Programmed or Not

A study about programming teachers' beliefs and intentions in relation to curriculum

LENNART ROLANDSSON

Doctoral Thesis
Stockholm, Sweden 2015
This doctoral thesis consists of a synthesis of four papers, a summary in Swedish, and
the following papers:


II Rolandsson, L. (2011). Teachers’ Beliefs Regarding Programming Education. Published in Technology Teachers as Researchers-Philosophical and empirical technology education studies in the Swedish TUFF research school. Inga-Britt Skogh and Marc de Vries (Eds). Sense Publishers: Rotterdam. 2011. (Published here with kind permission.)


Abstract

In the intersection of technology, curriculum and intentions, a specific issue of interest is found in the gap between teachers’ intentions and implementations of curriculum. Instead of approaching curriculum and technology as something *fait accompli*, teachers are considered crucial in the re-discovery of what and how to teach. The thesis depicts the mind-set of teachers and their beliefs in relation to computing curriculum. Three perspectives are covered in the thesis. Based on original documents and interviews with curriculum developers, the enactment of the computing/programming curriculum during the 1970s and 1980s is explored (Paper 1). This historical perspective is supplemented with a perspective from the present day where current teaching practice is explored through teachers’ statements (seminars with associated questionnaires) regarding their beliefs about teaching and learning programming (Paper 2). Finally with a view from a theoretical perspective, teachers’ perception of instruction is discussed in relation to a theoretical framework where their intentions in relation to theoretical and practical aspects of knowledge are revealed (Papers 3 & 4).

The initial incitement to offer computing education during the 1970s was discovered in the recruitment of a broader group of students within the Natural Science Programme and the perception that it would contribute to the development of students’ ability to think logically and learn problem solving skills. Data concerning teachers’ beliefs about teaching and learning programming unravels an instructional dependence among today’s teachers where students’ logical and analytical abilities (even before the courses start) are considered crucial to students’ learning, while teachers question the importance of their pedagogy. The thesis also discover two types of instruction; a large group putting emphasis on the syntax of programming languages, and a smaller group putting emphasis on the students’ experiences of learning concepts of computer science (not necessarily to do with syntax). In summary the thesis depicts an instructional tradition based on teachers’ beliefs where the historical development of the subject sets the framework for the teaching. Directly and indirectly the historical development and related traditions govern what programming teachers in upper secondary school will/are able to present to their students.

From deploying two theoretical approaches, phenomenography and logic of events, upon teacher’s cases it is shown that the intended object of learning (iOoL) is shaped by the teacher’s intentions (e.g., balancing the importance of theory and practice, using different learning strategies, encouraging learning by trial-and-error and fostering collaboration between students for a deeper understanding). The teachers also present a diverse picture regarding what theoretical knowledge students will reach for.

Keywords: computing, programming education, teachers’ beliefs, intentionality, curriculum development, curriculum studies, upper secondary school
Acknowledgements

First and foremost I am deeply grateful to my supervisor, Inga-Britt Skogh, who encouraged and fully supported my research process. She suggested the benefits of working with the teacher community. I am most grateful to my assistant supervisor, Sirkku Männikkö-Barbutiu, for her critical comments and enthusiasm for logic and values when writing for a deeper understanding. I extend my gratitude towards these two supervisors who guided me eloquently through the writing process.

I had the privilege to be part of the ‘Boost for Teachers program’ (Lärarlyftet) initiated by the Swedish government, and later in the research project ‘Theory and Practice in Programming Education’ (T-PIPE). My participation in these was funded by the Swedish government, the Swedish research council, and by the municipality of Nynäshamn. This support is therefore gratefully acknowledged.

The thesis is the outcome of a two phases: Phase 1: The graduate school Technology education for the future (Swedish: Teknikutbildning för framtiden, TUFF) involving coordinators and doctoral students from Stockholm University (SU), University of Gävle (HiG), as well as Royal Institute of Technology (KTH). It became an inspiring environment, with diverse perspectives on how school practices are investigated. Phase 2: At KTH, in the School for Education and Communication in Engineering Science (ECE), in collaboration with the Department of Mathematics and Science Education at SU. My thanks therefore goes to my colleagues at SU, HiG, and KTH. In the final stage of the thesis, Arnold Pears from Uppsala University (UU) and Niall Scery from University of Limerick made an impact and are gratefully acknowledged for their contributions. Finally, I would like to give honour to my colleagues at UU: Anna Eckerdal, Anders Berglund, and Michael Thuné, who gently introduced me to computing education research. In summary, this was a beneficial cross-disciplinary setting in its truest sense, with influences from many universities, research disciplines, and three research domains: engineering education research, computing education research, and education research.

A special thanks goes to my mother and father, who encouraged me to search beyond what seems obvious. Finally, and most of all, I would like to thank my beloved family, who has shown patience and confidence throughout these years.
Preface

We are facing a time where coding and algorithmic thinking has to be implemented in school on the same level as reading, writing, and arithmetic (Rushkoff and Purvis, 2011). In China, every child learns computer programming, compared to fewer than five percent in the U.S. So why shall we learn to code? Some say it’s all about political incentives, and some say it is a democratic right fostering pupils and students to learn about problem-solving, design thinking or systems thinking, digital confidence, and an understanding of the technical world.

Whether that is all, or if future technological innovations and democratic progression will bring other important skills to be taught, we cannot tell. But, for educational purposes, we can tell the deliberate mission of the teachers’ work - transforming and shaping curriculum - is something deserving attention, as otherwise a majority of pupils will be excluded. This is an important mission in all pedagogical activity, as content, society, and technology change in parallel. Or are we seeking the impossible?

Some months ago, I was invited to a Swedish publisher to experience in real-time a glint of what matters for such a curriculum change. The meeting became overwhelming as I felt the power of influencing what would be taught in school. I described to the publisher the significance of delivering content that describes human thinking ahead of IT, and why a computing machine does not always do what you intend. I presented the historical perspective pinpointing the fact that programming knowledge has been around for at least 100 years, and why we have to focus on human thinking instead of the language syntax that digital computers can “understand”. The editor was confused and could not visualize how the historical perspective on thinking would be profitable, as such a book would not fit today’s teaching and learning: repetition and routinized tasks in a programmed manner. I became confused as I thought I was right and she was wrong. But, in a split second I realised the difficulty of change, and some of the reasons for why teachers do things out of order.

Stockholm, 2015-02-15
Lennart Rolandsson
# Contents

Acknowledgements \hspace{1cm} v

Preface \hspace{1cm} vii

Contents \hspace{1cm} viii

1 Introduction 1
1.1 About Computing and Education \hspace{1cm} 1
1.2 Concepts in Use \hspace{1cm} 5
1.3 The Swedish School System \hspace{1cm} 6
1.4 Purpose \hspace{1cm} 7
1.5 Thesis Outline \hspace{1cm} 7

2 Theoretical Framework 9
2.1 Curriculum According to the Thesis \hspace{1cm} 10
2.2 Intention \hspace{1cm} 11
2.3 Intentionality \hspace{1cm} 12

3 Related Work 13
3.1 Curriculum Studies and Computing \hspace{1cm} 13
3.2 Beliefs and Worldviews \hspace{1cm} 14
3.3 Teachers’ Instruction \hspace{1cm} 17

4 Research Methodology 21
4.1 Study 1: The Curriculum Perspective \hspace{1cm} 23
4.2 Study 2: The Teacher Perspective \hspace{1cm} 24
4.3 Study 3: The Theoretical Perspective \hspace{1cm} 25
4.4 Trustworthiness \hspace{1cm} 27
4.5 Ethical Considerations \hspace{1cm} 30

5 Summary of Papers 33
5.1 Study 1: Programming in School \hspace{1cm} 33
5.2 Study 2: Teachers’ Beliefs Regarding Programming Education \hspace{1cm} 35
Chapter 1

Introduction

The background for what will follow, in the thesis, is found in my own practice as a teacher. In 2003, I searched nationwide for a joint project to collaborate with other programming teachers. The purpose was to experience the benefits of learning by somebody else’s needs, in a manner resembling the engineering approach. I intended to create a setting where students experienced the joy of delivering code that could be read and used by another student. This intention was not fulfilled as expected. It proved extremely hard to find other teachers prepared to work in such a setting. I searched for any particular explanation for this issue. Finally it dawned on me: something obvious to me is not always easily manageable for another teacher, as we each have our own strengths and priorities. This became the starting point for the thesis.

1.1 About Computing and Education

Over the years, computing machines have transformed our way of thinking about ourselves and what is possible with calculation, design, simulation, gaming, administration, etc. During the second World War, other computers - human computers (usually women) - managed algorithms in a skillful way, predicting the trajectories of ballistic projectiles. As could be imagined, they operated under heavy pressure leading to some failures, and pushing for new procedures and technological innovation.\(^1\) In the 60 years since, we have witnessed the evolution of digital computers and programming languages for dynamic interaction with computing machines. During this period, a multitude of technological innovations have emerged and continue to affect what and how computing is delivered as a subject in school (Micheuz, 2006; Syslo and Kwiatkowska, 2008).

These days the question of teaching computing in schools is often raised, with the main argument stating the outcome is not satisfactory (Furber, 2012) as the

\(^1\)This is an amazing story that should be taught in schools; the evolution from human computers to analogue computers to digital computers.
focus is mainly on basic digital literacy skills while the underlying principles are left uncovered. Besides computers, computer applications and IT in general become increasingly a natural part of everyday work, even in schools. In light of the growing need for people knowledgeable in computing and digital literacy, this situation could become problematic to individuals and society. Politicians, newspapers, and media reflect the significance of teaching computing instead of mainly IT or ICT. They suggest teachers should teach programming. The computer's technological heritage is pervasive in society, and voices are heard that pinpoint the necessity of offering computing and programming knowledge in the same way that chemistry or mathematics already exist in schools.\(^2\)

Some even state that learning to code is as important today as reading, writing, and counting were when compulsory schooling was introduced in the mid-1800s. This is not the first time similar concerns are raised for computing literacy in school (Rolandsson and Skogh, 2014). The outcome of such initiative expose a non-sufficient solution, as the subject matter easily becomes selective by nature, thus not suitable for the majority of pupils. The drop-off rate is high, even at the university level (Kurland et al., 1989; Robins et al., 2003). Offering programming in school accordingly has to be transformed into something more appropriate. An important target group in such a project is the teachers, holding epistemic attitudes and unknown reasons for their actions (until now, in this thesis).

The strategy of focusing primarily on general principles and concepts from computer science (CS) has become common and successful in education (e.g., Bell et al., 2011, 2012; Thies and Vahrenhold, 2012). The idea that concepts and principles do not need to be taught using digital computers as a necessity for learning has opened up new ways of teaching and learning computer programming that are expected to include the majority of students. As such transformation is considered necessary if computing (with CS) will be taught to all, we need to study the subject matter from a teacher’s perspective in contrast to recycling old lesson plans. In that tension between “hard core” programming and principles/concepts from CS, the thesis positions itself for a discussion about practical and theoretical matters in relation to curriculum from the teachers’ perspective.

The issue of programming in education (Griffiths and Tagg, 1985; Svensson, 1985; Turski, 1973) has been (and still is) debated vigorously. It has often been described as a too-demanding subject to learn. From time to time, the notion that only a minority of the students are able to understand a programming language for successful interaction has been put forward by both teachers and scholars (Björk et al., 1975; Luerhmann, 1980; Papert, 1980). Questions such as “should programming be taught to everybody?” and “should programming be taught in schools at all?” have been raised both prior to and during the period highlighted in this thesis. Over the years, computer programming evidently has been diffused and transformed into the school curriculum it is today.

---

\(^2\)E.g., the EU commissioner Neelie Kroes, (2014).
1.1. ABOUT COMPUTING AND EDUCATION

Teaching Programming

Educational computer technology aimed at enhancing learning has existed since the 1960s in different guises (e.g., Kollerbaur, 2005; Murray, 1983; Thomas and Kobayashi, 1987). However, many tend to forget the heritage of why it is taught at all. Today’s computing technology stands on earlier innovations, and lately during the 20th century, on the formalization made by Church and Turing (Watson, 2012), which is crucial for how we interact with computers.

For a dynamic use of computational thinking, the implementation of analogue computers became the initial step which eventually implied the construction of digital machines (abstract machines) with a multitude of different programming languages. As an example, the concept of machine is still in use, explaining how, for instance, Java language benefits from the “virtual machine” under the hood of the development environment in use when writing code.

The history of programming offers a diverse picture, favoring a multitude of different solutions and development environments (Bergin and Gibson, 1996). They all share ambitions to offer easy access to languages and efficient management of concepts within CS and/or informatics. Today, computer applications and cloud-services are made available at an evenly increasing pace. Therefore, today’s teachers in computer programming education are very fortunate, as they can benefit considerably from what hardware, programming paradigms, and development environments offer the learning process. Specific programming languages (e.g., Python, Pascal and Basic) and environments (e.g., Alice, BlueJ, Arduino and Raspberry Pi) are developed for educational purposes to lower the threshold of learning computer programming. However, there is research indicating (Gries, 2006; Linn and Clancy, 1992; McGettrick et al., 2005; Sloane and Linn, 1988) that new pedagogical ideas, the development of new educational environments, new educational programming languages, and the introduction of new informatics curricula have had little or no influence on the instructional pattern (content and work methods) offered in programming education in upper secondary school. The rationales for such a slow change could be in the teachers’ thinking about pedagogy (Booth, 2001; Lister, 2005, 2008; Tenenberg and Fincher, 2007).

Learning Programming

The process of learning programming - going from novice to professional - is considered by some to be a lifelong adventure (Kurland et al., 1989; Winslow, 1989), as you have to develop your ability to express yourself in code as distinctly as possible to make the computer interpret your intentions correctly. It is an unusual situation far from interacting and communicating with peers, and therefore demanding as you have to adjust your communication practices with precision in code for correct functioning by the computer.

---

3The stories about the early innovators e.g., Scheutz (Lindgren and McKay, 1987) and Babbage (Watson, 2012).
Kurland et al. (1989) show that students with two years of programming instruction commonly have not yet reached what would be called a deeper understanding; “Many students had only a rudimentary understanding of programming”. Winslow concludes that “[o]ne wonders [...] about teaching sophisticated material to CS1 students when study after study has shown that they do not understand basic loops [...]” (Winslow, 1996, p.21, in Robins et al., 2003).

A more optimistic view is expressed by Guzdial (2004) who claims that since computer programming has evolved considerably, and gradually diffused into different domains and gadgets, the problems in teaching and learning could be expected to diminish (diSessa, 2001). There are alternatives (Bell et al., 2011, 2012; Resnick and Klopfer, 1989) that emphasise thinking more than knowledge in programming skills, e.g., in embodiment, logical thinking and interplay, where focus is more on problem-solving in relation to “unplugged settings” (e.g., sorting and parity check).

**All Students can Learn Programming**

School is constantly transforming as new philosophical imperatives and political educational decisions shape our thinking about how education should or could be practiced. Resnick (2010) elaborates on what school is able to offer “through social processes that include participation in certain forms of high-demand learning” (Resnick, 2010, p.186) instead of as an “entity” or as something that people have a fixed amount of time to accomplish (Dweck and Molden, 2005; Resnick and Gall, 1997). In the 1990s “the public agenda of raising educational levels for all has been promoted under the banner of the standards movement, often accompanied by the phrase ‘All children can learn’”(Resnick, 2010, p.184). Resnick suggests a new way of school organization where teacher instruction and professional development should aim for and secure a “thinking curriculum” that has high cognitive demands on students’ and teachers’ conceptual learning, reasoning, explaining, and problem solving (Nisheta, 1993; Resnick and Klopfer, 1989).

**Rationales for Programming in Curriculum**

In criticism of democracies which serve economic and vocational purposes rather than general social and political ends, Carr (1998) pinpoints the importance of doing careful considerations when developing curriculum. If not reflected upon, we are in a fragile position where policy documents could be “reduced to a mundane technical expertise in which non-technical, non-expert questions about the social and political role of the curriculum are not even asked” (p.337). This is an important objection these days, as teachers’ professionalism demands something more than solely a “scripted activity where teachers are expected to deliver a prescribed curriculum under strong state guidelines” (Tatto, 2006, p.237). The question of how we interpret and manage curriculum is in focus.

For many years, computer programming has been perceived as school’s “new Latin”, with intention to foster problem-solving skills (Linn, 1985; Palumbo, 1990;
1.2. CONCEPTS IN USE

Reed and Palumbo, 1991), thinking skills, and understanding of computers (Urban-Lurain and Weinshank, 2011). Skills that is beneficial in an global economy, and therefore offered in the school system (Tatto, 2006). However, the transferability of skills and knowledge - between problem-solving within computer programming and problem-solving in other school subjects - has been questioned (Clements and Gullo, 1984; Dalbey and Linn, 1985; Griffiths and Tagg, 1985; Linn, 1985; Palumbo, 1990; Pea and Kurland, 1984; Reed and Palumbo, 1991; Soloway, 1993; Svensson, 1985; Turski, 1973). Research indicates that students’ understanding of computer programming concepts often remain firm and barely change (Kurland et al., 1989; Mayer et al., 1986). The understanding is not deep enough. Syntax and semantics in computer programming demands an abundance of experience before they can be used for problem-solving purposes (Urban-Lurain and Weinshank, 2011). In fact, Resnick (2009) even states that computer programming languages are too difficult to use in education. According to Resnick (ibid.), programming languages are either introduced in such a way that they do not appeal to students’ experiences or in a tutoring context where teachers fail to provide both sufficient guidance and appropriate challenges. In other words, teachers’ intentions for instruction are questioned.

