Going SOLO to Assess Novice Programmers

Judy Sheard
Faculty of Information Technology
Monash University, Australia
+61 (3) 9903 2701
Judy.Sheard@infotech.monash.edu.au

Angela Carbone
Faculty of Information Technology
Monash University, Australia
+61 (3) 9903 1911
Angela.Carbone@infotech.monash.edu.au

Raymond Lister
Faculty of Information Technology
University of Technology, Sydney
Broadway, NSW 2007, Australia
+61 (2) 9514 1850
raymond@it.uts.edu.au

Beth Simon
Computer Science and Eng.Dept.,
University of California, San Diego,
CA 92093, USA
+1 (858) 534-5419
esimon@cs.ucsd.edu

Errol Thompson
2 Haven Grove, Lower Hutt,
New Zealand
kiwiet@computer.org

Jacqueline L. Whalley
School of Computing and Mathematical Sciences
Auckland University of Technology
Auckland 1020, New Zealand
+64 (9) 921 9999 x5203
jacqueline.whalley@aut.ac.nz

ABSTRACT

This paper explores the programming knowledge of novices using Biggs' SOLO taxonomy. It builds on previous work of Lister et al. (2006) and addresses some of the criticisms of that work. The research was conducted by studying the exam scripts for 120 introductory programming students, in which three specific questions were analyzed using the SOLO taxonomy. The study reports the following four findings: when the instruction to students used by Lister et al. – "In plain English, explain what the following segment of Java code does" – is replaced with a less ambiguous instruction, many students still provide multistructural responses; students are relatively consistent in the SOLO level of their answers; student responses on SOLO reading tasks correlate positively with performance on writing tasks; postgraduates students manifest a higher level of thinking than undergraduates.

Categories and Subject Descriptors

K.3 [**Computers & Education**]: Computer & Information Science Education - *Computer Science Education*.

General Terms

Measurement, Experimentation, Human Factors.

Keywords

Novice programmers, CS1, comprehension, SOLO taxonomy.

1. INTRODUCTION

A number of psychological studies have shown that expert programmers organize their knowledge of code segments into a coherent whole the function performed by the code whereas novices may understand the parts of a program but struggle to organize those parts into a coherent whole (McKeithen, Reitman, Rueter & Hirtle, 1981; Adelson, 1984; Wiedenbeck, Fix and Scholtz, 1993). However, those psychological experiments do not lead directly to teaching techniques that help students see the relationships between the parts, nor are these psychological experiments viable approaches to assessing whether novices in a university environment have acquired the ability to see the relationships between the parts.

The BRACElet project (Lister et al., 2006) introduced a problem which represents a pragmatic approach to both teaching and assessing this type of knowledge in novice programmers. Students were instructed "*In plain English, explain what the following segment of Java code does*", where the code provided by Lister et al. is shown in Figure 1. The student responses to this question were then classified by BRACElet members according to the first four levels of the SOLO taxonomy (Biggs and Collis, 1982):

- **Prestructural ("P"):** This is the least sophisticated SOLO response, where the student manifests a significant misconception, or uses a concept irrelevant to programming.
- **Unistructural ("U"):** The student manifests a correct grasp of a part of the problem. For example, a student describes the functioning of one or two lines of code.
- **Multistructural ("M"):** The student manifests an understanding of most lines of code, but does not manifest an awareness of how the code as a single coherent whole – the student "fails to see the forest for the trees". For example, a student might translate each line of code into pseudo code.
- **Relational ("R"):** The student manifests an understanding of the code as a single coherent whole, by describing the function

performed by the code – the student "sees the forest". For the code given in Figure 1, a relational response might be "*checks to see if the array is sorted*".

```
boolean bValid = true;
for(int i = 0; i<iNumbers.length-1; i++)
{
    if(iNumbers[i] > iNumbers[i+1])
    {
        bValid = false;
    }
}
```

Figure 1: The *loop* code used in the BRACElet project

Among the students studied in the BRACElet project (Lister et al., 2006) one third provided relational responses and one half provided multistructural responses. Furthermore, the students studied could be assigned to performance quartiles, based upon programming-related multiple choice questions answered by the students. Approximately half of the students in each of the top two quartiles manifested a relational response to the 'explain in plain English' question, while multistructural responses dominated in the lower two quartiles. Lister et al. asserted that "*students who cannot read a short piece of code and describe it in relational terms are not intellectually well equipped to write similar code*" (page 122).

