Measuring Enrichment: The Assembly and Validation of an Instrument to Assess Student Self-Beliefs in CS1

Michael James Scott
Information Systems, Computing & Mathematics
Brunel University
Uxbridge, Middlesex, UB8 3PH
United Kingdom
michael.scott@brunel.ac.uk

Gheorghita Ghinea
Information Systems, Computing & Mathematics
Brunel University
Uxbridge, Middlesex, UB8 3PH
United Kingdom
george.ghinea@brunel.ac.uk

ABSTRACT

Educational research has shown that self-beliefs can have profound influences on learning behaviour and achievement. It follows, then, that beliefs about the nature of programming aptitude (e.g., students' mindset) and the way in which individuals perceive themselves as programmers (e.g., students' self-concept) could also have a salient impact on programming practice behaviour and the development of programming expertise. However, in order to test this hypothesis, a valid and reliable measurement instrument is needed. This paper draws upon the Control-Value Theory of Achievement Emotion to assemble such a measurement instrument. An evaluation of the proposed measurement instrument with three cohorts of undergraduate computing students ($N = 239$) then demonstrates that reliability and construct validity are adequate, while the concurrent validity of the conceptual framework is satisfactory. This suggests that the measurement instrument is suitable for further research into students' self-beliefs within the introductory programming context. However, it is important to note that this work represents only a first step and further validation is required to establish whether the measurement instrument is valid across different contexts and populations.

Categories and Subject Descriptors

J.4 [**Social and Behavioral Sciences**]: Psychology; K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Computer Science Education.*

General Terms

Human Factors, Measurement.

Keywords

Measurement; Self-Beliefs; Self-Concept; Self-Efficacy; Mindset; Anxiety; Interest; Practice; Programming; CS1.

ICER'14, August 11-13, 2014, Glasgow, Scotland, UK.

ACM 978-1-4503-2755-8/14/08 ...$15.00.
http://dx.doi.org/10.1145/2632320.2632350.

1. INTRODUCTION

Reliable and valid measurement is critically important in educational research. Variables of interest should be clearly defined and measured with minimal error in order to make meaningful conclusions from an analysis [8, 44]. However, there is not a strong history of instrument development and validation in computer science education research. Two systematic reviews of the literature found that the quantitative research in the field would benefit from improvements in methodology and reporting [37, 46]. Of particular interest, only 1.5% of articles published between 2000 and 2005 reported adequate psychometric information to support the validity, the reliability, and the generalisability of their claims [37].

A number of measurement instruments have since been developed and evaluated. For example, the Foundations of CS1 Test (FCS1) assesses students' performance in the cognitive-domain of introductory computing [45]. However, few instruments address the affective-domain of learning computing (e.g., attitude development [9]). In particular, there is a need to explore the emotive aspects of learning computer programming as, for some students, programming invokes strong negative feelings [19, 24, 38] and shapes their self-beliefs in counter-intuitive ways [22]. This is important to consider because self-beliefs play an important role in academic development [5, 26]. As an example, beliefs about the nature of programming aptitude, extending Dweck's mindset theory (see [7, 10, 11, 32, 42]), can lead to significant differences in the time that students report practising programming [41]. Nevertheless, to pursue this line of research, a valid measurement instrument is needed.

As such, this article will propose one such measurement instrument and will then address the research question: is the proposed measurement instrument reliable and valid? The following section will highlight a number of challenges encountered in introductory programming and will then situate these challenges within the Control-Value Theory of Achievement Emotions to illustrate the potential role which students' self-beliefs may play. Drawing from the theory, a parsimonious set of key variables is then identified to include in the measurement instrument. The next section then describes how the proposed measurement instrument was assembled. This leads into an evaluation with three cohorts of undergraduate computer students. The paper then closes with a brief discussion of the potential uses of the measurement instrument, its limitations, and a conclusion on its adequacy for future research.

2. BACKGROUND

2.1 Challenges in Programming Education

Programming is a craft which many would seem to find challenging to learn [21]. Notwithstanding practices that have been shown to improve retention and success [36], failure rates can be high [4] and there is a history of poor outcomes in the higher education context [17, 45]. The reasons for these poor outcomes are complex and multifaceted (see [3]). However, for this reason, introductory is considered a challenge for computer science education research [17, 30].

A key issue can sometimes be the amount *and the quality* of practice that novice programmers engage in [40]. This is because the discipline can require a substantial level of deliberate practice to master [13, 51]. That is, practice which is ongoing, focused, reflective, and situated at the right level of challenge for any individual student [13]. This type of practice, however, is inherently uncomfortable and demands that learners remain motivated.

