Appendix:
Breakout Sessions from
The Future of Computing Education Summit
June 25-26, 2009
Contents

Defining Computing Challenges Session ................................................................. 3
Revising the Curriculum Challenges Session .......................................................... 7
Professional Development and Lifelong Learning Challenges Session .............. 11
Creating More Attractive Pedagogy Challenges Session ................................ 16
Promoting Computing Challenges Session ........................................................... 20
Research and Measurement Strategy Session ..................................................... 23
University/Industry Partnership Strategy Session .............................................. 28
Cross-Computing Strategy Session ................................................................. 31
Public Relations Strategy Session ....................................................................... 34
Changing the University Strategy Session ....................................................... 37
Changing the University Action Items Session ............................................... 41
Community Collaboration and Policy Action Items Session .......................... 44
Public Relations Action Items Session ............................................................ 48
Defining Computing Challenges Session

Transcription: Defining Computing.doc
Participants: Alan Clements, Stephanie Hoeppner, Harriet Taylor, Heikki Topi.

Opinion from IEEE (Alan Clements): It’s important to define computing, because people in the IEEE Computing Society take a particular view of what computing is. However, computing is changing a lot and many of the more traditional organizations don’t appreciate that. Today, computing is being driven by games, graphics, visualization and so on. Yet a lot of the major societies are thinking of computing as large-scale aerospace-style projects, software engineering and so on. Recruitment at my own university is seeing a massive decline in traditional computing but an increase in the new computing like games and visualization. Of course, there is still the problem of needing people in conventional computing to run the banks and the networks and so on.

Harriet Taylor asks if computing always means there is a piece of technology involved, or are there some aspects that are not necessarily linked to technology, but may be fundamental to the use of a lot of technology? Heikki Topi believes this is an important question, and it is important not to define computing too narrowly, or we only provide a narrow window to a fascinating field.

This brings up the distinction between computing and computer science. Stephanie Hoeppner mentions that in high school, some students are good with programming, and others are good at multimedia. But corporations and colleges do not value these equally, and this confuses students. Within science, students have a choice of biology, chemistry, physics and other disciplines, but high school students don’t have the same sense when it comes to computing.

Computing education is largely PC-based, but most computers we encounter are in phones, automobiles etc. Both Britain and the US have an antiquated approach to thinking about technology in this way.

Are there any core issues common to all computing disciplines? Many say problem solving is. Unfortunately, students are choosing not to take subjects that they perceive as difficult with regard to mathematics and technology. In many universities, when there are options in computer sciences, enrollment numbers are falling for more technical subjects, such as architecture and system design, as compared to subjects seemingly related to social computing, such as net-based computing. The perception of computer science is that people sit in cubicles and only work on computers. Some of that perception may be accurate. Also, society feels that it’s okay not to understand computing. We are up against cultural trends, which may require top-end government action to fight. Some Eastern countries appreciate the problem better than we do.

Computer science is a space within computing, and should not be used as a term to describe the whole “space” of computing. Some large companies, such as Microsoft,
Google and Oracle, require computer science skills, but the full story is that there are other computing needs. Part of the challenge is in getting a broader definition of computing to the public.

Note that the message needs to work for young people, and not just for people already in professional societies. One barrier is that in high school, programming languages form the basis of computing education. A better approach is to provide a variety of activities, including web development, game development and programming, so that students get a sense of the range of options in the field. CSTA is trying to educate teachers that they need to bring a broader range of computing to students so that half the kids don’t get turned off by looking at Java code immediately. Also, around middle school, students notice that the technology at school is not as exciting as what they have at home, and this provides a negative connotation of computer science. This is true even at universities, which are often a generation behind everyone else.

Business students know how to use tools such as word processing, PowerPoint and Facebook, and can even create things by building from components at a very high abstraction level. This can be problematic when these students feel like they know everything and do not need to be taught further. However, in practice they do not understand the fundamental principles of problem solving, particularly algorithmic and computational problem solving, so it’s a challenge to convince them that there’s an entire field of knowledge that they don’t understand.

Can students effectively enter the field of computing without learning programming? One opinion is that tools are becoming so sophisticated that this is now true, especially in web design and multimedia. Another example would be a research project involving making animated characters seem human. This project uses sociology, art and computer tools, but no programming.

Work is being done on an Information Systems undergraduate model curriculum project, and programming is no longer in the core. There are heavy-duty data management, systems modeling and infrastructure issues that require truly complex modeling skills, but although programming is offered, it’s not part of the core curriculum (although students might believe that they were doing programming).

Programming is creating a sequence of steps to get an end result, and in some modern tools, this is accomplished by dragging-and-dropping, rather than writing text-based code. However, some programmers don’t know the basic underlying principles of computer science.

To broaden out computing beyond computer science, topics like computer security, identity theft, fraud and spam could be taught. This would appeal to students because they see this every day and it includes human and ethical aspects. Even younger students are more excited by—and interested in—real-world applications. This is difficult for teachers who have been pressed into teaching computing because of the shortage of CS teachers, and they tend to teach in an unimaginative way and create harm.
Computer science teaching certification exists in Ohio, but in few other places in the US. Professional development is a big issue for these teachers. There is insufficient literature that explains to teachers what computing is. There was a collaborative effort by ACM and IEEE computer society that ultimately resulted in two different concrete outcomes. One was the CC 2005 overview volume that provides an overview of the five computing disciplines that have the formal curricula, and the other was a four-page brochure describing what would be considered a very narrow model now. However, print media is not a way to reach today’s youth. CSTA has pilot projects for curricula that works for both computer science teachers or teachers from other disciplines who are teaching computing.

Another challenge is the lack of prestige in this work. Frontiers in Education, which is partially IEEE-based, used to try to communicate the excitement of technology, but the trend today is toward rigorously academic papers in pursuit of prestige in academia. The academic, obscure approach in IS has been labeled “good” and the populist approach (which often has a higher impact on students) has been labeled “bad.”

IS is computing plus business and transformation. The label “Computing + X” implies that X isn’t computing, which makes the division part of the label. In the case of information systems, the benefit occurs when the two pieces are integrated. Some see computing as a goal, but it’s really a tool to make things happen. For example, the media focuses on those times when computer failure is blamed for a plane crash, rather than on the daily effect computers have on keeping planes safer.

Society has gotten accustomed to receiving information for free, so students don’t want to pay for content. As a result, organizations like IEEE are struggling financially. In general, technology is taken for granted. If you ask someone how many times they came into contact with a computer before a morning meeting, they will probably forget about the computers synchronizing the traffic lights, keeping their cars running, allowing their phones to work, among others. We are increasingly reliant on technology, which is becoming increasingly invisible—until something goes wrong and causes frustration.

It would be interesting to have a new high school subject for computer-based problem-solving involving teamwork. For example, students could be asked to design a GPS system to keep cars driving safely. Unfortunately, teachers no longer seem able to get students to solve problems like this. Partly, students don’t want to fail, not realizing that it’s part of the process. There’s also the issue of being short on time, and not willing to spend a lot of time solving problems. We need some people who can do pure problem-solving, but today, most problem-solving is applied to a specific domain.

People get most of their information from the media, and certain professions are shown to be exciting, like the medical profession (E.R. and other shows), law enforcement, forensics and the legal profession. People in computing are often portrayed as nerdy and embarrassing. In the television show 24, technology plays a huge, exciting role, but the
technology experts are still portrayed as nerds. Trends on what profession is viewed as exciting goes through cycles.

Good computing puts computing into the background, making it invisible. For public perception, it hides many of the characteristics of the excitement that is associated with creating something. But computing is highly creative. Most of the successful models from the CPATH program are those that are doing the active learning, engaging the studio models, the deeper thinking or those qualities about whatever they are doing, where pedagogy are valued. In high schools, the kids that get into robotics competitions and similar pursuits seem to be more persistent. There can be a tremendous sense of excitement and accomplishment, but people are not aware of it. They certainly don’t perceive computing as a social activity.

Is having a definition important? Or does it limit the discussion and distract from the issue of perception? If we all have a general sense of what it means, is that good enough? We might disagree on what it is.

Why does computing need so many different organizations? Clearly AIS has a different mission than ACM, who focuses on a more traditional definition of computing, and so IS faculty feel as though they don’t belong. Computing may be uniquely diverse: Creating better animation models for Hollywood is a different skill than computing for banking and telecommunications. But we need some central organizations for cross-fertilization. Engineering manages to have a number of subdisciplines with a central core, and that central core is key to their identity. Computing is more diverse than engineering; the skills required to build physical objects use common tools and methodologies, whereas the skills to create animations are very different from designing an automatic braking system in a car, for example.

Students can resist gathering information outside of what is given to them, even though this is what is required in a computing field. They also can resist learning to write well, even though communication is an important skill that industry values. So, as part of our definition, computing includes human skills because computing is creating artifacts for people. Of course, nearly every discipline (finance, marketing, accounting etc.) requires these skills, but it’s important to emphasize, because some students are getting turned off from computing because they believe it doesn’t use social skills. Still, this raises the question of how can we define computing differently than other disciplines, especially engineering. In the case of computer science, there is a special skill that makes someone successful at a place like Google, Microsoft or Oracle.

Another unique aspect of computing is that many other sciences and engineering require expensive hardware and equipment, whereas computing allows you to implement ideas by applying pure mental capability to a fairly inexpensive computer. Computing is an inverted pyramid. At the bottom, there are probably 50 people in the world who work on the device physics; the next level has more people working on chips; and the applications are at the top, with the most people involved. The higher on the inverted pyramid, the
more domain knowledge is required. Society needs to recognize that we need capabilities in all areas of the pyramid.

Companies like Google and Microsoft have trouble finding qualified people because they are looking for a skill set that is more focused on narrowly defined, specialized computing skills. But there is a public perception that since the dot-com bust there are no jobs in this broad spectrum, which is not true. Are there openings for people who know programming especially? There are areas like data mining that do not involve programming, but do involve structured problem-solving using modeling capabilities. Computing involves not only different domains, but also the ability to make things usable and enjoyable, as demonstrated by the successful digital cameras.

In summary:
1. Computing involves an inverted pyramid and skills are required at every place in this pyramid.
2. Perception of computing is more important than its definition.
3. Unlike other fields, it is constantly changing and education needs to reflect the latest technology.
4. Computing has a social aspect.

Revising the Curriculum Challenges Session
Transcription: Curriculum2.doc
Participants: Mary Granger, Andy Van Dam, Stephen Seidman, Joe Trainer, Joan Peckham.

Mary is at a business school; the number of courses taught is limited, because ASEE accreditation requires arts, sciences and business core courses. This only leave room to take about five IS courses. Their IS program is not CSAB accredited because they don’t have enough courses. Joe is in the College of Engineering, which offers a masters in software engineering and a PhD in computer science. They are not accredited. He would like to align with the college the next opportunity. Andy is at Brown University, a private school with a bachelors, masters and a PhD in computer science. Like most top private schools in computer science, they are not accredited. Public schools, like UCS, UCLA, UC Santa Barbara and Georgia Tech, are accredited.

CRA is trying to address the educational needs of future researchers, and that is now extending into the sciences, since people are doing computational modeling, simulation and visualization.

What are the challenges for this subject? One is identifying different curricula. We need to convey what the different curricula are, since many students don’t understand the acronyms. Information Science can often mean library science. Some of the ISchools are still basically library schools, but others (Berkeley, U. Washington, U. Texas etc.) are quite sophisticated, although their roots are still in library sciences.
Possible challenges:

- Teaching students the things that will keep them in the field as it evolves at a frightening pace
- Attracting and retaining students in general, and especially women and underrepresented minorities
- Changing the perceptions about computer science and information technology as fields and careers

Andy and Stephen are both on the Changing the Image of Computing project, which is a purely PR effort with about 30 people. They hired marketing people and there was a joint grant with WGBH-Boston, who are experts at this kind of marketing and image changing.

Another challenge: How do we modernize the curricula on an ongoing basis so that they remain relevant when all other areas (both inside and outside computing) are also requiring time and attention?

Stephen has the opinion that we’ve been dominated for decades by the computer science curriculum of the 1960s and ’70s, when every major application that was developed in that period had its own course (like an operating system course, a compiler course, a database course etc.). Whether each needs to be a stand-alone course is hard to answer because there is a community defending curriculum as it is.

In the business schools, there is a challenge of integrating technology into other disciplines. It’s not just Information Systems by itself; it’s Marketing Information Systems, Accounting Information Systems, Financial Information Systems etc. So a challenge can be: How do we infiltrate and inform all the other disciplines so we help them take proper advantage of computational thinking? This applies to business, engineering, physics, chemistry, biology, medicine, social science and even music and arts. Currently, there’s no retention on new subdisciplines, such as health informatics. NSF might have graduation numbers, but probably not retention numbers.

Modernizing the curriculum needs to address what students will do in their careers, which is partly a marketing issue. Some turn down computing careers because they don’t want to work 80 hours a week. Mary said that she wonders if students in general are not willing to “pay their dues” when they first start out. She doesn’t think they realize that the real world is much more difficult than going to college in terms of working hours and meeting deadlines.

Some people say that you cannot fit all the information required into four years, so the masters programs become important. But the masters degree in the US can cover a wide range of studies. Bringing consistency to this space is important. In part, this is because different schools serve different communities, and the communities have different needs. However, students often move to new communities after graduation.
Another sub-challenge under modernizing the curriculum: How do you resolve the dynamic tension between teaching job skills that are useful now and for the next two to three years as far as we can see, versus teaching principles that are likely to survive over a 10-, 20- or 30-year career span? We need to teach the students how to learn over their lifetime, much of which is unguided learning. Mary commented that she doesn’t see students “playing” with technology as much as she would like. Students do have this mindset with games and cell phones, but don’t apply it to traditional software.

