# Exploring Learner Conceptions of Programming

**Errol Thompson**
Department of Information Systems
Massey University
Private Box 756, Wellington
New Zealand
Telephone +64 4 8015799 x6531
E.L.Thompson@massey.ac.nz

**Lynn Hunt**
Department of Management and International Business
Massey University
Private Bag 102 904, Albany
New Zealand
Telephone +64 9 414 0800 x9282
L.M.Hunt@massey.ac.nz

**Kinshuk**
Department of Information Systems
Massey University
Private Bag 11 222, Palmerston North
New Zealand
Telephone +64 6 356 9099 x2090
Kinshuk@massey.ac.nz

## Abstract

The objective of this paper is to argue for an approach to the scholarship of teaching based on conceptions of learning and task representation. A case study based on learning to implement a data structure using test-driven development is presented to illustrate the issues. Documented phenomenographic investigations into concepts of programming are referenced to lay a foundation for future research into improving learning and teaching of programming themes.

*Keywords:* Conceptions of programming, learning process, task representation

## 1 Introduction

Although a lecturer may specify learning objectives or outcomes, using an established cognitive hierarchy such as Bloom's taxonomy (Anderson et al. 2001; Bloom et al. 1956), a learner may not achieve the required level of outcome. Why does a learner fail to achieve the required level of outcome? Does the learner understand the required learning in the same way that the lecturer specified it? In this paper, it is argued that the student's conception of what is required, in other words their representation of the task, determines the outcome. If that representation does not match the lecturer's specification, then there is a possibility the student will fail to achieve the intended learning.

To illustrate this argument, a case study based on the teaching of the programming of data structures is used. The case study is discussed in terms of the lecturer's objectives, the approaches taken by students to complete the learning, and some of the possible reasons for the approaches taken.

To identify different approaches used by students, a model of learning based on the relational view of teaching and learning is applied. The relational view of learning contends that learning is the result of a change of in the way that the learner views the phenomenon or in their awareness of the phenomenon (Marton & Booth 1997; Ramsden 1988, pp. 13-31). This view of learning argues that learning outcomes are based on the student's learning strategies and their conceptions of learning and the subject (Bowden & Marton 1998, p. 310 p.; Ramsden 2003). Booth's conceptions of programming (Booth 1992, pp. 122-145 1992, p. 308 1997, pp. 135-158) are used to discuss the possible conceptions held by the learners in the case study. These conceptions of programming are also used to propose an approach to teaching aimed at fostering change in learner conceptions.

The approach to gathering data is based on a classroom research strategy where focus is placed on the small group of exceptions rather than endeavouring to analyse a representative sample (Cross & Steadman 1996). In this particular case study, a small group of students who completed the exercise quickly is compared and contrasted with a small group of students who were unable to complete the exercise as defined.

The next section describes the teaching case study. This is discussed in terms of what the learners where expected to do and then what the learners actually did. This is followed by a discussion of the relational model of learning and research outcomes that are specific to programming and the teaching case.

## 2 A Teaching Case Study

### 1.1 Case Background

This case is based on the teaching of part of a second year Advanced Programming paper utilizing Visual Basic .NET and the NUnit testing framework. Learners taking this paper had already completed two semesters of programming in Visual Basic 6. In these previous papers, there had been an emphasis on the development of testing plans and manual testing of completed code. Learners had been expected to submit with their completed code, testing logs and sample desk checks. Some of the learners had also taken or were concurrently taking papers in Java and/or C programming.

The analysis presented in this paper is based on in-class observations and discussions. The class had approximately twenty students and the students where given the opportunity to work in self selected groups.

The work of the group who completed the exercise was presented to the class as a solution to the learning exercise and analysed by all class members. There were other successful groups but there work was not collected for this study. More time, in tutorial sessions, was spent with the group of students who were unable to complete the exercise, with the intent to understand and to help them with their difficulties.

The core focus of the paper was on teaching object-oriented programming using a test-driven approach. The object-oriented concepts were introduced through coding exercises with provided automated test cases. Initially, the learners were provided with the automated test cases but these were slowly removed as the paper advanced until they were designing and coding their own tests cases.