Despite the best efforts of instructors (e.g., through encouraging and motivating students [20]), learners regularly report negative experiences when they engage with programming tasks [22, 38]. Some authors describe this phenomenon as *programming trauma* [19] and, to reinforce such striking language, there is some evidence which indicates that the type of task anxiety these experiences invoke are related to the activation of brain regions associated with visceral threat detection and pain [27]. Another concern is evidence suggesting that such affective factors worsen over a course of instruction [31, 41].

With this being the case, the emotions that learners feel may prompt them to reflect on themselves and their ability in several different ways [22]. Potentially, learners may start to believe that they no longer have the time or the motivation to overcome these challenges as they cannot envision success in the future [23]. In other words, learners may change their self-beliefs based on their experiences, through a process of self-appraisal, potentially diminishing the way that they identify with programming as a discipline and disengaging with deliberate forms of practice [35, 40].

2.2 The Control-Value Theory of Achievement Emotions

A framework that considers the role of self-beliefs and emotions in learning is the Control-Value Theory of Achievement Emotion [33, 34]. In this framework, students' self-appraisal of ongoing achievement activities, and of their past and future outcomes, are of key importance. This is because the emotions that they experience during a particular task will depend upon whether they feel in control of the outcome and that the outcome is subjectively important to them.

These emotions then influence academic engagement and performance through the model shown in Figure 1. The model proposes that instruction and support have an influence on the way in which individuals form the control and value appraisals. These appraisals then shape the specific achievement emotions that students may experience based on whether they feel they can control activities and outcomes that are subjectively important to them. These emotions then have a direct impact on self-regulated learning and performance. Specifically, emotions seem to influence cognitive resources, use of strategies, and dependence on external regulation of learning [33]. The overall model is also reciprocal in nature, such that outcomes can shape emotions while both emotion and performance shape the way students form their self-appraisals. In some cases, instruction and support may also respond to student needs. In particular, offering a range of interventions which could influence any part of the model. As this process continues over time, it could have substantial impact on learning behaviour and subsequently performance; as evidenced through the known co-variance between self-efficacy beliefs and success [47, 50].

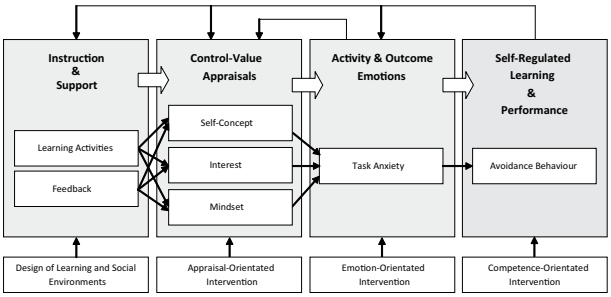

Figure 1: Overview of the Control-Value Theory of Achievement Emotion with an Embedded Conceptual Framework (Adapted From [33, 34])

As each component of the framework represent a broad range of different constructs, a parsimonious conceptual framework has been embedded within the model. This example has been derived from factors hypothesised to influence student programming practice [40] and illustrates how learning activities and feedback influence students' self-beliefs. Namely: self-concept, which is is understood to be a composite of "self-perceptions that are formed through experience with and interpretations of one's environment" [29]; interest, which is the extent to which an individual enjoys engaging with a set of tasks; and mindset, based on Dweck's [10] notion of mindsets. That is, students have a growth mindset, where they believe their capacities can be developed through practice, or students have a fixed mindset, where they believe their capacities are natural, inherent qualities. These, in turn, influence task anxiety which, consequently, may encourage avoidance behaviour.

2.3 Proposed Conceptual Framework and Instrument Assembly

To validate the framework and test such a hypothesis, it is necessary to develop an appropriate measurement instrument. Therefore, in line with the proposed conceptual framework shown in Figure 2, items for the key variables were assembled.

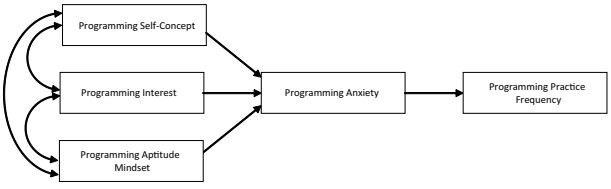

Figure 2: A Conceptual Framework for Enhancing Students' Programming Practice

The measurement model for the proposed measurement instrument consisted of four constructs: Programmer Self-Concept (PSC); Interest in Software Development (INT); Programming Anxiety (ANX); and Mindset Towards Programming Aptitude (APT). Additionally, in order to ensure appropriate discriminatory power between constructs, such as differences between self-concept and self-efficacy [6], items relating to software debugging task self-efficacy are also included (DSE). As existing instruments target similar constructs of interest, items were drawn from the literature and adapted to the introductory programming context. A self-report of programming practice behaviour, for the purpose of establishing the concurrent validity of the proposed framework, is also included (see [41]).

