An Extension Course for Training Trainers of Embedded Software

Masaki Yamamoto*, Shinya Honda*, Hiroaki Takada*, Kiyoshi Agusa*
Hiroyuki Tomiyama*, Kenji Mase**, Nobuo Kawaguchi*** and Nobuyuki Kaneko*

*Graduate School of Information Science, Nagoya University
**Information Technology Center, Nagoya University
***Graduate School of Engineering, Nagoya University
Furo-cho, Chikusa-ku
Nagoya, 464-8063, Japan

myamamoto@nces.is.nagoya-u.ac.jp, {honda, hiro}@ertl.jp,
{agusa, tomiyama}@ is.nagoya-u.ac.jp, {mase, kawaguti}@nagoya-u.jp,
kaneko@agusa.i.is.nagoya-ua.c.jp

ABSTRACT

The embedded system industry in Japan needs more and more embedded software engineers. In 2004, Nagoya University has started an extension course program for embedded software engineers, called NEXCESS (Nagoya university EXtension Courses for Embedded Software Specialists). In the first year of NEXCESS, we have got aware of the tremendous need for the introductory-level courses. With this demand, in the second year, we have started a new course to train future trainers of embedded software. After completion of the trainer course, they are expected to train introductory-level embedded software engineers at their companies. This paper reports the extension course for embedded software trainers.

Categories and Subject Descriptors

K.3.2 **[Computing Milieux**]: Computer and Information Science Education

General Terms

Human Factors

Keywords

extension course, embedded software, workshop, skill, trainer

1. INTRODUCTION

In the automotive and electronics industries over the world, Japan is one of the most competitive countries. A large number of embedded software engineers underpin the high competitiveness. It is reported in [1] that approximately 193,000 engineers are working on the development of embedded software in Japan. Nevertheless, the report also reveals a shortage of as many as 94,000 engineers.

In 2004, we have started an extension course program named NEXCESS (Nagoya university Extension Courses for Embedded Software Specialists) [2], supported by Ministry of Education, Culture, Sports, Science and Technology, Japan. NEXCESS is targeted towards embedded software engineers in industry. In the first year, NEXCESS offered eight courses at several levels of expertise, from the introductory level to the advanced level.

Among the eight courses, we have recognized that the strongest demand from industry is addressed towards the introductory-level course. The background of this fact is that few universities in Japan provide sufficient classes on embedded software development even for computer science students. Although NEXCESS opens the introductory course several times a year, its capacity is limited. Therefore, we have decided to develop a new class to train future trainers of embedded software. After completion of the trainer course, they are expected to teach introductory-level embedded software engineers at their companies.

This paper presents an outline of the introductory course and the trainer course in NEXCESS.

The rest of this paper is organized as follows. Section 2 describes the needs of embedded software engineers in industry. Section 3 presents the NEXCESS introductory courses. Section 4 presents the trainer course. Section 5 reports the results of the trainer course.

2. NEEDS

2.1 Courses

When we designed extension courses, we first classified technical skills into three levels as follows.

WESE2006, October 26, 2006, Seoul, South Korea

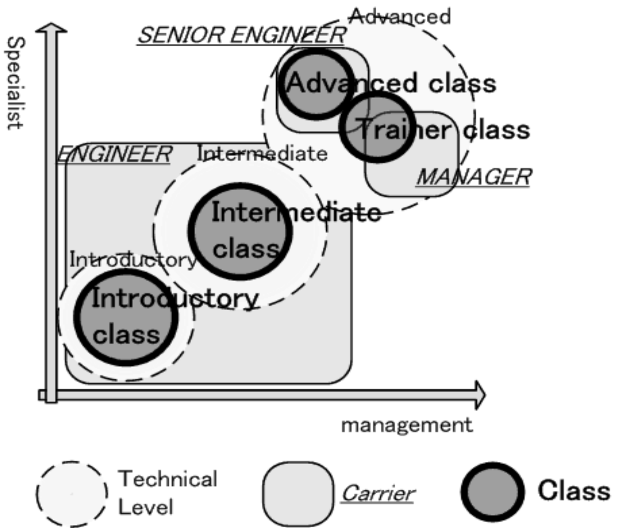


Figure 1. Course Map

(1) Introductory Level

(2) Intermediate Level

(3) Advanced Level

We also classified career paths for engineers in two directions as follows.

