Table of Contents

computer science and other fields, enriching both. We need to reach out to create programs that attract and educate the next generation of computer scientists, especially currently underserved populations. We need to reach out to policy makers, educating them on the importance and promise of our discipline. Finally, we need to reach out to each other, as always, to share our best ideas and experiences.

We are pleased that Trustee Professor Matthias Felleisen of Northeastern University will receive the SIGCSE award for Outstanding Contribution to CS Education and present Thursday's keynote address, and that Gordon Davies of the Open University will accept the SIGCSE Award for Lifetime Service to the CS Education Community. In addition, we've invited two keynote speakers to highlight our "reaching out" theme. Susan Landau, fellow at the Radcliffe Institute for Advanced Study at Harvard University, will give Friday's address and Luis von Ahn, professor at Carnegie Mellon University and researcher at Google, will address Saturday's luncheon.

Our sincere thanks go out to the people who made this Symposium extraordinary. First, our symposium committee: Ruth Anderson, Stacey Armstrong, Tim Bell, Dennis Bouvier, Tzu-Yi Chen, Pam Cutter, Adrienne Decker, Lynn Degler, John Dooley, Mary Ann Egan, Kurt Eiselt, Scott Grissom, Michael T. Helmick, Judy Hromcik, Deborah Hwang, Cary Laxer, Lester I. McCann, Larry Merkle, Brianna Morrison, Kristine Nagel, Susan Rodger, RoxAnn Stalvey, Kimberly Voll, Henry Walker, and Jian Zhang. Additional thanks go to our Associate Program Chairs who provided meta-reviews for papers: John Barr, Tracy Camp, Boots Cassell, Steve Edwards, Mike Goldweber, Chuck Leska, Sara Miner More, Sam Rebelsky, and Ingrid Russell. Special thanks to the trail bosses of the Robot Rodeo, Jennifer Kay and Tom Lauwers. We'd also like to thank all of our student volunteers who

Bob Beck and Scott Grissom, SIGCSE President Renee McCauley, former SIGCSE President Barbara Boucher Owens, and the entire SIGCSE Board, as well as from Ashley Cozzi of ACM. Lisa Tolles of Sheridan Printing brought all materials together. Thanks to the City of Dallas Convention and Visitor's Bureau, especially Heather Lovato, who provided valuable information and prizes. We were supported at the Sheraton Dallas by Mark Story, Victor Aguilar, Jerry Balousek and Jeffrey Webb.

Special thanks to our home institutions for providing needed resources: Carnegie Mellon University, Hiram College, College of the Holy Cross and Carleton College. We genuinely hope you enjoy the Symposium.

Symposium Chairs
Thomas J. Cortina, *Carnegie Mellon University*
Ellen L. Walker, *Hiram College*

Program Chairs
Laurie Smith King, *College of the Holy Cross*
David R. Musicant, *Carleton College*

Birds of a Feather	36	48	75 %
Video	7	10	70 %

Number of reviewers: 760 (plus 9 assoc. program chairs, a.k.a. meta-reviewers).
Number of paper reviews received: 1723 regular reviews and 312 meta-reviews
Number of reviews (including meta-reviews) assigned to each paper: ≥ 7
Number of papers with 6 or more regular reviews: 160
Number of papers with 5 regular reviews: 124

SIGCSE 2011 Award Winners

Award for Outstanding Contribution to Computer Science Education
Matthias Felleisen, *Northeastern University*

Award for Lifetime Service to the Computer Science Education Community
Gordon Davies, *Open University*

SIGCSE Video Exhibition

The 2011 SIGCSE Symposium is proud to present the SIGCSE Video Exhibition, which features videos relevant to computing education. The exhibit includes videos for outreach, recruiting, classrooms, and other purposes. **Videos may be viewed just outside the Exhibit Hall on the Lone Star level during regular Exhibit Hours.**

Symposium Evaluations Link

`http://www.sigcse.org/sigcse2011/attendees/evaluations.php`

Associate Program Chairs		John Barr, *Ithaca College*
	Tracy Camp, *Colorado School of Mines*	Boots Cassel, *Villanova*
	Steve Edwards, *Virginia Tech*	Mike Goldweber, *Xavier University*
	Chuck Leska, *Randolph-Macon College*	Sara Miner More, *McDaniel College*
	Sam Rebelsky, *Grinnell College*	Ingrid Russell, *University of Hartford*

Panels & Special Sessions	Pam Cutter, *Kalamazoo College*
Workshops	Ruth E. Anderson, *University of Washington* Adrienne Decker, *University at Buffalo (SUNY)*
Posters	Tzu-Yi Chen, *Pomona College*
Birds-of-a-Feather	Deborah Hwang, *University of Evansville*
Videos	Dennis Bouvier, *Southern Illinois University–Edwardsville*
Publications	Lester I. McCann, *The University of Arizona*
Student Volunteers and Activities	Brianna Morrison, *Southern Polytechnic State University* Mary Anne Egan, *Siena College*
Treasurer	Scott Grissom, *Grand Valley State University*
Evaluations	Kurt Eiselt, *University of British Columbia*
Registration	Lynn Degler, *Rose-Hulman Institute of Technology* Cary Laxer, *Rose-Hulman Institute of Technology* Larry Merkle, *Wright State University*
Webmaster	Michael T. Helmick, Google
Database Administrators	Henry Walker, *Grinnell College* John Dooley, *Knox College*
K-12 Liaisons	Stacey Armstrong, *Cypress Woods High School* Judy Hromcik, *Arlington High School*
International Liaison	Tim Bell, *University of Canterbury*
Supporters/Exhibitors	Susan Rodger, *Duke University*
Kids Camp Coordinators	Kimberly Voll, *University of British Columbia* RoxAnn Stalvey, *College of Charleston*
Pre-Conference Events	Kristine Nagel, *Georgia Gwinnett College*
Local Arrangements	Jian Zhang, *Texas Women's University*
Student Research	Ann Sobel, *Miami University (Ohio)*

Thursday	7:30 AM – 4:00 PM
Friday	7:30 AM – 5:00 PM
Saturday	8:00 AM – 12:10 PM
and	2:30 PM – 3:00 PM

Friday	10:00 AM – 4:30 PM
Saturday	9:30 AM – 12:00 PM

Poster presentation sessions will be held Friday from 10:00 to noon and 3:00 to 5:00 in Lone Star Preconvene.

SIGCSE Business Meeting will be held Friday at 5:10 PM in Lone Star A4.

CCSC Business Meeting will be held Friday at 6:00 PM in Lone Star A3.

NSF Project Showcase presentations will take place in Lone Star BC Thursday and Friday from 10:00 AM to 11:30 AM and 3:00 PM to 4:30 PM, and Saturday from 10:00 AM to 11:30 AM.

SIGCSE Special Projects Showcase presentations will take place in Lone Star A3 on Thursday from 10:45 AM until noon.

The ACM Student Research Competition poster presentations will be Thursday from 1:45 PM to 5:15 PM in Lone Star Preconvene. Presentations by the semi-finalists will be Saturday from 8:30 AM to 10:10 AM in Dallas D3 (graduate) and Dallas D2 (undergraduate). The winners will receive their awards during Saturday's luncheon (12:30 PM in Dallas BCD).

The SIGCSE Reception is Thursday evening 7:00 PM – 8:00 PM, in the Grand Hall

The SIGCSE Luncheon is Saturday in Dallas BCD

CS Education Research & K-12 Teachers Rooms

Descriptions are listed in the Workshops section starting on page 87 of this program. Registration is required for all workshops. Please be aware that workshops with low enrollment may be cancelled. Check the registration website or the registration desk at the Symposium for availability of workshops in which you might be interested.

1. **Advanced Scratch: Computer Science Through Storytelling and Games**	Lone Star A2
2. **Developing an Effective Assessment Program for Student Educational Outcomes**	Lone Star A3
3. **Pedagogical Progressions for Teaching Object-Oriented Design**	City View 7
4. **Explore, Customize, and Create: Getting Your Hands Dirty with UC Berkeley's Lab-Centric Curricula**	State Room 1
5. **Open Source and Freeware Tools for 3D Game Development Courses**	State Room 2
6. **Web Development with Python and Django**	State Room 3
7. **How to Use Algorithm Visualizations in Your Class**	City View 1
8. **Making Mathematical Reasoning Fun: Tool-Assisted, Collaborative Techniques**	City View 2
9. **General Purpose Computing Using GPUs: Developing a Hands-On Undergraduate Course on CUDA Programming**	City View 3
10. **Writing Effective NSF Proposals**	City View 6
11. **Audacious Android Application Programming**	Lone Star A4

Thomas Cortina, Symposium Co-Chair, *Carnegie Mellon University*
Ellen Walker, Symposium Co-Chair, *Hiram College*

2011 SIGCSE Lifetime Service Award: Gordon Davies, *Open University*

2011 SIGCSE Award for Outstanding Contributions to Computer Science Education: Matthias Felleisen, *Northeastern University*

Keynote Address: TeachScheme!
Matthias Felleisen, *Northeastern University*

In 1995, my research team and I decided to create TeachScheme!, an educational outreach project, with the hope that our work on programming languages could effect a dramatic change in K-12 computer science. At this point, we have a new design-oriented curriculum; a pedagogic program development environment to make it fun; and a series of matching programming languages. My talk will focus on just one aspect of the project: the design-oriented curriculum and its smooth path from middle school to college. I will first demonstrate how to teach an intellectually interesting and fun course on programming with something that looks like plain school mathematics. For the rest of the talk, I will sketch the path from there through college.

Thursday, 10:00 AM to 10:45 AM

Coffee Break & Exhibits Lone Star BC

Thursday, 10:00 AM to 11:30 AM

NSF Showcase #1 Lone Star BC

Problets and Codelets for Learning Introductory Programming *(CCLI/TUES)*: Amruth Kumar

"Hands-On" Collaborative Reasoning across the Curriculum *(CCLI/TUES)*: Jason O. Hallstrom, Joseph E. Hollingsworth, Joan Krone, Chong Pak, Murali Sitaraman

A Creative Collaboration between CSE and Biological Sciences through Cloud-enabled Evolutionary Biology Learning Tool *(CLTEAM)*:

As a consequence of this way to think about a proper role for computational thinking, what is emerging is a new discipline, computational science, that brings to bear the knowledge and tools of mathematics, statistics and computer science to solve problems in the sciences and engineering. This multidisciplinary approach is also leading to the creation of new scientific knowledge.

In the past few years computational science programs, both at the undergraduate and the graduate level, have been springing up at a number of colleges and universities. This panel is designed to explore this relatively new phenomenon. Each of the panelists has been a leader at her/his institution in the development of such programs. And, each will present what led them to take on this task, how they went about building a program, some of the features of each program and some of the successes and failures.

SPECIAL SESSION **SIGCSE Special Projects Showcase** Thur. 10:45 – 12:00 Lone Star A3

Chair: Doug Baldwin, *SUNY Geneseo*
Participants: Peter Sanderson, *Otterbein University*; Robert McCartney, *Univ. of Connecticut*; Stephanie Ludi, *Rochester Institute of Technology*; Narayanan Ramachandran, *Middle East College of Information Technology*; Carol Taylor, *Eastern Washington University*

SIGCSE funds a small number of special projects managed by SIGCSE members who have ideas on the investigation and introduction of new ideas in all aspects of computing education. Each project provides an identifiable benefit to our community. Such benefits include general resource development, the creation and assessment of educational practices, and basic advancement in computing knowledge.

SPECIAL SESSION **Computing and Music: A Spectrum of Sound** Thur. 10:45 – 12:00 Lone Star A4

Chair: Robert Beck, *Villanova University*
Participants: Jennifer Burg, *Wake Forest University*; Jesse Heines, *University of Massachusetts* Lowell; Bill Manaris, *College of Charleston*

This special session implements the first goal of the SIGCSE Committee on Computing and Music. It provides a venue in which the committee conveners can present their work to

Tadayoshi Kohno, *University of Washington*; Brian David Johnson, *Intel Corporation*

Computer systems and computer security interact intimately with the needs, beliefs, and values of people. This is especially true as computers become more pervasive. Therefore, in addition to the standard technical material, we argue that students in a computer security course would benefit from developing a mindset focused on the broader societal and contextual issues surrounding computer security systems and risks. We used science fiction (SF) prototyping to facilitate such societal and contextual thinking in a recent undergraduate computer security course. We report on our approach and experiences here, as well as our recommendations for future computer security and other computer science courses.

11:10 ***Security in Computer Literacy: A Model for Design, Dissemination, and Assessment***

Claude Turner, *Bowie State University*; Blair Taylor, *Towson University*; Siddharth Kaza, *Towson University*

While many colleges offer specialized security courses and tracks for students in computing majors, there are few offerings in information security for the non-computing majors. Information security is becoming increasingly critical in many fields, yet most computer literacy courses insufficiently address the security challenges faced by our graduates. This paper discusses the development and impact of a set of modules designed to integrate security into computer literacy across two universities and several community colleges in the state of Maryland. Results from our comparative analyses based on pre- and post-test analysis show significant improvements in post-test results.

11:35 ***Training Students to Steal: A Practical Assignment in Computer Security Education***

Trajce Dimkov, *University of Twente*; Wolter Pieters, *University of Twente*; Pieter Hartel, *University of Twente*

Courses in information security provide students with knowledge of technical security mechanisms and their weaknesses. Teaching students only the technical side of security leads to a generation of students that emphasize digital, but ignore the physical and the social aspects of security. In the last two years we devised a practical assignment which combines digital and physical penetration testing and social engineering. As part of the course the students stole laptops from unaware employees throughout the university campus. The course provided the students with practical overview of security and increased

MIMD parallelism. However, the shift to multicore architectures has made shared-memory MIMD parallelism increasingly important, and inexpensive manycore GPGPUs have revived SIMD parallelism. This paper presents a case study in designing and building a heterogeneous cluster as a learning platform for tera-scale distributed- and shared-memory MIMD parallelism, and GPGPU parallelism.

11:10 *Games as Motivation in Computer Design Courses: I/O is the Key*

Erik Brunvand, *University of Utah*

The design of computer games can be a powerful motivator as students learn about computer architecture and design. Students in classes where computer designs are developed and implemented (usually on Field Programmable Gate Arrays (FPGAs)) seem much more highly motivated if their computer can be used for something visual and interactive when the project is complete. Ensuring that the student teams can have a working game by the end of a semester requires careful planning of how their computer designs interact with the world. We describe the curriculum of a computer design course that uses game design as a "carrot" to encourage active student exploration and deeper understanding of computer architecture, I/O subsystems, and computer implementation.

11:35 *VIREOS: An Integrated, Bottom-Up Educational Operating Systems Project with FPGA Support*

Marc Corliss and Marcela Melara, *Hobart and William Smith Colleges*

In this paper, we present the VIREOS Project, a new operating system (OS) designed specifically for the classroom. VIREOS is a simple, Unix-like, OS, which runs on an educational, MIPS-like architecture. A VIREOS system can either be simulated or run on a pre-configured FPGA. The VIREOS Project can be well integrated with a course on introductory computer architecture. We have several resources available on the Web, which help reduce the overhead of adopting VIREOS. Finally, VIREOS has been used in one operating systems course already, and the feedback from students was generally favorable.

Lam, and Gwen Nugent, *University of Nebraska, Lincoln*

Learning objects (LO) have previously been used to help deliver introductory computer science (CS) courses to students. Students in such introductory CS courses have diverse backgrounds and characteristics requiring revision to LO content and assessment to promote learning in all students. However, revising LOs in an ad hoc manner could make student learning harder for subsequent deployments. To address this problem, we present a systematic revision process for LOs (LOSRP) using proven techniques from educational research. LOSRP uses these validation methods to answer seven questions in order to diagnose what needs to be revised in the LO. Then, LOSRP provides guidelines on revising LOs for each of the seven questions.

11:10 *The Impact of Problem-Oriented Animated Learning Modules in a CS1-Style Course*

Jeffrey Stone and Tricia Clark, *Penn State University*

CS educators face many challenges in teaching basic computer programming to first-year students. As a result, faculty must find new and interesting ways to engage students and provide opportunities for student success. This paper reports on a two-year study involving the Problem-Oriented Animated Learning Modules for Introductory Computer Science (PALMS for CS1) project. PALMS is a set of animated learning modules designed to enhance student engagement, success, and retention through the use of animation, video, audio, and storytelling. The results indicate that PALMS has been successful in engaging introductory students but more work remains to improve student success and retention. Marked improvement in student success and retention was seen during the second year of the study.

11:35 *Evaluating the Use of Learning Objects in CS1*

L. Dee Miller, Leen-Kiat Soh, Ashok Samal, Gwen Nugent, Kevin Kupzyk, and Leyla Masmaliyeva, *University of Nebraska, Lincoln*

Learning objects (LOs) have been previously used in computer science education. However, analyses in previous studies have been limited to surveys with limited numbers of LOs and students. This lack makes detailed analysis of LO usefulness problematic. Using an empirical approach, we have studied a suite of LOs deployed to CS1 courses from 2007-2010. Based on our analysis of student interaction data, we found that (1) students using LOs have significantly higher assessment scores than the control group, (2) several student attributes are significant predictors of learning, (3) active learning has a significant effect on

during the academic year and during summer. Co-op and internship opportunities are available to this group as well. Due to the fact that freshman and sophomore students do not have sufficient background, they are often left behind and are not involved in research activities. This paper shares some experiences with a program that was put in place through an NSF STEP grant that provides research opportunities to freshman, sophomore, and first year transfer students. The paper presents examples of projects in which computer science scholars were involved. We have learned that lower-division computer science students are excited about participating in research.

11:10 *Incorporating Social Issues of Computing in a Small, Liberal Arts College: A Case Study*

Henry Walker and Janet Davis, *Grinnell College*

CC2001 and CC2008 recommend that an undergraduate computing curriculum include 16 hours related to social and professional issues. An ITiCSE 2010 Working Group discussed approaches for incorporating this material in the curriculum and outlined seven contrasting implementation cases. Also, Baldwin et al discuss the implementation of computing curricula at five different liberal arts colleges [TOCE 2010]. However, none of these provides specific implementation details for addressing social issues in a liberal arts computing curriculum. This paper identifies successful strategies from one college and begins a general discussion of teaching social issues of computing in a liberal arts setting.

11:35 *Using Undergraduate Teaching Assistants in a Small College Environment*

Paul Dickson, *Hampshire College*

Use of undergraduate teaching assistants in computer science courses is not new but is primarily thought of as a way to help with large classes in universities and rarely for small classes in small colleges. In this paper we discuss the success we have had over the past 2 years using undergraduate students as teaching assistants for small computer science classes. Our experience has shown that having undergraduates as teaching assistants helps to engage students with the material, creates a more relaxed classroom environment in which students feel more free to ask questions, improves the effectiveness of class time, and improves class quality. We believe that our experiences using undergraduate teaching assistants can be beneficial not only to small colleges but also to large universities.

Information about supporter sessions can be found starting on page 68 of this program.

Thursday, 12:00 PM to 12:15 PM

Student Icebreaker	Grand Hall A

Thursday, 12:00 PM to 1:45 PM

Lunch Break	*On your own.*
First Timer's Lunch	Lone Star A1

Special Guest Speaker:
Gordon Davies, Open University
(SIGCSE 2011 Honoree for Lifetime Service to the Computer Science Education Community)

Thursday, 1:45 PM to 5:15 PM

Student Research Poster Session	Lone Star Preconvene

In 2010 a new annual symposium on Educational Advances in Artificial Intelligence (EAAI) was launched as part of the AAAI annual meeting. EAAI has a particular focus, however, as the event is specific to educational work in Artificial Intelligence. This panel introduces EAAI as a way of fostering more interaction between educational communities in computing. Specifically, the panel will discuss the goals of EAAI, provide an overview of the kinds of work presented at the symposium, and identify potential synergies between that EAAI and SIGCSE as a way of better linking the two communities going forward.

