ITiCSE 2011 Final Program and Abstracts Book
June 27-29, 2011

Welcome to Summer in Darmstadt!

Table of Contents

A Word of Appreciation

We thank our supporters, without whom this conference would not have been possible. Please show your appreciation by visiting them in the supporter room!

1. Welcome to ITiCSE 2011!

We are pleased to present the Final Program of the 16th Annual Conference on Innovation and Technology in Computer Science Education —ITiCSE 2011. Between the first idea about hosting an ITiCSE conference in Germany and today, much has been said and done in making this possible. We have put a large amount of thought and time into planning how ITiCSE 2011 can try to be "on a par" with the great past conferences.

This year, ITiCSE is set in the contrast of a very modern conference center right across the street from Darmstadt's historic city castle. The conference program should offer interesting elements for every attendee. Apart from the regular paper and poster sessions, we have three interesting working groups, two Tuesday excursions, keynotes, and a possible record number of tips & techniques. The conference dinner will be placed in the ruins of a castle south of Darmstadt.

We received 169 paper submissions, more than 50 posters and 20 tips & techniques submissions. The paper acceptance rate is 39%. All papers received at least four reviews, with most receiving five or even 6 reviews; the average number of reviews per paper was 4.92. All submissions were peer-reviewed, and we thank our more than 300 reviewers who helped in reviewing the submissions.

ITiCSE is located in Europe, but has a worldwide outreach. We received papers from 39 different countries covering every continent excluding Antarctica. Submissions were accepted from 19 countries, including 9 non-European ones. We tried to balance the program between quality and diversity, given the limited amount of time available. We believe that you will enjoy the program.

The organization of any conference is challenging, especially one so large and international in scope. We are very grateful to the many who helped make this conference possible. First we would like to thank our most excellent committee: Tom, Christian, Cary, Liz, Jürgen, Myles, Torsten, Steve, Rainer, Mikey, Stephen, Fabian and Michael. We also want to thank all the authors, reviewers, presenters, session chairs, working group leaders and participants. Thank you to our contact people at ACM headquarters, Lisa Tolles at Sheridan Printing, and the members of the SIGCSE Board for supporting all our crazy (innovative?) proposals. Thank you also to the conference exhibitors and sponsors.

Guido, Tom and Christian

2. Conference Committee

Conference Chair
Guido Rößling - TU Darmstadt, Germany

Program Co-Chairs
Thomas L. Naps - University of Wisconsin Oshkosh, USA
Christian Spannagel - PH Heidelberg, Germany

Treasurer and Registrar
Cary Laxer - Rose-Hulman Institute of Technology, USA

Working Groups
Elizabeth S. Adams - James Madison University, USA
Jürgen Börstler - Blekinge Institute of Technology, Sweden

Panels & Faculty Posters
Myles McNally - Alma College, USA

Student Posters
Torsten Brinda - Universität Erlangen, Germany

Tips, Techniques and Courseware
Steve Cunningham - Brown Cunningham Associates, USA
Rainer Oechsle - FH Trier, Germany

Proceedings
Michael Goldweber - Xavier University, USA

Student Activities
Stephen Cooper - Stanford University, USA
Fabian Rothmann - TU Darmstadt, Germany

Evaluations
Michael E. Caspersen - Aarhus University, Denmark

3. Conference Program

Monday, June 27, 2011

Monday, June 27, 2011 8:30 - 12:00	
8.30	**Monday Morning Plenary Session (europium 3.02/3.03/3.04)**
8.30	Welcome and Opening Session
9.00	Keynote: A Bouquet of Measures to Promote Computer Science in Middle and High Schools *Ulrik Schroeder (see bio on page 15)*
10.00	**Coffee Break (Foyer) & Posters I (argon 3.07)**
10.45	**Papers 1: Coding Skills (radon 3.05)** **Chair: Matt Bower, Macquarie University**
10.45	Security Injections: Modules to Help Students Remember, Understand, and Apply Secure Coding Techniques *Blair Taylor, Siddharth Kaza*
11.10	The Design and Coding of Greedy Algorithms Revisited *J. Ángel Velázquez-Iturbide*
11.35	Measuring Static Quality of Student Code *Dennis Breuker, Jan Derriks, Jacob Brunekreef*
10.45	**Papers 2: Web Development (neon 3.08)** **Chair: Lillian Cassel, Villanova University**
10.45	Awakening Rip Van Winkle: Modernizing the Computer Science Web Curriculum *Randy Connolly*
11.10	Experiences in Implementing a Studio Component into a Course for Novice Web Developers *Rebecca Grasser*
11.35	A Tool to Support the Web Accessibility Evaluation Process for Novices *Elaine Pearson, Christopher Bailey, Steve Green*

Monday, June 27, 2011 10:45 - 14:05	
10.45	**Panel Session (helium 3.09)** **Chair: William Honig, Loyola University Chicago**
10.45	Outreach Programs to Promote Computer Science and ICT to High School and Middle School Students *Mary Anne L. Egan, Catherine Lang, Reyyan Ayfer, Annemieke Craig, Jane Chu Prey*
12.00	**Lunch Break (Foyer)**
13.15	**Papers 3: Understanding OO (radon 3.05)** **Chair: Jean Goulet, Université de Sherbrooke**
13.15	Relationship Between Text and Action Conceptions of Programming: A Phenomenographic and Quantitative Perspective *Anna Eckerdal, Mikko-Jussi Laakso, Mike Lopez, Amitrajit Sarkar*
13.40	Automated Checks on UML Diagrams *Michael Striewe, Michael Goedicke*
14.05	AGUIA/J: A Tool for Interactive Experimentation of Objects *André L. Santos*
13.15	**Papers 4: Activities for Hardware Courses (neon 3.08)** **Chair: Myles McNally, Alma College**
13.15	Intelligent Systems Development in a Non Engineering Curriculum *Emily A. Brand, William L. Honig, Matthew Wojtowicz*
13.40	Design of Innovative Integrated Circuits in Education *Andre Schaefer, Matthias Mielke, Rainer Brueck*
14.05	Preparing Students for Future Architectures with an Exploration of Multi- and Many-Core Performance *Daniel Ernst*

Monday, June 27, 2011 13:15-16:00	
13.15	**Working Group Interim Reports (helium 3.09)** **Chairs: Elizabeth Adams, James Madison University** **Jürgen Börstler, Blekinge Institute of Technology**
13.15	Motivating All Our Students? *Janet E. Carter*
13.40	Informatics in Secondary Education *Peter Hubwieser, Torsten Brinda, Johannes Magenheim, Sigrid Schubert*
14.05	Information Assurance Education in Two and Four-Year Institutions *Lance Peréz, Steve Cooper, Elizabeth Hawthorne, Susanne Wetzel*
14.30	**Break between Sessions**
14.45	**Papers 5: Attracting K-12 Students to CS (radon 3.05)** **Chair: Moti Ben-Ari, Weizmann Institute of Science**
14.45	A Study in Engaging Female Students in Computer Science Using Role Models *Jonathan Black, Paul Curzon, Chrystie Myketiak, Peter McOwan*
15.10	Kinesthetic Learning of Computing via Off-beat Activities *Ursula Wolz, Michael Milazzo, Meredith Stone*
15.35	A Technology-Assisted Scavenger Hunt for Introducing K-12 Students to Sensor Networks *Sally K. Wahba, Yvon Feaster, Jason O. Hallstrom*
14.45	**Papers 6: Enhancing CS Lectures (neon 3.08)** **Chair: Mark Guzdial, Georgia Tech**
14.45	Does Lecture Capture Make a Difference for Students in Traditional Classrooms? *Amber Settle, Lucia Dettori, Mary Jo Davidson*
15.10	Impact of an e-learning platform on CSE lectures *Guillaume Jourjon, Salil Kanhere, Jun Yao*

Monday, June 27, 2011 14:45-18:00	
15.35	Evaluating How Students would use a Collaborative Linked Learning Space System *Kai Michael Höver, Michael Hartle, Guido Rößling, Max Mühlhäuser*
14.45	**Papers 7: Environments for Motivating Students (helium 3.09) Chair: Douglas Harms, DePauw University**
14.45	Efficient and Playful Tools to Teach Unix to New Students *Matthieu Moy*
15.10	Creativity Room 5555 *Timo Göttel, Jonas Schild*
15.35	Discovering Logic through Comics *Iliano Cervesato*
16.00	**Coffee Break (Foyer) & Posters II (argon 3.07)**
16.45	**Papers 8: Tool Support for Upper-Level Courses (radon 3.05) Chair: Guillaume Jourjon, Australian Technology Park Research Lab**
16.45	Integrating Google Technology in Artificial Intelligence *Elena Sanchez-Nielsen, Stefan Klink*
17.10	Interactive Tools in the Graphics Classroom *Dino Schweitzer, Jeff Boleng, Lauren Scharff*
17.35	Using The Score Software Package To Analyse Novice Computer Graphics Programming *Maximilian Wittmann, Matthew Bower, Manolya Kavakli-Thorne*
16.45	**Papers 9: Integrating Web-based Technologies in Courses (neon) Chair: Mark Pullen, George Mason University**
16.45	Evaluating a Web-Based Information System for Managing Master of Science Summer Projects *Till Rebenich, Andrew M. Gravell, Thanassis Tiropanis*

Monday, June 27, 2011 16:45-18:00	
17.10	A Study of Video-based versus Text-based Labs for a Management Information Systems Course *Eric Breimer, Michelle Conway, Jami Cotler, Robert Yoder*
16.45	**Papers 10: Collaboration and Peer Instruction in CS 1 (helium 3.09)** **Chair: Åsa Cajander, Uppsala University**
16.45	Effects of Team-Based Learning on a CS1 Course *Patricia Lasserre, Carolyn Szostak*
17.10	A Multi-classroom Report on the Value of Peer Instruction *Leo Porter, Cynthia Bailey Lee,Beth Simon,Quintin Cutts, Daniel Zingaro*
17.35	CoMoTo - The Collaboration Modeling Toolkit *Cinda Heeren, Charlie Meyer, Eric Shaffer, Jon Tedesco*

Tuesday, June 28, 2011

Tuesday, June 28, 2011 8:45-10:00	
8.45	**Papers 11: Free-Text Questions and Assessment (radon 3.05)** **Chair: Girija Krishnaswamy, Australian Catholic University**
8.45	A Marking Language for the Oto Assignment Marking Tool *Guy Tremblay, Lessard Paul*
9.10	Supporting student-generated free-response questions *Andrew Luxton-Reilly, Paul Denny, Beryl Plimmer, Daniel Bertinshaw*
9.35	Automated Assessment of Short Free-Text Responses in Computer Science using Latent Semantic Analysis *Richard Klein, Angelo Kyrilov, Mayya Tokman*
8.45	**Papers 12: Introductory Programming (neon 3.08)** **Chair: Don Goelman, Villanova University**
8.45	WeScheme: The Browser is Your Programming Environment *Danny Yoo, Emmanuel Schanzer, Shriram Krishnamurthi, Kathi Fisler*
9.10	Habits of Programming in Scratch *Mordechai Ben-Ari, Orni Meerbaum-Salant, Michal Armoni*