Summarizing this into a challenge: Designing a curriculum that attracts and retains students, but also prepares them for a career. Andy mentions that there is also the challenge of trying to cram too much into a four-year curriculum, so a strategy is needed for what should be removed when the additional components are added.

Another issue is relevance. Are students learning topics relevant to the real world? How do we keep the relevance when the technology is always changing? Computer science departments have a number of programming classes; this creates the perception that computing is programming and turns off some students. Is the number of programming classes required by CS departments too high? That depends on the expected outcomes for a CS graduate. If they require the experience of building a large software system, then they need to spend that time programming. However, Joe asked if redundancy can be reduced, and if the curricula could be reorganized so that the programming courses are not all at the beginning. Mary suggested that programming could be integrated into other material, such as project management.

Much depends on what employers want, and Andy pointed out that it’s difficult to get consensus on this. What a bank needs from a software developer is different from what Microsoft or Google wants. Employers are thinking about skill sets that they need now, not in 10 years. Graduates of business schools often go to consulting firms that say the most important thing they want from students is good communication skills, so the schools work on that. The firms also want the students to understand business organizations.

Mary is seeing more computer science students taking a minor in business, with core business classes like accounting, financing, marketing, management, human resources and quantitative analysis courses. She also sees people in the MIS program who are computer scientists who want to get more business, or they are doing too much programming and want to do more people-oriented subjects, such as project management and system analysis.

Andy again raised the issue of how to add more into a four-year curriculum because students want a broader education. So what is the core information that you need to get across?

Computer science loses people because of its math requirements; are the requirements still relevant? Tony Rowlston at State University of New York at Buffalo proposed teaching discrete mathematics before calculus, but it was never adopted. Starting in
middle school, girls seem to develop a bad perception of math. Part of the problem is that we are teaching the “wrong” kind of math (the kind that doesn’t get much use). Andy is part of a subcommittee at CRA that is looking at math, and will probably lean toward an emphasis on statistics, machine learning, techniques and linear algebra. Stephen added that every computer science student needs to know about concurrency and parsing, but they don’t need full courses on them. There is agreement that abstraction is a key skill, but how to teach that in a systematic way is far from obvious.

Many people don’t want to learn in-depth about computers; they use them as a tool. Although it’s useful to know how compilers and languages work, it may not be necessary any more. It’s no longer possible to know everything in the field. The number of tool builders is very small compared to the number of tool users. Building something rarely means building from scratch, but building out of components whose internal workings do not need to be understood (and they can be very unreliable).

What are the expected outcomes of our curriculum? There will be variations across institutions. Unlike other countries, the United States does not have an organized higher education system. In Germany, all disciplines must be accredited; this sort of legal requirement would never happen in the US.

Focusing now on strategies: what’s a strategy for attraction and retention? Some marketing strategies only work in the short term. Transforming high school level Computer Science is an example of a long-term strategy. It’s necessary; training math or science teachers for a few weeks to teach computer science isn’t effective. But even if you came up with a really good curriculum, it would be impossible to get universal agreement on it, and then you’d still need to get teachers up to speed who could actually teach it.

An example of a successful program is one used in Israel by David Harel and Judith Gal-Ezer. They studied what it means to learn computing in a scientific manner, and there was a national commitment.

Joan mentioned that in Rhode Island, when she asked to visit high school classrooms to recruit students to consider computer science as a major, she was frequently led to shop classes because they use CAD systems. Again, this suggests that people are thinking about computing as a tool.

One of the strategies that came out of a panel that Mary was involved with was that educating advisors, faculty and the media were critical. The media keeps writing about outsourcing, and parents are thinking, “Why go into computing when the jobs are being outsourced?” Other images that have been uncovered through the WGBH study are that it’s hard, it’s solitary (which women do not like) and there are no jobs. Andy mentioned that he used to open his introductory class saying that there were plenty of jobs for people who have a decent education and who know how to solve problems. But he has a student who has had trouble finding a job over the past year. Mary responded that her students in Washington have not had trouble finding jobs. PhDs may have a harder time.
Regarding high school teaching, NSF can jumpstart a program, but it needs to be sustained throughout the US, which is difficult. It may be more effective to try to infuse computational thinking into other science courses. Restructuring high school curricula is much harder than restructuring undergraduate curricula because there are more requirements and constraints imposed on high school curricula. It’s too big a job for NSF, but NSF can at least lay out what needs to be done.

Colleges can have requirements as well, especially state schools. Joan’s opinion is that it will take deans, department chairs and presidents of colleges to make decisions about including computation as part of a strong general education. It may involve engaging conferences that administrators go to, like AACU. The typical time a person stays a dean in a business school is five years.

**Professional Development and Lifelong Learning Challenges Session**

Transcription: Professional Development.doc

Participants: Mark Guzdial, Dennis Silage, Dave Chatterjee, Gail Chapman, Jan Cuny.

The ACM education board is not doing much in the way of professional development and lifelong learning, unlike many of the other organizations here. For ASEE, and their accreditation banner ABET (American Board for Engineering and Technology), one of the requirements is to demonstrate that your undergraduate programs have a commitment to lifelong learning and as they are imbued into the curriculum. In addition, you must demonstrate that both teaching and technical resources are made available for the faculty to get professional development. The IEEE student chapter has a professional awareness conference, ASPAC, about every two years. Engineers are brought in one year out, five years out, 10 years out etc. The latest one involved talking about cloud computing.

SIM (Society for Information Management) is practitioner-driven, with 3,500 members, primarily in the US, and you need to be a director or higher. SIM has some unique programs, such as the SIM-APC (Advanced Practices Council), which is comprised of 20 CIOs of Fortune 500 companies; they brainstorm on topics of research, and then researchers from across the world bid for the proposals. Grant recipients make a presentation in front of the panel every three months. This allows academics to do cutting-edge research and have their students involved. Papers are published first on the SIM website with member-only access; after six months, they are publicly released.

Another good program is RLF (Regional Leadership Forum). The goal of that forum is to develop management talent to become future CIOs. They work with mid-senior executives from different companies who are nominated. A third program involves monthly chapter meetings where topics of practitioner relevance are discussed. Academics come to share research, but they have to be able to make the connection between theory and practice. Finally, there is an international conference called Symposium held once a year, in which practitioners and CIOs come together to present
best practices and various insights (some academics are invited as long as their talks are relevant). Generally, SIM reached out to include academics. The SIM-net website has a library with research results. They now recognize that they get a lot more value from research projects done by academics than those done by more expensive consulting companies. Right now, SIM is not pitched at students or entry-level people, but SIM Memphis has been working with public schools to promote the significance of IS and discuss IS careers. SIM has 50 chapters. SIM does have a program for academic initiatives, although it’s been dormant lately.

CSTA is a K-12 membership organization with a mission to support K-12 computing teachers in a variety of ways, including professional development and providing resources. Programs vary by state, by district and by school. At the high school level, it tends to be AP classes. ACM has a K-12 model curriculum, and CSTA is trying to encourage schools to have a more formalized set of courses. The AP courses are somewhat standardized, with a topic outline and a standard exam. CSTA works with math teachers who teach computing, but some math teachers are not that interested in working with computer science. CSTA has an annual Computer Science and Information Technology Symposium with 200 participants. They also sponsor TECS (Teacher Engagement in Computer Science), which involves university/K-12 partnerships, and they host 10 workshops a year. Some of the smaller colleges cannot afford to host a workshop. The workshops cover modules from the K-12 model curriculum; for example, Java training. CSTA also has an NSF grant to build capacity in computer science by bringing two computer science K-12 teachers from each of the 50 states together for leadership and advocacy training; participants would return home and try to start local chapters and create a grassroots movement in their states. Last year’s workshop had 17 states represented and it was very successful. This year’s workshop has people from all states except Alaska. Dennis noted that getting AP classes into the Philadelphia school district was very difficult, but getting programs into the after-school setting was no problem.

NSF’s Broadening Participation in Computing Program works to get more women, African Americans, Hispanic people, Native American people and persons with disabilities into computing. That’s 70% of the population, so it’s a big target group. The goal is to get more students to achieve undergraduate and graduate degrees. They run programs for middle school through graduate faculty ranks. Their approaches include informal education, formal education, mentoring, research experiences for students and summer camps. Jan is increasingly convinced that the number-one problem we have to fix is high school and what happens in the classroom at high school, since the number of kids who show up at college saying they want to major in computer science is down 70% since 2000. Some colleges have seen a recent turnaround in their enrollment, but fewer people are coming into college with a desired to major in CS. She has a handout called “How to Transform High School Computer Science,” and she wants to put a new curriculum in place for an AP course. AP courses are important because it allows the curriculum to reach high schools nationally. And she is working to get 10,000 teachers trained to teach it in 10,000 schools by 2015, since that’s when the new AP course will be available. It’s a huge project, and NSF is not going to fund anything of this magnitude, so
it’s looking to form public and private partnerships with companies, professional
ingurations, foundations and other government agencies. Professional development is a
big piece of this effort. Although 36,000 people claim to be teaching computer science
currently, but a survey through CSTA saw just 770 people fill out the survey form. Many
teachers count keyboarding and word processing as computing. The new AP courses
would emphasize the fundamentals of computing and not emphasize programming. We
need to get students to understand that computing is a very social activity; that it involves
teamwork; that it cuts across many fields in terms of applications; that if you major in
computer science you don’t necessarily go work in a cubicle at Intel. We need to show
them that computer science has wide applications, that there is a “magic” component
whereby they can create solutions to problems that interest them across all fields, and that
there are fundamental ideas in computer science that are independent of any
programming language, although programming is an important aspect.

Dave mentioned that he doesn’t think we’ve done a good job of clarifying the distinction
between computer science and management information systems. At the high school
level, it would be hard to figure out which kind of career students would be interested in.
As organizations, we need to recognize what these disciplines are, how they are similar
and different, and then find a way of pitching them. Industry is looking for particular
skills; for example, one company would not recruit a computer scientist for an enterprise
architect, preferring an MIS background because they want to make sure the person
understands how to apply the logic of computing in the context of business.

But perhaps the high school classes and even the first year of undergraduate classes
should consist of general computing; later, students can choose the direction in which
they’d like to go. It’s important to explain how computing is used, and that explanation
can happen as early as 4th or 5th grade. A guidance counselor could do this, but the
average school now has a ratio of one guidance counselor to every 400 students. Using
faculty or graduate students is an idea, but schools are under pressure because of
mandated achievement exams to rate their schools, and so it’s hard to get them to do
anything different. There is general agreement that the process of teaching to these tests
is missing the point of good education. They education process is very siloed, and
connections are missed between reading, writing, science and technology.

NSF also has a project called the education workforce cluster, and Jan has been asked to
look at computational thinking. This brings up the issue of how you define computational
thinking. It’s a lifelong learning skill that students should learn in math class when they
are in 5th grade and in science class when they’re in 7th grade, and it should be
throughout the curriculum. Computer scientists are trying to define computational
thinking (see Computational Thinking by Jeannette Wing), but these definitions do not
help K-12 teachers. It’s especially hard for science teachers to integrate this. Dennis uses
Lego robots as a way to teach computational thinking and logic. But logic has a specific
definition, so we shouldn’t use that instead of computing.

Dave mentioned that he has students from human resources, marketing and finance, and
they have a phobia about programming. His approach is to talk about managing
information, asking if they need information for decision-making, and then talking about the role of technology.

Mark is the director of an effort called Georgia Computes, which strives to improve computing education across the pipeline. They’re training teachers, they’re involved with CSTA and they’re training undergraduate faculty and working to create new curriculum. He mentioned two projects in which his graduate students are involved. Lijun Ni is studying computer science teachers (both university faculty and high school teachers) and the first research question that she was interested in was: “What convinces a computer science teacher to change what they’re doing?” How do they decide whether to adopt a new curriculum, or that what they’re doing is wrong and needs to be fixed? And that turns out to be a difficult question to ask, because it’s hard to know how anybody adopts anything new in technology. The bottom line of the studies was that the major factor that influences somebody’s decision to change is their personal excitement about the new curriculum. What’s striking about that is, for example, we do many teacher workshops. We teach new kinds of curriculum and we talk about the research that has supported that this curriculum is a good idea. Research results have very little effect on teachers’ decisions to adopt something new. Lijun Ni is now working on notions of teacher identity. It turns out that how a teacher identifies him or herself (“I’m a math teacher” or “I’m a business teacher” or “I’m a science teacher”) heavily influences his or her commitment, the desire to change and the decision to stay with the field. Very few teachers identify themselves as computer science teachers. This is partly due to the lack of definition of what is a computer science teacher, and it’s partly because they see themselves as a math teacher who also teaches some computer science. Therefore, a really good strategy in trying to grow more computer science teachers is to define the identity of a computer science teacher and figure out what influences that sense of identity.