(1) Administrative Managers

(2) Technical Specialists

Then, we classified the engineers into four classes using the both sides of technical skill and management ability as shown in Figure 1. Engineers belonging to each class have the following responsibility.

(1) Introductory class

- ➢ Engineers in the introductory class are engaged in the programming process of software development, and in direct software production (see Figure 2).

(2) Intermediate class

- ➢ Engineers in the intermediate class are engaged in the design and high quality testing processes of software development, and in direct software production (see Figure 2).

(3) Advanced class

- ➢ Engineers in the advanced class work as a specialist of each advanced technology, and are engaged in indirect software production.

(4) Trainer class

- ➢ Engineers in the trainer class work as an educational specialist.

We developed nine extension courses, which educate engineers in these classes. Each course is only two to four days long so that industrial people can easily take the class. The titles of the nine courses are listed below.

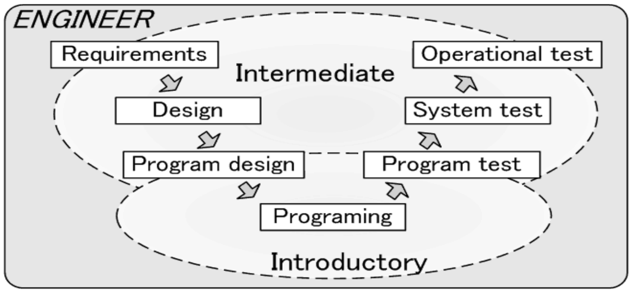


Figure 2. Software process and Introductory engineer

Table 1. Course list

Course	Total applicants	Applicants per course
Introductory class	273	54.6
Intermediate class	154	38.5
Advanced class	261	26.1
Trainer class	17	17

(1) Introductory class (one course):

(course-1) “Fundamentals of the embedded software development technology”

(2) Intermediate class (two courses):

(course-2) “Design methodology and management of embedded software”

(course-3) “Software design technology with a real-time OS”

(3) Advanced class (five courses):

(course-4) “Internal structure of a real-time OS”

(course-5) “C-based embedded hardware design”

(course-6) “System control middleware and application”

(course-7) “Software engineering for embedded systems”

(course-8) “Ubiquitous interface and embedded software programming for image processing”

(4) Trainer class (one course):

(course-9) “Training of future trainers who give a lecture on the introductory class”

The faculty members in Nagoya University offer most of the courses, but some courses are partially supported by engineers in industry, professors of the other universities and members of Specified Nonprofit Corporations (NPOs). Specifically, NPO TOPPERS Project [4] and NPO SESSAME [5] are primary contributors to NEXCESS.

2.2 The number of applicants

NEXCESS is to be carried out from 2004 to 2009. In 2004 and 2005, we held courses 20 times in total. The introductory course was held 5 times. The intermediate courses were held 4 times, and the advanced courses were held 10 times. The trainer class was held once.

Table 2. Introductory skill map

Technology component		
Information Processing	Information input	Read a switch using a PORT
	Information output	Turn on a LED using a PORT
	Data processing	Use the inside timer of CPU
Platform	Processor	Initialize CPU, Understand memory map
	Basic software	Use the real time operating system
Exploitation technology		
Design	Software design	Understand structured design
Programming and testing	Programming	Do C programming
	Testing	Generate boundary value
Integration	Integration testing	Do test design
Management skill		
Development process management	Process design	Understand development process

We accepted applications through the Web. Table 1 shows the number of applications to each class. The number of applications to every course exceeded the capacity. We selected the students suitable for the course.

The number of embedded software engineers at all the technical levels is insufficient [1]. For example, program managers are insufficient in most of large companies. Designers are insufficient in most companies that perform advanced research and development. Programmers are insufficient in all the companies. As such, most companies need much more embedded software programmers.

In many companies, engineers at the introductory level work on the programming and testing processes (see Figure 2). We assume that the introductory engineers must study structured programming, embedded C programming and testing. These skills are necessary to carry out the programming and testing processes. We have developed the introductory course titled “Fundamentals of the embedded software development technology”. We educate the introductory-level programmers in this course.

Table 1 clearly indicates the strong demand from the industry to education of introductory-level engineers. We open the introductory course several times a year, its capacity is limited due to the limitation of facilities. Therefore, we have decided to develop a new class to train future trainers of embedded software. After completion of the trainer course, they are expected to teach introductory-level embedded software engineers at their companies.