PANEL **Learning through Open Source Participation** Thur. 1:45 – 3:00 Lone Star A3

Moderator: Gregory Hislop, Drexel University
Panelists: Heidi Ellis, *Western New England College*; Mel Chua, *Red Hat, Inc.*; Matthew Jadud, *Allegheny College*

Free and Open Source Software (FOSS) and documentation projects provide excellent learning opportunities for students. In the context of active learning, FOSS is particularly interesting in providing transparent meritocracies that allow students to observe and contribute as part of their learning. This panel will present four different perspectives: (a) different ways that students can contribute to FOSS projects beyond coding, (b) an industry perspective on student involvement in FOSS projects, (c) how Humanitarian FOSS can provide a welcoming environment for student learning, and (d) barriers to faculty involvement and how such barriers can be overcome.

SPECIAL SESSION **The CS10K Project: Mobilizing the Community to Transform High School Computing** Thur. 1:45 – 3:00 Lone Star A4

Chair: Owen Astrachan, *Duke University*
Participants: Jan Cuny, *National Science Foundation*; Chris Stephenson, *Computer Science Teachers Association*; Cameron Wilson, *ACM*

The CS10K project is a large-scale, collaborative project bringing together stakeholders from wide-ranging constituencies with the goal of systematically changing the scale, Peter Hubwieser and Marc Berges, *Technische Universität München*

We describe a research project that investigates how far freshmen at the University are able to learn OOP with as less instruction as possible. We designed specific tasks for programming assignments and supporting worksheets that contained the only information input that the students received during the courses. After the completion, we investigated the program code the students produced in order to assess the quality of their products. The surprising result was that most of the students were able to write quite satisfying programs. Additionally, a cluster analysis of the results showed that there are two different types of students: some accept and apply the object oriented concepts quite willingly, while the others prefer to program in a more traditional, procedural style.

2:10 ***Extreme Apprenticeship Method in Teaching Programming for Beginners***

Arto Vihavainen, Matti Paksula, and Matti Luukkainen, *University of Helsinki*

Learning a craft like programming is efficient when novices learn from people who already master the craft. In this paper we define Extreme Apprenticeship, an extension to the cognitive apprenticeship model. Our model is based on a set of values and practices that emphasize learning by doing together with continuous feedback as the most efficient means for learning. We show how the method was applied to a CS I programming course. Application of the method resulted in a significant decrease in the dropout rates in comparison with the previous traditionally conducted course instances.

2:35 ***Expressing Computer Science Concepts through Kodu Game Lab***

Kathryn Stolee, *University of Nebraska – Lincoln*; Teale Fristoe, *University of California, Santa Cruz*

Educational programming environments such as Microsoft Research's Kodu Game Lab are often used to introduce novices to computer science concepts and programming. Unlike many other educational languages that rely on scripting and Java-like syntax, the Kodu language is entirely event-driven and programming takes the form of ‘when – do’ clauses. Despite this simplistic programming model, many computer science concepts can be expressed using Kodu. We identify and measure the frequency of these concepts in 346 Kodu programs created by users, and find that most programs exhibit sophistication through the use of complex control flow and Boolean logic. Through Kodu’s non-traditional language, we show that users express and explore fundamental computer science concepts.

In this paper we revisit previous research to elaborate on the question: “Can graduating students design software systems?” The work concluded that the answer was “not really”. We wished to determine if this was true currently at our institution and also to look at whether students were able to design software in groups, and evaluate others’ designs. In summary, it appears that our students, just as in the original experiment, struggle with doing design, even in a group situation. The representation of behavioral design was particularly lacking. That said, students were able to recognize weaknesses when evaluating other group designs. Based on our findings, we provide several pedagogic recommendations.

2:10 *The FCS1: A Language Independent Assessment of CS1 Concepts*

Allison Elliott Tew, *University of British Columbia*; Mark Guzdial, *Georgia Institute of Technology*

A goal of many education projects is to determine if an intervention has had an impact on student learning. However, computing lacks valid assessments for pedagogical or research purposes. We developed the FCS1 Assessment, the first assessment for introductory CS concepts that is applicable across a variety of current pedagogies and programming languages. We applied methods from educational test development, adapting them as necessary to fit the disciplinary context. We conducted a large scale empirical study to demonstrate that pseudo-code was an appropriate mechanism for achieving language independence. Finally, we established the validity of the assessment using a multi-faceted argument, combining interview data, statistical analysis of results on the assessment, and CS1 exam scores.

2:35 *Online vs. Face-to-Face Pedagogical Code Reviews: An Empirical Comparison*

Christopher Hundhausen, Pawan Agarwal, and Michael Trevisan, *Washington State University*

We have developed an adaptation of studio-based instruction for computing education called the pedagogical code review (PCR), which is modeled after the industrial code inspection process. PCRs are time-intensive, making them difficult to implement within a typical course. To address this issue, we have developed an online tool that allows PCRs to take place outside of class. We conducted an empirical comparison of a CS 1 course with online PCRs and a CS 1 course with face-to-face PCRs. We found that, in the course with face-to-face PCRs, students' self-efficacy and peer learning attitudes were higher, students taught to students. This is done in a way that emphasizes the relationships between them, and shows how considering abstraction and extreme cases can lead to the generation of new algorithms. In this paper we use a flexible priority queue, the d-heap, to derive three common sorting algorithms. We combine this with using a BST as a priority queue and some prior observations to show strong relationships between the main sorting algorithms that appear in textbooks. Using this view, students are able to explore elegant relationships between sorting algorithms. This approach can also lead to assessment that goes beyond desk-checking to evaluate students' understanding of sorting algorithms.

2:10 *Getting Algorithm Visualizations into the Classroom*

Cliff Shaffer, Monika Akbar, Alexander Joel Alon, Michael Stewart, and Stephen Edwards, *Virginia Tech*

Algorithm visualizations (AVs) are widely viewed as having potential for improving computer science education. However, the rate of AV use does not match the positive interest that instructors report. Surveys show that impediments to use of AVs include difficulties in finding quality AVs on desired topics, difficulties in adapting AVs to a given classroom setting, and lack of knowledge on how to deploy AVs. This indicates a need for better support for instructors, to get them past these barriers. We seek to provide support through an online educational community that focuses on community-driven content through members' discussions, reviews, and ratings of content items. The AlgoViz community effort will better focus the future direction of AV development and use.

2:35 *Two Experiments Using Learning Rate to Evaluate an Experimenter Developed Tool for Splay Trees*

Michael Orsega, *The University of West Georgia*; Bradley Vander Zanden and Christopher Skinner, *The University of Tennessee*

We conducted two experiments evaluating Sketchmate, a tool used to teach the splay tree data structure and its algorithms. Learning and learning rates were compared across two groups, one using Sketchmate and the other using paper and pencil on practice problems. Results from Experiment I showed that when students used Sketchmate with minimal feedback, there were no significant differences across learning, time spent learning, or learning rate. Experiment II used a version of Sketchmate that provided richer feedback. Results showed similar learning but Sketchmate took significantly less time. Thus when feedback was added, learning rates were significantly greater relative to the paper-and-pencil condition. Discussion focuses on measuring learning rates when evaluating

Requirements engineering, an integral part of the life of a software engineer, often receives little or no attention in the education of a computer science student. We report on our experiences in constructing an innovative curriculum that utilizes a three tier model of learning that provides students with hands-on experience on the various facets of requirements elicitation and management. This curriculum can be integrated into an existing course on software engineering, software requirements or the senior capstone experience. We believe our experience will be of use to other computer science and software engineering programs that are aiming at introducing requirements in the undergraduate curriculum.

2:10 *Collaborative Web-Based Learning of Testing Tools in SE Courses*

Peter Clarke, Jairo Pava, and Yali Wu, *Florida International University*; Tariq King, *North Dakota State University*

One of the main concerns in the software industry continues to be the development of high quality software. The training of software developers continues to grow in academia since more institutions are offering software engineering (SE) courses. However, little attention is usually given to testing topics. In this paper we describe an approach that non-intrusively integrates the use of software testing tools in SE courses. The cornerstone of our approach is the interaction students have with a Web-Based Repository of Software Testing Tools (WReSTT) that contains tutorials on testing concepts and testing tools. We present the results of a study that show an improvement in the students' conceptual understanding of software testing and the use of testing tools.

2:35 *Software Studio: Teaching Professional Software Engineering*

Tom Nurkkala and Stefan Brandle, *Taylor University*

Software Studio is a studio-based learning curriculum designed to train students as professional software engineers. Traditional software engineering courses remain important, but suffer significant gaps in preparing students for professional engagement. We describe our curriculum model, highlight ways in which it fills these gaps, and offer a SWOT analysis. As practical guidance, we reflect on our missteps and successes in implementing Software Studio over the past five semesters. Finally, we suggest future directions for Software Studio.

Russel J. Clark and Matthew Wolf, *Georgia Institute of Technology*

Information about supporter sessions can be found starting on page 68 of this program.

SUPPORTER SESSION **Amazon** Thur. 1:45 – 3:00 Dallas D3

Software as a Service, Cloud Computing, and Software Education
Armando Fox, *UC Berkeley*

Information about supporter sessions can be found starting on page 68 of this program.

Thursday, 3:00 PM to 3:45 PM

Coffee Break & Exhibits Lone Star BC

Thursday, 3:00 PM to 4:30 PM

NSF Showcase #2 Lone Star BC

Project MLeXAI: Machine Learning Experiences in the Undergraduate Curriculum *(CCLI/TUES)*: Ingrid Russell, Zdravko Markov

From CSI, an Integrated Exhibit and Web-based Learning Initiative, to TexNET, a Professional Development Network for Museum Professionals *(ISE)*: Charlie Walter

A Unifying Paradigm for Dynamic Simulation *(CPATH)*: Richard M. Salter

Project: CyberCHEOS: A Service-Oriented Cyberinfrastructure (SOCI) for

Technology

Much has been written about the decrease in the number of students pursuing computing in colleges. Unfortunately, most students form their (usually negative) opinions of computing long before they reach college, so it is beneficial for university faculty who want to increase college enrollment to do outreach aimed at a younger audience. Many college faculty are interested in doing K-12 outreach but often do not know where to start or do not know any successful strategies. The purpose of this panel is to present K-12 outreach strategies that have worked and then to provide the audience with the opportunity to brainstorm with the panelists to explore alternate ideas for outreach and to identify new approaches.

PANEL **Setting the Stage for Computing Curricula 2013: Computer Science - Report from the ACM/IEEE Joint Task Force** Thur. 3:45 – 5:00 Lone Star A3

Moderator: Mehran Sahami, *Stanford University*
Panelists: Mark Guzdial, *Georgia Institute of Technology*; Andrew McGettrick, *University of Strathclyde*; Steve Roach, *University of Texas at El Paso*

Following a roughly 10 year cycle, the Computing Curricula volumes have helped to set international curricular guidelines for undergraduate programs in computing. In the summer of 2010, planning for the next volume in the series, Computer Science 2013, began. This panel seeks to update and engage the SIGCSE community on the Computer Science 2013 effort.

SPECIAL SESSION **It Seemed Like a Good Idea at the Time** Thur. 3:45 – 5:00 Lone Star A4

Chair: Robert McCartney, *Univ. of Connecticut*
Participants: Jonas Boustedt, *Univ. of Gavle*; Josh Tenenberg, *Univ. of Washington Tacoma*; Stephen Cooper, *Stanford Univ.*; Daniel D. Garcia, *Univ of California Berkeley*; Michelle Hutton, *Stanford Univ.*; Nick Parlante, *Stanford Univ.*; Brad Richards, *Univ. of Puget Sound*

An effective CS1 approach has been developed for encouraging diverse students without prior computer science experiences to select computing majors. Separation of CS1 sections by prior experience level concentrates diverse students in the inexperienced section. Within that section we use several techniques to increase student comprehension and participation, including an integrated lecture/lab, many small examples and assignments, student participation, etc. We discuss the approach and evaluate its performance over a four-year time period.

4:10 ***Women Build Games, Seriously***

Elizabeth Sweedyk, *Harvey Mudd College*

Recruitment of students to computer science has been a major focus of effort for educators since the dot-com bust in 2001. Two largely disparate themes in these efforts are women and games. There have been numerous efforts to broaden participation in computer science by attracting women to the field. At the same time, games are increasingly used to attract new students. Our interest lies at the intersection of these methods. Our experience with games in the classroom suggests that they can work to reinforce gender stereotypes of Computer Science, that they can work to foster the so-called "Geek mythology." This paper reports on our solution. Rather than giving up games, we turned to serious ones.

4:35 ***Contextualized Approaches to Introductory Computer Science: The Key to Making Computer Science Relevant or Simply Bait and Switch?***

Jennifer Kay, *Rowan University*

The arguments in favor of contextualized approaches to attract non-CS-majors to our classes are very persuasive. But what about students who then choose to major or minor in CS? Of course we want to offer them interesting and engaging first courses in CS, and indeed this may help with our efforts to attract more students to our programs. But what happens in subsequent semesters? The purpose of this paper is to initiate a general discussion on the use of any sort of "cool" new approach to introductory CS. These approaches successfully attract students to study subjects that we ourselves are deeply engaged in. But we need to discuss as a community what happens to students who do choose to major or minor in CS when our individual classes conclude and the rest of their studies commence.

Shoop, *Macalester College*

WebMapReduce (WMR) is a strategically simplified user interface for the Hadoop implementation of the map-reduce model for distributed computing on clusters, designed so that novice programmers in an introductory CS courses can perform authentic data-intensive scalable computations using the programming language they are learning in their course. The open-source WMR software currently supports Java, C++, Python, and Scheme computations, and can readily be extended to support additional programming languages, and configured to adapt to the practices at a particular institution for teaching introductory programming. Potential applications in courses at all undergraduate levels are indicated, and implementation of the WMR software is described.

4:10 *Practical Parallel and Concurrent Programming*

Caitlin Sadowski, *University of California at Santa Cruz*; Thomas Ball, Judith Bishop and Sebastian Burckhardt, *Microsoft Research*; Ganesh Gopalakrishnan and Joseph Mayo, *University of Utah*; Madanlal Musuvathi and Shaz Qadeer, *Microsoft Research*; Stephen Toub, *Microsoft*

Multicore computers are now the norm and entail parallel and concurrent programming. There is therefore a pressing need for courses that teach effective programming on multicore architectures. We believe that such courses should emphasize high-level abstractions for performance and correctness and be supported by tools. This paper presents a set of freely available course materials for parallel and concurrent programming, along with a testing tool for performance and correctness concerns called Alpaca (A Lovely Parallelism And Concurrency Analyzer). These course materials can be used for a comprehensive parallel and concurrent programming course, a la carte throughout an existing curriculum, or as starting points for graduate special topics courses.

4:35 *Teaching Concurrency-Oriented Programming with Erlang*

Ariel Ortiz, *Tecnológico de Monterrey, Campus Estado de México*

Teaching how to write correct programs is hard; teaching how to write correct concurrent programs is even harder. There is a desperate need for a better concurrency programming model than what most people are currently using. The Erlang programming language might be a step in that direction. This paper provides an overview of Erlang and how it has been successfully used to teach concurrency-oriented programming (COP) in a sophomore level course at the Tecnológico de Monterrey, Campus Estado de México.

Tedford, *Texas A & M University, Corpus Christi*

This paper describes the benefits of Peer-Led Team Learning (PLTL), an NSF-sponsored program in the sciences, to peer leaders serving in the Computing Alliance for Hispanic Serving Institutions (CAHSI). Beyond the benefits to students enrolled in the PLTL courses, survey findings show the majority of peer leaders report increased self-efficacy in teaching computer science, improved content knowledge, and better communication and leadership skills following a semester of leading PLTL. Results from this diverse group of leaders indicate no differences in gains between underrepresented minority and majority students, suggesting the program may provide a path for improving retention of underrepresented groups in the field.

4:10 ***Lessons Learned from a PLTL-CS Program***

Christian Murphy and Rita Powell, *University of Pennsylvania*; Kristen Parton and Adam Cannon, *Columbia University*

At Columbia University, the Columbia Emerging Scholars Program has used Peer-Led Team Learning (PLTL) in an effort to increase enrollment in CS courses beyond the introductory level, and to increase the number of students who select Computer Science as their major, by demonstrating that CS is necessarily a collaborative activity that focuses more on problem solving and algorithmic thinking than on programming. Preliminary results indicate that this program has had a positive effect on increasing participation in the major. This paper discusses our experiences of building and expanding the Columbia Emerging Scholars program. We expect that this paper will provide a valuable set of lessons learned to other educators who seek to launch or grow a PLTL program at their institution as well.

4:35 ***Tutoring for Retention***

Suzanne Menzel, Joseph Cottam and Janet Greenblatt, *Indiana University*

Peer tutoring is a simple, low-cost intervention that can be implemented in CS1/2 courses. It is hypothesized that peer tutoring helps students build a sense of community, succeed in course work, and build confidence to take further courses in the major. This paper examines the latter two hypotheses by examining the predicted and actual behavior of students in CS1/2. Course performance improvements were observed, which also strongly influence retention in computing-related courses. The measures also point to further research directions, such as social influences and the impact of peer tutoring relative to office hours or online forums.

of New Jersey; Meredith Stone, *Independent Evaluator*

In this paper, we describe an ongoing multidisciplinary undergraduate seminar that we have developed, in which student teams build non-human systems that conduct our college orchestra. Students and faculty in the course come from four disciplines: computer science, interactive multimedia, music, and mechanical engineering. This paper describes the course structure, computer science components, final projects, team dynamics, and assessments. We evaluate the results to-date and discuss ongoing revisions and expectations for the future.

4:10 *Nelson: A Low-Cost Social Robot for Research and Education*

Michael Ferguson, Nick Webb and Tomek Strzalkowski, *ILS Institute, SUNY Albany*

A social robot is a robotic platform that supports natural interaction with people in a human-scale environment. Such a platform allows interesting opportunities for both traditional Computer Science students and students from other disciplines, such as psychology, philosophy, design and communications. In this paper, we describe a new social robotic platform for educational uses that is equipped with a social face, arms for gesturing, advanced sensory, mobile base, and ROS integration. By using off-the-shelf and rapidly prototyped components, together with open source software, this platform is low-cost, easy to use, and easy to reproduce.

4:35 *An Introduction to AI Course with Guide Robot Programming Assignments*

Nik Swoboda, *Universidad Politécnica de Madrid*; Juan Bekios-Calfa, *Universidad Católica del Norte*; Luis Baumela and Javier de Lope, *Universidad Politécnica de Madrid*

In this paper we describe a collection of course materials designed to be used in an undergraduate introduction to artificial intelligence (AI) course. These materials include three programming assignments, each touching upon core AI topics, which require that the students build the main functionalities of a guide robot. These assignments were carefully designed to allow the same solution to work both with a robot simulator and an inexpensive web-cam as well as with real robots. An overview of the course and the assignments is given along with references to online versions of the resources developed to teach the course.

Information about supporter sessions can be found starting on page 68 of this program.

