Aspects of the Evaluation and Improvement Process in an Online Programming Course

Case: The ViSCoS program

Sirpa Torvinen

University of Joensuu,
Department of Computer Science

Licentiate thesis

August 2004

Supervisors:
Professor Veijo Meisalo, University of Helsinki
Professor Erkki Sutinen, University of Joensuu

Abstract

The number of distance learning programs has rapidly increased during the past few decades but only few of them are evaluated in terms of functionality. In the ViSCoS (Virtual Studies on Computer Science) program, high school students in rural areas surrounding Joensuu, Finland, are able to do first-year university-level computer science (CS) studies while completing their regular high school studies. Students study mainly over the web using a web-based learning environment and web-based materials. We created the ViSCoS courses using the *CANDLE scheme* by first identifying the most essential principles needed for online course design and then by using electronic tools to meet the authentic learning needs of students. With the *CANDLE scheme*, we have successfully focused our design on the most essential parts of the virtual study process. We have evaluated the ViSCoS programming course through an action research / case study approach using questionnaires, interviews, and through analysis of student feedback, submitted exercises, and exams. Our goal was to minimize the number of students who dropped out of an online programming course in ViSCoS. The guiding principle in our research was to evaluate and improve the distance learning arrangements based on the needs and experiences of students. The time scheduling proved to be the most important factor behind the dropout phenomenon of ViSCoS; careful planning with scheduling is needed not only for students to manage their studies and free time, but also for internal scheduling between difficult and easy learning subjects in a ViSCoS programming course. The main elements in the *CANDLE scheme* (high school, university, electronic tools, and the Internet) proved to be a meaningful and adequate combination for a functional online learning setting. Finally the continuous improvement process with triangular research approaches based on the experiences and feedback given by distance learners, tutors, and supervisors has proved to be a successful way to approach the dropout phenomenon in a ViSCoS programming course. Since the design and implementation of ViSCoS has proved successful, we believe that it will be useful to expand the *CANDLE scheme* and ViSCoS to other academic subjects and educational contexts.

List of publications:

Paper I — Sutinen, E. & *Torvinen, S.*: "The Candle Scheme for Creating an on-line Computer Science Program – Experiences and Vision", *Informatics in Education*, 2003, Vol. 2, 93-102. Reprinted, with permission, from Journal of Informatics in Education.

Paper II — Meisalo, V., Suhonen, J., Sutinen, E. & *Torvinen, S.:* "Formative Evaluation Scheme for a Web-Based Course Design", *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2002)*, Aarhus, Denmark, June 24-26, 2002. ACM Press, 130-134.

Paper III — Meisalo, V., Sutinen, E. & *Torvinen, S.*: "How to improve a virtual programming course?", *The 32nd ASEE/IEEE Frontiers in Education Conference, FIE 2002*, November 6-9, 2002, Boston, MA, USA. CD-ROM.

Paper IV — Meisalo, V., Sutinen, E. & *Torvinen, S.*: "Choosing appropriate methods for evaluating and improving the learning process in distance programming courses", *The 33rd ASEE/IEEE Frontiers in Education Conference, FIE 2003*, November 4-8, 2003, Boulder, CO, USA. CD-ROM.

Preface

In my Master's thesis, I studied the difficulties associated with learning the basics of programming in a virtual learning environment. Through that study process, I found several future research questions which needed to be evaluated. Since one of the most important research questions was related to the dropout phenomenon within ViSCoS (Virtual Studies on Computer Science), I decided to continue in my Licentiate thesis with an investigation of the dropout phenomenon in a Programming 1 course taught at the University of Joensuu. The purpose of this investigation was to reduce the dropout rate by improving virtual learning settings.

This Licentiate thesis is based on the four publications mentioned, which were published from 2002 to 2003. In all of our papers, the authors are listed in alphabetical order, regardless of the role the authors played in producing the paper.

Although my name is listed second on the paper because of the alphabetical listing procedure, I was the main contributor in **Paper I.** Paper I describes the educational design scheme used in creating the ViSCoS program. In this paper, we describe the contents of ViSCoS, the main elements needed in a distance learning setting, and the learning methods used in ViSCoS. In addition, the authors write about their experiences with this online learning project during the years 2000-2002. (Professor Erkki Sutinen was the supervisor for this paper and he wrote the section labeled 'Future Vision'.) Professor Sutinen also contributed to this paper by providing the main ideas for section 3.2. of Paper I. Although the authors discussed the content of paper several times during the progress, the data was collected and the paper was written mainly by me. I also created Figure 1 and Tables 1 - 2.

Paper II describes the dropout phenomenon in the ViSCoS program. In this paper we write about the experiences and results of the first ViSCoS group, the main reasons for students to drop out, and about the most difficult topics in the Programming 1 course. Figures 1 – 3 and Tables 1 – 2 have been made by me and I have done the research work behind the data by creating and posting the questionnaires and by analyzing the data after the questionnaires have been returned.

The actual writing of the paper was done in collaboration with MSc Jarkko Suhonen. Although the authors worked together during the writing process, there were some sections that were written individually. Chapters 1 - 2 were mainly produced by Jarkko Suhonen, chapter 3 by professor Meisalo, chapters 4 - 5 by me, and chapter 6 by professor Sutinen. On behalf of the authors, I presented the research results at the 7th conference on Innovation and Technology in Computer Science Education (ITiCSE 2002) in Aarhus, Denmark.

Paper III describes the dropout phenomenon in the ViSCoS program. I was the main author of this paper. In Paper III, we describe the evaluation process and improvements made in the Programming 1 course. I created Tables 1 - 4 and Figure 1. I wrote the paper based on discussions with the other authors - professor Meisalo and professor Sutinen. Professor Meisalo and Sutinen collaborated on the conclusion. I conducted all of the research work behind Paper III. On behalf of the authors, I presented the paper at the 32nd Frontiers in Education Conference (FIE 2002) in Boston, MA, USA.

Paper IV describes the research methods used in the ViSCOS project. Although a collaborative process was used to produce Paper IV, I was the main author. Professor Meisalo was responsible, especially, for the chapters concerning triangulation. I created Figures 1 - 3 and Tables 1 - 3 and completed the research work behind Paper IV. On behalf of the authors, professor Meisalo and I presented the results of the paper at the 33rd Frontiers in Education Conference (FIE 2003) in Boulder, CO, USA.

TABLE OF CONTENTS

1. INTRODUCTION 1

1.1. BACKGROUND 2
1.2. NEEDS FOR RESEARCH 5
1.3. RESEARCH QUESTIONS 6

2. AN OVERVIEW OF THE ViSCoS PROGRAM 9

2.1. CONTENT AND SCHEDULE 9
2.2. ELEMENTS NEEDED IN VISCOS 10
2.3. THE TEACHING AND LEARNING METHODS USED 11

3. METHODOLOGY 15

3.1. RESEARCH PARADIGMS 15
3.2. RESEARCH STRATEGIES 17
3.3. THE RESEARCH STRATEGIES USED 19
3.4. THE RESEARCH SOURCES USED 21

4. THE *CANDLE SCHEME* 23

4.1. THE CREATION OF *CANDLE* 23
4.2. THE MAIN ELEMENTS IN *CANDLE* 23
4.3. THE *CANDLE SCHEME* FROM THE METHODOLOGICAL POINT OF VIEW 27

5. THE DROPOUT PHENOMENON 30

5.1. THE DROPOUT PHENOMENON IN ONLINE SETTINGS AND IN CS 30
5.2. THE DROPOUT PHENOMENON IN THE VISCOS PROGRAM 31
5.3. THE DROPOUT PHENOMENON FROM THE METHODOLOGICAL POINT OF VIEW 32

6. IMPROVEMENTS 34

6.1. EVALUATION AND IMPROVEMENTS 34
6.2. THE CONTINUOUS IMPROVEMENT PROCESS IN VISCOS 36
6.3. IMPROVEMENTS FROM THE METHODOLOGICAL POINT OF VIEW 38

7. CONCLUDING REMARKS AND FUTURE WORK 41

ACKNOWLEDGEMENTS 46
REFERENCES 47

Appendix I: **The questionnaire in Finnish**
Appendix II: **The questionnaire in English**
Appendix III: **Glossary**

1. INTRODUCTION

Distance learning programs have rapidly increased during the past few decades. Unfortunately, the most critical issue is far too often that institutions forget to evaluate and improve the quality of distance learning materials because the institutions are simply too urgent to produce new courses using new techniques without any kind of web-pedagogical design process (Klemm, 2001; Dahanarajan, 2001). Nevertherless, Kleinman & Entin (2002) found that administrators of many institutes of higher education think of distance learning courses as a cost-effective method of delivery since students do not need to be located in the same time or space with their instructors or with each other. In their research, which compared in-class and distance-learning courses, Kleinman & Entin theorized that a lack of face-to-face interaction between students and teachers, or between students, might be harmful to learning. Despite the lack of face-to-face interaction, they noticed that the attitudes of distance learners are generally positive. Additionally, they found that distance learning students were even more positive about the course than traditional on-site students.

Glass et al. (2004) examined the computing research in order to better understand where our field has been and to consider where it may be going in future. Glass et al. analyzed how different the topics upon which researchers of our field do research are and how similar (or dissimilar) the research approaches and research methods between the main three subdivisions of computing - computer science, software engineering, and information systems - are. They emphasized that the field of computing still seems to be in a state of transition.

Farrell et al. (1999) and Watanabe et al. (1999) have criticized the poor quality of distance courses. Both Farrell et al. (1999) and Watanabe et al. (1999) mention the fact that all too often institutions produce new teaching materials based on trends without considering whether the new materials really give any kind of benefit compared to the traditional, face-to-face lecture model. If you hasten to produce web-based learning setting without careful planning, you might not notice all the wonderful possibilities that web-based learning environments and learning materials could provide you.

Braun (1993) in his study of effectiveness of educational technology remarks that the roles of teachers and students are changing radically because of the trend from traditional face-to-face learning settings towards technology based (on-line) learning settings. The teaching environments of the future will be much more flexible than the traditional ones and they will meet each student's abilities and needs in a more flexible way. The teacher is seen in the technology based learning environment as a facilitator and a guide and students are seen as individuals with unique learning styles. Braun emphasizes that this new technology might open totally new worlds to students and make learning exciting and meaningful. Technology is being developed to prepare the way for new pedagogical progress and vice versa: new pedagogical approaches will generate new technologies for educational technology.

In addition, Rountree et al. (2004) discuss the predictors for success in studying CS at the university. They noticed that universities nowadays are no longer able to recruit the highest performing high school students and that the motivation of applicants might have more to do with getting a well-paying job than personal development. Rountree et al. also found that pre-university level CS studies in many countries are not readily available. Greening (1998) discusses the importance of the perceptions of so called 'potential students' and emphasizes the importance of presenting students with the realities of computing in order to minimize their misconceptions, which are hypothesized to lead to dropout. The ViSCoS[1] program is designed to give high school students a chance to try university level CS studies during their regular high school studies. If they decide enter the university after their high school graduation, they will already have some familiarity with university-level CS studies.

1.1. Background

The need for our distance learning project in computer science (CS), the ViSCoS program (**Vi**rtual **S**tudies on **Co**mputer **S**cience), arose out of the following two factors; first of all, since the Finnish Ministry of Education was expected to fund the Finnish Virtual University (FVU) project for the years 2000-2003 (FVU, 2004), we were encouraged to launch our project in the year 2000. In the framework of the FVU, ViSCoS was born, as if on cue, and the financial support given by Finnish Ministry of Education made it possible to start the development project. The majority of funding was provided by the Department of Computer Science at the University of Joensuu.

---

[1] More about ViSCoS in URL: http://cs.joensuu.fi/viscos/?lang=eng

Secondly, the need for higher quality CS teaching in local high schools became evident when teachers began to contact the University of Joensuu requesting that CS studies be made available for their students. We decided to meet these two needs by creating the virtual CS program, ViSCoS. While carrying out our virtual CS teaching program, we realized that ViSCoS could be an efficient student recruitment tool for our department. Since industry needs were high at that time and plenty of our CS students moved on to jobs even before they graduated, we wanted to encourage students to apply to our department. We hypothesized that having the first year of CS studies done before entering the university proper would motivate students to complete the remainder of their university studies.

We started our development process in close cooperation with high school teachers and created the *CANDLE scheme* (Paper I) to guide us in the creation process of the ViSCoS program. Although the *CANDLE scheme* has some context dependent elements, like the Finnish educational system or the target group of the students, we believe that *CANDLE* can be generalized to other environments, student groups, or other cultural contexts by making minor modifications.

In the ViSCoS program, high school students around the rural area of our university study a university level introductory computer science curriculum (22.5 European Credit Transfer System credits, cps, equals to about 600 hours study work) over the web. High school students study the first-year university-level CS courses simultaneously with their regular high school studies. Students' ages extend from 16 to 19 years. There are no prerequisite tests, so every high school student who is interested in participating, can participate.

The ViSCoS curriculum includes eight different courses that give students basic knowledge and skills in CS and computing. The content and schedule of the ViSCoS courses is shown in Table 1.

Table 1: Content and schedule of the ViSCoS courses

Title of the course	ECTS	Contents	Scheduled
Introduction to Computer Science	3 cps	General knowledge about IT and computing. Practical skills of using word processing and spreadsheet programs, basics of Unix.	Semester 1
Programming 1	3 cps	The idea of algorithms, basic structures of programming using Java (data types, logical operations and logical operands, variables, if-statements, loops, arrays and methods)	Semester 1
Programming 2	3 cps	The basics of object-oriented programming (classes, objects, graphical programming and event handling).	Semester 2
Research fields of Computer Science	3 cps	Introduction to a selection of research fields in computer science.	Semester 2
Programming project	3 cps	Independent work containing software design and programming.	Semester 2 and Semester 3
Introduction to Algorithms	3 cps	An overview of the central issues of computer science, such as algorithms, computation, and data structures.	Semester 3
Hardware, computer architecture and operating system	3 cps	An overview of issues such as architecture of computers, parsers, system software, databases, and information systems.	Semester 3
Introduction to the Ethics of computing	1.5 cps	General knowledge and practice in the ethics of computing.	Semester 3

During the first year (2000) we had nine different courses, but at the beginning of the second year (2001) we re-organized the curriculum so that the ViSCoS program consisted of the eight courses presented in Table 1. The scale has always been 22.5 ECTS credits and the content of the curriculum has been very much the same.

Students study independently over the web, but each high school has one *tutor* (referred to as a *tutor teacher* in the papers) who helps students, especially, at the beginning of distance studies. The main responsibility for teaching in the courses is taken by *supervisors* who are staff members at the Department of Computer Science at the University of Joensuu.

Earlier studies have shown that CS applicants might have the wrong reasons for applying to the university: students far too often just want to have a quick route to a job via CS studies because they believe that CS enhances general career prospects - even without any personal interest in the subject matter (Greening, 1998; Rountree et al., 2004). Rountree et al. (2004)

also emphasize that in many countries, pre-university CS is not readily available, and even if it is, the content and delivery is unlikely to correspond well with the university curriculum. With the ViSCoS program, high school students get pre-university CS studies that have been approved by university faculty. Based on our observations, students who work through the entire curriculum of ViSCoS – and even those who drop out – do not have any serious misconceptions about computing, and when (or if) they apply to the department of CS, they already know in advance what it is like to study computing at the university. We also agree with Rountree et al. (2004) that students' earlier (pre - high school) experiences with computing, like experiences with home computers or self-taught computing skills, might give them misconceptions about computing. It has been quite a challenging task for us to get students to understand the main principles of computing during ViSCoS in light of the misconceptions that many of our students have had prior to formal CS training.

1.2. Needs for research

Earlier findings show that dropout rates in university-level distance learning studies are, in general, higher than those in conventional university studies (Cornell & Martin, 1997). Dropout rates vary depending on the distance education system adopted and on the subject being studied. For example, in Europe, the dropout rate is between 20% to 30%; in Asian countries, the rate is about 50% (Xenos et al., 2002). In addition, Cornell & Martin (1997) found that students who have passed at least one distance course will probably pass others in the future.

In 1989, Denning et al, in the final report of the task force on the core of computer science (Denning et al., 1989), found that the view that 'computer science is the same as programming' was especially strong in most of the CS curricula. Usually programming was taught in introductory courses, technology was taught in core courses, and science was taught in elective courses. Although programming is part of the standard practice of the discipline, that does not necessarily mean that the curriculum should be based on programming or that introductory courses should be programming courses (Denning et al., 1989). Denning (2004) discovered that the first course in CS programs that use the Computing Curricula 2001 (CC2201) typically is centered around learning object-oriented languages (like Java or C++). Denning et al. (1989) emphasize that many novices found the language details complex and the course itself very difficult. Denning (2004) also reports high dropout rates during the first CS courses - from 35 to 50 %. Greening (1998) noticed that student dropouts in first year CS courses are typically high. Renaud et al. (2001) emphasize in their report that teaching programming is never a simple task; teaching it at distance is especially challenging. In ViSCoS we have combined these two demanding areas: the distance learning setting and CS teaching. Our experiences from ViSCoS, which are described in Paper III, are quite similar to Cornell & Martin's (1997) and Denning's (2004). In our research, we focus on evaluating and improving our first programming course for two reasons. Firstly, earlier findings of poor quality distance learning programs (Farrell et al, 1999; Watanabe et al., 1999) encouraged us to create a new program in a unique way. We wanted to create a high-quality, online CS program based on the needs of students and carefully measure its rate of success. This led us to create a formative scheme for evaluating and improving virtual learning settings (Papers II and III).

Secondly, we focused on evaluating and improving, in particular, our Programming 1 course since the difficulties that novice programmers experience, even in face-to-face learning environments, are so common (Ben-Bassat Levy et al., 2003; Wiedenbeck et al., 2004). In ViSCoS, we have determined that the programming courses have been the most difficult since the highest dropout rates occur there. In addition, we have noticed that students who passed Programming 1 in the Semester 1 and Programming 2 in the Semester 2 (see Table 1 in page 4), are likely to pass all the rest of the courses as well. Therefore, the importance of succeeding in these two programming courses is so significant that the rigorous evaluation and improvement of these courses is warranted.

1.3. Research questions

Our research work is focused on the dropout phenomenon in the ViSCoS Programming 1 course, which has given us access to quite a rich variety of research subjects. In the papers related to this thesis, we concentrate on the following three research questions:

The first question is a constructive question:

1) *What kind of model is suitable for creating a functional online CS program?*

The second question is a descriptive one:

2) *What are the characteristics of the dropout phenomenon in the ViSCoS program?*

And finally the third question is an analytical question:

3) *How do we improve the Programming 1 course in order to minimize the number of dropouts?*

The work concerning research question 1 is described in Paper I. In chapter 4 on page 23 of this thesis, we present the creation process of the *CANDLE scheme* and the main elements related to this model.

The work related to research question 2 is described in Paper II. In chapter 5 on page 30 of this thesis, we describe the dropout phenomenon in the ViSCoS program and consider the study process, especially, from the methodological point of view.

Finally in chapter 6 on page 34 of this thesis, we identify the improvements made in the years 2000-2002 to minimize the number of dropouts in the ViSCoS programming courses (Papers III and IV). The research work related to the third research question is mainly described in Paper III. Paper IV more or less describes the methodological issues in our research.

The relationships between the research questions and the papers are illustrated in Figure 1.

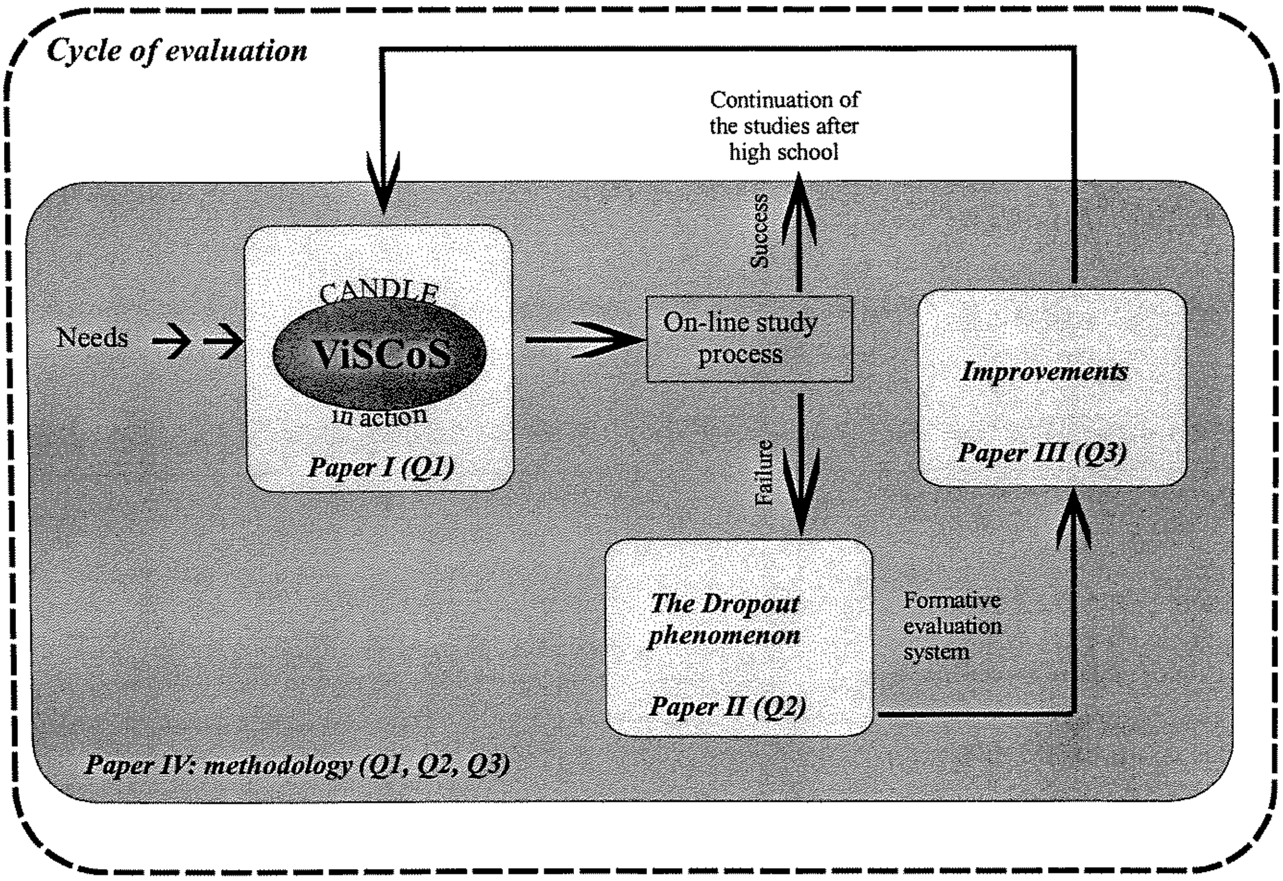


Figure 1. Relationships between papers and research questions

In Figure 1 the notations Q1, Q2 and Q3 correspond with research questions 1 – 3 as follows: notation Q1 corresponds with research question 1, notation Q2 corresponds with research question 2, and Q3 corresponds with research question 3.

2. AN OVERVIEW OF THE ViSCoS PROGRAM

The ViSCoS (Virtual Studies on Computer Science) program has been developed as a part of a three-year project (2001-2003) to establish the Finnish Virtual University (FVU, 2004). The FVU is a project organized to promote and develop a network of Finnish universities. The Eastern Finland Virtual University Network project, (ISVY, 2004) is one of the regional projects of the FVU. The ISVY project partners are the University of Joensuu, which coordinates the project, the University of Kuopio, and Lappeenranta University of Technology which all are located in the eastern part of Finland. In the fall of 2000, the Department of Computer Science at the University of Joensuu began offering, within the framework of the ISVY / FVU, a university level Introductory Computer Science Course to high school students in the rural regions surrounding Joensuu. The following fall, we extended this teaching experiment to the rural regions of the neighboring province as well. One of the main goals of our ViSCoS project is to develop new computer science education methods (Haataja et al., 2001). In this chapter we generally describe the ViSCoS program, we look at what kinds of elements are needed in ViSCoS and, finally, we present the teaching and learning methods used in ViSCoS.

2.1. Content and schedule

In the ViSCoS program, students study independently over the web using the WebCT learning environment. Almost 98% of the 600 hours of ViSCoS study time is delivered via the Internet.

During the 1.5-year study process, ViSCoS students visit the campus area four times as follows:

1. at the beginning of the study process for an orientation meeting where students get
    - a short presentation of the content and schedule of the ViSCoS curriculum,
    - a short introduction to the learning environment and how to use it,
    - a short presentation of web-based materials and how to use them,
    - a short introduction on the use of electronic tools like Jeliot 2000,
    - instructions on how to learn at distance,

- a guide for distance studies, and
- a guide for usage of the learning environment;

2.-3. for two programming exams and informational meetings after these exams, (during information meetings students get further information concerning the courses starting in the semester after the programming courses), and

4. for a graduation ceremony after 1.5 years of study.

In addition, we have a summer school period where students have lectures, are given teamwork assignments, and are asked to give presentations. Summer school, at the beginning of the second fall semester, lasts for one week.

The content and schedule of the ViSCoS courses are presented in Table 1 in Paper I, and in Table 1 on page 4 in this document.

2.2. Elements needed in ViSCoS

The main purpose of ViSCoS is to provide equitable access for high school students to university-level CS studies regardless of the location of the high school. Especially for rural small high schools, the importance of this program is significant. The opportunity to pursue university-level studies concurrently with regular high school studies in rural high schools will increase the attraction of rural high schools and enable them to compete with the high schools located near the university. It is generally known in Joensuu that since the perceived desirability of Joensuu's urban secondary schools is high, rural students often apply to Joensuu's urban high schools rather than their local high schools.

In ViSCoS, the following four main elements are needed:

1) **High school**: High school students study university-level CS studies over the web. In each high school, we have one *tutor* who helps students, especially at the beginning of online study process, with technical problems and with the learning environment.

2) **University**: The University of Joensuu offers the courses in the ViSCoS curriculum and provides *supervisors* (staff members of department of CS) who are responsible for all of the teaching and who provide the support needed when learning difficulties arise.
3) **Electronic tools**: Students study in a virtual learning environment with web-based learning materials. During the ViSCoS curriculum students use some optional electronic tools like Jeliot 2000, LEAP, or Ethicsar (Jeliot, 2004; Suhonen & Sutinen, 2003; Jetsu et al., 2004).
4) **Internet**: Since ViSCoS is an online study program, almost all of the studies are carried out *over the web* using different kinds of electronic tools. In order to study ViSCoS courses, students need only a computer with a web-browser and an Internet connection.

