



Issues and Trends in High School Computing

Chicago, Illinois

June 24, 2001

The second annual computer science and information technology symposium was attended by some 71 participants. It was a highly successful and interesting symposium, with something for everyone. This article is a synopsis of the three general sessions and six breakout sessions that made up the symposium. Some of the individual presenters' materials are linked from the ISTE Web site. Find presentation materials online for Simon Phipps, Jill Jones, Chris Stephenson, Tim Corica, and Graham Smyth at <http://www.iste.org/news/events/symposia/computer-science>.

General Session:

Emerging Technologies and Career Opportunities

Subtitle: Standards, Swarms, and Synergies: Understanding the Net Effect and Expecting New Career Paths

Speaker: Simon Phipps (simon.phipps@sun.com), Chief Technology Evangelist for Sun Microsystems, Inc.

The Firm Foundation Standards

With regard to a network, the value rises as the square of the participants. It is about the number of relationships going on. Layers of opportunity are coming up. The cost rises as the *sum* of the participants. This is the *Net Effect*. Shared standards guarantee that the value grows fastest (Phipps' Law). The Net rejects non-subscribers to the standards.

The NET Effect in This New Century

In the '80s, standards were inspired by hardware problems—avoiding them. In the '90s, standards were inspired by business; consortia formed the standards. There were only four acceptable consortia, and these were membership organizations. Although these groups were closed, people were happy to have the standards. In the 21st century, standards are community inspired. Look at Linux (e.g., using open source process). This is not anything like a standards process.

A *standard* is a technology that when it changes you are not surprised, even if the group it comes from seems all wrong.

Open source refers to the license that comes with the code. (All software comes with a license, even if free). It's a change of philosophy—anyone can borrow some source code, change it, and return it. There is some respected gatekeeper to keep it from chaos, and then there are opportunities above all that for someone to sell it. The *foundation* is determined by the community. The skills you need for such work are social, skills to work with a community.

The Shape of Tomorrow—Swarms

The trend in software is toward dis-integration and specialization. *A System*: The network is *the* computer now. *System Software*: There is a cloud of distributed network services (e.g., looking up of network addresses). *Application* disintegration is also taking place. Commercial services are of all different sorts, making possible, for example, the outsourcing of different pieces of the payroll.

Layers of the Net Effect: The Web is now 10 years old, and the nature of technology has completely changed:

- *On the Web*: HTML, Web server.
- *Consumer e-Commerce*: HTML, Server, Java. Everybody was empowered to spend money (Mastercard)! That was the key dynamic. Peer-to-peer computing is one of the big trends—doesn't go through the server (e.g., Napster, instant messaging).

- *Business-to-business*: HTML, Web Server, Java, XML; shared XML vocabularies. Data are transformed from this company to that, over language and culture barriers.
- *Consumer access*: HTML, servers, Java, XML, vocabularies. These transform service location and connection. Wireless is the catalyst and game changer. PCs are a shrinking percentage. You will have no idea what's on the end of the non-wire.
- *The hive mind*: Spontaneous federation. When you get in your car all your devices will check with each other and keep you informed—best route, cheapest gas, where to eat, whatever. Smart Web Services. Anyone, any time, anywhere, any device: they are always on.

Steps Toward Tomorrow—Synergy

1. Component based software—portable byte code, different pieces owned by different players: Java, XML
2. Companies will be in conversation all the time. Company partnerships are key; the marketplace is a conversation.
3. Communities: decisions are made by massively connected communities. People will find themselves increasingly in standards communities.

Summary

The network is the computer.
The network is a service network.
Community interaction is key.

There are two crucial technologies: open processing (Java) and open data (XML). These are not leading-edge skills. *They are current core skills that everyone needs to have.*

Resource

Presentation slides: <http://www.shipps.com/simon>

Breakout Session:

Professional Certification

Subtitle: Frills or Foundations? Certification and Curriculum Development

Speaker: Jay Wood (wood@admin.humberc.on.ca), Senior Program Coordinator in the School of Information Technology at Humber College of Applied Arts and Technology in Toronto, Ontario, Canada

Jay Wood described how Humber College has cooperated with Microsoft to allow the School of Information Technology to issue a Microsoft certificate. Most major software companies offer some variety of certificate (e.g., Novell, Cisco, Oracle, IBM, Sun, Nortel, and Microsoft).