The construct debugging task self-efficacy captures learners' cognitive self-assessments of whether or not they are confident in their ability to write and debug simple programs. This is based on the theoretical construct proposed by Bandura [1], as it relates to how self-assessments influence behaviour change. The items for this construct were created using guidelines regarding the domain-specificity of self-efficacy and its association with particular criterial tasks.

The construct of programmer self-concept has some conceptual overlap with debugging self-efficacy, however there are a range of theoretical and empirical differences [6, 14]. It represents a composite of self-perceptions that one can be a good programmer, which is "formed through experience with and interpretations of one's environment". This construct drives the affective elements of being a programmer as opposed to a cognitive assessment of success at programming because "self-concept better predicts affective reactions such as anxiety, satisfaction, and self-esteem, whereas self-efficacy better predicts cognitive processes and actual performance" [6]. The items for this construct were adapted from scales used by Ferla et al. [14] and Eccles & Wigfield [12]. These focus on the ability-belief component of self-concept.

The construct for interest in software development measures the extent to which an individual enjoys engaging with programming-related activities. This construct is believed to have a reciprocal relationship with self-concept, resulting in the pursuit of more achievement experiences in a domain [16]. The items for this construct were adapted from the scale used by Wigfield et al. [48], focusing on the enjoyment aspect of interest.

The programming anxiety instrument construct measures the self-reflected state of experiencing negative emotions, such as nervousness or helplessness, while writing and debugging programs. The items were drawn and adapted from the worry-component of the instrument used by Wigfield and Meece [49].

The mindset towards programming aptitude instrument construct represents the strength of a learners' belief in the notion of a fixed programming aptitude (e.g., aptitude is inherent and cannot change). The items were drawn from Dweck [10].

These items were then put together as a 5-point Likert instrument, with each item rated from strongly disagree to strongly agree. Each item was reviewed by 2 colleagues and a small convenience sample of undergraduate students, and revised to improve content validity and readability. This resulted in the instrument shown in Table 1.

3. METHOD

In order to evaluate the proposed measurement instrument, the psychometric properties are examined. Namely, based on the recommendations of Straub *et al.* [43] and other authors (e.g. [8, 44]), reliability and validity need to be established in order to deem a measurement instrument adequate. This involved a trial of the instrument with three cohorts of students at the conclusion of their first programming course and an analysis of their responses using a confirmatory factor analysis technique (see [18]).

3.1 Data Collection

The sampling frame for each cohort was set to all students who had submitted at least one assignment or code review to ensure participants had indeed attended the course. Minimum sample size requirements were calculated using Cochran's formula for continuous data with finite population correction and adjusted for anticipated non-response [2].

A random sampling procedure was used to select participants. Data was collected in three rounds: a paper-based survey was distributed to all students in the lab environment (unselected cases are not considered in analysis); a digital version was then advertised on the virtual learning environment and email alerts were distributed to those whom had not responded to the paper-version; after ten days, an additional series of follow-up emails were distributed to the non-respondents. All participants were offered an opt-out for further communication at each stage.

From 126, 115 and 98 invitations for each respective cohort, 91, 84, and 64 responded. This represents an overall response rate of 70%, noting that 34 cases in 2011-12, 30 cases in 2012-13, and 21 cases in 2013-14 were classified as late respondents. This is because their response was elicited after considerable follow-up during a third round of data collection.

3.2 Participants

Participants were all first undergraduate students following the sequential pathway for either 'Computer Science' or 'Business Computing'. The descriptive statistics show that less than 20% of the respondents were female, while the average age was 20 years, with approximately 15% respondents being mature students (over the age of 23 at entry).

Admission to the pathway required at least 300 UCAS Points (University & College Admission System Points), with a strong preference for STEM subjects (science, technology, engineering, and mathematics). Prior programming experience was not required. However, students without a relevant STEM qualification, or the required points, could opt to pursue a relevant foundation course.

During the introductory programming course, students would learn object-orientated design and the fundamental constructs of the Java language. This was conducted through a sequence of laboratory-based assignments and a collaborative project. The assignment for the 2011-12 cohort was a website and a lab-based programming examination. The assignments for the 2012-13 and 2013-14 cohort were robot scripting tasks, where students would program robots to complete activities such as maze navigation or communication in Morse Code. These assignments were examined by code review and oral viva.

