Malmi, Lauri; Sheard, Judy; Kinnunen, Päivi; Simon; Sinclair, Jane

Theories and Models of Emotions, Attitudes, and Self-Efficacy in the Context of Programming Education

Published in:
ICER 2020 - Proceedings of the 2020 ACM Conference on International Computing Education Research

DOI:
10.1145/3372782.3406279

Published: 10/08/2020

Document Version
Peer reviewed version

Theories and Models of Emotions, Attitudes, and Self-Efficacy in the Context of Programming Education

Lauri Malmi
Aalto University, Finland
Espoo, Finland
lauri.malmi@aalto.fi

Judy Sheard
Monash University, Australia
Melbourne, Australia
judy.sheard@monash.edu

Päivi Kinnunen
Helsinki University, Finland
Helsinki, Finland
paivi.kinnunen@helsinki.fi

Simon
University of Newcastle, Australia
Newcastle, Australia
simon@newcastle.edu.au

Jane Sinclair
University of Warwick, UK
Warwick, United Kingdom
j.e.sinclair@warwick.ac.uk

ABSTRACT

Research into the relationship between learning computing and students' attitudes, beliefs, and emotions often builds on theoretical frameworks from the social sciences in order to understand how these factors influence, for example, students' motivation, study practices, and learning results. In this paper we explore the computing education research literature to identify new theoretical constructs that have emerged from this research. We focus on empirical work in programming education that extends or adapts theories or instruments from the social sciences or that independently develops theories specific to programming. From an initial data set of more than 3800 papers published in the years 2010–2019, we identify 50 papers that present a range of domain-specific theoretical constructs addressing emotions, affect, beliefs, attitudes, and self-efficacy. They include 11 validated instruments and a number of statistical models, but also grounded theories and pedagogical models. We summarize the main results of many of these constructs and provide references for all of them. We also investigate how these constructs have informed further research by analysing over 850 papers that cite these 50 papers. We categorize the ways that theories can inform further research, and give examples of papers in each of these categories. Our findings indicate that among these categories, instruments have been most widely used in further research, thus affirming their value in the field.

CCS CONCEPTS

• **Social and professional topics → Computing education**.

KEYWORDS

computing education, theory, theoretical construct, emotion, affect, attitude, self-efficacy, belief, programming, research, instrument

ACM Reference Format:
Lauri Malmi, Judy Sheard, Päivi Kinnunen, Simon, and Jane Sinclair. 2020. Theories and Models of Emotions, Attitudes, and Self-Efficacy in the Context of Programming Education. In *Proceedings of the 2020 International Computing Education Research Conference (ICER '20), August 10–12, 2020, Virtual Event, New Zealand.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3372782.3406279

1 INTRODUCTION

How best to support our students' learning? There is no straightforward answer to this question, as the learning process is influenced by a complex set of interrelated factors, such as the academic and social capital that students bring with them as they start their studies [10]; students' beliefs, intentions, and orientations regarding specific subject matter; students' cognitive maturity; the nature of the subject matter; the physical, online, and social learning environments; and the pedagogical choices made by teachers and universities. Researchers and teachers have tried to understand the complexities of learning for decades. For instance, self-regulated learning (SRL) is a comprehensive theory that aims to describe and explain aspects related to the learning process, and incorporates several models [89] that entail components relating to both cognitive and motivational self-regulation of learning [8, 112]. A number of papers discuss the intertwined nature of emotions, motivation, self-efficacy, learning outcomes, and other concepts covered by different versions of the SRL framework [2, 77, 91, 112].

Programming education is no exception here. Widely reported difficulties that students face in learning programming have prompted many researchers to investigate relationships between attitudes, other non-cognitive factors, and students' motivation [16, 81, 111], self-efficacy or beliefs about one's skills [24, 99, 108], study practices [12, 18, 110], and learning results [17, 68, 80]. Lishinski and Yadav [67] provide a comprehensive summary of the research literature on students' motivation, attitudes, and other non-cognitive factors. Other authors discuss the relevant theories [79, 95]. These studies provide evidence of the important role that programming students' internal factors such as self-beliefs, academic emotions, and attitudes play in their learning outcomes and experiences. The findings are further supported by the recent comprehensive review of programming education research by Luxton-Reilly et al., which includes this challenge in its conclusion:

> "Although introductory programming courses are considered to cause significant levels of student anxiety and frustration, and student engagement levels in computing are benchmarked as among the lowest of any discipline, there are very few publications focusing on the student experience, particularly with respect to pastoral care. This is an important area that appears to be under-explored, and we feel that the community would benefit from further work in this area." [70, p85]

We propose, therefore, that an improved understanding of the complex factors influencing learning related to students' internal factors, such as beliefs, emotions, and attitudes, would help us to develop more effective interventions to influence students' perceptions about learning programming, and thus to improve pass rates and learning results.

Through exploring research in this area, we aim to build a theoretical understanding of students' experience in learning programming associated with their internal factors. Our work is motivated by our previous work [76] as well as the work by Nelson and Ko [84], who also propose that computing education researchers should undertake efforts to develop their own *domain-specific theories* of learning computing.

In a recent review [75], we explored domain-specific theories in all areas of computing education research. We identified several works [23, 32, 82, 83] presenting theoretical developments related to emotion, beliefs, attitudes, and self-efficacy. In that work, we applied a rather broad definition of theoretical constructs, which remains appropriate for the current work. The earlier work limited the search to two journals – ACM Transactions of Computing Education (TOCE) and Computer Science Education (CSEd) – and one conference – International Computing Education Research Conference (ICER). The analysis covered the years 2005–2015, so it did not include more recent developments. Here, we complement this work by reviewing more recent work in these venues. We also expand the scope of our work by including many more journals that publish research in computing education. The Scopus database was used for the survey, because it covers a very large pool of scientific journals and has good opportunities to limit the search using metadata. We thus aimed to build a more comprehensive picture of theoretical research in this area that would support future research, and also teachers who base their pedagogical innovations on research evidence. Using the definition by Grant and Booth [33], our current study can be considered a mapping review.

A recent survey by Szabo et al. [103] focuses on the use of learning theories used in computing education but developed in other areas. Our work complements this by focusing on theories developed within computing education.

2 BACKGROUND AND RESEARCH QUESTIONS

What is a theory? This question addresses the ontological nature of theory, and the answers reveal our research community's views on the relevant components of a theory, the kinds of question that we can or should ask, the kinds of representation of theory that are accepted, and what is regarded as a contribution to current knowledge [41]. Members of the research community are likely to give different answers to the question because we have multiple views and definitions of theory. Even though the authors of empirical research papers do not always explicate their viewpoint of theory, we can often deduce it from their choices of research questions, research designs, and the type of the contribution made by the paper.

Why do we need theories? Theories are at the very heart of a mature scientific discipline and carefully executed research. They can suggest which phenomena are worth studying in the first place, and may guide the choice of research methods and how observations are organized, analyzed, and interpreted [86]. Theories help us to see the complexity of seemingly mundane situations and to conceptualize complex problems [102]. However, there is a wealth of types of theory that serve different purposes. As Gregor [41] explains, theory can be used to analyze phenomena, to explain them, to make predictions about them, and to design subsequent actions. For more discussion on the roles of theories, see Malmi et al. [76] and Szabo et al. [103].

In this study, we seek to build a holistic understanding of the kind of research carried out into students' learning experience that goes beyond describing successful innovations, individual case studies, or simple statistical testing, and aims to contribute to a more comprehensive picture of the investigated phenomena by building theories, models, or instruments. Within the broad area of computing education, we limit our domain to programming education, and define our core research questions as follows:

- RQ1 What theoretical constructs, including instruments, have been developed in computing education research (CER) that address emotions, affects, attitudes, beliefs, or self-efficacy in teaching and learning programming, and where have they been published?
- RQ2 How have these constructs been developed?
- RQ3 What types of construct have been developed?
- RQ4 How have they been used to inform further research?

The terms in which we are interested are used in varying ways by different authors. In order to provide some sort of common ground for further discussion, we now provide some generic definitions.

- Emotion / academic emotion / affect: "the domain of academic emotions would include students' achievement emotions experienced in school or university settings, but goes beyond emotions relating to success and failure by also covering, for example, emotions relating to instruction or to the process of studying" [90, p92].
- Attitude: "how much we like (and/or dislike) something" [74, p xv]. The person's attitude varies based on several factors, such as certainty, extremity, valence, and functions that attitudes have for the person.
- Belief: a broad concept that can refer to students' beliefs about their own ability (see self-efficacy below) or their understandings and perceptions of the social and physical/online learning environment, the subject being studied, and/or possible career paths.
- Self-efficacy: "people's beliefs about their capabilities to produce designated levels of performance that exercise influence over events that affect their lives" [2, p1].

