Through the eyes of instructors: a phenomenographic investigation of student success

Päivi Kinnunen
The Laboratory for Software Technology
Helsinki University of Technology
Helsinki, Finland
pakinnun@cs.hut.fi

Robert McCartney
Dept. of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269 USA
robert@engr.uconn.edu

Laurie Murphy
Dept. of Computer Science & Computer Engineering
Pacific Lutheran University
Tacoma, WA USA
lmurphy@plu.edu

Lynda Thomas
Department of Computer Science
University of Wales
Aberystwyth, Wales
ltt@aber.ac.uk

ABSTRACT

In this paper we present a phenomenographic analysis of computer science instructors' perceptions of student success. The factors instructors believe influence student success fell into five categories which were related to: 1) the subject being taught, 2) intrinsic characteristics of the student, 3) student background, 4) student attitudes and behaviour and 5) instructor influence on student development. These categories provide insights not only into how instructors perceive students, but also how they perceive their own roles in the learning process. We found significant overlap between these qualitative results, obtained through analysis of semi-structured interviews, and the vast body of quantitative research on factors predicting student success. Studying faculty rather than students provides an alternative way to examine these questions, and using qualitative methods may provide a richer understanding of student success factors.

Categories and Subject Descriptors

K.3.2 [**Computers & Education**]: Computer & Information Science Education—*Computer Science Education*

General Terms

Experimentation, Human Factors

Keywords

Phenomenography, qualitative analysis, student performance, instructor perceptions

ICER'07, September 15–16, 2007, Atlanta, GA, USA.

1. INTRODUCTION

What causes some students to succeed at computer science while others struggle? This has been a central question in computer science education research for many years: during the dot-com boom, when educators longed for a means to filter students who didn't have "what it takes" from their over crowded classrooms; today, when those same educators wonder why capable students shy away from our discipline in droves; all along, to try and understand the leaking of women and minorities from our proverbial pipeline.

Research in this area has generally focused on correlations between quantifiable measures of student success, such as exam scores or course grades, and quantifiable student attributes, such as math background. These investigations have primarily focused on introductory programming students, with an eye toward being able to predict who among them would succeed and continue on in the discipline.

The current study takes quite a different approach: examining student success from the perspective of computer science educators. It is based on a qualitative analysis of interviews conducted with 16 computer science instructors from six countries. Using a phenomenographic approach, our analysis organises instructors' qualitatively rich understandings of student success into a set of distinct categories: understandings pertaining to the subject being taught; those focused on intrinsic qualities of the student; those related to students' previous experience; those highlighting student behaviours and attitudes; and finally those based on the role the instructor plays in student development.

A detailed analysis of these categories reveals many areas where these instructors' understandings of student success are well aligned with previous research. It also suggests some beliefs about students that are based largely on individual experiences and for which there appears to be little corresponding empirical research in the current literature.

These beliefs are important because of the role they have played in how these instructors have approached their teaching but also because they open up a forum for other CS educators to reflect on their own beliefs and their implications.

Furthermore, differences between educator beliefs and empirical evidence suggest potentially fertile areas for future research.

This phenomenographic study presents one facet of a larger body of work emerging from the Second International Workshop on Phenomenography in Computing Education Research (PhICER 2006), of which the authors were all participants.

The structure of this paper is as follows. In Section 2 we describe the specific research focus of this study. In Section 3 we present a brief overview of the largely quantitative body of work focused on predicting student success and introduce phenomenographical studies in CS Education research. In Section 4 we present the methodological approach employed in this study to collect and analyse data. We present the results of our analysis in Section 5, and discuss these results in relation to other work in Section 6. We conclude, in Section 7, with summary of our results and some suggestions for further study.

2. RESEARCH FOCUS

This study focuses on the following question: *What are instructors' perceptions of students' success?* Specifically, this study tries to identify *what instructors see* as the factors affecting student success, rather than trying to identify these factors. The value of such an approach is twofold: each instructor's view is based on experience teaching many students, and instructor views of the learning process affect the way they choose to teach.

As this is a phenomenographic study, we take a second-order perspective in studying the instructors' views—we observe their reflections on teaching, rather than their teaching itself. This is denoted in Figure 1 by arrow 1 to the cloud, which represents the instructor's view of the complex set of factors involved in the teaching and learning of a particular subject, relative to the success or failure of the students.

3. BACKGROUND

3.1 Student Success

Before the introduction of widespread computer science education, potential employers of computer programmers needed to hire people with no programming experience. Before hiring, they often employed predictive tests that sought to discover who might be good at computing (specifically programming) and who would not [41]. In more recent years, research from the CS Education community has focused primarily on students' performance in introductory programming courses as correlated with quantifiable factors, such as mathematical background, major or gender. Some recent studies have synthesised these two threads somewhat by correlating measurable psychological characteristics of students with their performance in introductory programming.

Research in this area can be categorised in different ways; first of all, in terms of the students and the subjects they are studying:

- Some investigators look at success across the curriculum [9], at graduation [27] or drop out rates [42], but most examine only success in introductory programming; some restrict their focus even further to a particular "flavour" of programming, such as objects-first [36] or model-driven [2].
- Within the studies of first year students, the sorts of measures of success used include: marks on programming assignments and programming tests [2, 4, 5]; exams [10, 39]; combinations of programs and exams [3, 25, 30, 36, 38]; and overall success in the first year [9].
- Most researchers have studied computer science majors but other studies look at non-majors [38], or consider the influence of major in mixed classes [2, 28].

Researchers have examined different factors that may help determine success:

- Gender, prior programming experience and mathematical background are very frequently considered in this kind of investigation [2, 4, 9, 10, 28, 30, 33, 37, 39], but some researchers have also focused on what sort of experience [27, 33, 36], and what sort of mathematics [28].
- Some studies have examined the influence of general learning characteristics and strategies such as: learning style [10], abstraction ability [3], knowledge organisation [38], approaches to study [16], self-efficacy [37, 38], expectations [30] and comfort level [4, 37, 39].
- Recent investigations have sought to link other factors to success in programming such as: map-drawing style [32], working memory and field independence [25] and a self-designed test on assignment [11].
- Since it has become apparent that no "silver bullet" exists to predict or explain student success (although, see [11] for a recent claim to the contrary) researchers often consider the influence of combinations of factors on success [30], particularly in programming.

Although many factors have been examined in these studies, the results are often inconclusive or contradictory. The number of variables, both in defining success and to be considered for predicting or influencing it, makes this a very hard problem. Mathematical background [2, 39, 10] and factors related to attitude to learning such as self-efficacy, expected grade and comfort level [5, 28, 37, 39] seem to have some effect. Intellectual characteristics, such as abstraction ability, knowledge organisation, and field independence may also have an effect [3, 38, 25], but this needs more research. Some commonly-examined factors such as gender, age, and learning style do not seem to be very significant factors in determining success. Relatively few studies have focused on how instructors can intervene with those particular populations who are identified to need different kinds of instruction [8, 34].