The main reason for a school to ask industry's cooperation is lack of funding. Software is expensive to acquire and requires an enormous amount of technical support. With industry support, school systems can get the latest software at unbeatable prices and also receive support services. This kind of cooperation also brings about public recognition and thus attracts incoming students. However, that reputation is just like two edges of a sword: it could help and also hurt you.

When negotiating with big corporations, schools need to pay attention to what is best for their students. We need to incorporate those certificate requirements into curricula, not just teach one specific program or software. We must prepare students in both breadth of knowledge and depth of understanding. Breadth will provide the ability to transfer knowledge to a different domain. Depth will give students problem solving skills. But it is very difficult and yet very important to find balance of these two.

There are pros and cons to this idea of a certificate. On the positive side are:

- Standards-based knowledge
- Certified people are said to be 10%–40% more productive (However, these data are provided by vendors)
- Credentials bring credibility
- Certificates pay: research shows that certified people get paid 10%–40% more

On the negative side are these factors:

- Very variable standard
- Very temporary life and product-specific. Software is updated every 2–3 years
- Very costly to own it: The average certificate costs \$5,000 to \$15,000.

Ontario has completely overhauled the K–12 curriculum into three streams: academic, applied, and workplace. All are foundation-based. Writers who developed this new curriculum came from all sectors of the educational system. At Humber College, we can see the power of the certification at the institutional level, but philosophically it was a hard sell.

The future of certification will ultimately be decided by money and by competition. For business and industry, education is a profit center; software and hardware products can shift very quickly. For education, IT standards will continue to evolve at a dramatically fast rate. There will be no substitute for a solid foundation—in the long run, learning is a lifelong process.

Resource

Electronic Experimenter Kit for an Activity Based Curriculum: <http://www.classictech.on.ca>

Breakout Session:

Ethics—The Dilemma of Technology

Speakers: Jill Jones (jjones@hayden.edu), Diane Braden, and Chris Walker, Center for Computer Studies, Carl Hayden Community High School, Phoenix Arizona. Jill is a computer science teacher. Diane teaches multimedia and electronic publishing, and Chris teaches desktop publishing and serves as program manager for the Center for Computer Studies.

The speakers began this interactive session by handing out a page called “The Commandments of Computer Ethics” (<http://www.cpsr.org/program/ethics/cei.html>) by the Computer Ethics Institute, and showing a short video on these commandments made by their high school students. They began their discussion by proposing to raise questions that teachers struggle with every day. Their main question was: How do we approach the dilemmas and set up procedures that people can follow to deal with these dilemmas?

Carl Hayden Community High School was described as a typical public school, in Phoenix, with a low socio-economic status and a high minority population. Six hundred students are enrolled in a four-year comprehensive magnet program, the Center for Computer Studies.

The presenters gave some quotes from freshmen about what computer ethics are and concluded from these that freshmen basically have no idea what they are. A quick review of the ethical problems they have had to deal with in their short history with computers reveals that the first computer problem (back in the days of no network, mouse, or hard drives) was gum in the disk drive. Next came cheating and using a home computer to bill long distance telephone calls to someone else. About 10–12 years ago the first virus to shut down a university appeared. Everything was on a floppy and was easy to copy.

The next years at Hayden brought mice, small disks, and the Internet. Mice balls and data disks were stolen; cheating became more sophisticated; equipment was mistreated. Other problems flourished in society at large during this period: pornography, fake ATMs in malls, altered photos, and hackers.

Currently: there are no floppies, one printer per room, scanners, digital cameras for each lab, and high-end software programs. Increasingly these problems occur in the classroom:

- programs deleted from the hard drive
- inappropriate screen savers
- viewing pornography
- altering a photograph for a poster
- equipment stolen or destroyed
- student work deleted when not logged out
- personal copies on printer
- cheating
- copyright violations
- Internet/e-mail/chatrooms under fire because of pedophiles
- identity theft
- fraud on Internet
- hackers
- cybercrimes
- misuse of e-mail (poor netiquette, personal attacks, junk mail, flaming)
- security problems
- stolen passwords

- writing virus programs
- students creating inappropriate Web sites

Ethics no longer apply to adults only!

What To Do?

- restrict Internet use
- supervision
- safe e-mail
- filters
- class discussions
- classroom policy or contract

The presenters focused on last two.

Class Discussion on 10 Commandments

After a unit on the 10 commandments, the presenters hope to have increased awareness among students. The interactive session at this point involved the audience in such a small-group discussion. Each group took a commandment and discussed and presented their experience with the problem and a possible solution. Find PDF documents on ISTE's Web site:
<http://www.iste.org/news/events/symposia/computer-science>

Policy

Why have a policy?