1.1 Research Questions

This paper extends the earlier work of the BRACElet project by exploring four research questions:

1. Is the instruction – '*In plain English, explain what the following segment of Java code does*' – ambiguous? Students capable of providing a relational answer may have thought that a line by line multistructural explanation was required. In this study we reworded the instruction to: "*Explain the purpose of the following segment of code*". We then explored the possible ambiguity of the new instruction by using two questions (*loop* and *swap*, Figures 1 and 2). If a student answers one question relationally but not the other, it is implausible to argue that the student understood the instruction for one question and not the other.

   We further explored this possible ambiguity by using a third question which asks students to supply a name for a method that "*reflects its purpose*" thus forcing a relational type response.

2. If asked more than one question, does a student tend to provide answers at the same SOLO level?
3. Consistent with the conjecture made by Lister et al., is there a correlation between student SOLO performance on reading tasks and on code writing tasks?
4. How do SOLO responses between postgraduate and undergraduate students compare in an introductory programming unit?

2. RESEARCH APPROACH

Our data is from written student responses in two exams, one for a class of undergraduates, the other for a class of postgraduates. Both courses are run within the faculty of the first two authors, and both courses are an introduction to programming, which students usually take at the commencement of their degree. The students sat the exam at the end of the semester.

2.1 Study Design

While the two exams were different in many respects, both exams used an identical set of three SOLO-related questions, referred to as *loop* (Figure 1), *swap* (Figure 2), and *average* (Figure 3). The "loop" code is the same code used by Lister et al., but in this study we replaced their instruction "*In plain English, explain what the following segment of Java code does*" with "*Explain the purpose of the following segment of code*".

Assume that a, b, c are declared as integers and have been initialized. Explain the purpose of the following segment of code.

```
a = b;
b = c;
c = a;
```

Figure 2: The *swap* question

Suggest a name for method10 below that reflects its purpose.

```
public float method10(int[] aiNumbers)
{
   int iSum = 0;
   for (int iLoop = 0; iLoop <
               aiNumbers.length; iLoop++)
   {
        iSum += aiNumbers[iLoop];
   }
   return iSum / aiNumbers.length;
}
```

Figure 3: The *average* question

2.2 Study Participants and Exam scripts

Exam scripts for 120 introductory programming students were analyzed. There were 79 scripts for the undergraduate students and 41 for the postgraduate students. The ratio of male/female students was 64/15 for the undergraduate cohort and there was a similar ratio (35/6) for the graduate cohort

The undergraduate exam contained five sections, worth a total of 100 marks. Section 1 (20 marks) comprised 20 short answer questions. The three SOLO-related questions analyzed in this study were placed in this section of the exam paper. Section 2 (21 marks) required students to write code for three small programming problems. Section 3 (33 marks) required students to design and implement a class that required the use of inheritance. Section 4 (18 marks) tested the students knowledge on algorithms and control structures. Section 5 (8 marks) focused on debugging.

The postgraduate exam paper was also worth a total of 100 marks but contained only four sections (A-D). Section A (15 marks) required students to give short written responses. Section B (20 marks) required students to write code to solve small programming problems and interpret segments of code. The three questions analyzed in this study were placed in this section of the exam paper. Section C (15 marks) required students to design a solution to a problem and to develop a testing strategy for the program. Section D (50 marks) was devoted to coding a more complex programming problem and also aspects of debugging.

SOLO was not used as part of the marking scheme for these two exams. The SOLO classification to the three questions was done subsequent to, and independently from, the exam marking.

2.3 Data Analysis

Student responses to the three questions were analyzed according to the SOLO taxonomy by the six authors. Initially, the six researchers independently coded the three answers of all 41 postgraduate students. This was followed by a discussion over differing classifications. To clarify coding and discussion, four new SOLO sub-categories were used:

- **Relational with extra ("Ra"):** For example, such a response for *swap* might mention that the values in "b" and "c" are exchanged, but adds that "a" contains the original value of "b".
- **Relational but error ("Re"):** For example, such a response for *swap* might mention that the code swapped the values in two variables, but specifies the wrong variables.
- **Relational incomplete ("Ri"):** A response where the student has not directly answered the question. For example, for *average*, some students wrote that the method computed the average of the numbers in the array, but omitted to provide a name for the method.
- **Multistructural with error ("Me"):** The student has made an error in their description of a line of code.