The design platform used to create ViSCoS is called the *CANDLE scheme* (Paper I).

2.3. The teaching and learning methods used

Designing functional and appropriate learning methods or tools for the distance learning context is a challenging task. The ViSCoS materials are composed mainly in HTML. The main goal of the web material was to bring the most essential parts of the course to the learners to quickly show them the structure of courses. Additionally, the web materials in ViSCoS include examples, optional exercises, visualizations using MacroMedia Flash 4.0 or Jeliot2000 (Jeliot, 2004). These materials create opportunities for interactive experimentation.

Ellis et al. (1999), Cordani & Tucker (1998), and Watanabe et al. (1999) have discussed the difficulties in selecting tools which are appropriate for use in higher education distance learning programs. Web-based learning environments are suitable for meeting students' individual learning needs because web-based learning environments are able to utilize a rich assortment of learning methods and tools. In the most optimal circumstances, learning environments should be able to provide a collection of different types of learning tools so that students can select the most appropriate method for their individual learning needs.

In virtual learning environments, the role of the student in the study process will be even more important than in off-line situations (Lawhead et al, 1997). In the planning process for functional distance learning courses, the most difficult task is guaranteeing adequate support for students who face difficulties during the study process (Watanabe et al., 1999). When you decide what kinds of tools you will need in your learning environment, you also have to take into account the possibilities for educationally supportive tools and methods.

Since students are individuals, students' personal features have a significant influence on the ways that they learn, the ways that they think, and on the ways that researchers understand theories of learning (Lee, 2002). Appropriately considering individual student differences makes the design of distance education courses an interesting and challenging task. PISA's report (The Programme for International Student Assesment) illustrates the differences within OECD countries (PISA, 2004). It was quite interesting to noticed in PISA's report that students did not necessarily perform any better at all in learning the subject, although they might have higher interest in reading than in general. On the other hand, in Finland reading interest and performance both were high overall (PISA, 2004). PISA also reported that the learning strengths were most highly concentrated in Finland where about 50% of the students were characterised as either the strongest or the weakest learners.

Chute et al. (1997) describes the student as the central point in a distance-learning environment. The student is never alone in a distance-learning environment; the student is connected via the Internet to other students and/or supervisors. In addition, the student might be connected to experts in the related subject, to virtual libraries, and given access to other methods of support. For example, the supporting methods can be technically-related (provided by PC advisors) or academically-related (provided by supervisors or experts in a specific area).

In ViSCoS we have taken care of the needs of students in the following two ways:

1) We have used *a variety of teaching methods, tools, and strategies*. These include using printed materials and web-based materials, portfolios, learning diaries, face-to-face team work, virtual team work, and individual work. (See Table 2 in Paper I). Using a variety of strategies accommodates a variety of learners and their preferred learning styles (Gardener, 2004).

2) We have offered *different kinds of learning tools* for students. Students can use visualization tools like Jeliot2000 in order to test their own programming codes. We have also created plenty of animations, visualizations, and interactive applets for our web-material in order to help students better understand the critical concepts in CS through visualization and experimentation.

In the ViSCoS program, half of the courses (Introduction to Computer Science, Programming 1, Programming 2, Introduction of Algorithms, and Hardware, computer architecture and operating system courses) have the following structure:

1) students study using both printed and web-based materials,
2) they submit weekly assignments, and
3) learning outcomes are evaluated by exam at the end of the course.

After students submit weekly exercises (four or five tasks), supervisors check the exercises and give comments. They are also available to give students detailed advice as needed. We have used interactive applets, non-interactive applets, and Flash animations in order to visualize the topic to be learned. We have also successfully used a program animation program, Jeliot2000 (Jeliot, 2004), in order to enhance the learning process in the Programming 1 course.

In addition to the tools mentioned above, we have also used *new tools* created at the Department of CS at the University of Joensuu.

1) **PPA / LEAP:** In the spring of 2003, we began using web-based software called a Problem Processing Assistant (PPA, later LEAP) which was developed to give online support to students for programming projects conducted in Semester 2. LEAP helps students through the entire programming process including conceptual and technical design, implementation, and software testing (Suhonen & Sutinen, 2003).

2) **Ethicsar:** In the introduction to the Ethics of computing course, we use a web-based tool called Ethicsar (Jetsu et al., 2004) in order to guide students in ethical issues related to computing and to support their individual writing processes. In that course, students

have to write an essay where they consider the ethical problems in some of the current topics in the area of CS, like piracy, viruses, or copyright issues. Students also are asked to create a learning portfolio where they reflect on the development of their own ethical viewpoints during the course.

During the 1.5-year study process, students have a one-week summer school session on campus (before Semester 3). During that week, we provide lectures and plenty of opportunities for team work. After the summer school is over, web-based courses continue.

We have used virtual team work in one of our courses by using team tools in our virtual learning environment. In this course, members of the virtual team, who are from different high schools, collaboratively produce a small-scale review of some of the research fields in CS at the University of Joensuu. In their reviews, students examine the newest research topics in CS. They are expected to contact researchers at the University of Joensuu to get more detailed information. Table 2 in Paper I shows the learning methods used in the ViSCoS program.

3. METHODOLOGY

In this chapter, we briefly describe the different kinds of research methods available in educational technology and social research that are relevant to technology education. In the following discussion of research methodologies, we discuss three aspects of research methodology - (a) research paradigms, (b) research strategies, and (c) methods for data collection. *Research paradigms* refer to how researchers are going to use the research; research methods are closely related to research paradigms. After we discuss paradigms, we describe several different *strategies* that researchers can use to carry out their research. Next, we discuss the *methods for data collection* that are available to social science researchers. After a general presentation of research methods and strategies, we describe the research strategies and methods used in our study.

3.1. Research paradigms

The main problem in educational technology research has been to find a suitable combination of methodologies since education technology research is neither pure CS nor pure educational research. We have found that the most meaningful methodology discussions revolve around the possibilities and advantages of different research paradigms.

Tashakkori & Teddlie (1998) compare four important paradigms - *positivism, postpositivism, pragmatism,* and *constructivism* - used in the social and behavioral sciences. They report that in the positivistic and postpositivistic approaches, researchers use mainly quantitative methods in their research; in the constructivist approach, researchers use only qualitative methods; and in the pragmatic approach, researchers use both quantitative and qualitative research methods.

Epistemologically, positivism is associated with an objective viewpoint and constructivism is associated with a subjective viewpoint. In pragmatism, both the objective and subjective points of view are used as the epistemological basis. Under the postpositivistic research paradigm, researchers deem their findings objectively "true". Tashakkori & Teddlie (1998) report that in positivism, inquiry is value-free. In postpositivism, values are involved in the inquiry process, but those values are supposed to be held in check by the researcher. In pragmatism, values play a larger role in the process of inquiry. In constructivism, inquiry is totally value-bound.

As we can see, there are quite interesting differences between these four paradigms. For example, the use of quantitative and qualitative methods strongly varies from one paradigm to another. For example, it varies from purely quantitative methods in positivism to purely qualitative methods in constructivism. The decisions regarding the use of either qualitative or quantitative methods, or both methods, depend on the research goals and the phase of the research cycle that is taking place (Tashakkori & Teddlie, 1998).

We were also interested in outlining the time periods for the different kinds of methodologies. We collected an outline of the evolution of methodological approaches based on Tashakkori & Teddlie (1998) in Figure 2.

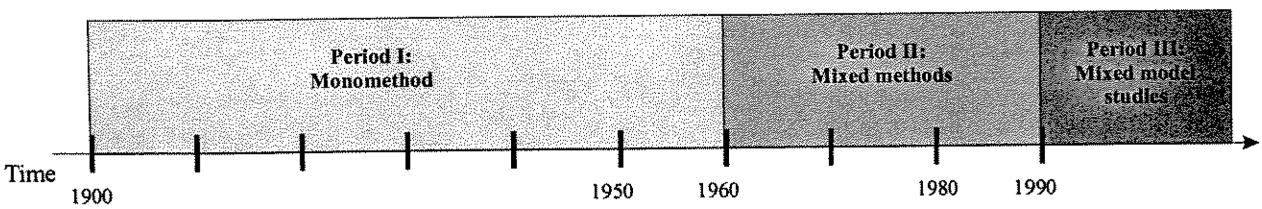


Figure 2. Evaluation of methodological approaches based on Tashakkori & Teddlie (1998)

As we can see from Figure 2, from the beginning of 1900 until to end of the 1950s, the methodological period is referred to as the *Monomethod* or the *Purist Era* (Tashakkori & Teddlie, 1998). During period I, researchers either used only a single qualitative or quantitative data source or they used multiple data sources within one paradigm. During the next four decades, researchers used qualitative and quantitative methods sequentially (e.g., using two-phase sequential studies) or used both methods in parallel. Pediod II is called the *Mixed-methods period* (Tashakkori & Teddlie, 1998). In the Mixed-methods period, qualitative and quantitative methods are used either equally or take on dominant and subordinate roles in the research design.

The third period, starting circa the 1990s, is the period of *Mixed model studies*. The characteristics of this period are that researchers choose a priori whether they will use qualitative or quantitative methods. If they decide to use only *one method for each stage* of the study, the research is called a "single application within stage study". In the "multiple applications within stage" study, *both qualitative and quantitative approaches* have to appear in *at least one stage of the study.*

Triangulation is commonly used in educational research. The different types of triangulation are *data triangulation, investigator triangulation, theory triangulation, or methodological triangulation* (Tashakkori & Teddlie, 1998). In data triangulation, a researcher uses a variety data of sources in order to answer the research questions. In investigator triangulation, several different researchers study the same research problem; each investigator's results are compared. In theory triangulation, a researcher uses multiple perspectives to interpret the results of a study; in methodological triangulation, a researcher uses multiple methods to study a research problem. In this study, we used both data and methodological triangulation in order to get a full picture of the dropout phenomenon.

3.2. Research strategies

Denscombe (1998) describes the following five research strategies available in social research - surveys, case studies, experiments, action research, and ethnography. We have collected a brief description, the types of sources, and the main benefits and disadvantages of each of these research strategies in Table 2.

In many research situations, the choice between different strategies is obvious, but in some cases, the choice might not be clear at all. As we can see in Table 2, since the chosen research strategy plays quite a significant role in the research project, researchers should have adequate knowledge about the focus of the research and about the target group in order to choose the appropriate research strategy before the study actually starts.

Table 2: Research strategies

Research strategy	Short description	Types of sources	Benefits/ Disadvantages
Survey	The idea is to get wide and inclusive coverage (breadth of view). Empirical research: to look at how things are going at a specific point in time.	Data collection using some of the following methods: postal questionnaires, face-to-face interviews, telephone interviews, documents or observations. Sampling after data collection.	Benefits: empirical data, surveys lend themselves to quantitative data, cost and time. Disadvantages: tendency towards empiricism, detail and depth of the data, accuracy and honesty of responses.
Case study	Focus on one instance (or only a few instances) of a particular phenomenon. Especially used in small-scale social research. In-depth study where researchers focus on individual instances instead of on a wide spectrum.	Multiple sources rather than one research method: postal questionnaires, face-to-face interviews, telephone interviews, documents and / or observations.	Benefits: focused on specific situations, multiple methods and multiple sources (triangulation), no pressure on the researcher to impose controls or to change circumstances, concentrates effort on one research site. Disadvantages: credibility of generalizations, often perceived as producing "soft" data (e.g., not based on statistical methods), boundaries of the case can prove difficult to define in an absolute and clear-cut fashion, access to case study settings, presence of the researcher might lead to an observer effect.
Experiment	Laboratory-like settings where researchers control the factors and observe their effect in detail. Both a test group (experimental group) and control group are needed.	Observations and measuring.	Benefits: repeatable, precision, convenience. Disadvantages: deception and ethics, artificial settings, representation of research subjects, control of relevant variables.
Action research	Real-world problems and issues. Typically not only a researcher but also practitioners are active in generating the knowledge from the research. Changes and professional self-development during the study process based on the knowledge received. Action research is a cyclical process where research involves a feedback loop.	Cyclical process with the following five steps: professional practice, critical reflection, research, strategic planning and action Multiple methods (e.g. observation, interviews, questionnaires, documents).	Benefits: research dealt with practical problems in a positive way feeding the results back into practice, personal benefits for practitioner, continuous cycle of development and change via on-site research, participation for practitioners. Disadvantages: involvement of the practitioner might limit the scope and the scale of research, setting for the research does not usually allow for the variables to be manipulated or for controls, ethical and permissible features, ownership of the research process might be problematic, at the beginning of research project the benefits for practitioners might not be very clear, researchers are unlikely to be detached and impartial in their approaches.

Ethnography	Researchers spend time in the field among the people whose lives and culture are being studied. Researchers study everyday life and try to understand how the members of the group / culture see their world, how they understand things, and in what way they perceive their reality.	Participant observation, lifestyles and meanings, comparison and contrast and / or stories.	Benefits: direct observation, empirical, links with theory, detailed data, holistic point of view, contrast and comparison between different groups or cultures, self-awareness, ecological validity. Disadvantages: tensions with the approach, stand-alone descriptions, story telling, reliability, ethics, access, insider knowledge.

3.3. The research strategies used

Glass et al. (2004) analyzed the research in field of computing by examining the topics, methods, and approaches used in the research of the following three most common subdivisions of computing: computer science (CS), software engineering (SE) and information systems (IS). The CS subdivision subsumes my research field (educational technology). The most interesting message in the Glass et al. report is that each of the fields (CS, SE and IS) has its own set of preferred research approaches and methods which do not necessarily command the respect of the other disciplines.

Based on the Glass et al. (2004) report, researchers in CS mainly expect to produce new things. Although CS researchers do not base their theories on research from other disciplines, their work is largely performed within the rules and practices of mathematics whereas, for example, SE researchers emphasize that their work is theory-based. Glass et al. (2004) have also categorized the subject matter of research into the nine different topics, the research approaches used in CS, SE and IS research into three different categories (descriptive, evaluative and formulative), and they also compared the use of different kinds of research methods in each of these subdivisions.

When we started our dropout studies, we had had to decide what kinds of research strategies and methods we were going to use in our research. In our case, we decided that we would receive the greatest benefits using both the case study and the action research types of approaches.

In educational and social research, *action research* is very common; whereas, in computing research, it is hardly used (Denscombe, 1998; Glass et al., 2004). Action research can be described as a family of research methodologies, which pursue action (or change) and research (or understanding) at the same time (Denscombe, 1998). In ViSCoS, we obviously have taken action and done research simultaneously; we have presented the loop of evaluation and improvement process in the ViSCoS, in Figure 1 in Paper IV and in Figure 4 in chapter 6 (page 36) of this thesis.

The *case study* focuses on one instance, or only a few instances, of a particular phenomenon (Denscombe, 1998). In our study, we focus on the dropout phenomenon in ViSCoS in a Programming 1 course which has proved to be one of the most difficult courses. A case study approach was used in the ViSCoS dropout study since it qualifies as small-scale social research that focuses on individual instances instead of on a wide spectrum of instances. By using a case study approach, we believe we got more in-depth knowledge about the dropout phenomenon than we otherwise would have using other research methods.

In the case study approach, the use of multiple data collection methods is recommended (Denscombe, 1998). In our case, the reasons to use both quantitative and qualitative methods and triangulation were quite obvious because we wanted to strengthen the validity of our findings. In Table 1 of Paper IV we describe, in general, the typical differences between quantitative and qualitative approaches. We collected the quantitative and qualitative methods used in our study in Table 2 of Paper IV.

In Glass et al.'s (2004) categorization of research approaches (descriptive, evaluative and formulative) report, they noticed that 79.1% of CS researchers use a formulative approach, 11.0% use an evaluative approach, and 9.9% use a descriptive approach in their research. In this thesis we can find all of these three approaches: the descriptive part of my job includes the review of literature and the research question 1 and Paper I where we describe the *CANDLE scheme* used in ViSCoS. The evaluative part of my study includes research question 3 and is presented in Papers III and IV, and, finally, the formulative part of my research includes research question 2 which gives formulative guidelines characteristics of the dropout phenomenon in ViSCoS Programming I course (presented in Paper II), and formulative guidelines and standards for evaluation process in ViSCoS as presented in Papers III and IV.

3.4. The research sources used

The significant feature of both of research strategies that we use is that, during the study process, it is recommended that multiple sources and multiple methods be used. We report the methods used in this study in Table 3 on page 22 of this thesis.

In Paper II we describe the results based on questionnaires and exam scores in the year 2000. In the beginning of fall 2001, we expanded ViSCoS to the neighboring province of Southern Savolax. In Paper III, we studied whether there were any differences between the student results in first year's area (North Karelia) and the second year's expanded area (Southern Savolax). Our research in Paper III focused, especially, on the role of tutors in these two areas.

In Paper IV, we describe the quantitative and qualitative data collection methods used in our study process (Tables 2 to 3 in Paper IV). The use of *triangulation* proved to be fruitful in our research. The main benefits are described in the section - "Benefits from the Triangular Evaluation Approach" - in Paper IV. In the same paper, we also describe our data collection methods from the qualitative and quantitative points of view.

The main data collection methods used in our study were questionnaires, interviews, action logs, analysis of exam scores, analysis of submitted exercises, and analysis of student feedback.

Table 3: Research methods used in the ViSCoS dropout study process

Year(s)	Source	Focus/Goals/Research questions	Target group
2000 to 2002	Questionnaires	Time when students drop out, reasons for dropping out, the most difficult topics in programming, and the usefulness of supervising, course materials, learning environment etc.	Students who dropped out in years 2000 to 2001, and all of the students in year 2002.
2000	Post-questionnaire by email	More detailed information based on data from questionnaires.	Five of the students who dropped out and answered the questionnaire.
2001 to 2002	Interviews	More detailed information based on data from questionnaires.	10 to 11 of the students who answered the questionnaire
2000 to 2002	Analysis of the submitted exercises	How the amount of submitted assignments differ between easy and difficult theory weeks. Comparison between students who did and didn't drop out.	All of the students, focusing on the students who dropped out.
2000 to 2002	Content analysis of exam	Possible correlation between amount of submitted assignments and the exam scores, and possible correlation between "difficult topics" of programming (defined in questionnaires) and the exam scores.	All of the students, focusing on the students who dropped out.
2000 to 2002	Log of actions Use of discussion forum, email messages and comments given by supervisors in the learning environment	Supervising and its contribution to the learning situation, frustration, or courage.	Supervisors, and all of the students, focusing on the students who dropped out.
2001 to 2002	Questionnaire	Tutors' feelings about working with the ViSCoS program, tutors' attitudes and their influence on the dropout phenomenon.	Tutors

4. THE *CANDLE SCHEME*

In this chapter, we describe the *CANDLE scheme* and the process by which it was created. First we discuss the background behind *CANDLE*, then we continue by presenting the main elements in *CANDLE*. Finally we look at *CANDLE* from the methodological point of view and discuss how well it was perceived to function.

4.1. The creation of CANDLE

At the beginning of the *CANDLE* creation process, we underlined the importance of program designers being knowledgeable about high school content and about the needs of high school students. The instructional design of the ViSCoS program was a result of close collaboration between high school and university teachers.

We found that the contribution from high school teachers was crucial for the development of the *CANDLE scheme*. High school teachers helped us coordinate the courses with the schedules of the high schools, which might differ quite a lot from school to school. A group of high school teachers monitored the course during the whole 1.5-year period (Haataja et al., 2001).

4.2. The main elements in CANDLE

Earlier Braun (1993) discussed the use of technology in schools. He outlined guidelines for the future paradigm for schools as follows: 1) learning environments should be flexible and take into account each student's abilities and needs, 2) the teacher should be a facilitator and a guide, 3) students should be seen as individuals with individual, unique learning styles, and finally 4) students should work cooperatively, gathering facts and developing skills (e.g., skills in decision making, problem solving and information processing.) We agree with Braun (1993) about the importance of the four features mentioned above, even when our learning setting is arranged at distance. We have created the *CANDLE* scheme to meet these demands.

Paper I (chapter 2.1) addresses research question 1 *(What kind of model is suitable for creating a functional online CS program?)* by introducing the educational technology design scheme called *CANDLE*. In *CANDLE* we started the creation process by combining *university* level CS studies into the high school context using *electronic tools* while taking a distance student's (*high school students*) point of view into consideration. The main elements of the *CANDLE scheme* are presented in Figure 1 of Paper I and in Figure 3 in this thesis.

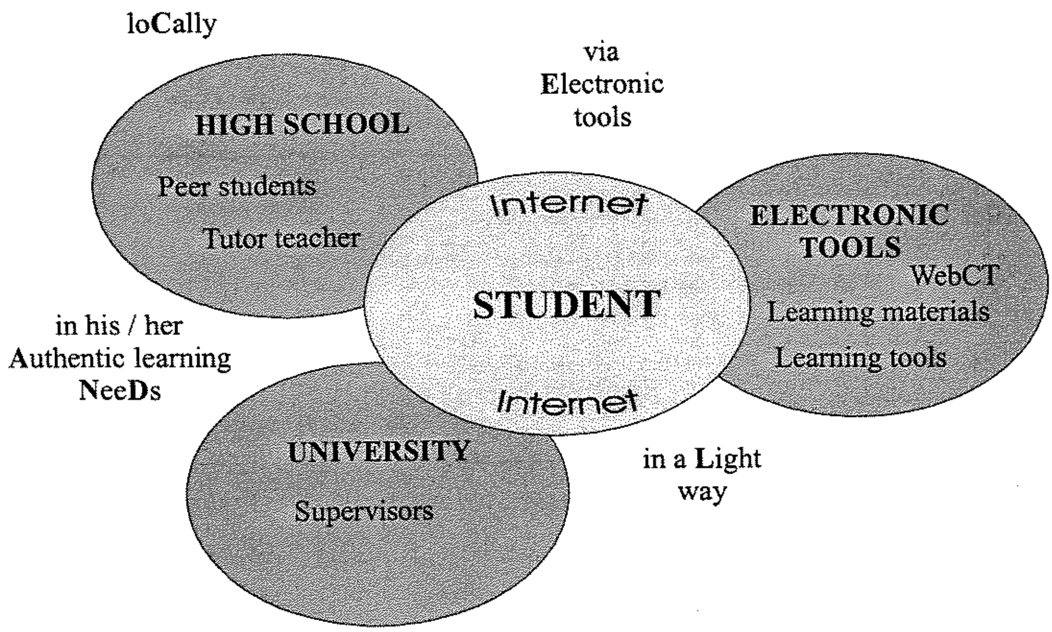


Figure 3. The main elements in the *CANDLE scheme* (Paper I)

We have described the main elements of *CANDLE* as follows:

1. ***High School***: One of the main purposes in *CANDLE* is to enable students to study ViSCoS courses locally from their own high schools. In Finland, the distances between university and high schools are often too demanding for daily (or even weekly) face-to-face university-level studies that occur simultaneously with high school studies. Based on our experiences, distance education increases the equality between high schools and makes it easier for students to take part in university-level studies. In each high school we have one tutor to help students, especially at the beginning of studies. For example, the tutor might show students how to download courses and assignments from the learning environment, how to use the tools of the learning environment, or how to

submit assignments into the learning environment. Tutors are either high school ICT or mathematics teachers.

2. **University:** The role and responsibility of the university is significant. In this case, the University of Joensuu offers eight online courses and organizes all the arrangements concerning the distance learning setting. The teaching responsibilities are carried out by university supervisors who act like facilitators and guides for students over the web.

3. **Electronic Tools:** *CANDLE's* study process is not possible without electronic or digital tools. All of the leaning materials, learning tools, and the learning environment itself are electronic and provide different level of exercises, examples, visualizations and animations to meet students' individual abilities, needs, and learning styles.

4. **Internet:** Students study over the Internet since all of the courses and assignments are available through a learning environment on the web. In order to take part in the ViSCoS program, students need only a computer with a web-browser and an Internet connection. All of the materials are downloadable so students only need to keep the Internet connection active when they download assignments, submit assignments, or when they read email or discussion forum messages. The learning environment is quite flexible in accommodating different kinds of learning materials and electronic learning tools. It has provided the possibility for students to cooperate with their peers over the web.

When we examine the four *CANDLE* elements mentioned above, we can easily notice that all of these four elements are needed in order to create a functional distance-learning program. Unfortunately the system is very fragile. If one of these elements does not function, the whole system is threatened. For example, if the attitudes of the high school's administration or tutors are not positive, it may easily have a negative influence on the motivation of the students as well. Similarly, if the technical components (e.g., electronic tools or Internet connections) break down, the whole system breaks down regardless of how enthusiastic the tutors, students, and supervisors are.

The educational technology design of ViSCoS has been a challenging task for two reasons. First, distance learning itself is not an easy task for every student. Secondly the difficulties in CS studies, especially in programming, have proved to be a challenge for novice students. Especially in programming, plenty of CS students have difficulties during the beginning of their studies, even in face-to-face learning situations (Ben-Bassat Levy et al, 2003; Renaud et al., 2001; Wiedenbeck et al, 2004). Many novices reported that the programming language details were too complex, found the whole course to be very difficult, and felt that the course was extremely hard to pass (Denning, 2004).

In distance learning settings, the most critical factor for students' success or failure is how the technology is set up and how well the technology operates at the beginning of semester (Kleinman & Entin, 2002). In the *CANDLE scheme* we have prepared for this in several ways:

1) students have local support from their tutors in practical problems, especially, at the beginning of studies,
2) students get a guide for distance studies,
3) students also get a guide for using the learning environment,
4) supervisors support students on-line in all of their difficulties, and finally
5) the learning environment also allows peer support in multiple ways via a discussion forum, email, or chat rooms.

In our case the technical difficulties at the beginning of studies have been quite similar to difficulties reported by Mock (2003). Mock found that many students had difficulties simply with installing Java on their home computer. Also, in our case the installation differences on different platforms (Windows 98/XP, Linux etc.) and setting the class path has been the most difficult hurdles in the beginning part of CS studies. Although we gave detailed instructions with screenshots, there were many students who didn't manage to install the Java environment on their home computer.

In *CANDLE* we emphasize the role of peer support by encouraging students to use discussion forums or email in order to ask for help not only from supervisors, but also from peers both in their own high school and from other high schools over the web as well. Many of our students found that the answers given by other students were useful. In fact, it might even be easier to ask help from a peer student than from supervisors (Sutinen & Torvinen, 2001). There is strong evidence in the literature for the value of collaboration among peers as well (Kleinman & Entin, 2002).

