Information Engineering Express
International Institute of Applied Informatics
2015, Vol. 1, No. 2, 21 – 30

Instructional Design of a Workshop "How a Computer Works" Facilitating Intuitive Comprehension with Driving Motivation

Susumu Yamazaki, Takashi Satoh [*], Taku Jiromaru [†],
Nobuyuki Tachi [‡], Masafumi Iwano [§]

Abstract

After teaching and observing students for several years, we hypothesize that learning programming is difficult for students who cannot imagine concretely how a computer works, or the process by which the CPU accesses memory and I/O via the bus according to coded programs. In this paper, we discuss why we believe it is important for programing education to help students understand how a computer works. We have developed a workshop to help students understand this more intuitively. We surveyed the students to assess their perceptions of the workshop, and we discuss its further development and progress toward use in a future full-scale course.

Keywords: How a computer works, instructional design, programming education, workshop.

1 Introduction

We are working and collaborating on instructing undergraduate and graduate students in information engineering in the University of Kitakyushu. We have been troubled that a proportion of the students remain weak at programming over the course of their degree in information engineering. This perception is common among instructors on information engineering.

Why are such students weak at programming? We have observed them for years, and we hypothesize that learning programming is difficult for students who cannot imagine concretely how a computer works, or the process by which the CPU accesses memory and I/O via bus according to coded programs.

Although we have formed this hypothesis, we have not proved it yet. As part of the proof, we assessed how many of our students could explain how a computer works (details are explained

[*] University of Kitakyushu, Fukuoka, Japan
[†] OME, Inc., Fukuoka, Japan
[‡] Nagoya University, Aichi, Japan
[§] NEOZEST, Fukuoka, Japan

in sections 2 and 3). Worse than expected, the result was very poor: none of the 51 students answered the pretest question (described later in the paper) correctly, even though all of them had acquired basic programming skills in C.

Therefore, starting in the spring of 2013, we revised the University of Kitakyushu curriculum to teach students how a computer works at the CPU level. This revision involved integration of the courses "Programming Language Processors (PLP)" in undergraduate year 2 and "Operating Systems (OS)" in undergraduate year 3 into the course "Computer Systems (CS)" in undergraduate year 2. We design the course CS to teach how a computer works and to include the basic content of the courses PLP and OS.

CS will start in the fall of 2014. We have developed and evaluated a workshop-based lesson called "How a Computer Works", which will become a core lesson in CS. We have adopted a workshop style to facilitate intuitive comprehension and motivation. We value intuitive comprehension over strict comprehension because of the above-mentioned hypothesis. We also value motivation, as it facilitates ongoing learning after the workshop. As reading material for the students before the start of the workshop, we reused the self-study materials from the PLP course temporarily. We could adapt this course for the workshop's flipped classroom style easily by using self-study materials, although we provided the materials to the students only during class.

The rest of this paper is organized into five sections: Section 2 describes the instructional design of the preparatory self-study materials and the workshop, Section 3 discusses student assessment, Section 4 discusses improvements to the CS course, Section 5 shows related works, and Section 6 summarizes this paper.

This paper is an extended version of our precede work [1].

2 Instructional Design

What learning objectives do students need to achieve in order to understand how a computer works at the CPU level? Based on our hypotheses developed by observing students, we formulated the following two objectives for knowledge and skill development, according to Gagne's method [2].

- O1: Students will explain the CPU, bus, memory, I/O, register, kinds of basic instructions, execution at the assembly language level, and related words and concepts. (verbal information)
- O2: Given a pseudo-program, students will demonstrate how the CPU exchanges information between the register, memory and I/O when the computer executes the program. (intellectual skill, rule)

We hypothesize that if a student can achieve these objectives, he/she will sufficiently understand how a computer works. O1 and O2 are multiple integrated objectives or enterprises [2].

Instruction materials from the older course could be reused for O1. Table 1 shows the instruction items included in O1. We developed new instruction materials for O2.