Another of Mark’s students, Brian Dorn, is studying non-computer scientists who use computing. These are programmers, such as people who add programming to their web designs and graphics designers who start programming PhotoShop to process images for themselves in batch mode. None of these people have taken any computer science classes; they’ve taught themselves. Brian interviewed people who fit this description, and he gave them a number of computer science concepts on cards and had them sort the cards into which ones related to what they do, and which are ones they teach in computer science classes. Concepts like commenting and structuring code in particular ways for reuse are examples of what they teach in computer science classes, but are irrelevant to what they do. They are clearly doing computational thinking, but they don’t care about working in teams and they don’t care about creating code that’s going to last for 10 years, because they’re just writing a piece of code to solve the particular problem. It’s a different mindset about what is important in computing. They want to learn more about computing; it was interesting to hear them say, “If I understood more about how it works I could do it better.” But they don’t want to take computer science classes, because they think that what we’re telling them in computer science class is irrelevant to what they’re doing.
Jan brings up the point that this is partly due to society’s view of education. Students don’t like to be taught things that they don’t need to know to solve the problems that she gave them; they seem to believe that in the rest of their lives, they wouldn’t need to know these things. It’s a bizarre view of learning, and it’s more noticeable in computing because it’s a field that is a means to an end. Dennis suggests that this is a result of high school education, in which students are only taught what is going to be on the exam.

High school teachers who are teaching computing don’t have an identity, but how can they, if we don’t have a definition? Having a high-level definition is good, but it’s not something high school teachers can use, and it’s not something that will influence the high school decision-makers, such as the principals and school boards.

There is general agreement to come up with a term that’s different from “computer science.” Dave asks, “What is beyond the purview of computer science?” He prefers to map his courses to a set of fundamental skills. He feels we have isolated ourselves with our labels, and there should be more communication between computer science departments and departments who use computing. Jan agrees, and says that the computer science faculty often does not value this kind of communication. There’s a $37,000-a-year program at NSF where students spend 17 hours a week in a K-12 classroom and work with teachers. There are physicists, chemists and biologists, but no computer scientists, because computer science faculty will not allow their students to participate. This is unfortunate, because high school teachers who are trained in another subject, such as math, need help in getting committed and excited about computer science. It’s very difficult for high school teachers to find the time to learn new subjects to teach, which is constantly required for computing.

Dennis mentions that engineering education is moving away from the lecture model. Prior to World War II, engineering education involved work in the laboratory with students. In Georgia, they’ve introduced a program called Scholarship of Engagement, which recognizes the significance of service. Jan would like to see undergraduates get into the classrooms as part of the 10,000 teachers project. They could act as lab helpers, since it’s difficult for teachers to manage 35 students in a lab.

In summary, the challenges are:

- We need to train more K-12 teachers, aiming for 10,000 by 2015.
- We need life-long professional development:
  - For K-12 teachers to sustain them;
  - For faculty to teach them new paradigms;
  - For practitioners across multiple fields

A question is raised whether professional development for practitioners is a challenge, or whether it is at an acceptable level. Also, is it a fundamental problem in academia that there’s too much focus on research and not enough on teaching?

Showing connections within the computing discipline, within subdisciplines and across science is also critical. An if/then statement is just as important to engineering as to MIS.
A strategy would be to get universities to recognize this, since the existing performance metrics for faculty do not reflect this. Universities value neither teaching nor outreach.

What should be the strategy for defining and branding? It works best to do it in a group that’s large enough and esteemed enough to get buy-in from the rest of the community. The branding needs to be inclusive, so if there is a large group, it needs to include people from engineering and MIS, for example. This conversation has been going on for many years—how can it be done differently?

Do we need more standardized tests? If teachers are going to teach to the test, we need to create some computer science or computing tests out there, so they’d be teaching in directions we like. Jan feels that we don’t want to do this. There is already a test being developed for the AP course and that’s hard enough, and there is the baggage left from No Child Left Behind. It’s better to focus on certification.

Related to this is the fact that in all states but two, taking computer science does not fulfill any requirement for high school graduation. In Texas it counts as a math; in Georgia it counts as a science, and those are the only two. There is an effort from the ACM education policy committee and the CSPA to work state-by-state, but that means 48 states, the District of Columbia and Puerto Rico need to be persuaded. It’s a complex problem: in George, for example, the Department of Education sets curriculum, but the Georgia Professional Standards Committee sets standards.

Under defining and branding, there should be a relationship to Career and Technical Education (vocational education). Microsoft and Cisco certification are examples of CTE classes, and more academic content is needed to form a foundation.

The foundation is important: If we’re producing students who can walk into Intel or walk into Microsoft without having to be taught anything, then perhaps we’re not having enough time to give them basic fundamentals. When that job changes in two years, are they going to be up to the change? It’s a balance, because if what we teach is too broad and abstract, then they don’t have a sense of what goes on in the business world. In order to implement enterprise resource planning systems, people need to have an understanding of both technology and its application.

**Creating More Attractive Pedagogy Challenges Session**

Transcription: pedagogy2transcript.doc
Participants: Lecia Barker, Boots Cassel, Tom Hilburn

First, it’s important to define what pedagogy is and how it differs from and overlaps curriculum. Pedagogy involves the process of teaching and learning. The traditional approach has been lecture and hands-on laboratories. This approach has been around for a long time and has its place.
Another approach that has growing interest is to use case studies. The University of Buffalo has a case study institute, and of the 500 case studies on their website, only two involve computing. The case study approach could involve ways to improve the quality of a software product. The most popular technique is testing, so you are teaching students how to inspect something. Students form groups of four or five and are given a problem, such as inspecting a requirements document. They go through an inspection process, produce results, and then you have a class discussion about those results.

Other case study examples: textbooks give sample code and ask to modify it. Boots saw a biology example where the goal is to hire a new graduate for a specific job, and students need to compare résumés to requirements. Students worked in small groups and presented results in front of the class. In the end, everyone voted on a résumé. This was a good way to prepare students for creating their own résumés.

These are examples of active, problem-based learning, which is not used widely enough. It’s a form of pedagogy that engages students. Ideally, it duplicates the real world in an academic environment, so students can see the value of what they are learning.

In-class exercises are valuable because they show that developing software is not all about programming. It allows students to learn analytic skills, conceptual modeling, teamwork and negotiation.

Tom describes a two-week course he took for industry elements, in which experienced programmers worked through problems in a group setting while getting help from instructors. He concluded that this was more effective than the way he was teaching: giving his students material and problems, telling them to work on their own, and come ask for help if they need it. Lecia mentioned a student who took an introduction to programming course and didn’t bother going to classes; not focusing on programming to begin with could change students’ attitudes.

There are some very good teachers out there. Some are good for teaching people who are already motivated, so the key is getting students motivated as well as improving the quality of teaching. One challenge is to engage students in the discipline, which means they need to understand early what the discipline is about. There is often a breadth-first approach with several small topics, but the general opinion was that it is not very engaging.

Lecia mentioned that research shows that collaborative approaches in the classroom increase retention of all students, women and men, of all races and ethnicities. The case study approach puts students in groups and has them talk about real, meaningful situations, so it’s both collaborative and meaningful. The challenge is using this approach at the beginning of and throughout the curriculum. It helps with both retention and attracting new students because it can change the culture and perception, as students talk to each other about what they are doing in classes.
Teamwork can be separated from case studies, although they overlap. Tom mentioned that they have an industrial advisory board, and they always list teamwork at the top of what they want to see in graduates. Teamwork involves assigning roles, including a management role. You can also have pair programming (part of Extreme Programming agile development methodology—Laurie Williams has done research on this.)

The challenge is really to engage students in the discipline. There’s agreement on the challenge, but the strategy is another issue. Many faculty have tried active learning, and many have discarded it. This could be due to insufficient resources and training. There is also a need to change attitudes: Some faculty and students believe that students should not talk to each other while doing assignments.

Lecia suggested that perhaps if faculty is careful to say that an introduction to programming is like learning to read and write, and you need to learn this first before you go off and do cool things, then maybe students would understand that computer science is not programming. Boots responded that children learn to read by being read to, which Tom said is what is done in the classroom. Boots questioned whether programming is the base that everything needs to be built on; she pointed out that this used to be true in the past in with computers, but now you can do very cool things without getting down the programming. Tom mentioned MatLab as an example of this.

There is agreement that students feel something is “cool” because it works. Lecia said this is supported by interviews. It’s the accomplishment, and now they can accomplish some of these things without programming.

Is part of the challenge providing resources? Faculty members don’t have time to make up their own case studies. Shared resources would be valuable, especially if they are easily adaptable to courses. Tom described a requirements course in which it was difficult to find a problem for the students that was realistic, but wasn’t so large that it couldn’t be handled in a classroom environment. Lecia mentioned that existing curricula often have prerequisite knowledge, which might make it hard to adapt to individual courses.

Other types of pedagogy besides active learning are applicable to computing. For example, there is the concept of abstraction, starting with the bigger picture and working toward the bits. Most programming courses start at the bottom. This may be historical, since many teachers learned programming in the early days of computers, where this was required. Ideas for starting at the top include the Java framework, Scratch and Alice.

How do you get faculty to actual adopt new pedagogies? In addition to explaining the benefits, you have to give them the resources. Workshops to demonstrate techniques can be very effective. How can this be implemented as a long-term strategy? One approach would be to have follow-up workshops, so people who did the first workshop come back as mentors for the new group, building enthusiasm. The program could be self-sustaining.

The first year is an important time, and a good place to focus efforts. But it’s important that advanced courses are engaging as well.
Active learning can be defined as learning by doing, as opposed to learning by reading or listening. It is problem-based or project-based, where you set out to solve a problem and you discover needs. Solving problems with constraints—scheduling and cost, for example—is very effective. However, active learning needs to be distinguished from a 20-minute lecture followed by an activity.

Frequent guest lecturers from industry can effectively motivate students; this may be more effective at the early part of the curriculum. Boots describes meeting with engineering students who felt that they had to take writing classes that were not relevant, and she had to explain how engineers spend much of their time writing requirements, reports and proposals. Even religious training is useful, since engineers need to interact with many cultures that are impacted by religious traditions.

A challenge is that many faculty have not had industry experience, so they cannot explain what industry is like to students. Again, resources would be useful, as well as more interaction with alumni.

Student/faculty interaction has been shown to be a big predictor of retention in a major. How can we get faculty to spend more time talking to students? Perhaps workshops are a way to teach faculty how to do this. However, we need to recognize that there are personality issues with switching from the “sage on the stage” to the “guide on the side” approach. There is much less control with the interactive approach, and it takes a while for faculty to get used to it.

What about post-undergraduate education? Women are not retained at the same rate as men. A study showed that while people might consider leaving for several reasons, overt sexism was a reason that women actually cited. Less overt sexism is an issue as well, such as how female students are treated inclusively or exclusively, or how women’s talents are used.

In K-12, there are a number of issues around what is required, and whether there is anyone capable of teaching computing. Sometimes computer science classes are just keyboarding, spreadsheets and word processors. Sometimes the classes are considered vocational. Perhaps a teacher knows computing but does not know how to teach in high schools, or the teacher knows teaching but does not know computing. Providing resources, such as curricula and information about computing careers, could help. Parents and policy-makers need to understand that computer science is not the same as putting computers in schools. But these are not pedagogy-related solutions. An engagement between college teachers and high school teachers could result in improved pedagogy.

SIGITE decided to establish some common idea of what an IT program should be, but some of their members had programs that were technically strong, and others did not. It was difficult to come to agreement, but eventually they did, and Boots believes it is fairly strong.
It’s important to emphasize that curriculum recommendations are not requirements.

Examples of resources include iTunes-U and lectures on YouTube.

**Promoting Computing Challenges Session**

Transcription: Promoting Computing.doc

Participants: Andy Bernat, Raymond McGhee, Barbara Boucher Owens, Han Reichgelt, Ty Znati.

Barbara has been trying to promote within ACM that all SIGs should have an education focus. In the past, CRA didn’t pay any attention to education, because it didn’t want to encroach on the territory of ACM and IEEE CS. CRA always surveys with the Taulbee survey and gets numbers. However, it’s difficult to get information beyond the Taulbee survey, and the survey focuses on R1 (i.e., PhD-granting) institutions. NSF does have a broader survey, but the results are hard to find, and they tend to be two to three years behind. Barbara would like to see surveys that include both numbers and salaries. However, the question remains: How do you compare salaries, and whom do you survey? CRA buys a commercial list of computing programs to do its survey.

Measurement through surveys is an important part of effective promotion of computing. How do you assess programs beyond ABET in a general way? What are graduation rates? What are diversity metrics? Han said he is looking at numbers in an attempt to evaluate diversity, and non-PhD-granting institutions have better minority numbers than PhD-granting institutions. A study worth looking at is the New Image for Computing, a joint effort by ACM and WGBH, led by Jill Ross. The research indicated that there weren’t significant racial or ethnic differences—but there was a significant gender gap, so they changed their focus to girls.

An important and unanswered question is: what is keeping kids out of computing? CSTA is working on this at both the individual and state levels. Currently, there is no central place to communicate and no consistent message. It’s important to figure out which messages work. For example, several messages were tested to see which was most effective: computing puts you in the driver’s seat, computing open doors, computing empowers you to do good, computing brings people together, computing is achievable, computing calls for creative problem solvers. The three most effective were: does good, opens doors and driver’s seat. Girls liked “does good” the best. The old approach from CRA was to emphasize money and great jobs, and was not effective for people studying toward a PhD.

Another concern is that when the discipline is promoted well, we still see a high rate of students dropping out of the major. Han loses 35%-50% of students in the first year; this is apparently typical. Does what is being sold match what they find when they get into the major? If they are focused on computer syntax for the first two years before doing anything exciting, it may feel like bait-and-switch. A study that discuss this is Women’s Underrepresentation in Science: Sociocultural and Biological Considerations. It shows
that most programs are designed for mathematically precocious youth, so we are designing are programs for the “tail” rather than the larger population, and this “tail” has a brain difference. It happens that there are more males in this very small tail than females, but in the larger population, you do not see a brain difference between males and female. There are also perceptual differences in how male vs. female students are viewed, both by family and teachers.