Students often provided both a relational response and a multistructural response. Such a response was deemed to be relational.

Once agreement was reached on the postgraduate exam responses, three of the authors coded the 79 undergraduate responses.

3. RESULTS

This section presents an analysis of the data collected for this study. To analyze the ordinal SOLO responses, we allocated a number to each SOLO level, from 1 (for prestructural) to 4 (for relational), and 0 for a blank response. Due to the sparseness of results for Ra, Re, Ri and Me responses, these subcategories were included in the parent "R" and "M" categories for statistical analysis.

3.1 Overview of SOLO responses

A summary of the SOLO responses for each question using the extended classification categories is shown in Table 1. A graphical summary of the responses classified according to the original SOLO responses is presented in Figure 4. This shows that the patterns of responses for the *swap* and *loop* questions are similar and these are different to the pattern for *average*.

3.2 Were there any differences between student cohorts and gender?

To investigate any differences in the level of responses between the undergraduate and postgraduate students, Mann Whitney U tests were used. The analysis showed that for each question the postgraduate group scored higher level responses than the undergraduate group. The results are shown in Table 2. Similar tests based on gender showed no differences in responses between the male and female students. No further analysis was conducted based on gender.

Table 1: Comparison of percentages of SOLO responses for undergraduate and postgraduate groups

SOLO response	Swap		Loop		Average	
	UG %	PG %	UG %	PG %	UG %	PG %
R	27	71	15	37	54	71
Ra	0	2	0	0	0	0
Re	3	5	6	5	1	5
Ri	0	2	3	5	0	10
M	25	10	20	32	1	0
Me	1	0	3	0	0	0
U	0	5	5	15	8	7
P	43	5	41	7	25	7
B	1	0	8	0	4	0

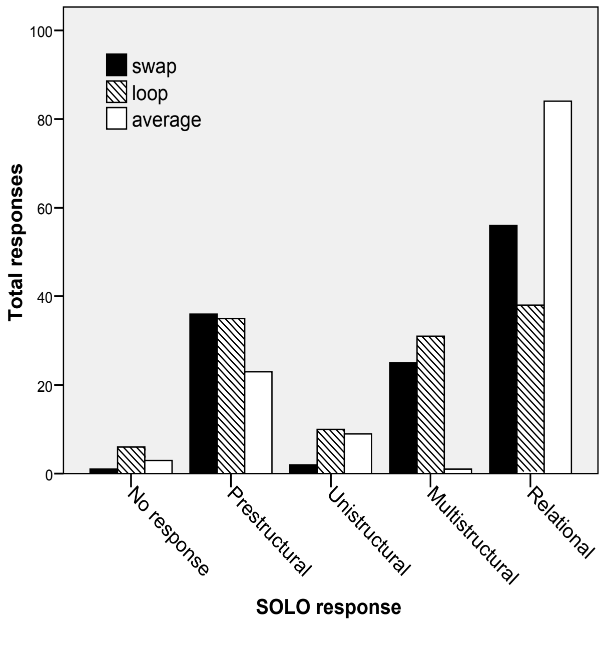

Figure 4: Total undergraduate and postgraduate SOLO responses for the *swap*, *loop* and *average* questions

Table 2: Comparison of mean and median of SOLO responses for undergraduate and postgraduate students

	Undergraduate		Postgraduate		U
	Mean	Median	Mean	Median	
Swap	2.39	3 (M)	3.66	4 (R)	723*
Loop	2.15	2 (U)	3.10	3 (M)	989*
Average	2.92	4 (R)	3.63	4 (R)	1214*
* $p \leq 0.05$ according to a Mann Whitney U test					

3.3 Were there any differences in SOLO responses between questions?

Relationships between the SOLO responses to the *swap*–and–*loop, swap*–and–*average,* and *loop*–and–*average* questions were tested using Spearman's R correlations. These showed moderate relationships between the students' responses for each pair of questions for both the undergraduate (R=0.54, 0.54, 0.48) and the postgraduate groups (R=0.54, 0.51, 0.48). In each case these relationships were significant at $p < 0.05$.