Table 1 Instruction items for O1

Title	Items
Computer architecture	CPU, registers, memory, address, data, code, RAM, ROM, I / O, interface, memory-mapped I / O, port-mapped I / O, bus
CPU abilities	"Programs performing complex functions comprise simple procedures"
Mechanism and principles of the CPU	Load, store, addressing mode, general register, special register, program counter, stack pointer, flag register, instruction cycle, fetch, decode, run, subroutine, machine language, instruction set, instruction set architecture, assembly language, assembler, opcode, operand, runtime memory, static variable, automatic variable, code area, data area, runtime stack, heap, activation record
Virtual machine	Virtual machines, intermediate code, immediate, absolute addressing, register direct, register indirect, base plus offset, register auto-increment indirect, PC relative, data transfer instruction, arithmetic instruction, logic instruction, shift instruction, rotation instruction, branch instruction, special instruction, pseudo-instruction
Code generation	Code generation rules (basic arithmetic operations, control structure, variables, functions), example of code generation (program for calculating factorials)
Optimization	Equivalence, strength, "Optimization must be equivalent", "Optimization selects an operation of the lowest strength as possible", "What is equivalent?", difference in speed between CPU and memory, negative correlation between capacity and speed, use a high-speed storage device as much as possible, to synchronize with large storage, execution frequency, optimize code that is executed frequently, importance of analyzing what the computer spends time on, parallel processing, dependency analysis of each instruction, basic block, procedure, register assignment, lifetime of variables, remove redundancy, constant propagation, dead code elimination, code motion, loop invariant, inlining, instruction scheduling, suppress optimization, volatile memory used for I/O or accessed by another CPU.

Figure 1 shows the pre- and posttests for O2 (the first and second questions, respectively). Q1 was the pretest conducted in 2012, before the workshop, and mentioned in section II, and Q2 was the posttest conducted in 2013, after the workshop. Q1 is described with a higher pseudo-language. Q2 is described with pseudo-assembly language. The reason of this change is that we have reviewed the pre- and posttests for this workshop. The students' results for these tasks are summarized in section 3.

Q1. Explain how information is exchanged in a computer when the following pseudo-program is run using the following computer system configuration diagram.
(1) Obtain coordinate data from a pointing device
(2) Calculate the screen coordinate based on the coordinate data
(3) Draw with a cursor based on the screen coordinate calculated in (2) and graphics data stored in memory
(4) Go to (1)

Q2. Explain how information is exchanged in a computer when the following pseudo-program is run using the following computer system configuration diagram. The initial valuable of the variable X is 2.
(1) Read the value from the variable X to Register B.
(2) Write 1 on Register A.
(3) Double the value of Register A.
(4) Subtract the value of Register B 1.
(5) Go to (3) if the value of Register B is not 0.
(6) Set the value of Register A as the value of the valuable X stored in memory.
(7) Exit

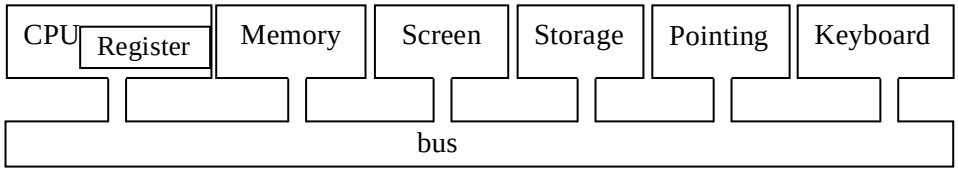

Figure 1. Pre- and Posttest for O2

As an instructional strategy, first, we first addressed O1 by distributing self-study materials followed by a test, after which we addressed O2 by the workshop itself, reflection after the workshop and the posttests (Q2).

We define it as such intellectual skills as O2 that the students will understand intuitively how a computer works. The skills means that the students can describe in details how a CPU in the computer computes along a given program with accessing its memory and I/O through the bus and with rewriting its registers. This means quite that the students can describe how a computer works.