- accountability for student and for teacher
- liability
- changing landscape of technology

What use would it serve?

- make us aware of the issues
- set boundaries on behavior
- common set of expectations

When creating policies, look at policy documents that already exist. Determine what is important for your situation. Discuss your policy with administrators and get their support. Align consequences with other school policies and rules.

Internet policy should:

- protect students and parents
- stress personal responsibility
- be against filtering
- provide guidance
- outline consequences

People don't realize the seriousness of the consequences yet.

Resources

10 commandments and AUPs: <http://www.davison.k12.mi.us/hahn/webinc/ethics.htm>

Center for Computing and Social Responsibility: <http://www.ccsr.cse.dmu.ac.uk/>

Computer Ethics Institute: http://www.brook.edu/ITS/CEI/CEI_HP.HTM

Computer Professionals for Social Responsibility: <http://www.cpsr.org>

Cyber Citizen Partnership: <http://www.cybercitizenpartners.org>

MainFunction: <http://www.mainfunction.com/realityCheck/>

National Infrastructure Protection Center: <http://www.nipc.gov>

Paper on computer ethics and social aspects of computing:

<http://www.ifl.uio.no/iris20/proceedings/12.htm>

Research Center on Computing and Society: <http://www.couthernct.edu/organizations/rccs/>

ThinkQuest: library.thinkquest.org/26658//?tqskip=1

General Session:

Equity

Subtitle: Improving Math & Science Education

Speaker: Jacelyn Swenson (jacelyn@us.ibm.com), Program Manager, Women in Technology/IBM

Jacelyn Swenson began with reference to a few recent *Time* and *Newsweek* lead articles: untipping the Leaning Tower of Pisa, robotic surgery, a smart shirt that will beam urgent messages to your doctor, and smart bandages. She claims that more fascinating possibilities like these are going to happen, and that our kids are well poised to go after the technology careers implied by these stories.

Swenson says that today's market has 400,000 tech jobs in the United States that we can't fill. Potential talent is not getting into the pipeline leading to such jobs. The number of college freshman women interested in science and engineering has declined during the past decade.

Look at the differences between females and males in these fields who actually graduate with the major they chose:

- Engineering—30% F vs. 80% M graduation rates.
- Biology—20% F vs. 40% M graduate as biology majors.
- Physics—33% F vs. 99% M stay with this major.

The Digital Divide is widening.

By 2006 there will be a need for 1 million new professionals in IT. Today:

- 50% of the workforce is made up of women,
- 40% of businesses are owned by women, and
- personal digital assistants (PDAs) and pagers are not designed with women in mind (most are designed to hang on a belt, but most women's business attire does not include a belt!).

How did we get into this situation in which women are not in the math/science/technology pipeline? They don't leave because of poor grades, but because of academic dissatisfaction and social factors:

- mass media portrayals
- social pressures
- materials—skill vs. kill
- influences discourage pursuit of these fields

Companies are asking themselves how they can attract and retain women in IT professions, and how they can help get girls into the pipeline to begin with. Below are some of the issues, possible solutions, and initiatives.

The challenge:

- neutral software
- encouragement
- role models
- girl-specific programs, resources
- reach to disadvantaged communities

The long-term solution (from the federal government):

- Stimulate student interest in math and science.
- Set strong curricula and high expectations.
- Produce teachers who have a strong interest, skill, and high expectations and can encourage and retain students.

An interim solution that the New Jersey Institute of Technology is working on:

Don't perpetuate the belief that young women won't be interested. *Do* ask the young women to explore the profession to see if it fits their talents:

- What do you care about?
- What do you like to do for fun?
- What interests and talents would you like to pursue?

IBM initiatives:

- informal programs to students—fun and interesting
- national engineering week
- introduce a girl to engineering day
- workshops, classroom visits
- institutionalized programs for women
- EXITE! camps: 21 camps worldwide, 600 girls, virtual and collaborative, build Web sites and computers, program a robot, construct towers, make liquid nitrogen ice cream.
- 600 e-Mentors

Building the Foundation

MentorNet is a national electronic mentoring project for women who are studying engineering and science. Women mentors in the industry are put in touch with college-age women interested in these careers. This is a personal, informal, and cost-effective way to support college women.

Camps, workshops, e-Mentors for college women: the college-age mentee becomes a mentor to precollege women.

