

A REPORT ON HIGH SCHOOL COMPUTER SCIENCE EDUCATION IN FIVE U.S. STATES

Chris Stephenson

ABSTRACT

For many years there has been considerable debate over the place of computer science education in the high school curriculum and despite increasing industrial demands for high skilled technology workers, figures released by the College Board indicate that fewer students are actually taking programming courses in high school. This article reports on a survey of over 2500 schools in five U.S. states. The results include information on teaching responsibilities, hardware use and purchase criteria, programming language use and purchase criteria, and teaching resource and skills upgrading preferences. This survey was supported by funding from IBM.

PURPOSE

This survey was instituted as a means of gathering data on the current state of hardware and software use in high school computer science programs. The goal was to provide detailed and reliable information relating specifically to computer science education (as opposed to information concerning the implementation of computers across the curriculum).

INTRODUCTION

While numerous research efforts have provided a body of information about teaching computer science at the college/university level, far less research has been conducted on computer science in high schools. The lack of information in combination with no commonly-applied curriculum have resulted in many schools making important decisions about course content and instructional tools (hardware and software) in isolation. This general lack of information further complicates the efforts of colleges and universities to plan course offerings to meet the needs of incoming students.

This survey was intended to fill in some of the information gaps about high school computer science. In particular, we were interested in the following:

- teaching responsibilities,
- hardware is being used and the selection criteria,
- grades at which computer science is offered,
- programming languages being used and the selection criteria,
- how teachers rank various instructional resources, and
- how teachers rank opportunities for their own skills upgrading.

It was our hope that by collecting and disseminating this data from a number of states, we could create a current snapshot that might help educators at all levels better understand the current state of high school computer science.

The previous lack of research relating to high school computer science may be explained in part by the fact that computer science is not generally considered part of the core curriculum of high school education (Merritt, 1995). Many scientists and educators argue, however, that knowledge of the fundamental concepts of computer science is essential for understanding the technology which increasingly facilitates the business, governmental, and personal interactions our society. As Merritt (1995) notes:

“The discipline of computer science has permeated our culture in much the same way as, for example, nuclear physics (atoms and molecules) has or as genetics (DNA, chromosomes and genes) has. Atoms, molecules, genes and chromosomes are now part of the K-12 curriculum. Similarly, I maintain that computer science (not just computer technology) needs to be part of K-12 education.”

Support for this argument is provided by Proulx (1995) who identifies the core computer science concepts that should be part of the high school curriculum. These concepts include:

- algorithms,
- programming languages,
- computer architecture,
- operating systems and user support,
- social, ethical and professional context, and
- applications.

Historically, resistance to high school computer science among college/university educators is based on the belief that it was not being taught “properly” and so would be better left until college. Taylor and Mountfield (1991) addressed this opinion directly and concluded that the research did not support the claim by university professors that high school computer science was detrimental to student performance at the college/university level. The researchers concluded, however, that the methodology that was taught contributed more to success in college than the course content.

There are, however, lingering questions about high school computer science relating to a number of issues. These include teacher preparation, hardware use, choice of instructional programming language, and teacher upgrading. We will now briefly examine these issues.

Computers in Schools

Between 1984 and 1994 American schools expended nearly \$500 million to add between 300,000 and 400,000 computers to their inventory. The total number of computers in high schools increased by 57% so that by 1994 the typical high school had 54 (median) computers (Becker, 1994). According to Coley, Cradlet, and Engel (1997) by 1997, 98% of all U.S. schools had computers and the average student-to-computer ratio was 10:1. At the state level, this ratio differed greatly, from approximately 6:1 in Florida, Wyoming, Alaska, and North Dakota to 16:1 in Louisiana. The reported student-to-computer ratio in the states which were the subject of this study were as follows:

California	2:1
New Jersey	12:1
New Hampshire	6:1
Massachusetts	14:1
Washington	2:1

Within each state, however, there are also differences from school to school. Students attending poor and minority high schools are reported to have less access to all types of technology than students attending other schools (Coley, Cradlet, and Engel, 1997).

By 1996, 64% of U.S. high schools also had access to the Internet and in some states all school were connected (Coley, Cradlet, and Engel; 1997). Only 14% of classrooms, however, had direct access. By 1997, the reported Internet access for our target states was as follows:

California	15%
New Jersey	72%
New Hampshire	75%
Massachusetts	60%
Washington	55%

Teacher Preparation

Many computer science teachers currently working in high schools were educators whose initial preparation was in another field, most-commonly math and science (Schollmeyer, 1996), before they retrained in computer science. Herrmann (1991), for example, reports that in the State of Virginia 71% of teachers who taught computer science/data processing reported some training in computer education while only 22% indicated computer science as their primary field of expertise. Schollmeyer (1996) also reported that very few high school computer science teachers self-identify as very knowledgeable in programming, have industrial programming experience, or a solid foundation in data structures and more advanced topics. These figures are somewhat surprising in light of Kushan's (1994) findings that both district coordinators and computer science teachers themselves consistently rank "knowledge of the subject" and "keeping up to date" in the top five criteria for effective computer science teachers.

This concern about teacher preparedness extends beyond original experience and qualifications to the ability to learn and implement with new technologies and teaching strategies as they emerge. As Becker (1994) notes, investment in the "people" costs of using technology, which he identifies as:

- formal teacher training,
- coordination and management of technology resources,
- technical support, and
- the time required for informal and latent growth of professional competence and integration with other curricular goals

has not had the same priority as spending on hardware and software.

Stager (1995) also notes that no single method of professional development can provide teachers with the array of experiences they require to continue to build their technical and pedagogical skills. Rather, school districts must

provide a combination of traditional workshops, in-classroom collaborations, mentoring, conference participation and whole learning residential workshops to enable teachers to learn at their own pace. Stager also concludes that computer-related professional development for teachers must extend beyond the purely technical, focus on the change process, and immerse teachers in meaningful educationally relevant projects that encourage teachers to reflect on and share ideas.

Programming and Instructional Language

Figures released by the College Board in 1996 indicated that while 24% of college-bound students had taken at least one computer programming course in high school there was a significant gender-based difference, with 29% of male students and 20% of female students. The College Board also reported an overall decline in the number of college-bound students taking computer programming courses from previous years. Percentages determined from previous years were as follows:

1987	45%
1990	40%
1993	30%

These results are supported by Coley, Cradlet, and Engel (1997) who report that in 1997, 29% of U.S. students of seventeen years of age had taken at least one programming course.

As is the case at the college/university level, choice of an instructional programming language for high schools is an issue fraught with debate and subject to a number of external considerations. In order to address this issue from a pedagogical rather than purely technical standpoint, Milbrandt (1993) proposed that the following criteria should inform the selection of a structured programming language for computer science instruction:

- ease-of-use,
- structured in design,
- powerful in computing capacity,
- simplicity of syntax,
- inclusion of variable declaration,
- easy input/output and output formatting,
- use of meaningful keyword names,
- allowances for expressive variable names, and
- the presence of a one-entry/one-exit structure.

He also notes the importance of immediate feedback to the learner-programmer and of good diagnostic tools for testing and debugging.

Many researchers have also attempted to evaluate the benefits and drawbacks of specific programming languages. Schollmeyer (1996), for example, notes while the use of Basic does not preclude the writing of good programs, its structure tends to encourage novice programmers to develop habits such as writing first and thinking later which must then be unlearned at the college/university level. Schollmeyer (1996) further argues that Pascal has a number of features which encourage the development of desirable programming practices such as modularity and top-down design, which are

required at the college/university level. These features include: the use of functions and procedure calls, variable declaration, and clean control structures.

Wittenburg (in Burd et al, 1997) contends that while visual programming tools such as Visual Basic are useful in the hands of expert users for rapid application development and prototyping, “they tend to drag the focus of the course away from solving problems using a programming language toward interface design”. This latter concern is echoed by Martin (1999) who concludes that students of poorer ability tend to be confused by Visual Basic’s profusion of features. The “potentially more nerdy” students, on the other hand, get carried away with these features to the detriment of their programming ability and ultimately these students’ programs demonstrate a surface knowledge of Visual Basic rather than an understanding of programming concepts. Martin’s concerns about Visual Basic in an educational environment, include:

- student misconceptions that objects such as command buttons were procedures,
- allowing students to be lazy in defining variable data types unless a type is declared, and
- students writing unfriendly programs.

Other researchers, however, argue that the design of an effective graphical user interface (GUI) requires a wider range of skills than those of software implementation and that these skills, particularly those applying to human-computer interfacing (HCI), are essential for students (Culwin, 1999). Workman (in Burd et al, 1997) argues that visual programming places a natural emphasis on designing a clear and attractive user interface and makes interesting application such as multimedia and the Internet assessable to beginning students.

Object-oriented programming languages are becoming increasingly important in state-of-the art software development and so have achieved popularity with practitioners and educators alike. As a result, an increasing number of educators are using an object-oriented programming language somewhere in the curriculum (Mazaitis, 1993; Luker, 1994; Stephenson and West, 1998). The inclusion of object-oriented languages, however, has given rise to a number of questions, including:

- when to teach the first object-oriented language,
- what set of concepts to include or exclude from the course, and
- which programming language to use (Kolling and Rosenberg, 1996).