3. INTRODUCTORY COURSE

The trainers should be able to teach at least the introductory course, i.e., “Fundamentals of the embedded software development technology”. This section explains the outline of the introductory course.

3.1 Requirement for application

In extension courses including NEXCESS, students generally have very different backgrounds such as graduated schools, ages, current companies and responsibilities in the companies.

At the introductory course in NEXCESS, exercises in embedded software programming are based on the C language. Therefore, we require applicants to the introductory course to have more than one year of experience in C programming. Applicants who do not satisfy this requirement are not selected if the number of applications exceeds our capacity, and fortunately, this always held so far. After this selection, although students have various backgrounds, at least, we can assume that all the students can write programs in C.

3.2 What to teach

We have defined the skills that an introductory embedded software engineer has to acquire.

An engineer at the introductory level works on the programming and testing processes, not on the design and requirement analysis. An engineer at the introductory level has to do C programming and test design. Of course, it is preferable for him/her to be able to do software design, but we think that software design is one of responsibilities of intermediate-level engineers.

Introductory-level engineers must be able to write embedded software programs. Programming of embedded software is different from that of enterprise- and business-oriented software. They must be able to directly manipulate CPUs and other hardware devices such as UART, timers, ports and so on, without OS or library supports.

Introductory-level engineers write embedded software under a direction of project managers. It is not necessary for the introductory engineers to have the ability of project management, but it is necessary for them to understand the overall software development process.

In Table 2, we have defined the skills that an introductory embedded software engineer has to acquire. The skills, which we have defined, are based on ETSS (Embedded Technology Skill Standards) which was defined by IPA [3]. For NEXCESS, we have modified ETSS.

3.3 How to teach

For engineers working in industry, teaching just theoretical knowledge is not sufficient. Not only knowledge but also practical skills need to be improved. They are expected to utilize the skills acquired during the course in their work in practice. They are not commentators but engineers. The introductory embedded software engineers should write embedded software programs.

In order to improve technical skills for introductory engineers, we have developed a set of exercises as follows.

(1) Documentation exercise

In software development, documentation is one of the most important processes. In this exercise, a student writes design specifications, test specifications, a review report, etc.

(2) Real machine exercise

In this exercise, a student uses a real microcomputer board. The student writes a program and runs it on the microcomputer board. It is effective for a programming exercise.

(3) Discussion exercise

Students discuss in a group. The exercise is effective for code review.

(4) Role playing exercise

In this exercise, a student plays a different role from his/her original role. For example, those who are always programming play a role of a manager. By acting in a different position, a student can obtain a wider scope

(5) Case study exercise

For example, if a student studies the example of the program bugs, he/she will not cause the same failure.

(6) PBL (Project Based Learning) exercise

A student undertakes a small project for education. The student will obtain a wider view of work by experiencing the project from the beginning to the end.

The introductory course is targeting to engineers who take charge of a part of software development processes. Their primary responsibility is programming. Therefore, we have decided to perform real machine exercise. Of course, all the software engineers need to write documentation. Therefore, we have decided to perform documentation exercise, too.

3.4 Real machine exercise

We use a microcomputer board with a 16-bit CPU (see Figure 3). This CPU has 2KB RAM and 64KB flash ROM. This board also has a UART port to connect to a host PC. We can develop a C program on the PC and download it to the target board. The development environment on the PC is an MS-Windows application, which is easy to use (see Figure 4). Therefore, it is suitable for the education purpose.

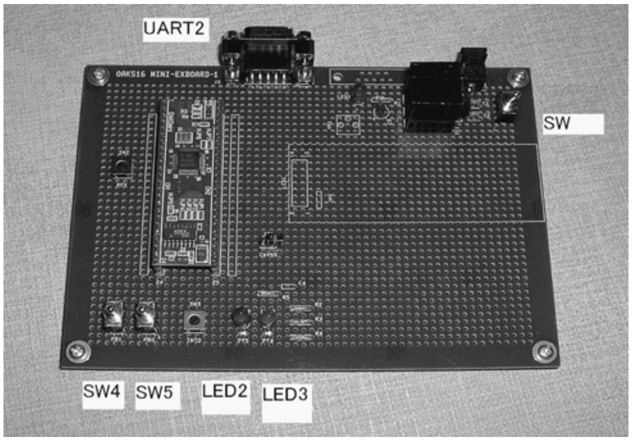
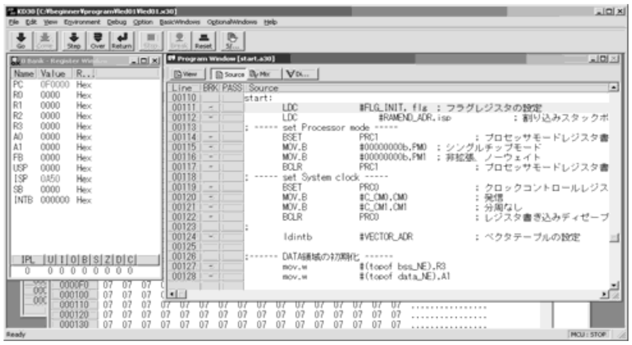