1.2 Concepts in Use

Informatics or Computer Science

School systems worldwide embrace the content of computer programming differently. It is offered under labels such as “Computing”, “Computer science”, or “Informatics” (Dagiene, 2006). However the computer science education research community seems to make no distinction between the terms “Computer science” and “Informatics” (Saeli et al., 2011). During the 1970s, 1980s, and 1990s, the Swedish informatics curriculum was labelled differently depending on context, educational level, and decade; “Datolära”, “Datateknik”, “Datakunskap”, “Datorkunskap”, and “ADB”, which in English translates as “Computing”, “Information technology”, “Information knowledge”, “Computer knowledge”, and “Automatic Data Processing” (ADP). The Swedish word “Data” is obscure, as it could nowadays be misunderstood as “the computer” and/or “the information in the computer”. However, history and the original meaning, which correlates to the international discourse, infer that “information” is the most appropriate translation. The use of the word “Computing” is more modern as it could embrace ICT, digital fluency, computational thinking, and programming (Department for Education, 2013). In the thesis “Computing” and “Informatics” is used to denote the subject, as the same subject translates, in summary of Swedish, to “Information technology” or IT.

The historical part of the inquiry reveals that the first curriculum developers in the early 1970s drew on the international discourses of informatics.
CHAPTER 1. INTRODUCTION

Instruction and Instructional Design

Instruction and instructional design elaborates on didactic issues like “what to teach” and “how to teach”. The concept of instructional design could be traced back to teaching machines in instructional settings and programmed instruction (Seel and Dijkstra, 2008). Today the concepts of instruction and instructional design are used among constructivists as well as among behaviorists (Mayer, 1999). The word “instruct” comes from the Latin word “instruere”: to set up, furnish, kit out, and teach. The thesis adopts the concept of instruction in a way similar to Brockenbrough (1993). He adds the prefix in to stuere5 which could be interpreted as “to build within”. The conclusion of the construction elicits the fact that instruction cannot occur unless it occurs inside people’s heads and that no amount of external activity or material can substitute for the existential fact that knowledge must be (re)created by each individual. (Brockenbrough, 1993, p.184).

1.3 The Swedish School System

To offer an insight into the educational context of the inquiry, a brief presentation of the Swedish school system and the early introduction of the subject matter of informatics will be given in the following section. The Swedish upper secondary school has undergone a series of changes during the past 40 years (Lindensjö and Lundgren, 2000). Three different curricula (Lgy70, Lp94 and GY2000) for gymnasium (upper secondary school) have passed, and today a fourth curriculum (GY-11) is on its way. The basic structure follows the same principles now as it did then; compulsory school (nine years) followed by two, three, or four years of upper secondary school. However, disruption emerged during the 1970s in the monolithic Natural Science Programme (NSP). The programme experienced a decline in student numbers after the introduction of the new curriculum.6 Some believed NSP was too theoretical, wherefore the Swedish National Board of Education (NBE) suggested a computing alignment within the NSP in 1976. The disruption is worth mentioning, as vocational education in ADP and natural sciences became an issue in the enactment of the informatics curriculum.7 The current design of upper secondary school was established during the 1970s. The organisation of upper secondary school education was designed to facilitate the separation of students aiming for higher education from students not aiming for higher education. In conjunction with this ‘main system’, special higher courses were offered as one-year extensions for those with a specific profession in mind. University studies

5Word from Latin which translates to place up together.
6Lgy70 (Skolverstyrelsen, 1971).
7The incorporation of computing into Natural Sciences is elaborated on further by Denning (2007).
1.4. **PURPOSE**

In a research review by Beswick (2007), mismatches between teachers’ beliefs and their practices were discovered. Marton (1994) describes the issue as teachers “lack an explicit and generalizable awareness of the relationships between means and ends in teaching” (p. 39), as Entwistle and Entwistle (1992) suggests that it is impossible to ‘de-understand’ what once had been understood. Besides, as most teachers are supposedly already familiar, in contrast to their students, to the content, a gap could be expected between the intended and the implemented curriculum. In the thesis a micro-macro perspective is used in the three following studies, to describe some of the characteristics of that issue, or gap, and unravel some of the rationales for these characteristics. The gap is described more in detail in the *Theoretical framework*.

1. Study 1 (Paper 1): The curriculum perspective, with a case from the 1970s and 1980s describing the Swedish curriculum development, addresses the question: *On what grounds were programming education in Sweden implemented and what lessons could be learned from this?*

2. Study 2 (Paper 2): The teacher perspective, where Swedish upper secondary teachers’ beliefs about programming and educational constraints are focused on with the question: *What beliefs do programming teachers express regarding teaching and learning computer programming in upper secondary school?*

3. Study 3 (Papers 3 and 4): The theoretical perspective; exploring teachers’ perception of teaching and learning. The perspective embraces two papers. Paper 3 is addresses the question: *The intentions invested in the object of learning - what is revealed in the complementarity of two theoretical approaches?* Paper 4 addresses the question: *What educational intentions and expectations do programming teachers express when they (in retrospect) describe their teaching during a practical assignment focusing on a principle from computer science?*

---

*The TEP has been through two revisions since 2000 to raise the numbers of students in tertiary studies in the field of technology. A fourth year was re-introduced as an experiment offering different alignments for those interested in a professional career instead of further studies.*

---

...
1.5 Thesis Outline

The thesis consists of four papers and a discussion on the main aspects of the studies. For those who are interested in the preceding work, I refer to the following papers.

1. Programming in School: Look back to move forward.
2. Teachers’ Beliefs Regarding Programming Education.
3. Bridging a Gap - In search of an analytical tool capturing teachers’ perceptions of their own teaching.
4. Intentions and Pedagogical Actions - A study of programming teachers’ construction of a learning objective.

The thesis covers a summary and a synthesis of these papers. Below is a short description of each chapter and the appendices.

The first chapter is an introduction to the theme of the thesis. The aim is to provide a platform to appreciate the issue brought forward by the thesis.

The second chapter shows a theoretical framework of perspectives.

The third chapter presents related work to position the thesis. Teachers’ instructions in relation to computing are in focus, and teachers’ beliefs and intentions are discussed.

The fourth chapter presents the methodology used in the thesis. Considerations and ethics are addressed.

The fifth chapter gives a summary of findings presented in the three studies.

The sixth chapter a discussion and synthesis of the studies is presented, followed by two issues found in the thesis and a message for future curriculum development. Further studies is suggested.

The seventh chapter presents a Swedish summary of the thesis.
Chapter 2

Theoretical Framework

In research, a discrepancy has been found between teachers' classroom practice and teachers' beliefs expressed beforehand (e.g., Ertmer, 2005; Fang, 1996; Schraw and Olafson, 2002), which I denote as a gap; experienced by the teachers, but not necessarily accessible through deliberate acts of reflection.\(^1\) For the work of this thesis I consider the gap, between what is intended and what is implemented, as significant for discussion about teachers' intentionality. In need of clarification, the following description is offered to exemplify how a gap could be materialized in education:

*A teacher assesses the student’s actions and achievements. Based on what the teacher observes, s/he responds in terms of the presumed cognition by the student. But, in that process of assessment, the intersection of the teacher’s intentions and the student’s actions do not necessarily correlate. In a teacher’s perspective, a gap is perceived, between the intended and the implemented curriculum that needs to be bridged.*

This is a common scenario in many teacher-student interactions: The teacher empowers the student’s understanding of a presented knowledge, as the teacher work (un)consciously to reduce the difference (the gap) between what s/he intends for student’s learning and what s/he perceives that the student has discerned. In order to pinpoint teacher’s perceptions of students’ actions, with implications for the teacher’s own action (commonly denoted as intentionality) an interpretivist approach has been applied in the thesis.

The conceptualisation of intentionality could, according to Noel (1993), be described as a spectrum:

At one side of the spectrum, scholars consider individuals as having mental states, usually called beliefs and desires, that make up the__________________________

\(^1\)Teachers can have a lot of teaching experiences, and therefore be able to bridge that gap by intuition.
reasons for the individual’s actions [...] At the other end of the spectrum [...] the intentionality of actions comes through linguistic or syntactic relations between the internal states.” (p.124)

Two sides in opposition with implications for what could be ascribed to one or many mental state(s); the former commonly ascribing meaning, as the latter commonly ascribing a cognitivist view for observation of the individual person (Noel, 1993). The thesis is more towards the first side of the spectrum, as it investigates teachers’ beliefs and intentions.

Phenomenography was considered a first choice for the thesis theoretical framework. Unfortunately, most research with a phenomenographical approach does not problematize what the teacher contributes, except that teachers’ contribution to the object of learning is supposed to exist (Häggström, 2008; Marton, 2014; Pang et al., 2006). Besides, commonly phenomenographic research investigates the phenomenon from the students’ perspective, even in literature that describes the intended object of learning, e.g., Pang et al. (2006). In search for ways of describing teachers’ thinking in relation to curriculum, the phenomenological heritage in phenomenography was searched for, to unravel what the teachers’ intentions contributes to the intended object of learning (See Study 3).

Based on the idea that there is a relation between the teacher’s intentionality (leading into teacher’s action) and the object of learning, it was considered necessary to apply the Logic of events theory (von Wright, 1983) in an attempt to unravel the intentions among the individual teachers. In other words, the thesis presume that teachers are holding intentions for students’ directedness/aboutness of the object of learning.

A brief description of curriculum according to the thesis, intentions, and intentionality is presented in the following sections.

2.1 Curriculum According to the Thesis

In contrast to a syllabus described by a document and in need of regular revisions approximately every tenth year, curriculum is considered a conceptualisation made with a multitude of interests in society. In the thesis, it is considered a dynamic entity which is transformed, enacted, and experienced in a continuous process by different parties (Goodlad, 1994; Goodson, 1993; Linde, 1993). As this thesis reveals, the existence of a curriculum where teachers are heavily involved in the development process, it is considered a social construction dependent on the interplay between society’s need and the actual classroom outcome (Connelly and Xu, 2010; Goodlad, 1994; van den Akker, 2003).

The intersection of curriculum and today’s teachers’ beliefs/intentions is a way of studying the curriculum as a continuous process from a macro- and a micro-level, where determinants could be discovered (Linde, 1993). Such an research approach makes it somewhat unusual as curriculum is biased towards the teacher. For the purpose of the thesis, such a bias is reasonable as the issue focus the
teachers’ thinking in relation to the gap, between the intended and the implemented curriculum. A curriculum distinction made by van den Akker (2003) where he differentiates the curriculum into practical representations for what different agents have in mind; intended (curriculum developers), implemented (by teachers), and attained curriculum (by students).

2.2 Intention

In an attempt to escape the logical positivism and the endless debate about body-mind issues, von Wright (1983; 1998) takes a stand to find reasons for our actions beyond a deterministic way of thinking. Instead of falling into behaviorism or phenomenalism, von Wright argues throughout his entire work for a parallelism (Curley, 1985) where the mental and the physical phenomena occur, without causal interaction between them. An interaction he named "psycho-physical parallelism" (von Wright, 1998). Such a stance has something of value as we reach out to the mental world (in this case, teachers’ intentions underpinning their behavior) to understand the process of the physical world (teaching and teachers’ intentionality).

Regardless of such an ambition, he identifies the “determinants of intentions”: wants, duties, abilities, and opportunities/possibilities, four determinants that shape our beliefs and attitudes without being too deterministic. In von Wright’s (1998) theoretical work he consider the determinants to change in apperance: as your role changes, you attend education becoming able, and you appropriate technology in new ways. In von Wright’s own wording he put his compatibilism (non-determinism) in this way:

In attributing reasons for action to an agent we normally also attribute to him various abilities, beliefs, desires and inclinations, the understanding of institutions and practices of the community, and other things which characterize him as a person. Some of these features may date far back in his life history. They constitute a kind of background or ‘program’ which has to be assumed if certain things he did or which happened to him shall count as reasons for subsequent action [...] These other things, then, speaking metaphorically, are ‘inputs’ playing on the ‘keyboard’ of his programmed personality. His action is the ‘output’. (von Wright, 1998, p.27).

In other words, the reasons for the agent’s action are difficult to extract, as the prevalent situation has an impact on her/his “programmed personality”. You therefore need to be aware of the limitations brought by a first-order perspective (the agent) or a second-order perspective (researcher) for an understanding of the rationales for the actions. Aware of such considerations between the “output” and the “input” the thesis will tell something about the teachers’ intentions.

The systematic structure of the model, built around the determinants of intentions identified by von Wright (1983) and Skogh (2013), allows access to information
not only about individuals’ perceptions of actions taken, but also about her/his reasons for these actions-information that would otherwise remain undiscovered and unused. Such a stance interpreting the reasons for actions increases awareness of the individual but also of the individual as part of a larger context—of the complexity that surrounds every (teaching) situation. Insight into this complexity constitutes a valuable basis for the development of educational practice.

2.3 Intentionality

Brentano distinguish between the psychological and non-psychological phenomena, as he re-introduced the principle of intentionality (Moran, 1996). Stanford Encyclopedia of Philosophy states that intentionality has nothing to do with the implicit knowledge of a subject, as it denotes “the power of minds to be about, to represent, or to stand for, things, properties, and states of affairs” (Jacob, 2014).

In an attempt to position von Wright’s philosophy in the spectrum of different opinions about intentionality, without going into the philosophy of language or the philosophy of mind, he is mostly concerned about how our actions disclose the intentions. In contrast to “behavioral explanations” he suggests “intentionalist explanations” as reasons for our action, with implications for how we can reflect upon human actions in terms of intentions (von Wright, 1983). Explanations that pinpoint the fact that we are actually free to transform our worldviews, but still determined by reasons for our actions. Husserl and many of his followers (Spiegelberg and Schuhmann, 1982) describe an awareness of our experiences through our senses (Smith and McIntyre, 1982), as von Wright argues that our experiences are consequences of reasons and motivation for or against a certain action, instead of being “governed by ‘iron laws’ of causal necessitation” (von Wright, 1998, p.3). von Wright describes it as

[...] intentionality is not anything ‘behind’ or ‘outside’ the behavior. It is not a mental act or characteristic experience accompanying it [...] it suggests a ‘location’ of the intention, a confinement of it to a definite item of behavior, as though one could discover the intentionality from a study of the movements. One could say—but this too might be misleading—that the behavior’s intentionality is its place in a story about the agent. (von Wright, 1971, p.115).

---

2Intentionality corresponds to the Latin verb *intendere*, interpreted as being directed towards some goal or thing. Sometimes words like aboutness or directedness are used synonymously with intentionality.

3Phenomenography and variation theory relate to the thoughts of Brentano (Marton and Booth, 1997; Pang, 2003). However, in phenomenography the concept is somewhat stretched (Harris, 2011). In comparison between the worldviews of a phenomenologist and the suggested interpretation of von Wright, it appears as if the different approaches relate to different kinds of intentionality.
Chapter 3

Related Work

This chapter presents work related to the purpose of the thesis. As curriculum development is considered a continuous process, a brief discussion about significant matters is presented from curriculum studies, teachers’ beliefs, the history of computing education and teachers’ instruction.

3.1 Curriculum Studies and Computing

Curriculum studies emphasise that, curriculum in comparison over time, is an important study object expressing “whether existing patterns of cultural, economic, and political life will be reproduced or transformed” (Carr, 1998, p.326). As a matter of fact, progress of democracy is closely related to curriculum development, as it incorporates “both a record of its past and a message for its future” (Carr, 1998, p.324). In a society totally dependent on digital computer technology, this becomes an interesting issue for teachers as technology continuously evolves, putting new demands and constraints on teachers’ ability to instruct properly with the purposes of students’ empowerment and understanding (Benade, 2015).

According to Pinar (1995), politicians and ministries of education commonly commission educational agencies and schools to offer education about computer technology. In this case, implicit values in technology diminish in favour of the process, the packaging, and presentation of the content, instead of focusing on the individuality of the learner or the applicability of content for school (Cuban, 1986, 2001). For many years, therefore, curriculum development has adopted a classical or technical-professional perspective instead of understanding what shapes and constrains it (Goodlad, 1994; van den Akker, 2003). In addressing these issues, especially in the formation of something never taught before, the thesis adapts a socio-political perspective on curriculum (Goodlad, 1994), including learning objectives, teaching methods, assessment procedures, and classroom organisation (Carr, 1998). A definition of curriculum that pinpoints the necessity to include the teacher as well as the students for an understanding of what is possible to
accomplish in classrooms. Besides, with a socio-political component in curriculum we become aware of “the important social and political role that curriculum plays in initiating pupils into the culture, practices and social relationships of their society” (Carr, 1998, p. 325)

According to Carr’s (1998) description of different co-existent ideologies for democratic progress, the development of Swedish curriculum could be described as a consequence of three ideologies working in parallel: the classical-humanists, the liberal-progressive, and the modernist-vocational. The first has roots in the pre-industrial society with the purpose of preparing an “intellectual elite for the task of preserving their society’s cultural heritage” (1998, p.327), the second has a political task offering rational autonomy and individual freedom, and the third has an economical task offering mass schooling.

Informatics education/curricula obviously vary from country to country regarding content and ambitions (Dagiene, 2005, 2006; Hubwieser et al., 2011; Micheuz, 2006). But, as we are living in a world built on ideologies pushing for a democratic progress, it could be expected that all pupils/students need to attend courses in computing. Research from the domain of computing history consequently reveals some commonality between different countries; the UK (Woollard, 2005), Austria (Micheuz, 2005), Ukraine (Spirin, 2005), Lithuania (Dagiene, 2005), Poland (Sysio and Kwiatkowska, 2005), and Sweden (Rolandsson, 2011). In many of these studies, the implementation of curriculum development in schools is described in relation to investments in hardware and software.

Some studies elicit the fact that computer technology raises considerable problems for instruction and pedagogy, as technological optimism commonly entails the innovation of information and communication technology (ICT) (Cuban, 1986, 2001; Donaldson and Knupfer, 2001; Pedersen, 2001; Pelgrum, 2001; Segal, 1996; Tapscott, 1998). In the field of education, ICT is believed to offer the solution to a number of pedagogical problems (Segal, 1996 in Karlsohn, 2009, p.353) similar to the optimism entailing the innovation of educational technology in the 1960s. Pedersen and Cuban (Cuban, 2001; Pedersen, 2001) raise a question about the implicit technological determinism within informatics education.