NSF funded a study that looked at real projects and found that 70% of software engineering is non-programming, and 30% is programming; girls might be less turned off if they knew this at the start. Classes at the middle and high school level could focus on the 70% at first, moving the students onto the 30% once they are hooked. If we start with the 30%, then the perception is that computing people sit in a cubicle and program.

However, we need the people who can do the 30%. Predictions from the Bureau of Labor indicate that there will be a serious shortage. We also need people who understand computing from a liberal arts perspective. Computer science is not the only path into computing. But the problem is that computer scientists can’t articulate what computing is if it’s not computer science. Lots of people have tried to describe computing as more of a liberal art, but it’s very difficult. It can be argued that in order to understand computing, you have to understand the concepts of programming. People defined computing as math 20 or 25 years ago, and that was a mistake. Kids do like programming in Alice or Scratch.

What do you teach in a computing course that’s not programming? One example is Computer Science Unplugged, by Tim Bell, and there is other work going in K-12. K-12 is likely to have the biggest impact. For undergraduates, it’s difficult to currently minor in computer science, so we need another approach. However, we got into trouble in the 1980s when we said it was okay to major in computer science without taking math. The problem was that it was designed for the “tail”, and so math was still needed. Could things be redesigned so students could major in computing and not need math, but still get the basic “footprint” of computational thinking?

How many computer science majors do we need, as compared to computing or information majors? We definitely need people who can learn and apply computing at a certain level. However, university departments are geared toward majors, not courses. Currently the number of students getting PhDs and masters degrees is rising, but that’s due to the economy. Why, after the dot-com bust, did students stop majoring in computing? The US graduates more PhDs in physics and astronomy than computer science, and no one goes into physics and astronomy for the money. They go in because there are exciting, interesting things to do, and computer scientists have lost the ability to notice what is exciting about computer science. CRA is working hard to bring this excitement back. CRA had an idea to copy astronomy, which has a picture of the day, and show a computer research highlight of the week. But they couldn’t get nearly enough people to submit them. There is exciting research being done, such as done by Ed Lazowska and Luis von Ahn. This excitement needs to counteract the bad publicity around off-shoring of technology jobs.
We need to focus on the excitement of computing, its impact, its ability to do good and its value. This requires that recruiting materials be made available, and they should be customizable. CSTA has discovered that their brochures were too generic, so now what they have is customizable.

Currently, not even other university departments understand what computing is. However, you can argue that if you want PhD students, you need undergraduate majors, which means that you need interest in K-12. So you need a broad range to capture the excitement.

The number of males (white males in particular) in higher education is shrinking, and the college population is becoming largely female, so we need to focus on attracting females. However, all of the studies have shown that if you attract females, you also attract males; designing a program to be gender-neutral will attract males at the detriment of females.

If universities are not good at self-promotion, could a partner be good at that? CRA hired a communications firm for the release of this year’s Taulbee survey results; this led to the results being published in the New York Times, and then in other papers, blogs and on the radio. It may be worth the expense to send someone to institutions to find out what is exciting to the students. Some universities already do this well for themselves, but the information is not getting to CRA. This brings up the issue that some of the exciting news might not be exciting to CRA.

The National Academy of Engineering had a Grand Challenges approach, with about 12 challenges, where half of them involved computing. It was not successful, and it got more attention overseas than in the US.

Han described a project where he has students working with fifth-graders from a nearby magnet school. Most of the students are minorities, and they work with his students on an educational game. The fifth graders do the design, and Han’s students program it. They created a press release that announced its success, sent it to the local paper and then the Atlanta Journal Constitution picked it up as well. There are many stories like these: Jean Sammet is organizing teaching computing to senior citizens. However, it’s hard to go from local press to national. Han mentioned that he sends out a monthly newsletter to the public relations people; they pick stories from the newsletter and collaborate with him to write a press release. It never occurred to him to send a copy to his congressman as well. A resource that describes how to publicize would be useful.

Working with communications firms is not always successful. The Image of Computing project was an example of this collaboration, and it dead-ended. One opinion is that the marketing people in charge did not understand computing well enough.

What about viral marketing, using resources like YouTube to disseminate exciting news? Facebook could also be an option, but first it must be determined if the target students are
using that particular social network. Several approaches could be used in parallel, even before the research is done.

Regarding partnerships, we discussed building alliances with organizations that can assist in getting the message out. What kind of organizations would we use for different sections of the pipeline? For K-12, Boys and Girls clubs have been focusing on computers (Club Tech). NCWIT and CSTA are working with the Girl Scouts and 4H Clubs. On the higher end, we’d love to have a partnership with the AAAS (American Association for the Advancement of Science), but right now their meetings do not discuss computing. There is a committee associated with the National Science Board that’s focused on computing and computational thinking.

Regarding messaging, part of the problem is that there’s no elevator speech. We have large volumes, such as the one that Eileen Trauth put together. Existing research needs to be collated and distilled.

**Research and Measurement Strategy Session**

Transcription: Tape 22

Online IT courses tend to have more diversity than equivalent classroom courses. Surveys show they work well for mothers who work or are looking to get back into the workforce. It may be similar for ethnic minorities. For example, Hispanic students can be reluctant to move away from their communities to go away to college. However, there was a review of educational research that showed that data from 30 studies indicated that the withdrawal failure rate online is twice what it is compared to face-to-face classes. One person said they have a higher retention online than face-to-face, but they are not completely online; they have a hybrid system where they bring people in for a week at the beginning and do real-time audio/web sessions once a week. Web classes at Georgia also have a very high retention rate; these classes begin with a face-to-face meeting, but then are not face-to-face after that.

SPSU and Georgia Southern are examples of Computer Science departments with significant ethnic minorities. George Southern is a residential university with 16,000 to 17,000 students, and approximately 30% undergraduates in computing were African American. SPSU a suburb setting of about 5000 students with a lot of commuters and the same percentage of African Americans as in the Georgia population as a whole, which is about 22%. By contrast, Georgia Tech has very few.

There are several non-traditional learners involved in computing. These include mothers returning to the workforce, students at non-PhD granting institutions, IS/IT/CS software engineering, computer engineering, informatics, and other engineering.

There is a spectrum of urban vs. rural. There are rural 4-year schools (no PhD program) and there are places like Ann Arbor, where there is nothing around the school, but it’s not
what you would call rural. It can be a challenge to attract minorities to places like Ann Arbor.

The Georgia Computes effort exists to improve the computing education progress across the state. The university system of Georgia, which all public universities in Georgia belong to, said that they have all the class data for all of the schools. Unfortunately, it turns out that although they keep it in all in one database, but there are 32 separate databases within this database for each of the institutions. This makes it impossible to compare them at all without knowing each of the institutions separately. In the end, Georgia Computes could not determine retention rates, which made it very hard to craft a policy argument to a state legislature.

There are many social science questions to be asked. One is the return on investment for new pedagogy. There are also diversity questions, such as: is it the case that there are differences in ethnic rate or gender differences between non-PhD granting institutions and PhD granting institutions? Or by IS/IT/CS departments? The numbers at Illinois are 83 men and 214 women, meaning more diversity would be more men. If diversity is theoretically important because it brings more different kinds of perspectives into the classroom and ultimately into problem solving industry and into education then it doesn’t really matter what kind of diversity you are lacking. It just matters that you don’t have it. Are there strategies we could set up to try to increase diversity where diversity in one area doesn’t necessarily mean the same as it does in another? In some areas, computing-related departments have a significant African-American population, but not Hispanics. The Hispanic population for the University System of Georgia is growing really, really fast, but they are not ending up in any computer-related fields, engineering, or business.

Another type of question that can be answered by research and measurement is: How are pedagogies being used?

How do we prioritize and raise questions as a community? How does the community decide which questions are going to give us the greatest leverage in addressing challenges in computing education?

There is also the issue of understanding culturally-specific issues around the perception of computing education. This could also be a research question to understand the difference between China and the US in the way that they see computing education, looking at national vs. multinational perceptions. One school has a good diversity record because they have a Masters program in computer science which is 80% of students from India, of which 60 or 70% are female.

Another research question involves understanding the effectiveness of different kinds of incentive schemes and different incentive strategies. For example, in the diversity group this morning they were talking about the desirability of assessing the social impact of NSF funded research and the way that is generally done is you require someone to say something about it when they ask for the money. Unfortunately, that’s not a very effective way to actually make sure that anything happens. If you wanted to make sure
that something happened, you’d measure it at the end of the project and not at the beginning, and then you’d have to make the next grant contingent on that outcome. That’s a different incentive strategy than simply mandating that people pay attention to it at the proposal stage. The Course Curriculum Laboratory Improvement (CCLI) program has a model like that where they give small grants, no more than $75,000, to try out an idea. You can then apply to the phase two grant which is up to a half a million in order to be able to use, but you have to have enough results to get to the next stage where you can do more formal assessment and national dissemination. Then if you get that and do that well then you can go to Phase Three, which is a national rollout where you have to do clinical studies with randomized sets. So the research question here would be actually assessing programs like that to see whether the program as a whole, not the individual projects within it, actually worked better promoting some outcome.

Learning effectiveness is a related research issue. No Computer Science effort has ever won a Phase 3 competition as I understand it. And the reason why is at the Phase 3 multimillion dollar grants, computer science education is up against physics, math, and chemistry education, where they have reliable standardized tests of learning that can determine if one group has done better than another. We don’t have those tests in Computer Science and therefore no reliable standardized test of computing learning. The AP is the closest test, but it is very language dependent. To test learning such as Matlab, Python, or Robots using Java, only the Java test is available. The lack of cross-disciplinary accepted measures of learning effectiveness is a real huge issue in terms of measurement.

In general, STEM disciplines are having trouble recruiting students. One question to ask is, “What is unique about computing compared to other disciplines?” Does its interdisciplinarity have value?

This raises the question, “What constitutes computing?” It would be good to get the answer from citation analysis (i.e., data mining of literature references) rather than defining it in advance. At an early iSchools meeting, someone from Michigan did some mapping and mining of course descriptions across the iSchools and basically clustering of vocabulary from these course descriptions and it was really interesting to see how the schools clustered in terms of their emphasis. Most of it turned out to be as expected (not counterintuitive), but this kind of method provides more confidence than sitting down in a group and trying to decide it.

Another question from the education board is about standards. The ACE and education board produces a number of curriculum volumes. We know very little about how they actually get used or what people think about them. There is data on who’s downloading them and how often they download them, but this is not useful compared to knowing who is not downloading them or who is downloading them and then rejecting them and why. That is really important information for us in terms of deciding where we should put emphasis. Also, it would be useful to know what employers think about the students that come out the programs that use these curricula or don’t use these curricula.
Unfortunately, standards bodies often don’t actually start with research at the end of the process, but make assumptions about what is needed. Examples include the AAAS standards called Science for All Americans, and the Virginia Standards of Learning.

Where can research and measurement address the challenges of computing education? Diversity, policy, attractive pedagogy, and revising the curriculum are some areas. (Note that pedagogy is separate from curriculum because pedagogy means covering material in different ways, such as active learning.) Incentive and rewards could also be helped by research and measurement.

Promoting computing is another area. If you look at recruitment issues or cultural issues, which messages about computing appeal to which group? That could be determined with research.

In order to define action items, we need to define our goals. This could require some sort of committee with representatives from the various organizations. One idea is to ask across these associations what the top five things are that would be useful to them to be able to measure in order to influence policy or increase diversity. Then you could analyze the answers and see if you’ve got three that keep cropping up.

It would be important to have industry involved as part of this. They tried to get more industry representation at the summit. Industry might be interested if asked about what characteristics they are looking for in employees and where research could help to understand how to get that. Industry in general has been more receptive to a project from ABIT CIC because they ask that question in terms of what are you looking for in terms of employers. They are not going to tell you they must have done a course in data structures, but they are going to tell you is they must have experience with Oracle, or something like that. Computing educators are the ones that should be able to translate those skills into curriculum. Industry has the ability to answer the question, “What are the top five characteristics you are looking for in employees?”

However, it may be that no one knows how anyone learns computing. There was a student who measured if people could solve in C a really simple problem. They found that they couldn’t solve it as a freshman or sophomore, but by their senior year they could do a lot, and were amazing programmers. There are a lot of underlying things being learned in the first two years which we don’t know how to measure. The only way we know how to measure is: “Write me this program.” We don’t know how to measure, for example, a student’s conception of data, conception of looping, or conception of variables. All of those must be developing during those first two years that it all starts appearing at a high level. Industry may say, “I want people to learn Oracle,” but that’s an outcome. What do we have to teach them inside of that? We don’t really know.

This is a big difference between computing and other disciplines. We know a lot about how math and science knowledge develops, but that sort of research has not been done about computing yet. So there are very few studies that have looked at people before they learned anything about programming to ask them to undertake activities involving
looping, variables, algorithms, etc., in order to understand something about how people think about these sorts of things before we get them into classrooms. This is important in order to develop a developmental model.

An example that illustrates a developmental model is a study where they took a number of classic textbook programs and methodically broke all the design rules and then asked students to compare them and pick which one is a better design. They always picked the other one with the broken rules because it did not use named constants, which required the trouble of looking up the name. So some hints exist for understanding how to learn computing, and what things we have to be getting past. But we know relatively little about the sequence, what do students learn first and what will that allow them to learn next.