3 METHOD

3.1 Theoretical Constructs (TCs)

As a young field, computing education research does not have an established terminology related to theoretical frameworks, and searching the literature using only terms such as 'theory', 'model', and 'framework' would be unlikely to give us a broad picture of domain-specific theories. As one of our starting points was our recent survey, we decided to use the same broad definition of *theoretical construct* (TC) as "a theory, model, framework, or instrument developed through application of some rigorous empirical or theoretical approach" [75, p188].

Using this as our guideline, we included in our search statistical models, such as regression, path analysis, factor analysis, structural equation modelling, and clustering, which have been generated from the data to explain the relationships between identified concepts. We also included validated instruments for measuring particular theory-based concepts, because they can be valuable tools for building TCs, and we were interested not only in the original publication of TCs but also in further developments in papers that cite the original papers. We excluded simple hypothesis tests unless they were combined into a greater whole such as a path model.

We also included qualitative data-driven categorizations that seek to generate a higher-level abstract description of the data, such as grounded theories. Simple lists of qualitative categories were excluded, unless there was a clear structural discussion of the relationships between the categories. Finally, we included other types of explanatory model such as taxonomies, typologies, figures, and formulas, which build abstraction from data or are derived from other theoretical frameworks and adapted to address a relevant domain, or were developed in terms of logical argumentation.

3.2 Data Sources

We used two approaches to identify papers in which we might identify development of TCs. It seemed unlikely that search terms such as 'theory' and 'model' would be helpful because many papers would not use these terms when presenting their results. Therefore, our strategy was first to identify papers that deal with 'emotions', 'affect', 'attitudes', 'beliefs', or 'self-efficacy' in the context of programming education, and then to manually examine them to determine whether they present the development of a relevant TC. We used these concepts as search terms for two reasons: first, they match with self-regulated learning theory as abstractions of different internal aspects of students' learning experience; second, they match the terms used in our prior work [75] to describe one category that we identified in our data-driven classification of TCs, and thus are likely to reflect the terms that we found in the literature. We added 'affect' to the list as one more abstract term that we had seen in the literature. However, we did not include a variety of possible synonyms or instances of emotions or attitudes, such as 'feeling', 'anger', 'frustration', 'joy', etc, as such a list would always be incomplete, and we would normally expect to see the more abstract terms in titles and abstracts.

In our prior search of the TOCE and CSEd journals and the ICER conference for 2005–2015 [75], we had already identified several papers in our current domain of interest. We took these findings as our starting point and then extended the search span to 2019. In the years 2016–2019 there were 111 papers published in ICER, 95 papers in TOCE, and 57 papers in CSEd. We added one further conference, Learning Analytics & Knowledge (LAK), as it seems a promising forum for the publication of empirical research in this area. We added all 457 papers presented at that conference from 2011, its first year, to 2019, and performed a complete search of all these papers for relevant content.

A complete search among other venues was beyond our resources. We conducted a number of pilot searches using Google Scholar with different phrases such as 'emotions', 'programming', and 'education', but the results were far too numerous for further analysis. Therefore, we decided to use the Scopus search engine, which gives us more scope to specify various types of metadata in the search. Trial searches with small numbers of hits seemed promising, so we performed the following Scopus search:

```
(TITLE-ABS-KEY (emotion) OR TITLE-ABS-KEY (affect) OR TITLE-
ABS-KEY (attitude) OR TITLE-ABS-KEY (belief)) AND TITLE-ABS-
KEY (programming) AND DOCTYPE (ar) AND PUBYEAR > 2009 AND
PUBYEAR < 2020 AND (LIMIT-TO (PUBSTAGE, "final")) AND
(LIMIT-TO (SUBJAREA, "ENGI") OR LIMIT-TO (SUBJAREA, "COMP")
OR LIMIT-TO (SUBJAREA, "PSYC")) AND (LIMIT-TO (LANGUAGE,
"English"))
```

That is, we searched for journal articles published in the years 2010–2019, which have the relevant terms in their title, abstract or keywords, which lie in the subject areas of engineering, computing, or psychology, and which are written in English. This search resulted in 3007 documents. We subsequently conducted a further search using similar metadata but limiting the search term only to 'self-efficacy'. This resulted in 87 new documents. Our total data pool thus comprised 3814 papers.

3.3 Analysis Methods

Rapid screening of the Scopus search papers was carried out by two researchers, each of whom screened half of the papers by looking first at the title and journal to assess whether they fall into a relevant domain. For example, many papers were found to concern health care, and were readily excluded. Papers that were not excluded on this basis were further screened by reading the abstract, and finally 252 papers remained for detailed screening, in which the whole paper was searched for signs of new theoretical constructs. This phase was carried out by two researchers who both identified potential candidates for novel TCs. For each paper in which one researcher identified a potential TC, the other researcher read the same paper. If they agreed, the paper was included. If they disagreed or were unsure, a third researcher looked at the paper, which was then included or excluded according to the majority decision.

This process, along with the complete search of papers from TOCE, CSEd, ICER, and LAK, finally identified 50 papers as *source papers* that present the development of some form of TC. Each source paper was then further analyzed based on the following dimensions: 1) whether the TC was a novel instrument, such as a validated survey, or some other form of TC; 2) what aspects were concerned in the TC (belief, self-efficacy, affect/feeling, emotion and attitude); and 3) what methodological approach was used in developing the construct. Two researchers reviewed these aspects for all papers, and in case of disagreement negotiated for consensus.

3.4 Use of Theoretical Constructs

We were also interested to find out whether the theoretical constructs had been used to inform further research, how they had been used, and whether the constructs had been extended. We used Google Scholar to identify papers citing each source paper, hereafter called *citation papers*. For each source paper, we downloaded all of the citation papers that we could access, unless there were too many for our resources. For source papers with more than 40 citation papers, we downloaded the most recent citation papers, starting from the most recent year and stepping back one year at a time until we had approximately 30 papers. This practice was motivated by the assumption that any earlier developments, not presented in the most recent citations, would probably themselves be cited in the most recent papers, and would thus give us hints for further snowballing. Using this process, a total of 868 citation papers were downloaded for citation analysis.

Using inductive content analysis, we devised a classification scheme with four main categories of use, each with subcategories, as shown in Table 1. It should be noted that in our classification, categories D1 and D2 were frequently used in cases where it was impossible to determine the main reason for citing the source paper, or where the source paper was cited for some reason other than the TC introduced in that paper. While we expected to find examples of each category of use, ultimately we found none in category A4.

Table 1: Categories of use of TCs: D – description; A – application; C – construction; V – validation

Code	Description
D1	Cited in related work – brief mention, no explanation
D2	Cited in related work – one-statement explanation (roughly)
D3	Discussed in related work – relevance to the current work is explained
D4	Described and critiqued
A1	Used as a framework to scope a study
A2	Used to develop a data collection instrument
A3	Used as a framework for data analysis
A4	Used to predict results of a study
A5	Used to interpret/compare/explain results of a study
A6	Used to design a new pedagogical method
A7	Used as an instrument in a study
C1	Modified and/or extended existing construct for use in a new context
C2	Developed new construct from existing construct and empirical work
C3	Developed new construct from existing construct and argumentation
V1	Used in a test to improve or discount an existing/new construct
V2	Tested the construct in a new context

4 RESULTS

Here we present results to address our four research questions about TCs developed in CER.

4.1 What Theoretical Constructs Have Been Developed and Where Have They Been Published? (RQ1)

Our analysis of the literature found 50 papers reporting the development of TCs that address one or more aspects of emotion, affect, attitude, belief, or self-efficacy in teaching and learning programming. Table 2 shows the numbers of papers found each year from 2010 to 2019 from the manual search of CSEd, ICER, LAK, TOCE, and the keyword search using Scopus. The growing interest in the area is shown by the distribution of the publications over the ten-year period, with 38% of the papers published in the last two years of the search period. A complete list of the corresponding publications is in Table 5.

Table 2: Numbers of papers reporting the development of an emotion, affect, attitude, belief, or self-efficacy TC in teaching and learning programming, 2010–2019

Year	Manual search of CSEd, ICER, LAK, TOCE	Keyword search of Scopus	Total papers
2010	1	2	3
2011	2	1	3
2012	1	0	1
2013	1	4	5
2014	1	0	1
2015	2	3	5
2016	4	5	9
2017	2	2	4
2018	1	9	10
2019	2	7	9
Total	17	33	50

The 50 new TCs were reported in 30 different venues, indicating a broad interest in investigating emotions, affect, attitudes, beliefs, and self-efficacy in the context of teaching and learning programming. Eleven TCs that are instruments were found in eight different venues. We take a particular interest in instruments, such as validated surveys, because of their high potential for use by researchers other than those who developed them.

Table 3 shows the total number of papers in each venue. Although the TCs were spread over a large number of venues, 34% of them, including 36% of the instruments, were found in papers from the four venues where a manual search was conducted. This shows a concentration of work in the key computing computing education research venues.

4.2 How Were the Theoretical Constructs Developed? (RQ2)

We found many different approaches used to develop the TCs, and different methods, sometimes two or three in combination, within each approach. Table 4 shows the frequencies of methods used.