4. DATA ANALYSIS

According to Straub *et al.* [43], there are three main forms of validity and reliability which are important in instrument development: content validity, construct validity and internal reliability. Content validity is the level at which items used to measure a construct reflect the meaning of the construct (and breadth of possible items which could represent the construct) to which the items will be generalised. Construct validity is the form of validity that deals with the degree to which items are an effective measure of a theoretical construct. This is often sub-divided into convergent validity and discriminant validity as evidence for both *imply* construct validity [18]. Convergent validity refers to the level at which multiple items which theoretically should be related are actually related. Conversely, discriminant validity assess the extent to which items which should be unrelated are actually unrelated. Reliability refers to the extend with which parallel items are consistent in what they are intended to measure (e.g. responses to a set of related items are internally consistent). Concurrent validity is also a consideration in cases where constructs should be related. That is, a construct is related to, or able to predict, another in the same instrument.

The data was analysed in PASW v20 and AMOS 21. All data was analysed. Items under consideration were modified to reflect feedback received from the 2011-12 cohort (see [39]), as such items were analysed on a pair-wise basis. This section follows the factor analysis procedure outlined by Hair *et al.* [18].

4.1 Descriptive Statistics

Descriptive statistics for the three samples are shown in Table 1 on the following page. This shows that learners tended to report high DSE, PSC, and INT. Many reported low ANX and, as indicated by low APT, many endorsed a growth view of programming aptitude.

Some analyses require the distribution of the data to follow a normal distribution. This was verified through an examination of skew and kurtosis, with skew indices greater than 3.0 and kurtosis indices greater than 10.0 often indicative of severe non-normality [25]. Table 1 shows these indices are within these guidelines.

4.2 Measurement Model

To verify the structure of the items for the proposed measurement model (i.e., checking that it was appropriate to group variables together into meaningful constructs), the proposed five-construct solution was evaluated using maximum-likelihood confirmatory factor analysis. As advised in [18], several fit indices were used to determine fit. One APT item was eliminated at this stage due to a low regression weight. Modifications were also made based on the modification indices to improve overall fit. These fit indices for the final set of items, shown in Table 2, indicate that the hypothesised model was 'not a bad fit' to the data (i.e., accepting the null hypothesis of having no significant difference between the prediction and the data).

This suggests that the expected model was adequately reflected by the structure of the data. However, it should be noted that alternative models with superior fit could still exist. An exhaustive review of alternative candidate models is beyond the scope of this article.

Table 2: Fit Indices and Criteria for the Measurement Model

Fit Index	Measurement Model	Adequate Fit Criteria [18]
$\chi^2(df = 153)$	267.312	N/A
χ^2/df	1.747	< 3.00
p	0.000	> 0.05
NNFI	0.950	> 0.90
CFI	0.960	> 0.90
SRMR	0.044	< 0.08
RMSEA	0.056	< 0.08

Note: df: degrees of freedom, NNFI: non-normed fit index, CFI: comparative fit index, SRMR: standardised root mean square residual, RMSEA: root mean square error of approximation.

Table 4: Regression Results for Relations in the Proposed Structural Model

Relationship	Estimate	Standard Error	Critical Ratio	p
APT $\rightarrow$ ANX	0.310	0.084	3.685	< 0.001
INT $\rightarrow$ ANX	-0.020	0.149	-0.135	0.893
PSC $\rightarrow$ ANX	-0.534	0.085	-6.301	< 0.001
ANX $\rightarrow$ PRACT	-2.335	0.389	-6.004	< 0.001

Note: APT: programming aptitude mindset, PSC: programming self-concept, ANX: programming anxiety, PRACT: frequency of programming practice.

4.2.1 Reliability

Reliability is assessed through examining the Composite Reliability (CR) of each construct. Values close to 1.0 indicate reliablilty, with 0.7 considered minimal [18]. Table 3 shows that the values are consistently above 0.7. Thus, the measurement instrument was reliable with this sample.

4.2.2 Construct Validity

In order to establish construct validity, each construct should demonstrate convergent and discriminant validity. Adequate convergent validity is demonstrated by an Average Variance Extracted (AVE) greater than 0.5 [18]. Table 3 shows all values were above this threshold. Adequate discriminant validity is demonstrated by the $\sqrt{}$AVE being greater than any correlation with another construct [15]. Table 3 shows that the $\sqrt{}$AVE of each construct was greater than its most significant correlation with another construct. Subsequently, these results imply construct validity.