We have two action items: (1) set goals for which we need for research and management, and (2) draw on industry to help set these goals. How can we get funding? There is not a clear part of NSF that funds computing education research right now. Some of CPATH does fund ways to revitalize undergraduate education, but research which informs that revitalization is currently not an NSF program. Getting behavioral sciences funding is difficult because most of us doing computing education research can’t really compete at that level, even if partnering with social scientists. FIPC still exists, but it doesn’t fund research. They fund changes in educational programs where you can assess what happens. NSF did fund actually Sally Fincher, one of the top computing education researchers in the world, to train a several Computer Science teachers who wanted to become Computer Science education researchers. There were two 2-year efforts where she taught them how to do methods and had them engage in a group project. A lot of those efforts have continued on because the methods that she taught them were things that they could do collaboratively, even though they are mostly full-time teachers with only partial research appointments. They will collaboratively come up with a research instrument, do it in all their classes, and collaboratively analyze and publish together. In a real sense, she revitalized computing education research in the United States.

One of the things that we talked about this morning in the diversity group was the math genealogy project, which tracks who was a student of which mathematician. You could award incentives and prizes based on similar empirical methods, such as “This person produced more women PhD students or more minority faculty members or more whatever over 3 academic generations, which is 15 years and a manageable period of time.”

Other action items: the education board should actually try to do some measurement of standards, such as uptake of standard and opinions about standards, so we would have some feedback on the volumes that we produce. Also, ABES should be doing something on accreditation.

More action items: Think about a better way of sharing the results of this research. Promote organization-specific studies such as ACM and IEEE on curriculum buy-ins, and
ABET on accreditation effectiveness. Also cover SIGC, SIGI, and AIS, like CRA does for its student populations.

Also look at the audiences that are impacted by online courses. For the faculty who do both kinds of teaching, teaching online changes the way they teach on campus often with positive effects. Students don’t necessarily prefer only face-to-face or online; they want something flexible. They’ll use both and they will graduate sooner, and they will probably graduate in higher numbers too as a result of being able to do that. For example, a student that is deployed to Afghanistan will graduate if he can finish his degree from over there, whereas otherwise he might not.

Some material is easier to learn online than others. Mechanical engineering is difficult. Teaching people how to do things with computers online is hard because students are using their computers for both the learning and the activity. In IT, you have to do physical activities, such as placing wires. Foreign languages are difficult to learn online as well.

Faculty and students would both be good audiences to ask questions of. A PhD student just studied why students would choose an informatics minor rather than go into computer science. It turns out they have a distorted idea of what the profession is like. Sometimes they say, “I’m just not very good at math and I don’t think I want to put myself through that.” But often they say things like, “I don’t want to spend the rest of my life alone in a server room.” These perceptions also show up in a BS program in computational media, where students believed they would have to take a difficult computer graphics class because they erroneously concluded that computational media meant video games and working for Pixar. So studying student perception is very important.

One other piece to study is student transitions from one institution to another, such as high school to university, two-year institution to a four-year institution, or a four-year institution to a PhD program.

**University/Industry Partnership Strategy Session**

Transcription: Tape 13

Forums that facilitate a connection between recruiters and students can be accomplished by presenting what recruiters needs are and what kinds of skill sets are required for students. In addition, one of the goals of the Society for Information Management, which is comprised of senior executives and academics, is to create partnerships between faculty and practitioners where they work collaboratively. There are several mechanisms to facilitate that, including monthly chapter meetings and monthly meetings with topics for discussion where primarily practitioners present, but academics are invited to attend. But the incentive system does not support faculty attending.
It would be useful to track where professors’ students end up in industry. This could provide a way to measure success. Unfortunately, many professors do not have industry experience and cannot teach what is relevant to industry.

Boeing has a program where they have research scholarships for new faculty members. They have projects and they seek proposals. As a result, faculty members come on-site and gain knowledge of how industry works. Faculty teams up with a practitioner and they jointly work on a project. This is a great model, but it needs to be more widespread.

How do those partnerships between faculty and industry groups form? Is it a self-selection process or is there a way to promote that? Boeing advertises, sending an email to their contacts in the various universities, who in turn send it to their faculty who will respond. Another model is best-practices research where practitioners will decide what kind of research they want done and then they send out a request for proposal on I.S. Word, which is a pretty huge distribution list.

How can this group encourage these partnerships? One way is to mandate that interaction with practice becomes part of the incentive system.

A large challenge is attracting and retaining students, and students are more likely to stay if they see the relevance. A program at Norfolk State had a half-credit freshman class where they met all these industry people and at the end wrote what they wanted their resume to look like when they graduated from college. Then they would meet with mentors who would tell them what they needed to learn in order to go into the mentor’s field. This is one option, or else people in the career center or faculty advisors need to be aware of the skills that industry needs. Right now, graduates go through the academic process and then have to learn things all over again when they are thrown into industry. So we need to get them working together better.

People in university career centers need to work with practitioners in order to understand the skills needed for different fields. Recruiters have an incentive to help with this information so that they can get good people. Some departments are very active here, but others are passive.

So far, these discussions have been around “bottom-up” solutions by faculty and practitioners. Is there anything that high-level leadership can do in both academia and industry? Academic work involves three things: research, teaching, and service. Unfortunately, this falls into the service category, and there is no incentive for doing service. There should be a system that should encourage how schools are involved from the acquisition of students, to nurturing the talent, to placement.

It would be useful to for freshmen to hear what it is like in industry from a variety of people working in computing. They may discover that they may want to be in marketing and needing to understand IT, but not needing to program. Similarly, they might be good at planning, for example, and be a good project manager. Just because a student is not a
good programmer doesn’t mean they don’t have a place in the field of technology. Hearing about this from practitioners could make the major attractive to them.

A related subject is how to get industry to recruit from a larger variety of schools, and not just the ones that are top-ranked (R1). A lot of smaller schools are more open to trying something new and different. This is an area where organizations like ACM and IEEE could push their industry members to pay attention to smaller schools.

In addition, the rankings for schools should be questioned. Some in the top 20 don’t even have an IS program. A potential action item is to have ACM, IEEE, and similar organizations come up with a ranking system based on where students get placed, how much effort is put into their career involvement, how well universities work with different industry partners in terms of internships, and mentoring, etc. This would be a much more accurate ranking than U.S. News and World Report, which is based on research dollars.

At a minimum, an action item would be to at least show how these rankings are coming about and how they may not reflect true abilities. For instance, if a person is graduating from the University of Central Florida, from the IS program, now that doesn’t mean that the graduate is always less than a graduate from Stanford or Harvard because those programs do not have IS programs. But this would be a hard battle to win. Perhaps using scoring instead of rankings would be easier.

Are there models of partnerships that have gone beyond just recruitment and employment and that involve more of a strategic alliance where resources are being shared? Unfortunately, this brings up issues of academic integrity. However, there are work-arounds, such as using consortia of companies. Still, in smaller schools where funding can be more difficult, it can be harder to hold onto integrity than with an R1 school that has more money.

With improved industry/academic collaboration, you could get the excellent theories and research “translated” from research journals into a form that industry can understand. Hooking up engineering students and business school students has been a useful way to make this connection.

Focusing now on challenges, a challenge is to get students from the universities into industry in a meaningful way. In other words, you want the students to hit the ground running so you want to equip them with both conceptual skills as well as tools which will help them and reduce the initial learning costs that companies have to incur. Also, how do we get industry involved to show applicability to the students, and how do we get the students to show up to listen?

Schools would like to see career-related programs customized, but faculty do not have the time for helping 30 to 80 students with their resumes. So we need to encourage mentoring programs. A number of companies will let people take every fourth Friday off to work with high school students. We need to provide companies with information on how to
help. A lot of departments have industry advisory boards that help with things like this, for example, providing a resource pool for class speakers. The incentive system needs to support activities like this. Also, if universities have career activities, they need to train industry people before they talk. Sometimes having them talk can do more harm than good.

Another challenge is how do we get industry to examine school rankings and recognize its limitations?

Collaboration needs to occur on multiple levels: placement, research, curriculum development/skill sets, and translating research from academic journals to practitioners.

Focusing on strategies, one strategy is to have professional organizations alter, modify, or propose a new ranking system. Ranking should be based on skills taught, skills developed, and real-world project experience.

NSF funding of these kinds of activities is another action item. This would add credibility and provide leverage. This funding needs to be enough ($3 million would be small to run an entire program.)

Other strategies and action items: Create joint practitioners/academic conferences where there are incentives for both to attend. Build on successful models for collaboration, such as SIM and Boeing. Get industry people to mentor students and advise faculty, and provide guidance for the industry people. Create more courses where students learn about the skills they need from practitioners. Team-teach classes with teachers from academia and industry. Encourage cross-functional collaborations between computer sciences and business schools. Have universities provide equal credit for faculty to attend practitioner conferences as academic conferences. Involve typically overlooked schools, such as community colleges and minority serving institutions (MSIs).

**Cross-Computing Strategy Session**

Transcription: Tape 18

One problem is that there is no unifying way to make cross-computing interaction happen. It tends to be one individual working with another individual. An example is the cross-computing curriculum. AITP is an organization that was involved in IS curriculum development in the IS 2002 version, but is not involved in the latest revisions. There is a dispute over whether they should have been involved, raising the issue of who are the right organizations to be involved.

The goal of this session is to specify what the strategy means, covering as many challenges as possible, building explicit links between strategies and challenges, and starting to identify possible action items. Three elements to start with are defining the space of computing, identifying computing, and joint activities to promote computing. Note that defining computing is a joint activity.
Who are the stakeholders in this problem? Perhaps it’s not the organizations, but the disciplines. For example, information systems as a sub-discipline of computing is more important than AIS or SIGMIS within ACM. Once you have the computing disciplines named, then you can choose who to represent them.

Three tasks identified are identifying the disciplines, identifying the organizations that represent them, and listing possibly activities these organizations would do.

What about strategies that are global, meaning that they apply to all organizations? These include life-long learning, professional issues, curriculum, diversity, public relations, and resource sharing. Certification and accreditation have some global aspects, and some that are specific to disciplines.

Resource sharing can involve ideas, media, and events. The SGB (SIG chairs that meet together) encourages groups with education events to collaborate. The IEEE educational activities site doesn’t link to all relevant education conferences, but it’s easier to find information than on the equivalent ACM site. Joint efforts of research creation are also desirable. A sense of competition gets in the way of this, especially if an organization feels that it is large enough that they don’t need the sharing and publicity. Smaller organizations like AIS are so financially strapped that they may not feel that it can contribute.

ACM has a policy on recognizing accepted curriculum models, although they do not include AIS. An example is the curriculum model CS2008. It’s been there for about ten years, and so is a little out of date. A sub-committee is working with the education board on this; this effort should include other organizations.

It would be useful if all this information were aggregated so that people didn’t have to search through several organizations. This doesn’t necessarily mean building a bureaucracy, but there needs to be some organization of players, perhaps like the old IFIP (federation of societies). So if the education board or council is successful, where would the information be housed? (Note that the difference between the education board and council is that one has 40 members and the other has 10.)

IEEE does not have an education SIG. They have an education board which is responsible for all education and training activities, and they have a professional activities board which is responsible for certification, professional support other than training, etc. Anything related to policy goes to the board of governors. So ACM and IEEE do not have the same structure, but they do not need to in order to share resources.

Certain things can be handled more easily than others through an umbrella organization. Information on conferences, events, and resources are examples of what could work well. NSF could fund some subset or group to at least create a Web presence that pointed to all the groups and the common sets of resources available. This is different from Boot’s project because her project focuses on instructional material. This would encompass a
calendar of events, curricular material, organizational descriptions, possibly job boards. This approach is a mechanism that’s needed and it can be part of a strategy that gets people to start collaborating over common interests. A funding source to pay people to do the legwork would keep it going. It’s important to have funding, even if it’s at a low level. An example of a project that was done without funding is ACM’s Computing Careers site. There was an initial level of interest that was sustained for about nine months. Then it became an effort for one or two people and then that died off because there was really no structure in place to support it. Now the site has been there in the same form probably for the past two years.

Would it be better to have NSF create a center for computing education, like NCWIT, or would it be better to just organize the efforts that already exist? Some people feel the latter would be better. One strategy is to build up, starting with a site that supports cross-cutting collaboration issues, followed by a simple organization of one or two organizations who set down a strategic plan of how they are going to cooperate, what are the benefits going to be, etc. Once they have a proof of concept and some accomplishments, then they come up with a more ambitious plan and get more funding.

Looking at a sustainably funded effort to just to communicate, how would it be funded? Normally, NSF wants to back off and find other sources of funding that will be perpetual. Perhaps over time it would have to be evaluated and individual societies would have to determine whether or not it were beneficial enough to them to fund. NSF is unable to commit funding indefinitely. There should be some level of volunteerism to make it sustainable. A ramp up time of two or three years would be expected to show how it adds value. Another idea: it could have a public and a private space for the organizations to have dialogs.

Another effort to look at is the Image of Computing. It started seven years ago as a place to share resources, but its mission changed. Now it doesn’t exist. Some say that the effort was led by public relations people without having the stakeholders that are practitioners. On the other side, we have the SIGCSE website which has information, but is visually uninteresting. So ideally you have visual styles from people who are good at that, but content from stakeholders.

Someone who wants to study computing education would have a hard time finding out about all the available programs, especially all the different majors. There are over 2,000 computer science programs alone. So a central site with all that information would be useful. That was one of the goals for the ACM computing careers sites, but it was too big for a volunteer crew to do.

Renee McCauley used to do an annual survey of what different programs were offering, which was very helpful in curriculum design, but that’s a research project. A Taulbee-type survey for undergraduate and state programs would also be very useful.