The level of responses to the *swap* and *loop* questions were compared using Wilcoxon Signed Ranks test. No significant differences were found for the undergraduate students (i.e. undergraduates tended to provide the same type of SOLO response for *swap* and *loop*). The postgraduates did show a statistically significant difference in SOLO response for the *swap* and *loop* questions ($Z = 3.22$, $p < 0.05$). Closer examination of the postgraduate data showed that each student had a SOLO response to the *swap* question either at the same SOLO level, or at a higher level, than their response to the *loop* question – there were no lower responses. The greater number of higher level responses to the *swap* question was consistent with the researchers' view that this was an easier question to answer than the *loop* question in that it tested understanding of *assignment*, rather than the more difficult concepts of *selection* and *iteration* in the loop question.

3.4 Were there any differences in exam results for different SOLO levels?

The mean exam results for each SOLO level for the undergraduate and postgraduate classes are shown in Table 3. These indicate that the average exam mark decreases as the level of SOLO response decreases, from relational to prestructural. ANOVA tests showed that these differences were significant for both the undergraduate and postgraduate groups.

Table 3: Comparison of mean exam marks for each SOLO level for undergraduate and postgraduate students

	Mean exam result (%)					
	swap		loop		average	
	UG	PG	UG	PG	UG	PG
Relational	60.1	75.4	55.0	80.2	52.3	76.7
Multistructural	44.9	58.4	55.6	71.0	23.0	-
Unistructural	-	54.8	54.6	64.5	26.7	29.7
Prestructural	29.2	25.0	29.6	31.8	28.3	36.7

3.5 Was there a relationship between exam results and overall SOLO responses?

An overall SOLO response was calculated for each student by summing the SOLO responses for *swap*, *loop* and *average*, giving a score in the range from 0 to 12. Spearman's R correlations conducted on overall SOLO responses and exam mark were significant for the undergraduate (R = 0.70) and the postgraduate (R=0.58) students.

Scatterplots of the exam mark and SOLO responses for both undergraduate and postgraduate students (Figures 5 & 6) showed interesting patterns. For both groups of students, a low level SOLO response (i.e. students responded with a U, P or B over the three questions) corresponded to a fail grade in the exam (i.e. lower than 50%), indicated by the empty top left quadrant in both scatterplots. All postgraduate students giving high level SOLO responses (i.e. ≥ 10) scored a passing grade (i.e. above 50%) for the exam. However, this was not the case for the undergraduate student group. This is indicated by the bottom right quadrant of the scatterplots

As described earlier, both exam papers contained a mixture of types of questions. Section D of the postgraduate exam was a large code writing task. Figure 7 is a scatter plot of Section D marks and overall SOLO responses. A further correlation of overall SOLO responses and the code writing section of the postgraduate exam (Section D) was also significant (R=.569).

