

LANGUAGE CHOICE AND KEY CONCEPTS IN CS1

Chris Stephenson
Computer Systems Research Institute
University of Toronto
chris@hsa.on.ca

Tom West
Computer Systems Research Institute
University of Toronto
west@hsa.on.ca

ABSTRACT

Because computer science is a complex and rapidly evolving discipline, educators at all levels have been required to revise their curricula as new paradigms and tools emerge. Since 1968 there have been three major computer science curriculum documents for the first course at the college/university level (CS1), and recent research shows no diminishment of this rate of evolution. This paper reports on a survey of 900 post-secondary institutions in North America which offer a CS1 course. The results demonstrate that despite differing course descriptions, there is a key set of computer science concepts common to most CS1 courses. They also indicate that over the last two years there has been a significant shift in programming language use for CS1, centering on the replacement of Pascal with C and C++. This ascendancy may, however, be short-lived as many of those presently using C++ indicate that they considering a further switch to Java over the next two years.

INTRODUCTION

For some years researchers and educators have noted the importance of ensuring that university computer science courses reflect the rapid advances taking place in the discipline (Foley, 1988; Hartmanis and Lin, 1992). The exponential rate of change in computer hardware, software and methodologies have therefore, if not driven, certainly strongly contributed to a constantly shifting set of curricula, published at regular intervals and all designed to reflect the changing demands of a young and dynamic field.

Since 1968 three major curriculum recommendations for a first course in computer science at the college/university level (CS1) have been published: Curriculum 68 which defined the scope of the discipline and maintained a strong link with mathematics, Curriculum 78 which emphasized programming and problem solving, and Curriculum 91 which focused on the laboratory aspects of computer science (Reinfelds, 1995). According to Gersting and Young (1997) the result of these revisions has been "a carefully designed coverage of academic material in a fashion that requires students to master core material and

guarantees student exposure to appropriate additional material that insures both breadth and depth of knowledge" (325).

This is not to say, however, that the CS1 curriculum has remained static since 1991. In fact, it is clear that it continues to change and evolve in terms of both content (Pattis, 1993; Tucker, 1994; Astrachan, 1995) and method of delivery (Connelly and Biermann, 1996; Feldman and Zelenski, 1996; Walker, 1997), and in this way will likely continue to improve student understanding (Joosten et al, 1993; Lambert, Lindsay, Robertson, 1993).

The challenge of developing viable curricula for computer science courses arises to some extent from the genesis of the discipline itself. As Merritt (1995) notes, it is "a volatile mix of engineering, mathematics, psychology, cybernetics, philosophy and organizational theory from which emerged the discipline of computer science or informatics" (p. 482). In light of continuing advancements and changes in the tools of computer scientists (hardware, operating systems, programming languages, paradigms), and evolving approaches to pedagogy, questions arise as to whether there is some solid and unchanging core of knowledge which remains central to computer science and hence an essential part of a first university level course, and whether a particular programming language may be particularly effective for imparting this knowledge to students.

This article describes a survey of current CS1 courses aimed at providing some indication of current and future trends in programming language use.

METHOD

The subjects for this survey were selected from a commercial database of four-year post-secondary institutions offering a first course in computer science across North America. Packages addressed to the Department Chair were sent to 900 institutions. Each package contained a letter explaining the purpose of the survey and the one-page survey itself. The letter requested that the Chair forward the correspondence and survey to the person on staff deemed most likely to have access to the

information requested. A FAX number and postal address were provided for return of the completed survey forms.

The survey consisted of a series of eight questions. The first three questions were intended to determine current, past, and future programming language use (within a two-year span in either direction) and the fourth to ascertain the criteria for language choice. Question five related to course description, asking respondents to indicate whether they would describe their course as depth-first, breadth-first, or other. In question six, respondents were provided with a list of nineteen computer science concepts determined by the researchers to be common elements of many CS1 courses, and asked to indicate which concepts they covered in their CS1 course. Additional spaces were provided for concepts not contained in the survey list. In question seven the subjects were asked to indicate whether there were concepts not covered because of language choice and if so, which concepts. The final question was similar except that it concerned concepts not covered due to time constraints.