Figure 3. The exercise board


KD30 [C:¥beginner¥program¥led01¥led01.x30]
File Edit View Environment Debug Option BasicWindows OptionalWindows Help
Go Come Step Over Return Stop Break Reset S/...
B Bank - Register Window
Program Window [start.a30]
View Source Mix Di...
Ready
MCU : STOP

```
Name  Value   R...
PC    0F0000  Hex
R0    0000    Hex
R1    0000    Hex
R2    0000    Hex
R3    0000    Hex
A0    0000    Hex
A1    0000    Hex
FB    0000    Hex
USP   0000    Hex
ISP   0A50    Hex
SB    0000    Hex
INTB  000000  Hex

IPL  U I O B S Z D C
 0   0 0 0 0 0 0 0 0
```

```
start:
        LDC     #FLG_INIT, flg  ; フラグレジスタの設定
        LDC        #RAMEND_ADR,isp         ; 割り込みスタックポ
; ----- set Processor mode -----
        BSET    PRC1                   ; プロセッサモードレジスタ書
        MOV.B   #00000000b,PM0  ; シングルチップモード
        MOV.B   #00000000b,PM1  ; 非拡張, ノーウェイト
        BCLR    PRC1                   ; プロセッサモードレジスタ書
; ----- set System clock -----
        BSET    PRC0                   ; クロックコントロールレジス
        MOV.B   #C_CM0,CM0             ; 発信
        MOV.B   #C_CM1,CM1             ; 分周なし
        BCLR    PRC0                   ; レジスタ書き込みディセーブ
;
        ldintb  #VECTOR_ADR            ; ベクタテーブルの設定

;------ DATA領域の初期化 ------
        mov.w   #(topof bss_NE),R3
        mov.w   #(topof data_NE),A1
```

```
0000F0  07  07  07
000100  07  07  07
000110  07  07  07  07  07  07  07  07  07  07  07  07  07  07  07  07  ................
000120  07  07  07  07  07  07  07  07  07  07  07  07  07  07  07  07  ................
000130  07  07  07  07  07  07  07  07  07  07  07  07  07  07  07  07  ................
```

Figure 4. The development environment

This board has switches and LEDs, too. Through this exercise, a student develops a simple countdown timer. The specification of the timer is as follows.

- When switch 5 is turned on, a set period is set to 1 minute and countdown is started.
- After starting the countdown timer, LED2 blinks at intervals of 10 seconds.
- LED2 blinks twice at intervals of 0.25 second.
- If the set period is reached, LED3 blinks for 15 seconds at intervals of 0.25 second.
- When switch 4 is turned on, a set period will be extended for 30 seconds.

Although this is a simple application, it is very effective for students to study embedded programming.

For example, in order to read a switch, the students have to set up a port appropriately. It is necessary to set up the direction of input and output of the port by a special register in the CPU. The students have to read a CPU manual. By reading a circuit diagram, one understands whether a switch is on or off at the time of '0'. In programming of enterprise- or business software, programmers do not have to be aware of such things, but they are very important in embedded software programming

A student also experiences characteristic coding by embedded C programming. For example, assume that in order to reflect the contents written in a port register, the port register needs to be read. An inexperienced engineer may write the following code.

```
void led_reflecte(void){
  unsigned char tmp;
  tmp = *((unsigned char *)(0x3ed));
}
```

However, an optimizing compiler will remove “`tmp`” because “`tmp`” is not referred to after it is written. Therefore, this program does not work correctly. For correct behavior, “`volatile`” needs to be specified as follows.

```
void led_reflecte(void){
  unsigned char tmp;
  tmp =
     *((volatile unsigned char *)(0x3ed));
}
```

In addition, many other skills are acquired through these exercises.