3.2 Teaching and learning theory

Educational theorists have considered the role of the instructor in teaching and learning—what they do to affect student success.

Biggs [7] has examined the role of the teacher in university education generally, and defined three levels of teaching competence based on instructors' assumptions about what affects success in learning. He claims that:

- A level one teacher focuses on *what the student is*, assuming that success or failure is primarily a function of student differences.

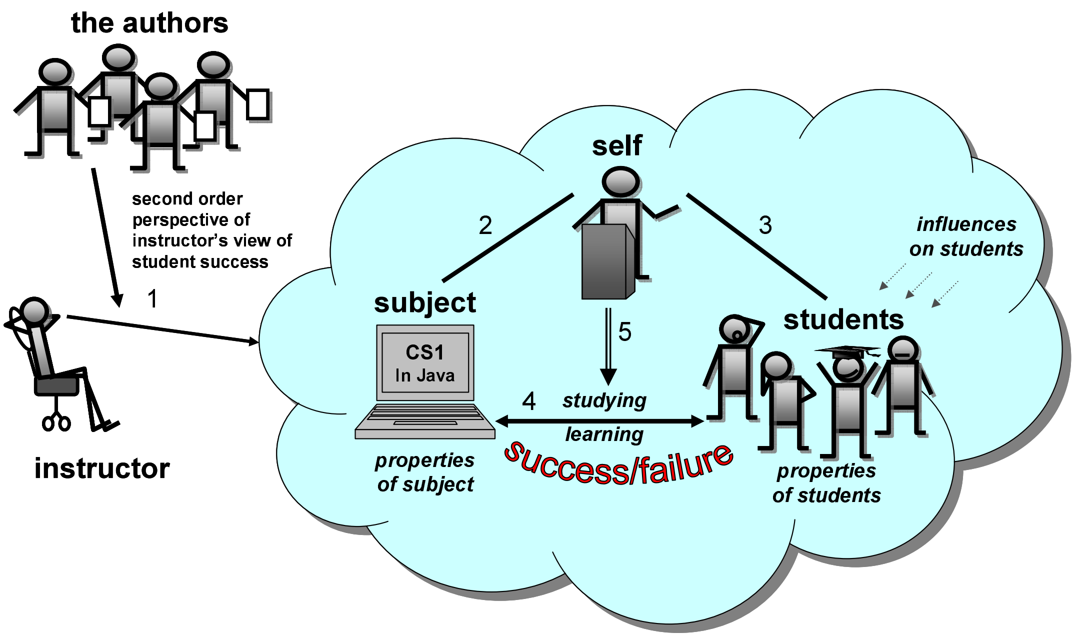

Figure 1: Research focus (interior of cloud adapted from Kansanen and Meri, 1999).

- A level two teacher focuses on *what the teacher does*, assuming that success or failure is primarily a function of the quality of the teaching.
- A level three teacher focuses on *what the students do*, assuming that success or failure is a primarily a function of the students' learning-focused activities–activities that work within the context of the class, are supported by the instructor, and lead to desired levels of understanding.

Based on his assessment of the superiority of level-three teaching, Biggs presents a model for teaching that relates the desired learning outcomes, the activities that support these outcomes, and the assessment tasks used to evaluate the degree to which the outcomes are achieved.

According to Kansanen, the core of the instructor's profession is to bring about and guide students' studying process [20]. The instructional process is a totality that includes teaching, studying and learning. Hence, it includes the instructor's acts (e.g. teaching), students' acts (studying), and the outcome of studying, that is, learning [19]. The relationship of the instructor, content and the students can be visualised as a triangle as in [21] (see modified version in Figure 1) where the instructor has relation both to the content (arrow 2) and to the students (arrow 3). The instructor's relation to the content expresses itself as the instructor's expertise on the subject matter. The connection between the instructor and the students (arrow 3) becomes concrete in pedagogical interaction, which can be described as an activity with a content and purpose. The students' relationship to the content becomes visible as an act of studying and learning (arrow 4). It is especially this relationship between the students and the content that is in the core focus of the instructor's profession. The didactic relation (arrow 5) portrays the instructor's intentions and acts to facilitate and guide the students' studying process and hence therefore indirectly also learning.

3.3 Phenomenography

Phenomenography is a research approach that is designed to elicit the variation in ways of experiencing things on a communal level. The roots of the approach are strongly connected to empirical studies of learning. The ulterior motive of studying the variation in how students experience certain studying/learning related situations or concepts is to make sense of how students handle those same situations. [26]. Therefore, the results of a phenomenographic study have the potential to provide concrete tools to understand and evolve instructional processes. The aim of phenomenographic research is to obtain a rich description, but not to necessarily find generalisable results. This aim originates from the desire to gain insights on the phenomenon studied. The interview is typically used to collect data in phenomenography, although other methods are possible [18, 26]. Due to the richness of the data, the number of interviews in phenomenographic research is usually restricted, for example Eckerdal and Thuné interviewed 14 students [14]. Selecting interviewees so that they represent a wide range of variation concerning the relevant characteristics is one way of ensuring unbiased and unrestricted results. It is expected that data collection reaches a saturation point, that is, after some number of interviews the same themes continually emerge and additional interviews do not uncover any new themes.

Examples of research applying phenomenography to the CS education domain include investigations of first year students' understandings of learning to program [13] and their conceptions of *object* and *class* [14], as well as upper-level students' approaches to learning CS [6]. Phenomenographic studies seeking to better understand CS education from the educator's perspective include investigations of CS educators intentions when teaching data structures [24] and an investigation of instructors' attitudes concerning their teaching [23]. Like ours, this last study looked at instructors' understandings of their own teaching. They considered teaching in relation to lab practicals, motivation, granularity of focus, and conceptions of when their own teaching was a success. In each of these areas, their results were consonant with Fox's identification of personal theories of teaching [15]: comprising either two categories (teacher- and student- focused), or four categories (transferring, shaping, traveling, and growing), corresponding to increasing amounts of student involvement in the teaching and learning process.

4. METHODOLOGY

This research originated with the Second International Workshop on Phenomenography in Computing Education Research (PhICER 2006) held in Canterbury (UK), September 11-12, 2006. At that workshop, three leaders[1] instructed a group of researchers in the use of phenomenography, starting with data collected by all of the workshop participants. This paper is an extended offspring of a section of a joint PhICER paper being written by the instructors of the workshop and the participants (manuscript in preparation).