Survey responses were collected and stored in a computer spreadsheet which provided maximum flexibility for adding new data fields as additional concepts were indicated. Responses were collected and subject to a quantitative analysis based upon averages calculated with respect to the language of instruction.

RESULTS

Responses were obtained from 309 post secondary institutions, giving a response rate of 34%.

Current Language Use:

As expected C++, C, and Pascal were reported to be the most widely used languages in CS1 courses with a number of other languages showing significantly less use. The following chart details the reported use for languages which exceeded 1% of reporting colleges/universities.

Current Language Use		
C++	154	50.1%
Pascal	62	20.5%
C	50	16.8%
ADA	15	4.9%
Java	15	4.9%
Scheme	15	4.9%
Visual Basic	7	2.2%
Basic	4	1.3%
Cobol	4	1.3%

Those languages for which reported use was less than 1% include: Fortran (0.9%), Modula-2 (0.9%), Turing (0.9%), Delphi (0.6%) OpenScript (0.6%), Eiffel (0.3%), Haskell (0.3%), Lisp (0.3%), Logo (0.3%), and Prolog (0.3%).

Previous Language Use:

Of the sites responding to the survey, 174 or 56.3% indicated that they had switched programming languages within the last two years, with 135 or 43.7% indicating that their language choice had remained unchanged during that time period. As the following chart indicates, of those who changed languages during this time period, the majority were moving away from Pascal.

Previous Language		
Pascal	129	41.4%
C	23	7.5%
C++	5	1.6%
Modula-2	5	1.6%
ADA	4	1.3%
Fortran	3	1.0%
Cobol	2	0.6%
Basic	2	0.6%

Those languages for which previous use was less than 1% include: Haskell (0.7%), Lisp (0.7%), and OPascal (0.7%).

Future Language Use:

Respondents were also asked whether or not they intended to change programming languages within the next two years. Here, 176 or 56.9% of the institutions indicated that they did not intend to switch languages in their CS1 course, while 133 or 43.1% indicated that they were considering adopting a new CS1 programming language. It should be noted that many of the institutions indicating an intention to change languages listed more than one possible choice, so the resulting figures exceed 100%. Again, however, the results indicate that certain languages, particularly Java and C++, appear to dominate the list of potential choices, with the remaining languages accounting for only a small number of sites.

Languages Considered by Changers		
Java	95	72.2%
C++	51	38.3%
C	9	6.7%
Smalltalk	3	2.2%
Eiffel	2	1.5%
Ada	2	1.5%
Scheme	2	1.5%
Visual Basic	2	1.5%
Haskell	1	0.7%
Lisp	1	0.7%
Pascal	1	0.7%
OPascal	1	0.7%

The data was also analyzed to determine whether or not patterns of change could be identified with respect to specific languages. The following chart combines current use with the percentage of those who have changed to that

language within the last two years, and those who are now considering an alternative language within the next two years. By "quickly changing" we mean those who have changed within the last two years and are considering changing again within the next two years.

In the case of Pascal, for example, the table shows that 11% of the responding colleges adopted it within the last two years. This is illustrative of the fact that as an instructional language, few institutions are switching to Pascal relative to the number of sites using Pascal.. More dramatically, 67% the current Pascal users are considering adopting a different language in the next two years. Some 29% of the schools that switched in the last two years are already considering switching to a different language. This is a strong indication of Pascal's abrupt decline as a language of instruction in colleges and universities.