Tuesday, June 28, 2011 8:45-12:00	
9.35	From Concrete to Abstract? Problem Domain in the Learning of Introductory Programming *Osvaldo Luiz Oliveira, Ana Maria Monteiro, Norton Roman*
8.45	**Supporter Session I (helium 3.09)**
8.45	Details to be announced
9.10	Details to be announced
9.35	Details to be announced
10.00	**Coffee Break (Foyer) and Posters III (argon 3.07)**
10.45	**Papers 13: K-12 Approaches I (radon 3.05)** **Chair: Yaakov L. Varol, University of Nevada, Reno**
10.45	Draw a Social Network *Sarah Carruthers, Todd Milford, Timothy Pelton, Ulrike Stege*
11.10	CS Education Re-Kindles Creativity in Public Schools *Vicki Bennett, Kyu Han Koh, Alexander Repenning*
11.35	A Pre-College Professional Development Program *Stephen Cooper, Wanda Dann, Dan Lewis, Pam Lawhead, Susan Rodger, Madeleine Schep, RoxAnn H. Stalvey*
10.45	**Papers 14: Visualization (neon 3.08)** **Chair: Leonard Mselle, The University of Dodoma, Tanzania**
10.45	GUIGraph – Editing Live Object Diagrams for GUI Generation Enables New Pedagogy in CS1/2 *Duane Buck*
11.10	Toward Replicating Handmade Algorithm Visualization Behaviors in a Digital Environment: A Pre-Study *Ming-Han Lee, Guido Rößling*
11.35	Improving Compilers Education through Symbol Tables Animations *J. Urquiza-Fuentes, F. Manso, J. Á. Velázquez-Iturbide, Manuel Rubio-Sánchez*

Tuesday, June 28, 2011 10:45-14:00 & Excursions	
10.45	**Tips,Techniques, and Courseware: Ideas and Insights (helium 3.09)** **Chair: Judy Sheard, Monash University**
10.45	A Medical Motif for Teaching Computer Graphics in Context *James Wolfer*
10.55	Courseware: Student Learning via FOSS Field Trips *Heidi Ellis, Gregory Hislop*
11.05	Infandango: Automated Grading For Student Programming *Michael Hull, Daniel Powell, Ewan Klein*
11.15	Scheduling and Student Performance *Cliff Shaffer*
11.25	Use a Little History *John Impagliazzo*
11.35	Best Practices for Teaching Mobile Application Development *Qusay Mahmoud*
11.45	A Mobile Web-based Approach to Introductory Programming *Qusay Mahmoud*
12.00	**Lunch Break (Foyer)**
14.00	**Excursions (see page 45 for more details)**

Wednesday, June 29, 2011

Wednesday, June 29, 2011 08:30-10:00	
8.30	**Wednesday Keynote (europium 3, 3.02/3.03/3.04)**
8.30	Technology for teaching the Rest of Us *Mark Guzdial (see bio on page 15)*
9.30	**TT&C II: Improving Computer Architecture Courses (radon 3.05)** **Chair: Rainer Oechsle, FH Trier Germany**
9.30	Two Kinesthetic Learning Activities: Turing Machines and Basic Computer Organization *Michael Goldweber*
9.40	*dropped from the program*
9.50	Teaching Computer Architecture with a Graphical PC Simulator *Michael Black, Manoj Franklin*
9.30	**TT&C III: Tools and APIs (neon 3.08)** **Chair: Dianna Xu, Bryn Mawr College**
9.30	A Java Implementation of the Myro API for Using Personal Robots in CS1 *Douglas Harms*
9.40	Changes to JFLAP to increase its Use in Courses *Susan Rodger, Henry Qin, Jonathan Su*
9.50	Merlin-Mo, an Interactions Analysis System for Moodle Raquel Hijon-Neira, J. Ángel Velázquez-Iturbide
9.30	**TT&C IV: Supporting Novice Programmers (helium 3.09)** **Chair: Diana Cukierman, Simon Fraser University**
9.30	UWA Java Tools: Harnessing software metrics to support novice programmers *Rachel Cardell-Oliver, Patrick Doran-Wu*

Wednesday, June 29, 2011 09:30-12:00	
9.40	Using Greenfoot in Teaching Inheritance in CS1 *Tamar Vilner, Ela Zur, Shay Tavor*
9.50	Animation Projects in CS1 from Scheme to Java *Mirela Djordjevic*
10.00	**Coffee Break (Foyer) and Posters IV (argon 3.07)**
10.45	**Papers 15: Facilitating Programming Instruction (radon 3.05)** **Chair: Janet Carter, University of Kent**
10.45	Understanding the syntax barrier for novices *Paul Denny, Andrew Luxton-Reilly, Ewan Tempero, Jacob Hendrickx*
11.10	Understanding Novice Programmer Difficulties via Guided Learning *Shuhaida Mohamed Shuhidan, Margaret Hamilton, Daryl D'Souza*
11.35	Continual And Explicit Comparison to Promote Proactive Facilitation During Second Computer Language Learning *Matt Bower, Annabelle McIver*
10.45	**Papers 16: Broadening the Perspective (neon 3.08)** **Chair: Charles Riedesel, University of Nebraska at Lincoln**
10.45	What Might Computational Thinking Mean and Imply? *Chenglie Hu*
11.10	Beyond Good and Evil Impacts: Rethinking the Social Issues Components in Our Computing Curricula *Randy Connolly*
11.35	Computing student practices of cheating and plagiarism: a decade of change *Judy Sheard, Martin Dick*
10.45	**Supporter Session II (helium 3.09)**
10.45	Lab explorations for multi-core introductory courses *Sergei Nemnyugin, Saint Petersburg University / intel*

Wednesday, June 29, 2011 12:00-14:30	
11.15	Out-of-the-box Education. Teaching Business Process Excellence in a self-explanatory, realistic setting *Jürgen Powik, Software AG*
11.30	Overview of the BlackBerry Academic Program *Christine Arsenault, Research in Motion*
11.45	E-Learning in the Future Internet: Quality First! The Darmstadt PhD School on E-Learning *Max Mühhäuser, TU Darmstadt*
12.00	**Lunch Break (Foyer)**
13.15	**Papers 17: K-12 Approaches II (radon 3.05)** **Chair: Catherine Lang, Swinburne University**
13.15	Computer Science and Nursery Rhymes: A Learning Path for the Middle School *Doranna Di Vano, Claudio Mirolo*
13.40	Experimental Evaluation of BeadLoom Game: How Adding Game Elements to an Educational Tool Improves Motivation and Learning *Acey Boyce, A. Campbell, Shaun Pickford, Dustin Culler, Tiffany Barnes*
14.05	Teaching CS Unplugged in the High School (with Limited Success) *Yvon Feaster, Luke Segars, Sally K. Wahba, Jason O. Hallstrom*
13.15	**Papers 18: Peer-Assisted Learning (neon 3.08)** **Chair: Amber Settle, DePaul University**
13.15	A scheme for improving ICT units with critically low student satisfaction *Jason Ceddia, Angela Carbone, Jessica Wong*
13.40	Combining Multiple Pedagogies to Boost Learning and Enthusiasm *Lori Pollock, Terry Harvey*
14.05	A Cooperative Learning-based Strategy for Teaching Relational Algebra *Alexandra Martinez, Arturo Camacho*

Wednesday, June 29, 2011 13:40-16:00	
13.15	**Papers 19: Engaging Students (helium 3.09)** **Chair: Henry Walker, Grinnell College**
13.15	Open Source Contribution as an Effective Software Eng. Class Project *Robert Marmorstein*
13.40	Extreme Apprenticeship Method: Key Practices and Upward Scalability *Arto Vihavainen, Matti Paksula, Matti Luukkainen, Jaakko Kurhila*
14.05	The Academic Enhancement Program in Introductory CS: A Workshop Framework Description and Evaluation *Rylan Egan, Diana Cukierman, Donna McGee Thompson*
14.30	**Break between Sessions**
14.45	**Papers 20: New Approaches in Undergraduate Instruction (radon)** **Chair: Roger McDermott, Robert Gordon University**
14.45	Getting CS Undergraduates to Communicate Effectively *Andreas Karatsolis, Iliano Cervesato, Nael Abu-Ghazaleh, Yonina Cooper, Khaled Harras, Kemal Oflazer, Thierry Sans*
15.10	Undergraduate Research: a Case Study *Herman Koppelman, Betsy van Dijk, Gerrit van der Hoeven*
15.35	Bringing Undergraduate Students Closer to a Real-World Information Retrieval Setting: Methodology and Resources *Julián Urbano, Mónica Marrero, Diego Martín, Jorge Morato*
14.45	**Papers 21: Automated Assessment (neon 3.08)** **Chair: Cliff Shaffer, Virginia Tech**
14.45	Combining Automated Exercise Evaluations with Algorithm Animations *Guido Rößling, Mihail Mihaylov, Jerome Saltmarsh*
15.10	Using Run Time Traces in Automated Programming Tutoring *Michael Striewe, Michael Goedicke*
15.35	A proposal for Automatic Evaluation in a Compiler Construction Course *Emilio Julio Lorenzo Galgo, Javier Vélez Reyes, Anselmo Peñas*

Wednesday, June 29, 2011 12:00-14:30	
14.45	**Papers 22: Attracting Girls and Women to CS (helium 3.09)** **Chair: Susan Rodger, Duke University**
14.45	Evaluation Framework Underpinning the Digital Divas Programme *Annemieke Craig, Julie Fisher, Helen Forgasz, Catherine Lang*
15.10	The Impact of IMPACT: Assessing students' perceptions after a day of computer exploration *Mary Anne Egan, Timothy Lederman*
15.35	Female Students' experiences of programming: It's not all bad! *Reena Pau, Marcus Grace, John Woollard, Wendy Hall*
16.00	**Closing Session**
18.00	**Conference Dinner**

3.1. Keynote Speakers

Ulrik Schroeder

Dr. Schroeder received his Diploma degree as well as his PhD in Computer Science from Technische Universität (TU) Darmstadt. Currently, he heads the research group Computer-supported Learning and Didactics in Computer Science in the computer science department at RWTH Aachen University as well as the university's Center for innovative Learning (CiL).

His research interests include assessment and intelligent feedback with a focus on learning processes, Web 2.0 applications and social software in education, mobile Internet and learning, gender mainstreaming in education, and Computer Science didactics. He teaches classes in CS1, Web Technologies, eLearning and Computer Science Education. He is active in the GI, the German sister of the ACM, as vice-chair of the special interest group eLearning and member of the Computer Science Education board.

Mark Guzdial

Mark Guzdial is a Professor in the School of Interactive Computing in the College of Computing at Georgia Institute of Technology. His research focuses on learning sciences and technology, specifically, computing education research. He has published several books on the use of media as a context for learning computing. He was the original developer of the "Swiki" which was the first wiki designed for educational use. He received the Ph.D. degree in Education and Computer Science from the University of Michigan in 1993. He serves on the both ACM's Education Board and the Special Interest Group in CS Education (SIGCSE) Board, and is on the editorial boards of the "Journal of the Learning Sciences," "ACM Transactions on Computing Education," and "Communications of the ACM."