There was a strong emphasis on learning the test-driven development process as an approach to object-oriented software development. Objects had been presented as mini-systems that could be developed and tested independently and then combined to achieve the greater goal of the application being developed.

Having fostered the learning of the process and the basics of using classes during the first half of the paper, the attention then shifted to issues in the use of objects to achieve specific programming tasks. It was envisaged that the learners would utilize their process learning to build their understanding of these new and unfamiliar tasks.

## 1.2 Working with Data Structures

The lecturer's primary focus was on the learner being able to develop a piece of code in a known object-oriented programming language (i.e. Java or .NET) to implement a data structure (i.e. a linked list) using a test-driven approach to programming (Astels 2003; Beck 2003). Before the learner was introduced to this exercise, they had already learnt enough of the programming language to write small applications using an object-oriented approach and test-driven development tools. The learner had also worked through a number of examples of developing code using the test-driven approach. The lecturer believed, from experience with test-drive techniques and from recorded examples (Astels 2003; Beck 2003), that with this background knowledge, the learner would be able to develop a series of tests that enabled the development of the data structure even though they had never seen code to implement the data structure in the programming language they were using.

Before the learners began work on the exercise, the lecturer went over the operation of the data structure using diagrams starting from an empty structure and slowly building it until a full working model had been developed. During each phase of this explanation, the lecturer emphasized the conditions that should exist within the data structure prior to the current operation and the conditions after the completion of the operation on the data structure. The lecturer believed that these conditions should translate to the tests that needed to be implemented for a test-driven development solution.

A key issue with test-driven development is that only the code required to pass the current test is implemented. At the beginning of a coding exercise, it is not necessary to know exactly what the end result (i.e. a fully implemented data structure) will look like. Usually a visual representation (i.e. a metaphor) of what is required is sufficient and an ability to identify a sequence of tests that will gradually grow in complexity. In this case, the lecturer has identified the key classes, their attributes, and methods, and the conditions required for the tests.

## 1.3 Expected Level of Learning Outcome

The lecturer developed a number of learning objectives based on of the taxonomy of learning objectives as described by Anderson et al. (Anderson et al. 2001),. This taxonomy is a revision of Bloom's taxonomy (Bloom et al. 1956). It uses a matrix with a knowledge dimension and a cognitive process dimension. The knowledge dimension categories are factual knowledge, conceptual knowledge, procedural knowledge, and metacognitive knowledge. The cognitive process dimension categories are remember (recognise and recall), understand (interpret, exemplify, classify, summarise, infer, compare, and explain), apply (execute, and implement), analyse (differentiate, evaluate, and attribute), evaluate (check, and critique), and create (generate, plan, and produce). A learning objective is specified in terms of a knowledge dimension category and the cognitive processing dimension category that should be applied. It is expected that for higher level cognitive processing categories that the lower level categories are also used.

In the case study, the learner would use prior knowledge of the language and test-driven development. However, the learning exercise itself involved multiple stages each with its own learning objective. Some of these objectives were:

1. Apply the known test-driven development method (process knowledge) to an unfamiliar task, the development of an implementation of the data structure (implementing)

2. Apply the known programming language knowledge (factual knowledge) to an unfamiliar task (implementing)

3. Apply known object-oriented concepts knowledge (conceptual knowledge) to an unfamiliar task (implementing)

4. Apply awareness of the data structure concepts and the operational conditions (conceptual knowledge) to develop a sequence of tests (conceptual knowledge) to create a valid working version of the data structure (implementing)

5. Verify that their code satisfies the known coding standards for the programming exercise (factual knowledge, checking)

6. Explain what their tests and code is doing (factual knowledge) and why they are using those constructs (explaining)

7. Evaluate (critiquing) their code and test strategy (factual knowledge) for effectiveness and performance

8. Evaluate their implemented code with respect to the criteria for an effective implementation of the data structure (conceptual knowledge, checking)

9. Utilise the known test-driven development process as a mechanism for learning how the data structure works (metacognitive knowledge, executing)

When specifying the objectives, there was an expectation that the learner would integrate a range of knowledge elements to accomplish the specific task. Specifying the integration of these techniques and knowledge elements requires the statement of multiple objectives. This expectation of integration is reflected in the relational and extended abstract levels of the SOLO taxonomy (Biggs & Collis 1982).