C++ has become an industry standard despite the fact that the language itself is not standardized (Weiss, 1997). Moylan (1992) argues, however that C++ provides poor support for modularity, minimal error checking, and continues the tradition that “everything should be legal”. These characteristics allow programmers to create unreadable code and limit the compiler’s ability to detect errors. He further argues that the extended features of C++ are extremely complex and cause programmers to misuse them.

As Weiss (1997) notes, Java attempts to simplify C++ by removing unnecessary features and cleaning up bad design decisions. As a result, many C++ features, such as templates, operator overloading, and pointers are not part of Java. Standing (1999) also notes the extendibility of Java as an instructional language, pointing out that as a multifaceted language, Java brings together a host of software technologies thus allowing a top-down approach in teaching and emphasis on computer science concepts.

A number of researchers contend, however, that despite its improvements upon C++, Java should in no way be considered a simple or easy language to learn. As a result many educators do not consider it suitable for novice programmers (Biddle and Tempero, 1999; Andreae et al. , 2000). The most-commonly noted Java complexity relates to its I/O model. While standard output in Java is relatively straightforward, standard input (for example from the keyboard) requires several declarations to set up the input stream and conversion methods to read numeric values. As Lewis (2000) notes, these steps are often considered unnecessarily complex for introductory students. Biddle and Tempero (1999) also identify a number of challenges that Java using educators must be aware of, including:

- poor enforcement of encapsulation,
- differences between primitive and object types,
- implicit pointers, and
- problems associated with genericity.

Other, more external factors affecting high school programming language choice include the influence of the Advance Placement (AP) exams, and the perception of industrial relevance. The AP in Computer Science is an intensive college-level curricula and examinations sponsored by the College Board and administered by Educational Testing Service. It provides motivated high school students an opportunity to earn advanced placement, college credit, or both in computer science (Baker, Chapman, Kmoch, Larson, Walker; 1998). The testing language mandated by the AP exams therefore directly impacts high school computer science instruction.

Brilliant and Wiseman (1996) also point out that the commercial failure of Pascal, combined with greater competition for jobs, has led to increasing pressure to teach a “marketable language”. Kavanagh (1998), refers to the influence of potential employers and Weiss (1997) notes, that Java in particular is “easily the most hyped language”.

METHODOLOGY

Instrument:

A one-page survey was designed to provide a composite picture of each school’s computer science program. The survey elicited information on the following:

- teaching responsibilities,
- current hardware use,
- plans to purchase more hardware,
- criteria for hardware selection,
- programming languages taught in each grade,

- plans to change programming language,
- criteria for programming language selection,
- current use of support materials, and
- methods of professional development.

Target

The survey was distributed in five states:

- California,
- Massachusetts,
- New Jersey,
- New Hampshire, and
- Washington.

These states were selected because they represent a diversity of school populations. New Jersey was chosen because it is known to have a strong computer science infrastructure with a number of magnet schools and close connections between the high schools and college/university institutions. Both California and Massachusetts have mixed school populations (both urban and rural) again with strong connections between the high school and college/university educational facilities. New Hampshire and Washington on the other hand, are smaller states with many rural schools and it was hoped that they would provide relevant data concerning the current state of computer science education in less populous states.

The contact information for the survey recipients was collected from mailing lists provided by a reputable data marketing company. The list parameters for schools included:

- public schools,
- county schools,
- state schools,
- private non-Catholic schools, and
- Catholic schools.

The list parameters for educators at each targeted school site were:

- computer science department chair, and
- computer science teacher.

The schools represented in the survey on a state-by state-basis were:

California	991
Massachusetts	539
New Hampshire	135
New Jersey	606
Washington	239

The total number of schools surveyed was 2510. In many cases the lists contained multiple teacher hits per school. Because the teacher population tends to be somewhat transient in many states, it was determined that all listed recipients would be contacted to ensure the greatest possible return rate. The total number of surveys mailed was 3069.

Distribution Method:

Personalized packages were sent to each targeted recipient at their school address. These packages contained:

- an introductory letter,
- the survey, and
- a postage paid return envelope.

The introductory letter specifically referenced the target state. Responding educators were informed that they would receive a copy of the results (the primary incentive for participation) via email as long as an email address was provided.

Tabulation:

Researchers collected the responses and tabulated the results using a database format specifically developed to provide numerical results for these survey questions. Results were tabulated on an overall and state-by-state basis.

RESULTS**Overall Response Rate:**

As Table 1 indicates, in all cases the state-by-state response rate exceeds the traditional survey return rate of 5% by a considerable margin. It should be noted, however, that the 8.71% response rate from Massachusetts suggests that any conclusions drawn for this state alone should be considered less reliable than for states with higher return percentages.

State	Schools Surveyed	Schools Responding	Percentage Responding
California	991	176	17.75
Massachusetts	539	47	8.71
New Hampshire	135	26	19.25
New Jersey	606	85	14.02
Washington	239	41	17.15
Total	2510	375	14.94

Table #1: Table of Response Rates

Teaching Responsibilities

In order to determine the extent and mix of their teaching responsibilities participants were asked to indicate whether they taught programming, computer applications, or other.

State	Programming	Applications	Both	Other
California	49.4%	75.6%	33.5%	58.7%
Massachusetts	67.1%	72.3%	38.3%	52.1%

New Hampshire	65.4%	69.2%	42.3%	46.2%
New Jersey	73.8%	50.6%	31.0%	49.4%
Washington	43.9%	43.9%	19.5%	43.9%
All States	57.7%	67.7%	32.6%	53.5%

Table #2: Teacher Responsibilities

The results, when compared on a state-by-state basis, indicate that a higher percentage of respondents considered themselves primarily computer applications teachers. California, for example, reports the second lowest percentage of programming teachers (49.4%) and the highest level of applications teachers (75.6%). The results for Washington state also show a greater than 20% difference in favor of applications teachers (43.9% versus 63.4%). The results for both Massachusetts and New Hampshire are more evenly distributed but still demonstrate a slight predominance of applications teachers. The results for New Jersey, however, differ significantly, with 73.8% of respondents reporting that they teach programming and only 50.6% reporting that they teach applications. When the responses for all of the states are combined, the results indicate an overall difference of 10.7%, with 57.7% of the respondents indicating that they teach programming and 67.7% indicating that they teach applications.

The results for Teaching Other (both individually and combined) indicate that approximately half of the teachers (53.3%) in all states teach subjects in addition to programming or applications, or both. Although these results vary slightly from state to state there are no marked differences.

The results for Teaching Neither are similarly closely distributed with a high of 12.2% reported in Washington state and a low of 4.3% reported in Massachusetts.

Current Hardware Use

All of the states in the survey report use of both IBM PC compatibles and Apple Macintosh computers. In all cases, however, schools reported a distinct difference in both the number of labs and machines within those labs. As Table # 3 indicates, IBM PC compatibles (PCs) greatly outnumber Apple Macintoshes (Macs). Massachusetts, for example, showed the greatest percentage of both hardware platforms with 78.7% of schools reporting that they had PC labs and 21.2% of schools reporting Mac labs. California showed the fewest overall number of labs with 67.6% of schools reporting that they had PC labs and 22.7% of schools reporting Mac labs. Washington state showed the fewest number of Mac labs at 14.6% of schools.

State	Av # PCs	Av # Macs	PC Lab Size	Mac Lab Size	% PC Labs	% Mac Labs
California	29.8	8.0	44.0	35.2	67.6%	22.7%
Massachusetts	25.1	2.7	31.9	12.9	78.7%	21.2%
New Hampshire	19.2	1.8	24.9	12.0	76.9%	15.3%
New Jersey	41.0	6.1	53.6	31.9	76.7%	19.0%
Washington	19.0	1.9	26.0	13.2	73.1%	14.6%

All States	29.8	5.8	41.3	28.6	72.2%	23.0%??
------------	------	-----	------	------	-------	---------

Table #3: Current Hardware Use

The number of machines per lab also vary significantly from state to state but all of the states report a greater number of PC's than Macs per lab. California and New Jersey report the highest average number of PC's per lab with 44.0 and 53.6 respectively. They also report the highest average number of Macs per lab with 35.2 and 31.9 respectively. Massachusetts reports an average of 31.9 PCs per lab and 12.9 Macs per lab, Washington shows 26.0 PCs per lab and 13.2 Macs, and New Hampshire reports 24.9 PCs per lab and 12.0 Macs per lab.

The average number of computers of each hardware platform per school is less relevant, since high schools tend to concentrate computers in lab settings. These results, however, further demonstrates the preeminence of PC's. In New Jersey, for example, schools on average have 41.0 PCs and 6.1 Macs. The consistency of these figures across the states is demonstrated by the overall results which show that schools on average use 29.8 PCs and 5.8 Macs to teach computer science.

Schools Purchasing New Machines

Fewer than half of all of the schools responding reported that they were intending to purchase new computers in the coming (2000-2001) school year. Massachusetts reported the highest percentage of schools planning to purchase new computers with 48.9%. New Hampshire, with 30.0% reported the lowest number of schools planning to purchase computers.