3.2. Faculty and Student Posters (argon 3.07)

Posters I: Monday, 10.00-10.45 (argon 3.07)

Using Videogames to Teach Security *Mario A.M. Guimaraes, Huwida Said, Richard Austin*
A System for Usable Unification of Interfaces of Learning Objects in M-Learning *Eva Garcia, Luis de-Marcos, Antonio Garcia, Jose-Ramon Hilera*
Best Practices for Peer Feedback in Interdisciplinary Research Groups *Sebastian Harrach*
Programming in Secondary Education: Benefits and Perspectives *Michail Giannakos, Spyros Doukakis, Panayiotis Vlamos, Christos Koilias*
Integrating Greenfoot into CS1 – a Case Study *Tamar Vilner, Ela Zur, Shay Tavor*
GLMP for Automatic Assessment of DFS Algorithm Learning *M. Gloria Sanchez-Torrubia, Carmen Torres-Blanc, Gracian Trivino*
An Innovative Teaching Tool Based on Semantic Tableaux for Verification and Debugging of Programs *Rafael del Vado Vírseda, Fernando Pérez Morente*
Moodle-Integrated Open Source Synchronous Teaching *J Mark Pullen, Nicholas Clark*
Teaching with CEOHP *Vicki Almstrum, Deepa Muralidhar, Mary Last, Barbara Boucher Owens*
Facilitating Learning Dynamic Programming Through a Previous Introduction of Exhaustive Search *Arturo Camacho, Alexandra Martinez*
A Bioinformatics E-learning Lab for Undergraduate Students *Feng Lu, Hui Liu, Yi Jian, Yanhong Zhou, Zhenran Jiang*
A Model for Visualizing Sentence Complexity *Stefanie Markham, Ying Zhu*

Posters II: Monday, 16.00-16.45 (argon 3.07)

Do Educational Software Systems Provide Satisfactory Learning Opportunities for 'Multi-Sensory Learning' Methodology? *Girija Krishnaswamy, Peter Chan*
Adaptation of Educational Contents to Mobile Devices *Antonio Garcia, Eva Garcia, Luis de-Marcos, Jose-Antonio Gutierrez*
A Comparison of Software Engineering Knowledge Gained from Student Participation in Humanitarian FOSS Projects *Heidi Ellis, Gregory Hislop, Ralph Morelli*
CROKODIL – a Platform Supporting the Collaborative Management of Web Resources for Learning Purposes *Mojisola Anjorin, Renato Domínguez García, Christoph Rensing*
Is Iteration Really Easier to Master than Recursion? An Investigation in a Functional-First CS1 Context *Claudio Mirolo*
Natural Language in Introductory Programming: an experimental study *Osvaldo Oliveira, Ana Maria Monteiro*
A first step Mapping IMS Learning Design and Merlin-Mo *Raquel Hijon-Neira, Ángel Velázquez-Iturbide*
Using Student Blogs For Documentation In Software Development Projects *Robert Law*
How Educators Find Educational Resources Online *Monika Akbar, Clifford A. Shaffer, Weiguo Fan, Lillian Cassel, Lois Delcambre, Edward A. Fox, Yinlin Chen*
A Contextualized Project-based Approach for Improving Student Engagement and Learning in AI Courses *Ingrid Russell, Zdravko Markov, Joy Dagher*
Java2Sequence - A Tool for the Visualization of Object-Oriented Programs in Introductory Programming *Joao Paulo Barros, Luis Biscaia, Miguel Vitoria*

Posters III: Tuesday, 10.00-10.45 (argon 3.07)

An Initial Look at Prospective Student Mentoring *Amber Settle, Sarah Pieczynski, Liz Friedman, Mary Jo Davidson*
IR2gT: A Report Generation Tool for Institutional Repository *Jayan Kurian Chirayath, Ashly Markose, Blooma Mohan John*
An Update On The Use Of Community-Based Non-Profit Organizations In Capstone Projects *Paul Leidig, David Lange, Roger Ferguson*
Findings from an ACM Strategic Summit on Computing Education in Community Colleges *Elizabeth Hawthorne, Robert Campbell, Karl Klee*
A Problem Solving Teaching Guide Based on a Procedure Intertwined with a Teaching Model *Ronit Ben-Bassat Levy, J. Ángel Velázquez-Iturbide*
A Normative Competence Structure Model for “Embedded Micro- and Nanosystems” Development *André Schäfer, Rainer Brück, Steffen Jaschke, Sigrid Schubert, Dietmar Fey, Bruno Kleinert, Harald Schmidt*
Identifying the Predictors of Educational Webcasts’ Adoption *Michail Giannakos, Panayiotis Vlamos*
Investigating Cognitive Structures of Object Oriented Programming *Peter Hubwieser, Andreas Mühling*
The Beaver Contest - Attracting Youngsters to Study Computing *Bruria Haberman, Avi Cohen, Valentina Dagiene*
Computing for the Social Good: A Service Learning Project *Michael Goldweber*
A Collaborative Linked Learning Space *Kai Michael Höver, Michael Hartle, Guido Rößling*
STEM and ICT Instructional Worlds: The 3D Experience *Yvon Feaster, Katherine Ross*

Collaborating Across International Boundaries … Using Twitter as a Tool in the Classroom
Stefanie Markham, Saeid Belkasim

Posters IV: Wednesday, 10.00-10.45 (argon 3.07)

Enhancing Learner Capability: Success of IT@School Project, Kerala, Region of India
Girija Krishnaswamy, V. Sasi Kumar

Deconstructing VLEs to Create Customized PLEs
Salvador Ros, Agustin C. Caminero, Antonio Robles, Roberto Hernandez

What Matters Most When Teaching CS1?
Scheila Martins, Ana Paula L. Ambrósio

Combining Memory Management and Filesystems in an Operating Systems Course
Hans-Georg Eßer

SyntaxTrain: Relieving the Pain of Learning Syntax
Andreas Leon Aagaard Moth, Joergen Villadsen, Mordechai Ben-Ari

The Impact of Memory Transfer Language (MTL) on Reducing Misconceptions in Teaching Programming to Novices
Leonard Mselle

Optimizing Collaborative Learning Processes by Using Recommendation Systems
Sebastian Harrach, Mojisola Anjorin

The Use of Mediating Artifacts in Embedding Problem Solving Processes in an E-Learning Environmnent
Orry Messer, Angelo Kyrilov

Introducing Students to Computer Science With Programmes That Don't Emphasise Programming
Bruria Haberman, Tim Bell, Paul Curzon

Integrating Scholarly Articles Within E-Learning Courses: A Framework
bee bee chua, Danilo Valeros Bernardo

Supporting Peer Learning with Ad-Hoc Communities *Johannes Konert, Kristina Richter*
Exploring Flow in Novice Programming Environments *Mark Zarb*
Muddy Hill Games *Elizabeth Sweedyk, Jessica Blevins, Andy Kearney, Eric Mullen, Emily Myers-Stanhope*

4. Abstracts

4.1. Monday Keynote: 09.00-10.00

A bouquet of measures to promote Computer Science in Middle & High Schools

Ulrik Schroeder (RWTH Aachen)

Computer science faces the same problem worldwide: too few women in computing, low enrollment and high failure rates in university programs. There is growing consciousness which has led to various ideas and programs to deal with these problems. My talk will introduce a set of measures which we have implemented at RWTH Aachen University to tackle the problem on different levels. Therefore, I will present our successful program *go4IT!* of robotic courses for girls in 5th and 6th grade and some empirical results of how these influence their interest in STEM topics. Currently we are developing follow-up measures based on Android and GUI programming on smartphones. These measures cumulate into our newly founded Computer Science Junior Academy InfoSphere as out-of-school educational lab. There we offer a variety of learning modules which can last between two hours and a few days. Complete school classes can book InfoSphere modules or interested pupils of different ages can collaborate on self-organized projects. We have implemented various learning programs for multi-modal devices such as multi-touch tables and smartphones in order to introduce different topics of computer science. Our vision for these educational labs is the enhancement to different disciplines which bring together learners of all ages from K-12, students, researchers and professionals for lifelong learning and vocational training.

4.2. Monday Paper Sessions 1, 2, Panel: 10.45-12.00

Security Injections: Modules to Help Students Remember, Understand, and Apply Secure Coding Techniques

Blair Taylor, Siddharth Kaza (Towson University)

With our global reliance on software, secure and robust programming has never been more important. Yet academic institutions have been slow to add secure coding to the curriculum. We present a model using checklist-based security injection modules to increase student awareness and ability to apply secure coding principles, specifically - identify, understand, and correct key security issues in code. The model is evaluated by mapping assessment questions to the cognitive dimension of the revised Bloom's taxonomy. Experiments with students in four sections of CS0 and CS1 show that students using our modules perform significantly better at remembering, understanding and applying secure coding concepts. Students exposed to the modules also show increased ability to write code to address specific security issues.

The Design and Coding of Greedy Algorithms Revisited

J. Ángel Velázquez-Iturbide (Universidad Rey Juan Carlos)

In this paper we argue that the most typical instruction method used to teach greedy algorithms is inadequate at achieving certain learning goals and we present several

contributions to alleviate this situation. Our first group of contributions highlights the role of selection functions and proposes separate treatment in their discovery and proof of optimality. For discovery, we outline some interesting cases of selection functions and for proofs, we examine the role of counterexamples. Furthermore, we argue that their separation provides more opportunities for instructional activities. Our second group of contributions concerns coding greedy algorithms. We discuss the role and adequacy of the template in current use, and also the role of sorting candidates and how to implement sorting.

Measuring Static Quality of Student Code

Dennis Breuker, Jan Derriks (Hogeschool van Amsterdam), Jacob Brunekreef (Fontys Hogeschool)

In this paper we report about a large-scale measurement programme concerning the static quality of student-written Java code. The goal of the programme is two-fold: we investigate what metrics are useful for measuring static quality in an educational setting, and we investigate what conclusions can be drawn from the measurement results.

Awakening Rip Van Winkle: Modernizing the Computer Science Web Curriculum

Randy Connolly (Mount Royal University)

The world of web development has experienced a great deal of change over the past decade. The importance and complexity of web development is currently not adequately reflected in the ACM Computer Science 2008 Curriculum, nor in most reported computer science programs. This paper examines published literature on teaching the web since 2001 and argues that the computer science curriculum needs to be woken up and modernized in regards to the importance of web development. The paper critiques the approach of teaching web development topics within a single course. It articulates a wide variety of web development topics that need to be covered in any contemporary computer science program and which are often absent in other published accounts of this course. The paper concludes by arguing that a multi-course stream in web development can help the students integrate the discrete pieces of knowledge garnered during their undergraduate education.

Experiences in Implementing a Studio Component into a Course for Novice Web Developers

Rebecca Grasser (Lakeland Community College)

This paper discusses an educational experience aimed at producing better computer-based projects by including in the assessment process an art appreciation-like critique component. We tested this idea at a medium-size Community College offering many degrees in various applied areas of computing (such as web programming). In a way that at times parallels the constructive criticism of work done by art students we asked our participants to critique their classmates as well as themselves. New elements of assessment such as aesthetics, user-engagement, naturalness, completeness, and unrealized potential of the projects appeared in the discourse. In our opinion, the assessment of computer code as just being correct or

incorrect is enriched when designers look at their artifacts as pieces to be shown to others (technical and non-technical viewers) in a studio-like environment. Although it is very early to extrapolate results, our conclusion is that a noticeable improvement of the overall quality of the final products as well as a higher level of collaboration, participation, and student satisfaction resulted from this approach.

A Tool to Support the Web Accessibility Evaluation Process for Novices

Elaine Pearson, Christopher Bailey, Steve Green (Teeside University)

The Accessibility Evaluation Assistant (AEA) is designed to assist novice auditors in the process of an accessibility evaluation of websites. The tool has been incorporated into an undergraduate computing module in Accessibility and Adaptive Technologies. It takes a structured walkthrough approach to guide the novice through a series of checks for established web accessibility principles with the goal of conducting a comprehensive accessibility evaluation. An initial evaluation of the AEA with 38 students confirms its potential for supporting the development of skills in auditing of websites for accessibility.