Most TCs were developed from literature or grounded studies. We found few examples of adaptations or extensions of other constructs. One example of adaptation is the Zones of Proximal Flow, a pedagogical framework developed by Basawapatna et al. [5] that

Table 3: Numbers of papers reporting the development of an emotion, affect, attitude, belief, or self-efficacy TC in teaching and learning programming in each venue, and the type of search conducted

Venue	Total papers	Search
ICER	7	manual
TOCE	5	manual
CSEd	4	manual
Computers and Education	3	keyword
Educational Computing Research	3	keyword
IEEE Transactions on Education	2	keyword
Informatics in Education	2	keyword
Science Education and Technology	2	keyword
one other conference and 21 other journals with a single paper each	22	keyword
Total	50	

Table 4: Methods used in the development of TCs concerning emotion, affect, attitude, belief, or self-efficacy in teaching and learning programming; some papers used several approaches, all of which are listed in the table

Approach	Source papers
regression	[16, 17, 31, 42, 46, 47, 61, 72, 80, 87, 98, 100]
path analysis	[12, 13, 19, 64, 65, 68, 69, 96]
exploratory factor analysis (EFA)	[14, 17, 24, 28, 52, 59, 61, 105]
confirmatory factor analysis (CFA)	[14, 28, 51, 52, 54, 59, 99]
structural equation modelling (SEM)	[18, 51, 58, 110, 111]
literature analysis	[54, 72, 99, 111]
grounded theory	[25, 39, 48, 49]
partial least squares (PLS)	[18, 58, 66]
thematic analysis	[81, 101]
machine learning	[45, 80]
content analysis	[57]
implicative statistics	[1]
principal components analysis (PCA)	[31]
case study analysis	[85]
other (argumentation, factor analysis, linear mixed-effects, other empirical)	[5, 9, 38, 97, 108]

integrates Vygotsky's Zone of Proximal Development theory with Csikszentmihalyi's ideas about Flow. An example of an extension to a TC is the case study reported by Nikula et al. [85], whose Three-Motivator Theory for course success extends the two-factor Herzberg's Motivation-Hygiene Theory.

Most approaches (86%) involved some form of quantitative method. The most common quantitative approach was a form of regression including partial least squares analysis (30%). The next most common approach was a form of structural equation modelling including path analysis (26%). We found few cases (16%) where TCs had been developed through qualitative approaches. The most common qualitative method used was grounded theory.

The developments of instruments all involved various forms of factor analysis. The most common approach was a literature search or an exploratory factor analysis in combination with a confirmatory factor analysis. Three instruments were adapted from existing instruments [17, 24, 28].

4.3 What Types of Construct Have Been Developed? (RQ3)

As part of our analysis we classified the TCs according to the particular areas of emotion, affect, attitude, belief, or self-efficacy that they addressed. Table 5 lists the numbers of papers reporting on each area. More than half the papers (56%) addressed two or more areas. Common combinations were beliefs & attitudes, emotions & attitudes, self-efficacy & attitudes, and self-efficacy & emotions. Almost half of the instruments (45%) involved self-efficacy.

Space limitations prevent us from listing and describing all of the TCs that we identified. Instead, we have selected a number of examples to highlight some of our findings.

4.3.1 Quantitative Models. Many researchers have investigated the role of *self-efficacy* in learning performance. Lishinski et al. [69] presented a path analysis model of the interaction of self-efficacy, intrinsic and extrinsic goal orientations, and metacognitive strategies, and demonstrated the impact on student performance in a CS1 course. A similar method was used by Cernusca and Price [13], whose path analysis model showed that self-efficacy, perceived engagement, and perceived difficulty are significant predictors of students' final performance in CS1. Further evidence of the role of self-efficacy was provided by Kuo et al. [58], whose model indicated that goal-setting and self-efficacy in learning to program positively influence learning performance while self-satisfaction serves as a feedback for learners to evaluate their performance. Moreover, goal commitment and self-efficacy in learning to program lead to successful learning, generating a high level of satisfaction that further heightens students' sense of commitment to learning.

Other aspects of the role of self-efficacy have been explored by Gurer et al. [42], who investigated pre-service computer science teachers' Attitudes Towards Computer Programming (ATCP). The constructed regression model indicated that achievement in computer programming courses, computer programming self-efficacy, and perceived learning were significant predictors of ATCP.

Several researchers have investigated how to improve self-efficacy itself. Law et al. [61] showed that individual attitude and expectation, challenging goals, social pressure, and competition have a significant and positive relationship with efficacy in programming courses. The regression model of Nugent et al. [87] demonstrated that learning resources can support self-efficacy by showing that wearable e-textiles for upper elementary school students had a significant positive relationship with programming self-efficacy.

Research on *emotions* has mainly explored the modelling of various forms of scaffolding, as illustrated by a few examples.

Cabada et al. [12] analyzed the impact of the web-based environment Java Sensei for learning Java programming. The system provides students with adapted and individualized programming

Table 5: Numbers and citations of papers reporting emotion, affect, attitude, belief, or self-efficacy TCs in teaching and learning programming; several papers address two or more areas

Focus of TC	Total papers	Source papers
attitudes	28	[1, 12, 14, 16, 18, 25, 28, 38, 42, 45, 46, 51, 52, 57, 59, 61, 62, 64, 65, 72, 80, 81, 85, 96, 99, 108, 110, 111]
emotions	21	[1, 9, 12, 17, 19, 38, 39, 48, 49, 51, 58, 61, 64–66, 68, 80, 81, 97, 100, 101]
self-efficacy	21	[1, 13, 18, 24, 42, 48, 49, 51, 54, 58, 61, 69, 72, 87, 96, 99, 100, 105, 108, 110, 111]
beliefs	7	[13, 17, 25, 31, 52, 59, 99]
affect	6	[5, 72, 81, 85, 99, 111]

instruction by using a recommender and mining system, an affect recognizer, a sentiment analyzer, and an authoring tool. Their regression model concerning use of the system indicated that perceived ease of use has an impact on perceived enjoyment, which then impacts both attitude to the system and intention to use it. Another automated scaffolding system was used by Bosch and D'Mello [9], who also analyzed students' emotional states. Students first undertook scaffolded programming tasks with automatic assessment and hints for incorrect submissions; then they moved on to more challenging exercises without hints or explanations. Their emotional states were recorded retrospectively by collaborative analysis of video recordings of the activity. The authors analyzed the transitions between various affective states, building a complex model of students' transitions relating to their activities during the process.

Liew et al. [65] built on emotional response theory [44] to investigate whether the well-known relationship between instructor's enthusiasm and students' learning experience transferred to a multimedia learning environment with an enthusiastic pedagogical agent. They compared the effect of enthusiastic and neutral agents and found that the former significantly enhanced students' emotions, intrinsic motivation, affective perceptions, and cognitive outcome. They built path models, which "demonstrated the full mediating role of positive emotion in affecting learners' approach–avoidance behaviors, such as intrinsic motivation, perception of the virtual learning environment, and perception of the pedagogical agent".

Attitude is a broad concept with some connection to personality. Karimi et al. [47] used the International Personality Item Pool[1] to analyze the influence of prior experience, attitude, self-assessed survey performance, and personality on students' performance in computer programming. Several factors, including prior programming experience, attitude towards programming, academic performance, and personality factors such as openness to experience, conscientiousness, extroversion, and agreeableness had a positive effect on programming performance. Conversely, negative effects were found with facets of neuroticism and with one facet of openness to experience: high imagination (a possibly surprising result). An interesting continuation study [46] looked further at programming working styles. Using the classification proposed by Cox and Fisher [21] and Vessey [107], they built a complex model of the influence of various factors on programming working styles. An interesting finding was that openness to experience had a positive association with breadth-first style, that is, exploring alternatives in debugging instead of the depth-first approach of immediately trying the first hypothesis. On the other hand, conscientiousness had a positive association with the depth-first style.

4.3.2 Qualitative Constructs. Kinnunen and Simon [48] investigated the emotional experiences of students working on programming assignments during a CS1 course. They developed a partial theory identifying six stages whose dimensions capture the variation in emotional experiences, for instance when encountering and dealing with difficulties. They extended the analysis to provide a descriptive model of how computer science majors build self-efficacy perceptions, reported via four narratives that present either positive or negative development in students' self-efficacy [49]. In an interesting quantitative continuation, Lishinski et al. [68] built a path model of how students' emotional reactions correlate with their performance on programming projects, with the valence of the emotion dictating the direction of the correlation. Self-efficacy and three categories, happy/proud, frustrated/annoyed, inadequate/disappointed, were based on the work of Kinnunen and Simon. Path models indicated that with the exception of feelings of pride, students' emotional reactions to programming projects had significant effects on the outcomes of future programming projects.

