the sun	How far is it from the earth to the sun

Fig. 3. Abstraction worksheet.

In "Representation" step, the learners understand the relationship between data and consider how to represent to the computer. Therefore, the learners can perform simulation using squared paper such as Fig. 4. Like step 2, they can use the object that they created at step 1. Learners are able to understand the relationship of the object by attempting to represent all state of objects.

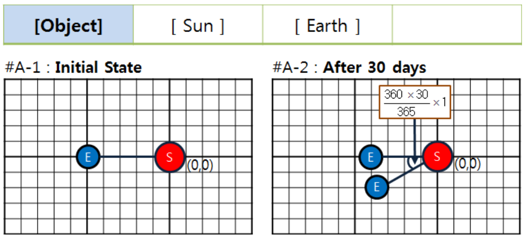
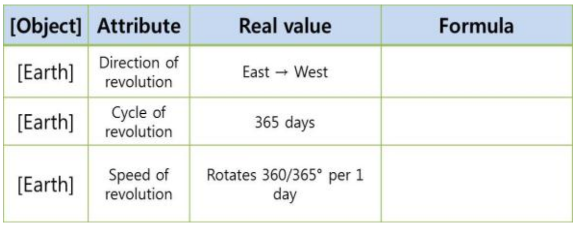


Fig. 4. Representation worksheet.

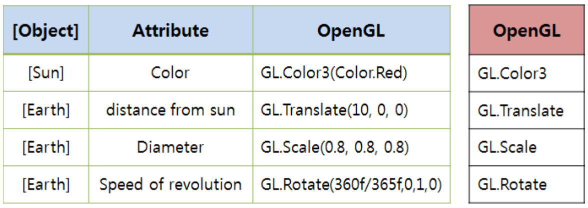
In "Formula" step, the learners expresse the relationship between data by mathematical formula. They can make mathematical formula of the relationship between objects by "Representation Step". The learners must use object that they created at step 1, like other steps. The following Fig. 5 is an example of the making formula activity.



[Object]	Attribute	Real value	Formula
[Earth]	Direction of revolution	East → West	
[Earth]	Cycle of revolution	365 days	
[Earth]	Speed of revolution	Rotates 360/365° per 1 day	

Fig. 5. Formula worksheet.

In “Instruction” step, the learners are possible to learn how to use the tools to implement the real-world problems based on a computer. Fig. 6 refers to the command of OpenGL. Learner can create an algorithm and implement using provided commands.



[Object]	Attribute	OpenGL
[Sun]	Color	GL.Color3(Color.Red)
[Earth]	distance from sun	GL.Translate(10, 0, 0)
[Earth]	Diameter	GL.Scale(0.8, 0.8, 0.8)
[Earth]	Speed of revolution	GL.Rotate(360f/365f,0,1,0)

OpenGL
GL.Color3
GL.Translate
GL.Scale
GL.Rotate

Fig. 6. Instruction worksheet.

In “Algorithm” step, the learners create a problem solving process for implementing on a computer. The learners can create problem solving procedure by using the worksheet such as Fig. 7.

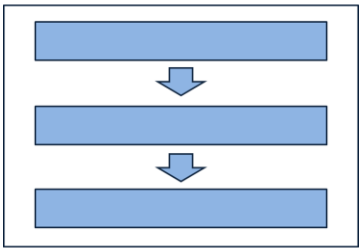


Fig. 7. Algorithm worksheet.

In “Implementation” step, the learners implement an algorithm using programming language. Implementation of an algorithm uses C++ and OpenGL. The following Fig. 8 is an example of the implementation activity.

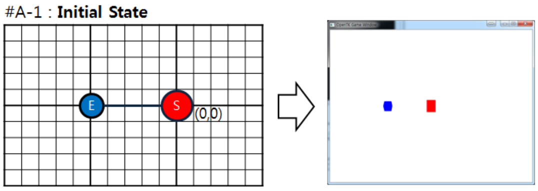
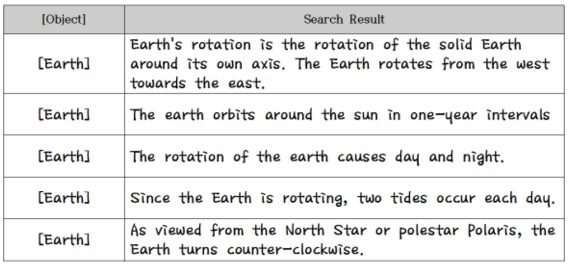


Fig. 8. Implementation worksheet.