Panel: Outreach Programs to Promote Computer Science and ICT to High School and Middle School Students

Mary Anne L. Egan (Siena College), Catherine Lang (Swinburne University of Technology), Reyyan Ayfer (Bilkent University), Annemieke Craig (Deakin University), Jane Chu Prey (Microsoft Research)

Can hands-on computing programs for high school and middle school students create interest in Information and Communication Technology (ICT)? Which outreach program is appropriate for your school or company? We will explore several established programs with a view of sharing resources and ideas. We will begin a conversation to determine if current workshops do enough to engender student desire to act by committing to a higher degree in ICT. We will be presenting information on existing programs in different countries, such as the Digital Divas program (Australia), the Digigirlz program (various countries) and project IMPACT (US).

Attendees of this panel will be exposed to details of each of the named programs with the objective of enabling them to be able to organize and present a suitable outreach program of their own.

This panel is timely and necessary because the 'shrinking pipeline' of women and men in ICT is a problem in many countries and has dwindled to a trickle with students often making poor decisions due to misinformation and stereotypes. These ICT programs aimed at middle school and high school students expose them to the potential for exciting careers in ICT and increase awareness that technical innovation plays a critical role in virtually every sector of the global economy. As academics and industry professionals, we are concerned that a shrinking talent pool leads to reduced innovation and competitiveness. Current trends indicate that we need to act now to encourage more young people into ICT to meet future employment and creativity demands.

The materials presented will include sample activities, sample agendas, coding exercises, etc which will be available to all attendees.

4.3. Monday Paper Sessions 3, 4: 13.15-14.30

Note: the information about the Working Group Session is on page 43.

Relationship Between Text and Action Conceptions of Programming: A Phenomeno–graphic and Quantitative Perspective

Anna Eckerdal (Uppsala University), Mikko-Jussi Laakso (University of Turku), Mike Lopez (Manukau Institute of Technology), Amitrajit Sarkar (Christchurch Polytechnic Institute of Technology)

Phenomenographic research studies have identified different understandings of the concepts class and object by novice programmers. Aspects of understanding include a focus on artefacts of text, syntax and structure (text), as active agents in a program (action) and as models of an external reality (model). We explore the hypothesis that these aspects of conceptual understanding form a hierarchy in which mastery of the text aspect is a necessary precondition for understanding objects as active agents and the action aspect is a precondition for model understandings. We use empirical data from the final examination of an introductory programming course to test the relationship between the text and action aspects. Our findings do not support the hypothesis of a hierarchy but rather suggest that text and action understandings develop in parallel.

Automated Checks on UML Diagrams

Michael Striewe, Michael Goedicke (Paluno - The Ruhr Institute for Software Technology)

Automated checks for software artifacts like UML diagrams used in automated assessment or tutoring systems do often rely on direct comparisons between a solution and a sample solution. This approach has drawbacks regarding flexibility in face of different possible solutions which are quite common in modeling tasks. This paper presents an alternative technique for checking UML class diagrams based on graph queries which promises to be more flexible.

AGUIA/J: A Tool for Interactive Experimentation of Objects

André L. Santos (Lisbon University Institute)

Learning and teaching object-oriented programming are still perceived as being difficult tasks. This paper presents AGUIA/J, a pedagogical tool for interactive experimentation and visualization of object-oriented Java programs. The approach is based on having a graphical environment for experimenting a set of user-developed classes where objects of such classes can be created and controlled interactively. The main innovative aspects of the tool comprise the visualization of objects in widgets that take different forms according to their classes and state, a mechanism to address the query-command separation principle, and the capability of runtime adaptation of the objects in the workbench to new versions of their classes. An experiment using AGUIA/J as courseware in pilot lab classes has resulted in higher approval rates for the involved students, as well as significantly lower drop-out rates.

Intelligent Systems Development in a Non Engineering Curriculum

Emily A. Brand, William L. Honig, Matthew Wojtowicz (Loyola University Chicago)

Much of computer system development today is programming in the large–systems of millions of lines of code distributed across servers and the web. At the same time, microcontrollers have also become pervasive in everyday products, economical to manufacture, and represent a different level of learning about system development. Real world systems at this level require integrated development of custom hardware and software.
How can academic institutions give students a view of this other extreme–programming on small microcontrollers with specialized hardware? Full scale system development including custom hardware and software is expensive, beyond the range of any but the larger engineering oriented universities, and hard to fit into a typical length course. The course described here is a solution using microcontroller programming in high level language, small hardware components, and the Arduino open source microcontroller. The results of the hands-on course show that student programmers with limited hardware knowledge are able to build custom devices, handle the complexity of basic hardware design, and learn to appreciate the differences between large and small scale programming.

Design of Innovative Integrated Circuits in Education

Andre Schaefer, Matthias Mielke, Rainer Brueck (University of Siegen)

Teaching practical ASIC design, one faces a lot of problems, such as high manufacturing costs, long workflow, and heterogeneous previous knowledge of the students. On one hand, it is difficult to find topics that motivate students and, at the same time, are not too complex. On the other hand, the industry requires students who are trained not only theoretically, but also have practical experience in team work and project management. We regard these problems as a challenge and are going to create a concept of a project-oriented ASIC design course that focuses on teaching hard and soft skills.
In this paper, we describe our concept for a student project work, which leaves to the students a lot of degrees of freedom in the design process and offers the possibility to realize an own idea as integrated circuit.

Preparing Students for Future Architectures with an Exploration of Multi- and Many-Core Performance

Daniel Ernst (University of Wisconsin - Eau Claire)

The recent progression of modern computer architectures from serial to multi-core to many-core has raised numerous questions about the placement of parallel computing topics in undergraduate computer science curricula. While several papers have explored how programming and algorithms courses might introduce parallel programming topics, there has been very little discussion of the changes needed in computer organization and architecture coursework to help students understand key issues about the behavior of these platforms.
To increase the relevance of these courses and to tackle more modern architectural issues, we propose extending the performance coverage of a traditional computer organization course to include an exploration of the quantitative characterization of a program's performance on a variety of architectural platforms, including modern GPU hardware.

In this paper, we outline the changes made to the Computer Organization and Design course at University of Wisconsin – Eau Claire. These modifications included a 3 week unit on the basics of parallel and GPU architectures, along with a tiered project on program optimization.

4.4. Monday Paper Sessions 5, 6, 7: 14.45-16.00

A Study in Engaging Female Students in Computer Science Using Role Models

Jonathan Black, Paul Curzon, Chrystie Myketiak, Peter W. McOwan (Queen Mary, University of London)

An effective approach to engaging young women to take computing in higher education is to provide examples of successful female computer scientists. Can a print publication that combines core computing concepts with inspiring stories of women in the field be effective? In this paper, we describe a campaign that distributed a 60-page booklet on women in computing to UK secondary schools. We analyse the initial response from teachers, and draw some general conclusions from the project. Teachers expressed strong enthusiasm for the booklet, and also report the desire for recruitment and retention of girls in their computing programmes. They had confidence in the potential for this booklet to inspire young women to take computing.

Kinesthetic Learning of Computing via "Off-beat" Activities

Ursula Wolz (The College of New Jersey), Michael Milazzo (g8four), Meredith Stone

To broaden participation in computing requires exposing students to a variety of experiences. CS Unplugged promotes learning through kinesthetic activities. This paper identifies three types of learning (1) problem solving, (2) creative construction, and (3) open-ended invention, that lend themselves to activities that engage middle school students in physical movement to learn computing. Via surveys and a personality traits activity "True Colors" we examined whether self-identified personality types were predisposed to particular kinesthetic learning activities. Our results suggest that personality type as defined by True Colors does not predict selection of an activity type. Furthermore, the students in our summer Interactive Journalism Institute were significantly more predisposed to pick open-ended invention. These results suggest directions in which K-12 computing curriculum should take to reach the broadest constituency.

A Technology-Assisted Scavenger Hunt for Introducing K-12 Students to Sensor Networks

Sally K. Wahba, Yvon Feaster, Jason O. Hallstrom (Clemson University)

Sensor networks serve as a powerful recruiting vehicle to excite and engage students in socially-relevant applications of computing. In this paper, we describe a technology-assisted scavenger hunt for introducing young learners—from grade school through high school—to sensor networks and computer science. The desired outcome is to expand students' content knowledge and positively impact their impressions of the discipline. We describe the outreach program and present promising evaluation results across three pilots involving 5th graders, 7th graders, and 11th graders.

Does Lecture Capture Make a Difference for Students in Traditional Classrooms?
Amber Settle, Lucia Dettori, Mary Jo Davidson (DePaul University)

The College of Computing and Digital Media (CDM) at DePaul University has recorded thousands of courses using an in-house system called Course Online (COL) since 2001. These recordings are available not only to students enrolled in online CDM courses, but also to students in traditional classrooms at CDM. In this study we analyzed survey responses and grade data in order to determine whether traditional students found COL recordings to be a valuable substitutional tool and whether the recordings had any impact on student performance. We found that a large majority of traditional CDM students find the recordings useful and believe that they improve performance. Somewhat counter to the students' perceptions, we found that there were no large differences in performance prior to the introduction of COL recordings and after COL recordings began to be available.

Impact of an e-learning platform on CSE lectures
Guillaume Jourjon (NICTA), Salil Kanhere, Jun Yao (UNSW)

This article presents a comprehensive summary and recommendations towards the use of IREEL, an e-learning plat- form designed for network studies in CSE courses, based on our hands-on experience in a large hybrid undergraduate/postgraduate course at the UNSW. We found that the tool was well received by the students for understanding key concepts, especially when compared to legacy tools used in labs. Furthermore we show that our tool was able to handle a very large number of experiments in a relatively short amount of time.

Evaluating How Students would use a Collaborative Linked Learning Space System
Kai Michael Höver, Michael Hartle, Guido Rößling, Max Mühlhäuser (TU Darmstadt)

Personal interaction with learning materials and arranging learning resources in a semantically meaningful way is important for comprehension and personal knowledge construction. Current learning systems only enable learners to add simple ink or text notes. How would users work with a system for collaboratively augmenting and semantically connecting learning materials with related knowledge resources? This paper presents user studies conducted with both students and educators to elicit users' acceptance, needs and preferences regarding such a learning system.

Efficient and Playful Tools to Teach Unix to New Students
Matthieu Moy (Grenoble-INP (Ensimag))

Teaching Unix to new students is a common tasks in many higher schools. This paper presents an approach to such course where the students progress autonomously with the help of the teacher. The traditional textbook is complemented with a wiki, and the main thread of the course is a game, in the form of a treasure hunt. The course finishes with a lab exam, where students have to perform practical manipulations similar to the ones performed during the treasure hunt. The exam is graded fully automatically.
This paper discusses the motivations and advantages of the approach, and gives an overall view of the tools we developed. The tools are available from the web, and open- source, hence re-usable outside the Ensimag.

Creativity Room 5555: Evoking Creativity in Game Design amongst CS Students
Timo Göttel (University of Hamburg), Jonas Schild (University of Duisburg-Essen)

This paper highlights the importance of computer-free meeting places for Computer Science (CS) students participating in game design courses in the sense of learning spaces. Such environments have the potential of countering possible negative attitudes of CS students towards creative processes. During a game design course we offered a unique meeting room that offered playful and creative elements. Furthermore, the room had to be rearranged by the participants of the course. We describe the course and analyze it through observations and three formal surveys. Both the results and the quality of resulting prototypes indicate that CS students strongly benefit in their creative processes from working in individually arranged playful environments.