VISIT OUR EXHIBIT HALL
AND SEE THE LATEST IN TOOLS & TEXTS
FOR CS EDUCATION
AND CHECK OUT OUR VIDEO EXHIBITION

EXHIBIT HALL: LONESTAR BC
VIDEO EXHIBITION: LONESTAR PRECONVENE

Exhibit Hall & Video Exhibition Hours
Thursday 10:00AM-6:00PM
Friday 10:00AM-4:30PM
Saturday 9:30AM-12:00PM

Web Programming	Dallas A2
Using Game Development to Teach Parallelism: Where can I find Resources to Get Started?	Dallas A3
Attention and Learning in the Computer Science Classroom	Dallas D1
Technology that Educators of Computing Hail (TECH) site in the Ensemble Computing Portal	Dallas D2
Program by Design: TeachScheme/ReachJava	Dallas D3
Teaching Open Source (Software)	State Room 1
Undergraduate Information Security Curriculum Development	State Room 2
Assessing Interdisciplinary CS Initiatives	State Room 3
Teaming up to Change K-12 CS Education, One State-at-a-Time	State Room 4
An iOS BOF	City View 1
Mathematical (and Other) Reasoning in Computer Science Education	City View 2
AP CS A: Sharing Teaching Strategies and Curricular Ideas	City View 3
Report of Findings: ACM Summit on Computing Education at Community Colleges	City View 4
Connecting Biomedical and Health Informatics with Computer Science	City View 5
Teaching and Learning with Scratch	City View 6
SIGCSE and the International Community	City View 7
Capstone Projects in CS and SE: How Do You Reach Out?	City View 8

Media Computation	Dallas A2
Technology Available to Educators: How Does Access to Next-Generation Hardware Impact Students?	Dallas A3
What Makes a Good Scratch Program? Examining Structure and Style in Scratch Programs	Dallas D1
Teaching Secure Programming	Dallas D2
Lifesavers! Favorite Online Tools and Resources for Teaching CS1	Dallas D3
Designing Open Source Labs for Distance Education	State Room 1
Logisim and Circuit Simulation: Future Directions	State Room 2
Computing Education that Supports Research: Connecting with the Committee of Education of the Computing Research Association (CRA-E)	State Room 3
Outreach & Reaching Out in K-12: An International Perspective	State Room 4
Introducing Software Engineering Principles in the First Two Years of Computer Science Education	City View 1
Exploring Computer Science: Fostering Computational Thinking through Engaging and Relevant Curriculum	City View 2
Creating a Game Development Course for Pre-Collegiate Schools	City View 3
A Neglected Pipeline? How Faculty Teach, Advise, and Mentor Transfer Students	City View 4
Sustainability and Computing Education	City View 5
Education, Computers and Society	City View 6
Enhancing Undergraduate Education through the ACM Distinguished Speakers Program	City View 7
Computer Science: Small Department Initiative	City View 8

SIGCSE Keynote Address Dallas BC

Welcome

Laurie Smith King, Program Co-Chair, *College of the Holy Cross*
David R. Musicant, Program Co-Chair, *Carleton College*

SIGCSE Keynote Address: A Computer Scientist Goes to Washington: How to be Effective in a World Where Facts are 10% of the Equation

Susan Landau, *Radcliffe Institute for Advanced Study, Harvard University*

Government's role in computer science is much larger than funding agencies. Digital rights management, net neutrality, and cybersecurity are hot topics in Washington, hot topics where regulation or legislation may have major impact on the computer systems we develop and enjoy. Yet the rules governing DC are very different than the rules that govern science and engineering, and learning how to operate in a world where facts are only ten percent of the equation can be a challenging experience for someone more accustomed to proving theorems and building systems. I'll describe what it takes for a nerd to be effective in the world of government, and give some specific examples in the hot area of cyberwar.

Friday, 10:00 AM to 10:45 AM

Coffee Break & Exhibits Lone Star BC

Friday, 10:00 AM to 11:30 AM

NSF Showcase #3 Lone Star BC

The CyberCIEGE Video Game in Cybersecurity Education *(SFS)*: Cynthia Irvine, Michael Thompson

Quahog Cohorts: A Community of Student Scientists *(S-STEM)*: Kathryn Sanders

Project EXCE2L (Excellence in Computer Education with Entrepreneurship and Leadership Skills) *(CPATH)*: Wendy Tang, Simona Doboli

CI-Facilitators: Accelerating Knowledge Development across the STEM

Panelists: Hans-Peter Bischof, *Rochester Institute of Technology*; Daniela Raicu, *DePaul University*; Susan Urban, *Texas Tech University*

A working group at a recent National Science Foundation (NSF) Computer and Information Science and Engineering (CISE) Research Experiences for Undergraduates (REU) PI's meeting identified four important issues in undergraduate research: 1) how to design a good research project, 2) how to prepare students for research, 3) how to measure outcomes of undergraduate research and 4) incentives for undergraduates to publish as a result of their participation in research. The panelists will each address one of the issues identified above, sharing their expertise while addressing the issue and providing solid guidance to anyone interested in promoting undergraduate research. A significant amount of time will be set aside for audience participation and discussion.

SPECIAL SESSION **Report on Qualitative Research Methods Workshop** Thur. 10:45 – 12:00 Lone Star A3

Chair: Sue Fitzgerald, *Metropolitan State University*
Participants: Renée McCauley, *College of Charleston*; Vicki Plano Clark, *University of Nebraska-Lincoln*

This special session will recap qualitative research design and analysis as discussed during an NSF-sponsored two-part workshop for computer science education researchers held in 2009-2010. Several workshop participants will illustrate the use of qualitative methods by describing their research projects. They will briefly present their methodologies, analyses and findings. Attendees will be encouraged to ask questions about how qualitative methods can be used in their own research projects.

SPECIAL SESSION **NCATE Standards for Preparation of Secondary Computer Science Teachers** Thur. 10:45 – 12:00 Lone Star A4

Chair: Philip East, *University of Northern Iowa*
Participants: Stephen Rainwater, *University of Texas-Tyler*; Chris Stephenson, *CSTA*; Joe Kmoch, *Milwaukee Public Schools*; Charmaine Bentley, *Dallas Independent School District*

However, while there is good evidence, these environments are effective, the question remains 'what students actually learn'. For our purpose, we would like to know if students can apply the knowledge obtained from programming games to creating science simulations. Specifically, we want to better understand if students can recognize Computational Thinking Patterns from game programming. Computational Thinking Patterns are abstract programming patterns that enable agent interactions in games and science simulations. The participants in a summer institute were administered a Computational Thinking Pattern Quiz. This quiz tested the participants' ability to recognize patterns in a context removed from game programming.

11:10 *Initial Experience with a Computational Thinking Course for Computer Science Students*

Dennis Kafura and Deborah Tatar, *Virginia Tech*

Experience with the first offering of a computational thinking course for computer science (CT4CS) students is reported. The course is grounded in student interaction with fundamental, recurring concepts suggested by comparison with two sets of computer science principles. By using specialized, freely available tools and physical simulations it is possible to provide concrete, tangible learning experiences that neither require knowledge of nor the overhead of programming. Student end-of-term reflections indicate that the course deepened and broadened their understanding of computer science even when they had previously encountered a topic, and improved their computer science vocabulary.

11:35 *A Model for Piloting Pathways for Computational Thinking in a General Education Curriculum*

Charles Dierbach, *Towson University*; Harry Hochheiser, *University of Pittsburgh*
Christopher Ariza, *MIT*; Tina Kelleher, Gerald Jerome, Samuel Collins, William Kleinsasser, Josh Dehlinger and Siddharth Kaza, *Towson University*

In this paper, we report on the progress and experiences of the first two years of a three-year project for incorporating computational thinking into the undergraduate, general education curriculum at our university. We discuss our experience with the model in generating faculty interest, receiving institutional support, and receiving positive response from students. Thus far, an Everyday Computational Thinking course and four other discipline-specific computational thinking general education courses have been developed, piloted, and assessed. Initial assessments show promising and significant student, instructor and administration interest in computational thinking as a basis in courses covering multiple

development of many areas in computer science. Commensurately, in this paper, we argue for expanding the coverage of probability in the computing curriculum. Specifically, we present details of a new course we have developed on Probability Theory for Computer Scientists. An analysis of course evaluation data shows that students find the contextualized content of this class more relevant and valuable than general presentations of probability theory. We also discuss different models for expanding the role of probability in different curricular programs that may not have the capacity to teach a full course on the subject.

11:10 ***Mathematical Induction is a Recursive Technique***

Robert L. Scot Drysdale, *Dartmouth College*

This paper argues that learning and using proof by induction is easier if the recursive nature of proof by induction is made explicit, especially for students familiar with recursion. It can be useful to view an inductive proof as a template for a recursive program that takes a specific instance as a parameter and generates a complete direct proof for that instance. The abstract idea of assuming and invoking an inductive hypothesis is replaced by the concrete idea of making a recursive call to prove a lemma. Viewing induction as a recursive process also allows us to give a rule for determining what base cases need to be proved in strong induction and simplifies creating correct inductive proofs.

11:35 ***Teaching Discrete Structures: A Systematic Review of the Literature***

James Power, Thomas Whelan and Susan Bergin, *National University of Ireland, Maynooth*

This survey paper reviews a large sample of publications on the teaching of discrete structures and discrete mathematics in computer science curricula. The approach is systematic, in that a structured search of electronic resources has been conducted, and the results are presented and quantitatively analyzed. A number of broad themes in discrete structures education are identified relating to course content, teaching strategies and the means of evaluating the success of a course.

the concepts involved in operating systems and their interface to the hardware. As an operating system is typically the first reactive system which students encounter in their studies, the goal of the class is to develop an understanding of the tools and reasoning which are involved in understanding and working with the internals of such a system. This course is currently in its third year with enthusiastic responses from students, especially those who have been able to apply its lessons in co-operative work assignments, and an undergraduate class teaching substantially the same material is currently underway.

11:10 *Structured Linux Kernel Projects for Teaching Operating Systems Concepts*

Oren Laadan, Jason Nieh and Nicolas Viennot, *Columbia University*

Students learn more about OS through hands-on project experience. Linux has emerged as a widely-used platform to enable such learning, but developing programming projects for a production OS is challenge. We address the challenges of designing Linux kernel programming projects that provide educational value for an OS course. We describe assignments that gradually introduce students to core OS topics and components, while implicitly teaching them about aspects of real-world OS. We require students to understand core components of the OS to make suitable changes, while keeping the coding complexity modest. Our experiences in teaching the course over the past decade have been very positive, and we now build on it to draw guidelines for designing pedagogically effective projects.

11:35 *LINQ ROX! Integrating LINQ into the Database Curriculum*

Suzanne Dietrich and Mahesh Chaudhari, *Arizona State University*

The Language INtegrated Query (LINQ) language is a declarative query language integrated within an object-oriented programming language that provides a unified paradigm for querying relations, objects, and XML (ROX). This paper describes a suite of exercises, from cooperative in-class activities to larger-scale graded assignments, for incorporating LINQ into the database curriculum. These exercises support various student learning outcomes and illustrate the applicability of LINQ by querying the same database enterprise across the ROX data models.

Over the course of several semesters, we had the same professor teach several CS1 and CS2 courses for computer science and game development majors. As part of his classroom approach, the professor used a student response system to engage the students in the flow of the lecture. In this paper, we examine the relationships between student participation using the student response system and student performance in the course assessments. We also explore the relationship between each student's perceived mastery of course topics and their demonstrated mastery of those topics on the Final Exam. Finally, we explore several differences between the multiple courses included in the study.

11:10 *Effectiveness Of Cognitive Apprenticeship Learning (CAL) And Cognitive Tutors (CT) For Problem Solving Using Fundamental Programming Concepts*

Wei Jin, *Shaw University*; Albert Corbett, *Carnegie Mellon University*

In this paper, we describe our approach in addressing learning challenges students experience in introductory programming courses. We combine two effective instructional methodologies to help students learn to plan programs prior to writing code: Cognitive Apprenticeship Learning (CAL) and Cognitive Tutors (CT). In the CAL component, the instructor models program planning in class and paper handouts are used to scaffold program planning in homework assignments. In CAL-CT, the program-planning process is also supported by a computer tutor which provides step-by-step feedback and advice. The results show that the combined CAL-CT approach yielded substantial gains over traditional instruction. Its advantage over the CAL-Only approach is also significant.

11:35 *Can AlgoTutor change Attitudes toward Algorithms?*

Jungsoon Yoo, Sung Yoo, Suk Seo and Chrisila Pettey, *Middle Tennessee State University*

The ability to design an algorithm is one of the most important learning outcomes of a computer science program. Unfortunately, students frequently think that designing algorithms is difficult and unimportant. To counteract these two attitudes, we developed AlgoTutor, a web-based algorithm development tutoring system. AlgoTutor’s primary components are the algorithm composer and the algorithm tracer. A third component, ProgramPad, was added to show the connection between algorithms and code. This paper presents the results of experiments that assessed AlgoTutor’s effectiveness in changing student attitudes about algorithm development. The results show that students who used AlgoTutor in CS-I were more likely to believe that they could design an algorithm and that

Peli de Halleux and Nikolai Tillmann, *Microsoft Research, Redmond;* Pat Yongpradit

Information about supporter sessions can be found starting on page 68 of this program.

Friday's Lunchtime Activities

Lunch Break (on your own)	Fri. 12:00 – 1:45
Upsilon Pi Epsilon (UPE) Annual Convention	Fri. 12:10 – 1:35 Dallas BC

Keynote Speaker and Recipient of the 2011 UPE Abacus Award:
Donald Knuth; *Professor Emeritus, Stanford University*

Robot Hoedown & Rodeo

Never programmed a robot before? Give it a try!
Already programmed a robot? Try out a different one!

Stop by the "Robot Corral" (Live Oak Room) – take the skywalk to the hotel and go down the hallway to the right.

Mini-training tutorials start at 10:45 on Thu & Fri
in Dallas D3 - for more details, see our flyer or visit:

Panelists: Zachary Dodds, *Harvey Mudd College*; Timothy Huang, *Middlebury College*; Samuel A. Rebelsky, *Grinnell College*

At the SIGCSE Symposium in 2007, we presented a panel in which seasoned teaching faculty from large universities shared the teaching tips we wished we'd known before starting our careers. The session was well-received; however, since all of the presenters regularly taught large classes, many of their suggestions were not relevant to the experience of teaching small classes. Quite a few attendees suggested there be a follow-up session with presenters who could address the challenges specific to small college or university classes. To that end, we present the "Small College Class" edition, with seasoned educators who have expertise teaching smaller classes at their university or college.

SPECIAL SESSION **Understanding NSF Funding Opportunities** Fri. 1:45 – 3:00 Lone Star A3

Chair: Scott Grissom, *National Science Foundation*
Participants: Victor Piotrowski, Sue Fitzgerald, Harriet Taylor and Joan Peckham, *NSF*

We highlight programs in the National Science Foundation's Division of Undergraduate Education, Office of Cyberinfrastructure and Directorate of Computer and Information Science and Engineering. The focus is on providing descriptions of several programs of interest to college faculty, and discussing the requirements and guidelines for programs in these areas. It includes a description of the proposal and review processes as well as strategies for writing competitive proposals. Discussion from participants is encouraged.

SPECIAL SESSION **Role and Value of Quantitative Instruments in Gauging Student Perspectives in Computing Curriculum** Fri. 1:45 – 3:00 Lone Star A4

Chair: Ali Erkan, *Ithaca College*
Participants: Henry Walker, *Grinnell College*; Mark Guzdial, *Georgia Institute of Technology*; Stephen Cooper, *Stanford University*

We will discuss the role and value of quantitative instruments in gauging student perspectives in computer science education. After creating the appropriate context, we will engineering (CSE) students with art students to engage in joint engineering design and creative studio projects. These projects combine embedded system design with sculpture to create kinetic art. We believe that this is a natural pairing of two disparate disciplines, and one that provides distinct educational benefits to both groups of students. In this paper we describe the course content, the collaborative process, the materials used in the class, and experience with a pilot version of the course taught in Fall 2009 at the University of Utah.

2:10 ***Cooperative Expertise for Multidisciplinary Computing***

Ursula Wolz, *The College of New Jersey*; Lillian Cassel, *Villanova University*; Thomas Way, *Villanova University*; Kim Pearson, *The College of New Jersey*

We present outcomes of an experiment to implement a model of cooperative expertise at two geographically separated institutions through three courses (two at one institution, one at the other), by faculty in computer science, media and English. Results reported include faculty analysis of student achievement in each course and student surveys of attitudes toward multidisciplinary collaboration. Overall, student learning and attitudes are enhanced by the collaborative experience.

2:35 ***Teaching Biologists to Compute using Data Visualization***

Kay Robbins, David Senseman and Elizabeth Pate, *University of Texas at San Antonio*

The accelerating use of computation in science continues to widen the gap between student skills and expectations. This paper describes an alternative to traditional approaches that introduces students to computing using data analysis and visualization with MATLAB. Our goal is produce computationally qualified young scientists by teaching a highly relevant computational curriculum early in their college career. The course, which integrates writing, problem-solving, statistics, visual analysis, simulation, and modeling, is designed to produce students with usable data analysis skills.

The τέχνη method is an approach to undergraduate computer science education that is based on cognitive constructivism, in the sense of Piaget, and which invokes several course design directives that include re-combining art and science, problem-based learning, problem selection from the visual problem domain, and cognitive apprenticeship. The paper describes a new τέχνη course in data structures. It includes a full comparative assessment of the realized improvement in student problem solving capability and, for the first time, cognitive authenticity in problem selection, in that the course problem is a variation on a very recent research result.

2:10 *Student Attitudes and Motivation for Peer Review in CS2*

Scott Turner, *UNC Pembroke*; Manuel Perez-Quinones, *Virginia Tech*; Stephen Edwards, *Virginia Tech*; Joseph Chase, *Radford University*

CS students need practice with vital concepts and professional activities. Peer review is one way to meet these goals. In this work, we examine the students' attitudes towards and engagement in the review process. Using three groups in two CS2 classes (one reviewing their peers, one reviewing the instructor, and one completing small design or coding exercises), we measured their attitudes and the number of reviews completed. We found moderately positive attitudes that were not significantly different between groups. We also saw a lower completion rate for those reviewing peers than for the other groups. The students' internal motivation was not shown to be strongly related to our other measures. Overall, our results show a need for external motivation to engage students in peer reviews.

2:35 *Applying Data Structures in Exams*

Briana Morrison, *Southern Polytechnic State University*; Mike Clancy, *University of California, Berkeley*; Robert McCartney, *University of Connecticut*; Brad Richards, *University of Puget Sound*; Kate Sanders, *Rhode Island College*

It is important for students to be able to select and apply the appropriate data structure for the problem at hand. Testing this knowledge on exams can be difficult, however. We examined 59 data structures final exams and found only 36 that contained questions involving the application of data structures. To promote assessment of this knowledge in the data structures course, we present a framework for classifying "apply" exam questions, with illustrations from the exams collected. We then show how a number of questions can be developed by varying a single rich apply question along the dimensions of this framework.

In this paper, we describe a programmable emulator for the Princeton IAS/Von Neumann machine. We describe our efforts to make the emulator as faithful as possible to the original, preserving the quirks and eccentricities of the machine. We also describe our efforts to make the tool user-friendly and robust, suitable for undergraduate architecture and programming classes as a teaching tool. The emulator permits users to write non-trivial programs in IAS assembly code or machine code. We present some examples here, and discuss assignments from its first use in two undergraduate classes. IASSim is a Java application publicly available at no cost.

2:10 *A Full System x86 Simulator for Teaching Computer Organization*

Michael Black and Priyadarshini Komala, *American University*

This paper describes a new graphical computer simulator developed for computer organization students. Unlike other teaching simulators, our simulator faithfully models a complete personal computer, including an i386 processor, physical memory, I/O ports, floppy and hard disks, interrupts, timers, and a serial port. It is capable of running PC software such as FreeDOS and Windows, and can run as a Java applet. Graphical user interfaces allow students to view and modify the processor, memory, disks, and hardware devices at runtime. The simulator includes a processor development utility that allows students to design their own datapath and control units, and run their custom processor alongside the x86 processor.