Prior to the workshop, participants interviewed one to three computer science instructors at their home universities. Each researcher chose whom they wanted to interview, without limits on the topics and level of the courses the interviewees taught. The general topic of the interview was how instructors approach their teaching, specifically: what do instructors perceive is difficult for their students, and what do they do if they notice struggling students? The PhICER leaders supplied an interview script (see Appendix for script) and general guidance about interviewing techniques and the expected length for the interviews. Each interviewer was encouraged to ask his or her own questions in addition to the ones in an interview plan. Hence, methodologically speaking, the interviews followed the interview guide approach [18]. All interviews were recorded and transcribed verbatim and translated to English if in another language. Our data consists of 16 English-language transcripts of interviews conducted by 12 researchers from six countries: Finland, Germany, Ireland, Sweden, the U.K., and the U.S.

During the workshop, the participants became familiar with the data by closely reading the transcripts. In smaller subgroups they discussed themes that were emerging from the data and could be approached using phenomenography. It was at this point that the authors of this paper chose our research question: What are instructors' perceptions of students' success?

After our subgroup decided on our research question, we proceeded to analyse the data as follows. With a focus on students' success we first divided the transcripts among the authors of this paper (a "vertical" division); each member then read his or her transcripts to see what categories emerged. Potential categories were listed and discussed and then the authors went through the transcripts again, looking to see if the set of potential categories was complete and consistent.

Subsequently the data were divided "horizontally", i.e., by category rather than by transcript, with each researcher looking for particular categories in all of the transcripts. As a result of this and subsequent discussion, a minor redefinition of the categories emerged.

[1]PhICER was organised and led by Anders Berglund, Anna Eckerdal, and Arnold Pears, of Uppsala University.

Once we had a set of categories, we examined the relationships among them, trying to determine in particular what they had in common and how they differed. This involved a progression of alternatives that were discussed among the authors until they reached agreement. This final set of categories (the outcome space) and how they are related are discussed in the next section.

5. RESULTS

The results of a phenomenographic study are a description of the different categories of understanding found in the interviews, and the relationships among these categories. In this section we describe the categories, with examples of illustrative quotes, and examine the relationships among these categories.

5.1 The categories

The different instructor understandings of students' success or failure in learning "their" subject are summarised in Table 1. This table gives an overview of five categories—the different things that affect student success in the instructors' opinions. It provides a brief description, the focus of the category (what was being discussed), and the dominating aspect (what thing was primary in the discussion).

In the following, we discuss each category in more detail.

Category 1 – Subject

In category 1, the instructor's focus is on how the nature of the subject being taught impacts students' success. Their emphasis is on aspects of their particular computing topic area, rather than on the students or the instructor. These may include very specific concepts and topics, characteristics of different programming paradigms, or general issues like the nature of learning to program.

Some of the quotes in this category focused on the inherent difficulty of specific topics.

> *D1:* Because you know, pointers are less concrete than values. So they don't always grasp that whole concept of the address... I just think they're hard to visualise. (q1)

Other instructors spoke in broader terms, focusing on the nature of computers and computer science in general.

> *P2:* And what comes to the errors and such, computers are cruel. They expect things to be exact and they just don't work, will not co-operate with you unless you are pretty exact about telling them to what to do. (q2)

Category 2 – Intrinsic

In category 2, the instructors see student success as due to intrinsic student abilities or capabilities. Sometimes instructors referred to "magical" attributes—those that allow students to succeed, for which the instructor has no specific explanation. Other times they mentioned specific attributes: natural level of ability, ways of thinking, and attributes relating to gender, culture, and language. These features do not seem to be a result of the student's current or past education, nor are they affected by the instructor, the student, or the college learning experience.

There were quotes expressing students' success as a "magical" feature—for some students it "just clicks" and for some

	Category	Students' success is understood...	What is the focus?	Dominating aspect
1	Subject	as being influenced by the inherent nature or quality of the subject matter to be learned.	The nature of the subject being taught and how it impacts students' success.	Subject
2	Intrinsic	as being caused by an intrinsic quality or "magical" ability of the student, something neither the instructor nor the student can significantly impact or change.	Students' intrinsic attributes and how they are connected to success.	Student
3	Previous experience	as being influenced by the amount, quality, or design of preparation before coming to the current course (e.g., in high school or in prerequisite CS classes).	How students' background knowledge and previous experience impact their success.	Student
4	Attitude/ Behaviour	as being influenced by the student's attitude or behaviour (e.g., hard work or being proactive).	Students' attitudes or behaviours and how they are connected to success.	Student
5	Developmental	as a process of developing an understanding, way of thinking or skill, with the instructor's help.	Strategies the instructor uses to help students succeed.	Instructor and Student

Instructor may also see each level through the filter of his/her own experience

Table 1: Different categories of instructors' understanding of student success/failure.

it does not. Instructors were not able to specify why it happens.

> *J1:* No, no there are definitely some students that it just clicks [snaps fingers] with them. They have no trouble at all. They'll be, and in fact, quite a few students, quite a few students just have a knack for that. (q3)

Some instructors see success as connected to students' natural level of ability, that there are strong students, weak students, and those in between. This instructor sees success or failure as largely predetermined in the top and bottom groups.

> *D2:* And I'm thinking some of these top students, you know, I could hand them the book and disappear, and they would figure it out, and maybe I'd help smooth things over, but they'd be learning pretty much in any environment...And then there are some weak students who are not making it, and frankly, I'm not sure it would matter who was teaching them... (q4)

Some quotes reveal more specifically instructors' conceptions of the reasons behind different ability levels. For example, a specific intrinsic way of thinking, which enhances success.

> *D2:* I don't think - it seems like the good programming students, I guess, the students who come in maybe with no programming experience at all, but if they just think that if they have a good ability think logically and organise their thinking, they somehow take to it really pretty well... (q5)

Gender, culture, and language are also identified as attributes that affect the student's way of studying, willingness to ask questions, and comfort level in the course and therefore also their likelihood to succeed.

> *D1:* He grew up in a society where they did do things as groups. All his learning was in group learning. And he was attending his class at age 7 with a group of other boys. And also the African society is very concerned with what other people think. So, in the regular classroom situation it had been really difficult for him to ask questions because he didn't want to show any lack of understanding or stand out in any way. (q6)

Category 3 – Previous experience

In the third category, instructors understand success as being affected by the students' previous experience, background knowledge, and previous education. Some of this experience is directly applicable to their current learning—previous coursework or experience in computing, for example, that provide the student with particular skills or a general familiarity with using computers. Others are less direct, a result of educational background in other subjects such as mathematics. As with the second category, these are factors that are outside the direct influence of the instructors in the context of their classes. In contrast to the category 2 (intrinsic ability) quotes which generally tried to explain the highly successful students, these quotes tended to link student problems with inadequate preparation.

Many of the quotes discussed the links between previous mathematics experience (in areas such as logic or abstract reasoning) and their current level of success.

> *D2:* I think most haven't thought too abstractly before they get into a CS program. Maybe if they took their math a long way as in high school, but even in high school, high school math is mostly, like, do the calc stuff and not logic and things like logic, and so I think most college students really haven't had a lot of practice. (q7)

The source of some difficulties was attributed to an expected lack of familiarity with the subject.