In Paper I, we describe the use of the *CANDLE scheme* in the creation process of ViSCoS.

4.3. The CANDLE scheme from the methodological point of view

In *CANDLE* we have four elements - the distance student, the high school context, the university, and the electronic tools (see Figure 3 of this thesis). All of these elements have been investigated by the research team of Educational Technology at the University of Joensuu (EdTech, 2004). A study which investigated a combination of three of the elements – the context of high schools, the distance student, and university-level studies – is included in this thesis as Paper I. Our research group has created several electronic tools, like LEAP, former PPA, (Suhonen & Sutinen, 2003) and Ethicsar (Jetsu et al, 2004) which have been investigated by the other members of Educational Technology research group. In addition, we do co-research with the Weitzmann Institute of Science (Weitzmann, 2004) to improve the Jeliot2000 program and to create the newest version of the Jeliot family, Jeliot 3 (Jeliot, 2004). Table 4 presents the methods we used to investigate each aspect of the *CANDLE scheme*.

As we can see in Table 4, we studied the usefulness of *CANDLE* in several ways. The functionality of loCality is investigated mainly through questionnaires and interviews. We asked both the students and the tutors about their experiences and satisfaction concerning the opportunity to study university-level studies in the high school through distance education. Based on our experiences, both students and tutors have been mainly enthusiastic about studying and working in the ViSCoS program.

Table 4: *CANDLE* research methodology

CANDLE supports a student	**Method used to enhance the feature**	**Target group**
loCally	Questionnaires, interviews	Students
	Questionnaires	Tutors
In his / her Authentic learning NeeDs	Questionnaires, interviews, discussion forums, email messages	Students
	Questionnaires	Tutors
	Experiments, mutual information, comments from students	Supervisors
in a Light way	Questionnaires, interviews, discussion forums, email messages	Students
	Questionnaires	Tutors
	Experiments, mutual information	Supervisors
via Electronic tools	Questionnaires, interviews, discussion forums, email messages	Students
	Questionnaires	Tutors
	Experiments, mutual information, discussion forums	Supervisors

More detailed information concerning the focus of questionnaires and interviews is presented earlier in Table 3 on page 22.

Earlier, Watanabe et al. (1999) found that the most difficult task in web-based course design is to predict the learning situations where students will encounter difficulties and what kinds of support they will need at that point in time. We noticed that students need special support for their Authentic learning NeeDs, especially, at the beginning of the distance study process and in situations when difficulties occur in the learning process. Our students have been generally satisfied with the support given by supervisors based on the answers reported in the questionnaires and interviews. Also feedback given by tutors has been good. There have been only few students who might want something more (e.g., more individual support in certain cases). The individual comments given by supervisors on submitted assignments tell us much about the supervisors' skills in supporting students in the online study process.

In *CANDLE*, we offered several ways for providing support. *Tutors* help at the local level by giving advice (especially, at the beginning of studies) and in response to technical questions like how to use the learning environment or how to submit assignments into the learning environment. If students encounter problems concerning the subject to be learned, they can contact their *supervisors* via email, a discussion forum in the learning environment, or even by telephone. *Peer students support* is available both locally, from their high school, or at a distance via the Internet from students from other high schools. In addition, the different kinds of *visualizations tools* (like Jeliot2000 / Jeliot 3, Flash animations, and interactive and non-interactive Java applets) have been created *for supporting authentic learning needs*.

The course material was constructed in a Light way in order to make the material easy to download and use over the web. Both students and tutors found this an excellent and functional solution. From the supervisor's point of view, the course materials were created to serve as a supplement for the course book by giving optional examples, visualizations, and exercises. There is no point to rewrite an already functional course book in the form of Hypertext; instead we emphasize that the web-material should supplement the course book.

The analysis of messages in the discussion forum and email messages gave us quite a rich picture about the levels of satisfaction concerning general arrangements in the ViSCoS program. The functionality of electronic tools is the most essential criterion for success in a web-based course system. Although there were several harmless firewall problems, there were not any major problems.

5. THE DROPOUT PHENOMENON

The virtual learning environment itself is a challenge to overcome for teachers and students alike. For the learning material (and the teachers who create the learning material), the virtual learning setting is quite demanding since the web-based material has to somehow replace the role of teacher in traditional face-to-face learning situations. Students play the largest role in determining if a virtual learning environment will succeed or not; students have to be much more independent and take more responsibility for their own learning process than in other educational settings. Some of the students have not been able to do this and they have dropped out (Sutinen & Torvinen, 2001).

In this chapter, we will describe the dropout phenomenon first in virtual learning settings in general and then describe the dropout phenomenon in more detail as it pertains to the ViSCoS program. We will end this chapter by looking at the dropout phenomenon in the ViSCoS program from the methodological point of view.

5.1. The dropout phenomenon in online settings and in CS

The distance learning setting is quite a demanding one. For example Xenos et al. (2002) and Cornell & Martin (1998) found that even 30 – 50 % of students are going to drop out during the virtual study process. They found that in Europe, the drop out rates are in general over 30 % and in Asia, even 50 % of those who started an online course might give up before the end of the course.

Teaching programming is a challenging task even in face-to-face learning settings (Wiedenbeck et al., 2004). When we move the teaching of programming to the distance learning setting, it is even more challenging (Renaud et al., 2001). Mock (2003) found it extremely challenging to design the preparation course, CS0[2], that students study before taking CS1 remotely. His results show that of the large number of students who dropped out

[2] Topics covered by the course included among others the basic knowledge of programming with Java, like data types, if-statements, while loop and introduction to methods (Mock, 2003), that equals to our Programming I as well.

of the class, the dropout rate for the distance course was 42%. That was much higher than the 12% and 26% for the same class in face-to-face learning setting in previous years taught by the very same teacher. In his study, the results were mixed. Students were reportedly satisfied with the class, although there was a high rate of drop out (Mock, 2003). The survey concerning students' satisfaction with course arrangements was completed only by the students who finished the course. Mock (1993) noted that the next study should also include interviews with the dropout and not only with those students who remained in the course.

Wiedenbeck et al. (2004) have reported several factors that may influence novices' success in introductory university-level programming courses. There are several studies (like Byrne & Lyons, 2001; Hagan & Markham, 2000; Wilson & Shrock, 2001) that clearly show a positive correlation between previous programming experience and success in an introductory university course. Byrne & Lyons (2001) and Wilson & Shrock (2001) also found a positive relationship between mathematics or science background and success in computer programming. The relationship between students' learning styles and learning to program is investigated by Byrne & Lyons (2001) as well. Wiedenbeck et al. (2004) also found that there are several research articles which concern student's mental models of programming in relation to success in specific programming tasks (see Cañas et al., 1994, Wiedenbeck et al., 1999).

Although numerous factors have been investigated, we are still based on Wiedenbeck et al. (2004) far from having a full understanding of the difficulties in learning programming – they wonder in their study why some of students learn to program easily and quickly while the others flounder.

5.2. The dropout phenomenon in the ViSCoS program

In the ViSCoS program, the coursewise drop out rates varied from 0 % to 42.9 % in the years 2000-2002 (Paper III). During the first year, 2000, we noticed that the highest dropout rates were in the programming courses. Our experiences show that students who pass Programming 1 in Semester 1 and Programming 2 in Semester 2, will very probably pass the rest of the ViSCoS courses as well (Paper III). So our findings support the earlier findings from Cornell & Martin (1998) who noticed that students who passed at least one virtual course will pass the future virtual courses as well. Wiedenbeck et al. (2004) noticed that if a student drops out, fails, or passes with a struggle, that student is unlikely to enroll in the following courses.

Already during the first year of ViSCoS, we noticed that the dropout phenomenon was unavoidable; we became interested in studying the dropout phenomenon in more detail in order to understand the cause of the difficulties that students in programming courses were having. Our intent was to deepen the understanding of the dropout phenomenon in general and to decrease the rate of student attrition from the demanding online programming study process.

In order to understand the reasons leading to the dropout phenomenon, we collected questionnaire data from the students who dropped out in the years 2000 and 2001 and from all of the students in the year 2002. The original questionnaire for the year 2002 can be found in Finnish in Appendix I and in English in Appendix II. Based on our questionnaires, we noticed that, in general, the students who dropped out were either the students who had scheduling problems between their regular high school studies and online university level ViSCoS studies or they were students who had difficulties with the topic concerned, especially when studying programming. PISA reported that high self-confidence and interest in mathematics does not correlate with other strengths (PISA, 2004). Researchers in the PISA group also found that the weakest learners who had not yet succeeded in acquiring the prerequisites for lifelong learning and were very likely to need some additional support in order to succeed.

5.3. The dropout phenomenon from the methodological point of view

In Paper II we concentrate on research question 2 *(What are the characteristics of the dropout phenomenon in the ViSCoS program?)* by describing the dropout phenomenon in ViSCoS and introduce the formative evaluation system that we used to evaluate our Programming 1 course. The study is a very typical case study; we limited the research to the dropout phenomenon in the Programming 1 course since we wanted to get deeper knowledge about the phenomenon itself.

We used both qualitative and quantitative methods in order to collect data. We sent out student questionnaires where we asked, among other things, the reasons for dropping out, the time of drop out, the most difficult topics in the Programming 1 course, and suggestions for improvements. We also used quantitative methods such as analyzing exam scores to determine the success and failure rates of students who did and did not drop out.

The visualization tools have proved to be especially useful for novice-level students (Ben-Bassat Levy et al., 2003). In our research, we analyzed the usefulness of Jeliot2000, Flash animations, and interactive and non-interactive Java applets through questionnaires and interviews. We got both quantitative and qualitative data concerning the experiences of students who dropped out.

If we look at the research paradigms presented in the chapter 3.1 on page 15 of this thesis, we can define 'pragmatism' as the research paradigm of our study. The main idea was to *enlarge the understanding of the dropout phenomenon in general.* The pragmatic paradigm allows *both quantitative and qualitative methods for data collection,* as we have. As for the theory of knowledge in our drop out research, *both objective and subjective points of view* were used.

In Paper II, we describe the dropout phenomenon in the ViSCoS program in detail.

6. IMPROVEMENTS

During the past few decades the quantity of web-based courses has rapidly increased. Especially in the beginning of the 1990s, teachers simply recreated their existing courses in Hypertext without considering web-pedagogy during the development process. Farrell et al. (1999) have done an excellent global survey on the situation of virtual education. In their survey, they noticed that, all too often, web-based materials are created just because they are trendy and that they are created without regard to whether the form of Hypertext gives any benefit at all to the learning situation. The other mistake, especially in the early web-based courses, was also that people far too often did not have enough time to improve web-based based materials after their initial creation.

In this chapter we first briefly look at the evaluation and improvement methods presented in the literature. Then we describe the continuous evaluation system created to evaluate and improve the ViSCoS program in general and especially for the Programming 1 course. Finally in chapter 6.3 we describe the improvement process from the methodological point of view.

6.1. Evaluation and improvements

Almstrum et al. (1996) found that the only reason for evaluating the whole system is to learn how to do it even better. In order to improve our programs, we have to accept all the results and look at our work truthfully based on the feedback derived from students, tutors, supervisors, and through the analysis of learning materials.

Carbone & Kaasbøll (1998) reported the methods used to evaluate CS teaching in their literature review. They outlined the techniques used for evaluation of teaching difficult areas that were reported in recent research as follows:

1. *General questionnaires:* many universities examine students' general impression of a course right after the course is finished. These kinds of questionnaires are quite general and the responses might be useful for evaluating a certain area or unit of teaching.

2. *Examination marks:* Carbone & Kaasbøll (1998) emphasize that if the examination marks reflect students' competence, the marks should also be indicators of the effects of teaching. They noticed that examination marks are useful when measuring the success of teaching a difficult topic; naturally the marks of these topics have to be separable from those of other topics in a course. Carbone & Kaasbøll (1998) found that the populations have to be matched if examination marks are used to compare a change of teaching from one population of students to another.
3. *Experiments:* In CS education research, experiments are infrequent. There are, however, some types of CS research, (e.g.,human-computer interaction), where laboratory experiments can be used. The teaching and learning process often becomes too complex to set up controlled experiments. (Carbone & Kaasbøll, 1998)

One very important key to improving CS teaching is to improve the whole curriculum by paying careful attention to the most difficult curricular components. Denning et al. called for the improvement of CS teaching in 1989 by criticizing the misleading view that "computer science equals programming" (Denning et al., 1989, on page 9). They found that programming clearly is part of the standard practices of CS principles and that every computing major should have competence in it, but they also reported that the curriculum should not be based entirely on programming and that introductory courses should not be programming courses at all (Denning et al., 1989). Although their report was designed to provoke new thinking about computing as a discipline, Denning still reports 14 years later that even the existing CS curricula used this misleading image (Denning, 2003). He argued that when CS teaching started, the first courses were programming courses, but in those days the languages were easier for beginners (Denning, 2003). Currently, students are exhausted by the complexity of languages that many experts find challenging, like Java or C++. Denning (2003) also mentioned that many students have even turned to cheating and plagiarism in order to pass the courses and that the dropout rates have increased to 35% - 50%.

Although CS teachers do not have many additional resources for evaluating their teaching improvements, some easy solutions can still be found. Data from sources like questionnaires, examination marks, literal quotations from students, and materials from students are relatively easy to collect. Gathering data from more than one source usually strengthens the validity of the findings (Carbone & Kaasbøll, 1998). If we look at the data collection methods used in our research (shown in Table 3 on page 22), we can see clear evidence for using an improvement process as rich and versatile as possible by collecting data from students, tutors, and supervisors using different kinds of methods.

6.2. *The continuous improvement process in ViSCoS*

Our evaluation system is based on the principle of continuous evaluation and improvement. In all of our papers, we describe how we improved our course materials immediately after receiving the feedback given by students and tutors mainly via questionnaires and interviews but also via discussion forums and email. Especially in Paper III and Paper IV, we focus on the improvement process. Figure 4 (also found in Paper IV) illustrates the idea of the continuous improvement process used in ViSCoS.

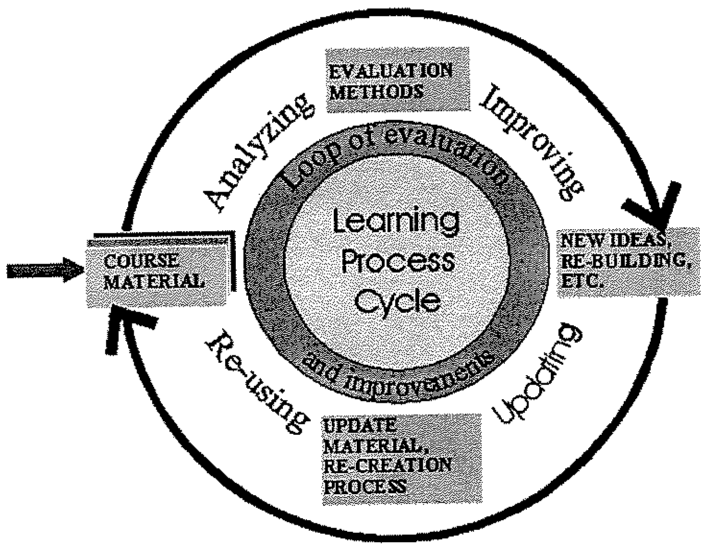


Figure 4: Loop of evaluation and improvements (Paper IV)

Since the main idea in ViSCoS is to take care of the needs of students, we created the course material in collaboration with high school tutors and the staff of the university. We had to ensure the rigor of university-level studies while keeping in mind the needs of high school students. The high school aspect was taken into account in the planning process when we catered to the target group by integrating examples and optional exercises into the web-based material. Actually we had one secret that helped us to succeed in this challenging task; at the beginning of the process, we already had a few high school students who were responsible for helping us create assignments that would appeal to other high school students. Naturally the supervisors from the university had to modify and complete the assignments, but the original ideas came from these young people.

We noticed that the general satisfaction with the ViSCoS program and its arrangements has increased over the years. Based on our interviews, the new students are obviously more ready for independent work partly because of the knowledge they have received from earlier ViSCoS students and tutors. In the ViSCoS orientation session at the beginning of the study process, supervisors underlined the demanding nature of this online learning process so students would understand the amount of time and resources required in this study process. Kleinman & Entin (2002) have found similar kinds of results with their students. They found that the attitudes of distance learners have been generally positive and, sometimes, even more positive than the attitudes of traditional in-class students. In our case, the motivation of high school students has to be high in order to manage the challenging task of successfully completing ViSCoS while attending high school.

The evaluation methods mentioned in Figure 4 are shown in Table 3 on page 22. After analyzing the data collected by these multiple methods, we have focused on improving the most complex part of Programming I course. The annual schedule of our cyclical improvement process is shown in Table 5.

Table 5 Annual schedule of the improvements in ViSCoS

Type of action	**Point in academic year**
Studying	Semester 1
Analyzing	Semester 2
Improving and updating	Summertime (the school holidays)
Re-using	Semester 3 (equals to Semester 1 for new students)

6.3. Improvements from the methodological point of view

The research strategy used in ViSCoS is a combination of action research and case study. In section 5.3, we have already described the characteristics of our case study research. The features of action research in our study are evident in the improvement process described in Papers III and IV. Denscombe (1998) has described action research as a cyclical process where research and improvement alternate; the development of the process is the main focus of the research.

In Paper III we focused on the improvement process of ViSCoS by concentrating on research question 3 (*How can we improve the Programming 1 course in order to minimize the number of dropouts?*). Paper IV is also closely related to research question 3. In Paper IV, we describe the continuous improvement cycle (shown in Figure 4 on page 36 of this thesis), and describe the qualitative and quantitative data collection methods used in our research.

As presented in 3.2 in page 17 in this thesis, action research focuses on real-world problems and issues. Typically in action research, not only researchers but practitioners are also active in generating knowledge from the research. Based on Denscombe (1998), action research is a cyclical process where research involves a feedback loop. These action research features are very obvious in our study.

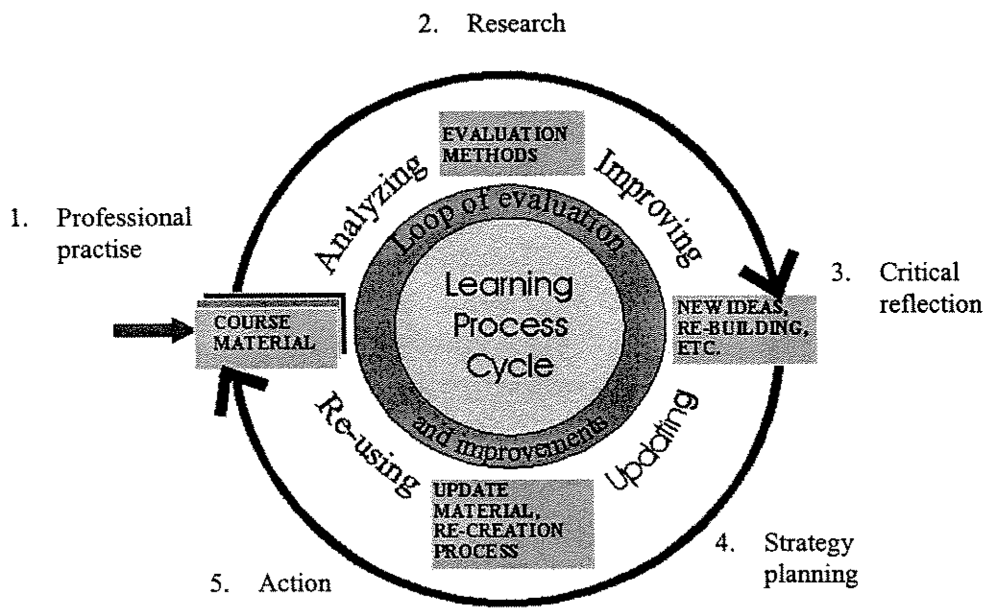


Figure 5: Loop of evaluation and improvements complemented with features of action research

In Figure 5 we illustrate the relationships between features of action research approach (Denscombe, 1998) and our loop of evaluations and improvements by combining them together and in Table 6 we present the action research approaches combined with the annual schedule of improvements in ViSCoS.

Table 6 Action research approaches situated in annual schedule of improvements in ViSCoS

Action research approach	Point in academic year
Professional practice	Semester 1
Research	Semester 2
Critical reflection and strategy planning	Summertime (the school holidays)
Action	Semester 3 (equals to Semester 1 for new students)

The main idea in our research has continually been to promote the improvement process. Within the formative evaluation scheme (presented in Paper II), we succeeded in collecting important ideas for improvement. The most efficient way to improve the Programming 1 course seemed to be to focus on the students who dropped out; the opinions, experiences, and suggestions given by them are more valuable than the opinions, experiences, and suggestions of those students who succeeded in the ViSCoS program without any serious problems. Of course, the talented students have to be taken into account as well, and that is why we expanded the questionnaires in year 2002 to include those students who had not dropped out.

During the first two years, our improvement process was based solely on the information given by those who dropped out. This enabled us to focus on the improvement process, especially, regarding the most difficult topics (e.g., arrays, loops and methods) that were taught in the Programming 1 course. For example, we increased the number of optional exercises and examples in the web-based material. We also used the Jeliot2000 program and created Flash animations to visualize the use of arrays and loops. To enhance the teaching of programming methods, we created interactive applets and Flash animations.

7. CONCLUDING REMARKS AND FUTURE WORK

The development of a functional web-based learning environment is a challenging task. One of the most typical difficulties in the early stages of web-based learning implementations has been the lack of continuous follow-up and improvement (Farrell et al., 1999). This thesis and the enclosed four papers describe the design process used for improving a virtual online setting in computer science called ViSCoS. Within ViSCoS, the emphasis was on an online programming course (Programming 1).

At the beginning of the process we had already emphasized the importance of the students' point of view and focused our design process on students' needs (Figure 3 in page 24). We found out that a student needs to be the central hub of a learning environment, similarly to what Chute et al. (1997) described.

The main idea in the *CANDLE scheme* has been to offer different kinds of supporting methods to students in order to avoid one of the most difficult design problems - how to know the point in time when students will need support and what kinds of support they will need at that point (Watanabe et al., 1999). The *CANDLE scheme* described in Paper I is based on student needs at both the local level (tutors, peer students in the same high school) and at the web-based learning environment level (supervisors, peer students over the web, supporting material, discussion forum, email etc.). With this rich variety of support methods, we have succeeded in meeting the needs of students. In the end, the activities of the students themselves have the most impact on their success in a web-based learning setting anyway. Our task is to offer the most adequate methods and tools available.

In the development process of the ViSCoS program, focusing on the dropout phenomenon has proved to be the most efficient and successful way to improve the ViSCoS program. We found that the students who dropped out either had scheduling conflicts between their regular high school studies and online university level ViSCoS studies, or they were students who had difficulties with the topic concerned (especially programming). In paper II, we describe a formative evaluation scheme that we created in order to affect the dropout phenomenon immediately after the first year. Our first improvements (re-scheduling, the visualizations, and the optional examples, especially, in the most difficult topics of programming) have already proved fruitful for the development process. Like Almstrum et al. (1996) observed, the only reason for evaluating the learning system is to learn how to do it even better. In Paper III we describe the improvements done in the ViSCoS Programming 1 course based on our research work.

Although we have not taken into account the annual variance in the academic skills of the tutors and student populations (since programming skills or ability for online study process might differ a lot from year to year), we can see that the results described in Paper III are quite encouraging for the development process. The general satisfaction with the ViSCoS program and its arrangements has increased, and based on our observations the new, starting students are obviously more ready for independent work because of knowledge passed on to them from former ViSCoS students and tutors. Since supervisors have also begun to emphasize the demanding nature of this online learning process in the introductory session of ViSCoS, students should not have misconceptions, regarding the ease of university-level distance study, like they had in the first year when plenty of enthusiastic high school students started university-level studies with rosy hopes.

In order to get a picture of the dropout phenomenon that is as wide and rich as possible in this action research type case study, we have collected data with both qualitative and quantitative methods, and analyzed them with both of these approaches. In Paper IV, we describe the methods used and the benefits of data triangulation that we successfully utilized in our research.

To summarize, the original three research questions resulted in the respective answers:

1) *What kind of model is suitable for creating a functional online CS program?*

We created our model with the close cooperation of high school teachers and with students at the same age as students in our target group. The online course setting has its own special demands for the supervisors as facilitators, and for the students as independent and individual workers who work in close cooperation with their peers and supervisors. It has been a challenging task to create a model that teaches a difficult subject *and* addresses the demands of the distance learning setting. ViSCoS has been quite a demanding challenge for us because in ViSCoS we offer university level CS studies (including programming) to high school students over the web.

We built the model based on the following four features:

i) High School (lo**C**ally, support by tutors),

ii) University (supervisors help students in students' **A**uthentic learning **N**ee**D**s),

iii) Course materials (**L**ight way: easy to download and use, easy to produce),

iv) via **E**lectronic tools (electronic learning environment, Internet, web-based tools)

The model which we created is the *CANDLE scheme* which is presented in Paper I and in chapter 4 in this thesis.

2) What are the characteristics of the dropout phenomenon in the ViSCoS program?

The dropout rates among distance learners are in general higher than among on-site students. In general, dropouts among first year CS students are common. In ViSCoS the dropout phenomenon is most evident in the programming courses. Since most of the students who pass the Programming I and Programming II courses will pass the rest of ViSCoS courses, it is important to find the characteristic features of the dropout phenomenon in order to encourage students to complete their ViSCoS studies. Most students dropout because of lack of time, but other factors can be found as well. The majority of students who dropped out did not have any kind of previous programming skills before ViSCoS. Loops, arrays, if-statements, and methods have been found to be the most difficult topics in ViSCoS. Most of the students who were absent during the theory weeks were likely to drop out. The dropout phenomenon in the ViSCoS program is described in Paper II and in chapter 5 in this thesis.

3) How can we improve the Programming 1 course in order to minimize the number of dropouts?

Earlier literature describes plenty of methods used in order to improve distance materials and CS teaching. We have created our own model, based on continuous loops of evaluation and improvement which are grounded in the action research approach. Our process has the following four steps:

i) carry out studies during Semester 1,

ii) research and analyze (questionnaires, interviews etc.) during Semester 2

iii) improve, rebuild and update the program during summer time (the school holidays), and

iv) re-use the updated material with a new population of students during Semester 3 (equals to Semester 1 for new students).

We have now successfully used this cyclical loop of evaluation and improvements for four years in the ViSCoS program. The improvements done in the Programming 1 course are described in Papers III and IV and in chapter 6 in this thesis.