2:35 *IBCM: The Itty Bitty Computing Machine*

Aaron Bloomfield and William Wulf, *University of Virginia*

We present the development and implementation of the Itty Bitty Computing Machine (IBCM), a machine language designed specifically to be taught to lower-level undergraduate students. The presentation of the material takes about one-week of lecture, and allows understanding of all the concepts of machine language without having to deal with the complexity of a modern machine language, such as x86 and MIPS. A number of pedagogical aspects are addressed concisely via IBCM, such as treating all data as untyped and performing arithmetic on instructions. While we are not the first to introduce a short machine language module, we provide a number of benefits over older versions. All of the necessary materials, including compilers, simulators, and documentation, are freely available online.

emphasized possible future careers that rely on computer technology. During the week-long camp middle school girls created an original computer project, visited campus computer labs and listened to invited speakers – all with computer technology as the central theme. This paper discusses the program, girls' reactions to their experiences, and the projects they created during the summer camp.

2:10 ***Camps on a Shoestring: How We Survived a Summer***

Deborah Dunn, Robert Strader and Michael Pickard, *Stephen F. Austin State University*

As we are well aware, there has been a significant nationwide decline in enrollment for computer science programs, as well as other STEM fields. One of the primary reasons the lack of participation and diversity in the STEM fields is becoming increasingly important is the potentially adverse effect it may have on the U.S. work force. Many successful programs have been put in place to combat this decline. But how do the small regional universities with limited resources and a limited "audience" (with limited resources) contribute to the field? In this paper we will discuss the mechanisms that may be implemented which will allow more educators to become a part of the solution.

2:35 ***Introducing Computer Science to K-12 through A Summer Computing Workshop for Teachers***

Jiangjiang Liu, Cheng-Hsien Lin, Ethan Philip Hasson and Zebulun David Barnett, *Lamar University*

In this paper, we describe a one-week summer computing workshop for teachers to improve computer science education in K-12. Our workshop focuses on using Scratch to introduce computing concepts to teachers in computer, technology, math, and science at all K-12 levels to expose students computing at their early age and to reach more students. During the workshop, the teachers developed curriculum materials for the subjects they will teach in the following semesters with the help of our workshop tutors. We present our workshop strategies, lessons learned, and assessment results in this paper.

Bob Chesebrough, *Intel*

Information about supporter sessions can be found starting on page 68 of this program.

SUPPORTER SESSION **Google** Fri. 1:45 – 3:00
Dallas D3

Google's Education Initiatives for 2011
Multiple Presenters, *Google*

Information about supporter sessions can be found starting on page 68 of this program.

Friday, 3:00 PM to 3:45 PM

Coffee Break & Exhibits Lone Star BC

Friday, 3:00 PM to 4:30 PM

NSF Showcase #4 Lone Star BC

Computational Art and Creative Coding: Teaching CS1 with Processing *(CCLI/TUES)*: Ira Greenberg, Deepak Kumar, Dianna Xu

Presentation of the 2011 AlgoViz Awards *(NSDL)*: Cliff Shaffer

Demonstration of Remote Robotic Exploration and Experimentation *(CI-TEAM)*: William Smart, Bruce Maxwell

Modular CS1 from the Inside Out *(CPATH)*: Christine Alvarado, Zach Dodds

See the separate showcase bag insert for more information.

In this panel, three computer science professors who have completed a total of five Fulbright grants in the last 12 years offer their experiences, anecdotes, and insights of the Fulbright Scholar program, the flagship academic exchange program of the U.S. Department of State. The goal of the panel is to promote and inform the CS Education community about benefits of the Fulbright Scholar program, address questions or misconceptions regarding the program, and present realistic expectations for both the application process and the program itself. Each of the three panelists will present a 15 minute overview of their Fulbright experiences, leaving ample time for an interactive question and answer period.

SPECIAL SESSION **CS Principles: Piloting a New Course at National Scale** Thur. 3:45 – 5:00 Lone Star A3

Chair: Owen Astrachan, *Duke University*

Participants: Tiffany Barnes, *University of North Carolina*, Charlotte; Daniel D. Garcia, *University of California, Berkeley*; Jody Paul, *Metropolitan State College of Denver*; Beth Simon, *University of California, San Diego*; Lawrence Snyder, *University of Washington*

Since 2008, NSF and The College Board, have been developing a "Computer Science: Principles" curriculum to "introduce students to the central ideas of computing and CS, to instill ideas and practices of computational thinking, and to have students engage in activities that show how computing and CS change the world". We report on the initial pilot of the CS Principles curriculum at 5 universities in 2010/11. The instructors from the pilot schools will describe their classes, the piloting experience (teaching under a microscope), and successes and failures. Emphasis will be on: mapping the CS Principles curriculum to a college's specific needs, and how others can use or modify the existing materials for pilots at their schools.

attractive hands-on laboratory platform in network courses. We present classroom and research lab evidence for the usefulness of this platform as a network education tool, and discuss its significance in the context of a wide spectrum of competitor systems. This project is part of a larger effort to bring cost-effective, hands-on embedded system laboratory experiences into systems courses throughout the undergraduate computer science core.

4:10 ***Introducing Networking and Distributed Systems Concepts in an Undergraduate-Accessible Wireless Sensor Networks Course***
Sami Rollins, *University of San Francisco*

The field of wireless sensor networks (WSN) is growing rapidly, but curriculum in the field is fairly limited. Most courses reach only advanced graduate students. Undergraduate students lack the background to digest the topics and assignments of a standard WSN course. In this work, we present our approach to teaching WSN to undergraduates. We discuss a unique, integrated approach to introducing relevant distributed systems and networking concepts in the context of WSN applications. Our experience suggests that there is ample opportunity to expand curricula in sensor networking and reach a broader population of students.

4:35 ***Follow the River and You Will Find the C***
Jae Woo Lee, Michael S. Kester and Henning Schulzrinne, *Columbia University*

We present a one-semester transition course intended to bridge the gap between a Java-based introductory sequence and advanced systems courses. We chose to structure our course as a series of lab assignments that, while independent, are also milestones in a single main project, writing a web server from scratch. By anchoring the course on a single real-world application, we were able to provide depth, instill good programming practices, give insight into systems, and generate excitement.

responsibility and information literacy. Its target audience, first-year, non-computer science majors, learn what they need to know to use technology safely, effectively, efficiently, and ethically. The course is grounded in active learning and critical thinking. It is a radical alternative to a traditional software packages approach. The paper documents the need for this course and its blend of content and pedagogy. Data from three years of offering the course provide an assessment of its effectiveness. A review of SIGCSE literature in the last ten years finds no representation of this creative approach.

4:10 *Exploring the Appeal of Socially Relevant Computing: Are Students Interested in Socially Relevant Problems?*

Cyndi Rader, Doug Hakkarinen, Barbara Moskal and Keith Hellman, *Colorado School of Mines*

Prior research indicates that today's students, especially women, are attracted to careers in which they recognize the direct benefit of the field for serving societal needs. Traditional college level computer science courses rarely illustrate the potential benefits of computer science to the broader community. This paper describes a curricula development effort designed to embed humanitarian projects into undergraduate computer science courses. The impact of this program was measured through student self-report instruments. Through this investigation, it was found that students preferred projects that they perceived as "fun" over the projects that were social in nature.

4:35 *Increasing Engagement and Enrollment in Breadth-First Introductory Courses Using Authentic Computing Tasks*

Ryan McFall and Matthew DeJongh, *Hope College*

Lab assignments for breadth-first introductory courses tend to focus on learning to program or simulations of program execution. These activities unfortunately fail to build on the foundations laid by a breadth-first approach, and serve to perpetuate the computer science = programming misperception. We have developed a set of lab activities based on what we call authentic computing tasks: everyday tasks that students want to know how to accomplish. Explicit connections are made between these authentic computing tasks and the concepts covered in the lecture portion of the course. We have seen dramatic increases in enrollment, and have evidence that students see the connections, rather than coming to believe that performing computing tasks well is the essence of computer science.

curriculum. We try to structure the debate along a number of dimensions and then present the solution that we adopted for an engineering-oriented curriculum. We added an introduction to parallel programming to the list of mandatory classes in the 2nd semester. The class exposes students to three styles of parallel programming: threads with shared memory, CSP-style message passing, and OpenMP-based parallel programming. Within these models, the class aims to focus the student's attention on communication as the key issue in parallel programs.

4:10 ***Encouraging Parallel Thinking through Explicit Coordination Modeling***

Sirong Lin and Deborah Tatar, *Virginia Tech*

Parallel thinking is a mindset that allows people to create support for activities that happen concurrently in a program. It crosscuts extant computer science boundaries, including parallel processing, network programming and multi-user systems, indeed, any system that involves the distribution and reintegration of work. Recent efforts to integrate parallelism across the CS curriculum begin to address the support of parallel thinking. We approach the pedagogy of parallel thinking by teaching students to model coordination explicitly using a specialized coordination language. We report a study of an experimental class taking this approach, finding that advanced CS students lack a good understanding of coordination but that the explicit modeling of coordination can address this lack.

4:35 ***Modules in Community: Injecting More Parallelism into Computer Science Curricula***

Elizabeth Shoop, *Macalester College*; Richard Brown, *St. Olaf College*

Given the recent emergence of multi-core and distributed computing that is transforming mainstream application areas in industry, demand is rising for teaching more parallelism and concurrency in CS curricula. We argue for teaching these topics incrementally in CS courses at all undergraduate levels, and propose a comprehensive approach involving flexible teaching modules with experiential programming exercises, technical and instructor supplementary materials, and an online community of educators to support adopters and module contributors. Progress on developing these materials and online resources is reported.

In order to attract a diverse audience, we developed a summer program based on culturally-relevant themes that appealed to our two target audiences, females and Latina/os. This paper describes our success in developing and implementing a computing curriculum and recruiting materials for a summer camp integrating animal conservation and Mayan culture. Scratch programming was used to engage students in creating animations about animals and Mayan culture, allowing them an interdisciplinary experience that combined programming, culture, art, and writing. Our recruiting efforts resulted in an application pool 73% female and 67% Latina/o, with only 6.5% in neither group. The camp resulted in double the students choosing CS as their top career and triple considering it.

4:10 ***K-12 Game Programming Course Concept using Textual Programming***

Antti-Jussi Lakanen, Ville Isomottonen and Vesa Lappalainen, *University of Jyvaskyla*

Several programming environments have been constructed to facilitate novice programming at K-12 and CS0/CS1 levels. The environments can be roughly divided into those using visual or textual programming. This paper presents a K-12 game programming course concept based on textual programming. The concept is based on an easy-to-use C# library, called Jypeli, built on top of Microsoft XNA Framework. The library tries to maintain advantages of visual programming and avoid challenges of textual programming. In particular, the library helps beginners to program their first games in a short period of time and without a heavy syntactic load. The course concept and an initial evaluation consisting of student feedback and a literature rationale are presented.

4:35 ***Introducing Computational Thinking in Education Courses***

Aman Yadav, Ninger Zhou, Chris Mayfield, Susanne Hambrusch and John T. Korb, *Purdue University*

As computational thinking becomes a fundamental skill for the 21st century, K-12 teachers should be exposed to computing principles. This paper describes the implementation and evaluation of a computational thinking module in a required course for elementary and secondary education majors. We summarize the results from open-ended and multiple-choice questionnaires given both before and after the module to assess the students' attitudes towards and understanding of computational thinking. The results suggest that given relevant information about computational thinking, education students' attitudes towards computer science becomes more favorable and they will be more likely to integrate programming. A barrier to providing such activities is the effort required to set up the programming environment. Testing is an important component to writing good software, but it is difficult to motivate students to write tests. In this paper we describe and evaluate CodeWrite, a web-based tool that provides drill and practice support for Java programming, and for which testing plays a central role in its use. We describe how we have used CodeWrite in a CS1 course, and demonstrate its effectiveness in providing good coverage of the language features presented in the course.

4:10 *E-Learning Experience using Recommender Systems*

Jesus Bobadilla, Antonio Hernando and Angel Arroyo, *Universidad Politécnica de Madrid*

This paper presents the results obtained using a real e-learning recommender system where the collaborative filtering core has been adapted with the aim of weighting the importance of the recommendations in accordance with the users' knowledge. In this way, ratings from users with better knowledge of the given subject will have greater importance over ratings from users with less knowledge. The results obtained show a notable improvement regarding traditional collaborative filtering methods and suggest balanced weightings between the importance assigned to users with more or less knowledge.

4:35 *A Web-Based Generation and Delivery System for Active Code Reading*

Daniel Hoffman, Ming Lu and Timothy Pelton, *University of Victoria*

Learning to write computer software is difficult. In Computer Science courses, we ask students to write a lot of code. All too often, the resulting code quality is poor: the students have many misunderstandings about their programs and about the features of the programming language they are using. While it is widely believed that students would write better code if they spent more time reading code, it is difficult to get students to read code effectively. We present a web-based application intended to support generation and delivery of quizzes designed to evaluate and improve code reading skills.

Information about supporter sessions can be found starting on page 68 of this program.

SUPPORTER SESSION **Microsoft** Fri. 3:45 – 5:00 Dallas A3

Microsoft .NET Gadgeteer: A New Way to Create Electronic Devices
Nicolas Villar, *Microsoft Research*

Information about supporter sessions can be found starting on page 68 of this program.

Friday, 5:10 PM to 5:55 PM

SIGCSE Business Meeting Fri. 5:10 – 5:55 Lone Star A4

Friday, 6:00 PM to 6:45 PM

CCSC Business Meeting Fri. 6:00 – 6:45 Lone Star A3 *enrollment may be cancelled. Check the registration website or the registration desk at the Symposium for availability of workshops in which you might be interested.*

13. Developing an Android Mobile Application for the Google App Engine Cloud	Lone Star A2
14. Computer Science Unplugged And Outreach Activities	Dallas D3
15. Basic 2D Graphics and Animation Concepts for iPhone/iPad Games	Lone Star A4
16. Making the Most of Undergraduate Research	Dallas A1
17. 2D Game Design and Development 101	Dallas A2
18. Listening to Linked Lists: Using Multimedia to Learn Data Structures	Dallas A3
19. Multithreading (Pretty) Early for Everyone: Parallelism and Concurrency in Second-Year Data-Structures	Dallas D1
20. Reinvigorating CS1 with Creative Web 2.0 Programming	Dallas D2
21. Computational Art and Creative Coding: Teaching CS1 with Processing	Lone Star A3
22. Build Your Own Blocks: A Scratch Extension for CS Courses for Non-Majors	State Room 1
23. Broadening Participation in Computer Science with Scratch, Jeroo, and GridWorld	State Room 2
24. Greenfoot: Introducing Java with Games and Simulations	State Room 3
25. Using the FIRST (robotics) Experience to Reach Pre-College Students via Near Peer	*Canceled*

		Science Program	Project Showcase	of Sound	and Modules
THU	Noon	First-Timers’ Lunch (Lone Star A1)			
	1:45	Educational Advances in AI	Learning Through Open Source Participation	The CS10K Project	Algorithms
	3:45	Successful K-12 Outreach Strategies	Setting the Stage for Computing Curricula 2013	It Seemed Like a Good Idea at the Time	Peer Teaching and Tutoring
	5:10	Birds-Of-A-Feather, Flocks I & II (See schedule for rooms and times)			
	7:00PM	SIGCSE Reception (Grand Hall) 7:00-8:00PM			
FRI	7:15AM			Brkfst w/ Alice	
	8:30	Plenary Session / Keynote (Dallas BC)			
	10:45	Top Issues in Successful UG Research Exper.	Report on Qual. Research Methods Workshop	NCATE Standards for Secondary CS Teachers	Operating Systems and Databases
	Noon	Lunch (On Your Own); UPE National Meeting (12:10, Dallas BC)			
	1:45	Teaching Tips, Small College Class Edition	Understanding NSF Funding Opportunities	Role and Value of Quantitative Instruments	Computer Architecture Teaching Tools
	3:45	CS Fulbright Experiences Abroad	CS Principles: Piloting a New Course at Nat Scale	Networks	K-12 Instruction
	5:10			SIGCSE Bus. Mtg.	
	6:00		CCSC Bus. Mtg.		
	7:00PM	Workshops 13–25 (See schedule for rooms)			
SAT	7:00AM			Greenfoot Brkfast	
	8:30	Teaching and Studying Novice Programmers	Scratching the Surface: Infusing Computing in K-12	Nifty Assignments	Teacher Endorsement and Preparation
	10:55	Rediscovering the Passion, Beauty, Joy, & Awe: Part 4	Progress in Surfacing CS in STEM	NSF/IEEE-TCPP Curric Init on Para & Dist Computing	Cooperative Learning
	12:10	Robot Hoedown Finale (Grand Hall A)			
	12:30	Luncheon (Dallas BCD)			

Innovations	Microsoft	Architecture	Society	
First-Timers' Lunch (Lone Star A1)				
Software Engineering	Supporter Session: **Intel**	Assessing and Reviewing	Teaching Programming: Non-Trad. Appr.	Supporter Session: **Amazon**
Musical, Social, and Intelligent Robots	Supporter Session: **IBM**	Parallel / Concurrent Programming	Recruitment and Retention	
Birds-Of-A-Feather, Flocks I & II (See schedule for rooms and times)				
SIGCSE Reception (Grand Hall) 7:00-8:00PM				
Plenary Session / Keynote (Dallas BC)				
CS 1: Tools	Supporter Session: **Microsoft**	Discrete Mathematics	Computational Thinking	
Lunch (On Your Own); UPE National Meeting (12:10, Dallas BC)				
Summer Experiences	Supporter Session: **Intel**	Data Structures / CS 2	Computing in the Arts and Sciences	Supporter Session: **Google**
Web-Based Tools	Supporter Session: **Microsoft**	Parallelism Across the CS Curriculum	Relevant Computing	Supporter Session: **Google**
Workshops 13–25 (See schedule for rooms)				
Expanding the Community	Mobile Computing	Communication Skills	Stud. Research Competition - Undergraduate	Stud. Research Competition - Graduate
Researching and Evaluating Teachers	Software Design and Development	(Session moved to Dallas A3)	Intro CS: Panoptic Views: Moved to Lone Star A1	
Robot Hoedown Finale (Grand Hall A)				
Luncheon (Dallas BCD)				
Workshops 26–35 (See schedule for rooms)				

PANEL Scratching the Subject Surface: Infusing Computing into the K-12 Curriculum

Sat. 8:30 – 9:45
Lone Star A3

Moderator: Ursula Wolz, *The College of New Jersey*
Panelists: Youwen Ouyang, *CSU San Marcos*; Scott Leutenegger, *University of Denver*

Two identified problems with K-12 computing curriculum are how to (1) bring computing into an already over-burdened curriculum, (2) provide substantive professional development for teachers. The power of computing lies in its broad applicability to facilitate creativity in other domains. The panelists will describe using Scratch in language arts, science and social studies where they show middle and high school students how computing is essential, fun to learn and that programming is an accessible tool for creativity. The panelists will present using Scratch for humane games, scientific modeling and interactive storytelling.

SPECIAL SESSION Nifty Assignments

Sat. 8:30 – 9:45
Lone Star A4

Chairs: Nick Parlante and Julie Zelenski, *Stanford University*
Participants: Dave Feinberg, *Carnegie Mellon University*; Keith Schwarz, *Stanford University*; Michelle Craig, *University of Toronto*; Stuart Hansen, *University of Wisconsin – Parkside*; Michael Scott, *University of Texas at Austin*

The Nifty Assignments special session presents successful assignments ready for adoption.

PAPERS Teaching and Studying Novice Programmers

Sat. 8:30 – 10:10
Lone Star A2

Chair: Mark Johnson, *Central College*

8:30 ***Experience Report: Getting Novice Programmers to THINK about Improving their Software Development Process***
Tammy VanDeGrift, Tamara Caruso and Natalie Hill, *University of Portland*
Beth Simon, *University of California, San Diego*

Expertise is developed through self-reflection and making useful plans for improvement. Traditional novice programming assignments require neither of these skills. Could we get students to think about improving their software development processes? What would they

Good error messages are critical for novice programmers. Many projects attempt to rewrite expert-level error messages in terms suitable for novices. DrScheme's language levels provide a powerful alternative through which error messages are customized to pedagogically-inspired language subsets. Despite this, many novices still struggle to work effectively with DrScheme's error messages. To better understand why, we study the effectiveness of DrScheme's error messages. Unlike existing work in this area, we study messages at a fine-grained level by analyzing the edits students make in response to various classes of errors. We present our rubric, apply it to a course-worth of student lab work, and describe what we have learned about using the rubric effectively.