> *J1:* I think, it's not so much that they've never seen event-driven programs. I mean, they've seen programs where events occur, and they've used those applications, but to actually see it from the inside where they're doing the programming. I'm sure that's totally foreign to them. (q8)

Students' successes or difficulties in introductory programming courses were also linked to whether they came to the course with previous programming experience, although it was not a prerequisite requirement.

> *J1:* If I remember it seemed that those that didn't understand (flow of control) were those that didn't have prior experience. (q9)

In general, the variability in students' programming background was acknowledged. Although most quotes identified experience and background knowledge as positive features, there were others that pointed out potential problems.

> *X1:* ...what makes things even worse is that they have a priori understanding of what it is because they've heard about it through friends, through the media, through whatnot. They've heard the word, and they believe they know what it represents, and they come with this aggravating factor that biases them very often in the wrong direction. (q10)

> *P1:* One thing could be that if you have learned to program without objects before you come to the course then you have some idea what programming is about. Then objects don't correspond with your preconceptions. (q11)

Category 4 – Attitude and Behaviour

In the fourth category, the instructors perceive student success as being influenced by the students' attitudes or behaviour. Much of the discussion centered on student work habits—the need for students to put forth sufficient effort and to take responsibility for their own learning. They also discussed more general issues of the sorts of attitude needed to be successful. Among the attributes mentioned in the interviews under this category were: amount of work done, study habits, being proactive, general attitude, expectations, emotions and motivation. Unlike the previous categories, these attributes are seen as things that the students can affect and change.

In the following, the instructor addressed the importance of hard work and persistence—features that are central in learning.

> *D1:* I think that students just aren't always willing to put in the time that it takes. They think that things should come easily and I think that if people spend enough time looking at code and trying things out themselves, you can't just watch what's done. You have to try it yourself. And a lot of it's just – you know, programming just takes a ton of time. (q12)

On the other hand, procrastinating and reluctance were seen as issues hindering success.

> *D1:* Because they're used to procrastinating and you can't procrastinate on coding because you have no idea how long it's gonna take you (q13)

During the interviews, instructors addressed the importance of study habits. Good study habits can promote success and less ideal habits on the other hand can hinder it. The importance of students' meta-knowledge concerning their learning preferences was also acknowledged.

> *P1:* ...students come to the class without any preparatory studying. They haven't read chapters that relate to the topic. Instead they expect that teaching assistant will explain them everything from the beginning... (q14)

> *D1:* Well, no, I think that our students have some responsibility to be aware of the way that they learn. And just because a student next to them for example doesn't need to write down the algorithm doesn't mean they don't need to write down the algorithm... (q15)

The active role of students was related to success. Students need to ask questions and seek help if they do not understand something.

> *D2:* So if they're very proactive - and I can think of a number of students over the years who just did so well, and part of it was because they were so – they worked hard, and they came, and when they struggled, they came in for help... (q16)

A positive and inquisitive attitude was perceived as furthering success.

> *G1:* ... I find that the people who pick it up the best are the people that ... are A) interested in the subject and they are inquisitive and willing to explore within their programming, they might do a simple example and then they might extend it and play around with it, and they are the people who do well, because they are interested, inquisitive, and willing to explore; and other that are afraid of the whole computer they can't make... yeah... the are not able to create the program are a bit less successful. (q17)

Students' presumptions concerning CS or their own inability to learn CS were seen as hindering success. Instructors also mentioned negative emotions, such as fear, as impediments to success.

> *O1:* Some people are always afraid of things and either haven't figured out to make it not fearsome or there's just some things that are always gonna be fearsome to them. (q18)

Category 5 – Developmental

In the fifth category, the instructor looks at his or her role in helping the student succeed–how his or her actions can influence student success. The approaches are varied—teaching material in alternative ways to account for differences in student learning styles; giving the students appropriate examples and problems to work on; effectively assessing and monitoring student progress; and providing a positive learning context in his or her class. The focus here is not on the instructor, rather it is on the instructor-student interactions.

When differences in learning styles were thought to be the source of students' difficulties, instructors attempted to help students by presenting the material in multiple ways.

> *A1:* ...you have explained something a thousand times but then when you do it again then you realize that, 'Ah, okay. I have to come up with a new way of saying this because this student doesn't get my other ways.' (q19)

Some stressed the importance of of building on the appropriate examples.

> *G1:* I start with simple problems and I think that this builds up their ability to see though the components of the problem. (q20)

One expressed frustration when students wanted examples in a situation where examples were not possible.

> *D2:* And then in the automata, they're asking me, 'Can you give me an example?' and at that point, I'm at a loss, because, I mean, we're trying to show that something doesn't exist. (q21)

Others indicated the importance of gauging where students were having difficulties by using formal or informal assessments.

> *J1:* So, I think it's important that you do that kind of formative assessment to see how they're doing. And if they don't understand, you know, hit the topic again. (q22)

Others stressed the importance of providing a context that maximises student learning.

> *O1:* I think what I should be doing is putting my students in environments where they have the maximum opportunity to learn what it is they're trying to learn ... putting them in a context where they can, with the right experiences, learn the things they need to learn. (q23)

> *X1:* It's really throwing them in deep water...Just give them...something giant and let them sink in it, and the ones that are sinking say, 'See? That's why we have abstraction.' Showing them then with abstraction it becomes manageable. (q24)

Instructor-centric filter

Some responses concerning students' success were explained in terms of instructors' abilities and experiences. There are two broad categories of these: those where the instructor compares his or her current self to the students, and those where the instructor compares him- or herself as a student to the current students.

In some quotes, the instructor considers student behaviour in comparison with his or her own current behaviour.

> *J1:* So you'll see for example, that they'll save files, and I do the same thing, accidentally, where I'll save a file and I think I'm saving it in one directory and in fact it's in another. And I'll try to compile and you know the class won't be there that it's looking for because I'm in the wrong directory, in the wrong folder. But you know it's pretty clear to me when I get the error message what I've done wrong. For them they get the error message – some exception, some class not found – and they're not sure what's wrong. (q25)

In other quotes, the instructor introspectively recognises that one's own current knowledge and ability may not be a good predictor of student abilities and knowledge.

> *D2:* So I was thinking about other things—introductory programming—just the idea that things that are so ingrained in us, we don't realize that they can have trouble with just the idea that the sequential execution of an algorithm or a method, or an assignment statement. (q26)

Other quotes show the influence of the instructor's learning experience on his or her teaching.

> *P1:* It is easy to understand since I have learned to program with language that wasn't an object language. I remember that at first I had difficulties to understand what this all is about. (q27)

Another instructor bases his pedagogical approach on his or her experiences as a student.