SOLO (Structured Observed Learning Outcome) is an alternative method of defining the required level of learning outcome (Biggs & Collis 1982). In SOLO, the focus is on the structure of the knowledge and the relationships between knowledge elements. Hattie and Purdie (Hattie & Purdie 1998, pp. 145-176) contend that although there are similarities between the learning objectives taxonomy (Bloom et al. 1956) and the SOLO taxonomy (Biggs & Collis 1982), that there are also fundamental differences. The main difference is that the learning objectives taxonomy is based on 'behavioural tendency' models while the SOLO taxonomy is based on the cognitive processing model.

Rather than focussing on the cognitive skills (behavioural characteristics) that should be utilised in completing an assessment, SOLO uses an alternative assessment proposal based on the structural complexity and relationships in the knowledge (Biggs & Collis 1982; Hattie & Purdie 1998, pp. 145-176). Hattie and Purdie (Hattie & Purdie 1998, pp. 145-176) contend that SOLO is based on "the processes of understanding used by the students" (p 160).

The Structure of the Observed Learning Outcome (SOLO) has five levels. These are

- prestructural – a learner operating at this level performs the preliminary preparation but does not complete the learning task or does not approach the task in an appropriate way

- unistructural – a learner operating at this level can recall individual knowledge elements or performs aspects of a learning task but does not see any sequence or relationship in the knowledge elements or task aspects

- multistructural – a learner operating at this level relates the knowledge elements in an appropriate sequence or completes aspects of a task in sequence but fails to see the interrelated nature of the knowledge or the task aspects

- relational – a learner operating at this level integrates the knowledge elements into a coherent structure and meaning, and performs a task as an integrated whole

- extended abstract – a learner operating at this level applies higher levels of abstraction to the coherent body of knowledge

The development of the data structure code using a test-driven development strategy could not be completed satisfactorily using individual pieces of knowledge or the application of a single technique (SOLO's unistructural level) or with generalising from a limited amount of knowledge (SOLO's multistructural level). The knowledge had to be combined in a way that worked toward the targeted outcome, an implementation of the data structure with its associated set of tests that verify its execution (SOLO's relational level).

The lecturer expected the learner to work with the conceptual model of the data structure and the supplied operational conditions drawing from it appropriate test cases which are then implemented in the testing framework. In order to write a test within the testing framework, the learner had to picture the design of the classes that would be used in the data structure implementation. Finally, the learner had to code the classes so that they passed the test cases that they had selected and implemented. The learner integrated a range of knowledge to produce the required range of outputs. In the SOLO taxonomy, this represents performance at the relational level.

The lecturer also looked for signs that the learner had evaluated the exercise to draw out some principles that might be used to implement a completely different data structure or program architecture (SOLO's extended abstract level).

## 1.4 Learner Approaches to the Task

One group of learners searched the library and the internet for example code in the required language rather than working through the concepts and developing a test strategy based on the illustrated operation. The search for example code bypassed some of the lecturer's objectives for the exercise by removing the need for a conceptual understanding of the operation of the data structure. The learners stated that they needed to see a code example in order to understand how the data structure worked. From the lecturer's perspective, the learners reduced the exercise to copying a specific implementation of the data structure (factual knowledge, recalling). It allowed the learners to focus on the