3.5 Documentation exercise

For an introductory engineer, programming is the main mission. However, he/she needs to study the design process, too. This is because a better program can be made if he/she has knowledge on design.

Moreover, he/she needs to consider test specification. If there is capability to test, he can make a better program.

Then, we provide a document exercise in the design and test processes. Students write a 10-page document in the design exercise. They also write a 10-page document in the test exercise.

3.6 Course schedule

A lecture is required before an exercise. If the students do not understand the reading of a CPU manual, they cannot do these exercises. Therefore, after giving a lecture first, they do these exercises.

In NEXCESS, a class starts at 9:30 and finishes at 17:00. An introductory course is four days long. The overall schedule for the four days is as follows where mark ‘*’ indicates an exercise.

The 1st day

> Embedded system, Software development process, Structure design, Structure design exercise*, Check of development environment*

The 2nd day

> Hardware knowledge, Embedded C programming, How to use the board for an exercise*, Embedded C programming exercise*

The 3rd day

> Real time OS, ITRON programming*

The 4th day

> Test process, Test specification*, Down count timer application*

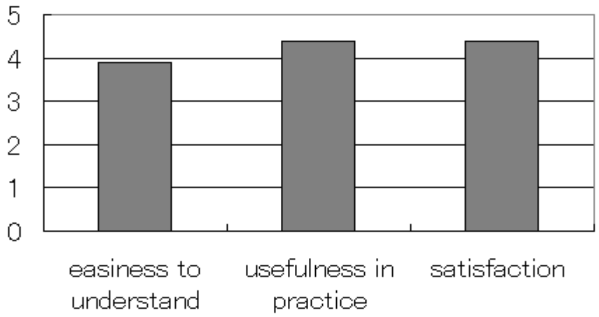


Figure 5. Introductory course evaluation: availability

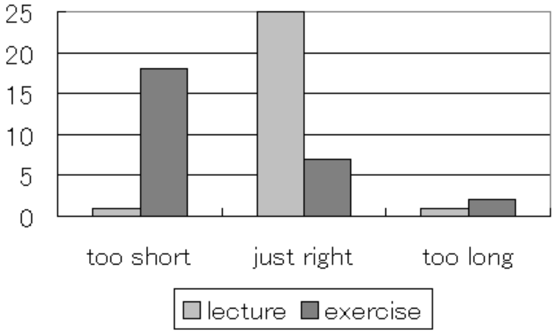


Figure 6. Introductory course evaluation: time

3.7 Evaluation by the students

We ask students to evaluate the course. Figure 5 shows a part of the evaluation results where 5 is the highest. The score of the usefulness in practice is 4.4. This shows that the contents of the introductory course are directly applicable to their work.

Evaluation on time length is shown in Figure 6. Students think that the time of a lecture is just right. However, they think that exercise time is too short.

In most companies, embedded software engineers are too busy to spend many days for the education purpose. Therefore, at NEXCESS, the introductory course is carried out in four days. However, the evaluation results show that four days are not enough. The right length may be about 6 days.

4. THE TRAINER COURSE

In NEXCESS, the capacity of the introductory course is 30, and the course is held two or three times every year. This means that only about 100 introductory engineers per year can be trained, which is not sufficient at all

In 2005, we have developed a new course to train future trainers of introductory engineers. We train 10 trainers per year in the

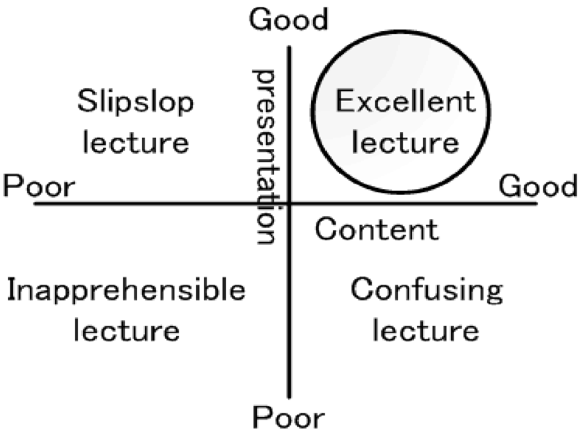


Figure 7. Teaching skill is useful

course. If each trainer trains 20 introductory engineers per year, then 200 introductory engineers are trained per year.