The systematic development process using the continuous loop of evaluation and improvement presented in Figure 4 (and in Paper IV) has proved to be a fairly functional and efficient way to develop a web-based learning setting. The results of our research work give us positive signals and encourage us to continue with our work.

Our future work within the dropout phenomenon will continue with my PhD-level research that focuses on the weekly assignments, especially, during difficult theory weeks. We have already done the groundwork for that; using exercise classification, we determined which theory weeks in the Programming 1 course are the most difficult (Meisalo et al., 2004).

Our research team has been augmented by a few undergraduate students. In addition, part of other PhD theses in the Educational Technology Resource group at our department (Edtech, 2004) are related to ViSCoS as well. We illustrate the relationships of this thesis and future work made by me and by other PhD or undergradute students in Figure 6.

One of the undergraduate students is studying the ViSCoS students who passed the program. She studies how the ViSCoS students have managed after high school and what kinds of influences the ViSCoS program has had on them in their post-matriculation studies and what kind of influence it has had on their decisions concerning their future studies (MSc thesis I in Figure 6). Another undergraduate student is studying the on-going ViSCoS program from the counseling and guidance point of view, especially, in the Programming 1 course (MSc thesis II in Figure 6). The third undergraduate student has started her research with a classification of programming textbooks' exercises (MSc thesis III in Figure 6), which is very closely related to our ongoing research with ViSCoS's exercise classifications (Meisalo et al., 2004).

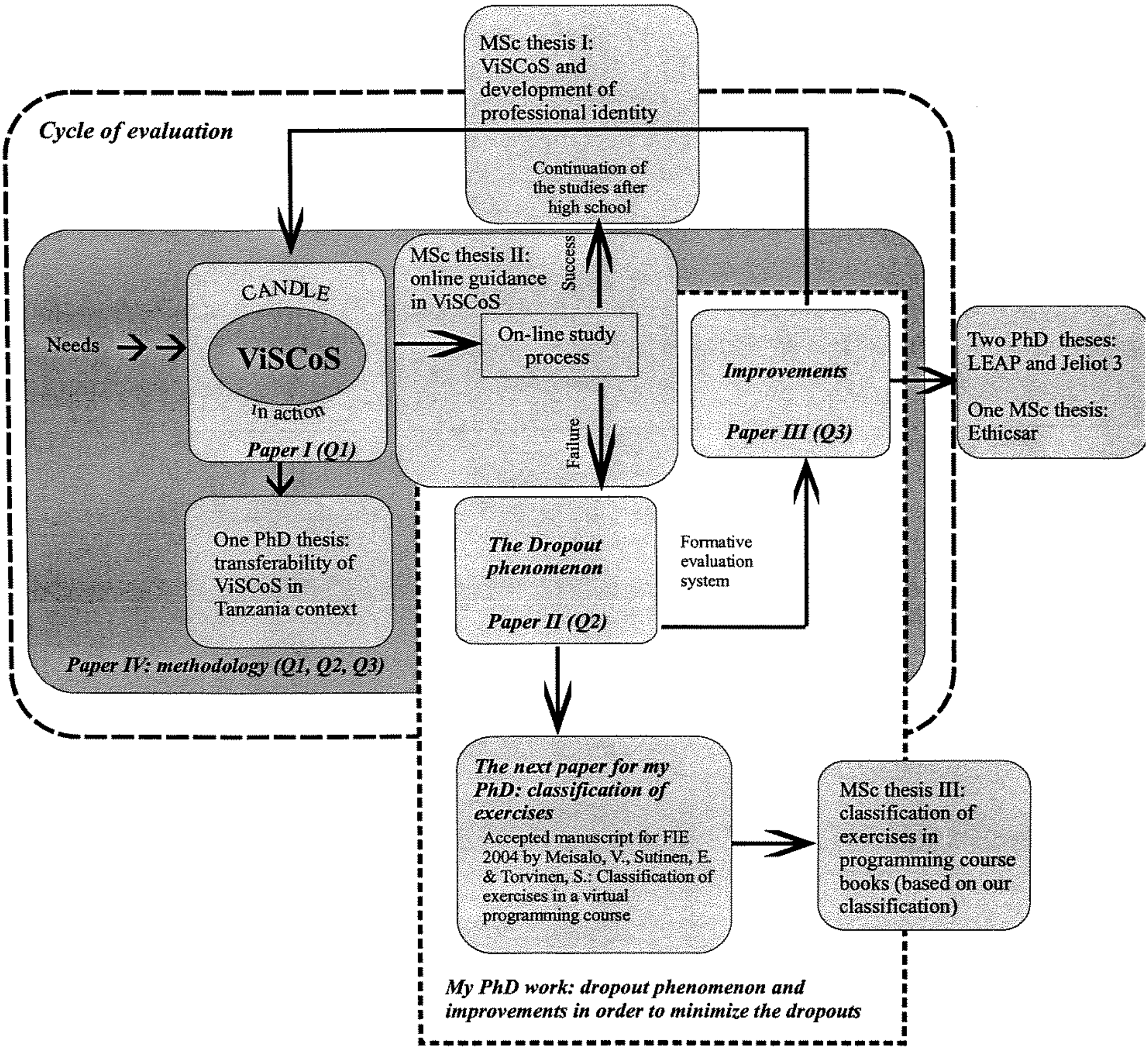


Figure 6 Future work within ViSCoS

Concerning future work, we think that it will be interesting to test the transferability of *CANDLE* into different contexts as well. At this moment, we have already started one experiment in Tanzania, and we have also planned some co-projects with partners in Kazahkstan, Kyrgys, Uzbekistan, Lithuania, and England. In addition, ViSCoS will offer a wonderful platform to test new web-based learning tools and adaptive systems as well.

Acknowledgements

First of all I would like to thank my supervisors professor Erkki Sutinen from the University of Joensuu and professor Veijo Meisalo from the University of Helsinki who have been just wonderful benefactors for me during my post-graduate study project. Their encouraging and confidential attitudes towards my work has carried me through the most difficult moments when my own skeptical feelings towards my research work arose every now and then.

I would like to give my warmest thanks to Justus Randolph who has patiently helped me with grammar checking for this thesis. Justus, you can not even imagine how thankful I am for your wonderful help in making my writing more fluent.

I would also like to thank the team of ViSCoS for their cooperation during my research. The discussions about the successes or failures in ViSCoS arrangements during our team meetings have led my research in the right direction and made it possible to examine the most demanding development matters.

And I would like to thank the rest of our educational technology research group and all the other people who have read my work for your constructive comments during my research work. By reading and commenting on the papers and my thesis during the various stages of production, you have helped me to focus on the most crucial points and present them concisely.

And naturally my warmest thanks to all of the tutors and students who have participated to my research in years 2001 - 2003. Within your contribution, this thesis is now completed. My whole-hearted thanks to all of you!

And last, but certainly not least, I would thank my wonderful children, my 9-year-old son Eetu-Markus and my 7-year-old daughter Jenna, who have understood the importance of the mama's writing project. I'm really thankful that you can sleep so well at nights – it has made it possible for me to finalize my thesis. *Eetu ja Jenna: Lämmin halaus ja KIITOS!*

References

Almstrum, V.L., Dale, N., Berglund, A., Granger, M., Little, J.C., Miller, D.M., Petre, M., Schragger, P. and Springsteel, F.: "Evaluation: turning technology from toy to tool. Report of the Working Group on Evaluation". *Proceedings of the 1st Conference on Integrating Technology into Computer Science Education, ITiCSE 1996*, ACM Press, NY, USA, 201-217, 1996.

Ben-Bassat Levy, R., Ben-Ari, M. & Uronen, P.A.: "The Jeliot 2000 Program Animation System", *Computers and Education*, **40**(1), 1-15, 2003.

Braun, L.: "Help for all the students", *Journal of Communications of the ACM,* **36**(5), 66-69. 1993.

Byrne, P. & Lyons, G.: "The Effect of Student Attributes on Success in Programming.", *Proceedings of the 6th annual Conference on Innovation and Technology in Computer Science, ITiCSE 2001*, ACM Press, NY, USA, 49-52, 2001.

Cañas, J. J., Bajo, M. T. & Gonzalvo, P.: "Mental Models and Computer Programming.", *International Journal of Human-Computer Studies,* **40**(5), 795-811, 1994.

Carbone, A. & Kaasbøll, J. J.: "A Survey of Methods Used to Evaluate Computer Science Teaching", *Proceedings of the 6th annual Conference on the Teaching of Computing and the 3rd annual Conference on Integrating Technology into Computer Science Education, ITiCSE 1998,* ACM Press, NY, USA, 41-45, 1998.

Chute, A.G., Sayers, P.K. & Gardner, R.P.: "Networked Learning Environments." In Cyrs, T.E. (Eds.): *Teaching and Learning at a Distance: What It Takes to Effectively Design, Deliver, and Evaluate Programs. New directions for teaching and learning.* Number 71. Jossey-Bass Publishers, San Francisco, 75-83, 1997.

Cordani, J.R. & Tucker, R.J.: "Tools for Higher Education Distance Teaching", *Proceedings of the 26th annual Conference on Special Interest Group on University and College Computing Services, SIGUCCS 1998*, Bloomington, USA, 71-76, 1998.

Cornell, R., Martin, B.L, "The Role of Motivation in Web-Based Instruction", In Khan, B. H. (Eds.), *Web-Based Instruction. Educational Technology Publications*, New Jersey, USA, 93-106, 1997.

Dahanarajan, G.: "Distance Education: promise, performance and potential. Open learning.", *The Journal of Open and Distance Learning,* **16**(1), 61-68, 2001.

Denning, P. J., Comer, D. E., Gries, D., Mulder, M. C., Tucker, A., Turner, A. J. & Young, P. R.: "Computing as a discipline", *Communications of the ACM*, **32**(1), 9-23, 1989.

Denning, P. J.: "Great Principles in Computing Curricula", *Proceedings of the 35th annual Conference on ACM Special Interest Group on Computer Science Education, SIGCSE 2004,* USA, 336-341, 2004.

Denscombe, M.: *The good research guide for small-scale social research projects*. Open University Press, Buckingham, 1998.

Edtech, Home site of Educational Technology research group at the Department of Computer Science, University of Joensuu, URL: http://cs.joensuu.fi/pages/edtech/ (June 21, 2004)

Ellis, A., Lowder, J., Robinson, J., Hagan, D., Doube, W., Tucker, S., Sheard, J. & Carbone, A., "A collaborative strategy for developing shared Java teaching resources to support first year programming", *Proceedings of the 4th annual Conference on Innovation and Technology in Computer Science Education, ITiCSE 1999,* ACM Press, NY, USA, 84-87, 1999.

Farrell, G. Introduction. In Farrell, G. et al. "The Development of Virtual Education: A global perspective". *The Commonwealth of Learning*, Department for International Development, London, 1999. URL: http://www.col.org/virtualed/chapter1_intro.pdf (July 6, 2004).

Finnish Virtual University, Web-site, URL: http://www.virtuaaliyliopisto.fi/?language=eng (July 6, 2004)

Gardener, H.: "Seven intelligence model", *Learning skills research,* University of Westminster, UK, URL:
http://www2.wmin.ac.uk/eic/learning-skills/cognition/learning_styles/howard_model.html (June 28, 2004)

Glass, R. L., Ramesh, V. & Vessey, I.: "An Analysis of Research in Computing Disciplines", *Communications of the ACM,* **47**(6), 89-94, 2004.

Greening, T.: "Computer Science: Through the eyes of potential students", *Proceedings of the 3rd Australasian Conference on Computer Science Education*, The University of Queensland, Australia, ACM Press, New York, NY, USA, 145-154, 1998.

Haataja, A., Suhonen, J. , Sutinen, E., Torvinen, S., "High School Students Learning Computer Science over the Web", *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning (IMEJ)*, **3**(2), 2001.

Hagan, D. & Markham, S.: "Does it help to have some programming experience before beginning a computing degree program?", *Proceedings of the 5th annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education, ITiCSE 2000*, ACM Press, NY, USA, 25-28, 2000.

ISVY, The Eastern Finland Virtual University Network,
URL: http://www.joensuu.fi/isvy/isvyeng/ (June 21, 2004)

Jeliot 3 and the Jeliot family home site, URL: http://cs.joensuu.fi/jeliot/ (June 21, 2004)

Jetsu, I., Meisalo, V., Myller, N. & Sutinen, E.: "Ethical argumentation with Ethicsar", *Proceedings of the IADIS International Conference on Web Based Communities*, Lisbon, Portugal, 255-261, 2004.

Kleinman, J. & Entin, E.B.: "Comparison of In-Class and Distance-Learning Students' Performance and Attitudes in an Introductory Computer Science Course", *The Journal of Computing in Small Colleges*, **17**(6), 206-219, 2002.

Klemm, W. R.: "Creating Online Courses: A Step-by-Step Guide", *The Technology Source*, May/June 2001, URL: http://64.124.14.173/default.asp?show=article&id=861 (June 28, 2004)

Lawhead, P.B., Albert, E., Bland, C.G., Carswell, L., Cizmar, D., DeWitt, J., Dumitru, M., Fahraeus, E.R. & Scott, K.: "The Web and distance learning: what is appropriate and what is not.", *Report of ITiCSE'97 Working Group on the Web and Distance Learning. SIGCSE/SIGCUE Integrating Technology into Computer Science Education, ITiCSE 1997*, Uppsala, Sweden, 27-37, 1997.

Lee, M.G.: "Profiling students' adaption styles in Web-based learning". *Computers & Education,* **36**(1), 121-132, 2001.

Meisalo, V., Sutinen, E. & Torvinen, S.: "Classification of Exercises in a Virtual Programming Course": *Manuscript for the 34th ASEE/IEEE Frontiers in Education Conference, FIE2004,* Savannah, GA, USA, 2004.

Mock, K.: "The Development of a CS0 Course for Distance Delivery", *The Journal of Computing in Small Colleges*, **19**(2), 30-38, 2003.

PISA, The Program for International Student Assessment, URL: http://www.pisa.oecd.org/links.htm (June 21, 2004)

Renaud, K., Barrow, J. & le Roux, P.: "Teaching Programming from a Distance: Problems and a Proposed Solution", *ACM SIGCSE Bulletin,* **33**(4), 39-42, 2001.

Rountree, N., Vilner, T., Wilson, B.C. & Boyle, R.: "Predictors For Success In Studying CS": *Proceedings of the 35th annual SIGCSE Technical Symposium on Computer Science Education,* Norfolk, Virginia, USA, 145-146, 2004.

Suhonen, J. & Sutinen E.: "Portfolio-Based Analysis in Developing Tool for a Web-Based Programming Project Course". *Proceedings of the IASTED International Conference on Computers and Advanced Technology in Education, CATE 2003*, Rhodes, Greece, 2003. ACTA Press. CD-ROM.

Sutinen, E. & Torvinen, S.: " Instructional Intervention In a Web-Based Programming Course", *Proceedings of the First Annual Finnish/Baltic Sea Conference on Computer Science Education*, University of Joensuu, Report A-2002-1, Koli, Finland, 53-60, 2001.

Tashakkori, A. & Teddlie, C.: *Mixed Methodology – Combining Qualitative and Quantitative Approaches,* Applied Social Research Methods Series, Volume 46. SAGE Publications, International Educational and Professional Publisher, Thousand Oaks, California, 1998.

Watanabe, S., Nakabayashi, T., Satoh, H., Jiang, T. & Oda, T.: "Web-based Educational System: Monitoring and Assisting Learners". In Cumming, G. (Eds.) *Advanced Research in Computers and Communications in Education.* IOS Press, Amsterdam, 693-700, 1999.

Weitzmann Institute of Science, Israel, URL: http://www.weizmann.ac.il/ (June 21, 2004)

Wiedenbeck, S., Ramalingam, V., Sarasamma, S., & Corritore, C. L.: "A Comparison of the Comprehension of Object-oriented and Procedural Programs by Novice Programmers.", *Journal of Interacting with Computers*, **11**, 255-282, 1999.

Wiedenbeck, S., LaBelle, D. & Kain, V. N. R.: "Factors Affecting Course Outcomes in Introductory Programming", *Proceedings of 16th annual Workshop of the Psychology of Programming Interest Group, PPIG 2004,* Institute of Technology, Carlow, Ireland, 97-110, 2004.

Wilson, B. C. & Shrock, S.: "Contribution to Success in an Introductory Computer Science Course: study of twelve factors.", *Proceedings of the 32nd SIGCSE Technical Symposium on Computer Science Education,* ACM Press, NY, USA, 184-188, 2001.

Xenos, M., Pierrakeas, C., Panagiotis, P.: "A survey on student dropout rates and dropout cousescauses concerning the students in the Course of Informatics of the Hellenic Open University", *Computers and Education*, **39**, 361-377, 2002.

Paper I:

Sutinen, E. & *Torvinen, S.*: "The Candle Scheme for Creating an on-line Computer Science Program – Experiences and Vision", *Informatics in Education*, 2003, Vol. 2, 93-102.

Informatics in Education, 2003, Vol. 2, No. 1, 93–102

The Candle Scheme for Creating an on-line Computer Science Program – Experiences and Vision

Erkki SUTINEN, Sirpa TORVINEN
Department of Computer Science, University of Joensuu
P.O. Box 111, FIN-80101 Joensuu, Finland
e-mail: {sutinen,sirpa.torvinen}@cs.joensuu.fi

Received: April 2003

Abstract. Distance learning programs have rapidly increased during the past few decades. In fall 2000 the University of Joensuu started to offer distance Computer Science (CS) studies to the high school students in surrounding rural areas of Joensuu. In this program high school students study the first year's university level CS studies over the web simultaneously with their regular high school studies. We describe the creation process of our virtual curriculum which is based the so-called Candle scheme. The Candle scheme search the most essential principles needed in on-line course design, supporting a student locally in her authentic learning needs via electronic tools in a light way. With the Candle scheme we have successfully focused in our design process on the most essential parts of the virtual study process. Our experiences of the Candle scheme in the creation process of the on-line CS program during years 2000–2002 indicate that the scheme is the functional one and expandable to other contexts as well.

Key words: Computer Science education, distance learning, on-line learning, virtual learning.

1. Introduction

In the past two decades all kinds of distance learning programs have rapidly increased. In the beginning teachers simply turned their learning material to the form of hypertext bringing plenty of web-based materials available all the way from the elementary schools up to the university-level. Farrell *et al.* (1999) made in year 1999 an extensive snapshot of the state of virtual education development in the major regions of the world. They defined *virtual education* as something that is heard with an increasing frequency, as the use of information and communication technologies becomes ever more present in conducting open and distance education. They considered that the label virtual is frequently used mutually with other labels like *open and distance learning, distributed learning, networked learning, web-based learning,* and *computer learning*. Current strategies typically involve the use of either synchronous or asynchronous digital networks for the delivery of courses, management of services like registration and records, and the provision of learner support services (Farrell *et al.*, 2001).

In a virtual learning environment, the roles of students and teachers are changing. Students have to be much more active, they are assumed to be motivated to find out information and search optional materials via Internet. Hope (2001) noticed that teachers and instructors are facilitators who support active technology-mediated student learning rather than sole sources of knowledge. When we change our paper-based teaching model to an interactive one we have to rethink the design and production of our learning resources (Sumner and Taylor, 1998). New media demand new practices; some of the traditional teaching methods are useful even in the web-based teaching, but distance learning has plenty of characteristic methods itself – like communication over the web, shared delivery of materials, interactive working over the web, or virtual team working. Dhanarajan (2001) has developed ten key issues that need to be addressed when designing virtual courses. The most essential demands are the pedagogical skills of teachers and educational administrators, access and equity, standards and assessments, staff training and development, and research. Several distance learning programs have been developed due to the pressure of fashion and trend rather than real need.

The creation process of our distance learning Computer Science (CS) program arose out of two factors; the expectations of the Finnish Ministry of Education funded the Finnish Virtual University project, and the need for higher quality CS teaching in high schools. We made plans for implementation in close co-operation with high school teachers, and created the Candle Scheme to guide us in the process. Thus the Candle scheme has some context dependent elements, like the Finnish educational system or the target group of the students. However, we believe that it can be expanded to other environments, like different kinds of student groups or other cultural contexts, by minor modifications.

2. First Year University Level Studies in CS over the Web

Our distance learning program has been developed as a part of the three-year project (2001–2003) to establish the Finnish Virtual University. One of the particular goals of our project is to develop new Computer Science education methods (Haataja *et al.*, 2001). In fall 2000, the Department of Computer Science at the University of Joensuu began offering, within the framework of that project, a university level Introductory Computer Science Curriculum (22.5 European Credit Transfer System credits, cps) to high school students in the surrounding rural region of Joensuu, extending this teaching experiment in the following fall to the rural region of the neighboring province as well. We have now students from a total of 20 rural high schools.

Table 1 gives the contents and schedule of courses during the 1.5 years period. The Curriculum provides basic knowledge of Computer Science to the students.

Students study independently over the Internet using the WebCT learning environment. Almost 98% of our studies are arranged over the web. During the 1.5 years study process students visit the campus area four times: in the beginning of the study process at the info meeting, two exams and, for the graduation. In addition, we have one week's summer school period where students have some lectures and plenty of teamwork and presentations.

Table 1

The contents and outline of courses in our on-line CS program

Title of the course	ECTS	Contents	Scheduled
Introduction to Computer Science	3 cps	General knowledge about IT and computing. Practical skills of using word processing and spreadsheet programs, basics of Unix.	First fall semester
Programming I	3 cps	The idea of algorithms, basic structures of programming in Java.	First fall semester
Programming II	3 cps	The basics of object-oriented programming (classes, objects, graphical programming and event handling).	First spring semester
Research fields of Computer Science	3 cps	Introduction to a selecting of research fields in Computer Science.	First spring semester
Programming project	3 cps	Independent work containing software design and programming.	First spring semester and summer
Design of algorithms	3 cps	An overview of the central issues of Computer Science, such as algorithms, computation and data structures.	Second fall semester
Hardware, computer architecture and operating system	3 cps	An overview of issues such as; architecture of computers, parsers, system software, databases, and information systems.	Second fall semester
Introduction to the ethics of computing	1.5 cps	General knowledge and practice in the ethics of computing.	Second fall semester

3. The Candle Scheme

The implementation was based on the request of high school teachers who wanted to improve the level of CS teaching in their schools. At the same time the Finnish Ministry of Education started the Finnish Virtual University program. We decided to combine these two interests and created the virtual introductory CS studies for high school students over the web.

3.1. *Structure of the Candle Scheme*

The instructional design of our program was constructed in close collaboration between high school and university teachers. We found that the contribution from the high school teachers was crucial for the development of the *Candle scheme*, a selecting of principles to guide the design process. Teachers helped us to coordinate the courses with the schedules of the high schools – these differ quite a lot from school to school. A group of high school teachers monitored the course during the whole period of 1.5 years (Haataja *et al.*, 2001).

Fig. 1 illustrates the main elements of the Candle scheme.

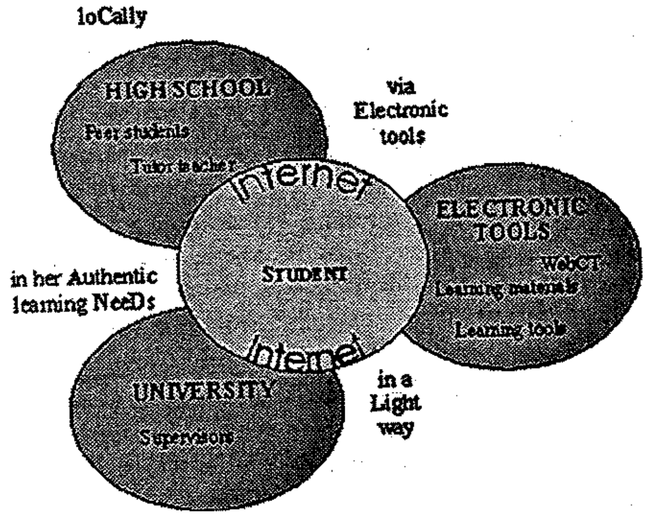


Fig. 1. The main elements in the Candle scheme.

The basis of our creation process was the needs of the students, so the student is the well-justified center point in Fig. 1. Three main elements, *university, high school* and the *Internet*, enclose the student. All learning materials and tools are offered to the students by university. Students can get support from tutor teachers and peer students both face-to-face in their own high school and over the web from other high schools as well. Supervisors are staff members of university and co-operate with high schools and with students over the web.

The Candle scheme is designed to support a student (Haataja *et al.*, 2001):

- ☐ loCally,
- ☐ in her **A**uthentic learning **N**ee**D**s,
- ☐ in a **L**ight way,
- ☐ via **E**lectronic tools.

In **C**andle, almost all *teaching is carried out via the Internet* (Fig. 1) while the students come from all around the surrounding rural region of the university. Students can study locally, at home or at their own high school, instead of spending their time and money for traveling between home and university. One of the major principles has been the equality between high schools; no matter where the high school is situated, students can take part in our program.

The c**AND**le scheme provides supports in students in their **A**uthentic learning **N**ee**D**s by offering *real human contacts* for the learners. The web site helps the students to communicate with the supervisors and peer students in multiple ways. *Supervisors* work from the university and hold a at least BSc or MSc degree in CS. In each high school we have one *tutor teacher*, who usually is either an IT or Mathematics teacher, who helps students especially in the beginning of distance studies. The main purpose of a tutor teacher is to ensure that everybody can use the learning environment and find the courses and assignments. Tutor teachers advice students with the practical problems like submitting exercises or installing Java compiler. In addition they act as contact persons between the university and high schools. The main responsibility of teaching lies on the supervisors. The students get valuable help via *peer students*, both locally and over the web. Real human contacts are also illustrated in Fig. 1.

While sophisticated user interfaces in many web-based educational settings might be confusing for novices we decided to construct our scheme in a Light way. Therefore, candLe keeps the user interface as simple as possible. To make the interface of the learning environment more student-friendly, we hired a fresh high school graduate to design the pages, and give us youthful ideas for assignments. We wanted also keep our web-based learning materials as light as possible, so even over the slow modem connections it is available to use or download our materials.

Another major design principle in candlE is to *link the printed learning material with an activating learning environment on the web*. The printed material gives students an overall structure and knowledge of the topic while the web materials guide the learners step by step in minor weekly pieces. Additionally we used Electronic learning tools (Fig. 1) like activating *visualizing tools*, e.g., Jeliot, as students' virtual laboratory. *Exercises* play an important role in the learning process – especially in programming, learning-by-doing is the most effective way to learn. Students have to complete at least one third of all the course assignments; those who exceed the minimum requirement get bonus points for the grade. The bonus point system has been a motivating factor for increasing the response rate.