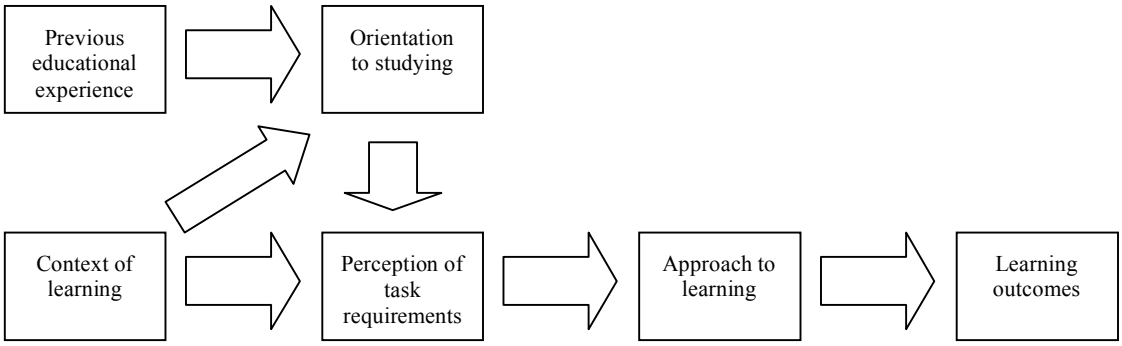


**Figure 1: Ramsden's "Student learning in context" model**

problem by completing one part at a time or in a unistructural manner.

On failing to find example code, the learners complained that it was not possible to complete the exercise since there was no example code to follow. Although the learners had completed previous test-driven development exercises, they still lacked the knowledge to use test driven development to complete an unfamiliar programming task. This group stated that they needed example code in order to comprehend the operation of the data structure. They also lacked an understanding of why the learning exercise was given. The learners' perceptions appeared to be that the focus of the exercise was on knowing how to use the programming language and not learning to apply the test-driven strategy to implement and understand the operation of the data structure.

When examining the learners' approaches in terms of learning outcomes, the learners had focused on those related to coding in the required language. The lecturer assessed that the higher level conceptual and process learning was not achieved by these learners.

In contrast, another group of learners worked stepwise through the operation of the data structure defining the tests for each of the required conditions for successful completion of each step and created working code for the data structure. Although these learners also started with no knowledge of how to implement the data structure in the language being used, these learners successfully developed a conceptual understanding of the data structure and successfully applied the test driven development to build knowledge of how to implement the data structure in the specified language.

These learners demonstrated not only a comprehension of the data structure but an understanding of the operation of the data structure. As well, these learners demonstrated that they understood test driven development by applying a test driven approach to the development of a working solution for the data structure. Their solution revealed an understanding of the features to select in the testing framework and the programming language to achieve the desired outcome.

These learners achieved all of the learning objectives and developed a relational level understanding of the content of learning. The learners used the range of desired skills in unison and in the process built an understanding of the data structure and its implementation.

This group of learners on completion of the exercise was able to describe the operation of the data structure and the reasoning for the approach that they had taken and the tests that they had implemented. From a SOLO perspective, they were beginning to operate at an extended abstract level.

Of concern to the lecturer was the variation in responses to the exercise. In the lecturer's evaluation, the first group of learners appeared to have misunderstood the level of knowledge required and the primary focus of the exercise. The second group of learners appeared to be more on target with the lecturer's objectives of the exercise. Was this a problem with the way the exercise was expressed, the approach taken by the learner to the exercise, or some other cause?

## 3 Relating To a Theoretical Framework

### 1.5 An Educational Model

Biggs proposed a 3P (presage, process, product) model to illustrate the impacts on learning outcomes (Biggs 1993, pp. 73-85; Biggs & Moore 1993). He proposed that the product of learning or the learning outcome is the result of the process (learning focussed activities). This in turn is influenced by student factors and the teaching context (presage). Amongst the student factors are issues such as the learner's comprehension of the requirements and the concept of learning. Biggs recognises that there are influences in both directions between the teaching context and the student factors.

Ramsden (Ramsden 2003) places more emphasis on the perception of task requirements as an influence on the approach to learning in his model (Figure 1). Hunt (Hunt 1995) refers to this as the task representation. In Ramsden's model the perception of the task requirements is influenced by the learner's orientation to study, and the context of learning. The orientation to learning is seen as being influenced by previous educational experience and the context of learning.

Ramsden warns against taking the diagram as being deterministic. He contends that it helps us "to reason about possible relationships between different aspects of learning and teaching" (p 81). There may be other influences that are not depicted in the diagram. The

connections highlight points of possible intervention where it might be possible to influence the learner and as a consequence enhance learning. For example, if the lecturer is able to apply an intervention that changes the learner's perception of the task requirements then the learner may change their approach to learning in a way that delivered an improved learning outcome.