What You Can Do

Teach: Reassess your software; integrate computers into all subject areas; access content-rich sites (e.g. science.org).

Reach: Engage girls in tinkering activities. Hands-on activities are meaningful.

Coach: Bash the stereotype (solitary, antisocial); find and nurture role models. When they come to you, keep them!

Resources

A Girl's World Online: <http://www.agirlsworld.com>

Expect the Best from a Girl: <http://www.academic.org>

Girl Scouts USA: Just 4 Girls <http://www.gsusa.org/girls>

Girls, Inc.: <http://www.girlsinc.org>

Girlstart: <http://www.girlstart.org>

Independent Means: <http://www.independentmeans.com>

New Moon: <http://www.newmoon.org>

Role Model Project for Girls: <http://www.womenswork.org/girls>
Scitechmatics: <http://www.can.ibm.com/k12/scitechmatics>
TAP Junior: <http://www.cs.yale.edu/homes/~tap/tap-junior.html>
The Engineer Girl: <http://www.nae.edu/nae/cwe/egmain.nsf?OpenDatabase>
Tryscience: <http://www.tryscience.org>
The Backyard Projects: <http://www.backyard.org>
Women of NASA: <http://quest.arc.nasa.gov/women/intro.html>

Breakout Session:

AP with Java

Subtitle: APCS: Plus ça change, plus c'est la même chose

Speaker: Tim Corica (tcorica@peddie.org), Director of Academic Technology and mathematics teacher at The Peddie School, a private boarding school in Hightstown, New Jersey. Corica has extensive experience with the APCS exams, both in development and reading.

Presentation and accompanying files: <http://www.peddie.org/local/Corica2001.zip> and <http://www.peddie.org/local/Corica2001.ppt>

APCS Curriculum & Exams

Corica began his talk by describing the current Advanced Placement exam situation in computer science:

- Level A—equivalent of the first semester; fundamental programming concepts and skills
- Level AB—adds to the fundamentals data structures, algorithm analysis, class construction—may be equivalent of two semesters of CS.
- 24,000 AP exams were given in CS this year.
- Current language is C++.

Languages & APCS

The APCS exam was based on Pascal for 14 years, and C++ for the years 1999–2003, but starting in 2004, Java will be the language of choice.

This decision was made by convening an ad hoc committee of college and high school instructors, chaired by Henry Walker (Samuel R. and Marie-Louise Rosenthal Professor of Natural Science and Mathematics, Department of Mathematics and Computer Science, Grinnell College, Iowa; Chair of SIGCSE, ACM's Special Interest Group on Computer Science Education), to discuss what they believe is important, the future of the AP, and more. Not all of the college people were happy about it.

Now in 2001 the College Board has announced that we are going to change in 2004; this is about as fast as the AP can change.

It's not about programming languages; it's about the programming concepts. According to Corica, the real issues are not about which language to teach, but about the programming concepts that students need to learn. Some historical events and developments highlight such important concepts:

- Dijkstra's 1968 letter: structured programming
- 1977 Jensen/Wirth Pascal: procedural abstraction
- 1980 Wirth's Modula-2/Ada: data abstraction
- 1985 Stroustrup's C++ makes object-oriented programming (OOP) possible
- 1994 Sun's Java nearly legacy-free, fully OOP, and meshes with the Web explosion

According to Brooks (1995), programming tools are improved nearly to their maximum. The remaining software productivity problem is the task of conceptualizing complex systems. After 20 years, OOP may be the closest to a silver bullet addressing this issue.

What is OOP?

Corica demonstrated how to think about OOP by showing a screen shot of a game of solitaire. There are objects—things—that are on the screen. That's the first thing you do in OOP is identify objects. According to Alan Kay (as cited in Eckel, 2000), there are five basic principles of OOP:

1. Everything is an object.
2. A program is a group of objects telling each other what to do by sending messages.
3. Each object has its own memory made up of other objects.
4. Every object has a type.
5. All objects of a particular type (class) can receive the same messages.

How to get ready for Java without actually teaching it yet:

- Make intelligent use of classes/objects to construct programs.
- Look carefully at the two AP case studies:
 - LargeInt: two hierarchical classes
 - Marine Biology: 6–8 interlocking classes
- Use Visual Basic as a starting point (the first thing you do is build objects)
- Model/View approach
- Event-driven OOP models

Colleges have found that the transition from Pascal to C++ is very difficult, but C++ to Java is not so bad. Corica thinks the switch to Java is a good move:

- Syntax is nearly identical. Java is inherently more OOP than C++.
- Java supports strings, range-checked arrays, without adding special libraries.
- There is a clear and defined standard.
- Once again, Java is a College Board subset.