Focusing on strategies, there is defining which sub-disciplines, organizations, and possible joint actions would be involved. Other strategies include: creation and sharing of
resources, and the creation and sharing of tools for assessment. Another is to identify the top five Grand Challenges in computing education and perhaps each year focus on one of them for the year; for example, these could be focus of joint meetings. Each institution could create its own editorial take on the Grand Challenges. These challenges would include the image of computing, the content of computing, and the resources for computing. It would be fascinating if it there was an opportunity for the organizations to report back to each other and share experiences regarding what they did and what was achieved.

Another challenge involves rewards for faculty advancement in education. Some universities are significantly reducing their teaching faculty, although others are doing the opposite. How do we make this interesting enough for administrators to buy in?

Funding will be needed for people to spearhead the incentive and faculty rewards. Maybe the responsibilities can rotate so that responsibility is shared. Funding would significantly increase the likelihood of success and sustained effort, and would make it easier to sell to the deans. A unified proposal from all the groups at this summit would have a powerful chance at getting funding from NSF.

Computer scientists become very enmeshed in their problems and they forget to celebrate success. We should have an annual, very public, formal recognition of several levels of success towards whatever goal is being focused on. One idea is giving awards for young or mid-career educator. They need to be significant awards, not just a laminated plaque.

CPATH attempted to recognize science distinguished education, but it was fraught with problems. There is the SIGCSE award for research and education.

Faculty interacting with industry is another area that is not currently rewarded, such as bringing in industry practices, having industry advisory boards, and getting support for student internships. High school teachers can be recognized and celebrated as well. However, getting success stories could be difficult, as illustrated by the CRA website that was supposed to show highlights of exciting projects. NSF also has trouble getting real highlights beyond descriptions of events where people came and got excited. Computer scientists need to be trained in PR. Are IT people more savvy in this area?

We need to show engagement with industry, somehow showing collaboration with large partners across disciplines. This is what large companies are looking for – solutions to significant problems. Sounds like a Grand Challenge.

**Public Relations Strategy Session**

Transcription: Public Relations Strategy.docx

The key issue is to be solved is why there’s a shortage of people going into computing. We need to attract a more diverse, bigger audience that includes not only traditional
higher education students, but also K-12 and people who return to the university (another
degree, returning from the military, etc.).

One challenge that was identified earlier is that society doesn’t understand what
computing is about, and has the image of a solitary male in a room hacking away. Society
doesn’t see the excitement of designing mobile technologies, but sees the problems when
computers cause accidents, even though human error is much more common. Young
people use technology, but it’s often invisible, so they are unaware of it.

The job of public relations here is crafting controlling, managing and disseminating the
messaging and imaging of computing as a career and a lifestyle. To do this, you need to
understand what influences decisions that young people are making. There’s a National
Academy of Engineering study that studied similar issues, and it talks about identifying
the messages. (Changing the Conversation). There were action items around focusing on
key messages and messages that talk about the impact on lives, society and humanity,
which young people are very interested in. The messaging should not focus on what’s
hard in a field. People were saying, “Engineering is really hard but you can do it,” and the
young people were not motivated by doing something that was hard. Girls, especially,
might not be able to think of themselves as someone who is good at doing something
hard.

Is it better to target the public as a mass, or to separate out K-12 as a different target? K-
12 students are definitely affected by the mass culture. CSTA works with computer
science teachers, but is up against the mass culture, as well as school boards,
superintendents, and state departments who aren’t listening. School boards approve
curriculum for school districts, and they report to the states. Individual states are the ones
who set what graduation requirements. Some states, such as Massachusetts, are
discussing developing K-12 engineering, technology, computer class standards within the
curriculum. How can we help that effort? In England, computing is part of a national
curriculum, but it’s taught badly, so it hasn’t been effective; it has focused on office skills
like using spreadsheets.

Often the problem is that the computer science teachers are math teachers who have gone
through some retraining and do not have the enthusiasm for technology. Also, in the
United States, teachers have very little flexibility in terms of topics they can teach
because they’re governed by the state-approved curricula. This is further complicated by
the fact that every state is slightly different, although there is certainly some
commonality.

CSTA is working with teachers on improving perception, trying to get colleagues within
that school district to also be excited about computer science. One of the teachers in the
CSTA cohort leadership initiative works to educate kids and their parents that computing
permeates all different fields. He hands out a flyer that lists all the majors at the local
universities that require at least one computer science course. His name is Stacey
Armstrong, and he has a website: http://apluscompsci.com/.
The CSTA cohort leadership group picks two teachers from each state to be the ringleaders to bring together all the teachers that are teaching computer science in that state. They know more about their state legislation, their laws, and whatever was going on with computer science education in that state. The goal was to bring them together, perhaps create a chapter, and compile resources. Wouldn’t the national math and science teacher organizations also be using computing somehow?

Imagine interviewing a student and asking, “Do you know what you want to do when you grow up? Do you like computer science?” and they say, “No, I don’t because and so on. I’d like to be an historian.” And they go to a real historian who describes how they use computers to analyze things so. Or they’d like to be an airline pilot and they talk about navigation and control systems and the student finds out at the end that there’s no job that doesn’t accept computers.

There needs to be a place where people can find public relations resources that will help with perception, similar to how NCWIT provides resources.

There are advertising campaigns by Apple and Microsoft that say “I’m a Mac” or “I’m a PC.” What about one that says “I’m a computer scientist” and talks about what they do? But the term “computer scientist” can be a turn off. It’s a problem that we don’t have a good name for what we do. People might accept “computer scientist” if they understood that it is not the same as programming, but that’s a hard problem to solve.

Names need to reflect the changing field. Originally, engineering was called mechanical arts, but it’s not longer just mechanics. Computing now encompasses computer engineers, software engineers, information scientists, and information technology.

Faculty needs to convey this information to first year students, and they typically start with programming. They need to explain that you need to learn programming in order to create things. It’s like learning reading. However, that may be changing as more first years are being taught Intro to Computing Theory first. At some universities, there are computer-related subjects that have no programming. For example, there’s a computer games degree where most of the work is on subject like sociological areas, such as how do you represent a character’s animation.

University of Washington produced videos that illustrated how diverse computing has become, talking to recent women graduates. (http://www.cs.washington.edu/WhyCSE). When universities are talking about a drop off in enrollment, are they talking about conventional computing? Are the more creative areas increasing?

CSTA’s experience with girls at the high school level shows that they want to be more well-rounded and experience a lot of different types of classes. So although they liked programming, they didn’t want to continue on to AP computer classes because they wanted to try other things. A lot of women in engineering want to do work that has a positive social impact. Some boys may be in the field just so they can hide out from people, but in general, both boys and girls want to have social impact.
Most students going to college have their majors picked out, but most of them change them when they get there. It varies whether teachers talk about careers and majors. Schools are really not going to be able to take on the entire public relations responsibility. Summer camps can also help because they provide hands-on experience. But teachers could be helped with talking points resources, much like NCWIT provides.

Partnerships are critical. Partnerships between K-12 and universities depend a great deal on whether universities are physically nearby. It’s similar for industry, although industry has a field organizations that could be local, even if their headquarters are not. Can we encourage partnerships with the local press? ACM SIGCSE did a special issue on women in computing, and it had good articles, but it was just a onetime occurrence. In small communities, organizations like Rotary Clubs can be a big deal. These kinds of partnerships are another opportunity for a centralized resource, containing presentations and templates for presentations.

To summarize, public relations is about crafting, controlling, managing, and disseminating the message about the image of computing. It’s also about creating resources, such as a central database of templates, speakers, talking point cards, etc. Target audiences for K-12 include students, teachers, school boards, state departments of education, parents, and national math and science teacher organizations. Young people should be getting the same message from multiple sources, especially if it’s not their parents. The media technology should make use of technologies that young people use (social media, etc.) The message needs to be repeated to be effective.

Diversity and curriculum have not been discussed yet. It seems like the first job is to get the message out, and then focus on curriculum. But if curricula are being changed, then there should be public relations to let people know. Also, PR can be used to help people change their curriculum and still meet their state requirements. This needs to be done state by state. Some states, such as Texas and Georgia, are going to be easier.

There can be more PR efforts to make computing more visible. It’s helping in keeping water clean, helping with AIDS research, etc. Perhaps there are celebrities out there who have a math or engineering background who can help. As a historic example, the actress Hedy Lamarr invented a frequency-hopping technology during World War II that is now used in cell phones. (http://www.inventions.org/culture/female/lamarr.html). Similarly, Florence Nightingale applied scientific methods to nursing. Many of these stories are told in a book called Her Story: A Timeline of the Women Who Changed America by Charlotte S. Waisman, Jill S. Tietjjen.

**Changing the University Strategy Session**

Transcription: Changing University Strategy.doc

There are two ways of changing universities: one is by creating actions at individual universities, and the other is trying to implement policies that could apply across universities, such as a serious computing general education requirement.
One problem with a computing general education requirement is that there aren’t enough computers for all the students. Even though more students have laptops these days, not all of them do, even though $400 laptops are now available.

The issue of a general education requirement should possibly be the subject of a study produced by a panel of professional societies, similar to how the curriculum is developed for undergraduate schools. Many universities have a core curriculum which is supposed to be spread across all the majors, and this could be used as a model. Politically, it would be very advantageous if there were a national professional society recommendation. Even stronger would be to involve industry.

From a branding perspective, it makes sense not to call it a computer science requirement, because that ties it to a department. Science requirements are not tied to individual departments. In fact, computing classes don’t necessarily need to be taught by computer science faculty, if the faculty were supported by NSF.

 Universities respond to accreditation. This can be a way to make universities prioritize teaching. However, using accreditation in this way can backfire if universities decide to shut down programs rather than do what’s required for accreditation.

Another approach is to offer models for how these kinds of programs can be done. You have to start small, helping multiple schools establish successful programs, and then promulgating the most successful approaches. Schools are too diverse to have one approach work for all schools. This would apply to computing general education as well as other issues such as restructuring of curricula. Georgia Tech’s Threads model is a good one to look at; although it’s still early, it is showing increased interest by students and a higher percentage of women than typical. Note that this model requires quite a bit of human infrastructure to support. Other universities (Stanford, CMU, Princeton, etc.) are also trying new approaches.

Business schools are locked into the ASEE model, but the rules have become more relaxed. For example, they are no longer requiring a 50%/50% model for classes outside of the business school. But program requirements are pretty much the same, so there might be a slight move down to 45% or 40%, but it’s not going to be overwhelming.

On the subject of tenure, California state schools now have two tracks, one for teaching and one for research. This may be part of a trend to eliminate tenure altogether. (Some say that European academia does not have tenure, although this depends on how you define it.) The measurement and research group could suggest that this is something to be investigated to see if it is successful.

Revising curriculum and pedagogy is another issue. Part of the problem is that computer science departments are behind the times in technology and how they are used in the classroom. Another pedagogy idea is to use accreditation. For example, nursing educators in the UK require accreditation that demonstrates continuing professional education and
that could be in teaching, or in application areas, etc. The Computing Accreditation Commission of ABED has similar requirements.

For pedagogy, we have to get faculty away from the lecture plus lab model. Most research universities use the German model from the 19th century. Using novel approaches are not rewarded. (Service is especially not rewarded.) Some schools, such as Villanova, reward teaching more, although less so than they have in the past. There is an issue with focusing on teaching, in that you need to give the faculty enough time to stay on top of the latest technology; Swarthmore is an example where they only allowed time for this every 2 ½ years. In the United States, there are more teaching universities than research universities. Another approach to the research/teaching conflict is to have research around pedagogy and then publish results on that research.

In order to have changes actually occur, you might need an established organization at the university, such as a teaching and learning center, where faculty can go and learn how to teach with these newer approaches. The universities would need to strongly encourage faculty to participate. Universities are strongly motivated by money, so they would respond to an organization like NSF rewarding faculty who prioritize teaching with healthy-sized grants.

Would it be reasonable to award grants to deans or department chairs for teaching? The general opinion is that this is a good idea, but unlikely to happen. Some universities have pots of money for faculty development, but they tend to go to humanities and social sciences, because they have so little money themselves. There is a discussion of whether grant overhead could be used to fund teaching changes, but some feel this is against federal regulations. However, on further discussion, it could be written into a grant in the form of a check box on the grant application that says, “I’m willing to try new teaching innovations if you can give me X amount of dollars under this line item.” This is like an educational version of an RUI supplement, but bigger.

Other programs, such as CPATH exist for this kind of potential funding, but although they are good ideas, they are not big enough. A special incentive program that cuts across size and education tenure tracks would be desirable.

The idea of treating computing as a broad discipline like “science” (which covers biology, chemistry, and physics) is brought up again. Humanities majors take a science course – they don’t need to take courses in all of biology, chemistry, and physics. Brown has done well because it threw out core requirements altogether. But many students need a core curriculum. You can make a good case that computing should be part of this core curriculum. There may be some opposition from math departments, since many math departments support their graduate students through T.A. positions; they would object if we were taking away students from them. But computers science departments could follow that same model.

Of the action items identified so far, which could be taken on by the organizations represented here? One is a white paper on curriculum changes. This is focusing on
problems such as defining an introductory sequence, determining what math should be like, figuring out how to deconstruct the overly-demanding graduate requirements in most computer science degrees, and determining the needs for the computation thinking for people in natural sciences. CRA-E will tackle a larger agenda once this first report is released, hopefully within the year. One good focus for the committee is finding exemplars that show early promise. Georgia Tech’s Threads program is an example. It’s different from having a core plus tracks, as is required in electrical engineering, since the field is so wide.