4.2.3 Concurrent Validity

Adequate concurrent validity is established through a cursory examination of the correlation matrix and an examination of hypothesised relationships in a structural model. Table 3 does not show any anomalies within the correlation matrix. As such, the proposed structural model was assessed along with a self-report measure of programming practice. The results, shown in Table 4, reveal that most of the expected regression relationships were statistically significant. However, the regression between INT and ANX was not statistically significant. This suggests that either there is no relationship or the size of effect is small. Nevertheless, with the exception of INT, the conceptual model appears to be valid.

Table 1: Mean, Standard Deviation, Skewness and Kurtosis of the Instrument Items

Item	Item Description	*M*	*SD*	*Sk*	*K*
Debugging Self-Efficacy					
DSE1	I am confident that I can understand Java exceptions (e.g., NullPointerException)	3.65	0.96	-0.27	-0.61
DSE2	I am confident I can solve simple problems with my programs	3.48	1.02	-0.18	-0.55
DSE3	I am confident I can implement a method from a description of a problem or algorithm	3.87	0.98	-0.68	-0.26
DSE4	I am confident I can debug a program that calculates prime numbers	3.68	0.92	-0.35	-0.37
Programming Self-Concept					
PSC1	I am just not good at programming	2.44	1.18	0.44	-0.67
PSC2	I learn programming quickly	3.42	1.11	-0.28	-0.58
PSC3	I have always believed that programming is one of my best subjects	3.41	1.17	-0.28	-0.82
PSC4	In my programming labs, I can solve even the most challenging problems	3.34	1.10	0.07	-1.01
Programming Interest					
INT1	I enjoy reading about programming	3.66	1.10	-0.44	-0.50
INT2	I do programming because I enjoy it	3.93	0.95	-0.75	0.13
INT3	I am interested in the things I learn in programming classes	3.72	0.98	-0.52	-0.39
INT4	I think programming is interesting	3.87	1.03	-0.72	0.67
Programming Anxiety					
ANX1	I often worry that it will be difficult for me to complete debugging exercises	2.77	1.06	-0.27	-0.85
ANX2	I often get tense when I have to debug a program	2.83	1.18	-0.08	-0.95
ANX3	I get nervous when trying to solve programming bugs	2.82	1.16	0.06	-0.96
ANX4	I feel helpless when trying to solve programming bugs	2.76	1.21	0.11	-0.83
Programming Aptitude Mindset					
APT1	I have a fixed level of programming aptitude, and not much can be done to change it	2.08	0.97	0.82	0.35
APT2	I can learn new things about software development, but I cannot change my basic aptitude for programming	2.22	0.95	0.34	-0.59
APT3	To be honest, I do not think I can really change my aptitude for programming	1.90	0.92	0.87	0.23

Note: Pooled Sample (N = 175); M: mean, SD: standard deviation, Sk: skew, K: kurtosis.

Table 3: Construct Validity of the Latent Constructs in the Measurement Model

	Loadings	Reliability	Variance Explained			Correlations				
Items	*FL*	*CR*	*AVE*	*MSV*	*ASV*	DSE	PSC	INT	ANX	APT
Debugging Self-Efficacy		0.868	0.624	0.530	0.418	(0.790)				
DSE1	0.776									
DSE2	0.808									
DSE3	0.696									
DSE4	0.870									
Programming Self-Concept		0.703	0.655	0.494	0.413	0.703	(0.809)			
PSC1	-0.710									
PSC2	0.800									
PSC3	0.882									
PSC4	0.835									
Programming Interest		0.842	0.579	0.530	0.363	0.728	0.686	(0.761)		
INT1	0.781									
INT2	0.847									
INT3	0.846									
INT4	0.522									
Programming Anxiety		0.888	0.664	0.475	0.364	-0.681	-0.689	-0.507	(0.815)	
ANX1	0.817									
ANX2	0.762									
ANX3	0.834									
ANX4	0.844									
Programming Aptitude Mindset		0.865	0.682	0.262	0.214	-0.431	-0.462	-0.439	-0.512	(0.826)
APT1	0.782									
APT2	0.858									
APT3	0.836									

Note: Values on the diagonal represent $\sqrt{AVE}$; FL: factor loading, CR: composite reliability, AVE: average variance explained, MSV: maximum shared variance, ASV: average shared variance, DSE: debugging self-efficacy, PSC: programming self-concept, INT: programming interest, ANX: programming anxiety, APT: programming aptitude mindset.