9:20 *Which Aspects of Novice Programmers' Usage of an IDE Predict Learning Outcomes?*

Gregory Dyke, *École des Mines de Saint-Étienne*

We present the preliminary analysis of a study whose long term aim is to track IDE usage to identify novice-programmers in need of support. Our analysis focused on the activity of 24 dyads over 3 sessions. We correlated frequencies of events such as use of code generation and of the debugger with assignment grades, final exam grades, and the difference in rankings within dyad on the final exam. Our results show several significant correlations. In particular, code generation and debugging are correlated with the final grade, and running in non-debug mode is correlated with differences in ranking. These results are encouraging as they show that it is possible to predict learning outcomes with simple frequency data and suggest more complex indicators could achieve robust prediction.

9:45 *The Novice Programmer's 'Device To Think With'*

David Barnes, *University of Kent*; Dermot Shinners-Kennedy, *University of Limerick*

We present some ideas for course material for the introductory teaching of programming that are based on the principle of allowing the students to be the domain experts. The idea is that the students' familiarity with the domain of discourse will make course material more motivating, and that it will be more likely that they will be able to model the concepts and artifacts being discussed. This approach thereby seeks to scaffold the students' understanding of programming-related concepts. For reasons discussed in the paper, we have chosen mobile phone technology for this discussion, but there is no reason why the same principles should not be applied to other culturally-accessible domains.

Soft skills such as communication, teamwork, and organization are important to students' future success. Faculty members know it, students know it, and employers are explicitly asking for these skills. Are CS departments responsible to teach these skills? If so, where in the curriculum should they be covered? This paper explores the soft skills that employers want, and possible places to include the teaching of those skills in the curriculum. It then shows how an extensive set of soft skills were incorporated into a service learning course for the students in the CS department at Point Loma Nazarene University. Finally, it makes suggestions as to how other service learning or capstone courses could be altered to afford more opportunity for soft skill education.

8:55 ***Interdisciplinary Teaching: Introductory Programming via Creative Writing***

Mary Elizabeth Jones, Melanie Kisthardt and Marie Cooper, *Immaculata University*

Seizing and retaining student interest in programming is a difficult task. Teaching introductory programming via creative writing connects the writing of a story to the programming process. Creative concepts are connected to an equivalent programming concept. Student pairings are assigned by matching a student with analytical tendencies with a student majoring in one of the humanities disciplines. They create a story and design/implement an animation using the Alice. This research recognizes the creative nature of programming, invites students who would not consider studying programming to potentially identify an unrecognized talent, and attempts to develop a new approach for teaching introductory programming. This research and teaching is sponsored by a NSF Grant.

9:20 ***Gumshoe: A Model for Undergraduate Computational Journalism Education***

Sarah Monisha Pulimood, Donna Shaw and Emilie Lounsberry, *The College of New Jersey*

This paper describes a collaboration between computer science and journalism students and professors at our small, primarily undergraduate college, and a large metropolitan newspaper. Our students' work was a catalyst for a hard-hitting series of investigative stories, with far-reaching consequences. The Gumshoe project is a model for computational journalism at an undergraduate institution. The project demonstrates that when computer

significantly based on intuition. In this paper we present an engineering approach to writing, in which engineering principles are used to teach and assess writing. The results are as good, and much better in some cases, and teaching and learning become easier.

PAPERS Teacher Endorsement and Preparation — Sat. 8:30 – 10:10 Dallas A1

Chair: Sherri Goings, *Carleton College*

8:30 *Teaching Computer Science Majors about Teaching Computer Science*

Tim Bell, *University of Canterbury*; Lynn Lambert, *Christopher Newport University*

This paper describes the design, implementation, and evaluation of a course teaching Computer Science majors about teaching Computer Science. The course was designed to address the need for teachers and resources to support rapid changes in topics being taught in high schools. It also helped prepare students for research in Computer Science Education, and for careers involving computing and education. The course is described in detail, and is evaluated based on student feedback and the outcomes from the course.

8:55 *Implementing a Computer Science Endorsement Program for Secondary School Teachers*

Christopher Whitehead, Lydia Ray, Shamim Khan, Wayne Summers and Rodrigo Obando, *Columbus State University*

Computer Science has been increasing productivity in all sectors of our economy for decades. The capacity of our education system to produce adequate CS graduates will determine the US's ability to expand and maintain the IT infrastructure vital for economic growth. Unfortunately, there has been a serious decline in the number of high school graduates applying to study Computer Science over the last decade. One strategy that can possibly reverse this decline is to create an awareness of Computer Science as an interesting and prospective discipline within the secondary school system. This objective can only be achieved if we have skilled CS teachers in our schools. This paper describes one method for realizing this goal-a Computer Science endorsement program for secondary school teachers.

local community through the Disciplinary Commons for Computing Educators (DCCE) project. The DCCE project is an effort to exploring ways of supporting HS CS teachers through creating a local community and promoting teacher reflection. DCCE achieves this goal through an academic-year-long program where a cohort of CS teachers engages in collaborative portfolio creation and peer observation of classroom teaching. We describe the design of the DCCE activities and present preliminary results from initial evaluations. Our short-term evaluation shows that this project was successful in creating a local community of CS teachers, promoting teacher reflection and advancing pull transfer of teaching practice

9:45 ***A Study on Attitudes and Emphases in Computer Science Teacher Preparation***

Noa Ragonis, *Beit Berl College, Technion - Israel Institute of Technology*; Orit Hazzan, *Technion - Israel Institute of Technology*; Judith Gal-Ezer, *The Open University of Israel*

This paper focuses on the development and implementation of computer science (CS) teacher preparation programs. The paper presents the second stage of a study in which we approached relatively wide community of CS teacher educators (from Israel, Europe, and the USA) and explored its perspective on one element of teacher preparation programs – the Methods of Teaching CS (MTCS) course. The conclusions from this stage can be viewed as a comprehensive framework for the design of an MTCS course, both in terms of topics to be included in the course as well as the relative weight (in time) that is to be dedicated to each topic.

PAPERS Expanding the Community

Sat. 8:30 – 10:10
Dallas A2

Chair: Chun Liew, *Lafayette College*

8:30 ***The Images of Computing: Engaging Undergraduates in the Broad Issues of Computer Science***

Carol Frieze, *Carnegie Mellon University*

In this paper we describe a new "research and action" based course designed to give undergraduate students the opportunity to think beyond the classroom, to reach out and examine some of the broader issues surrounding computing. "Understanding and Broadening the Images of Computing" researches the images, the realities and the science. Students who wish to enter the discipline must overcome significant technological and educational barriers to succeed. In an attempt to help, we are engaged in a three-year research project to build and deploy an educational infrastructure for blind and visually impaired middle and high school students. We present here two preliminary results from this research: 1) a new auditory programming environment called Sodbeans, a programming language called Hop, and a multi-sensory (sound and touch) curriculum, and 2) an empirical study of our first summer camp with the blind students. Results show students reported a significant increase in programming self-efficacy after participating in our camp.

9:20 *Enhancing Participation and Education in CS through Guided Research Projects in Underserved Communities*

Ermine Teves, *Carnegie Mellon University*; Yonina Cooper, *Carnegie Mellon University in Qatar*; M. Bernardine Dias, Sarah Belousov and M. Freddie Dias, *Carnegie Mellon University*

While the needs and applications for computing technology have been growing, the enrollment and interest in Computer Science (CS) at the university level has not been growing in proportion. The increasing prevalence of globalization requires a new set of skills for future technology leaders such as the ability to work well in multidisciplinary and globally distributed teams, create innovative solutions for problems that arise in unfamiliar settings, and think outside the box to solve a variety of problems while building effectively upon related work in the literature. We report our experience in designing and deploying an innovative internship that addresses these issues and seeks to enhance participation and education in CS through guided research projects in underserved communities.

9:45 *Reaching Out to Aid in Retention: Empowering Undergraduate Women*

Rebekah Overdorf and Matthew Lang, *Moravian College*

Creating programs that engage undergraduate women with the broader community and encourage them to take an active role in changing the underrepresentation of women in computer science can effectively address both retention and recruitment of women in the discipline. This paper is an experience report describing the creation and outcomes of an outreach program for K--12 girls run entirely by undergraduate women. The contributions of this paper are the description of the creation of a successful student-led outreach program and a set of active-learning modules for K--12 students that illustrate advanced topics.

Hollingsworth, *Elon University*

A new class of mobile devices is appearing that combines the functionality of an eBook reader with a web browser, and many are based on the same programming frameworks as the smart phones. We report on our successes with mobile device programming courses. We report on the accomplishments of students developing software to leverage the newer multi-function devices in supporting CS education. We have developed software that allows students to compile and test programs written in Java/C++ that can be invoked from these devices and software allowing these devices to function like Tablet PCs. We discuss our current efforts to use these mobile devices in CS education.

8:55 ***Human Computer Interaction that Reaches Beyond Desktop Applications***
Susan Loveland, *Adams State College*

Recently, several frameworks have been developed for writing mobile and web applications in Java making the development of web and mobile applications accessible to HCI students with only a CS1 Java background. In this paper we describe using student projects based on the Google Android mobile platform and Google's Web Toolkit to provide students with experience designing and implementing user interfaces for mobile and web applications. Specific examples demonstrate how programming on these platforms reinforces standard HCI topics. As a result of being able to learn mobile device programming in the context of "cool" Google platforms, students expressed increased interest in studying HCI.

9:20 ***App Inventor and Real-World Motivation***
David Wolber, *University of San Francisco*

App Inventor is a visual "blocks" language for creating mobile apps. As part of a Google pilot program, App Inventor was taught to university students in a core curriculum course at the University of San Francisco. This paper introduces App Inventor and the course, focusing on how the language facilitated interactions with the world outside of the classroom.

9:45 ***Smart Smartphone Development: iOS versus Android***
Mark Goadrich, *Centenary College of Louisiana*; Michael Rogers, *Northwest Missouri State University*

In a remarkably short timeframe, developing apps for smartphones has gone from an arcane curiosity to an essential skill set. Employers are scrambling to find developers capable of

STUDENT RESEARCH COMPETITION: UNDERGRADUATE Sat. 8:30 – 10:10 Dallas D2
Organizer: Ann Sobel, *Miami University of Ohio*

Saturday, 10:00 AM to 11:30 AM

NSF Showcase #5 Lone Star BC

Ensemble: Connecting Computing Educators *(NSDL)***:** Peter Brusilovsky, Steve Carpenter, Lillian Cassel, Lois M. L. Delcambre, Steve Edwards, Edward A Fox, Richard Furuta, Daniel D. Garcia, Gregory Hislop, Haowei Hsieh, Frank Shipman

Leveraging Map-based CI Resources for Middle School Earth Science Curriculum *(CI-TEAM)***:** Youwen Ouyang

Encouraging Digital Literacy, College Readiness and Persistence among Native Americans *(CI-TEAM)***:** Richard Mosholder

Computational Thinking as a Foundation for Interdisciplinary Undergraduate Education *(CPATH)***:** Suzanne Westbrook

See the separate showcase bag insert for more information.

Saturday, 10:10 AM to 10:55 AM

Coffee Break & Exhibits Lone Star BC

Panelists: Michelle Friend Hutton, *Stanford University*; Eugene Lemon, *Ralph J. Bunche High School*; Josh Paley, *Henry M. Gunn High School*

At SIGCSE 2007, Grady Booch exhorted us to share the "passion, beauty, joy and awe" (PBJA) of computing. This led to a series of room-packed sessions at the following three SIGCSE symposia to explore that idea from different perspectives. They have provided a forum for sharing best practices, curriculum changes and pedagogy suggestions. This year, we have invited three K-12 teachers who, collectively, have taught computing at an all-girls middle school, an under-served high school, and an affluent high school. This panel will drill down and understand the K-12 space, in terms of extolling the PBJA of computing.

SPECIAL SESSION **Progress in Surfacing Computer Science in STEM** Sat. 10:55 – 12:10 Lone Star A3

Chair: Susan Rodger, *Duke University*

Participants: Mark Stehlik, *Carnegie Mellon University*; Cameron Wilson, *ACM*; Chris Stephenson, *Computer Science Teachers Association*

Major policy issues still exist for K-12 computer science education. There is deep confusion about computer science teacher certification, courses, gender and diversity gaps in students, and whether computer science courses “count” toward a student's graduation requirements. We present groundbreaking research reflecting how computer science education is treated in each of the 50 states coupled with initiatives to transform the national education policy landscape for K-12 computer science education. It will connect the broad SIGCSE community by giving them new data and a call to action to get involved in a new coalition called “Computing in the Core.”

SPECIAL SESSION **NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributive Computing: Core Topics for Undergraduates** Sat. 10:55 – 12:10 Lone Star A4

Chair: Sushil Prasad, *Georgia State University*

Participants: Arnold Rosenberg, *Colorado State University*; Alan Sussman, *University of Maryland*; Chip Weems, *University of Massachusetts*; Anshul Gupta, *IBM T.J. Watson Research Center*; Andrew Lumsdaine, *Indiana University*

A working group from IEEE, NSF, and the sister communities, including ACM, is

Bothell

Games assignments are increasingly popular in computer science education. This paper advocates and analyzes the inclusion of board, card, and dice games as programming assignments in introductory programming courses. The simple interface and strategy-based play of these types of games complement the immersive multimedia and agility-based play of video games. The implementation of board, card, and dice games typically demands less background knowledge from the instructor and offers fewer opportunities for extraneous work by the students. The paper lists 32 specific games that are suitable for teaching the major topics in CS1/2 and discusses the implementation of some of these games and their successful use as programming projects.

11:20 *A Snapshot of Current Practices in Teaching the Introductory Programming Sequence*

Jennifer Polack-Wahl, Stephen Davies and Anewalt Karen, *University of Mary Washington*

We present results from a nationwide survey of undergraduate computer science departments regarding languages and techniques taught in CS0, CS1, and CS2. This snapshot of 371 schools provides an intriguing look into the state of computing education today in the U.S., quantifying which practices are actually in common use. Among other things, the study reveals the great variety in CS0 approaches, the relative uniformity of CS1 and CS2 approaches, the dominance of Java as a language for the introductory major sequence, and the tendency for departments to teach CS1 and CS2 in a consistent manner, rather than exposing students to different ideas in each.

11:45 *Reviewing CS1 Exam Question Content*

Andrew Petersen, *University of Toronto - Mississauga*; Michelle Craig and Daniel Zingaro, *University of Toronto*

Many factors have been cited for poor performance of students in CS1. To investigate how assessment mechanisms may impact student performance, nine experienced CS1 instructors reviewed final examinations from a variety of North American institutions. The majority of the exams reviewed were composed predominantly of high-value, integrative code-writing questions, and the reviewers regularly underestimated the number of CS1 concepts required to answer these questions. An evaluation of the content and cognitive requirements of individual questions suggests that in order to succeed, students must internalize a large amount of CS1 content. This emphasizes the need for focused assessment techniques to

for incorporating Mercurial into introductory and advanced computing courses in the entire CS curriculum. Using Mercurial, instructors can create unique opportunities to engage students in collaborative, real-world projects and activities, giving them critical exposure to the expectations and assumptions prevalent in the software development community. Early introduction to version control provides students with an important foundation in both personal and collaborative development excellence, offering them a competitive edge in the marketplace and a superior understanding of software development best practice.

11:20 *LIFT: Taking GUI Unit Testing to New Heights*

Stephen Edwards, Jason Snyder and Manuel Perez-Quinones, *Virginia Tech*

The Library for Interface Testing (LIFT) supports writing unit tests for Java applications with graphical user interfaces (GUIs). Current frameworks for GUI testing provide the necessary tools, but are complicated and difficult to use for beginners, often requiring a significant amount of time to learn. LIFT takes the approach that unit testing GUIs should be no different than testing any other type of code. By providing a set of frequently used filters for identifying GUI components and a set of operations for acting on those components, LIFT lets programmers quickly and easily test their GUI applications.

11:45 *Promoting Creativity in the Computer Science Design Studio*

Katherine Cennamo, *Virginia Tech*; Sarah A. Douglas, *University of Oregon*
Mitzi Vernon, Carol Brandt and Brigitte Scott, *Virginia Tech*; Yolanda Reimer, *University of Montana*; Margarita McGrath, *Virginia Tech*

Advances in technology will require computer science professionals who are able to develop innovative software solutions. In order to identify techniques that can lead students to creative insights in their work, we have conducted an ethnographic study of the studio method as enacted in architecture, industrial design (ID), and human-computer interaction (HCI) classes. Our analysis of the activities conducted during studio critiques revealed that while the ID and architecture studios had a primary focus on experimentation, the primary emphasis of the HCI studios was on idea refinement. In this paper, we describe four barriers to creative thought observed in the HCI classrooms and identify ways that the architecture and ID instructors helped students to overcome similar challenges.

Pair programming is a programming technique where two programmers work together on the same programming task. Previous research has shown that it is effective for improving students' learning effectiveness. Much research has also been dedicated to determining effective pairing strategies. This paper discuss two empirical studies conducted to a) test the feasibility of using pair programming in introductory computer science courses and b) determine whether or not major-based pairing produces effective pairs. The results of these studies provide support for implementing pair programming in introductory courses and show that pairing of computer science and non-computer science students may produce less compatible pairs.

11:20 *Beyond Clickers: Using ClassQue for Multidimensional Electronic Classroom Interaction*

Steven Robbins, *University of Texas at San Antonio*

ClassQue is a classroom response system that goes beyond clickers to allow a wide selection of classroom interactions: teacher to individual student, teacher to all students, student to teacher and student to student. Questions are not restricted to multiple choice, and multiple questions can be pending at one time. One student can anonymously comment on another student's answer. After the class, students and teachers can receive reports of the classroom interactions. The current version of ClassQue is available for use in an environment in which each student is seated at a computer.

11:45 *Overcoming Barriers among Israeli and Palestinian Students via Computer Science*

Shiri Azenkot, *University of Washington*; Theodore Golfinopoulos, Adam Marcus, Alessondra Springmann and Jonathan Varsanik, *MIT*

The Middle East Education Through Technology (MEET) program is an organization based in Jerusalem that aims to empower Israeli and Palestinian students by teaching them computer science and business. From the perspective of MEET’s instructors, this paper describes how MEET uses computer science to foster professional and personal contact between Israeli and Palestinian high school students, two groups who otherwise would have little or no interaction with each other. MEET’s primary method of overcoming the barrier is teamwork: students are divided into groups that include both Israelis and Palestinians and are assigned software engineering tasks. We believe that MEET can serve as an example for other programs that overcome barriers between groups in the United States and other countries.

San Diego; Stephen Cooper, *Stanford University*

Single-instance workshops, where instructors are brought together to learn about a new technique or system so that they can possibly adopt it, are a common dissemination method for pedagogical and technical advances in education. Unfortunately, rarely do we see reports of their effectiveness. In this case study, we report on two NSF-funded workshops to support adoption of the Ubiquitous Presenter active learning classroom presentation system. Though only 44% of workshop attendees used UP in their classrooms, 65% of those used it repeatedly. Overall this impacted 1570 students. We reflect on the aspects of the workshops which seemed to promote, and hinder, instructor adoption and, finally, suggest some metrics for evaluating innovation dissemination workshops in general.