The College Board defines what will be tested, but not what can or should be taught. There is no I/O of any kind (file or console or GUI). There are specific things that students do have to be able to do. Inheritance and the expanding role of OOP are topics that will be added into Java by the AP.

Corica wrote a Java and a C++ version of an answer to an AP question (see Web site for problem) to show that the versions are nearly identical, and that it would be fairly straightforward to use C++ to get ready for teaching Java.

The Model/View Approach to Teaching Java

Corica admits that he is not certain how to teach Java, but recommends the Model/View approach:

- *Model*: class to house the data
- *View*: class to display the data
- *Controller*: class to interpret user input

In early coursework, rather than fall back on non-OOP, the teacher can write the view, and students can write the model.

What Can You Do Now?

- Add more OOP to your present courses.
- Experiment—or get kids to experiment—with Java.

- Put some Java into your courses; it mixes well with C++.
- Keep an eye out for opportunities for training.

Resources

College Board APCS: <http://www.collegeboard.org/ap/computer-science/>
College Board Java page: http://www.collegeboard.org/ap/computer-science/html/java_announcement.html

Free/low-cost Java systems:

Ready to Program: <http://www.hsa.com>

JCreator: <http://www.jcreator.com>

BlueJ: <http://www.bluej.org>

Sun JDK: java.sun.com (combined with TextPad editor: <http://www.textpad.com>)

Higher-end commercial Java systems

CodeWarrior (Metrowerks, for Mac and PC)

JBuilder (Borland)

Visual Studio (MS)

VisualAge for Java (IBM)

References

- Brooks, F. (1995). *Mythical man-month: Essays on software engineering*. Anniversary edition. Reading, MA: Addison-Wesley.
- Eckel, B. (2000). *Thinking in Java*. (2nd Ed.) Upper Saddle River, NJ: Prentice Hall. [Online] <http://www.eckelobjects.com>.
- Lazere, C. A. & Shasha, D. E. (1998). *Out of their minds: The lives and discoveries of 15 great computer scientists*. New York: Copernicus Books.

Breakout Session:

Getting to the Bottom of Computer Hardware: Creating Chips, Gates & Interfaces

Speaker: Graham Smyth (smyth1@mnsi.net), 32-year veteran of teaching computer science, former head of Computer Science Department with the Lambton-Kent District Board of Education in Ontario, Canada.

This was a hands-on session. Participants actually constructed some type of hardware. Smyth and his associates wanted to show how easy it is to understand the basics of computer hardware. The PowerPoint presentation, which has pictures of the hardware used and which accompanied the hands-on session, is available at: <http://www.iste.org/news/events/symposia/computer-science>

The session began with the simple explanation of the interfacing system: the computer, the interface (wires, resistors, etc.), and peripherals (LEDs, robots, etc.). Almost everyone in the audience had a set of equipment that contained a portable PC, toolbox, and circuit board.

Smyth explained the structure of a parallel port, which we know as a printer port. It consists of two types of pins: output pins and ground pins. Later on, he explained the D sub connector, which is a regular connector for the printer. In this session, he used the programming language Turing, which has an easy-to-learn syntax and is supported by a student-friendly programming environment. This language was chosen because it facilitates hands-on learning by providing direct software access to the parallel port. It also provides easy access to graphics and mouse control. Turing also provides students with a conceptual understanding of programming that is highly transferable to other programming languages.

The activity for the audience was to build a one-bit interface. Also, participants were asked to write the program to turn the LED on and off. After seeing the result, everyone was surprised at themselves: they had actually built a simple computer hardware device. It was a really good way to show the audience how easy and understandable hardware can be.

Resource

Classic Technology: <http://www.classictech.on.ca>.

Breakout Session:

Knowing What to Teach— Computing in High Schools Panel

Panelists: Chris Stephenson (chris@hsa.on.ca), is President of Holt Software; a Research Associate at University of Waterloo, Waterloo, Ontario, Canada; and symposium chair. Philip East (east@cs.uni.edu), is Associate Professor of computer science education at the University of Northern Iowa.

First Speaker: Chris Stephenson

Topic: Curriculum is Just the Beginning

Stephenson set the scene for the discussion of knowing what to teach by asking some big questions:

- Who sets the vision?
- Is it an inside or outside job?
- Is there anything more than paper?
- A new curriculum but the same old people?
- New people but the same old resources?
- The same people with the same frustrations?