The previous works focus on students learning introductory programming. In contrast, Graziotin et al. [39] explored learning in advanced programming and software engineering contexts. Using the valence-arousal-dominance framework, they collected data with the Self-Assessment Manikin (SAM) instrument, which measures these aspects of the framework [11]. Valence (or pleasure) reflects the attractiveness/adverseness of an event, object, or situation; arousal represents the intensity of emotional activation [60]; and dominance (or control) represents the perception that one's skills are higher than needed for the task [11]. The result was a linear model of how valence, arousal, and dominance affect self-assessed productivity. This work was augmented with a grounded explanatory theory of the impact of affects on software development performance, describing how different types of event trigger either positive or negative affects, some of which earn importance and priority to become attractors that, together with affect, influence the focus of work and performance. This is a complex theory which is based on substantial previous research [35–38].

Finally, we give an example of a pedagogical model. Lee et al. [62] combined Kolb's experiential learning (ELT) [50] and a modified version of Self-Regulated Learning Theory (SRL) [89] to produce a model that they call Self-Directed Regulated Learning (SDRL). They explain that "to create a positive learner-directed learning experience and provide optimal levels of learner control ... ELT was used as a guideline for the domain-knowledge content with the four learning modes. SDRL was used as a guideline for the

[1] https://ipip.ori.org/

meta-knowledge (i.e., study skills) content that was integrated and contextualized into the model. This model caters to learner control with the consideration of freedoms of pace, content, and media" [62, p222].

4.3.3 Instruments. We provide examples of three instruments that focus on attitudes. Dorn and Tew [28] developed the computing attitudes survey (CAS), an extension of the Colorado Learning Attitudes about Science Survey, to measure novice to expert attitude shifts in the nature of knowledge and problem solving in computer science. Studying pair programming, Chen and Rea [17] modified and extended the Attitudes Towards Mathematics Instruction instrument (ATMI) [104] for use in computing education. They identified four main factors related to pair programming: confidence – visceral reaction; confidence – perceived ability; value; and motivation. Moreover, they developed a hierarchical linear regression model which indicates that the key factors of performance in pair programming tasks include enhanced perceived expertise and enhanced sense of accomplishment. To measuring Turkish university STEM students' attitudes towards computer programming, Cetin and Ozden [14] developed a scale that includes three dimensions: 'affection', cognition, and behavior.

Several researchers have built instruments for measuring self-efficacy. Tsai et al. [105] developed the Computer Programming Self-Efficacy Scale (CPSES), based on students studying natural science, social science, and engineering in Taiwan, and concluded that their scale works for all students above middle school levels. The scale covers five subscales: logical thinking, algorithm, debugging, control, and cooperation. The study also confirmed a positive correlation between computer programming experience and computer programming self-efficacy. Another scale covering self-efficacy in learning programming was developed by Kong et al. [51], who conceptualized a programming empowerment instrument for primary school students that comprises four components: meaningfulness, impact, creative self-efficacy, and programming self-efficacy. They also built a structural equation model, which indicated that students with greater interest in programming perceived it as more meaningful and allowing them to make more impact with it. Such students also had greater creative self-efficacy and programming self-efficacy.

Two other instruments in the same area are the Positive Youth Programming Development instrument (PYPD) [54], which measures students' positive qualities while learning programming with App Inventor, with three components: programming confidence, programming connection, and programming contribution; and the scale developed by Scott and Ghinea [99], which measures students' self-beliefs within the introductory programming context.

Narrower scales include a scale for measuring self-efficacy in an introductory algorithms course [24], a scale to measure students' subjective perceptions of their difficulties in learning recursion [59], and a survey of factors motivating students to participate in activities with tangible programming technologies [31].

Finally, Kong et al. [52] developed an instrument measuring parents' perceptions of programming education among P-12 schools.

4.4 How Have the Theoretical Constructs Been Used to Inform Further Research? (RQ4)

Our analysis of the 868 citation papers determined what type of use each citing paper made of the theoretical construct. We found that most citation papers (90%) made no use of the TC beyond a brief description or critique. Only 8% of the papers reported an application of the TC, and fewer reported construction (3%) or validation (2%). Overall, only 54% of the source papers had one or more citation papers where the use was application, construction, or validation. Table 6 shows the TCs that have been used in further work and the type of use reported. A code in the column showing the type of use indicates that at least one citation paper used the TC in the manner indicated. Note that we sometimes identified different uses of a TC in the same citation paper.

The table shows that we found examples of almost all types of use. The most common use was application, where a construct was used, for example, to scope a study, develop a data collection instrument, or interpret results. There were fewer examples where a TC had been modified or extended to develop a new construct, and very few examples of construct validation. Below, we present examples of ways that some of the TCs we identified have informed further research. We focus on a few cases where building on previous work has led to considerable progress. We have added information on which TC use cases they demonstrate, according to the categories in Table 1.

4.4.1 Examples of Instrument Use. Instruments were most often used for data collection (code A7). For example, Dawson et al. [26] compared student pass rates, satisfaction, and attitudes after they had developed and delivered a CS0.5 introductory programming course for non-CS majors. They used the CAS instrument [28] to measure students' attitudes following the course, and found a good match with the attitudes of students taking the traditional CS1.

Often, parts of an instrument were used to build a new survey for data collection (code A2). For example, Bockmon et al. [7] used the Computing Attitudes Survey [28] as one starting point when building and validating their own instrument, which added questions on gender issues and computing utility to the original survey [6]. In other cases, researchers used only the questions from the original instrument that they deemed relevant for their current study [88].

In a few cases, the results of a new study were contrasted with those using a different instrument (code A5). For example, Duran et al. [30] compared their own findings on comprehending programming concepts with those of Danielsiek et al. [24], who explored students' algorithm skills and pseudocode writing and tracing.

As mentioned in section 4.3.3, Kong et al. have developed instruments to measure programming self-efficacy, as well as parents' perceptions on programming, in the context of K-12 programming education. For this research they used and further developed instruments to build a structural equation model on programming empowerment and students' interest and attitude toward collaboration in programming [51, 52, 54]. They also developed a new instrument to measure digital empowerment of primary school students [56]. Their work, which was also based on interest-driven creator theory [15], explored in detail how the interest loop (triggering interest, immersing interest, and extending interest) influences students' robotic creations [53]. This work resulted in a structural equation

Table 6: Source papers of TCs concerning emotion, affect, attitude, belief, or self-efficacy that have been used in further work, their total citations (how many were analyzed, if not all), and the type of use reported (A: application, C: construction, V: validation); see Table 1 for definitions

Year	Source paper	Citations (analyzed)	Types of use
2010	[61]	297 (47)	A2, A7
2010	[100]	71 (48)	A5, C2
2010	[48]	59 (31)	A2
2013	[5]	55	A1, A3, A5, A6, C2, C3, V1
2011	[85]	55 (28)	A5
2012	[49]	54 (31)	A1
2016	[69]	52 (32)	A5
2015	[38]	49	A1, A3, C2, C3, V1, V2
2014	[98]	45 (34)	A1, A2, A5, C1, C2
2015	[28]	32	A2, A7
2016	[97]	29	A5, A6, C1, C3, V1
2013	[98]	27	A2, A7
2016	[71]	23	A5, V2
2015	[14]	22	A7, C2
2019	[105]	19	A2, A7
2018	[51]	19	C2
2013	[66]	17	C2
2017	[24]	8	A5, A7, V2
2019	[18]	8	C1
2018	[110]	7	C2
2018	[31]	6	C2
2018	[110]	2	A1, A5
2013	[9]	1	A7, C2

model describing the relation of these factors in the context of using programmable robots in primary education. Furthermore, they investigated computational thinking (CT) and computational identity (CI) among primary school students, developing both a CT perspectives instrument and a CI instrument for this context [55].

Another interesting example is the use of the computer programming attitude scale (CPAS) of Cetin and Ozden [14]. Gurer et al. [42] used CPAS in research that investigated the factors affecting pre-service computer science teachers' Attitudes Towards Computer Programming (ATCP). Their main findings were that participants' ATCP had significant correlations with their achievement in programming courses, programming self-efficacy, and perceived learning. They built a regression model which indicates that these three variables were significant predictors of teachers' ATCP. Another study, by Mahatanankoon and Sikolia [73], investigated the factors behind student retention in an IT major, studying the effect of harmonious passion and obsessive passion [106] and grit [29] on computer programming attitude (i.e., coding affect, cognition, and behavior), which in turn can affect student retention. Using these scales and CPAS as data collection instruments, they built a path model which showed that harmonious passion leads to positive coding affect and cognition, while obsessive passion has a negative impact on coding affect but contributes significantly to coding behavior. The paper finds that "passion and grit provide sufficient predictability for computer programming attitude and retention."

These examples demonstrate the power of using instruments in building new models to describe the complex relationships among various factors in learning to program (code C2).