Thus, we define the instructional strategies for O2 in the workshop as a reproducing work how a CPU works. In this paper, we call the strategies the Role Playing Workshop, which means that the students play a role of a CPU. We will show other instructional strategies in previous works to teach how a computer works in Section 5.

In the workshop, we instructed the students to work in groups of four to trace a simple program written in Z80 CPU assembly language. The students can refer provided reference texts and describe the changes in the registers. The program used in the exercise was quite simple, e.g. addition of a variable in memory, and multiplication by a simple repetition of addition. As references, we provide the Z80 instruction set manual for beginner [3] and a catalogue of Z80 mnemonics, which map between machine language and assembly language.

This program tracing exercise was designed for the intellectual skill of O2 by a demonstration of program tracing: we aimed for the students to achieve O2 by practicing repeatedly tracing the flow of assembly language programs like that used in the workshop.

3 Assessment

We took a survey to assess the relationships of students' perceptions and motivation between the classes on PLP before the workshop (e.g. the runtime environment, virtual machines, code generation and optimization) and the workshop how a computer works. We define questions on the students' motivation, according to the ARCS model [4], and enjoyment.

The questions in the survey questionnaire (response options) were as follows:

- Before this workshop, how well did you understand the runtime environment, virtual machine, code generation, and optimization? (Very well, a little, not so much, not at all.)
- Before this workshop, were you curious about how a computer works? (Very much, a little, not so much, not at all.)
- Before this workshop, were you familiar with how a computer works? (Very familiar, a little, not so much, not at all.)
- Before this workshop, were you confident about your understanding of how a computer works? (Very confident, a little, not so much, not at all.)
- Were you curious about this workshop? (Very much, a little, not so much, not at all.)
- After this workshop, are you familiar with how a computer works? (Very familiar, a little, not so much, not at all.)
- After this workshop, are you confident in your understanding of how a computer works? (Very confident, a little, not so much, not at all.)
- Were you satisfied with this workshop? (Very much, a little, not so much, not at all.)
- Did you enjoy this workshop? (Very much, a little, not so much, not at all.)
- Do you think that role-playing to perform the CPU's actions in this workshop was better than learning about this in a normal lecture? (Very much, a little, not so much, not at all.)
- Do you think that knowledge about the runtime environment, virtual machine, code generation and optimization was useful for your learning in this workshop? (Very much, a little, not so much, not at all.)

We received 58 valid responses from among 60 students. The following is a summary of the survey responses:

1) Fifty-four of the 58 (93%) students answered that the workshop was enjoyable.
2) Enjoyment of the workshop seemed to relate to the students' confidence about achievement of O1 and O2: 32 of the 54 Students who enjoyed the workshop (59%) answered positive confidence.

We conclude that the workshop was effective in motivating the students to learn about how a computer works. According to Ichikawa [5], students' enjoyment to learning is the strongest intrinsic motivation. Thus, students' enjoyment to the workshop contributes to bring learning motivation.

3) Achieving O1 may lead to the achievement of O2 in this workshop: 22 of the 28 (73%) students who answered that they understood the instruction items of O1 "very well" or "a little" on O1 before this workshop answered that they were "very" or "a little" confident after this workshop
4) Students who does not achieve O1 may slightly achieve O2: 12 of 30 (40%) students who answered that they did not understand the runtime environment, virtual machines, code

generation, and optimization (the instruction items of O1) before the workshop answered that they were confident after this workshop.

We conclude that improvement in student comprehension of O1 will be effective in improving student outcomes from this workshop.

5) Experience of role-playing on the workshop may lead to confidence: 31 of the 52 (60%) students who answered positively to the experience of role-playing answered to gain confidence after the workshop.

We conclude that the workshop is effective at least to drive students' motivation.

4 Discussion Towards The New Course

According to the conclusion in section 3, improvement in students' comprehension of the runtime environment, virtual machines, code generation, and optimization, as described in O1, will be effective in improving student outcomes from this workshop. Thus, we propose that a better instruction strategies to provide a lecture or self-learning materials for the achievement of O1 before the workshop.