> *X1:* I enjoyed it a lot more when it was engaged like that, and very often it's fun to see the students after a while start to enjoy the game, and see really that it's a game... But it's based on my personal bias and subjectivity as to what I think is most effective for me, and that's how I relate more easily to the problem. (q28)

Initially, we considered these instructor-centric quotes to represent a different category of understanding. On closer examination, however, we saw these as attempts to understand students' success by comparing his or her own intrinsic abilities, previous experiences or behaviours with those of the students, and so could be used for any of our categories—a lens for viewing the other categories rather than a category of its own.

5.2 Relationships among the categories

The five categories in our outcome space illustrate different ways that instructors understand student success, that is, the categories capture the different understandings, but not necessarily the different instructors. Indeed, all of the instructors showed more than one of these understandings in the interviews.

The *Subject* level attributes student success to inherent subject features. As instructors are generally subject experts, they all presumably understand connections at this level.

The *Intrinsic* level focuses on the student's innate ability and characteristics. This is done, however, in the context of learning the particular subject at hand (as is reflected in the quotes), so it can be considered an enhancement of *Subject* understanding—taking what is known about subject characteristics and adding in knowledge about individual student variations.

The *Previous experience,* and *Attitude/Behaviour* levels focus on other largely disjoint aspects of the students. Each of these builds on *Intrinsic* level understanding—the effects on success of experience or attitude and behaviour are considered in the context of individual innate student characteristics, so these levels are both enhancements to the *Intrinsic* level. Rather than one including the other, however, these provide distinct enhancements.

The *Developmental* level understanding focuses on the instructor-student interactions—things the instructor does to help the student learn. A *deep* understanding at this level encompasses all of the other levels, and builds on the *Intrinsic, Previous Experience,* and *Attitude/Behaviour* levels—understanding the student differences is a prerequisite for effective interaction. The extent to which understandings at this level consider the previous levels may vary, however.

In contrast to most phenomenographic work, we do not see our categories as forming a strict hierarchy. Such a hierarchy could be formed by combining categories 3 and 4, but the text indicated that these understandings were different and distinct. In general, the higher-numbered levels correspond to increasingly rich understandings, and the relationships among the levels are fairly clear: levels can be seen as extensions of other levels formed by adding things to those levels.

6. DISCUSSION

In this section we discuss our results in relation to the background from Section 3: work on student success prediction, and teaching and learning theory. Additionally we consider the reliability, or trustworthiness of these results.

6.1 Predicting student success

Previous studies highlight an array of factors computer science educators believe may be important predictors of student success. However, because so many of these studies are quantitative, they have obviously focused largely on factors that researchers could quantify. The discussion that follows compares the overall landscape of this vast body of research with the qualitative descriptions of factors contributing to student success we found during our analysis.

Category 1 – Subject

Sentiments expressed by Dijkstra's "On the Cruelty of Really Teaching Computing Science" [12] were echoed by the instructors in this study; with one stating, when it "comes to the errors and such, computers are cruel" and others describing introductory programming as "quite overwhelming" and "totally foreign."

In addition to being thought of as generally challenging, we see two themes where the nature of CS is highlighted in the success literature:

1. where success studies have been conducted within the context of a particular paradigm or flavour of programming (e.g., OO [4, 36] or model-driven [2]), and
2. where measurements of ability (e.g., abstraction [3]) or background (e.g., previous mathematics experience [2, 4, 9, 10, 40, 39]) suggest that researchers believe these skills comprise essential elements of computer science, or that they may transfer effectively to the computer science domain.

Instructors in the current study expressed contrasting views of OO programming, similar to those found in the literature, which suggest OO may be a positive factor pedagogically, by levelling the playing field for students with less experience, particularly women [36], or by providing, as one instructor in this study expressed, an "easier way to think." In contrast, another instructor stated that OO "makes [programming] more difficult," which supports well-known assertions in the CS Ed literature suggesting OO may present serious obstacles to beginning students because it "adds the overheads of class structure to a procedural system" [29](p. 145), and also because "there must be an *object-oriented paradox*: how is it possible to forget detail that you never knew or even imagined?" [1] (p. 260).

Abstraction, proclaimed recently as possibly "the key to computing" [22], was also seen as fundamental for student success by instructors in this study. They expressed that abstraction was something they wish students came to their classes knowing more about and that it is an important and challenging topic to teach. Despite the view that abstraction is key for success in CS, a recent study [2] found no significant correlation between abstraction level, using Adey and Shayer's cognitive development stages as an abstraction measure, and exam grades in a CS1 course.

During the current study, instructors also mentioned characteristics of specific topics or concepts, such as algorithm analysis, pointers and concurrency, as presenting unique challenges or obstacles to students' success. While the challenges involved in teaching and learning these various concepts is widely accepted, with a few exceptions (e.g., [27]), specific concepts or topics are rarely considered in studies investigating student success.

Category 2 - Intrinsic

Similar to quotes from the instructors in our study, much of the success literature focuses on intrinsic student attributes as possible predictors of success.

Relationships between gender and success have been frequently explored (see e.g., [2, 4, 16, 28]); data on gender is easily gathered and highly relevant to concerns about low enrolment of women in CS. While evidence suggests that gender is not generally a factor for success (e.g., [2, 16, 28]), and comments by instructors in our study neither disputed nor highlighted that evidence, one instructor did suggest that, from his or her experience, it appeared that "women would do better on exams" and that some men "may have done more with their assignment." The notion that men and women may have different strengths or approaches to learning computing has also been debated in the literature [35]. Studies investigating a host of other gender differences, such as expectations and attitudes toward computing [31] and previous programming experience [27, 33, 37], appear in the literature as well but were not expressed as factors for success among instructors in the current study.

Other demographic variables, such as age and years in college, were similarly not mentioned, and typically have been found to have no significant relevance for predicting success (e.g., [2]), although [30] did find that students age 16 to 18 and looking for an 'A' in the course tended to be highly successful.

Correlations between non-demographic, yet seemingly intrinsic, student attributes have been investigated, usually by administering established psychological tests or surveys.

These have included personality type inventories [8], learning styles [10, 16, 34] and tests of cognitive variables [3, 4, 5, 25, 32].

Introductory students' personality types, as determined by the Myers-Briggs type inventory, in [8] have been found to influence programming performance (*sensing* students outperformed *intuitive* students and *judging* did better than *perceptive*) but not their exam grades or overall course average. Personality type *per se* was not mentioned by instructors in the current study, although they did mention personality traits such as being positive, inquisitive and motivated as factors contributing to success. Culture and language, thought by instructors in this study to be important factors influencing students' comfort level and willingness to ask questions, to our knowledge, are not addressed explicitly in the success literature.