4.4.2 Other Examples of Use of TCs. Basawapatna et al. [5] presented a novel pedagogical framework, entitled the zones of proximal flow (ZPF), which integrates Vygotsky's zone of proximal development theory [109] with Csikszentmihalyi's ideas about flow [22]. Essentially, they state that students should aim at flow that is "an ideal condition for learning" and pedagogy should guide them to work in their zone of proximal development, which lies between the extremes of boredom and anxiety. This concept has been extensively used and developed in further work. The authors developed this theory framework in the context of the scalable game design project [93], where children were encouraged to develop games using end-user programming tools. The ZPF framework has informed further pedagogical development in their work, including the construction of a tool for the teacher to monitor students' progress and set them individual goals [4, 94] (code A6). While measuring flow is not straightforward, they did develop a 'retention of flow' method [92] to reflect participants' activity during a computer science education week activity (codes A1, A3). Their further analysis of ZPF tutorial [3], in the context of analyzing data from two games developed at 'hour of code', confirmed that the ZPF-based tutorial (code C2) led to better student retention than a sophisticated previous tutorial (code V1).

ZPF has been further applied to learner modelling in a different context, primary school mathematics. Clement [20] designed an intelligent tutoring system which exploits machine learning and cognitive sciences to automatically personalize activities and keep students motivated by keeping them in flow or in their zone of proximal development (codes C1-C3), while still allowing them to make choices about contextual features or pedagogical content.

In the context of larger programming projects and software engineering, we have mentioned the TCs developed by Graziotin and several others [38, 39]. While their further work falls in the area of professional programming and programming communities, two papers are worth mentioning here. Graziotin et al. [34] apply the scale of positive and negative experience instrument (SPANE) [27, 63], developed in psychology to measure happiness, to analyze the feelings of more than 2000 participants whose contact information was available in GitHub Archive[2]. The scale has been validated in many large-scale studies, but we are not aware of studies that have used the instrument in a programming education context. Another interesting paper is Graziotin et al. [40], in which the authors note that "SE lacks in theoretical background of affects and guidelines for using psychology". They write a comprehensive literature review on affect theory and give guidelines for conducting psychoempirical research in software engineering, which are also relevant for studies in programming education.

Graziotin et al. [38] focused on individual software developers. Hedberg Griffith and Nguyen [43], building on the same work, focused on team projects. Using similar data collection and the

[2] https://githubarchive.org

same linear model, they confirmed the findings of Graziotin et al. [38] concerning the role of valence on performance (code V2).

5 DISCUSSION

A holistic understanding of learning programming is needed if we are to design and support the learning process in meaningful ways. As a means to contribute to this goal, we have focused on components of learning relating to students' internal factors. Taking our prior work [75] as a starting point, we surveyed a wide pool of CER literature to identify papers that demonstrate development of new TCs related to these internal factors in programming education. This area has gained considerable interest, as evidenced by our finding that 74% of the 50 identified constructs were published during the past five years. Further, since our analysis in this paper concerns only publications that deal with these factors *and* present some novel theoretical construct, the total number of publications addressing these factors is much higher. It is significant that these TCs have been published in a wide variety of venues. Only 32% of them were published in TOCE, CSEd and ICER, which we initially expected to be the main venues for publishing this kind of research. The result encourages us to follow the literature more widely.

We found that it was not feasible to use Google Scholar's full-text searches. Instead we needed database searches that allow the incorporation of appropriate metadata in the search query. It is also interesting that a large share of the publication venues are not visible in the ACM Digital library or IEEE Xplore, and thus the Scopus database proved to be a fruitful tool for our exploration of the literature.

We found that many different approaches have been used to develop these TCs. Very clearly the dominant approaches were quantitative, resulting in numerous statistical models. We found only a handful of qualitative approaches, mainly grounded theories. There is a striking difference between these findings and the findings in our prior work [75], which explored computing education literature published in TOCE, CSEd, and ICER in the years 2005–2015, and identified theoretical constructs using the same criteria that we used in the current work. For example, more than a third of the 65 TCs in the earlier results were based on phenomenographical research or grounded theories, while in the current data pool of 50 TCs we found only four papers using grounded theory and none using phenomenography. We do not consider it plausible that this difference can be explained by the difference in the analyzed publication years (2005–2015 vs 2010–2019). Neither do we expect that the result follows from narrowing the domain focus, from any TCs in computing education research to TCs in programming education that address emotions, affect, attitudes, beliefs, and self-efficacy (the earlier research having found only four TCs addressing these factors). There seems to be no obvious reason why these factors would be researched mainly using quantitative approaches. We speculate that different journals have different policies concerning the type of research that they publish. There might also be stronger emphasis on quantitative methods in research communities outside the ACM SIGCSE research community, which publishes heavily in TOCE, CSEd, and ICER.

We found a substantial number of instruments being developed in this area, and our findings clearly complement the recent survey of instruments by Margulieux et al. [78]. Naturally, the publications also reported using instruments developed mainly in psychology research. While we could not report in full the results of the citation paper analysis, we observed that many other instruments were developed in citation papers, either based on the TCs we found, or as independent tools. Most of them were published in conferences, which Scopus search does not reach, and therefore we cannot claim that our search was comprehensive. However, our set of TCs gives strong support for the recommendation that the CER community should use more existing instruments and invest in developing and validating new ones.

We did not identify many replications of previous research, either in the source papers or in the citation papers. The few cases that ensued from the citation paper analysis, coded with V1 or V2, include some work of this kind, but there were almost no cases where TC research was replicated. This is unfortunate, because replication would be a strong tool to help establish the external validity of TCs that have been developed.

The citation paper analysis revealed a few strong research groups that build heavily on previous work. Examples include the works of Kong, Graziotin, Repenning and Basawapatna, and their colleagues. Also it was clear that many citation papers built on the instruments that were in our source paper pool. Moreover, there was clear evidence that many such papers were published by researchers other than those who had developed the original instruments, which helps to affirm their significance for research. For other TCs, there were very few examples where someone other than the original authors had used or extended the TCs.

We finally note that the findings presented in the various quantitative models could benefit teachers when designing their courses. A better understanding of the relations, dependencies, and impacts among learners' internal and non-cognitive factors provides us with stronger arguments when selecting the teaching methods, tools, learning resources, and tasks that best match our contexts. On the other hand, qualitative findings – while only a few were identified in this survey – can inform teachers about the richness of student experiences, as well as laying the groundwork for further quantitative analysis by identifying factors that can enrich statistical model building.

5.1 Limitations

The terminology of emotion, attitude, and belief in educational and psychological research differs according to the adopted theoretical frameworks and research traditions. Therefore, we acknowledge that our search terms 'emotions', 'affect', 'attitude', 'belief', and 'self-efficacy' do not cover all related concepts that are explored in educational and psychological research. However, the search terms that we used are those that seem to be commonly used among computing education researchers, and thus we are confident that we have identified many of the relevant papers. On the other hand, there is variation in researchers' definitions of the terms, if any, and their interpretations, making it more difficult to conduct comparisons across the literature.

Another limitation is that our keyword search did not cover conference proceedings and thus we are sure to have missed some relevant papers. This was necessary to keep the amount of work within the scope of our resources. As we conducted the citation analysis we noticed that we had a handful of papers with high

numbers of citations. We did not have the resources to analyze all citations, and chose to check the most recent 30 or so citation papers to keep the amount of work reasonable. We acknowledge that we might have made some misinterpretations as we analyzed the approaches that the source papers had used. Finally, we did not measure inter-rater reliability for our classification, instead relying on the researchers discussing ambivalent cases until they reached consensus. However, all three researchers who conducted this part of the analysis are experienced researchers with a good knowledge of both qualitative and quantitative research approaches.

6 CONCLUSION

In this survey, we have identified substantial theoretical development addressing emotions, attitudes, and self-efficacy in programming education. We cannot claim that this survey has captured all such work in the field, due to the sheer number of potential publication venues, but we are confident that our work captures a substantial part of the relevant work and our summary and references provide valuable support for further research in this area.

Much of the identified work relates to the development of new validated surveys, the use of the instruments to build quantitative models of the relations and impact of the target factors on one another, along with their impact on learning performance and motivation. Our findings thus confirm the importance of instruments in computing education research. Instruments are needed for measuring factors such as emotions and attitudes, which are otherwise difficult to capture reliably; and they are also tools for other researchers, helping them to build more elaborate models of the complex experience of learning programming. When analyzing the citing papers, we found considerable evidence that instruments had indeed been used to inform further work by researchers other than the original developers. On the other hand, some theoretical constructs were rarely used by others, possibly reflecting their contextual dependency.

Our findings provide evidence that computing education research, as a field, is maturing because it develops its own theoretical constructs and models to build deeper understanding of the complex phenomena related to learning programming. There is methodological richness in quantitative research, which has the potential to produce results with stronger external validity.

In future work, we will compile a more comprehensive analysis of the identified models. They have been developed to address specific contexts and a variety of research questions. However, they already provide such a large body of findings that we expect to be able to identify some more general findings concerning the relations and impact of the factors we have been exploring. While we found little evidence of actual replication or validations of the generated theoretical models in new contexts, a summary of common features of the models could go some way to producing evidence like that produced by replication. This would be supported by a more comprehensive analysis of how the identified TCs have been used in the citation papers.