11:20 *Analysis of Undergraduate Teaching Evaluations in Computer Science*

Joshua T Guerin and Daniel Michler, *University of Kentucky*

Undergraduate teaching evaluations are widely believed to be biased. We analyzed the evaluations from our department for all classes from 2007 through 2009, looking for specific biases. We looked for correlations between evaluation numbers and whether the course was required; how much time students spent on the course; student classification (freshman, sophomore, etc.), and expected grades. While all of these show correlations with teaching evaluation responses, the effect of each factor is not consistent over all questions and varied strongly depending on the evaluation question and the course level.

11:45 *The Use of Evidence in the Change Making Process of Computer Science Educators*

Davide Fossati, *Carnegie Mellon University*; Mark Guzdial, *Georgia Institute of Technology*

This paper explores the issue of what kind of evidence triggers changes in the teaching practice of Computer Science educators, and how educators evaluate the effectiveness of those changes. We interviewed 14 Computer Science instructors from three different institutions. Our study indicates that changes are mostly initiated from instructors' intuition, informal discussion with students, and anecdotal evidence.

Saturday, 12:30 PM to 2:30 PM

SIGCSE Luncheon and Keynote Address Dallas BCD

Announcements, Awards and a Preview of SIGCSE 2012

Thomas J. Cortina, Symposium Co-Chair, *Carnegie Mellon University*
Ellen L. Walker, Symposium Co-Chair, *Hiram College*
Laurie Smith King, Program Co-Chair, *College of the Holy Cross*
David R. Musicant, Program Co-Chair, *Carleton College*

Keynote Address: Three Human Computation Projects

Luis von Ahn, *Carnegie Mellon University*

At the height of its construction, 44,733 people worked on the Panama Canal. The Great Pyramid of Giza required 50,000 workers and the Apollo Project 400,000. No matter what you put on this list, humanity's largest achievements have been accomplished with less than a few hundred thousand workers because it has been impossible to assemble (let alone pay!) more people to work together --- until now. With the Internet, we can coordinate the efforts of billions of humans. If 400,000 people put a man on the moon, what can we do with 100 million? My research aims to develop theories and build computer systems that enable massive collaborations between humans and computers for the benefit of humanity. I am working to develop a new area of computer science called human computation, which studies how to harness the combined power of humans and computers to solve problems that would be impossible for either to solve alone.

In this talk I will discuss several examples of my work in human computation, including reCAPTCHA, The ESP Game, and Duolingo. I have used these projects in many of my courses to illustrate state of the art concepts in computer science.

enrollment may be cancelled. Check the registration website or the registration desk at the Symposium for availability of workshops in which you might be interested.

26.	**Making Music With Scratch**	Lone Star A2
27.	**Hands-on Teaching Modules for Secure Web Application Development**	Lone Star A3
28.	**Storytelling in CS Education - Examples from Programming and Software Engineering Courses**	*Canceled*
29.	**Agility Training**	State Room 1
30.	**Testing Graphical User Interfaces in the Classroom**	State Room 2
31.	**Teaching a Consulting Course to Develop Communication and Leadership Skills**	*Canceled*
32.	**Reaching Out Internationally with Programs to Promote Computing to High and Middle School Students**	*Canceled*
33.	**Teaching with Greenfoot - From development of material to delivery in the classroom**	State Room 3
34.	**Exploring Wonderland: Teaching with Alice and Media Computation**	*Canceled*
35.	**Short Mobile Game Development Projects for CS1/2**	Lone Star A4

determined by conference attendee evaluations of their research projects. Initially, students use the interactive nature of a visual presentation to highlight different aspects of their research to individual evaluators. These presentations are evaluated on their quality, the significance of the work, and the clarity of the informal discussion. The semi-finalists present their contributions using the standard forum of conference presentation during a conference session. This venue provides selected audience attendees with another platform for evaluation, the student with experience in formal presentations, and conference participants with the opportunity to learn of ongoing, current research in computer science.

The first round of competition takes place in the Lone Star Preconvene exhibits area from 1:45 – 5:15 PM on Thursday and the semi-finalists give their conference presentations in Dallas D2 (undergraduate) and Dallas D3 (graduate) from 8:30 – 10:10 AM on Saturday. The ACM SIGCSE SRC winners will receive their awards during Saturday's luncheon.

NSF Project Showcases

Thursday, 10:00 AM – 11:30 AM and 3:00 PM – 4:30 PM
Friday, 10:00 AM – 11:30 AM and 3:00 PM – 4:30 PM
Saturday, 10:00 AM – 11:30 AM
Lone Star BC

The NSF Showcase participants are recipients of National Science Foundation Division of Undergraduate Education grants. All showcase sessions focus on educational issues: effective ways to present particular concepts in a classroom, using and evaluating new teaching techniques, anticipating future directions in the curriculum, and a host of other ideas directly applicable to college faculty. Showcase participants are faculty members from small liberal arts colleges to research-oriented universities. Your registration bag insert includes a detailed schedule of presenting faculty and abstracts of their projects. Also, individuals *willingness to work with the SIGCSE 2011 Committee to coordinate these events.*

MICROSOFT Thur. 10:45 – 12:00

F#: From Foundations to Modern Parallel and Data Rich Functional Programming Dallas A3

Presenters: Don Syme, *Microsoft Research, Cambridge*
R. Nigel Horspool, *University of Victoria*

F# is a multi-paradigm programming language, targeting the .NET/Mono platforms. F# brings you type safe, succinct, efficient and expressive functional programming language supporting both the imperative and object-oriented programming disciplines. It is a simple and pragmatic language, and has particular strengths in data-oriented programming, parallel I/O programming, parallel CPU programming, scripting and algorithmic development. F# combines the advantages of typed functional programming with a high-quality, well-supported modern runtime system and set of libraries. This tutorial will teach F# from several angles:

- F# as a cross platform language for teaching, including using F# on Windows, Macs, with Linux and MonoDevelop, and on the TryF# system
- The advanced features of F#, including asynchronous and parallel programming and units of measure
- F# type providers and their applications to strongly typed programming with web ontologies
- F# as a modern data programming environment, including writing data-rich phone applications

Attendees should come away with a good idea of the utility and power of F# in the classroom context, and its use all the way up to postgraduate and research level.

Have you heard the buzz about MeeGo? The newly formed MeeGo Operating System — a combination of Intel's Moblin and Nokia's Maemo—provides a unified development environment for developers with worldwide distribution channels to consumers.

Learn how Intel's Global MeeGo University program can help you integrate application development on netbooks, tablets, and smartphones into your curricula. Professors from the University of Jyväskylä and Georgia Institute of Technology will share insight into classroom modules that meet demand for rich application development on client devices. They will also share student feedback, in addition to how, why, and when you should start introducing MeeGo to your students.

AMAZON WEB SERVICES

Thur. 1:45 – 3:00
Dallas D3

Software as a Service, Cloud Computing, and Software Education

Presenter: Armando Fox, *UC Berkeley*

UC Berkeley leverages the combination of cloud computing and Software as a Service (SaaS), with its emphasis on productively creating well-tested, maintainable, reusable code, to let "one-pizza" teams of Berkeley undergrads design, develop, test, and deploy their own SaaS applications. Iteration-based agile development rewards regular progress, test-first design results in students actually enjoying testing, and cloud computing showcases deployed projects to friends, colleagues, and future employers, all while students absorb "big ideas" such as higher-order programming and metaprogramming. I'll also discuss other uses of cloud computing at Berkeley from lower-division through graduate CS courses.

Armando Fox is an Adjunct Professor at UC Berkeley, a co-author of "Above the Clouds: A Berkeley View of Cloud Computing", and a researcher at the intersection of cloud computing, machine learning, and parallel computing.

In the "real world" of software development, developers rarely work on a project alone. Development has become a "team sport" with accepted rules, best practices, and preferred tools. However, too often, students graduate only knowing how to develop code as an individual - not as part of a team. With IBM's Rational Team Concert (free for classroom use from IBM's Academic Initiative) students can work in a real-world team development scenario -- each with different roles on the team. Work items can be sized and assigned to each team member, and anyone can view real-time status and project health. Even better, professors can use the dashboards and reports to view each student's contribution to the project to help them assess each student individually.

MICROSOFT Fri. 10:45 – 12:00
Dallas A3

XNA Game Studio and PexForFun: Teaching Serious Computer Science and Developing Algorithmic Thinking

Presenters: Peli de Halleux and Nikolai Tillmann, *Microsoft Research, Redmond;* Pat Yongpradit

A Game Programming Framework for the PC, Xbox 360, and Windows Phone 7
Programming Exercises and Automatic Assessment in the Cloud

Delivering core computer science concepts and promoting deep algorithmic thinking with engaging activities — such as game development — can be a challenge. It can be an additional challenge to infuse game development into an existing CS program with a prescribed curriculum such as Advanced Placement Computer Science. We will preview a 5-week XNA Game Studio mini-course which is easily integrated into an existing curriculum schedule and is designed to engage students in game development, reinforce CS concepts, and develop complex algorithmic skills with the XNA/C# framework. PexForFun is integrated into the mini-course as a tool for practice, mastery, and analysis.

Students in high school through college can use PexForFun as a stand-alone tool to learn and practice programming concepts. With PexForFun, students edit code within any browser; the code is executed and analyzed in the cloud for immediate feedback. PexForFun supports the C#, Visual Basic, and F# programming languages. PexForFun helps students more deeply understand what is happening within the code by using dynamic symbolic execution to thoroughly explore feasible execution paths. The real fun starts with Coding Duels in which students write code that implements a specification unique, global, remote-access facility made available at no charge to members of the Intel Academic Community. The Intel Manycore Testing lab can be used to test, validate, and improve the scalability of classroom labs, homework, and capstone projects. The lab supports both Linux and Microsoft Windows with a 32-CPU/64-thread development environment, including up-to-date, essential performance tools to assist professors and their students.

Bob Chesebrough will demonstrate a command-line ray tracer application created by Master's degree student Dave Frogley. The application was tested, scaled, and tuned on the Intel Manycore Testing Lab. Discussions will include scaling methodologies and state-of-the-art software tools. Bob Chesebrough is a senior course architect in Intel's Innovative Software Education department.

GOOGLE Fri. 1:45 – 3:00

Google's Education Initiatives in 2011 Dallas D3

Multiple Presenters, *Google*

Google believes that all students should have the opportunity to become active creators of tomorrow's technology. Our goal is to leverage Google's strengths and infrastructure to increase access to high-quality, open educational content and technology in science, engineering, and math. We support access to computing curriculum and educational technology for all students, leveling the playing field so that students and educators alike have the opportunity to shape the technologies of their future.

During this session we will share what Google Education is planning for 2011 and we'll dive into how CS educators can use the following Google programs to inspire students to shape the technologies of tomorrow: App Inventor for Android, Google Summer of Code, Computer Science 4 High School, and Computational Thinking.

Google supports access to computing curriculum and educational technology for all students, leveling the playing field so that students and educators alike have the opportunity to shape the technologies of their future. The creators of tomorrow's innovations are everywhere, ready to be engaged and inspired.

This session will be your opportunity to talk in depth with Google representatives from key Google CS education programs. We'll also share material from other Google education programs and provide an opportunity to explore how, together, we can inspire students to shape the technologies of tomorrow. Representatives from the following programs will be present: App Inventor for Android, Google Summer of Code, Computer Science 4 High School, and Computational Thinking.

MICROSOFT Fri. 3:45 – 5:00

Microsoft .NET Gadgeteer: A New Way to Create Electronic Devices Dallas A3

Presenter: Nicolas Villar, *Microsoft Research*

Getting started building embedded devices is relatively complex which limits the students able to be successful and limits the complexity of the projects that they can undertake. With the Microsoft .NET Gadgeteer, someone with basic programming skills can be building sophisticated projects immediately. .NET Gadgeteer is a toolkit for building small electronic devices that combines the advantages of object-oriented programming, solderless assembly of electronics, and quick physical form factor design. Small devices can be iteratively designed, built and programmed in a matter of hours rather than days or weeks.

This session will provide an overview of the various hardware and software elements of the .NET Gadgeteer platform. Attendees will be introduced to the modular electronics system, and learn how to build sophisticated devices. They will learn about how .NET Gadgeteer devices can be programmed easily using the C# language, and interactively debugged Visual Studio IDE. Finally, the session will demonstrate how custom enclosures for .NET Gadgeteer projects can be built on demand by using 3D-printing technologies and predefined templates.

Parallelism and Concurrency in the Curriculum **Dallas A1**

Daniel Ernst, *University of Wisconsin - Eau Claire*; Jens Mache, *Lewis and Clark College* ; David Bunde, *Knox College*; Matthew Wolf, *Georgia Institute of Technology;* Libby Shoop, *Macalester College;* Richard Brown, *St. Olaf College;* Michael Wrinn, *Intel Corporation*

Multicore, multiprocessor, and clustered platforms have become the standard computing platforms available for executing programs. Thus, Computer Science education increasingly entails teaching programming in parallel environments. This BOF is designed to gather colleagues to exchange ideas and discuss questions such as the following: 1) What fundamental ideas of concurrency and parallelism should every CS graduate know? 2) How should concurrency and parallelism be integrated in the curriculum? At what levels and depths? If broad integration is appropriate, what resources and initiatives are needed to educate the educators? 3) What are some good/proven practices and platforms for teaching this material to undergraduates in these various contexts?

Web Programming **Dallas A2**
Marty Stepp and Jessica Miller, *University of Washington*

For the last three years we have held BoF sessions about teaching web programming at the college level. Our feedback indicates that the most valuable aspect of the past sessions was to get everyone together for an exchange of ideas and information. We'd like to bring together two groups of instructors: Newcomers who have not yet taught this material before (or are just starting), and veterans who may have insights to share with the group. Newcomers can ask questions about materials, languages and technologies to use, what works, and so on. Veterans can share tips from the classroom and helpful resources. We can also discuss the latest web technologies such as HTML 5, ECMAScript 5, Flash, iPhone, Android, Rails, Zend, Django, and .NET.

Using Game Development to Teach Parallelism: **Dallas A3**
Where Can I Find Resources to Get Started?
Robert Chesebrough, *Intel Corporation;* Tom Murphy, *Contra Costa College;* Joel Adams, *Calvin College*

What resources are available for students to learn parallel programming? Can parallel programming be taught effectively using game codes? Isn’t game programming difficult and parallel programming more so? What advantage is there in teaching parallelism through games development? This BOF will discuss opportunities for computer science students enhancing learning in the CS classroom. The discussion that we hope will ensue should provide a scaffold to this cohort of students who classify themselves as the net generation. We recognize that there is not one set of strategies that work for all educators, so our intention is to provide a forum for computer science instructors to share their own best practices.

Technology that Educators of Computing Hail (TECH) Site in the Ensemble Computing Portal — Dallas D2

Daniel D. Garcia, *UC Berkeley;* Sally Fincher, *University of Kent;* Don Bailes, *East Tennessee State University*

The judicious use of technology in computing education (in and out of the classroom) can be empowering and transformative. However, it is very difficult to discover what tools are available and how effective they have been. The ACM Education Council Technology and Tools task force has been developing a website, "Technology that Educators of Computing Hail (TECH)", that hopes to provide a central, organized collection of links to teaching technology resources. It will feature search, rating, tagging and commentary. We're delighted to announce that we have moved our site into the Ensemble Computing Portal. The goal of this BOF is to demonstrate the site, gather feedback, and collect "experience reports" from educators who have used technology for teaching computing – with success or not!

Program by Design: TeachScheme/ReachJava — Dallas D3

Viera Proulx, *Northeastern University*

Program by Design is a new name for the comprehensive introduction to programming at all levels that began with TeachScheme/ReachJava. This unconventional introductory computing curriculum covers both functional and the object-oriented program design in a systematic design-based style, enforcing test-first design from the beginning. The Bootstrap curriculum makes programming and algebra exciting for children ages 11-15. Special libraries support the design of interactive graphics-based games, musical explorations, client-server and mobile computing. We invite you to come and meet those who have used the curriculum, learn about new additions, libraries, bring in your experiences with the curriculum, show your projects, or ask questions about how it works and how you can use it.

project. Student contributions can range from the simple (e.g., writing user instructions), to the complex (e.g., coding an enhancement or new feature). Would you like to involve students in FOSS? This BoF, run by members of the Teaching Open Source `http://teachingopensource.org`) community, will host discussion on the process of turning students into FOSS contributors.

Undergraduate Information Security Curriculum Development **State Room 2**

Michael Locasto, *University of Calgary;* Richard Weiss, *Evergreen State College*

Incorporating information security into the undergraduate curriculum has been a topic of ongoing interest to SIGCSE attendees for quite some time. The purpose of this BOF is to help sustain the existing community of educators and researchers interested in bringing ethical hacking skills and an understanding of security into the classroom and relating these topics to the foundations of Computer Science. We plan to facilitate communication with BOF attendees to ask questions, suggest curriculum approaches, and share their own experiences. We also plan to briefly discuss our ongoing efforts and programs (e.g., infosec curriculum modules, the expansion of the SISMAT (Secure Information Systems Mentoring and Training) course, dissemination of infosec laboratory exercises).

Assessing Interdisciplinary CS Initiatives **State Room 3**

Leen-Kiat Soh, *University of Nebraska;* Jesse Heines, *University of Massachusetts Lowell*

With the revitalization of computational thinking and increasing recognition of the potential impact of interdisciplinary CS curricula in attracting students of both genders to CS and broadening participation in computing in other disciplines, we have seen a number of initiatives anchored with interdisciplinary CS curricula, accompanied with program evaluation and student assessment mechanisms. However, due to the often-convoluted multiple purposes of such initiatives, it is not entirely clear how best to assess such initiatives. This BOF aims at exploring this issue as a platform for a more comprehensive, future treatment of the subject.

Teaming up to Change K-12 CS Education, One State-at-a-Time **State Room 4**

Mark Guzdial, *Georgia Tech;* Susan Rodger, *Duke University;* Barbara Boucher Owens, *Southwestern University;* Chris Stephenson, *CSTA*

The CSTA has established a Leadership Cohort: Two K-12 teachers in each state who are supported and charged with improving CS education in their state. The SIGCSE Board their colleagues on the vanguard.

Mathematical (and Other) Reasoning in Computer Science Education **City View 2**

Doug Baldwin, *SUNY Geneseo*

This birds-of-a-feather session explores the relationships between mathematical and other kinds of reasoning typical of computer science, and the implications of those relationships for computing education. For example, is mathematical thinking a foundation for learning other forms of thinking? Or are other forms of thinking a necessary context for computing students to appreciate mathematics? All points of view are welcome.

AP CS A: Sharing Teaching Strategies and Curricular Ideas **City View 3**

Paul Tymann, *Rochester Institute of Technology;* Karen Donathan, *George Washington High School*

This BOF will provide an opportunity for high school and college faculty to discuss the AP CS A curriculum and to explore possibilities for collaborations and outreach activities between high schools and colleges.

Report of Findings: ACM Summit on Computing Education at Community Colleges **City View 4**

Elizabeth Hawthorne, *Union County College;* Robert Campbell, *CUNY Graduate Center;* Karl Klee, *Alfred State College;* Anita Wright, *Camden County College*

At the request of and funded by the National Science Foundation, the ACM Two-Year College Education Committee (TYCEC) conducted a Strategic Summit on the Computing Education Challenges at Community Colleges. A distinguished group of academicians and employers, accomplished in their various fields, were assembled not to identify solutions to challenges, but to ferret out and articulate the nature and scope of the challenges confronting computing programs in community colleges. At this session, the TYCEC will facilitate a dialogue about these challenges and associated opportunities. Attendees should come away from the session with ideas for crafting funding proposals and other initiatives. Many of the findings are applicable at and beyond the scope of the community college environment.

medicine, or healthcare. We will share our expertise and experience on such questions as choosing software and textbooks, developing case studies and projects, and developing links with biologists and clinicians. We will also discuss how best to continue the conversation, perhaps via electronic mailing list or blog.