Discovering Logic through Comics
Iliano Cervesato (Carnegie Mellon University - Qatar)

This paper describes a new experimental course introduced in the Spring of 2010 at Carnegie Mellon University in Qatar. Discovering Logic is an introduction to logic for Computer Science majors in their freshman year. It targets students who have had little or no exposure to logic and has the primary objective of 1) preparing them for sophomore classes which require proficiency with understanding formal statements expressed in English, elementary reasoning skills, and a sense of mathematical rigor. The course is structured as to achieve two additional objectives: 2) develop the students' communication skills, and 3) give them some historical depth into Computer Science and logic. This led to a somewhat unconventional approach that used a comic book, Logicomix, as the course textbook and that empowered the students to be active agents in the learning process through presentations and numerous open discussions. Preliminary analysis hints at an improved performance in follow-up courses, indicating that it may be achieving its primary objective.

4.5. Monday Paper Sessions 8, 9, 10: 16.45-18.00

Integrating Google Technology in Artificial Intelligence
Elena Sánchez-Nielsen (Universidad de La Laguna), Stefan Klink (Karlsruhe Institute of Techn.)

The use of Google technology in artificial intelligence courses gives new opportunities to focus the academic topics and learning objectives in order to meet the needs of students and computer science curricula. With the integration of Google developer toolkits, we can transform the class in order to use industry software tools as a motivating topic and as the domain for real-world software projects. This paper describes the methodology and motivating results using Google developer toolkits in teaching and practicing computer applications, which are based on artificial intelligence theory and techniques.

Interactive Tools in the Graphics Classroom
Dino Schweitzer, Jeff Boleng, and Lauren Scharff (United States Air Force Academy)

Computer graphics is a fun course for both teachers and students. The topics are filled with interesting images and animations, there is a wealth of support material available, and students are motivated to express creativity in projects. There are also underlying math

concepts and algorithms that some students find challenging to fully understand. At our institution, we teach a computer graphics course to junior and senior-level computer science majors as an elective. To assist their understanding of fundamental concepts and algorithms, we created and employed a collaborative learning approach using locally developed interactive tools during each lecture. The Think-Pair-Share model was used to facilitate collaborative interaction between students. The results of this approach were measured through in-class feedback questions and student performance on individual exam questions. Students enjoyed using the tools, highly rating them on the feedback forms, but were less enthusiastic about the classroom methodology used to present them. These results along with lessons learned will be addressed.

Using the SCORE software package to analyse novice Computer Graphics programming

Maximilian Wittmann, Matthew Bower, Manolya Kavakli-Thorne (Macquarie University)

This paper presents the SCORE (Student Coding Observation and Recording Engine) software package designed to capture and analyse student coding processes. The package consists of an Eclipse [1] plug-in to gather observational data while students code a programming task, and an analysis tool that allows researchers to visualise, categorise and annotate changes in code. Because the SCORE package supports code text level analysis it enables more in-depth understanding of student programming and problem solving approaches than meta-data or program output analysis tools. SCORE also provides features to assist the analysis of Computer Graphics programs. An example analysis of a student's Computer Graphics assignments demonstrates how SCORE was used to reveal the dominant role of general programming issues in the early assignment, whereas spatial programming issues persisted throughout both assignments.

Evaluating a Web-Based Information System for Managing Master of Science Summer Projects

Till Rebenich, Andrew M. Gravell, Thanassis Tiropanis (University of Southampton)

We describe the design of a web-based information system for monitoring MSc summer projects in the School of Electronics and Computer Science at the University of Southampton, and a mixed method quasi-experimental study involving 290 MSc project students, 19 monitors, and 69 supervisors in electronics and computer science, using the system over a period of 17 weeks. Statistically significant results presented here are: Students making heavy use of the system achieved higher marks on their project dissertation, while no such correlation was found with marks for other parts of their MSc. Likewise, student's monitor activity is significantly correlated with their own activity and dissertation mark. These results suggest that educational information and project management systems positively affect student achievement and academic staff involvement is crucial for these systems to be successful. Future work includes a more detailed analysis of success factors and their impact on student performance.

A Study of Video-based versus Text-based Labs for a Management Information Systems Course

Eric Breimer, Michelle Conway, Jami Cotler, Robert Yoder (Siena College)

In this paper we study key differences between video-based and text-based instructions by developing and testing an interactive website for delivering lab material in our Management Information Systems course. In a face-to-face lab setting, we tracked the performance and surveyed the impressions of 80 students where approximately half received video instructions while the other half received text instructions. The results indicate no statistically significant difference in students' correctness in answering lab questions, but slight differences in completion time and impression. We discuss our results and suggest ways to use video instruction effectively based on our results and observations.

Effects of Team-Based Learning on a CS1 Course

Lasserre Patricia, Carolyn Szostak (UBC Okanagan)

Many active learning techniques have been used and described over the years, including team-based learning (TBL). While this technique is well established, it is only recently that analyses that compare it to other teaching techniques have been reported. In this paper, we evaluate the impact of team-based learning on two major concerns for computer science instructors: the drop/attrition rates, and students' success in CS1. The results show some major improvements both in terms of the drop rate and students' success, as measured by final exam grades. For example, the number of students obtaining 50% or more on the final exam has increased from 54% to 75.5%. Moreover, the drop rate has decreased from more than 30% to 6.4%.

Experience Report: A Multi-classroom Report on the Value of Peer Instruction

Leo Porter, Cynthia Bailey-Lee, Beth Simon (University of California, San Diego), Quintin Cutts (University of Glasgow), Daniel Zingaro (University of Toronto)

Peer Instruction (PI) has a significant following in physics, biology, and chemistry education. Although many CS educators are aware of PI as a pedagogy, the adoption rate in CS is low. This paper reports on four instructors with varying motivations and course contexts and the value they found in adopting PI. Although there are many documented benefits of PI for students (e.g. increased learning), here we describe the experience of the instructor by looking in detail at one particular question they posed in class. Through discussion of the instructors' experiences in their classrooms, we support educators in consideration of whether they would like to have similar classroom experiences. Our primary findings show instructors appreciate that PI assists students in addressing course concepts at a deep level, assists instructors in dynamically adapting their class to address student misunderstandings and, overall, that PI encourages students to be engaged in conversations which help build technical communication skills. We propose that using PI to engage students in these activities can effectively support training in analysis and teamwork skills.

CoMoTo - the Collaboration Modeling Toolkit

Charlie Meyer, Cinda Heeren, Eric Shaffer, Jon Tedesco (University of Illinois at Urbana-Champaign)

We are excited to introduce CoMoTo – the Collaboration Modeling Toolkit – a new, web-based application that expands and enhances well-known software similarity detection systems. CoMoTo is an end-to-end data management, analysis, and visualization system whose purpose is to assist instructors of courses requiring programming exercises to monitor and investigate the extent of student collaboration, both allowed and illicit. We describe CoMoTo's interface, which was designed to facilitate scrutiny of collaboration data projected along student, course, assignment, etc. attributes, and to allow for interactive visualization of pairwise similarity measures via a dynamic graph. We also elaborate on the details of CoMoTo's implementation. Finally, we briefly discuss two use cases that foreshadow CoMoTo's broad utility in student code analysis, not only for plagiarism detection, but also for investigating early student coding styles, and for evaluating software similarity detection systems, themselves.

4.6. Tuesday Paper Sessions 11, 12: 8.45-10.00

A Marking Language for the Oto Assignment Marking Tool

Guy Tremblay, Paul Lessard (Univeristé du Québec a Montréal)

Marking programming assignments involves a lot of work, and with large classes, the feedback provided to students through marking is often rather limited and late.
Oto is a customizable and extensible marking tool that provides support for the submission and marking of assignments. Oto aims at reducing the marking workload and, also, at providing early feedback to students.
In this paper, we present Oto's new marking language and give an overview of its implementation as a Domain-Specific Language.

Supporting student-generated free-response questions

Andrew Luxton-Reilly, Paul Denny, Beryl Plimmer, Daniel Bertinshaw (University of Auckland)

Although a number of existing systems support student-generated multiple choice questions, supporting free-response questions creates additional challenges. StudySieve is a web-based tool that extends student-generated questions to the free-response domain.
We report on the use of StudySieve in three large undergraduate Computer Science courses. Students produce more content than required, provide feedback to their peers and report that they learn from question authoring, question answering, seeing the answers produced by their peers and evaluating those answers.

Automated Assessment of Short Free-Text Responses in Computer Science using Latent Semantic Analysis

Richard Klein, Angelo Kyrilo (University of the Witwatersrand), Mayya Tokman (University of California, Merced)

In the last few decades, much research has focused on the evaluation and assessment of students' knowledge. The idea that computers can now be used to aid assessment is appealing. While implementing automatic marking of multiple choice questions is trivial, most

educators agree that such form of assessment provides only limited insight into students' knowledge. Due to this limitation, teachers prefer to use unstructured questions in assessments. These questions, however, are much more complicated to mark as the semantic meaning of a response is more important than any of the individual keywords. Latent Semantic Analysis (LSA) has a remarkable ability to infer meaning from a text in this way. This paper describes the design, implementation and evaluation of an automatic marking system based on LSA and designed to grade paragraph responses to exam questions. In addition to presenting the algorithm and the theoretical basis of the system, we describe the tests that were conducted to test its efficacy. The tests included comparing the marks for several computer science courses exams generated by the system with the original grades awarded by a human examiner. The various settings of the system are studied to understand their effect on the accuracy. Using this understanding, along with trial and error, good configurations for each question are found. Under ideal configurations the system is capable of generating marks with correlations above 0.80 compared to the human examiner's grades. This is considered an acceptable variance. Generating these ideal configurations is nontrivial but possible by designing effective ways to extract appropriate settings from features of the data. If there is enough training data the system easily performs at rates that match the inter-correlation between human markers.

WeScheme: The Browser is Your Programming Environment

Danny Yoo (WPI), Emmanuel Schanzer (Harvard University), Shriram Krishnamurthi (Brown University), Kathi Fisler (WPI)

We present a programming environment called WeScheme that runs in the Web browser and supports interactive development. WeScheme programmers can save programs directly on the Web, making them accessible from everywhere. As a result, sharing of programs is a central focus that WeScheme supports seamlessly. The environment also leverages the existing presentation media and program run-time support found in Web browsers, thus making these easily accessible to students and leveraging their rapid engineering improvements. WeScheme is being used successfully by students, and is especially valuable in schools that have prohibitions on installing new software or lack the computational demands of more intensive programming environments.

Habits of Programming in Scratch

Orni Meerbaum-Salant, Michal Armoni, Mordechai Ben-Ari (Weizmann Institute of Science)

Visual programming environments are widely used to introduce young people to computer science and programming; in particular, they encourage learning by exploration. During our research on teaching and learning computer science concepts with Scratch, we discovered that Scratch engenders certain *habits of programming*: (a) a totally bottom-up development process that starts with the individual Scratch blocks, and (b) a tendency to extremely fine-grained programming. Both these behaviors are at odds with accepted practice in computer science that encourages one: (a) to start by designing an algorithm to solve a problem, and (b) to use programming constructs to cleanly structure programs. Our results raise the question of whether exploratory learning with a visual programming environment might actually be detrimental to more advanced study.