3.2. *Learning Methods: Candle in Action*

Designing appropriate learning methods and tools for distance learning context is a challenging task. Earlier, e.g., Ellis *et al.* (1999), Cordani and Tucker (1998) and Watanabe *et al.* (1999) have discussed the difficulties to select tools appropriate for use distance learning in higher education. Anyway, the common opinion is that a web-based learning environment should provide a rich assortment of learning methods and learning tools in order to provide for each student suitable methods which help him/her in the most efficient way.

Four courses out of eight in our program have the following structure: students study both printed and web-based material; submit weekly assignments and their learning incomes evaluated by exam. Students submit weekly from four to five exercises. Supervisors check all of them, give comments and more detailed advice when needed. Our web-materials include multiple kinds of visualizations: both interactive and non-interactive applets, and Flash animations. We have also successfully used a program animation program, Jeliot 2000 (Ben-Bassat Levy *et al.*, 2003), in order to deepen the study process in the Programming I-course. Table 2 presents the learning methods and tools used in our program.

In the programming project students develop either individually or in pairs a small piece of software and document it. In spring 2003 we start to use the Problem Processing Assistant (PPA) (Suhonen and Sutinen, 2003) which is developed to support students all the way through their programming process, from conceptual and technical design through implementation up to testing.

Table 2
Learning methods used

Method	C	AND	L	E
Printed material + web-based material, assignments and exam	x	x	x	x
Animations (Jeliot 2000, Flash, applets)	x	x	x	x
Programming project + PPA	x	x		x
Virtual team work	x	x		x
Essays	x			
Portfolios		x		x
Lectures		x		
Face-to-face team work		x		
Self-tests	x	x		x
Optional virtual team work (discussion forum, IRC, chat, e-mail)	x	x		x
Internet (search for information, contacts etc.)	x		x	x

During one of our courses we have used team work over the web. The members of the group are from different high schools, they work together using the WebCT learning environment and its group work area, producing a common thesis. In additional each student writes his/her personal portfolio where he/she reflects the impacts of virtual teamwork. In "Introduction to the ethics of CS" students write individual essays where they have to consider the ethical point of view in a given case, like illegal copies or using the Internet for terrorism. In this course students have to write a portfolio as well. Students reflect on the development of their attitudes or opinions towards given topic.

During the 1.5 years' period we have a one-week long contact learning period. During this week we have lectures in "Introduction to the Algorithms" course, group work related to both the Algorithm and Ethic courses and feedback session on the programming project.

Based on our experiences we have noticed that a significant part of successful learning process lies in students' own activities like using self-tests, activity in optional virtual team working and use of the Internet (Table 2). The activity of using the discussion forum in the WebCT varied course-wise; the activity is highest in programming courses but not so lower for easier courses. An indication of high motivation is that our students have created discussion groups in IRC where they discuss freely our courses, assignments and problems in the study process.

4. Experiences

The on-line CS program was arranged for the first time in fall 2000. For the time being the third group of high school students study in our on-line program. The study process simultaneously with regular high school studies has proved to be a hard one and resulted in several dropouts during the hardest programming courses. Our dropout rates have varied from zero to 42.9 %, i.e., depending on the course. Earlier findings show that dropout rates in university-level distance learning are, in general, definitely higher than those in conventional university studies. They vary depending on the adopted distance education system and the subject being studied; in Europe between 20% and 30% while in Asian countries they around 50% (Xenos *et al.*, 2002).

After the first year's experiences we evaluated our courses and arrangements by a formative evaluation scheme (Meisalo *et al.*, 2002a) which led to improving especially the difficult parts of the programming courses. In our program the main reason to drop out have been both in year 2000 and 2001 the lack of time. Students have obviously had difficulties to estimate and prioritize the time required for the distance study process along with their high school studies.

In Table 3 we present the amount of students during years 2000–2002. Numbers of registered students include all students who have registered – no matter whether they start to study at all. In each year there have been 10 to 50 students who actually did not do anything at all. Figures in parentheses give the amounts of female students. It is significantly lower than that for our ordinary students at the university. At the university the distribution between female and male students is almost even but in virtual CS studies the main part of, both started and graduated, students are male. The survey of Xenos *et al.* (2002) illustrates the very same results of female students' reluctance to start university level distance courses, and the dropouts in general.

Programming has proved to be the most difficult part in our program. In brief we can characterize that *students who pass Programming I and Programming II, will very probably pass the rest of the courses as well.* Cornell and Martin (1997) found that a students who has passed at least some distance courses, would probably also pass others in the future. Our findings agree with them (Meisalo *et al.*, 2002b).

Furthermore we have been satisfied with our visualization tools: both interactive and non-interactive applets, Flash-animations and Jeliot 2000 program. Especially mid-performers benefit from visualizations; an observation analyzed in detail by Ben-Bassat Levy *et al.* (2003). Masters *et al.* (2002) created several Java applets for the electrical engineering course and found out that simulations and on-line materials made it easier for students to take a more active role in learning. Animations have helped especially the understanding of difficult concepts. Our experiences support the findings of Master *et al.* (2002): animations of arrays, loops or binary search trees have helped students to understand these crucial structures of programming.

Table 3

Number of students in our on-line program

Year	Number of high schools	Registered	Graduated
2000	12	89 (14)	20
2001	21	184 (24)	37
2002	19	156 (22)	Ongoing

5. Future Vision

The Candle scheme is an outcome from a process where we designed an on-line course according to the actual needs of the potential users. The pros and cons of the scheme are rooted in this starting point. In particular, they can be seen in the future applicability of the Candle scheme.

The main advantage of the scheme is the relatively fast results it guarantees. This is based on, first, being sensitive to the students' authentic learning needs, and secondly, implementing the course in a light way. In our case, we had the first course implemented just after three months from the beginning of the project. This means, however, that the design and implementation process is followed by another process, called formative evaluation. Formative evaluation, utilizing both quantitative and qualitative methods for analyzing users' experiences and learning outcomes, identifies the weaknesses of the on-line program. The feedback must be taken into account by immediate modifications of the course.

The scheme emphasizes an extensive use of digital tools, like visualization environments. This is in contradiction to the commonly accepted emphasis on constructing – mainly static – learning materials. We did not even try to replace a textbook but strived after using ICT in a functional way. Thus, we created smart learning gadgets for identified topics where students face learning difficulties. These tools can easily be transferred to other contexts as well.

Motivation, rather than the needs required by a certain paradigm of the subject area, has been the practical approach for organizing our learning materials. We think that on-line students need to be able to use their knowledge as fast as possible. In the area of programming, this means that exercises for producing animations have been more important than building a solid conceptual background.

In the future, we hope that we can light our Candle not only around our university but in other cultural contexts and for other subject areas as well. In fact, the candling process is about learning rather than designing. There are no more teachers at one end of the cable, students at the other one. There are just learners with slightly different roles. The learning environment is not primarily a clean and polished one, but of an inspiring and provocative character. A zone where one fights for understanding and mutual comprehension.

6. Conclusions

We developed our on-line Computer Science program as a part of the Virtual University of Finland funded by the Finnish Ministry of Education during 2001–2003. At the same time high school teachers of the surrounding rural area of the university were keen on to improve their CS teaching and asked whether the university could offer first year's university level CS studies to the high schools. Our department decided to combine these both interests and developed within this project virtual a CS program to high school students.

The design model used in our on-line course creation process, the Candle scheme, proved to be a functional one. With Candle we have succeeded to focus our efforts on the highest priority items in the creation and evaluation process. Candle combines the essential elements in the on-line learning process from the view both of students and the university. Candle takes care of the students' need in learning process offering multiple ways of learning. In on-line courses equality of students and availability of services are the most essential elements, and Candle provides these both elements. Watanabe *et al.* (1999) noticed that the most difficult part in the creating process of distance learning programs seems to be to estimate the need for support that students will need during the study process. The Candle scheme focuses on that in multiple ways, on-line support complemented with face-to-face support given by tutor teachers at the local level.

Our scheme gives main guidelines for teachers who are going to create on-line courses, and we believe that Candle is likely expandable to other subjects or other countries as well. Our future interest is apply the Candle scheme to the developing countries for creating on-line CS programs in the third world.

References

Ben-Bassat Levy, R., M. Ben-Ari and P.A. Uronen (2003). The Jeliot 2000 program animation system. *Computers and Education*, **40** (1), 1–15.

Cordani, J.R., and R.J. Tucker (1998). Tools for higher education distance teaching. In *Proceedings of the 26th SICUCCS Conference on User Services*. Bloomington, USA, pp. 71–76.

Cornell, R., and B.L. Martin (1997). The role of motivation in web-based instruction. In B.H. Khan (Ed.), *Web-Based Instruction. Educational Technology Publications*. New Jersey, USA, pp. 93–106.

Dhanarajan, G. (2001). Distance education: promise, performance and potential. *Open Learning*, **16** (1), 61–68.

Ellis, A., D. Hagan, J. Sheard, J. Lowder, W. Doube, A. Carbone, J. Robinson and S. Tucker (1999). A collaborative strategy for developing shared Java teaching resources to support first year programming. In *Proceedings of the 4th Annual SIGCSE/SIGCUE on Innovation and Technology in Computer Science Education*. Krakov, Poland, pp. 84–87.

Farrell, G.M. (Ed.) (1999). A development of virtual education: a global perspective. In *The Commonwealth of Learning*. `http://www.col.org/virtualed/index.htm`

Farrell, G.M. (Ed.) (2001). *The Chancing Faces of Virtual Education. The Commonwealth of Learning*. `http://www.col.org/virtualed/index2.htm`

Haataja, A., J. Suhonen, E. Sutinen and S. Torvinen (2001). High school students learning computer science over the web. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* (IMEJ), **3** (2). `http://imej.wfu.edu/articles/2001/2/04/index.asp`

Hope, A. (2001). Quality assurance. In G.M. Farrell (Ed.), *The Chancing Faces of Virtual Education. The Commonwealth of Learning*. `http://www.col.org/virtualed/virtual2pdfs/V2_chapter7.pdf`

Masters, J., T.M. Madhyastha and A. Shakouri (2002). Educational applets for active learning in properties of materials. *The 2002 Frontiers in Education* (FIE2002), Boston, USA. `http://fie.engrng.pitt.edu/fie2002/papers/1070.pdf`

Meisalo, V., J. Suhonen, E. Sutinen and S. Torvinen (2002a). Formative evaluation scheme for a web-based course design. In *Proceedings of 7th Annual Conference on Innovation and Technology in Computer Science Education* (ITiCSE 2002). ACM Press, Aarhus, Denmark, pp. 130–134.

Meisalo, V., E. Sutinen and S. Torvinen (2002b). How to improve a virtual programming course? *The 2002 Frontiers in Education* (FIE2002), Boston, USA. `http://fie.engrng.pitt.edu/fie2002/papers/1180.pdf`

Suhonen, J., and E. Sutinen (2003). Portfolio-based analysis in developing tool for a web-based programming project course. *Accepted to the IASTED International Conference on Computers and Advanced Technology in Education* (CATE 2003), Rhodes, Greece.

Sumner, T., and J. Taylor (1998). New media, new practices: experiences in open learning course design. In *Conference Proceedings on Human Factors in Computing Systems*, Los Angeles, CA, USA, pp. 432–439.

Watanabe, S., T. Nakabayashi, H. Satoh, J. Tianfeng and T. Oda (1999). Web-based educational system: monitoring and assisting learners. In G. Cumming *et al.* (Eds.), *Advanced Research in Computers and Communications in Education*. IOS Press, pp. 693–700.

Xenos, M., C. Pierrakeas and P. Panagiotis (2002). A survey on student dropout rates and dropout cousescauses concerning the students in the course of informatics of the Hellenic Open University. *Computers and Education*, **39**, 361–377.

E. Sutinen is a professor of computer science at the University of Joensuu. After receiving his PhD from the University of Helsinki in 1998, he has been working also at Purdue University, USA, and University of Linköping, Sweden. Currently, his main research focus is educational technology from the computer science point of view: how to design functional ICT tools to overcome learning barriers. He is a director for the Thinking Tools for the Net Project, funded by the National Technology Agency, Finland. He is also involved in applications such as Jeliot for programming visualizations, support collaborative learning by writing, and on-line computer science program for teaching.

S. Torvinen, MSc (2001) is a PhD student in CS. Her special interest is in educational technology and difficulties in virtual programming courses. She works as manager of the on-line CS project at the Department of Computer Science in the University of Joensuu. Her educational background is in educational technology in the field of computer science. Her research deals with the on-line CS program focusing to find the indicators of the learning process for the intervention in the virtual programming courses.

"Žvakės" schema kuriant nuotolinio mokymo kompiuterijos programą: patirtis ir vizija

Erkki SUTINEN, Sirpa TORVINEN

2000 m. Joensuu universiteto suprojektuota nuotolinio mokymo programa buvo tris metus trukusio projekto, skirto įkurti Suomijos Virtualų universitetą, dalies tąsa. Vienas iš specialių projekto tikslų – naujų kompiuterijos mokymo metodų parengimas ir plėtotė. Dabar šiuo metodu mokomi mokiniai iš dvidešimties kaimo vidurinių mokyklų.

Straipsnyje apibūdinamas virtualios mokymo programos kūrimo procesas. Programa remiasi vadinamąja "žvakės" schema (Candle scheme). "Žvakės" schema atitinka svarbiausius principus, taikomus nuotolinėms studijoms bei padeda įvairiose vietose gyvenantiems studentams patenkinti jų individualius poreikius pakankamai nesudėtingu būdu: naudojantis elektroninėmis priemonėmis. Pasitelkus "žvakės" schemą susikoncentruota ties esminių virtualaus mokymo dalių kūrimu.

Straipsnyje aprašytos autorių sukurtos nedidelės mokymo priemonės, skirtos įvairioms temoms. Jas nesunku pritaikyti ir kitiems dalykams. Projektuojant mokymo medžiagą vyravo praktinis požiūris veikiau į motyvaciją nei į poreikius, reikalingus tam tikrai mokomojo dalyko paradigmai. Programavimo srityje tai reiškia, kad pratimų animacija labiau svarbi nei tvirto konceptualaus pagrindo suformavimas. Autoriai tikisi ateityje uždegti savo "žvakę" ne tik aplink Joensuu universitetą, bet ir kituose kultūriniuose kontekstuose bei kitose mokslo srityse. Iš esmės, šis procesas labiau susijęs su mokymusi, nei su projektavimu. Autorių sukaupta patirtis apie "žvakės" schemos taikymą sudarant kompiuterijos programą, skirtą nuotolinėms studijoms, rodo, jog ši schema yra funkcionali bei jos panaudojamumas gali būti išplėstas mokant ir kitų mokslo sričių.

Paper II:

Meisalo, V., Suhonen, J., Sutinen, E. & *Torvinen, S.:* "Formative Evaluation Scheme for a Web-Based Course Design", *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2002)*, Aarhus, Denmark, June 24-26, 2002. ACM Press, 130-134.

Formative Evaluation Scheme for a Web-based Course Design

Veijo Meisalo
Department of Teacher Education
University of Helsinki
P.O. Box 38
FIN-00014, Helsinki, Finland
+358 9 1912 8129
veijo.meisalo@helsinki.fi

Jarkko Suhonen
Department of Computer Science
University of Joensuu
P.O. Box 111
FIN-80101, Joensuu, Finland
+358 13 251 7917
jarkko.suhonen@joensuu.fi

Sirpa Torvinen
Department of Computer Science
University of Joensuu
P.O. Box 111
FIN-80101, Joensuu, Finland
+358 13 251 5272
sirpa.torvinen@joensuu.fi

Erkki Sutinen
Department of Computer Science
University of Joensuu
P.O. Box 111
FIN-80101, Joensuu, Finland
+358 13 251 7934
erkki.sutinen@joensuu.fi

ABSTRACT

In Fall 2000, University of Joensuu offered for the first time a Web-based university level Computer Science course to high school students in the surrounding rural region of North Karelia. To improve the course design, a formative evaluation scheme was developed for monitoring learning outcomes and identifying existing problems, pedagogical as well as technical. An analysis of the feedback given by those who dropped out of the pilot program offered important insights to the difficulties encountered by the students especially during the programming course. The main reasons for quitting were the problems to synchronize high school and university studies, given tight time constraints. Use of arrays and designing methods proved to be among the most difficult topics of programming. Based on the analyzed feedback, the curriculum has been modified to fit better to students' potential.

Categories and Subject Descriptors

K.3.1. **[Computer Uses in Education]**: Collaborative Learning, Computer-Assisted Instruction, Computer-Managed Instruction, Distance Learning.

General Terms

Design, Experimentation, Human Factors, Management

ITiCSE'02, June 24-26, 2002, Aarhus, Denmark.

Keywords

Computer Science Education, Distance Learning, Evaluation, Virtual Learning

1. INTRODUCTION

Difficulties in learning to program among the first year Computer Science students are well-known. In distance education these problems become even more evident: there is a wide experience that far more effort is needed in designing courses than in face-to-face teaching.

In Fall 2000, University of Joensuu offered for the first time a Web-based university level Computer Science program to high school students in the surrounding rural region of North Karelia. Roughly half of the contents dealt with programming. To get a common pedagogical framework for the course design, we developed the Candle model [4]. The main design principles in the Candle model are:

- offer students real human contacts (online instructors and tutor-teachers at the schools);
- link the printed learning material with an activating learning environment on the web;
- utilize the principle of learning-by-doing, e.g. apply visualization tools; and
- keep the user interface simple and use an otherwise light technology.

Because of the novelty of our teaching approach, there was an urgent need for improving the pilot courses as fast and deeply as possible. This was understood as the only way to market the program also for the coming high school generations, and at the same time, to make Computer Science studies at a university an attractive choice after the high school.

The methodological goal of our study was to develop a formative evaluation scheme for assessing and improving the quality of a Web-based programming course. In particular, we were interested in an ideal course design for supporting the study process of programming, especially among novice programmers. The methodological goal leads to the formative evaluation scheme.

The practical goal was to get insight into the high school students' learning difficulties. Therefore we focussed on the students who dropped out of the web-based course. As a research task, we wanted to identify the topics of programming that these high school students regarded as difficult to understand. We also searched for the main reasons to drop out of the pilot course during the school year 2000-2001. To reflect our findings, we analyze the results of the examinations and the impact of doing exercises. Carbone and Kaasbøll (1998) [2] recommend a simple yet powerful way to evaluate the teaching innovations at computer science education. We believe that we have succeeded to accomplish this. The practical goal leads to the ideas and proposals for further development of the courses.

Although high schools students differ from the normal university student body, we believe that their experiences and feedback are important for developing academic teaching of programming among regular students. In particular, they are tied to the university practically only by the web-based course and so their criticism could be interpreted more neutral and subject-centered compared to regular students that might have other motivations to their criticism. Moreover, they belong very clearly to the group of novice programmers whose teaching is specially challenging.

2. THE VIRTUAL CERTIFICATE PROJECT

In Finland, the Ministry of Education is funding a three-year project to establish the Virtual University of Finland, during years 2001-2003. One of the particular goals in the project is to develop new methods for science education. The three universities in eastern Finland, University of Joensuu, University of Kuopio, and Lappeenranta University of Technology, work jointly in the virtual university project. One of the concrete objectives is to create a web-based learning environment for an introductory Computer Science course, intended for high school students [5].

The project is called *Virtual Certificate*, indicating that high school students can take 15 credits of Computer Science studies in one and a half year via the Internet. Courses will give students basic knowledge of three main domains: Introduction to Computer Science (5 credits), Basics of Programming with Java (7 credits), and Preliminaries of Computers (3 credits). In the Finnish system, each credit equals 40 hours of studying; 120 credits are required for the Bachelor's degree. One of these credit point equals two credit points in the International Credit Transfer System. Thus, after passing all the offered 15 credits a student has the first year Computer Science studies completed. Moreover, if the student passes the program with grade 2/3, she will be free to enter the university as a Computer Science major.

Why have we decided to offer Computer Science studies for high school students? There are three major viewpoints to support our decision.

First there are very *few qualified Computer Science teachers* in schools. Moreover, many students live far from universities and from each other. Therefore, it is not possible to solve the problem by sending visiting teachers to these schools or invite the students to the university.

Secondly there is a great *need for professionals* in the computing industry. As a partial solution, Computer Science departments have to be able to attract more students. By entering the university with the first courses completed already at high school, the students will have better chances to finish their studies.

From the research point of view, it is important to *assess the benefits Internet offers when integrated in the learning environment*. Like the extensive popularity of mobile technology in Finland indicates, high school students are enthusiastic about new opportunities in web-based education. Moreover, they are open to give very frank and well-grounded feedback on the systems they use.

3. PEDAGOGICAL CHALLENGES OF A WEB-BASED PROGRAMMING COURSE

Pedagogically, a pilot program like the present one puts several challenges on the researchers: There is seldom enough time for designing a pilot course. However, there is a need to start the pilot course more or less immediately. This means a sort of learning-by-doing method in designing — hopefully there is enough capacity for simultaneous research effort to prevent repeating many mistakes.

There have been several studies on the importance of using methods of distance education to augment the possibilities of rural schools to offer expert teaching [6]. It is most important also in the equal opportunities perspective. This is one of the reasons for launching the national Virtual University Project in Finland.

It is not at all obvious how to design and implement a university level programming course in a way that it motivates high school students and gives them opportunities to progress efficiently in their studies. Especially, it is difficult to activate true interaction and group work in problem solving over the Internet. Problems of tutoring-at-distance are related to this.

Distance education arrangements should be far more flexible than in the context of ordinary courses on campus. Especially in Finland each school has the freedom to organize their time schedules with very few constraints. Students were on three different grade levels, which added to the challenge. Students taking the national Matriculation Examination have practically no time for other studies over several weeks during the spring term.

Many difficulties in learning programming seem to be independent of the programming language in use. However, it is important to activate on the school level at least some visualization methods. For young students, Papert (1980) [8] classically recommended the Logo language and drawing exercises, and Lavonen et al. (2001) [7] study an icon-based approach. However, this course was designed closely on the grounds of the on-going courses on the university campus.

4. METHOD

We evaluated the second completed course, Programming, part 1 (2 credits), which covers the introduction to the concepts of programming in Java. We analyzed the course design from two viewpoints:

First we analyzed *the learning outcomes*: the examination papers and exercises of *all the students* in the course were inspected. Secondly we looked after *students' attitudes*: students who *dropped out* from the curriculum either during this course or after passing it were interviewed.

The questionnaire sent to the drop-outs, or the *test group*, included ten questions. The first two questions were focused on the learning environment and the course materials on it. The following questions dealt with the study process: which fields of programming the students regarded as difficult, are there any needs for support measures like on-line lessons, videoconferencing or chat in WebCT. Students were also asked the main reason for dropping out. The last two questions were open questions: what would the students like to change in the course, are there any ideas to improve the quality of the materials or arrangements of the course etc. A total of 25 students completed the query form. A more detailed follow-up questionnaire was sent to six volunteers by e-mail.

5. RESULTS OF THE COURSE DESIGN EVALUATION

5.1 Learning Outcomes

To participate in the examination, student had to complete 1/3 of the given 50 exercises. Out of 79 students, 76 submitted at least one exercise and 62 of them returned the compulsory amount of exercises. Out of 62 students allowed to participate 56 took part in the exam; a total of 20 in this group dropped out after the course.

Table 1 Results of exam by passed and dropped out students

[illegible]	[illegible]	[illegible]	[illegible]
1: if-statement	6	5.04	4.08
2: loop and if-statement	6	4.69	2.90
3: arrays and random	6	4.11	1.90
4: applets	6	3.67	0.45
[illegible]	[illegible]	[illegible]	[illegible]

The exam included four questions on the following themes:

- Question No.1: Declare the meaning of the if-statement
- Question No.2: Writing a Java program with an if-statement and a loop
- Question No.3: Concept of randomized numbers
- Question No.4: Coding of Java applet

Table 1 presents the results of the exam by those students who continued and those who dropped out. One can clearly see from the table that especially from the difficult questions the drop-outs got less points than those students who continued.

5.2 Students' Attitudes Towards the Course

A total of 39.5 % of students dropped out during the first programming course. According to Cornell & Martin [3], it is very common that as many as 30–50% of students drop out during virtual courses. They also found out that a student, who has passed at least some distance courses, will probably also pass others in the future. Our experiences are very similar to the above.

Figure 1 shows the main reason to drop out of the course. A total of 44% of students quitted because of the lack of time. Also 20% of students founded exercises too difficult and 8% did not pass the retake exam.

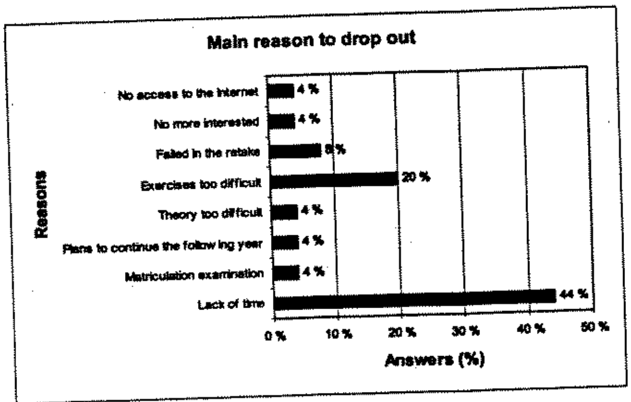


Figure 1 Main reasons to drop out

Table 2 shows the topics that were considered difficult by test group. Quite unexpectedly, as many as 48% of students regarded if-statements as hard to learn. Surprisingly, a total of 72% considered arrays difficult.