IV. Results

A. Result of the Student’s Activity

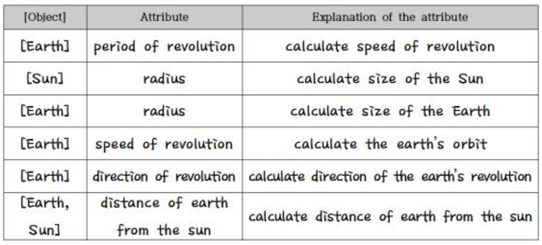
In “Collection” step, the learners collected information of rotation and revolution of the Earth using the internet. Fig. 9 shows the result of collection activity. Learner displays objects using the “[ ]” because the objects name are used in the next step.



[Object]	Search Result
[Earth]	Earth's rotation is the rotation of the solid Earth around its own axis. The Earth rotates from the west towards the east.
[Earth]	The earth orbits around the sun in one-year intervals
[Earth]	The rotation of the earth causes day and night.
[Earth]	Since the Earth is rotating, two tides occur each day.
[Earth]	As viewed from the North Star or polestar Polaris, the Earth turns counter-clockwise.

Fig. 9. Result of collection activity.

In “Abstraction” step, learner defined the object they want to use. The learners can be defined properties of the object that they create in “Collection” step. Fig. 10 shows an abstraction activity that students have defined. Students performed using worksheet like Fig. 11.



[Object]	Attribute	Explanation of the attribute
[Earth]	period of revolution	calculate speed of revolution
[Sun]	radius	calculate size of the Sun
[Earth]	radius	calculate size of the Earth
[Earth]	speed of revolution	calculate the earth's orbit
[Earth]	direction of revolution	calculate direction of the earth's revolution
[Earth, Sun]	distance of earth from the sun	calculate distance of earth from the sun

Fig. 10. Result of abstraction activity.

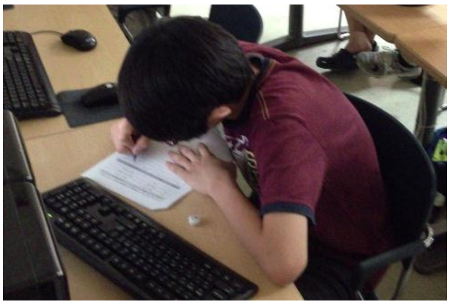


Fig. 11. Activity using the worksheet.