From Concrete to Abstract? Problem Domain in the Learning of Introductory Programming

Osvaldo Luiz de Oliveira, Ana Maria Monteiro (Faculty of Campo Limpo Paulista), Norton Trevisan Roman (University of São Paulo)

A good deal of research on learning introductory programming have been carried out along the past years based on a generalization to mature individuals of Piaget's theory which states that learning among children progresses from concrete to abstract. In this research, we set up two problem domains—a concrete and an abstract one—along with specific programming languages and compilers. We experimentally investigated how these domains were used by two groups of undergraduate students without previous programming knowledge. Results suggest that the type of domain (either concrete or abstract), when taken in and on itself, does not affect the learning of introductory programming. On the other hand, the previous knowledge students have about the domain does influence learning.

4.7. Tuesday Paper Sessions 13, 14: 10.45-12.00

Draw a Social Network

Sarah Carruthers, Todd Milford, Timothy Pelton, Ulrike Stege (University of Victoria, BC)

We investigate the effect of graph theory instruction on the representations of social networks by grade six students. In this quasi-experimental study, treatment groups participated in graph theory lessons as part of their mathematics class. In evaluating student responses to pre and posttests we observed that students shifted in their approach to representing a social network problem, from less to more abstract - starting with complex vertices (superfluous detail) and planar graphs (no crossed edges) toward simple vertices and non-planar graphs.

CS Education Re-Kindles Creativity in Public Schools

Vicki Bennett, Kyu Han Koh, Alexander Repenning (University of Colorado Boulder)

Creativity is an important aspect of industry and education. The lack of creativity in current students has become a concern for educators. Through the process of implementing the Scalable Game Design project to teach computer science through game authoring, fostered/increased creativity occurred in public middle schools. Despite some structural limitations of the US educational system, creativity among the participating students was recognized. This paper describes a unique solution to fostering creativity while teaching game design in the limiting public school environment.

A Pre-College Professional Development Program

Stephen Cooper (Stanford University), Wanda Dann (Carnegie Mellon University), Dan Lewis (Santa Clara University), Pam Lawhead (University of Mississippi), Susan Rodger (Duke University), Madeleine Schep (Columbia College), RoxAnn Stalvey (College of Charleston)

In this paper, we describe the results of a four-year collaborative project conducted among six higher education institutions and their partner pre-college school systems across the US. The primary goal of the project was to offer professional development to middle and high school teachers to enable those teachers to create modules and courses to excite their students about computing. The project used Alice, a software program that utilizes 3-D

visualization methods, as a medium to create a high-level of interest in computer graphics, animation, and storytelling among middle and high school students, to build understanding of object-based programming. More than 100 middle and high school teachers participated in the project, with approximately 80% of those reporting that they had used what they learned during summer workshops in their classrooms during the subsequent years.

GUIGraph – Editing Live Object Diagrams for GUI Generation Enables New Pedagogy in CS1/2

Duane Buck (Otterbein University)

The GUIGraph software tool supports a new pedagogy, motivates students, and solves early user-interface issues. Regardless of the type of curriculum, it can provide an initial, intuitive introduction to object-oriented thinking, even before coding is discussed. By editing a UML-like object diagram, the student creates and links virtual Java Swing objects representing a user-interface, and can instantly view and manipulate its realization. GUIGraph is unique in that the student specifies an object structure to be created, equivalent to writing complex source code. When requested, GUIGraph generates the Java source code of an abstract class that constructs the object structure. The student then completes the coding of a concrete class that implements its application specific abstract methods. The functionality of the application is cleanly separated from its user-interface, which helps build design intuition, and iterative refinement of the user-interface is supported by regenerating the abstract class.

Toward Replicating *Handmade* Algorithm Visualization Behaviors in a Digital Environment: A Pre-Study

Ming-Han Lee, Guido Rößling (Technische Universität Darmstadt)

Low fidelity algorithm visualizations (AV) made manually using simple art supplies are believed to have several pedagogical advantages over *high fidelity* visualizations generated by computer. Our research thus aims to introduce the kind of paper-and-pen, handmade AV construction experience into a computer-based environment. We videotaped ten students constructing handmade visualizations of their chosen algorithms to determine user behaviors we need to translate into our AV system. Eight key operational behaviors are identified, which leads to further derived operational behaviors. Based on the pre- study, we propose three new lo-fi AV design requirements. Implementation of a browser-based AV system that supports these operational behaviors and meets these design requirements is underway.

Improving Compilers Education through Symbol Tables Animations

Jaime Urquiza-Fuentes, Francisco Manso, J. Ángel Velázquez-Iturbide (Rey Juan Carlos University)

This paper presents the evaluation of an educational tool focused on the visualization of the symbol table in the context of a compiler course. In a first evaluation we used simulation exercises and tested basic concepts of symbol tables. We detected efficiency improvements, students who used the tool completed the exercises with the same grading and significantly faster than the students who did not use the tool. In addition students' opinion was positive. In a second evaluation we used more active tasks, and tested students' skills on writing parser specifications regarding symbol table management. We have detected significant improvements. Students who used the tool outperformed those who did not by 22%.

4.8. Wednesday Keynote: 08.30-09.40

Technology for Teaching the Rest of Us
Mark Guzdial (Georgia Institute of Technology)

The motivated student is easy to teach. You facilitate learning and get out of the way. It's much more challenging to teach the student who is less motivated, or who needs knowledge to support their main interest. Think of the graphics designer who chooses to learn scripting to make their job easier, but doesn't want to learn to "program" and whose many (simple) mistakes cost valuable time. Think of the secondary-school business teacher who wants to teach computer science, but who doesn't want to learn to be a professional programmer. The number of people who need some knowledge of a domain may be much greater than those who need expertise in that domain. Providing learning opportunities tailored to the needs and interests of the learner, potentially motivating that interest where necessary, is a great and important challenge in an increasingly technological society. My talk will describe characteristics of these challenges and suggest where computing technologies and computing education research insights may provide solutions.

4.9. Wednesday Paper Sessions 15, 16: 10.45-12.00

Understanding the syntax barrier for novices
Paul Denny, Andrew Luxton-Reilly, Ewan Tempero, Jacob Hendrickx (University of Auckland)

Mastering syntax is one of the earliest challenges facing the novice programmer. Problem solving and algorithms are the focus of many first year programming classes, leaving students to learn syntax on their own while they practice writing code. In this paper we investigate the frequency with which students encounter syntax errors during a drill and practice activity. We find that students struggle with syntax to a greater extent than we anticipated, even when writing short fragments of code.

Understanding Novice Programmer Difficulties via Guided Learning
Shuhaida Shuhidan, Margaret Hamilton, Daryl D'Souza (RMIT University)

Learning to program is known to be problematic for a significant number of students as evidenced by high failure rates reported by Computer Science schools. Students either fail to comprehend a range of fundamental programming concepts or carry misunderstandings and misconceptions about programming well into the semester, leading to summative assessment failures. Multiple-choice questions(MCQs) in summative assessments are a popular choice of instrument to test novice learners of programming, yet during their formative stages such questions are typically used in the traditional "practice" or "rote learning" contexts, leaving gaps in understanding of programming concepts. In addition to the use of multiple-choice questions for practice, our guided learning tool aims to involve them to help identify cognitive lapses in learning programming. In this paper we report the use of multiple-choice exercises in guided learning, within the learning context of novice programmers who are typically first-time university students of a Computer Science program. We report results of a pilot study

that uses partially-completed guided learning exercises, to prevent students at the outset from falling into the cognitive traps that often ensnare the novice programmer.

Continual And Explicit Comparison to Promote Proactive Facilitation During Second Computer Language Learning

Matt Bower, Annabelle McIver (Macquarie University)

This paper describes a Continual And Explicit Comparison (CAEC) approach to overcoming proactive inhibition and amplifying proactive facilitation in a second year Java course. The approach utilizes continual and explicit comparison to students' prior learning (in this case C++ programming knowledge) in early stages of learning the new language in order to more rapidly build understanding and more definitively form concept boundaries between the two languages. The majority of students felt the approach supported learning of the second language (proactive facilitation) without causing any interference with second language learning (i.e., minimal proactive inhibition). Some students also indicated that the approach enhanced their understanding of the first language (retroactive facilitation) and overwhelmingly agreed that the approach did not interfere with their understanding of the first language (i.e., minimal retroactive inhibition). Students also indicated that their Java programming ability and their enjoyment of programming increased during the period that the continual and explicit comparison approach was applied.

What Might Computational Thinking Mean and Imply?

Chenglie Hu (Carroll University)

Computational thinking has been promoted in recent years as a fundamental skill at the same level as reading, writing, and arithmetic. However, what computational thinking really means remains speculative. While wonders, discussions and debates will likely continue, this article provides some analysis on the notion. It argues that computational thinking is likely a hybrid thinking paradigm that must accommodate different thinking modes in terms of the way each would influence what we do in computation. Furthermore, a description of computational thinking is attempted to connect thinking elements to the known modes of thought. Finally, some implications are discussed.

Beyond Good and Evil Impacts: Rethinking the Social Issues Components in Our Computing Curricula

Randy Connolly (Mount Royal University)

It is by now widely accepted that social and professional issues are an important part of any computer science curriculum. The approach taken in most social issues courses is to articulate the social impacts of different computer technologies and then apply macro-ethical theories to those impacts. This paper argues that this approach has a number of drawbacks. First, it is based on a technological deterministic style of social explanation that has been in disrepute in the academic social sciences for decades. Second, it uses an algorithmic approach to ethics that simplifies the social complexity and the uncertainty that is the reality of socio-technological change. It concludes by suggesting that the alternative to the ethical evaluation of impacts is to focus the course instead on the social context; that is, on clarifying and unpacking the complexity involved in the relationship between technology and society.

Computing student practices of cheating and plagiarism: A decade of change
Judy Sheard (Monash University), Martin Dick (RMIT University)

Cheating in undergraduate computing courses is an ongoing and widespread concern. In this paper we report an investigation of changes over the last decade in computing students' attitudes towards cheating practices, the extent of cheating behavior, and factors which influence cheating at an Australian university. A comparative analysis of data from surveys in 2000 and 2010 of undergraduate students in a School of Information Technology found that students in 2010 considered cheating less acceptable and the practice of cheating was reportedly lower. These results are discussed in terms of measures that have been taken to address this problem.

4.10. Wednesday Paper Sessions 17, 18, 19: 13.15-14.30

"Computer Science and Nursery Rhymes" A Learning Path for the Middle School
Doranna Di Vano (Scuola Media Statale "Via Petrarca"), Claudio Morilo (University of Udine)

We have tried to introduce some ideas and way of thinking of computer science through a set of extra-curricular activities on nursery rhymes. In this paper we discuss our experience in an Italian middle school. The chosen subject is naturally connected to what the pupils see, or listen to, in the primary school as well as in their home. Starting from this material which is familiar to them, the pupils are guided to explore the "computational paradigm". This is accomplished through gradual steps, where they are solicited to observe, to analyze, to devise models and, eventually, to develop simple programs in Logo. Our work is an attempt to suggest a different perspective on computation, since most of the opportunities for the pupils to interact with the new technologies tend to reinforce a view that relegates all the computing sphere to a merely instrumental role.