Table 2 Topics of programming feeling as difficult

[illegible]	[illegible]
Variables and symbols	24 %
Assignment statement & input/output	32 %
If-statement and logical operations	48 %
Loops: for, while and do-while	24 %
Arrays and random	72 %
Methods	60 %
Applets and graphics	60 %
Making animations	56 %

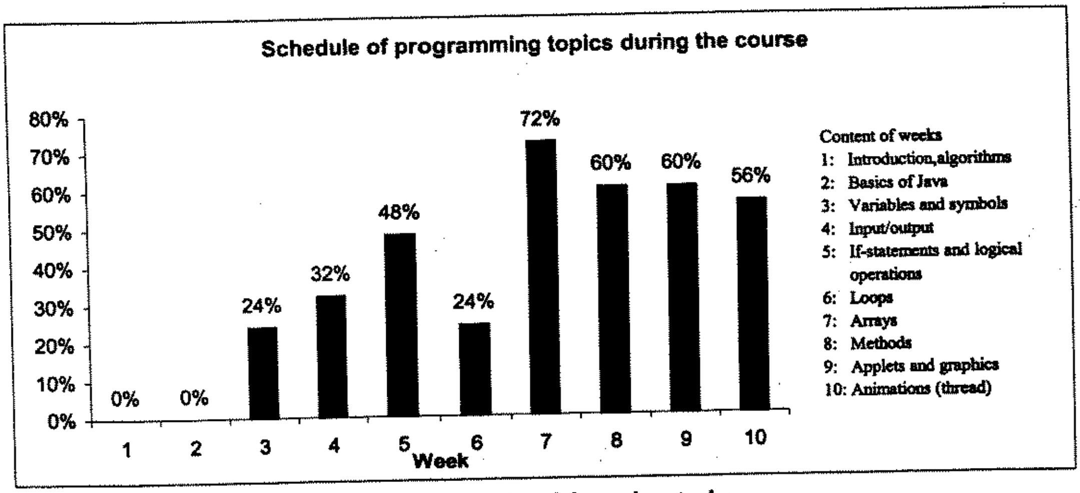


Figure 2 Difficulty of the various topics

Let us then look at how the topics were scheduled along the course. Figure 2 shows that most of them occurred in the last half of the course and at the same time there was also an exam period found in Figure 2: the main reason for dropping out the course was lack of time. Partly, this is due to the instruction-centric design of the course: instead of allowing the students to learn the topics according to their personal timing preferences, they were required to submit their exercises weekly—if a tough school period was hardened even more by difficult program topics and related assignments, that was students' problem, not instructors' one. Therefore, our findings indicate there is a need for a more flexible schedule, or alternatively, the topics of the course should be organized in a new way.

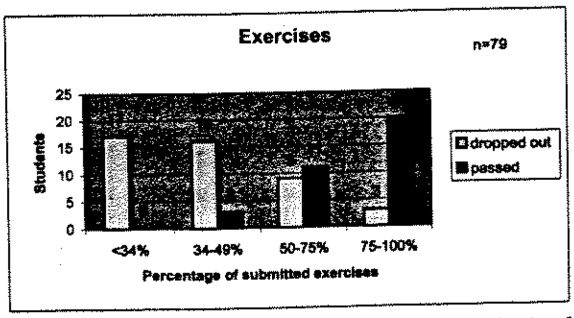


Figure 3 How dropped –out and passed students submitted exercises

In the Candle model, one of the key ideas is learning by doing. If a student does not have time enough to do exercises, she seems to confront serious problems with the examination. Figure 3 shows how dropped-out students and passed students submitted exercises. In Figure 3 we can see that the main portion of dropped out students submitted less than half of the exercises. On the other hand, slightly over 90% of passed students (31 out of 34) returned at least 50% of exercises. The benefit of doing for learning is obvious. On the other hand, some of the exercises were and will be hard by nature. There will always be students who cannot make in high schools. So quite many of the students had difficulties to synchronize their high school studies with the "Virtual Certificate" courses. This correlates directly with the information use of these exercises to improve their learning outcomes. However, the clear correlation between the number of submitted exercises and passing the course illustrates the need for a more careful content analysis of the exercises. In particular, the course material should have a continuum of assignments, in terms of difficulty.

When we review the results of the examination, we find out the phenomenon shown in Table 2. If we look at the difficult subjects and questions in exam, the exam was quite difficult for those who dropped out. All the questions dealt with topics which students founded difficult. On the other hand we can see how important it is to do exercises. If we compare the information of Figure 3 and Table 1, we find out that passed students (who mainly made more than one half of the exercises) got also very good points in exam. Instead, dropout students did not do very well exercises and even in the exam their results are quite poor.

6. CONCLUDING REMARKS

The methodological goal of our study was reached: we designed and applied a formative evaluation scheme for further development of the course. Moreover, the evaluation method gave answers to our practical questions.

This study shows the benefit of exercises for learning programming. The analysis emphasizes also the importance of an appropriate schedule of the course. Most of the drop-outs discontinued their studies due to the lack of time. For the following school year 2001-2002, we allocated more time to topics like arrays, methods, random numbers, and coding applets. On the other hand, we also cut down the number of exercises but at the same time increased the number of examples and easy optional practices to the Web-based course material. It will be interesting to see the impact of these changes to the success of studies in year 2001-2002.

How can the evaluation scheme be improved? In the ongoing semester some of the tutor-teachers working in the local high schools are participating in continuing education program at the University of Joensuu. In this program teachers take an evaluation course (2 credits), where they collect data from the students. This way we can deepen the formative evaluation process of the courses, in order to get better understanding about students' performance.

On the other hand it would be useful to expand the evaluation to include all the students, not particularly those who dropped out. To get new dimensions to the analysis also the high performers' opinions should be taken into consideration. One interesting outcome of the expanded analysis would be information about the characteristics of the students who succeed in their studies.

By combining the evaluation of the learning results and the analysis of the student feedback we were able to get valuable information about the most problematic aspects of the web-based programming course. If we had concentrated for example solely on the students' feedback, the gathered information would have been insufficient. In addition, in some cases the opinions of the drop-outs are not totally neutral. For example, the frustration of not succeeding in the course can interfere the feedback. There will always be students who think that the learning difficulties are someone else's fault, not theirs.

Although the need for a deeper analysis of the course design is obvious, we are still convinced that a simple method to collect the data must be used. Data from sources like questionnaires, examination marks, literal quotations from students, and submitted assignment materials are easy to collect. These kinds of methods for collecting the information remain as a basis of our evaluation scheme also in the future.

From the practical point of view there are naturally various ways to improve teaching settings:

- It might be beneficial to develop the courses to have a more efficient contact teaching and group-work period in the beginning of each course. However, in our experience high school students are reluctant to participate in these kinds of activities because of the long distances.
- It might be feasible to recommend that the course could be spread over three high school years, but that exceptionally talented students could do it even in one year. On the other hand, this might cause too much difficulties for practical arrangements.
- It should be possible to use the learning environment more efficiently for group work and to activate creative problem-solving methods in doing the exercises [1]. Nevertheless, the common platforms (e.g. WebCT) designed for web-based courses lack the proper tools for collaborative work, especially discussing programming exercises.
- An electronic portfolio for each student could essentially help tutoring. Portfolio helps students to take responsibility, to see gaps in their learning, to identify strategies that support learning and to set goals for the future [9].

7. REFERENCES

[1] Carbone, A., Doube, W., Ellis, A., Hagan, D., Lowder, J., Robinson, D., Sheard, J., and Tucker, S. A collaborative strategy for developing shared Java teaching resources to support first year programming. In Proceedings of the 4th annual SIGCSE/SIGCUE on Innovation and technology in Computer Science education (Krakow, Poland, June 1999), 84-87.

[2] Carbone, A., and Kaasbøll, J. J. A Survey of methods used to evaluate computer science teaching. Proceedings of ITiCSE '98 (August 1998), ACM Press, 41-45.

[3] Cornell, R. & Martin, B.L, "The Role of Motivation in Web-Based Instruction", In Khan, B. H. (Eds.), Web-Based Instruction. Educational Technology Publications (New Jersey, USA, 1997), 93-106.

[4] Haataja, A., Suhonen, J., Sutinen, E., and Torvinen, S. High School Students Learning Computer Science over the Web, Interactive Multimedia Electronic Journal of Computer-Enhanced Learning (IMEJ), 3(2), October 2001. http://imej.wfu.edu/articles/2001/2/04/index.asp

[5] Haataja, A., Suhonen, J., and Sutinen, E. How to Learn Introductory Programming over Web. In Proceedings of the 7th International Conference of European University Information Systems (Eunis 2001), (Berlin, March 2001), Humboldt-University, 309-312.

[6] Kynäslahti, H. Act Locally, Th/Link Translocally. An Ethnographic View of the Kilpisjärvi Project. Department of Teacher Education, University of Helsinki, Media Education Publications 10, 2001.

[7] Lavonen, J., Meisalo, V., Lattu, M., and Sutinen, E. Concretizing the Programming Task: A Case Study on Icon-Based Visual Programming in Comprehensive School. Manuscript 2002.

[8] Papert, S. Mindstorms: Children, Computers and Powerful Ideas. The Harvester Press, 1980.

[9] Porter, C., and Cleland, J. The Portfolio as a Learning Strategy. Boynton/Cook Publisher, Portsmouth, 1995.

Paper III:

Meisalo, V., Sutinen, E. & *Torvinen, S.*: "How to improve a virtual programming course?", *The 32nd ASEE/IEEE Frontiers in Education Conference, FIE 2002*, November 6-9, 2002, Boston, MA, USA. CD-ROM.

HOW TO IMPROVE A VIRTUAL PROGRAMMING COURSE?

Veijo Meisalo[1], Erkki Sutinen[2] and Sirpa Torvinen[3]

Abstract *– The University of Joensuu started to offer virtual Computer Science studies to high school students in the surrounding rural area of Joensuu beginning in fall 2000. We evaluated the first programming course and modified the course based on the analyzed feedback of the students who dropped out. In fall 2001, we branched out to the rural region of a neighboring province and evaluated the differences between the new and old areas. The percentage of dropouts is somewhat larger in the new area. The course arrangements are the same in both areas, but we found factors, such as the knowledge of the course standards, the experience and attitudes of tutor-teachers, the students' own activities, as well as the students' earlier experience in programming, which may have influenced the dropout phenomenon. We are using stepwise dropout treatment during the first programming course.*

Index Terms – computer science education, distance learning, dropouts, programming, virtual learning

INRODUCTION

The Finnish Ministry of Education is funding a three-year project during 2001-2003 to establish the Virtual University of Finland. One of the particular goals in the project is to develop new Computer Science education methods [5]. In the fall of 2000, the Department of Computer Science at the University of Joensuu began offering, within the framework of that project, a university-level Introductory Computer Science Curriculum (22.5 European Credit Transfer System credits, cps) to high school students in the surrounding rural region of the province of Pohjois-Karjala. Almost half of the Curriculum concerned programming in Java.

We evaluated in this action research oriented case study the first programming course. We focused on students who dropped out either during the course or right after it. On the basis of the analyzed feedback, the curriculum was modified for the following academic year to better suit the students' potential.

In the fall of 2001, we extended this teaching experiment to a rather similar rural region in the neighboring province of Etelä-Savo. We now have students from a total of 21 rural high schools. In this report, we present the contents and arrangements of this virtual course, the feedback collected during the first year, and describe the changes made. We also analyze the percentage of dropouts both in the old and new areas, and discuss the factors that might help decrease the number of dropouts.

STUDYING THE VIRTUAL COURSE

The Curriculum of the Virtual University Course (22.5 cps) includes nine different courses that will give students a basic knowledge of three main domains: Introduction to Computer Science (7.5 cps), Basics of Programming (10.5 cps) and Preliminaries of Computer Architecture and Operating Systems (4.5 cps). Students study independently via the Internet, but each high school has one tutor-teacher, who helps students especially at the beginning of the studies. He or she may, for example, advise on the use of WebCT or how to install the Java compiler. However, the main responsibility for all the courses lies with a supervisor, who is a staff member in University's Computer Science department.

Most of the courses have a similar structure: students work from both printed and web-material, they do weekly assignments, and their learning outcome is evaluated through an exam. Almost all teaching is carried out via the Internet using a WebCT learning environment. One of the design principles was to link the printed learning material with a learning environment on the Web [5], designed to activate students.

Each student learns in his or her own personal way and the learning environment should offer a rich assortment of learning tools for different kinds of students [3]. Our Web material is composed mainly in HTML and it supports the printed material. The Web material brings the most essential parts of the course to the learners and quickly shows the student the structure of the domain. Additionally, the Web material includes examples and optional exercises, visualizations using MacroMedia Flash 4.0 or Jeliot2000 [1], and offers an opportunity for interactive experimentation.

THE DROPOUT PHENOMENON

During the fall semester, some students decided to drop out, some without even passing any of the courses. The high

[1] Veijo Meisalo, University of Helsinki, Department of Teacher Education, P.O. Box 9, FIN-00014, Helsinki, veijo.meisalo@helsinki.fi
[2] Erkki Sutinen, University of Joensuu, Department of Computer Science, P.O. Box 111, FIN-80100, Joensuu, erkki.sutinen@joensuu.fi
[3] Sirpa Torvinen, University of Joensuu, Department of Computer Science, P.O. Box 111, FIN-80100, Joensuu, sirpa.torvinen@joensuu.fi

school students studied for their matriculation examinations at the same time as they did the Virtual University course, and they may have been rather stressed during school exam periods, for example, even if they had not had the extra studies. Students had obvious difficulties scheduling their parallel studies [8].

It is well known that first-year Computer Science students have difficulty learning to program, so the number of dropouts in Table I is not surprising.

TABLE I
PERCENTAGE OF DROPOUTS IN OUR COURSES

Title of course	Scheduling	Percentage of dropouts
Preliminaries of Computer Architecture and Operating Systems (4.5 cps):		
Computers, operating systems, and networking (1.5 cps)	Fall 2000	6.1 %
Practical uses of computers, essential applications, and networks (1.5 cps)	Fall 2000	*)
Introduction to the ethics of computing (1.5 cps)	Fall 2001	0.0 %
Introduction to Computer Science (7.5 cps):		
Design of algorithms (1.5 cps)	Fall 2001	8.0 %
Hardware, structure of computers and operating system (3 cps)	Spring 2001	32.5 %
Research fields of Computer Science (3 cps)	Fall 2001	0. 0 %
Programming (10.5 cps):		
Programming, part 1 (3 cps)	Fall 2000	39.5 %
Programming, part 2 (4.5 cps)	Spring 2001	42.9 %
Laboratory project (3 cps)	Summer 2001	0.0 %

*) Data not available

Naturally, a possible explanation for the smaller percentage of dropouts in some courses might be that courses like "Introduction to ethics of computing" or "Practical uses of computers, essential applications, and networks" have a much easier content than the programming courses. Be that as it may, the difference between the percentage of dropouts from the programming courses and the other courses is conspicuous.

The other interesting point in Table I is that so many students dropped out during the fall of 2000 and in the beginning of the second programming course, Programming, part 2, Spring 2001, but very few later. According to Cornell & Martin [2], it is very common that as many as 30–50% of students drop out during virtual courses. They also found out that a student, who has passed at least some distance courses, will probably also pass others in the future. Our experiences are very similar to the above. When we look in more detail at the Summer or Fall 2001 courses, we find that every one who actively started a course also passed it.

Feedback and Course Evaluation

Since so many dropouts gave up during the first programming course, we focused on the problems with learning the programming basics and the most difficult topics therein. We wished to be able to pinpoint the main reasons for dropping out. We also analyzed the need for various support measures.

We use the Java programming language. There are several evaluations of the use of Java for introductory programming courses [4], [6] and [7]. We can agree with these reports regarding the difficulties using Java as the first programming language. On the other hand, we find many benefits in the use of Java when teaching high school students from 16 to 19 years old. These benefits include Java's general popularity as a programming language and its graphical features that seem to motivate students to learn programming.

One major aspect of our case study was to listen to the students and further develop our courses on the basis of the analyzed feedback. We analyzed the course from two viewpoints:

- **Learning outcome:** the examination papers and exercise portfolios of all the students were inspected.
- **Student attitudes:** students who dropped out, either during the first programming course or right after passing it, were enquired.

The questionnaire sent to the dropouts consisted of ten questions.

- **The first two questions focused on the learning environment:** usability of WebCT and the virtual course material.
- **The following six questions dealt with the study process:** which programming fields did the students regard as difficult; the benefit of online support given by supervisors (such as advice given by email, discussion forum, or feedback from exercises); was there a need for support measures like face-to-face sessions, videoconferencing, or a chat-room in WebCT-environment. The students were also asked when and why they dropped out.
- **The last two questions were open questions:** what would the students like to change in the course and were there any ideas on how to improve the quality of the material or course arrangements, etc.

A total of 25 students completed the original questionnaire and a more detailed follow-up questionnaire was sent to six volunteers by email.

We found the main reasons for dropping out were *lack of time*, *exercises too difficult*, and *failing the re-exam*. We also found the dropout students did not benefit from the support given by supervisors via Internet. Instead, students found the support of their peers very important. Many students found it difficult to study independently via Internet, particularly learning programming. They wished for traditional face-to-face lessons on the difficult topics of programming. Such topics include arrays, methods, applets, and animations. The exam results indicate that loops were more difficult for students than they themselves evaluated.

CHANGES MADE IN THE COURSE ARRANGEMENTS

On the basis of the analyzed feedback, we have modified our curriculum to better suit the students' potential. The major changes are as follows:

- **Re-scheduling within Programming, part 1:** In order to give more time for the difficult topics of programming, we re-arranged the weekly assignments giving more time for difficult topics. For example, there is now twice the time for learning arrays.
- **Re-scheduling of courses:** we looked at the scheduling of all the courses and modified it so as to avoid having several hard courses going on simultaneously.
- **Examples and optional exercises:** we added new examples and easy optional exercises to the Web material to support individual learning processes and make it easier to pass the weekly exercises. In this way students were supposed to feel successful at programming thus motivating and helping them with their learning process.
- **Individual feedback:** we gave more individual feedback on submitted exercises since some students clearly benefit from more detailed comments.
- **Visualization:** we offered more Jeliot animations particularly on difficult topics (like arrays). The main point was to offer different ways of learning to the students who had difficulty understanding the basics of programming. Our observation confirms the earlier findings [1] that average performers, in particular, found visualizations useful.
- **Encouraging the use of support measures offered in the learning environment:** we encouraged students to communicate with each other, for example, using team activities like discussion forums or email, and to ask for help with their learning difficulties.

In virtual courses students learn on their own in a responsible fashion. Students also need the support of tutor-teachers, supervisors, and their peers. We found the role of instructional intervention important in our virtual programming course. [8]

INFLUENCE OF CHANGES

Let us then look at how the above changes influenced the dropout phenomenon. When we look at the percentage of dropouts, we see the following figures in Table II. The data in Table II show the relative percentage of dropouts compared with the students who started the studies when seen right after the first programming course (Programming, part 1). In the first year, there were no relevant statistics for the students so the percentage 37% is calculated comparing the number of students who *did not participate in the exam* in December 2000 with those who started their studies in August 2000.

TABLE II
THE PERCENTAGE OF DROPOUTS

Year	Total	Old area	New area
2000	37.0%	37.0%	-
2001	26.0%	22.7 %	29.5%

In the school year 2001-2002, students have an opportunity to pass only part of the Virtual University course in the first year and the rest with a new group the following academic year. In the first school year 2000-2001 such an opportunity did not exist because it was a pilot project and we did not know whether this Virtual University course would be arranged again the following academic year. This may influence the apparent dropout percentages.

In the school year 2001-2002, students are obliged to inform us if they drop out. So the figure of 26% is calculated by comparing the *announced dropouts* with those who started studying in August 2001. But when we look at these announcements more closely, we see that there is a difference between those who did not participate in the exam and those who have announced they were dropping out. This means that we have not been informed of all the dropouts or that the students are going to complete the programming course later.

We wondered whether we could compare these two numbers, 37% and 26%, with each other and so we examined the number of dropouts from a different point of view. In Table III, we can see the percentage of dropouts calculated *comparing those who did not participate in the course exam in December with those who started the course in August.* The figures shown in Table III show that the relative number of dropouts is decreasing and the number of students who passed is increasing.

TABLE III

PARTICIPATION IN PROGRAMMING, PART 1

	2000	2001
Started course (Step 2)	76	146
Took part in exam (Step 4)	73.7 %	71.2 %
Passed course (Step 5)	60.5 %	64.4 %
Dropped out	39.5 %	35.6 %

Let us then compare the percentage of dropouts in the new area to the old area. If we look at Table II, we can see that the percentage of dropouts in the new area (29.5%) is clearly higher than in the old area (22.7%). These figures were calculated by comparing the number of dropouts to those who started the Course. We discuss later the factors that might influence the differences between the new and the old area.

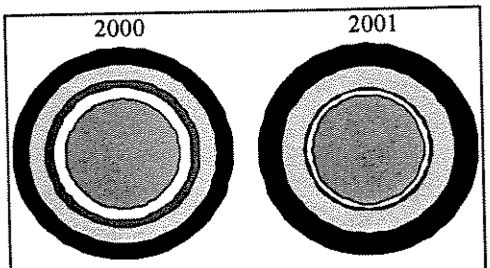
We were interested in finding out what might influence to the dropout phenomenon and found that students drop out step by step as follows:

- **Step 1:** Students who sign in the Virtual University course.
- **Step 2:** Students who start the Programming, part 1.
- **Step 3:** Students who submit the compulsory number of exercises.
- **Step 4:** Students who take part in the exam.
- **Step 5:** Students who pass the course.

We have illustrated this phenomenon in Figure 1. Our target is to reduce the differences between Steps 1 and 2, and between Steps 3 and 4. The first goal means that we hope that all those who become inspired by our virtual curriculum would take part in at least the first programming course even if they do not initially intend to take further courses. Our goal is to show students that they do not know what programming is if they do not even try to study it. We are especially interested in how to encourage female students to try programming. There are few females who sign in the Virtual University and in the school year 2000-2001 the percentage of female dropouts (79%) was clearly larger than that of males (59%). We hope that female students become more enthusiastic towards programming and computer science studies generally.

In Figure 1 we can see that for the second year (circles on the right) the differences between Steps 1 and 2 are slightly larger than for the first year (circles on the left). One explanation may be that students have scheduled their studies differently: Many students in 2001 decided to study all the other courses during the first academic year and left programming to the second year. So there might be some students who started in August 2001 and who will pass the course later.

When we look at Steps 3 and 4, we can see that in 2000 there were many students who did not take part in the exam although they had returned the compulsory number of exercises (Step 3 minus Step 4). The second year only two students who had the right to take part in the exam did not do so. We can see this in Figure 1: the dark gray circle (Step 3) on the right has almost disappeared behind the white (Step 4).



	2000	2001
● Step 1	89	184
● Step 2	85.4 %	79.3 %
● Step 3	81.6 %	72.6 %
○ Step 4	73.7 %	71.2 %
● Step 5	60.5 %	64.4 %

FIGURE 1

THE DROPOUT PHENOMENON ILLUSTRATED STEPWISE BY PERCENTAGES OF PARTICIPATING STUDENTS

On the other hand, when we look at the figures in Step 4 and Step 5, we can see that the most interesting difference between these two years may have an obvious reason. In 2000 there were as many as 13.2% of students who took part in exam but did not pass the course (Step 4 minus Step 5). In 2001 the analogous percentage is 6.8%. This means that in 2001 students who started the Programming, part 1, passed it better than in the first year. Let us then look the dropout numbers in the new and old areas shown in Table IV.

TABLE IV

THE DROPOUT PHENOMENON IN THE NEW AND OLD AREAS DESCRIBED BY PERCENTAGES OF PARTICIPATING STUDENTS

Steps	Old area	New area
Step 1	97	87
Step 2	81.4 %	77.0 %
Step 3	58.8 %	56.3 %
Step 4	58.8 %	54.0 %
Step 5	53.6 %	48.3 %

When we look at Table IV we can see that in total the participants percentages in the new area are somewhat lower than figures in the old area. This may explain a major part of the difference quoted above.

DIFFERENCES BETWEEN THE NEW AND OLD AREAS

It has been interesting to see that there really is a big difference between the dropout percentages of the old and the new areas, although the areas themselves appear rather similar. We found some factors that may have similarly influenced dropouts in both areas:

- **Common arrangements:** the course arrangements are exactly the same for students in both areas. Same timing, same courses, same information, etc.
- **Course material:** the web-material, visualizations, and exercises are the same for students in both areas.
- **Individual feedback:** there is no difference in feedback for students in the new or old areas. The feedback from exercises or advice concerning email questions or the discussion forum are the same.
- **Student experience in programming:** almost half of the dropouts (40%) in 2000 had no earlier experience programming and the rest of the dropouts had very little relevant experience. So there seems to be a clear negative correlation between earlier experience in programming and a disposition to drop out.

Because the percentage of dropouts differs in the new and old areas, there must be some other factors that influence these figures. We suggest the following:

- **Knowledge that courses are demanding:** the old area students have heard comments about courses from the first-year students and have some preconceptions about the courses. Thus, students in the old area are better prepared for the virtual studies.
- **Experience of tutor-teachers:** tutor-teachers in the old area can tell more about arrangements and courses. They can give more useful advice because they have had one year's experience of the same virtual courses.
- **Activities:** if a student has a problem, any help he or she may receive depends entirely on what he or she does about it. For example, using the discussion forum or email are very good ways of getting help. There were no significant differences between the old and new area students' behavior when discussing in the forum or by email.

We assume that the experience of tutor-teachers and knowledge about courses might explain the better figures in the old area shown in Table IV.

Furthermore we propose the following factor that has influence to the dropout phenomenon:

- **Attitudes:** both tutor-teachers' and students' attitudes are very important. If a tutor-teacher has a negative attitude, it has an obvious correlation to the number of dropouts. Similarly, if peers have positive (or negative) attitudes toward virtual studies, it is clearly seen in increasing (or reducing) motivation.

THE ROLE OF TUTOR-TEACHERS

We found the role of tutor-teachers to be quite important for success in virtual studies. We sent a questionnaire to 21 tutor-teachers: 10 of them in the new area and 11 in the old area. 18 returned the questionnaire.

We wanted to know how a tutor-teacher feels about the role of tutoring in virtual courses, whether a tutor-teacher has some need for support and whether he or she feels that students need more support than he or she can give. We also asked whether he or she has arranged regular meetings in the high school with the Virtual University students. We also asked whether there are any ideas for developing our course arrangements.

We assumed that the answers of the first-year tutor-teachers might differ from those of the second-year tutor-teachers. The differences were actually quite minimal, although some were found. Tutor-teachers in the new area are more qualified and some of them even had quite a negative attitude towards all the tasks given them. It was very clearly seen that tutoring was an uncomfortable task, and comments like "Giving advice is not tutors' duty" say much about the attitude of some tutors. The percentage of dropouts was correlated with negative attitude of tutor-teachers in our data. On the other hand, the reverse situation is also clearly seen: a large number of dropouts reduces the motivation of a tutor-teacher and some tutor-teachers feel really sorry about the dropouts.