In order to understand why a learner approached the learning in a way that was different to the lecturer's conception of the task, it is necessary to examine the learner's perception of the task requirements or their task representation.

## 1.6 Application to Programming

In the phenomenographic approach to research, emphasis is placed on the conceptions or ways of experiencing that the learner has in relation to the phenomenon or object of learning. It is argued that in the relationship between a learner and a phenomenon there is a finite set of distinct descriptions that describe the learner's conceptions. These descriptions can be organized in a sequence that identifies increasing understanding or awareness of the phenomenon by the learner.

Booth has utilized the phenomenographic approach to explore learners' conceptions of learning to program (Booth 1992, pp. 122-145 1992, p. 308 1997, pp. 135-158). In her studies, she has sought to identify how the learners conceptualize the task and what they are endeavouring to achieve. Some of these are specific to learning to program specific concepts such as recursion. Others are more general and focus on the learners' conceptions of the nature of programming and of programming languages. Those that are of most relevance to this case relate to the conceptions of learning to program and approaches to writing programs.

With respect to learning to program, Booth identifies four conceptions (pp 119-129):

- Learning to program as learning a programming language
- Learning to program as learning to write programs in a programming language
- Learning to program as learning to solve problems in the form of programs
- Learning to program as becoming a part of the programming community

These conceptions parallel the cognitive skills hierarchy in the sense that the first conception of "learning to program as learning a programming language" would tend to utilize memorization strategies while the last conception, "learning to program as becoming part of the programming community" would involve more reflection and deeper understanding of the issues.

Since the case was not studied from this perspective, it is difficult to determine the conceptions that are closest to the two groups of learners described in the case. Phenomenographers argue that the descriptions are not related to a particular learner's conception but are drawn from the population under study. A particular learner may at different times, exhibit different conceptions.

In reviewing the evidence from the case, the struggling group of learners appears to be focused more on "learning to program as learning a programming language" or "as learning to write programs in a programming language". This conclusion is drawn from the students' insistence on needing example code in the language. The successful group of learners may be more focused on "learning to solve problems in the form of programs". This is unclear from the case notes. These students did describe the operation and conditions of the data structure in their description of their program code.

The exercise, as conceptualized by the lecturer, was focused on "learning to solve problems in the form of programs" with a minor emphasis on "becoming part of the programming community". The lecturer's conception was that the skills and techniques being developed in the study are designed to enable the learners to develop those skills and techniques required to participate in the programming community. However, the teaching style was more focused on the solving of problems in the form of programs.

The issue here is that by understanding the conceptions that learners may have, the lecturer may be able to apply appropriate interventions to encourage change in their conceptions. Without this awareness, the lecturer may have difficulty encouraging the struggling learners to overcome the difficulties.

Booth's conceptions of approaches to programming seem to be easier to apply than the conception of learning to program to this case study. The distinction between these two sets of conceptions is that the first focuses on how the learner conceives learning to program. This set of conceptions focuses on how the learner approaches the programming task. These conceptions (pp 203-238) are described as:

- Expedient approach where the focus is on utilizing existing programs or adapting some known programs to the requirements of the new problem
- Constructional approach where features of the programming language are recognized as relating to details of the problem and can be used to construct an appropriate program
- Operational approach where the focus is on "what operations the program has to perform"
- Structural approach where the structure of the problem is used to devise a program solution

The learners who sought example code possibly saw the data structure and its object-oriented representation only in terms of a language specific implementation. It would appear as though these learners were unaware that they may have been able to use an example from another programming language or an understanding of the concept of the data structure and object-oriented principles independently of a particular language

implementation to create an implementation in the language currently being used.

This conception may be valid in other situations and as a result, these learners may be successful in other programming classes or contexts. However, in the situation described in the case, these learners struggled and found it difficult to achieve the required learning objective.

Using Booth's conceptions of approaches to writing programs, these learners might be judged as using an expedient approach to writing programs. They sought existing code that can be adapted to the new programming problem. Is this the conception that these learners brought to the coding of the data structure?