Change Patterns in Language Choice			
Language	Changing	Recently Changed	Quickly Changing
Ada	40%	40%	17%
C	38%	50%	48%
C++	35%	77%	30%
Java	0%	93%	0%
Pascal	67%	11%	29%
Scheme	33%	33%	60%

The results for C++, however, indicate a much more rapid turnover. Of the sites indicating that they use C++ in CS1, 77% have adopted it within the last two years. At the same time, 35% of the C++ sites, including 30% of the sites who just began using it, are considering changing languages. What this appears to indicate is that despite C++'s recent appearance on the scene, its lifespan as an educational language may be much shorter than languages that preceded it.

Language Selection Criteria:

When asked to indicate the criteria for language selection, a slight majority (165 or 53.4%) responded that teachability was the most important factor, followed closely by industrial relevance (155 or 50.1%). It should be noted, however, that many respondents reported that both of these criteria are important in their choice of teaching language. A total of 35 (11.3%) post-secondary institutions indicated that there were other factors involved in the selection of a programming language for CS1. These included external influences such as the Advanced Placement (AP) exam or state mandate, the extent to which the language facilitated the introduction of key concepts and conformed to ideals of a "good" language from a computer science perspective, compatibility with overall curriculum goals and other courses, faculty consensus, the availability of inexpensive software and good textbooks, power, simplicity, elegance and transparency, usefulness for various levels of applications, and support for modern features. Only one

Key CS Concepts Available in Each Language						
	C	C++	Java	Pascal	Scheme	Ada
data types	96%	99%	100%	100%	93%	100%
variables	100%	100%	100%	100%	100%	100%
text I/O	94%	96%	73%	98%	73%	100%
control constructs	100%	99%	93%	94%	93%	100%
subprograms	94%	99%	87%	97%	93%	100%
arrays	100%	96%	100%	95%	73%	100%
records	70%	64%	40%	68%	47%	93%
sorting	70%	64%	60%	68%	73%	93%
pointers	66%	42%	27%	31%	73%	60%
correctness	54%	44%	47%	58%	73%	60%
efficiency	50%	43%	27%	45%	93%	53%
trees	12%	12%	20%	11%	67%	33%
ethics	24%	31%	20%	24%	27%	33%
graphics	12%	18%	53%	18%	47%	13%
linked lists	34%	27%	27%	22%	87%	20%
internet	20%	18%	40%	8%	20%	0%
objects	22%	66%	87%	17%	60%	13%
classes	26%	68%	87%	6%	67%	7%
inheritance	16%	21%	47%	8%	40%	0%

Figure 1

respondent admitted that the language choice was "an administrative fiat". A total of 22 colleges or universities (7.2%) indicated that student preference was an important factor in language choice for CS1. These results appear to indicate that language selection is a complex process requiring the weighing of a multiple of criteria

CS1 Course Descriptions

When asked to choose the term that best described their CS1 course content, 138 or 44.6% chose "depth-first", 124 or 40.1% chose "breadth-first" and the remaining 40 or 12.9% chose "other". Those indicating "other", commonly noted that their course was either a mixture of "depth-first" and "breadth-first" or simply an "introductory programming" course.

CS Concepts and Programming Languages

One of the important questions when considering programming language choice for CS1 is the extent to which a language is amenable to teaching key computer science concepts. **Figure 1** indicates the percentage of users of each language who indicated that their course included the listed concepts.

One way of simplifying the above data is to rate each language in terms of how many respondents report using it to teach the key concepts (regardless of whether the language could be used). While the differences between the languages are not statistically significant, they are intriguing none the less. The ranking in terms of greatest to least is as follows:

Scheme

Java

C++

Ada

C

Pascal

It is also interesting to note that while space was allotted for the inclusion of additional concepts, no single concept received a sufficient number of responses to warrant inclusion in the list of key concepts. Still, readers should be aware that the concept list provided as part of the survey represents certain researcher biases as to which concepts are fundamental within the CS1 curriculum.