In terms of experiential learning [6], it will be effective for students to reflect on the lessons learned and their experiences in this workshop by discussing these with each other after the workshop.

It is important to change the instruction strategy after the workshop according to the workshop according to the students' comprehension and enjoyment of the workshop:

1) Students who become confident and enjoyed the workshop reported that they had learned well about how a computer works. We will provide advanced learning materials for deeper learning about how a computer works.
2) Students who did not gain confidence but enjoyed the workshop may be able to achieve the objectives by relearning the materials. To help them, we will provide opportunities for relearning with reflection. It is important to relate the students' experiences in the workshop to knowledge of the O1 instruction items.
3) Students who gained sufficient confidence but did not enjoy the workshop may have already learned the information, may not have agreed with the instructional strategies of the workshop, or may have the illusion that "I understand already". We will provide the advanced learning materials because they achieved O1 and O2, though we must interview them about why they did not enjoy the workshop and follow-up on their subsequent progress.
4) For students who did not become confident and did not enjoy the workshop, the learning obviously failed. We plan to offer the same information to help them relearn it by providing individual follow-up, including a detailed analysis. Hopefully, there will be few students requiring such follow-up.

Figure 2 shows the flowchart of this lesson including the workshop.

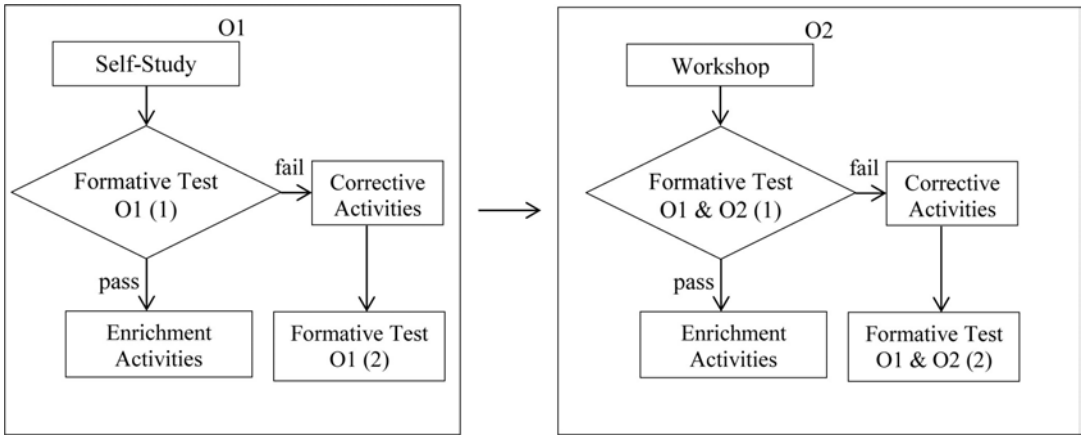

Figure 2. The Flowchart of the lesson including the workshop

5 Related Works

We have conducted a survey on instructional strategies of "How a computer works". We categorize them into four approaches:

1. Advance Organizer approach;

2. Role Playing Workshop approach;

3. Simulated Virtual Machine approach; and

4. Real Machine approach.

The Advance Organizer is an instructional strategy, which is proposed by David Ausubel [7], to facilitate learning unknown information by relating to known information. Kawamura [8][9] proposed to teach high school students how a computer works by relating to functions, structures and behavior of a Japanese word processor, which was popular and familiar to the students in 1995.

The Role Playing Workshop approach, which is adopted in this paper, is aimed at instruction of how a computer works by getting into a role of a computer. We found a case study of this category, Maeda et al. in MIT [10].

The Simulated Virtual Machine approach and the Real Machine approach are instructional strategies how a computer works by operating a simplified computer: in the former, the learners operate a simulated computer realized by software on a PC, and in the latter, they operate a micro-computer board. We don't use PCs directly to teach how a computer works because they are too complex for the beginners to understand. Case studies of the former are Tamagawa University [11], Kimuro et al. [12], Dolittle [13][14], Li and Li [15], etc. Case Studies of the latter are

KITE [16], GAINER [17], KERNEL [18], Yatsushiro National College of Technology [19], Tokuyama College of Technology [20], etc.