4.1 Course schedule

The trainer course is a little longer than the other courses. The course consists of four stages.

(1) First stage (2 days)

Teaching the education skill

(2) Second stage (4 days)

Introductory class observation

(3) Third stage (4 days)

Practice teaching of introductory class

(4) Fourth stage (1 days)

Review

Students of the trainer course have to understand the technical contents of the introductory course. In addition, they have to acquire educational skills.

4.2 Education skills

What are the education skills? METI (Ministry of Economy Trade and Industry, Japan) has defined education skills as follows [6]. Education skills are categorized to three types (a) to (c) as follows.

(a) Investigate a training trend skill, Project management skill, Leadership skill, Communication skill, Negotiation skill

(b) Planning skill of a training course, Course development skill

(c) Instruction skill

Type (a) skills are general and not specific to education.

Type (b) skills are the planning skills which are not necessary for the trainer course at NEXCESS since we have already developed the introductory course.

We put our focus onto type (c) skill. The instruction skill is the core skill for education. We have further classified the instruction skill into two, i.e., a teaching skill and a coaching skill.

Table 3. Teaching and Coaching

Teaching	Coaching
The educational method to clue up	The educational method to draw forth
It teaches that a student does not know.	It helps for a student to discover.
Required also of individual guidance or class room.	It is required when doing individual guidance.
Capability required as an educator.	Capability required also as a boss.

4.3 Teaching skill

Teaching skill is trainer's capability to give a lecture. The trainer cannot give a lecture without teaching skill, even if teaching materials are prepared. Good presentation as well as good contents is required for a good lecture (see Figure 7). In the trainer course at NEXCESS, we teach the following teaching skill.

- Preparation of a lecture
- How to talk
- How to use a whiteboard
- The method of a question to the student
- How to hear the opinion of a student

We prepare several exercises to train the teaching skill. For example, a student gives a short lecture for several minutes, followed by discussion on the teaching skill.

4.4 Coaching skill

What is the coaching skill? What is the difference between teaching and coaching?

Coaching is an educational method, which grow the capability of students. The difference between coaching and teaching is summarized in a table 3.

Recently, the importance of coaching has been recognized in the business field. In particular, in embedded software industry, it is very important due to the following reasons.

One reason is due to the technical contents to be educated. As for the embedded software industry, technology is rapidly progressing. In such domains, it is very difficult for even leaders or senior engineers to catch up with all of the technology. A leader of a group has to survey the technology together with the other members in his/her group. The leaders may not always have more knowledge on the technology than other members may. Even in that case, however, if the leader has the coaching skill, he/she can guide the other members.

The second reason is concerned with a mentality. In software industry, it is necessary for workers to be highly motivated. The software developed with a high motivation has high productivity and quality. For this reason, a leader needs to manage his/her group members so that they keep a high motivation. Moreover, their mental illness can also be prevented by raising their motivation.

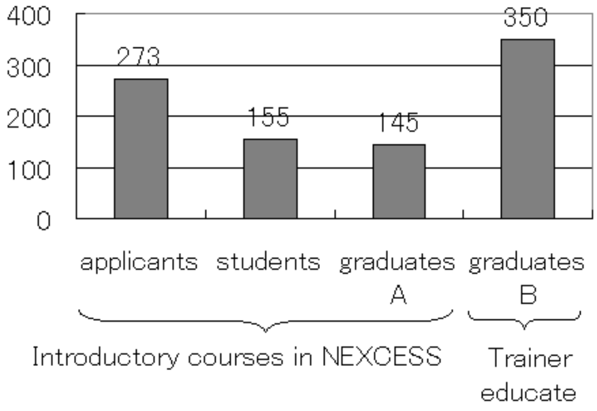


Figure 8. Introductory engineers

In the trainer course at NEXCESS, we educate the following coaching skills.

- How to adjust to the motion of a partner. (Mirroring)
- How to adjust to the way a partner speaks. (Pacing)
- How to listen closely.
- How dose a partner speak about real intention.
- How to make a good action plan.

We have developed several exercises to train the coaching skill. Many managers tend to teach without hearing subordinate's talk. Intentionally mirroring and pacing exercise improve the attitude to listen.

5. RESULT

We held the introductory courses five times in 2004 and 2005, and the trainer course once in 2005.