5. DISCUSSION

Adequate measurement in computing education research is important. This is because researchers need to know whether the measures being selected and used by other researchers are valid. Straub *et al.* [43] highlight several key concerns that researchers may have: Does the instrument truly represent the essence or content of the target construct? Is the instrument unidimensional and therefore only representing the target construct? Has the target construct been confused with another similar construct? Are the estimates of the true values of latent constructs appropriate? Rigorous approaches to measurement address such questions.

Unfortunately, there has not been a strong history of reporting psychometric information in the field [37] and few measurement instruments are readily available to researchers in the computing education community. Particularly, measurement instruments that capture constructs concerned with the affective-domain of learning computer programming. This may be because developing adequate measurement instruments can be fraught with difficulties [43]. However, there is a strong case for pursuing such work [43, 44] and there is a range of literature which can be drawn from for support (e.g. [8, 18, 43]).

This paper has assembled one such measurement instrument and demonstrated that it has adequate psychometric properties in terms of reliability, construct validity, and concurrent validity. This instrument focuses on student self-beliefs in the introductory programming context and measures five different constructs: programming aptitude mindset, programming self-concept, debugging self-efficacy, programming anxiety, and programming interest.

It is interesting to note that interest in software development did not predict programming anxiety. In hindsight, other value appraisals such as 'importance of programming for future prospects' may have been more appropriate for anxiety. Nevertheless, programming self-concept and mindset towards programming aptitude were shown to be related to programming anxiety and, subsequently, programming practice behaviour. These relationships have not been firmly established as causal relationships nor are the directions of the relationship clear. This suggests that the measurement instrument will be useful for future work investigating hypotheses raised by the theory in, for example, longitudinal survey studies.

The measurement instrument may also be useful in other similar areas of work. There is a vast range of techniques which educators could attempt to apply in order to enrich their students' beliefs, practice, and performance (see [28]). Using a validated instrument, such as the one proposed here, will improve the rigor of such explorations. To illustrate, the authors previously embedded a fantasy role-play within an e-learning tool to evaluate its impact on students' programming self-concept through a pre-post experiment [39]. The ongoing development of this measurement instrument will support future such experiments, increasing confidence that such design experiments present useful and meaningful conclusions. Other uses of the measurement instrument may include educators using the measurement instrument to identify potential problems in their introductory programming classes or researchers evaluating student outcomes across different course designs and cohorts.

6. LIMITATIONS

It should be noted that this work only represents a first step and future development is needed to overcome a number of limitations. Most importantly, the instrument has only been administered to students at a single institution. Therefore, it may not generalise to populations from other higher education institutions; particularly, those with a different culture. Therefore, there is a need to further validate the tool beyond the institution. Of particular note, the cross-cultural validity of the measurement instrument also needs to be considered in addition to the appropriateness of adapting the framework for different educational contexts. In its present form, it is not clear whether the instrument would be suited for a range of programming topics or age groups.

A small number of items have been included in this scale to facilitate the collection of data from a large group with a short questionnaire. As such, it should not be used to make fine-grain judgements about any individual student. However, estimation of the true values of the latent constructs for individual students would likely improve with additional items.

7. CONCLUSION

Valid measurement is important, however only a small number of validated measurement instruments are available to computing education researchers. This limits research being conducted into educational theory, teaching practice and the the use of instructional technologies which aim to enrich beliefs and learning behaviour. The study presented in this paper contributes to this gap in the literature through the assembly and validation of a measurement instrument that could be used for such research. Specifically, for the investigation of student self-beliefs within the introductory programming context.

Three administrations of the instrument at the authors' institution demonstrated that the proposed measurement model had a good fit to the data. Furthermore, there was adequate support for reliability, construct validity, and concurrent validity. However, there are a number of limitations. Critically, the results may not generalise to different age-groups, cultures or educational contexts.

Future work will involve further validation of the conceptual framework in addition to an examination of appropriate descriptive statistics across a range of students and contexts. This will support further research into teaching practice and instructional technology used in introductory programming.

8. REFERENCES

[1] A. Bandura. Self-efficacy: Toward a unifying theory of behavioral change. *Psychological Review*, 84(2):191 – 215, 1977.

[2] J. E. Bartlett, J. W. Kotrlik, and C. C. Higgins. Organizational Research: Determining Appropriate Sample Size in Survey Research. Information Technology, Learning, and Performance Journal. 19(1):43 – 50, 2001.

[3] T. Beaubouef and J. Mason. Why the High Attrition Rate for Computer Science Students: Some Thoughts and Observations. *SIGCSE Bulletin*, 37(2):103 – 106, 2005.