Teaching and Learning with Scratch **City View 6**
Karen Brennan, John Maloney and Ricarose Roque, *MIT Media Lab*

Do you use Scratch as part of your teaching practice? Would you like to? Join members of the MIT Scratch Team for a discussion about how Scratch is being used in a variety of educational settings and learning environments. You will have an opportunity to share your experiences and resources and to find out about what others are doing. For those who are new to Scratch, we will provide a brief introduction to the Scratch authoring environment and to ScratchEd, an online community for Scratch educators, where more than 2300 educators have joined – sharing stories, contributing resources, participating in discussions, and connecting to other educators.

SIGCSE and the International Community **City View 7**
Alison Young, *Christchurch Polytechnic Institute of Technology*

ACM are very clear about their focus, they include the word "world" in their introductory statement. "ACM is an Educational and scientific society uniting the world's computing educators, researchers and professionals to inspire dialogue, share resources and address the field's challenges." SIGCSE has over 2600 members from 63 countries. There are many opportunities for international members however it has become apparent that not many of the international members are aware of these opportunities. This session is to help the international community to become more involved in SIGCSE activities, to promote the current opportunities and to feedback to the Board ideas for the international community to become more involved.

Capstone Projects in CS and SE: How do you reach out? **City View 8**
Carsten Kleiner, *University of Applied Sciences & Arts,* Dean Knudson, *North Dakota State University*

The goal of this birds-of-a-feather session is to bring together instructors in computer science and software engineering that teach capstone courses and who have interest in doing outreach within these courses. Most importantly it is meant as a forum to gather ideas and experiences on how to do outreach in such classes. There will also be the opportunity to discuss possibilities on how to set up an electronic platform for the exchange of future concerning successful gender issues projects, along with group discussion and brainstorming, in order to create committee goals for the coming year. We select projects to highlight through listserv communication and through our connections with NCWIT, ABI, ACM-W, etc. This year we will highlight the new NSF Broadening Participation in Computing grant – a grant that encompasses projects we presented in previous BOFs and a grant that builds on an alliance among ACM-W, ABI and NCWIT.

Media Computation — Dallas A2

Mark Guzdial, *School of Interactive Computing, Georgia Tech;* Barbara Ericson, *College of Computing, Georgia Tech*

Media Computation is an approach to teaching computing (introductory and data structures, in Python and Java) in which students learn through manipulation of digital media: images, sounds, text, video, and animations. Research on Media Computation has demonstrated effectiveness at increasing retention, especially among women and members of under-represented minorities. At Media Computation BOF's, faculty share their favorite Media Computation inventions and assignments, presented with samples of some of their students' productions.

Technology Available to Educators: How Does Access to Next-Generation Hardware Impact Students? — Dallas A3

Lauren Dankiewicz, *Intel Corporation;* Mike Pearce, *Intel Corporation;* Tom Murphy, *Contra Costa College;* Skylar Thompson, *University of Washington*

This BOF will explore opportunities for CS students to gain hands-on experience using state of the art hardware and software in the classroom. Options include cloud service providers, university hosted cloud solutions, and the Intel® Manycore Testing Lab—a remotely accessible, 32-core, 64-thread server that teachers and students can access at no charge. Audience members will share ideas and collaborate on actionable solutions that will directly impact their students. Discussion points will cover: What resources are available for students to learn parallel programming? Can parallelism be taught effectively without state of the art hardware? What are the minimum hardware requirements to teach parallelism? How can teachers provide students with a consistent environment and tool set?

studying their programs? Do some problem domains result in richer learning experiences for students than others? What programming structures and coding styles do we hope to see as students gain experience and understanding? This BOF is an opportunity to explore these questions together. As a way to start the discussion, the organizers will show an experimental visualization tool that may shed light on some of these questions. They will also share a few things they have learned from studying student Scratch programs.

Teaching Secure Programming **Dallas D2**

Blair Taylor, *Towson University;* Diana Burley, *George Washington University;* Bill Chu, *University of North Carolina, Charlotte;* Stephen Cooper, *Stanford University;* Ron Dodge, *United States Military Academy at West Point;* Trish Gregory, *Anne Arundel Community College;* Siddharth Kaza, *Towson University*

There is growing concern among educators that secure programming is inadequately addressed in the curriculum. In order to better prepare our graduates to address tomorrow's security challenges, we wish to examine the following questions: What do CS students need to know about secure software development? Are we teaching secure programming in CS1 and CS2? How can we encourage a security mindset among faculty? What steps are needed to include secure coding in the curriculum? How can we teach secure programming across different schools- community colleges, universities, high schools?

Lifesavers! Favorite Online Tools and Resources For Teaching CS1 **Dallas D3**

Ria Galanos, *Centennial High School;* Jill Pala, *Girls Preparatory School*

Do you ever wish there was a “one stop shop” to find all of the cool online resources for teaching introductory CS? Do you wish you could read reviews from teachers who use these resources? If so, then this is the BOF for you! Some of the best gems for teaching CS1 remain hidden to many teachers. This BOF will allow attendees to share their favorite online tools/resources and discuss how they use them to introduce or reinforce computer science topics. These lifesavers include: online practice tools, multiple-choice practice question generators, web-based IDEs, code visualization tools, automatic grading programs, and great teacher websites. Attendees will be given an annotated list of resources at the beginning of the BOF; the hope is for the list to grow with contributions from the group.

creation of online lab solutions that can be more easily tailored to curriculum needs than traditional commercial solutions by publishers.

Logisim and Circuit Simulation: Future Directions State Room 2

Carl Burch, *Hendrix College*

Logisim is an open-source, cross-platform tool that provides a graphical environment for designing and simulating circuits. It is used widely for classes such as introductory surveys of computing, sophomore-level computing systems courses, and upper-division computer organization and architecture. The software is in the midst of heavy development; revision should be complete by summer 2011, with intermediate versions released along the way. In this BOF led by Logisim's primary developer, attendees will have the opportunity to discuss how they use circuit simulation in their courses (if at all), to provide feedback on recent developments, and to discuss openly directions for Logisim's near-term future.

Computing Education that Supports Research: Connecting with the Committee of Education of the Computing Research Association (CRA-E) State Room 3

Edward Fox, *Virginia Tech;* Jane Prey, *Microsoft;* Richard DeMillo, *Georgia Tech*

Few US students move into computing research. CRA-E seeks to address this challenge by supporting community-wide activities. NSF CISE and DUE, as well as other agencies/foundations, have new and continuing programs that fund undergraduate students interested in research, e.g., individual or site REU awards. Nevertheless, there are few umbrella mechanisms to collect and disseminate effective practices regarding how best to attract students to engage in research, nor on how to expand and improve the pipeline guiding such students to research activities/studies/careers. We seek to engage SIGCSE attendees involved or interested in better understanding and supporting future computing research. This must involve those in R1 universities, 4-year colleges, 2-year colleges, and K-12 education.

Outreach and Reaching Out in K-12: An International Perspective State Room 4

Margot Phillipps and Chris Stephenson, *CSTA;* Judith Gal-Ezer, *Open University*

CSTA's international mandate includes a commitment to sharing knowledge and expertise with educators around the world who want to build CSTA-like organizations. As CSTA has grown to represent members more than 100 countries, many of these countries, including Armenia, England, France, India, New Zealand and Nigeria have approached CSTA for phase of the software lifecycle. Courses that teach software engineering principles tend to be near the end of the undergraduate curriculum. Introducing software engineering principles early would allow the maturing of students' knowledge through the application of the principles in their advanced courses. Time and resource limitations can prevent additional courses to teach software engineering principles. What software engineering principles can be introduced in the first two years of the computer science curriculum?

Exploring Computer Science: Fostering Computational Thinking through Engaging and Relevant Curriculum — City View 2

Gail Chapman, *University of California, Los Angeles*

Computer science teachers are always looking for new projects to use with their students that are engaging and relevant while providing an opportunity for computational thinking at a high level. The Exploring Computer Science curriculum is based on projects of this type. This BOF will provide a platform for the discussion of what teachers are currently doing in ECS classrooms and provide teachers who are not teaching ECS an opportunity to learn more about the kinds of projects that have been successful.

Creating a Game Development Course for Pre-Collegiate Schools — City View 3

Alfred Thompson and Patricia Philips, *Microsoft*

Game development computer science courses are gaining in interest as a means to increase excitement and enrollment in computer science. What are the tools, topics and curriculum modules that actually work in high schools and middle schools? This BOF will give teachers who have created game development courses and teachers who are considering creating them a chance to share ideas, solutions and concerns.

A Neglected Pipeline? How Faculty Teach, Advise, and Mentor Transfer Students — City View 4

Mihaela Sabin, *University of New Hampshire;* Susan Miertschin, *University of Houston*

Two-year community colleges are critical entry points to postsecondary education for low-socioeconomic status students. Record numbers of students enroll in Information Technology programs at community colleges. Many of these students transfer into bachelor's IT programs, which reside in CS departments and/or have overlapping curricula with CS programs whose faculty teach and advise both CS and IT students. Articulation agreements between two- and four-year computing programs facilitate student transfer. However, these agreements do not spell out support structures that help transfer students provide an opportunity to create an ongoing list of faculty with similar interests.

Education, Computers and Society **City View 6**
Joseph Oldham, *Centre College;* Florence Appel, *Saint Xavier University*

This session is organized by ACM SIGCAS (Computers and Society) for those interested in addressing the social and ethical consequences of computing within their curricula. We will emphasize sharing resources and ideas, and strengthening our network. Discussion typically includes best practices and approaches to teaching computer ethics and social impact, developing programs for recruitment and retention of under-represented populations, and implementing service learning course components. We offer an update on happenings in SIGCAS and encourage discussion of ways in which SIGCAS and SIGCSE can collaborate, including organizing a future pre-SIGCSE conference dedicated to computers and society issues. Attendees are welcome to contact the discussion leaders with suggestions or questions.

Enhancing Undergraduate Education through the ACM Distinguished Speakers Program **City View 7**
Barrett Bryant, *University of Alabama at Birmingham*

As indicated on the ACM Distinguished Speaker (DSP) website (`http://www.dsp.acm.org`), a core mission of the DSP is to provide students with access to computing professionals by facilitating personal visits by DSP speakers to ACM Student Chapter meetings. DSP speakers come from all areas of computer science and from throughout the world and include Turing Award winners. The opportunity to have such a distinguished speaker visit is a highlight for ACM Student Chapters. Yet the program is currently underutilized. This session will provide an overview of the program along with discussion of how to use it to enhance undergraduate education, for both universities which have and do not have active ACM Student Chapters.

Computer Science: Small Department Initiative **City View 8**
James Jerkofsky, *Walsh University*

Faculty in small departments (perhaps 3 FTE, perhaps only 1 or 2) face special situations – both challenges and strengths. In this BOF, members will have a chance to talk about both. Challenges include maintaining a well-rounded curriculum and attracting students. Strengths include a close relationship with other members of the department and majors. These and other topics are open for discussion; the specific topics will be based upon the composition and interests of the group assembled.

Session I: Friday, 10:00AM – 12:00PM, Lone Star Preconvene

Incorporating Application Domains in Software Engineering Education
Sriram Mohan, *Rose-Hulman Institute of Technology*

dLife: A Java Library for Robotics, AI and Computer Vision
Grant Braught, *Dickinson College*

Recruiting via a First-Year Seminar: Storytelling through Computer Animation
Mark LeBlanc, *Wheaton College*

A Java Ray Tracer for Introductory Computer Graphics Courses
Helen Hu, *Westminster College*

Building and Managing Paths of Web-based Computer Science Content
Richard Furuta and Frank Shipman, *Texas A&M University*

Pacific Rim Summer School in Global Distributed Software Development
Virginia (Ginnie) Lo, Andrzej Proskurowski, Arthur Farley and Stuart Faulk, *University of Oregon*; Lian Yu, *Peking University*; Kathleen Freeman-Hennessy, *University of Oregon*

Building a Generator-Based Cyber Platform for Automating Production of PPT-Based Algorithm Visualization Teaching Materials
Sen Zhang and James Ryder, *SUNY College at Oneonta*

Scrape: A Tool for Visualizing the Code of Scratch Programs
Ursula Wolz, Christopher Hallberg and Brett Taylor, *The College of New Jersey*

Lego NXT Mindstorms Robot Simulator
Richard James, R. Taylor Hanson and Addison Sims, *Rollins College*

Computer Science for the Liberal Arts: Reaching More Students
Madeleine Schep and Nieves McNulty, *Columbia College*

SPLICE: Self-Paced Learning in an Inverted Classroom Environment
Matthew Boutell and Curtis Clifton, *Rose-Hulman Institute of Technology*

Empirical Evaluation of Success Factors for Capstones in Software Engineering and their Interdependencies
Carsten Kleiner and Arne Koschel, *University of Applied Sciences & Arts Hannover*

Mark Lewis, *Trinity University*

Inspire-CT: Vertically Integrated Student Teams
Gregory Hislop, *Drexel University*; Massood Towhidnejad, *Embry-Riddle Aeronautical University*; Joseph Urban, *Texas Tech University*

Revitalizing Labs: Lessons from 2.5 Years of Iterative Development and Assessment of Digital Logic Labs
Elizabeth Patitsas, Steven Wolfman and Meghan Allen, *University of British Columbia*

Increasing Security Awareness: The Next Level
Helen Schneider, Mary Jo Geise and Loren Wagner, *The University of Findlay*

An Inter-Disciplinary Workshop based on Social Robotics
Nick Webb, Jennifer Goodall, Michael Ferguson, Kathryn DeCorah and Kristen Kielbasa, *University at Albany, SUNY*

Project Expression: A Cloud Computing & Multimedia-based Java Course that Helps Solve the Empathy Crisis
Joseph Shanahan and Daniella Marghitu, *Auburn University*

Curricular-Wide Problem Based Learning for Computer Science and Programming Education
Amanda Holland-Minkley, Samuel Fee and James Gralka, *Washington & Jefferson College*

The Virtual Studio: Implementing Studio-Based Learning Techniques in an Online Introductory Programming Course to Address Common Programming Errors
Blanca Polo, *Leeward Community College*

Integrating "Soft Skills" into a Computing Degree
Alison Young, *Christchurch Polytechnic Institute of Technology*

SKA for RBTs: An Algorithm Visualization System to Support Red-Black Trees
Kadian Davis and Ashley Hamilton-Taylor, *The University of the West Indies*

Reaching Out to Many Majors: A Database Approach
Suzanne Dietrich, *Arizona State University*; Don Goelman, *Villanova University*

GUIGraph: Editing Live Object Diagrams for GUI Generation Enables New Pedagogy in CS 1/2
Duane Buck, *Otterbein University*

Introduction to Programming with Alice: Bringing a College Course to High Schools in Memphis
Juan Carlos Olabe, *Christian Brothers University*; Laine Agee, *White Station High School*

Deconstruction Kits in Scratch: Designing Scratch Debugems for Learning Core Programming Concepts
Jean Griffin, Eliot Kaplan, Quinn Burke and Yasmin Kafai, *University of Pennsylvania*

The Laptop Instrument
Kristin Raudonis, *Villanova University*

Getting CS Undergraduates to Communicate Effectively
Andreas Karatsolis, Iliano Cervesato, Yonina Cooper, Khaled Harras, Kemal Oflazer, Nael Abu-Ghazaleh and Thierry Sans, *Carnegie Mellon University Qatar*

What Today's College Students Know about Technology and What They Don't
Mary Jo Geise and Helen Schneider, *The University of Findlay*

Student-Authored Wiki Textbooks in Computer Science
Edward Gehringer, *North Carolina State University*

Impact of Direct Tutoring that Utilizes Visual Learning and Translations for Deaf Students
Brian Trager and Raja Kushalnagar, *National Technical Institute for the Deaf*

Programming By Voice with Scratch

Major Challenges Facing Computing Programs at Community Colleges
Elizabeth Hawthorne, *Union County College*; Robert Campbell, *CUNY Graduate Center*; Karl Klee, *Alfred State College*; Anita Wright, *Camden County College*

Students Believe Live Coding Helps Them Learn
Kristin Vadas Marsicano and Jay Summet, *Georgia Institute of Technology*

A Deconstruction Kit for the LilyPad Arduino: Designing Debugging Sets for Learning about Circuitry & Programming for High School Students
Eliot Kaplan, Jean Griffin, Yasmin Kafai and Quinn Burke, *University of Pennsylvania*

Introducing Case Study Based Teaching in Computing Curricula
Salamah Salamah, Massood Towhidnejad and Thomas Hilburn, *Embry Riddle Aeronautical University*

Engaging Participants in the Ensemble Computing Pathway
Luke Segars and Daniel D. Garcia, *University of California, Berkeley*

Graph Algorithms using Highway Mapping Data and the Google Maps API
James Teresco, *Siena College*

Universal Design for Learning in CS1
Suzanne Rivoire, *Sonoma State University*

A Grounded Embodied Approach to the Instruction of Computational Thinking
Cameron Fadjo and John Black, *Teachers College, Columbia University*

Wednesday Workshops, 7:00 PM to 10:00 PM

1. **Advanced Scratch: Computer Science Through Storytelling and Games** **Lone Star A2**

   Ursula Wolz, *The College of New Jersey;* John Maloney, *Massachusetts Institute of Technology;* Christopher Dunne, *Self-employed*

   A set of Scratch storytelling and game project starters are presented to those with some Scratch experience. Each project demonstrates good media design, intermediate/advanced programming and core computing concepts. These projects are used in a CS 0 course, a one semester CS1/CS 2 course, and a course in game design. Computing concepts include modular design via game engine architecture, data management and communication via sprite interaction, and algorithm efficiency via sound production and sprite animation. Laptops are required. Participants will have ample time to extend the project starters and discuss the ramifications for computing curriculum. See `http://www.tcnj.edu/~wolz/AdvScratch`

2. **Developing An Effective Assessment Program For Student Educational Outcomes** **Lone Star A3**

   Donald Sanderson, *East Tennessee State University*

   New requirements, especially among ABET accredited programs; require measuring the performance of graduates against a set of program level student outcomes. This workshop is aimed at those new to this approach, and focuses on the techniques needed to move from high level student outcome statements to specific methods to collect, aggregate and evaluate the needed data. Participants can choose to use example materials, or bring their own outcomes and related course syllabi. They will leave with the tools needed to create evaluable performance indicators for student outcomes, develop appropriate direct and indirect measures of performance, integrate measurement into existing classroom activities, and aggregate the collected data into useful information. **Laptop recommended**.

3. **Pedagogical Progressions for Teaching Object-Oriented Design** **City View 7**

   Carl Alphonce, *University at Buffalo, SUNY;* Michael Caspersen, *Aarhus University;* Dale Skrien, *Colby College*

   This workshop is intended for anyone teaching object-oriented programming who has an interest in developing the pedagogy used to teach students sound object-oriented developed a wide range of curricula using a variety of programming languages (including Scratch, Java, C, and Scheme), used by students ranging from junior high to college. This workshop will first introduce our experiences and approach. Next, we'll help instructors integrate our customizable curricular modules and relevant technology, such as our customized Moodle, into their classes. We'll end with a look at our online community site, designed to support instructors as they adopt the lab- centric approach in their classroom. **Laptop required**.

5. **Open Source and Freeware Tools for 3D Game Development Courses** **State Room 2**

   Victor M. Larios, *University of Guadalajara CUCEA;* Kelvin Sung, *University of Washington Bothell*

   3D game development is a complex and labor intensive process that involves the creation, management, and integration of multimedia resources from diverse domains involving drastically different tools. A 3D game development class should touch upon all of these aspects of game development process. This workshop is design specifically for interested faculty members new to the field, where it can be overwhelming to identify the required software tools and to work with the potential overburdening budget. Based on experience teaching 3D game development courses, this workshop will: 1) Introduce a streamlined 3D game production pipeline, 2) Demonstrate simple, free, and compatible tools for each stage of the pipeline and 3) Describe our strategies for managing the resources.