The tension between the technicalities within computing and the didactics has been a long-lasting process with many backdrops (Donaldson and Knupfer, 2001; Pelgrum, 2001). Implicit technicalities in computer technology commonly dominate instructional design, wherefore Cuban underlines the necessity to discuss and elaborate on methodological questions (Cuban, 1986, 2001).

3.2 Beliefs and Worldviews

Thompson (1992) distinguishes between beliefs and knowledge, where the concept of belief holds specific features:

1. Beliefs are held with varying degrees of conviction, and
2. There is no consensus as there is no need to satisfy a truth condition.

The concept of knowledge is attained in consensus within a cultural context holding a specific belief (Abelson, 1979; Ernest, 1991).\(^1\) It should be noted that beliefs sometimes move to the status position of knowledge, and vice versa. Teachers’ epistemology in relation to their daily work and subjects they teach is commonly labelled as epistemological beliefs or personal epistemology.\(^2\)

Beliefs-oriented research can be traced back to the 1920s, but it was not until the 1970s—when it was fueled by a shift in paradigms from a focus on teachers’ behaviours to a focus on teachers’ thinking and decision-making processes (Thompson, 1992)—that it became widespread among education scholars.

**Instructional Patterns**

Bruner (1996) and Lister (2008) emphasise the importance of teachers liberating themselves, from “folk” pedagogy (Bruner, 1996; Lister, 2008), where content is commonly taught based on intuition and personal educational experiences. In general, folk pedagogy and educational beliefs are seen as resistant to external influences and accordingly are very difficult to change (Kagan, 1992; Luft and Roehrig, 2007; Olson, 1981; Yerrick et al., 1997), as they depend on the individuals’ personal growth and ability to reflect upon and understand her/his own teaching practice (Baird et al., 1991; Brookfield, 1995; Schön, 2003).

If a group of teachers show similar patterns in knowledge and beliefs, it is of importance to examine the implications of these patterns. The underpinnings of such patterns could be the reason for the difficulties in learning what computer programming offers. In that case, the implicit constraints in education depend on the messengers (teachers) as well as on the content itself (Thompson, 1984).

In studies by Pajares (1992) and Schraw and Olafson (2002), the transformation of knowledge in classrooms is scrutinised as dependent on teachers’ epistemological beliefs in relation to teaching computer programming. The concept of belief is described using two different dimensions: beliefs about classroom practices and ontological beliefs.

Our beliefs influence our understanding of the world (Abelson, 1986; Alexander and Dochy, 1995). Accordingly, teachers’ beliefs will influence their perception of learning environments, instructional materials, and different instructional approaches available to them (Alexander and Dochy, 1995; Bungum, 2003; Calderhead, 1996). Kagan postulated, “[...] most of a teacher’s professional knowledge can be regarded more accurately as a belief” (Kagan, 1992, p.73). Thompson (1984) argues for the importance of teachers’ conception (their beliefs, views, and preferences) in research about education, as it constitutes a primary mediator

\(^1\)Ernest (1991) dissolves the distinction between knowledge and beliefs as he re-labels the two concepts in relation to the social aspect: individual constructions (subjective knowledge) or social constructions (objective knowledge).

\(^2\)For an explanation of the origin and flavours of personal epistemology, I refer to Hofer (2001).
between the subject and the learners. This is supported by research – for instance, in mathematics – when studying teachers’ interpretation and implementation of mathematics curricula (Clark and Peterson, 1986; Romberg and Carpenter, 1986; Thompson, 1984).

Beliefs and Teachers’ Knowledge

In an overview by Pajares (1992) it is obvious that research concerning teachers’ epistemological beliefs is perceived as “a messy construct”. The messiness is partly due to the fact that different concepts, like construct, personal theories, attitudes, beliefs, and knowledge (Hashweh, 2005; Kagan, 1992; Pajares, 1992), are used interchangeably among researchers.

Alexander et al. (1995) highlight the differences between peoples’ beliefs and knowledge, linking it to the number of years in education. The outcome of their research showed that people use the words belief and knowledge interchangeably and they could therefore be perceived as the same.

Are knowledge and beliefs, in actuality, synonyms marking the same semantic territory, or are they antonyms denoting orthogonal dimensions of human understanding? Or, is it possible that the concepts of knowing and believing share a common ground, while still retaining some unique and unshared terrain? (Alexander and Dochy, 1995, p.415)

Belief Systems and Classroom Practices

The need to cluster different beliefs into systems is linked to the discovery that the same individual can hold contradictory beliefs (Leatham, 2006). A belief system could be described as a metaphor for an individual’s organisation of beliefs in similar ways, as conceptual knowledge is conceived in cognitive structures (Green, 1971, in Thompson, 1992). Some research claims that belief systems are more episodic in nature than knowledge systems, as they tend to connect to specific situations or experiences (Abelson, 1979), which would explain why knowledge is situated and dependent on context (Leatham, 2006).

A great deal of empirical evidence points to the significance of beliefs for understanding teacher behavior (e.g., Calderhead, 1996; Clark and Peterson, 1986; Pajares, 1992). Several scholars suggest that teaching practices are strongly associated with teachers’ beliefs about teaching and learning (e.g., Hofer, 2004; Hofer and Pintrich, 1997; Kagan, 1992), including interaction with students, instructional materials, and instructional design (Kagan, 1992; Song et al., 2007).

However, research describes inconsistencies between teachers’ beliefs and their classroom practices (e.g., Ertmer, 2005; Fang, 1996; Schraw and Olafson, 2002), which could be explained by contextual factors; teacher’s ability to apply their beliefs in practice do not match answers to a self-report conducted in a research context (Fang, 1996).
3.2. BELIEFS AND WORLDVIEWS

Teachers’ Epistemological Beliefs

Research about teachers’ epistemological beliefs in general have a more holistic approach compared to research about students’ epistemological beliefs, which is why the concept of worldviews was introduced (Schraw and Olafson, 2008) in order to depict teachers’ beliefs or belief systems (Olafson and Schraw, 2006; Schommer-Aikins, 2004; Schraw and Olafson, 2008). This could be a way of acknowledging the difference between epistemological beliefs, in relation to specific subject domains, and holistic epistemological stances. According to Schraw and Olafson it is important to distinguish clearly between epistemological beliefs and epistemological worldviews. The former consist of specific beliefs about a particular dimension of knowledge such as its certainty, simplicity, or origin. The latter consist of a set of beliefs that collectively define one’s attitudes about the nature and acquisition of knowledge. Each adult presumably has a set of epistemological beliefs that are included within an epistemological worldview. (Schraw and Olafson, 2002, p.102)

When studying the instructional implications of teachers’ beliefs regarding knowledge and learning, Schraw and Olafson (2002) identify/compare three epistemological worldviews. The identified worldviews are: 1) the realist, 2) the contextualist and 3) the relativist, all with their associated nine beliefs about knowledge, curriculum, pedagogy, assessment, reality and standards for judging truth, constructivism, the role of the teacher, the role of the student, and the role of peers. Below is a short summary of each distinctive worldview:

1. In the realist worldview there is an objective body of knowledge that is acquired via transmission and reconstruction. Teachers with the realist worldview perceive students as passive recipients. The knowledge at stake is pre-established and agreed upon by experts. To acquire high levels of skill you have to work systematically under the governance of the teacher.

2. In the contextualist worldview learners construct understanding in a collaborative context where the teacher acts as a facilitator. The learning process is more important than the knowledge it constructs, as knowledge will change over time, wherefore emphasis is on students’ skills so they learn to acquire knowledge on their own. Teachers holding this worldview encourage peer work and expert scaffolding. Authentic co-operative assessment is desirable.

3. In the relativist worldview each learner constructs their individual understanding of the same subject content. Teachers with the relativist worldview commonly design environments where students are encouraged to think independently. Self-regulation is an objective in itself, as peers are important promoters of self-regulation. Criterion-based assessment is used on an individual basis.

---

3See Schraw and Olafson (ibid.) for a description of each worldview.
In later publications Schraw and Olafson (2006) summarise these three worldviews into the concept of \textit{ontological beliefs}. The model they suggest consists of two dimensions and a four-quadrant scale, depicting the teachers as relativists or realists in each dimension—one for teachers’ epistemological belief and one for teachers’ ontological belief. In the study at hand, the model is used to propose the existence of commonality and specific beliefs caused by the subject domain.

### 3.3 Teachers’ Instruction

According to Guzdial, teachers are tasked “to make computation available to thinkers of all disciplines”. He pinpoints that the educational discourse about computer programming started early in the 1960s “…[where] programming was an exploration of process, a topic that concerned everyone, and that the automated execution of process by machine was going to change everything” (Guzdial, 2008, p.25). However, it seems that instructions in computer programming have remained the same since the 1960s. According to Kaplan (2010, p.1), computer programming “is largely taught today the way it was taught 60 years ago” while research has not informed the computer programming teaching community.

### Theories and Models for Teaching

According to McCormick (1992), research in informatics curriculum for secondary level education is rare. The absence of research papers dealing with established theories or models of learning and teaching programming is obvious (Sheard et al., 2009). Holmboe et al. (2001) highlight the nature and the scope of computer science education (CSE) research.

We argue that there has been a lack of reference to pedagogical theory, underlying most past research studies. This has resulted in a failure to provide teachers with ‘pedagogical content knowledge’, critical to gaining useful insights into cognitive and educational issues surrounding learning. (Holmboe et al., 2001)

Despite the troublesome lack of theories, there is some research related to teachers’ instruction from different disciplines. In an investigation by Postareff and Lindblom-Ylänne (2008), two approaches to teaching are identified: \textit{learning-focused} and \textit{content-focused}. Similar findings were reported by Kember et al. (2000) and Trigwell et al. (1994), who exposed the presence of different teaching approaches: \textit{student- and teacher-centred} instruction. The outcomes of these three investigations are not exactly comparable, but they do show the dichotomy between existing teaching instructional strategies. In a study performed by Boulton-Lewis et al. (2001) with 24 upper secondary school teachers, four instructional approaches were identified.
3.3. TEACHERS’ INSTRUCTION

1. Transmission of content/skills – focus is on the content and the students are somewhere in the background.

2. Development of skills/understanding – teachers direct the learning process and students are perceived as participants.

3. Facilitation of understanding – teachers and students work together to construct personal meaning

4. Transformation – the teacher organizes the situation to provide enough stimulus for students to take action, while the teacher fades into the background.

These four investigations pinpoint a distinction between teachers’ instruction and students’ engagement or ability. The second and third categories added by Boulton-Lewis et al. (ibid.) include approaches of importance for this study that will be further discussed.

Defenders and Partisans

During the 1980s, two main divisions of educators existed ‘side by side’, perceiving learning and teaching in programming differently. Teaching and learning computer programming was debated by the defenders of programmed teaching and the partisans of learning through discovery and self-teaching (Solomon, 1986 in Mendelsohn et al., 1990). In the spirit of behaviourism, the defenders perceived programmed teaching through repeated sequences as an effective learning tool (Suppes, 1979). In the spirit of constructionism, the partisans advocated learning through discovery as a way of supporting children’s own knowledge building (Papert, 1980). Today, the debate has faded while these two groups of defenders and partisans still exist among teachers. This will be discussed further.

In an attempt to find an instructional theory for computer programming education, Linn and Sloan studied “naturally occurring instructions” (Sloane and Linn, 1988):

Historically, programming classes built on experiences of expert programmers who taught themselves. Students were provided with assignments and access to computers and were expected to learn through trial and error and unguided discovery. (Sloane and Linn, 1988, p.208)

Teachers commonly used discovery learning, problem-solving procedures, or extensive feedback. An effective method for teaching programming seems hard to discover (Linn, 1985; Linn and Dalbey, 1985). A quotation from Linn and Clancy (1992) explains the situation that many students faced during the 1990s:

Programming instructors often assume that students can take their general problem-solving skills and discover specific programming design skills on their own. Thus, students learn program design through unguided discovery [...]. In programming, the acquisition of design skills is
further impeded . . . For example, texts frequently feature what is called ‘top-down design’, the process of designing a program by breaking the high-level statement of the problem into parts and then continuing with this process until the program is completed [...] Instruction that suggests program design proceeds in an uncomplicated, top-down fashion confuses and frustrates students [...] while teachers often describe the features of completed programs or the characteristics of the language syntax. [...] As a result, students may think they should know how to design solutions to problems without actually learning the skills. Programming instruction often implies that design skills are available to everyone. Students lacking a clue as to how to proceed in designing the solution to programming problem may conclude they are incapable of learning, when in fact, they are actually unaware of how to proceed. (Linn and Clancy, 1992, p.125-126)

The quotation describes a situation where teachers expect learning to emerge through unguided discovery and students to be able to draw from their former abilities or skills for success or failure. There are reasons to believe the situation still remains today. In 1996, East and Wallingford pinpoint the same instructional problem:

There is indeed little discussion of the teaching of programming that relates to pedagogy and almost none that addresses how the process of learning might or should affect instruction. (East et al., 1996, p.1)

Approximately 10 years later, the problem still remains on the agenda, as teaching programming is one of the seven challenges in computing education (McGettrick et al., 2005). Gries (2006) takes it one step further when he pinpoints the difference between teaching facts and teaching for ‘deeper’ understanding, which correlates with Resnick (2010) and a thinking curriculum.

We need to look seriously at how we teach programming. The purpose of an education should not simply be to pour facts into students, but rather to teach them to think. (Gries, 2006, p.82)
Chapter 4

Research Methodology

Galileo is considered one of the first to approach the celestial objects systematically.\(^1\) In a similar way, regardless of the fact that our tools to understand thinking are blunt by nature, the thesis at hand should be seen as an attempt to unravel teachers’ thinking in relation to curriculum. In search for the understanding of the phenomenon of teaching, the what and the how questions are continuously processed throughout the thesis. As a matter of fact, the shaping of a critical theory in opposition to the positivist beliefs that “research could describe and accurately measure any dimension of human behavior” (Steinberg and Kincheloe, 2010, pp.141) has grown to be somewhat familiar to me, as I had to fight the battle of how to approach the general and the particular.

In the beginning, I was primarily focusing on the collective level, with ambitions to gather the number in quantitative ways, delivering a “general knowledge” about teachers’ beliefs and practices (Study 2). Then in a later record, shifting towards the particular by studying teachers’ thinking I approached the individual level (Study 3). In that order working from the collective to the individual level, some of the characteristics in teachers’ thinking have been discovered.

For a better understanding of the subject matter, I found it necessary to gain perspective in time and space. This decision made me start the research project conducting two investigations in parallel: 1) A historical case, depicting the development of computing curriculum in Sweden (Study 1), and 2) exploring the diversity/conformity of teachers’ beliefs concerning programming content, the use of assessment/environment and student learning (Study 2).

The concept of curriculum is interpreted differently among scholars, leaving it somewhat difficult to discuss, as the borders between different actors could vary, and involved actors could be engaged in different stages of the curriculum development process. As the work with Study 1 and Study 2 proceeded, and empirical data was analysed, it was – for the matter of coherency – decided that the thesis would focus on the intended and implemented curriculum, leaving the attained curriculum for

---

\(^1\)Regardless, the fact that lenses were not produced as they are today.
future studies. Later, while approaching an interpretative position (Studies 2 & 3), it was considered necessary to narrow the phenomenon under investigation and limit the thesis to a teacher’s perspective. An overview of the research process, including the perspectives used (time and approach), is described in Table 4.1.

Table 4.1: A summary of the studies, approaches used and allocation in time

<table>
<thead>
<tr>
<th>Study #</th>
<th>Perspective</th>
<th>Approach</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Study 1</td>
<td>Curriculum</td>
<td>Historical</td>
<td>2009-2013</td>
</tr>
<tr>
<td>Study 2</td>
<td>Teacher</td>
<td>Hermeneutical</td>
<td>2009-2011</td>
</tr>
<tr>
<td>Study 3</td>
<td>Theoretical</td>
<td>Inductive</td>
<td>2012-2015</td>
</tr>
</tbody>
</table>

The design of the research could resemble other applied research designs, as action research or designed-based research. However, as the research was more in cooperation than partnership with the teachers, it is not described as any of these two. The research design is considered to be pragmatic and interpretative, as former teaching experiences were considered distractive in the research process. The selected three perspectives therefore became tools, to distance myself and bracket my natural attitude (Uljens, 1997); studying the subject from a historical, hermeneutical and inductive approach.

Trustworthiness and Ethical Considerations

The perspective of the thesis is situational, where insights and findings emerge throughout the data collection and analysis. As the data is qualitative, the test of validity and rigor is assured by trustworthiness (Lincoln and Guba, 1985). However, as such a research design requires ethical considerations where “the interconnectivity between production of knowledge at the ethics of production” (Trainora and Bouchard, 2012, p.3) becomes relevant. Ethical considerations are accounted for in accordance with Gustafsson, Hermerén, and Peterson (2005), who distinguish two aspects of considerations: researcher-ethical considerations and research-ethical considerations. This will be discussed in relation to the concepts embraced by trustworthiness (Denzin and Lincoln, 2000) and the ethical principles suggested by the Swedish Research Council (Vetenskapsrådet, 2002): information, consent, confidentiality, and use.

In the following each study will be described briefly, as the trustworthiness and ethical considerations of the thesis will be discussed more in detail. The reader is

---

2 For a discussion about the three levels of curriculum, see van den Akker (2003).
3 In the spirit of phenomenology, it is considered that the phenomenon of teaching a specific content could be shared between the teacher and student, as well as the phenomenon of learning a specific content could be shared likewise.
4 The discussion is influenced by reciprocity and rigorous reflexivity (Trainora and Bouchard, 2012) in research design.
4.1 STUDY 1: THE CURRICULUM PERSPECTIVE

referred to the papers Programming in School: Look Back to Move Forward, Teachers’ Beliefs Regarding Programming Education, Bridging a Gap, and Intentions and Pedagogical Actions, for a thorough description of the methods used.