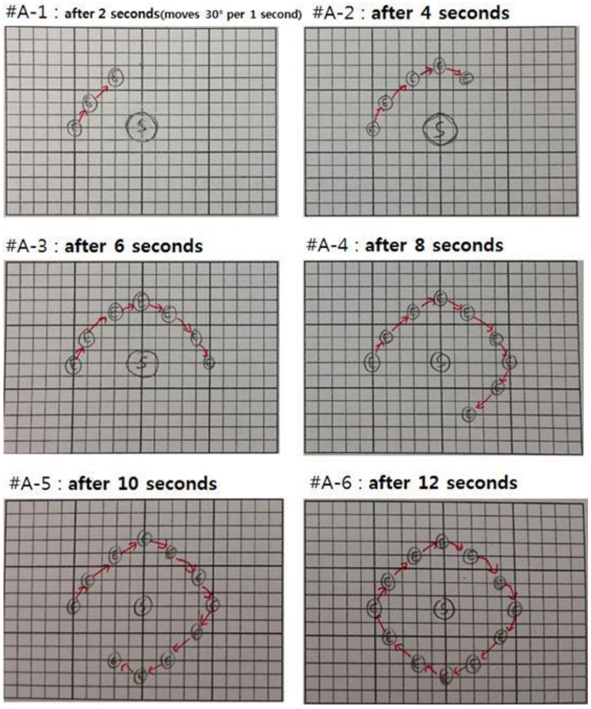


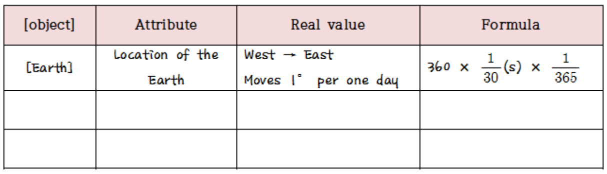
Fig. 12. Result of representation activity.

Fig. 13. Teacher described relationship of objects using gestures.

In “Representation” step, the learners express all situations such as Fig. 12. From this step, they could understand the relationships between objects. Like “Abstraction” step, they can use the object that they created in “Collection” step.

Many students felt difficult to express the relationship between the objects. Therefore, the teacher described the relationship of the objects using gestures such as Fig. 13.

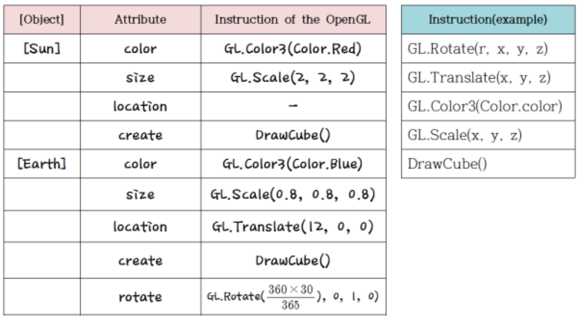
In “Formula” step, the learners could define the relationships between objects. In this step, they can create a formula to calculate the position of the sun and the earth. The formula was consisted by using the result of the search. Fig. 14 shows a formula that students have created.



[object]	Attribute	Real value	Formula
[Earth]	Location of the Earth	West → East Moves 1° per one day	$360 \times \frac{1}{30}(s) \times \frac{1}{365}$

Fig. 14. Result of formula activity.

In “Instruction” step, they can learn how to use programming language. C++ is very difficult to middle school students. So, we provided the only command that is required to implement the rotation and revolution of the Earth. Fig. 15 is an example of using the command to be used when the learners implements.

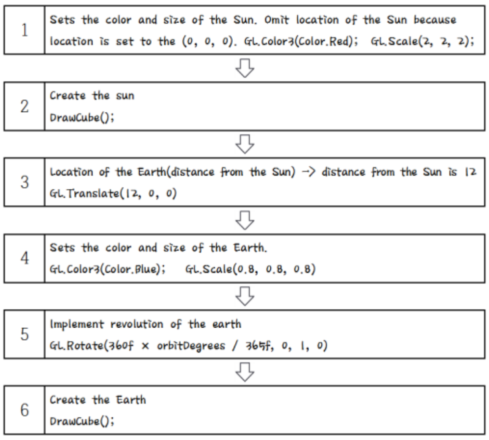


[Object]	Attribute	Instruction of the OpenGL
[Sun]	color	GL.Color3(Color.Red)
	size	GL.Scale(2, 2, 2)
	location	–
	create	DrawCube()
[Earth]	color	GL.Color3(Color.Blue)
	size	GL.Scale(0.8, 0.8, 0.8)
	location	GL.Translate(12, 0, 0)
	create	DrawCube()
	rotate	GL.Rotate($\frac{360 \times 30}{365}$), 0, 1, 0)

Instruction(example)
GL.Rotate(r, x, y, z)
GL.Translate(x, y, z)
GL.Color3(Color.color)
GL.Scale(x, y, z)
DrawCube()

Fig. 15. Result of instruction activity.

In “Algorithm” step, the learners can make a problem solving procedure using formula and instruction. Fig. 16 shows an algorithm that students have created. In this step, they can organize their idea on how to solve the problem.