In contrast, the second group of learners who developed a working model by designing an appropriate testing strategy would appear to see the data structure as a concept independent of a specific language implementation. In the approach to programming, these learners appeared to utilize an operational approach to writing programs. They expressed an understanding of the operation of the data structure and from that perspective developed the code to implement the desired data structure.

If the conception of programming is influencing the learning outcome then to improve learning it is desirable to change the conceptions of the learner and therefore their representation of the task (Ramsden 2003).

## 1.7 Other Conceptualization Issues

Both of these groups of learners learnt to program using a procedural programming paradigm. This learning exercise was to be completed using an object-oriented paradigm. Although there is ongoing debate about the approach to introducing learners to the object-oriented paradigm (Bergin 2000), there are no studies of the conceptions of learners with respect to the procedural or object-oriented paradigms and the impact of these conceptions on the learning of these paradigms.

Neither are there studies that show how the conceptions of objects as utilized in object-oriented programming impact on a learner's success in object-oriented programming. If these conceptions of objects were available, how would they be changed by the programming paradigm in which the learner developed their initial programming skill?

Although this case study may have revealed such conceptions and their effectiveness, no data was gathered that would have enabled such analysis. One of the authors has used the concepts of procedural, procedural objects, and object-oriented as a way of describing a possible transition path from the procedural paradigm to the object-oriented paradigm but empirical studies are required to verify these conceptions and their effectiveness in transitioning learners. These conceptions are also not necessarily those that the learners have as they approach the programming tasks.

A procedural paradigm may be described as focusing on control structures with particular emphasis on sequences (Booth 1992, p. 308; Dale & Weems 1992). A procedural objects paradigm is where classes are implemented as code modules. That is, the class is seen as a way of encapsulating the data and methods for a common piece of functionality. The object-oriented paradigm may be introduced as a community of interacting entities (Stein 2003). Are these two extremes of an imperative programming paradigm or do they represent fundamentally different programming paradigms?

Answers to these questions may assist in resolving learning difficulties of some learners and to address some of the issues in the procedural verses object-first debate.

## 1.8 A Teaching Strategy

If understanding learner conceptions and applying interventions at the appropriate times will foster improved learning then how can the lecturer achieve these goals?

Phenomenography approaches suggest that the teacher needs to learn to see the object of learning in the same way as the learner sees it (Marton & Booth 1997). That is the teacher needs to understand the variations of conceptions that the learners bring to the classroom and how to challenge those variations to improve learning. Through this process and by focusing on the learners' awareness of the object of learning, the teacher helps to mould that object of learning for the learner (Marton & Tsui 2003).

Cope et al. (Cope, Garner & Prosser 1996, pp. 125-129) endeavoured to use learner conceptions of Information Systems and of learning as a way of generating discussion of the conceptions held by learners and to foster the use of those conceptions that bring deeper understanding of the subject. By having the learners describe their conceptions and then through sharing and comparing with others their conceptions, the learners were challenged to rethink their conceptions of what they were learning and how they were approaching that learning. This proved effective were the learner conceptions of Information Systems had previously been identified and were of relevance to the learners (Cope 2002, pp. 67-79).

By applying a similar approach to Cope, it might be expected that in a programming class, the conceptions of programming expressed would relate to those from Booth's study (Booth 1992, p. 308). Using the conceptions expressed by the learners and those from Booth's study as a starting point for discussion may cause some learners' conceptions to move toward those that fostered greater understanding.

Although Booth did not intervene in this way in her study, she did track changes in perceptions during the teaching of a paper. These were seen to change during the paper but not necessarily for greater understanding.

Cross and Steadman (Cross & Steadman 1996) contend that the scholarship of teaching should mean that the teacher is endeavouring to discover what is happening in

their classroom. Why are the learners not learning as expected? What is the impact of changes in teaching strategy? The teacher should aim through classroom research to better understand what is happening in the classroom and how to improve the learning that is occurring.

Based on these studies, a teaching strategy that regularly enables review and discussion of conceptions is likely to aid the lecturer in becoming aware of the learners conceptions of what is being taught. These regular reviews would also enable the lecturer to determine whether the learner conceptions where changing as desired. The changes in conceptions should then be reflected in improved selection of learning strategies or approaches to learning and improved learning outcomes.