4.1 Study 1: The Curriculum Perspective

Curriculum and syllabi are, by nature, a creation of a legitimate discourse done in society, of significance to what will be in a school context and what teachers with adequate degrees are supposed to offer in education (Apple, 1990; Carr, 1998). However, they are markers of something more relevant, as they describe values, norms, and practices expected to be taught, highlighting the fact that we have to scrutinize former curricula for an understanding of what we value today.

The following section describes the methods used in the construction of a historical case. Considerations and delimitations are discussed, as they have defined the discoveries as much as the search for documents and people involved. The section begins with a short introductory text about the use of arenas as a way of structuring the historical data.

Arenas - a Way to Structure the Data

The curriculum history in informatics is explored through studies of the continuous curriculum development process where both the micro- and the macro-levels are addressed in the search for existing determinants in the transformation of a school subject (Linde, 1993). This has been a common approach in educational research in Sweden since the 1960s, which brought forward the frame-factor theory (Dahllöf, 1969; Lundgren, 1972, 1977, 1983). The distinction between micro- and macro-levels were later made visible by the distinguishing of different arenas (Klasander, 2010; Linde, 1993; Lindensjö and Lundgren, 2000). The arena concept offers a rational way to describe the development process and reification of a curriculum at different levels: the arena of enactment, the arena of transformation, and the arena of realisation. Linde (1993) refers to the following arenas:

- The arena of enactment concerns the curriculum development process, from the initial steps to the final enactment of a curriculum document.

- The arena of transformation in particularly concerns teachers’ reification work of concepts described in the curriculum. Teachers are considered gatekeepers of what can be offered and established in the classroom. Principals and school publishers belong to this arena.

- The arena of realisation concerns the acts in classrooms according to the intentions written in the curriculum.

In this study the first and the second arenas are primarily focused.
Document Analysis and Interviews with Key Persons

The National Archives (RA) and Stockholm City Archive (SSA) have been valuable resources in the reconstruction of the development process, as they possess an abundance of documents from the time of interest. Specific effort has been made to understand on what arena/level, and by whom, the Swedish informatics curriculum was processed for the first time. Historical documents from archives depicting the curriculum development were scanned and compiled to digital volumes to enable study of the interaction between and within different groups at the National Board of Education (NBE) and upper secondary schools. Contemporary educational journals from the 1970s, 1980s, and 1990s were scrutinized to understand the overall process and communication within the development process.

Parties involved in the curriculum development process were identified and interviewed if still alive; around ten interviews were conducted. Resources (proposals, government bills, decrees, submissions, reports, and written communications) from the governmental library, Riksdagen Library in-house and on-line, were studied to understand how the Ministry of Education (ME) was involved in the development process.

4.2 Study 2: The Teacher Perspective

The following methods were used in order to unravel characteristics of commonalities among teachers' beliefs. The study is an explorative study and adopts a qualitative hermeneutic approach to unravel characteristics of commonalities among teachers' beliefs. The analysis is inductive and data-driven with the ambition to identify specific themes of interest.

Statistics

To get an overview of the number of schools and teachers involved in programming education at the time of the study, information from Statistics from the Swedish National Agency of Education (SNAE) was collected, and used for estimation of an upper limit of teachers engaged in programming education (Table 4.2).

Table 4.2: The number of schools that offered computer programming courses on the levels A, B, and C.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>A+B</th>
<th>A+B+C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>341</td>
<td>233</td>
<td>158</td>
</tr>
</tbody>
</table>

5 A resume of the most prominent people involved in the curriculum development is found in Emanuel (2009a; 2009b).
4.2. STUDY 2: THE TEACHER PERSPECTIVE

Questionnaires
The expected diversity of existent beliefs among teachers was of value for the investigation, wherefore questionnaires were considered a proper instrument choice to picture the mindset of the teachers’ community. A major rationale to offer seminars was discovered while teachers showed an interest in participating in forthcoming seminars. Another (logistic) rationale for the seminars was the possibility to distribute and manage the questionnaires.6

The questionnaire design (See Appendix B) was a result of the seminars, as they became important to better understand teachers’ experiences teaching and learning programming. The main questionnaire (Quest1) was sent to 250 teachers nationwide. For a thorough discussion about the questionnaires, I refer to the paper describing this study.

Informants and Seminars
Based on the numbers of upper secondary schools, an upper limit of computer programming teachers was set to 1050. As not all of them offered courses in programming, all existent school web pages were scrutinized. It resulted in 196 potential schools and 206 teachers. In the process, the initial list continuously evolved and reached approximately 250 computer programming teachers from all over Sweden who together constituted the Swedish Informatics Teachers’ Network (SITSNET). When the study was conducted the teachers in the community was estimated to 300 individuals, and therefore the search for members on the list was considered successful.

The four seminars were offered regularly from 2009 to 2011 focusing on didactic/pedagogical aspects of programming education, with the intention of vitalizing teachers’ reflections about programming education. For a detailed description of each seminar the reader is referred to Study 2, Teachers’ Beliefs Regarding Programming Education. The seminars became a valuable resource and an inspiration in the search for future seminar activities (e.g., content of importance to the teachers as well as educational problems faced by teachers). In total, 90 individual teachers attended one or more of the four seminars and 45 teachers attended two or more of the seminars. Each seminar was attended by 20 to 50 teachers.

Analysis of Data
The analyses were inspired by inductive analysis (Lincoln and Guba, 1985; Miles and Huberman, 1994) and the critical hermeneutic approach (Steinberg and Kincheloe, 2010) searching for characteristics or patterns among teachers. Based on what the informants answered, successive questionnaires focused on specific themes. In

---

6It should be noted that teachers’ thinking is not considered to be the actual outcome on the arena of realization (what the students encounter).
that procedure, specific themes and issues were elaborated and scrutinized in at least two seminars.

4.3 Study 3: The Theoretical Perspective

The third study in this thesis provides a theoretical examination of the teachers’ way of thinking about their teaching.

Literature Review

The first substudy (3a), is based on extensive literature reviews of relevant research/literature about the origins of phenomenography and logic of events (von Wright).

Phenomenography

In short, the ‘classical’ phenomenography chisels out a phenomenon in our world according to what a number of distinctive ways of experiencing can tell us (Bowden, 2005), as the variation theory pinpoints the importance of critical aspects and variation for individual learning – in other words, two ways of dealing with learning with an OoL playing one of the main roles. It should, however, be remembered that the OoL does not reside in someone’s thinking, as it embraces a hypothetical object framed by what we focus on when learning:

[...] we argue that independently of human beings we cannot define the object of learning. On the contrary, we suggest that learners contribute to defining this by learning about it and by doing so the object of learning is set in new human contexts and its meaning is enriched in the process [...] In fact the idea that students not only learn from teachers and the textbook, but actually contribute to the joint constitution of knowledge about the world. (Bowden and Marton, 1998, p.190)

In phenomenography, the experiences of understanding exist in the relation between the subject and the phenomenon. However, as I consider the teacher as a member of that “joint constitution of knowledge” when scrutinizing the intended OoL, it has been of interest to search beyond the phenomenon described by phenomenography for an understanding of the teacher’s intentions.

Logic of Events Analyses

In the second substudy (3b) analysis of statements from five teachers regarding their thinking about teaching programming is presented. To unravel implications and meanings of the intentions invested in the intended object of learning, five teacher cases were analysed with the framework presented in Study 3a to differ between teachers’ intentions and expectations. The interview transcripts were
subject to careful, detailed readings by the all the authors of the papers. In the analysis, specific intentions were identified, as well as the phenomena-content specific intentions and the teachers’ direct and indirect outspoken/“portrayed” pedagogical expectations, the latter with implications for what and how the OoL is manifested.

4.4 Trustworthiness

According to Denzin and Lincoln (Denzin and Lincoln, 2000), trustworthiness research involves establishing credibility, transferability, dependability and confirmability. In the following, these characteristics of qualitative research is discussed in each of the studies.

Credibility

Study 1  In order to make the research task in the study manageable, a limitation of the investigation has been made regarding the level of education and the timespan. The delimitation in time of Study 1 is linked to the revision of curriculum that started at the end of the 1980s, which brought about the curriculum Lpf94, later published in 1994. Of course there were other curriculum reforms succeeding the one studied. To make a proper choice of reform to study, each one of the curriculum reforms was investigated and data was collected to decide upon the quality of research that could be achieved (based on the identified actors and documentation preserved in the archives). However, it was decided the studied reform in this thesis was most appropriate.\footnote{A short description of each successive curriculum is found in the Study 1.}

Study 2  The research design was based on the idea that teachers hold a diversity of different beliefs about the educational situation, but commonality would appear if the beliefs were compared. In that process, a diversity of beliefs was presumed to be the research outcome. However, the method used will never explain any individual’s beliefs, as reflections are collected during different seminars.\footnote{The individuals did not participate in all seminars, while some joined later and some only attended the first seminar.} The study is therefore considered as explorative by nature.

One third of the teachers’ community expressed, early on in the research process, a sincere interest in participating in the seminars. It was therefore assumed that these informants represented those with a specific interest in teaching and learning, as well as those interested in socialising with peers having similar questions and experiences. A group of teachers on-site would therefore fulfill a social need, as well as offering opportunities to register characteristics. It could be expected that the population of informants was skewed, as the informants attending the seminars...
embraced a specific pedagogical interest or had some demographical data in common (e.g., age). However, as an explorative study investigating characteristics and instructional patterns, it was considered a successful design with the discovery of some critical features.

**Study 3** To make the outcome of the study generable, different cases were selected (Schofield, 1993, p.101) working in different levels of the educational system and cities. The outcome of nine expected “objects of learning”, together with the original learning objective, depicts the fact that the cases selected are heterogeneous.

The number of teacher informants are low, as each site demanded a lot of resources, interviews with students, and collection of material with implications for how many interviews that could be done. The rational for doing the teacher interviews afterwards with the ambition to anchor the discussion as much as possible to “realities and less likely to be ‘rhetorical’ in nature” (Abrahams and Millar, 2008, p.1950). The interviews and the outcome of the study would otherwise risk not offering anything exceptional.

**Transferability**

**Study 1** It was considered important to delimit the study to a teachers’ perspective on programming in education. Former research about computing in Swedish curriculum was scrutinized, but as it mainly concerned the diffusion of computer hardware in school (Jedeskog et al., 1991; Nissen and Riis, 1985; Nissen et al., 1991, 2000; Riis, 1987, 1991, 2000), it was of minor relevance for the thesis.

**Study 2** The method used in the study elicits the fact that curriculum development in informatics is the outcome of work carried out by many teachers and experts, and is officially engaged in a continuous process that started at least 40 years ago. An understanding of the development and the underpinnings that shaped today’s curriculum and instruction was believed to be of huge value to teachers’ reflection in practice and to future curriculum developers.

**Study 3** In the archeology of how ‘intended’ has been used in phenomenology, it was noticed that Husserl used the concept of ‘intentional object’ in phenomenology synonymously with the ‘intended object’ to pinpoint “whatever entity it is that makes an act intentional” (Smith and McIntyre, 1982, p.46). It is unclear to me, as well as for (Harris, 2011), exactly how to interpret intentionality in the phenomenographical way, especially when it comes to the teachers’ perspective. It could therefore be questioned whether the ‘intended object of learning’ described in the study, is a phenomenographical study object, as phenomenographers usually consider phenomenography to stand on its own prerequisites (Svensson, 1997).
Dependability

**Study 1** Specific attention in the investigation of archive documents has been given in relation to upper secondary school and vocational education in ADP (VADP). During the 1970s and 1980s, administration of the Swedish NBE expanded into different bureaus and divisions, whereof the following bureaus have been in focus:

2. G division for upper secondary school (1982 to 1991): with Bureau for curriculum development (LPU), and Bureau for course development and syllabus development (KPU 1 and KPU 2).

The ME archives have been another source of information. They show the curriculum process, in particular from 1986 onwards where different working groups focusing on computing education were initialized within the ME. Documents from the CSB archives revealed information on why two specific schools (Berzeliuskolan and Bandhagens gymnasium) and associated teachers became influential in the beginning of the curriculum development process. The SCSB archives revealed few documents of importance to this study.⁹

**Study 2** The thesis was dependent on teachers’ willingness and ability to share their beliefs and shortcomings. Offering professional development for teachers was considered an opportunity to do something more than just conduct the research process with recruitment, data collection, and analysis. The methodology used in this study depends on the interplay between teachers, researcher, and context of different sites, where the study - as well as the researcher’s learning - is the outcome. Examples of instruction were searched for among individual participants, especially those that brought something different to the teachers’ community. The seminars offered potential situations in which such differences could be revealed.

To unravel characteristics and instructional patterns from teachers’ experiences is a delicate project, as the number of informants correlates to the value and the respect invested in the seminars. Besides, the study is built on the principle of reciprocity (Trainora and Bouchard, 2012), with consequences for what could be discussed between the teachers and the researcher. As many of the teachers participated in all seminars, a specific subset could been extracted, discussing their beliefs in comparison between the age-groups. It was not done, as such a categorisation would strengthen the researcher’s own prejudice; e.g., teachers with many years of experience would be bound by former technologies, as younger teachers would be more open-minded to alternative instructional methods. In

---

⁹The CSB in Stockholm (SCSB) was scrutinised in relation to Bandhagens gymnasium in order to understand what sort of documents were preserved.
retrospect, such a step would have revealed something of importance for the research process, as teachers with experience from many reforms and different programming technologies presumably hold a reluctance towards how instruction and environments could enhance the students’ learning processes. A rational for not taking that step is caused by the ambition to treat every teacher with respect, not excluding any teacher based on age, gender, ethnicity, pedagogical ambitions, former degree, etc.

The seminars in the study were believed to offer an environment for research through interaction with teachers, as it would take too much effort and time to dispute my own beliefs regarding programming education.\(^{10}\) To stimulate reflection, the four seminars were offered at different sites: two academic ones (Stockholm University and KTH), an industrial one (Microsoft), and one upper secondary school. With such a design, teachers were exposed to at least three different perspectives: industry, university, and school.

**Study 3** In an essay, von Wright (1976, p.377) debates the question of the individuals’ awareness of an action. In line with common sense, von Wright agrees with other scholars (e.g., Wittgenstein\(^{11}\)) about our unawareness (ignorance) of many of our basic actions, but still acknowledges our awareness of the results from our actions. Teachers are therefore most likely ignorant of many of their actions, but in interviews reflecting their actions in retrospect, it is believed teachers are delivering their reasons for the “causes of the results of [their] actions” (von Wright, 1976, p.377). This could become an issue as the results from the interview are expected to offer teachers’ intentions and attitudes about their pedagogical actions.\(^{12}\) But, as this study is processed with similar assumptions as phenomenographical studies with a second-order perspective, the outcome of analysis is considered to be comparable.

**Confirmability**

As the purpose of the thesis – the gap between teachers’ intentions and the implementation of curriculum – concern three different perspectives, it is with comfort that all three studies emphasise the importance of a “practical approach” for learning programming, with hands-on activities instead of the other way around – offering theory to explain the principles of computer science. Approach the object of learning from a student’s perspective could probably reveal another picture of programming in education. Especially if the “theoretical students” are observed (See Discussion).

\(^{10}\) At the time I had been teaching computer programming for at least 10 years.

\(^{11}\) “The freedom of the will consists in the fact that future actions cannot be known now” (Wittgenstein and Wedberg, 1992, 5.1362).

\(^{12}\) The teachers in the study are considered experienced with at least 10 years of teaching programming; they are expected to be aware of common actions in classroom.
4.5 Ethical Considerations

The following text shows the ethical principles followed in the thesis

1. *Information*: During the whole research process (during interviews, seminars, and when distributing the questionnaires) information was presented to inform about the anonymity and the confidentiality used when processing the information.\(^{13}\)

Names of people involved have been used while they were officially employed by NBE or government. Whenever people were interviewed, they were also informed about intentions for how information will be processed and used for research. To fulfill the ethical principles of *information* and *use* (Vetenskaprådet, 2002), each informant was asked whether their information could be used for research.

2. *Consent*: The principle of consent could be questioned, but there never existed any sort of obligation to become an informant or to attend the seminars. Each questionnaire was conducted in relation to a seminar, except the second (Quest2).\(^{14}\) They were distributed beforehand along with research intentions and information about confidentiality, etc. The methodological design of the questionnaires in relation to the seminars might put pressure on the informants, as informants could feel they had to give something back in return. However, this was never the intention, and each informant was asked beforehand if s/he would be willing to share his/her experiences.

3. *Confidentiality*: It is always easier to be critical in hindsight, wherefore I as a researcher have to be careful with how I handle the informants’ values and experiences. During the whole investigation process I have been in contact with a large number of people who are/were committed to what they believe is/was necessary and important. These teachers’ stories and witness accounts of engagement made an impression, which has been of great value to me and my research. Therefore, it is important to live up to and pay back the trust informants showed me. During the whole research process (in seminars as well as within questionnaires), letters and information were delivered to inform about the research intentions and the confidentiality that would be used when processing the information.

4. *Use*: The information offered by informants has been used for research purposes, this thesis, papers, and future research work. The informants were informed of this.

\(^{13}\) See Appendix B.

\(^{14}\) The second questionnaire was distributed during the seminar while informants were informed of the research intentions and the confidentiality that would be used when processing information.
Chapter 5

Summary of Papers

The chapter gives a brief presentation of each paper and some of the results. Conclusions and results are discussed in the next chapter. For a comprehensive understanding, the reader is referred to the papers.

5.1 Study 1: Programming in School

(Published in ACM Transactions on Computing Education, Vol. 14, No. 2, Article 12, Publication date: June 2014)

The development process, the final enactment, and the early implementation of the informatics curriculum is described. It is a case describing the informatics curriculum developed in Swedish upper secondary school during the 1970s and the 1980s. The research question addressed reads: *On what grounds were programming education in Sweden implemented and what lessons could be learned from this?*

To unravel the interaction between schools, teachers, and the National Bureau of Education (NBE), documents regarding the curriculum development process available at the National Archives (RA) and city archives were studied.