The Advance Organizer and the Role Playing Workshop approaches are corresponding to the Analogical Thinking-based Learning (Analogy dropping method) and Self Role-playing-based Learning (Self role-play method) of Analogical Thinking Theory [21][22], respectively.

In the order of the Real Machine, the Simulated Virtual Machine, the Role Playing Workshop, and the Advance Organizer, the learners have a potential to understand how a computer works more deeply and correctly, while it needs more knowledge and skills to operate computers to learn.

We may have more advantages to facilitate the learning by using the approaches together. In this case, we should use in the order of the Advance Organizer, the Role Playing Workshop, the Simulated Virtual Machine and the Real Machine, because of easiness of acquiring knowledge and skills of computer operations. In general, we use some of these approaches together because of time constraints of learning.

In this workshop, we decide to adopt the Role Playing Workshop approach because it is the best way that students can understand how a computer works as precisely as possible, although the whole CS class cannot have enough time to master operations of virtual machine or real machine.

Maeda [10] shows only a concept video movie, which doesn't include details of the workshop lesson and its assessments of effectiveness. Thus, we adapt and rebuild it in the context of system programing, based on instructional design, and assess and analyze its effectiveness.

A future full-scale course of the workshop will be followed by the experiment courses that adopt the Real Machine approach. This will bring more effectiveness.

6 Conclusion

In this paper, we have described the workshop "How a Computer Works", designed to help students understand this topic intuitively because it is important for programming education. We have described the design of the workshop and its assessment through a survey of students.

We have also discussed the further development of the workshop and its progress toward use in the future full-scale course. We have taught the full-scale course in the fall of 2014, and will report it.

We have showed effectiveness of students' motivation. We will assess effectiveness of students' intuitive comprehension.

Acknowledgement

We thank: Toshihiro Shima from Seiko Epson Corporation; Prof. Katsuaki Suzuki, Prof. Junko Nemoto and Prof. Atsuko Takahasi from Kumamoto University; Prof. Satoshi Asano in

Yokohama Digital Arts Technical College; and the students of University of Kitakyushu.

We also thank the students and teachers of Kumamoto University Graduate School of Instruction Systems.

References

[1] S. Yamazaki, T. Satoh, T. Jiromaru, N. Tachi and M. Iwano, "Instructional Design of a Workshop How a Computer Works Aimed at Improving Intuitive Comprehension and Motivation", 3rd International Conference on Learning Technologies and Learning Environments (LTLE 2014), 2014.

[2] R. Gagné, W. Wager, K. Golas, and J. Keller, "Principles of Instructional Design", 5th edition, Wadsworth Pub, Belmont, CA, 2004

[3] Chunichi Denko, Inc. "Z80 Instruction Set Manual" (in Japanese), http://www.alles.or.jp/~thisida/nd3setumeisyo/nd3_z80meirei.pdf

[4] J. Keller. "Motivational Design for Learning and Performance: The ARCS Model Approach", Springer, NY, 2010.

[5] S. Ichikawa, "Manabu Iyoku no Shinrigaku (Psychology for Learning Motivation)" (in Japanese), PHP Institute, Tokyo, Japan, 2001.

[6] D. Kolb, "Experiential Learning: Experience as the Source of Learning and Development", Pretice Hall, NJ, 1984.