The vision of where high school computing is headed is often created outside the school and watered down by the time it gets to you, the teacher. The Ontario curriculum, according to Stephenson, was designed in a way that maximized the potential for it to be implemented effectively in the classroom. A clear choice was made to create a real CS stream that offered some new opportunities, was more rigorous, and was designed specifically for Grades 9–12. This curriculum was developed by representatives from high school, university, technical college, and industry.

The curriculum has two strands, a traditional Computer and Information Science (CIS) strand is offered in Grades 10–12. It focuses on programming with a major emphasis on software design concepts. The new Computer Engineering (CE) strand is also offered in Grades 10–12. It focuses on hardware and interfaces, with more emphasis on computer logic. Both strands contain expectations relating to computer networking.

Computer and Information Science Strand

These courses contain sets of expectations (what the student must know or be able to do) relating to programming concepts, logic, design, hardware, interfaces, and networking. There is also a major section (about 20% of the curriculum) that deals with careers and social issues relating to the impact of technology. Ontario is starting to introduce object-oriented programming in Grade 12, along with code re-use, software maintenance, and algorithm comparison.

Computer Engineering Strand

These courses contain sets of expectations (what the student must know or be able to do) relating to computer hardware (internal and external), peripherals, gates and chips, computer logic (including Boolean algebra), and computer networks. There is some programming content, but it is related primarily to writing programs to control peripherals using the computer. Another 20% deals with careers and social issues relating to the impact of technology.

Choice of Programming Language

While Turing is now their standard for Grades 10–11 and Java for Grade 12, Ontario does not mandate that teachers use a particular programming language. It does, however, provide a set of criteria for choosing a suitable programming language:

- it must meet all of the programming expectations;
- it must be grade-level appropriate;
- you must consider whether it can be used in both CIS and CE;
- you must consider availability of curriculum/technical support resources.

Assessment

As in many states, assessment has become something of an obsession for the educational policy makers in Ontario. The downside is that there appears to be more of an emphasis on what and how to assess than on what is in the curriculum and how to teach it. The good news, however, is that there has been a real improvement in using assessment to help students understand how they can do better.

Teachers are now being trained to use three distinct kinds of assessment:

- diagnostic assessment—pretesting
- formative assessment—ongoing, continuous feedback for improvement
- summative assessment—determines student achievement at the end.

They are also being encouraged to take advantage of great opportunities for performance-based assessment: programs, documentation, presentations, peer teaching, portfolios, skill demonstrations, and debates.

Achievement charts and rubrics are used to establish performance standards. Ontario is now also developing exemplars to show students specific examples of actual student work that has been evaluated at four different levels, which roughly correspond to blocks of grades or averages (for example, 50%, 60%, 70%, and 80% and above). Assessment and evaluation is based upon demonstrations of work over time and not on student behavior.

Next Big Steps

It will be important to create time for teachers to get together and learn together, and to put money into curriculum development and teacher preparation. New legislation has also been passed in Ontario that requires mandatory year-to-year updating of skills and professional development in order for teachers to maintain their teaching certificate. Unfortunately, as of yet there is no plan of how, where, who will pay, or whose time it will come out of.

Resources

Curriculum policy documents: <http://www.edu.gov.on.ca/eng/document/curricul/curricul.html>
(See Technological Education)

Association for Computer Studies Educators: <http://www.acse.net>

Second Speaker: Philip East

Topic: The High School Computing Course

East focused on a single high school course for those schools that could not have a sequence of courses. East believes that it is important that all students leave high school understanding computing, which he believes is second in frequency and importance of use only to reading and writing.

The design of a single course should be based on the fluency report: *Being Fluent with Information Technology* (Committee on Information Technology Literacy, 1999). This report suggests that we focus on the enduring concepts of computing, skills, intellectual capabilities, and project-based learning. The report listed 10 topics under each of these headings:

- how computers work
- information systems
- how networks work
- digital representation of data, information
- information storage, retrieval
- problem representation, abstraction
- algorithmic thinking, programming
- universality
- limits of computing
- social impact (the technical basis for social concerns)

East believes that information systems can and should be covered under the social impact concept. He also believes that universality should be discussed along with the limitations of IT.