6. **Web Development with Python and Django** **State Room 3**

   Ariel Ortiz, *Tecnológico de Monterrey, Campus Estado de México*

   Many instructors have already discovered the joy of teaching programming using Python. Now it's time to take Python to the next level. This workshop will introduce Django, an open source Python web framework that saves you time and makes web development fun. It's aimed at CS instructors who want to teach how to build elegant web applications with minimal fuss. Django is Python's equivalent to the popular Ruby on Rails framework. Topics that will be covered include: setup and configuration, template language, and database integration through object-relational mapping. Participants should have some familiarity with Python, HTML and SQL. **Laptop required.** Please check the following link to get more information: `http://webcem01.cem.itesm.mx:8005/django.`

are needed to introduce this new approach. We will guide you past the real-life pitfalls that get in the way of using AVs in the classroom. We show you how to find the resources that you need, and present case studies of successful classroom deployments. This workshop is about helping you to make the changes you have said for years that you want to make. You will get time to try out some recommended AVs during the workshop. **Laptop recommended**.

8. Making Mathematical Reasoning Fun: Tool-Assisted, Collaborative Techniques **City View 2**

Jason Hallstrom, *Clemson University* ; Joe Hollingsworth, *Indiana University Southeast;* Joan Krone, *Denison University;* Murali Sitaraman, *Clemson University*

Is it possible to excite students about learning the mathematical principles that underlie high-quality software? Can we teach them to apply these principles using modern software tools? Can this be accomplished without displacing existing content? Yes! But it takes the right set of pedagogical principles, teaching tools, and classroom exercises. This hands-on laboratory will introduce a set of principles, tools, and exercises that work. Participants will be better prepared to teach students to reason rigorously about the software they develop and maintain. Funding is available to cover registration for some attendees; please contact the workshop organizers for details. **Laptop required.**

9. General Purpose Computing Using GPUs: Developing a Hands-On Undergraduate Course on CUDA Programming **City View 3**

Barry Wilkinson, *University of North Carolina Charlotte;* Yaohang Li, *Old Dominion University*

GPUs (graphical processing units) with large numbers of cores are radically altering how high performance computing is conducted. With the introduction of CUDA for general-purpose GPU programming, we can now program GPUs for computational tasks and achieve orders of magnitude improvement in performance over using the CPU alone. With CUDA-enabled GPUs, any desktop or laptop computer can become a very high performance computer. Some HPC clusters now incorporate large numbers of GPUs, which now impact on how clusters are programmed. The purpose of this workshop is to provide CS educators with the fundamental knowledge and hands-on skills to teach CUDA materials at the undergraduate level. **A laptop is required**, which does not need to be CUDA-enabled as remote GPU servers will be provided.

group exercises, and provide "expert" opinions on these issues. The intended audience is faculty at two-year and four-year colleges and universities who are seeking NSF funding in support of undergraduate education. Participants will include novice proposal writers as well as those who seek to improve their proposal writing. **Laptop optional.**

11. Audacious Android Application Programming — Lone Star A4

Frank McCown, *Harding University*

As smartphones and mobile devices become ubiquitous, many CS departments are adding mobile computing electives to their curriculum. Google's Android OS is a freely available and popular smartphone platform with applications programmed in Java. Workshop participants will be introduced to mobile app development and the Android SDK. We will write some simple Android apps with Eclipse and run them on an emulator. For those interested in teaching an upper-level Android course, reusable programming labs and projects will be distributed, and we will discuss some teaching strategies. Participants should be capable of writing Java programs in Eclipse and should bring their own laptop preloaded with Eclipse and the Android SDK. `http://www.harding.edu/fmccown/android/workshop.html`
Laptop recommended.

12. Using Map-Reduce to Teach Parallel Programming Concepts Across the CS Curriculum — City View 8

Richard Brown, *St. Olaf College;* Elizabeth Shoop, *Macalester College*; Patrick Garrity and Timothy Yates, *St. Olaf College*

Map-reduce, the cornerstone of Google- and Yahoo!-style computing, has star appeal to draw students to the study of parallelism. Workshop participants will carry out exercises that introduce data-parallel processing concepts, using the open-source Hadoop map-reduce programming environment. The exercises are designed for the CS1, intermediate, and advanced curricular levels. CS1 exercises use a simplified interface developed by the presenters for Hadoop programming by introductory students, using Java, Python, Scheme, or C++. Intermediate and advanced exercises use standard Hadoop programming in Java or C++. Expository materials and portable software infrastructure provided to participants. Intended audience: CS instructors. **Laptop required** (Windows, Mac, Linux).

The web is the dominant computing platform of our time. Google has led the rapid development of technologies to create mobile and cloud computing applications and has freely provided them to the world. This workshop provides hands on exposure using Eclipse plugins from Google to develop a mobile application with both stand alone and cloud based services for data persistence. In the first of three sessions, the Android plugin will be used to develop a fully functional Android application. In the second session, the App Engine plugin will be used to develop cloud based services using Java Servlets and JPA. In the third session, participants will extend the stand alone mobile application to support cloud based services. For more information: `http://www.cs.elon.edu/sigcse2011`. **Laptop required**.

14. Computer Science Unplugged And Outreach Activities — Dallas D3

Tim Bell, *University of Canterbury;* Bengt Aspvall, *Blekinge Institute of Technology*; Daniela Marghitu, *Auburn University;* Lynn Lambert, *Christopher Newport University*

You've been asked to talk to an elementary or high school class about Computer Science, but how can you ensure that the talk is engaging? Or perhaps you're trying to introduce a concept from Computer Science to a school group, but you want a fun way to get the class engaged. This workshop introduces CS Unplugged (`www.csunplugged.org`), a widely used set of kinesthetic, fun activities that cover many core areas of computer science without using high technology. We will explore how to use the activities in a variety of situations (including with special needs students), show some novel applications, and discuss ways that they can be combined with "plugged-in" activities. Attendees will receive a copy of a handbook for teachers and a collection of video files demonstrating the activities.

15. Basic 2D Graphics and Animation Concepts for iPhone/iPad Games — Lone Star A4

Jonathan Lartigue, Russell Thackston and Prateek Hejmady, *Auburn University*

iOS and Android devices and their respective app stores have revived the individual- and small- publisher game industry. App downloads in Apple's iTunes store - mostly games - now exceed song downloads. Developers are learning or refreshing game development skills to capitalize on this trend. A first step is understanding simple, 2D game design. We start at the beginning - presenting basic 2D graphics, animation, and game development on the iPhone platform. We introduce concepts such as basic sprites; versatile sprite class design; drawing, transforming, and animating sprites; and simple game engines. Hands-on activities apply these concepts to create a simple but institutions, it's a rewarding way for faculty to remain engaged in their own research. We will present proven mentoring strategies for undergraduate research, equip participants with materials to run mentoring workshops for undergraduates, and brainstorm with attendees. This workshop is intended for all college-level CS educators. See `www.cs.williams.edu/~andrea/SIGCSE11`. **Laptop Optional**.

17. 2D Game Design and Development 101 **Dallas A2**

Scott Leutenegger and Rafael Fajardo, *University of Denver*

You have attended or heard about the game talks at SIGCSE over the past few years. You know inside that there is something there, your students want to make games, but, you have no clue how to go about designing and creating a game. You can write/teach the code to move objects around and detect collisions between objects, but how do you design the game? We will teach participants how to create a simple 2D game using their ideas and then transition to a rapid development exercise using Scratch. This workshop is intended for any CS instructor wishing to use game creation as an assignment but who is uncomfortable and/or unknowledgeable about how to design and create a simple game.

18. Listening to Linked Lists: Using Multimedia to Learn Data Structures **Dallas A3**

Mark Guzdial and Barbara Ericson, *Georgia Institute of Technology*

Everybody teaches linked lists, with homework like implementing duplicate, weave, and reverse. When those nodes contain strings or numbers, these are pretty boring assignments. When these nodes contain music (MIDI), these operations are composing music, which can then be played. This workshop shows how to use music, images, and sounds to teach the basic data structures, including linked lists, circular linked lists, stacks, queues, and trees. These pieces are then tied together through the use of simulations to generate animated movies. We will be using Java, though many of the methods can also be used in Python. Laptop recommended, if you want to play along.

19. Multithreading (Pretty) Early for Everyone: Parallelism and Concurrency in Second-Year Data-Structures **Dallas D1**

Dan Grossman, *University of Washington*

There is wide interest in introducing students to threads earlier, but how? An answer: a 3-week unit in data structures. It should appeal to data- structure instructors and those tasked with curriculum revision. A key approach is to distinguish parallelism (using

XHTML, without learning JavaScript or AJAX. See a full set of CS1 assignments leveraging this strategy. See how students can write a personal "Facebook-lite" they can show to friends. Play with live demos yourself. Leave with new assignment ideas. **No laptop required**.

21. Computational Art and Creative Coding: Teaching CS1 with Processing **Lone Star A3**

Ira Greenberg, *Southern Methodist University;* Deepak Kumar, *Bryn Mawr College;* Dianna Xu, *Bryn Mawr College;* Ursula Wolz, *The College of New Jersey*

This workshop showcases a new approach to teaching CS1 using computational and generative art as a context. Participants will be introduced to the Processing programming language and environment, designed for the construction of 2D and 3D visual forms. Its IDE is light-weight, but well-suited for the kind of rapid proto-typing needed for dynamic visual work. We hope to bring the excitement, creativity, and innovation fostered by Processing into the computer science education community. Instructors of all experience levels - including none - are welcome. At least one-half of the workshop will be hands-on: participants will be able to learn and explore Processing and create a variety of visual effects on the fly. **Laptop required**.

22. Build Your Own Blocks: A Scratch Extension for CS Courses for Non-Majors **State Room 1**

Brian Harvey, Daniel D. Garcia, Colleen Lewis and Luke Segars, *University of California, Berkeley;* Josh Paley, *Henry M. Gunn High School, Palo Alto*

This workshop is for high school and college teachers of general- interest ("CS 0") CS courses. It presents the programming environment used in one of the five "AP CS Principles" pilot courses. BYOB (Build Your Own Blocks) is a graphical, drag-and-drop programming language based on Scratch. Scratch is unthreatening, object-oriented, and multithreaded. But a Scratch program is written as "scripts" without names, arguments, or return values. BYOB supports older learners by adding named procedures (thus recursion), procedures as data (thus higher order functions), and structured lists. Participants will learn BYOB through discussion, programming exercises, and exploration. See `http://byob.berkeley.edu` for details and software. **Laptop required**.

in a relevant and engaging way, including iteration, decision- making, procedural abstraction, and inheritance. Tool differences and limitations will be covered as well as ways to use each tool to teach a particular topic more effectively. Examples will be relevant and engaging for a diverse audience of learners. The workshop will be hands-on and no prior experience will be required. **Laptop required. Java 6.0 pre-installation is required and pre-loading Jeroo, Scratch, and GridWorld is strongly recommended.**

24. Greenfoot: Introducing Java with Games and Simulations — State Room 3

Michael Kölling, *University of Kent*

Greenfoot is a programming environment, from the creators of BlueJ, that allows teaching of object- oriented programming concepts – using Java – in a highly motivating context. Built to be interactive and graphical, Greenfoot offers a more engaging experience than previous systems. Building widely differing scenarios, such a simulations or games, is easy and quick. This workshop is aimed at teachers of introductory Java programming courses (high schools and universities) who have no or little experience with Greenfoot. Laptop recommended for hands-on exercises. Participants without laptops will be paired with laptop owners. The workshop is practically oriented and allows participants to use Greenfoot in their classroom immediately. More information at `www.greenfoot.org`. **Laptop Recommended**.

25. Using the FIRST (robotics) Experience to Reach Pre-College Students via Near-Peer Mentors — *Canceled*

Stephanie Ludi, *Rochester Institute of Technology*

This workshop introduces participants to using FIRST (For Inspiration and Recognition of Science and Technology) robotics challenges and technologies as a means of conducting outreach with middle and high school students. The use of FIRST goes beyond programming providing linkages to various domains in a manner that promotes teamwork and research. Participants will learn about how to conduct a FIRST team, using university students as near peer mentors. The workshop is divided into outreach strategies, managing a FIRST team, and practicing robot design and programming using a sample FIRST challenge. You can learn more about FIRST at: `http://www.usfirst.org`

This workshop introduces playing and generating music with Scratch, a media-rich visual programming system (`scratch.mit.edu`). It is based on lessons learned using Scratch to teach both music and computer science in an interdisciplinary GenEd course. As students write programs that make music, they learn control flow, user interaction, synchronization, real- time programming, and data structures. Participants will use their own laptops to explore progressively complex musical Scratch programs (see `www.scratchmusic.org`). They will also write their own programs and use external sensor devices to make custom musical instruments. Samples and an extensive handout will be provided. The workshop will culminate in a concert of participant-created music. **Headphones and laptop required**.

27. Hands-On Teaching Modules for Secure Web Application Development **Lone Star A3**

Li-Chiou Chen and Lixin Tao, *Pace University*

This workshop will discuss security issues in web application development and demonstrate a set of teaching modules in this area through hands-on exercises, developed by a NSF-funded project called SWEET (Secure WEb dEvelopment Teaching). The workshop will guide the participants to run through a couple of web security hands-on exercises, such as web server threat assessment, security testing, and secure web transactions. The workshop will also discuss examples of incorporating these teaching modules in computing curriculum. To facilitate the exercises, the workshop will provide every participant a lab DVD, which contains the teaching materials and software needed. Further information is available at `http://csis.pace.edu/~lchen/sweet/`. **Laptop Required.**

28. Storytelling in CS Education: Examples from Programming and Software Engineering Courses ***Canceled***

Henrik Baerbak Christensen and Michael E. Caspersen, *Aarhus University*

Storytelling is a viable technique to structure courses and exercises to improve student motivation and learning: a realistic story is constructed such that new learning topics are introduced and revisited as demanded by the story, typically of a software product that evolves in response to customers' new requirements. We will exemplify storytelling as used in an advanced programming course, discuss pitfalls, and shortly relate it to cognitive learning theories. Next, the organizers will coach participants in developing stories for their own courses. Finally, a plenary discussion will sum up ideas, tips and tricks. Visit `www.daimi.au.dk/~hbc/story`. **Laptop Optional**

The workshop follows the introduction of Scrum into a senior-level SE course with the use of Wiki documentation, an online product Backlog, Git version control, Ant, JUnit, and Maven for building, testing, and integrating projects. The use of individual tools in other courses is discussed with an eye toward easing the learning curve in Sofware Engineering.

30. Testing Graphical User Interfaces in the Classroom State Room 2

Stephen Edwards, Manuel Perez-Quinones and Jason Snyder, *Virginia Tech*

Software testing has become popular in introductory courses. But many introductory courses use graphical user interfaces (GUIs) as an "attention grabber" for students and as a metaphor for teaching object-oriented programming. Unfortunately, developing software tests for programs that have significant GUIs is beyond the abilities of typical students (and, for that matter, many educators). This workshop introduces participants to LIFT, the Library for InterFace Testing. LIFT works with the ACM Java Task Force library, Java GUIs using Swing, and Objectdraw. LIFT was developed at Virginia Tech by the Web-CAT team. Participants will learn how to use LIFT, including how to write simple, student-friendly tests for GUIs used in typical CS1/CS2 courses as well as more complex user interfaces.

31. Teaching a Consulting Course to Develop Communication and Leadership Skills ***Canceled***

Scott McElfresh, *Wake Forest University;* Joseph Mertz, *Carnegie Mellon University*

The daunting logistics of managing service projects and the difficulty of defining their academic benefits tempers interest in service learning. Our approach is to teach students consulting skills as they work each week as consultants for non-profit organizations (`http://cmu.edu/tcinc`). Students learn to lead a consulting project, communicate effectively in oral and written forms, and engage in a systematic inquiry into the social and organizational context of technology. Workshop participants will receive sample materials for implementing such a course. All participants will share ideas for addressing the problems associated with developing and managing relationships with community partners and ideas for promoting new proposals for community engagement to their peers and administrators.

a view of sharing resources and ideas. We will begin a conversation with university and high school faculty to build a future strategy to determine if current workshops do enough to engender student desire to act by committing to a higher degree in ICT. We will present information on existing programs in different countries, such as the Digital Divas program(Australia), the Digigirlz program(various countries), project IMPACT (US) and an ACM-W Student Chapter in Turkey. Current trends indicate that we need to act now to encourage more women into ICT to meet future employment and creativity demands. **Laptop Optional**.

33. Teaching with Greenfoot: From Development of Material to Delivery in the Classroom — State Room 3

Michael Kölling, *University of Kent;* Frances P. Trees, *Drew University;* Stephanie Hoeppner, *Clermont Northeastern Schools;* Daniel Green, *Oracle Corporation*

Greenfoot is an introductory Java programming environment that gives teachers a high level of control over the nature and context of teaching examples. This workshop is aimed at teachers of introductory programming courses (high school/university) who have seen Greenfoot and want to learn how to use it more effectively in their teaching. Participants will develop their own teaching project for use in their classroom under consideration of pedagogical and technical aspects. Other topics discussed include educational strategies for programming, advanced Greenfoot programming techniques, transition from/to other system (Scratch, Alice, BlueJ) and integration of media. Laptop recommended for hands-on exercises. Participants without laptops will be paired with laptop owners. **Laptop Recommended**.

are learning computing concepts and then uses Media Computation to reinforce the same concepts in Java. We will introduce the approach and show some sample work. Participants will do hands-on Alice and Media Computation projects. For example they will use chromakey to change the background of an Alice movie. We will leave 15 minutes at the end of the workshop to answer questions. **Laptop Required**.

35. Short Mobile Game Development Projects for CS1/2 Lone Star A4

Stan Kurkovsky, *Central Connecticut State University;* Delvin Defoe, *Rose-Hulman Institute of Technology*

Game development and mobile computing have been successfully used to increase student motivation. However, instructors with no background in mobile computing, computer graphics, and/or game development may find it difficult to develop or adopt course materials on these topics. This workshop is designed to address these concerns. Using Java Micro Edition, we have developed several project-based course modules focused on mobile game development and designed to study fundamental programming principles (e.g. loops) while also exposing students to more advanced concepts (e.g. databases). Using a mobile phone emulator, participants will test- drive one of our modules and develop a simple game, which can then be transferred to and played on a mobile device. **A Windows or Mac laptop is recommended**.

(SIGCAS)

- CSTA Source Web Repository Cataloging Workshop
- Department Chairs' Pre-Symposium Workshop *
- Managing the Academic Career for Women Faculty in Undergraduate Computing Programs
- Program by Design: Bootstrap, TeachScheme, ReachJava, and Beyond
- Programming with Alice Workshop (Alice 2.2)
- Programming with Alice Workshop (Alice 3)
- The 3rd Annual Humanitarian FOSS Symposium 2011
- Innovative Approaches to Introducing Computer Science
- Hawaii Project Companion for Cloud-Enabled Mobile Computing Courses
- New Educators Roundtable *
- Teaching Professional Ethics in Computer Science: Tips and Traps

* Sponsored by the SIGCSE Board

More information about these events can be found online at:

`http://sigcse.org/sigcse2011/attendees/pre_symposium_events.php`

Technische Universität Darmstadt
Darmstadt, Germany
June 27-29, 2011
`http://www.sigcse.org/iticse2011`

Conference Chair
Guido Rößling - *TU Darmstadt*

ICER 2011
Providence, Rhode Island, USA
August 8-9, 2011
`http://icer-conference.org/`

General Chair
Kate Sanders*, Rhode Island College*

SIGCSE 2012
Raleigh, North Carolina, USA
February 29 - March 3, 2012
`http://sigcse.org/sigcse2012/`

Symposium Co-Chairs
Laurie Smith King, *College of the Holy Cross*
David R. Musicant, *Carleton College*