State	Buying	Buying PCs	Buying Macs
California	43.2%	89.5%	18.4%
Massachusetts	48.9%	78.3%	34.8%
New Hampshire	30.8%	62.5%	37.5%
New Jersey	40.0%	88.2%	17.6%
Washington	46.3%	84.2%	15.8%
All States	42.7%	85.6%	21.3%

Table #4: New Computer Purchases

Among the schools planning to purchase computers, again results for the PCs and Macs differ significantly. Of the 43.2% of California schools planning to purchase new computers, for example, 89.5% report planning to purchase PCs and 18.4% report planning to purchase Macs. New Hampshire reported the highest percentage of school planning to purchase Macs with 37.5% of schools reporting plans to purchase computers. The overall results for all states show 42.7% of schools planning to purchase computers within the 2000-2001 school year, with 85.6% planning to purchase PCs and 21.3% planning to purchase Macs.

Although the researchers attempted to gather more detailed information concerning the numbers of machines each school was planning to purchase, the results proved unreliable since most respondents indicated that they were unable to accurately predict numbers and so these results are not reported.

Hardware Purchase Criteria

The researchers surmised that the criteria on which schools based their computer purchases would be quite broad. In order to provide respondents with the widest possible latitude, the survey did not include suggested responses. The data therefore consists entirely of educator-generated categories.

For the sake of clarification it is important to note that the category "compatibility" refers to compatibility with existing types of hardware and compatibility with computer networks. Teachers also tended to differentiate between "durability" and "reliability", with the former indicating the ability of the hardware to stand up to student use over long periods of time and the latter referring to, as one teacher noted "my feeling that it kept doing what it was expected to do on a day-to-day basis with no nasty little surprises".

While there was some difference in emphasis from state to state, overall the reported criteria was fairly consistent. The cumulative responses from all states indicate that cost (26.9%), compatibility (18.4%), reliability (16.5%), speed (12.8%), and memory capacity (9.9%) are the most commonly identified criteria for computer hardware selection.

In California cost (24.4%), compatibility (18.8%), reliability (15.9%), speed (8.5%), and ease-of use (8.0%) were the most frequently reported criteria. California educators also were more likely to list technical support as a purchase criterion (6.3%) while Massachusetts and Washington responses contained no reference to this criterion.

Cost (28.2%) was also the most frequently reported criterion in New Jersey, followed by speed and compatibility (both 14.1%), and durability (7.1%). Upgradability (5.9%) was more commonly reported in New Jersey than in any other state. Cost was also the predominant concern in Washington state (31.7%), followed by speed (24.4%), memory capacity (19.5%), reliability (17.1%), and compatibility (14.6%).

Purchase Criteria	CA	MA	NH	NJ	WA	All
School Board Decision	6.8%	6.4%	0.0%	2.4%	4.9%	5.1%
Compatibility	18.8%	27.7%	19.2%	14.1%	14.4%	18.4%
Cost	24.4%	25.5%	34.6%	28.2%	31.7%	26.9%
Durability	7.4%	0.0%	0.0%	7.1%	0.0%	5.1%
Ease-of-Use	8.0%	6.4%	7.7%	0.0%	7.3%	5.9%
Industry Standard	2.3%	6.4%	3.8%	3.5%	9.8%	4.0%
Internet Capabilities	2.3%	2.1%	0.0%	0.0%	2.4%	1.6%
Networkable	4.0%	0.0%	7.7%	2.4%	4.9%	3.5%
Manufacturer	1.7%	0.0%	0.0%	1.2%	0.0%	1.1%
Memory Capacity	5.7%	10.6%	11.5%	12.9%	19.5%	9.9%
Reliability	15.9%	19.1%	38.5%	9.4%	17.1%	16.5%
Speed	8.5%	14.9%	15.4%	14.1%	24.4%	12.8%
Teacher Decision	1.7%	2.1%	0.0%	1.2%	0.0%	1.3%
Technical Support	6.3%	0.0%	3.8%	2.4%	0.0%	3.7%
Upgradeable	1.7%	4.3%	0.0%	5.9%	0.0%	2.7%
Warranty	6.3%	0.0%	0.0%	3.5%	0.0%	3.7%

Table #5: Criteria for New Computer Purchases

In Massachusetts compatibility (27.7%) was the most frequently mentioned criterion with cost (25.5%), reliability (19.1%), speed (14.9%), and memory capacity (8.0%) providing the remainder of the five most frequently reported criteria. The results for New Hampshire are similar, with reliability (38.5%) as the most frequently mentioned criterion, followed by cost (34.6%), compatibility (19.2%), speed (15.4%), and memory capacity (11.5%).

Teaching Programming in Grades 10, 11, and 12

Schools were asked to indicate all grades in which programming was taught. With the influence of the Advanced Placement exams at the higher grade levels it was expected that more schools would indicate teaching programming in later rather than earlier grades. This was not, however, always the case.

State	Grade 10	Grade 11	Grade 12
California	38.6%	47.7%	48.3%
Massachusetts	44.7%	55.3%	57.4%
New Hampshire	61.5%	57.7%	57.7%
New Jersey	56.5%	62.4%	69.4%
Washington	29.3%	36.6%	34.1%
All States	44.0%	51.5%	48.0%

Table #6: Programming in Grades 10 through 12

As expected the combined results for all states indicated the smallest percentage for Grade 10 (44.0%). They also indicated, however, that a greater percentage of schools taught programming in Grade 11 (51.5%) than in Grade 12 (48.0%). The results for individual states also indicate differences. New Hampshire, for example, reported more schools teaching programming in Grade 10 (61.5%) than in either Grade 11 or 12 (both 57.7%). New Jersey reported the largest number of Grade 11 classes (62.4%) and Grade 12 classes (69.4%) and the second highest number of Grade 10 classes (56.5%). This is consistent with earlier results which showed that more teachers in New Jersey teach programming. The results for Washington state were also consistent with earlier results, showing the lowest number of classes in all three grades with 29.3% of schools reporting programming classes in Grade 10, 36.6% in Grade 11, and 34.1% in Grade 12.

Programming Languages Taught in Grade 10

Respondents were asked to indicate which programming languages they taught in each grade. (Because instruction often involves more than one programming language in each grade, totals often exceed 100%.)

Language Taught in Grade 10	CA	MA	NH	NJ	WA	All
Basic	26.5%	0.0%	18.8%	52.1%	0.0%	27.9%
C++	48.5%	57.1%	62.5%	43.8%	50.0%	49.7%
HTML	11.8%	23.8%	18.8%	2.1%	25.0%	12.1%
Java	5.9%	0.0%	12.5%	10.4%	0.0%	6.7%
JavaScript	0.0%	4.8%	6.3%	0.0%	8.3%	1.8%
Pascal	10.6%	4.5%	25.0%	13.2%	8.3%	11.2%
Perl	0.0%	4.8%	6.3%	0.0%	0.0%	1.2%
Visual Basic	17.6%	42.9%	31.3%	20.8%	50.0%	24.2%

Table #7: Programming Languages in Grade 10

As Table #7 indicates, C++ and Visual Basic are the most commonly taught programming languages among those teaching programming in Grade 10. The results for Massachusetts (57.1%), California (48.6%), and New Hampshire (62.5%) indicate that C++ is the most widely used programming language. This is also reflected in the overall tabulations which show 49.7% of reporting teachers use C++ for Grade 10 instruction.

Use of Basic is also widely reported in California, New Hampshire, and New Jersey. In Washington State and Massachusetts, however, use of Visual Basic is extensive and may indicate that these states have simply upgraded their version of Basic. Although Visual Basic is not the most commonly used language in any of the states except Washington (where both C++ and Visual Basic use is reported at 50.0%), in all cases it is the second most commonly reported language for Grade 10.

Use of Pascal (which was once predominant in educational programming) is still reported across all states but, with the exception of New Hampshire (25.0%), its use has been significantly decreased.

Washington state respondents report the greatest use of Hypertext Markup Language commonly referred to as HTML (25.0%), followed by Massachusetts (23.8%), New Hampshire (18.8%), and California (11.8%).

California (5.9%), New Hampshire (12.5%), and New Jersey (10.4%) are the only states reporting use of Java in Grade 10.

Programming Languages Taught in Grade 11

Language Taught in Grade 11	CA	MA	NH	NJ	WA	All
Basic	13.1%	11.5%	6.7%	18.9%	6.7%	13.5%
C++	64.3%	61.5%	73.3%	77.4%	53.3%	67.4%
HTML	13.1%	19.2%	20.0%	1.9%	20.0%	11.9%
Java	4.8%	11.5%	13.3%	9.4%	13.3%	8.3%
JavaScript	3.6%	0.0%	6.7%	0.0%	0.0%	2.1%
Pascal	8.3%	7.7%	26.7%	11.3%	6.7%	10.4%
Perl	1.2%	0.0%	6.7%	0.0%	0.0%	1.0%
Visual Basic	23.8%	30.8%	33.3%	20.8%	40.0%	25.9%
Visual C++	2.4%	0.0%	0.0%	1.9%	0.0%	3.0%

Table #8: Programming Languages in Grade 11

Table #8 shows that in Grade 11 C++ begins to outstrip Visual Basic by a sizable margin (greater than 30%) in all states except Washington (where the difference between C++ and Visual Basic is only 13.3%). This use pattern is also evident from the overall tabulations which show 67.4% of reporting teachers use C++ for Grade 11 instruction. Considerable use of Visual Basic, however, is reported in all of the states and reflected in the overall results with 25.9% of respondents reporting use.