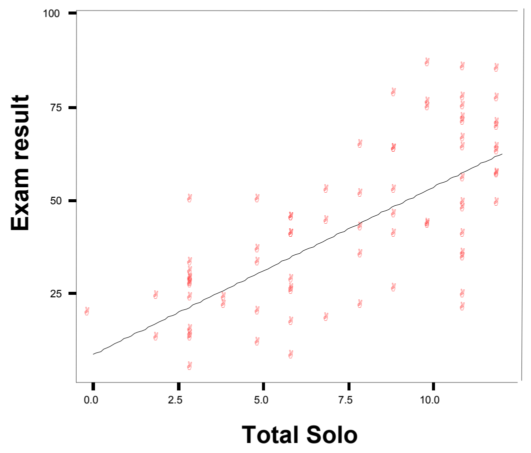

Figure 5 Scatterplot of UG exam results and SOLO responses

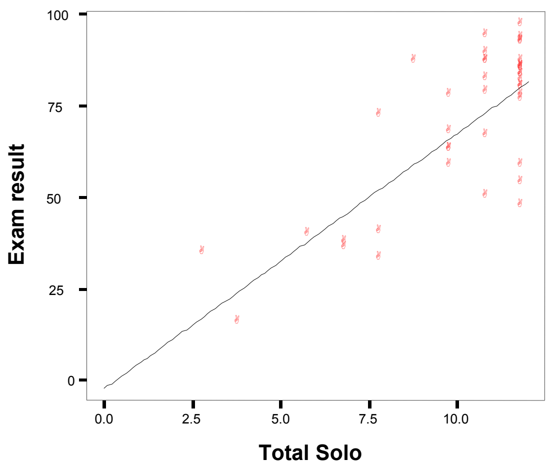

Figure 6 Scatterplot of PG exam results and SOLO responses

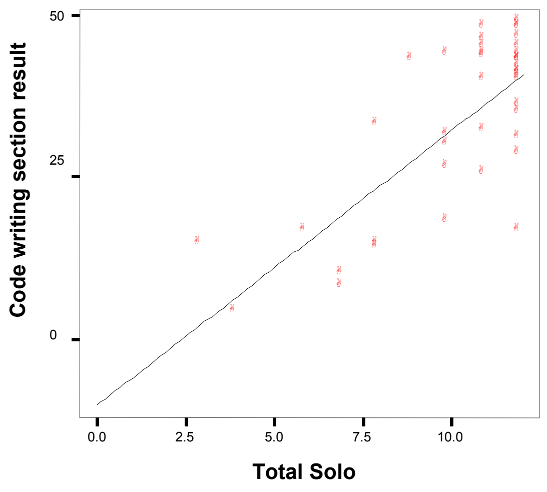

Figure 7 Scatterplot of postgraduate results for the code writing section (section D) of the exam and SOLO responses

4. DISCUSSION

We discuss the four research questions raised in section 1.1:

1. Is the instruction – 'In plain English, explain what the following segment of Java code does' – ambiguous? Similarly, is our instruction – Explain the purpose of the following segment of code – also ambiguous?

   Our *swap* and *loop* used the same instruction. For *swap*, 71% of postgraduates answered "R" but only 37% (i.e. 34% less) answered "R" for *loop*. This suggests that – at least for 34% of postgraduate students – the instruction is not ambiguous. However, we cannot eliminate the possibility that some of postgraduates misunderstood the instruction.

   The results for *average* are harder to assess. The high rate of "R" responses for *average* might suggest that the instruction given for *swap* and *loop* was ambiguous. However, the use of meaningful variable names in *average* (especially "sum") may have given a student a clue as to what the code was doing, without the student actually understanding the code. Asking students to nominate a method name is a promising approach if our aim is to gain evidence of relational thinking. However, this approach does not give comprehensive information about students' lower level thinking.

2. If asked more than one SOLO question, does a student tend to provide answers at the same SOLO level?

   Students were relatively consistent in the SOLO level of their answers across *swap* and *loop*. The overall differences in level of the responses are probably due to the code for *swap* being easier to interpret than the code for *loop*.

   As above, the results for *average* are harder to assess, because the use of meaningful variable names may have given a student a clue as to what the code was doing, without the student actually understanding the code.

3. Consistent with the conjecture made by Lister et al., is there a correlation between student performance on SOLO reading tasks and on code writing tasks?

   Figure 7 and its correlation coefficient (R value) indicate that, in this study, there was a positive correlation.

4. How do SOLO responses between postgraduate and undergraduate students compare in an introductory programming unit?

   The higher level of SOLO responses from the postgraduate group is consistent with our understanding of postgraduate students as having developed higher level thinking skills during their undergraduate degree.

5. CONCLUSION AND FURTHER WORK

The findings from this study support our plans to use the SOLO taxonomy to develop an instrument to assess a student's ability to see higher level relationships in their code and to develop teaching techniques to help students to acquire this ability.

6. REFERENCES

[1] Adelson, B. When novices surpass experts: The difficulty of a task may increase with expertise. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 10, 3 (1984), 483-495.

[2] Biggs, J. B. & Collis, K. F. *Evaluating the quality of learning: The SOLO taxonomy* (*Structure of the Observed Learning Outcome*). New York, Academic Press, 1982.

[3] Lister, R., Simon, B., Thompson, E., Whalley, J. L., and Prasad, C. (2006). *Not seeing the forest for the trees: novice programmers and the SOLO taxonomy.* In Proceedings of the 11th Annual SIGCSE Conference on innovation and Technology in Computer Science Education. (Bologna, Italy, June 26 - 28, 2006), 118-122.

[4] McKeithen, K., Reitman, J.., Rueter, H., & Hirtle, S. (1981). Knowledge organization and skill differences in computer programmers. *Canadian J. of Psychology*, 13, 307-325.

[5] Wiedenbeck, S., Fix, V. & Scholtz, J. (1993) Characteristics of the mental representations of novice and expert programmers: An empirical study. *International Journal of Man-Machine Studies*, 39, 793-812.