The 10 intellectual capabilities:

- sustained reasoning
- management of complexity
- testing a solution
- management of faulty solutions
- finding/using information
- collaboration
- communication with other audiences
- expecting the unexpected
- anticipation of new technologies
- thinking about IT abstractly

The 10 skills:

- setting up a computer
- basic OS tasks
- text documents
- slides, images
- connecting to network
- finding resources on Internet
- communication
- spreadsheets
- database
- use computer-based instruction, documentation

East says that this list will no doubt change. For example, setting up a computer now requires connecting to a network; it did not 10 years ago.

Project-based learning

This type of learning involves the production of some type of artifact. Such a construction provides the opportunity to apply skills and practice intellectual capabilities, as well as to demonstrate understanding.

Programming

In the *Fluency Report*, programming is included in the 10 concepts. It involves specifying instructions precisely and "primitively" for some agent to carry out. To learn these concepts, traditional programming languages are not required. For example, a spreadsheet or HTML might be a reasonable substitute. East believes that designing significant projects is the important issue, not traditional programming.

Planning the Course

Skill activities: The goal is to have students learn independently using modern information access and learning aids.

Knowledge activities: East uses a textbook, and during class time, provides insights and connections to help students understand the material in the book.

Projects: Students examine some of the big ideas. East plans four big projects during a semester, usually two individual and two group projects. Students put their projects on the Web.

Collaboration: This is difficult to manage but important to do.

Assessing student learning is always difficult. It involves trying to answer these two questions:

- Did you do it?
- How well did you do it?

Quizzes help with some of the assessment, allowing East to focus on evaluation of the larger projects. He is moving toward:

- having students do more public display of projects,
- self-assessment, and
- talking with him about student learning.

Conclusions

1. No single bit of knowledge is critical.
2. Trust the process.
3. Intellectual capabilities are enduring; skills are not.
4. You need to be the one to pick the skills, concepts, and intellectual capabilities that you want to include in such a course.

Resources

Philip East (not complete or perfected): <http://www.cns.uni.edu/~east/teaching/021>

Larry Snyder (one of the *Fluency Report* authors):

<http://www.cs.washington.edu/education/courses/100/CurrentQtr/>

Reference

Committee on Information Technology Literacy (1999). *Being fluent with information technology*.

Washington, DC: National Research Council. [Online] <http://www.nap.edu/catalog/6482.html>.

Group Discussion

The following comments were captured during the group discussion:

- The expectation of being magically able to do collaboration is not likely to be fulfilled. [We] need to teach them how to do it.
- Problem of CS not being a real course (like science, history, math).
- Even some computer scientists don't believe their field is important to everybody.
- East thinks it really is as important as physics.
- CS is the discipline with no home. Business dept? math dept? broadbased technology/shop?
- Chris Stephenson has a *Careers in Computers* book that is really helpful. The book helps to counteract the stereotype. There are jobs. This gets to the parents.
- Computer entrance requirements would be the single greatest thing universities could do to help the high schools along. But, if we do it, and other universities don't, then we will lose students. Unless there is some help for administrators making sensible decisions, there will be no such decisions. Stephenson noted that if industry is saying we can't hire people the Government will listen to industry. Lillian Israel (Director of Membership, Association for Computing Machinery, <http://www.acm.org>) can step into a place like this. Can help get both higher education and industry behind it.

Breakout Session:

Identifying Web Resources for Teaching Computer Science

Speakers: Douglas Peterson (doug_peterson@gecdsb.on.ca) and Harry Groves. Doug is a Computers in Education Program Coordinator for the Greater Essex County District School Board, Windsor, Ontario, Canada. Harry is a Computer Education Consultant in Ontario, Canada.

Often CS teachers do not have supplementary materials, especially contemporary materials that are applicable in today's real world. Peterson and Groves believe the CS teacher needs to be on the lookout for such materials, and the Internet is a great source for information about current issues, computer programming contests, and supplementary tutorials and resources.

The presenters showed the audience the many links they have found and put together in a Web site. Readers should go directly to the Web site (see Resources) to check out the many links. The main menu includes these items:

- What is CS?
- The Basics
- Issues
- Current events
- Computer science contests
- American Computer Science League (also has contests)
- Intel (Journey Inside the Computer materials)
- Scavenger hunt
- Various online resources, courses, and tutorials
- CNET's Web Builder, news, and how-to's
- Three Dead Trolls alternative/underground news
- Advanced placement
- WebQuests on CS (a link to a NECC presentation on CS WebQuests)

Resources

Peterson & Groves: <http://www.gecdsb.on.ca/d&g/NECCMining/>

Chilly Beach: <http://www.chillybeach.com/games/quiz.html>

Dead Troll: <http://www.deadtroll.com>

<http://webservices.adobe.com/optimize/main.html>

Department of Control Engineering and Information Technology (Budapest, Hungary):

<http://www.inf.bme.hu/contests/tasks>

University of Maryland, Department of Computer Science:

<http://www.cs.umd.edu/Outreach/hscontest00/>

General Session:

Preparing Your Students with the Best Skills for Their Future

Speaker: Todd Knowlton (tknowlton@smoothfusion.com), President, Internet Business Consultant, SmoothFusion Digital Media Group (substituting for Janie Schwark, Microsoft Corp.)