AIS is focusing on curriculum, working on improving the pedagogy links on the AIS site, but it is difficult to find anyone to help. Unlike ACM, there are few practitioners in AIS. The idea is for the website to have syllabi, cases, databases, and ideas of what to try in the classroom. There is also a conference strictly devoted to pedagogy, and the proceedings will be in the AIS library.

ASEE just had their national meeting in Austin two weeks ago and they did have a plenary session. There were several talking points put out, and the most popular one was the idea of changing the orientation towards teaching and its importance. So the ASEE board is very interested in it and feels that it can be accomplished through accreditation.

IEEE CS is not doing much differently because of the financial downturn right now. The normal activities include accreditation, curriculum projects, and professional development. The IEEE education society plays a big role in this arena, but it is more electrical engineering than computer science. This is reasonable, since it’s IEEE, but the computer society is the largest society in IEEE. But the educational activities board is completely separate from the computer society. IEEE also has standards, which eventually affect education, as the standards are taught.

ACM has an education board and an education council. The board had grown to an unwieldy group of 30 people, so it became the council, and the board is now a group of 10. There is one chair for both groups, and they interact quite a bit. The education board is responsible for getting things done, and that includes forming a curriculum committee, forming taskforces for specific items that report back when a job’s done, etc. The exact nature of the council is still under development. For the council, what seems to work best is to have panel presentations, such as panels from industry, graduates, etc. This resulted in discussions in the council which generated action items for the board. ACM also collaborates very closely on curriculum issues with the IEEE computer society and AIS.

Could the government play a role? There is skepticism, since we wouldn’t want something like “No Child Left Behind” to be extended to the college level.

There is a discussion of how the field is changing so rapidly, it is hard to keep up. Also, one of the dominant questions right now is that there are so many variations emerging in curriculum recommendations and you can’t restructure for each of these variations. Curriculum development projects have been very labor intensive, involving large committees and many meetings. The last ACM version was 2001, although a 2008
version will be released soon. Software engineering was last updated in 2004. The ACM IT document is more focused on outcomes.

Another model to look at: the Learning Federation was an attempt in 2000 to form a large project to make progress in new ways of teaching and pedagogy using tools such as visualization and simulation. It created a research roadmap, and unfortunately, in the economic meltdown of 2000, it was hard to get anyone to think big. This was done as a partnership with Henry Kelly of the Federation of American Scientists and Randy Hinrichs, who was a Microsoft Researcher.

Business schools often believe that information systems is spreadsheets or simple databases like Microsoft Access. Data mining is a better thing to look at from a computing point of view, and some business schools are doing that. They are not developing algorithms, but rather using existing ones. This is fine, as long as students have an understanding of the algorithm and therefore the merit of the data they get back. Overreliance on data produced by bad models has been largely responsible for our current economic meltdown.

**Changing the University Action Items Session**

Transcription: Tape 23

In some schools, if students can’t do math, they put them in computing. But in others, they put them in a high school math class called “College Algebra”. In a nursing school, the nurses can take a one credit course called “Math for Nurses” which covers fractions and dose levels, and this one credit satisfies the math requirement.

What are some action items? Improve accreditation. Include pedagogy in faculty development. (Side note: Should math be taught by engineering faculty or by the math department? There is some disagreement here.) Gather data on experience with dual track tenure lines for research and teaching. (Side note: do we need to look beyond tenure tracks? Is tenure going to be going away for most faculty?) Make sure computing is an essential part of general education.

A model to look at is for science where in some places they have A, B, and C courses. For example, A might be a classical physics course, B might be an environmental science course, and C might be an earth science course. Different majors require different levels. Perhaps this could be applied to computing. Another model is an MBA program, where there is on an IS course in the core, but students are required to take one of three “selectives”, and the IS course that is one of the selectives.

An action item could be getting the societies to strongly recommend putting computing in general education and accompanying that with specific curriculum recommendations. Information should be gathered on what is currently being done that has been successful. Then you could analyze the successes and look for common characteristics of good
computing courses. Unfortunately, societies making recommendations will not have enough impact. This is where new accreditation could play a role.

There is a general recognition by the national accrediting bodies that people need to understand science, but there is not the same recognition that everyone needs to understand computers to some degree. This mindset needs to change. As an example, one school’s core requirements include two semesters of science and two semesters of math or computing, of which one must be math-based. This means that students only choose computing if the courses are attractive enough, such as computing with images, computing and the web, etc. The computing with images class is attractive because at the end, each student produces a poster which they can take back to their dorm and hang on the wall. Friends think it’s cool, and it makes for effective outreach.

Technical schools can require computing electives, but non-technical schools, such as business schools, need to be advised. We don’t want simply computer literacy, which is using computer applications productively; we want computational thinking. There is some gray area: one school teaches a class where students have to find a problem in an existing company or come up with a new problem and solve it. You could say that there is some computation, but it depends on how you define it. There is a publicity event, like a “bake off” of the students’ projects that is effective for outreach.

One school has something called “Science Day”, where undergraduates create posters in biology, geology, etc. Perhaps we should have a “Computing Day”? One college has their computer science department participate in their Science Day. But what about computing that is not computer science?

An action item can be doing a survey of exemplars for successful courses. This could be a resource for those who want to influence universities. It would be called a survey of existing computational thinking courses for general education. An example would be an MIS course which has a popular lab component where they do advanced Excel and database work and some Web design. The lecture part focuses on technology in the business environment. Students have done some interesting things, including going out into the university and finding things they want to fix.

Another issue is that technology classes often use substandard technology. An action item could be to make some positive recommendations as to what universities should do to enhance the use of advanced instructional techniques in classrooms and other spaces.

Still, associations are limited in how much impact they can have through recommendations. What if there were resources to establish some sort of recognition or award for innovations in pedagogy that are applied in computing classrooms? ACM currently only has an award for outstanding contributions to computer science education. Other organizations have similar awards, but they are more for long term contributions. Which meetings would be appropriate for recognizing pedagogy? The award could be based not on the best paper, but on the best educational software system, which could be distributed by CD. Universities like these awards because they are good publicity.
universities have centers for teaching and learning that could be a place to work out pedagogical ideas. Some schools have an instructional teaching lab where you can have your class in to get videotaped, and they can help with cost design or educational background; they are an underused resource.

There are some conferences on educational development, such as ICED and the Professional Organizational Development (POD) network.

Research, such as measurement and evaluation, would be valuable, but it needs to be done by social scientists, not computing people. The only way that CRA is doing an education project is by creating a separate group of people and funding them to do it. If money were available, more research could be done. The research should be short-term, mission-oriented and applied. So an action item could be sponsorship for this kind of research.

Some universities have advisory boards or groups from industry, but many do not. ACM could sponsor a study to find out from a well-represented group of industries what they would like to see universities do better. (Unfortunately, many departments have no interest in having anyone outside tell them what to put in their curriculum.) SIM does something like this that could be looked at as a model – Jerry Lufkin does a survey. When this kind of survey has been done before, the advisory boards typically say that the students’ technical abilities are fine, but they need to develop teamwork and communication skills. It would be useful to make them elaborate. We don’t want information at too low a level to find the systematic holes in students’ education. We could ask them to look 10 years into the future, but it’s not clear that industry knows what it will need in 10 years.

Europe has had an effort here where they created a skill framework for the general IT arena. The British had the first effort, which was a collaboration between the British computer society, government, and industry. This was followed by French and German efforts and a European community meta-framework. However, the presentation on it that Boots heard was oriented towards business, management, and company finances, and had hardly any technical content. This is an issue with US industry as well, where there is a belief that people would do technical work for a few years and then move into management. Industry really wants project management now.

Another example is a US school where industry was brought in to help with a curriculum. There was a disconnect between what the recruiters believed students needed and what the managers believed. Also different types of industry groups will have different needs, such as an IT-oriented group, a software development application house, and a national laboratory. A focus group might be more effective than a survey. We need to ask what they are looking for in the next couple of years and in the next 10 years, and how do they see job categories changing?

Having industry people talk to students can help dispel the myth that computing jobs involve geeks sitting in a dark room programming 10 to 12 hours a day. Students need to
understand the variety of roles in industry. An example is a software engineer at BP who
got to go out to a glacier to understand the sensors that BP was using. One survey should
that kids understood that IT was important to their lives and they like using it, but when
asked to describe who creates the products, they gave the usual stereotypes, and felt that
people who majored in IT were not the kind of person that they were.

Another question to ask industry is, “What do you want from employees who get a
general education?” Europe has an effort to develop skills requirements, such as word
processing, databases, spreadsheets, or even how to use an ATM, how to use a Smartcard
for the metro rail. It’s a 38 page syllabus. But this kind of effort is not popular in the U.S.

**Community Collaboration and Policy Action Items Session**

Transcription: Cross-Community.doc

Consider forming a center for computing education. How would we get that, and would
we give it to NSF to fund? Would people object to the word “computing”? In general,
people believe that “computer science” is more of an issue than “computing”.

An objection is raised to starting a new organization, suggesting that we do not need
another top-down organization, but acknowledging that an umbrella organization might
work another way. The organization could be structured to help cross-collaboration, or it
could be structured so that it subsumes existing organizations. One suggestion is that the
focus should be to solve the big challenges, like creating 10,000 new computer science
teachers. There is currently no infrastructure to support that.

Another possible structure would be a clearinghouse of information. NCWIT got started
as a clearinghouse in the area of women in computing, and there used to be one for
computer education a while ago, perhaps called NCIC. It took about four years for
NCWIT to get running.

One model where organizations are working together is CSAP, which has membership
from ACM, IEEE, and, until recently AIS. CSIM doesn’t have computer engineering.

The iSchools consist of 25 schools. They don’t all go to one conference, and there is no
one single accreditation agency in common. In fact, some are not accredited by any
agency. If the American Society of Information were to offer some kind of accreditation
scheme, 95% of the schools would not be interested because they have alternatives, but
some schools would like it since they are not currently accredited. So it’s dangerous to
make assumptions on what all the schools have in common. Perhaps there is a more
grassroots way to create opportunities or incentives for common goals to be realized,
because that will happen sooner than organizing a super-structure that needs funding.

The essential question is: would a new organization be an additional layer of bureaucracy
or would it genuinely be a mechanism with which we could signal to the rest of the world
that we are working together? There are a unified set of concerns, issues, and
opportunities that would benefit from having a central voice. It could be a powerful way to tell a strong story. The story would cover issues discussed in this summit: sharing resources, diversity issues, how to change pedagogy, etc.

NCWIT started in a similar fashion, building community, sharing resources, and creating a unified front. It has some definite advantages to it, but there are difficulties because everyone has different needs and a different perspective. Still, that does not negate the advantages. What they are seeing is that once you build a community structure, then people’s behavior changes such that they aren’t reinventing resources but are coming to a central location for them. Lucy says this is the hardest job she’s ever done because it requires a lot of leadership and a lot of infrastructure to support everyone else in their jobs. So it’s not merely a matter of making connections.

Changing the image of computing is a lot of work, and one could argue that it’s made more difficult if each organization takes a different tack on it. Conversely, one could argue that taking multiple approaches is required to successfully change the image. A variety of skills is needed, and may require help from departments as diverse as information science, social science, psychology, and education.

Mixing social scientists and computing professionals is not easy. It takes a lot of work to get to the point where they can have meaningful conversations. This is not the kind of collaboration that starts up immediately; it requires the nurturing of human relationships and negotiation. It helps to have very clear goals – this was advantageous for CSAP. Four goals for this possible organization have already been mentioned: changing the image of computing, addressing diversity, improving industry/university collaboration, and improving the perceived value of education within universities. The transition from high school to computing major is another potential goal.

This organization wouldn’t be the only organization addressing these issues, but several organizations could use it to join forces and share best practices and resources. This is different from SIGSCE, which focuses on curriculum. Could SIGSCE’s mission be expanded in order to become this new organization? SIGSCE is part of ACM and focused on computer science. IS people are more likely to come to a new organization.

Individual schools can make progress on many of these goals (addressing diversity, improving industry/university collaboration, and improving the perceived value of education), but cannot do much by themselves to change the image of computing. However, a central organization can make those first goals easy to achieve by providing resources.

Such a central organization can help with diversity. However, if the organization gets too diverse it can be a problem. NCWIT has so many players that it is now looking at ways to separate them into different roles. This new center would need to have a very clear focus.

The center’s focus would cover several areas: centralized web resources, grand challenges, celebration of achievements, and government policy. In addition, one of the
center’s functions could be research. We would want to solicit information on what kinds of research the organizations are interested in. Would organizations then bid on the funding to do that research?

An example of policy work is messaging about K-12 education policy around computing and diversity of thought. ACM has a new K-12 education policy committee, but CRA does nothing with K-12. ASEE works with K-12, and is discussing how to have partnerships that can be effective around government policy. A government roundtable with ACM, CRA, and ASEE brought in government relations people from large corporations and explained to them about computing education and why it was important. The center could create meetings like this.

Math teachers are good when it comes to influencing government policy, and CSTA would like to do this, but it would be more effective if there was a collective policy voice. Aviation safety is another example where this is done.

In fact, CRA is a model for this center. Their activities include a workshop to help new faculty in their careers, to get more women into computing. But CRA focuses on research and not on education, so it’s not the same organization. This centralized approach is good for policy, because Congress hates it when they are presented opposing views because they have no way to judge who is right because they don’t have the technical competence. Astronomers are another example: they have their disagreements in private and then present one view publicly.