In areas where the variations in learner conception are not known to the lecturer, then exercises in uncovering and discussing the conceptions would assist the lecturer to better understand the conception hierarchies that are assisting learners and to uncover interventions that would assist the learners.

## 4 References


Anderson, LW, Krathwohl, DR, Airasian, PW, Cruikshank, KA, Mayer, RE, Pintrich, PR, Raths, J & Wittrock, MC (eds.) 2001, *A taxonomy for learning and teaching and assessing: A revision of Bloom's taxonomy of educational objectives*, Addison Wesley Longman.

Astels, D 2003, *Test-driven development: A practical guide*, Prentice Hall PTR, Upper Saddle River.

Beck, K 2003, *Test-driven development by example*, Addison Wesley, Boston, MA.

Bergin, J 2000, *Why procedural is the wrong first paradigm if OOP is the goal*. from http://csis.pace.edu/~bergin/papers/Whynotproceduralfirst.html.

Biggs, JB 1993, 'From theory to practice: a cognitive systems approach', *Higher education research and development,* vol. 12, no. 1, pp. 73-85. Retrieved: 1993, from

Biggs, JB & Collis, KF 1982, *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcome)*, Academic Press, New York.

Biggs, JB & Moore, PJ 1993, *The process of learning,* 3rd edn, Prentice Hall, New York.

Bloom, BS, Engelhart, MD, Furst, EJ, Hill, WH & Krathwohl, DR 1956, *Taxonomy of educational objectives Handbook 1: cognitive domain*, Longman Group Ltd, London.

Booth, SA 1992, 'The experience of learning to program. Example: recursion', in *5 éme Workshop sur la Psychologie de la Programmation (the fifth workshop of the Psychology of programming Interest Group)*, ed. Détienne, F, INRIA, Paris, pp. 122-145.

Booth, SA 1992, *Learning to program: A phenomenographic perspective*, Acta Universtatis Gothoburgensis, Göteborg.

Booth, SA 1997, 'On phenomenography, learning and teaching', *Higher Education Research & Development,* vol. 16, no. 2, pp. 135-158.

Bowden, J & Marton, F 1998, *University of learning*, Routledge Falmer, London.

Cope, C 2002, 'Educationally critical aspects of the concept of an Information System', *Informing Science,* vol. 5, no. 2, pp. 67-79.

Cope, C, Garner, M & Prosser, M 1996, 'Using phenomenographic perspectives in the classroom', in *Annual Conference of the Higher Education Research and Development Society of Australasia*. Higher Education Research and Development Society of Australasia Inc., Sydney, pp. 125-129,

Cross, KP & Steadman, MH 1996, *Classroom research: Implementing the scholarship of teaching*, Jossey-Bass Publishers, San Francisco.

Dale, N & Weems, C 1992, *Introduction to Pascal and structured design,* 3rd Turbo version edn, Heath, Lexington, Mass.

Hattie, J & Purdie, N 1998, 'The Solo model: Addressing fundamental measurement issues', in Dart, B & Boulton-Lewis, GM (eds.), *Teaching and learning in higher education*, Australian Council of Educational Research, Camberwell, Vic, pp. 145-176.

Hunt, LM 1995, *Approaches to learning: The selection and use of learning strategies*, Dissertation, Massey University.

Marton, F & Booth, SA 1997, *Learning and awareness*, Lawrence Erlbaum Associates, Mahwah, NJ.

Marton, F & Tsui, ABM (eds.) 2003, *Classroom discourse and the space of learning*, Lawrence Erlbaum Associates, Publishers, Mahwah, NJ; London.

Ramsden, P 1988, 'Studying learning: Improving teaching', in Ramsden, P (ed.), *Improving learning: new perspectives*, Kogan Page; Nichols Pub. Co, London: New York, NY, pp. 13-31.

Ramsden, P 2003, *Learning to teach in higher education,* 2nd edn, Routledge Falmer, London.

Stein, LA 2003, *Interactive programming in Java*. Retrieved: 28 September, from http://www.cs101.org/ipij/.