[7] D. Ausubel, "The Psychology of Meaningful Verbal Learning". Grune & Stratton , NY, 1963

[8] K. Kawamura, "A Primer Course of Computer Science used Japanese Word Processor" (in Japanese), Proceedings of the 51th National Convention of IPSJ, 277-278, 1995-09-20, Information Processing Society of Japan (IPSJ), 1995. http://ci.nii.ac.jp/naid/110002876921

[9] K. Kawamura, "A Guide Education of Computer Science as the subject Information at a high-school" (in Japanese), The Special Interest Group Technical Reports of IPSJ, IPSJ Special Interest Group on Computers in Education (IPSJ-SIGCE), 96(52), 45-51, 1996-05-24, Information Processing Society of Japan (IPSJ), 1996. http://ci.nii.ac.jp/naid/110002776389

[10] J. Maeda, "Human Powered Computer Experiment",

http://www.youtube.com/watch?v=KaIxBIclGUQ, 1993.

[11] M. Tsuchiyama, "Development of an educational computer simulator", Electronics and Communications in Japan (Part III: Fundamental Electronic Science), Volume 74, Issue 3, pages 25–35, DOI: 10.1002/ecjc.4430740304, 1991.

[12] Y. Kimuro, M. Matsumoto and H. Yasuura, "An Educational Method of Computer Princi-

ples for Elementary and Junior High School Students" (In Japanese), JOURNAL-INSTITUTE OF ELECTRONICS INFORMATION AND COMMUNICATION ENGINEERS, 86(11), 868-873, 2003. http://ci.nii.ac.jp/naid/110003231115

[13] S. Kanemune, T. Nakatani, R. Mitarai, S. Fukui and Y. Kuno, "Dolittle-Experiences in Teaching Programming at K12 Schools," In C5 (pp. 177-184), 2004.

[14] S. Kanemune and Y. Kuno, "Dolittle: an object-oriented language for K12 education", In EuroLogo (pp. 144-153), 2005

[15] J. Li and S. Li, "Exploration of computer-principle virtual laboratory", Proceedings of 2011 International Conference on Electronics and Optoelectronics. Vol. 4., 2011.

[16] T. Sueyoshi, K, Tanaka and H. Shibamura, "KITE: An Educational Microprocessor Using Field Programmable Gate Array" (In Japanese), Proceedings of the 45th National Convention of IPSJ, 67-68, 1992-09-28, Information Processing Society of Japan (IPSJ), 1992. Available at http://ci.nii.ac.jp/naid/110002890024

[17] K. Harada, and S. Kobayashi, "GAINER: A Reconfigurable I/O Module for Media Artists" (in Japanese), IPSJ Special Interest Group on Music and Computer, 2005(129), 7-11, 2005-12-23, Information Processing Society of Japan (IPSJ), 2005. http://ci.nii.ac.jp/naid/110003496057

[18] T. Hananoi, K. Ushijima and M. Nishioka, "A Bi-directional Practical Education through Industry-University Cooperation", The Special Interest Group Technical Reports of IPSJ, IPSJ Special Interest Group on Information Systems (SIG-IS), 2007(25), 71-74, 2007-03-14, 2007. http://ci.nii.ac.jp/naid/110006249693

[19] S. Oshima, Y. Shirai, J. YUJI, I. Inoue, T. Moriuchi, M. Isogai and Y. Fujimoto, "Short-Term PBL Education for Learning Skills of Microcomputer System Development", Journal on Japanese Society for Engineering Education, 55(3), 105-110, 2007-05-20, Japanese Society for Engineering Education, 2007. http://ci.nii.ac.jp/naid/10019556877

[20] T. Shigemura, T. Furukawa, M. Ohchi and T. Hayashi, "Development and Application of Educational Microcomputer System to Instruct Machine Language Programming with a Console Panel", IPSJ Journal, 48(9), 3318-3327, 2007-09-15, Information Processing Society of Japan, 2007. http://ci.nii.ac.jp/naid/110006423008

[21] T. Matsuo and T. Fujimoto, "An Analogical Thinking Based New Software Engineering Education Methodology", Computer and Information Science, SCI 131, pp. 77-86, Springer-Verlag Berlin Heidelberg, 2008.

[22] T. Matsuo and T. Fujimoto, "Electronic Learning Support System Based on Analogy Reuse", In Proceedings of IEEE International Conference on Information Reuse and Integration 2008 (IRI 2008), IEEE, 2008.