Another branch of future work would be to categorize the works according to the work of Gregor [41], i.e., what kind of purpose theories have, which ones merely analyze or explain phenomena, and which ones can be used for predicting results. This would help to better summarize where research in this area is proceeding and to identify gaps where new theoretical developments would be of value. Finally, comparing how theoretical models related to learning programming differ from learning models developed in other contexts [89] would provide an insight into what, if anything, makes learning programming different from learning other topics.

REFERENCES

[1] Sofia D Anastasiadou and Alexandros S Karakos. 2011. The beliefs of electrical and computer engineering students regarding computer programming. *The International Journal of Technology, Knowledge and Society* 7, 1 (2011), 37–51.

[2] Albert Bandura. 1994. Self-efficacy. In *Encyclopedia of Human Behavior*, VS Ramachandran (Ed.). Vol. 4. Academic Press, 71–81. Reprinted in H Friedman (Ed.), Encyclopedia of Mental Health. Academic Press, 1998.

[3] Ashok Basawapatna, Alexander Repenning, and Mark Savignano. 2019. The zones of proximal flow tutorial: designing computational thinking cliffhangers. In *50th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2019)*. 428–434.

[4] Ashok Ram Basawapatna, Alexander Repenning, and Kyu Han Koh. 2015. Closing the cyberlearning loop: enabling teachers to formatively assess student programming projects. In *46th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2015)*. 12–17.

[5] Ashok R Basawapatna, Alexander Repenning, Kyu Han Koh, and Hilarie Nickerson. 2013. The zones of proximal flow: guiding students through a space of computational thinking skills and challenges. In *Ninth International Computing Education Research Conference (ICER 2013)*. 67–74.

[6] Ryan Bockmon, Stephen Cooper, Jonathan Gratch, and Mohsen Dorodchi. 2020. Validating a CS attitudes instrument. In *51st SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2020)*. 899–904.

[7] Ryan Bockmon, Stephen Cooper, William Koperski, Jonathan Gratch, Sheryl Sorby, and Mohsen Dorodchi. 2020. A CS1 spatial skills intervention and the impact on introductory programming abilities. In *51st SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2020)*. 766–772.

[8] Monique Boekaerts. 1996. Self-regulated learning at the junction of cognition and motivation. *European Psychologist* 1, 2 (1996), 100.

[9] Nigel Bosch and Sidney D'Mello. 2017. The affective experience of novice computer programmers. *International Journal of Artificial Intelligence in Education* 27, 1 (2017), 181–206.

[10] Pierre Bourdieu. 2011. The forms of capital (1986). *Cultural theory: an anthology* 1 (2011), 81–93.

[11] Margaret M Bradley and Peter J Lang. 1994. Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of Behavior Therapy and Experimental Psychiatry* 25, 1 (1994), 49–59.

[12] Ramón Zataraín Cabada, María Lucía Barrón Estrada, Francisco González Hernández, Raúl Oramas Bustillos, and Carlos Alberto Reyes-García. 2018. An affective and Web 3.0-based learning environment for a programming language. *Telematics and Informatics* 35, 3 (2018), 611–628.

[13] Dan Cernusca and Clayton E Price. 2013. Can undergraduates learn programming with a "virtual professor"? Findings from a pilot implementation of a blended instructional strategy. *The ASEE Computers in Education Journal* 4, 4 (2013).

[14] Ibrahim Cetin and M Yasar Ozden. 2015. Development of computer programming attitude scale for university students. *Computer Applications in Engineering Education* 23, 5 (2015), 667–672.

[15] Tak-Wai Chan, Chee-Kit Looi, Wenli Chen, Lung-Hsiang Wong, Ben Chang, Calvin CY Liao, Hercy Cheng, Zhi-Hong Chen, Chen-Chung Liu, Siu-Cheung Kong, Heisawn Jeong, Jon Mason, Hyo-Jeong So, Sahana Murthy, Fu-Yun Yu, Su Luan Wong, Ronnel King, Xiaoqing Gu, Minhong Wang, Longkai Wu, Ronghuai Huang, Rachel Lam, and Hiroaki Ogata. 2018. Interest-driven creator theory: towards a theory of learning design for Asia in the twenty-first century. *Journal of Computers in Education* 5, 4 (2018), 435–461.

[16] Chen Chen, Paulina Haduong, Karen Brennan, Gerhard Sonnert, and Philip Sadler. 2019. The effects of first programming language on college students' computing attitude and achievement: a comparison of graphical and textual languages. *Computer Science Education* 29, 1 (2019), 23–48.

[17] Kuanchin Chen and Alan Rea. 2019. Do pair programming approaches transcend coding? Measuring agile attitudes in diverse information systems courses. *Journal of Information Systems Education* 29, 2 (2019).

[18] Gary Cheng. 2019. Exploring factors influencing the acceptance of visual programming environment among boys and girls in primary schools. *Computers in Human Behavior* 92 (2019), 361–372.

[19] Jody Clarke-Midura, Frederick J Poole, Katarina Pantic, Chongning Sun, and Vicki Allan. 2018. How mother and father support affect youths' interest in computer science. In *14th International Computing Education Research Conference (ICER 2018)*. 215–222.

[20] Benjamin Clement. 2018. *Adaptive Personalization of Pedagogical Sequences using Machine Learning*. Ph.D. Dissertation. Bordeaux.

[21] Anthony Cox and Maryanne Fisher. 2009. Programming style: influences, factors, and elements. In *Second International Conference on Advances in Computer-Human Interactions*. IEEE, 82–89.

[22] Mihaly Csikszentmihalyi. 2000. *Beyond Boredom and Anxiety*. Jossey-Bass.

[23] Quintin Cutts, Sarah Esper, Marlena Fecho, Stephen R Foster, and Beth Simon. 2012. The abstraction transition taxonomy: developing desired learning outcomes through the lens of situated cognition. In *[Eighth] International Computing Education Research Conference (ICER 2012)*. 63–70.

[24] Holger Danielsiek, Laura Toma, and Jan Vahrenhold. 2017. An instrument to assess self-efficacy in introductory algorithms courses. In *13th International Computing Education Research Conference (ICER 2017)*. 217–225.

[25] Salihu Dasuki and Ago Quaye. 2016. Undergraduate students' failure in programming courses in institutions of higher education in developing countries: a Nigerian perspective. *The Electronic Journal of Information Systems in Developing Countries* 76, 1 (2016), 1–18.

[26] Jessica Q Dawson, Meghan Allen, Alice Campbell, and Anasazi Valair. 2018. Designing an introductory programming course to improve non-majors' experiences. In *49th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2018)*. 26–31.

[27] Ed Diener, Derrick Wirtz, William Tov, Chu Kim-Prieto, Dong-Won Choi, Shigehiro Oishi, and Robert Biswas-Diener. 2010. New well-being measures: short scales to assess flourishing and positive and negative feelings. *Social Indicators Research* 97, 2 (2010), 143–156.

[28] Brian Dorn and Allison Elliott Tew. 2015. Empirical validation and application of the computing attitudes survey. *Computer Science Education* 25, 1 (2015), 1–36.

[29] Angela Lee Duckworth and Patrick D Quinn. 2009. Development and validation of the short grit scale (GRIT–S). *Journal of Personality Assessment* 91, 2 (2009), 166–174.

[30] Rodrigo Duran, Jan-Mikael Rybicki, Juha Sorva, and Arto Hellas. 2019. Exploring the value of student self-evaluation in introductory programming. In *15th International Computing Education Research Conference (ICER 2019)*. 121–130.

[31] Michail N Giannakos and Letizia Jaccheri. 2018. From players to makers: an empirical examination of factors that affect creative game development. *International Journal of Child-Computer Interaction* 18 (2018), 27–36.

[32] Ken Goldman, Paul Gross, Cinda Heeren, Geoffrey L Herman, Lisa Kaczmarczyk, Michael C Loui, and Craig Zilles. 2010. Setting the scope of concept inventories for introductory computing subjects. *ACM Transactions on Computing Education (TOCE)* 10, 2, Article 5 (2010), 29 pages.

[33] Maria J Grant and Andrew Booth. 2009. A typology of reviews: an analysis of 14 review types and associated methodologies. *Health Information & Libraries Journal* 26, 2 (2009), 91–108.

[34] Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Pekka Abrahamsson. 2017. On the unhappiness of software developers. In *21st International Conference on Evaluation and Assessment in Software Engineering*. 324–333.

[35] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2014. Happy software developers solve problems better: psychological measurements in empirical software engineering. *PeerJ* 2 (2014), e289.

[36] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2014. Software developers, moods, emotions, and performance. *arXiv preprint arXiv:1405.4422* (2014).

[37] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2015. The affect of software developers: common misconceptions and measurements. In *2015 IEEE/ACM Eighth International Workshop on Cooperative and Human Aspects of Software Engineering*. IEEE, 123–124.