Experimental Evaluation of BeadLoom Game: How Adding Game Elements to an Educational Tool Improves Motivation and Learning
Acey Boyce, Antoine Campbell, Shaun Pickford, Dustin Culler, Tiffany Barnes (University of North Carolina at Charlotte)

The Virtual Bead Loom (VBL) is a Culturally Situated Design Tool that successfully teaches students middle school math concepts while they learn about and create their own Native American bead artifacts. We developed BeadLoom Game to augment VBL with game elements that encourage players to apply the computational thinking skills of iteration and layering while optimizing the number of steps they take to solve a puzzle. In our prior work, we showed that BeadLoom Game is effective at teaching Cartesian coordinates, iteration, and layering. In this study, we use a switching replications experimental design to compare performance of BeadLoom Game with the VBL. Our results from two summer camps, one for middle school and one for college-bound high school students, show that through the addition of game based objectives, BeadLoom Game teaches Cartesian coordinates as well as the VBL but also teaches the computational thinking practices of iteration and layering.

Teaching CS Unplugged in the High School (with Limited Success)

Yvon Feaster (Clemson University), Luke Segars (UC Berkeley), Sally K. Wahba, Jason O. Hallstrom (Clemson University)

CS Unplugged is a set of active learning activities designed to introduce fundamental computer science principles without the use of computers. The program has gained significant momentum in recent years, with proponents citing deep engagement and enjoyment benefits. With these benefits in mind, we initiated a one-year outreach program involving a local high school, using the CS Unplugged program as the foundation. To our disappointment, the results were at odds with our enthusiasm—significantly. In this paper, we describe our approach to adapting the CS Unplugged mate- rial for use at the high school level, present our experiences teaching it, and summarize the results of our evaluation.

A Scheme for Improving ICT Units with Critically Low Student Satisfaction

Angela Carbone, Jessica Wong, Jason Ceddia (Monash University)

Unit evaluations across many Australian universities indicate that close to 10% of units in Information and Communication Technology (ICT) and Engineering disciplines are flagged as needing critical attention, and as such these faculties often struggle to meet university and national targets on educational performance. Further, ICT and Engineering repeatedly have the highest student dropout rates. This paper reports on the efficacy of activities undertaken to improve teaching quality and student satisfaction. Specifically, this paper outlines a Peer Assisted Teaching Scheme (PATS) as a process that was embedded in the Faculty of Information Technology (FIT) at Monash University to build peer assistance capacity in the faculty to improve student satisfaction of units in need of critical attention.

Combining Multiple Pedagogies to Boost Learning and Enthusiasm

Lori Pollock, Terry Harvey (University of Delaware)

This paper describes the pedagogy we applied in a 5-week class, in which students taught themselves (and each other) a new language, new OS, GUI programming, and simple networking for collaborative games. They learned communication, negotiation, collaboration, presentation and teamwork skills; and project design and iterative development. We had four goals: increased learning, enthusiasm about CS, confidence in technical ability and communication skills. To achieve these goals, we decided to rely solely on the integration of teaching techniques that we believed would be highly effective: collaborative teams, student presentations, student critique of work, open-ended projects of student design, iterative process, journal reflection, and motivation through helping others. The students had to learn about each technique through discussion, modeling, and moderated practice. We focused on this process learning and trusted that the technical material would come from solving the (unspecified) assignments. This focus left no time for traditional teaching activities. We present quantitative and qualitative results from a student survey and the students' reflective journals. Students reported learning at a greater rate than in other CS courses while maintaining (and in some cases acquiring) a high level of enthusiasm and confidence.

A Cooperative Learning-based Strategy for Teaching Relational Algebra

Alexandra Martinez, Arturo Camacho (Universidad de Costa Rica)

This paper presents the design, implementation, and assessment of a cooperative learning-based teaching strategy to introduce relational algebra in an undergraduate database course. It has been implemented in four course sections across two semesters. The strategy was assessed from both students and teachers perspective. Assessment results show that between 78% and 92% of the students considered that the group work enriched their learning, providing support for the use of cooperative learning. Also, the results from a homework and an exam on the subject show an improvement on the students' learning of relational algebra.

Open Source Contribution As An Effective Software Engineering Class Project

Robert Marmorstein (Longwood University)

Software engineering courses often involve a semester project designed to give students experience with real-world programming challenges and to familiarize them with the phases of software development cycle not covered in other programming classes. One means of exposing students to realistic programming challenges is to make participation in open source development a part of the semester project. Open source projects usually have well established code bases and are maintained by a team of experienced developers.

This paper describes an assignment in which students contribute to an open source project of their choosing. The project is designed to immerse students in the open source community and expose them to the work flow and design strategies of a large project.

In addition to selecting the project they will work on, students have freedom to choose the specific ways in which they contribute to the project. For instance, students can choose to focus on implementation of new features over software maintenance or can focus on documentation and design over both. The assignment contains a proposal phase that allows the instructor to ensure that students are exposed to a healthy cross section of the development cycle.

Extreme Apprenticeship Method: Key Practices and Upward Scalability

Arto Vihavainen, Matti Paksula, Matti Luukkainen, Jaakko Kurhila (University of Helsinki)

Programming is a craft that can be efficiently learned from people who already master it. Our previous work introduced a teaching method we call Extreme Apprenticeship (XA), an extension to the cognitive apprenticeship model. XA is based on a set of values that emphasize doing and best programming practices, together with continuous feedback between the master and the apprentice. Most importantly, XA is individual instruction that can be applied even in large courses. Our initial experiments (n = 67 and 44) resulted in a significant increase in student achievement level compared to previous courses. In this paper, we reinforce the validity of XA by larger samples (n = 192 and 147) and a different lecturer. The results were similarly successful and show that the application of XA can easily suffer if the core values are not fully adhered to.

The Academic Enhancement Program in Introductory CS: A Workshop Framework Description and Evaluation

Rylan Egan (Memorial Univ.), Diana Cukierman, Donna McGee Thompson (Simon Fraser Univ.)

The Academic Enhancement Program (AEP) is a student focused proactive intervention initiated in 2006 by the School of Computing Science and the Student Learning Commons at Simon Fraser University to provide academic self-regulation skills. This program has been offered as a required component in selected first year courses since 2008. In this paper we provide a description and evaluation of AEP 101 - a two-hour workshop provided in a first year programming course. The workshop material includes a proposed framework to address academic challenges in general and a learning model customized for CS but applicable to other disciplines. Statistical evaluations of the workshop provide evidence that students are interested and receptive to the program and that it influenced their study-strategy choices. In addition, some statistically detectable correlations between workshop satisfaction, academic attitudes, and academic performance are described.

4.11. Wednesday Paper Sessions 20, 21, 22: 14.45-16.00

Getting CS Undergraduates to Communicate Effectively

Andreas Karatsolis, Iliano Cervesato, Nael Abu-Ghazaleh, Yonina Cooper, Khaled Harras, Kemal Oflazer, Thierry Sans (Carnegie Mellon University – Qatar campus)

In the last decade or so, the ACM, the IEEE and other organizations have acknowledged that there is a problem with the way communication is taught in the Computer Science curriculum: the writing, speaking, and presentation skills students learn in the classroom do not match what is expected of them in the workplace. The proposed solution, adopted by many undergraduate colleges, was to add a technical communication course to the CS curriculum. This does not appear to be enough, as mainstream accreditation boards are still emphasizing the need for improvement of communication skills instruction in their recent reports and recommendations. For the last two years, we have experimented with a complementary transversal approach where many “traditional CS” courses in our program have added a communication component to their syllabus, while at the same time our technical communication course has been revamped to expose students to realistic practices as promoted by situated learning theory. The results, so far anecdotal, point to improved student performance and attitude across several communication dimensions, in particular writing and presentation. We plan to develop this experiment by spreading it across more classes and by starting to collect rigorous measurements of students’ communication performance.

Undergraduate Research: a Case Study

Herman Koppelman, Betsy van Dijk, Gerrit van der Hoeven (University of Twente)

This paper describes a one semester research course for undergraduates of computing programs. Students formulate a research proposal, conduct research and write a full paper. They present the results at a one-day student conference. On the one hand we offer the students a lot of structure and support; on the other hand an important feature of the course is that they are in control of their own research. A key aspect of the pedagogical approach is that the students are supervised in small teams by experienced staff. The results of

evaluations show that the students are positive about the course. One of the main findings is that they feel well prepared to conduct research in the graduate program.

Bringing Undergraduate Students Closer to a Real-World Information Retrieval Setting: Methodology and Resources

Julián Urbano, Mónica Marrero, Diego Martín and Jorge Morato (University Carlos III, Madrid)

We describe a pilot experiment to update the program of an Information Retrieval course for Computer Science undergraduates. We have engaged the students in the development of a search engine from scratch and they have been involved in the elaboration, also from scratch, of a complete test collection to evaluate their systems. With this methodology they get a whole vision of the Information Retrieval process as they would find it in a real-world setting, and their direct involvement in the evaluation makes them realize the importance of these laboratory experiments in Computer Science. We show that this methodology is reliable and feasible, and so we plan on improving and keep using it in the next years, leading to a public repository of resources for Information Retrieval courses.

AnimalSense: Combining Automated Exercise Evaluations with Algorithm Animations

Guido Rößling, Mihail Mihaylov (Technische Universität Darmstadt), Jerome Saltmarsh (Queensland University of Technology)

Exercises with an automatic evaluation can be helpful for students, if they cover meaningful tasks and are sufficiently challenging. We have designed an exercise system that combines exercise tasks with automatic evaluation and integrated algorithm animation. The paper describes the current status and sample tasks our system can handle.

Using Run Time Traces in Automated Programming Tutoring

Michael Striewe, Michael Goedicke (Paluno – The Ruhr Institute for Software Technology)

Running test cases against a student's solution of a programming assignment is one of the easiest ways to generate feedback. If black-box tests are used, students may have difficulties to retrace the complete system behaviour and to find erroneous programming statements. This paper dis- cusses the use of automated trace generation for assisting students in this task. Both manual and automated trace interpretation is discussed and evaluated by examples.

A proposal for Automatic Evaluation in a Compiler Construction Course

Emilio Julio Lorenzo, Javier Vélez, Anselmo Peñas (UNED)

Experience in teaching the subject of Language Processors within the degree for Technical Engineering Computer Science, highlights the difficulties developing a compiler, for both students and instructors. The former because they are often disoriented when they are unable to get some feedback on their work and the latter because of the amount of effort and time involved to perform the evaluation. This paper presents a system for assisting evaluation using test cases, aligned with the trends of European Higher Education Area ongoing evaluation methodology. This approach is implemented in four phases where different problems are faced. The first phase consists of designing a test case set, accurate enough to discern whether the student successfully absorbed the different concepts and contents of the

subject. The second phase involves the implementation of a runtime environment for the designed tests, so that many of the tasks required by the teaching staff can be automated. In the third phase, automatic assessment, according to different criteria is implemented. The fourth phase is the report of the results of the previous execution like straightforward feedback for both students and teachers.

Evaluation Framework Underpinning the Digital Divas Programme

Annemieke Craig (Deakin University), Julie Fisher, Helen Forgasz (Monash University), Catherine Lang (Swinburne University of Technology)

In Australia, as elsewhere, women's participation rates in Information Technology (IT) have been low. IT is the generic term used to refer to the many courses in the Computer Science, and Information Systems disciplines. While there have been a number of intervention programmes implemented aimed at encouraging women into IT and retaining them once there, few have included evaluations of the efficacy of the intervention. Thus little is known about the factors contributing to the success, or lack of success, of the interventions, or of the medium and longer term impacts for the participants. In this paper we briefly describe an intervention programme implemented with girls in the high school years. We present an evaluation framework providing a detailed overview of the aims and processes involved in the evaluation of the programme. Data for the evaluation were embedded within the data gathering methods associated with the research on the intervention itself.