Findings reveal that the Natural Science Programme (NSP) was remodeled during the mid-1970s. While holding a pragmatic educational approach, the new informatics curriculum was believed to attract more students. The new curriculum was also believed to offer enhanced learning in Mathematics and Physics, as logical computer technology and these two subject matters were believed to hold similar rational intentions.

The interaction between and within different groups at the NBE was studied to better understand the early steps taken towards the enactment of the informatics curriculum. Some teachers and principals worked at many levels in the school system. Three upper secondary schools in particular had previous experience with vocational education in ADP-Hedbergska skolan in Sundsvall, Carlforska skolan in Västerås, and Berzeliuskolan in Linköping, whereof the last one, Berzeliuskolan, became significant in the enactment of the informatics curriculum.
A pre-existing vocational education during the 1960s and 1970s became the blue-print for the formation of the new subject matter. However, other schools also contributed to the development. At Sunnerboskolan in Ljungby (Björk, 1974, 1977, 1983; Björk et al., 1975; Kollerbaur, 2005), a working group of enthusiastic teachers of different subject matters showed a tremendous driving force in using computer technology for interaction and enhanced learning. This enthusiasm could be traced back to the early 1970s when curriculum developers in Mathematics and Physics (Kollerbaur, 2005; Utbildningsdepartementet, 1971) saw computers as tools for a dynamic curriculum. Computer programming was first introduced to the NSP within a specific alignment for informatics. Later it was introduced in other programmes as well, within experimental work. The outcome of this introduction was commonly perceived as unsatisfactory, as computer programming was time-consuming and not appropriate for students with a small interest in the technocentric environment that computers could offer during the 1980s.

The implementation of the school subject informatics exposed another problem caused by the heritage from vocational education in ADP. The societal concern and system development were believed to be important for understanding the true value of computer programming. This concern became an issue for the curriculum development group as teachers generally did not have the appropriate competence to teach system development/systemisation. Besides, the content of system development/systemisation was experienced as vocational and non-technical, which made the content appear somewhat strange and not appropriate for the NSP. However, these opinions seemed to vanish with time, as the computer application market offered educational software more appropriate for subject matters like Mathematics and Physics. In 1985, the Swedish association of computer teachers (Swedish: Svenska datalärarföreningen) stated that

[...] technologically oriented modules like computer programming should decrease in size in favour of information technology, system development, and project collaboration. (Utbildningsdepartementet Datautbildningsgruppen, 1986, Appendix 3, p.5)

The time when computer programming was believed to foster logical and analytical reasoning in schools had now passed. Computer technology itself offered new educational technologies with new hopes for enhanced learning and better understanding of concepts in Natural sciences and Mathematics. Computer programming became an optional knowledge domain for students with a specific computer technology career in mind. In the curriculum reform of 1994, Lp94, computer programming was re-introduced as a subject matter on its own after the electricity trade organisation put pressure on the Swedish National Agency of Education (SNAE) to offer computer programming courses.

In this paper, the development of the Swedish informatics curriculum during the 1970s, 1980s, and 1990s is studied and described. The study’s design is inspired by the curriculum theory presented by Lindensjö and Lundgren (Lindensjö and
Lundgren, 2000), who suggest using the concept of arenas (the arenas of enactment, transformation, and realisation) when discussing curriculum development. Data collection in this study comprises activities and actors in the arenas of enactment and transformation. Collected data include contemporary papers, journals, reports, booklets, government documents, and archived documents. Findings show that informatics education in Sweden evolved from primarily focusing on programming knowledge related to automatic data processing and offered exclusively in vocational education (the 1960s and 1970s) to later (early 1980s) being introduced in the upper secondary school curriculum under the heading Datakunskap. The enactment of the informatics curriculum in 1983 encompassed programming, system development, and computing in relation to applied sciences and civics. Mathematics teachers did much of the experimental work. It is shown that the competencies of upper secondary school teachers at the time rarely corresponded to the demands of the subject (content knowledge, resources, and pedagogical skills). Stereotypical examples were therefore developed to support teachers in instructing the subject content. When implemented in the theoretical natural science programme, system development/systemisation was transformed into a twofold issue comprising vocational attributes and societal aspects of computer programming. The implementation of today’s informatics education, including programming in the curriculum, should draw from lessons learned from history. For a successful outcome, this study emphasises the necessity to understand 1) the common incentives for introducing computer programming in the curriculum, 2) the requirement for teachers’ pedagogical content knowledge, and 3) the stakeholders’ role in the curriculum development process.

5.2 Study 2: Teachers’ Beliefs Regarding Programming Education

The second paper explores the beliefs of today’s programming teachers from the following research question: What beliefs do programming teachers express regarding teaching and learning computer programming in upper secondary school? The paper revolves around data collected during a series of seminars focusing on upper secondary programming education. Between 20 and 50 of the teachers in the study attended one or more of the seminar/s. At each seminar, a questionnaire designed to elicit teachers’ beliefs about aspects of importance for their instructional design and students’ learning was given to the teachers/informants.

The responses were analysed (Lincoln and Guba, 1985; Miles and Huberman, 1994) to find common themes underpinning the work of computer programming teachers. The analysis showed four themes in relation to teachers’ beliefs about learning and teaching.
CHAPTER 5. SUMMARY OF PAPERS

1. Students’ individual connective time.
2. Teachers’ pedagogy.
3. Students’ abilities.
4. Students’ interest and motivation.

Further analyses reveal that the assessment process is crucial to teachers’ choice of instruction strategies. This is particularly valid in the beginners’ course, where collaboration among students is often practiced, and where skills essential to working in groups are commonly considered not to be important. This is considered a common practice among secondary school technology teachers (Hennessy and Murphy, 1999; Sidawi, 2009). In conclusion, it could be said that two distinctive instructional patterns exist among teachers—individual support and instruction for learning experience. This fully corresponds to previous research (Boulton-Lewis et al., 2001).

The data analysis exposed that a majority of the teachers in the study express a number of expectations concerning their students’ abilities; specific abilities such as logical and analytical thinking are emphasized as important for successful learning, while the ability to work in a group and to communicate is perceived as beneficial but not of any concern during the assessment process. The paper raises the question of whether teachers perceive abilities as fixed and inherent (naive belief) or something that students could acquire with some effort (sophisticated belief). Findings suggest that a majority of the teachers hold a naive belief. Findings also show that the teachers in the study focus on the individual, constructivist-based learning which, according to Schraw and Olafson (2008), indicates that the teachers in the study commonly hold on to relativistic worldviews. To understand whether computer programming teachers hold a realistic or a relativistic epistemology, further investigation would be needed.

A specific feature appeared in the study regarding the concept of pedagogy. Depending on how the questions were phrased, teachers’ belief in pedagogy varied from question to question. When teachers were asked, “Do you consider pedagogy as important for programming education?” 95 percent perceived pedagogy as important, while in another session only five percent perceived pedagogy as an important factor for students’ learning. On the other hand, only 23 percent of the teachers perceived a degree in pedagogy as important when asked “What kind of education do you perceive as necessary for being a teacher in programming?” An interpretation of these different percentage numbers reveals the existence of a relationship between instruction and pedagogy, but it does not necessarily coincide with teaching (Nuthall, 2004).

5.3 Study 3a: Bridging a Gap

(Submitted to International Journal of Technology and Design Education)
Computing and computers are introduced in school as important examples of technology. Sometimes there is a separate subject about them, sometimes they are used as tools, but in principle, learning about computers is part of learning about technology. Lately, the subject is implemented in the curriculum to explain society’s dependence on programming knowledge and code. There are some considerations related to teaching programming, as the questions of what and how to teach offers different aspects to the learning object.

In phenomenography, intended object of learning (OoL) is suggested to describe the teacher’s perspective on teaching and learning. There is an analytical reduction made in phenomenography which makes such a construction hard to distinguish in action. To find ways of bridging that reduction and deepen our understanding of teachers’ work, the paper discusses how determinants of intentions by von Wright are complementary to understanding the intentions behind such a construction.

Two theoretical approaches, Phenomenography and Logic of events, are deployed in a teachers’ case to reveal that the intended OoL is shaped by teachers’ intentions, balancing importance of theory, different learning strategies, discovery learning by trial-and-error, and collaboration between students for a deeper understanding of the OoL. In conclusion, the paper reveals the teachers’ intentions as complementary to phenomenography.

5.4 Study 3b: Intentions and Pedagogical Actions

(Submitted to Computer Science Education)
The teacher focuses on one or several objects of learning in parallel. In the process of teaching and learning, students are invited to share the enacted object of learning (eOoL), as teachers hold intentions to shape the intended object of learning (iOoL). The step going from the eOoL to the iOoL is considered a gap, as students hold different prerequisites and ambitions to be involved. In laboratory work theory and practice, there is supposedly an interplay for enhancement of students learning. The study therefore considers that gap to be of interest, addressing the question What educational intentions and expectations do programming teachers express when they (in retrospect) describe their teaching during an assignment focusing on a principle from computer science? Interviews are conducted with five teachers from different sites (secondary and tertiary level). A second order perspective is used to unravel the expected phenomenon, as well as teacher’s intentions. The study reveals the existence of other OoL in interplay with the studied learning objective. Among the teachers’ a strong opinion of practice as a means for learning theory is discovered, as theoretical knowledge is expected in three qualitatively different students’ actions. For an understanding of the gap, further work is suggested to explore the teachers’ reflection about what is materialized in the classroom.
5.5 The Author’s Contribution

The research design, data collection and production of papers is described in the following, as

- Paper I: Rolandsson full authorship.

- Paper II: Rolandsson first authorship, and by that taken full responsibility for research design, data collection and production of the paper in discussion and close collaboration with Skogh.

- Paper III: Rolandsson, first authorship, and by that taken responsibility for the full paper (draft, outline & final) in continuous discussion with the co-authors. Skogh has contributed by particular theoretical input (theory of logic of events) and, together with Männikkö Barbutiu, taken the holistic view of the paper by processing the text.

- Paper IV: Rolandsson, first authorship, and by that taken responsibility for research design, data collection and production of the paper in discussion and close collaboration with Männikkö Barbutiu and Skogh, who has also contributed by theoretical aspects from philosophy. Data underpinning this study comes from a larger data collection undertaken by Rolandsson and Skogh, financed by The Swedish Research Council.
Chapter 6

Discussion

Now, as well as in the past, novel subjects are introduced into the national curriculum, putting emphasis on teachers’ ability to teach the content. It could therefore be expected that in future classrooms other challenges, brought forward by technological endeavors and human thinking will put heavy demands on teachers to transpose the content, making it applicable for all pupils and students. Accordingly, Shulman (1987) emphasises that there is a difference between knowing a topic and being able to teach it, which - in the case of computers and programming knowledge - raises both practical and didactical concerns. The title of the thesis, *Programmed or Not*, presume such a distinction made by Shulman, as it challenges the teachers to fight their own epistemic attitude to become aware of students’ thinking. However, the title stress as well, the fact that teachers’ instruction could be a consequence of curriculum ideologies. In agreement with Apple (1990) and Carr (1995; 1998), the thesis embraces the purpose with curriculum as a reflection of the society. A statement with implications for the teachers to determine, transform, and reflect upon the outcome in the classroom. But, if teachers are already determined in their instructions, what could possibly be challenged and changed?

The thesis consider teachers as able to transform their attitudes and break their bounds and gain new knowledge in teaching. Like Wittgenstein, the thesis argues for ways to emancipate the individual (teacher) from a world where their actions are determined by their thinking. In von Wright’s opinion the human race is not following a strict determinism, as nobody can ever assume what is to come. However, in some sense we are determined, as we are looking for explanations and reasons from our experiences (von Wright, 1971), a fact that von Wright stretched to its extreme where individuals are “programmed” with the possibilities to be “re-programmed” (von Wright, 1983, 1998). In the following, the three studies are synthesised into a conclusion to discuss the difference between teachers’ intentions and what is implemented in classroom. Based on the discoveries, there is a message for future curriculum. The next step for a change, reflection, and professional development is considered for another study.
6.1 Study 1: The Curriculum

Study 1 is an exemplary case extracted from the history of Swedish curriculum reforms, as it offers a record of documents as well as identified actors involved in the formation of a computing curriculum on a micro- and micro-level. Besides, it offers a setting where technological innovations (computer hardware and programming knowledge) pushed for readjustment of the existent curriculum in already existing subject matters, e.g., Mathematics and Natural sciences.

On a macro-level, the case depicts a situation where a former institution, education in vocational ADP (VADP), influenced the constitution of a new curriculum embracing an anti-theoretical attitude to attract a broader student group at a theoretical upper secondary programme. The case is considered interesting as it depicts how computing technology worked as a means for politicians’ ambition, where two specific ideologies are working in parallel in the midst of a school system transformation - the liberal-progressive ideology and a modernist-vocational ideology (Carr, 1995, 1998). In other words, an enactment and implementation conducted among curriculum developers and teachers with consequences for NBE to find out how the subject should be offered in curriculum for as many students as possible.

On a micro-level, system development/systemisation was taught in VADP schools, at that time, which was considered appropriate even in the new computing curriculum for upper secondary school. However, that specific content demanded teachers’ expertise that existed mainly among VADP schools. The teachers destined for the new computing curriculum mainly belonged to the Mathematics and Natural science departments, wherefore it became a teacher’s competence issue never properly solved. Later, as computer applications successively diminished the need to teach programming, system development/systemisation was considered an option by the NBE, as it was believed to offer a thinking curriculum more than the technicalities in programming ever could offer.

According to the study, NBE searched for a societal context-related approach for delivering programming as a tool. If programming was not offered anymore, something else had to be taught about computing. To teach programming as a separate skill never reached any significant status. This is considered an important discovery, as it pinpoints an issue about what teachers at that time could deliver, no matter what politicians initiated and curriculum developers (teachers) proposed. There were some implicit deficits in the curriculum, such as the rationale for its existence, since it was attended by a specific group - boys attending an elite programme in natural science, NSP. This was an outcome that probably caused the programming content to be marginalised or surpassed by computer applications replacing the need to teach programming skills.

As the arguments in the introduction of the thesis describe, the drop-off rate in programming courses is high. It could be expected that the subject matter and teachers’ instruction are somewhat skewed towards “especially talented students” - in this case boys - and not following the ambition of a liberal-progressive ideology.
which expects an “active development of pupil’s understanding [... instead of ...] passive transmission of society’s knowledge” (Carr, 1998, p.328), something that hardly materialize among the students attending programming courses (even) today. An educational outcome of minor interest during the 1980s, as well as today.¹ In the following section, some of the rationales for such a pattern, most likely 35 years with a biased outcome towards boys in an elite programme, will be discussed, as the subject never materialized as an egalitarian curriculum.

6.2 Studies 2 & 3: The Teachers

There is one specific aspect of concern when computing is transposed from a subject domain to a subject matter in school - teachers’ pedagogy. During the 1970s, a positive attitude towards programming was commonly present, primarily among curriculum developers. This could be a possible reason that “how to teach programming” became a non-question in the early stages. It was not until the end of the 1980s that NBE experts questioned the appropriateness of the pedagogy in the subject matter (Björk, 1987; Dahland, 1990).

According to Pears et al. (2007), some teachers delivering beginners’ programming courses at university level believe their teaching has little, if any, impact on the learning outcome of the students, which seems to be the case even among teachers at upper secondary levels (Study 2). As a matter of fact, Study 2 reveals that the majority of teachers emphasise students’ logical/analytical ability as crucial for the educational outcome, which could imply that teachers’ pedagogy matters for the students that already have the abilities that the programming courses intend to develop.

In search for what could have caused such a slow pedagogical development, the two following paragraphs discuss society’s ideologies and teachers’ worldviews, beliefs, and intentions.

**Teachers’ worldviews and ideologies:** Research on teachers’ beliefs is commonly based on the assumption that teachers’ deeds are a product of their beliefs (Hofer, 2004; Hofer and Pintrich, 1997; Kagan, 1992; Song et al., 2007). It should be remembered that the interplay between deeds and beliefs is complex in nature (Beswick, 2005). Besides, these beliefs are probably a result of the ideologies embraced in society; e.g., teachers embracing a liberal-progressive ideology primarily guide rather than instruct, with the ambition to “stimulate pupils’ natural curiosity and facilitating their own enquiries” (Carr, 1998, p.328). This correlates well with the outcome from Study 2, with teachers embracing a relativistic worldview (a worldview emphasising the non-importance of lecturing in favour of individual learning in a constructivist manner).

---

¹In a recent Swedish curriculum reform in 2011, an alignment in the NSP, focusing on programming skills, closed down for the second time. The programming courses are still offered. However they are mainly attended by boys at Technology and Electricity programmes.
Study 2 discovers a tension regarding teachers’ thinking about students’ ability. Some of the teachers express confidence in their students’ ability to learn and some did not. As an example, teachers could occasionally express “I am one of those that believe any student can learn programming”, while others rejected such a statement. This is a thinking that depicts to what extent teachers are working to bridge the gap - holding intentions for enhancement of students’ ability to learn the content, or if they consider student to have inert abilities. The outcome from Study 2 discloses the instruction of teachers’ community as divided into two strategies - individual scaffolding and instruction for the learning experience. The former focuses on the specific issues each student will have, as the latter focuses on a liberal-progressive attitude towards learning. The result aligned with what Boulton-Lewis et al. (2001) discovered, as suggesting two main types of instructional strategies: 1) development of students’ skills/understanding and 2) teaching for facilitation of understanding. The two studies are difficult to compare in detail, but still pinpoint the fact that the distinction found in Study 2 has some correlation to the two ideologies - modernist-vocational and liberal-progressive.