[38] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2015. Do feelings matter? On the correlation of affects and the self-assessed productivity in software engineering. *Journal of Software: Evolution and Process* 27, 7 (2015), 467–487.

[39] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2015. How do you feel, developer? An explanatory theory of the impact of affects on programming performance. *PeerJ Computer Science* 1 (2015), e18.

[40] Daniel Graziotin, Xiaofeng Wang, and Pekka Abrahamsson. 2015. Understanding the affect of developers: theoretical background and guidelines for psychoempirical software engineering. In *Seventh International Workshop on Social Software Engineering*. 25–32.

[41] Shirley Gregor. 2006. The nature of theory in information systems. *MIS Quarterly* (2006), 611–642.

[42] Melih Derya Gurer, Ibrahim Cetin, and Ercan Top. 2019. Factors affecting students' attitudes toward computer programming. *Informatics in Education* 18, 2 (2019).

[43] Kevin Hedberg Griffith and Erik Nguyen. 2018. *Sampling Affects of Software Developers to Understand Individual & Team Performance*. Master's thesis. Chalmers University of Technology.

[44] Sean M Horan, Matthew M Martin, and Keith Weber. 2012. Understanding emotional response theory: the role of instructor power and justice messages. *Communication Quarterly* 60, 2 (2012), 210–233.

[45] Ryosuke Ishizue, Kazunori Sakamoto, Hironori Washizaki, and Yoshiaki Fukazawa. 2018. Student placement and skill ranking predictors for programming classes using class attitude, psychological scales, and code metrics. *Research and Practice in Technology Enhanced Learning* 13, Article 7 (2018).

[46] Zahra Karimi, Ahmad Baraani-Dastjerdi, Nasser Ghasem-Aghaee, and Stefan Wagner. 2016. Links between the personalities, styles and performance in computer programming. *Journal of Systems and Software* 111, C (2016), 228–241.

[47] Zahra Karimi, Ahmad Baraani-Dastjerdi, Naser Ghasem-Aghaee, and Stefan Wagner. 2016. Using personality traits to understand the influence of personality on computer programming: an empirical study. *Journal of Cases on Information Technology* 18, 1 (2016), 28–48.

[48] Päivi Kinnunen and Beth Simon. 2010. Experiencing programming assignments in CS1: the emotional toll. In *Sixth International Computing Education Research Conference (ICER 2010)*. 77–86.

[49] Päivi Kinnunen and Beth Simon. 2012. My program is OK – am I? Computing freshmen's experiences of doing programming assignments. *Computer Science Education* 22, 1 (2012), 1–28.

[50] David A Kolb. 2014. *Experiential Learning: Experience as the Source of Learning and Development*. FT press.

[51] Siu-Cheung Kong, Ming Ming Chiu, and Ming Lai. 2018. A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education* 127 (2018), 178–189.

[52] Siu-Cheung Kong, Robert Kwok-Yiu Li, and Ron Chi-Wai Kwok. 2019. Measuring parents' perceptions of programming education in P-12 schools: scale development and validation. *Journal of Educational Computing Research* 57, 5 (2019), 1260–1280.

[53] Siu-Cheung Kong and Yi-Qing Wang. 2019. Nurture interest-driven creators in programmable robotics education: an empirical investigation in primary school settings. *Research and Practice in Technology Enhanced Learning* 14, 1 (2019), 20.

[54] Siu-Cheung Kong and Yi-Qing Wang. 2019. Positive youth development from a "3Cs" programming perspective: a multi-study investigation in the university. *Computer Science Education* 29, 4 (2019), 335–356.

[55] Siu Cheung Kong and Yi Qing Wang. 2020. Formation of computational identity through computational thinking perspectives development in programming learning: a mediation analysis among primary school students. *Computers in Human Behavior* 106 (2020), 106230.

[56] Siu-Cheung Kong, Yi-Qing Wang, and Ming Lai. 2019. Development and validation of an instrument for measuring digital empowerment of primary school students. In *ACM Conference on Global Computing Education*. 172–177.

[57] Sevda Kucuk and Burak Sisman. 2018. Pre-service teachers' experiences in learning robotics design and programming. *Informatics in Education* 17, 2 (2018), 301–320.

[58] Feng-Yang Kuo, Wen-Hsiung Wu, and Cathy S Lin. 2013. An investigation of self-regulatory mechanisms in learning to program Visual Basic. *Journal of Educational Computing Research* 49, 2 (2013), 225–247.

[59] Carmen Lacave, Ana I Molina, and Miguel A Redondo. 2018. A preliminary instrument for measuring students' subjective perceptions of difficulties in learning recursion. *IEEE Transactions on Education* 61, 2 (May 2018), 119–126.

[60] Richard D Lane, Phyllis ML Chua, and Raymond J Dolan. 1999. Common effects of emotional valence, arousal and attention on neural activation during visual processing of pictures. *Neuropsychologia* 37, 9 (1999), 989–997.

[61] Kris MY Law, Victor CS Lee, and YT Yu. 2010. Learning motivation in e-learning facilitated computer programming courses. *Computers & Education* 55, 1 (2010), 218–228.

[62] Stella Lee, Trevor Barker, and Vivekanandan Suresh Kumar. 2016. Effectiveness of a learner-directed model for e-learning. *Journal of Educational Technology & Society* 19, 3 (2016), 221–233.

[63] Feng Li, Xinwen Bai, and Yong Wang. 2013. The scale of positive and negative experience (SPANE): psychometric properties and normative data in a large Chinese sample. *PloS one* 8, 4 (2013).

[64] Yi-Wen Liao, Yueh-Min Huang, Yong-Ming Huang, Zhi-Yuan Su, and Chun-Wang Wei. 2019. An empirical evaluation of online cooperative programming platforms based on the PACT framework and technology readiness. *Journal of Internet Technology* 20, 2 (2019), 345–352.

[65] Tze Wei Liew, Nor Azan Mat Zin, and Noraidah Sahari. 2017. Exploring the affective, motivational and cognitive effects of pedagogical agent enthusiasm in a multimedia learning environment. *Human-Centric Computing and Information Sciences* 7, 1 (2017).

[66] Hui Lin, Weiguo Fan, and Linda Wallace. 2013. The effects of social and technical factors on user satisfaction, sense of belonging and knowledge community usage. *International Journal of e-Collaboration* 9, 3, Article 2 (2013).

[67] Alex Lishinski and Aman Yadav. 2019. Motivation, attitudes, and dispositions. In *The Cambridge Handbook of Computing Education Research*, Sally A Fincher and Anthony V Robins (Eds.). Cambridge University Press, 801–826.

[68] Alex Lishinski, Aman Yadav, and Richard Enbody. 2017. Students' emotional reactions to programming projects in introduction to programming: measurement approach and influence on learning outcomes. In *13th International Computing*

Education Research Conference (ICER 2017). 30–38.

[69] Alex Lishinski, Aman Yadav, Jon Good, and Richard Enbody. 2016. Learning to program: gender differences and interactive effects of students' motivation, goals, and self-efficacy on performance. In *12th International Computing Education Research Conference (ICER 2016)*. 211–220.

[70] Andrew Luxton-Reilly, Simon, Ibrahim Albluwi, Brett A Becker, Michail Giannakos, Amruth N. Kumar, Linda Ott, James Paterson, Michael James Scott, Judy Sheard, and Claudia Szabo. 2018. Introductory programming: a systematic literature review. In *ITiCSE 2018 Working Group Reports (ITiCSE WGR 2018)*. 55–106.

[71] Alejandra J Magana, Michael L Falk, Camilo Vieira, and Michael J Reese. 2016. A case study of undergraduate engineering students' computational literacy and self-beliefs about computing in the context of authentic practices. *Computers in Human Behavior* 61, C (Aug. 2016), 427–442.

[72] Brian Magerko, Jason Freeman, Tom McKlin, Mike Reilly, Elise Livingston, Scott McCoid, and Andrea Crews-Brown. 2016. EarSketch: a STEAM-based approach for underrepresented populations in high school computer science education. *ACM Transactions on Computing Education (TOCE)* 16, 4 (2016), 1–25.

[73] Pruthikrai Mahatanankoon and David Sikolia. 2017. Intention to remain in a computing program: exploring the role of passion and grit. In *Americas Conference on Information Systems (AMCIS)*.

[74] Gregory R Maio, Geoffrey Haddock, and Bas Verplanken. 2018. *The Psychology of Attitudes and Attitude Change*. Sage Publications Limited.

[75] Lauri Malmi, Judy Sheard, Päivi Kinnunen, Simon, and Jane Sinclair. 2019. Computing education theories: what are they and how are they used?. In *15th International Computing Education Research Conference (ICER 2019)*. 187–197.