The results for New Hampshire (20.0%), Massachusetts (19.2%), and to a lesser extent California (13.1%) all indicate use of HTML for programming instruction.

Use of Pascal is also reported in Grade 11 across all states but again, with the exception of New Hampshire (26.7%), its use has been significantly decreased.

Programming Languages Taught in Grade 12

As Table 9 indicates, 15.5% of respondents from all states reported that they were considering changing programming languages. The results for each state varied only slightly. New Hampshire reported the highest percentage of schools considering new programming languages with 19.2%, followed by Massachusetts (17%), then California and New Jersey (both 16.5%). Teachers in Washington State showed the least inclination to adopt new programming languages (14.6%).

Language Taught in Grade 12	CA	MA	NH	NJ	WA	All

Basic	9.4%	7.4%	6.7%	22.0%	0.0%	12.0%
C++	71.8%	70.4%	80.0%	76.3%	64.3%	73.0%
HTML	11.8%	14.8%	26.7%	8.5%	14.3%	12.5%
Java	7.1%	7.4%	13.3%	16.9%	7.1%	10.5%
JavaScript	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Pascal	9.4%	7.4%	20.0%	5.1%	7.1%	8.3%
Perl	1.2%	0.0%	6.7%	0.0%	0.0%	1.0%
Visual Basic	18.8%	29.6%	33.3%	22.0%	42.9%	24.0%
Visual C++	2.4%	3.7%	0.0%	1.9%	7.1%	3.0%

Table #9: Programming Languages in Grade 12

Programming Languages Under Consideration

The following Table shows the percentages for the languages under consideration by those considering a change. (Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.)

New Language Considered	CA	MA	NH	NJ	WA	All
C++	41.4%	50.0%	40.0%	21.4%	16.7%	35.5%
Java	31.0%	37.5%	60.0%	42.9%	33.3%	37.1%
Visual Basic	24.1%	25.0%	40.0%	50.0%	33.3%	32.3%
Visual C++	3.4%	0.0%	0.0%	0.0%	0.0%	1.6%
Other	13.8%	0.0%	0.0%	0.0%	16.7%	8.1%

Table #10: New Programming Languages Considered

The results for Massachusetts (50.0%) and California (41.4%), indicate a strong preference for C++. In New Hampshire both Visual Basic (40.0%) and C++ (40.0%) are outstripped by Java (60%), In New Jersey, both Visual Basic (50.0%) and Java (40.0%) are also popular choices. The results are similar for Washington, where reported consideration of Java and Visual Basic are identical (33.3%).

The strong performance of more than one language in each state may indicate that schools are attempting to choose between a number of possible options. The overall results indicate that C++, Java, and Visual Basic are all strong candidates for schools considering a change in programming language with Java showing a slight edge. These results are likely to be influenced by external factors which will be addressed in the Discussion section.

Selection Criteria for Instructional Programming Languages

As Table 11 demonstrates, when the results from all schools are combined, a total of twenty-one different criteria for the selection of instructional programming language are reported. This indicates a broad-range of factors affect language selection. The results, both individually and combined, however, show that there are three predominant criteria: industrial relevance, ease-of-use, and the Advanced Placement exam.

Programming Language Criteria	CA	MA	NH	NJ	WA	All
Advanced Placement Exam	19.9%	8.5%	15.4%	23.5%	4.9%	17.3%
College Requirements	5.7%	2.1%	3.8%	9.4%	0.0%	5.3%
Concepts	2.3%	2.1%	0.0%	3.5%	0.0%	2.1%
Curriculum Guidelines	0.6%	2.1%	0.0%	0.0%	4.9%	1.1%
District Decisions	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Ease-of-Use	8.0%	17.0%	26.9%	15.3%	17.1%	13.1%
Environment	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Features	0.0%	2.1%	0.0%	1.2%	2.4%	0.8%
Graphics	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
Hardware Requirements	0.0%	0.0%	0.0%	3.5%	0.0%	0.8%
Industrial Relevance	19.3%	19.1%	42.3%	29.4%	22.0%	23.5%
Object Oriented	1.1%	4.3%	3.8%	0.0%	2.4%	1.6%
Open Source	0.6%	2.1%	0.0%	1.2%	2.4%	1.1%
Price	1.1%	0.0%	0.0%	0.0%	0.0%	0.5%
Resources Available	2.3%	2.1%	0.0%	1.2%	0.0%	1.6%
Skills Transfer	1.1%	6.4%	3.8%	3.5%	2.4%	2.7%
Structured	0.0%	2.1%	3.8%	0.0%	0.0%	0.5%
Student Interest	1.1%	2.1%	3.8%	2.4%	2.4%	1.9%
Teacher Knows It	2.3%	0.0%	3.8%	2.4%	0.0%	1.9%
Variety	0.0%	0.0%	0.0%	1.2%	0.0%	0.3%
Vendor (Brand Name)	0.0%	2.1%	0.0%	1.2%	0.0%	0.5%

Table #11: Programming Language Selection Criteria

Overall 23.5% of schools report that perceived industrial relevance (use in business and industry) plays a major role in programming language selection. This is echoed in the state-by-state results, where industrial relevance is the most commonly reported criterion in New Hampshire (42.3%), New Jersey (29.4%), Washington (22.0%), and Massachusetts (19.1%). In California it is the second most reported criterion (19.3%).

The second most common criterion reported by all states is the Advanced Placement exam. Overall 17.3% of respondents indicate that the programming language used for this exam (as mandated by the College Board) determines which technology they will use for instructional purposes. While California is the only state where the Advanced Placement exam is the most commonly reported criterion (19.9%), it is also a major decision factor in both New Jersey (23.5%) and New Hampshire (15.4%). In Massachusetts (8.5%) and Washington (4.9%) it is the third most commonly reported criterion.

Educators also indicate that ease-of-use is a factor in instructional programming language selection. Overall 13.1% of respondents from all states list it among their selection criteria. Results for the individual states also indicate its importance, with New Hampshire (26.9%) reporting it most-

frequently, followed by Washington (17.1%), Massachusetts (17.0%), New Jersey (15.3%), and California (8.0%).

Ranking of Instructional Support Materials

Respondents were asked to indicate which of six listed teaching resources they found most valuable by rating them on a scale of one to six with one being the most valuable and six being the least. These responses were tabulated to produce an overall rating for each resource. While the results differ from state-to-state, the combined results show an essentially two-tiered response, with textbooks (2.4), teachers guides (2.4), and example programs (2.4) achieving the highest rating, followed by electronic lessons (3.0), student workbooks (3.0) and finally project collections (3.3).

Support Materials	CA	MA	NH	NJ	WA	All
Electronic Lessons	2.8	3.3	3.5	3.5	2.4	3.0
Example Programs	2.5	2.6	2.3	2.5	2.0	2.4
Project Collections	3.1	3.6	3.3	3.6	3.1	3.3
Student Workbooks	2.8	3.1	3.1	3.5	2.6	3.0
Teachers Guide	2.2	2.5	3.0	2.6	1.9	2.4
Textbook	2.4	2.4	2.5	2.5	2.0	2.4

Table #12: Support Material Ranking

Ranking of Sources of Continuing Education

Respondents were also asked to indicate which of six listed sources of continuing education and skills upgrading they found most valuable by rating them on a scale of one to six with one being the most valuable and six being the least. These responses were then tabulated to produce an overall rating for each source.

Continued Education	CA	MA	NH	NJ	WA	All
Upgrading	3.4	3.9	3.1	3.4	3.6	3.4
Conferences	2.3	2.4	2.4	2.3	2.4	2.4
Internet	3.3	4.0	3.7	3.8	3.7	3.6
Self-directed Learning	2.3	2.9	2.2	2.7	2.7	2.5
Inservice	3.1	3.0	3.1	3.3	2.8	3.1
Vendor Events	4.2	4.6	4.8	4.5	4.4	4.4

Table #12: Continued Education Source Ranking

Unlike Table 12, the results in Table 13 indicate a hierarchical rating. Conferences (2.4) closely followed by self-directed learning (2.5) are reported as the most valuable sources of continuing education for computer teachers. Inservice (workshops provided by the school or school district) and upgrading (courses provided by college/university institutions) fall into the middle ground with ratings of 3.1 and 3.4 respectively. The Internet (3.6) is ranked as the second least valuable source for continuing education and vendor events (4.4) are ranked least valuable.

DISCUSSION

Teaching Responsibilities

The predominance of educators who identify themselves as computer applications teachers as opposed to computer science teachers may be related to the perception that computer applications skills (that is, learning to use the computer as a tool) should be the focus of computer education at the high school level. While there is growing acknowledgment that all students require some level of competency with computers, there is less awareness of the importance of fundamental computer science concepts in the attainment of information fluency. This emphasis on teaching students how to use the computer as a tool also diminishes the likelihood that computer science will be part of the core (mandatory) curriculum at a given school.