Results comparing learning style preferences with academic performance have been mixed; "no long ranging effect" in a study using Kolb and Gregorc learning style assessments [16], no significant differences in performance across Kolb learning styles in an introductory programming course [10]; significant differences in performance between different learning style groups using the Felder-Silverman scale [34]. Learning styles were mentioned by instructors in the current study both as a factor that students must "take responsibility for" and something the instructor should take into account when teaching.

The relationship between success and a range of cognitive tests has been explored, again with mixed results. As mentioned previously, [3] used Adey and Shayer's cognitive development stages as an abstraction measure, and found no significant correlation between abstraction ability and final grade in CS1. In experiments with students enrolled in a third-level OO course, [4] found only a weak correlation between programming and cognitive abilities, as measured by an in house cognitive test, while [5] found that the use of cognitive strategies did not appear to relate to performance. Visual and spatial tests appear to be more predictive, with cognitive tests of "field dependency" found to be very highly correlated with CS students' exam scores [25] and spatial visualisation skills significantly correlated with course marks in a multi-institutional study of CS1 students [32]. In the same study, map drawing styles and "measures of richness of articulation of search strategy" (p. 189) were also found to be correlated with higher marks [32]. Of the cognitive abilities that have appeared in the success literature, abstraction was the only one specifically mentioned by the instructors in this study.

What many instructors did mention, and that we almost never explicitly see in the success literature, is a seemingly "magical" notion that there are students who just "get it", or for whom it "just clicks", while there are others for whom it never will, or for whom it will be much more difficult. On the one hand, it stands to reason that this magical ability to "get" computer science cannot be measured, yet it also seems clear that CS educators' desire to understand this "get it" factor goes to the heart of many previous investigations of student success, particularly the aforementioned studies investigating intrinsic qualities such as personality, learning styles and cognitive abilities. At the extreme, a total belief in intrinsic ability would render teaching superfluous, a level of belief expressed by none of this study's interviewees.

Category 3 – Previous Experience

Given the importance of abstraction in computing, it is unsurprising that instructors mentioned students' lack of previous abstraction experience (as distinct from ability) as a factor that negatively impacts success. This seems closely tied to their belief that students' mathematics backgrounds were of relevance, which has been widely investigated and often found to be correlated with success [2, 4, 10, 28, 30, 39, 40]. Instructors also mention background deficiencies in particular kinds of math, such as discrete math or logic, as being problematic. Recent research has made a similar distinction in types of math, showing that students taking a discrete math course and a calculus course performed better than those having taken two calculus courses [28].

Instructors in this study mentioned previous programming experience as an important consideration, discussing the variability in students' backgrounds as a particular challenge to teaching. The role of previous programming experience on success has also been widely researched (e.g., [4, 9, 10]), with mixed results. Some studies have tried to categorise the kind of background experience, either by level of course [33], language paradigm (OO or non-OO) [36] or by assessing experience with specific programming concepts [27]. Instructors in this study made similar distinctions, one acknowledging that "even just a bit of programming background" can be helpful and another suggesting that previous non-OO programming experience may not be helpful for students in an OO class, a finding in line with those reported in [36].

Background experience using computers in general was also mentioned as an advantage for students, particularly general skills such as managing files and opening the editor. However, one instructor also suggested that previous computer experience can be "an aggravating factor that biases [students] very often in the wrong direction" and that they must "unlearn somehow." This lack of a viable mental-model on which to build has been presented in the literature as a significant challenge to applying a constructivist approach to teaching CS [1].

Category 4 – Attitude/Behaviour

While much of the early research on predictors of success tended to focus on students' intrinsic qualities or background experience, more recent work has expanded to include students' attitudes, such as comfort level [4, 37, 39], and behavioural characteristics, for example, approaches to study [16].

Hard work and persistence were identified by our subjects, with one acknowledging that programming "just takes a ton of time," although such factors have only been investigated indirectly (e.g., [2] found that course work grades, which they largely attributed to hard work, were found to be predictors of success in a CS1 course). Being proactive and asking questions is also not directly addressed in the literature, although comfort level, found to be a factor for success in [4, 37, 39], suggests students who are more comfortable might be more likely to ask questions.

Study habits and meta-knowledge, such as the ability to self-assess one's own knowledge or learning style, were mentioned as key factors for success by instructors in the current study. Similarly, students' perceptions of their understanding of a module have been found to be correlated with their programming performance [4], as have "high levels of metacognitive and resource management strategies" [5].

Instructors also mentioned interest in and a general positive attitude toward programming/CS as being important for success. Likewise, the positive expectation of an 'A' in the course was found to be a contributor to success for some groups in [30]. While self-efficacy was found to be significant for non-majors in [38], results have been mixed for CS1 students [39, 40]. Previous success research has also found that a "dislike of programming had [a negative] effect" on success [16]. For instructors in the current study, negative emotions, such as fear (e.g., "Some people are afraid of things" or "afraid of the whole computer"), were mentioned as hindering success. However, the role of fear and other emotions does not appear to have been investigated directly.

Category 5 – Developmental

When reflecting on student success, many instructors naturally considered the influence of their own actions; how they might explain things more clearly, employ appropriate pedagogical techniques, teach to different learning styles or assess students' progress. However, among studies investigating predictors of student success, instructors' roles are rarely directly taken into account. Yet it stands to reason that many of the variables under an instructor's purview during the teaching of a course—such as content coverage, textbook selection, pedagogy, instructor-student rapport, classroom atmosphere and difficulty of assignments—may have a significant impact on many of these studies' variables—including comfort level, self-efficacy, attitude and academic performance. Yet, in the success literature, these factors are largely considered only to be "student attributes". This highlights a likely reason why many of the results in the success literature are contradictory and often inconclusive. It also suggests the need for more multi-institutional studies, such as [32], which help to mitigate the effects of individual instructor's choices and thus may offer more reliable results.

6.2 Teaching and learning theory

As Biggs' teaching levels are based on instructor assumptions about what affects learning success, it is natural to compare our categories to his levels. He has no level corresponding to our *Subject* category. This may be due to his attempt at a universal teaching model that is independent of subject; it may be that computing teachers are more focused on the specific characteristics of their subject. His level one, *what the student is*, naturally contains our *Intrinsic* category, but it also contains our other student-focused categories, *Previous experience* and *Attitude/Behaviour*. Our categories provide a finer-grained view of what students bring to the learning process. Finally, our *Developmental* category corresponds to Biggs' levels two and three, *What the teacher does* and *What the students do*. While we see some quotes that seem to match Biggs' level 2, e.g. (q19) in section 5.1, most of the quotes discuss the teacher's actions in response to particular student needs, and some talk explicitly about providing the context in which learning occurs, as in quotes (q23) and (q24). Our data show a range of *Development* quotes, while the Biggs levels 2 and 3 seem to be quite a distance apart.