[4] J. Bennedsen and M. E. Caspersen. Failure rates in introductory programming. *SIGCSE Bulletin*, 39(2):32 – 36, 2007.

[5] L. Blackwell, K. Trzesniewski, and C. S. Dweck. Implicit Theories of Intelligence Predict Achievement Across an Adolescent Transition: A Longitudinal Study and an Intervention. *Child Development*, 78:246 – 263, 2007.

[6] M. Bong and E. Skaalvik. Academic Self-Concept and Self-Efficacy: How Different Are They Really? *Educational Psychology Review*, 15(1):1 – 40, 2003.

[7] Q. Cutts, E. Cutts, S. Draper, P. O'Donnell, and P. Saffrey. Manipulating mindset to positively influence introductory programming performance. In *Proceedings of the 41st ACM technical symposium on Computer science education*, pages 431 – 435, 2010.

[8] R. F. DeVellis. *Scale Development: Theory and Applications*. Sage: London, 3rd edition, 2012.

[9] B. Dorn and A. E. Tew. Becoming experts: Measuring attitude development in introductory computer science. In *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13)*, pages 183 – 188, 2013.

[10] C. S. Dweck. *Self-Theories: Their Role in Motivation, Personality, and Development*. Psychology Press, Philadelphia, PA, 1999.

[11] C. S. Dweck and A. Master. Self-Theories Motivate Self-Regulated Learning. pages 31 – 51. Lawrence Erlabaum, New York, NY, 2008.

[12] J. S. Eccles, WigĭňĄeld, and A. In the mind of the actor: The structure of adolescents' achievement task values and expectancy-related beliefs. *Personality and Social Psychology Bulletin*, 3:215 – 225, 1995.

[13] K. Ericsson, R. Krampe, and C. Tesch-Romer. The Role of Deliberate Practice in the Acquisition of Expert Performance. *Psychological Review*, 100(3):363 – 406, 1993.

[14] J. Ferla, M. Valcke, and Y. Cai. Academic Self-Efficacy and Academic Self-Concept: Reconsidering Structural Relationships. *Learning and Individual Differences*, 19(4):499 – 505, 2009.

[15] C. Fornell, Larker, and D. F. Evaluating Structural Equation Models with Unobservable Variables and Measurement Error. *Journal of Marketing Research*, 18:39 – 50, 1981.

[16] F. Guay, H. W. Marsh, and M. Boivin. Academic self-concept and academic achievement: Developmental perspective on their causal ordering. *Journal of Educational Psychology*, 95:124 – 136, 2003.

[17] M. Guzdial. From Science to Engineering. *Communications of the ACM*, 54(2):37 – 39, 2011.

[18] J. Hair, B. Black, B. Babin, and R. Anderson. *Multivariate Data Analysis*. Psychology Press, NJ, USA, 7th edition, 2009.

[19] M. Huggard. Programming Trauma: Can it be Avoided? In *Paper presented at the BCS Conference on Grand Challenges in Computing: Education*, page 50, 2004.

[20] T. Jenkins. Teaching Programming: A Journey from Teacher to Motivator. In *2nd HEA Conference for the ICS-LTSN*, pages 53 – 58, 2001.

[21] T. Jenkins. On the Difficulty of Learning to Program. *3rd HEA Conference for the ICS-LTSN*, pages 1 – 8, 2002.

[22] P. Kinnunen and S. Beth. My Program is OK – Am I? Computing Freshman's Experience of Doing Programming Assignments. *Computer Science Education*, 22(1):1 – 28, 2012.

[23] P. Kinnunen and L. Malmi. Why Students Drop Out CS1 Courses? In *Proceedings of the 2006 International Computing Education Research Workshop*, pages 97 – 108, 2006.

[24] P. Kinnunen and B. Simon. Experiencing Programming Assignments in CS1: The Emotional Toll. In *Proceedings of the 6th International Workshop on Computing Education Research*, pages 77 – 86, 2010.

[25] R. B. Kline. *Principles and Practice of Structural Equation Modeling*. New York, NY: The Guilford Press, 2nd edition, 2005.

[26] D. R. Krathwohl, B. S. Bloom, and B. B. Masia. New York, NY: David McKay Co, 1973.

[27] I. M. Lyons and S. L. Beilock. When Math Hurts: Math Anxiety Predicts Pain Network Activation in Anticipation of Doing Math. *PLOS One*, 7:e48076, 2012.

[28] E. MacLellan. How might teachers enable self-confidence? A Review Study. *Educational Review*, 66(1):59 – 74, 2014.