The researchers were somewhat surprised to discover that fewer teachers than expected reported teaching both computer science and applications, since we had assumed that the computer science teacher was likely to be the school “computer expert” and thus responsible for all curriculum areas relating to computers. The fact that fewer than half of the respondents in any state reported teaching responsibilities for both computer science and computer applications and more than half indicated that they held teaching responsibilities in other curriculum areas may indicate that many schools tend to offer either computer science or computer applications rather than both.

Hardware Use and Purchase

The results relating to hardware use indicate that PC compatibles are the dominant hardware platform in computer science education, both in terms of overall use in schools and the number of machines per school. Despite its earlier predominance in some states, the Apple Macintosh maintains a persistent but much less prominent place.

Further evidence of the PC compatible’s dominance can be seen in the reported purchasing intentions of the survey respondents. More than 80% of the schools who are planning to purchase new computers within the next school year indicate that they are planning to purchase PC compatibles as contrasted to the approximately 20% planning to purchase Macs. This purchasing pattern is consistent with current use.

Although the researchers did not attempt to establish a direct connection between hardware platforms actually used in the schools and the educator’s stated criteria for hardware purchases, this may prove fertile ground for future research, since the results of this survey indicate a fairly consistent hierarchy of educator concerns. The importance of cost as a criterion for hardware purchase came as no surprise to the researchers in light of the current environment of fiscal hardship in many schools and school districts.

The concern with compatibility may also be seen as a resource issue. Often, schools do not have easy access to computer technicians so the responsibility for maintaining the hardware and networks often falls to the computer teacher. Ensuring compatibility with existing machines and networks reduces overall complexity by allowing teachers to concentrate on a single hardware platform. This emphasis on compatibility may also promote long

term stability in hardware use patterns since schools are more likely to continue using and purchasing the hardware platforms with which they are already comfortable.

Despite this apparent consistency of use and purchase, schools do not place similar emphasis on the hardware brand or manufacturer and so platform loyalty has little or no connection to vendor loyalty. This, again, may reflect the high level of price point sensitivity in the education market which leads schools to search for, what one teacher described as “the best bang anyone will give us for the buck”. When teachers refer to the “bang” most often they are referring to the most frequently cited performance criteria: reliability, memory capacity, and speed. The researchers found it interesting that even in an academic setting these criteria would be considered more important than pedagogical concerns such as ease-of-use.

The researchers were also somewhat surprised by the infrequent reference to hardware industry standards (what schools perceive as being most commonly used in business and industry) since industrial relevance is the most-widely reported criterion for programming language selection. One possible explanation is that schools, with no hope of matching the frequent replacement cycle for computer hardware in industry, have simply abandoned this criterion as unattainable.

The relatively infrequent mention of Internet capabilities as a hardware selection criterion may result from a number of possible factors. While an increasing percentage of schools are now reported to be wired (have Internet access), this capability is often limited to a small number of machines in the schools rather than the larger labs used for computer science instruction. In fact, in a traditional computer science lab setting, teachers may find such access more of a hindrance than a help due to the potential for students to be distracted from classroom tasks. Internet capabilities may also be perceived as less related to the computer science curriculum (which focuses on algorithm development and programming) than to the computer applications curriculum (which focuses on using the computer as a tool). One final possibility is that Internet capabilities are now so ubiquitous that educators no longer consider this a criterion requiring consideration.

Computer Science in Grades 10, 11, and 12

Considering the widely-reported career opportunities for computer professionals, the results indicate that fewer than 50% of schools teach computer science in Grades 10 and 12 and just slightly more than 50% do so in Grade 11. These results are particularly surprising in California, for example, since its Silicon Valley is world renown as a site of both hardware and software innovation and employment opportunities.

The relatively low number of high schools offering computer science instruction may be related to a number of factors. Schools may perceive that, due to its resource requirements and/or academic rigor, computer science instruction is best left to the college/university institutions. Fisher, Margolis, and Miller (1997), for example refer to frequently expressed student impressions of computer science which include the prevalence of “really smart” students and an extremely heavy workload,

They also note that many young women still consider computer science a predominantly male domain. This preconception may discourage young women from taking computer science classes which may in turn contribute to smaller class sizes which may make it untenable for smaller schools to offer such courses.

The lack of qualified or interested teachers may also affect the education system's ability to offer computer science courses, since the salaries and working conditions offered by most schools do not compete with those in business and industry, especially for individuals with skills which are in such high demand.

Programming Languages

The reported use of programming languages in Grades 10 through 12 show a pattern similar to that reported in earlier studies of college/university institutions (Stephenson and West, 1998). In both educational environments there is a shift from structured programming languages such as Pascal and some versions of Basic to object-oriented programming languages such as C++ and Java.

Approximately 25% of teachers reported using some version of Basic for instruction in Grade 10. This may be because Basic is widely available and known to many long-time computer science teachers. In Grades 11 and 12, however, Basic use drops by greater than 50%, indicating that it is not widely perceived as a suitable language for students pursuing computer science at college or university.

The growing popularity of Visual Basic may arise from former Basic teachers upgrading to a version of Basic which they perceive will provide students with object-oriented programming experience. Visual Basic's facilities for the creation of graphical user interfaces (GUIs) may also prove attractive to students interested in graphics and web applications as opposed to traditional computer science. This facet of Visual Basic use, however, has created some concern among educators that using GUI features may actually interfere with the learning of core computer science concepts (Wittenburg, 1997; Martin, 1999).

The growing interest in webpages may also explain the fairly consistent use of HTML (and to a lesser extent JavaScript) in Grades 10 through 12. Again, however, there is some debate as to whether web scripting languages support the learning of core computer science concepts.

The popularity of C++ in all grades may be explained in a number of ways. First, C++ is an object-oriented language with all of the required object-oriented features such as classes, inheritance, and concurrency. C++ is also seen as an industrially relevant language, leading students and some educators to believe that learning C++ contributes significantly to employability upon graduation. Finally, the Advanced Placement exams in Computer Science must now be written in C++. This final factor provides a powerful incentive for schools to use C++ as an instructional programming language, especially in Grades 11 and 12.

The extent to which the Advanced Placement exams influence high school programming will become even more apparent in the next four years. In 2000 the College Board announced that beginning in the 2003-2004 school year the

exams must be written in Java (as opposed to C++). This decision, combined with the growing number of college/university institutions that have already switched to Java or are planning to do so (Stephenson and West, 1998) is likely to have a profound impact on high school computer science over the next few years. The results reported for new programming languages being considered provides some early evidence of this impending shift to Java. Among the schools reporting that they are considering switching to C++, Java, or Visual Basic, the results for Java are marginally higher.

Review of the results relating to selection criteria for instructional programming languages further supports the conclusion that use of object-oriented programming languages in general, and Java in particular, will continue to increase at the high school level. The importance educators place on industrial relevance provides an obvious advantage to C++ and Java, which have received considerable coverage in both the technical and popular press. The numerous citations of the Advanced Placement exam as a criterion in language selection supports the current use of C++, but this support will begin to shift as the transition date to Java approaches (Biddle and Tempero, 1999; Andrae et al 2000).

The only frequently reported criterion which may argue against the use of object-oriented languages such as C++ and Java is ease-of-use. Some educators in general see the complexity of object-oriented languages as a barrier to student learning and at the very least perceive them to be of questionable suitability as an introductory programming language.

Instructional Support Materials

The results relating to instructional support materials indicate a slight preference for traditional text-based resources such as textbooks and teacher's guides. These results may prove surprising to administrators who assume that the presence of computer hardware and software eliminates the need for traditional teaching resources. This is not to say, however, that teachers do not value and use computer-based learning materials. The results indicate that educators perceive computer-based resources such as electronic lessons to be as useful as student workbooks. This is not surprising, since both of these resources are likely to be viewed primarily as tools for individual student drill and practice.

Opportunities for Continuing Education

Constantly changing technology creates a pressing need for computer science teachers to upgrade their skills but opportunities for doing so can be restricted by resources available for inservice (learning opportunities provided by the school district) and limited access to upgrading by outside agencies such as local colleges and universities. The latter is particularly difficult for teachers in isolated rural areas. It is therefore not surprising that many teachers rely primarily on self-directed learning (teaching themselves about new hardware, networks, and software for example by reading textbooks and manuals). While this kind of learning can support teachers in learning new technical skills, it is less effective in helping teachers improve and expand their teaching strategies. The value computer teachers place on conferences therefore comes as no surprise, since discipline-based

educational conferences provide a unique “all in one” opportunity to learn about new tools and ideas and to build networks with teachers from other schools, districts, states, and even countries.

The significantly lower rating for the Internet as a vehicle for teacher skills upgrading may come as a surprise to some, especially if they see distance education as a panacea to teacher learning needs. This data may result from a lack of access, lack of awareness, or lack of materials specially designed to address the computer science curriculum from the teacher’s perspective.

It is likewise not possible to determine from the data why vendor events were perceived to be the least useful venue for teacher learning. Again, the data may result from too few vendors providing teacher training or from a perceived lack of direct classroom relevance for events that are vendor-sponsored.