As Kansanen's view of the instructional process includes the relationships among the instructor, the course content, and the student, it can represent all of our categories: the categories emphasise different aspects of the subject-student-instructor triangle (Figure 1). All learning involves the subject, the students, the instructor, and the connections among these three. Our *Subject* category puts relatively more emphasis on the subject; our *Intrinsic* and *Previous experience* categories put more emphasis on the student, and *Attitude/Behaviour* puts more emphasis on the student subject connection. Finally, in our *Developmental* category the emphasis is on the link between the instructor and the subject-student link—that is, the instructor tries to affect the way the student interacts with the subject to promote learning. Kansanen sees this relationship as fundamental to the role of the instructor, consistent with both our *Developmental* category and Biggs' level-three teachers.

6.3 Trustworthiness

The trustworthinss of phenomenographic research can be considered for several issues, such as, the selection of interviewees, data analysing process and the significance of the results [43]. Firstly, the 16 interviewees in this study represent CS instructors from six countries in Europe and the US. Both male and female instructors were represented. There was also variation concerning the age and experience of instructors as well as the courses they taught among the interviewees. Therefore, we are confident that our sample covers a sufficiently wide range of relevant characteristics and a rich variety of understanding is possible to obtain. Secondly, our data analysing procedure supports realibility; according to Åkerlind [43], the reliability of phenomenographic results can be enhanced when two researchers independently code interview transcripts and compare their categorisations (coder reliability check) and by reaching agreement on interpretations through discussion (dialogic reliability check). Our analysis procedure was a shared effort and we authors discussed the possible categories several times over the analysis process. Thirdly, there is a question of relevance of the results (communacative validity) and usefullness of the results (pragmatic validity) [43]. The CS instructors perceptions on instructional process is less studied and therefore, we hope to provide a fresh insight that can be used, for example, in improving teacher education and development.

7. CONCLUSIONS

The fact that there is much agreement between beliefs espoused by instructors in the current study, which appear to be grounded largely in those instructors' experiences, and the success factors investigated in the previous research, which are based almost entirely on quantitative measures of student attributes, gives credence to these results. This study suggests that studying instructors rather than students provides an alternative way to explore questions about students and learning, and that phenomenography could be an appropriate methodology to use in such studies.

The current phenomenographic results serve to extend and improve our understanding of instructors' perceptions of student success. They provide context by revealing instructors' conceptions of students beyond CS1; identifying ways in which instructors relate their own experiences to those of the students; highlighting instructors' beliefs about students both as individuals and as cohorts within particular classes or the major; and informing the relationships between what instructors believe are causes of student difficulty and the range of their pedagogical responses.

The emphasis placed on abstraction by these instructors corroborates recent discussions in CS education [17, 22] and highlights its rather thin coverage and discordant results within the existing success literature [3], suggesting this is a particularly fertile area for investigation. Our results also highlight factors that instructors believe are important but that do not necessarily lend themselves to quantitative investigation. Examples include emotions, persistence, hard work, culture and language—also possible foci for future research.

The five different categories in our outcome space highlight the variation in how teachers perceive students' success. The relationships among the categories suggest that the most effective teachers will be those who ascribe to the *Developmental* viewpoint, the category that encompasses and uses all of the other levels of understanding to design instructor-student interactions. The connection between better understanding of student success and more effective instruction means that further investigation into student success factors and their interactions could be valuable for the discipline. This study suggests that qualitative study of instructors might be the best approach for such investigation.

8. ACKNOWLEDGEMENTS

We would like to thank Anders Berglund, Anna Ecerdal, and Arnold Pears from Uppsala Universty, Sweden, for organizing PhICER and therefore making this research paper possible. Thanks to the many discussions with the leaders of the workshop we gained a better understanding of the phenomenographic research approach. We would also like to thank all the PhICER participants for their contributions in data collection and interesting discussions. This work was partially supported by the National Science Foundation through award REC-0633128.

9. REFERENCES

[1] M. Ben-Ari. Constructivism in computer science education. In *SIGCSE '98: Proc. of the twenty-ninth SIGCSE technical symposium on Comp. sci. educ.*, pages 257–261, 1998.

[2] J. Bennedsen and M. Caspersen. An investigation of potential success factors for an introductory model-driven programming course. In *Proc. of the 1st Intl. Computing Education Research Workshop (ICER 2005)*, pages 155–164, 2005.

[3] J. Bennedsen and M. E. Caspersen. Abstraction ability as an indicator of success for learning object-oriented programming? *SIGCSE Bull.*, 38(2):39–43, 2006.

[4] S. Bergin and R. Reilly. Programming: factors that influence success. *SIGCSE Bull.*, 37(1):411–415, 2005.

[5] S. Bergin, R. Reilly, and D. Traynor. Examining the role of self-regulated learning on introductory programming performance. In *ICER '05: Proc. of the 2005 Intl. workshop on Computing education research*, pages 81–86, 2005.

[6] A. Berglund and M. Wiggberg. Students learn CS in different ways: insights from an empirical study. In *ITICSE '06: Proc. of the 11th annual SIGCSE conf. on Innovation and technology in comp. sci. education*, pages 265–269, 2006.

[7] J. Biggs. What the student does: teaching for enhanced learning. *Higher Education Research & Development*, 18(1):57–75, 1999.

[8] C. Bishop-Clark and D. D. Wheeler. The Myers-Briggs personality type and its relationship to computer programming. *Journal of Research on Computing in Education*, 26(2):358–371, Spring 1994.

[9] R. Boyle, J. Carter, and M. Clark. What Makes Them Succeed? Entry, progression and graduation in Computer Science. *Journal of Further and Higher Education*, 26(1):3–18, 2002.

[10] P. Byrne and G. Lyons. The effect of student attributes on success in programming. In *ITiCSE '01: Proc. of the 6th annual conf. on Innovation and technology in comp. sci. education*, pages 49–52, 2001.

[11] S. Dehnadi and R. Bornat. The camel has two humps (working title). Draft paper. Retrieved at http://www.cs.mdx.ac.uk/research/PhDArea/saeed/paper1.pdf, 2/23/07, February 2006.

[12] E. Dijkstra. On the cruelty of really teaching computer science. *Commun. ACM*, 32(12):1398–1404, 1989.

[13] A. Eckerdal and A. Berglund. What does it take to learn 'programming thinking'? In *Proc. of the 1st Intl. Computing Education Research Workshop (ICER 2005)*, pages 135–143, 2005.

[14] A. Eckerdal and M. Thuné. Novice java programmers' conceptions of "object" and "class", and variation theory. In *ITiCSE '05: Proc. of the 10th annual SIGCSE conf. on Innovation and technology in comp. sci. education*, pages 89–93, 2005.

[15] D. Fox. Personal theories of teaching. *Studies in Higher Education*, 8(2):151–163, 1983.

[16] A. Goold and R. Rimmer. Factors affecting performance in first-year computing. *SIGCSE Bull.*, 32(2):39–43, 2000.