A distinction made between two ideologies that most likely been around ever since the early days of computing education. In search for when such an instructional pattern was initiated, it was discovered from the history of computing education in the 1980s, that the existence of partisans emphasised the importance of repetition in small steps while the defenders emphasised the advantages of ‘discovery-learning’ where the students construct their own learning. The partisans emphasise a mechanistic step-by-step procedure in harmony with the modernist-vocational, as the defenders harmonize well with the liberal-progressive ideology.

In other words, there seems to be some rationale found from the past that correlates with teachers’ beliefs today, where teachers are working towards one of the two different curriculum ideologies in our society - a modernist-vocational curriculum with the purpose of providing the knowledge and skills appropriate in a market driven economy, or liberal-progressive curriculum with the purpose of offering rational autonomy and freedom to work individually. In other words, teachers’ instructional strategy could be a consequence of the overall democratic progress, where they reach out for the novices becoming computing professionals or the individuals becoming literate in computing.

**Teachers’ intentions:** In Study 3, teachers expressed a skepticism towards theoretical aspects of knowledge for beginners to learn. They showed dependence on practice as a means for students’ learning. Perhaps this is a consequence of how programming knowledge always has been taught, but according to the teachers in the study, it is a consequence of the students’ learning strategies or personalities being theoretical or practical. This distinction made among teachers in the study could be a consequence of prejudice, but is still relevant as it is used in their

---

2The Swedish curriculum developers during the 1970s found evidence that supported the partisans’ argumentation.
thinking. No matter what, regardless of students’ strategy or personality, practice is considered among the teachers in the study to be a major prerequisite for learning both theory and practice.

In an attempt to transform such a preconception about theory and practice, Study 3 embraces a critical approach to study teachers’ intentions. In Study 2, an instructional pattern according to teachers’ beliefs was discovered (See Figure 6.1), where theory and the practice is biased towards practice, except in the advanced course (at upper secondary school).

Figure 6.1: The diagram pictures how teachers answered the question “What sort of education do you consider beneficial for students learning programming?” on a scale between 1 and 6.

Such a pattern could be a consequence of what teachers are inclined to emphasise by tradition or what could be expected by an advanced course. Either way, teachers have to deal with the issue of theory, as it otherwise could easily track students into those that are inert or innate by birth for learning. This issue has existed ever since the school system transformed from the elite school system to a school for
the masses, putting demands on teachers’ ability to offer an appropriate setting for all (Carr, 1998). The teachers in Study 3b have partly solved that issue, as they emphasise the importance of practice ahead of theory until students have reached the level of acquaintance to apply the theoretical aspects of knowledge.\(^3\) In other words, teachers in beginning courses are constrained by the students’ achievement with hands-on code before they can tutor, teach, or instruct. Teachers in Study 3b do expect theory to be discovered in time by the students. The study reveals the existence of qualitatively different understandings of theory. Teachers expect to find expressions of theoretical understanding in students’ actions, in one of the following ways:

- read and understand course literature,
- verbally communicate the code, and
- deploy structure and functionality in code.

**In summary:** The distinction, made by the teachers, between theoretical and practical students is considered an important discovery, as it could be expected that “theoretical students” with ambitions to read-before-writing will have a hard time compared to the “practical students” that are in the process of deploying code. Both categories depict students actively searching for (theoretical) knowledge, whereof the “theoretical student” has not understood on what grounds teachers assess their understanding of knowledge. In other words, a practical subject that could been implemented differently among teachers with ability to reflect on the proposed gap when observing students in action.\(^4\) However, the subject of programming is actually taught and considered important for problem-solving in general and the development of students’ thinking abilities in particular. Two arguments commonly dissolved and processed in curriculum development, e.g., in technology education (Herschbach, 1995). As a matter of fact, the practical aspects of programming knowledge during the 1970s were one of the major arguments when computing was introduced in one of the theoretical programmes at upper secondary school.

### 6.3 Conclusion

Even today computer programming courses remain a challenge for teachers to teach (Berglund and Lister, 2010) and for students to learn (du Boulay and Benedicte, 1989; Haberman, 2006; Perkins et al., 1989) because students are incapable of understanding essential concepts and ideas in computer science (Kurland et al., 1989; Winslow, 1989).\(^5\) It seems like teachers’ instruction is not enough to enhance

---

\(^3\)Teachers from secondary as well as tertiary school.

\(^4\)According to Tedre (2014) it was theoretically based in former times.

\(^5\)Scholars use the metaphor of threshold concepts (Boustead et al., 2007; Meyer and Land, 2003) to describe the fact that computer programming could be experienced as troublesome for students.
students’ learning. According to Kaplan, computer programming “is largely taught today the way it was taught 60 years ago” (2010, p.1) while research has not informed the computer programming teaching community. In the following three rationales have been identified that could explain the characteristics of the gap investigated in the thesis, that probably existed during all these years:

1. **The teachers’ beliefs about the significance of pedagogy**: The tension between the two ideologies described above, pinpoints the fact that the implementation of computer technology could become a mediator for curriculum change, as technology is believed to improve the teachers’ instruction as well as the enhancement of students’ problem-solving skills. This scenario is commonly discussed, even today, in school for enhancement of learning and new literacies (Vee, 2010). The thesis unravels two groups of teachers with different beliefs about what is beneficial for students’ learning process. The thesis proposes two groups as teachers elaborating on how to bridge the gap between what they expect and what they perceive of students’ learning.

2. **The teachers’ beliefs about students’ ability**: Originally programming knowledge was offered in Sweden to a specific target group, and later was tried out in a broader context. It was dropped during the late 1980s when programming skills were not considered important enough. Could it be that curriculum developers were forced to leave the original intention with programming as a tool, and instead focus on the application of computing? Or, did school function as an experimental setting to observe what sorts of marvels teachers can achieve with minimal understanding of the technological agenda, minimal professional development, and minimal technological resources in education? In comparison with today’s teachers, I hope we can learn something from history, as even today’s teachers describe the educational outcome as dependent on students’ logical/analytical abilities.

3. **The instructional strategy of using practice and theory**: As discussed, the curriculum is a reflection of political, economic and cultural aspects of life (Carr, 1998) embracing ideological ideas in society. The thesis has unraveled teachers’ beliefs and intention in an attempt to study the gap between what is intended and what is perceived among teachers. The problem arises as it could be questioned whether curriculum is the outcome of teachers’ intention or predetermined by the needs in society.⁶

Even theoretical knowledge (among students) is interpreted by the teachers (Study 3) in relation to three qualitatively different students’ action: in communication, in reading the course literature, and in the deployment of structure and functionality of code. Accordingly, a pedagogy where students’

---

⁶Elgström and Riis (1990, p.87-89) describe the ambitions heralded by the NBE to find a balance between theory and practice in school. This and other issues of ICT in relation to society and political intentions for upper secondary school are discussed by Murray (1988).
theoretical understanding is fostered in action. This outcome is supported by the fact that computer science has adopted an anti-theoretical attitude (Eden, 2008), an opposite position in former times when theoretical aspects on the subject were adopted to a higher extent (Tedre, 2014). This is a shift towards practice shaping the nature of the subject, with consequences for how computing education is presented and experienced among teachers as well as students.

As a final remark for discussion I will focus on two issues: *Resources for Curriculum Change* and *Programming for All to Learn*. I will do this in order to raise some considerations about the impact of ideology and technology on teachers’ pedagogy.

**Issue 1: Resources for Curriculum Change**

According to von Wright (1976), innovations of technology can become an important prerequisite for change of epistemic attitudes and change of our intentions, as we are not determined by obligations, educational institutions, or educational technology. Instead, with new technological innovations, former experiences are diminishing and educational practices and teachers’ instruction are open for change. As a matter of fact, von Wright argues that the human race has to be cautious about how technology influences our way of thinking, as it could constrain us as well as become useful for finding new ways of doing things (von Wright, 1976). Caution should be raised when studying to what extent teachers’ pedagogy are actually influenced by computer technology or determined by the curriculum ideologies existent in society, or both.

According to Mendelsohn et al. (1990), teachers are allowed to elaborate in three different areas to achieve successful computer programming education—the level of the computer programming language, the intended transferability of the language, and the overall teaching style, where beliefs about teaching influence the teachers’ approach (Mendelsohn et al., 1990). Study 1 reveals this does not have to be the case, as there existed a specific trust and partnership between teachers and people from the SNAE, a partnership that was prosperous from the 1970s and throughout the 1980s. As this was a time of new technology releases, it could be expected in line with von Wright’s reasoning that technology became a significant impetus for some teachers. Teachers’ beliefs are according to research (Kagan, 1992; Luft and Roehrig, 2007; Olson, 1981; Yerrick et al., 1997) perceived as resistant to change. But, as Study 1 describes, teachers can make a difference if they are provided with the right conditions. As the study reveals, many teachers involved were authors of school literature, which in turn could become an incentive for their contribution to the process.

There are, however, some recommendations for empowerment of teachers’ instruction as well as students’ learning. As noted, technological innovations in history offer alternatives for curriculum with implications for how curriculum is
6.3. CONCLUSION

expected to change. In search for new instructional strategies, the following are forwarded (the determinants of intentions) as examples:

- teachers’ wants: with technology, teachers could offer education with thinking before application of code. There are different alternatives embracing music, graphics, and art to visualise a computational principle.\(^7\)

- teachers’ abilities: ambitions to increase their pedagogical content knowledge challenging their presumptions about teaching and learning with unplugged activities that pinpoint technological thinking.\(^8\)

- teachers’ duties: roles for assessment, making someone else do the final grading of students’ achievement with the help of e-learning platforms.

- teachers’ opportunities: instruction, design of collaborative settings where students will experience engineering interdependence.

Issue 2: Programming for All to Learn

In one interview, the informant expressed concern about computer programming while he spontaneously said:

“Programming, the dinosaur that survived in school!”

I found that utterance remarkable and conclusive, as it describes computer programming education as archaic. Perhaps it mirrors how curriculum has been perceived over the years or the old-fashioned way of communicating with computing machines via a specific programming language. In my opinion, it is awkward to expect all students to learn the syntax of a language without any further context.\(^9\)

So, what actually did survive? According to me teachers’ ignorance to explore what sorts of benefits the content could unravel in relation to a broader context. As it is now, teachers’ instruction mainly offers a setting students are expected to try out their code until it works with the presumption that they will probably understand theory by then (See Study 3). I consider that sort of instruction excludes many of the interested students. Currently, the content is offered with piles of syntax and often inaccessible content. I therefore suggest a re-imagining of programming, where the thinking of computer science/computing are more in focus than the implicit technicalities of the syntax.

\(^7\)The programming language of Processing is one example. (2015, February 2). Retrieved from https://processing.org/.


\(^9\)As teachers are not fully aware of the heritage of programming in our society, in resemblance with the teachers during the 1980s who were expected to teach system development/systemisation without any vocational experiences.
6.4 Message to Curriculum Developers

Currently, computing and programming are experiencing a revival in our society that most likely has not been witnessed since the textual representation of human affairs became prevalent. Over the years, many attempts have been made to open up computing in education and lower the barriers for beginners to learn programming (Holt and Cordy, 1988; Kelleher and Pausch, 2005).\textsuperscript{10,11} It should be remembered that computing curriculum needs teachers’ reflection. For the purposes of this thesis, the curriculum could be approached differently than commonly done. I suggest two curriculum alternatives emphasising the thinking of programming:

- It is important for the pupils/students to understand the limitations of what computers can do; the computing machine does not do anything else than what you explicitly state in code what to do, and this might not necessarily be what you had in mind in the first place. With such an objective in mind, you could draw from such a constraint by instead exploring the nature of interactions between humans and between computers and humans. This embraces the what, but when it comes to how, programming could be taught in the aesthetics of recycling code, for instance. Another methodology for instructional practice could be found, in the tinkering processes, as students start to appreciate what is actually inside their “heads” in similar paths as an artist commonly crafts when expressing herself.\textsuperscript{12}

- Historical aspects of computing are essential when reflecting on teaching and learning the subject (Akerja and Aspray, 2004; Micheuz, 2006). As a matter of fact, the original ideas by Babbage were based on a language for thought (Swade, 2010) emphasising the computational aspects in a rational way, but still transparent for others to develop their own thinking.\textsuperscript{13} Today we are fortunate with digital tools (instead of pens) where our design (by shapes, code, or mark-up languages) can be done in real-time rendering. This a tremendous shift going from mechanical devices to digital devices in exploration of the logics of our thinking. Why do we then depend on writing code for an understanding of the impact of our achievements? Perhaps skills in

\textsuperscript{10}Commonly, such attempts imply the construction of programming languages where one “avoids the languages that enforce many new abstract ideas, or have too many ‘programming tricks’ to be learnt, or give too little immediate concrete feedback” (Mendehlson et al., 1990).

\textsuperscript{11}A description of the work established, 1963 to 2005 by the International Federation for Information Processing (IFIP) and the technical committee in education, TC3, was made by Watson (2006).


\textsuperscript{13}A fundamental principle of use in a democratic society. At the time when Babbage constructed the Analytical Engine, he realised it was too expensive to work out the necessary mechanics for problem solving. As paper and pen are easier and less expensive to work with, he invented a descriptive language that he thought would become the standard design method taught in the engineering schools-in other words a dynamic language for thought.
computer programming are all there is to an understanding of the engineering prototyping and simulations? Or, are there other ways to experience the enhancement of thinking with technology? Further studies are needed to answer these questions, but such an approach for curriculum has a lot more to offer than syntax in a programming language.

6.5 Further Studies

Different incitements in curriculum development were noticed during the investigation, each holding their own specific perspective on computer programming education - industry in need of people with skills, industries with interest in selling hardware, politicians with democratical ambitions, the curriculum reflecting society, and the teacher with the subject matter in mind. The following headlines summarise what I found to be relevant for further studies. Each headline raises a research project impacting curriculum studies and teaching and learning experiences.

Evolution of Technology

Investigations about vocational education in ADP (VADP) were conducted in different archives in Solna, Växjö, and Örebro municipalities and interviews with principals and school personnel from Linköping, Umeå, and Stockholm were carried out. These elicit the fact that VADP schools were self-governed to a high extent (finances, personnel, and computers), while hosted by upper secondary schools until the mid-1970s. Based on these investigations – and the tentative results that people from VADP were only involved in the initial phase until the mid-1970s when VADP was incorporated into tertiary education (Huvudmannaskapskommittén, 1980) – it is considered important that computing was experienced differently between people from VADP and the teachers from Mathematics and Natural science school departments. At that time, computer technology was hard to manage – especially for education – and not as today with cost-efficient and easily-manageable computers. Further studies could reveal to what extent the technological evolution and the teachers’ competence are the main factors of importance for the experiences of curriculum.

The National Agency of Education

During the 1980s, the informatics application group was initiated by the ME, with intentions to facilitate computer technology in schools (Emanuel, 2009b; Lindberg, 1988; Utbildningsdepartementet, 1988) and supply educational software in different subject matters.14 This was a specific work group started by the Ministry of Education which initiated a new programming language suitable for Swedish

---

14The informatics application group (Swedish: Dataprogramgruppen, DPG), existed 1985 to 1988. The group was later transferred to the NBE.
circumstances, DPG-PROLOG (Utbildningsdepartementet, 1988), and a computer for the school market, COMPIS (Emanuel, 2009b). It is believed that the formation of the group and the incitements to develop a new programming language held much of the discourse at that time, in relation to the intersection of education and computer technology. A study in comparison with today’s incentives of programming knowledge in school, could most probably unravel what beliefs we are still fighting in relation to computer and computing technology.

Teacher’s Network

During the investigation it became obvious that other interest groups pushed for a change, like teachers’ associations (ALF, SIGYM, ASKI-gruppen) in collaboration with computer manufacturers (Luxor, Norsk Data, and IBM). The documents that were scrutinised reveal a heavy focus on computer hardware technology as prerequisites to offer computer programming in schools instead of pedagogical aspects. A comparative study between today’s movement of technologically enhanced learning and the 1980s, when computer resellers were surfing the optimistic wave, would be an opportunity to study some of the rationales for curriculum change.

Teacher

For the complementarity of a teachers’ perspective it would be of interest to conduct a study with the students’ perspective. Teachers and students experiences is expected to intersect, as they experience a collective phenomenon (Marton, 2014; Uljens, 1997), wherefore it is expected that learning and meaning making are components that are discovered in the joint action of the phenomenon. It would therefore be of interest to study to what extent teachers’ and students’ directedness are alike, and how they interplay over time.

There are though an issue that arises as the phenomenon is experienced differently among the students, leaving them with one or many object(s) of learning to focus, where the teachers’ intentions (intended object of learning) do not necessarily correspond with students’ experiences (lived object of learning). Accordingly, the object of learning can be observed from at least two different perspectives. Translated into the context of phenomenology: in the optimal situation, the teacher perceives students’ directedness towards the same object of learning as s/he intended. In such a study it would be of value to describe the presumed “difficulties” from a teacher’s and the students’ perspectives, and to what extent teachers are aware of these difficulties.
Kapitel 7

Sammanfattning
(summary in Swedish)

Svensk titel: Programmerad eller Inte - programmering i skolan från ett lärarperspektiv

7.1 Inledning


Att uppgadera lärarens ämneskunskaper inom informationsteknologi (IT)\(^1\) är

\(^1\)På grund av en viss begreppsförvirring och olika namn på kurser inom datortek-
av stor vikt, men det räcker inte. Den didaktiska förmågan i relation till ämnes-
innehalten är minst lika viktigt (Shulman, 1986, 1992). Ämnesdidaktisk forskning
har mer sällan uppmärksammat problemet från ett lärarperspektiv för att förstå
underliggande faktorer i IT-undervisningen.