What is IT?—the study, design, development, implementation, support or management of computer-based information systems, particularly software applications and computer hardware.

What is an IT worker?—programmer/software developer, database administrator, Web administrator, network systems specialist, etc.

Why do we care about IT?—There is a worldwide shortage of IT workers.

Changes in Demands

2000: There was a demand for 1.6 million IT workers, and a gap (between demand and availability) of 843,000.

2001: This year there is a demand for .9 million, and the gap is 425,000. This gap is fairly significant. If we look at a breakdown of the gap, which represents an unfilled need this year, we find that the top three categories, which are particularly in need, are:

- tech support,
- network design/administration, and
- software engineering.

The next three are:

- Web administration/development,
- database development/administration, and
- enterprise systems.

By region, the highest to lowest gaps are found in the Midwest, the West, the South, and finally the Northeast.

In IT companies the top three areas of job openings are:

- software engineer,
- database developer/administrator, and
- Web developer/administrator.

In Non-IT companies the top three areas of job openings are:

- enterprise,
- software engineer, and
- tech support.

Skills needed by these job categories:

Tech Support: troubleshooting; facilitation/customer service; hardware/software installation, configuration, upgrades; system operations, monitoring, and maintenance

Database Developer/Administrator: analyze needs and design database; develop and implement database; perform administration and management; security administration; client services

Software Engineer: perform needs analysis; develop structure; design and develop a program; implement a program; test and validate programs; release programs

Web Developer/Administrator: content and technical analysis; develop Web applications, sites; implement a Web application or site design; maintain applications; manage a Web environment; manage enterprise-wide Web activities

Network Designer/Administrator: design and analysis; configuration/implementation; testing; security; monitoring and management; administration/management

Technical Writer: analyze project requirements; perform research; design documents; develop and write documents; publish and package

Enterprise Systems Integration: define customer requirements; determine system solutions; configuration and interoperability; provide high-level technology management; implement enterprise-wide systems

Digital Media: perform needs analysis; produce visual and functional design; media production and acquisition; implement and test design

It is important as you think about the skills your students will need that you don't forget the "soft skills." These are communication, analytical and business skills, and interpersonal skills.

Resources

MainFunction: <http://www.mainfunction.com> (a curriculum resource from Microsoft for teachers and students of computer programming and the Internet.

MSDN Academic Alliance: <http://www.msdnaa.net> (makes available low-cost technology tools for computer science instructors for an annual subscription fee)

Acknowledgments

The Symposium Committee wishes to acknowledge Misook Ji, Alex Hwu, and Daisy Huang for their efforts in documenting the Symposium sessions and Dr. Diane McGrath for her editorial assistance with this document. Dr. McGrath (dmcgrath@ksu.edu) is an Associate Professor of Educational Computing, Design & Telecommunications at Kansas State University, and a longtime member of SIGCS and former editor of its *Journal of Computer Science Education*; she also is a former editor of ISTE's *Journal of Research on Computing in Education*. Misook Ji, a former math and computer science teacher, has just received her doctorate from Kansas State University and accepted a position in educational technology at the University of San Diego. Alex Hwu and Daisy Huang are doctoral students in Educational Computing, Design & Telecommunications at Kansas State University.



This synopsis is a publication of ISTE's Special Interest Group for Computer Science (SIGCS). ISTE members have permission to photocopy this document for educational use. Non-members may contact the Copyright Clearance Center, 978.750.8400 (voice), 978.750.4470 (fax), www.copyright.com for permission to reproduce this document. The copyright statement below must appear on all copies.

Issues and Trends in High School Computing: The Computer Science & Information Technology Symposium 2001, Edited by Diane McGrath
Copyright © 2001, ISTE (International Society for Technology in Education),
800.336.5191 (U.S. & Canada) or 541.302.3777 (Int'l),
iste@iste.org, www.iste.org.

8-15-2001