The tutor-teachers mostly had correct preconceptions about tutoring, based on our tutor-guide and the information meeting. Only two tutors in the new area felt surprised by the tasks to be done. It was very clearly seen that students studied quite independently in both areas and mainly asked the tutor-teacher for help with problems in programming. Then it was not surprising that almost one quarter of the tutor-teachers hoped for an introduction course for Java as well.

The tutor-teachers' programming skills differ from "not at all" to expert. So the support given to the students by a tutor-teacher particularly during programming courses differs very much from school to school. In some schools, the tutor-teacher can only encourage students to continue, but in some others he or she can even help with difficult problems in programming.

It was also interesting to see that some of the tutor-teachers expected students to come and ask them for help. Others asked students how they were getting along with virtual studies and were there any problems. It seems that few tutor-teachers have organized weekly meetings with students. Most of them found it difficult because both the teacher and the students were too busy.

CONCLUSIONS

In our case study we have focused on developing an improving mechanism for our virtual programming courses. It consist of gathering data by multi-faceted ways like questionnaires to the students and tutor-teachers, analyzing the feedback submitted via the learning environment or email, observing activities of students in the learning environment by supervisors and analyzing data about students' behavior. Based on all this feedback and data, we managed to improve our virtual programming courses, and the whole Virtual University course as well so that the dropout percentages grew much lower. Stepwise treatment that we have built up helps us to pay our attention to certain groups of students and we look forward to developing an adaptive learning environment to our courses. We aim at improving not only the course materials but also the adaptive observation of students' learning skills in order to get help for students in their learning difficulties at an appropriate time.

While it is well known that first programming courses suffer from serious dropout problems, it is most important to focus research effort in this field. Our case study shows that there are many ways of cutting down the number of dropouts by improving practical arrangements during a virtual computer science course. It also shows that the number high-school dropouts may not be relatively higher than university-student dropouts on courses of a similar level. However, one must be very cautious in generalizing the results of our case study. The action research type of effort will continue, and we hope to show further results in due course.

REFERENCES

[1] Ben-Bassat Levy, R., Ben-Ari, M. & Uronen, P.A, "An Extended Experiment with Jeliot 2000", *Proceedings of the First Program Visualization Workshop*, Joensuu, Finland, 2001, pp. 131-140.

[2] Cornell, R. & Martin, B.L, "The Role of Motivation in Web-Based Instruction", In Khan, B. H. (Eds.), *Web-Based Instruction. Educational Technology Publications*, New Jersey, USA, 1997, pp. 93-106.

[3] Ellis, A., Lowder, J., Robinson, J., Hagan, D., Doube, W., Tucker, S., Sheard, J. & Carbone, A., "A collaborative strategy for developing shared Java teaching resources to support first year programming", *Proceedings of the 4th annual SIGCSE/SICCUE on Innovation and technology in Computer Science Education*, Krakow, Poland, 1999, pp. 84-87.

[4] Fleyry, A, E., "Programming in Java: Student-Constructed Rules", *Proceedings of the thirty-first SIGCSE technical symposium on Computer Science Education*, Austin, TX USA , 2000, pp. 197-201.

[5] Haataja, A., Suhonen, J. , Sutinen, E. & Torvinen, S., "High School Students Learning Computer Science over the Web", *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning (IMEJ)*, 3(2), October 2001.

[6] Kluit, P. G., Sint, M. & Wester, F., "Visual Programming with Java: Evaluation of an Introductory Programming Course", *Proceedings of the 6th annual conference on the teaching of computing / 3rd annual conference on integrating technology into computer science education on Changing the delivery of computer science education*, Dublin, Ireland, 1998, pp. 143-147.

[7] Roberts, E., "An Overview of MiniJava", *Proceedings of the Thirty Second SIGCSE Technical Symposium on Computer Science Education*, Charlotte, North Carolina, Feb 21-25, 2001, pp. 1-5.

[8] Sutinen, E. & Torvinen, S. "The Instructional Intervention in a Web-based programming course", *Proceedings of the first annual Finnish/Baltic Sea conference on Computer Science Education, Report A-2002-1*, October 19-21, Koli, Finland, 2001, pp. 53-60.

Paper IV:

Meisalo, V., Sutinen, E. & *Torvinen, S.*: "Choosing appropriate methods for evaluating and improving the learning process in distance programming courses", *The 33rd ASEE/IEEE Frontiers in Education Conference, FIE 2003*, November 4-8, 2003, Boulder, CO, USA. CD-ROM.

CHOOSING APPROPRIATE METHODS FOR EVALUATING AND IMPROVING THE LEARNING PROCESS IN DISTANCE PROGRAMMING COURSES

Veijo Meisalo[1], Erkki Sutinen[2], and Sirpa Torvinen[3]

Abstract *- The University of Joensuu started to offer virtual Computer Science studies to high school students in the surrounding rural area of Joensuu beginning in fall 2000. In this program high school students are studying first-year university level studies in CS over the web parallel with their regular high school studies. Almost half of the virtual CS studies offered focus on programming that has proved to be the most difficult part of this Curriculum. Most dropouts already quit during the first programming course and at the latest during the second one. We ascertained the difficulties in learning programming especially in a virtual learning setting. We used both quantitative (e.g. statistical analysis and analysis of closed questions in questionnaires) and qualitative methods (e.g. interviews, written products, and analysis of open questions in questionnaires) in order to evaluate the study process in the virtual programming courses during the years 2000-2002 and improve the course. It turned out that the use of qualitative methods is essential to explain the quantitative findings. The crucial issue is to distinguish between the needs of quantitative versus qualitative approaches, as well as to integrate them. The goal of the analysis is to improve our formative evaluation system and furthermore, to improve the whole distance learning scheme.*

Index Terms – distance learning, dropouts, evaluation methods, programming, virtual learning

INTRODUCTION

In fall 2000, the Department of Computer Science at the University of Joensuu began offering, within the framework of the Finnish Virtual University Project, a university level Introductory Computer Science (CS) Curriculum (22.5 European Credit Transfer System credits, cps) to high school students in the surrounding rural regions of Joensuu. In this distance education program, high school students study the first-year university-level CS courses over the web parallel with their regular high school studies. Students' age extends from 16 to 19 -years and we allow participation for all high school students who is interested in our distance CS curriculum without any pretest.

The Curriculum includes eight different courses that give students basic knowledge and skills of CS and computing. Students study independently over the Internet, but each high school has one *tutor teacher*, who helps students, especially at the beginning of the studies. The main responsibility for all the courses lies with a *supervisor*, who is a member of the staff of the Department of Computer Science at the University [7].

Students use both printed and web-materials, they complete weekly assignments, and their learning outcomes are evaluated through an exam. Almost all teaching is carried out via the Internet using the WebCT platform. Studies take altogether 1.5 years, but some students already drop out during the first fall semester. In our survey, we define a student as *dropped out* if he/she does not complete assignments for several weeks. Some students quit the study process without participating in any of the courses, but most dropouts quit during the programming courses in the first fall semester or the following spring semester. Dropout rates in university-level distance learning are, in general, definitely higher than those in conventional university studies. They vary depending on the adopted distance education system and the subject being studied [5], [13]. Dropout rates in Europe have varied from 20% to 30% while in Asian countries they have even been about 50% [13]. Cornell & Martin [5] found that a student, who has passed at least some distance courses, would probably also pass others in the future. Our experiences are similar to the above [10].

In order to improve our distance learning program, we have concentrated on our programming courses [10]. Our goal is to reduce the number of dropouts and motivate students to learn programming despite possible preconceptions of these being the most difficult courses. We noticed that there are no significant differences in the dropout rates of face-to-face programming courses and distance programming courses, although the students in our program are younger and the distance learning method itself is more demanding.

For example, Xenos *et al.* [13] have examined earlier student dropout rates and dropout causes among Informatics students at the Hellenic Open University. They used the following techniques for data collection: student registry, tutor's records, questionnaire-based surveys, and telephone interviews. The course at the Hellenic Open University was

[1] Veijo Meisalo, University of Helsinki, Department of Teacher Education, P.O. Box 9, FIN-00014, Helsinki, veijo.meisalo@helsinki.fi
[2] Erkki Sutinen, University of Joensuu, Department of Computer Science, P.O. Box 111, FIN-80100, Joensuu, erkki.sutinen@joensuu.fi
[3] Sirpa Torvinen, University of Joensuu, Department of Computer Science, P.O. Box 111, FIN-80100 , Joensuu, sirpa.torvinen@joensuu.fi

designed for adult students. The method used by Xenos *et al.* [13] to review the dropout phenomenon is quite similar to ours [10]. Our findings concerning the dropout phenomenon are the same as those of Xenos *et al.* [13]: the main reason for dropping out in our distance learning project has been lack of time. Students have been unable to estimate the time required for university-level studies and perceived the computer course to be difficult.

Our survey focus is not only to find the dropout rates and reasons course-wise, but also to identify the key issues as to why dropouts occur and to improve our course settings. How could we help our students succeed in their virtual study processes? Which topics are the most difficult and what kind of help do the students need during the study process?

Anastasiades [2] presents a different kind of evaluation methodology and results concerning distance learning in elementary schools. This paper presents evaluation methodology for a pilot distance learning program, called ODYSSEUS, between two elementary schools with the main target being to familiarize students with a step-by-step hybrid-learning environment. This approach consists of 11 steps of the ODYSSEUS evaluation methodology including, for example, evaluation target, object of evaluation, evaluation criteria, and evaluation tools. Evaluation methodology focuses on three levels: the evaluation of technological and human resources, learning effectiveness, and the social implications of the suggested distance learning model [2].

Selwyn [12] studied students' attitudes towards computers in education for 16 to 19 years old students. This study collected data on students' use of, and their attitudes towards, computers via a self-administered questionnaire. Selwyn noticed that he might argue that research into attitudes towards educational computing has concentrated too much on quantitative methods and future research should strive to use qualitative findings to illuminate those features not seen in quantitative findings.

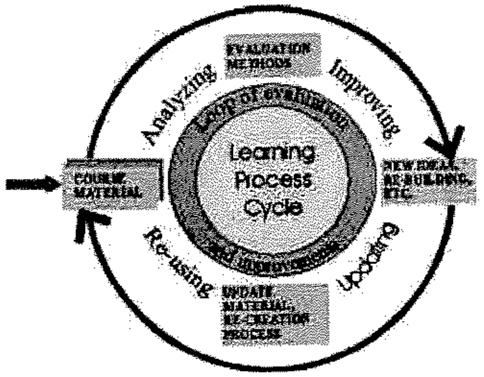


FIGURE 1
CYCLE OF EVALUATION

Figure 1 shows the principle of the evaluation process in our CS courses. In order to get a more detailed picture of the difficulties in our virtual programming courses, we used the following techniques for data collection: interviews, student registry, and a questionnaire-based survey. We conducted interviews in face-to-face situations and recorded them on audiotape. Additionally, we analyzed the feedback given by students, as well as the statistics and the log of actions. Although there are many differences between our surveys, like the students' age and the breadth of the courses, our goals are similar to those of Xenos *et al.* [13]. We both investigate the factors and causes related to dropout rates. We both also hope that our research will be useful to those involved in distance learning processes for identifying dropout-prone students.

In the future, we shall concentrate on intervening especially in the dropout-prone student groups, which will hopefully minimize the number of dropouts. In addition our focus is to improve our materials and arrangements at the same time as we look for indicators connected with dropouts. Earlier surveys have emphasized evaluating the system and methods, and reporting the results of these evaluations. In this paper, we introduce the cycle of evaluation for a continuous improvement process of the learning environment, based on functional evaluation methodology (Figure 1).

QUANTITATIVE AND QUALITATIVE METHODS AND TRIANGULATION

It is possible to use both quantitative and qualitative methods in evaluation studies. Traditionally, Computer Science Education (CSE) has trusted in quantitative methods, but there are many features in CSE studies where even qualitative approaches have proved to be valuable. For example, Medley [9] noticed that most CSE researchers would rely on software for only the quantitative methods even if both quantitative and qualitative methods were needed. Qualitative research methods are, indeed, well-developed tools for educational research [6]. Actually, there is already a long tradition of combining qualitative and quantitative methods even for conducting research on distance education (e.g. [4]). Such an approach is called *triangulation* in educational research.

The purposes for qualitative research projects are not the same as those for quantitative. With qualitative methods we aim to get a better understanding of the study process. Medley [9] defines the use of quantitative and qualitative methods as follows: *quantitative techniques* are used for tasks like counting the number of students who got correct answers in an exam or who handed in a completed programming assignments and using these results to generate statistics. In addition to these quantitative findings, it is fruitful to use *qualitative methods* to get answers for trends and tendencies. Table 1 shows the differences between the qualitative and quantitative approach by Merriam [11].

TABLE 1
DIFFERENCES BETWEEN QUANTITATIVE AND QUALITATIVE RESEARCH

Attribute	Quantitative research	Qualitative research
Starting point	Definite hypothesis that must be proved or disproved	Attempts to discover research ideas, sometimes for the purpose of generating qualitative projects
Scale of research field	Very small, specific facts about clearly defined situations	Entire environment or context and try to finding patterns within it
Methods	Specific methods for choosing samples to generate data	Chooses samples from individuals that are the most likely to contribute to the understanding of the problem
Tools for data collection	Instruments, tests, objective means as data	Interviews, essays, observation as data

THE NEED FOR EVALUATION

The virtual CS studies take altogether 1.5 years. Thus, we currently have the third group of high school students studying in this program. During the first fall semester, we offer two courses: the Basics of Computer Science, and Programming I. The latter one in particular has proved difficult for students, while most dropouts quit during the first programming course or right after it [10]. This aroused our interest in the dropout phenomenon. Furthermore, we have noticed that students who pass Programming I in the first fall semester and Programming II in the following spring semester will very likely graduate after 1.5 years. Thus, an intervention during the first programming course or at the beginning of the second one will very likely minimize the dropout rates.

The dropout phenomenon piqued our curiosity. There were several unanswered questions:

- Why do the students drop out?
- When do the students drop out?
- Why is learning programming so difficult?
- Were there any other difficulties (e.g. web-based learning itself) in the virtual learning process?
- Which are the most difficult topics in the web-based programming courses?
- Is there a need for some other kind of support during web-based courses?
- Is there any need for optional materials?
- What learning techniques have students used?
- Did the students get help when needed?
- How should the course materials or the course arrangements be improved?

In order to find answers to these questions, we decided to evaluate our distance learning project focusing especially on the course Programming I. In our evaluation system, improving courses has been the most important goal [10]. Earlier Almstrum *et al.* [1] have claimed that there is only one reason to evaluate: in order to learn. We can easily agree that; there is no point in evaluating if we are not ready to look at our work truthfully and accept all the results from the evaluation process. No matter whether the results are good or bad. After analysis and improvements we are more satisfied with our work and like Almsturm *et al.* noticed: the ultimate beneficiaries of our learning are our students [1]. We may also note, that the evaluation process should not only give us information, but also our students, as well as the administrators, providers of learning materials, etc.

METHODS USED

The CSE research area is rather new and the research in it has become increasingly important in recent years. Problems in the distance learning of CS can be due to the students, teachers and supervisors, the institution itself, and/or the medium used to deliver the subject matter [8]. How can we generalize results, for example, in the psychology of programming while students are different and individual differences account for most of the observed variance? Anyway, the complexity of the individual is no reason to abandon evaluation [1].

TABLE 2
QUANTITATIVE AND QUALITATIVE METHODS USED

Type of method	Methods used
Quantitative	Statistical analysis of submitted exercises Exam scores Analysis of closed questions in questionnaires
Qualitative	Analysis of open questions in questionnaires Interviews, observations, tape recordings, field notes Analysis of the feedback given by students and tutor teachers Written products: quality of submitted exercises End products: results of examinations

Table 2 presents the quantitative and qualitative methods used in our survey. In Table 3 we have illustrated the number of students per method of collected data. We have listed the methods used and the number of students for each year 2000-2002. There are two sets of numbers in the third column of Table 3 meaning how many of students participated in our survey, e.g. 25/55 means 25 students out of 55.

TABLE 3
NUMBER OF STUDENTS PER METHOD IN THE COLLECTED DATA

Method	Year	Number of students
Questionnaires	2000	25 / 55
	2001	42 / 76
	2002	(ongoing)
Follow-up questions by e-mail	2000	5 / 5
	2001	-
	2002	-
Interviews	2000	-
	2001	11 / 11
	2002	(ongoing)
Feedback analysis	2000	29 / 56
	2001	52 / 52
	2002	18 / 18
Analysis of submitted exercises	2000	76 / 76
	2001	146 / 146
	2002	109 (ongoing)
Analysis of exams	2000	56 / 56
	2001	104 / 104
	2002	67 / 67
Web-based questionnaires at the beginning of the studies	2000	-
	2001	-
	2002	54 / 109

We started with quantitative methods like the analysis of log of actions and analysis of submitted exercises. We carried out this analysis throughout the course. In the beginning of the course the student filled in the web-based questionnaire, in the middle of the course we got feedback on feelings, motivations, and attitudes towards course materials and arrangements during the study process, and then we got feedback on feelings again at the end of the course before the exam. An exam was arranged immediately after the course ended. Exams included five questions: in the first one students had to define five essential concepts, the second one was an essay and the remaining three were programming tasks. After the course we sent questionnaires to the students: in the first two years to dropouts only and in the third year (2002) to all the students who participated in Programming I.

The questionnaire included both closed and open questions. After analyzing the questionnaires we interviewed some students to get a better understanding of, and learn more about, the study process and the difficulties in distance learning programming. Figure 2 shows how the methods distributed over the course period.

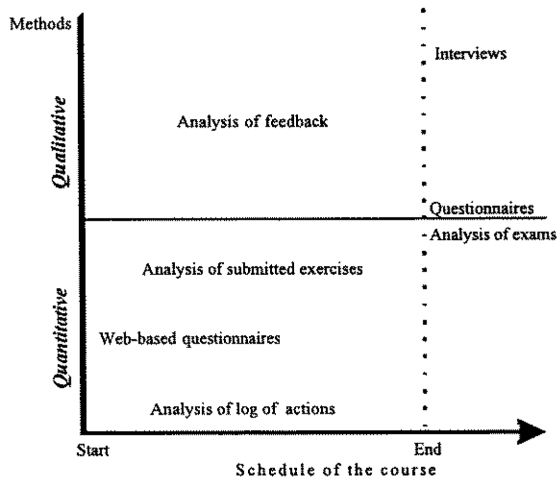


FIGURE 2
THE COLLECTION METHODS DESCRIBED BY DIMENSIONS OF QUALITATIVE VERSUS QUANTITATIVE, AND THE SCHEDULE OF THE COURSE

EXPERIENCES

The cycle of evaluation applied to a particular course implementation is an open process where both the course and its evaluation influence each other and keep changing over time. In our case, we started in the first year (2000) to collect data by questionnaires for the dropout students. A major idea of the case study was to listen to the students and develop the courses further based on the analyzed feedback in a formative way.

The questionnaire included ten questions [10] dealing with the usability of the WebCT platform and the course materials on it, the fields of programming that students considered as difficult, the benefit of on-line support given by supervisors from the University (like advice given by e-mail, discussion forum, or feedback of exercises), need for support measures like face-to-face lessons, videoconferencing, or chat in the WebCT environment. The students were also asked the time and reason for dropping out. There were also two open questions on the changes required in the course and ideas for improving the quality of the materials or course arrangements.

The data collected by questionnaires posed new questions. For example, when a student said the main reason for dropping out was "lack of time", we wanted to know why. We tried to interview students by telephone but encountered problems. The students were rather young, 15-19 years old, and although we had agreed what time to call, they were unavailable. Finally, we decided to send more detailed follow-up questions by email to five students who then all answered. Thus, the further we proceeded in the analysis, the deeper and more detailed questions we encountered. These were answered by a more limited student group, giving us novel insights to our original observations.

We got important information on the causes of dropping out and students' difficulties in programming through the use of questionnaires. We noticed that most dropouts had only very limited skills, if any, in programming prior to the course. Therefore, we decided to improve our materials for the following year and focused, especially at the beginning of the studies, by giving more detailed instructions to the tutor teachers and students during the information meeting.

The number of exercises submitted had a significant negative correlation with the number of dropouts, as well as a positive one with the results in the exam. It is clearly seen that if a student submits assignments weekly (at least two or three exercises out of five) he/she will succeed in the examination and pass the course.

For the following years, we also improved our questionnaires based on our first year's experience. However, the main structure of the questionnaires remained the same. Interesting new questions have appeared throughout the project requiring new techniques for collecting and analyzing data. For example, in the beginning it was interesting to identify the lack of time as the main reason for dropping out. This observation led us to analyze the lack of time more carefully. Likewise, many other answers given by students prompted us to ask more details. For the group that began in 2002, we have developed a new technique (Web-based questionnaire at the beginning of studies). During 2001, we also interviewed students face-to-face instead of using email questions and got much better information on the motivation and attitudes towards distance learning and programming.

BENEFITS FROM THE TRIANGULAR EVALUATION APPROACH

Our study process has helped students in their studies in the virtual programming course. We wanted to ascertain their difficulties in learning programming using questionnaires. In Figure 3, we can see which topics students found difficult when using Java as the programming language. We have used letters A to G to show the programming topics. The symbols are as follows:

- A: Variables, symbols, and assignments
- B: Input and output
- C: Loops
- D: Conditional statements
- E: Arrays
- F: Methods
- G: Applets and graphics

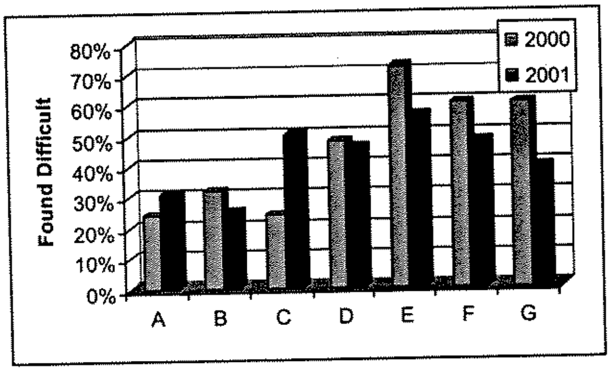


FIGURE 3
DIFFICULT TOPICS IN THE PROGRAMMING I COURSE

The data shown in Figure 3 were collected by quantitative methods using questionnaires. The question format was: "Estimate the difficulty of the following topics using a scale of 1 to 5 where 1=very easy, 2=quite easy, 3=neutral, 4=quite difficult, 5=very difficult". Students had a list of topics included in the Programming I course. Figure 3 shows very clearly that the students of 2000 did not find loops (symbol C) particularly difficult (alternatives 4 or 5). On the other hand, in the 2001 group almost half the students found loops difficult to learn. This difference could be explained by different skills in the student groups. However already in 2000 the exam answers indicated that loops were more difficult for students than they estimated in the questionnaire [10]. Thus in order to get a better picture of this we need more techniques, not just quantitative analysis of closed questions. Both qualitative and quantitative methods are needed, and the only problems seem to be related to finding a proper combination of useful methods in each case. Without a more detailed analysis of the results of the exams and interviews or follow-up questions by email, we might get a totally wrong picture of students' skills or difficulties.

Let us now look at the main contribution of each technique mentioned in Table 3 and Figure 2 used in our analysis of the dropout phenomenon:

- **The pretest questionnaire at the beginning of the studies:** gives information about skills before studies. Based on this information we can intensify the support measures or general advice via the discussion forum when there are many students with no programming skills.
- **Questionnaires** provide basic data on dropouts and their skills, indicate the most difficult programming topics, give information about the usefulness of materials, the learning environment, support methods, and the need for improvements. Based on this data, we can concentrate on improving the most difficult parts in

our courses and target various support tools or new materials especially for difficult topics.

- **Interviews** provide more detailed information about difficult programming topics (*what is difficult, why it is difficult, etc.)*, study processes, and learning techniques, as well as attitudes and motivation. Based on this information, we can inform new students and teach them techniques that have proved to be fruitful. We get valuable information about motivation in web-based courses.
- **Statistical analysis of submitted exercises and exams** provide information about the correlation between submitted exercises and learning results, and on the learning results in general. Based on this information, we can compare the data collected and the results of exams for a more detailed analysis of study processes and students' profiles.
- **Analysis of feedback** provides much information about satisfactory or unsatisfactory course materials, learning environments, and other arrangements. Based on this information, we have been able to quickly improve our courses and arrangements, updating materials or making changes in scheduling when needed.
- **Statistics from the log of actions** provides information about activities in the learning environment. This method has not yet been very fruitful for us, because all our materials are available for downloading and this is supposed to minimize the actions in the learning environment. Anyway, WebCT track lists provide information about the first and last logons and logouts, so some kind of activity is seen in these lists.

CONCLUSIONS

The aim of the research conducted in the Virtual CS Curriculum has been to combine continuing improvements and an evaluation of the course. The main reason to drop out has been the lack of time. Students were clearly unable to estimate the time required for university-level studies and thought the course was difficult. High school context seems not to have a significant role while the dropout percentages for our traditional programming courses on our campus are equally high. The focus of our study has not been in analyzing the dropout numbers only but at the same time in improving our distance CS program. We find our formative evaluation model (Figure 1) a practical and fruitful way to quickly improve course materials and arrangements. We recommend it for everyone who arranges web-based courses. Quantitative methods such as analysis of log of actions or analysis of submitted exercises and closed questions in questionnaires give us the basic data about the dropout phenomenon. We complement the quantitative findings with qualitative methods. We can recommend this bottom-up scheme starting with the basic quantitative collection of data and completing with qualitative methods of data collection.

Our methods are currently relatively fruitful and comprehensive compared with earlier studies in the same area, but we find for example Chin's empirical evaluation of user models and user-adapted systems [3] very interesting. In future, our goal is to find possible dropouts beforehand and try to improve our materials more and more towards adaptive materials, which take into account students' individual needs. Our further intentions also include applying these versatile evaluation methods towards more creative approaches in learning programming.

REFERENCES

[1] Almstrum, V.L., Dale, N., Berglund, A., Granger, M., Little, J.C., Miller, D. M., Petre, M., Schragger, P., Springsteel, F.: "Evaluation: turning technology from toy to tool. *Report of the Working Group on Evaluation.", Proceedings of he 1st Conference on Integrating Technology Into Computer Science Education*, 1996, Barcelona, Spain, ACM Press, 201-217.

[2] Anastasiades, P.S., "Distance learning in elementary schools in Cyprus: the evaluation methodology and results", *Computers and Education*, 40, 2003, 17-40.

[3] Chin, D, N., "Empirical Evaluation of User Models and User-Adapted Systems", *User Modeling and User-Adapted Interaction*, 11, 2001, 181-194.