Nya ämnen introduceras i skolan med jämna mellanrum. Nyligen fick Storbri-
tannien en ny läroplan i “Computing” som ersatte den tidigare IT-undervisningen.
Förutom att ämnet fick ett innehåll som motsvarade samhällets behov av nya
kunskaper, så är syftet med ämnet att alla elever, från de lägre åldrarna i grund-
skolan upp till och med gymnasiet, skall få möjligheten att lära sig något om
datavetenskap. Liknande satsningar förbereds i andra länder. Då IT är så pass
allmängiltigt och gängbart i samhället, kommer detta initiativ att följas av andra
liknande läropansreformer. Följande avhandling undersöker lärarens föreställningar
inom IT-undervisning, för att diskutera läroplanens begränsningar och framtida
behov av reformer. Avhandlingen bygger på att IT i skolan har förändrats minimalt
sedan ämnet introducerades på 1970-talet (2010), samtidigt som IT i samhället i
övrigt har förändrats mycket. En teknisk förändring som inte påverkat skolans
undervisning nämnvärt. Nyligen gjordes en svensk utredning som visade på behovet
av “digital kompetens” (Digitaliseringskommisionen, 2014). Utredningen visar band
annat att skolans syn på IT måste förändras. Därför måste lärare måste få bättre
kunskaper i IT. Det framgår dock inte huruvida ämnesdidaktisk kunskap är en del
av en sådan kompetenshöjning. Den ämnesdidaktiska forskningen har inte tagits
på tillräckligt stort allvar under ett flertal år (Kaplan, 2010). Faktum är att lärare
får stora svårigheter att undervisa IT om de inte får en ämnesdidaktisk kunskap
som visar på något annat bortanför datorer och mjukvara. Det är därför hög tid
att politiker, lärarhögskolor och Skolverk gör något radikalt för att förändra IT-
undervisningen. Ett antal förslag finns att läsa under ‘Discussion’ och rubriken
‘Message to Curriculum Developers’.

Forskningsfokus

Det finns ett moment i all typ av undervisning där läraren arbetar med att iscens-
sätta sina intentioner. För vissa lärare kanske det är mera regel än undantag att
undervisningen inte blir som det var tänkt och därför svår att analysera.² Men
faktum är att läraren som inte reflektar omkring skillnaden, mellan intention och
iscensättning i klassrummet, kan få svårt att styra sin undervisning mot elevers

²Resonomangen bygger på att lärare har föreställningar som är svåra att förändra (Abelson,
7.2. TEORETISKT RAMVERK


2. Studie 2 (Lärare): En undersökning som beskriver lärarens föreställningar kring ämnets och undervisningens förutsättningar. Frågeställningen är: Vilka föreställningar uttrycker programmeringslärare om programmeringsutbildning i gymnasieskolan?


7.2 Teoretiskt ramverk

Hur avhandlingen ser på läroplanen


\textsuperscript{3}Skillnaden mellan lärarens intention och icensättning beskrivs mera i detalj under ‘Theoretical framework’.

\textsuperscript{4}Se Bruner (1996, ss.44) understryker behovet av att lämna vardagspedagogiken.
KAPITEL 7. SAMMANFATTNING (SUMMARY IN SWEDISH)

denna uppdelning i tre delar och ett socio-politiskt perspektiv som ramverk för att förklara lärares föreställningar och intentioner.

Intentioner


Intentionalitet


Inom fenomenografi används begreppet för att visa på hur personen blir allt mera medveten om olika aspekter av lärandeobjektet. I avhandlingen utgår jag ifrån att läraren har en intentionalitet om elevens lärande och kunskap. I en jämförelse mellan denna fenomenografiska/fenomenologiska typ av intentionalitet och den intentionalitet som von Wright syftar på framgår att i den fenomenografiska/fenomenologiska kan vi vara (olika) medvetna om våra upplevelser genom våra sinnen, medan i den sistnämnda utgår vi ifrån en persons handlingar för att förstå dess intentionalitet.

---

5För en diskussion om parallelism se Curley (1985).
7.3 Tidigare forskning


Läroplanen förändras kontinuerligt


Den stora frågan blir därför: Hur får man en hel lärarkår och ett skolsystem att motsvara samhällets behov av en modern syn på IT (Donaldson and Knupfer, 2001; Pelgrum, 2001)? Cuban understryker behovet av att utveckla användbar undervis-
KAPITEL 7. SAMMANFATTNING (SUMMARY IN SWEDISH)


Programmering i skolan


Lärares föreställning om kunskap


Lärares föreställningar är ett begrepp som speglar lärare kunskapssyn och förhållande till sin undervisningspraktik. Man kan därför peka på vikten av ett ämnesdidaktiskt forum (t.ex. lärarutbildning och lärarnätverk) för att IT-ämnem med sitt kunskapsinnehåll skall få genomslag i praktiken. Avhandlingen använder sig av lärares föreställningar om undervisning i programmering för att uttala sig om dagens undervisning. Vad gäller lärare så har man kunnat visa att deras föreställ-

6IT som skolämne är enligt mig något som ständigt förhandlas på olika samhällsnivåer.
7Detta finns beskrivet i Studie 1.
7.4 Metod

I det stora hela har avhandlingen antagit ett pragmatiskt-tolkande tillvägagångssätt. Nedan följer de olika metoderna som har använts i respektive perspektiv.

Läroplansperspektiv

Historiska källdokument från Riksarkivet och olika Stadsarkiv har använts. Ett tiotal personer som var engagerade i kursplaneutvecklingen intervjuades. Även källor från Riksdagens bibliotek, som belyser politiska intentioner och investeringssatsning på datorer i skolan har använts.

Data av relevans för kursplaneutveckling, från Riksarkiv och olika stadsarkiv, skannades och sammanställdes i digitala volymer för att granskas i efterhand; interaktionen mellan olika arbetsgrupper inom SÖ och mellan SÖ och olika skolor/lärare studerades speciellt.

Lärarperspektiv


I samband med seminarierna tillfrågades lärarna om sina erfarenheter och didaktiska överväganden i programmeringsundervisningen. Insamlade data analyserades och kategoriserades i en tvästegsprocess. En preliminär analys visade på två olika synsätt vad gäller eleverns förutsättningar att lära sig programmering; det fanns lärare som ansåg, och inte ansåg, att alla elever kan lära sig programmering. I nästa steg genomfördes en induktiv dataanalys där olika teman arbetades fram (Lincoln and Guba, 1985; Miles and Huberman, 1994).

Teoretiskt perspektiv

Ett teoretiskt ramverk baserat på händelserlogik och en persons handlingar arbetades fram för att tolka lärarens intentioner. Ramverket bygger på ett andrahands perspektiv, på liknande sätt som fenomenografi; d.v.s man utgör ifrån den transkriberade texten för att kategorisera och tolka erfarenheter och intentioner. Ramverket ansågs av värde då det fokuserade på lärarens och elevernas handlingar snarare än det förväntade fenomenografiska fenomenet. Med data från olika lärare, analyserades

8Om möjlighet ges finansiellt och resursmässigt kommer seminarieserien fortgå. Träffarna genomfördes i samarbete med Teknikum/Stockholms universitet, Uppsala universitet, KTH och Microsoft Sweden AB.
lärares intentioner, för att undersöka hur ett ämnes teoretiska och praktiska be-
ständsdelar bygger upp undervisningen.

7.5 Diskussion

Avhandlingen arbetar med ett mikro-makro perspektiv, där läroplansbegreppet är
långt mer än styrdokumenten. Med detta som bakgrund undersöker avhandlingen
skillnaden mellan lärares föreställningar och det som läraren avser att implemente-
ra.

Nedan följer en syntes av avhandlingens tre studier där läroplanen och lärarens
föreställningar diskuteras. Därefter följer några förslag av betydelse för framtida
reformer.

Läroplanen

I sökandet efter intressanta fall inom den svenska läroplanshistorien, valdes ett
fall från 1970 och 1980-talet, där skolmyndigheten fokuserade på programmering
och systemutveckling. Det var en tid med positiv politisk vilja, där programme-
ring skulle erbjudas så många som möjligt. IT som ämne var nytt och dåvarande
läroplan var föräldrad till innehållet. Teknikutvecklingen var påtaglig i samhället.
Man utgick från en samtida eftergymnasial yrkesutbildning för programmerare. Det
fanns löften om utveckling av elevers logiska förmågor och bättre problemlösning.
Ämnet som fick namnet Datakunskap underströk med emfas att programmering
måste undervisas i ett större sammanhang. Försöksverksamheten slutade med att
elever vid naturvetenskaplig linje fick möjligheten att lära sig ämnet. Med andra ord
användes erfarenheter och föreställningar från en yrkesutbildning för att förändra
innehållet i en teoretisk utbildning. I ett makroperspektiv kan man se det som två
läroplansideologier i samband. Den utlösende faktorn var datortekniken i samhället,
som läroplansutvecklare skulle omsätta till något undervisningsbart. I ett mikroper-
spektiv, implicerade ämnet flera didaktiska överväganden, som t.ex. avsaknaden av
lärare med tillräckliga kunskaper inom området. Så småningom försvann delar av
ämnet innehåll, då programvara ersatte behovet av att undervisa programmering.
Problemöverförsling med egenkonstruerade algoritmer i t.ex. matematik och fysik var
inte längre beroende av kunskaper i programmering. Det till trots, så ville Skol-
myndigheten att systemutveckling skulle erbjudas fortsättningsvis, då sådant ämne
lyfter fram tänkandet och visar på programmering utifrån ett samhällsperspektiv.
Kunskaper i programmering nådde aldrig någon högre status. Dessutom fanns en
inbyggd problematik i ämnet då det mestadels undervisades bland stude
motiverade pojkar. Ett faktum (med pojkar) som återfinns även i dagens programmeringskur-
ser.

Nedan diskuteras några av skälen till denna utveckling, baserat på ett nutids-
perspektiv med fokus på lärarens föreställningar och intentioner.
Läraren

I avhandlingen framgår att en majoritet av dagens lärare anser att inte alla elever har förmågan att lära sig programmering. Dessutom ifrågasätter många lärare huruvida en annorlunda pedagogik kan förbättra elevers lärande. Med andra ord en mycket besvärlig undervisningssituation för ett ämne, där man kan misstänka en avsaknad av en didaktisk repertoar. Dessutom existerar bland lärare i program-mering ett speciellt förhållningssätt, då lärare lever i föreställningen om behovet av studenters logiska och analytiska förmåga (innan kursen startar). Man skall dock komma ihåg att pedagogik är ett mångfasetterat begrepp; i studien visar det sig att det finns två grupper av lärare; de som

undervisar för elevers individuella lärande

och

undervisning för elevers erfarenheter av lärande.

Det finns en klyfta mellan de lärare som anser att “alla elever kan lära sig att programmera” och de som anser att “alla elever kan inte lära sig att programmera”. Dessa två grupper av lärare anses spegla de två ideologierna (’modernist-vocational’ och ‘liberal-progressive’) som länge har samexisterat i skolan.9 Två ideologiska idéer som troligen funnits så länge som IT-ämnet erbjudits i skolan. Med andra ord finns det två lärargrupper: En lärargrupp där syftet med undervisningen är att sälla fram de mest lämpade och en annan lärargrupp där syftet är att undervisa eleverna så att de utvecklar sina förmågor.10 Lärares föreställningar pekar därför på kopplingen mellan samhällets demokratiska ambition och de ideologiska strömningar som påverkar läroplansutvecklingen.


Slutsatser

Baserat på lärarens olika föreställningar och intentioner söker avhandlingen förstå lärarens tänkande i förhållande till läroplanen. Studieobjektet för avhandlingen ligger i skillnaden mellan lärarens intentioner och det som läraren iscensätter. Huruvida lärarna lyckas omsätta ämnet till något som passar alla elever är problematiskt. I spänningsfältet mellan praktik och teori har lärarna gjort ett (o)medvetet val till

9 Ideologier som kan sammanfattnas i “Så många elever som möjligt skall kunna utbildas” och “alla skall ha möjligheten att utveckla sina förmågor till problemlösning”.
10 Den förstnämnda utifrån en marknadsdriven ideologi (modernist-vocational) och den andra utifrån en bildningsideologi (liberal-progressive ideology).
fördel för de elever som lär sig praktiskt. Programmeringsundervisningen kan därför uppfattas som exkluderande. I ett större sammanhang kan man därför fråga sig om dagens undervisning är tillräckligt utvecklad, för att alla elever kan erbjuda de kunskaper som IT omfattar.

Theses in Education and Communication in the Technological Sciences from KTH Royal Institute of Technology


Appendix A

Swedish School System
### The Swedish School System

#### The management

<table>
<thead>
<tr>
<th>Level</th>
<th>The parliament</th>
<th>The government – Ministry of education</th>
<th>The National Bureau of Education, NBE</th>
<th>County School Board</th>
<th>School Board</th>
<th>Principal</th>
<th>Teacher</th>
<th>Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The parliament</td>
<td>The parliament</td>
<td>The National Bureau of Education, NBE</td>
<td>County School Board</td>
<td>School Board</td>
<td>Principal</td>
<td>Teacher</td>
<td>Student</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Jurisdiction

<table>
<thead>
<tr>
<th>Example of decisions</th>
<th>General objectives for the society</th>
<th>School law</th>
<th>Decrees and curriculum</th>
<th>Central planning</th>
<th>Regional planning</th>
<th>Organization plan</th>
<th>Working plan</th>
<th>Education</th>
</tr>
</thead>
<tbody>
<tr>
<td>The parliament</td>
<td>Specific laws</td>
<td></td>
<td>Decrees</td>
<td>Administrative</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The parliament</td>
<td></td>
<td></td>
<td></td>
<td>decisions for</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The parliament</td>
<td></td>
<td></td>
<td></td>
<td>execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure A.1: Translated from the original version summarized in Swedish by Marklund (1987, p.69).
Appendix B

Questionnaires
KONSTEN ATT UNDERVISA I PROGRAMMERING


Som ett led i datainsamlingen har jag tagit fram följande enkät med frågor som jag ber Dig besvara. Enkäten vänder sig till lärare som anmält intresse till programmeringsolympiaden. Den information Du lämnar kommer att behandlas konfidentiellt och data kommer att avidentifieras när resultatet avrapporterats.

I ett senare skede kommer jag att kontakta några av dem som besvarat enkäten för ett uppföljande samtal. I formuläret kan Du med ett kryss ange att Du vill vara med också i denna datainsamling.

Den besvarade enkäten skickas lättast till <min.epostadress@xxx.se>
För dig som önskar skicka den besvarade enkäten med posten kommer ett separat utskick att göras. Om så är fallet vill jag att Du meddelar mig per e-post.

Tack för Din medverkan - Dina synpunkter är viktiga!
Har du frågor kring studien eller enkäten får du gärna kontakta mig.

1. Personuppgifter

<table>
<thead>
<tr>
<th>Namn</th>
<th>Ålder</th>
<th>Man</th>
<th>Kvinna</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-post</td>
<td>Skola</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Vill Du medverka i en uppföljning av Dina svar på enkäten? (via telefon alt. via personligt möte)

JA □ NEJ □
2. Programmering och programmeringsspråk

A. Vilken typ av utbildning har Du i programmering?

B. Vilken utbildning tycker Du att man behöver för att undervisa i programmering?

C. Hur länge har Du undervisat i programmering?

D. Vilka åskurser och gymnasieprogram har Du undervisat i den senaste femårsperioden?

E. Vilka av följande kurser har Du undervisat i den senaste femårsperioden? Fyll även i det programmeringsspråk du använt.
   a) Programmering A  JA   NEJ
   b) Programmering B  JA   NEJ
   c) Programmering C  JA   NEJ
   d) Webbdesign  JA   NEJ
   e) Databashantering  JA   NEJ

F. Vilket programmeringsspråk föredrar Du?

G. Vilken inställning har Du till produkter med öppen källkod?

H. Har Du arbetat utanför skolan med programmering under den senaste femårsperioden?
   JA   NEJ

I. Om svaret är JA på föregående fråga ber jag Dig ange typ av arbete som Du haft.

3. Undervisningen

A. Har Du och Dina elever arbetat tillsammans med andra skolor, eller klasser/grupper i programmering?
   JA   NEJ

B. Om svaret är JA på föregående fråga ber jag att Du beskriver Ditt syfte med detta samarbete.
**KONSTEN ATT UNDERVISA I PROGRAMMERING**

<table>
<thead>
<tr>
<th>C.</th>
<th>Brukar Dina elever använda Internet (forum, chatt, skype, wiki m.fl.) för interaktion med andra elever och andra programmeringsintresserade för att lösa uppgifter i programmering?</th>
</tr>
</thead>
<tbody>
<tr>
<td>JA ☐</td>
<td>NEJ ☐</td>
</tr>
</tbody>
</table>

| D. | Använder DU andra bedömningsgrunder än prov för att värdera elevers kunskaper? Om svaret är JA ber jag att Du kort beskriver det underlag Du använder vid betygssättningen. |

<table>
<thead>
<tr>
<th>E.</th>
<th>Använder Du film (video på nätet, youtube) i Din undervisning?</th>
</tr>
</thead>
<tbody>
<tr>
<td>JA ☐</td>
<td>NEJ ☐</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>F.</th>
<th>Använder Du facebook, twitter, blogg eller wiki i Din undervisning?</th>
</tr>
</thead>
<tbody>
<tr>
<td>JA ☐</td>
<td>NEJ ☐</td>
</tr>
</tbody>
</table>

| G. | Om Du använder utveckling av spel i undervisningen; Vilken typ av spel? |

| H. | Brukar Dina elever arbeta i huvudsak enskilt eller tillsammans(parvis eller grupper)? |

| I. | Vilken typ av undervisning anser Du är bäst för att elever skall lära sig programmering? |

| J. | Beskriv vilka utvecklingsmiljöer (ex. Notepad, Eclipse, BlueJava eller Dreamweaver) Du använder i programmeringsundervisningen? |

| K. | Vilken är enligt Dig den tydligare kopplingen mellan matematik och programmering? |

| L. | Finns det, enligt Dig, någon koppling mellan programmering och annat ämne i gymnasieskolan? |

| M. | Arbetar Dina elever företrädesvis med enkla och mindre uppgifter eller med komplexa och större programmeringsuppgifter? |

### 4. Lärande

| A. | Vilka kvaliteter anser Du att en elev behöver för att bli duktig i programmering? |
**KONSTEN ATT UNDERVISA I PROGRAMMERING**

<table>
<thead>
<tr>
<th>B. Vad innebär det enligt Dig att lära sig problemlösning?</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. Anser Du att ALLA kan lära sig programmering? Motivera varför.</td>
</tr>
<tr>
<td>D. Lägger Du in estetiska värden i programmering? JA □ NEJ □ VET INTE □</td>
</tr>
<tr>
<td>E. Om svaret är JA på föregående fråga ber jag att Du beskriver egenskaper som gör koden estetiskt tilltalande?</td>
</tr>
<tr>
<td>F. Hur många timmar/vecka förväntar Du Dig att Dina elever arbetar hemma med programmering?</td>
</tr>
<tr>
<td>G. I Programmering B och C finns flera begrepp som syftar mot en objekt-orienterad programmering. Vilket eller vilka är, enligt Din mening, det svåraste begreppet/begreppen då elever skall lära sig objekt-orienterad programmering?</td>
</tr>
<tr>
<td>H. Många elever kan lära sig syntax och programmeringsspråkets funktion, men stöter på svårigheter då de skall lösa större uppgifter. Vad är enligt, Din mening, den vanligaste orsaken till elevers problem?</td>
</tr>
<tr>
<td>I. Vilka kvalitéer skiljer, enligt Din mening, elever som får betyget G respektive betyget MVG i programmering?</td>
</tr>
<tr>
<td>5. Nätverk</td>
</tr>
<tr>
<td>A. Använder Du nätverksgemenskaper för att lära Dig mer om programmering? Om svaret är JA får Du gärna dela med Dig av Dina kontakter.</td>
</tr>
<tr>
<td>B. Har eleverna på Din skola en grupp med programmeringsintresserade?</td>
</tr>
<tr>
<td>C. Är Du intresserad av att ingå i ett nätverk med lärare för att a) utbyta idéer i skolämnet programmering? JA □ NEJ □ b) utbyta programmeringsuppgifter? JA □ NEJ □ c) utveckla samarbetsmöjligheter mellan elevgrupper och skolor? JA □ NEJ □</td>
</tr>
</tbody>
</table>
KONSTEN ATT UNDervISA I PROGRAMMERING

d) fortbilda Dig i programmering?  