1 Sets the color and size of the Sun. Omit location of the Sun because location is set to the (0, 0, 0). GL.Color3(Color.Red); GL.Scale(2, 2, 2);

2 Create the sun
DrawCube();

3 Location of the Earth(distance from the Sun) -> distance from the Sun is 12
GL.Translate(12, 0, 0)

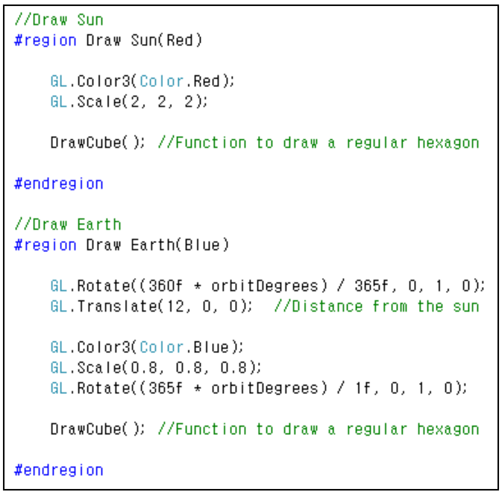
4 Sets the color and size of the Earth.
GL.Color3(Color.Blue); GL.Scale(0.8, 0.8, 0.8)

5 Implement revolution of the earth
GL.Rotate(360f × orbitDegrees / 365f, 0, 1, 0)

6 Create the Earth
DrawCube();

Fig. 16. Result of algorithm activity.

In “Algorithm” step, the learners can implement an algorithm to the programming language. Fig. 17 shows the C++ code that students have created. They can make a programming code using their algorithm.



```
//Draw Sun
#region Draw Sun(Red)

    GL.Color3(Color.Red);
    GL.Scale(2, 2, 2);

    DrawCube(); //Function to draw a regular hexagon

#endregion

//Draw Earth
#region Draw Earth(Blue)

    GL.Rotate((360f * orbitDegrees) / 365f, 0, 1, 0);
    GL.Translate(12, 0, 0);   //Distance from the sun

    GL.Color3(Color.Blue);
    GL.Scale(0.8, 0.8, 0.8);
    GL.Rotate((365f * orbitDegrees) / 1f, 0, 1, 0);

    DrawCube(); //Function to draw a regular hexagon

#endregion
```

Fig. 17. Result of implementation activity.

B. *Student’s Feedback*

After the class, we asked to students about what was the most important activity. Fig. 18 shows students sorting the cards that named by activities in order of importance. Finally, students ordered the cards such as Fig. 19.

Fig. 18. Students who are considered about the order.

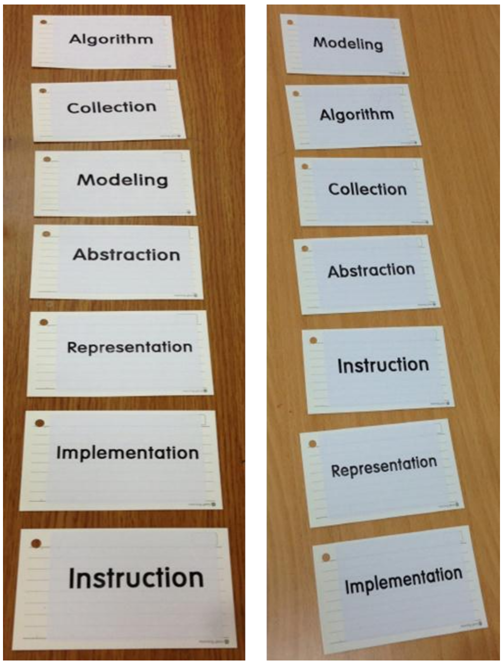


Fig. 19. Priority for students to think.

Most students though “Collection” step was the most important step because information is required to solve the problem. They said that the least important step is “Instruction”. Because it is more important what they do make rather than how they used the programming language.

Table II is the student’s feedback after the class.