[4] Coldeway, D. "Methodological Issues in Distance Educational Research", *American Journal of Distance Education* 2(3), 1988, 45-54.

[5] Cornell, R., Martin, B.L, "The Role of Motivation in Web-Based Instruction", In Khan, B. H. (Eds.), *Web-Based Instruction. Educational Technology Publications*, New Jersey, USA, 1997, 93-106.

[6] Greene, J. "Qualitative Program Evaluation. In N. Denzin & Y. Lincoln (Eds.), Handbook of Qualitative Research. London, Sage, 530-541.

[7] Haataja, A., Suhonen, J. , Sutinen, E., Torvinen, S., "High School Students Learning Computer Science over the Web", *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning (IMEJ)*, 3(2), October 2001.

[8] Jones, D.: "Computing by Distance Education: problems and solutions", *Proceedings of the Conference on Integrating Technology Into Computer Science Education*, ACM SIGCSE Bulletin, Vol (28) 1996, 139-146.

[9] Medley, M.D., "Using Qualitative Research Software for CS Education Research", *Proceeding of the 6th annual Conference on Innovation and Technology in Computer Science Education"*, Vol 33(3), 2001, 141-144.

[10] Meisalo, V., Sutinen, E., Torvinen, S., "How to improve a virtual programming course?", *Proceedings of the 32nd ASEE/IEEE Frontiers in Education Conference (FIE2002)*, Boston, USA, 2002.

[11] Merriam, S: *Qualitative Research and Case Study Applications in Education: Revised and Expanded from Case Study Research in Education*, Jossey-Bass, 1997

[12] Selwyn, N., "Students' attitudes towards computers in sixteen to nineteen education", *Education and Information Technologies*, 4(2), 1999, 129-141.

[13] Xenos, M., Pierrakeas, C., Panagiotis, P.: "A survey on student dropout rates and dropout causes concerning the students in the Course of Informatics of the Hellenic Open University", *Computers and Education*, Vol (39) 2002, 361-377.

KYSELYLOMAKE

Taustatietoja:

Vastaajan nimi: ______________________________
Oppilaitoksen nimi: ______________________________

Olen aloittanut opinnot nykyisessä oppilaitoksessa

☐ v. 2000 ☐ v. 2001 ☐ v. 2002 ☐ 1999 ☐ vuonna __________

Minulla on / oli lukiossa

	Kyllä	*Ei*
Pitkä matematiikka	☐	☐
Laaja fysiikka	☐	☐
Kemia	☐	☐

Kysymykset

Arvioi seuraavia väittämiä käyttäen asteikkoa 1-5 (1 = täysin eri mieltä, 5 = täysin samaa mieltä).

1. Ohjelmointi 1- kurssimateriaali ja sen käytettävyys.

Arvioitava osa-alue	*1*	*2*	*3*	*4*	*5*
WebCT:n teoriasivut toimivat hyvin oppikirjan tukena	☐	☐	☐	☐	☐
Teoriasivuilta löytyi tarvittava tieto helposti	☐	☐	☐	☐	☐
Teoriasivut olivat rakenteeltaan johdonmukaiset ja selkeät	☐	☐	☐	☐	☐
Harjoitustehtäviä oli riittävästi asian oppimisen kannalta	☐	☐	☐	☐	☐
Harjoitustehtävät kattoivat hyvin kurssin eri osa-alueet	☐	☐	☐	☐	☐
Teoriasivujen esimerkit helpottivat asian omaksumista	☐	☐	☐	☐	☐
Teoriasivujen esimerkkien ja harjaantumistehtävien läpikäyminen helpottivat varsinaisten tehtävien tekemistä	☐	☐	☐	☐	☐

2. Visualisointivälineiden käyttö
Ohjelmointi, osa 1 –kurssin aikana hyödynnettiin kolmea visualisointitapaa: Flash-animaatioita, appletteja ja Jeliot 2000 –ohjelmaa.

a) **Arvioi, auttoivatko teoriasivustoilla olevat Flash-animaatiot ja appletit sinua ohjelmoinnin oppimisessa? Kerro myös miksi auttoi tai miksi ei auttanut.**

____
____
____

b) **Arvioi, auttoiko Jeliot 2000 ohjelma sinua ymmärtämään paremmin ohjelmakoodia ja/tai tallennusrakenteita (esimerkiksi taulukkoa)? Kerro myös miksi auttoi tai miksi ei auttanut.**

____
____
____

Kysely liittyy Sirpa Torvisen Joensuun yliopistolle tekemään jatkotutkimukseen.
Yhteystiedot: sirpa.torvinen@joensuu.fi

KYSELYLOMAKE

c) Kerro, kuinka koit Jeliot 2000 ohjelman asentamisen ja käyttämisen yleisesti. Onko sinulla parannusehdotuksia Jeliot 2000 ohjelman ominaisuuksien suhteen?

____

____

____

____

3. Oppimisympäristön tekninen hallinta.
Arvioi seuraavia väittämiä käyttäen asteikkoa 1-5 (1 = täysin eri mieltä, 5 = täysin samaa mieltä).

Arvioitava osa-alue	*1*	*2*	*3*	*4*	*5*
WebCT –oppimisympäristöä oli helppo käyttää	☐	☐	☐	☐	☐
WebCT -oppimisympäristö oli rakenteeltaan selkeä	☐	☐	☐	☐	☐
Harjoitustehtävien palauttaminen sujui helposti	☐	☐	☐	☐	☐
Malliratkaisut olivat helposti saatavilla	☐	☐	☐	☐	☐
Java-kääntäjän ja -tulkin asentaminen sujui ongelmitta	☐	☐	☐	☐	☐
Java-kääntäjää ja –tulkkia oli helppo käyttää	☐	☐	☐	☐	☐

Kerro myös, mitä toiveita tai parannusehdotuksia sinulla on oppimisympäristön kehittämiseksi:

____

____

____

4. **Ohjauksen saaminen ongelmatilanteissa.**
Arvioi seuraavia väittämiä käyttäen asteikkoa 1-5 (1 = täysin eri mieltä, 5 = täysin samaa mieltä).

Arvioitava osa-alue	*1*	*2*	*3*	*4*	*5*
Yliopistolta saamani ohjaus auttoi harjoitustehtävien ratkaisemisessa	☐	☐	☐	☐	☐
Harjoitustehtävistä saamani palaute auttoi ymmärtämään vaikeita kohtia	☐	☐	☐	☐	☐
Kysyin neuvoa kurssikavereilta WebCT:n keskustelufoorumin kautta ja sain sieltä apua	☐	☐	☐	☐	☐
Seurasin WebCT:n keskustelupalstaa ja löysin toisten oppilaiden tekemistä kysymyksistä ja niihin tulleista vastauksista ratkaisun myös itseäni askarruttaviin kysymyksiin	☐	☐	☐	☐	☐
Keskustelin omalla koulullani olevien kurssikavereiden kanssa ja sain heiltä apua ongelmallisissa tehtävissä tai tilanteissa	☐	☐	☐	☐	☐
Malliratkaisut auttoivat löytämään omassa ratkaisussani olleita puutteita tai virheitä	☐	☐	☐	☐	☐
Käytännön ongelmissa sain apua ongelman ratkaisemiseen oppilaitokseni tutoropettajalta	☐	☐	☐	☐	☐

Yhteystiedot: sirpa.torvinen@joensuu.fi

KYSELYLOMAKE

Oppilaitokseni tutoropettaja oli tavoitettavissa ja osasi neuvoa opiskeluun liittyvissä ongelmatilanteissa eteenpäin.	☐	☐	☐	☐	☐

Sain apua ohjelmoinnin opiskelussa kohtaamiini ongelmiin muualta, mistä?

____

____

____

5. Ohjauksen kehittäminen

Onko sinulla parannusehdotuksia virtuaaliohjauksen suhteen? Kerro, kuinka ohjausta tulisi muuttaa ja mitä asioita virtuaalikurssien ohjauksessa ja / tai ohjeistuksessa tulisi huomioida?

____

____

____

____

____

____

____

6. Opiskelun tukimateriaalin käyttäminen

	Kyllä	*Ei*
Osallistuin Virtuaaliapprobaturin aloitusinfotilaisuuteen elokuussa 2002	☐	☐
Tutustuin Opiskelijan itseopiskeluoppaaseen	☐	☐
Tutustuin kunkin kurssin alkaessa kurssin esittelysivuun WebCT:ssä	☐	☐
Tutustuin Opiskelijoiden tietopankin ohjeisiin	☐	☐
Tutustuin WebCT:n käyttöohjeeseen	☐	☐

Onko sinulla toiveita tai parannusehdotuksia virtuaaliopintojen oppaiden suhteen? Kerro, mitä:

____

____

____

KYSELYLOMAKE

7. **Aikaisempi ohjelmointikokemus.**
Vastaa seuraaviin kysymyksiin sen mukaan mikä tilanne oli **ennen** Virtuaaliapprobatur-opintojen alkamista.

a) Minulla on kokemusta seuraavista ohjelmointikielistä:

____

____

____

____

b) Kerro, minkä tyyppistä ohjelmointia olet harrastanut edellä mainitsemillasi ohjelmointikielillä?

____

____

____

____

c) Kerro, kuinka olet opiskellut ja oppinut kyseisiä ohjelmointikieliä?

____

____

____

____

8. **Ohjelmoinnin osa-alueiden vaikeus**
a) Arvioi seuraavien ohjelmoinnin osa-alueiden vaikeutta asteikolla 1-5 (1= helppo, 2= melko helppo, 3= en osaa sanoa, 4 = melko vaikea, 5= vaikea).

Arvioitava osa-alue	***1***	***2***	***3***	***4***	***5***
Muuttujat ja tunnukset	☐	☐	☐	☐	☐
Tietotyypit	☐	☐	☐	☐	☐
Sijoitusoperaatiot	☐	☐	☐	☐	☐
Syöttö- ja tulostuslauseiden käyttö	☐	☐	☐	☐	☐
Vertailuoperaatiot ja boolean lauseet	☐	☐	☐	☐	☐
Valintalauseet (if, if-else, if-lohko, if – else-lohko)	☐	☐	☐	☐	☐
Toistolauseet (for, while ja do-while)	☐	☐	☐	☐	☐
Loogiset operandit (&&, ||, ^, !)	☐	☐	☐	☐	☐
Taulukot	☐	☐	☐	☐	☐
Sisäkkäiset toistolauseet, vaihtolajittelu	☐	☐	☐	☐	☐
Satunnaisluvut	☐	☐	☐	☐	☐
Metodit	☐	☐	☐	☐	☐
Sovelmien (applettien) tekeminen	☐	☐	☐	☐	☐
Grafiikkakäskyt	☐	☐	☐	☐	☐
Näppäinohjelmointi	☐	☐	☐	☐	☐

b) Jos vastasit a-kohdassa jonkin käsitteen kohdalla valinnalla 4 tai 5, niin kerro miksi koet asiat vaikeaksi tai mikä ko. asian opiskelussa oli hankalaa?

____

____

____

____

KYSELYLOMAKE

9. Opetuksen tukimenetelmät
a) Kaipaisitko Web-pohjaisen opetuksen tueksi jotakin seuraavista monimuoto-opetusmenetelmistä?

	kyllä	ei
Luento yliopistolla Savonlinnassa tai Joensuussa	☐	☐
Videoluento omasta oppilaitoksesta käsin	☐	☐
Ohjattu keskustelu (esim. Netmeeting tai WebCT)	☐	☐

Muuta, mitä?

____

____

____

b) Mikäli vastasit a-kohdassa johonkin kohtaan kyllä, niin kerro mihin ohjelmoinnin osa-alueeseen erityisesti kaipaisit tukea ja miksi?

____

____

____

____

____

____

10. Jos olet keskeyttänyt approbatur-opintosi, niin merkitse missä vaiheessa (laita rasti keskeytysajankohtaan)

Syyskuu	Lokakuu	Marraskuu	Joulukuu	Tammikuu	Helmikuu	Maaliskuu
Virtuaaliopin-tojen aloitus Ohjelmointi, osa1 -kurssin aloitus	Tietotekniikan perusteet -kurssin tentti		Ohjelmointi, osa 1 -kurssin tentti	Ohjelmointi, osa 2 -kurssin aloitus Ohjelmointi, osa 1 -kurssin uusintatentti	Tutkimusaloja –kurssin aloitus	
☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐

11. Mikä oli pääsyy keskeyttämiseesi (voit myös numeroida useammankin syyn käyttäen järjestystä 1= tärkein syy, 2 = toiseksi tärkein jne)?

☐osallistuin yo-kirjoituksiin
☐aion jatkaa opintoja ensi lukuvuonna
☐teoria tuntui liian vaikealta
☐tehtävät tuntuivat liian vaikeilta
☐en läpäissyt Ohjelmointi, osa 1-kurssin varsinaista tenttiä
☐en läpäissyt Ohjelmointi, osa 1-kurssin uusintatenttiä
☐tuli muutoksia opiskelutilanteeseeni
☐kurssisisältö ei vastannut ennakko-odotuksiani
☐kurssi ei kiinnostanut
☐muu syy, mikä?

____

☐aikapula,
kerro myös mistä arvelet aikapulan johtuvan?

____

____

KYSELYLOMAKE

12. Opiskeluun käytetty aika:

a) Arvioi kuinka paljon käytit aikaa viikossa ohjelmoinnin opiskeluun. Käytä asteikkoa 1 = alle tunti, 2 = 1-2 tuntia, 3 = 3-4 tuntia, 4 = 5-6 tuntia, 5 = enemmän kuin 6 tuntia viikossa.

Arvioitava osa-alue	*1*	*2*	*3*	*4*	*5*
Oppikirjasta teorian opiskeluun	☐	☐	☐	☐	☐
Web-materiaalista teorian opiskeluun	☐	☐	☐	☐	☐
Web-materiaalista harjaantumistehtävien tekemiseen	☐	☐	☐	☐	☐
Harjoitustehtävien (demotehtävien) tekemiseen	☐	☐	☐	☐	☐
Kurssin tehtävien lisäksi omana harrasteena ohjelmointiin	☐	☐	☐	☐	☐
Lukion ohjelmointikurssille osallistumiseen samanaikaisesti	☐	☐	☐	☐	☐
Muuhun, mihin?					
______________________	☐	☐	☐	☐	☐
______________________	☐	☐	☐	☐	☐

b) Arvioi lisäksi käytitkö enemmän aikaa vaikeana kokemiesi asioiden kohdalla kuin muutoin kurssilla? Jos, niin mikä tai mitkä aihealueet vaativat enemmän aikaa?

13. Tarvitsitko hakea lisätietoja WebCT:ssä olleen verkkomateriaalin ja kurssilla käytettävän oppikirjan lisäksi esimerkiksi Internetistä tai muista Java-ohjelmointikirjoista kurssin aikana? Kerro mihin ohjelmoinnin osa-alueen opiskeluun tarvitsit lisätietoja ja mistä sitä sait.

14. Mitä muuttaisit ohjelmointikurssista? Ottaisitko jotakin pois tai haluaisitko jotakin lisää?

Yhteystiedot: sirpa.torvinen@joensuu.fi

KYSELYLOMAKE

15. Onko sinulla muita toiveita kurssimateriaalin tai käytännön teknisen toteutuksen parantamiseksi?

16. Kerro mitä koet oppineesi tai kuinka koet hyötyneesi virtuaaliapprobatur-kursseista.

17. Muita ehdotuksia tai kommentteja Virtuaaliapprobaturin järjestäjille?

Kiitos Sinulle vastauksestasi!
Jokainen annettu parantamisidea auttaa meitä työssämme kurssien kehittämisessä.

Kysely liittyy Sirpa Torvisen Joensuun yliopistolle tekemään jatkotutkimukseen.
Yhteystiedot: sirpa.torvinen@joensuu.fi

QUESTIONNAIRE

Baclground:

Your name: ______________________
Name of the educational institute: ______________________

I have started the studies at the institute mentioned above in year

☐ 2000 ☐ 2001 ☐ 2002 ☐ 1999 ☐ in year ________

I study / studied in the high scool

	Yes	*No*
Extensive syllabus in mathematics	☐	☐
Extensive syllabus in physics	☐	☐
Chemistry	☐	☐

Questions:

Please estimate using scale 1 to 5 (1 = fully not-agreed, 2 = somewhat not-agreed , 3 = neutral, 4 = somewhat agreed, 5 = fully agreed).

1. Programming 1 –course material and it's usability.

Field to be estimated	*1*	*2*	*3*	*4*	*5*
Theory in WebCT supports the book well	☐	☐	☐	☐	☐
Information in Web-site was easy to find	☐	☐	☐	☐	☐
Web-sites were well-structured	☐	☐	☐	☐	☐
Adequate number of excercises	☐	☐	☐	☐	☐
Exercises covered all of the fields of the course	☐	☐	☐	☐	☐
Examples in Web-site facilitate the learning of subject	☐	☐	☐	☐	☐
Examples and optional exercises in Web-site facilitate the submitting exercises	☐	☐	☐	☐	☐

2. Usefulness of Visualizations tools

There were the following form of visualizations in Programming 1 course: animations made by Flash, interactive and non-interactive applets and Jeliot 2000.

a) **Please, estimate whether the Flash-animations and applets situated in Web-site facilitate the learning of those subjects. Please describe also why they helped or did not help you.**

____
____
____

b) **Please, estimate whether the Jeliot 2000 program helped you to better understand your programming code and / or data structures (like arrays). Please describe also why it helped or did not help you.**

____
____
____

QUESTIONNAIRE

c) Please describe how you felt the installation and use of Jeliot 2000 in general. Do you have any suggestions for improvements of Jeliot 2000 and its features?

____

____

____

____

3. Technical resources: learning environment.
Please estimate using scale 1 to 5 (1 = fully not-agreed, 2 = somewhat not-agreed , 3 = neutral, 4 = somewhat agreed, 5 = fully agreed).

Field to be estimated	*1*	*2*	*3*	*4*	*5*
Learning environment (WebCT) was easy to use	☐	☐	☐	☐	☐
Structure of learning environment was clear	☐	☐	☐	☐	☐
It was easy to submit assignments	☐	☐	☐	☐	☐
Example solutions were easily at handy	☐	☐	☐	☐	☐
No problems with installing Java compiler	☐	☐	☐	☐	☐
Java compiler was easy to use	☐	☐	☐	☐	☐

Please describe also your wishes or suggestions for improving the learning environment:

____

____

____

4. Support in problems
Please estimate using scale 1 to 5 (1 = fully not-agreed, 2 = somewhat not-agreed , 3 = neutral, 4 = somewhat agreed, 5 = fully agreed).

Field to be estimated	*1*	*2*	*3*	*4*	*5*
Advices given by supervisors at the University helped me to solve the exercises	☐	☐	☐	☐	☐
Advices given by supervisors at the University helped me to understand the difficult parts of the exercises	☐	☐	☐	☐	☐
I asked help and got it from other students via discussing forum in WebCT	☐	☐	☐	☐	☐
I followed regularly the discussions in WebCT and found the solusions to my problems as well based on the questions and answers made by peer students	☐	☐	☐	☐	☐
I got helped via discussing with peer students at my own high school	☐	☐	☐	☐	☐
Examplesolutions after exersices helped me to understand the bugs of my solution	☐	☐	☐	☐	☐
I got help from tutor in practical problems	☐	☐	☐	☐	☐

QUESTIONNAIRE

Tutor was usually available and ability to help in my learning problems. ☐ ☐ ☐ ☐ ☐

If you got help to your programming learning difficulties from otherside, pls name where?

____

____

____

5. Improvements for support

Do you have any suggestions for how to improve the support given over the web? Pls tell what kinds of changes the support will need, and what kinds of things we should cater for?

____

____

____

____

____

____

____

6. The usefulness of support

	Yes	*No*
I participated into the orientation meeting of ViSCoS in August 2002	☐	☐
I got to know the content of the guide for distance studies	☐	☐
I got to know the introduction site of courses in WebCT	☐	☐
I got to know the instructions delivered via "Students' Databank" in WebCT	☐	☐
I got to know the instructions for using of WebCT	☐	☐

Do you have any wishes or suggestions for online guides offered by supervisors? If so, pls describe what kinds of improvements they will need:

____

____

____

QUESTIONNAIRE

7. **Preceding programming skills.**
Pls answer to the following questions as the situation was **before** starting the ViSCoS studies.

a) I know the following programming languages:

____

____

____

____

b) Describe what kinds of tasks did you produce using the languages mentioned above (what kinds of programming)?

____

____

____

____

c) Describe how you have studied and learned these programming languages?

____

____

____

____

8. **The difficulty of programming.**
   a) **Estimate the difficulty of the following programming topics using scale 1 to 5, where 1 = easy, 2 = quite easy, 3 = neutral, 4 = quite difficult, 5 = very difficult.**

Field to be estimated	*1*	*2*	*3*	*4*	*5*
Variables and symbols	☐	☐	☐	☐	☐
Primitive data types	☐	☐	☐	☐	☐
Assingments	☐	☐	☐	☐	☐
Input and output	☐	☐	☐	☐	☐
Relational operators (<, <=, >, >=, ==, !=) and boolean data (true, false)	☐	☐	☐	☐	☐
Conditional statements	☐	☐	☐	☐	☐
Loops (for, while and do-while)	☐	☐	☐	☐	☐
Boolean operations (&&, ||, ^, !)	☐	☐	☐	☐	☐
Arrays	☐	☐	☐	☐	☐
Nested loops, selection sort	☐	☐	☐	☐	☐
Randomize numbers	☐	☐	☐	☐	☐
Methods	☐	☐	☐	☐	☐
Applets	☐	☐	☐	☐	☐
Graphics	☐	☐	☐	☐	☐
KeyEvent	☐	☐	☐	☐	☐

   b) **If you ticked 4 or 5 in some of the topics mentioned in 8a, pls describe why you felt those as difficult to learn?**

____

____

____

____

QUESTIONNAIRE

9. Support methods.
a) Did you wish for some of the following multiplicity learning techniques as supporting ViSCoS learning?

	yes	no
Face-to-face lecture at the university	☐	☐
Lecture via videoconference	☐	☐
Real-time discussions via Netmeeting or WebCT's chatroom	☐	☐

Something else, what?

____

____

____

b) If you ticked 'yes' in some of the previous supporting methods, pls tell in which of the programming topics you will especially wish for these support methods and why?

____

____

____

____

____

____

10. If you have dropped out, pls mark the time of dropping out using 'x' to the following time zone

September	October	November	December	January	February	March
Starting ViSCoS studies Programming 1 course starts	Exam for Introduction to Computer Science -course		Exam for Programming 1	Programming 2 starts Re-exam for Programming 1	Research fields of Computer Science – course starts	
☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐	☐☐☐☐

11. What was the main reason to drop out (you can also number several reasons using order 1 = the most important reason, 2 = the second important reason, etc.)?

☐matriculation
☐coming back on next academic year
☐felt theory too complicated
☐felt assingments too complicated
☐did not pass Programming 1 exam in December
☐did not pass Programming 1 re-exam in January
☐personal changes in learning situations
☐course did not meet my expections
☐course did not interested me
☐some other reason, what?

____

☐lack of time,
pls, estimate also why you had lack of time?

____

____

____

QUESTIONNAIRE

12. Learning time:

a) Estimate the time you used to study programming weekly. Use the following scale 1 = less than one hour, 2 = from 1 to 2 hours, 3 = from 3 to 4 hours, 4 = from 5 to 6 hours, 5 = more than 6 hours in week.

Field to be estimated	*1*	*2*	*3*	*4*	*5*
Study the theory of coursebook	☐	☐	☐	☐	☐
Study the web-based theory in WebCT	☐	☐	☐	☐	☐
Optional assingments in WebCT (no need to submit)	☐	☐	☐	☐	☐
Weekly assignments (need to submit)	☐	☐	☐	☐	☐
Voluntary programming by myself	☐	☐	☐	☐	☐
Participating some other programming course in high school simultaneously with the ViSCoS Programming 1	☐	☐	☐	☐	☐
Other type of studying, what?					
______________________________	☐	☐	☐	☐	☐
______________________________	☐	☐	☐	☐	☐

b) Estimate whether you used more time for studying the most difficult topics than others. If so, pls name the field(s) demanding more time.

____

____

____

____

____

____

____

13. Did you need to find some optional materials in addition to web-based theory in WebCT and the coursebook (e.g. using Internet or other Java course books)? If so, pls tell in what progamming topics you needed more information and where did you get it.

____

____

____

____

____

____

____

____

14. Do you want to change something in ViSCoS Programming 1 course? Do you want to decrease or increase something to the syllabus?

____

____

____

____

____

____

____

QUESTIONNAIRE

15. Do you have any other wishes for improving the Programming 1 coursematerial or technical feasibilities?

16. Please, describe what you learnt and how did you benefit from the ViSCoS courses.

17. Any other suggestions or comments to the ViSCoS organizers?

Thank you for your answers!
We appreciate every single feedback that will help us in our work within improving the ViSCoS courses.

This questionnaire is related to the post-graduate research work of
Sirpa Torvinen, University of Joensuu
Contact: sirpa.torvinen@joensuu.fi

GLOSSARY

Guide for distance studies	It is a guide for students which consists information about ViSCoS courses, contact information, information on related coursebooks, recommendations for independent online study etc. It is delivered to every student during the orientation meeting at the beginning of ViSCoS studies.
Guide for usage of the learning environment	It is a guide for students which consists of instructions for the use of the learning environment WebCT. It also includes the address for WebCT server, login instructions, password instructions, and instructions on how to submit exercises into the learning environment. It is delivered for every student locally via tutors by CD-ROM.
Semester 1	The first fall semester during the 1.5 –year ViSCoS curriculum.
Semester 2	The spring semester during the 1.5. –year ViSCoS curriculum.
Semester 3	The second fall semester during the 1.5 –year ViSCoS curriculum.
Student who dropped out	Students who dropped out are student who started Programming 1 course but did not participate in the final exam in the end of the course or in the re-examination right after the course. These students did not either continue on to the following ViSCoS courses in Semester 2.
Tutor, tutor teacher	The teacher who cooperates with the university during the ViSCoS program. Each high school has one tutor who is responsible for their own high school's ViSCoS students. Responsibilities for tutors are the following: at the beginning of studies tutors deliver the WebCT login usernames and passwords to students (supervisors from university send these to the tutor) and ensure that every student can • login to the learning environment • download material from the learning environment • submit exercises into the learning environment • install a Java compiler After the studies have started, tutors proctor the exams at the high school (except the programming exams which are at the university).