CONCLUSIONS

The results of this study highlight the multiplicity of factors affecting computer science education in U.S. high schools. While the PC compatible is now well established as the dominant hardware platform for computer science instruction, their domination is not complete and as the Macintosh maintains a small but loyal following in education. Despite the complexity of issues involved in hardware selection, schools also appear to base acquisition decisions on a relatively stable set of criteria. School focus on cost, compatibility, and reliability. They may wish for the latest and greatest but they buy from the vendor who offers the biggest bang for the buck.

Programming language wars have raged at all levels of education and the high schools are subject to even greater external pressures than their post-secondary colleagues. High schools, in their attempt to prepare students for future study and work are influenced by both education and industry. New programming paradigms such as object-oriented programming are trickling down to the high school level. The influence of the College Board also plays a major role in high school computing and the announced intention to require the Computer Science Advanced Placement exams to be written in Java by the 2003-2004 school year is likely increase the number of Java-using schools dramatically over the next six years.

Although in a less direct manner, this survey also points to the need for a profound engagement with issues relating to teacher training (both pre-service and inservice) as teachers endeavor to keep up with constantly changing technology and, at the same time, improve their teaching skills. If teachers are to meet the demand for professional development their academic discipline and their students require, more effective and assessable kinds of training will have to be made available.

ACKNOWLEDGEMENTS

The author would like to thank George Milbrant and JCSE Editor for their careful read of an early draft of this article. The researchers would also like to thank IBM for their generous support of this project.

CONTACTS

For further information please contact Chris Stephenson at 1-800-361-8324 or chris@hsa.on.ca.

APPENDIX A

CUMULATIVE RESPONSES (ALL STATES)

Teaching Responsibilities	Percentages
% Teaching Programming	57.0%
% Teaching Applications	67.7%
% Teaching Other	53.3%
% Teaching Prog+Applications	32.6%
% Teaching Neither	7.8%

Current Hardware Use	Results
Average Number of PCs	29.8
Average Number of Macs	5.8
Average PC Lab Size	41.3
Average Mac Lab Size	28.6
Percentage with PC Lab	72.27%
Percentage with Mac Lab	23.00%

Intended Hardware Purchase	Percentage
Buying new machines	42.7%
Buying PC (if buying)	85.6%
Buying MAC (if buying)	21.3%

Hardware Purchase Criteria	Percentage
School Board Decision	5.1%
*Compatibility	18.4%
Cost	26.9%
Durability	5.1%
Ease-of-Use	5.9%
Industry Standard	4.0%
Internet Capabilities	1.6%
Networkable	3.5%
Manufacturer (Brand Name)	1.1%
Memory Capacity	9.9%
Reliability	16.5%
Speed	12.8%
Teacher Decision	1.3%
Technical Support	3.7%
Upgradeable	2.7%
Warranty	3.7%

* Compatibility refers to compatibility with schools networks and with existing hardware.

Grade	Teaching Programming
Grade 10	44.0%
Grade 11	51.5%
Grade 12	48.0%

Language Taught	Grade 10	Grade 11	Grade 12
Basic	27.9%	13.5%	12.0%
C++	49.7%	67.4%	73.0%
HTML	12.1%	11.9%	12.5%
Java	6.7%	8.3%	10.5%
JavaScript	1.8%	2.1%	0.0%
Pascal	11.2%	10.4%	8.3%
Perl	1.2%	1.0%	1.0%
Visual Basic	24.2%	25.9%	24.0%
Visual C++	0.0%	1.6%	3.0%

A total of 15.5% of respondents reported that they were considering changing programming languages. The following table shows the percentages for the languages under consideration by those considering a change. Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.

New Language	Percent
C/C++	35.5%
Java	37.1%
Visual Basic	32.3%
Visual C++	1.6%
Other	8.1%

Programming Language Criteria	Percentage
Advanced Placement Exam	17.3%
College Requirements	5.3%
Concepts	2.1%
Curriculum Guidelines	1.1%
District Decisions	0.0%
Ease-of-Use	13.1%
Environment	0.0%
Features	0.8%
Graphics	0.0%
Hardware Requirements	0.8%

Industrial Relevance	23.5%
Object Oriented	1.6%
Open Source	1.1%
Price	0.5%
Resources Available	1.6%
Skills Transfer	2.7%
Structured	0.5%
Student Interest	1.9%
Teacher Knows It	1.9%
Variety	0.3%
Vendor (Brand Name)	0.5%

Teachers were asked to rank the importance/usefulness of a list of teaching resources. Here is their ranking from **most** (1) to **least** important (6) .

Example Programs (2.4)
 Teachers Guide (2.4)
 Textbook (2.4)
 Electronic Lessons (3.0)
 Student Workbooks (3.0)
 Project Collections (3.3)

Teachers were asked to rank the importance/usefulness of a list of potential sources of continued learning/skills upgrading. Here is their ranking from **most** to **least** important.

Conferences (2.4)
 Self-directed Learning (2.5)
 School/Board Inservice (3.1))
 Upgrading Courses (3.4)
 Internet (3.6)
 Vendor Events (4.4)

APPENDIX B

CALIFORNIA

A total of 991 schools were surveyed in the state of California. The following tables provide the results for the 176 schools responding. The response rate for this state was 17.75%.

Teaching Responsibilities	Percentages
% Teaching Programming	49.4%
% Teaching Applications	75.6%
% Teaching Other	58.7%
% Teaching Prog+Applications	33.5%
% Teaching Neither	8.5%

Current Hardware Use	Results
Average Number of PCs	29.8
Average Number of Macs	8.0
Average PC Lab Size	44.0
Average Mac Lab Size	35.2
Percentage with PC Lab	67.61%
Percentage with Mac Lab	22.73%

Intended Hardware Purchase	Percentage
Buying new machines	43.2%
Buying PC (if buying)	89.5%
Buying MAC (if buying)	18.4%

Hardware Purchase Criteria	Percentage
School Board Decision	6.8%
*Compatibility	18.8%
Cost	24.4%
Durability	7.4%
Ease-of-Use	8.0%
Industry Standard	2.3%
Internet Capabilities	2.3%
Networkable	4.0%
Manufacturer (Brand Name)	1.7%
Memory Capacity	5.7%
Reliability	15.9%
Speed	8.5%

Teacher Decision	1.7%
Technical Support	6.3%
Upgradeable	1.7%
Warranty	6.3%

* Compatibility refers to compatibility with schools networks and with existing hardware.

Grade	Teaching Programming
Grade 10	38.6%
Grade 11	47.7%
Grade 12	48.3%

Language Taught	Grade 10	Grade 11	Grade 12
Basic	26.5%	13.1%	9.4%
C++	48.5%	64.3%	71.8%
HTML	11.8%	13.1%	11.8%
Java	5.9%	4.8%	7.1%
JavaScript	0.0%	3.6%	0.0%
Pascal	10.6%	8.3%	9.4%
Perl	0.0%	1.2%	1.2%
Visual Basic	17.6%	23.8%	18.8%
Visual C++	0.0%	2.4%	2.4%

A total of 16.5% of respondents reported that they were considering changing programming languages. The following table shows the percentages for the languages under consideration by those considering a change. Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.

New Language	Percent
C/C++	41.4%
Java	31.0%
Visual Basic	24.1%
Visual C++	3.4%
Other	13.8%

Programming Language Criteria	Percentage
Advanced Placement Exam	19.9%
College Requirements	5.7%
Concepts	2.3%
Curriculum Guidelines	0.6%
Ease-of-Use	8.0%
Industrial Relevance	19.3%
Object Oriented	1.1%

Open Source	0.6%
Price	1.1%
Resources Available	2.3%
Skills Transfer	1.1%
Student Interest	1.1%
Teacher Knows It	2.3%

Teachers were asked to rank the importance/usefulness of a list of teaching resources. Here is their ranking from **most** (1) to **least** important (6) .

Teachers Guide (2.2)
 Textbook (2.4)
 Electronic Lessons (2.8)
 Student Workbooks (2.8)
 Example Programs (2.5)
 Project Collections (3.1)

Teachers were asked to rank the importance/usefulness of a list of potential sources of continued learning/skills upgrading. Here is their ranking from **most** to **least** important.

Conferences (2.3)
 Self-directed Learning (2.3)
 School/Board Inservice (3.1)
 Internet (3.3)
 Upgrading Courses (3.4)
 Vendor Events (4.2)

APPENDIX C

MASSACHUSETTS

A total of 539 schools were surveyed in the state of Massachusetts. The following tables provide the results for the 46 schools responding. The response rate for this state was 8.71%.

Teaching Responsibilities	Percentages
% Teaching Programming	61.7%
% Teaching Applications	72.3%
% Teaching Other	52.1%
% Teaching Prog+Applications	38.3%
% Teaching Neither	4.3%

Current Hardware Use	Results
Average Number of PCs	25.1
Average Number of Macs	2.7
Average PC Lab Size	31.9
Average Mac Lab Size	12.9
Percentage with PC Lab	78.72%
Percentage with Mac Lab	21.28%

Intended Hardware Purchase	Percentage
Buying new machines	48.9%
Buying PC (if buying)	78.3%
Buying MAC (if buying)	34.8%

Hardware Purchase Criteria	Percentage
School Board Decision	6.4%
*Compatibility	27.7%
Cost	25.5%
Ease-of-Use	6.4%
Industry Standard	6.4%
Internet Capabilities	2.1%
Memory Capacity	10.6%
Reliability	19.1%
Speed	14.9%
Teacher Decision	2.1%
Upgradeable	4.3%

* Compatibility refers to compatibility with schools networks and with existing hardware.