[76] Lauri Malmi, Judy Sheard, Simon, Roman Bednarik, Juha Helminen, Päivi Kinnunen, Ari Korhonen, Niko Myller, Juha Sorva, and Ahmad Taherkhani. 2014. Theoretical underpinnings of computing education research: what is the evidence?. In *Tenth International Computing Education Research Conference (ICER 2014)*. 27–34.

[77] GC Marchand and AP Gutierrez. 2012. The role of emotion in the learning process: comparisons between online and face-to-face learning settings. *Internet and Higher Education* 15, 3 (2012), 150–160.

[78] Lauren Margulieux, Tuba Ayer Ketenci, and Adrienne Decker. 2019. Review of measurements used in computing education research and suggestions for increasing standardization. *Computer Science Education* 29, 1 (2019), 49–78.

[79] Lauren E Margulieux, Brian Dorn, and Kristin A Searle. 2019. Learning sciences for computing education. In *The Cambridge Handbook of Computing Education Research*, Sally Fincher and Anthony V Robins (Eds.). Cambridge University Press, 208–230.

[80] Fagbola Temitayo Matthew, Adeyanju Ibrahim Adepoju, Oloyede Ayodele, Obe Olumide, Olaniyan Olatayo, Esan Adebimpe, Omodunbi Bolaji, and Egbetola Funmilola. 2018. Development of mobile-interfaced machine learning-based predictive models for improving students' performance in programming courses. *International Journal of Advanced Computer Science and Applications* 9, 5 (2018).

[81] Robert McCartney, Jonas Boustedt, Anna Eckerdal, Kate Sanders, Lynda Thomas, and Carol Zander. 2016. Why computing students learn on their own: motivation for self-directed learning of computing. *ACM Transactions on Computing Education (TOCE)* 16, 1, Article 2 (Jan. 2016), 18 pages.

[82] Monica M McGill. 2012. The curriculum planning process for undergraduate game degree programs in the United Kingdom and United States. *ACM Transactions on Computing Education (TOCE)* 12, 2, Article 7 (April 2012), 47 pages.

[83] Orni Meerbaum-Salant, Michal Armoni, and Mordechai (Moti) Ben-Ari. 2013. Learning computer science concepts with Scratch. *Computer Science Education* 23, 3 (2013), 239–264.

[84] Greg L Nelson and Amy J Ko. 2018. On use of theory in computing education research. In *14th International Computing Education Research Conference (ICER 2018)*. 31–39.

[85] Uolevi Nikula, Orlena Gotel, and Jussi Kasurinen. 2011. A motivation guided holistic rehabilitation of the first programming course. *ACM Transactions on Computing Education (TOCE)* 11, 4, Article 24 (Nov. 2011), 38 pages.

[86] Mogens Niss. 2007. The concept and role of theory in mathematics education. *Relating practice and research in mathematics education. Norma* 5 (2007), 97–110.

[87] Gwen Nugent, Bradley Barker, Houston Lester, Neal Grandgenett, and Dagen Valentine. 2019. Wearable textiles to support student STEM learning and attitudes. *Journal of Science Education and Technology* 28, 5 (2019), 470–479.

[88] Keith J O'Hara, Kathleen Burke, Diana Ruggiero, and Sven Anderson. 2017. Linking language & thinking with code: computing within a writing-intensive introduction to the liberal arts. In *22nd Conference on Innovation and Technology in Computer Science Education (ITiCSE 2017)*. 269–274.

[89] Ernesto Panadero. 2017. A review of self-regulated learning: six models and four directions for research. *Frontiers in Psychology* 8 (2017), 422.

[90] Reinhard Pekrun, Thomas Goetz, Wolfram Titz, and Raymond P Perry. 2002. Academic emotions in students' self-regulated learning and achievement: a program of qualitative and quantitative research. *Educational psychologist* 37, 2 (2002), 91–105.

[91] Reinhard Pekrun, Stephanie Lichtenfeld, Herbert W Marsh, Kou Murayama, and Thomas Goetz. 2017. Achievement emotions and academic performance: longitudinal models of reciprocal effects. *Child Development* 88, 5 (2017), 1653–1670.

[92] Alexander Repenning, Ashok Basawapatna, Dorit Assaf, Carmine Maiello, and Nora Escherle. 2016. Retention of flow: evaluating a computer science education week activity. In *47th SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2016)*. 633–638.

[93] Alexander Repenning, David Webb, and Andri Ioannidou. 2010. Scalable game design and the development of a checklist for getting computational thinking into public schools. In *41st SIGCSE Technical Symposium on Computer Science Education (SIGCSE 2010)*. 265–269.

[94] Alexander Repenning, David C Webb, Kyu Han Koh, Hilarie Nickerson, Susan B Miller, Catharine Brand, Ian Her Many Horses, Ashok Basawapatna, Fred Gluck, Ryan Grover, Kris Gutierrez, and Nadia Repenning. 2015. Scalable game design: a strategy to bring systemic computer science education to schools through game design and simulation creation. *ACM Transactions on Computing Education (TOCE)* 15, 2 (2015), 1–31.

[95] Anthony V Robins and Lauren E Margulieux. 2019. Cognitive sciences for computing education. In *The Cambridge Handbook of Computing Education Research*, Sally Fincher and Anthony V Robins (Eds.). Cambridge University Press, 231–275.

[96] Mary Beth Rosson, John M Carroll, and Hansa Sinha. 2011. Orientation of undergraduates toward careers in the computer and information sciences: gender, self-efficacy and social support. *ACM Transactions on Computing Education (TOCE)* 11, 3, Article 14 (2011), 23 pages.

[97] Samara Ruiz, Sven Charleer, Maite Urretavizcaya, Joris Klerkx, Isabel Fernández-Castro, and Erik Duval. 2016. Supporting learning by considering emotions: tracking and visualization a case study. In *Sixth International Conference on Learning Analytics & Knowledge*. 254–263.

[98] Michael J Scott and Gheorghita Ghinea. 2013. On the domain-specificity of mindsets: the relationship between aptitude beliefs and programming practice. *IEEE Transactions on Education* 57, 3 (2013), 169–174.

[99] Michael James Scott and Gheorghita Ghinea. 2014. Measuring enrichment: the assembly and validation of an instrument to assess student self-beliefs in CS1. In *Tenth International Computing Education Research Conference (ICER 2014)*. 123–130.

[100] Ruey-Shiang Shaw. 2010. A study of learning performance of e-learning materials design with knowledge maps. *Computers & Education* 54, 1 (2010), 253–264.

[101] Johanna M Silvennoinen and Jussi PP Jokinen. 2016. Appraisals of salient visual elements in web page design. *Advances in Human-Computer Interaction* (2016).

[102] Patrick Suppes. 1974. The place of theory in educational research. *Educational Researcher* 3, 6 (1974), 3–10.

[103] Claudia Szabo, Nickolas Falkner, Andrew Petersen, Heather Bort, Kathryn Cunningham, Peter Donaldson, Arto Hellas, James Robinson, and Judy Sheard. 2019. Review and use of learning theories within computer science education research: primer for researchers and practitioners. In *ITiCSE 2019 Working Group Reports*. 89–109.

[104] Martha Tapia and George E Marsh. 2004. An instrument to measure mathematics attitudes. *Academic Exchange Quarterly* 8, 2 (2004), 16–22.

[105] Meng-Jung Tsai, Ching-Yeh Wang, and Po-Fen Hsu. 2019. Developing the computer programming self-efficacy scale for computer literacy education. *Journal of Educational Computing Research* 56, 8 (2019), 1345–1360.

[106] Robert J Vallerand, Céline Blanchard, Genevieve A Mageau, Richard Koestner, Catherine Ratelle, Maude Léonard, Marylene Gagné, and Josée Marsolais. 2003. Les passions de l'ame: on obsessive and harmonious passion. *Journal of Personality and Social Psychology* 85, 4 (2003), 756.

[107] Iris Vessey. 1985. Expertise in debugging computer programs: a process analysis. *International Journal of Man-Machine Studies* 23, 5 (1985), 459–494.

[108] Camilo Vieira, Alejandra J Magana, R Edwin García, Aniruddha Jana, and Matthew Krafcik. 2018. Integrating computational science tools into a thermodynamics course. *Journal of Science Education and Technology* 27 (2018), 322–333.

[109] Lev Vygotsky. 1987. Zone of proximal development. In *Mind in Society: the Development of Higher Psychological Processes*. Harvard University Press, 52–91.

[110] Hatice Yildiz Durak. 2018. Flipped learning readiness in teaching programming in middle schools: modelling its relation to various variables. *Journal of Computer Assisted Learning* 34, 6 (2018), 939–959.

[111] Yulei (Gavin) Zhang and Yan (Mandy) Dang. 2015. Investigating essential factors on students' perceived accomplishment and enjoyment and intention to learn in web development. *ACM Transactions on Computing Education (TOCE)* 15, 1, Article 3 (March 2015), 21 pages.

[112] Barry J Zimmerman. 2013. From cognitive modeling to self-regulation: a social cognitive career path. *Educational Psychologist* 48, 3 (2013), 135–147.