CONCLUSIONS:

This data appears to indicate that although course descriptions may vary, CS1 course content is fairly consistent in terms of the presence of key concepts. Programming language choice, however, appears to be quite fluid as institutions adapt to changing technology, paradigms, and software availability within a complex context of internal and external criteria. Specifically, the data indicates that C++ dominates current use despite Pascal's continuing presence at many institutions, and that Pascal will continue to decline as newer languages such as C++ and Java gain in popularity. This ascendancy of C++

and Java appears to be driven primarily by perceived of industrial relevance and student preference.

The domination of C and C++ may be short-lived, however, if Java continues to gain converts. Its long-term viability is likely to depend, however, on the extent to which institutions now adopting Java are satisfied with it as a medium for teaching key computer science concepts and for meeting the strong demand for industrial relevancy.

Because computer science is both complex and changing it would be foolish to contend that this data represents more than a flash photo of selected aspects of the current state of CS1. The researchers hope, however, that by providing this representation as a starting point, future research undertakings will provide greater enlightenment as to changing patterns of course content and programming language use over time, and that this information will further assist instructors in making the best possible choices for their institutions and for their students.

REFERENCES:

- ACM Curriculum Committee. (1968). Curriculum 68. *Communications of the ACM*, 11(3), 151-197.
- ACM/IEEE-CS Curriculum Task Force. (1991). *Computing Curriculum 1991*. IEEE Computer Society Press.
- Astrachan, Owen. (1995). AAA and CS 1: The Applied Apprenticeship Approach to CS 1, *SIGCSE Bulletin*, 21(1), March, 1-5.
- Austing, R. (ed). (1978). Curriculum 78. *Communications of the ACM*, 22(3), 147-166.
- Connelly, Christopher and Biermann, Alan W. (1996). Home-Study Software: Flexible, Interactive, and Distributed Software for Independent Study. *Proceedings of the 27th SIGCSE Technical Symposium on Computer Science Education*, Philadelphia, 63-67.
- Feldman, Todd J. and Zelenski, Julie D. (1996). The Quest for Excellence in Designing Cs1/Cs2 Assignments. *Proceedings of the 27th SIGCSE Technical Symposium on Computer Science Education* (pp. 63-67). Washington, DC: Association for Computing Machinery.
- Foley, J. (Ed.). (1988). *Report on the National Science Foundation Disciplinary Workshops on Undergraduate Education*. Washington, DC: National Science Foundation.
- Gersting, Judith L. and Young, Frank H. (1997). "Content + Experiences = Curriculum". *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education* (pp. 319-323). Washington, DC: Association for Computing Machinery.
- Hartmanis, J and Lin, H. (1992). *Computing the Future: A Broader Agenda for Computer Science and Engineering*. National Academy Press.
- Joosten, S. van den Berg, K and van der Hoeven, G. (1993). "Teaching Functional Programming to First Year Students". *Journal of Functional Programming*, 3(1), 49-65.
- Lambert, T., Lindsay, P. and Robinson, K. (1993). "Using Miranda as a First Programming Language". *Journal of Functional Programming*, 3 (1), 5-34.

- Merritt, Susan M. (1995). Reflections of a Computer Scientist for Teachers and Teacher Educators. In Tingley, J. David and van Weert, Tom J. (eds.) *Proceedings of the Sixth IFIP World Conference on Computers in Education* (pp. 479-486). London: Chapman and Hall.
- Pattis, Richard E. (1993). The 'Procedures Early' Approach in CS 1: A Heresy. *SIGCSE Bulletin*, 25(1), March, 122-126.
- Reinfelds, Juris. (1995). Logic in First Courses for Computer Science Majors, In Tingley, J. David and van Weert, Tom J. (Eds.) *Proceedings of the Sixth IFIP World Conference on Computers in Education* (pp. 467-478). London: Chapman and Hall.
- Tucker, Allen B. (1994). "New Directions in the Introductory Computer Science Curriculum". *SIGCSE Bulletin*, 26(1), March, 11-15.
- Walker, Henry. (1997). "Collaborative Learning: A Case Study for CS1 at Grinnell College and UT Austin". *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education* (pp. 209-213). Washington, DC: Association for Computing Machinery.