Grade	Teaching Programming
Grade 10	44.7%
Grade 11	55.3%
Grade 12	57.4%

Language Taught	Grade 10	Grade 11	Grade 12
Basic	0.0%	11.5%	7.4%
C++	57.1%	61.5%	70.4%
HTML	23.8%	19.2%	14.8%
Java	0.0%	11.5%	7.4%
JavaScript	4.8%	0.0%	0.0%
Pascal	4.5%	7.7%	7.4%
Perl	4.8%	0.0%	0.0%
Visual Basic	42.9%	30.8%	29.6%
Visual C++	0.0%	0.0%	3.7%

A total of 17% of respondents reported that they were considering changing programming languages. The following table shows the percentages for the languages under consideration by those considering a change. Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.

New Language	Percent
C/C++	50.0%
Java	37.5%
Visual Basic	25.0%

Programming Language Criteria	Percentage
Advanced Placement Exam	8.5%
College Requirements	2.1%
Concepts	2.1%
Curriculum Guidelines	2.1%
Ease-of-Use	17.0%
Features	2.1%
Industrial Relevance	19.1%
Object Oriented	4.3%
Open Source	2.1%
Resources Available	2.1%
Skills Transfer	6.4%
Structured	2.1%
Student Interest	2.1%

Vendor (Brand Name)	2.1%
---------------------	------

Teachers were asked to rank the importance/usefulness of a list of teaching resources. Here is their ranking from **most** (1) to **least** important (6) .

- Textbook (2.4)
- Teachers Guide (2.5)
- Example Programs (2.6)
- Student Workbooks (3.1)
- Electronic Lessons (3.3)
- Project Collections (3.6)

Teachers were asked to rank the importance/usefulness of a list of potential sources of continued learning/skills upgrading. Here is their ranking from **most** to **least** important.

- Conferences (2.4)
- Self-directed Learning (2.9)
- School/Board Inservice (3.0)
- Upgrading Courses (3.9)
- Internet (4.0)
- Vendor Events (4.6)

APPENDIX D

NEW HAMPSHIRE

A total of 135 schools were surveyed in the state of New Hampshire. The following tables provide the results for the 26 schools responding. The response rate for this state was 19.25%.

Teaching Responsibilities	Percentages
% Teaching Programming	65.4%
% Teaching Applications	69.2%
% Teaching Other	46.2%
% Teaching Prog+Applications	42.3%
% Teaching Neither	7.7%

Current Hardware Use	Results
Average Number of PCs	19.2
Average Number of Macs	1.8
Average PC Lab Size	24.9
Average Mac Lab Size	12.0
Percentage with PC Lab	76.92%
Percentage with Mac Lab	15.38%

Intended Hardware Purchase	Percentage
Buying new machines	30.8%
Buying PC (if buying)	62.5%
Buying MAC (if buying)	37.5%

Hardware Purchase Criteria	Percentage
*Compatibility	19.2%
Cost	34.6%
Ease-of-Use	7.7%
Industry Standard	3.8%
Networkable	7.7%
Memory Capacity	11.5%
Reliability	38.5%
Speed	15.4%
Technical Support	3.8%

* Compatibility refers to compatibility with schools networks and with existing hardware.

Grade	Teaching Programming
Grade 10	61.5%
Grade 11	57.7%
Grade 12	57.7%

Language Taught	Grade 10	Grade 11	Grade 12
Basic	18.8%	6.7%	6.7%
C++	62.5%	73.3%	80.0%
HTML	18.8%	20.0%	26.7%
Java	12.5%	13.3%	13.3%
JavaScript	6.3%	6.7%	0.0%
Pascal	25.0%	26.7%	20.0%
Perl	6.3%	6.7%	6.7%
Visual Basic	31.3%	33.3%	33.3%
Visual C++	0.0%	0.0%	0.0%

A total of 19.2% of respondents reported that they were considering changing programming languages. The following table shows the percentages for the languages under consideration by those considering a change. Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.

New Language	Percent
C/C++	40.0%
Java	60.0%
Visual Basic	40.0%

Programming Language Criteria	Percentage
Advanced Placement Exam	15.4%
College Requirements	3.8%
Ease-of-Use	26.9%
Industrial Relevance	42.3%
Object Oriented	3.8%
Skills Transfer	3.8%
Structured	3.8%
Student Interest	3.8%
Teacher Knows It	3.8%

Teachers were asked to rank the importance/usefulness of a list of teaching resources. Here is their ranking from **most** (1) to **least** important (6).

Example Programs (2.3)
Textbook (2.5)
Teachers Guide (3.0)
Student Workbooks (3.1)
Project Collections (3.3)
Electronic Lessons (3.5)

Teachers were asked to rank the importance/usefulness of a list of potential sources of continued learning/skills upgrading. Here is their ranking from **most** to **least** important.

Self-directed Learning (2.2)
Conferences (2.4)
School/Board Inservice (3.1)
Upgrading Courses (3.1)
Internet (3.7)
Vendor Events (4.8)

APPENDIX E

NEW JERSEY

A total of 606 schools were surveyed in the state of New Hampshire. The following tables provide the results for the 85 schools responding. The response rate for this state was 14.02%.

Teaching Responsibilities	Percentages
% Teaching Programming	73.8%
% Teaching Applications	50.6%
% Teaching Other	49.4%
% Teaching Prog+Applications	31.0%
% Teaching Neither	6.0%

Current Hardware Use	Results
Average Number of PCs	41.0
Average Number of Macs	6.1
Average PC Lab Size	53.6
Average Mac Lab Size	31.9
Percentage with PC Lab	76.47%
Percentage with Mac Lab	19.05%

Intended Hardware Purchase	Percentage
Buying new machines	40.0%
Buying PC (if buying)	88.2%
Buying MAC (if buying)	17.6%

Hardware Purchase Criteria	Percentage
School Board Decision	2.4%
*Compatibility	14.1%
Cost	28.2%
Durability	7.1%
Industry Standard	3.5%
Networkable	2.4%
Manufacturer (Brand Name)	1.2%
Memory Capacity	12.9%
Reliability	9.4%
Speed	14.1%
Teacher Decision	1.2%
Technical Support	2.4%
Upgradeable	5.9%

Warranty	3.5%
----------	------

* Compatibility refers to compatibility with schools networks and with existing hardware.

Grade	Teaching Programming
Grade 10	56.5%
Grade 11	62.4%
Grade 12	69.4%

Language Taught	Grade 10	Grade 11	Grade 12
Basic	52.1%	18.9%	22.0%
C++	43.8%	77.4%	76.3%
HTML	2.1%	1.9%	8.5%
Java	10.4%	9.4%	16.9%
JavaScript	0.0%	0.0%	0.0%
Pascal	13.2%	11.3%	5.1%
Perl	0.0%	0.0%	0.0%
Visual Basic	16.7%	20.8%	22.0%
Visual C++	0.0%	1.9%	3.4%

A total of 16.5% of respondents reported that they were considering changing programming languages. The following table shows the percentages for the languages under consideration by those considering a change. Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.

New Language	Percent
C/C++	21.4%
Java	42.9%
Visual Basic	50.0%

Programming Language Criteria	Percentage
Advanced Placement Exam	23.5%
College Requirements	9.4%
Concepts	3.5%
Ease-of-Use	15.3%
Features	1.2%
Hardware Requirements	3.5%
Industrial Relevance	29.4%
Open Source	1.2%
Resources Available	1.2%
Skills Transfer	3.5%

Student Interest	2.4%
Teacher Knows It	2.4%
Variety	1.2%
Vendor (Brand Name)	1.2%

Teachers were asked to rank the importance/usefulness of a list of teaching resources. Here is their ranking from **most** (1) to **least** important (6) .

Example Programs (2.5)
 Textbook (2.5)
 Teachers Guide (2.6)
 Electronic Lessons (3.5)
 Student Workbooks (3.5)
 Project Collections (3.6)

Teachers were asked to rank the importance/usefulness of a list of potential sources of continued learning/skills upgrading. Here is their ranking from **most** to **least** important.

Conferences (2.3)
 Self-directed Learning (2.7)
 School/Board Inservice (3.3)
 Upgrading Courses (3.4)
 Internet (3.8)
 Vendor Events (4.5)

APPENDIX F

WASHINGTON

A total of 239 schools were surveyed in the state of New Hampshire. The following tables provide the results for the 41 schools responding. The response rate for this state was 17.15%.