TABLE II : STUDENT'S FEEDBACK AFTER THE CLASS

Feel	Student's Opinion
Advantages	It was good to be able to understand the principle of rotation and revolution of the earth, and to create myself rather than just memorization
	It was interesting to learn earth science using computer.
	It was good experience that I check the results visually.
	It was good to realistically implement about differences in the distance and the size of the Earth and Sun
	Implement of the algorithm that I made is the best memory.
	Worksheet is helpful when I programming in a systematic manner
Disadvantages	It was difficult to be used for the first time the programming tool. It takes a lot of time to understand.
	It was hard to understand command that I saw for the first time.
	It would be satisfying if the design is much better.

V. CONCLUSION

In this paper, we proposed modeling and simulation steps in a programming activity that can learn the concept of automation. And, we tried to see how modeling and simulation steps can help in the process of modeling the real-world problems and solving the problem. The results showed that student seems to be satisfied with learning earth science to programming activity. And they think "Collection" step is the most important step among 7 steps because information is required to solve the problem. They said that the least important step is "Instruction". Because it is important what they do make rather than how they used the programming language. From this class we recognized that our modeling and simulation steps are helpful to students do modeling practical problems. And we can confirm that the worksheet is helpful when a student programs in a similar manner.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea Grant funded by the Korean Government [NRF-2013R1A2A2A03016926]).

REFERENCES

[1] J. M. Wing, "Computational thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33-35, 2006.

[2] J. J. Lu and G. H. Fletcher, "Thinking about computational thinking," *ACM SIGCSE Bulletin*, ACM, March 2009.

[3] J. M. Wing, "Computational thinking and thinking about computing," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3717-3725, 2008.

[4] P. J. Denning, "The profession of IT Beyond computational thinking," *Communications of the ACM*, vol. 52, no. 6, pp. 28-30, 2009.

[5] Computational Thinking - School of Computer Science - Carnegie (2006). Computational Thinking - School of Computer Science – Carnegie. [Online]. Available: http://www.cs.cmu.edu/afs/cs/usr/wing/www/publications/Wing06.pdf

[6] G. Futschek, "Algorithmic thinking: The key for understanding computer science," *Informatics Education–The Bridge between Using and Understanding Computers*, pp. 159-168, 2006.

[7] A. Gomes and A. J. Mendes, "Learning to program-difficulties and solutions," in *Proc. International Conference on Engineering Education–ICEE*, Sept. 2007.

[8] A. Eckerdal. (2009). Novice Programming Students' Learning of Concepts and Practise. [Online]. Available: http://user.it.uu.se/~annae/FullAvh-Spikenheten.pdf

[9] W. Lee, "Design of programming learning process using hybrid programming environment for computing education," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 5, no. 10, pp. 1799-1813, 2011.

[10] G. Fessakis, E. Gouli, and E. Mavroudi, "Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study," *Computers and Education*, 2012.

[11] The OpenGL Graphics System: A Specification, Version 4.0 (Core Profile), March 11, 2010.

Yong Cheon Kim received the B.S. and M.S. degrees in computer science education from the Korea University, Seoul, Korea, in 2010 and 2012, and is currently pursuing the Ph.D. degree in computer science education at Korea University. He has also been teaching computing education for elementary and secondary school students in Korea University Gifted Education Center. His research interests include algorithmic thinking, teaching and learning method for improving programming abilities, and educational programming languages.

Dai Young Kwon received the B.S., M.S., and Ph.D. degrees in computer science education from Korea University, Seoul, Korea in 2000, 2006, and 2011, respectively. He became a Research Professor with the Creative Informatics & Computing Institute, Korea University, in 2011. His research interests include algorithmic thinking, educational programming languages, and cognitive experiments for evaluating thinking abilities.

Won Gyu Lee received B.A. degree in English language and literature from Korea University, Seoul, Korea in 1981, and the M.S. and Ph.D. degrees in computer science from Tsukuba University, Tsukuba, Japan, in 1989 and 1993, respectively. He is a professor of computer science education with Korea University. He was the president of the Korea Association of Computer Education from 2002 to 2006. Currently, he is the chief director of the Creative Informatics & Computing Institute, Korea University. Prof. Lee is a member of the Korea Association of Computer Education and the Korea Information Processing Society.