[29] H. Marsh and A. Martin. Academic Self-Concept and Academic Achievement: Relations and Causal Ordering. *British Journal of Educational Psychology*, 81(1):59 – 77, 2011.

[30] A. McGettrick, R. Boyle, R. Ibbett, J. Lloyd, G. Lovegrove, and K. Mander. Grand Challenges in Computing: Education - A Summary. *The Computer Journal*, 48(1):42 – 48, 2005.

[31] D. McKinney and L. F. Denton. Houston, We have a Problem: There's a Leak in the CS1 Affective Oxygen Tank. *SIGCSE Bulletin*, 36(1):236 – 239, 2004.

[32] L. Murphy and L. Thomas. Dangers of a Fixed Mindset: Implications of Self-Theories Research for Computer Science Education. *SIGCSE Bulletin*, 40(3):271 – 275, 2008.

[33] R. Pekrun. The control-value theory of achievement emotions: Assumptions, corollaries, and implications for educational research and practice. *Educational Psychology Review*, 18(4):315 – 341, 2006.

[34] R. Pekrun and E. J. Stephens. Achievement Emotions: A Control-Value Approach. *Social and Personality Psychology Compass*, 4:238 – 255, 2010.

[35] A.-K. Peters and A. Pears. Engagement in Computer Science and IT – What! A Matter of Identity? In *Learning and Teaching in Computing and Engineering Conference*, pages 114 – 121, 2013.

[36] L. Porter and B. Simon. Retaining nearly one-third more majors with a trio of instructional best practices in CS1. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, pages 165 – 170, 2013.

[37] J. Randolph, G. Julnes, E. Sutinen, and S. Lehman. A Methodological Review of Computer Science Education Research. *Information Technology Education*, 7(1):135 – 162, 2008.

[38] C. Rogerson and E. Scott. The Fear Factor: How it Affects Students Learning to Program in a Tertiary Environment. *Information Technology Education*, 9(1):147 – 171, 2010.

[39] M. J. Scott and G. Ghinea. Integrating Fantasy Role-Play into the Programming Lab: Exploring the 'Projective Identity' Hypothesis. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education*, pages 119 – 122.

[40] M. J. Scott and G. Ghinea. Educating Programmers: A Reflection on Barriers to Deliberate Practice. In *Proceedings of the 2nd HEA Conference on Learning & Teaching in STEM Disciplines*, pages 85 – 90, 2013.

[41] M. J. Scott and G. Ghinea. On the Domain-Specificity of Mindsets: The Relationship Between Aptitude Beliefs and Programming Practice. *IEEE Transactions on Education*, in print.

[42] B. Simon, B. Hanks, L. Murphy, S. Fitzgerald, R. McCauley, L. Thomas, and C. Zander. Saying Isn't Necessarily Believing: Influencing Self-Theories in Computing. In *Proceedings of the 4th Int. Workshop on Computing Education Research*, pages 173 – 184, 2008.

[43] D. Straub, M.-C. Boudreau, and D. Gefen. Validation Guidelines for IS Positivist Research. *Communications of the Association for Information Systems*, 13:380 – 427, 2004.

[44] A. E. Tew and B. Dorn. The Case for Validated Tools in Computing Education Research. *Computer*, 46(9):60 – 66, 2013.

[45] A. E. Tew and M. Guzdial. The FCS1: A Language Independent Assessment of CS1 Knowledge. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, pages 111 – 116, 2011.

[46] D. W. Valentine. CS Educational Research: A Meta-Analysis of SIGCSE Technical Symposium Proceedings. In *Proceedings of the 35th ACM Technical Symposium on Computer Science Education*, pages 255 – 259, 2004.

[47] S. Wiedenbeck. Factors affecting the success of non-majors in learning to program. In *Proceedings of the 1st Int. Workshop Computing Education Research*, pages 13 – 24, 2005.

[48] A. Wigfield, J. S. Eccles, K. S. Yoon, R. D. Harold, A. J. A. Arbreton, and C. Freedman-Doan. Change in children's competence beliefs and subjective task values across the elementary school years: A 3-year study. *Journal of Educational Psychology*, 89:451 – 469, 1997.

[49] A. Wigfield, Meece, and J. L. Math anxiety in elementary and secondary school students. *Journal of Educational Psychology*, 80(2):210 – 216, 1988.

[50] B. C. Wilson and S. Shrock. Contributing to success in an introductory computer science course: A study of twelve factors. *SIGCSE Bulletin*, 33(1):184 – 188, 2001.

[51] L. E. Winslow. Programming Pedagogy - A Psychological Overview. *SIGCSE Bulletin*, 28(1):17 – 22, 1996.