The Impact of IMPACT: Assessing students' perceptions after a day of computer exploration

Mary Anne L. Egan, Tim Lederman (Siena College)

Declining enrollments in computer science are a cause of great concern. There has been a 30% decline in enrollments in US computer science bachelor programs over the previous decade and more than a 50% decline in the enrollment of women in computer science [20]. Some research suggests this is due to the negative perceptions high school students hold of computer science [6,16]. IMPACT is a unique way of attracting excellent students to the major by inviting students and their teachers to a day of exploration and competition hosted by our college. The aim of our research is to investigate how the students' attitudes change and to assess the impact of this daylong event. This paper will demonstrate that IMPACT increases awareness of computer science and changes the perceptions of the students who attend the event.

Female Students' experiences of programming: It's not *all* bad!

Reena Pau, Wendy Hall, Marcus Grace, John Woollard (University of Southampton)

Programming has been cited as a barrier for female students to enjoy and pursue computing as a career or at higher education. However, there are examples of good practice, which demonstrate that programming can act as a bridge rather than a barrier. As a result of surveying 103 students and interviewing 60 students from 3 different UK higher education institutions and this paper demonstrates that female students can enjoy programming and take it further for their careers.

5. Working Groups

Three working groups will convene concurrently with the conference beginning on Saturday, June 25th. Each working group will present a work in progress report at the Monday morning plenary session.

WG 1: Motivating All Our Students?

Leader: Janet Carter (University of Kent)

This working group is a research study which addresses the course related area of classroom management.
The aim of this working group is to continue previous work which explores the ways in which academics around the world enthuse their high achieving students, and to create a repository of effective ideas. This includes determining the scope and usage of differentiated teaching techniques in educational institutions worldwide; the educational soundness of the tech–niques used, and the impact of these techniques on areas including student achievement, plagiarism, and staff workloads.

WG 2: Informatics in Secondary Education

Leaders: Peter Hubwieser (TU München), Torsten Brinda (Universität Erlangen-Nürnberg), Johannes Magenheim (University of Paderborn, Sigrid Schubert (University of Siegen)

This working group will collect, evaluate, integrate and present research findings about Infor–matics in secondary schools. As a result we expect a comparison of the effects of different organizational conditions, teaching approaches, curricula, teaching methods of Informatics courses in secondary schools in different countries. As a theoretical framework for the synopsis we propose the Berlin Model of Heimann, Otto and Schulz.

WG 3: Information Assurance Education in Two and Four-Year Institutions

Leaders: Steve Cooper (Stanford University), Lance Peréz (University of Nebraska, Lincoln), Elizabeth Hawthorne (Union County College), Susanne Wetzel (Stevens Institute of Technology)

This working group builds on the work of the 2009 and 2010 working groups on information assurance (IA) education. The focus of the 2011 working group is the examination of the educational missions and curricula of two and four-year institutions with respect to IA education. More specifically, this working group will define and describe the distinct and complementary missions of two and four-year institutions with respect to IA education, describe the differences and similarities of the educational programs at two and four-year institutions, and document the challenges and opportunities for IA course articulation between two and four-year institutions.

6. Space for your notes!

7. Social Events

7.1. Lunches/Coffee Breaks and Poster Sessions

All conference lunches and coffee breaks will take place in the foyer area on the same level as the conference sessions.
The poster sessions always take place in room *argon (3.07)*.
Exhibitors, supporters, and the presentation for ITiCSE 2012 can be found in room *xenon (3.06)*.

7.2. Sunday Evening Reception

There will be an informal welcome reception for all conference participants and their guests, starting at 18.30 on Sunday, June 26th. The reception will be held on the west side of the Computer Science building, close to the conference center. The CS building is also the site for the Working Groups and the Sunday registration.

7.3. Conference Banquet

The conference banquet will be held on Wednesday, June 29, starting at 19.00. Note that the banquet location, *Castle Auerbach*, is about a 20 minute bus ride outside of Darmstadt. The busses are scheduled to leave at the conference site at 18.00. Participants are encouraged to arrive on time, or if they prefer to skip the banquet or arrange private transportation, to let us know of this beforehand.
Please make sure that you bring your banquet ticket with you, as well as (if applicable) the banquet ticket for your accompanying person(s).

7.4. Excursions

For Tuesday afternoon, we have arranged two excursion possibilities for conference attendees and their guests. Each of the excursions requires a ticket which you may also be able to purchase at the conference registration desk, if space is still available.

1. ITiCSE is not complete without a walking tour of the hosting city. This year, the walking tour will explore the main sights of downtown Darmstadt as well as the City Castle Museum, which opens especially for our guided tour! The walking tour will start at 15.00 from the front of the *darmstadtium* conference center. It will take roughly two hours.
2. The bus tour to Heidelberg, widely named one of the most beautiful towns, will start from the *darmstadtium* conference center at 14.00 *precisely*. After a bus trip over the German highways, you will explore the old town (with guides), see a selection of the main sights, and take the furnicular train up and down to the castle. The return from the city is at roughly 19.00 in front of the *darmstadtium* conference center.

8. Essentials

8.1. Emergencies

Hopefully, there will be no emergencies! If there are, the conference ("registration") desk will be able to help. The phone number for the registration desk is ***+49 1577 530 4162.***

If the registration desk is unavailable, or for an "out of hours" emergency, please call the conference chair, Guido Rößling, at ***+49 1577 252 4013***.

To reach the *police*, simply call ***110***. This will route your call to the next police emergency station. The closest police station is the 1st "Revier" (police station), located Bismarckstr. 16—about a 500m walk straight across the Herrngarten from the CS department and then straight on.

To call an ambulance in a medical emergency, call ***112***.

In general, both the police, firefighters and doctors will be able to speak English, although the "technical terms" may be difficult since they cannot be "translated" from German to English or vice versa: one needs to know them, or have a dictionary at hand, to be on the safe side.

8.2. Getting Around

Darmstadt is not very large, so reaching almost any place by a short walk is possible. All hotels should have a small map of the city available, and we strongly encourage you to pick one up.

The *darmstadtium* conference center—the site of the conference—is directly across the city castle, a sight that any inhabitant will be able to point you to.

The CS department is situated right next to the *Herrngarten*[1] park. If you leave the castle on the north side, cross the moat and then the street. Past the "Greek temple"-like building, you can enter the Herrngarten and will directly see the CS department in front and to your right. The CS department has the shape of an "E", with the open doors at the end of the middle horizontal stroke.

8.3. Advice for the Unwary

Pickpockets and robbers are by no means common in Darmstadt, but—as in any other town—they do exist. Three simple guidelines that apply to essentially all "publicly placed" ITiCSE conference also apply in 2011:

1. Before leaving the conference sites, *hide your conference badge*. Such a badge essentially broadcasts "I am a foreigner and may be an easy target", especially when traveling in groups. Also, the city inhabitants do not really need to know your name ☺.
2. Keep an eye out on your belongings, especially your wallet. The more crowded a place (including the tram) is, the more likely this will be a place for a pickpocket.

[1] For those knowing some German: this is not a typo; it's Herrngarten, not Herrengarten.

3. As all towns, Darmstadt has more “risky” areas. Alas, this can be the Herrngarten (next to the CS department) once darkness has fallen. It would be best to avoid the Herrngarten after dark by simply walking around it (it’s not *that* big to make a large detour, after all!).

8.4. Computer Access

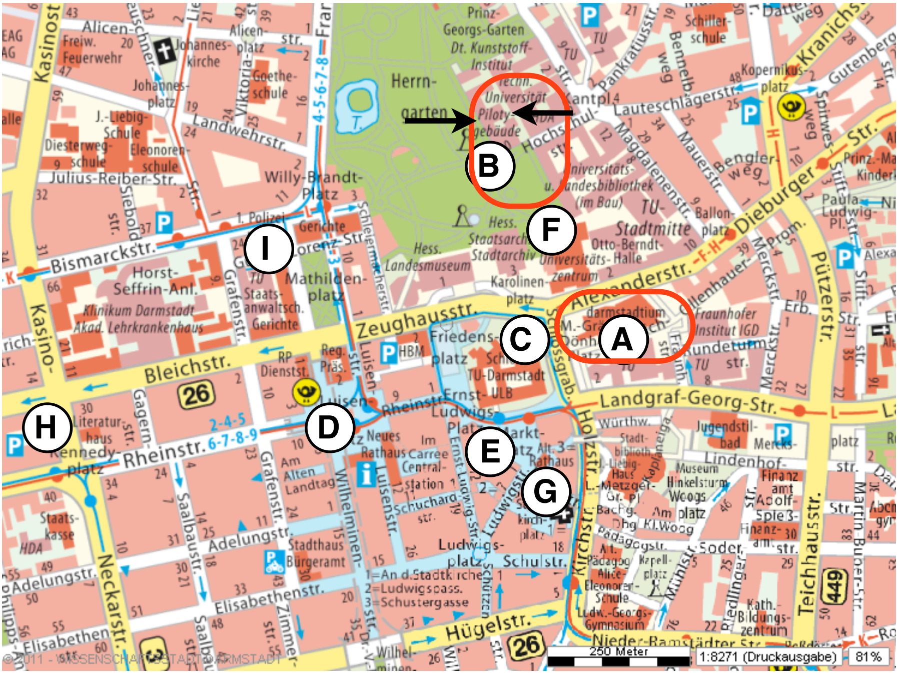
We have reserved a small computer pool of about 16 Unix-based computers in the CS department. The room is called **C003** and can be found in the basement of the CS department (“B” on the City center map). To reach the pool, enter through either of the two entrances to the CS department at the “end of the central horizontal stroke of the E”. Then take the stairs down; the reserved pool is the smaller room attached to the large computer pool.

To log in in this pool, you need to have a CS department account. Of course, you can also use the *eduroam* wireless network, provided that you have an eduroam login from your home university. Information on how to log in to the computers will be provided in your registration package.

The *darmstadtium* conference center also provides wireless LAN access. The access key for this will also be provided in your registration package.

As a courtesy to the other conference participants, we request that you refrain from downloading huge updates (e.g., operating system patches or previews, larger software updates such as from Microsoft) while using the wireless facilities. Please keep in mind that other presentations may require a working wireless network to run smoothly!

9. Map of Darmstadt City Center



Legend:

A. *darmstadtium* conference center
B. CS Department (entrances at center stroke, indicated by arrows)
C. City Castle
D. Luisenplatz (main tram/bus station, shopping center)
E. Market place (secondary tram/bus station)
F. *Welcome Hotel*
G. *Bockshaut* Hotel
H. *Ibis and Etap* Hotels
I. Police Department (Bismarkstr. 16)

All areas marked in blue—essentially those framed by the elements **D, C**, and **G**—are pedestrian areas with shops. The small blue numbers and letters on the map indicate tram (numbers) and bus (letter) lines.

10. Conference rooms in *darmstadtium*

The conference rooms inside the *darmstadtium* conference center are on level 3. The above image is oriented roughly northwards. The rooms on the left are facing the City Castle; the CS departments is to the top and left of this image.

A. The **keynotes** are held in room *europium 3 (3.02+3.03+3.03).*
B. The **paper sessions** are held in rooms *radon (3.05), neon (3.08),* and *helium (3.09).*
C. Special sessions—the **panel**, **Working Groups reports**, and the **single TT&C** *session*—are held in *helium (3.09).*
D. The **poster presentations** are placed in room *argon (3.07)*.
E. The **supporters and registration desk** is placed in room *xenon (3.06).*
F. The **coffee** and **lunch breaks** are served in the lobby area just off the image.

11. Conference at a Glance