5.1 The number of introductory engineers

We held five the introductory courses in 2004 and 2005. The number of applicants is 273. Out of them, we selected 155 persons. Since 10 persons were absent, 145 persons successfully completed the course (see Figure 8).

17 persons took the trainer course in 2005. Many of them are working in the educational section of their companies. In their companies, they teach the introductory engineers in 2006. A total of 350 engineers are educated in their report (see Figure 8).

Only 145 persons have been educated through 5 times of introductory engineer courses. However, 350 introductory engineers' education was made by a single trainer course. We can conclude that the trainer course is very effective to educate introductory engineers.

5.2 Evaluation by the trainers

We asked the students of the trainer course to evaluate the course. Figure 9 shows a part of the evaluation results where 5 is the highest. The average score of usefulness in practice is 4.7. This shows that the contents of the trainer course are applicable to their work.

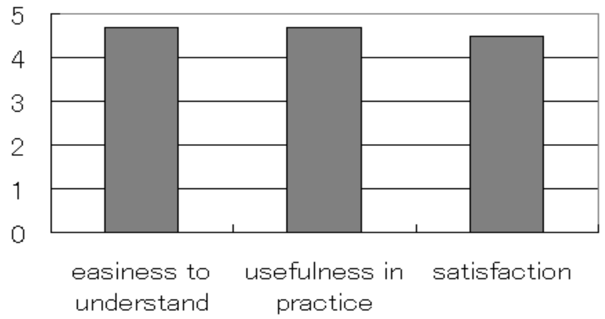


Figure 9. Trainer course Evaluation: availability

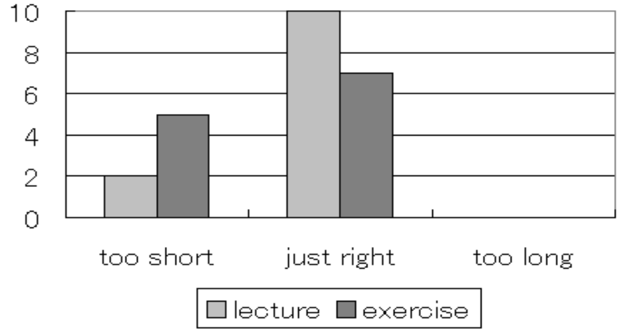


Figure 10. Trainer course Evaluation: time

Evaluation on time length is shown in Figure 10, indicating that the students need longer time.

5.3 Evaluation of the education skill

Especially the student in the trainer course got interested in the coaching skill. The students wrote in the questionnaire as follows.

- I understood the definition of coaching.
- I got to know that not only teaching skill but also coaching skill are required.
- When I tried it, I found that it was difficult.
- I learned many things that I had never experienced or investigated.

Many of the students of the trainer course are working in the educational section of the companies. However, many of them had not sufficiently studied the education skill. In order to raise an educational effect, it is required to improve the education skill. We understood that this course was important.

6. CONCLUSION

Education of embedded software engineers in industry is very important. Nagoya University has started an extension program on embedded software, named NEXCESS. Through the first year of NEXCESS, we have found that introductory engineer training is highly required.

We have developed a trainer course to educate future trainers. Students of the trainer course are expected to train introductory-level embedded software engineers at their companies. Thereby, many introductory engineers can be educated efficiently.

NEXCESS is now in the third year. We are continuing to improve our courses.

ACKNOWLEDGMENTS

NEXCESS is supported in part by Ministry of Education, Culture, Sports, Science and Technology, Japan.

REFERENCES

[1] Ministry of Economy Trade and Industry. *2006 Embedded software actual condition survey,* https://sec.ipa.go.jp/download/200606es.php, 2006.

[2] M.Yamamoto, et. al., “NEXCESS: Nagoya University Extension Courses for Embedded Software Specialists”, Workshop on Embedded Systems Education, Sep. 2005.

[3] Ministry of Economy Trade and Industry. *2006 Embedded Technology Skill Standards,* https://sec.ipa.go.jp/download/200606eb.php, 2006.

[4] NPO TOPPERS Project, http://www.toppers.jp/

[5] NPO SESSAME, http://www.sessame.jp/

[6] Ministry of Economy Trade and Industry. *IT skill standard Ver. 2,* http://www.ipa.go.jp/jinzai/itss/download.html, 2006.