[17] O. Hazzan. How Students Attempt to Reduce Abstraction in the Learning of Mathematics and in the Learning of Computer Science. *Computer Science Education*, 13(2):95–122, June 2003.

[18] B. Johnson and L. Christensen. *Educational Research. Quantitative, qualitative, and mixed approaches.* Pearson, Boston, 2004.

[19] P. Kansanen. Teaching as teaching-studying-learning interaction. *Scandinavian Journal of Educational Research*, 43(1):81–89, 1999.

[20] P. Kansanen. Studying – the realistic bridge between instruction and learning. an attempt to a conceptual whole of the teaching-studying-learning process. *Educational Studies*, 29(2/3):221–232, 2003.

[21] P. Kansanen and M. Meri. The didactic relation in the teaching-studying-learning process. In B. Hudson, F. Buchberger, P. Kansanen, and H. Seel, editors, *Didaktik/Fachdidaktik as Science(-s)of the Teaching profession*, pages 107–116. TNTEE Publications, 1999.

[22] J. Kramer. Is abstraction the key to computing? *Commun. of the ACM*, 50(4):36–42, 2007.

[23] R. Lister, A. Berglund, I. Box, C. Cope, A. Pears, C. Avram, M. Bower, A. Carbone, B. Davey, M. de Raadt, B. Doyle, S. Fitzgerald, L. Mannila, C. Kutay, M. Peltomaki, J. Sheard, Simon, K. Sutton, D. Traynor, J. Tutty, and A. Venables. Differing ways that computing academics understand teaching. In *Proc. of the Ninth Australasian Comp. Educ. Conf.*

(ACE2007), 2007.
[24] R. Lister, I. Box, B. Morrison, J. Tenenberg, and D. S. Westbrook. The dimensions of variation in the teaching of data structures. In *ITiCSE '04: Proc. of the 9th annual SIGCSE conf. on Innovation and technology in comp. sci. education*, pages 92–96, 2004.
[25] R. Mancy and N. Reid. Aspects of cognitive style and programming. In E. Dunican and T. Green, editors, *Proc. of the 16th Workshop of the Psycology of Programming Interest Group*, pages 1–9, 2004.
[26] F. Marton and S. Booth. *Learning and Awareness.* Lawrence Erbaum Assoc., Mahwah, NJ, USA, 1997.
[27] L. Murphy, B. Richards, R. McCauley, B. B. Morrison, S. Westbrook, and T. Fossum. Women catch up: gender differences in learning programming concepts. In *SIGCSE '06: Proc. of the 37th SIGCSE technical symposium on Comp. sci. educ.*, pages 17–21, 2006.
[28] B. T. Pioro. Introductory computer programming: gender, major, discrete mathematics, and calculus. *J. Comput. Small Coll.*, 21(5):123–129, 2006.
[29] A. Robins, J. Rountree, and N. Rountree. Learning and teaching programming: A review and discussion. *Computer Science Education*, 13(2):137 – 172, 2003.
[30] N. Rountree, J. Rountree, A. Robins, and R. Hannah. Interacting factors that predict success and failure in a CS1 course. *SIGCSE Bull.*, 36(4):101–104, 2004.
[31] C. Schulte and J. Magenheim. Novices' expectations and prior knowledge of software development: results of a study with high school students. In *Proc. of the 1st Intl. Computing Education Research Workshop (ICER 2005)*, pages 143–153, 2005.
[32] Simon, S. Fincher, A. Robins, B. Baker, I. Box, Q. Cutts, M. de Raadt, P. Haden, J. Hamer, M. Hamilton, R. Lister, M. Petre, K. Sutton, D. Tolhurst, and J. Tutty. Predictors of success in a first programming course. In *ACE '06: Proc. of the 8th Austalian conf. on Comp. educ.*, pages 189–196, 2006.
[33] H. G. Taylor and L. C. Moundfield. Exploration of the relationship between prior computing experience and gender on success in college computer science. *Educational Computing Research*, 11(4):291–306, 1994.
[34] L. Thomas, M. Ratcliffe, J. Woodbury, and E. Jarman. Learning styles and performance in the introductory programming sequence. In *SIGCSE '02: Proc. of the 33rd SIGCSE technical symposium on Comp. sci. educ.*, pages 33–37, 2002.
[35] S. Turkle and S. Papert. Epistemological pluralism and the revaluation of the concrete. *Journal of Mathematical Behavior*, 11:3–33, 1992.
[36] P. Ventura and B. Ramamurthy. Wanted: CS1 students. no experience required. *SIGCSE Bull.*, 36(1):240–244, 2004.
[37] P. R. Ventura. Identifying Predictors of Success for an Objects-First CS1. *Computer Science Education*, 15(3):223–243, 2005.
[38] S. Wiedenbeck. Factors affecting the success of non-majors in learning to program. In *Proc. of the 1st Intl. Computing Education Research Workshop (ICER 2005)*, pages 13–24, 2005.
[39] B. C. Wilson. A study of factors promoting success in computer science including gender differences. *Computer Science Education*, 12(1–2):141–164, 2002.
[40] B. C. Wilson and S. Shrock. Contributing to success in an introductory computer science course: a study of twelve factors. *SIGCSE Bull.*, 33(1):184–188, 2001.
[41] J. M. Wolfe. Perspectives on testing for programming aptitude. In *Proc. of the 1971 26th annual conf.*, pages 268–277, 1971.
[42] M. Xenos, C. Pierrakeas, and P. Pintelas. A survey on student dropout rates and dropout causes concerning the students in the Course of Informatics of the Hellenic Open University. *Computers & Education*, 39:361–194, 2002.
[43] G. S. Åkerlind. Variation and commonality in phenomenographic research methods. *Higher Education Research & Development*, 24(4):321–334, 2005.

APPENDIX. THE INTERVIEW SCRIPT

From instructions, paraphrasing: Discuss several (2-5) concepts (ideas, kinds of problems), but discuss them one by one. That is, go through questions 1 - 11 several times, once for each concept.

1. Mention something (a topic, a concept, a technique, a class of problems etc.) that you think that the students find difficult.
2. In what way does it show to you that the students find X to be difficult? Which signs do you see on this?
3. Why do you think X is difficult for them?
4. Is X difficult for everyone?
   If not, how do those students, who succeed with X handle their learning of X?
5. Which strategies do [or did] you use to handle their difficulties concerning X?
6. How did you think to invent this strategy?
7. Why did you think this strategy would work?
   Did it work?
   Why?/Why not? What do you think?
8. What responsibility would you say that you have to solve this problem?
9. What responsibility lies with the students, would you say?
10. What responsibility would you say that you have to solve this problem?
11. What responsibility lies with the students, would you say?
12. Is there anything else you'd like to tell me about the students' difficulties, in general or any specific problem?
13. Is there anything else you want to say about what you do to help?
14. Is there anything else you want to say about the results of your efforts?