In this way, the organization is more like an umbrella organization than like a center. This brings up the issue of who would be in charge. One model is that Coalition to Diversify Computing, which is a joint effort of ACM, IEEE computer society, and CRA and it’s quasi-independent and responsible to all three organizations. NCWIT is structured in a similar fashion.

IT and IS organizations have not been involved in these efforts, however. (This is starting to change with NCWIT, which is putting CIOs on its board, but still does not have IS business schools.) In general, there has not been good communication and involvement with IS organizations such as AIS. This must be rectified if we are to have a common voice. (An example of where not having a common voice is a problem: the supercomputer centers have an organization and they approach Congress and present the message to put money into supercomputing, which is not what the rest of the computing societies want, and this leaves congress confused.) So the center must not be run just by the big players.

NCWIT is an organization of organizations, and people want to volunteer their time as individuals, which was unexpected. So they send them to their member organizations. In other words, the center could provide a way to increase membership to individual organizations.
There needs to be a very clear goal, or at least a prioritization of goals, or a central theme. Diversity could be this theme, or creating a 21st century work force. Quality computing education is a good central theme, but it is not self-explanatory. Diversity is more self-explanatory, but even that can be unclear as to what kind of diversity is needed. And diversity is important, but there are other factors that are also important – is there a common thread? Diversity involves two issues: fairness and the fact that having diversity can make computing better. Industry is more influenced by the latter argument. Most industry websites have something about diversity, but they don’t know what they are doing. The center could provide a way to help them spend their money effectively to improve diversity.

The workforce could be a common theme. Industry has the money to make a difference, more so than government. Industry understands that if academia increases the number of graduates in computing, that it serves their needs. Increasing enrollment is measurable.

MSRI Math Institute Berkeley is a good model to look at. One opinion is that this is a good model because it doesn’t require organizations massively coordinate.

It’s not easy to define the details of this center in a two hour meeting. Can an action item to be to form a small group and write a proposal with goals? The goal of this meeting was to specify three things: the areas involved, the stakeholders, and the collaborative actions. If we define something and people want to be involved, then it’s a good sign that this is something that people will want.

Is this a center or an umbrella organization? Centers usually imply a physical space and are associated with research. Organizations would be more focused on policy objectives. One opinion is that it would be a mistake to have the identity of the center be focused on research. It can do research, but it shouldn’t be central to its identity.

One possible action item is a white paper, which would give people something to look at and discuss. Another option is a meeting. An analogy: the center is not providing a community for people who don’t have a home, but is providing a neighborhood community center so that people in different homes can learn from each other effectively.

There is still no precise statement of the problem to be solved. Is it lack of unanimity, is it lack of communication, or is it lack of diversity? What is happening with the organizations: are there too many of them, not enough membership, or the curricula is not being used? One of the lessons of NCWIT is that you can over-think this kind of thing, and although that you need to create some structure and alignment up front, a lot of it is invented as you go. There was an attitude of, “Look, I have no idea what I’m doing, but let’s do this.” Things very quickly go beyond gender issues to general computing issues.

There’s a lot of material out there that can be pulled together. An example is projects by Deborah Estrin, which are participatory science projects that use cell phone technology. Kids all across the country taking pictures of bird nests, which creates a huge database of
when birds are hatching all kinds of birds and when they are hatching all over the country. Then the kids can do some analysis with that.

These kinds of projects can have a big impact on the image of computing. The image was much better in the mid to late 90s, but then the bubble burst.

Can the organization speak with one voice for a variety of types of organizations? It would be beneficial to the smaller organizations, especially if it truly represented them. The center could also validate those who have concentrated on education. Education has lacked credibility in the field, but having the center give awards could raise the visibility of it being important. It also feeds into state departments of education making choices about the standards that students have to meet to graduate, and if computing can play a role here.

To wrap up: let’s leave the choice of organization vs. center as a blank. The goal is to speak as a united front for the big five education goals, even though we haven’t defined what they are yet. It should speak to and drive the issues, and the issues should be global. Each year, the center can focus on one of these issues, such as diversity. In addition, it provides an information clearinghouse. A future Computing Education Act would come from the center. Also, it would implement the strategy to create 10,000 K-12 computing teachers. Examples of analogous organizations include astronomers, CRA, NCWIT, and MSRI.

Side note: what appears to be a similar organization is the National Center for Research and Advanced Information and Digital Technologies, also called the “digital promise”. It turns out that it is focused on using technology to teach (TechEd), rather than computing education.

This new center would be the place that is accountable for computing education, since right now no one organization is. A goal is a Computing Education Act of 2012.

**Public Relations Action Items Session**

Transcription: Public Relations Action.doc

Participants: Mark Guzdial, Lecia Barker, Joan Peckham, Stephanie Hoeppner, Tom Hilburn.

It’s important to research what’s an appropriate message for a given audience. For example, if you are talking to a school board, find out what they care about so that your message can persuade them to what you are saying is important. Is this something that individual organizations should do, or should resources be pooled and a marketing firm hired? Real marketers would be great, but there needs to be control over the message so that they don’t mislead people about what computing is really like. Marketing firms cost money, and at least some organizations are having financial difficulties at the moment. So an action item can be: Develop funding models by which we could hire a marketing firm for audiences chosen by the organizations. For example, CSTA could say that its
most important issue is school principals, and organizations such as ACM will help get funding for that. If organizations pooled resources, this could make it less of a financial burden.

Another action item: Identifying audiences that have the greatest potential for making a difference in the shortest time is another action item. Since this is a research item, it would require funding.

CSTA has identified the state board of education as the most critical audience for K-12. The other big proponent for change is an audience that consists of superintendents, principals, and school boards, who all work together to make decisions for the school district. At the college level, it’s the department who is the critical audience.

Another action item: Offering workshops on messaging and talking points to faculty. Talking points will be more cost-effective, and so another action item is to get funding to develop a series of talking points that are relevant to different audiences. So in summary, the items are:

- Identify audiences for greatest impact
- Develop the appropriate message
- Obtain funding for these

An example of why messaging needs to be different: a political audience, such as the state departments of education, will want to hear about impact on economic development for the state, whereas school boards will be interested in hearing something different.

Another action item: Develop and disseminate resources. Note that a common idea that has been brought up is a central area for resources. This will be discussed in the final session where everyone will choose to be part of different resources.

Some of these resources (talking points, research results, etc.) already exist. Another action item is to determine what these are.

Unfortunately, once a message has been developed, a lot of faculty won’t agree with it. Mark sometimes asks people why they think students in a class did badly, and they respond that they don’t belong in computer science. That’s not the appropriate outward facing message to be sending. It may take more than talking points to change people’s opinions. So change the action item to: To develop resources to inform faculty about appropriate messages.

As an example, consider the talking point cards for girls that were developed at NCWIT. The first side of the card is basically to persuade parents that it’s worthwhile to do what we want them to do. It overcomes some misconceptions about whether there’s jobs, about off-shoring, it talks about salaries, and it talks about getting this degree – computing has the highest entry-level salaries in a four-year degree. That came out of research that showed that parents were concerned that they couldn’t put their daughters in school long enough to be a doctor or lawyer. Also, it talks about the nature of the job, that their
daughter will be happy, that she will have money and there will be jobs, and she’ll have the flexibility to go across industries. Once the parents buy into that, then they turn the card over and it describes what you should tell the girl. The same kind of approach can work on undergraduate educators. One of the barriers to persuasion is that there are a lot of faculty who strongly believe that since women don’t do well in computer science they should not be encouraged to go into computer science, despite the fact that the women who leave have a higher grade point average than the men who stay. It’s important to fight these stereotypes using scholarly-based research with the same rigor as technical research. ASEE is encouraging and promoting this.

Sometimes women do better in fields like computing and aviation because the ones who survive there have to work harder. The book Outliers by Malcolm Gladwell discusses some of these issues and provides some explanation about how not wanting to lose face causes mistakes to be made. Men are more likely than women to not want to show weakness. On the other side, women tend to internalize problems as being their fault.

Regarding the strategy of working with local press and local groups: there should be an action item encouraging that. This counteracts the issue that computing is often invisible. For example, you get clean water because water is monitored by computers. There could be a resource of well-written articles to look at. The Rotary Club and school boards are also examples of people who are looking for positive stories.

Again, this comes back to building a repository. The repository might just have links to resources elsewhere, or might contain the resources themselves. The repository would cover messages, elevator speeches, a database of speakers, etc. It would involve finding out what exists already and then filling in the missing pieces.

Another idea is a template for press releases that talk about the importance of computing and could be used when a teacher has done a successful computing project. There could be mini-grants for these kinds of projects for them to develop teaching resources, and then these projects could be written about and sent to the press. An example of a successful program like this is Nerd Girls, which got a lot of press. The grant could be used to develop curriculum or it could be used to get new computers and software. An example for younger students is a digital photo club, as long as it emphasizes the power of computing.

Another idea is to use the mini-grants for competitions, which involve participations and enthusiasm. The disadvantage is that they are labor intensive. A number of competitions already exist. However, competitions often emphasize programming, which is what we are not trying to do, and can emphasize quick solutions as opposed to longer-term team-based projects.

Mark disagrees, and describes the research of one of his students. He says that the people who are doing “X + computing” work are mostly doing quick programming and that’s what gets them excited. Examples are graphic designers who are programming in Photoshop and web designers who write Java PHP. The research involved them sorting
concepts into what was relevant to them versus what they teach in computer science classes. They rated things like team development standards as not related to them. In other words, emphasizing proper practice too much can destroy the fun parts. We want to encourage long-term team-based work as well, but to get people to see why computing is amazing, it’s often the little bit of code that makes the computer do something that gets people excited.

How does public relations relate to professional development and pedagogy? How can we publicize teaching practices such that they get adopted? One idea are newsletters that describe a promising practice, such as pair programming or programming with Alice or Scratch. Ideally the newsletter would just have headlines and a three-line summary; if you click on the summary, you get a PDF summary, and if you click on the summary, you get the full source. However, you want to be sure that there has been real evaluation and assessment on these practices. Educators can be skeptical about adopting something new if it hasn’t been shown to work or scale. Also, research has shown that the enthusiasm and skill of the teacher has more impact than teaching techniques.

Mark has a PhD student studying when computer science teachers adopted something new. Whether the research findings supported the new approach had very little effect – it was much more dependent on whether the teacher liked it. Research results are more likely to prevent adoption. This study worked with undergraduate teachers. K-12 is significantly different, because of requirements coming from the outside, and this will vary significantly depending on the school board.

Another point about proving techniques work: Tom says that the Frontiers in Education (FIE) conference used to have exciting papers about the joys of teaching and now it seems to have gotten into evidence-based research and it’s statistics and comparison and very dull.

Who offers faculty workshops on pedagogy? ASEE does, and ACM does at conferences such as SIGCSE, and IEEE does at the FIE. ASEE has a large workshop called the National Engineering Teaching Institute (NETI). It’s a 3 or 4 day intensive faculty development workshop, limited to 50 young engineering faculty. SIGCSE are four-hour long workshops.

What about reaching out to mass media? Maria Klawe is working with a producer in Hollywood about creating a House-like series where it’s a computer science person who is solving complicated mysteries. The Crime Scene Investigation (CSI) TV show seriously impacted the number of courses that universities offered about forensics. But shows like House and CSI involve overcoming badness – can computer science do that? There are aspects of science and math in shows like “Numbers”, “Myth Busters”, and National Geographic reconstruction of disasters.

There are dramatic approaches being tried in classrooms. Jen Murdoch at the University of Victoria uses media computation with assignments such as giving everyone 5 sound files and shows them a diagram of a bank and says it comes from 5 microphones and asks
what time did the bank robbers arrive? Doug Blank from Bryn Mawr has students using graphical computing with actual CAT scan images. Tom mentioned that a project where students helped a legal professor with a legal issue.

On the subject of advocacy, CSTA has an advocacy section. Although CSTA focuses on all K-12 computer science education, often AP computing is the only sure-fire way to get computer science into a school. (AP is Advanced Placement, getting college credit for taking a high school course.) Not all schools can have AP classes, but CSTA is working with the Cohort Leadership Group to create a grassroots movement for computer science. Distance technology and partnering with community colleges are examples of other approaches. Since only two states give any credit for computer science, if schools don’t offer AP, they’re unlikely to have any computer science teacher at all at the high school because the AP is the only reason to offer it if computer science doesn’t count toward graduation. Changing 52 state curricula is really challenging, although CSTA is working on it.

Mark related the story of how computer classes in high schools in Georgia obtained a status where they count towards graduation. They first tried to get it to count as math, but the math lobby in Georgia is very strong, and even if they could convince them, the universities would not have accepted computer science as math. Science, on the other hand, has a very poor lobby, so they were able to convince people to have it count as science. In Texas, they were able to do it more rationally, and it counts as a math credit. Texas and Georgia are the only two states where computer science classes in high school count towards graduation. In Ohio, they are making progress towards computer science classes counting as math, but they are not willing to let teachers with computer science degrees be certified to teach classes that count as math. This is despite the fact that Ohio is one of the few states that certify computer science teachers.

Setting up teacher standards is also important. The issue is that several things need to be in place in order to be successful: standards, classes, curriculum, and education certification. Therefore an action item is to develop partnerships through our organizations with CSTA to provide in some sense advocacy support for the teachers through its Cohort Leadership program. Partners could write letters to the school district or go with them to meetings. Part of Georgia’s success was because Kennesaw State and Georgia Tech were pushing. So, the action item could involve people from our organizations talking to the school board, which means that as part of the action item, materials and messages should be developed to help.