Teaching Responsibilities	Percentages
% Teaching Programming	43.9%
% Teaching Applications	63.4%
% Teaching Other	43.9%
% Teaching Prog+Applications	19.5%
% Teaching Neither	12.2%

Current Hardware Use	Results
Average Number of PCs	19.0
Average Number of Macs	1.9
Average PC Lab Size	26.0
Average Mac Lab Size	13.2
Percentage with PC Lab	73.17%
Percentage with Mac Lab	14.63%

Intended Hardware Purchase	Percentage
Buying new machines	46.3%
Buying PC (if buying)	84.2%
Buying MAC (if buying)	15.8%

Hardware Purchase Criteria	Percentage
School Board Decision	4.9%
*Compatibility	14.6%
Cost	31.7%
Ease-of-Use	7.3%
Industry Standard	9.8%
Internet Capabilities	2.4%
Networkable	4.9%
Memory Capacity	19.5%
Reliability	17.1%
Speed	24.4%

* Compatibility refers to compatibility with schools networks and with existing hardware.

Grade	Teaching Programming
Grade 10	29.3%
Grade 11	36.6%
Grade 12	34.1%

Language Taught	Grade 10	Grade 11	Grade 12
Basic	0.0%	6.7%	0.0%
C++	50.0%	53.3%	64.3%
HTML	25.0%	20.0%	14.3%
Java	0.0%	13.3%	7.1%
JavaScript	8.3%	0.0%	0.0%
Pascal	8.3%	6.7%	7.1%
Perl	0.0%	0.0%	0.0%
Visual Basic	50.0%	40.0%	42.9%
Visual C++	0.0%	0.0%	7.1%

A total of 14.6% of respondents reported that they were considering changing programming languages. The following table shows the percentages for the languages under consideration by those considering a change. Note that many respondents report that they are considering more than one potential new programming language so total responses exceed 100%.

New Language	Percent
C/C++	16.7%
Java	33.3%
Visual Basic	33.3%
Other	16.7%

Programming Language Criteria	Percentage
Advanced Placement Exam	4.9%
Curriculum Guidelines	4.9%
Ease-of-Use	17.1%
Features	2.4%
Industrial Relevance	22.0%
Object Oriented	2.4%
Open Source	2.4%
Skills Transfer	2.4%
Student Interest	2.4%

Teachers were asked to rank the importance/usefulness of a list of teaching resources. Here is their ranking from **most** (1) to **least** important (6) .

Teachers Guide (1.9)
Example Programs (2.0)
Textbook (2.0)
Electronic Lessons (2.4)
Student Workbooks (2.6)
Project Collections (3.1)

Teachers were asked to rank the importance/usefulness of a list of potential sources of continued learning/skills upgrading. Here is their ranking from **most** to **least** important.

Conferences (2.4)
Self-directed Learning (2.7)
School/Board Inservice (2.8)
Upgrading Courses (3.6)
Internet (3.7)
Vendor Events (4.4)

References:

- Andreae, P.; Biddle, R.; Dobbie, G.; Gale, A.; Miller, L.; Tempero, E. (2000). Experience Teaching CS1 with Java. *Journal of Computer Science Education*, 14 (1&2), April, (pp.19-28).
- Baker, L.; Chapman, G.; Knoch, J.; Larson, K.; Walker, H. (1998). SIGCSE Panel: Approaches for Encouraging High School/College Faculty Interaction. In *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education* (pp. 368-369). Washington, DC: Association for Computing Machinery.
- Becker, H. J. (1994). *Analysis and Trends of School Use of New Information Technology*. Prepared for the U.S. Office of Technology Assessment. <http://www.gse.uci.edu/doehome/EdResources/Publications/EdTechUse/TEXTCHI1.HTM>.
- Biddle, R. and Tempero, E. (1999). Java Pitfalls for Beginners. *Journal of Computer Science Education*, 13 (3&4), November, (pp.8-13).
- Brilliant, S. and Wiseman, T. (1996). First Paradigm and Language Dilemma. In *Proceedings of the 27th SIGCSE Technical Symposium on Computer Science Education* (pp. 338-342). Washington, DC: Association for Computing Machinery.
- Burd, B.; Spies, W.; Wittenburg, L. and Workmann, R. (1997). Visual Programming Tools in the C.S. Curriculum. In *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education* (pp. 388-389). Washington, DC: Association for Computing Machinery.
- Carrasquel, Jacobo. (1999). Teaching CS1 On-line: The Good, the Bad, and the Ugly. In *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education* (pp. 121-216). Washington, DC: Association for Computing Machinery.
- Coley, R. J.; Cradlet, J.; and Engel, P. (1997). *Computers In Classrooms: The Status of Technology in U.S. Schools*. Educational Testing Service. Princeton, N. J.
- Culwin, F. (1999). Object Imperatives. In *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education* (pp. 31-36). Washington, DC: Association for Computing Machinery.
- Fisher, A.; Margolis, G.; and Miller, F. (1997). Undergraduate Women in Computer Science: Experience, Motivation and Culture. In *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education* (pp. 106-115). Washington, DC: Association for Computing Machinery.
- Heermann, R. W. (1991). Computer's in Virginia's Public High Schools. *Computers and Education*, 13 (1), (pp. 85-93).
- Hiltz, S. R. and Wellman, B. Asynchronous Learning Networks as a Virtual Classroom. In *Communications of the ACM*, 40(9), (pp. 44-49). Washington, DC: Association for Computing Machinery.
- Jarvinen, K.; Kyaruzi, J.; Sutinen, E. (1999). Between Tanzania and Finland: Learning Java Over the Web. In *Proceedings of the 30th SIGCSE Technical Symposium on Computer Science Education*, (pp. 217-221). Washington, DC: Association for Computing Machinery.
- Kavanagh, J. (1998). Java or Bust. *The Computer Bulletin*, IV (10), Part I, (pp. 28-30).

- Kolling, M., and Rosenberg, J. (1996). Blue - A Language for Teaching Object-Oriented Programming. In *Proceedings of the 27th SIGCSE Technical Symposium on Computer Science Education*, (pp. 190-194). Washington, DC: Association for Computing Machinery.
- Kushan, B. (1994). Effective Computer Science Teachers. *Journal of Computer Science Education*, 8 (4), Summer, (pp. 23-25).
- Lewis, J. (2000). Myths About Object-Orientation and Its Pedagogy. In *Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education*, (pp. 245-249). Washington, DC: Association for Computing Machinery.
- Luker, P. (1994). There's More to OOP than Syntax. *SIGSCE Bulletin*, 26 (1), (pp. 56-60). Washington, DC: Association for Computing Machinery.
- Martin, J. (1999). Teaching with Visual Basic. *Journal of Computer Science Education*, 13 (1), March, (pp. 12-15).
- Mazaitis, D. (1993). The Object-Oriented Paradigm in the Undergraduate Curriculum: A Survey of Implementation and Issues. *SIGCSE Bulletin*, 25 (3), (pp. 58-64). Washington, DC: Association for Computing Machinery.
- Merritt, S. (1995). Reflections of a Computer Scientist for Teachers and Teacher Educators. In Tinsley, J. D. and van Weert, T. J (Eds.) *Proceedings of the World Conference on Computers in Education VI*, (pp. 479-486). London: Chapman and Hall for the International Federation for Information Processing.
- Milbrandt, G. (1993). Using Problem Solving to Teach a Programming Language in Computer Studies. *Journal of Computer Science Education*, 8 (2), Winter, (pp. 14-19).
- Moylan, P. J. (1992). The Case Against C. *Technical Report EE240*. Centre for Industrial Control Science. Department of Electrical and Computer Engineering. The University of Newcastle, Australia.
- Proulx, V. (1995). Computer Science/Informatics: The Study of the Information World. In Tinsley, J. D. and van Weert, T. J (Eds.) *Proceedings of the World Conference on Computers in Education VI*, (pp. 495-503). London: Chapman and Hall for the International Federation for Information Processing.
- Schollmeyer, M. (1996). Computer Programming in High School VS College. In *Proceedings of the 26th SIGCSE Technical Symposium on Computer Science Education*, (pp. 378-382). Washington, DC: Association for Computing Machinery.
- Stager, G. (1995). Constructing Staff Development and Education Change. In Tinsley, J. D. and van Weert, T. J (Eds.) *Proceedings of the World Conference on Computers in Education VI*, (pp. 1079-1087). London: Chapman and Hall for the International Federation for Information Processing.
- Standing, C. (1999). Teaching Software Technologies and Programming with Java. *Journal of Computer Science Education*, 13 (2), July, (pp. 6-14).
- Stephenson, C and West, T. (1998) Language Choice and Key Concepts in Introductory Computer Science Courses. *Journal of Research on Computing in Education*, 31(1), (pp. 89-95). Eugene, OR: International Society for Technology in Education.

- Taylor, H. The Evolution of Standards for Accrediting Computer Science Teacher Preparation Programs. In *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education*, (pp. 67-71). Washington, DC: Association for Computing Machinery.
- Taylor, H and Mountfield, L. (1991). An Analysis of Success Factors in High School Computer Science: High School Methodology is a Key Element. *Journal of Research on Computing in Education*, 24 (2), (pp. 240-245). Eugene, OR: International Society for Technology in Education.
- Weiss, M. A. (1997). Experiences Teaching Data Structures with Java. In *Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education*, (pp. 164-168). Washington, DC: Association for Computing Machinery.