JA ☐  NEJ ☐

Stort tack för Din medverkan!
**Quest2 - Kunskaper i Programmering**

### Personuppgifter

<table>
<thead>
<tr>
<th>Namn och skola</th>
<th>Man</th>
<th>Kvinna</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lärarutbildning eller pedagogisk påbyggnadsutbildning</td>
<td>Ja</td>
<td>Nej</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Antal år med undervisning i Programmering</th>
</tr>
</thead>
</table>

**Under de senaste fem åren har Du undervisat följande kurser:**

- Programmering A
- Programmering B
- Programmering C

**Kan alla elever lära sig kunskaper motsv. Programmering A?**

| Ja | Nej |

**Kan alla elever lära sig kunskaper motsv. Programmering B?**

| Ja | Nej |

**Kan alla elever lära sig kunskaper motsv. Programmering C?**

| Ja | Nej |

### I vilken omfattning förutsätter DU att eleven har följande förkunskaper i *Programmering*?

<table>
<thead>
<tr>
<th>Lite</th>
<th>Mycket</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2 3 4 5</td>
<td>att kunna kommunicerar kunskaper och programmeringsstrategier</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>förmågan att arbeta i grupp</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>att se helheter och skapa strukturer</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>att ”bryta ner” ett problem i detaljer (analytiskt tänkande)</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>ett logiskt tänkande</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>ett matematiskt tänkande</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>ett abstrakt tänkande</td>
</tr>
<tr>
<td>1 2 3 4 5</td>
<td>i det aktuella programmeringsspråket (syntaxkunskaper)</td>
</tr>
</tbody>
</table>
I vilken omfattning arbetar Du med att eleven skall utvecklas i följande kunskaper i *Programmering*?

<table>
<thead>
<tr>
<th>Lite ... Mycket</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>att kunna kommunicerar kunskaper och programmeringsstrategier</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>förmågan att arbeta i grupp</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>att se helheter och skapa strukturer</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>att ”bryta ner” ett problem i detaljer (analytiskt tänkande)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>ett logiskt tänkande</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>ett matematiskt tänkande</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>ett abstrakt tänkande</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>i det aktuella programmeringsspråket (syntaxkunskaper)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

I vilken omfattning anser Du att följande kunskaper är viktiga för att eleven skall få högsta betyg i *Programmering*?

<table>
<thead>
<tr>
<th>Lite ... Mycket</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>att kunna kommunicerar kunskaper och programmeringsstrategier</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>förmågan att arbeta i grupp</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>att se helheter och skapa strukturer</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>att ”bryta ner” ett problem i detaljer (analytiskt tänkande)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>ett logiskt tänkande</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>ett matematiskt tänkande</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>ett abstrakt tänkande</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>i det aktuella programmeringsspråket (syntaxkunskaper)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

Kommentarer (skriv på kanten om det inte räcker):
I vilken grad är det viktigt för Din undervisning?


För att resultatet skall kunna sammanställas i tid, behöver jag ditt svar senast, torsdag förmiddag. Även andra svar som kommer senare är välkomna, då sammanställningen är av betydelse i framtida seminariediskussioner. Den information som Du lämnar behandlas konfidentiellt och data kommer att avidentifieras.

Den besvarade enkäten skickas lättast till <min.epostadress@xxx.se>

Tack för Din medverkan - Dina synpunkter är viktiga!
Har du frågor kring studien eller enkäten får du gärna kontakta mig.

<table>
<thead>
<tr>
<th>1. I vilken grad är följande faktorer viktiga för elevers lärande?</th>
<th>Inte alls</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Mycket</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Programmeringsspråket har enkel syntax</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>B. Det finns färdiga moduler/klasser/bibliotek</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>C. Språkets syntax är generaliserbart till andra språk</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>D. Användargränssnittet är lätthanterligt</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>E. Det finns funktioner för programdesign</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>F. Det finns verktyg för felsökning av buggar mm</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>G. Programvaran är gratis o. tillgänglig via Internet</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>H. Det finns grafiska representationer av objekt (typ spelliknande miljöer som Alice, Scratch)</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>I. Programköring går att visualisera stegvis</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>J. Språket och/eller miljön används inom industrin</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>K. Möjlighet att arbeta tillsammans via Internet</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>L. Möjlighet att arbeta nära datorsystemet (komplera, länka och bygga skript)</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>M. Du kan programmera objektorienterat</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>N. Du kan programmera funktionsorienterat</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>O. Det finns bra pedagogisk litteratur/material</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>P. Programmeringsspråket kräver ett strukturerat arbetssätt vid implementation</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Q. Man kan programmera externa enheter (LEGO robotar, styr- och mätlaborationer mm)</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

Forts på nästa sida
2. I vilken grad är följande faktorer viktiga för din undervisning?

<table>
<thead>
<tr>
<th>Faktorer</th>
<th>Inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Mycket</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Elevers önskemål</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B. Elevers tidigare erfarenheter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C. Elevers intresse och motivation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D. Elevgruppens sammanhängande tid vid datorn</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E. Elevers nytt av språket/miljön i högre studier</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F. Elevers vilja till samarbete med andra elever</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G. Elevers logiska förmåga</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H. Elevers strukturella förmåga</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I. Elevgrupp bestående av NV - elever</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J. Elevgrupp bestående av TE - elever</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K. Elevgrupp bestående av EC - elever</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L. Elever läser programmering som individuellt val</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. I vilken grad är följande faktorer viktiga för att utveckla din undervisning?

<table>
<thead>
<tr>
<th>Faktorer</th>
<th>Inte alls</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Mycket</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Dina kunskaper i liknande språk/miljö</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B. Dina möjligheter (tid, vilja mm) till fortbildning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C. Ett nätverk med kunniga personer i programmeringspråket/utvecklingsmiljön</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D. Skolledningens stöd och engagemang i Din utveckling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E. Samarbete medan lärare för att utveckla nytt undervisningsmaterial (text och mjukvara)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F. Endagars seminarier under en längre tid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>G. Kortare kurser</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H. Längre kurser</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I. En koppling till lärarutbildning med lärarkandidater som praktiserar hos Dig</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
En enkät för lärares perspektiv på undervisning i programmering


Som ett led i datainsamlingen har denna enkät utarbetats med frågor som jag ber Dig besvara. Den bygger på tidigare enkäter, seminarier och intervjuer med enskilda lärare. Informationen du lämnar kommer att behandlas konfidentiellt och data kommer att avidentifieras när resultatet avrapporterats.


Den besvarade enkäten skickas snarast möjligt till <min.epostadress@xxx.se> för en sammansättning inför seminariet.

Tack för Din medverkan - Dina synpunkter är viktiga!
Har du frågor kring studien eller enkäten får du gärna kontakta mig.
### ATT UNDERVISA I PROGRAMMERING  
#### Quest 4

#### 1. Personuppgifter

<table>
<thead>
<tr>
<th>Namn</th>
<th>Ålder</th>
<th>Man</th>
<th>Kvinna</th>
<th>E-post</th>
<th>Skola</th>
</tr>
</thead>
</table>

Är du intresserad av en pdf-kopia av avhandlingen?  
(via e-post)  
JA □ NEJ □

#### 2. Lärares pedagogisk utbildning

A. Hur många universitetspoäng har Du (på ett ungefär) i  
   - Fysik  
   - Programmering  
   - Matematik  
   - Pedagogik

B. Om Du har genomgått lärarutbildning eller motsvarande. Vilken typ av pedagogisk examen har Du?

C. Har du en lärarexamen med kurser i programmering? Vilka kurser?

D. Varifrån har du fått Dina ämneskunskaper i programmering?

E. På vilken (högsta) nivå har Du studerat programmering?

#### 3. Lärares pedagogisk erfarenhet

A. Hur många år har Du undervisat i programmering?
### ATT UNDERVISA I PROGRAMMERING  
**Quest 4**

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>B.</strong> Vilka kurser i programmering har Du undervisat den senaste femårsperioden?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>C.</strong> Har Din syn på undervisning i programmering förändrats sedan Du började undervisa i ämnet?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>D.</strong> Anser Du att pedagogik är viktigt för programmeringsundervisning?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>E.</strong> Vad har påverkat Dig i valet av undervisningsmetod och val av resurser (Internet, Litteratur mm) i Din programmeringsundervisning?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| **F.** Hur ställer Du Dig till följande lärarcitat?  
"Programmering är mer matematik än vad matematikämnet i skolan är" |   |   |   |   |   |   |
| (Håller INTE med) | 1 | 2 | 3 | 4 | 5 | 6 | (Håller med) |
| **G.** Hur ställer Du Dig till följande lärarcitat?  
"Vi [lärare] lägger för mycket tid på syntax och för lite tid på struktur. När elever får en större uppgift ‘drunknar’ de i mängden av kod, istället för att dela upp den i delproblem." |   |   |   |   |   |   |
| (Håller INTE med) | 1 | 2 | 3 | 4 | 5 | 6 | (Håller med) |
| **H.** Hur ställer Du Dig till följande lärarcitat?  
"Bättre om [eleverna] kunde strukturerar problemet med papper och penna (flödesschema) och tänka igenom lösningen innan man kodar." |   |   |   |   |   |   |
| (Håller INTE med) | 1 | 2 | 3 | 4 | 5 | 6 | (Håller med) |
| **I.** Hur ställer Du Dig till följande lärarcitat?  
"Man måste hjälpa eleverna att se behovet av en strategi istället för att angripa hela problemet på en gång" |   |   |   |   |   |   |
| (Håller INTE med) | 1 | 2 | 3 | 4 | 5 | 6 | (Håller med) |
| **J.** Hur ställer Du Dig till följande lärarcitat?  
"Elever har svårt att dela upp uppgiften/problemet i mindre delar och lösa dem var och en för sig." |   |   |   |   |   |   |
| (Håller INTE med) | 1 | 2 | 3 | 4 | 5 | 6 | (Håller med) |
**ATT UNDERVISA I PROGRAMMERING**

**Quest 4**

<table>
<thead>
<tr>
<th>4. För Dig som undervisar/undervisat i Programmering A</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Anser Du att ALLA elever som börjar läsa Programmering A kan bli godkända?</td>
</tr>
<tr>
<td>JA ☐ NEJ ☐ VET INTE ☐</td>
</tr>
</tbody>
</table>

I vilken omfattning spelar följande faktorer en roll för lärandet i Programmering A?

<table>
<thead>
<tr>
<th>Faktor</th>
<th>(Lite)</th>
<th>(Mycket)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lärarens pedagogik</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Tiden som eleven arbetar på egen hand</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Elevers motivation och intresse</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Elevers förmåga att strukturera koden</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Elevers analytiska och logiska förmåga</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>(Enskilt)</th>
<th>1 2 3 4 5 6</th>
<th>(Tillsammans)</th>
</tr>
</thead>
<tbody>
<tr>
<td>B. Brukar Dina elever, i Programmering A, arbeta i huvudsak enskilt eller tillsammans (parvis eller grupper)?</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(Enkla/Mindre)</th>
<th>1 2 3 4 5 6</th>
<th>(Komplexa/Större)</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. Arbetar Dina elever företrädesvis med enkla och mindre uppgifter eller med komplexa och större programmeringsuppgifter i Programmering A?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(Teoretisk)</th>
<th>1 2 3 4 5 6</th>
<th>(Praktisk)</th>
</tr>
</thead>
<tbody>
<tr>
<td>D. Vilken typ av undervisning (teoretisk eller praktisk) anser Du är bäst för att elever skall lära sig Programmering A?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**ATT UNDERVERSA I PROGRAMMERING**

**Quest 4**

5. **För Dig som undervisar/undervisat i Programmering B**

A. **Anser Du att ALLA elever som börjar läsa Programmering B kan bli godkända?**
   - JA □
   - NEJ □
   - VET INTE □

I vilken omfattning spelar följande faktorer en roll för lärandet i Programmering B?

<table>
<thead>
<tr>
<th>Faktor</th>
<th>(Lite)</th>
<th>(Mycket)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lärarens pedagogik</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Tiden som eleven arbetar på egen hand</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Elevers motivation och intresse</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Elevers förmåga att strukturera koden</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
<tr>
<td>Elevers analytiska och logiska förmåga</td>
<td>1 2 3 4 5 6</td>
<td></td>
</tr>
</tbody>
</table>

B. **Brukar Dina elever, i Programmering B, arbeta i huvudsak enskilt eller tillsammans (parvis eller grupper)?**

<table>
<thead>
<tr>
<th>(Enskilt)</th>
<th>1 2 3 4 5 6</th>
<th>(Tillsammans)</th>
</tr>
</thead>
</table>

C. **Arbetar Dina elever företrädesvis med enkla och mindre uppgifter eller med komplexa och större programmeringsuppgifter i Programmering B?**

<table>
<thead>
<tr>
<th>(Enkla/Mindre)</th>
<th>1 2 3 4 5 6</th>
<th>(Komplexa/Större)</th>
</tr>
</thead>
</table>

D. **Vilken typ av undervisning (teoretisk eller praktisk) anser Du är bäst för att elever skall lära sig Programmering B?**

<table>
<thead>
<tr>
<th>(Teoretisk)</th>
<th>1 2 3 4 5 6</th>
<th>(Praktisk)</th>
</tr>
</thead>
</table>
**ATT UNDERVISA I PROGRAMMERING**  
**Quest 4**

<table>
<thead>
<tr>
<th>6. För Dig som undervisar/undervisat i Programmering C</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A. Anser Du att ALLA elever som börjar läsa Programmering C kan bli godkända?</strong></td>
</tr>
<tr>
<td>JA ☐ NEJ ☐ VET INTE ☐</td>
</tr>
<tr>
<td>I vilken omfattning spelar följande faktorer en roll för lärendet i Programmering C?</td>
</tr>
<tr>
<td><strong>Lärarens pedagogik</strong></td>
</tr>
<tr>
<td><strong>Tiden</strong> som eleven arbetar på egen hand</td>
</tr>
<tr>
<td><strong>Elevers motivation och intresse</strong></td>
</tr>
<tr>
<td><strong>Elevers förmåga att strukturera kod</strong></td>
</tr>
<tr>
<td><strong>Elevers analytiska och logiska förmåga</strong></td>
</tr>
</tbody>
</table>

| **B. Brukar Dina elever, i Programmering C, arbeta i huvudsak enskilt eller tillsammans (parvis eller grupper)?** |
|-----------------|----------------|
| (Enskilt) | 1 2 3 4 5 6 |
| (Tillsammans) | |

| **C. Arbetar Dina elever företrädesvis med enkla och mindre uppgifter eller med komplexa och större programmeringsuppgifter i Programmering C?** |
|-----------------|----------------|
| (Enkla/Mindre) | 1 2 3 4 5 6 |
| (Komplexa/Större) | |

| **D. Vilken typ av undervisning (teoretisk eller praktisk) anser Du är bäst för att elever skall lära sig Programmering C?** |
|-----------------|----------------|
| (Teoretisk) | 1 2 3 4 5 6 |
| (Praktisk) | |

**Stort tack för Din medverkan!**
Bibliography


Bell, T., P. Curzon, Q. Cutts, V. Dagiena, and B. Haberman (2011). Introducing students to computer science with programmes that don’t emphasise programming. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education, ITiCSE ’11*, New York, NY, USA, pp. 391–391. ACM.


BIBLIOGRAPHY


Bungum, B. (2003). Perceptions of technology education: a cross-case study of teachers realising technology as a new subject of teaching. Trondheim: Norwegian University of Science and Technology, Faculty of Science and Technology, Department of Physics.


Tenenberg, J. and S. Fincher (2007, March 7-10). Opening the door of the computer science classroom: The disciplinary commons. In SIGCSE 07 Technical Symposium on Computer Science Education, Covington, Kentucky, USA.


Papers